

GENERIC DYNAMIC PATTERNS: TESTING BY EMPIRICAL EVIDENCE

by

Sercan Soylu

B.S. in C.E., Boğaziçi University, 2002

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree
of Master of Science

Graduate Program in Industrial Engineering
Boğaziçi University
2006

GENERIC DYNAMIC PATTERNS: TESTING BY EMPIRICAL EVIDENCE

APPROVED BY:

Prof. Yaman Barlas
(Thesis Supervisor)

Prof. İ. Kuban Altınel

Assist. Prof. Pınar Yolum

DATE OF APPROVAL: 15. 09. 2006

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Professor Yaman Barlas for his invaluable guidance and help during the preparation of this thesis dissertation. I would like to mention his patience, giving me inspiration and hope when I was struck at dead-ends.

I would also like to present my special thanks to Prof. Kuban Altinel and Prof. Pınar Yolum, to Eczacıbaşı Menkul Değerler Research Department and to Boğaziçi University Industrial Engineering SESDYN group, for their participation into the testing phase of this thesis study, and to Korhan Kanar for his valuable support and suggestions for the completion of this thesis.

Finally, I would like to thank also to my parents Cengiz and Birsal Soylu, and to my best friends Özge Kılıçlı and Çağrı Kaplan, for their invaluable support throughout the thesis.

ABSTRACT

Validation in system dynamics (SD) models can be classified into two major groups, structure and behavior validation. While behavior validation involves comparison of the model output with the real data, the structural validation focuses on the structure of the system and the mathematical relationships between the model variables and comparison with the relations in the real world. Indirect structure testing is a special type of structure testing that tests if the model generates the type of dynamic patterns that the real system is hypothesized to generate under special conditions. Indirect Structure Testing Software (ISTS) is a Hidden Markov Model based pattern recognition algorithm that implements indirect structure testing. In preliminary tests, the original version of ISTS is discovered to have certain flaws in pattern detection and classification of generic dynamic patterns. In the first part of this study, the flaws and erroneous classification problems have been resolved, by modifying/extending the algorithm and its coding application. Next, trained database of the algorithm was expanded. As an ultimate goal, we aim ISTS to be capable of detecting the generic patterns in the real noisy data sets. In addition, our goal is to improve ISTS, so that it is able to detect the generic patterns in the real data as well as the human eye can. Therefore, a Turing Test procedure has been developed, based on a pattern recognition test applied to twelve expert subjects, and their responses and ISTS results were analyzed statistically. Directly linked to the nature of our Turing test procedure, since the responses of the subjects and ISTS are qualitative, relative statistical analysis was used. First of all, it was observed that with respect to their first choices, both within subject differences and between subjects and ISTS differences were significant. On the other hand, looking at two types of outlier analysis, we obtained the results that outlier numbers of ISTS in the test are less than the average of the subjects. In our modal analysis, although the first choice results between ISTS and the subjects are not satisfactory, the results for the first or second choice analysis were satisfactory and ISTS performed as well as the subjects. In the last analysis, the new benchmark for the responses of ISTS and the subjects are our own classifications or hypotheses. Based on the results from this last test, we conclude that ISTS performs again as well as the subjects.

ÖZET

Sistem dinamiği modellerindeki geçerlilik kavramı temel olarak yapısal ve davranışsal geçerlilik olmak üzere iki ana gruba ayrılabilir. Davranışsal geçerlilik, model ve gerçek dünya arasındaki davranışsal karşılaştırma üzerine yoğunlaşırken, yapısal geçerlilik sistem değişkenleri ve parametreleri arasındaki matematiksel ilişkilerin gerçek dünya ile karşılaştırılması üzerine odaklanmaktadır. Dolaylı yapısal geçerlilik testleri, yapısal geçerliliğin sağlanabilmesi için tamamlayıcı test imkanları sunmaktadır. ISTS algoritması ise dolaylı yapısal geçerlilik testlerin uygulanmasında kullanılmak üzere hazırlanmış bir dinamik davranış tanımlama ve sınıflandırma algoritmasıdır. Yapmış olduğumuz testler, ISTS'in önceki versiyonun dinamik davranışların belirlemede bazı eksiklik ve hatalar saptamıştır. Bu çalışmanın ilk bölümü boyunca algoritmadaki eksiklikler ve hataların geliştirilmesi üzerine çalışmalar yapılmış ve algoritmanın Matlab'da hazırlanmış otomasyon kodu geliştirilmiştir. Bu güncellemelerin ardından, algoritmanın veri tabanı büyütülerek, algoritmanın öğrenme bölümü tamamlanmıştır. Bu tez araştırmasının esas amacı, ISTS algoritmasını gerçek dünyadan temin edilen zaman serilerinin belli bir dinamik davranış sınıfı içerip içermediğini test edebilir hale getirmektir. Buna paralel olarak, ISTS algoritmasının davranış sınıflandırmalarındaki başarı seviyesinin uzman gözüyle yapılan sınıflandırmalar ile karşılaştırılması hedeflenmiştir. Bu nedenle, tezin ikinci ana bölümü Turing testi ve bu testin istatistiksel analizleri üzerine kuruludur. Turing testi hem ISTS'e hem de on iki ayrı uzman deneye uygulanmış ve verilen cevaplar istatistiksel olarak incelenmiştir. Verilen cevaplar nitel içerikli olduğundan ötürü göreceli istatistiksel analizler üzerine yoğunlaşarak karşılaştırma analizleri yapılmıştır. İlk olarak, ISTS ve deneklerin ilk seçimleri incelendiğinde, hem denekler kendi aralarında hem de ISTS ile istatistiksel olarak önemli farklılık göstermiştir. Diğer yandan, seçimler arasında dışa düşen seçimlerin iki farklı analizinden de ISTS'in sonuçları deneklerin sonuçlarının ortalamasından daha iyi çıkmıştır. ISTS ve deneklerin ilk tercihlerinin ortancası üzerinden yaptığımız analizden, ISTS'in sonuçları moddan birazdaha kötü çıkmasına karşın, ilk ya da ikinci seçimler üzerinden yapılan analizden ISTS'nin sonuçlarının moddan daha iyi olduğu görülmüştür. Son olarak, bizim sınıflandırma seçimlerimize göre yaptığımız analizden ISTS'nin sonuçlarının en az uzman deneklerin sonuçları kadar başarılı olduğu saptanmıştır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	i
ABSTRACT	ii
ÖZET	iii
LIST OF FIGURES	vi
LIST OF TABLES	viii
LIST OF SYMBOLS	xi
1. INTRODUCTION	1
2. BASIC STRUCTURE OF INDIRECT STRUCTURE TESTING ALGORITHM.....	4
2.1. Continuous Density Hidden Markov Models	4
2.2. Algorithmic Structure and Function Relations	8
2.3. Training Part	10
2.4. Classification Part	11
2.5. Problem issues in the original algorithm	12
3. IMPROVEMENT METHODOLOGY	14
3.1. Solution for the Oscillatory Pattern Recognition Problem	14
3.2. Solution for the erroneous classification problem	17
3.3. Inclusion of Linear Pattern to HMM Based Algorithm	20
3.4. Smoothing	24
3.5. Solution for the overtraining problem	27
4. TESTING THE NEW ALGORITHM AND TURING DESIGN	29
4.1. Preliminary Test by Synthetic Data	29
4.2. Turing Test Design	32
4.3. Smoothing (Filtering) Effect.....	35
5. STATISTICAL ANALYSIS OF TURING TEST.....	39
5.1. Test Results and Statistical Methods	39
5.2. Standard t-test	47
5.3. Outlier Analysis	52
5.4. Modal Analysis	58
5.5. Comparison with Our Classification.....	64
6. CONCLUSIONS	68
APPENDIX A: APPENDIX A: REVISED MATLAB CODES for ISTS	70

APPENDIX B: PRELIMINARY TEST PATTERNS	87
APPENDIX C: TURING TEST	100
Source of Turing Test Data.....	130
REFERENCES	132

LIST OF FIGURES

Figure 2.1. Generic dynamic patterns	6
Figure 3.1. Oscillatory pattern example.....	14
Figure 3.2. Example for <i>S-shaped decline</i> pattern.....	18
Figure 3.3. Example 1 for positive linear growth pattern	21
Figure 3.4. Example 2 for positive linear growth pattern	23
Figure 3.5. Example for the noise in a real time-series (Akbank Market Value, in million dollars, between March10, 2004 and July 23, 2004)	24
Figure 3.6. Real noisy data and its smoothed version with Savitzky-Golay algorithm (span length =3, degree of polynomial = 1).....	25
Figure 3.7. Real Noisy data and its smoothed version with moving-average algorithm (span length = 3)	26
Figure 3.8. Example for transitory zone pattern between SSHGR and D2GRA.....	27
Figure 4.1. Synthetically generated example for D1PEG.....	30
Figure 4.2. Synthetically generated example for D2GRB	31
Figure 4.3. Sample Turing Test Data Sheet.....	34
Figure 4.4. Noisy real data, hypothesized as the generic dynamic pattern D2PEG	36

Figure 4.5. 4th degree smoothed version of the series displayed in Figure 4.4.....	37
Figure 5.1. Generic dynamic patterns in Turing test	42
Figure 5.2. Sample subject response sheet (Time Series 1, Subject 1).....	42

LIST OF TABLES

Table 2.1. List of dynamic behavior classes	7
Table 3.1. Likelihood values for the pattern in Figure 3.1, without normalization	15
Table 3.2. Likelihood values for the pattern in Figure 3.1, with normalization	15
Table 3.3. Likelihood values for the pattern in Figure 3.1, with improved algorithm	17
Table 3.4. Likelihood values for the pattern in Figure 3.1, with the improved database ...	19
Table 3.5. Likelihood values for the pattern in Figure 3.2 (original version of ISTS).....	19
Table 3.6. Likelihood values for the pattern in Figure 3.2 (improved version of ISTS).....	19
Table 3.7. Likelihood values for the pattern in Figure 3.3 (old version of ISTS)	22
Table 3.8. Likelihood values for the pattern in Figure 3.3 (new version of ISTS).....	22
Table 3.9. Likelihood values for Figure 3.4 (original algorithm).....	23
Table 3.10. Likelihood values for Figure 3.4 (improved algorithm)	23
Table 4.1. Likelihood values for the pattern in Figure 4.1 (improved ISTS)	30
Table 4.2. Likelihood values for the pattern in Figure 4.1 (original ISTS)	30
Table 4.3. Likelihood values for the pattern in Figure 4.2 (improved ISTS)	31
Table 4.4. Likelihood values for the pattern in Figure 4.2 (original ISTS)	32

Table 4.5. Likelihood values for the pattern in Figure 4.4, without smoothing	36
Table 4.6. Likelihood values for the pattern in Figure 4.4, after smoothing	37
Table 5.1. Codes for the generic dynamic patterns.....	39
Table 5.2. Responses of ISTS and subjects to Turing test queries (1-26)	43
Table 5.3. Responses of ISTS and subjects to Turing test queries (27-52)	44
Table 5.4. Responses of ISTS and subjects to Turing test queries (53-78)	45
Table 5.5. Responses of ISTS and subjects to Turing test queries (79-106)	46
Table 5.6. t-test results for the comparison of first choices of ISTS and Subject 1.....	50
Table 5.7. t-test results for all combinations of the top choices of subjects and ISTS	51
Table 5.8. Turing test results of ISTS and subjects (type-1 outliers indicated).....	53
Table 5.9. Coded results (1:outlier) for the type-2 outlier analysis, for top choices	54
Table 5.10. Turing test results of ISTS and subjects (type-2 outliers indicated).....	56
Table 5.11. Coded results (1: outlier) for the type-2 outlier analysis, for top choices	57
Table 5.12. Turing Test results of ISTS, subjects and the modes (top choices).....	59
Table 5.13. Coded results (1: deviation from mode) for the modal analysis (top choices) .	60
Table 5.14. Turing test results of ISTS and subjects and the modes ⁶¹ (first or second choices, test number 1-60)	61

Table 5.15. Turing test results of ISTS and subjects and the modes (first or second choices, test number 61-106)	62
Table 5.16. Coded results (1: deviation from mode) for the modal analysis (top two choices)	63
Table 5.17. Turing Test Results of ISTS and subjects compared to our selections (top choices)	65
Table 5.18. Coded results (1: deviation from ours) comparison with our classification choices).....	66

LIST OF SYMBOLS

A	State transition matrix
a_{ij}	State transition probability from state i to state j
B	Set of posteriori density of observation vectors
b	A posteriori density of observation vectors
N	Number of states
O	Sequence of observation vectors
o	Observation vector
P	Number of classes
S	State sequence
s_i	State at step i
T	Number of time segments
V_i	Covariance matrix of state i
y	Normalized model-generated time sequence
λ	Set of Hidden Markov Model parameters
Π	Initial probability vector
π_i	Initial probability of state i

1. INTRODUCTION

System Dynamics (SD) is a methodology that emphasizes modeling the structure of the “real world” processes. SD models are based on identifying the causal influences among the variables within the system and expressing these relations in terms of nonlinear differential (or difference) equations. Computer simulations with SD models are ultimately aimed at analyzing alternative policies and modifications on the structures of systems, in order to acquire improved system performance (Sterman, 2000).

Model validation has been recognized as one of the primary issues in system dynamics field (Forrester, 1968; et al., 1974; Forrester and Senge, 1980; Sterman, 1984; Barlas, 1989a; Barlas and Carpenter, 1990). Richardson (1996) identifies “confidence and validation” as one of the eight key problems for the future SD discipline.

Kanar (1999) quotes that validation of a SD model consists of two broad components: *structure* validation and *behavior* validation. Structure validation means establishing that the relationships used in the model construct an adequate representation of the real relationships, in regard to the purpose of the study. Behavior validation consists of demonstrating that the behavior of the model is “close enough” to the observed real behavior. In SD validation, without an acceptable level of structure validity, the behavior validity testing is redundant. Behavior validation is preceded by structure validation and the tests are meaningful only if confidence has been established first in the structure of the model.

Setting aside behavior validity, in the SD literature, testing the structure validation is composed of two major parts: *direct* structure testing and *indirect* structure (or structure-oriented behavior) testing. In an attempt to initiate research in structure validation, Barlas (1989b and 1996) distinguishes those test types. Direct structure tests assess the validity of the model structure, by direct comparison with knowledge about the real system structure. There is no simulation involved and these tests are as such highly qualitative in nature (Forrester and Senge, 1980). Indirect structure (or structure oriented behavior) tests, on the other hand, assess the validity of the structure indirectly, by applying certain behavior tests on model-generated behavior patterns (Barlas, 1989b; Forrester and Senge, 1980). These

tests involve simulation, and may be applied to the entire model, as well as to isolated sub-models.

In a typical structure-oriented behavior test, the modeler makes a claim of the form: “if the system is operated under condition C, then the behavior B would be the result. For instance extreme condition behavior analysis is the most widespread structure-oriented test, to compare the results of the real system and the model, under the selected extreme conditions. The resulting behavior can be expressed in several forms. Glancing at a simple case, one can put forward a hypothesis on whether or not a specific state variable converges to equilibrium. In the latter case, a further hypothesis may state that the equilibrium has a zero or non-zero value. Moving one-step forward, the modeler can hypothesize a certain “dynamic pattern“ (like “gradual increase to a non-zero equilibrium”) and then test if the model yields the anticipated dynamic pattern. Such tests are very useful and informative tools to detect potential structural flaws in the system, in the formulations or causal relations.

In his M.S. thesis (1999), Kanar developed *Indirect Structure Testing Software (ISTS)*, computerized application, based on an algorithm, which tries to capture the dynamic pattern of the input signals, constructed by one-dimensional arrays. There are two major groups of patterns, linear-type and non-linear type. The linear patterns are composed of ‘zero’, ‘constant’, ‘linear growth’ and ‘linear decline’. The first type of pattern group is evaluated under standard curve fitting methods. ISTS is predominantly devoted to the non-linear segment of the generic patterns. The non-linear patterns are composed of ‘positive exponential pattern’, ‘negative exponential pattern’, ‘S-shaped pattern’, all possible combination of these patterns and some variants of these and ‘oscillation on a constant equilibrium’, ‘oscillation on a growing equilibrium’ and ‘oscillation on a declining equilibrium’. The abbreviations for all the patterns, which are used in the ISTS is given in Table 2.1.

The configuration of the non-linear part of the ISTS is based on the “Hidden Markov Model” (HMM). Roughly dissecting the formation of the Hidden Markov Model, in HMM-based pattern recognition, one-dimensional data is divided into segments and a sequence of feature vectors is extracted.

Kanar (1999) discusses statistical and syntactic pattern recognition methods and shape and waveform analysis techniques. In the statistical pattern recognition, a set of characteristic features are extracted from the pattern data. On the other hand, syntactic pattern recognition focuses on the relationships between simpler sub-patterns rather than the absence/presence of information or the numerical value of the features. Besides, in the techniques of shape and waveform analysis, contours and waveforms are described using shape representation techniques. Kanar (1999) selected HMM algorithm for dynamic pattern classification, which is mainly statistical pattern recognition, enhanced with the advantageous aspects of syntactic recognition. The selection of HMM is attributable primarily to the dynamic pattern nature of the data, and Kanar (1999) decided that the best algorithm for dynamic pattern recognition is HMM, among its competitors in the literature.

Although ISTS is an important contribution to the indirect structure tests in the structural validation of SD models, it has not been extensively tested yet. Initial tests in this research indicate that ISTS algorithm may have some performance problems in certain pattern types. The flaws emanate primarily from the certain weak points in the algorithm and the inadequacy of its trained database. The major aim of this research is to test ISTS so as to determine its weaknesses, make the necessary improvements in the algorithm and finally test it against human eye recognition with real data collected from the Istanbul Stock Exchange.

Improving the algorithm of the software is a critical issue, where the aggregate structure is to be evaluated and the potential structural problems are to be discovered first. Moreover, since the algorithm uses the built-in trained database for all of the nonlinear generic dynamic patterns, the inadequacies in the databases should be detected and eliminated to obtain more successful pattern detection. Hence, the new database is to be trained and it should provide better detection capability for the synthetic or real data. The ultimate goal is to detect the inherent dynamic patterns in the real data, as well as the human expert-eye can.

2. BASIC STRUCTURE OF INDIRECT STRUCTURE TESTING ALGORITHM

2.1. Continuous Density Hidden Markov Models

Statistical pattern recognition is based on the classification of feature vectors extracted from the data. Especially in dynamic pattern recognition problems, representation of the whole data with a single set of feature vectors is not desirable (Kanar, 1999). It is difficult to model the variations in the characteristics along the shape with a single vector (He and Kundu, 1991). Hidden Markov Models do not represent the whole data with a single feature vector. In HMM-based pattern recognition, one-dimensional data is divided into segments and a sequence of feature vectors is extracted. Kanar (1999) states that the primary goal of ISTS is to recognize the dynamic patterns, labeled as “generic dynamic patterns” in this study (Table 2.1 and Figure 2.1), in any set of time series, which are inherently one-dimensional and suitable for HMM implementation.

Regarding the continuous density HMM, the facts of a Markov process should be remembered. In a Markov process, a new state $s_t \in \{1, 2, \dots, N\}$ is entered at each step $t = 1, 2, \dots, T$, depending on an initial probability vector Π and a state transition matrix A , where

$$\Pi = \{\pi_i\}, \pi_i = \Pr(s_1 = i), \quad i = 1, 2, \dots, N \quad (2.1)$$

$$A = \{\alpha_{ij}\}, \alpha_{ij} = \Pr(s_{t+1} = j \mid s_t = i), \quad i, j = 1, 2, \dots, N. \quad (2.2)$$

Here,

T = length of state sequence,

N = number of states.

The resulting state sequence is denoted by $S = \{s_t \mid t = 1, 2, \dots, T\}$ and its realization probability is given by

$$\Pr (S | A, \Pi) = \pi_{s_1} \prod_{t=2}^T \alpha_{s_{t-1}, s_t} . \quad (2.3)$$

In a continuous density HMM the states of the process are not observed directly. The outcome of the process is a sequence of vectors $O = \{o_1, o_2, \dots, o_T\}$, where $o_t \in \mathfrak{R}^M$, M -dimensional Euclidean space (Kanar, 1999). The output vector is produced according to a probability distribution, depending on the current state. There are N number of observation-vector probability distributions, denoted by a vector B , where

$$B = \{b_j(o_t)\}, b_j(o_t) = \text{a posteriori density of observation vector } o_t \text{ in state } j.$$

Here, an assumption is made such that the observation densities are Gaussian. In this case, a conditional mean vector μ_j and a conditional covariance matrix V_j , determine the density corresponding to state j . This is denoted by the density $N(\mu_j, V_j)$.

A continuous density HMM is thus specified by the parameters A , Π , and B . The parameters can be compactly represented as a set denoted by $\lambda = (A, \Pi, B)$.

With this model, a process is described such that a transition probability from a step t to $t+1$ is only dependent on the state at step t .

The states of a HMM only reflect clustering properties of the features and should not be regarded as having a physical meaning (He and Kundu, 1991).

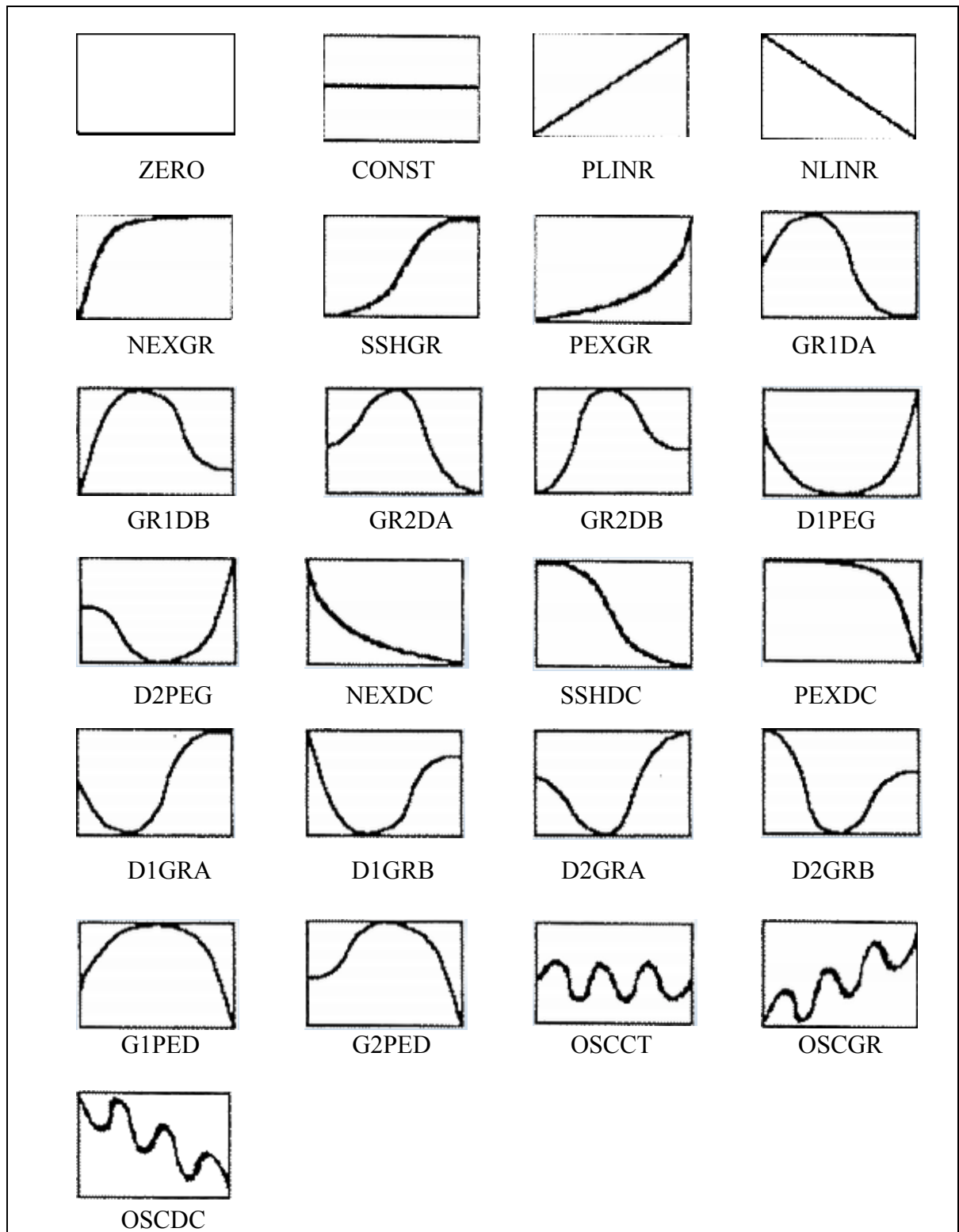


Figure 2.1. Generic dynamic patterns (Kanar, 1999)

Table 2.1. List of dynamic behavior classes (Kanar, 1999)

Abbreviation	Description
ZERO	Zero
CONST	Constant
PLINR	Linear with positive slope
NLINR	Linear with negative slope
NEXGR	Negative exponential growth
SSHGR	S-shaped growth
PEXGR	Positive exponential growth
GR1DA	Growth with decreasing rate followed by decline to equilibrium (growth level is less than decline level)
GR1DB	Growth with decreasing rate followed by decline to equilibrium (growth level is greater than decline level)
GR2DA	S-shaped growth and decline to equilibrium (growth level is less than decline level)
GR2DB	S-shaped growth and decline to equilibrium (growth level is greater than decline level)
D1PEG	Decline with increasing rate followed by positive exponential growth
D2PEG	S-shaped decline followed by positive exponential decline
NEXDC	Negative exponential decline
SSHDC	S-shaped decline followed by positive exponential decline
PEXDC	Positive exponential decline
D1GRA	Decline with increasing rate followed by growth to equilibrium (decline level is less than growth level)
D1GRB	Decline with increasing rate followed by growth to equilibrium (decline level is greater than growth level)
D2GRA	S-shaped decline and growth to equilibrium (decline level is less than growth level)
D2GRB	S-shaped decline and growth to equilibrium (decline level is greater than growth level)
G1PED	Decline with increasing rate followed by positive exponential decline
G2PED	S-shaped growth followed by positive exponential decline
OSCCT	Oscillation around constant mean
OSCGR	Oscillation around linearly growing trend
OSCDC	Oscillation around linearly declining trend

2.2. Algorithmic Structure and Function Relations

Focusing on the algorithmic structure of ISTS, the overall algorithm can be divided into two primary sections. The first one is the *training* part and the second is the *classification* section.

The *training* segment is the part, which develops the database for each type of generic patterns listed in Table 2.1. First of all, *training* algorithm is based on continuous density HMM, including the empirical state and segment numbers, which are determined by the algorithm-applier. Namely, Hidden Markov Model is trained via N states, dividing each sample time-series into T segments. In this research, state number N is defined as 7 and the segment number is determined as 12.

On the other hand, glancing at the classification segment, the primary goal is to test whether the hypothesized pattern is classified into the same pattern or not, based on the state-optimized likelihood functions over the Viterbi algorithm (He and Kundu, 1990). The details of the Viterbi algorithm are given at the end of this paragraph. Throughout the algorithm, first of all, the given signal is tested whether it falls into the *zero*, *constant* or *linear* categories (in the improved version, linear type will also be tested under HMM). The other nonlinear generic patterns are tested under the structure of HMM. There are 25 different generic dynamic patterns defined in this study (Figure 2.1). With the exception of *zero* and *constant* patterns, remaining twenty-three patterns will be tested under HMM. However, ten of them have mirror images with respect to the x-axis and *oscillation* pattern may display growing or declining behaviors. Therefore at the end, there are eleven major pattern classes, which are tested under HMM. The first ten *growth*, *decline* or *growth-decline* combinations and *oscillation* detection have some differences from each other. The analysis for *growth*, *decline* or *growth-decline* combinations is a standard process based on the state-optimized likelihood values, depending on features evaluated under each segment, such as mean, slope and curvature. On the other hand, the *oscillation* pattern class needs pre-processing and autocorrelation analysis will be applied to catch the first peak in the oscillation. After the pre-process, oscillation is also tested in the standard manner, that is to say by HMM-oriented pattern recognition and classification.

Viterbi Algorithm (See Kanar, 1999 for discussion)

Step 1. Initialization For $1 \leq i \leq N$,

$$\delta_1(i) = \pi_i b_i(o_i) \quad (2.4)$$

$$\psi_1(i) = 0 \quad (2.5)$$

Step 2. Recursive computation

For $2 \leq t \leq T$ for $1 \leq j \leq N$

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(o_t) \quad (2.6)$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] \quad (2.7)$$

Step 3. Termination

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \quad (2.8)$$

$$s^*_T = \arg \max_{1 \leq i \leq N} [\delta_T(i)] \quad (2.9)$$

Step 4. Tracing back the optimal sequence

For $t = T-1, T-2, \dots, 1$

$$s^*_{t+1} = \psi_{t+1}(s^*_{t+1}) \quad (2.10)$$

P^* is the state-optimized likelihood function, and

$S^* = \{s1^*, s2^*, \dots, sT^*\}$ is the optimal state sequence.

In the first step of the algorithm, for each state at segment 1, state likelihood functions $\delta_1(i)$ and transitional probabilities ($\psi_1(i)$) are defined. In the second step, the recursive computation phase, state likelihood functions for each segment t , $t = 2, \dots, T$, at each state j , $j = 1, \dots, N$, are calculated. In addition, transitional probability function values are computed at each segment and each state. The calculation in the second step is terminated in the third step and state optimized likelihood value is found at the last segment T , by taking the maximum of the state likelihood values at each state. Likewise, the optimum state is also found at segment T . At the fourth and the last phase, optimal state sequence is found by tracing back to previous segments, $T = T-1, T-2, \dots, 1$. As a result, Viterbi algorithm provides the state optimized likelihood function P^* and the optimal state sequence $S^* = \{s1^*, s2^*, \dots, sT^*\}$.

2.3. Training Part

The training procedure is the process in which the model parameter set $\lambda = (A, \Pi, B)$ for a class is adjusted so that the state optimized likelihood function is maximized for the "training set" of that class. (A training set for a given class is a collection of dynamic patterns, all of which are "noisy" versions of the basic dynamic pattern that define that class). At the end of the training phase, the algorithm essentially "learns" the basic dynamic pattern of a given class. For Gaussian density functions, there is a "segmental K-means algorithm" that converges to the state-optimized likelihood function (Juang and Rabiner 1990, Kanar 1999). Segmental K-means algorithm can be outlined briefly as below.

Given an initial model $\hat{\lambda}^0$, calculate

$$\hat{\lambda}^{k+1} = \arg_{\lambda^k} \max \left\{ \max_s p(O, S^* | \hat{\lambda}^k) \right\} \quad (2.11)$$

iteratively until $\hat{\lambda}^{k+1} = \hat{\lambda}^k$, where k is the iteration number. Maximization of the state-optimized likelihood $p(O, S^* | \lambda)$ in (2.11) for each training observation sequence is achieved using the Viterbi algorithm. Given a model $\lambda = (A, \Pi, B)$ and an observation sequence $O = \{o_1, o_2, \dots, o_T\}$, the Viterbi algorithm finds the state-optimized likelihood function and the optimal state sequence. At each iteration, the optimal state sequences are assigned to the observation vectors of each training sample. The new states are again used to estimate new model parameters. The iteration proceeds until none of the state assignments change at the end of the maximization (Kanar, 1999).

As the training process proceeds, the database for every generic pattern is evaluated and a *trained* file is generated, which are going to be used in the classification part. The *trained* file includes the samples for each pattern, and their major features, the mean, the slope and the curvature information of every sample time-series, (i.e. an HMM for each pattern class), included in the database of each pattern.

2.4. Classification Part

In the classification part, major goal is to test whether a pattern classification hypothesis is validated or not. As a result of the training procedure, we obtain one HMM for each class. We denote the P models by λ_p , $p = 1, 2, \dots, P$. When a signal O of unknown class is given, we calculate $p(O, S^* | \lambda_p)$ for each class $p = 1, 2, \dots, P$ using the Viterbi algorithm. The goal is to classify the given signal in one of the known pattern classes. The classification is based on the state-optimized likelihood function, which is a measure of how well the input signal is representative of a given class. However, the likelihood values for different classes by themselves may not be suitable for direct comparison, depending on the degree of dispersion of the training feature vectors within their clusters. He and Kundu (1990) normalize the state-optimized likelihood function for a class by dividing it to the mean of the likelihoods of the training set used for that class. Kanar moves one step further and takes into account the variation of the likelihood function values of the training samples within classes. Kanar's criterion is the state-optimized likelihood value of the input signal normalized by the mean, slope and the standard deviation of the likelihood function of the training set, $\hat{\sigma}_p$. We denote the normalized likelihood function by $h(O, S | \lambda_p)$ such that,

$$h(O, S | \lambda_p) = \frac{p(O, S^* | \lambda_p) - m_p}{\sigma_p} \quad (2.12)$$

Here,

$$m_p = \frac{1}{n} \sum_{i=1}^{n_p} p(O_i^p, S^* | \lambda_p) \quad p = 1, 2, \dots, P \quad (2.13)$$

and,

$$\hat{\sigma}_p = \left\{ \frac{1}{n_p - 1} \sum_{i=1}^{n_p} (p(O_i^p, S^* | \lambda_p) - m_p)^2 \right\}^{1/2} \quad p = 1, 2, \dots, P \quad (2.14)$$

where, n_p is the number of training samples used for class p and O_i^p is the observation (feature) vector sequence extracted from the i^{th} training sample of class p .

When an input does not belong to any of the classes, it should not be erroneously classified into one of the classes. The equation of $h(O, S | \lambda_p)$ also allows us to define a common lower rejection region. Since the likelihood values of the training set are approximately normal, $\hat{\sigma}_p$ is an estimator for the population standard deviation. We take the ad hoc value of 3 times the standard deviation of the likelihood functions within a class as our rejection region. Thus, if the outcome of the classifier $h(O, S | \lambda_p)$ is less than -3.0 for all P classes, it is classified that the input signal into the “none” class (Kanar, 1999). Therefore, the classification rule becomes:

$$\text{choose} \begin{cases} \underset{p}{\operatorname{argmax}} [h(O, S^* | \lambda_p)] & \text{if } \max_p (h(O, S^* | \lambda_p)) > -3.0 \\ \text{none} & \text{otherwise} \end{cases} \quad (2.15)$$

2.5. Problem issues in the original algorithm

Throughout the initial tests, we have perceived that there may be some inefficiency in the algorithm and databases. During the assessment, we have figured out that there are three types of weaknesses or inadequacies in the algorithm.

The first one is the lack of detecting oscillatory patterns due to an inadequacy in the algorithm. This situation stemmed primarily from a lack of normalization in the pre-process for oscillation detection algorithm. This is the first issue to be resolved and improved, as it may be an inadequacy in the algorithm or bug in the coding. We have also figured out that the linear approximation coding in the first peak detection algorithm in the pre-process of oscillation classification may create flaws, when slight noisy oscillatory time-series are to be tested. Therefore, using the difference function directly and analysis of the slope sign changes more accurately, should probably solve the oscillatory pattern detection problem.

Another inadequacy in the original ISTS was the *linear* pattern detection, which was evaluated by binary, zero-one, acceptance or rejection criterion. However, our tests showed that linear class must also be trained under HMM and tested with regard to its state-optimized likelihood functions. This was a major design issue for the *linear* pattern recognition and we decided to include it into the HMM-based pattern detection algorithm.

The last and maybe the most significant problem in the older version of ISTS was the insufficiency of the database for most pattern classes. This is very significant in terms of real noisy data detection, resulting often in wrong classification. We have discovered that when the algorithm is asked whether the tested data falls in the hypothesized pattern class or not, even if its answer is satisfactory, it also classifies the same data in other generic patterns cited above. Insofar as the major goal of this study is concerned, with this type of errors, it is often impossible to detect and classify noisy real data under ISTS. So, we hold the view that in addition to ISTS algorithm improvement, the database should also be expanded and calibrated so as to provide an optimal classification with real noisy data. Therefore, new data entries for each pattern are generated and entered.

In the upcoming chapter, the resolution for the above problems and inadequacies were investigated and implemented. Without these improvements, it was almost impossible to increase the reliability of the algorithm and code, so further testing with real noisy data was meaningless.

3. IMPROVEMENT METHODOLOGY

3.1. Solution for the Oscillatory Pattern Recognition Problem

As mentioned above, we have discovered that an oscillation detection problem exists in ISTS. In order to overcome the trouble, there should be an improvement in the algorithm. By inference, we have recognized some weakness in the algorithm, in terms of normalization in the pre-process for discovery of the first peak in the data. During the preparation process in classification phase, the characteristic properties of the data should be determined from its normalized version, so that its slope, mean, and curvature may be comparable with the embedded values in the trained file. However, most probably emanating from a coding inefficiency, the required normalization was not included into the algorithm and it created undesired state optimized likelihood function results, even for the very smooth oscillatory patterns. Including normalization in the algorithm has solved the problem. Considering the following example oscillatory pattern in Figure 3.1, the difference of the normalized likelihood functions may be observed, depending on normalization included in the algorithm.

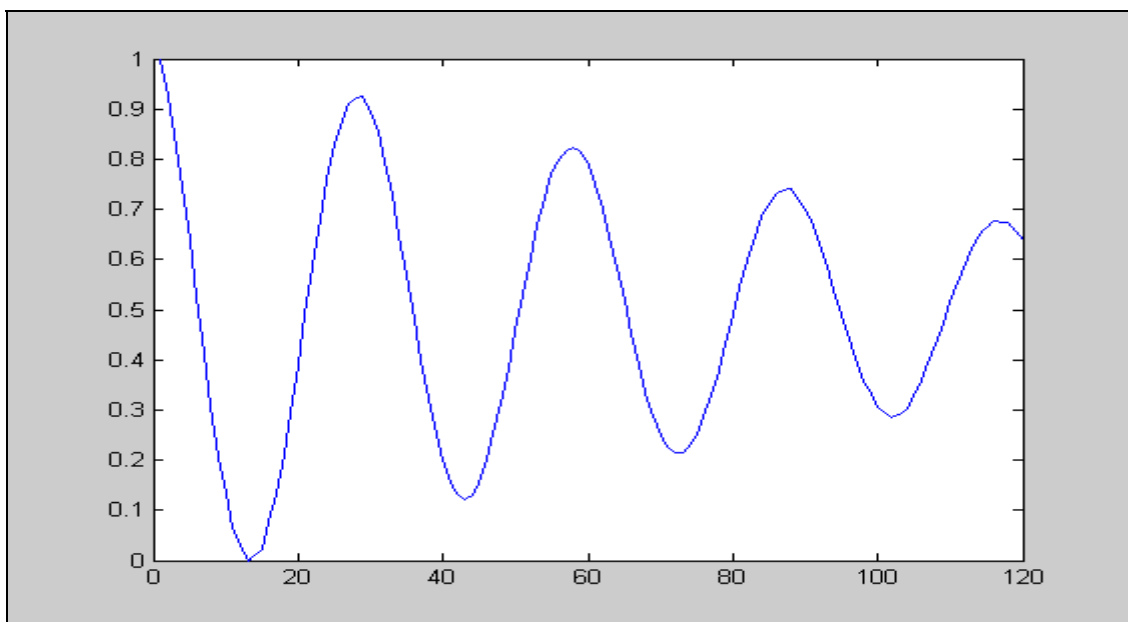


Figure 3.1 Oscillatory pattern example

Table 3.1. Likelihood values for the pattern in Figure 3.1, without normalization

ZERO	-10.00	GR1DA	-14.69	SSHDC	-27.15	G2PED	-22.75
CONST	-10.00	GR1DB	-18.17	PEXDC	-21.71	OSCCT	-19.05
PLINR	-10.00	GR2DA	-12.82	D1GRA	-12.91	OSCGR	-10.00
NLINR	-10.00	GR2DB	-14.72	D1GRB	-18.20	OSCDC	-25.56
NEXGR	-11.97	D1PEG	-15.39	D2GRA	-13.88		
SSHGR	-17.61	D2PEG	-16.34	D2GRB	-14.52		
PEXGR	-17.09	NEXDC	-18.63	G1PED	-19.78		

Table 3.2. Likelihood values for the pattern in Figure 3.1, with normalization

ZERO	-10.00	GR1DA	-14.69	SSHDC	-27.15	G2PED	-22.75
CONST	-10.00	GR1DB	-18.17	PEXDC	-21.71	OSCCT	-10.22
PLINR	-10.00	GR2DA	-12.82	D1GRA	-12.91	OSCGR	-10.00
NLINR	-10.00	GR2DB	-14.72	D1GRB	-18.20	OSCDC	-25.56
NEXGR	-11.97	D1PEG	-15.39	D2GRA	-13.88		
SSHGR	-17.61	D2PEG	-16.34	D2GRB	-14.52		
PEXGR	-17.09	NEXDC	-18.63	G1PED	-19.78		

In this context, it is observed that normalization error has a significant effect on the “oscct” pattern result – which is actually the proper pattern class for this test data. Therefore, we pursue with-normalization route.

However, for the oscillatory pattern recognition, the lack of normalization was not the only problem. There are some other algorithmic problems and also the coding issues. First of all, let us summarize the preprocessing mechanism of oscillation detection. HMM likelihood of oscillation class, OSCCT, is evaluated for using the autocorrelation function of the input data. Sample autocorrelation function $r(k)$ of a signal $y(k)$ is defined as,

$$r(k) = \frac{\sum_{t=1}^{K-k} (y(t) - \bar{y})(y(t+k) - \bar{y})}{\sum_{t=1}^K (y(t) - \bar{y})^2} \quad k = 1, 2, \dots, K \quad (3.1)$$

where $\bar{y}$ is the mean of data points in sequence $y(k)$. Here, the number of data points in the sample autocorrelation K is taken to be the half of data points in $y(k)$.

If the oscillations do not have a trend component, the autocorrelation functions of oscillatory patterns typically exhibit a behavior such that the function drops from around 1.0 to negative values and then elevates to positive region to make a peak at the period of

oscillation. This is a characteristic shape, which is very different than the autocorrelation function of a non-oscillatory time-sequence.

For the recognition purpose, the part of the signal until the first peak is taken and the rest is discarded. For this reason, Kanar (1999) developed two heuristic peak detection algorithms. The first one locates the peak region using piecewise linear approximations, while the second algorithm searches for the peak at different resolutions.

We have combined Kanar's two different algorithms. The revised combined peak detection algorithm is:

- Step 1. Set $i = 1$.
- Step 2. $d(i) = \text{sign}(r(i+1) - r(i))$. (3.2)
- Step 3. if $d(i) \leq 0$ then peak is found at i and go to step 5, else $i = i + 1$.
- Step 4: repeat until the peak is found or the end of array $r(i)$. Go to step 5.
- Step 5. end the algorithm.

The autocorrelation function is truncated at the first peak detected by the algorithm and the remaining portion is discarded. Denoting the truncated autocorrelation sequence until first peak by $r(k)$, the number of points in this sequence may be too small for HMM. Hence, a new sequence is generated (the same size as the model generated data) with values obtained by sampling at intermediate and equally spaced time intervals, using Lagrangian interpolation by Kanar (1999). Each new point in this sequence is obtained by interpolation using two preceding and the following points from $r(i)$. As a small change, we have decided to apply linear interpolation with two preceding and following points instead of Lagrangian interpolation, due to its simplicity and smoother interpolation and extrapolation capability.

After the completion of the improvement phase for the oscillatory pattern class, the normalized likelihood values for the series in Figure 3.1 are given in Table 3.3.

Table 3.3. Likelihood values for the pattern in Figure 3.1, with improved algorithm

ZERO	-10.00	GR1DA	-14.69	SSHDC	-27.15	G2PED	-22.75
CONST	-10.00	GR1DB	-18.17	PEXDC	-21.71	OSCCT	-5.42
PLINR	-10.00	GR2DA	-12.82	D1GRA	-12.91	OSCGR	-10.00
NLINR	-10.00	GR2DB	-14.72	D1GRB	-18.20	OSCDC	-25.56
NEXGR	-11.97	D1PEG	-15.39	D2GRA	-13.88		
SSHGR	-17.61	D2PEG	-16.34	D2GRB	-14.52		
PEXGR	-17.09	NEXDC	-18.63	G1PED	-19.78		

Here, it is clearly observed that oscillatory pattern class detection goal is achieved. For the *oscct* pattern our improved algorithm yielded a normalized likelihood value of -2.09 , which is satisfactory compared to the acceptance level of -3 . In addition, there is no confusion with the other classes, which was also desirable.

3.2. Solution for the erroneous classification problem

As may be recalled, hypothesis testing (classification) part of ISTS tests whether a given time-series can be classified into the hypothesized pattern class or not. However, if the question asked to the software is of form: "which class does this time-series fall into?" the algorithm has difficulty in dealing with this harder question. Due to the structure of the algorithm, false classifications may come into scene. For instance, if an S-shaped growth candidate pattern was tested under ISTS, the algorithm can properly decide if it falls into that category or not. However, dealing with the above more general question, it may classify the candidate into several other classes as well, which is an intrinsic trouble in the algorithm. Therefore, a sharper separation of the generic dynamic pattern classes is the most critical issue throughout the improvement portion of this study. The following examples illustrate this harder problem of pattern recognition. After the improvement and expansion of trained database, the first example shows the results for *oscct* pattern, given in Figure 3.1.

Table 3.4. Likelihood values for the pattern in Figure 3.1, with the improved database

ZERO	-10.00	GR1DA	-11.21	SSHDC	-25.19	G2PED	-13.54
CONST	-10.00	GR1DB	-15.69	PEXDC	-18.36	OSCCT	-1.61
PLINR	-12.74	GR2DA	-9.07	D1GRA	-8.34	OSCGR	-10.00
NLINR	-12.74	GR2DB	-10.49	D1GRB	-13.37	OSCDC	-7.24
NEXGR	-11.36	D1PEG	-11.65	D2GRA	-9.52		
SSHGR	-16.98	D2PEG	-12.77	D2GRB	-12.00		
PEXGR	-17.98	NEXDC	-14.87	G1PED	-13.31		

As may be recalled from the previous section, the oscillatory pattern recognition problem was improved in two steps, after normalization inclusion into the algorithm and first peak detection algorithm improvement phases. As a reminder, in the original version of ISTS, for the example the oscct pattern, the result was -19.1 , after normalization it increased to -10.2 and after the peak detection algorithm improvement the result has risen to -5.4 . However, the results were still out of the acceptance border of -3 . Here, after the improvement of trained database of oscillation pattern, the result for oscct reached -1.6 , which is inside the border of -3 , and no confusion with other patterns exists (Table 3.4).

The following example illustrates another erroneous classification problem in ISTS.

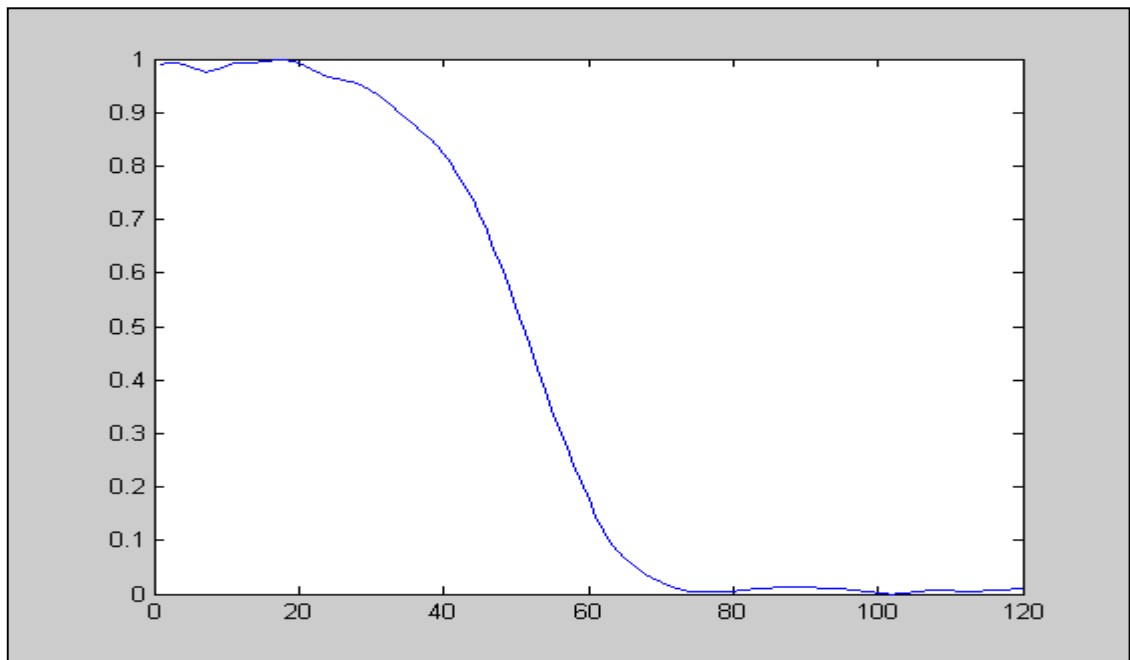
Figure 3.2. Example for *S-shaped decline* pattern

Table 3.5. Likelihood values for the pattern in Figure 3.2 (original version of ISTS)

ZERO	-10.00	GR1DA	-1.29	SSHDC	-0.77	G2PED	-13.92
CONST	-10.00	GR1DB	-5.27	PEXDC	-19.69	OSCCT	-31.31
PLINR	-10.00	GR2DA	-0.89	D1GRA	-12.57	OSCGR	-10.00
NLINR	-10.00	GR2DB	-10.42	D1GRB	-16.96	OSCDC	-31.31
NEXGR	-14.74	D1PEG	-10.13	D2GRA	-7.63		
SSHGR	-28.91	D2PEG	-8.92	D2GRB	-4.74		
PEXGR	-24.64	NEXDC	-8.87	G1PED	-10.64		

Here, it is clearly seen that the hypothesis for *s-shaped decline* has passed the barrier of -3.0 (SSHDC yielding a value of -0.7). However, it should also be noticed that the pattern classes *gr1da* (-1.3) and *gr2da* (-0.9) are also candidates for the classification. Thus, the example shows that false classifications may arise even with rather smooth and distinct samples of any generic pattern class.

After the database is expanded by training with extensive new patterns from each class, the likelihood values for the time-series in Figure 3.2 become as given in Table 3.6

Table 3.6 Likelihood values for the pattern in Figure 3.2 (improved version of ISTS)

ZERO	-10.00	GR1DA	-3.35	SSHDC	0.31	G2PED	-10.85
CONST	-10.00	GR1DB	-4.07	PEXDC	-8.99	OSCCT	-7.24
PLINR	-10.00	GR2DA	-3.59	D1GRA	-12.35	OSCGR	10.00
NLINR	-10.00	GR2DB	-8.32	D1GRB	-12.99	OSCDC	-8.55
NEXGR	-12.48	D1PEG	-10.51	D2GRA	-9.69		
SSHGR	-26.66	D2PEG	-11.18	D2GRB	-7.56		
PEXGR	-13.54	NEXDC	-5.64	G1PED	-10.12		

So as to overcome such problems, the database for each generic pattern should be improved by different sample normalized arrays. Hence, our fundamental goal in this phase is to improve and expand the databases of each pattern (excluding the mirror patterns and oscillation with trends), as much as possible without including any time-series, which may fall into transition territory of the patterns. It is very important not to include any indistinct shaped series in the pattern databases, since the algorithm must avoid biasing some of the patterns, which are trained with indistinct patterns. In that context, the sample series must display concrete and distinct shapes, and any confusion possibilities must be avoided. One more critical aspect of the sample series inclusion to the databases is

avoiding of excessively noisy series. Considering the shape of each sample series, inclusion of harsh noise may create detection problems and unavoidable confusion in the classification of the related data. However, for the sake of obtaining a richer database, simple and quite smooth noise may be included into the sample series. As a reminder, the primary goal of this algorithm improvement work is to detect the inherent patterns in the noisy real data. Without keeping track of these two significant criteria during the expansion phase, ultimate improvement goal is unreachable.

One other important problem may arise, even if the two critical rules are obeyed. We call this problem overtraining, and investigate its effects in a separate section. If a pattern class is trained with too many data sets, then that pattern class may be dominant, in the sense that it will not only properly classify its own patterns, but also improperly claim ownership of the patterns that belong to some neighboring classes. The details and problematic issues will be expressed later in Section 3.5.

3.3. Inclusion of Linear Pattern to HMM Based Algorithm

In the unimproved algorithm of ISTS, we define a pattern $Y(k)$, $k = 1, 2, \dots, K$, as linear, if each data point in the normalized sequence $y(k)$, $k = 1, 2, \dots, K$, remains within a band around the least-squares line-fitted to the data points:

$$|y(k) - f(k)| < 0.05 \quad \text{for } k = 1, 2, \dots, K \quad (3.3)$$

where $f(k)$ is the fitted least-square-line evaluated at point k .

But the above simple classification rule may have some problems when tested with real data, as will be shown below. Since linear pattern is one of the most recognizable generic patterns in the real world with some noise additions, we think that more accurate classification approach is needed instead of simple binary zero-one comparison method.

Like the nonlinear generic dynamic patterns, we determined to include the linear pattern class into HMM-based algorithm.

Instead of the least square band approach, including the linear pattern into the HMM based algorithm yields results comparable with other different growth (or decline) patterns. Regarding the original ISTS algorithm, class relations are defined for the patterns treated with the HMM approach. For instance, *nexgr*, *sshgr* and *pexgr* (and their mirror patterns) are defined as a related group, which represents the growth (decline) super-class. Therefore, first of all linear pattern is defined in the algorithm in the same way as it belongs to general growth (or decline) super-class. Then, the *positive linear growth* and its mirror pattern *negative linear growth* (or linear decline) should also be recognized by the class recognition structuring table defined in the algorithm.

After the incorporation of the linear pattern to HMM algorithm, in the classification procedure, the feature vector of any test array is compared with the feature vectors embedded in the trained set. Since the HMM classification scheme needs a training for each class, for the linear pattern, a new large-scale database (just as for the other patterns) was constructed.

Following example illustrates how linear pattern likelihood values were included and the linear pattern classification becomes comparable with all the other patterns.

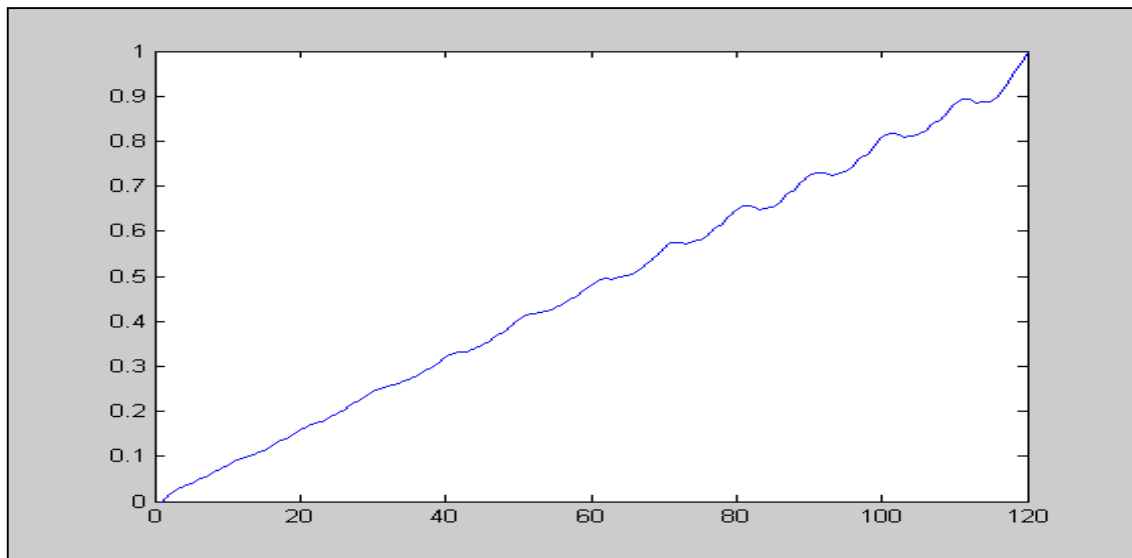


Figure 3.3. Example 1 for positive linear growth pattern

Table 3.7 Likelihood values for the pattern in Figure 3.3 (old version of ISTS)

ZERO	-10.00	GR1DA	-25.25	SSHDC	-33.35	G2PED	-13.92
CONST	-10.00	GR1DB	-13.11	PEXDC	-22.35	OSCCT	-31.31
PLINR	10.00	GR2DA	-11.14	D1GRA	-12.57	OSCGR	-13.25
NLINR	-10.00	GR2DB	-10.09	D1GRB	-16.96	OSCDC	-31.31
NEXGR	-14.74	D1PEG	-11.15	D2GRA	-14.46		
SSHGR	-28.91	D2PEG	-12.11	D2GRB	-15.45		
PEXGR	-24.84	NEXDC	-10.09	G1PED	-13.35		

Table 3.8 Likelihood values for the pattern in Figure 3.3 (new version of ISTS)

ZERO	-10.00	GR1DA	-15.80	SSHDC	-27.88	G2PED	-10.19
CONST	-10.00	GR1DB	-16.39	PEXDC	-18.08	OSCCT	-7.24
PLINR	-1.07	GR2DA	-9.92	D1GRA	-13.86	OSCGR	-3.08
NLINR	-12.74	GR2DB	-8.95	D1GRB	-13.59	OSCDC	10.00
NEXGR	-8.39	D1PEG	-11.66	D2GRA	-12.65		
SSHGR	-20.77	D2PEG	-10.81	D2GRB	-10.05		
PEXGR	-14.40	NEXDC	-11.46	G1PED	-13.31		

As may be observed from the Table 3.7 and Table 3.8 the *positive linear growth* is properly classified under HMM. Although classification in both algorithms yields correct results, the second new algorithm is much more coherent and healthy, because it yields a metric compatible with entries for the other classes seen in the Tables 3.6 and 3.7. (Note that the PLINR value yields 10 and ZERO, CONST and NLINR VALUES yield the same -10 values in the old version; so that the old algorithm is unable to generate metric compatible results with the other classes). The difference will be even clearer and more important in the case of noisier time-series.

Let us observe the difference between the original and improved algorithms with a noisy example. Another synthetic noisy data is given in the Figure 3.4. The respective normalized likelihood values are given in the Table 3.9 and Table 3.10, for the original and improved algorithms.

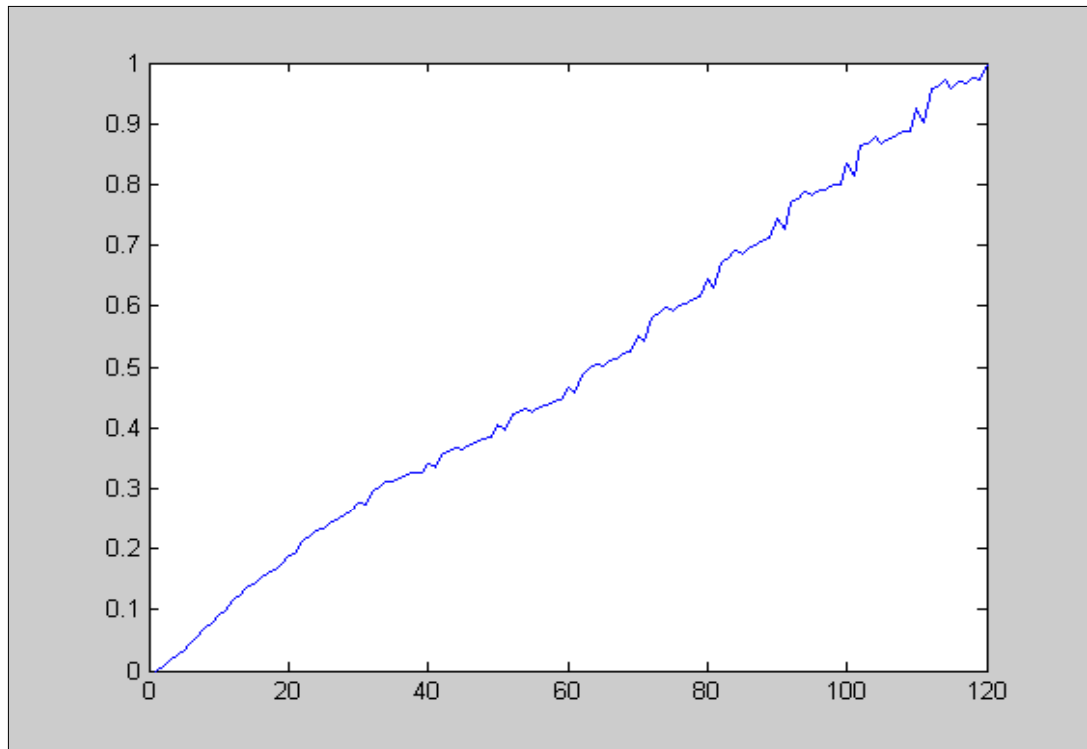


Figure 3.4. Example 2 for positive linear growth pattern

Table 3.9. Likelihood values for Figure 3.4 (original algorithm)

ZERO	-10.00	GR1DA	-19.77	SSHDC	-36.55	G2PED	-15.96
CONST	-10.00	GR1DB	-16.27	PEXDC	-34.01	OSCCT	-26.79
PLINR	-10.00	GR2DA	-10.32	D1GRA	-14.13	OSCGR	-21.23
NLINR	-10.00	GR2DB	-19.50	D1GRB	-18.89	OSCDC	-10.00
NEXGR	-14.01	D1PEG	-13.01	D2GRA	-22.22		
SSHGR	-29.15	D2PEG	-18.17	D2GRB	-21.75		
PEXGR	-28.77	NEXDC	-14.35	G1PED	-16.73		

Table 3.10. Likelihood values for Figure 3.4 (improved algorithm)

ZERO	-10.00	GR1DA	-15.70	SSHDC	-28.29	G2PED	-9.08
CONST	-10.00	GR1DB	-16.34	PEXDC	-18.40	OSCCT	-9.18
PLINR	-1.59	GR2DA	-10.00	D1GRA	-12.03	OSCGR	-5.50
NLINR	-12.74	GR2DB	-9.75	D1GRB	-13.22	OSCDC	-10.00
NEXGR	-9.36	D1PEG	-11.00	D2GRA	-11.94		
SSHGR	-19.29	D2PEG	-10.68	D2GRB	-9.04		
PEXGR	-10.73	NEXDC	-11.40	G1PED	-13.31		

As can be observed from Tables 3.8 and 3.9, PLINR pattern is detected by the improved algorithm, with a value of -1.6 and the result is metric compatible with the results of other patterns. On the other hand, the original algorithm is unable to detect PLINR pattern, the normalized likelihood value being -10 .

3.4. Smoothing

Regarding the hypothesis testing of the noisy data in ISTS, it is a hard task to teach the algorithm to detect the intrinsic generic pattern buried in noise. In terms of the database preparation for training domain, the negative effects of noise in the sample series in the database were explained before in details. To demonstrate the excessive noise in a real series, let us consider the time-series in the following graph, obtained from the Istanbul Stock Exchange (ISE).

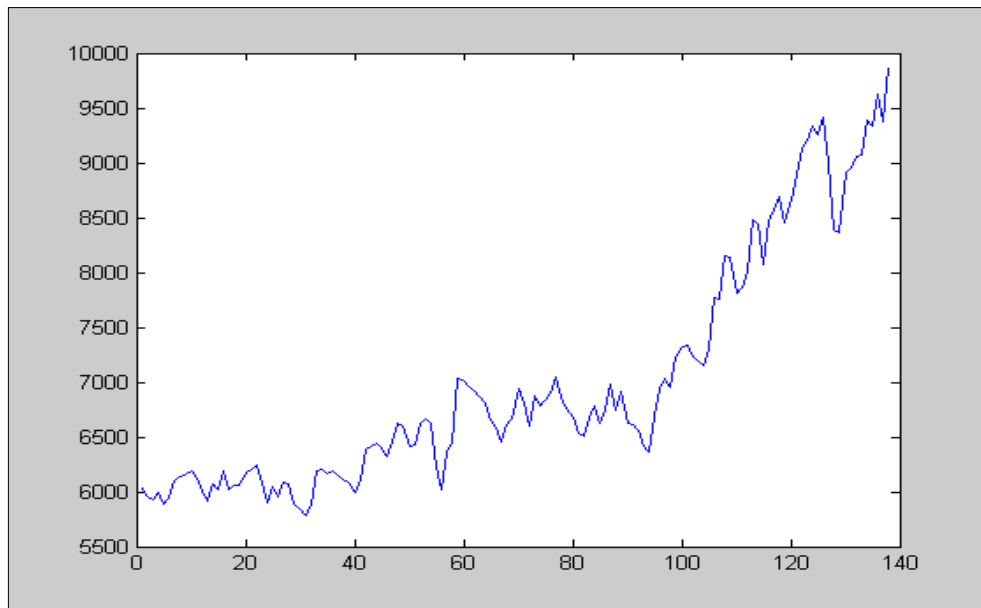


Figure 3.5. Example for the noise in a real time-series (Akbank Market Value, in million dollars, between March10, 2004 and July 23, 2004)

Here, different experts may assert different generic patterns, which the series may include inherently. However, without any smoothing, and the results for such pattern hypotheses will most likely fail in the ISTS. For the sake of inherent generic dynamic pattern recognition for such noisy time-series, there should be a smoothing option in the ISTS, as an auxiliary and complementary part.

There exist different smoothing methods in the literature, predominantly grouped under filtering (averaging) and local regression. The most common used algorithms for these two types of smoothing classes are the “moving average method” and the “Savitzky-Golay smoothing”. By inference, each smoothing method requires a span, which defines a window of neighboring points to include in the smoothing calculation for each data point. This window moves across the data set as the smoothed response value is calculated for each predictor value. Focusing on the size of the span, a large span increases the smoothness but decreases the resolution of the smoothed data set, while a narrow span decreases the smoothness but increases the resolution of the smoothed data set. The optimal span value depends on the data set and the smoothing method, and usually requires experimentation to find.

Choosing and including one of the two alternatives into the overall algorithm, a high-noised sample time-series (picked up from the ISE, Akbank’s market worth, in million dollars, between June12, 2004 and December 12, 2004) are used for visual inspection. Figure 3.6 and 3.7 represent samples for the two smoothing methods. For the sake of comparability, in both of the methods, the span length is determined as 3, and for the Savitzky-Golay method the degree of polynomial is set to 1.

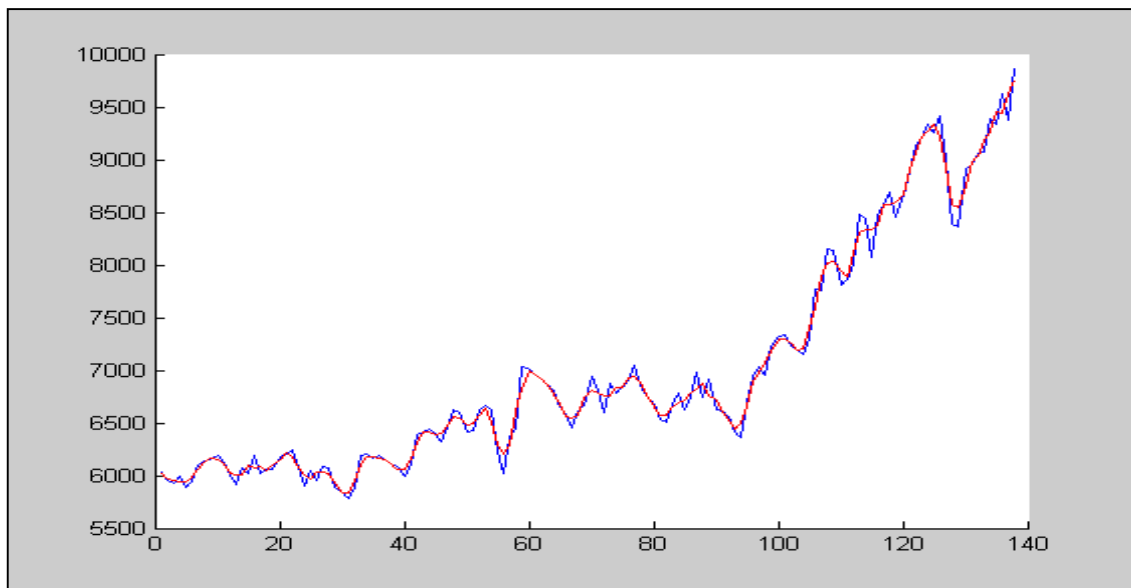


Figure 3.6. Real noisy data and its smoothed version with Savitzky-Golay algorithm (span length =3, degree of polynomial = 1)

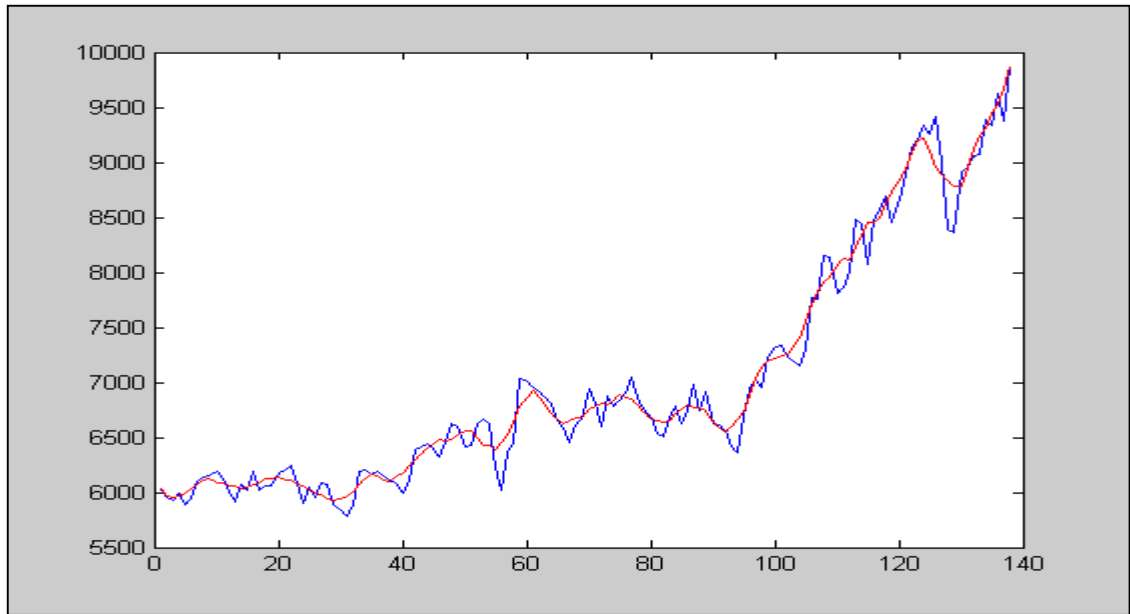


Figure 3.7. Real Noisy data and its smoothed version with moving-average algorithm
(span length = 3)

As may be observed from the figures 3.6 and 3.7, the smoothing capability of the moving average algorithm is higher than the Savitzky-Golay procedure. Since the ultimate goal of the improvement part of this study is to cope with detecting the innate generic patterns in the real noisy time-series, rather than dealing with local noise filtering Savitzky-Golay procedure, we decided to apply moving average filtering (smoothing) and included an optional smoothing algorithm into the ISTS. The smoothing algorithm is optional, since direct inclusion of smoothing into the general flowchart of ISTS may yield oversmoothing results for slight noisy or noise-free patterns; there may be information and recognition capability losses in the algorithm. If the user thinks that the smoothing process is redundant, he/she does not have to apply the smoothing algorithm. As an extra feature for the smoothing algorithm, we decided to allow adjustment for the degree of smoothing. Since the moving average algorithm works by averaging fixed number of neighboring points, the number of points is left to the users' choice.

3.5 Solution for the overtraining problem

As explained in brief in the preceding sections, *overtraining* of database must be avoided. Structurally, ISTS is exposed to such *overtraining* risk. A pattern class may be called as overtrained, either if many excessive noisy series or many series with patterns that may fall into transitory zones of generic pattern classes, are included in its trained database or if it has been trained with excessively high number of sample series.

Regarding the role of noise discussed above, an additional side effect is the *overtraining* risk in order to recognize noisy data. However, more significant risk for *overtraining* is the transitory zone patterns. We define a transitory zone pattern as a dynamic pattern, which includes certain dynamic patterns in its different segments, resembling to two or more generic patterns. The transitory zone patterns are the most crucial factors in the *overtraining* case. If the database includes several number of such training time-series, then *overtraining* is unavoidable and confusion with other neighboring patterns during the classification procedure may come into scene. Following figure is an example for the transitory zone pattern, between the SSHGR and D2GRA.

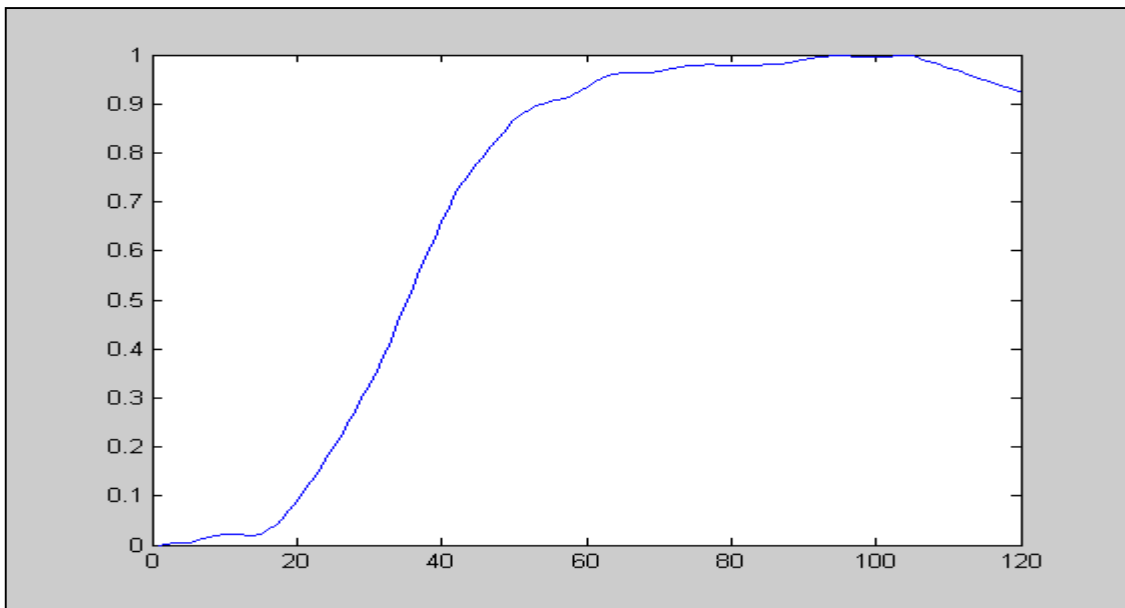


Figure 3.8. Example for transitory zone pattern between SSHGR and D2GRA

If many such patterns are included into databases of a given class, confusion in the classification is unavoidable. Therefore, we were very careful during the construction of the databases, to avoid *overtraining* problem.

Another significant issue is the database-size differences between the patterns. Since in the database expansion phase, the same or very close shaped patterns are not included simultaneously, size of the databases for each type of pattern must not differ much. Therefore, during the database expansion process, we have expanded the size of each eleven database in the interval of 1000-1300 data sets, which were generated by hand and visual inspection. Hence, we avoided the *overtraining* trouble, in terms of the different database sizes. As will be seen in the following chapter, the final training database is set after having tested against *overtraining* and reconstructed several times.

4. TESTING THE NEW ALGORITHM AND TURING DESIGN

4.1. Preliminary Test by Synthetic Data

After the improvement methodology was finalized and a comprehensive trained database has been constructed, testing the revised and rearranged algorithm phase is next. As a reminder, our major goal is to test the improved ISTS, tested under many noisy time-series from the real life. However, beforehand, it is necessary to test whether there remains any significant problematic situations in the classification scheme. For this reason, prior to the real data testing, we test the algorithm with a set of synthetic data specifically designed to check for potential problematic classes. There are 76 sets of synthetic data, constructed for 19 different generic patterns. We have decided to test the *growth-then-decline* classes, under a grouped format. Namely, *GR1DA* and *GR1DB* is tested under *GR1D*, while *GR2DA* and *GR2DB* is analyzed under a super-class of *GR2D*. The same analysis procedure was applied for its mirror patterns too. As defined before, Kanar (1999) determined 25 different generic dynamic patterns. By the reduction of 4 pattern types from the *growth-then-decline* and *decline-then-growth* classes, as well as the exclusion of *constant* and *zero* classes, we have had 19 different generic patterns in hand, for the preliminary test and final Turing test. Following two examples are different samples of our preliminary test, which can be fully traced in Appendix B.

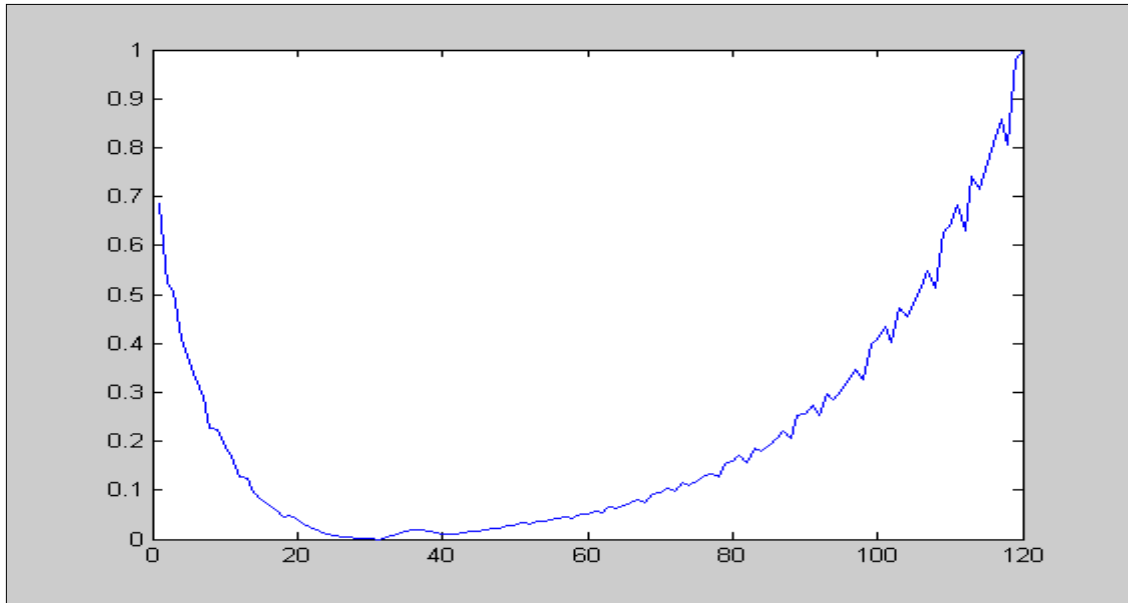


Figure 4.1. Synthetically generated example for D1PEG

Table 4.1 Likelihood values for the pattern in Figure 4.1 (improved ISTS)

ZERO	-10.00	GR1DA	-12.25	SSHDC	-20.06	G2PED	-14.35
CONST	-10.00	GR1DB	-13.88	PEXDC	-18.25	OSCCT	-5.26
PLINR	-10.44	GR2DA	-9.30	D1GRA	-4.21	OSCGR	-5.31
NLINR	-12.74	GR2DB	-6.34	D1GRB	-7.14	OSCDC	-10.00
NEXGR	-13.34	D1PEG	-1.17	D2GRA	-3.88		
SSHGR	-12.93	D2PEG	-3.64	D2GRB	-4.70		
PEXGR	-4.81	NEXDC	-8.26	G1PED	-13.31		

Here, one can observe that classification of the pattern is successful and consistent with the hypothesis D1PEG. Looking at the original version of ISTS, the normalized density values for Figure 4.1 is given in the following table.

Table 4.2 Likelihood values for the pattern in Figure 4.1 (original ISTS)

ZERO	-10.00	GR1DA	-18.28	SSHDC	-23.56	G2PED	-20.54
CONST	-10.00	GR1DB	-20.52	PEXDC	-19.11	OSCCT	-18.84
PLINR	-10.00	GR2DA	-18.57	D1GRA	-13.15	OSCGR	-10.00
NLINR	-10.00	GR2DB	-19.98	D1GRB	-14.74	OSCDC	-16.64
NEXGR	-16.87	D1PEG	-11.21	D2GRA	-15.24		
SSHGR	-22.21	D2PEG	-13.18	D2GRB	-16.49		
PEXGR	-10.15	NEXDC	-15.84	G1PED	-23.55		

As may be observed from Table 4.2, the original ISTS was not adequate to detect the inherent D1PEG pattern from the time-series in Figure 4.1. Hence, improved ISTS is more reliable than the original one in detecting patterns embedded in the noisier time-series.

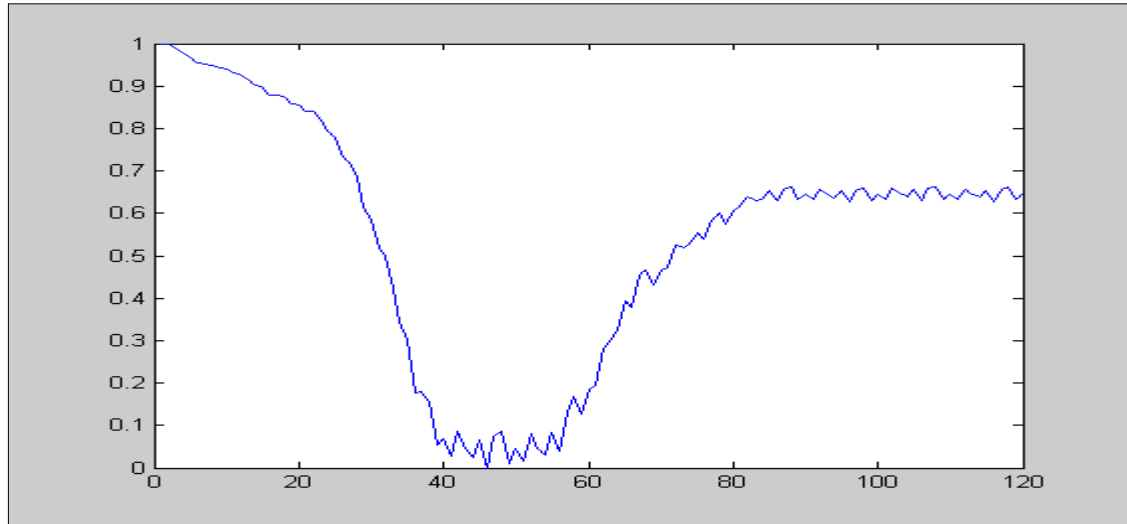


Figure 4.2. Synthetically generated example for D2GRB

Table 4.3. Likelihood values for the pattern in Figure 4.2 (improved ISTS)

ZERO	-10.00	GR1DA	-11.25	SSHDC	-17.53	G2PED	-9.33
CONST	-10.00	GR1DB	-11.97	PEXDC	-13.78	OSCCT	-6.01
PLINR	-10.11	GR2DA	-11.34	D1GRA	-11.92	OSCGR	-10.00
NLINR	-9.25	GR2DB	-10.06	D1GRB	-6.51	OSCDC	-7.24
NEXGR	-11.27	D1PEG	-10.61	D2GRA	-6.74		
SSHGR	-21.01	D2PEG	-8.16	D2GRB	-0.05		
PEXGR	-14.20	NEXDC	-10.88	G1PED	-10.92		

Here, again the algorithm has accepted the hypothesis and the test result validated that the given time-series pertain to the generic pattern class of *D2GRB*. On the other hand, turning back to original ISTS, Table 4.4 shows the normalized density values for the data set in Figure 4.2. As may be seen in the Table 4.4, the original ISTS is again incapable of clarifying noisier series, even if the embedded pattern is obvious.

Table 4.4. Likelihood values for the pattern in Figure 4.2 (original ISTS)

ZERO	-10.00	GR1DA	-22.43	SSHDC	-28.84	G2PED	-23.54
CONST	-10.00	GR1DB	-20.61	PEXDC	-15.95	OSCCT	-18.82
PLINR	-10.00	GR2DA	-25.21	D1GRA	-14.16	OSCGR	-10.00
NLINR	-10.00	GR2DB	-22.69	D1GRB	-16.52	OSCDC	-16.41
NEXGR	-15.54	D1PEG	-22.30	D2GRA	-18.34		
SSHGR	-26.55	D2PEG	-18.84	D2GRB	-13.05		
PEXGR	-19.95	NEXDC	-23.32	G1PED	-26.29		

4.2. Turing Test Design

It is not unusual for models to be validated by comparing the model's performance to a human expert's opinion. For example, O'Leary (1987) discusses the validation of expert systems, Sauder and Westerman (1983) evaluate a decision support system and Charnes et al. (1988) test a data envelopment analysis model, all using comparisons to human experts. This approach is generally known as a 'Turing test'. Turing (1950) proposed that the intelligence of computing machinery be assessed using a 'game'. The computer and a human counterpart would be asked to perform the same tasks. Then an external interrogator would be asked if the two could be distinguished. Since many models are designed for complex decisions, typically a Turing test would be administered to a number of human actors or a number of data points would be administered to a single human-computer pair. O'Leary (1991) performed statistical analysis of tests, such as Turing test comparison, with particular emphasis on the use computer intensive statistics.

Prior to focus on the statistical analysis of the test, first of all the Turing test procedure should be constructed. Our basic aim in this part of study is to prepare a test for different time-series patterns from the real-life. In order to include noisy real-life time-series, which may represent different generic dynamic patterns, we have selected Istanbul Stock Exchange (ISE) as our data-collection pool. As its inherent nature, stock market data are dynamic and noisy. As a footnote, in this study our aim is not to prove that the stock-market data include generic dynamic patterns; instead, we use ISE data pool just to collect a large sample of real time-series. Hence, our analysis cannot be used for any forecasts for the ISE or any stock market related variables. We have gathered a pile of long-run data, all of which are daily (every point in the series represent a daily value).

We chose on average 5 to 7 representative samples for every generic dynamic pattern as explained before we test 19 patterns instead of 25, excluding *zero* and *constant* in addition to grouping *gr1da* and *gr1db* as *gr2da* and *gr2db*. Totally, we determined to expose 106 data sets to the experts and the algorithm. Here, we determined to construct to test, based on the answers of the experts (selections from the generic pattern table). Focusing on the subjects of our Turing test approach, we have determined to use 12 subjects, 6 from the academic life and 6 from the professional businesses. Our academic subject group is composed all from the Industrial Engineering Department of Boğaziçi University; one professor, one graduate student, two system dynamics graduate students, and two senior undergraduate system dynamics students. As to our professionals, all of them are data and equity analysts, who are all experts in the ISE-related brokerage houses.

After the determination of the real data series for pattern recognition and classification, it is important to shuffle the candidate series in a random manner, to avoid the bias (conditioning) that the subjects may be exposed to, even though ISTS has no bias in that context. After the random shuffling process applied for the candidate time-series, the general structure of our Turing Test procedure is formed.

The next stage is how to display the candidate series to the subjects. As may be recalled, during the methodology improvement part of this study, we have incorporated additional smoothing algorithm for ISTS. We decided to display the original raw nature of the candidate series as well as the smoothed version to subjects. One other crucial point here is the degree of smoothing. The degree of smoothing for the time-series is determined with respect to the degree of smoothing, which ISTS' results provided good classification in prior tests. We have figured out that ISTS' classifications turn out to be successful around fourth to fifth degree smoothing, on the average. Therefore we have also displayed fourth degree-smoothed versions of the candidate time-series to the human subjects, as assistance for easier detection, like we applied in ISTS. A sample Turing test data sheet is shown in Figure 4.3. Complete Turing test patterns are provided in Appendix C.

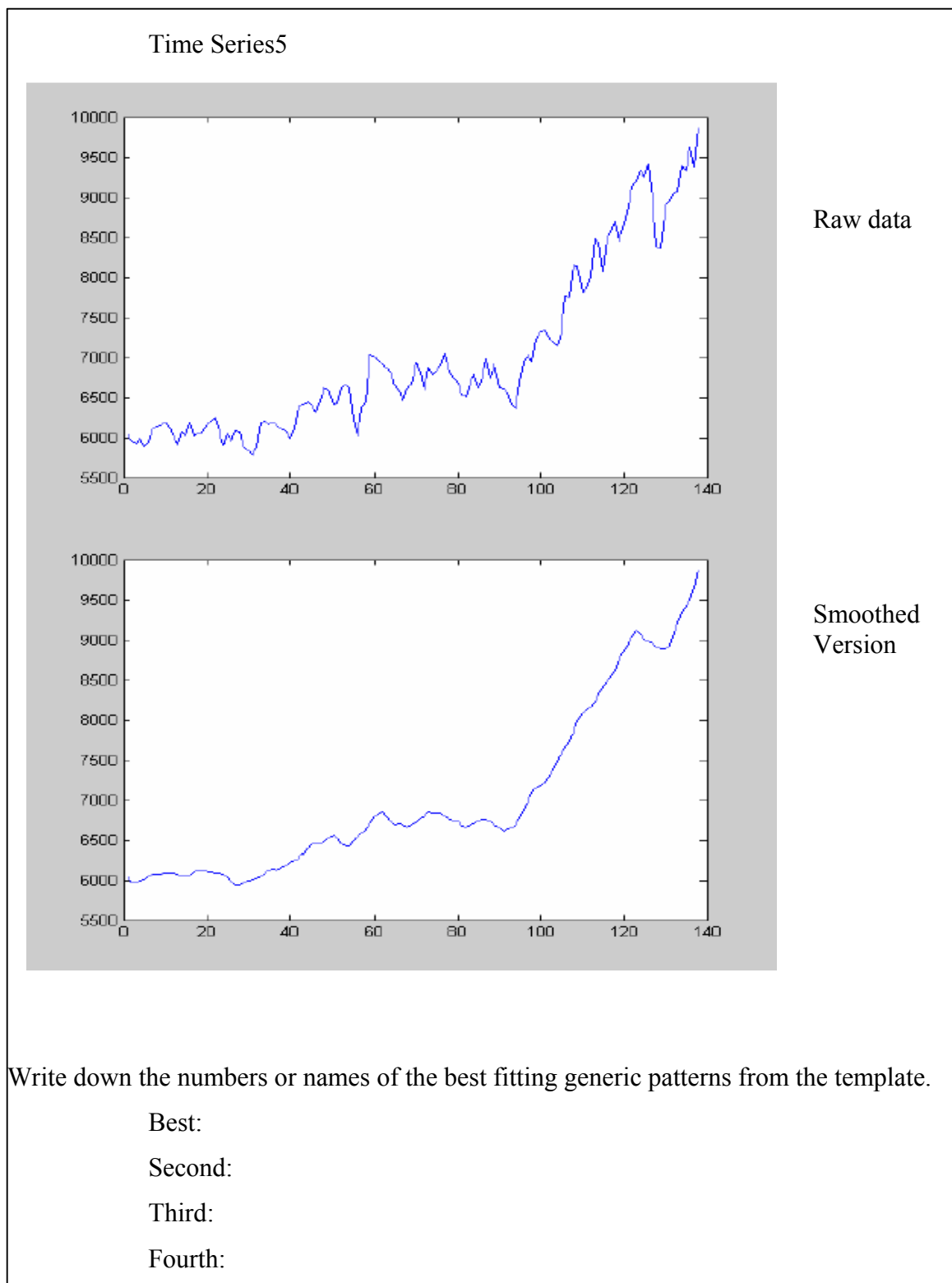


Figure 4.3. Sample Turing Test Data Sheet

The last phase of the design for the Turing Test is how to collect the responses of the subjects and compare with the results of ISTS. We decided to instruct the subjects to write down their first four picks from the template. As we know that the candidate sample time-series does belong to a generic dynamic pattern, at least one selection from the pattern list is mandatory, and the other three are optional. A subject is able to select one, two, three or four different generic patterns from the table. For the ease of detection and a speedy response for the subjects, we have enumerated every pattern with a code. The subject is able to note down the name of the patterns or their code from 0 to 20 (See Figure 5.1 and Table 5.1).

After the design of the test, subjects are supposed to complete the test without any break; therefore the tests were performed under a controlled environment. However, there is no time-restriction to complete the test. The results from all subjects are collected to be later compared with the responses of ISTS.

Firstly, it was observed that no subject has responded with four different alternative classes; while the maximum response number for any time-series was three, but even the three-response case is very rare. Subjects' response for each question is either one or two responses. Results of the tests will be shown and analyzed in Chapter 5.

4.3. Smoothing (Filtering) Effect

As cited before, smoothing has significance for the dynamic pattern detection for real data that include inherent noise. Focusing on our moving averaging filtering algorithm, the degree of smoothing can be adjusted with respect to the nature of the data and its noise level. Therefore, smoothing is an optional step for the user of ISTS, according to the level of noise in the candidate time-series.

To illustrate, we compare ISTS' results on the raw data and its fourth degree smoothed version. The following example displays the effect of smoothing in the detection of the pattern.

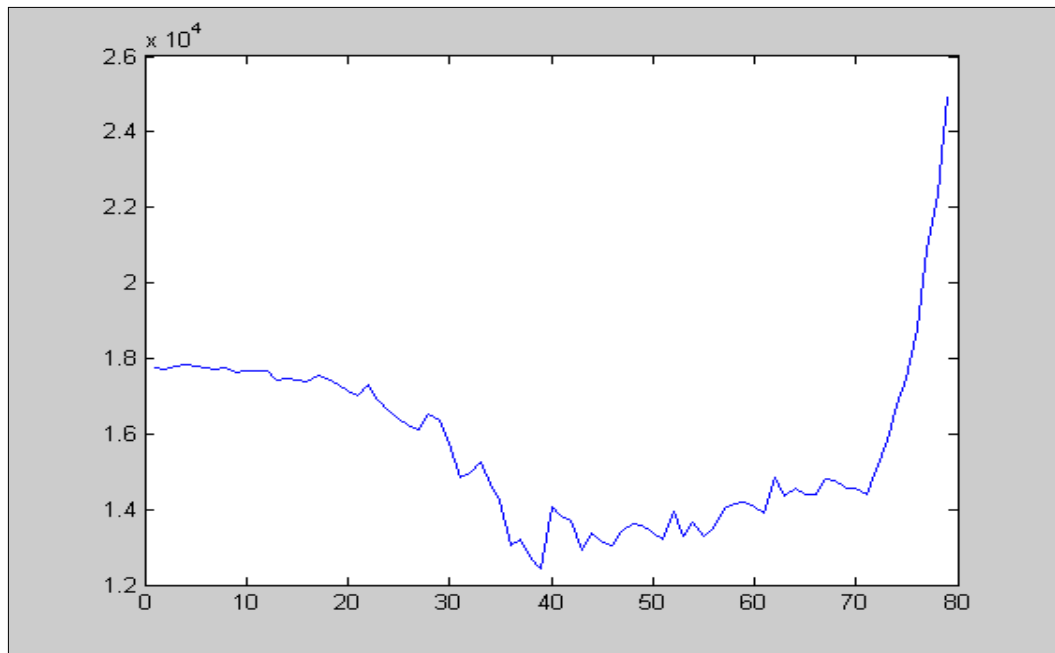


Figure 4.4. Noisy real data, hypothesized as the generic dynamic pattern D2PEG

Table 4.5. Likelihood values for the pattern in Figure 4.4, without smoothing

ZERO	-10.00	GR1DA	-14.91	SSHDC	-22.49	G2PED	-12.97
CONST	-10.00	GR1DB	-14.18	PEXDC	-19.05	OSCCT	-8.55
PLINR	-12.51	GR2DA	-11.45	D1GRA	-12.87	OSCGR	-10.00
NLINR	-11.97	GR2DB	-15.02	D1GRB	-13.22	OSCDC	-9.24
NEXGR	-14.93	D1PEG	-11.30	D2GRA	-10.14		
SSHGR	-28.80	D2PEG	-8.62	D2GRB	-12.27		
PEXGR	-15.83	NEXDC	-9.69	G1PED	-13.31		

Although naked eye may detect the generic pattern in the hypothesized series, ISTS is unable to make an exact classification. (All class values are inconclusive, nevertheless the best result is obtained for *D2PEG*). The next figure shows the smoothed version of the same time-series, with moving averages of degree four.

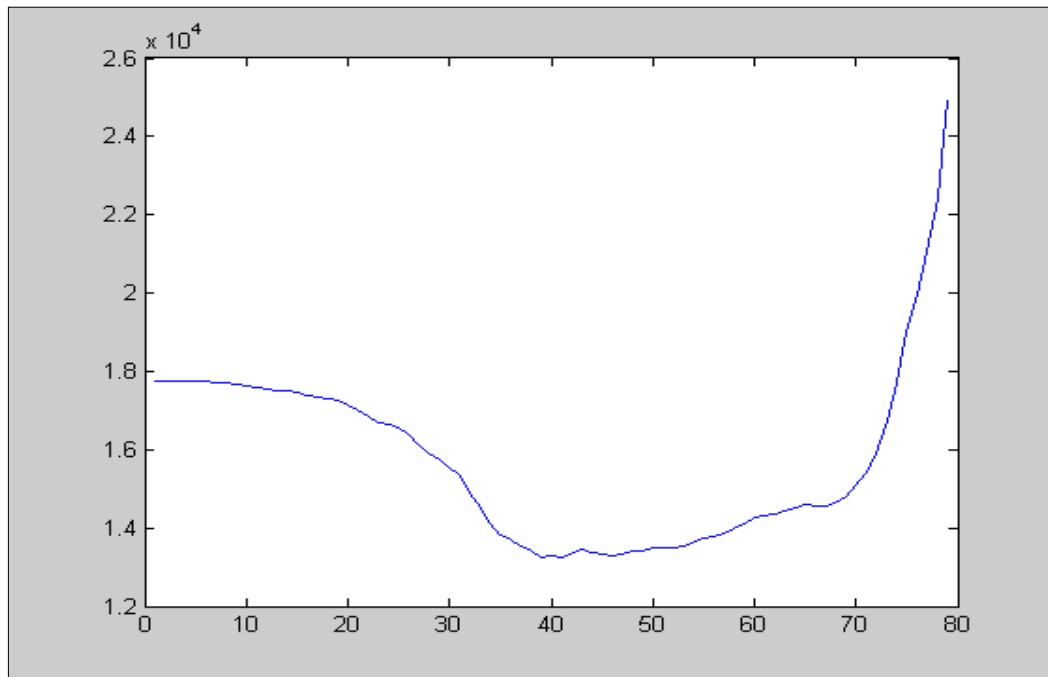


Figure 4.5. 4th degree smoothed version of the series displayed in Figure 4.4

Table 4.6. Likelihood values for the pattern in Figure 4.4, after smoothing

ZERO	-10.00	GR1DA	-11.88	SSHDC	-15.37	G2PED	-12.73
CONST	-10.00	GR1DB	-12.03	PEXDC	-16.59	OSCCT	-7.00
PLINR	-12.35	GR2DA	-9.43	D1GRA	-10.51	OSCGR	-10.00
NLINR	-9.82	GR2DB	-8.87	D1GRB	-10.00	OSCDC	-9.24
NEXGR	-13.73	D1PEG	-5.69	D2GRA	-6.56		
SSHGR	-25.74	D2PEG	-2.49	D2GRB	-7.45		
PEXGR	-11.14	NEXDC	-7.89	G1PED	-13.31		

As may be observed from Figure 4.4, the correct hypothesis is approved by ISTS, if the data is smoothed under the moving averaging algorithm with degree 4. As we cited before, noise cannot be taught to an algorithm or a computer code. Inherently, noise is random and training databases should not include random noise, for the sake of avoiding false classification. It is better to use smoothing rather than overtraining with noisy training set. Throughout the development and testing processes, smoothing idea assisted us to avoid overtraining.

In terms of the human eye perception of the generic dynamic patterns, we hold the view that smoothing has a positive influence for detection. Although we have not

considered to test explicitly the effect of smoothing on the subjects, we presented both raw and smoothed versions to the subjects in the Turing tests. All results will be presented and analyzed in the next Chapter.

5. STATISTICAL ANALYSIS OF TURING TEST

5.1. Test Results and Statistical Methods

As stated in the previous chapter, responses of the subjects are qualitative figures which standard statistical tests are hard and invalid to apply. As known, ISTS has quantitative likelihood function values, but human subjects cannot generate such values, they simply give the identification code of the classified pattern (Table 5.1). Therefore, throughout the analysis part, we cannot use any quantitative tests such as ANOVA etc. directly.

Table 5.1. Codes for the generic dynamic patterns

ZERO	0	NLINR	6	G1PED	12	OSCCT	18
CONST	1	NEXDC	7	G2PED	13	OSCGR	19
PLINR	2	SSHDC	8	D1GR	14	OSCDC	20
NEXGR	3	PEXDC	9	D2GR	15		
SSHGR	4	GR1D	10	D1PEG	16		
PEXGR	5	GR2D	11	D2PEG	17		

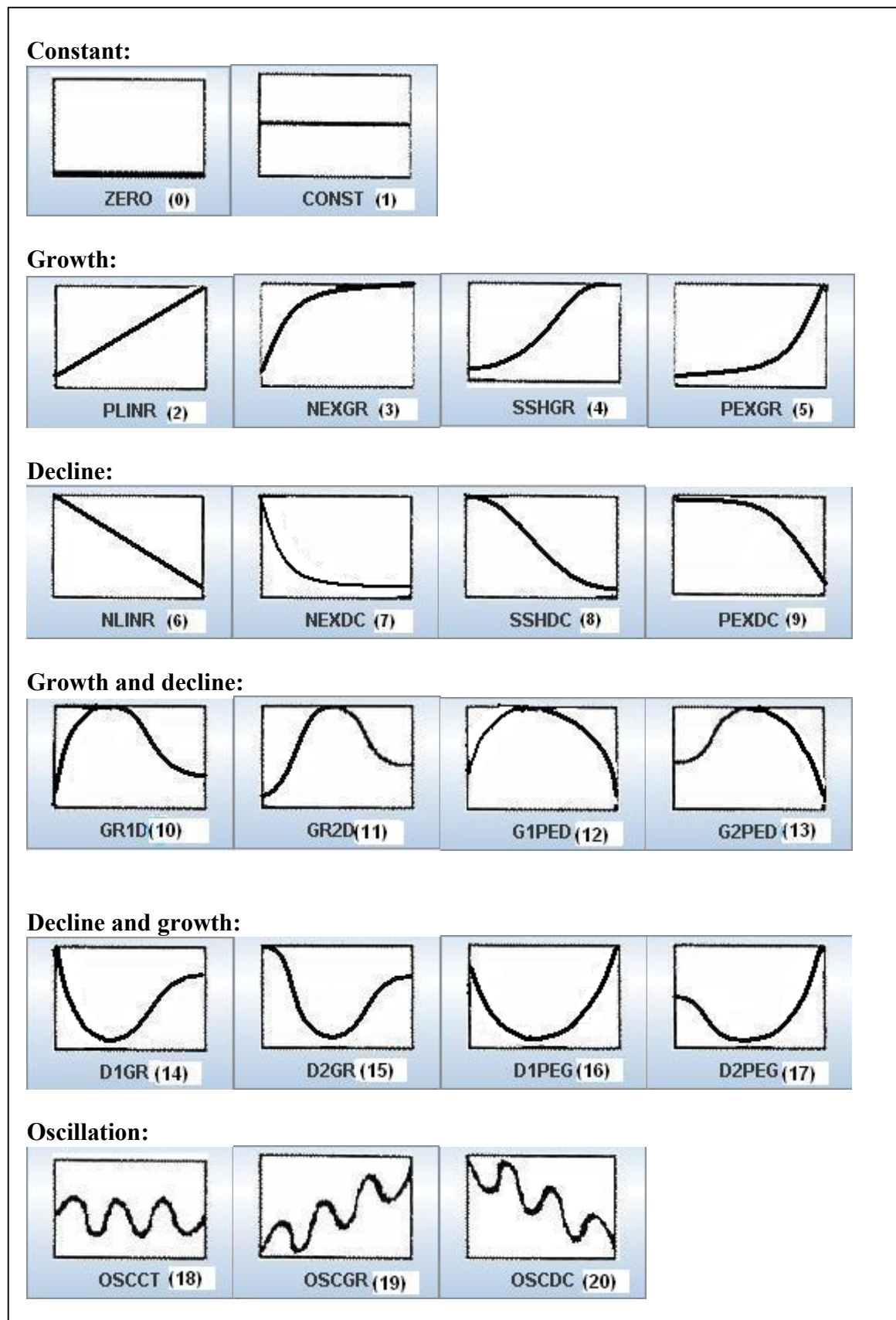


Figure 5.1. Generic dynamic patterns in Turing test

Hence, we contemplated a series of relative tests, whereby the qualitative numbers are transformed into quantitative figures, in order to attach any statistical meaning and have solid results about the performance of ISTS and the deviation among the sample space of subjects and ISTS. The following sets of tables are the results from ISTS and responses from the subjects. Each number from 0 to 20 represents the related generic pattern classification.

To exemplify the responses of the subjects, the following example test sheet shows the response of the first subject for the first time series (see Appendix C for details).

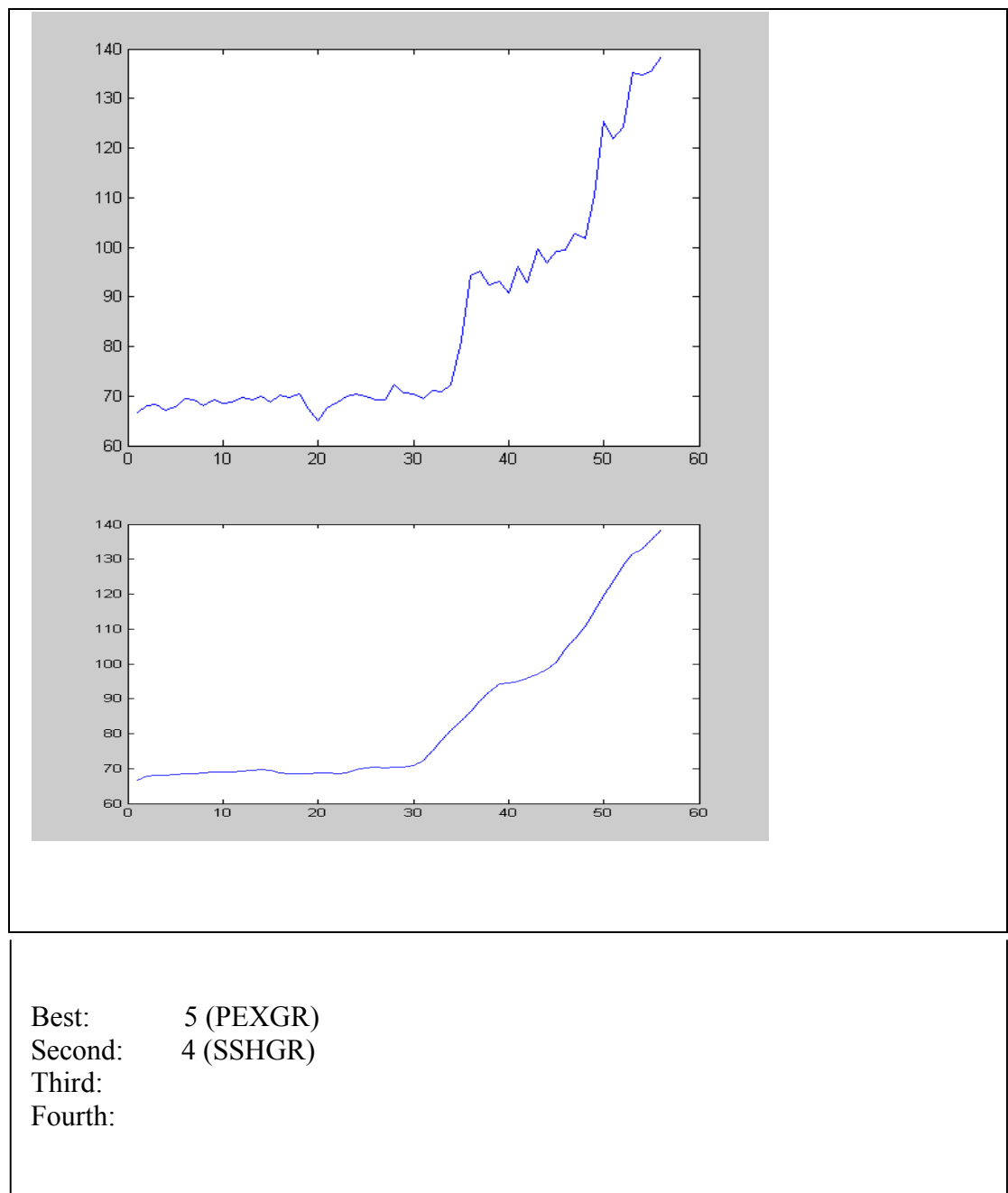


Figure 5.2. Sample subject response sheet (Time Series 1, Subject 1)

Table 5.2. Responses of ISTS and subjects to Turing test queries (1-26)

[illegible]

Table 5.4. Responses of ISTS and subjects to Turing test queries (53-78)

Results of the ISTS and the subjects	Best-fit order													
Time Series	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	
53	2	19	19	2	2	2	2	2	2	2	2	2	2	
	15, 19, 5	2												
54	3	3	3	3	3	4	3	3	11	3	3	3	3	
	19, 18, 11					3	19		3					
55	19	2	4	4	4	4	19	4	4	2	4	19	4	
	18, 2, 3					19						2		
56	20	20	20	20	20	20	20	20	20	20	20	20	20	
	18, 15, 12	18												
57	6	6	6	6	6	6	6	6	6	6	6	6	6	
	20, 11, 7													
58	19	19	19	19	19	18	19	19	19	19	19	18	19	
	18, 15, 16			18		19								
59	8	8	8	8	8	8	8	8	7	8	8	8	8	
	7, 20, 18					7, 10	20		8	7				
60	20	6	7	7	8	7	8	8	7	20	8	20	8	
	7, 18, 6	7				8	20	7		7	7	6		
61	18	18	18	18	18	18	18	18	18	18	18	18	18	
	20, 11, 15													
62	7	7	7	7	7	7	7	7	7	7	7	7	7	
	20, 8, 17													
63	17	16	16	17	15	17	17	16	16	16	16	16	15	
	15, 20, 16										17			
64	20	18	18	20	18	18	18	18	18	20	18	18	18	
	18, 11, 15					20								
65	19	19	18	19	19	18	19	19	19	18	19	19	19	
	18, 11, 15					19				19				
66	7	7	7	7	7	7	7	7	7	7	7	7	7	
	20, 8, 10			8		8								
67	16	16	16	16	16	16	16	16	17	16	16	16	17	
	17, 15, 19											17		
68	20	18	18	20	18	20	20	18	20	20	20	20	20	
	18, 11, 10	20	20					20						
69	11	13	11	10	13	11	10	13	13	13	15	13	10	
	10, 12, 20											12		
70	9	9	9	9	12	9	9	9	9	9	12	9	12	
	12, 15, 13					12						12		
71	19	20	19	19	19	19	19	19	19	19	19	19	19	
	15, 18, 14								4					
72	2	2	19	2	2	2	2	2	2	2	2	2	2	
	19, 15, 3	19				19								
73	17	17	17	17	16	17	17	16	16	17	16	16	17	
	20, 16, 18										17			
74	6	6	6	6	6	6	6	6	6	6	6	6	6	
	20, 15, 11													
75	19	3	19	3	4	3	19	3	3	4	3	3	4	
	3, 18, 15	2	3					2	4	2		2		
76	15	15	15	15	15	15	15	15	15	15	15	15	15	
	20, 18, 7													
77	3	3	3	3	3	3	3	3	3	3	3	3	3	
	19, 4, 15													
78	4	4	4	4	4	4	19	4	4	4	4	11	4	
	3, 19, 18	3				3	3		3		3	10		

Here, the qualitative raw data in the above tables will be converted to zeros and ones, with respect to the type of relative analysis. Zero (0) means compared responses for a given sample are the same with the other while one (1) means a difference.

As mentioned before, our analysis will be based on relative transformation of the qualitative data, such that it is possible to analyze statistically the transformed quantitative data. The statistical tests to be considered are the standard t-test with the relatively transformed data of each subject and ISTS, two different outlier analysis, modal analysis and relative analysis with respect to the designer (our) classifications.

5.2. Standard t-test

First, our basic statistical test focused on the comparison of the means of subject (or ISTS) responses, which are transformed into binomial variables. Standard t-test procedure is our first analysis, in the statistical analysis and comparison portion. Every combination of pairs is investigated therefore potential existence of significant difference in their means. To comply with the requirements of this test, two sample subjects (or ISTS) are picked up from the sample pool, and relative to each other, the first selections for each pattern are compared. If the responses are the same for a pattern, then the statistic for that question is set to 0, otherwise it is 1. Since the analysis works on a true-false case, the distribution of the statistic has a binomial behavior. Therefore, throughout the t-test procedure, our basic assumption is the transformed data for the comparative responses are binomially distributed.

As a statistical background, the binomial distribution for a random variable X with parameters n and p represents the sum of n independent variables x which may assume the values 0 or 1. If the probability that each x assumes the value 1 is equal to p , then the mean of each variable is equal to $1 \cdot p + 0 \cdot (1-p) = p$, and the variance is equal to $p(1-p)$. By the additional properties for independent random variables, the mean and variance of the binomial distribution are equal to the sum of the means and variances of the n independent x variables, so

$$\mu_x = np \quad (5.1)$$

$$\sigma_x = \sqrt{np(1-p)} \quad (5.2)$$

If we know that the count X of "successes" in a group of n observations with success probability p has a binomial distribution with mean np and variance $np(1-p)$, then we are able to derive information about the distribution of the sample proportion $\hat{p}$, the count of successes X divided by the number of observations n . By the multiplicative properties of the mean, the mean of the distribution of X/n is equal to the mean of X divided by n , or $np/n = p$. This proves that the sample proportion $\hat{p}$ is an unbiased estimator of the population proportion p . The variance of X/n is equal to the variance of X divided by n^2 , or $(np(1-p))/n^2 = (p(1-p))/n$.

For the t-test analysis of the subject responses, we ask whether the means of them differ significantly or not. For the binomial distribution of the transformed values x , we test the hypothesis

$$H_0 : \hat{p} = 0 \quad (5.3)$$

$$H_1 : \hat{p} > 0 \quad (5.4)$$

$$\hat{p} = \frac{\sum_{i=1}^N x_i}{N} \quad \text{where } N = 106 \quad (5.5)$$

and the standard deviation for the sample mean may be defined as

$$\sigma_{\hat{p}} = \frac{\sqrt{\hat{p}\hat{q}\nu}}{\nu} \quad (5.6)$$

where ν is the degree of freedom and equals to $N - 1$.

Here $\sqrt{\hat{p}\hat{q}\nu}$ is the standard deviation of the population, where for one sample the standard deviation is given in the above formula. In addition $\hat{q}$ is defined as $(1 - \hat{p})$.

By definition a t-statistic is defined as

$$t = \frac{\hat{p}}{\sigma_{\hat{p}}} \quad (5.7)$$

The t-test works on whether the t statistic is less than t_0 , where t_0 is a critical value obtained from t-tables for $\nu = 105$, $\alpha = 0.05$. We seek for the condition $t \leq t_0$, for the subjects not to differ significantly, in statistical terms.

Applying the above formulas we have obtained the results for the t-tests for every pair from the sample pool. Considering the results we see that every subjects and ISTS differ from each other, with respect to their means. The following sample illustrates our t-test analysis.

Table 5.6. t-test results for the comparison of first choices of ISTS and Subject 1

Test Number	ISTS-S1	Test Number	ISTS-S1
1	0	54	0
2	0	55	1
3	0	56	0
4	1	57	0
5	0	58	0
6	0	59	0
7	1	60	1
8	1	61	0
9	0	62	0
10	0	63	1
11	1	64	1
12	0	65	0
13	1	66	0
14	0	67	0
15	1	68	1
16	0	69	1
17	0	70	0
18	1	71	1
19	0	72	0
20	0	73	0
21	1	74	0
22	0	75	1
23	0	76	0
24	0	77	0
25	0	78	0
26	0	79	0
27	0	80	0
28	0	81	0
29	0	82	0
30	0	83	0
31	1	84	0
32	0	85	0
33	0	86	0
34	0	87	0
35	1	88	1
36	0	89	0
37	0	90	1
38	1	91	0
39	1	92	0
40	0	93	1
41	0	94	1
42	0	95	0
43	1	96	0
44	1	97	0
45	0	98	0
46	0	99	0
47	0	100	0
48	0	101	0
49	0	102	0
50	1	103	1
51	1	104	1
52	0	105	0
53	1	106	0
Sum	17		14
Total Sum	31		
phat	0.29		
qhat	0.71		
Standard Dev.of phat	0.044		
t = phat/Standard Dev.of phat	6.588		
t0	1.987		
Significant (YES or NO)	YES		

Here, it may be observed that $t > t_0$, so that ISTS and Subject 1 differ significantly from each other in their top choices, regarding the means of the binomial variable.

Results for t-tests performed for all pair-wise combinations of all subjects and ISTS versus subjects are tabulated in the following table.

Table 5.7. t-test results for all combinations of the top choices of subjects and ISTS

First selections	t values	Significance	First selections	t values	Significance	First selections	t values	Significance
ISTS-S1	6.619	YES	S2-S5	4.811	YES	S5-S7	6.318	YES
ISTS-S2	5.720	YES	S2-S6	6.019	YES	S5-S8	7.229	YES
ISTS-S3	4.656	YES	S2-S7	6.469	YES	S5-S9	6.169	YES
ISTS-S4	6.922	YES	S2-S8	7.075	YES	S5-S10	4.811	YES
ISTS-S5	4.811	YES	S2-S9	5.869	YES	S5-S11	4.500	YES
ISTS-S6	5.720	YES	S2-S10	5.869	YES	S5-S12	6.469	YES
ISTS-S7	7.229	YES	S2-S11	5.420	YES	S6-S7	5.720	YES
ISTS-S8	7.855	YES	S2-S12	6.922	YES	S6-S8	7.075	YES
ISTS-S9	6.019	YES	S3-S4	6.318	YES	S6-S9	5.720	YES
ISTS-S10	5.420	YES	S3-S5	3.849	YES	S6-S10	4.811	YES
ISTS-S11	5.117	YES	S3-S6	5.420	YES	S6-S11	4.965	YES
ISTS-S12	6.019	YES	S3-S7	6.318	YES	S6-S12	6.770	YES
S1-S2	4.656	YES	S3-S8	7.539	YES	S7-S8	6.169	YES
S1-S3	5.869	YES	S3-S9	6.019	YES	S7-S9	6.169	YES
S1-S4	6.169	YES	S3-S10	4.500	YES	S7-S10	4.811	YES
S1-S5	5.720	YES	S3-S11	4.811	YES	S7-S11	5.269	YES
S1-S6	5.117	YES	S3-S12	6.169	YES	S7-S12	7.229	YES
S1-S7	4.965	YES	S4-S5	7.075	YES	S8-S9	6.770	YES
S1-S8	6.770	YES	S4-S6	6.019	YES	S8-S10	6.469	YES
S1-S9	5.269	YES	S4-S7	5.269	YES	S8-S11	6.318	YES
S1-S10	4.656	YES	S4-S8	8.015	YES	S8-S12	7.855	YES
S1-S11	4.811	YES	S4-S9	6.770	YES	S9-S10	5.420	YES
S1-S12	7.075	YES	S4-S10	5.117	YES	S9-S11	5.420	YES
S2-S3	5.420	YES	S4-S11	6.169	YES	S9-S12	7.383	YES
S2-S4	6.619	YES	S4-S12	7.229	YES	S10-S11	3.849	YES
S2-S5	4.811	YES	S5-S6	5.720	YES	S10-S12	6.169	YES
t0	1.987							

We thus see that testing the strict equality of top choices is a very hard requirement. Statistically, with respect to this test, all subjects differ significantly and all subjects differ from ISTS. So these results are inconclusive in terms of evaluating the performance of ISTS and we proceed with other tests.

5.3. Outlier Analysis

We have determined two types of outlier analysis for the responses of the subjects and ISTS, hence, we define two types of outlier. According to our first definition, if all subjects (or ISTS) have chosen one pattern, but only one subject (or ISTS) has picked up a different pattern then different selection is an outlier. Namely, common selection of all subjects cannot be found only in one's choices. In our second definition for outlier, if a subject picks up a pattern, and none of the others have selected the same pattern, then the single selection becomes an outlier. In the first outlier definition there should be a common choice between all other subjects, but in the second there is no requirement for such condition.

In regard to our type 1 outlier analysis the results turned out to be that ISTS does not differ from the subjects significantly. There is only one outlier in ISTS' selections, while the overall outlier figure is at 1.08. Therefore, the type-1 outlier analysis for the first selections of the subjects and ISTS, ISTS does not generate worse results than the subjects, actually even better than the average of the subjects (Table 5.8).

In the analysis we have again used transformed responses, as also will be used in the forthcoming analysis. Here 1 represents there is an outlier in the related test, where 0 means no outlier in the corresponding query.

Table 5.8. Turing test results of ISTS and subjects (type-1 outliers indicated)

Test Number	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Test Number	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
1	5	5	5	8	5	5	5	5	5	5	5	5	5	54	3	3	3	3	3	4	3	3	11	3	3	3	3
2	7	7	7	7	7	7	7	7	8	7	7	7	7	55	19	2	4	4	4	4	19	4	4	2	4	19	4
3	13	13	13	13	13	13	13	13	12	13	13	13	11	56	20	20	20	20	20	20	20	20	20	20	20	20	20
4	4	5	4	4	5	4	5	4	4	4	5	4	4	57	6	6	6	6	6	6	6	6	6	6	6	6	6
5	5	5	5	5	19	5	5	5	5	5	5	5	4	58	19	19	19	19	19	18	19	19	19	19	19	18	19
6	10	10	12	10	10	10	12	10	12	12	10	10	10	59	8	8	8	8	8	8	8	8	7	8	8	8	8
7	11	13	11	13	13	13	13	13	11	13	13	11	10	60	20	6	7	7	8	7	8	8	7	20	8	20	8
8	16	17	17	16	17	16	17	17	5	17	16	16	16	61	18	18	18	18	18	18	18	18	18	18	18	18	18
9	3	3	3	3	3	3	3	3	4	3	3	3	3	62	7	7	7	7	7	7	7	7	7	7	7	7	7
10	17	17	17	17	17	17	17	17	17	17	17	17	15	63	17	16	16	17	15	17	17	16	16	16	16	16	15
11	15	17	15	15	15	15	17	17	17	17	17	17	15	64	20	18	18	20	18	18	18	18	18	20	18	18	18
12	4	4	4	4	4	4	4	4	4	4	4	4	4	65	19	19	18	19	19	18	19	19	19	18	19	19	19
13	6	20	20	6	6	6	6	6	6	6	6	6	6	66	7	7	7	7	7	7	7	7	7	7	7	7	7
14	12	12	12	12	12	12	12	12	12	12	12	12	10	67	16	16	16	16	16	16	16	16	17	16	16	16	17
15	15	14	14	14	14	14	14	14	14	15	14	14	15	68	20	18	18	20	18	20	20	18	20	20	20	20	20
16	2	2	2	2	2	2	2	2	2	2	2	2	2	69	11	13	11	10	13	11	10	13	13	13	15	13	10
17	19	19	19	19	19	19	19	19	19	19	19	19	19	70	9	9	9	9	12	9	9	9	9	9	12	9	12
18	11	13	13	13	13	13	13	12	11	13	13	13	11	71	19	20	19	19	19	19	19	19	19	19	19	19	19
19	20	20	20	6	20	20	20	20	20	20	20	20	8	72	2	2	19	2	2	2	2	2	2	2	2	2	2
20	12	12	12	12	12	12	12	12	12	12	12	12	12	73	17	17	17	17	16	17	17	16	16	17	16	16	17
21	19	18	18	19	19	19	19	19	19	19	19	19	18	74	6	6	6	6	6	6	6	6	6	6	6	6	6
22	5	5	5	5	5	5	5	5	5	5	5	5	5	75	19	3	19	3	4	3	19	3	3	4	3	3	4
23	6	6	6	6	6	6	6	6	6	6	6	6	6	76	15	15	15	15	15	15	15	15	15	15	15	15	15
24	19	19	19	19	19	19	19	19	19	19	19	19	19	77	3	3	3	3	3	3	3	3	3	3	3	3	3
25	9	9	9	9	9	9	9	9	8	9	9	9	9	78	4	4	4	4	4	4	19	4	4	4	4	11	4
26	11	11	11	11	11	11	11	11	11	11	11	11	11	79	9	9	9	9	9	9	9	9	9	9	9	9	9
27	11	11	11	11	11	11	11	11	11	11	11	11	10	80	7	7	7	7	8	7	7	8	7	7	7	7	7
28	20	20	20	7	20	7	20	20	8	20	20	6	8	81	7	7	7	8	8	7	20	8	7	7	7	7	7
29	20	20	20	20	20	20	20	6	6	6	6	20	20	82	12	12	12	12	12	12	12	13	12	9	12	12	12
30	14	14	14	14	14	14	14	15	16	14	14	14	15	83	12	12	12	12	12	12	12	12	12	12	12	12	12
31	19	18	18	19	19	18	18	18	18	18	19	18	18	84	2	2	2	2	2	2	2	2	2	2	2	2	2
32	4	4	4	4	4	4	4	4	4	4	4	4	4	85	17	17	17	17	17	17	17	17	17	15	17	17	17
33	16	16	16	16	14	16	16	14	16	16	16	16	17	86	13	13	13	13	13	13	12	13	12	13	13	13	10
34	5	5	5	5	5	5	5	5	5	5	5	5	5	87	16	16	16	16	16	16	16	16	16	16	16	16	16
35	11	13	13	13	13	13	13	13	13	13	13	13	13	88	11	10	10	10	11	10	10	10	11	11	10	10	10
36	8	8	8	8	8	8	8	8	8	8	8	8	8	89	17	17	17	17	17	17	17	17	17	17	17	17	17
37	9	9	9	9	9	9	9	9	9	9	9	9	9	90	17	16	17	17	16	17	16	16	16	16	16	17	17
38	13	12	12	12	10	12	12	12	12	12	12	12	13	91	11	11	17	11	10	11	11	11	13	11	11	11	11
39	11	12	10	12	10	12	12	12	12	12	12	12	12	92	12	12	12	12	12	10	12	12	12	12	12	12	12
40	15	15	15	15	15	15	12	15	15	14	15	15	15	93	11	13	13	13	13	11	11	13	13	13	13	13	13
41	5	5	5	5	5	5	5	5	5	5	5	5	5	94	10	12	12	11	13	10	11	10	12	12	10	10	10
42	20	20	20	20	20	20	20	20	20	20	20	20	20	95	14	14	14	14	17	14	14	17	15	14	14	14	14
43	20	6	8	8	8	8	20	6	6	8	8	8	20	96	9	9	9	9	9	9	9	9	9	9	9	9	9
44	14	16	14	14	14	14	14	15	14	16	14	14	14	97	14	14	14	14	17	14	14	14	14	15	14	14	14
45	6	6	6	6	6	6	6	6	6	6	6	6	6	98	10	10	10	10	11	10	10	10	10	10	10	10	10
46	15	15	15	15	15	15	15	15	14	14	15	15	15	99	3	3	3	3	19	3	19	19	4	3	3	3	3
47	11	11	11	11	11	11	11	11	11	11	11	11	11	100	14	14	14	14	14	14	14	14	14	14	14	14	14
48	20	20	20	20	20	20	20	20	20	20	20	20	20	101	14	14	14	14	14	14	14	15	15	14	14	14	14
49	16	16	16	16	16	16	16	16	16	16	16	16	16	102	2	2	2	2	2	2	2	2	2	2	2	2	2
50	19	18	18	19	19	18	19	18	18	19	19	19	19	103	14	16	14	14	16	14	16	16	15	14	16	16	14
51	8	7	8	8	7	8	7	7	7	7	7	7	7	104	11	12	12	11	12	11	12	12	12	12	11	12	12
52	5	5	5	5	5	5	5	5	5	5	5	5	5	105	15	15	15	15	15	15	15	15	15	15	15	15	15
53	2	19	19	2	2	2	2	2	2	2	2	2	2	106	17	17	17	17	17	17	17	17	17	17	17	17	17

Table 5.9. Coded results (1: outlier) for the type-1 outlier analysis, for top choices

[illegible]

In accordance with the results obtained in the second type outlier analysis, ISTS generated only two outliers, where the average of the subjects yielded an outlier figure of 4.33. Again, we have a satisfactory result for ISTS. We conclude that regarding the second outlier analysis, ISTS' performance is not worse than the subjects' detection capability and it performs even better.

Table 5.10 displays the results of the type-2 outlier analysis for the first selections of the subjects and ISTS.

Table 5.10. Turing test results of ISTS and subjects (type-2 outliers indicated)

Test Number	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Test Number	IST	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
1	5	5	5	8	5	5	5	5	5	5	5	5	5	54	3	3	3	3	3	4	3	3	11	3	3	3	3
2	7	7	7	7	7	7	7	7	8	7	7	7	7	55	19	2	4	4	4	4	19	4	4	2	4	19	4
3	13	13	13	13	13	13	13	13	12	13	13	13	11	56	20	20	20	20	20	20	20	20	20	20	20	20	20
4	4	5	4	4	5	4	5	4	4	4	5	4	4	57	6	6	6	6	6	6	6	6	6	6	6	6	6
5	5	5	5	5	5	19	5	5	5	5	5	5	5	58	19	19	19	19	19	18	19	19	19	19	19	19	18
6	10	10	12	10	10	10	12	10	12	12	10	10	10	59	8	8	8	8	8	8	8	8	8	7	8	8	8
7	11	13	11	13	13	13	13	13	11	13	13	11	10	60	20	6	7	7	8	7	8	8	8	7	20	8	19
8	16	17	17	16	17	16	17	17	5	17	16	16	16	61	18	18	18	18	18	18	18	18	18	18	18	18	18
9	3	3	3	3	3	3	3	3	3	4	3	3	3	62	7	7	7	7	7	7	7	7	7	7	7	7	7
10	17	17	17	17	17	17	17	17	17	17	17	17	15	63	17	16	16	17	15	17	17	16	16	16	16	16	15
11	15	17	15	15	15	15	17	17	17	17	17	17	15	64	20	18	18	20	18	18	18	18	18	20	18	18	18
12	4	4	4	4	4	4	4	4	4	4	4	4	4	65	19	19	18	19	19	18	19	19	19	18	19	19	19
13	6	20	20	6	6	6	6	6	6	6	6	6	6	66	7	7	7	7	7	7	7	7	7	7	7	7	7
14	12	12	12	12	12	12	12	12	12	12	12	12	10	67	16	16	16	16	16	16	16	16	17	16	16	16	17
15	15	14	14	14	14	14	14	14	14	15	14	14	15	68	20	18	18	20	18	20	20	18	20	20	20	20	20
16	2	2	2	2	2	2	2	2	2	2	2	2	2	69	11	13	11	10	13	11	10	13	13	13	15	13	10
17	19	19	19	19	19	19	19	19	19	19	19	19	19	70	9	9	9	9	12	9	9	9	9	9	12	9	12
18	11	13	13	13	13	13	13	12	11	13	13	13	11	71	19	20	19	19	19	19	19	19	19	19	19	19	19
19	20	20	20	6	20	20	20	20	20	20	20	20	8	72	2	2	19	2	2	2	2	2	2	2	2	2	2
20	12	12	12	12	12	12	12	12	12	12	12	12	12	73	17	17	17	17	16	17	17	16	16	17	16	16	17
21	19	18	18	19	19	19	19	19	19	19	19	19	18	74	6	6	6	6	6	6	6	6	6	6	6	6	6
22	5	5	5	5	5	5	5	5	5	5	5	5	5	75	19	3	19	3	4	3	19	3	3	4	3	3	4
23	6	6	6	6	6	6	6	6	6	6	6	6	6	76	15	15	15	15	15	15	15	15	15	15	15	15	15
24	19	19	19	19	19	19	19	19	19	19	19	19	19	77	3	3	3	3	3	3	3	3	3	3	3	3	3
25	9	9	9	9	9	9	9	9	8	9	9	9	9	78	4	4	4	4	4	4	19	4	4	4	4	11	4
26	11	11	11	11	11	11	11	11	11	11	11	11	11	79	9	9	9	9	9	9	9	9	9	9	9	9	9
27	11	11	11	11	11	11	11	11	11	11	11	11	10	80	7	7	7	7	8	7	7	8	7	7	7	7	7
28	20	20	20	7	20	7	20	20	8	20	20	6	8	81	7	7	7	8	8	7	20	8	7	7	7	7	7
29	20	20	20	20	20	20	20	6	6	6	6	20	20	82	12	12	12	12	12	12	12	13	12	9	12	12	12
30	14	14	14	14	14	14	14	15	16	14	14	14	15	83	12	12	12	12	12	12	12	12	12	12	12	12	12
31	19	18	18	19	19	18	18	18	18	18	19	18	18	84	2	2	2	2	2	2	2	2	2	2	2	2	2
32	4	4	4	4	4	4	4	4	4	4	4	4	4	85	17	17	17	17	17	17	17	17	17	15	17	17	17
33	16	16	16	16	14	16	16	14	16	16	16	16	17	86	13	13	13	13	13	13	12	13	12	13	13	13	10
34	5	5	5	5	5	5	5	5	5	5	5	5	5	87	16	16	16	16	16	16	16	16	16	16	16	16	16
35	11	13	13	13	13	13	13	13	13	13	13	13	13	88	11	10	10	10	11	10	10	10	11	11	10	10	10
36	8	8	8	8	8	8	8	8	8	8	8	8	8	89	17	17	17	17	17	17	17	17	17	17	17	17	17
37	9	9	9	9	9	9	9	9	9	9	9	9	9	90	17	16	17	17	16	17	16	16	16	16	16	17	17
38	13	12	12	12	10	12	12	12	12	12	12	12	13	91	11	11	17	11	10	11	11	11	13	11	11	11	11
39	11	12	10	12	10	12	12	12	12	12	12	12	12	92	12	12	12	12	12	10	12	12	12	12	12	12	12
40	15	15	15	15	15	15	12	15	15	14	15	15	15	93	11	13	13	13	13	11	11	13	13	13	13	13	13
41	5	5	5	5	5	5	5	5	5	5	5	5	5	94	10	12	12	11	13	10	11	10	12	12	10	10	10
42	20	20	20	20	20	20	20	20	20	20	20	20	20	95	14	14	14	14	17	14	14	17	15	14	14	14	14
43	20	6	8	8	8	8	20	6	6	8	8	8	20	96	9	9	9	9	9	9	9	9	9	9	9	9	9
44	14	16	14	14	14	14	14	15	14	16	14	14	14	97	14	14	14	14	17	14	14	14	14	15	14	14	14
45	6	6	6	6	6	6	6	6	6	6	6	6	6	98	10	10	10	10	11	10	10	10	10	10	10	10	10
46	15	15	15	15	15	15	15	15	14	14	15	15	15	99	3	3	3	3	19	3	19	19	4	3	3	3	3
47	11	11	11	11	11	11	11	11	11	11	11	11	11	100	14	14	14	14	14	14	14	14	14	14	14	14	14
48	20	20	20	20	20	20	20	20	20	20	20	20	20	101	14	14	14	14	14	14	14	15	15	14	14	14	14
49	16	16	16	16	16	16	16	16	16	16	16	16	16	102	2	2	2	2	2	2	2	2	2	2	2	2	2
50	19	18	18	19	19	18	19	18	18	19	19	19	19	103	14	16	14	14	16	14	16	16	15	14	16	16	14
51	8	7	8	8	7	8	7	7	7	7	7	7	7	104	11	12	12	11	12	11	12	12	12	12	11	12	12
52	5	5	5	5	5	5	5	5	5	5	5	5	5	105	15	15	15	15	15	15	15	15	15	15	15	15	15
53	2	19	19	2	2	2	2	2	2	2	2	2	2	106	17	17	17	17	17	17	17	17	17	17	17	17	17

5.4. Modal Analysis

The third statistical analysis is the modal analysis, which subjects' and ISTS' each responses in the relevant questions are tested with respect to the *mode* of that question. The test result and evaluation is based on the total numbers of deviations from the *modes*.

In regard to the test results, with respect to the first choices of the subjects and ISTS, we found that ISTS scored 22 deviations from the modal response, while the average number of the subjects stood at 19.2. Therefore, ISTS has a slightly larger deviation than the subjects' average, insofar as the first selections are concerned. But, since most of the time-series in the Turing test can be classified into more than one groups, we next analyzed the first or second selections of ISTS and the subjects. In the related analysis, we determined that ISTS' deviation number from the mode was 11, while the average number of deviation from the mode was 11.8. Hence, we conclude that even if it generated slightly worse result than the subjects regarding the top selections, considering the first or second (which is meaningful for this type of test) choices, ISTS' results are not worse than the subjects'.

Table 5.12. Turing Test results of ISTS, subjects and the modes (top choices)

[illegible]

Table 5.13. Coded results (1: deviation from mode) for the modal analysis (top choices)

[illegible]

Table 5.14. Turing test results of ISTS and subjects and the modes
(first or second choices, test number 1-60)

First or second choices		Best-fit order																First or second choices		Best-fit order															
Time Series	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Mode	Time Series	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Mode						
1	5	5	5	8	5	5	5	5	5	5	5	5	5	5	30	14	14	14	14	14	14	14	15	16	14	14	14	14	15	14					
	16	4		4		4				4						15																			
2	7	7	7	7	7	7	7	7	8	7	7	7	7	7	31	19	18	18	19	19	18	18	18	18	18	19	18	18	18	18					
	20						8									19	18		18		19	19			19	18									
3	13	13	13	13	13	13	13	13	12	13	13	13	11	13	32	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4					
	11			12		11							12			5							5	5											
4	4	5	4	4	5	4	5	4	4	4	5	4	4	4	33	16	16	16	16	14	16	16	14	16	16	16	16	17	16						
	15	4	5			5					5	4				19												16							
5	5	5	5	5	19	5	5	5	5	5	5	5	4	5	34	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5					
	19			2		4	2						5			16																			
6	10	10	12	10	10	10	12	10	12	12	10	10	10	10	35	11	13	13	13	13	13	13	13	13	13	13	13	13	13	13					
	11	12		11									12			13					11														
7	11	13	11	13	13	13	13	13	11	13	13	11	10	13	36	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8					
	13			11		11										10																			
8	16	17	17	16	17	16	17	17	5	17	16	16	16	17	37	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9					
	17	16	16	14		17					5	17				8																			
9	3	3	3	3	3	3	3	3	4	3	3	3	3	3	38	13	12	12	12	10	12	12	12	12	12	12	12	13	12						
	19															19				12								12							
10	17	17	17	17	17	17	17	17	17	17	17	17	15	17	39	11	12	10	12	10	12	12	12	12	12	12	12	12	12	12					
	16	16	16			16						16	17			13					12	13													
11	15	17	15	15	15	15	17	17	17	17	17	17	15	17	40	15	15	15	15	15	15	12	15	15	14	15	15	15	15	15					
	19		17			14		16								20	14		17				14		15										
12	4	4	4	4	4	4	4	4	4	4	4	4	4	4	41	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5					
	20															19						2													
13	6	20	20	6	6	6	6	6	6	6	6	6	6	6	42	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20					
	15	6	6													18																			
14	12	12	12	12	12	12	12	12	12	12	12	12	10	12	43	20	6	8	8	8	8	20	6	6	8	8	8	20	8						
	11					13							12			7			7		20	6	8				6	8							
15	15	14	14	14	14	14	14	14	14	15	14	14	15	14	44	14	16	14	14	14	14	14	15	14	16	14	14	14	14	14					
	20											15				15																			
16	2	2	2	2	2	2	2	2	2	2	2	2	2	2	45	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6					
	11															20																			
17	19	19	19	19	19	19	19	19	19	19	19	19	19	19	46	15	15	15	15	15	15	15	15	14	14	15	15	15	15	15					
	18												4			20			17					15											
18	11	13	13	13	13	13	13	12	11	13	13	13	11	13	47	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11					
	19			12												10																			
19	20	20	20	6	20	20	20	20	20	20	20	20	8	20	48	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20					
	18	6	20										20			18																			
20	12	12	12	12	12	12	12	12	12	12	12	12	12	12	49	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16					
	20				20											19																			
21	19	18	18	19	19	19	19	19	19	19	19	19	18	19	50	19	18	18	19	19	18	19	18	18	19	19	19	19	19						
	18					18	18									18		19				19	18												
22	5	5	5	5	5	5	5	5	5	5	5	5	5	5	51	8	7	8	8	7	8	7	7	7	7	7	7	7	7						
	11															7			7			7					8								
23	6	6	6	6	6	6	6	6	6	6	6	6	6	6	52	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5					
	20															16																			
24	19	19	19	19	19	19	19	19	19	19	19	19	19	19	53	2	19	19	2	2	2	2	2	2	2	2	2	2	2	2					
	18							2		5						15	2																		
25	9	9	9	9	9	9	9	9	8	9		9	9	9	54	3	3	3	3	3	4	3	3	11	3	3	3	3	3	3					
	12															19					3	19		3											
26	11	11	11	11	11	11	11	11	11	11	11	11	11	11	55	19	2	4	4	4	4	19	4	4	2	4	19	4	4						
	19															18				19						2									
27	11	11	11	11	11	11	11	11	11	11	11	11	10	11	56	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20					
	19															18	18																		
28	20	20	20	7	20	7	20	20	8	20	20	6	8	20	57	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6					
	18		6	6		20		6		6						20																			
29	20	20	20	20	20	20	20	6	6	6	6	20	20	20	58	19	19	19	19	19	18	19	19	19	19	19	18	19	19						
	6		6	6	6					20		6				18			18		19														
30	14	14	14	14	14	14	14	15	16	14	14	14	15	14	59	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8					
	15															7								8	7										
31	19	18	18	19	19	18	18	18	18	18	19	18	18	18	60	20	6	7	7	8	7	8	8	7	20	8	20	8	7						
	18			18		19	19			19	18					7	7					8	20	7		7	7	6							

Table 5.15. Turing test results of ISTS and subjects and the modes
(first or second choices, test number 61-106)

First or second choices	Best-fit order														First or second choices	Best-fit order													
Time Series	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Mode	Time Series	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Mode
61	18 20	18	18	18	18	18	18	18	18	18	18	18	18	18	84	2 19	2	2	2	2	2	2	2	2	2	2	2	2	2
62	7 20	7	7	7	7	7	7	7	7	7	7	7	7	7	85	17 15	17	17	17	17	17	17	17	17	15	17	17	17	17
63	17 15	16	16	17	15	17	17	16	16	16	16	16	15	16	86	13 11	13	13	13	13	13	12	13	12	13	13	13	10	13
64	20 18	18	18	20	18	18	18	18	18	20	18	18	18	18	87	16 17	16	16	16	16	16	16	16	16	16	16	16	16	16
65	19 18	19	18	19	19	18	19	19	19	18	19	19	19	19	88	11 10	10	10	10	10	10	10	10	11	11	10	10	10	10
66	7 20	7	7	7	7	7	7	7	7	7	7	7	7	7	89	17 20	17	17	17	17	17	17	17	17	14	17	17	17	17
67	16 17	16	16	16	16	16	16	16	17	16	16	16	17	16	90	17 20	16	17	17	16	17	16	16	16	16	16	17	17	17
68	20 18	18	18	20	18	20	20	18	20	20	20	20	20	20	91	11 19	11	17	11	10	11	11	11	13	11	11	11	11	
69	11 10	13	11	10	13	11	10	13	13	13	15	13	10	13	92	12 10	12	12	12	12	10	12	12	12	12	12	12	12	
70	9 12	9	9	9	12	9	9	9	9	9	12	9	12	9	93	11 20	13	13	13	13	13	11	13	13	13	13	13	13	
71	19 15	20	19	19	19	19	19	19	19	19	19	19	19	19	94	10 11	12	12	11	13	10	11	10	12	12	10	10	10	
72	2 19	2	19	2	2	2	2	2	2	2	2	2	2	2	95	14 15	14	14	14	14	17	14	14	17	15	14	14	14	
73	17 20	17	17	17	16	17	17	16	16	17	16	16	17	17	96	9 12	9	9	9	9	9	9	9	9	9	9	9	9	
74	6 20	6	6	6	6	6	6	6	6	6	6	6	6	6	97	14 15	14	14	14	14	17	14	14	14	15	14	14	14	
75	19 3	3	19	3	4	3	19	3	3	4	3	3	4	3	98	10 11	10	10	10	10	11	10	10	10	10	10	10	10	
76	15 20	15	15	15	15	15	15	15	15	15	15	15	15	15	99	3 19	3	3	3	3	19	3	19	19	4	3	3	3	
77	3 19	3	3	3	3	3	3	3	3	3	3	3	3	3	100	14 15	14	14	14	14	14	14	14	14	14	14	14	14	
78	4 3	4	4	4	4	4	19	4	4	4	4	11	4	4	101	14 15	14	14	14	14	14	14	14	15	15	14	14	14	
79	9 12	9	9	9	9	9	9	9	9	9	9	9	9	9	102	2 19	2	2	2	2	2	2	2	2	2	2	2	2	
80	7 20	7	7	7	8	7	7	8	7	7	7	7	7	7	103	14 15	16	14	14	16	14	16	16	15	14	16	16	16	
81	7 20	7	7	8	8	7	20	8	7	7	7	7	7	7	104	11 12	12	12	11	12	11	12	12	12	12	11	12	12	
82	12 13	12	12	12	12	12	12	13	12	9	12	12	12	12	105	15 14	15	15	15	15	15	15	15	15	15	15	15	15	
83	12 11	12	12	12	12	12	12	12	12	12	12	12	12	12	106	17 20	17	17	17	17	17	17	17	17	17	17	17	17	

Table 5.16. Coded results (1: deviation from mode) for the modal analysis
(top two choices)

[illegible]

5.5. Comparison with Our Selection

Finally, we also decided to apply statistical tests to compare ISTS and subjects' results, against our own choices as designer of the tests. The same procedures as in the outlier and modal analysis are also applied here.

After the analysis, we have come up with a conclusion that ISTS' results are better than the subjects', as compared to our selections. While ISTS generated a deviation number of 14 in 106 questions, the average of the subjects yielded a figure of 21.7. With respect to our choices, ISTS is the second best after the fifth subject (See Tables 5.17 & 5.18).

Table 5.17. Turing Test Results of ISTS and subjects compared to our selections
(top choices)

First Selections														First Selections															
Test Number	Our selection	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Test Number	Our selection	ISTS	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
1	5	5	5	5	8	5	5	5	5	5	5	5	5	5	54	4	3	3	3	3	3	4	3	3	11	3	3	3	3
2	7	7	7	7	7	7	7	7	7	8	7	7	7	7	55	4	19	2	4	4	4	4	19	4	4	2	4	19	4
3	13	13	13	13	13	13	13	13	13	12	13	13	13	11	56	20	20	20	20	20	20	20	20	20	20	20	20	20	20
4	4	4	5	4	4	5	4	5	4	4	4	5	4	4	57	6	6	6	6	6	6	6	6	6	6	6	6	6	6
5	5	5	5	5	5	19	5	5	5	5	5	5	5	4	58	19	19	19	19	19	19	18	19	19	19	19	19	18	19
6	10	10	10	12	10	10	10	12	10	12	12	10	10	10	59	8	8	8	8	8	8	8	8	8	7	8	8	8	8
7	11	11	13	11	13	13	13	13	13	11	13	13	11	10	60	7	20	6	7	7	8	7	8	8	7	20	8	20	8
8	16	16	17	17	16	17	16	17	17	5	17	16	16	16	61	18	18	18	18	18	18	18	18	18	18	18	18	18	18
9	3	3	3	3	3	3	3	3	3	4	3	3	3	3	62	7	7	7	7	7	7	7	7	7	7	7	7	7	7
10	17	17	17	17	17	17	17	17	17	17	17	17	17	15	63	17	17	16	16	17	15	17	17	16	16	16	16	16	15
11	15	15	17	15	15	15	15	17	17	17	17	17	17	15	64	18	20	18	18	20	18	18	18	18	18	20	18	18	18
12	4	4	4	4	4	4	4	4	4	4	4	4	4	4	65	18	19	19	18	19	19	18	19	19	19	18	19	19	19
13	6	6	20	20	6	6	6	6	6	6	6	6	6	6	66	7	7	7	7	7	7	7	7	7	7	7	7	7	7
14	12	12	12	12	12	12	12	12	12	12	12	12	12	10	67	16	16	16	16	16	16	16	16	16	16	17	16	16	17
15	14	15	14	14	14	14	14	14	14	14	15	14	14	15	68	20	20	18	18	20	18	20	20	18	20	20	20	20	20
16	2	2	2	2	2	2	2	2	2	2	2	2	2	2	69	11	11	13	11	10	13	11	10	13	13	13	15	13	10
17	19	19	19	19	19	19	19	19	19	19	19	19	19	19	70	9	9	9	9	9	12	9	9	9	9	9	12	9	12
18	11	11	13	13	13	13	13	13	12	11	13	13	13	11	71	19	19	20	19	19	19	19	19	19	19	19	19	19	19
19	20	20	20	20	6	20	20	20	20	20	20	20	20	8	72	2	2	2	19	2	2	2	2	2	2	2	2	2	2
20	12	12	12	12	12	12	12	12	12	12	12	12	12	12	73	17	17	17	17	17	16	17	17	16	16	17	16	16	17
21	19	19	18	18	19	19	19	19	19	19	19	19	19	18	74	6	6	6	6	6	6	6	6	6	6	6	6	6	6
22	5	5	5	5	5	5	5	5	5	5	5	5	5	5	75	3	19	3	19	3	4	3	19	3	3	4	3	3	4
23	6	6	6	6	6	6	6	6	6	6	6	6	6	6	76	15	15	15	15	15	15	15	15	15	15	15	15	15	15
24	19	19	19	19	19	19	19	19	19	19	19	19	19	19	77	3	3	3	3	3	3	3	3	3	3	3	3	3	3
25	9	9	9	9	9	9	9	9	9	8	9	9	9	9	78	4	4	4	4	4	4	4	19	4	4	4	4	11	4
26	11	11	11	11	11	11	11	11	11	11	11	11	11	11	79	9	9	9	9	9	9	9	9	9	9	9	9	9	9
27	11	11	11	11	11	11	11	11	11	11	11	11	11	10	80	7	7	7	7	7	8	7	7	8	7	7	7	7	7
28	20	20	20	20	7	20	7	20	20	8	20	20	6	8	81	7	7	7	7	8	8	7	20	8	7	7	7	7	7
29	20	20	20	20	20	20	20	20	6	6	6	6	20	20	82	12	12	12	12	12	12	12	12	13	12	9	12	12	12
30	14	14	14	14	14	14	14	14	15	16	14	14	14	15	83	12	12	12	12	12	12	12	12	12	12	12	12	12	12
31	18	19	18	18	19	19	18	18	18	18	18	19	18	18	84	2	2	2	2	2	2	2	2	2	2	2	2	2	2
32	4	4	4	4	4	4	4	4	4	4	4	4	4	4	85	17	17	17	17	17	17	17	17	17	15	17	17	17	17
33	16	16	16	16	16	14	16	16	14	16	16	16	16	17	86	13	13	13	13	13	13	13	12	13	12	13	13	13	10
34	5	5	5	5	5	5	5	5	5	5	5	5	5	5	87	16	16	16	16	16	16	16	16	16	16	16	16	16	16
35	13	11	13	13	13	13	13	13	13	13	13	13	13	13	88	10	11	10	10	10	11	10	10	10	11	11	10	10	10
36	8	8	8	8	8	8	8	8	8	8	8	8	8	8	89	17	17	17	17	17	17	17	16	16	16	16	16	17	17
37	9	9	9	9	9	9	9	9	9	9	9	9	9	9	90	17	17	16	17	17	16	17	16	16	16	16	16	16	17
38	12	13	12	12	10	12	12	12	12	12	12	12	12	13	91	11	11	11	17	11	10	11	11	11	13	11	11	11	11
39	12	11	12	10	12	10	12	12	12	12	12	12	12	12	92	12	12	12	12	12	10	12	12	12	12	12	12	12	12
40	15	15	15	15	15	15	15	12	15	15	14	15	15	15	93	11	11	13	13	13	13	11	11	13	13	13	13	13	13
41	5	5	5	5	5	5	5	5	5	5	5	5	5	5	94	10	10	12	12	11	13	10	11	10	12	12	10	10	10
42	20	20	20	20	20	20	20	20	20	20	20	20	20	20	95	14	14	14	14	14	17	14	14	17	15	14	14	14	14
43	8	20	6	8	8	8	8	20	6	6	8	8	8	20	96	9	9	9	9	9	9	9	9	9	9	9	9	9	9
44	14	14	16	14	14	14	14	14	15	14	16	14	14	14	97	14	14	14	14	14	17	14	14	14	14	15	14	14	14
45	6	6	6	6	6	6	6	6	6	6	6	6	6	6	98	10	10	10	10	10	11	10	10	10	10	10	10	10	10
46	15	15	15	15	15	15	15	15	15	14	14	15	15	15	99	3	3	3	3	3	19	3	19	19	4	3	3	3	3
47	11	11	11	11	11	11	11	11	11	11	11	11	11	11	100	14	14	14	14	14	14	14	14	14	14	14	14	14	14
48	20	20	20	20	20	20	20	20	20	20	20	20	20	20	101	14	14	14	14	14	14	14	14	14	15	15	14	14	14
49	16	16	16	16	16	16	16	16	16	16	16	16	16	16	102	2	2	2	2	2	2	2	2	2	2	2	2	2	2
50	19	19	18	18	19	19	18	19	18	18	19	19	19	19	103	16	14	16	14	14	16	14	16	16	15	14	16	16	14
51	8	8	7	8	8	7	8	7	7	7	7	7	7	7	104	11	11	12	12	11	12	11	12	12	12	12	11	12	12
52	5	5	5	5	5	5	5	5	5	5	5	5	5	5	105	15	15	15	15	15	15	15	15	15	15	15	15	15	15
53	2	2	19	19	2	2	2	2	2	2	2	2	2	2	106	17	17	17	17	17	17	17	17	17	17	17	17	17	17

Table 5.18. Coded results (1: deviation from ours) comparison with our classification (top choices)

[illegible]

Summing up the results of the three different tests for the performance of ISTS with the real data, in comparison to the experts both from the professional and the academic life, we have come up with an overall conclusion that ISTS detects the generic dynamic patterns as well as the expert human eye.

6. CONCLUSIONS

In the first phase of this research, Indirect Structure Testing Software (ISTS), developed by Kanar (1999), was tested and improved against noisy synthetic and real data sets. For the detection of classification problems and algorithmic flaws, several synthetic smooth and noisy time-series were used.

It was observed that ISTS had a significant inadequacy in detection of oscillatory patterns, due to the lack normalization procedure, in the pre-process of the oscillation detection algorithm. Moreover, Kanar's (1999) two complementary first peak detection algorithms were merged and a single and more effective algorithm has been constructed.

Another problem area is the weakness of ISTS in detecting linear patterns. Since linear pattern is one of the most recognizable generic patterns in the real world with some noise additions, more precise classification approach is needed instead of simple binary zero-one comparison method of ISTS. Like the nonlinear generic dynamic patterns, we included the linear pattern class into HMM-based algorithm. After the incorporation of the linear pattern to HMM, a large-scale trained database was constructed, for all patterns. Hence, the normalized likelihood values for the linear pattern become a metric compatible with the other patterns.

In the second phase of the study, we have focused on applying systematic tests to measure the reliability of ISTS. Firstly, we have conducted a preliminary test, with 76 synthetic noisy data. ISTS has generated successful classification results. Then, we have designed and applied Turing test, which is composed of 106 real time series. We have compared the responses of 12 human experts and the results of ISTS.

To summarize the statistical tests; first of all it was observed that with respect to their first choices, both within subject differences and between subjects and ISTS differences were significant. This test is applied with respect to the means of transformed binomial variables. Since the significant differences exist between all subjects as well, this variation may be natural, due to the nature of the test. Turing test for such a problem is

hard to apply, since the number of patterns in the survey is rather high and there is substantial noise as well. Noisy nature of the data may cause confusion among the similar patterns; even human subjects may have sharply differing visual properties. Hence, we continued to focus on relative analysis of the above raw differences.

The results of the relative statistical analysis, turn out quite satisfactory, as mentioned before. Focusing on two types of outlier analysis, we obtained the results that outlier numbers of ISTS in the test are less than the average of the subjects. In the modal analysis considering the top (first) selections of ISTS and the subjects, ISTS performed slightly worse than the average of the subjects. Due to the noisy nature of the data, we investigate whether the trouble continues for this first or second selection case. Here, we consider whether this problem continues for the first or second selection case. As we expected, the results for this second step analysis were satisfactory and ISTS performed as well as the subjects. In the last analysis, the new benchmark for the responses of ISTS and the subjects are our own classifications or hypotheses. Based on the results from this last test, we conclude that ISTS performs again as well as the subjects.

As a future research idea, ISTS may be integrated into system dynamics simulation software and indirect structure validation tests may be directly in the modeling tools.

As another future study, the same real data set we use in Turing test can be classified with other existing classification algorithms in literature and the results can be compared with the performance of ISTS.

APPENDIX A: REVISED MATLAB CODES for ISTS

```

function [shapedec,mxi]=testthy(y,i,eql)

% TESTHY   Test hypothesis
%
% [SHAPE_DECISION,MXI]=TESTHY(Y,CID,EQL) tests the hypothesis that
% dynamic pattern Y is an instance of class id represented by CID.
% If stated EQL=0 means converges to zero,
% EQL=1 means convergence to non-zero.
%
% SHAPE_DECISION is either 0(FAILE D), 1(PASSED), 2(UNSURE).
% MXI is the id of the class for to which the likelihood is greatest.
% Final decision after convergence check is printed on the screen
% after check for convergence.
%
% Calls CONST, LINEAR, CLASST, ID, CHCONV, MESSAGE
%
% 05.03.2006 - IsEmpty kullanildi.

% -----
MODEL_DOSYASI = 'trained\model_12_7_AUGREV1';

y=changeSize(y,120);

% Initializations -----
normvec=[];                                % Normalization vector
table = -10*ones(1,id(0));                 % Initialize table
sub=0;

% Test for CONST and LINEAR -----
c=const(y);                                % Check for constant pattern
if c==1                                     % If a zero constant pattern is detected
    table(id('zero0'))=10;
end
if c==2 | c==3                             % If a nonzero constant pattern is
found                                     found
    table(id('const'))=10;
end

% if c==0      | c==3                      % If not a constant pattern
%      l=linear(y);                         % Check for linear pattern
%      if l==1;
%          table(id('plnr'))=10;            % Positive linear found
%      elseif l==2;
%          table(id('nlinr'))=10;          % Negative linear found
%      end

```

```

% end

% HMM ile test edilenler -----
if c~=1 & c~=2
    table = classt(y,MODEL_DOSYASI,table);
end

mx=-20;
mxi=0;
for j=1:id(0)
    tj=table(j);
    fprintf(2,'%8s %8.2f\n',id(j),tj);
    if tj>mx & tj>-3.0
        mx=tj;
        mxi=j;
    end
end

% *** Class Relations ***

%
clrel=eye(id(0));

clrel([8 9 10 11],[8 9 10 11])=ones(4,4);
clrel([17 18 19 20],[17 18 19 20])=ones(4,4);
clrel([12 13],[12 13])=ones(2,2);
clrel([21 22],[21 22])=ones(2,2);

clrel(6,7)=2;

relrow=clrel(i,:);
hyplkd=max(table(relrow==1));
likelihoods

% Max of sibling class

if hyplkd >= -3.0
    otherlkd=table(find(relrow==1));
    if isempty(find(otherlkd>hyplkd))
        shapedec=2;
    else
        shapedec=1;
    end
else
    fslkd=table(find(relrow==2));
    if isempty(find(fslkd>=-3.0))
        shapedec=0;
    else
        shapedec=0;
    end
end

SIMULATION REQD
sub=4;

```

```

        end
    end
    dec=shapedec;

% *** CHECK FOR
CONVERGENCE ***

% id    01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
convtype=[ 0  0  1 -1  1  1  0 -1 -1 -1 -1  0  0 -1 -1  0 +1 +1 +1 +1 +1 +1  0  0  0];

if shapedec==1 & convtype(i)~=0                                % If shape PASSES and
    converging class                                           % find slope & mean at the end
        [endsl,endmean]=chconv(y) ;

        if eql==0 & convtype(i)==-1                            % Form decision matrices
            D=[0 0 3; 0 1 0; 0 0 0];                          % depending on eql hypothesis
            DN=[0 1 0; 2 0 2; 0 1 0];                          % and class type
        elseif eql==0 & convtype(i)==+1
            D=[0 0 0; 0 1 0; 3 0 0];
            DN=[3 1 0; 2 0 2; 0 1 0];
        elseif eql==1 & convtype(i)==-1
            D=[0 0 0; 1 0 1; 0 0 0];
            DN=[1 0 1; 0 4 0; 0 0 0];
        elseif eql==1 & convtype(i)==+1
            D=[0 0 0; 1 0 1; 0 0 0];
            DN=[0 0 0; 0 4 0; 1 0 1];
        end
        dec=D(endsl+2,endmean+2);                                % Decision
        sub=DN(endsl+2,endmean+2);                                % Explanation
    end

    message(dec,sub)                                             % Report results

function mes=message(decision,expl)

% MESSAGE(DECISION,EXPL) returns a message string depending
% on the parameters of DECISION and EXPL.

if decision==1
    mes='INCONCLUSIVE';
elseif decision ==2
    mes='PASSED';
elseif decision ==3
    mes='SIM. FURTHER';
elseif decision ==0
    mes='FAILED';
end

```

```

if expl==1
    mes=setstr([real(mes) real(' due to nonzero slope')]);
elseif expl==2
    mes=setstr([real(mes) real(' due to nonzero equilibrium')]);
elseif expl==3
    mes=setstr([real(mes) real(' due to zero equilibrium')]);
elseif expl==4
    mes=setstr([real(mes) real(' one of its precedents found')]);
end

```

```

function [y,a]=pfit1(x,n,N)

```

```

% [y,a]=pfit1(x,n,N)
%     fits a polynomial of order n to
%     the data given as vector x.
%     n=1 for linear, n=2 for quadratic
%     The independent variable ranges from 0 to (length of x)/N
%     Returns polynomial coefficient vector a, and,
%     points of fitted vector y

```

```

[cx rx]=size(x);

```

```

t=linspace(0,(rx-1)/(N-1),rx);
B(1,:)=ones(size(x));
B(2,:)=t;
for i=3:n+1
    B(i,:)=B(2,:).^ (i-1);
end
a=inv(B*B')*B*x';

```

```

y=a'*B;

```

```

function y=nrmlz(f,mx)

```

```

% NRMLZ(f,mx)
%     normalizes values of f by dividing each point
%     by the maximum of absolute values of all points
%     irrespective of the value of mx.
% NRMLZ(f)
%     normalizes values of f such that min->0, max->1

```

```

if nargin==1
    mx=1;
    mn=0;

```

```

    y=(f-min(f))/(max(f)-min(f)).*(mx-mn)+mn;
else
    y=f/max(abs(f));
end

```

```

function y=id(arg)
% ID(ARG)
% For string input returns the id number of class named ARG
% For integer input returns the name of the class for that id.
% ID(0) returns the number of classes defined

```

```

IDS=['zero0';'const';'plinr';'nlinr';'nexgr';'sshgr';'pexgr'
    'gr1da';'gr1db';'gr2da';'gr2db';'d1peg';'d2peg'
    'nexdc';'sshdc';'pexdc';'d1gra';'d1grb';'d2gra';'d2grb'
    'g1ped';'g2ped';'oscct';'oscgr';'oscdc'];

```

```

[ri ci]=size(IDS);
if isstr(arg)==1
    if size(abs(arg))==[1 5]
        for i=1:ri
            if IDS(i,:)==arg
                y=i;
            end
        end
    end
else
    if arg==0
        y=ri;
    else
        y=IDS(arg,:);
    end
end
end

```

```

function table=classt(y,fname,table)
% CLASST (Y,FNAME,TABLE)
% Forms likelihood table for vector Y according to the
% HMM definitions stored in file FNAME.DAT in the current directory
% Input is the partly filled likelihood table TABLE.
%
% Calls NRMLZ, VITERBI, PFIT1

```

```

file=setstr([real(fname) real('.dat')]);
fid=fopen(file,'r');

```

% Open Model file

% Read training parameters

```

type=fread(fid,1,'int');
key=fread(fid,1,'int');

```

```

t=fread(fid,1,'int');
n=fread(fid,1,'int');
nclass=fread(fid,1,'int');
source=fread(fid,[1 nclass],'int');

```

```

% *** Normalizations ***

```

```

ly=length(y);

```

```

YN(1,:)=nrmlz(y);
YN(2,:)=1.0-nrmlz(y);
[YOSC,oscslp]=osprep(y);
YN([3 4],:)=YOSC;

```

```

% 1 : min-max

```

```

% 2 : max-min (mirror)

```

```

% 3-4 : autocorrelation function based

```

```

% Normalization vector

```

```

% id: 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25...

```

```

nv = [ 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 4 4];

```

```

mr = [ 0 0 4 0 14 15 16 17 18 19 20 21 22 0 0 0 0 0 0 0 0 0 0 0];

```

```

% *** MAIN LOOP ***

```

```

i=fread(fid,1,'int');

```

```

while feof(fid)==0

```

```

    p=fread(fid,[n 1],'double');

```

```

% Read class parameters

```

```

    if type==1

```

```

        A=fread(fid,[n n*(t-1)],'double');

```

```

    else

```

```

        A=fread(fid,[n n],'double');

```

```

    end

```

```

    m=fread(fid,[3 n],'double');

```

```

    V=fread(fid,[3 3*n],'double');

```

```

    samsz=fread(fid,1,'int');

```

```

    lkd=fread(fid,[1 samsz],'double');

```

```

    st=fread(fid,[samsz t],'float');

```

```

    if type==1

```

```

        [lkdin,optseq]=viterbi(p,A,m,V,feat(YN(nv(i,:),t))); % evaluate likelihoods

```

```

    else

```

```

        [lkdin,optseq]=viterbi0(p,A,m,V,feat(YN(nv(i,:),t)));

```

```

    end

```

```

    table(i)=(lkdin-mean(lkd))/std(lkd);

```

```

% ***** MIRRORS *****

```

```

    mir=mr(i);

```

```

    if mir~=0

```

```

        if type==1

```

```

            [lkdin,optseq]=viterbi(p,A,m,V,feat(YN(2,:),t));

```

```

        else

```

```

            [lkdin,optseq]=viterbi0(p,A,m,V,feat(YN(2,:),t));

```

```

        end

```

```

        table(mir)=(lkdin-mean(lkd))/std(lkd);
    end

% *** OSCILLATORY PATTERNS
***
% if i==23
%     if type==1
%         [lkdin,optseq]=viterbi(p,A,m,V,feat(YN(4,:),t));
%     else
%         [lkdin,optseq]=viterbi0(p,A,m,V,feat(YN(4,:),t));
%     end
%     if oscslp>0.2
%         table(24)=(lkdin-mean(lkd))/std(lkd);
%     elseif oscslp<0.2
%         table(25)=(lkdin-mean(lkd))/std(lkd);
%     end
% end
    if i==23
        if type==1
            [lkdin,optseq]=viterbi(p,A,m,V,feat(YN(4,:),t));
        else
            [lkdin,optseq]=viterbi0(p,A,m,V,feat(YN(4,:),t));
        end
        if oscslp>0.1
            table(24)=(lkdin-mean(lkd)*1.5)/std(lkd);
        elseif oscslp<0.1
            table(25)=(lkdin-mean(lkd)*1.5)/std(lkd);
        end
    end
    i=fread(fid,1,'int');
% Proceed with next class

end
fclose(fid);

function [sl,mn]=chconv(y)

% [S,M]=CHCONV(Y) Check Convergence
% Returns S and M depending on the values of mean and slope of the
% last 10 per cent of all the data points in array Y.

ly=length(y);
yn=nrmlz(y,1);
% Normalize with max. absolute value

nendp=fix(ly/10);
% Take last 1/10 data points
endt=(ly-nendp+1:ly);
endy=yn(endt);

f=feat(endy,1);
% Evaluate slope / curvature / mean

```

```

sl=0;
if f(1)>0.02
    sl=+1;
elseif f(1)<-0.02
    sl=-1;
end

```

```

mn=0;
if f(3)>0.02
    mn=+1;
elseif f(3)<-0.02
    mn=-1;
end

```

```

function Y=feat(X,t)
% FEAT(X,t) extracts features from t segments obtained from
% the sample vectors given as the rows of vector X.
% The features are based on 1st and 2nd order polynomial curve
% fitting on each segment.
% Returns matrix with columns as the observation vectors.
% The resulting matrix is 3 by (t * no of rows).
% feature 1 : slope of linear fit
%      2 : curvature at the beginning of the segment
%      3 : curvature at the middle of the segment
%
% Calls PFIT1

```

```

[rX cX]=size(X);
Y=[];
for i=1:rX
    f=X(i,:);
    n=length(f);
    sbeg=1;
    for segleft=t:-1:1
        lseg=round(n/segleft);
        send=sbeg+lseg-1;
        rmid=0.5*(lseg-1)/(cX-1);
        feature
        % brx(segleft)=send;bry(segleft)=f(send);

        r=f(sbeg:send);
        [w1,p1]=pfit1(r,1,cX);
    end
end

```

```

%    g1(sbeg:send)=w1;                                % can be used to check 1st order points

    v(1)=p1(2);                                         % 1st feature : slope
    [w2,p2]=pfit1(r,2,cX);                             % 2nd order approximation

%    g2(sbeg:send)=w2;                                % can be used to check 2nd order
points

    a1=p2(2);
    a2=p2(3);

    par=a1^2+4*a1*a2*rmid+4*a2^2*rmid^2;
    v(2)=2*a2/(1+par)^1.5;                             % 2nd feature : curvature at mid-point

    v(3)=mean(r);                                       % 3rd feature : segment mean

    Y=[Y v'];
    n=n-lseg;
    sbeg=sbeg+lseg;
end

%    plot(f);hold on;plot(g1,'r');plot(g2,'b')         % use to plot approximated functions
%    plot(brx,bry,'o')
%    hold off

end

```

```

function [lkd,optseq]=viterbi(p,A,m,V,S)
% [lkd,optseq]=viterbi(p,A,m,V,S) calculates the state-optimized
% likelihood function and the optimal state sequence for a
% nonstationary HMM model.
% INPUTS :
% p : the initial probability vector (column)
% A : [A1 A2 ... A(t-1)] (n*t n),
%     Ai as the state transition matrix from segment i to i+1
% m : [m1 m2 .. mn], mi is the mean vector of state i
% V : [V1 V2 .. Vn], Vi as the covariance matrix of the vectors for state i
% S : matrix of input sequence of observation vectors (column)
% OUTPUTS :
% lkd : state optimized likelihood function
% optseq : optimal state sequence, [q1 q2 ... qt]
% where,
% n=number of states,
% t=number of segments.
%
% Calls LOGL.
[N nt1]=size(A);

```

```

[rS T]=size(S);

%***** DENSITY FUNCTION VALUES *****
for i=1:N                                % for all states
    C=V(:,((i-1)*rS+1:i*rS));
    cons=1/((2*pi)^rS*(det(C)^0.5));
    for j=1:T                            % and for each obs.vector
        omm=S(:,j)-m(:,i);
        b(i,j)=cons*exp(-.5*omm'*inv(C)*omm);    % calculate density
    end                                  % function values
end

%***** LOGARITHMIC CONVERSION *****
p=logl(p);
A=logl(A);
b=logl(b);
%***** VITERBI ALGORITHM : INITIALIZATION *****
d(1,:)=p'+b(:,1)';
psi=zeros(1,N);
%***** RECURSIVE COMPUTATION *****
for t=2:T
    for j=1:N
        dd=(d(t-1,:)+A(:,j+(t-2)*N)'+b(j,t));
        d(t,j)=max(dd);
        argmax=find(dd==max(dd));
        psi(t,j)=argmax(1,1);
    end
end
%***** TERMINATION *****
lkd=max(d(T,:));
qt=find(d(T,:)==lkd);
optseq(T)=qt(1,1);
for t=T-1:-1:1
    optseq(t)=psi(t+1,optseq(t+1));
end

```

```

function [lkd,optseq]=viterbi0(p,A,m,V,S)
% viterbi0(p,A,m,V,S) calculates the state-optimized likelihood
% function and the optimal state sequence for an HMM where,
% p : the initial probability vector (column)
% A : state transition matrix (nxn)
% m : [ m1 m2 .. mn], mi are the mean vectors of n states
% V : [V1 V2 .. Vn] with Vi as the covariance matrices
% S : matrix of input sequence observation vectors (column)
% Returns [ p*, q1, q2, ..., qt ]

```

```

[N N]=size(A);

```

```

[rS T]=size(S);

%***** DENSITY FUNCTION VALUES *****
for i=1:N                                % for all states
    C=V(:,((i-1)*rS+1:i*rS));
    cons=1/((2*pi)^rS*(det(C)^0.5));
    for j=1:T                            % and for each obs.vector
        omm=S(:,j)-m(:,i);
        b(i,j)=cons*exp(-.5*omm'*inv(C)*omm);    % calculate density
    end                                    % function values
end

%***** LOGARITHMIC CONVERSION *****
p=logl(p);
A=logl(A);
b=logl(b);
%***** VITERBI ALGORITHM : INITIALIZATION *****
d(1,:)=p'+b(:,1)';
psi=zeros(1,N);
%***** RECURSIVE COMPUTATION *****
for t=2:T
    for j=1:N
        dd=(d(t-1,:)+A(:,j)'+b(j,t));
        d(t,j)=max(dd);
        argmax=find(dd==max(dd));
        psi(t,j)=argmax(1,1);
    end
end
%***** TERMINATION *****
lkd=max(d(T,:));
qt=find(d(T,:)==lkd);
optseq(T)=qt(1,1);
for t=T-1:-1:1
    optseq(t)=psi(t+1,optseq(t+1));
end

```

```
function [Y,coef2]=osprep(y)
```

```

% [Y,SLOPE]=osprep(y) preprocesses for oscillation
% calculates autocorrelation function of input y and returns the
% values until the first peak as the first row of Y.
% The second row of Y is the autocorrelation function of the detrended data.
% SLOPE is the slope of the trend.

```

```

yn=nrmlz(y);
ly=length(y);

```

```
MIN_SLOPE = 0.1;
```

```

MIN_PERIOD = ly/2;

% Direct evaluation of the autocorrelation function

acy=autocf(nrmlz(y));
% figure
% plot(acy)
acfpeak=fpeak(acy);

if acfpeak < MIN_PERIOD
    Y1=changeSize(acy(1:acfpeak),ly);
    Y1 = nrmlz(Y1);

else
    Y1=zeros(size(y));
end

% autocorrelation function after removal of trend

[ylf,coefs]=pfit1(yn,1,ly);
coef2=coefs(2);

if abs(coef2) > MIN_SLOPE
    acy=autocf(nrmlz(yn-ylf));
    acfpeak=fpeak(acy);

    if acfpeak < MIN_PERIOD
        Y2=changeSize(acy(1:acfpeak),ly);
        Y2 = nrmlz(Y2);
    else
        Y2=zeros(size(y));
    end
else
    Y2=zeros(1,ly);
end

Y=[Y1;Y2];

function p=fpeak(y)
% FPEAK          First peak
%
% fpeak(y) finds the first peak in the signal y

```

```
%
d = sign(diff(y));

pos = 0;
p=length(d);
i = 1;
peak_found = 0;

while peak_found == 0 & i <= length(d)

if pos & d(i) <= 0
    p = i;
    peak_found = 1;
end
if d(i) > 0
    pos = 1;
end
i = i + 1;
end
```

```
function y1 = changeSize(y, N)

ly = length(y);

x = 0 : ly-1;

x1 = linspace(0, ly-1, N);

y1 = interp1(x,y,x1);
```

```
function ys = smoothM (y,N)
m = length(y);
for i=1:m
    if i==1 | i==m
        ys(i) = y(i);
    elseif (i - N) < 0 & (i + N) > m
        sum = 0;
        for j=1:m
            sum = sum + y(j);
        end
        ys(i) = sum / m;
    elseif (i - N) < 0 & (i + N) < m
```

```

sum1 = 0;
for j=1:i
    sum1 = sum1 + y(j);
end
sum2 = 0;
for k=(i + 1):(i + N)
    sum2 = sum2 + y(k);
end
ys(i) = (sum1 / (i) + sum2 / N) / 2;
elseif (i - N) > 0 & (i + N) > m
    sum1 = 0;
    for j=(i + 1):m
        sum1 = sum1 + y(j);
    end
    sum2 = 0;
    for k=(i-N+1):i
        sum2 = sum2 + y(k);
    end
    ys(i) = (sum1 / (m - i) + sum2 / N) / 2;
else
    sum = 0;
    for j=(i - N + 1):(i + N)
        sum = sum + y(j);
    end
    ys(i) = sum / (2 * N);
end
end
end

```

```

function train(t,n,type,source,file,key,fileref)
% TRAIN(T,N,TYPE,SOURCE,FILE,KEY,REFERENCE)
%   Trains Hidden Markov Models using N states dividing each
%   sample signal into T segments.
%   TYPE is 0 for stationary model and 1 for nonstationary model.
%   Any other special characteristic originating from feature extraction to
%   training algorithm is represented by the integer KEY.
%   The sample sets are read from the directory C:\MATLAB\THESIS\SAMPLES
%   with the version numbers represented by the vector SOURCE.
%   Training results for all classes are written to FILE.dat under
%   C:\MATLAB\THESIS\HMM\TRAINED
%   A summary file FILE.sum is created in the same directory.
%   To reduce training time a present trained data file REFERENCE.dat
%   can be specified if trained with the same parameters and the same KEY.
%
%   Calls FEAT, HMMTR, MONITR, CTSTR, ID

```

```

%***** Open Output Files
*****

dirout=real('trained\'); % Output directory
opf=fopen(setstr([dirout real(file) real('.dat')]),'w'); % Open training output file
inf=fopen(setstr([dirout real(file) real('.sum')]),'w'); % Open training info file

%***** Write Model and Training Parameters
*****

fprintf(inf,'%s\n\n', '***** HMM TRAINING REPORT *****');
fwrite(opf,type,'int');
model(1,:)=real(' STATIONARY');model(2,:)=real('NONSTATIONARY');
fprintf(inf,'%22s %s\n',MODEL TYPE : ',setstr(model(type+1,:)));
fwrite(opf,key,'int'); % Write KEY
fprintf(inf,'%s\n',sprintf('KEY FOR TRAINING : %2i',key));
fwrite(opf,t,'int'); % Write number of segments
fprintf(inf,'%s\n',sprintf('NUMBER OF SEGMENTS : %2i',t));
fwrite(opf,n,'int'); % Write number of states
fprintf(inf,'%s\n',sprintf('NUMBER OF STATES : %2i',n));
nclass=length(source);
fwrite(opf,nclass,'int'); % Write length of source
fwrite(opf,source,'int'); % Write source array

%***** Determine Available Results
*****

prevtr=zeros(size(source)); % Initialize 'previously trained' array
if nargin==7 % If a reference file is given,
    fprintf(inf,'%s\n\n',sprintf('REFERENCE FILE : %12s',fileref));

    k=monitr(fileref,'key');
    nn=monitr(fileref,'n');
    tt=monitr(fileref,'t');
    refsource=monitr(fileref,'source');

    if ((k==key)&(t==tt)&(n==nn))
        ms=min(size(refsource),size(source));
        sc=source(1:ms(2));rsc=refsource(1:ms(2));
        iptr=find((sc==rsc)&(sc~=0));
        prevtr(iptr)=ones(size(iptr)); % Mark classes already trained
    end
end

    fprintf(inf,'%s\n\n','REFERENCE FILE : NONE');
end

% ***** MAIN LOOP
*****

fprintf(inf,'%4s %6s %10s %12s\n','ID ','CLASS','VERSION','STATUS');
fprintf(inf,'%4s %6s %10s %12s\n','----','-----','-----','-----');

```

```

for i=1:nclass                                % For each class in source array,
    flag=0;
    if(prevtr(i)==1)                          % If available in reference file
        % get it
        p=monitr(fileref,'p',i);A=monitr(fileref,'A',i);
        m=monitr(fileref,'m',i);V=monitr(fileref,'V',i);
        lkd=monitr(fileref,'lkd',i);st=monitr(fileref,'st',i);
        flag=1;
        s=size(st);
        fprintf(inf,'%3i %7s %10i %12s\n',i,id(i),source(i),'inherited'); %NEW
    else                                       % otherwise,
        file=setstr(['samples\' real(id(i)) real('-') ctstr(source(i)) real('.txt')]);
        fr=fopen(file,'r');                  % open sample file with version no.c
        if fr~= -1;                          % If file is opened successfully,
            % s=fread(fr,[1 2],'int');        % read size of matrix
            % X=fread(fr,[s(1) s(2)],'float'); % read matrix of samples
            X = dlmread(file,'\t');
            s = size(X);
            fclose(fr);
            fprintf('%s\n',file)
            [p,A,m,V,st,lkd]=hmmtr(feats(X,t),t,n,type); % Train with data
            flag=1;
            fprintf(inf,'%3i %7s %10i %12s\n',i,id(i),source(i),'trained'); %NEW
        end
    end
    if flag==1;                              % If results obtained either way
        fwrite(opf,i,'int');                  % Write id no for class
        fwrite(opf,p,'double');               % Write p,A,m,V,m,lkd,st
        fwrite(opf,A,'double');
        fwrite(opf,m,'double');
        fwrite(opf,V,'double');
        fwrite(opf,s(1),'int');
        fwrite(opf,lkd,'double');
        fwrite(opf,st,'float');
    end
end
fclose(opf);
fclose(inf);

```

```

function y=monitr(file,var,id)
% MONITR(FILE,VAR,ID)
%     Monitors the training output file FILE.dat existing in
%     the directory TRAINED.
%     VAR is the requested variable name which is one of the following:
%     't','n','key','nclass','source' (ID can be omitted)
%     'p','A','m','V','lkd','st' (ID is necessary)

```

```

% **** OPEN FILE IN 'TRAINED' DIRECTORY ****
dir = 'trained\';

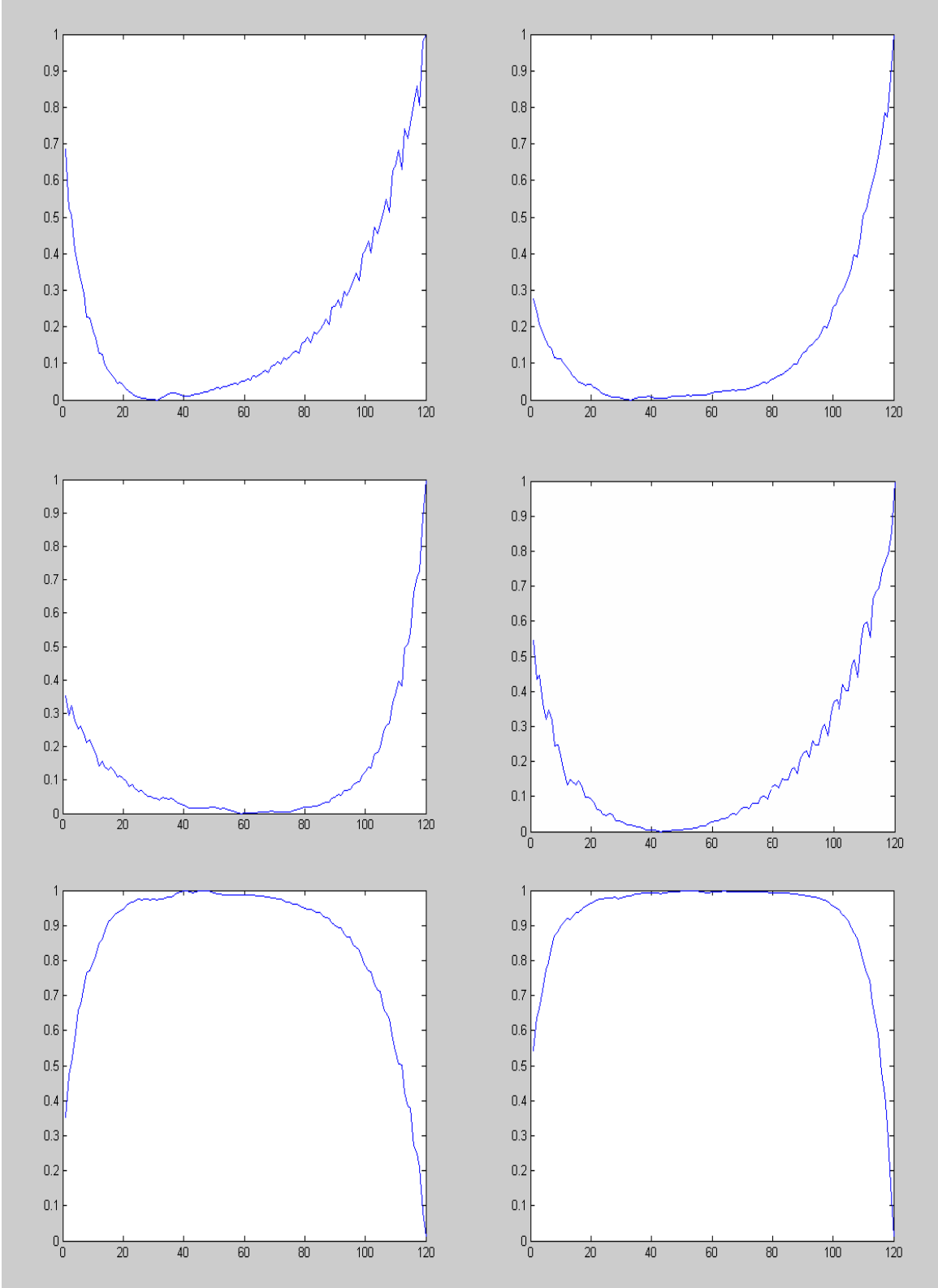
trfile=setstr([dir real(file) real('.dat')]);
fid=fopen(trfile,'r');
% *****
type=fread(fid,1,'int');
key=fread(fid,1,'int');
t=fread(fid,1,'int');
n=fread(fid,1,'int');
nclass=fread(fid,1,'int');
source=fread(fid,[1 nclass],'int');

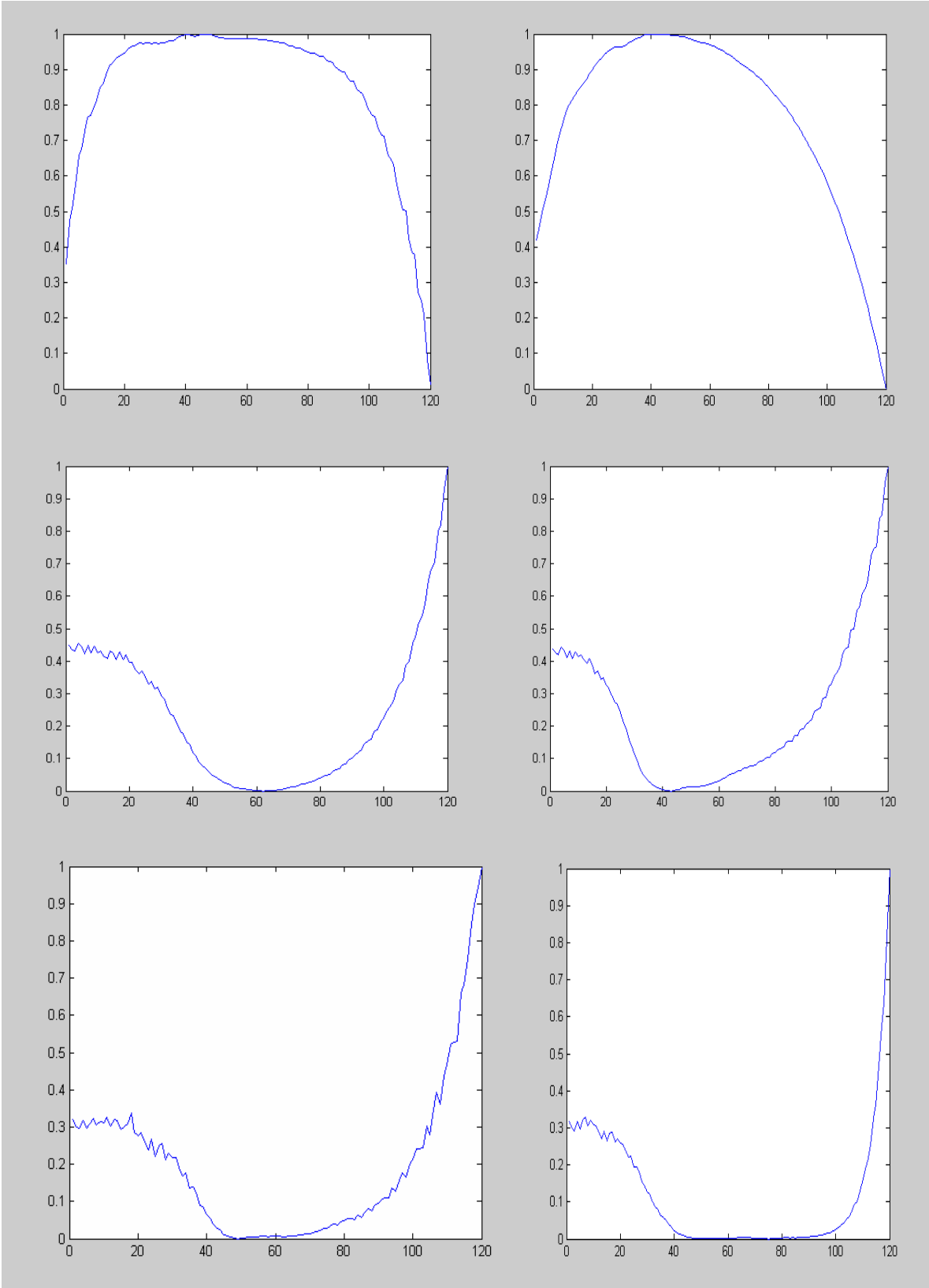
if nargin==2
    eval(setstr([real('y=') real(var) real(';')]));
else

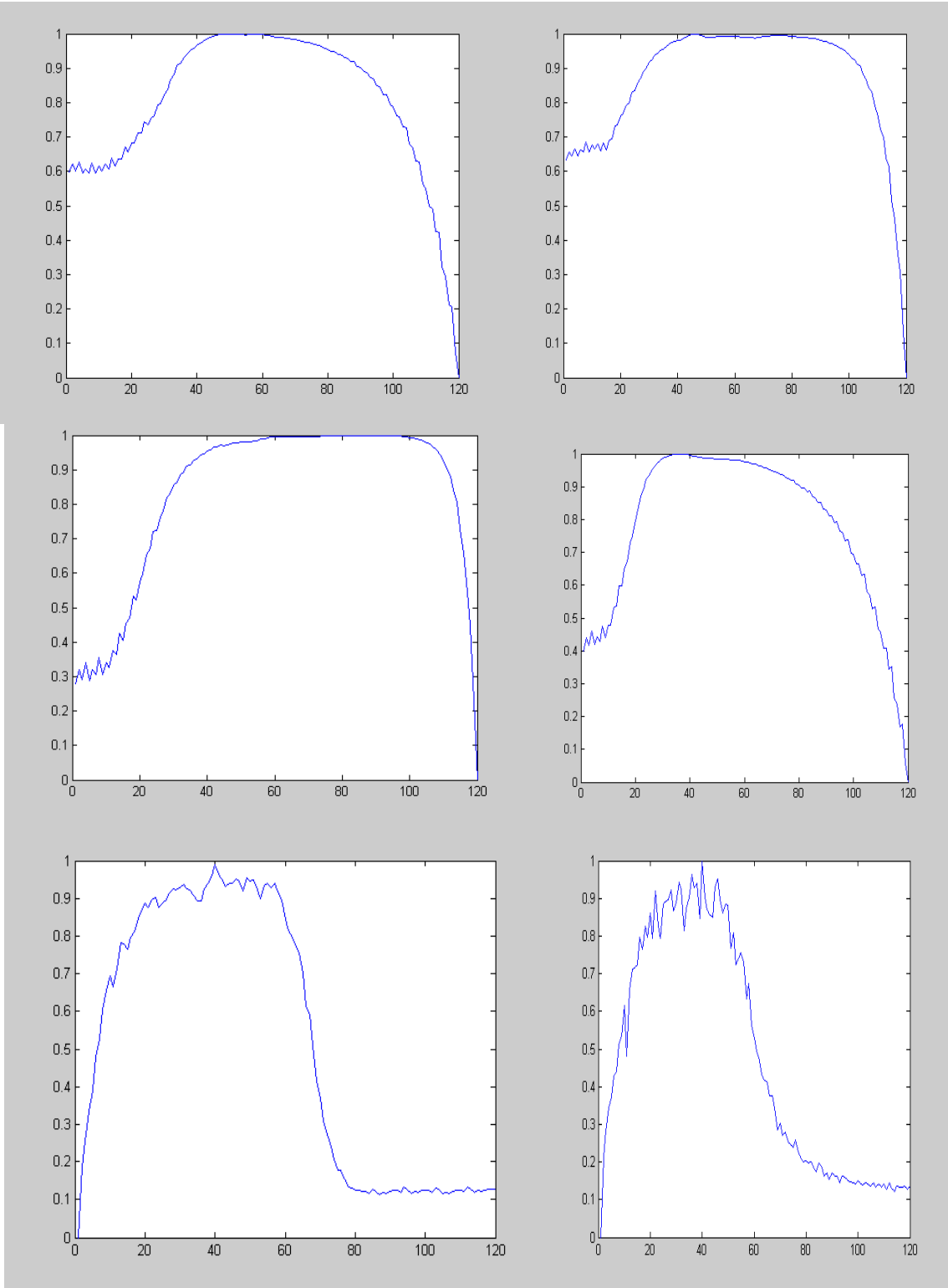
    i=fread(fid,1,'int');
    while feof(fid)==0
        p=fread(fid,[n 1],'double');
        if type==1
            A=fread(fid,[n n*(t-1)],'double');
        else
            A=fread(fid,[n n],'double');
        end
        m=fread(fid,[3 n],'double');
        V=fread(fid,[3 3*n],'double');
        samsz=fread(fid,1,'int');
        lkd=fread(fid,[1 samsz],'double');
        st=fread(fid,[samsz t],'float');
%       st=order(st,t);
        if(i==id)
            eval(setstr([real('y=') real(var) real(';')]));
        end
        i=fread(fid,1,'int');
    end
end
fclose(fid);

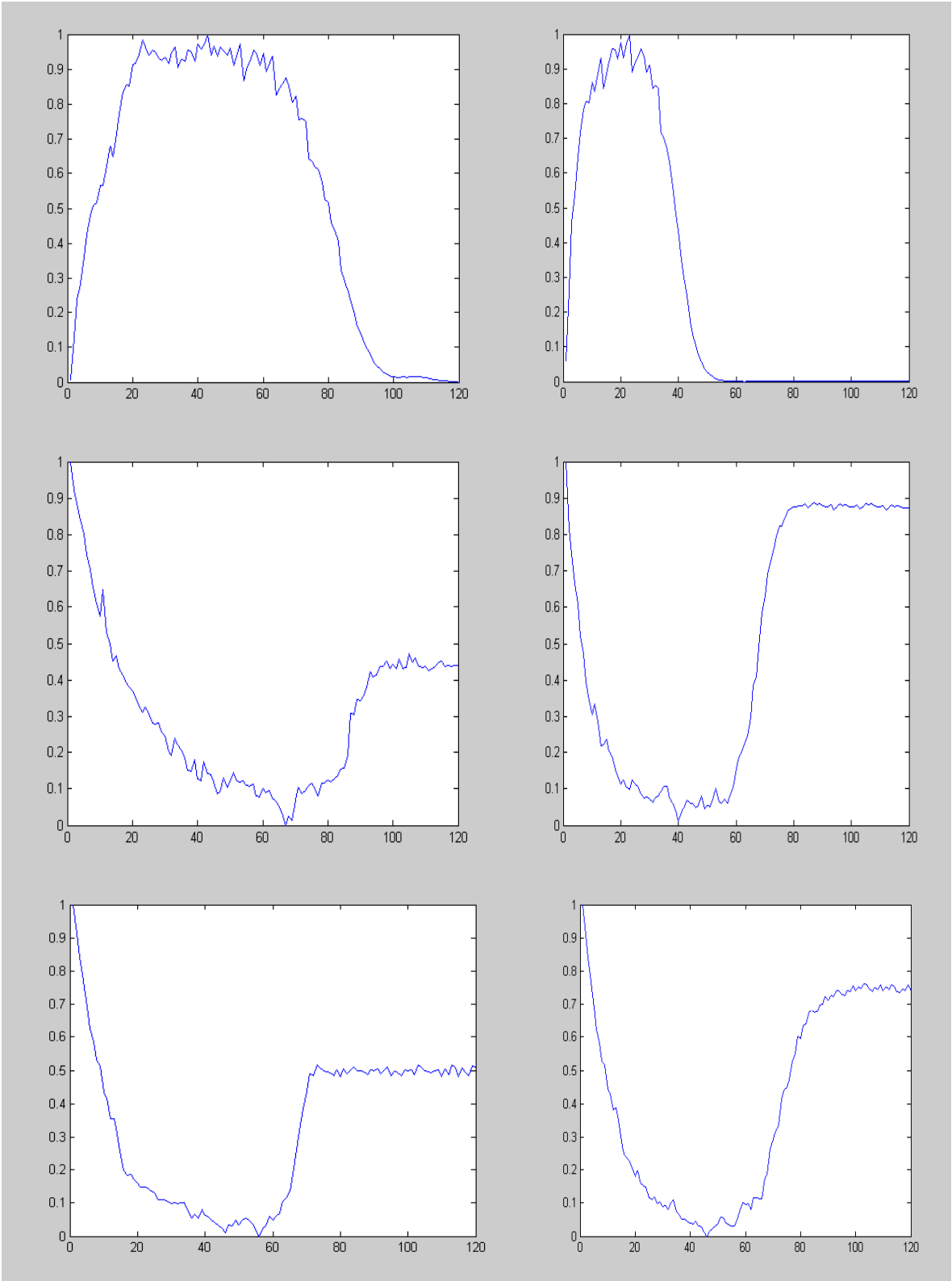
```

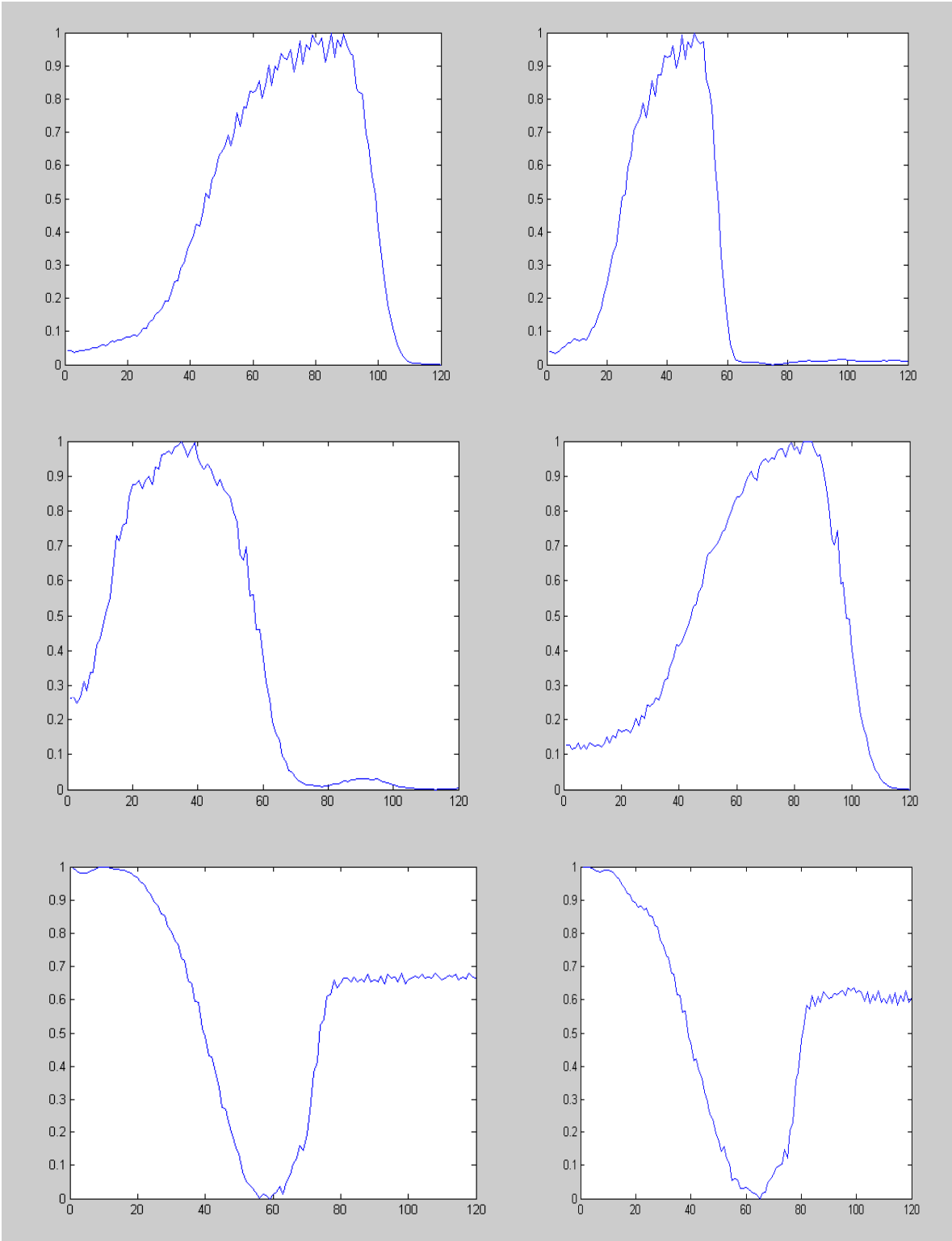
APPENDIX B: PRELIMINARY TEST PATTERNS

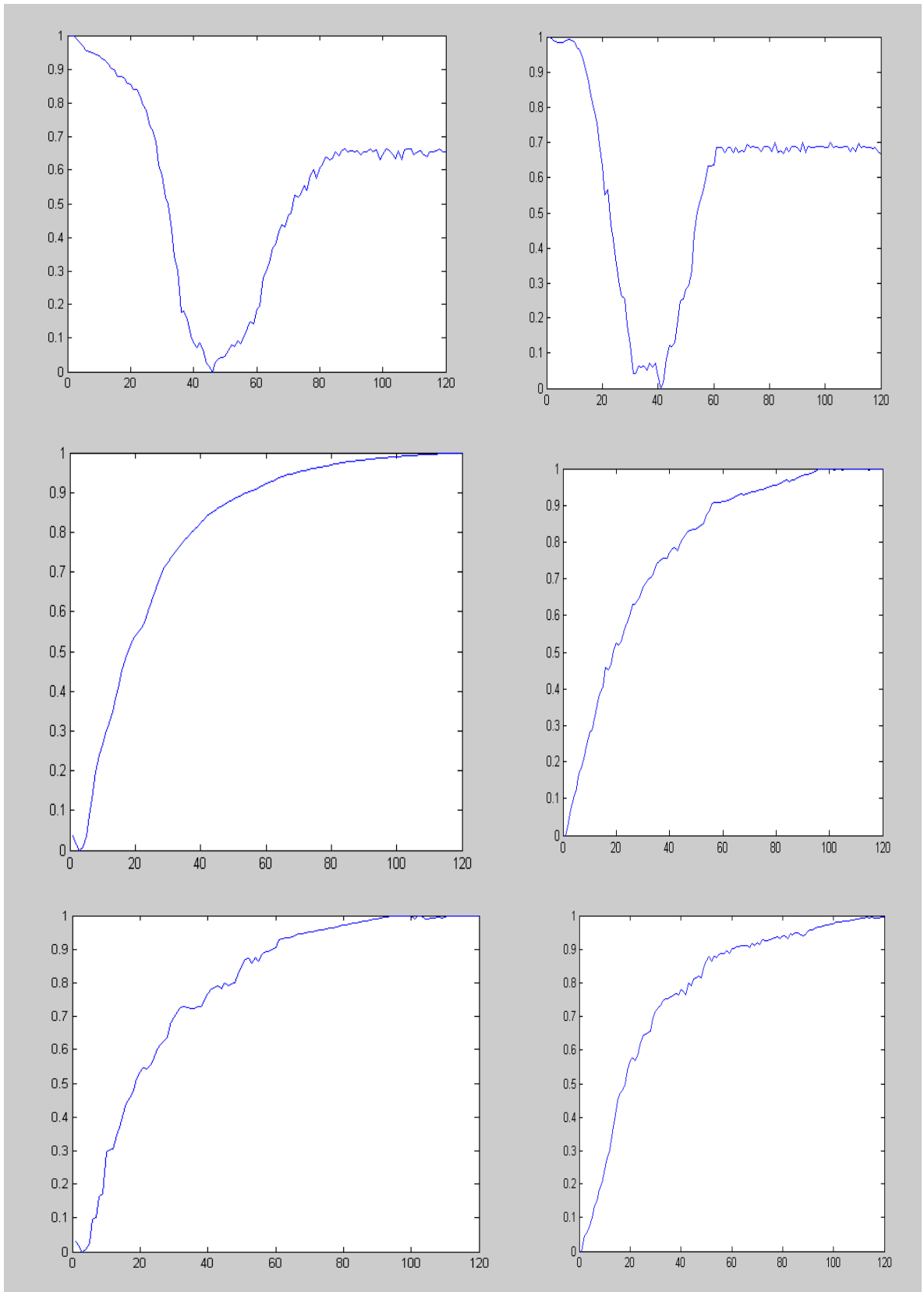


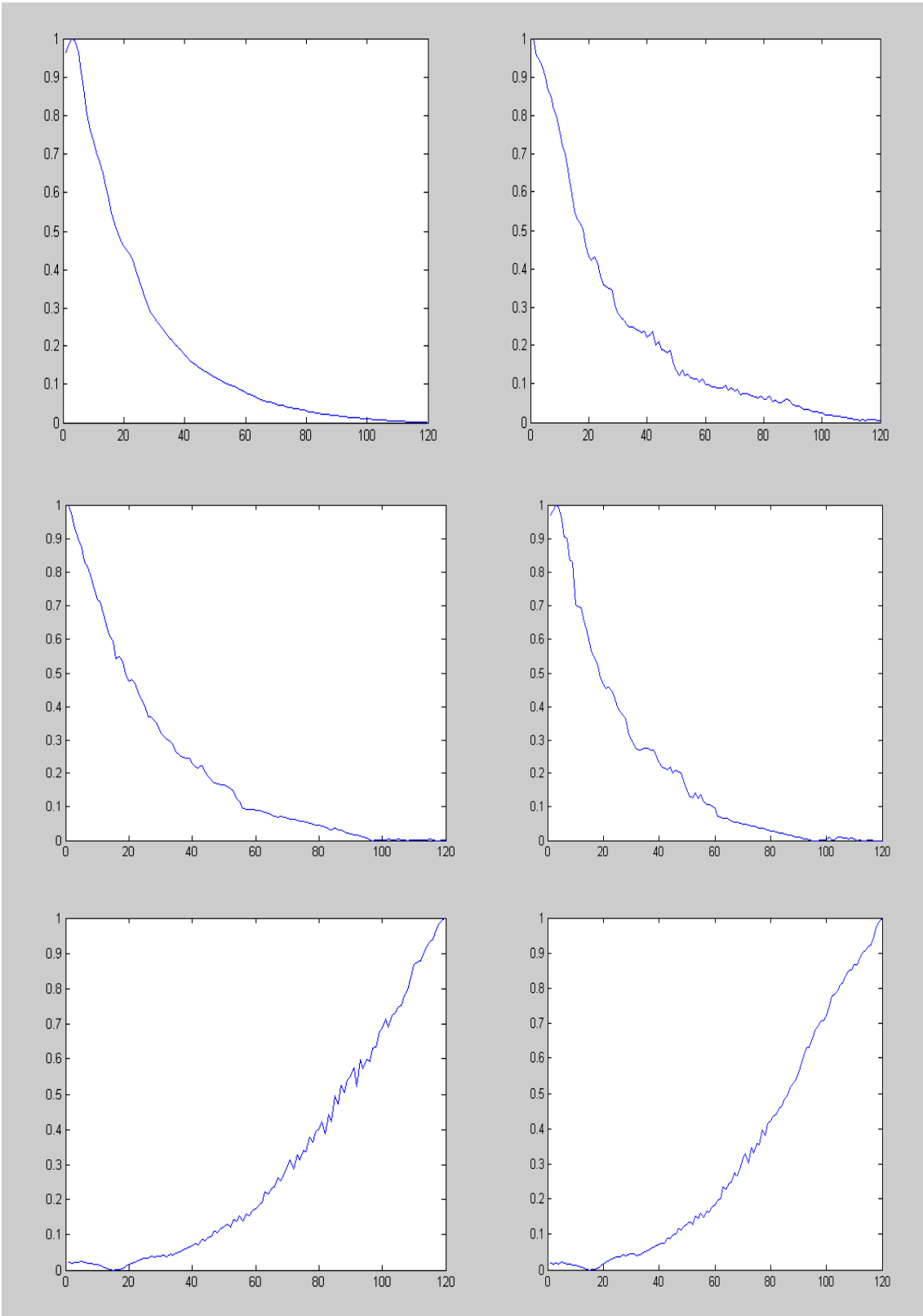


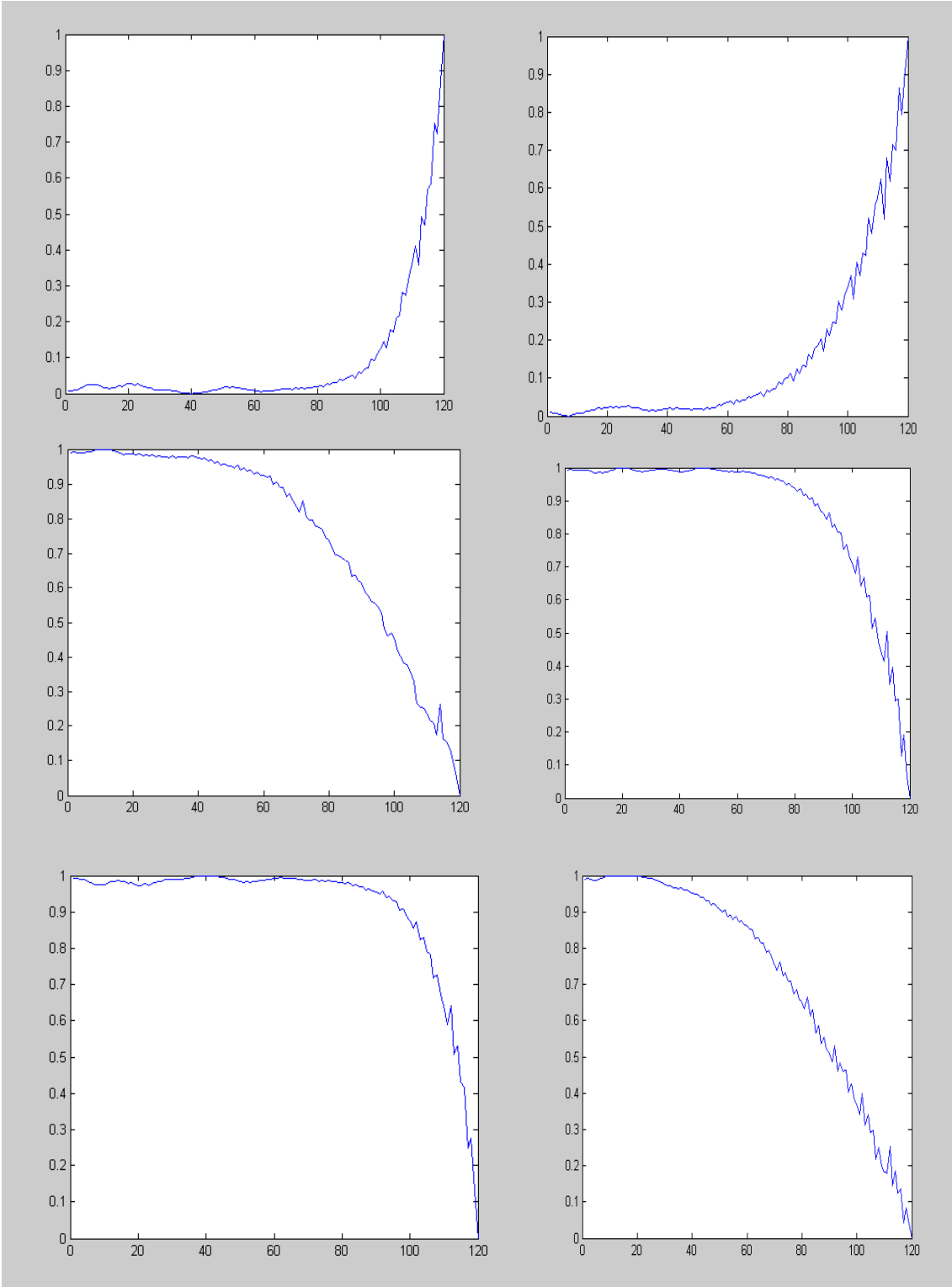


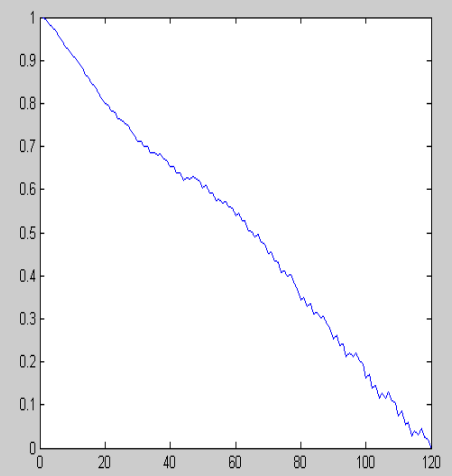
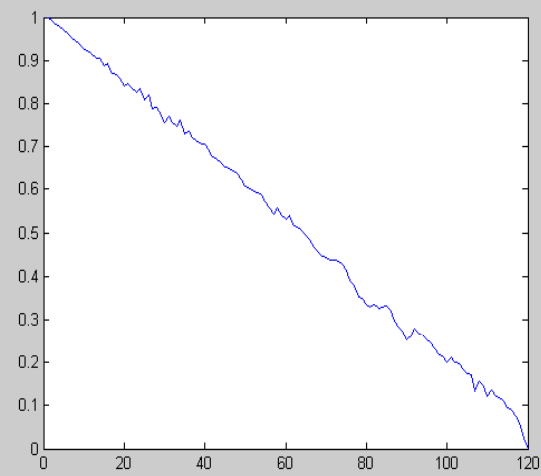
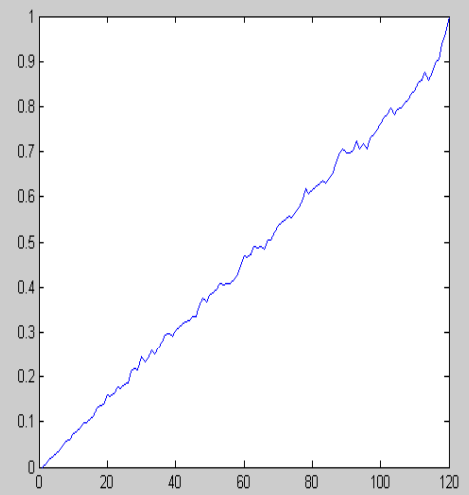
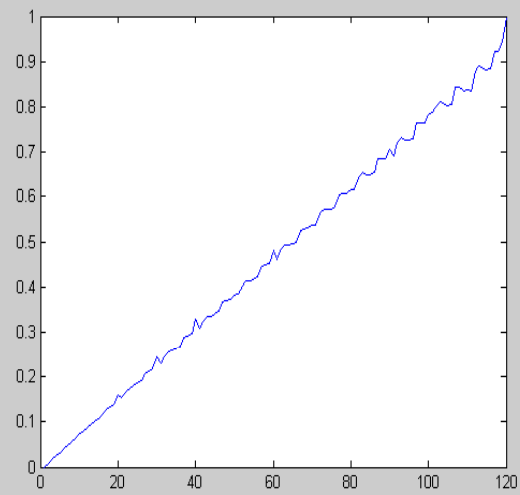
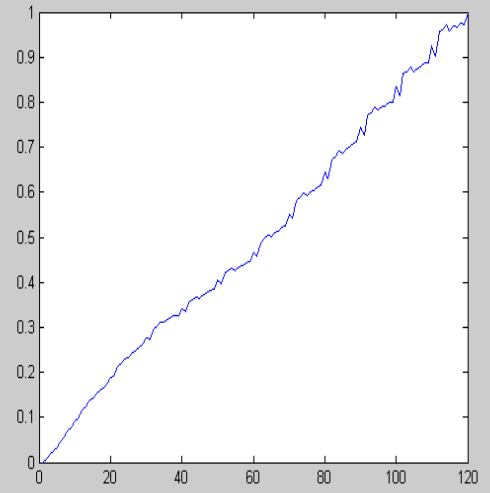
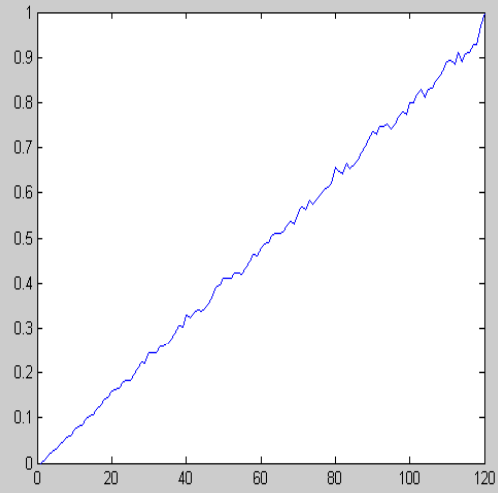


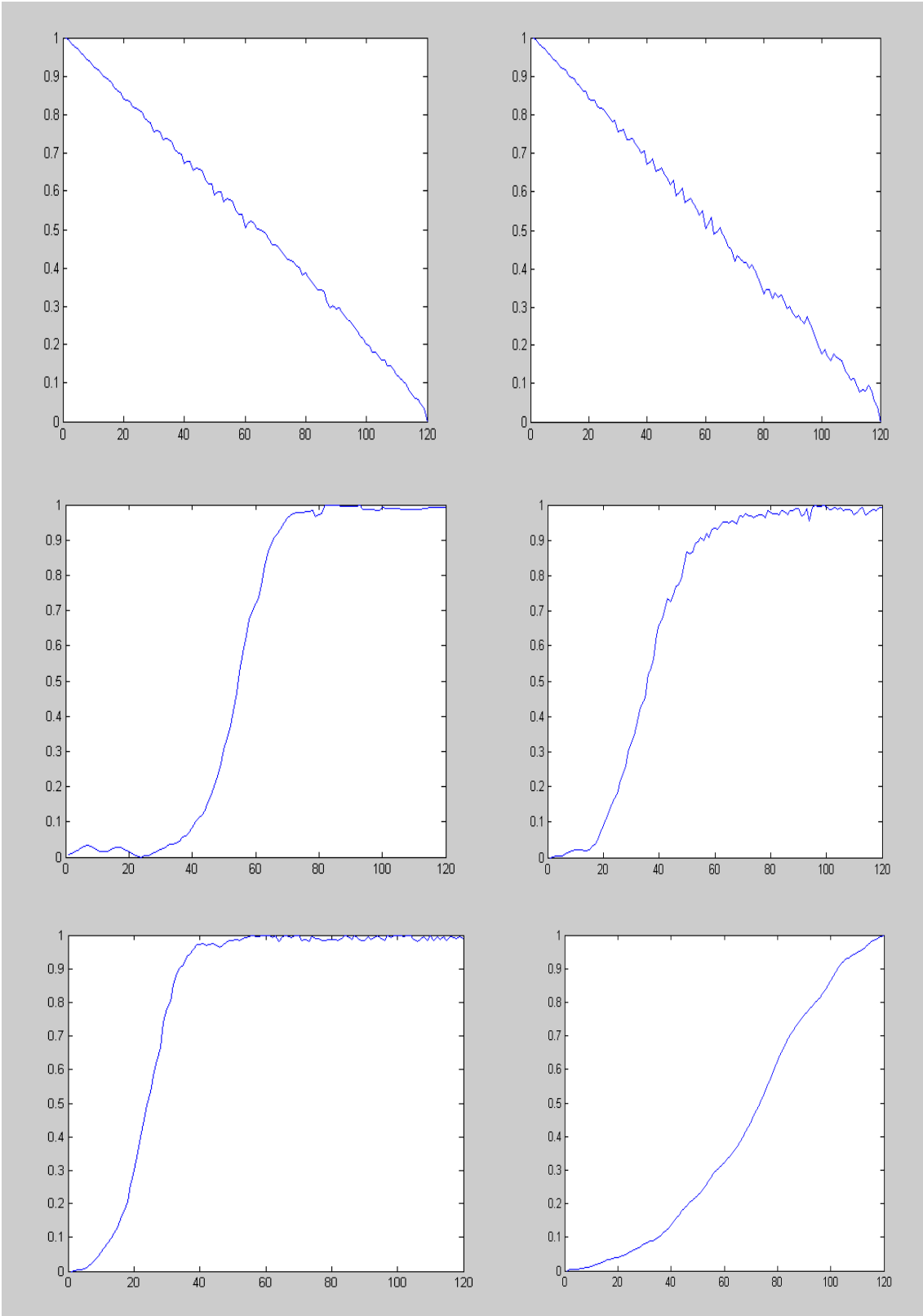


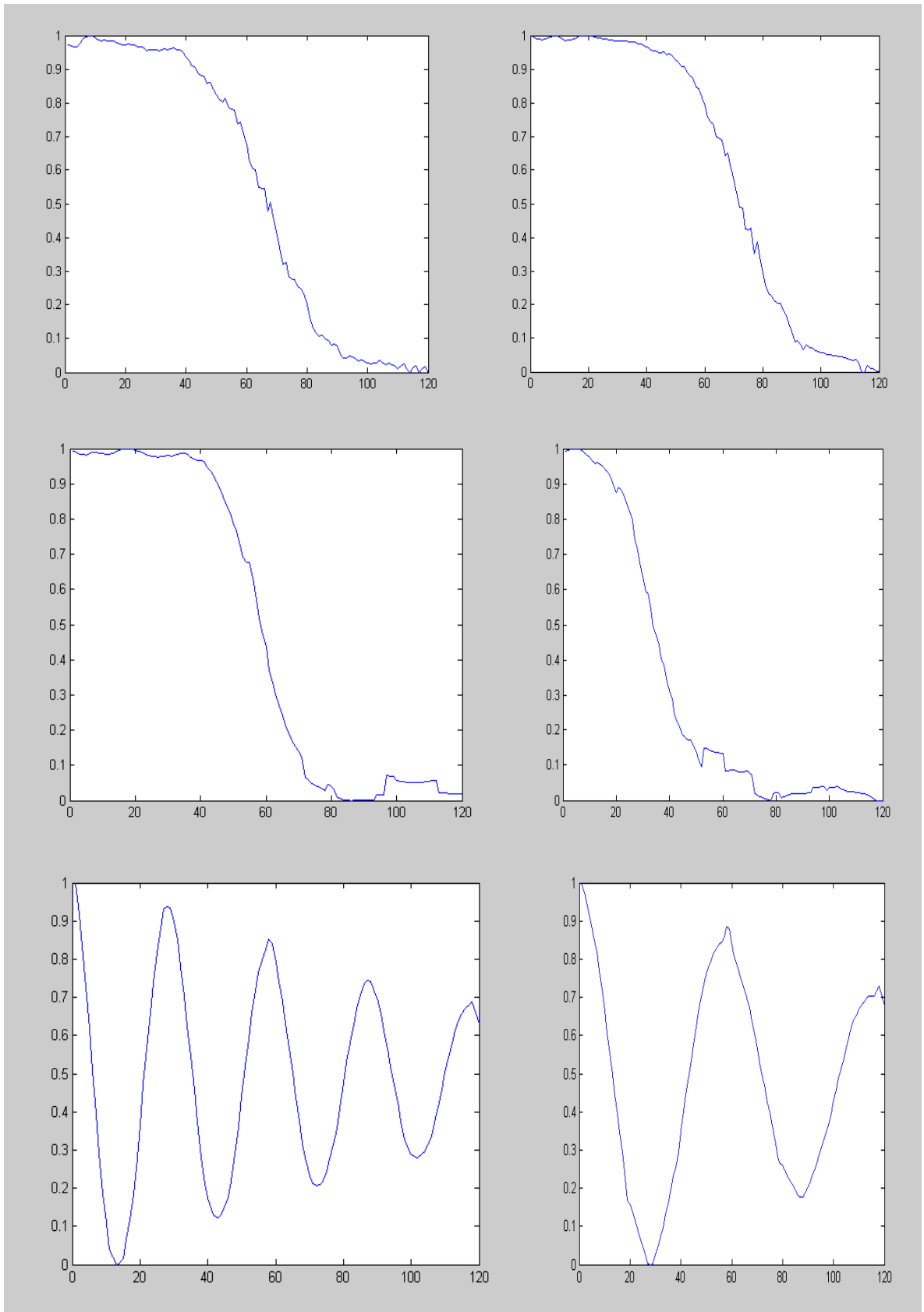


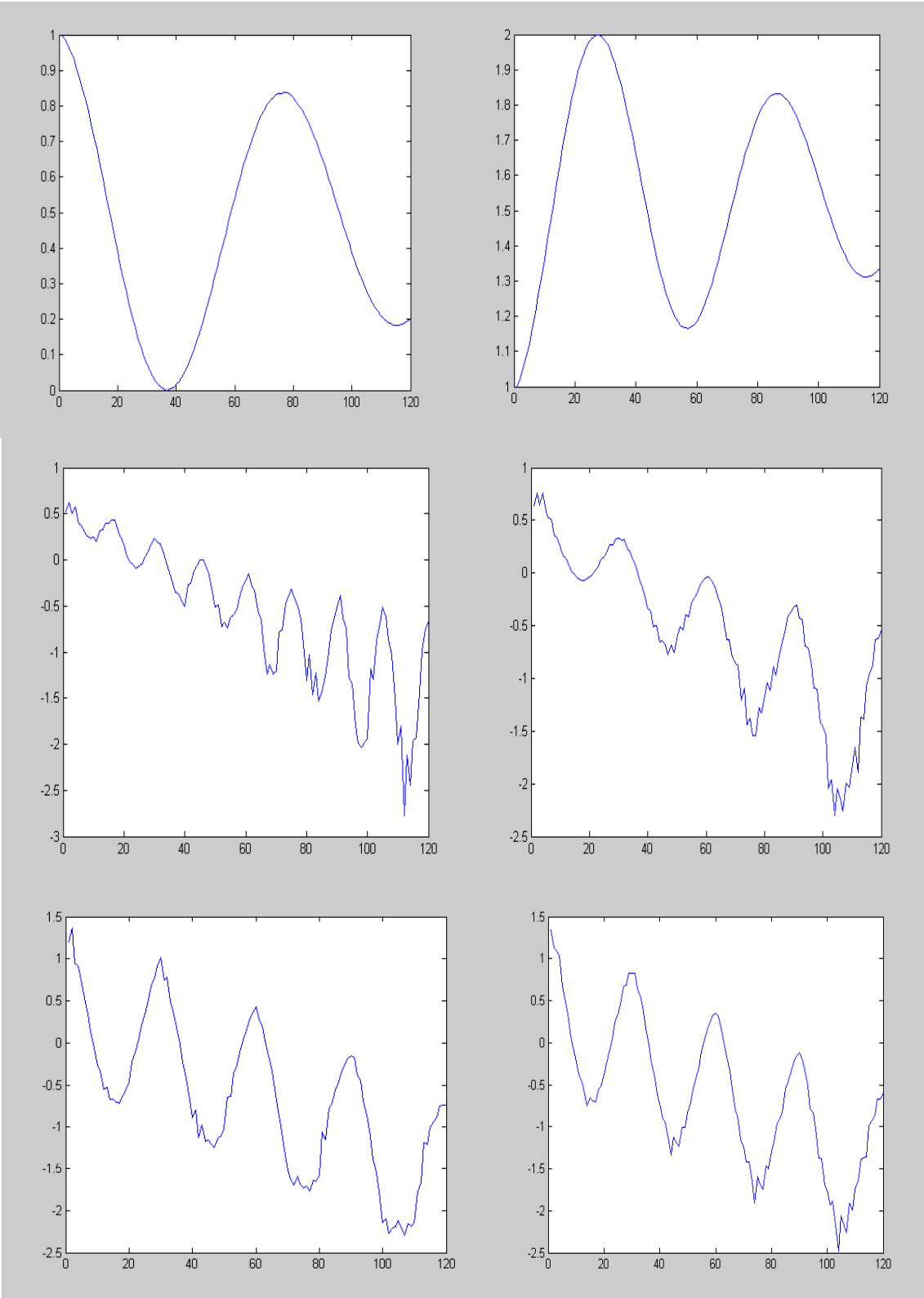


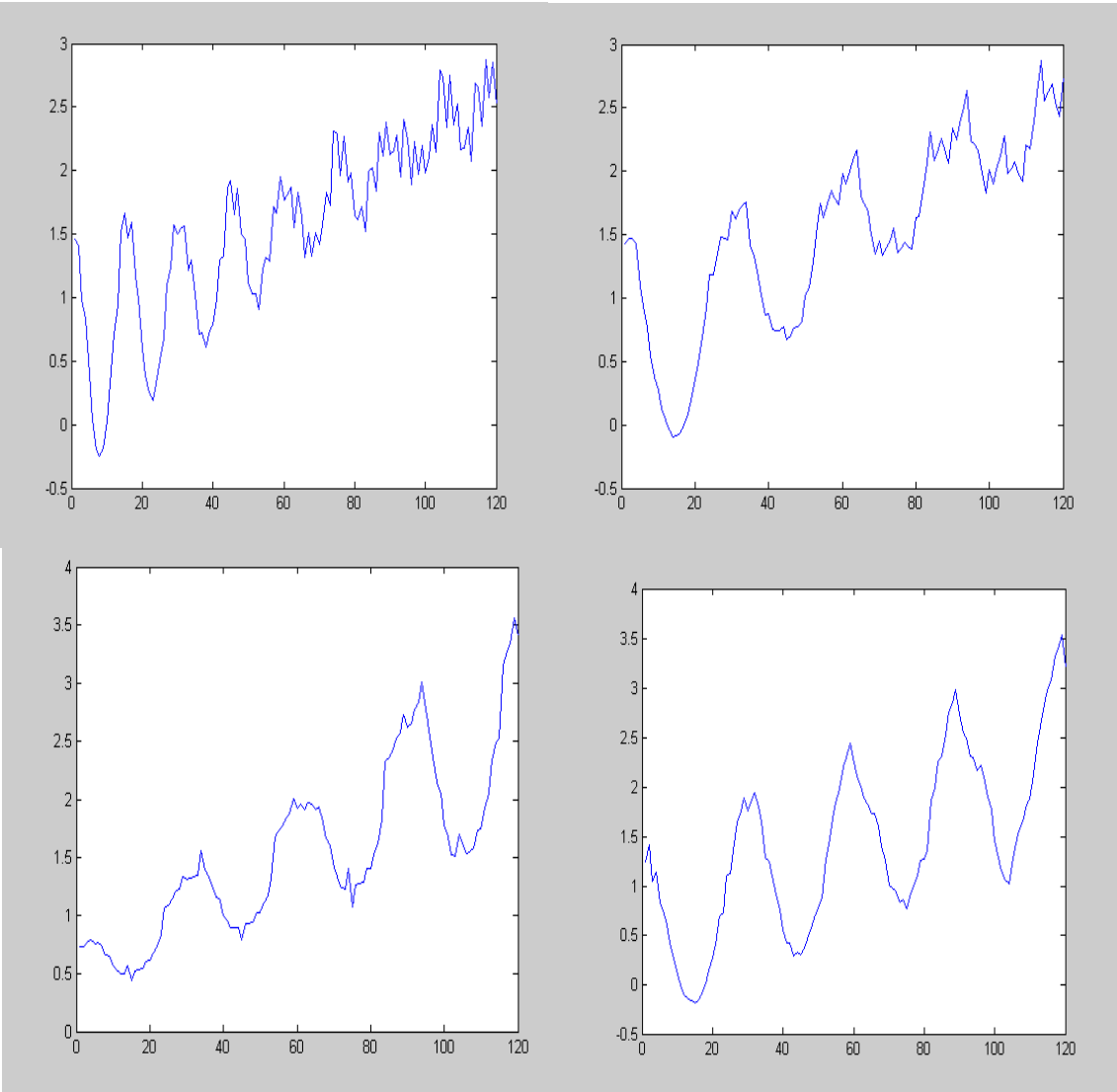












APPENDIX C: TURING TEST

Instructions:

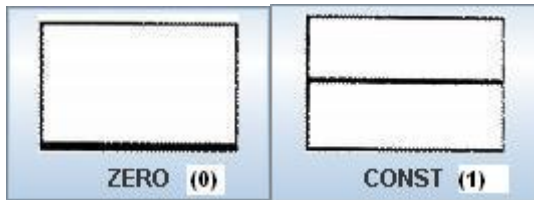
This test is about the reliability of ISTS (indirect-structure testing software), which is designed to classify generic dynamic patterns.

There are 106 dynamic series patterns obtained from different Istanbul Stock Exchange (ISE) data. The purpose of the Turing test is to have human experts (subjects) and ISTS algorithm classify these 106 patterns into one or more of the existing generic patterns, and then compare the results of human expert vs. machine classifications. There are 21 generic patterns in the attached template.

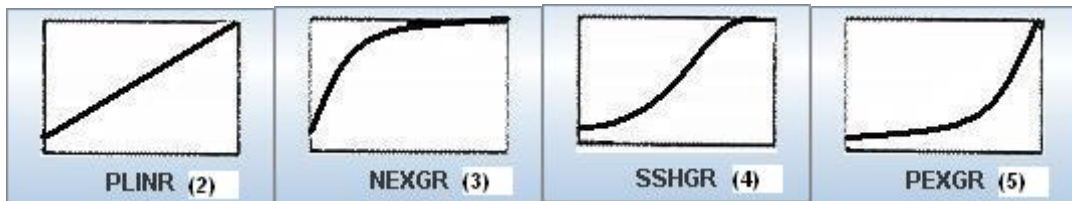
Since there exists substantial noise in real data and since ISTS algorithm does pre-smoothing in such situations, both raw and smoothed versions of each real time series is presented to the human subjects. For each time series, the subject is asked to classify in one to four pattern classes, from the existing template. The subject can thus provide up to four classifications, with decreasing order of goodness of fit. Note that the subject is required to choose at least one class, but is free to choose up to three other classes if she/he feels that there are other close candidate classes that the data may belong to. Finally, the subject may also suggest a near generic pattern class if she/he believes that a certain generic pattern missing in the given template that fits much better for the given data.

Template of generic dynamic patterns

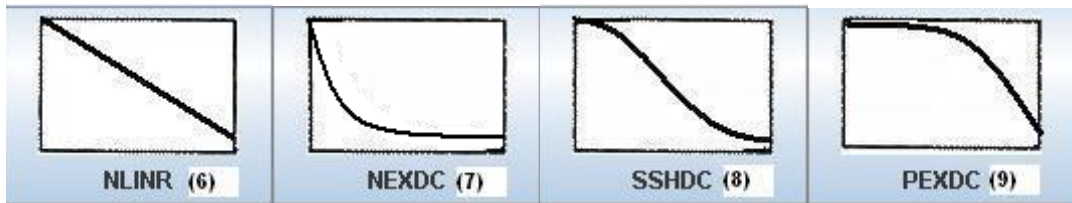
Constant:



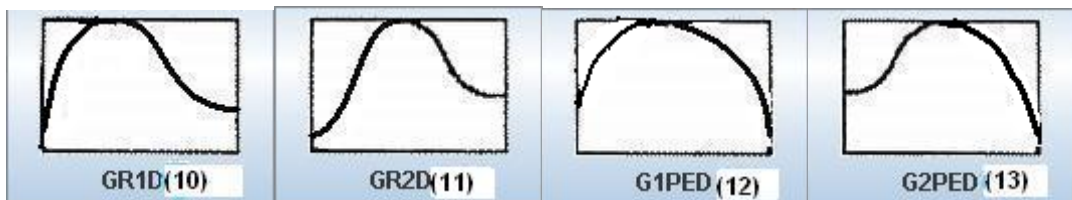
Growth:



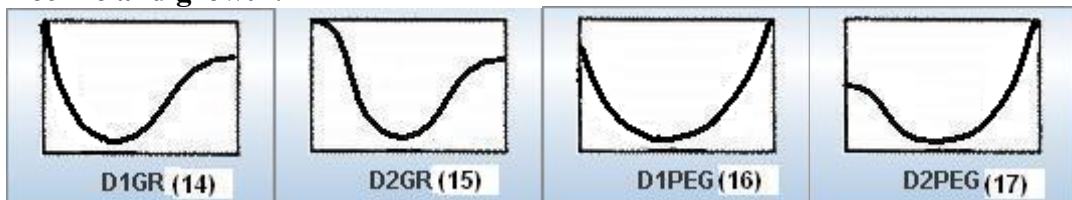
Decline:



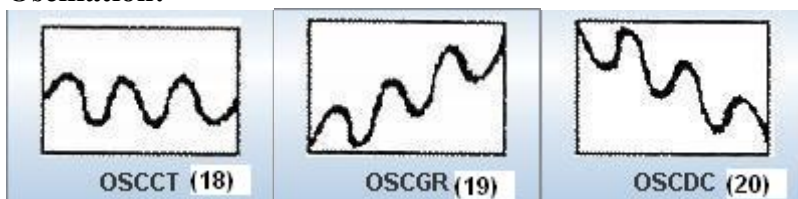
Growth and decline:



Decline and growth:



Oscillation:



List of generic dynamic patterns

Zero0: Zero

Const: Constant

Plinr: Linear growth

Nexgr: Negative exponential growth

Sshgr: S-shaped growth

Pexgr: Positive exponential growth

Nlinr: Linear decline

Nexdc: Negative exponential decline

Sshdc: S-shaped decline

Pexdc: Positive exponential collapse

Gr1d: Growth with a decreasing rate followed by decline to equilibrium

Gr2d: S-shaped growth and decline to equilibrium

G1ped: Growth with decreasing rate followed by a positive exponential collapse

G2ped: S-shaped growth followed by positive exponential collapse

D1gr: Decline with a decreasing rate followed by growth to equilibrium

D2gr: S-shaped decline followed by a growth to equilibrium

D1peg: Decline with decreasing rate followed by positive exponential growth

D2peg: S-shaped decline followed by positive exponential growth

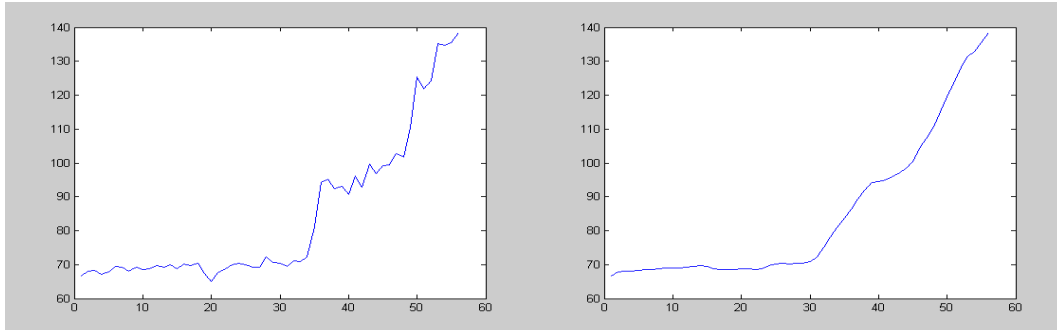
Oscct: Oscillation around constant mean

Oscgr: Oscillation around a linearly growing trend

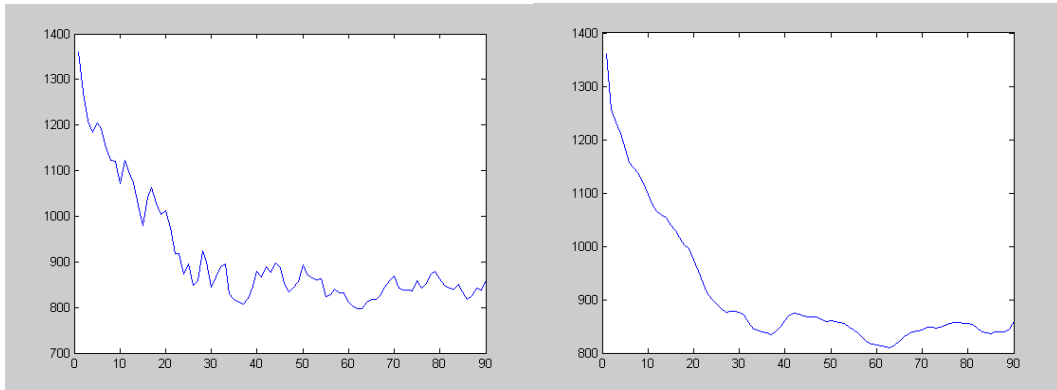
Oscdc: Oscillation around a linearly declining trend

TIME SERIES EXPOSED TO HUMAN SUBJECTS IN TURING TEST

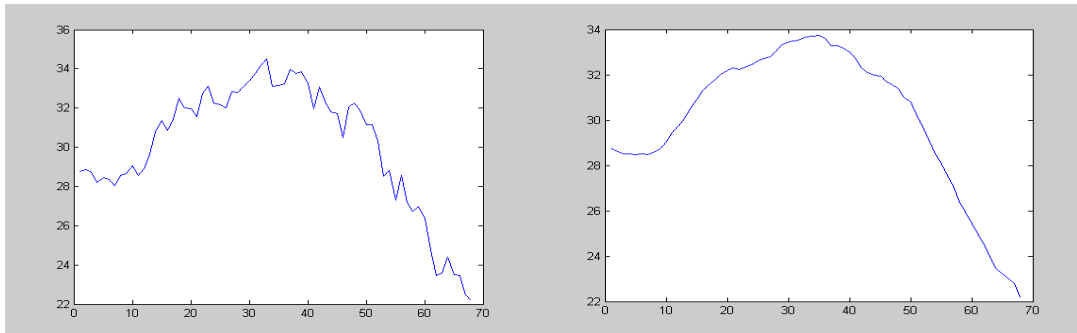
Time Series 1



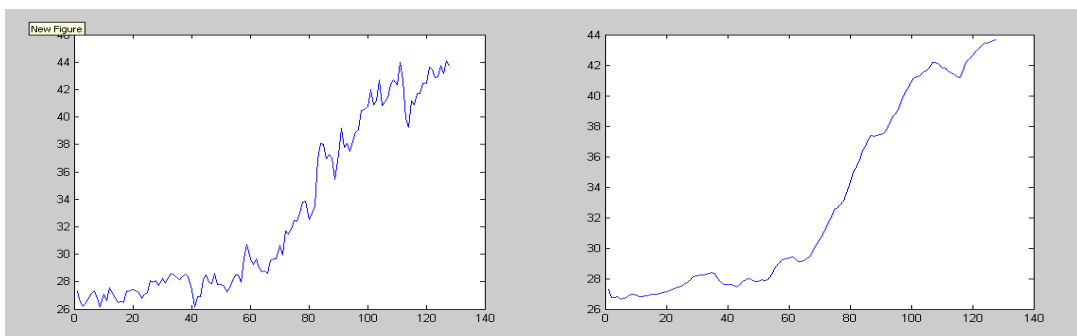
Time Series2



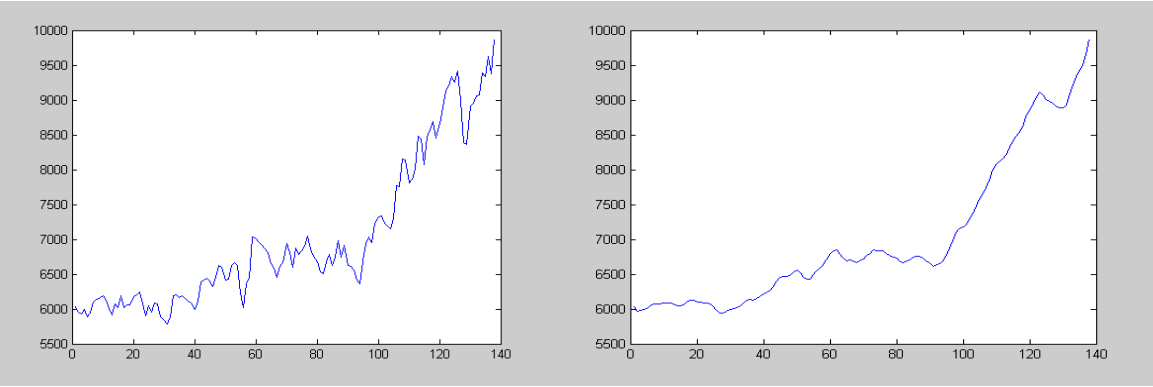
Time Series 3



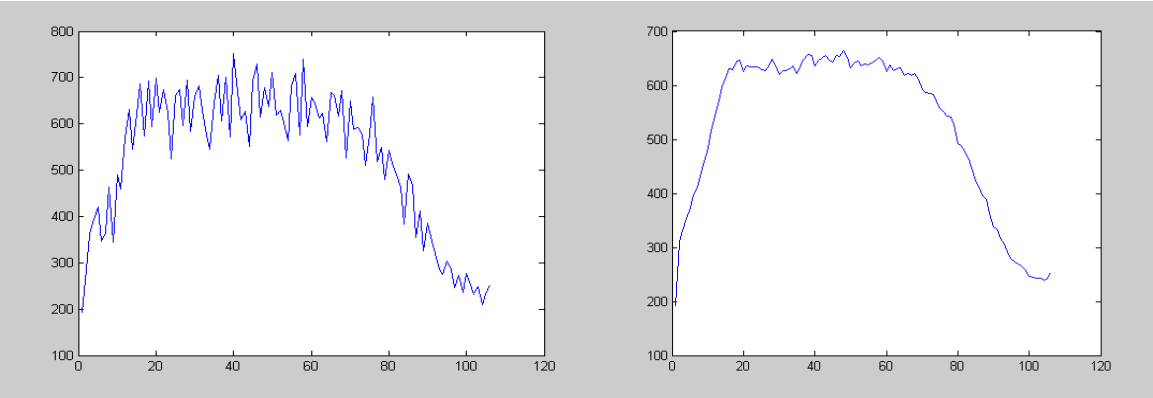
Time Series 4



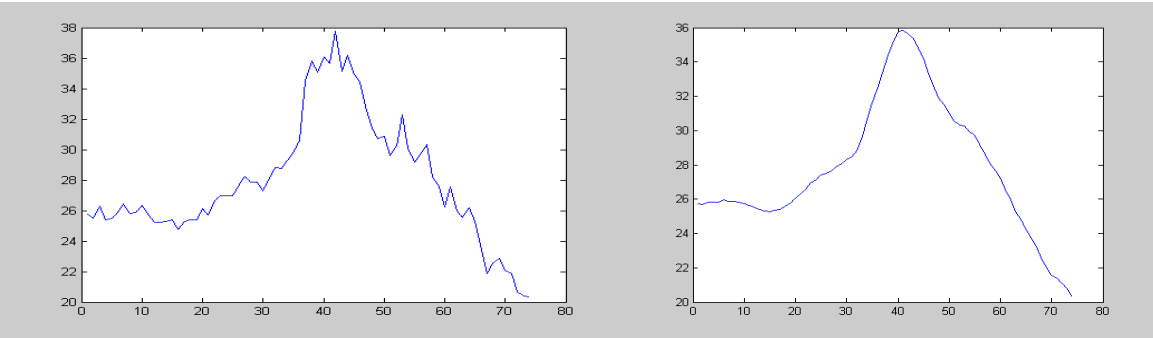
Time Series 5



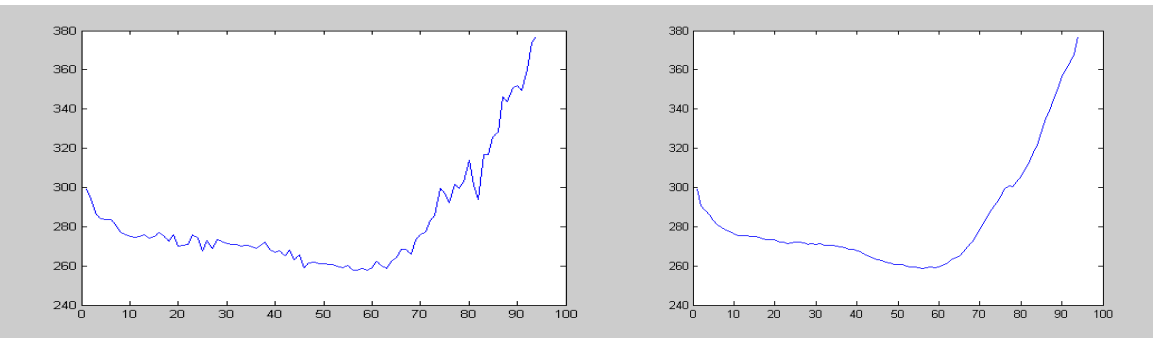
Time Series 6



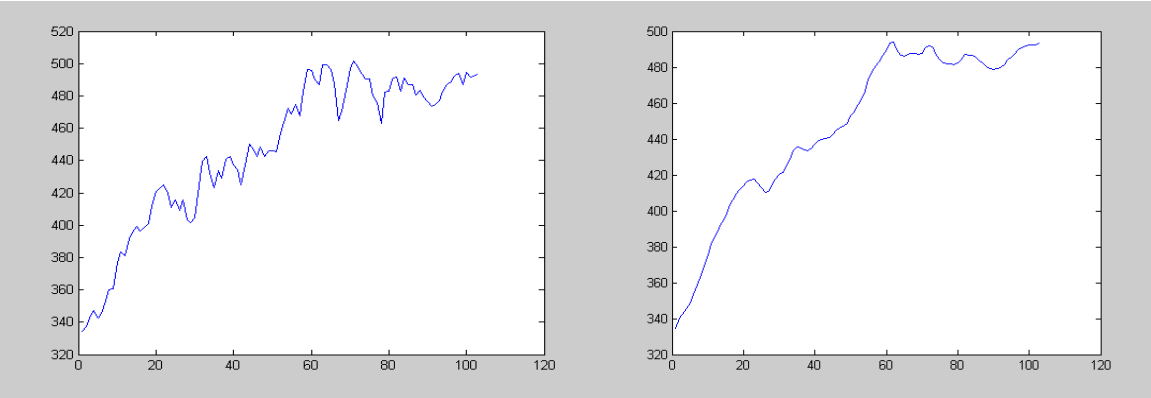
Time Series 7



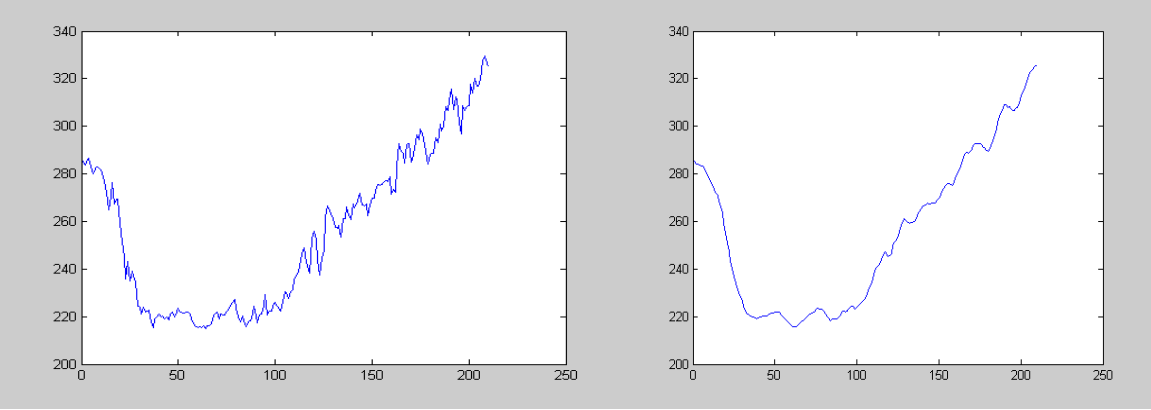
Time Series 8



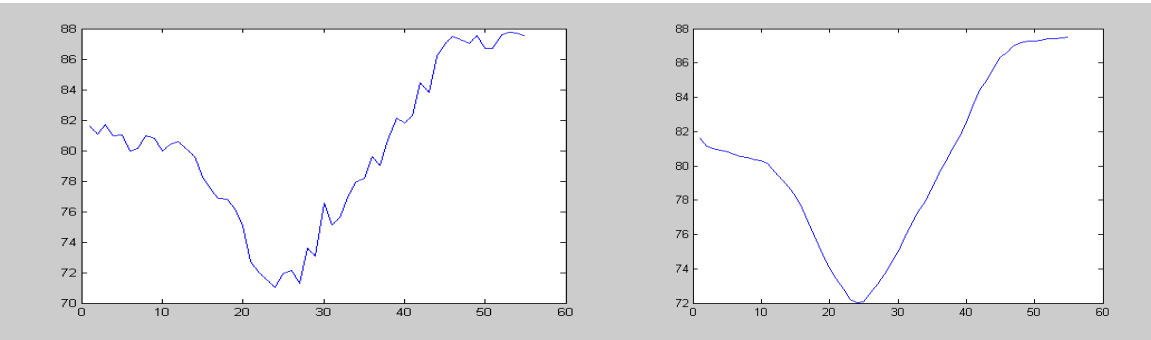
Time Series 9



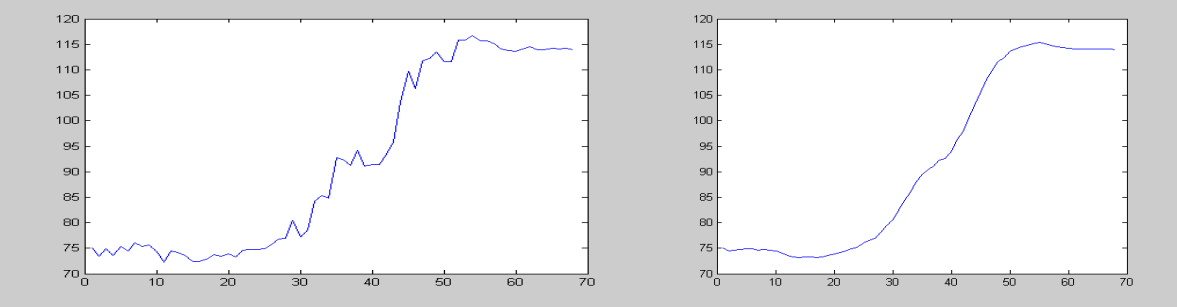
Time Series 10



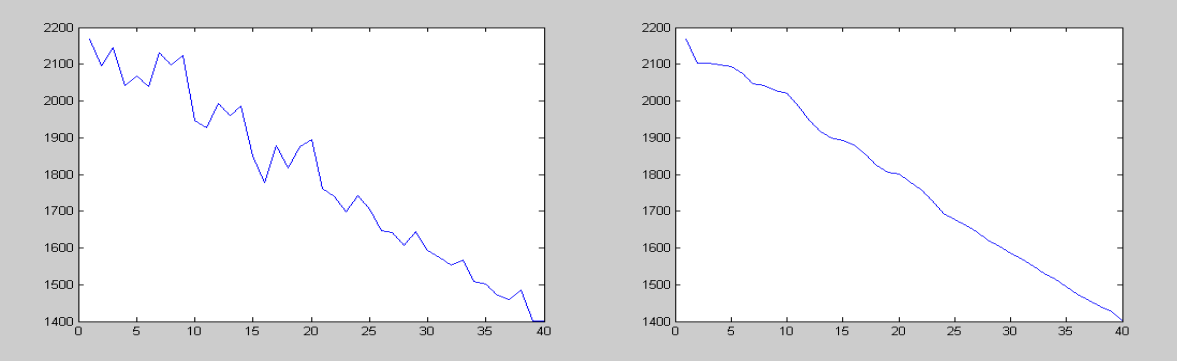
Time Series 11



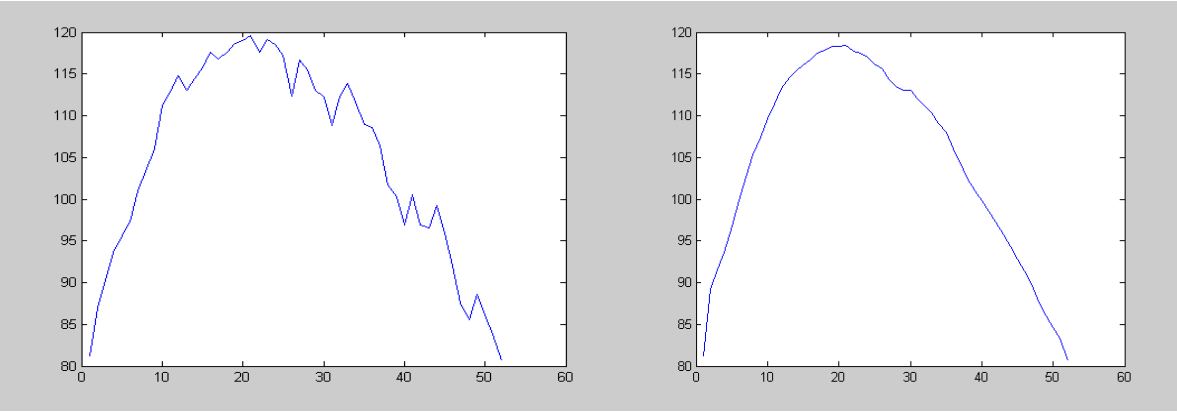
Time Series 12



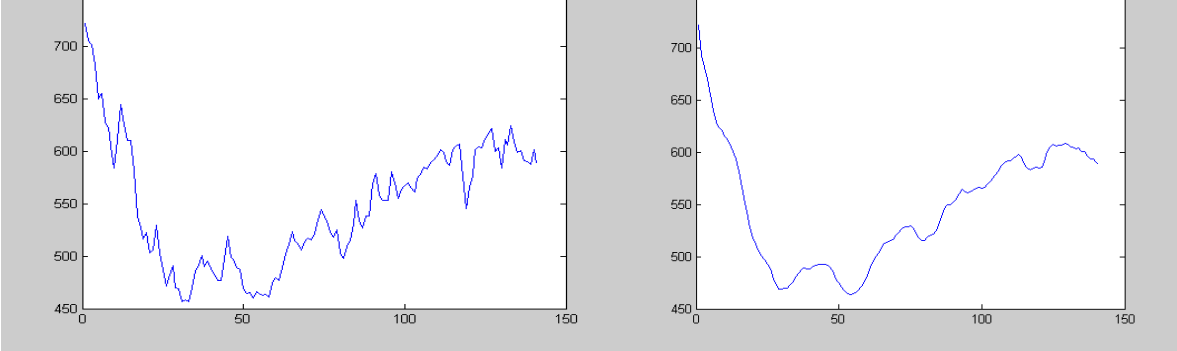
Time Series 13



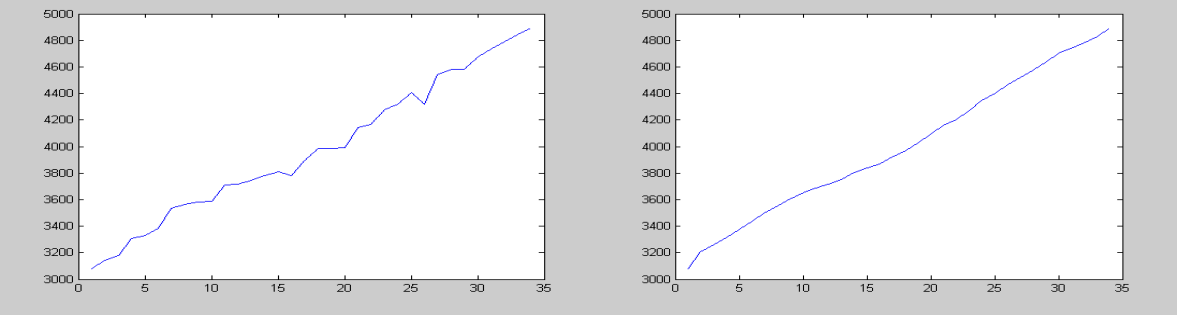
Time Series 14



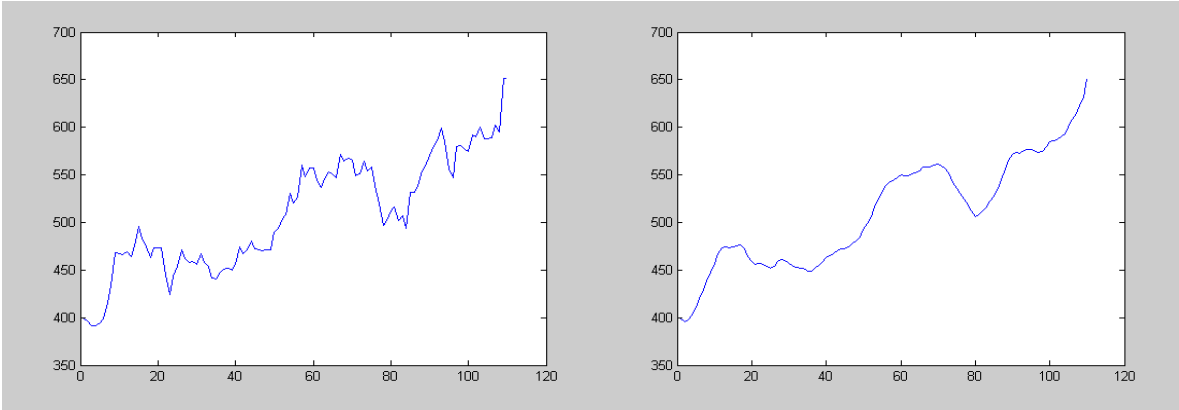
Time Series 15



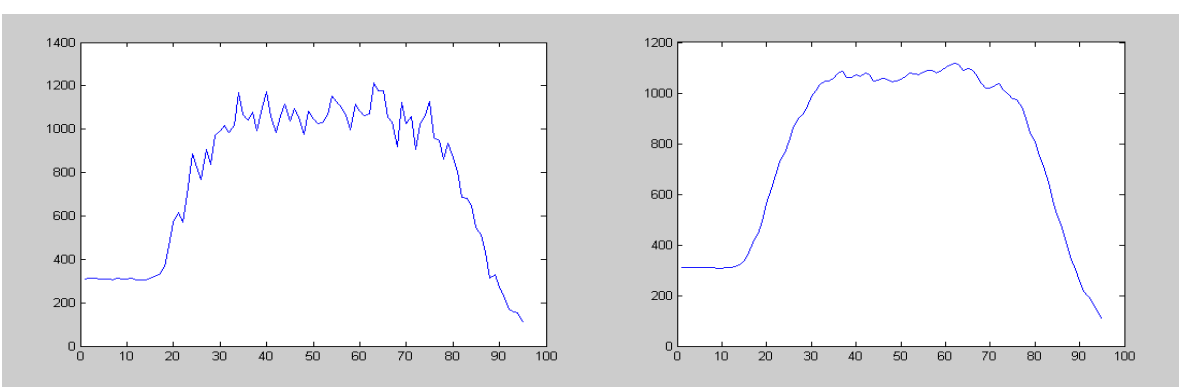
Time Series 16



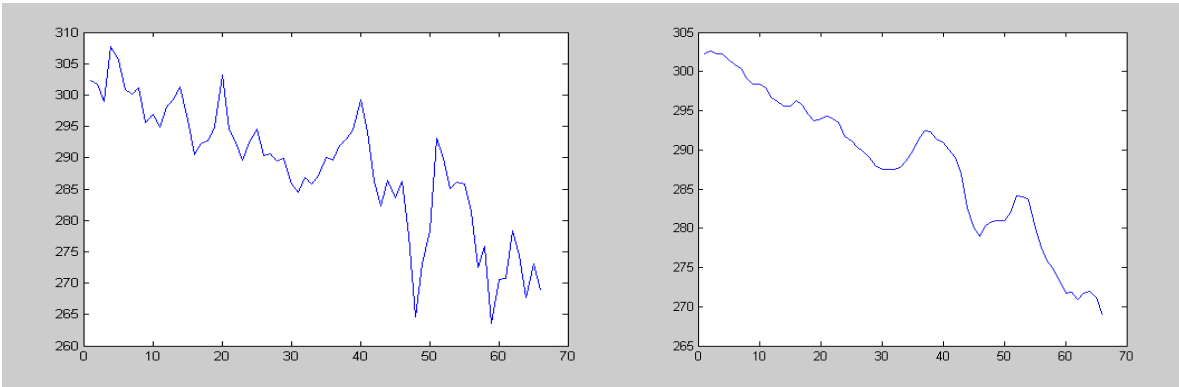
Time Series 17



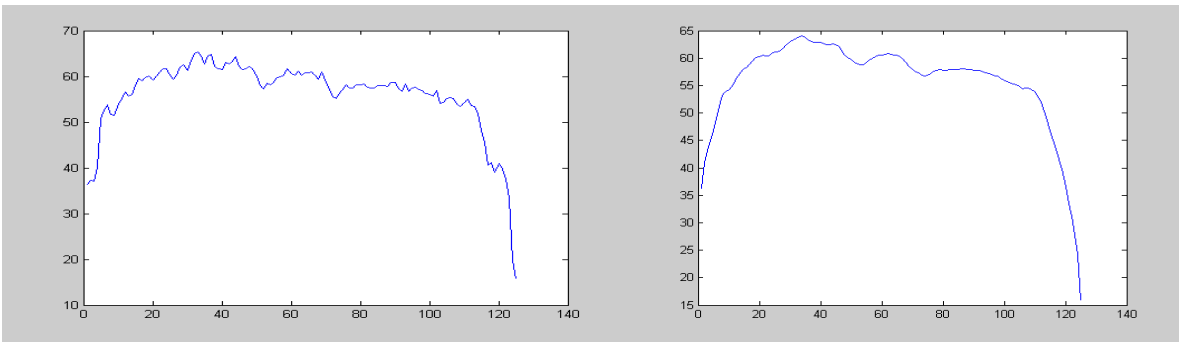
Time Series 18



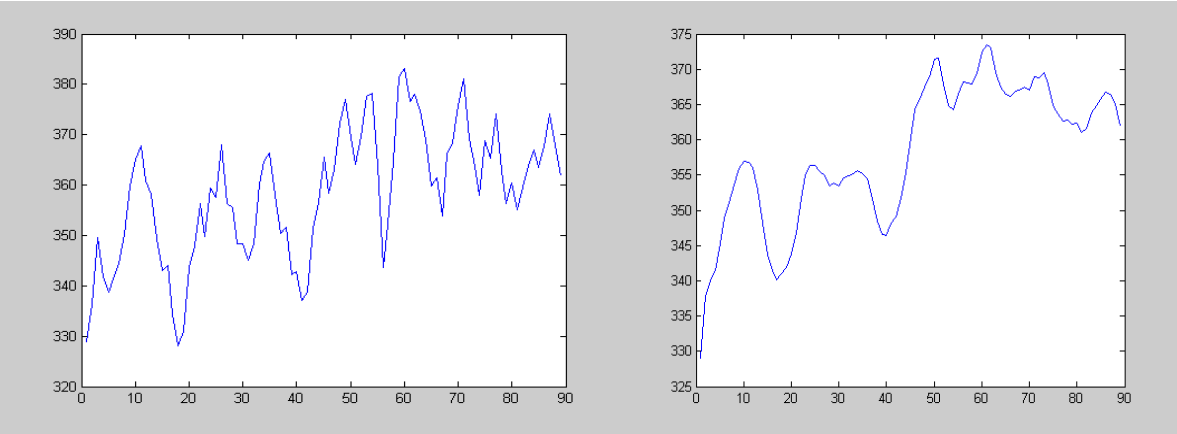
Time Series 19



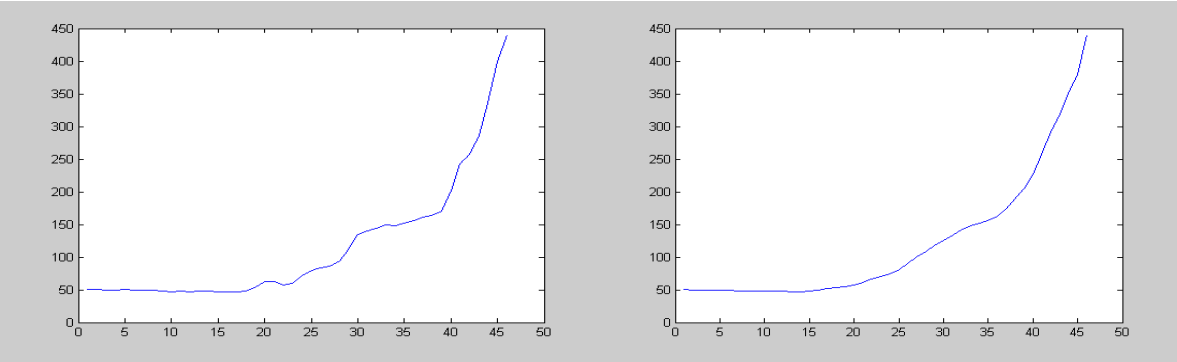
Time Series 20



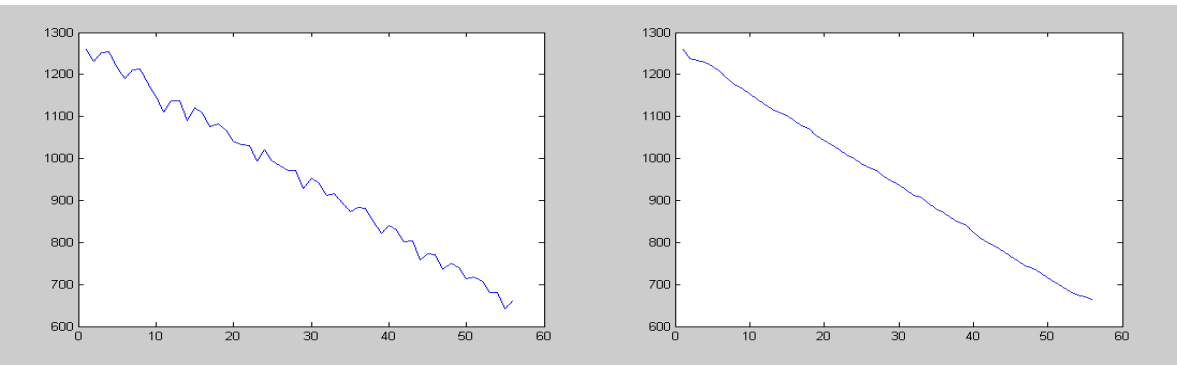
Time Series 21



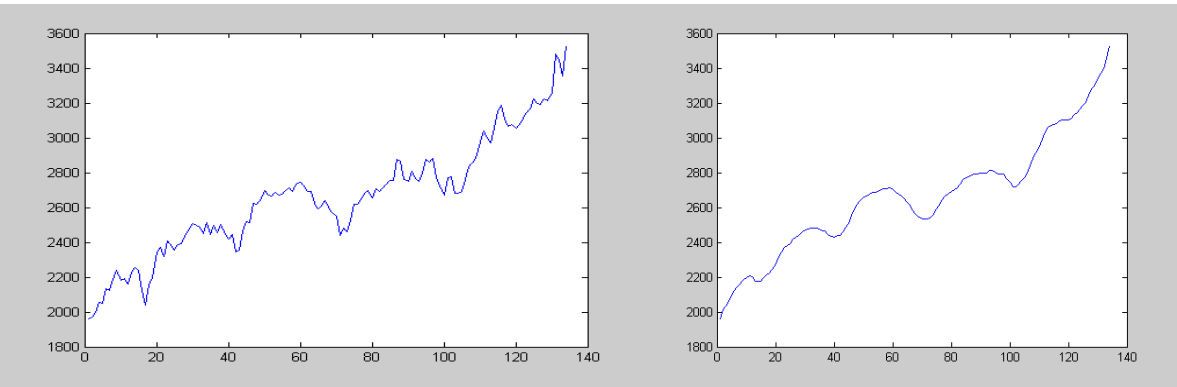
Time Series 22



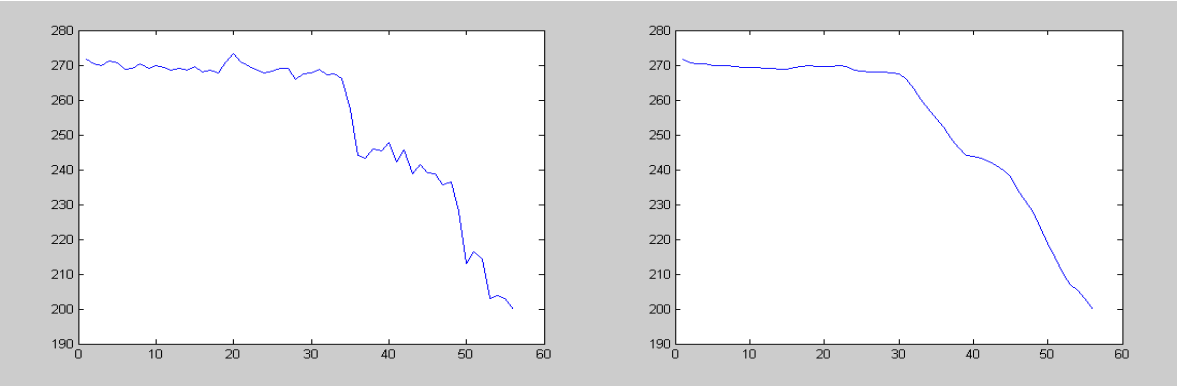
Time Series 23



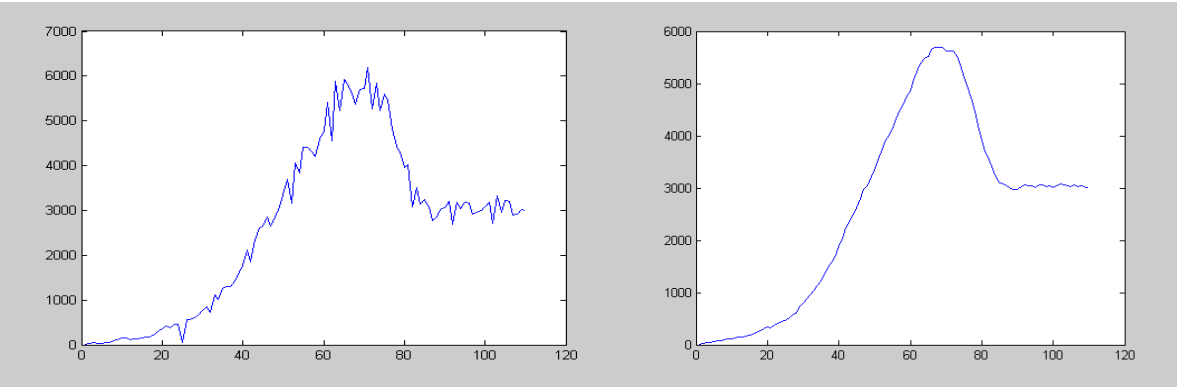
Time Series 24



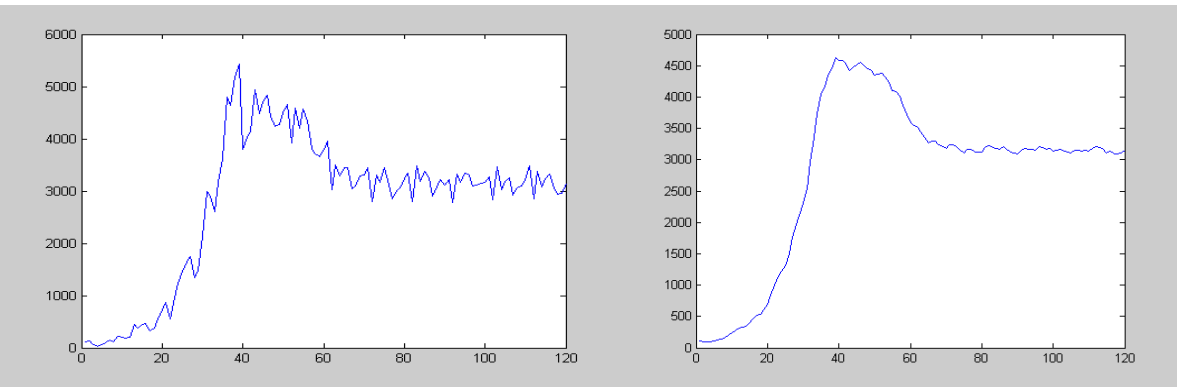
Time Series 25



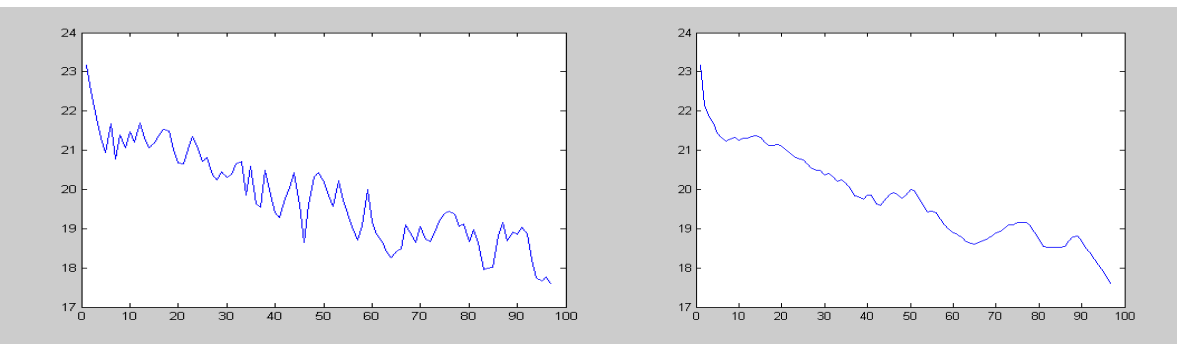
Time Series 26



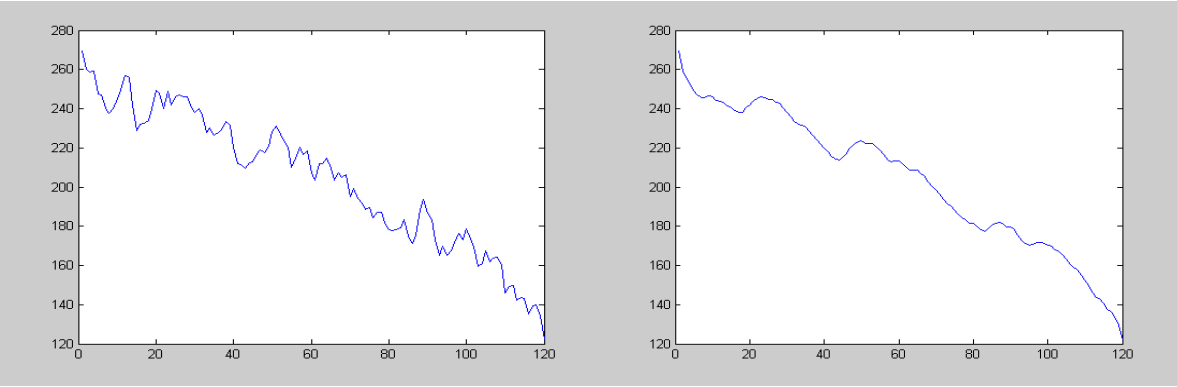
Time Series 27



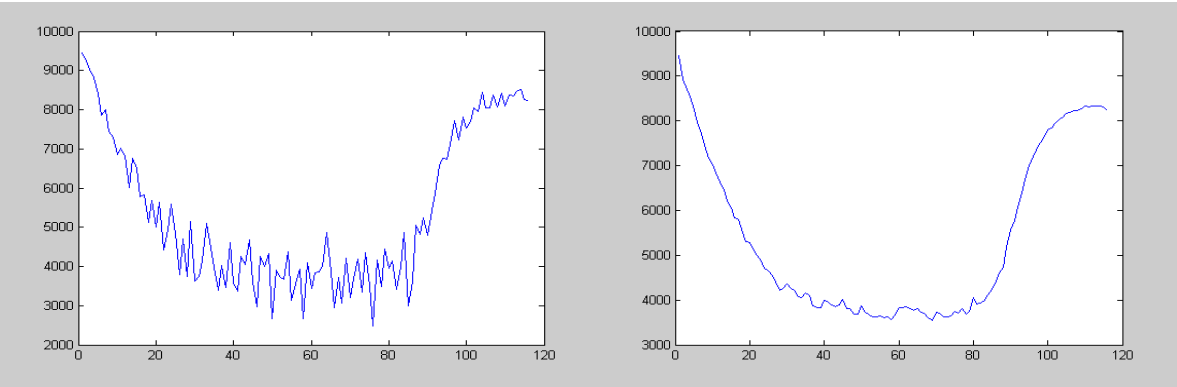
Time Series 28



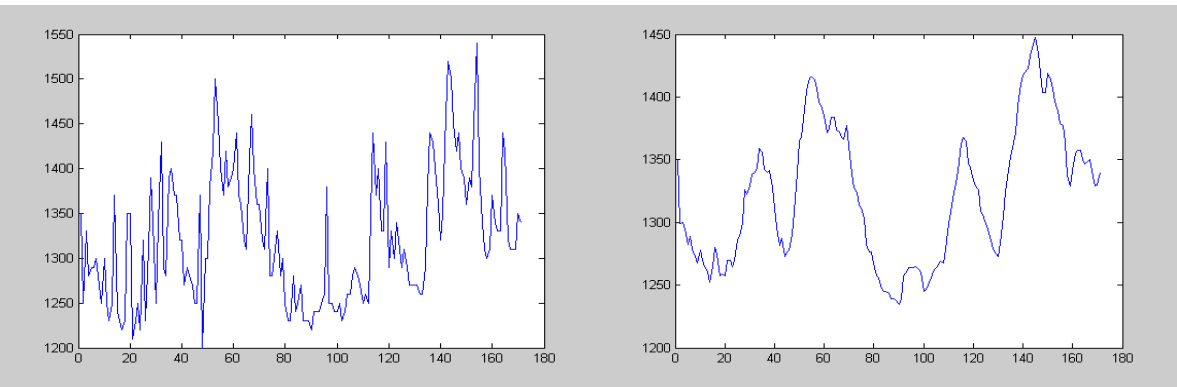
Time Series 29



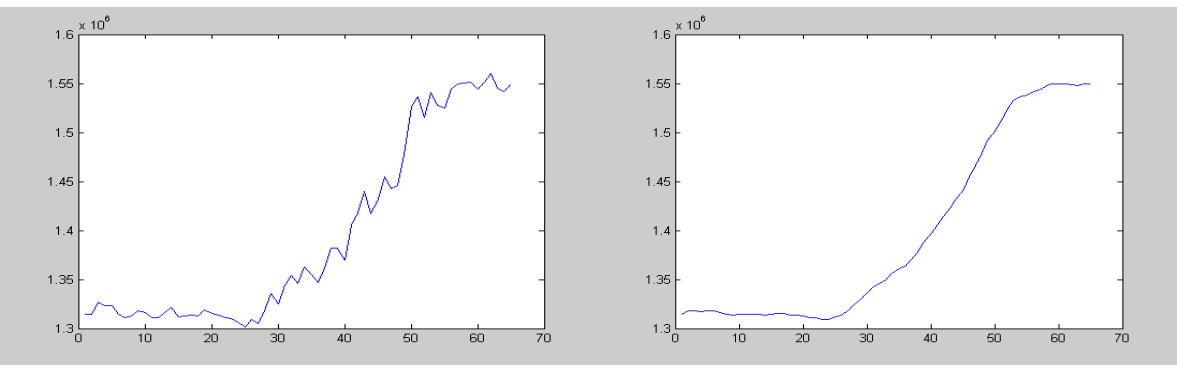
Time Series 30



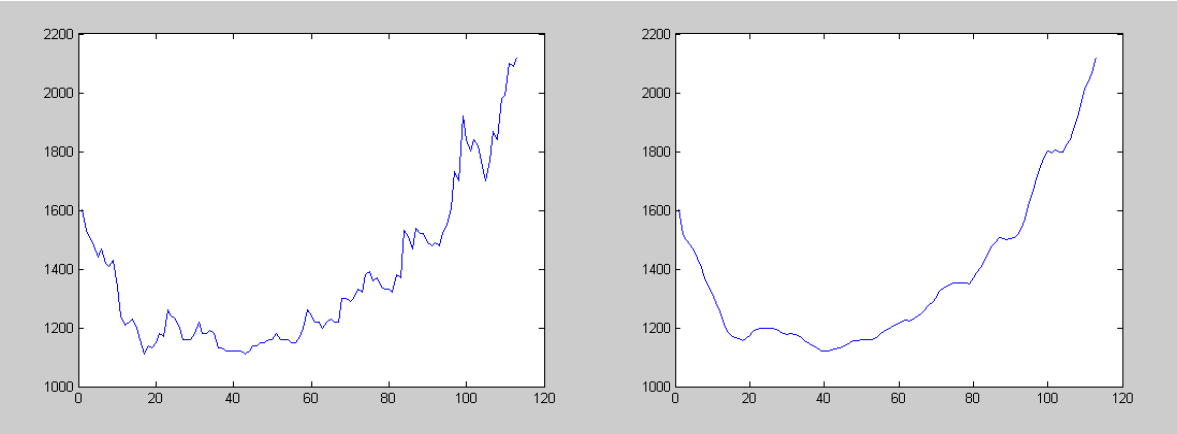
Time Series 31



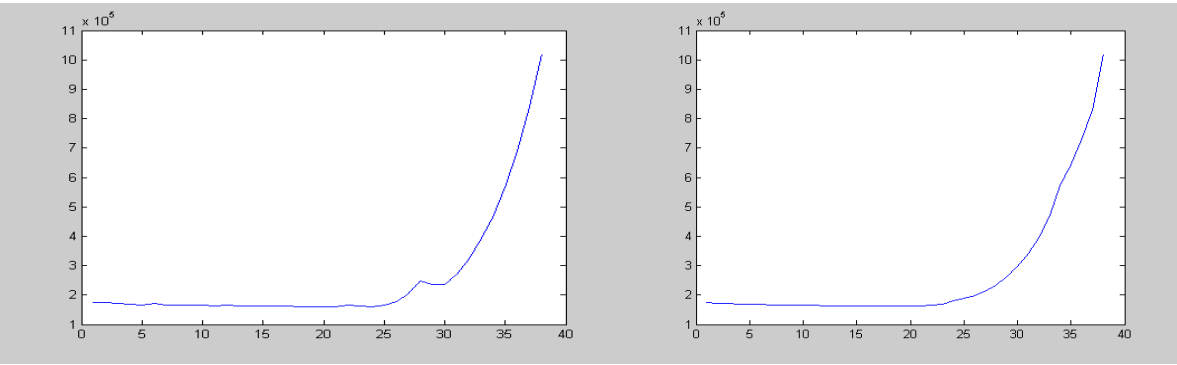
Time Series 32



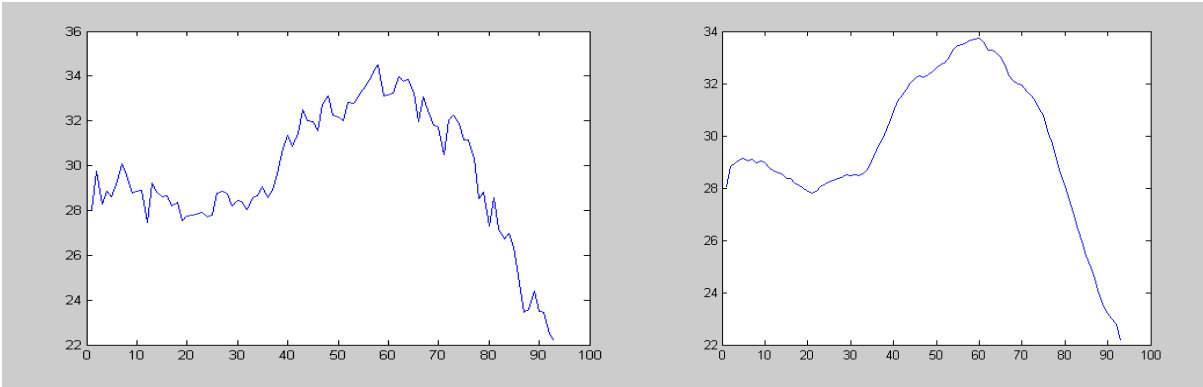
Time Series 33



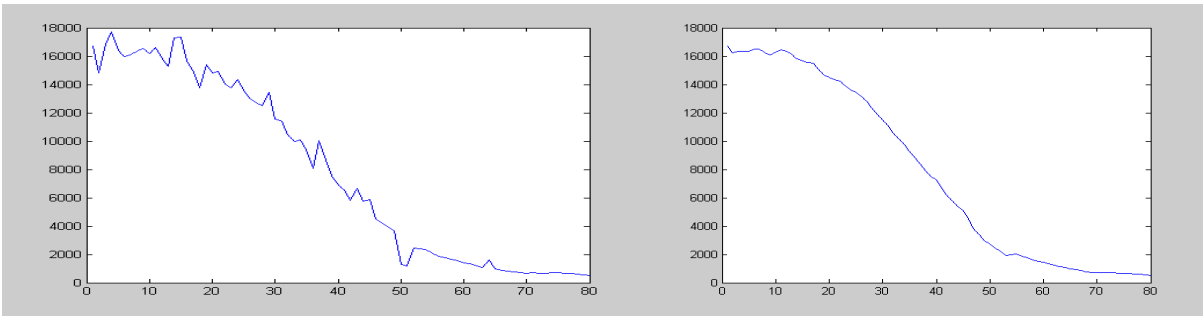
Time Series 34



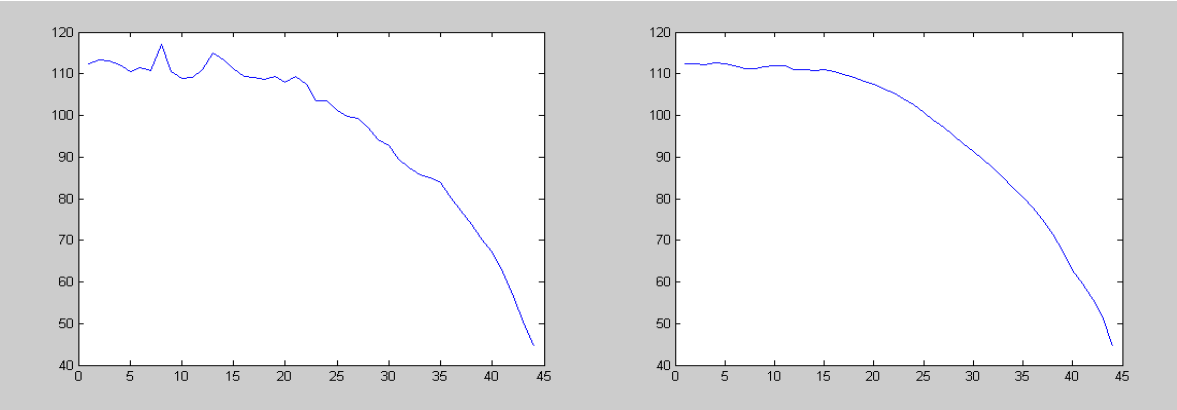
Time Series 35



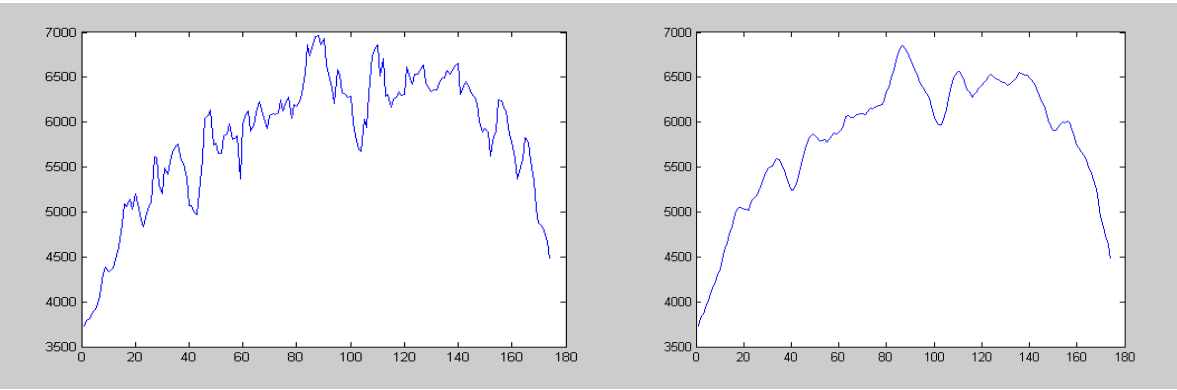
Time Series 36



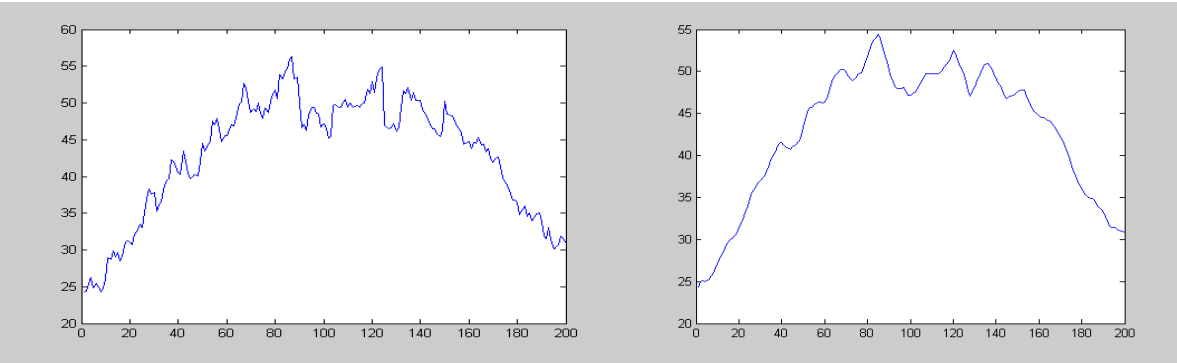
Time Series 37



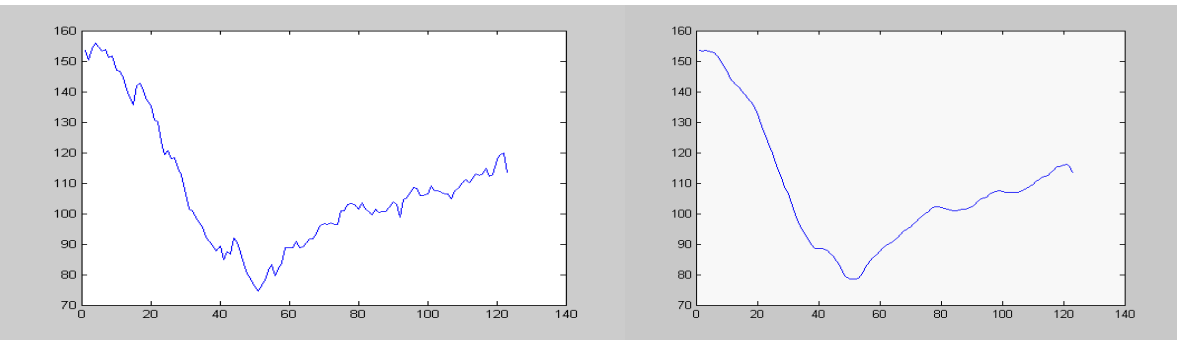
Time Series 38



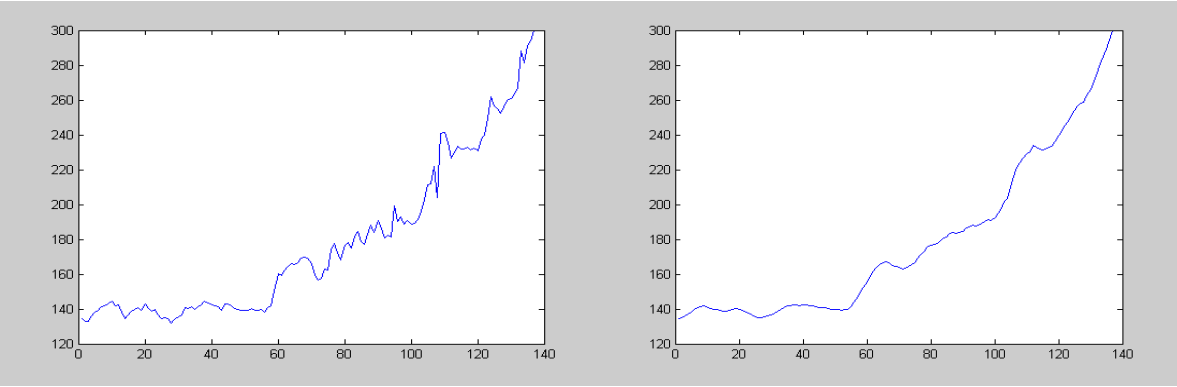
Time Series 39



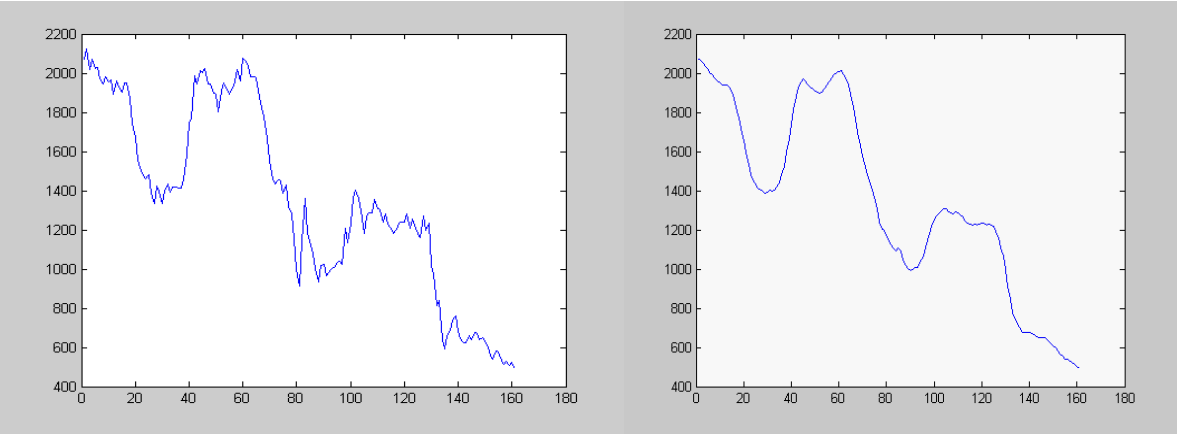
Time Series 40



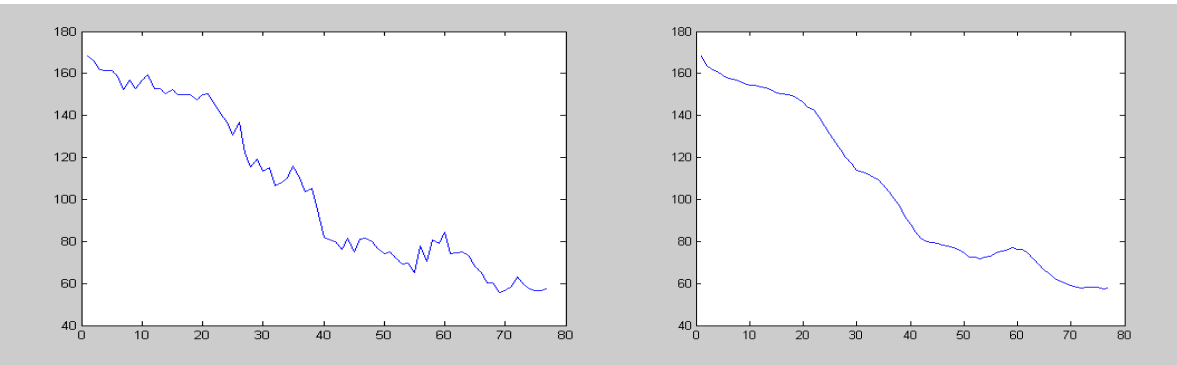
Time Series 41



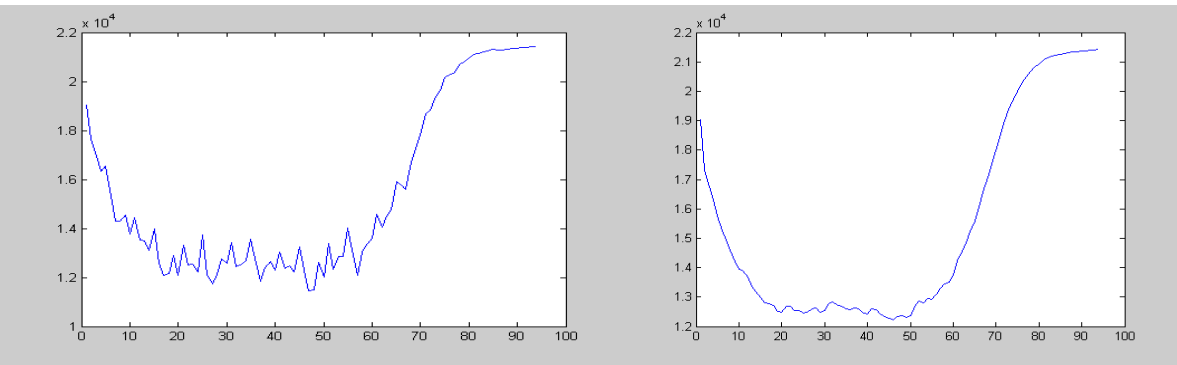
Time Series 42



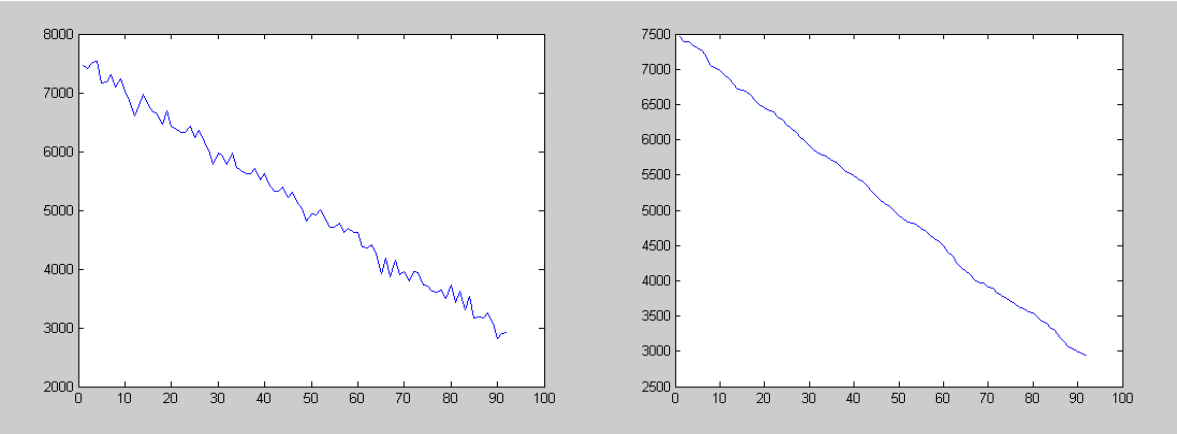
Time Series 43



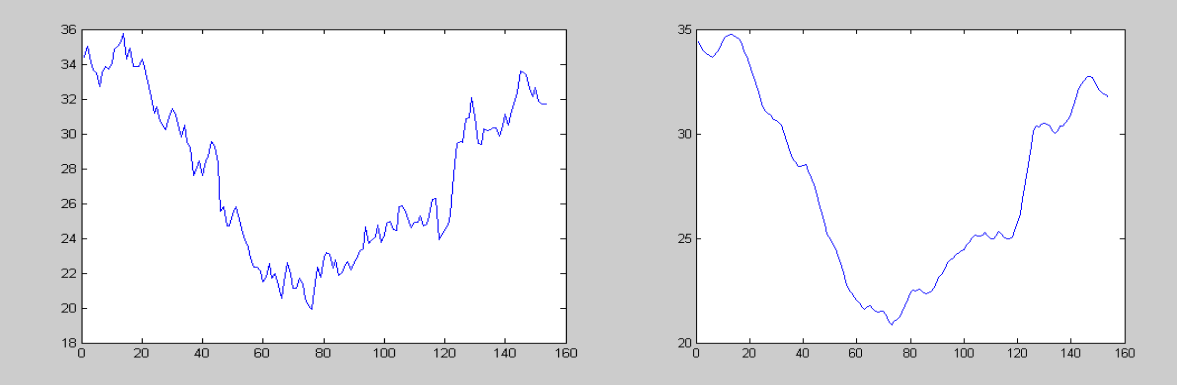
Time Series 44



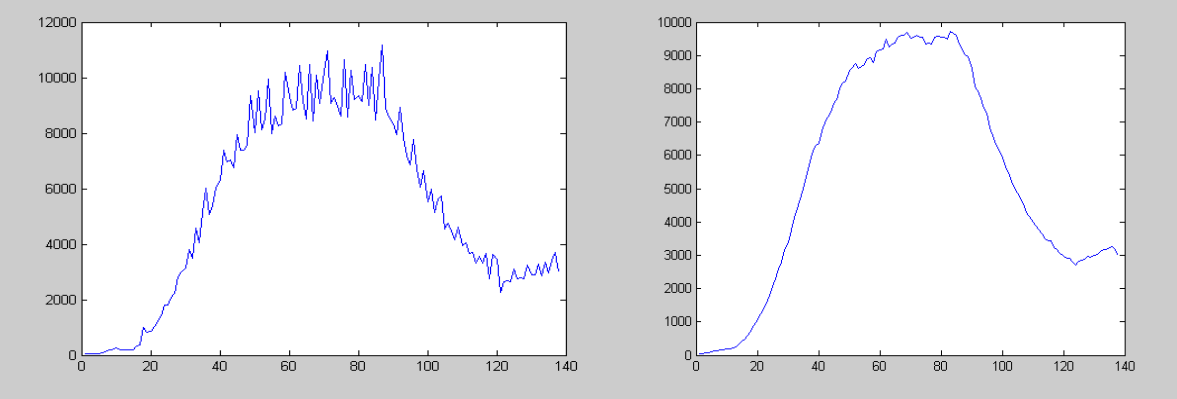
Time Series 45



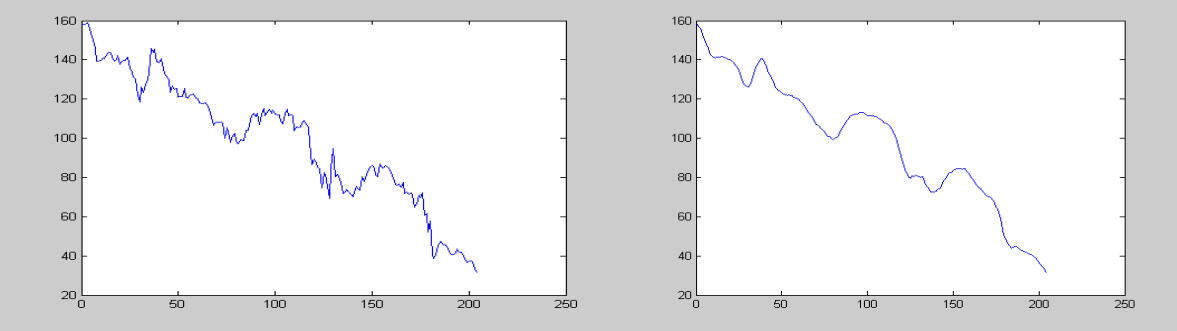
Time Series 46



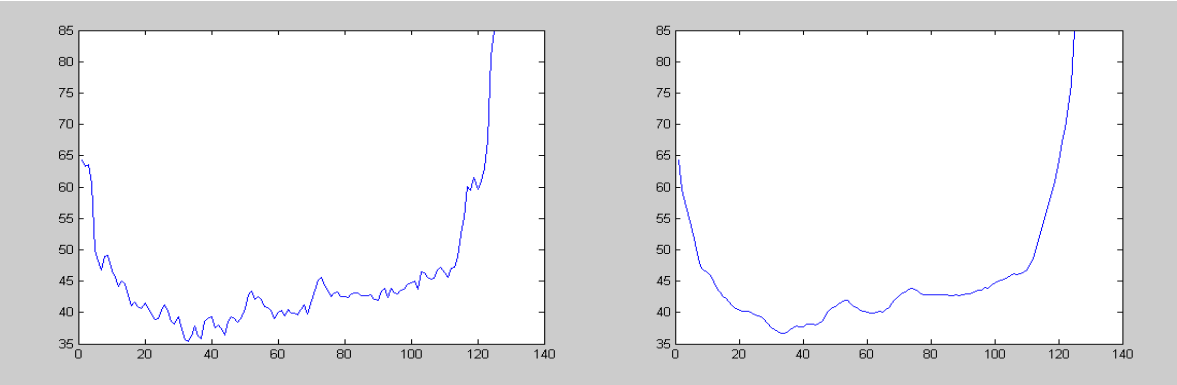
Time Series 47



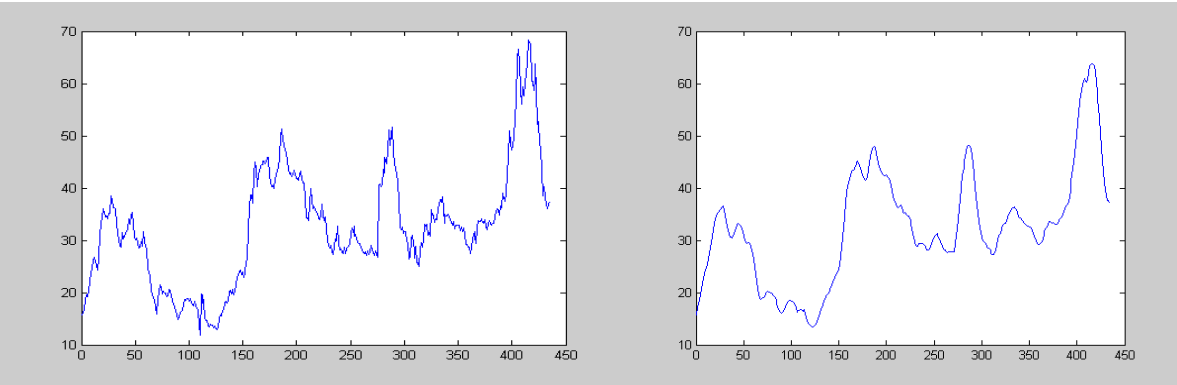
Time Series 48



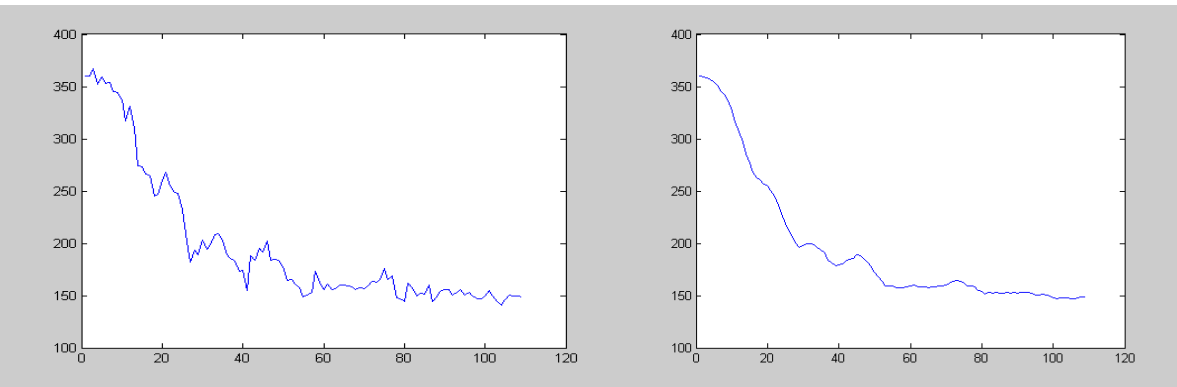
Time Series 49



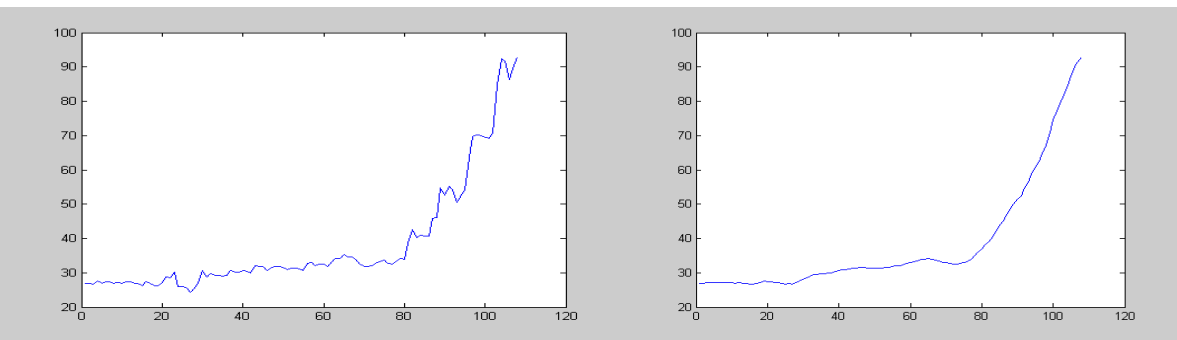
Time Series 50



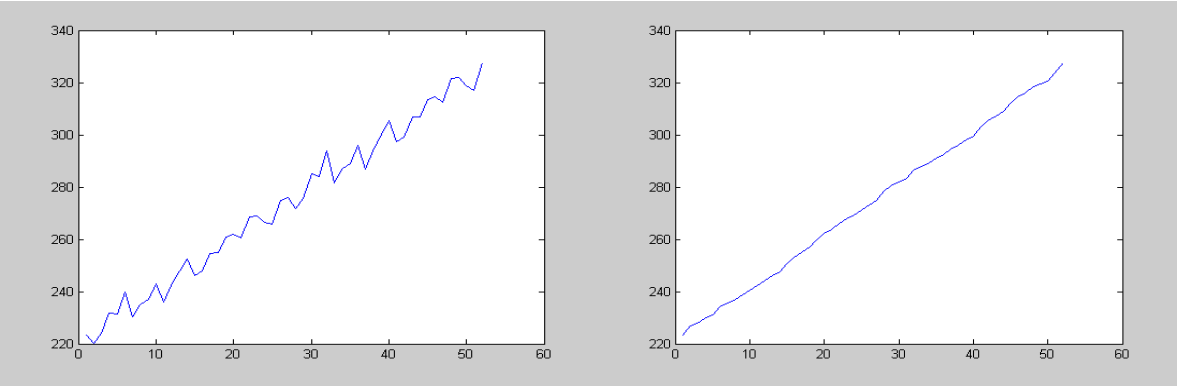
Time Series 51



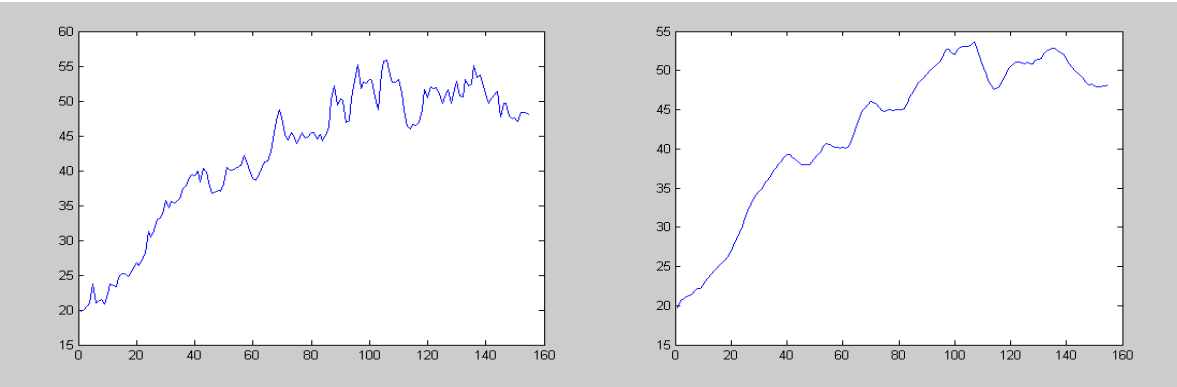
Time Series 52



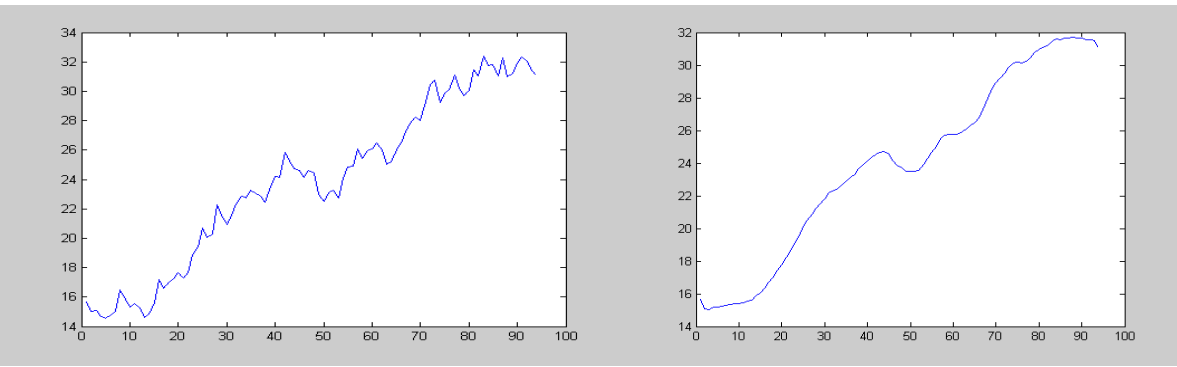
Time Series 53



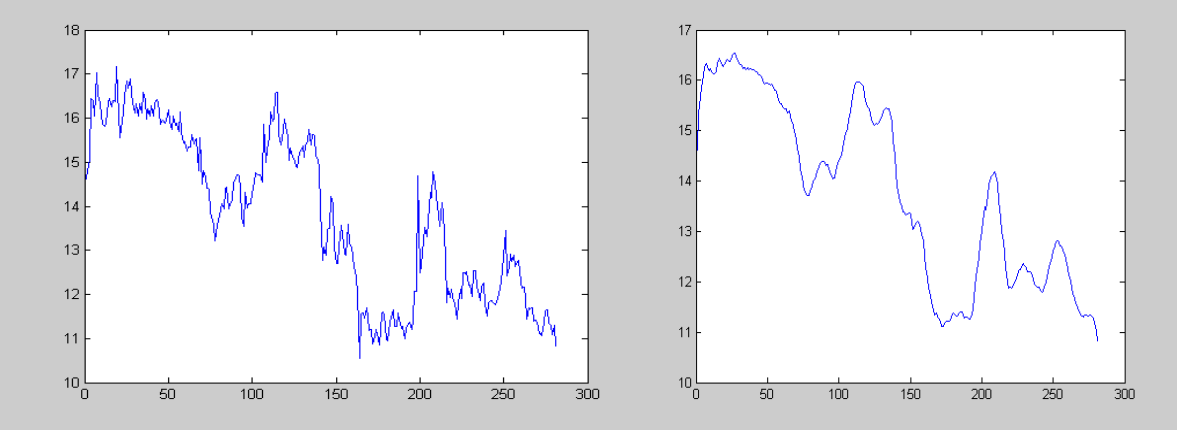
Time Series 54



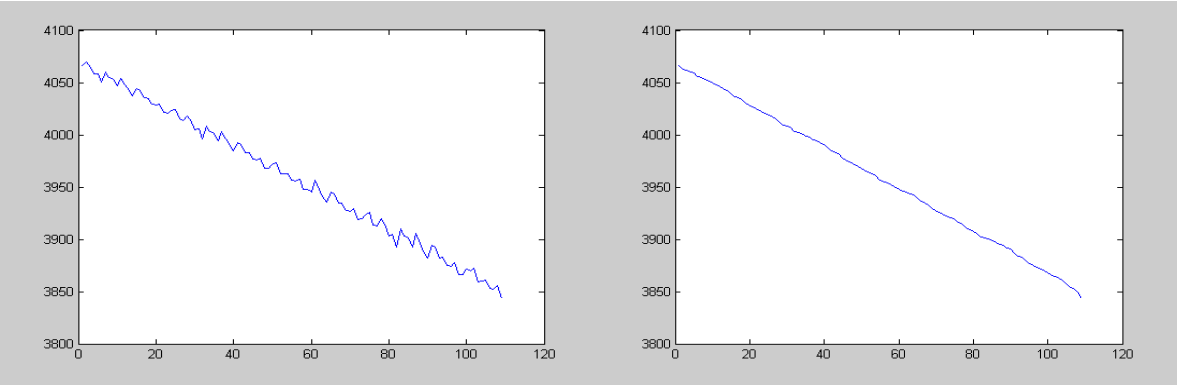
Time Series 55



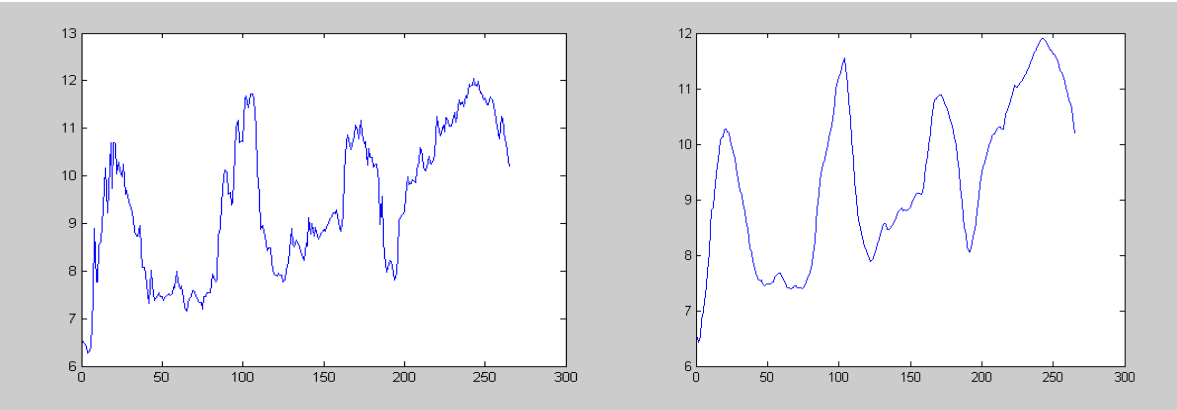
Time Series 56



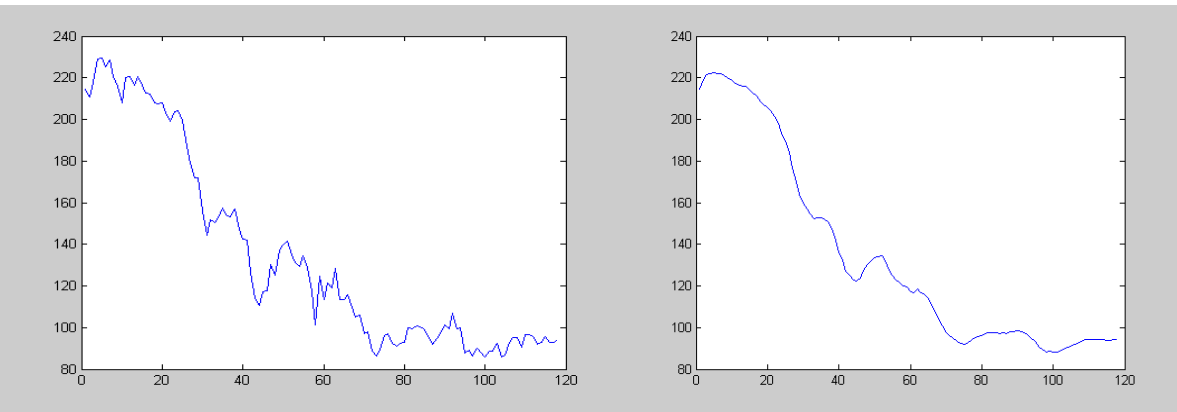
Time Series 57



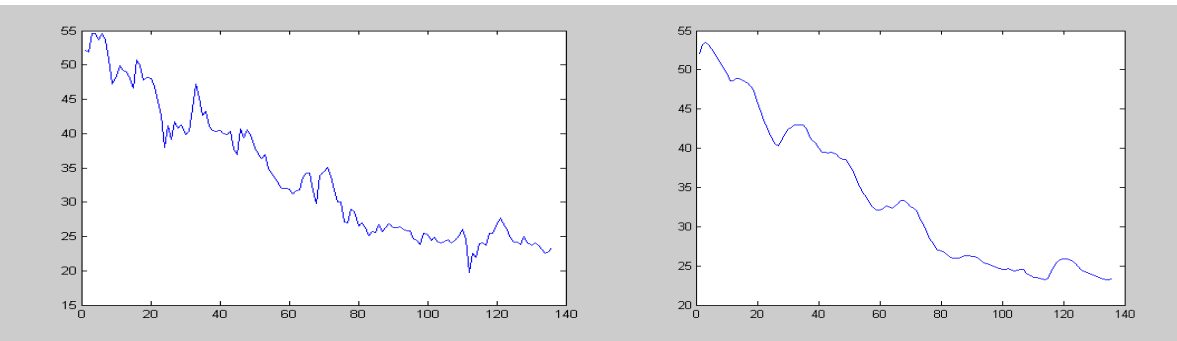
Time Series 58



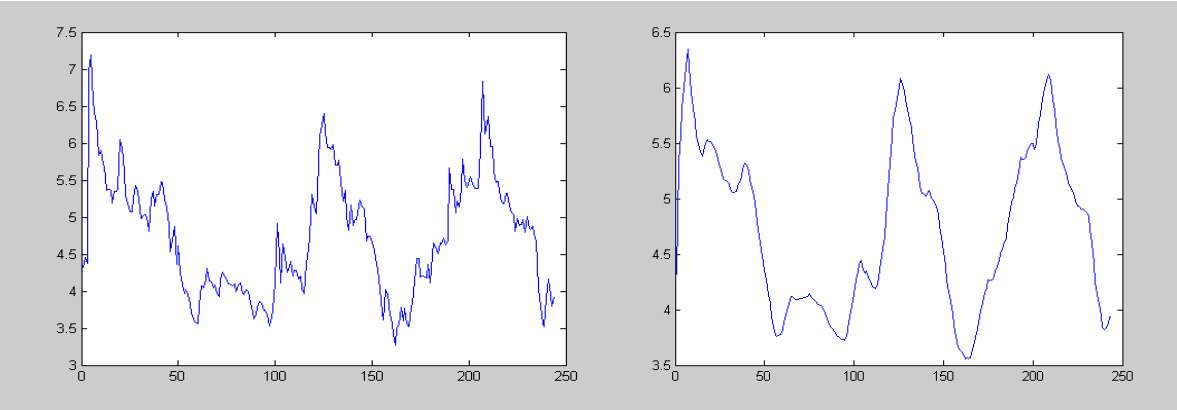
Time Series 59



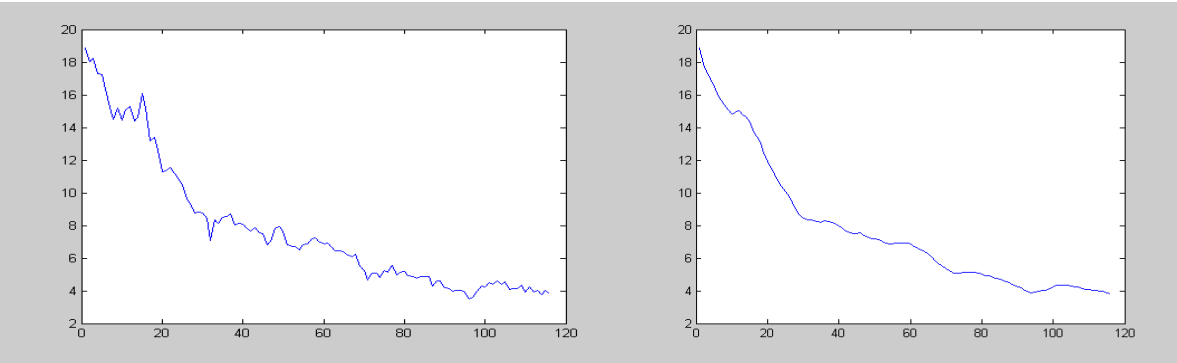
Time Series 60



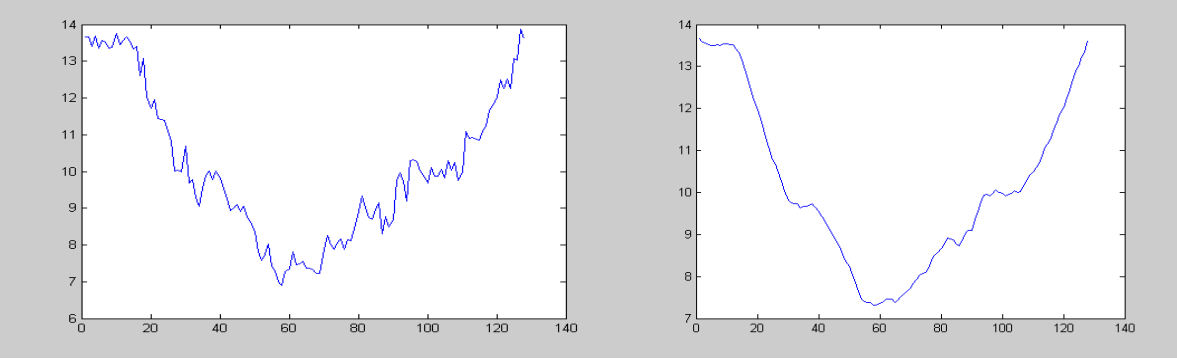
Time Series 61



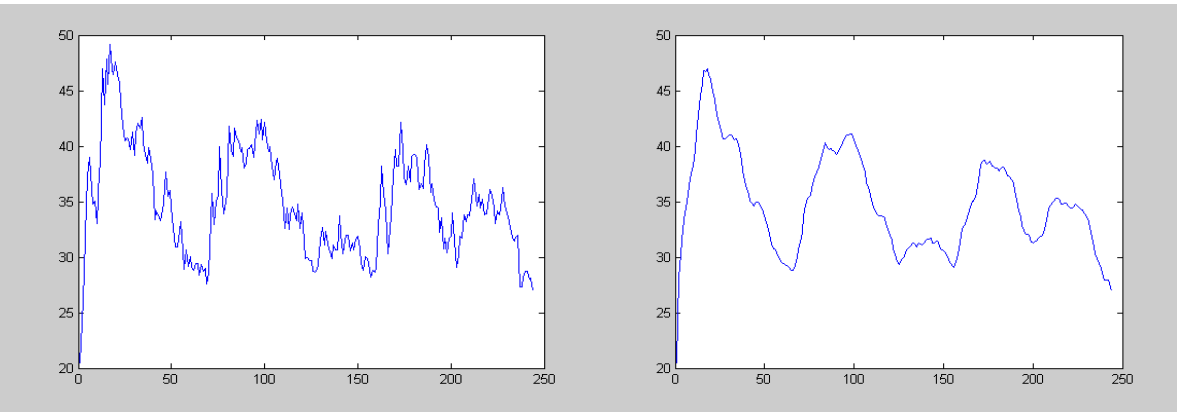
Time Series 62



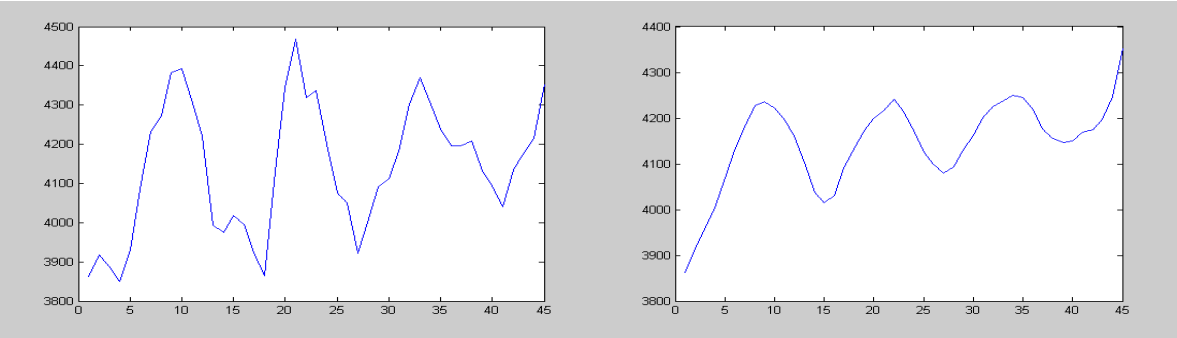
Time Series 63



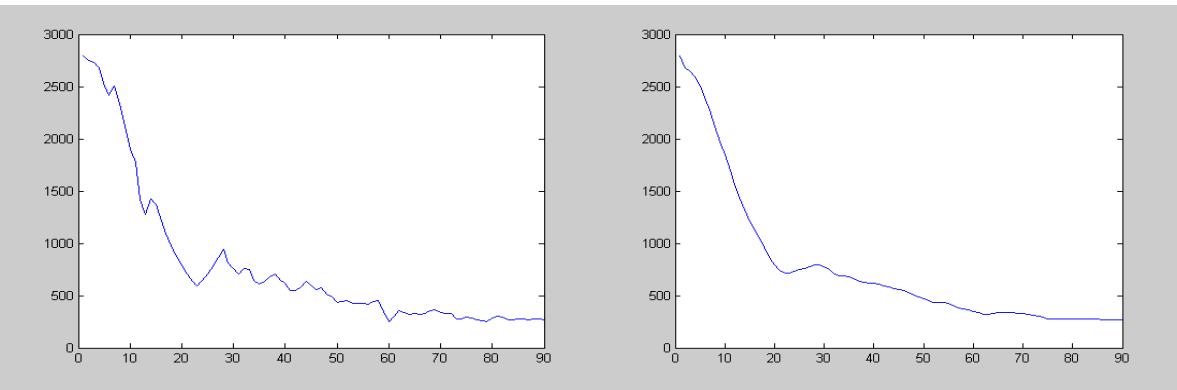
Time Series 64



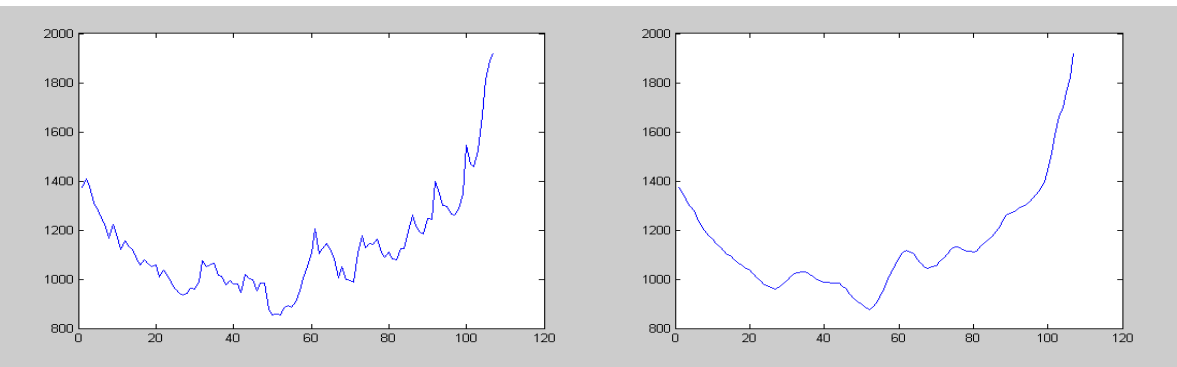
Time Series 65



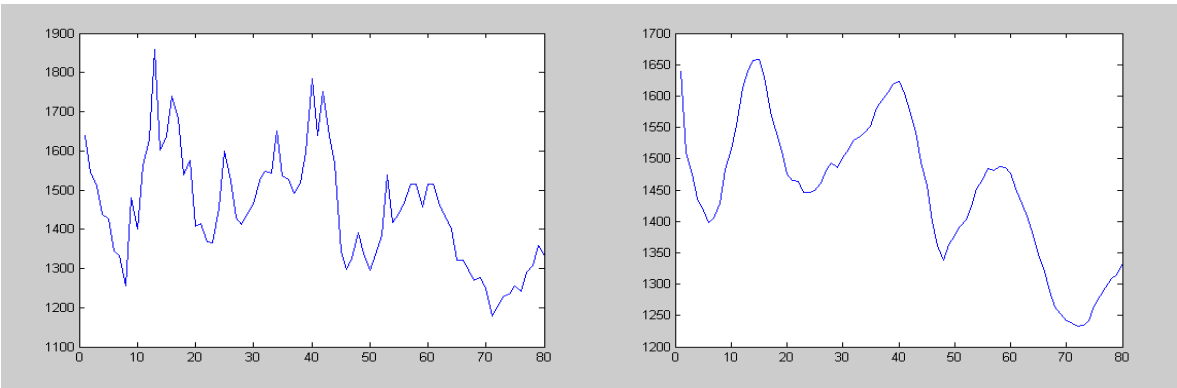
Time Series 66



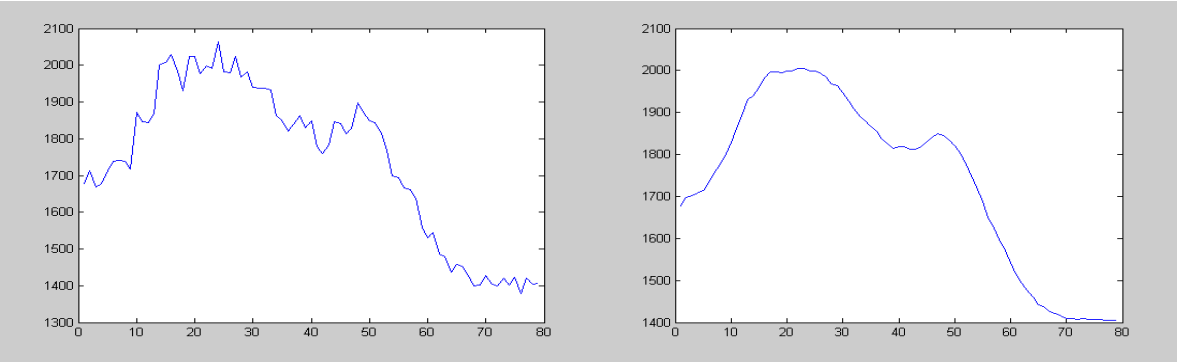
Time Series 67



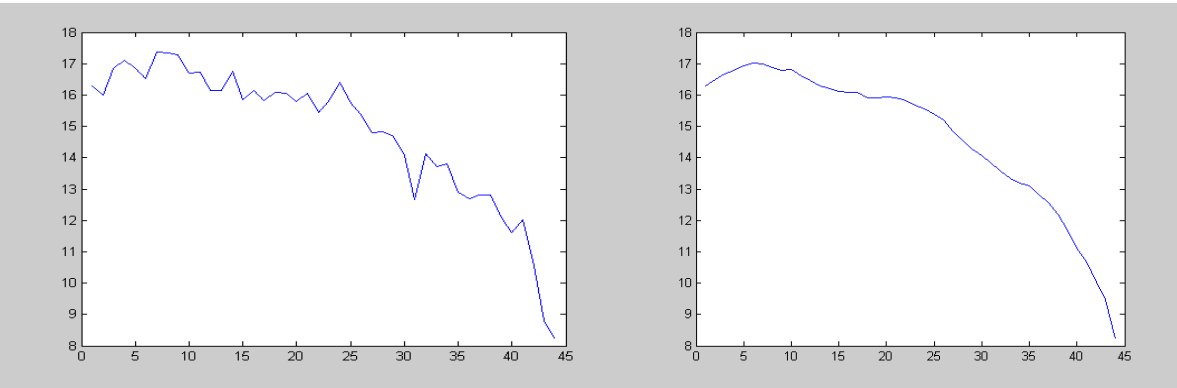
Time Series 68



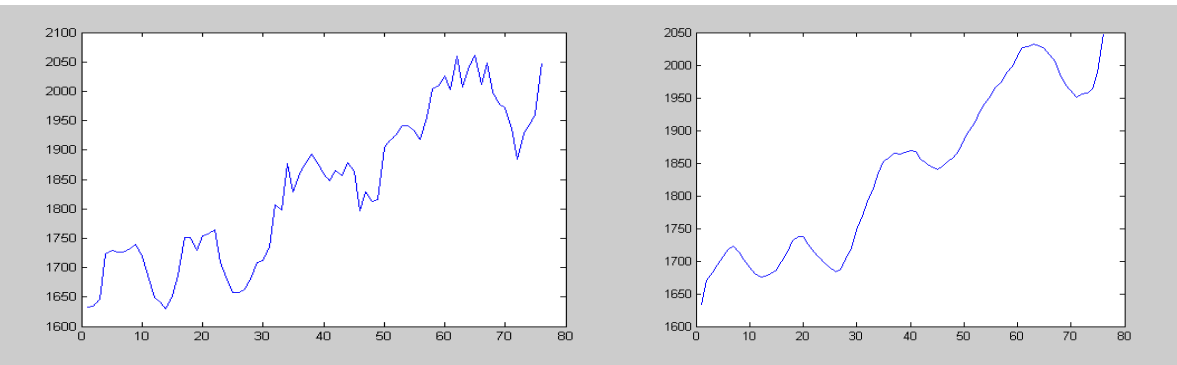
Time Series 69



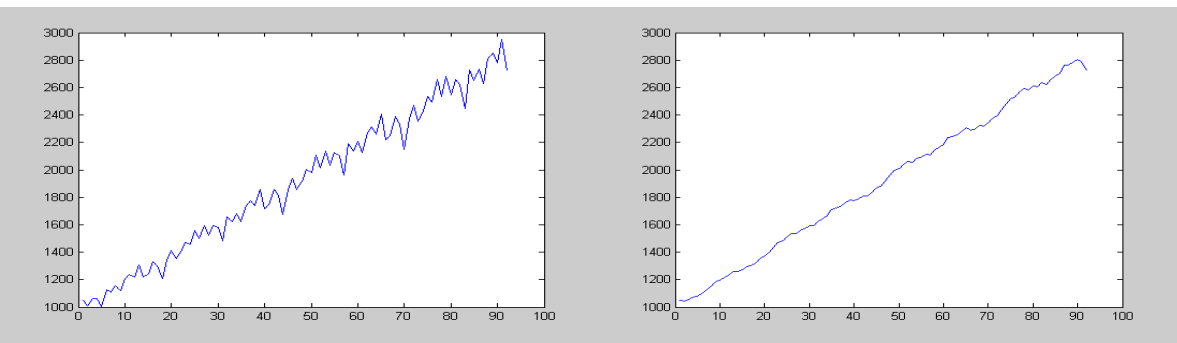
Time Series 70



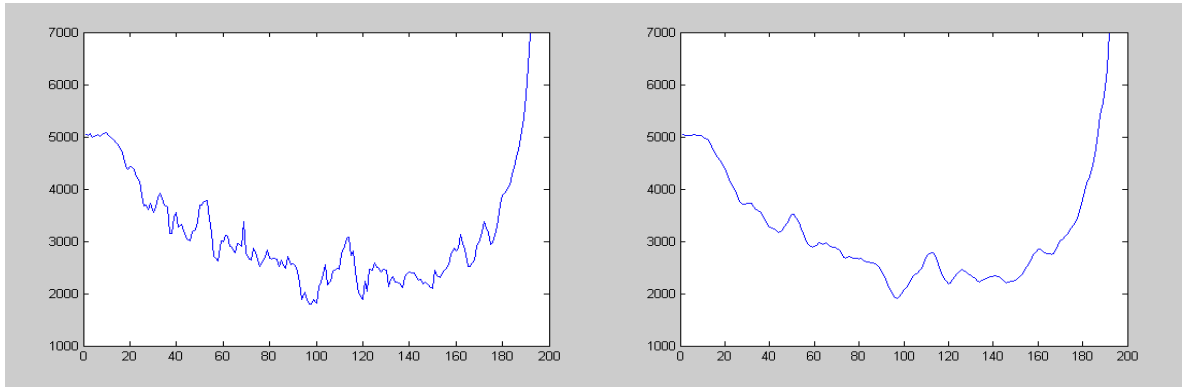
Time Series 71



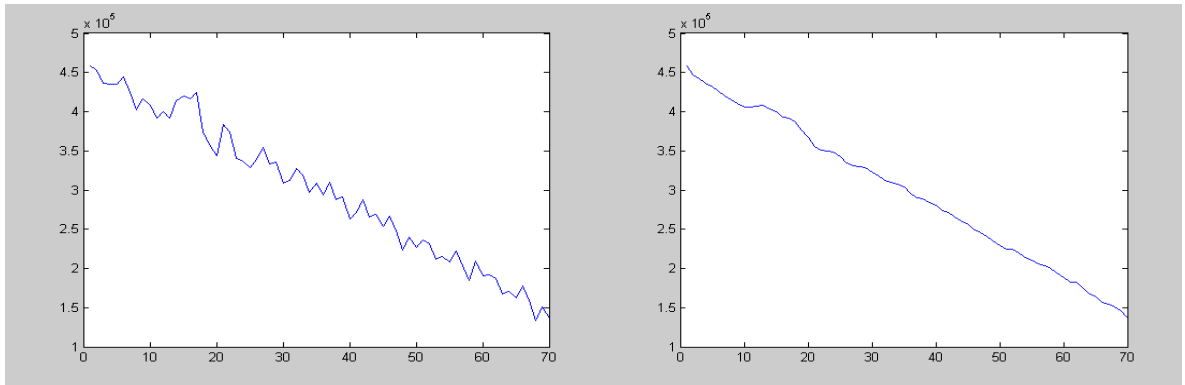
Time Series 72



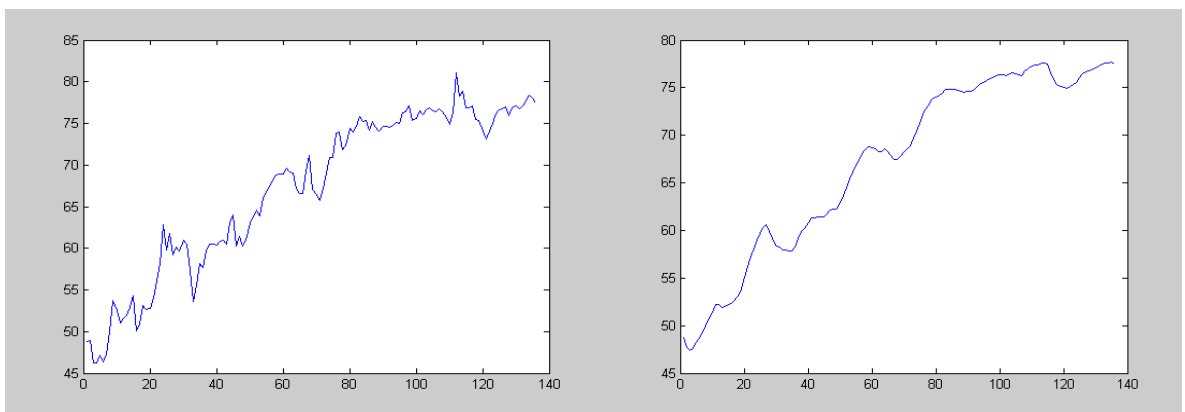
Time Series 73



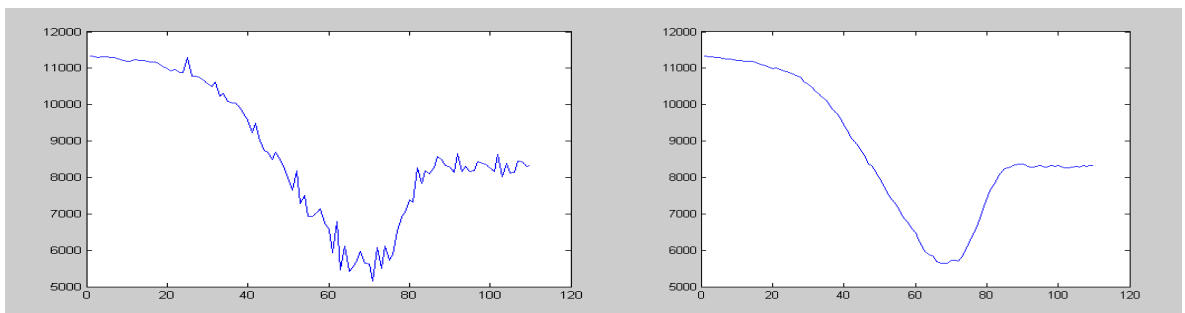
Time Series 74



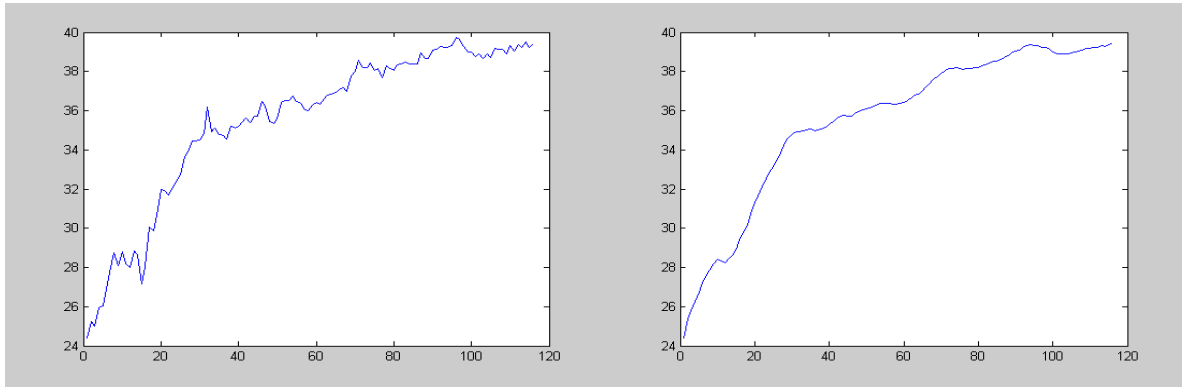
Time Series 75



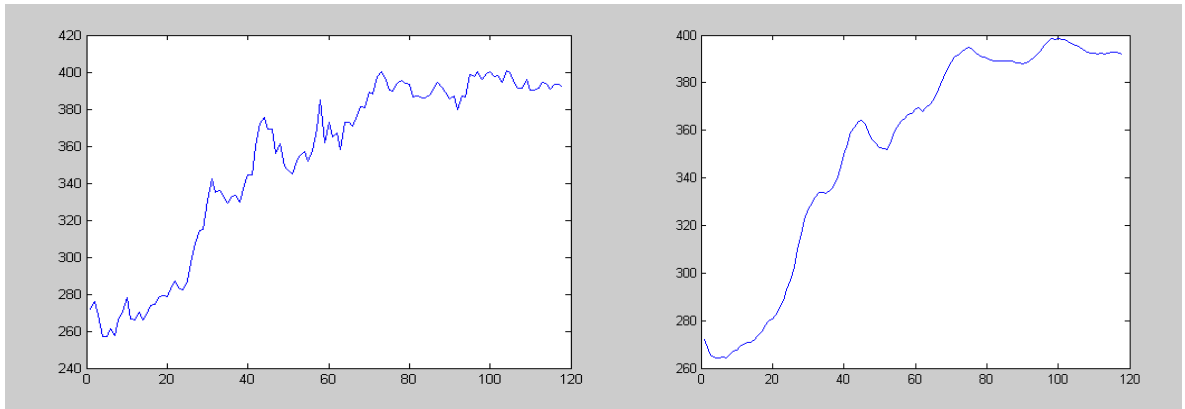
Time Series 76



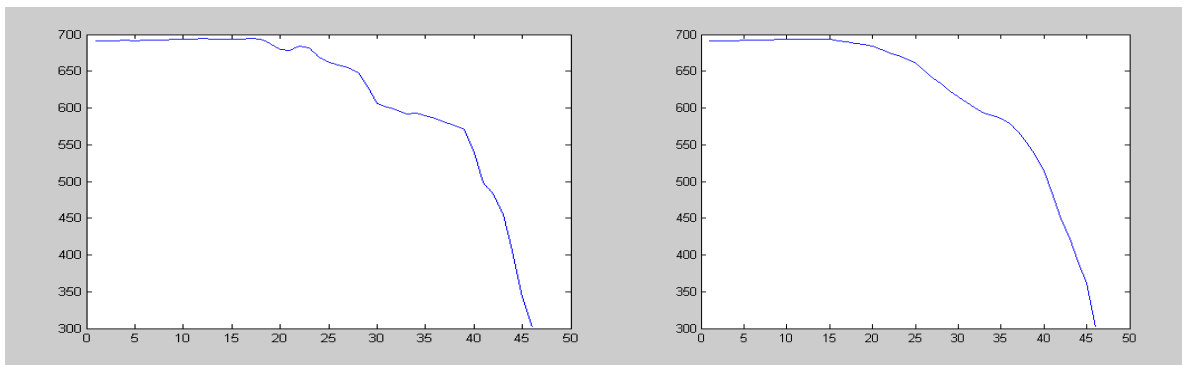
Time Series 77



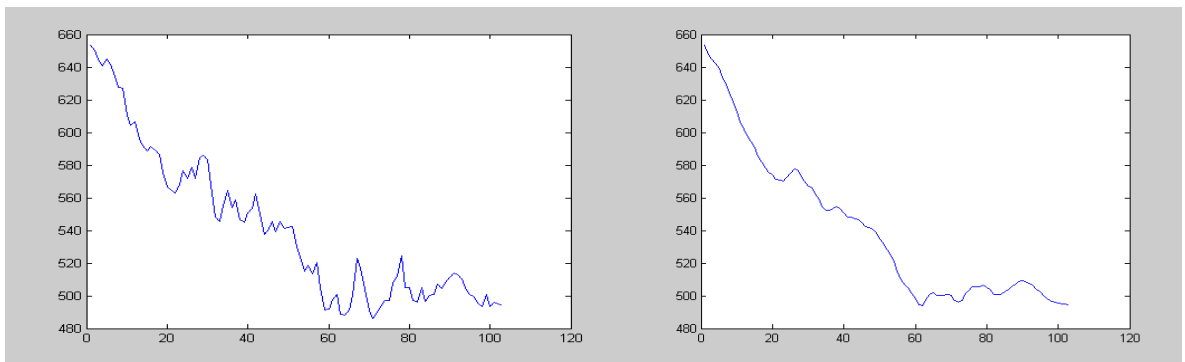
Time Series 78



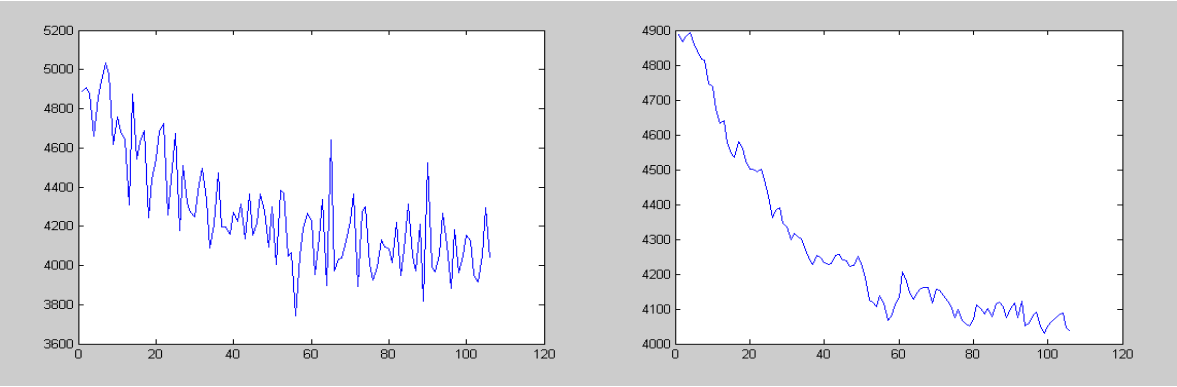
Time Series 79



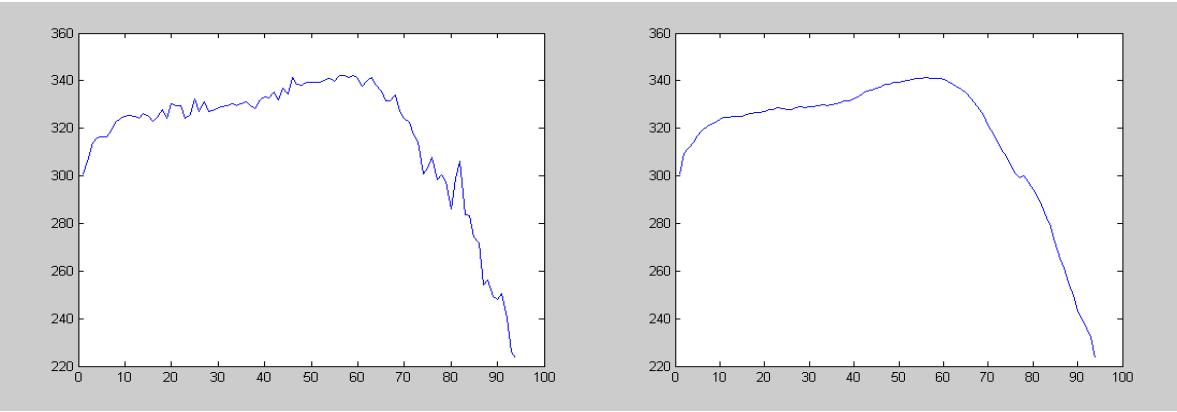
Time Series 80



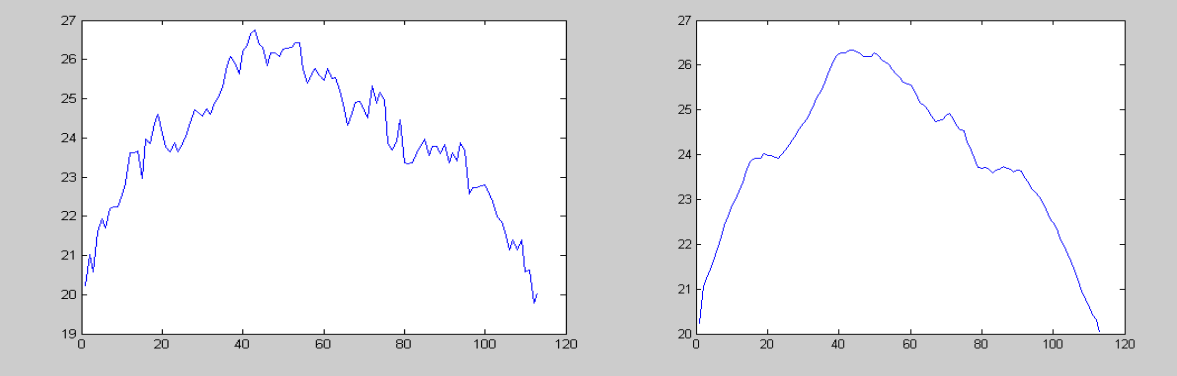
Time Series 81



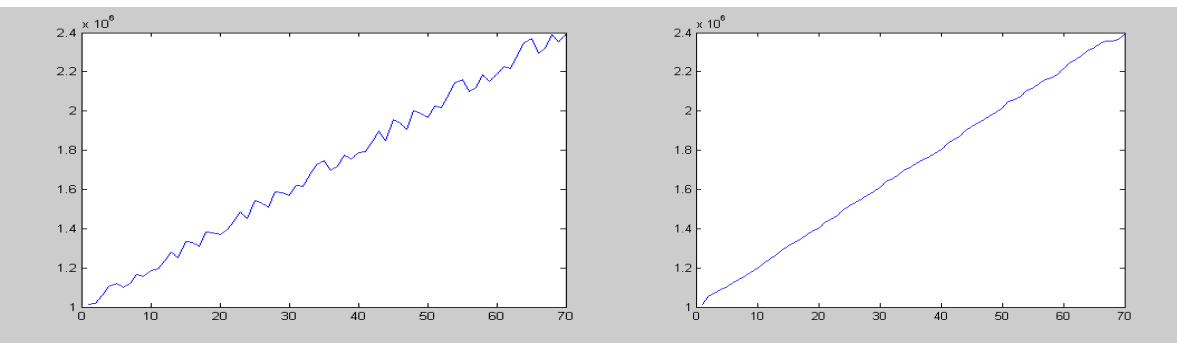
Time Series 82



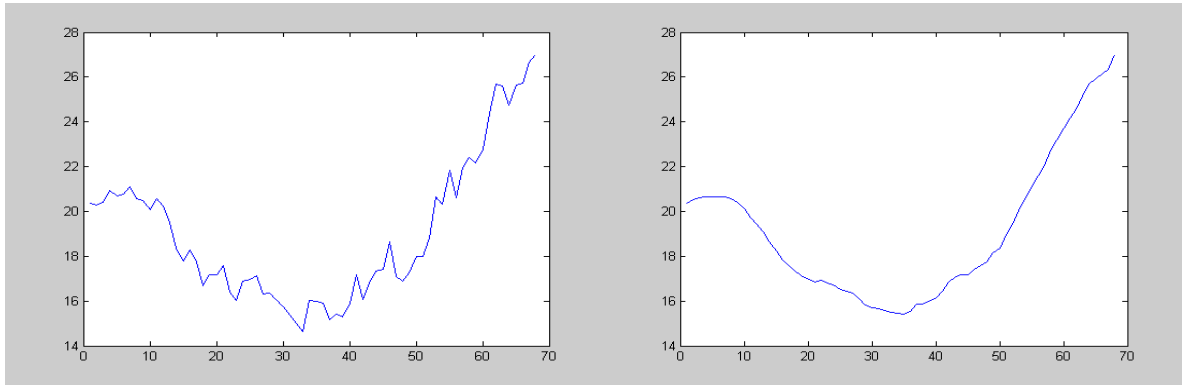
Time Series 83



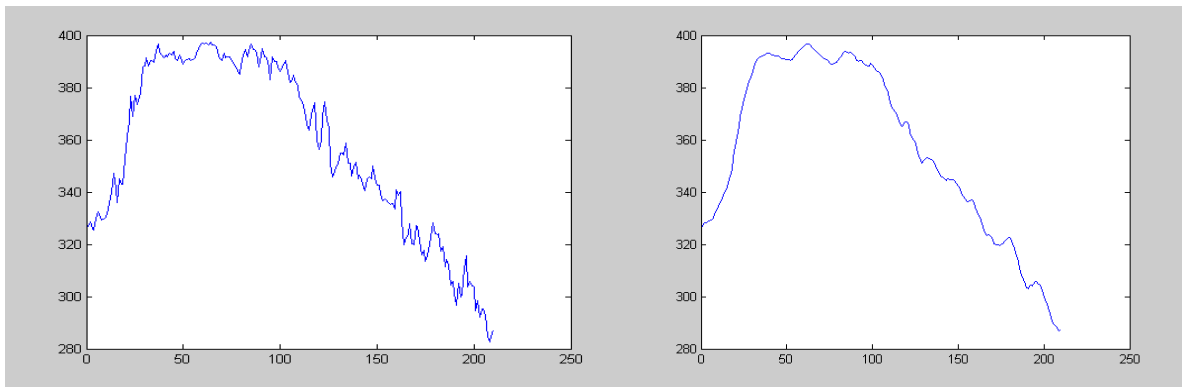
Time Series 84



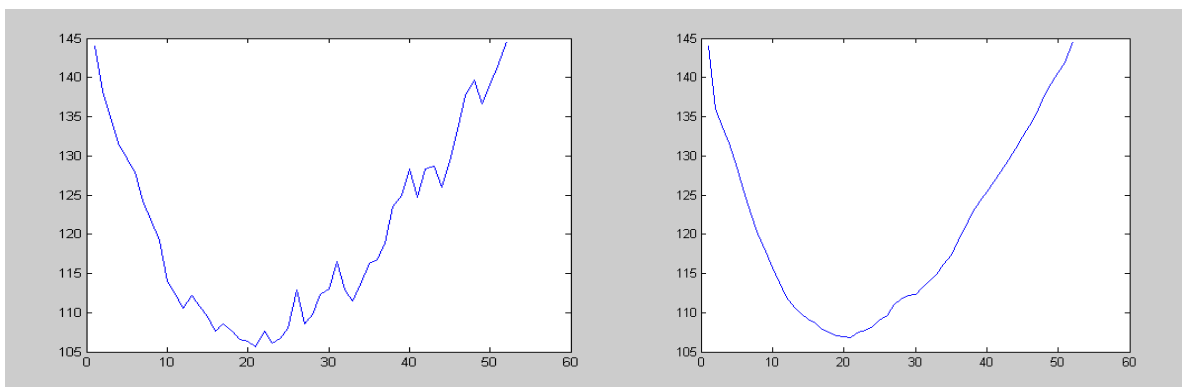
Time Series 85



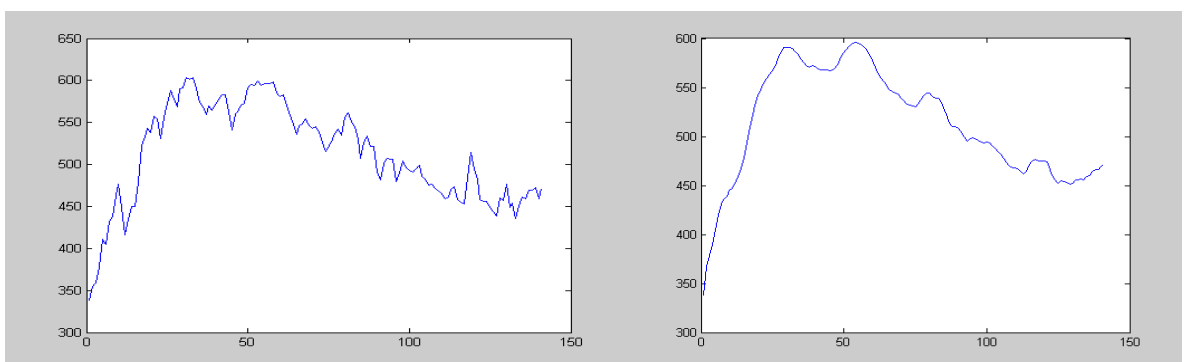
Time Series 86



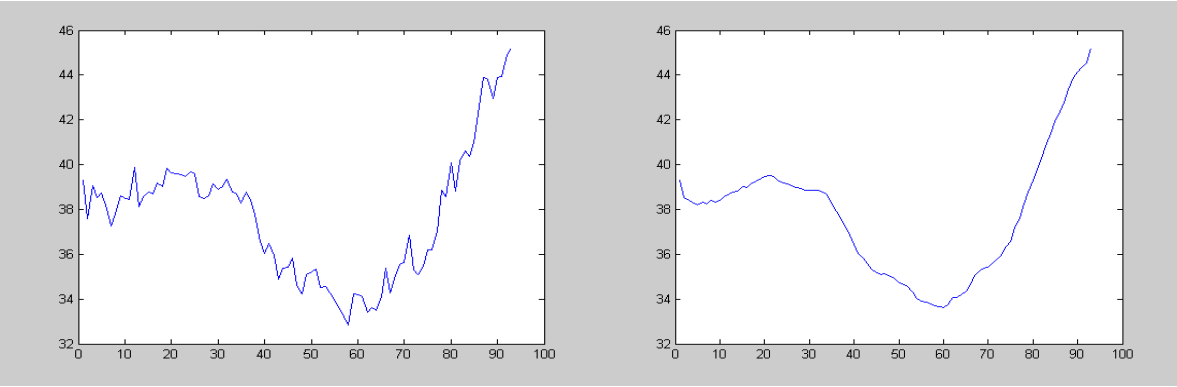
Time Series 87



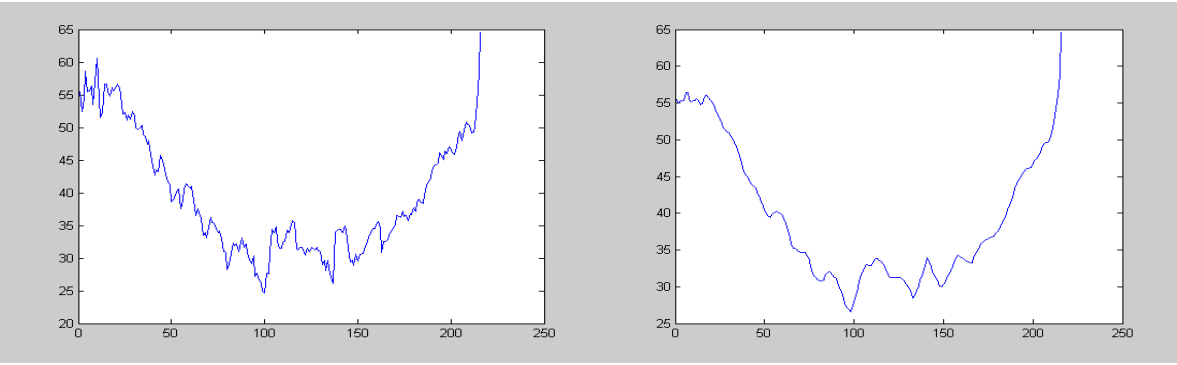
Time Series 88



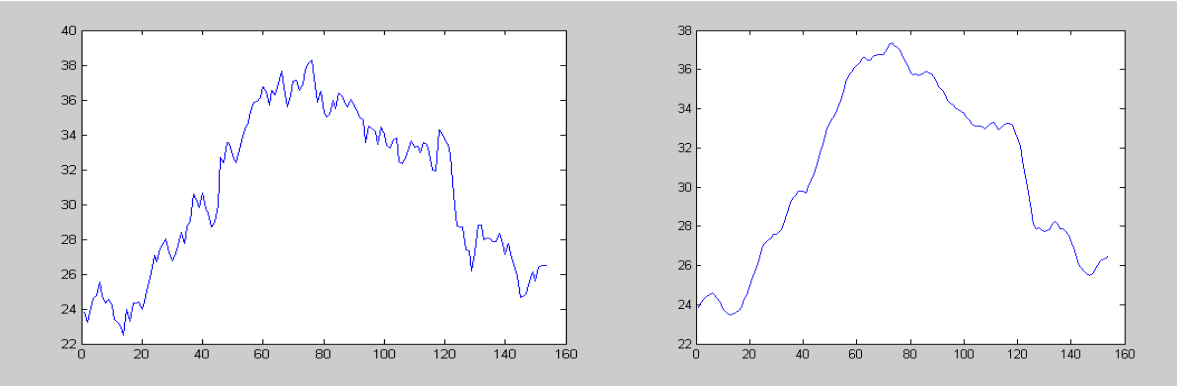
Time Series 89



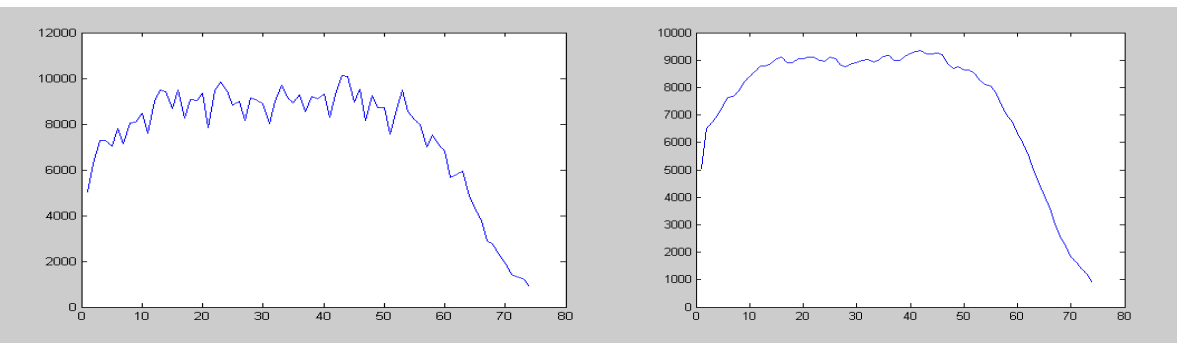
Time Series 90



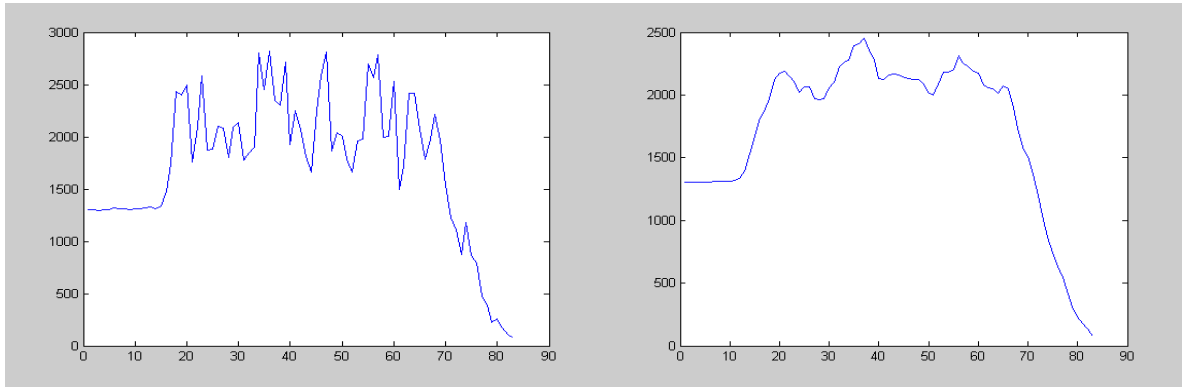
Time Series 91



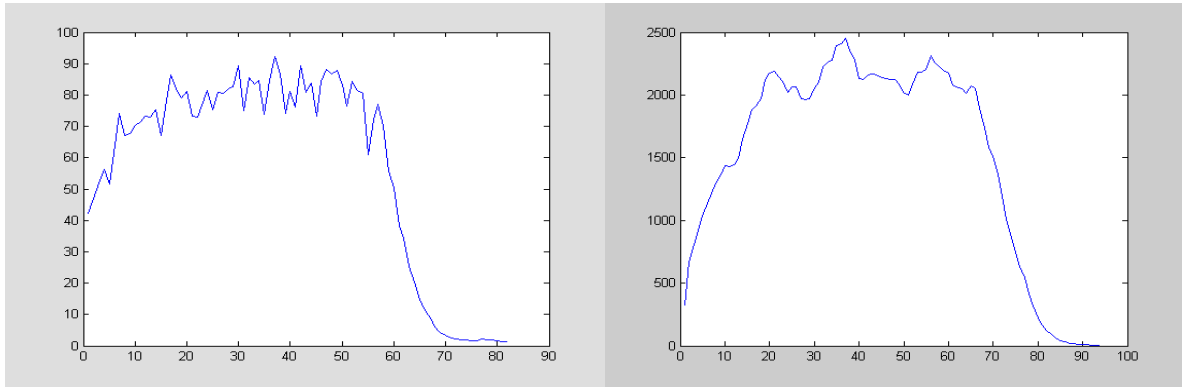
Time Series 92



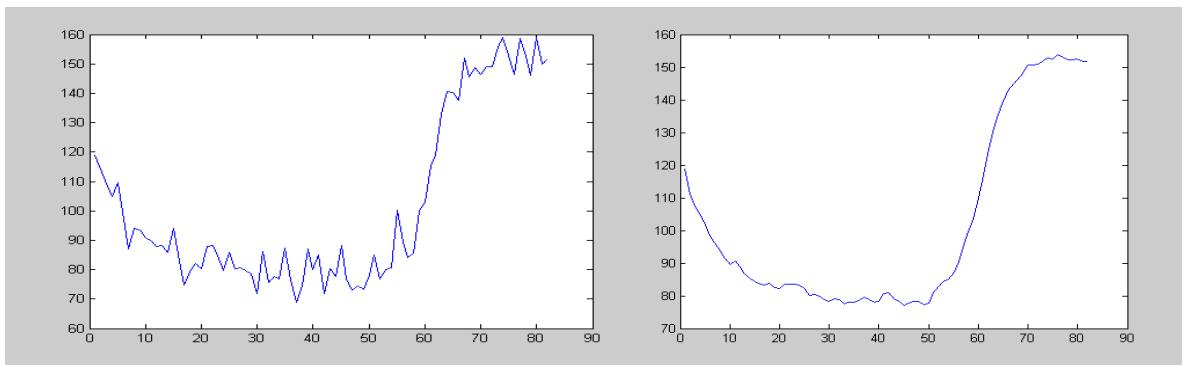
Time Series 93



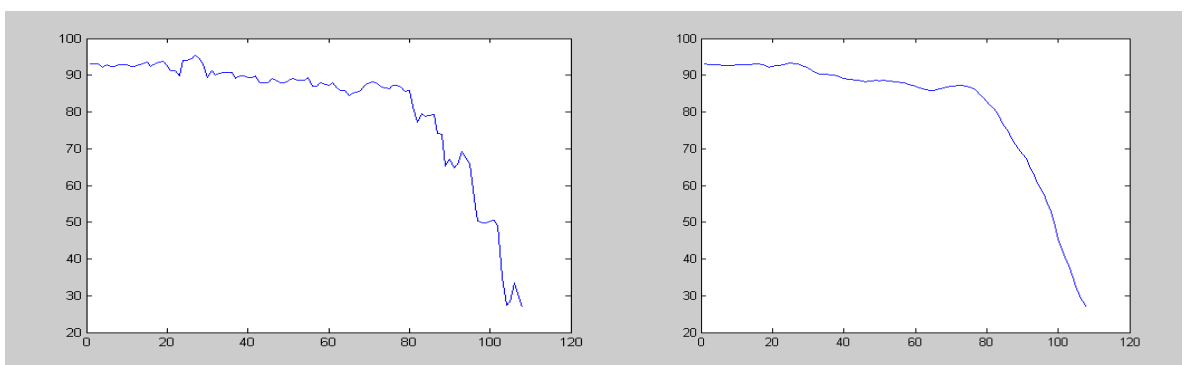
Time Series 94



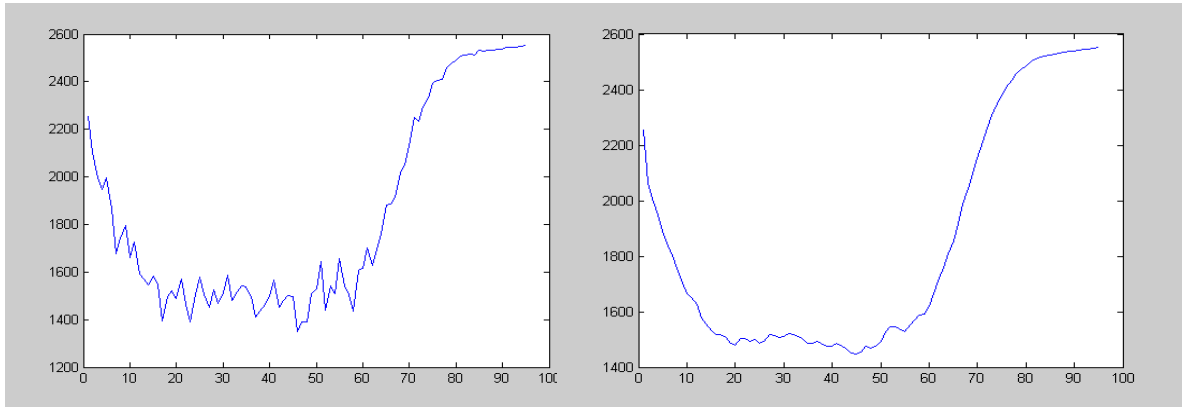
Time Series 95



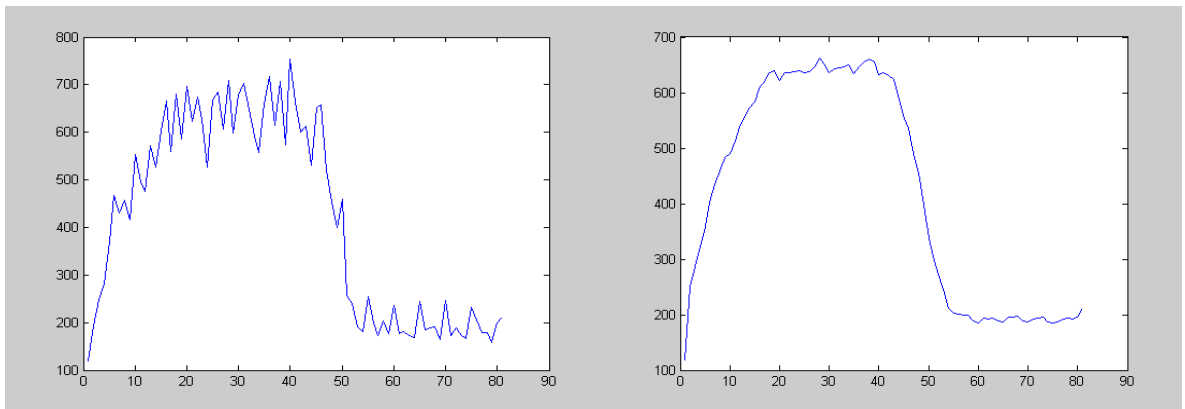
Time Series 96



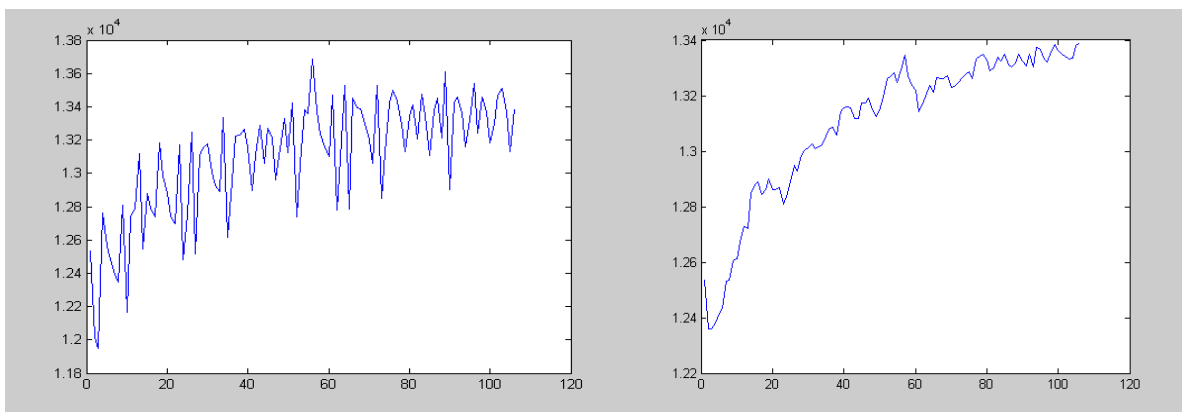
Time Series 97



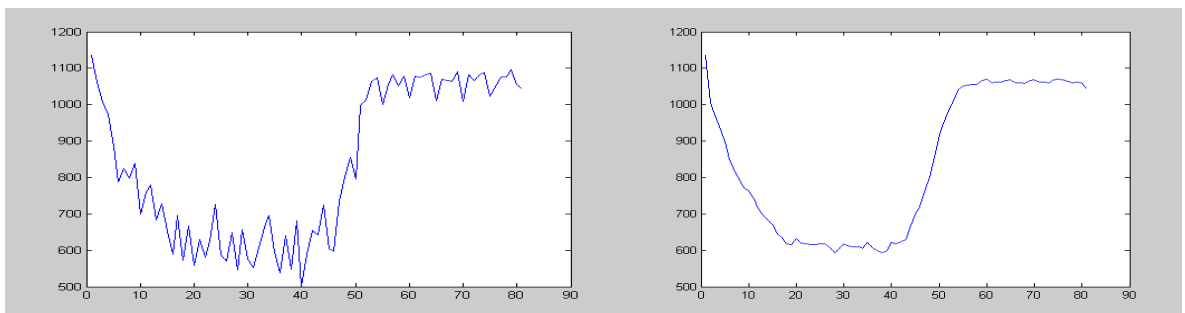
Time Series 98



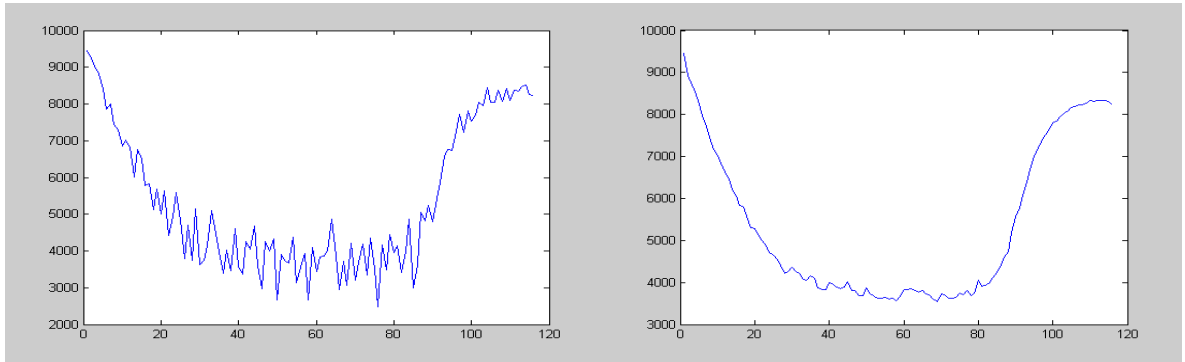
Time Series 99



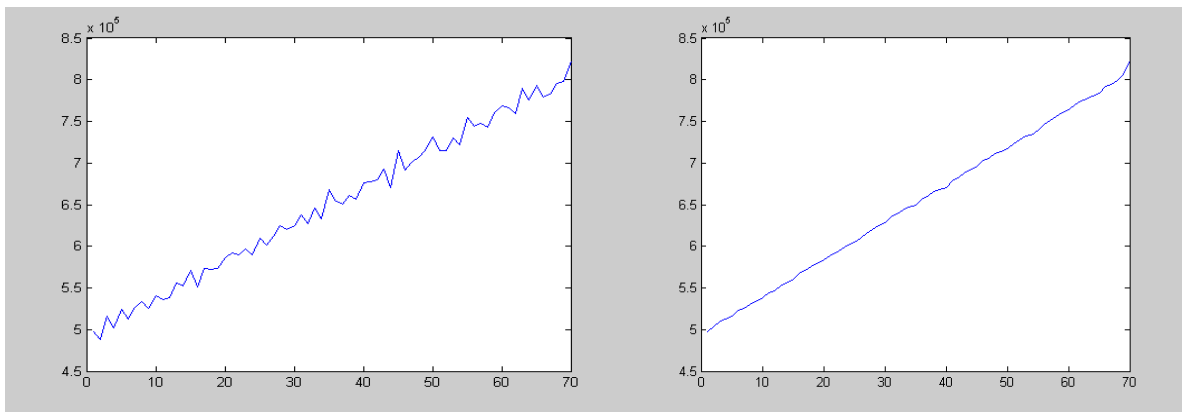
Time Series 100



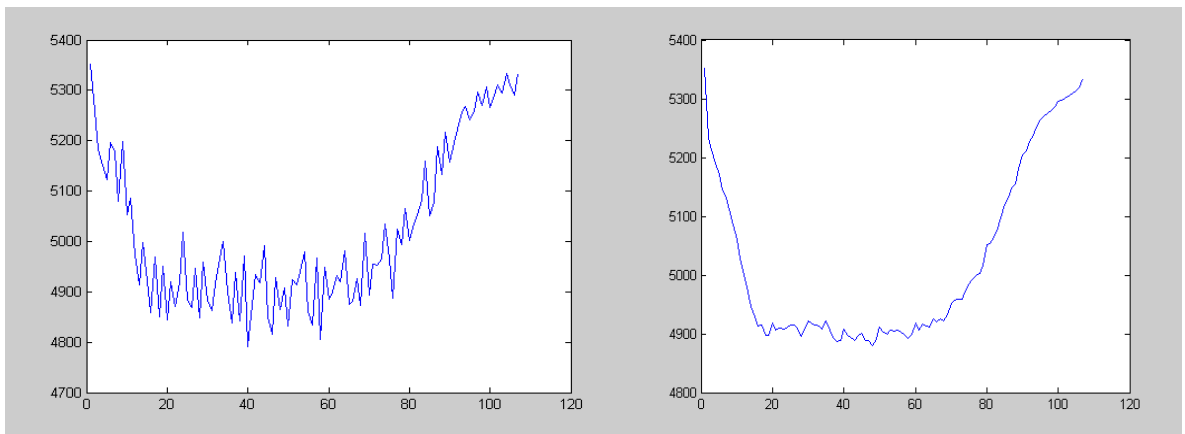
Time Series 101



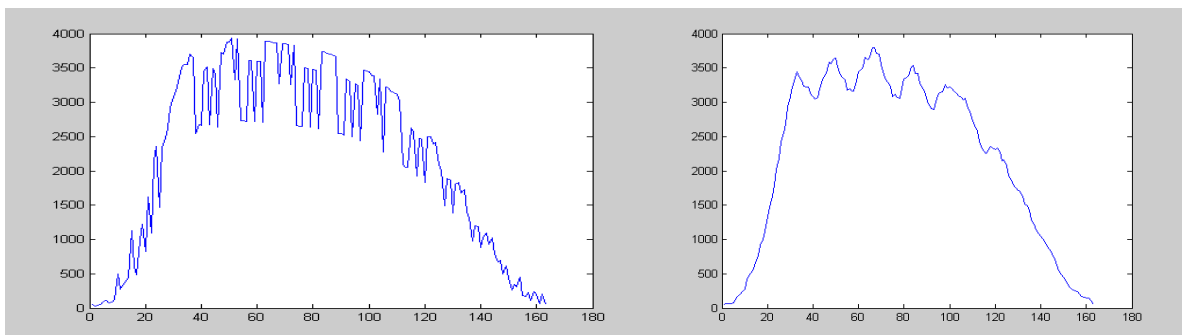
Time Series 102



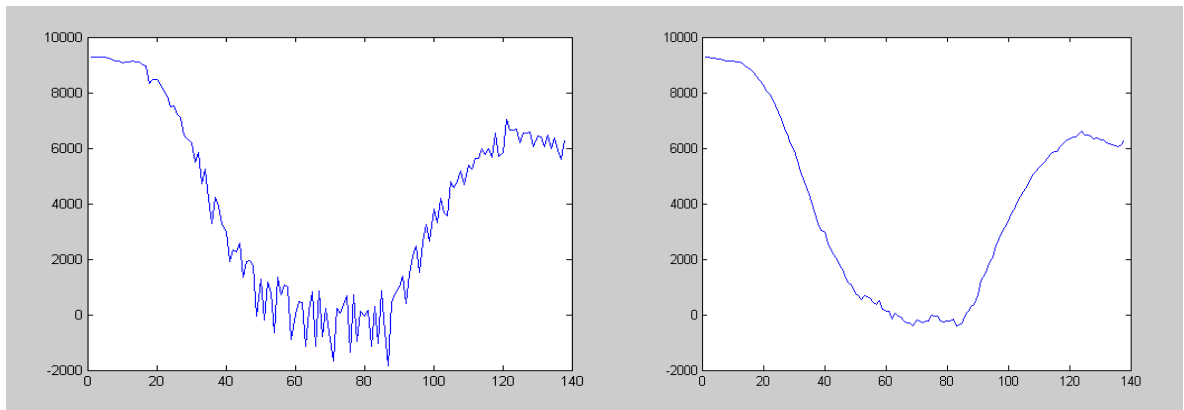
Time Series 103



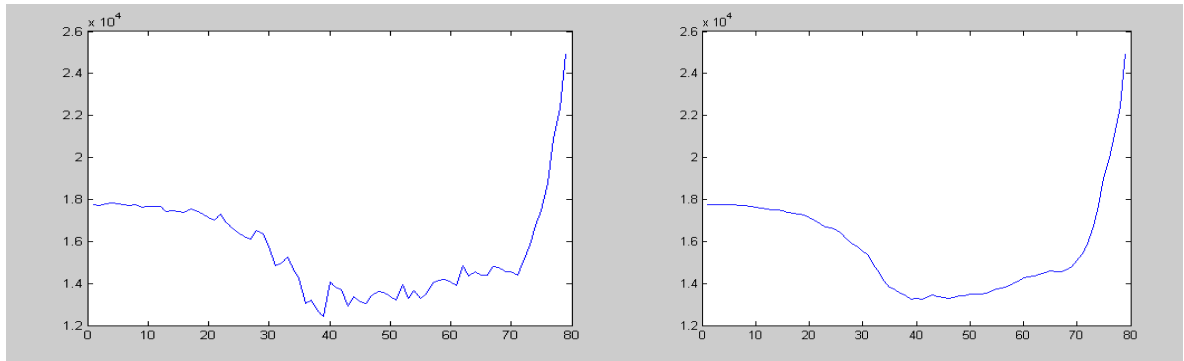
Time Series 104



Time Series 105



Time Series 106



Source of Turing Test Data

Time Series	Source Data	Market Worth (in mn\$ or 1000\$)	Starting Date	Closing Date
1	Tekstilbank	million\$	3-Jan-01	27-Feb-01
2	Erdemir	million\$	5-Dec-96	28-Feb-97
3	Tümteks	million\$	13-Jun-98	17-Aug-98
4	Karsan	million\$	11-May-00	23-Sep-00
5	Akbank	million\$	10-Mar-04	23-Jul-04
6	Bosch Profilo	million\$	11-Feb-97	27-May-97
7	Karsu Tekstil	million\$	10-Aug-00	19-Oct-00
8	İş GYO	million\$	12-Feb-04	12-May-04
9	Finansbank	million\$	15-Jun-98	18-Sep-98
10	Dışbank	million\$	17-Aug-97	10-Mar-98
11	Anadolu Sigorta	million\$	15-May-99	4-Jul-99
12	Anadolu Hayat	million\$	16-Nov-01	20-Jan-02
13	Ereğli	million\$	11-Jan-99	15-Feb-99
14	Gümüşsuyu	million\$	11-Feb-95	2-Apr-95
15	Demirbank	million\$	12-May-98	4-Oct-98
16	Tüpraş	million\$	10-Feb-03	12-Mar-03
17	Şişecam	million\$	11-Apr-00	25-Jul-00
18	Petrol Ofisi	million\$	10-Apr-94	9-Jul-94
19	Finansbank	million\$	12-Sep-00	16-Nov-00
20	İş Yatırım Ortaklığı	million\$	11-Apr-03	9-Aug-03
21	Nuh Çimento	million\$	10-May-00	3-Aug-00
22	Aksa	million\$	12-Feb-99	29-Mar-99
23	Ereğli	million\$	20-Jul-96	13-Sep-96
24	Tüpraş	million\$	25-Dec-99	8-May-00
25	İzocam	million\$	10-Feb-00	31-Mar-00
26	Doğusan Boru	1000\$	9-Aug-98	22-Nov-98
27	İzmir Demir Çelik	1000\$	10-May-97	23-Aug-97
28	Soda Sanayi	million\$	10-Jun-03	13-Sep-03
29	Kipa	million\$	15-May-99	7-Sep-99
30	Abana	1000\$	10-May-97	8-Aug-97
31	Anadolu Efes	million\$	8-Apr-00	10-Oct-00
32	Adel	1000\$	7-Jun-98	11-Aug-98
33	Yapı Kredi Yat. Ort.	1000\$	6-May-99	29-Aug-99
34	Bosch Fren	1000\$	4-Jul-04	8-Aug-04
35	Aktif Finans	million\$	6-May-96	9-Aug-96
36	Alfa Menkul	1000\$	5-Apr-95	19-Jun-95
37	Sarkuysan	million\$	6-Aug-94	15-Sep-94
38	Alkim Kimya	1000\$	12-Feb-98	6-Aug-98
39	Alkim Kağıt	million\$	11-Feb-97	25-Aug-97
40	Banvit	million\$	9-Sep-99	22-Jan-00
41	Bolu Çimento	million\$	8-Sep-01	21-Jan-02
42	Bayraklı Boya	1000\$	8-Jun-96	30-Nov-96
43	Esbank	million\$	5-Aug-97	19-Oct-97
44	Ceylan Giyim	1000\$	4-May-99	7-Aug-99
45	Birlik Mensucat	1000\$	11-Nov-01	14-Feb-02
46	Borova Yapı	million\$	10-Oct-00	14-Mar-01
47	Mudurnu	1000\$	11-Nov-94	26-Mar-95
48	Milliyet	million\$	12-Jan-97	5-Aug-97
49	Pastavilla	million\$	11-Oct-95	23-Feb-96
50	Pınar Et	million\$	24-Jan-98	14-Apr-99
51	Sabah	million\$	21-Feb-96	15-Jun-96
52	Maret	million\$	1-Oct-97	14-Jan-98
53	Trakya Cam	million\$	1-Dec-99	25-Jan-00

Time Series	Source Data	Market Worth (in mn\$ or 1000\$)	Starting Date	Closing Date
54	Bursa Çimento	million\$	12-Feb-99	17-Jul-99
55	Burçelik Vana	million\$	18-May-98	21-Aug-98
56	Selçuk Gıda	million\$	22-Apr-00	11-Feb-01
57	Tezsan	1000\$	19-May-94	11-Sep-94
58	Sevgi Sağlık	million\$	20-Feb-96	11-Nov-96
59	Söksa	million\$	12-Dec-93	6-Apr-94
60	Viking Kağıt	million\$	10-Nov-99	24-Mar-00
61	Boyasan	million\$	11-Apr-00	17-Dec-00
62	Berdan Teks.	million\$	8-Jun-98	1-Oct-98
63	Gediz İplik	million\$	7-Jul-99	19-Nov-99
64	Kerevitaş	million\$	8-Sep-98	11-May-99
65	Mensa	1000\$	15-Jul-98	24-Aug-98
66	Alternatif YO	1000\$	12-Feb-99	8-May-99
67	Enka	million\$	10-Nov-98	5-Mar-99
68	Petrol Ofisi	million\$	12-Dec-96	25-Feb-97
69	Enka	million\$	8-Sep-97	22-Nov-97
70	Arsan	million\$	6-Nov-00	16-Dec-00
71	Yatırım Finansman YO	1000\$	7-Jul-97	20-Sep-97
72	TSKB YO	1000\$	8-Sep-99	12-Dec-99
73	Deniz YO	1000\$	9-Nov-99	22-May-00
74	Eczacıbaşı YO	1000\$	8-Aug-94	12-Oct-94
75	Alternatifbank	million\$	11-Jun-00	24-Oct-00
76	Demir YO	1000\$	12-Aug-95	5-Dec-95
77	Oysa	million\$	7-Jun-94	30-Sep-94
78	Petkim	million\$	8-Jun-95	1-Oct-95
79	Medya Holding	million\$	15-Oct-00	29-Nov-00
80	Sinai Yat. Bankası	million\$	6-Jun-97	19-Sep-97
81	Mazhar Zorlu Holding	1000\$	7-Dec-00	1-Apr-01
82	Aksa	million\$	11-Aug-00	19-Nov-00
83	Nurol GYO	million\$	8-Aug-99	1-Dec-99
84	Garanti YO	1000\$	7-Apr-00	16-Jun-00
85	İş YO	million\$	6-Dec-00	14-Feb-01
86	Tofaş	million\$	5-Nov-97	8-Jun-98
87	Döktaş	million\$	12-Apr-01	6-Jun-01
88	Dışbank	million\$	7-Sep-01	30-Jan-02
89	Olmuxsa	million\$	8-Mar-99	11-Jun-99
90	Rant Leasing	million\$	6-Jan-96	18-Aug-96
91	Net Turizm	million\$	7-Aug-00	14-Jan-01
92	Ray Sigorta	1000\$	6-May-01	25-Jul-01
93	Tacirler YO	1000\$	5-Apr-00	29-Jun-00
94	Arat	million\$	5-May-99	29-Jul-99
95	Fenerbahçe	million\$	12-Nov-04	5-Feb-05
96	Bayraklı Boya	million\$	5-Jun-96	28-Sep-96
97	Tekstil Leasing	1000\$	7-Aug-01	15-Nov-01
98	Yasarbank		6-Nov-98	30-Jan-99
99	Sifariş	1000s	5-Aug-96	28-Nov-96
100	Petrokent	1000s	26-Nov-99	24-Feb-00
101	Marmaris Altinyunus	1000s	6-Nov-00	16-Mar-01
102	Mardin Çimento	1000s	5-Aug-00	19-Oct-00
103	Mustafa Yılmaz YO	1000s	24-Oct-99	16-Feb-00
104	Deniz YO	1000s	28-Feb-00	11-Aug-00
105	Global YO	1000s	27-Oct-99	20-Mar-00
106	Gedik YO	1000s	21-Oct-00	14-Jan-01

REFERENCES

- Barlas, Y., 1996, "Formal Aspects of Model Validity and Validation in System Dynamics," *System Dynamics Review*, Vol. 12, No.3, pp. 183-210.
- Barlas, Y. And S. Carpenter, 1990, "Philosophical Roots of Model Validation: Two Paradigms," *System Dynamics Review*, Vol. 6, No. 2, pp. 148-166.
- Barlas, Y., 1989a, "Multiple Tests for Validation of System Dynamics Type of Simulation Models," *European Journal of Operational Research*, Vol. 42, No. 1, pp. 59-87.
- Barlas, Y., 1989b, "Tests of Model Behavior That Can Detect Structural Flaws: Demonstration With Simulation Experiments," In P. M Milling and E. O. K. Zahn (Eds.), *Computer Based Management of Complex Systems*, pp. 246-254, Berlin: Springer-Verlag.
- Boğ, S. and Y. Barlas, 2005, "Automated Dynamic Pattern Testing, Parameter Calibration and Policy Improvement," *System Dynamics Society Conference*, Boston: USA.
- Charnes, A., W. W. Cooper, D. Divine, T. W. Ruefler, and D. Thomas, (1988), "Comparisons of DEA and Existing Action and Regression Systems for Effecting Efficiency Evaluations of Regulated Electric Cooperatives in Texas," Unpublished Paper, center for Cybernetic Studies, Austin, TX.
- Forrester J. W. and P. M. Senge, 1980, "Test for building Confidence in System Dynamics Models," In A. A. Legasto , J. W. Forrester and J. M. Lyneis, (Eds.) , *System Dynamics*, Amsterdam: North Holland.
- Forrester J. W., G. W. Low and N. J. Mass, 1974, "The Debate on World Dynamics: A Response to Nordhaus," *Policy Sciences*, Vol. 5, pp. 169-190.

- Forrester, J. W., 1968, "A Response to Ansoff and Slevin", *Management Science*, Vol. 14, pp. 601-618.
- He, Y. and A. Kundu, 1991, "2-D Shape Classification Using Hidden Markov Model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-13, No. 11, pp. 1172-1184.
- Juang, B. and L. R. Rabiner, 1990, "The Segmental K-means Algorithm for Estimating Parameters of Hidden Markov Models," *IEEE Transactions in Acoustics Speech and Signal Processing*, Vol. ASSPO-38, No. 9, pp. 1639-1641.
- Kanar K. and Y. Barlas, 1999, "A Dynamic Pattern-oriented Test for Model Validation," *System Dynamics Society, 17th International Conference*, Wellington: New Zealand.
- Kanar, K., 1999, "Structure Oriented Behavior Tests in Model Validation," *Boğaziçi University Institute for Graduate Studies in Science and Engineering, Industrial Engineering Department M.S. Thesis*.
- O'Leary, D., 1991, "Measuring the Quality of Computer Model Performance," *European Journal of Operational Research*, Vol. 56, pp. 319-331.
- O'Leary, D., 1987, "Validation of Expert Systems", *Decision Sciences*, Summer, pp. 468-486.
- Richardson, G. P., 1996, "Problems for the Future of System Dynamics," *System Dynamics Review*, Vol. 12, No. 2, pp. 141-157.
- Sauder, R. and W. Westerman, 1983, "Computer Aided Train Dispatching: Decision Support Through optimization," *Interfaces*, Vol. 13, No. 6, pp. 24-37.
- Sterman, J. D., "Business Dynamics, systems Thinking and Modeling for a Complex World," Irwin McGraw-Hill, Inc, 2000.

Sterman, J. D., 1984, "Appropriate Summary Statistics for Evaluating the Historical Fit of system dynamics Models," *Dynamica*, Vol. 10, No. 2, pp. 51-66.

Turing, A. M., 1950, "Computer Machinery and Intelligence," *Mind*, Vol. 59, pp. 433-460.