Learning Short Multivariate Time Series Models through Evolutionary and Sparse Matrix Computation

Stephen Swift[1], Joost Kok[2], Xiaohui Liu[1,2]

[1]School of Information Systems, Computing and Mathematics,

Brunel University, Uxbridge, Middlesex, UB8 3PH, United Kingdom.

Phone: +44 (0) 1895 203397, Fax: +44 (0) 1895 251686, Email: Xiaohui.Liu@brunel.ac.uk

[2]Leiden Institute of Advanced Computer Science,

Leiden University, P.O. Box 9512, 2300 RA Leiden, The Netherlands.

ABSTRACT

Multivariate Time Series (MTS) data are widely available in different fields including medicine, finance, bioinformatics, science and engineering. Modelling MTS data accurately is important for many decision making activities. One area that has been largely overlooked so far is the particular type of time series where the data set consists of a large number of variables but with a small number of observations. In this paper we describe the development of a novel computational method based on Natural Computation and sparse matrices that bypasses the size restrictions of traditional statistical MTS methods, makes no distribution assumptions, and also locates the associated parameters. Extensive results are presented, where the proposed method is compared with both traditional statistical and heuristic search techniques and evaluated on a number of criteria. The results have implications for a wide range of applications involving the learning of short MTS models.

Keywords: Natural Computation, Short Multivariate Time Series, Sparse Matrices, Short-term Forecasting, Glaucoma

Running Header: Learning Short MTS Models

Abbreviations

Term	Meaning
GA	Genetic Algorithm
HC	Hill Climbing
LS	Least Squares
ML	Maximum Likelihood
MTS	Multivariate Time Series
SSV	Seeded Sparse-VARGA

Term	Meaning
SVNP	Sparse-VARGA-No-Padding
SVP	Sparse-VARGA-Padding
VAR	Vector Auto-Regressive
VARGA	VAR Genetic Algorithm
WK	Weighted-Kappa
YW	Yule-Walker

Introduction

Much research has gone into the development of ways of analysing multivariate time series (MTS) data in both the statistical and artificial intelligence communities. Statistical MTS modelling methods include the Vector Auto-Regressive process [Lütkepohl 1993], the Vector Auto-Regressive Moving Average process [Lütkepohl 1993], and other non-linear [Casdagli 1992] and Bayesian approaches [Pole et al. 1994], while various AI methods have been developed for different purposes including dependence detection in MTS of categorical data [Oates et al. 1999], knowledge based temporal abstraction [Kadous 1999, Shahar 1997], and forecasting [Meir 2000, Weigend 1994]. However, one area that has been largely overlooked is the particular type of time series where the data set consists of a large number of variables but with a small number of observations.

A multivariate time series (MTS) is a series of n observations, $x_i(t)$; [i=2, ..., n; t=1,...,T], made sequentially through time where t indexes the different measurements made at each time point, and i indexes the number of variables in the time series. Throughout this paper the terms "observation" and "variable" will be used synonymously. The vector notation $\underline{x}(t)$ is a shorthand way of referring to the whole set of observations made at time t, i.e. $\underline{x}(t)$ stands for the observations $x_i(t)$ where $2 \le i \le n$. A commonly used statistical method for modelling MTS is the Vector Auto-Regressive Process, usually denoted as VAR(p) for a model of order p, as defined in equation 1. An introduction can be found in [Holden 1995].

$$\underline{x}(t) = \sum_{i=1}^{p} A_i \cdot \underline{x}(t-i) + \underline{\varepsilon}(t) \tag{1}$$

Where $\underline{x}(t)$ is the next data vector of size n (the number of variables in the model), A_i is a $n \times n$ coefficient matrix at time lag i, and $\underline{\varepsilon}(t)$ is an n dimensional zero mean noise vector at time t (usually Gaussian). The value of each element in A_i is usually a bound real number.

Definition 1. The notation nVAR(p) will refer to an n variable VAR process of order p, where p is an integer greater or equal to zero.

Definition 2. The *Noise Model* will refer to an nVAR(0) process, i.e. one generated by noise. This model is very easy to construct and simulate, and will be used as a simple benchmark with which to rate other VAR processes against. A VAR process that is designed to model a particular MTS is expected to perform better than the corresponding *Noise Model*, since the *Noise Model* simply assumes that any time series observations were totally randomly generated, with no structure and no reliance on previous observations.

The standard statistical methods for fitting a VAR process to a set of data often consist of two steps: order selection (determining a suitable p) and parameter estimation (calculating the matrices A_i from the data). Order selection is commonly performed through the use of information theory based metrics such as AIC (Akaike's Information Criterion) [Akaike 1974, Lütkepohl 1993]. Many of these metrics will impose a restriction on the minimum length of a

MTS (i.e. setting a lower limit for the series length *T*), based on the number of degrees of freedom of the model being estimated, namely $T>np + 1$. Here, *n* is the number of variables being modelled, and *p* is the order of the VAR process. For example, with an MTS involving 10 variables, in order to find a VAR process with a maximum order of five, *T* must be at least 52 (the time series length). This restriction is unacceptable for modelling many short time series such as gene expression data produced by DNA array technology [Lockhart 2000] or many medical series such as visual field data [Haley 1987].

Additionally the parameter estimation step can experience some difficulties when dealing with a short MTS. The standard methods for parameter estimation include *Maximum Likelihood* (ML) methods, the *Yule-Walker* (YW) equations method, and the *Least Squares* (LS) method [Lütkepohl 1993]. With ML methods, there must be some distribution assumptions on the noise vector. If the MTS values are exclusively within a set interval, an appropriate continuous distribution might be difficult to find. This is the case with dataset that is the subject of this paper. The YW method can involve matrix inversion, which is computationally expensive with large matrices and can fail if the matrix is singular (especially with shorter time series). The LS method is often used in preference to the YW equations, but it too can involve matrix inversion, and can also impose the degree of freedom restriction mentioned above (this is involved in computing an unbiased estimator for the covariance matrix of the associated noise vector).

In this paper we describe the development of a novel computational method based on genetic algorithms (GA) and sparse matrices. This research significantly extends our previous work. In [Swift 1999a] we presented VARGA, a novel genetic algorithm for determining VAR processes where each gene was a parameter matrix; and in [Swift 2002a] we improved on

VARGA by introducing additional crossover and mutation operators and providing an analysis of their performance. We now approach the problem from a new perspective. Our new method has the advantage that it allows the bypassing of the time series length restrictions that are inherent in some traditional statistical MTS methods, whilst making no distribution assumptions and additionally computing the associated parameters. The method is evaluated extensively against other statistical and heuristic search methods using a number of different criteria.

This paper is organised as follows. First we introduce Sparse-VARGA, the subject of this paper. A description of how we evaluate this new method, using both real world medical and synthetic MTS datasets, follows. Finally we draw some conclusions and describe the opportunities for further research.

The Sparse-VARGA Method

Previously we have identified the difficulties in modelling short high dimensional MTS and in [Swift 1999a] these difficulties were dealt with for the prediction of visual deterioration in glaucoma patients using a genetic algorithm approach. The initial idea was that an nVAR(p) process would be represented as a *chromosome* of p $n \times n$ matrices. Suitably modified genetic operators were then applied over subsequent generations improving the population’s fitness and thus finding the required parameters. A further improvement upon this method was presented in [Swift 2002a]. Both these methods were referred to as VARGA which stands for *Vector Auto-Regressive model fitting through Genetic Algorithms*. However the overall representation was inefficient in terms of algorithm speed and to some extent accuracy, and also suffered a slight bias towards models with a low order.

In this paper we extend this work in three key areas: improving algorithm efficiency, ensuring unbiased order selection and conducting an extensive evaluation.

VAR subset models [Lütkepohl 1993] have been used for short term forecasting. A VAR subset model is a VAR process that has one or more of its parameters set to zero, representing the situation where not every variable depends on every other variable at all possible time lags. In adopting such an approach, the number of dependences to search for is reduced; however deciding which parameters to set to zero is a difficult task. In [Bearse 1998] a binary GA is used towards these ends, where the binary chromosome represents which of the parameters are set to zero, i.e. the chromosome consists of n^2p *genes* where each '1' means that the corresponding element in a parameter matrix is set to zero, and a '0' means it is left unchanged. However this method still relies upon the initial parameters and order being determined by some other method. It is proposed that all of these steps can be performed as one whole process, i.e. order determination, parameter computation and VAR subset selection. VAR processes that are highly sparse (lots of zeros for parameters) are a lot easier to work with than a densely populated VAR process.

Sparse Matrix Theory [Zlatev 1991] suggests that if most of a matrix contains zeros, then one should only store the values that are not zero as opposed to all of the elements, i.e. assume that all elements are zero unless indicated otherwise. It would be more efficient to store the parameter matrices using a sparse representation; this is the idea behind *Sparse*-VARGA. This idea was also motivated from a similar representation introduced in [Tucker 1999], where a collection of 3-tuples represented the potential links in the search for dynamic Bayesian network structures. In [Swift et al. 1999b] it was also noted that sometimes highly sparse parameters matrices were often beneficial, and aided mutation. Sparse-VARGA stores the non-zero elements of each parameter matrix of a MAXORDER VAR process as a variable length unordered list, where MAXORDER is the maximum order (number of parameter matrices) any VAR process under consideration can have. The Sparse-VARGA algorithm is similar to a standard genetic algorithm but has enhancements to deal with variable length chromosomes.

Algorithm 1: Sparse-VARGA

```
1)    Input:   The Sparse-VARGA Parameters (table 2) and the Fitness Function (equation 3)
2)    Create POPULATION chromosomes
3)    For g = 1 to GENERATIONS
4)             *Shuffle* population to create Children
5)             Clone the population
6)             *Mutate* the Clones and add back to the population
7)             Add the Children back to the population
8)             *Select* the new population
9)    Next g
10)   Output:  The best VAR process; which is the chromosome from the final population with
               the largest fitness score
```

By using VAR subset models, there will be a reduction in the number of parameters being searched for. This will result in a much simpler model thereby making the VAR process easier to generalise. By using a sparse matrix representation, the algorithm will be computationally more efficient. Algorithm 1 describes Sparse-VARGA. All of the operators described in this algorithm are similar to those of a standard genetic algorithm [Holland 1975], where the *Shuffle* operator is analogous to crossover. The rest of this section explains each part of this algorithm.

Representation

A chromosome is an unordered variable length list of genes; each chromosome will range in size from one gene to some upper limit (say MAXSIZE). A gene is a tuple (*lag*,*row*,*col*,*value*), where *lag* is the particular parameter matrix (assuming there are 1,..,MAXORDER parameter matrices), *row* is the row of a parameter matrix, *col* is the column of the same matrix and *value* is the value the particular coefficient defined by (*lag*,*row*,*col*); hence $a_{(lag)(row)(col)}$=*value*. CHROMESIZE $(\leq \text{MAXSIZE})$ will be defined as the number of genes in a given chromosome. In order to reconstruct the parameter matrices, the genes of the *chromosome* are used to fill a set of matrices. It is assumed that all of the possible parameter matrices have zero for each element unless there is a gene that indicates otherwise. Each gene element is bounded as in table 1.

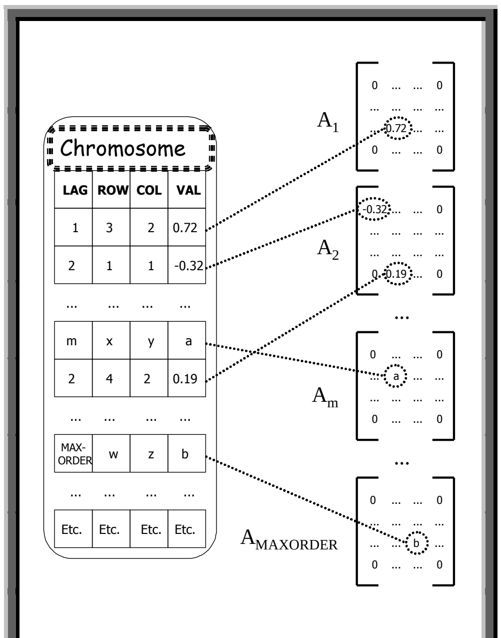

Figure 1: Sparse-VARGA Representation

Element	Type	Range
Lag	Integer	1…MAXORDER
Row	Integer	1…n
Col	Integer	1…n
Value	Real	-1.25…+1.25

Table 1: Gene Part Types and Limits

Figure 1 shows the representation for Sparse-VARGA. Duplicate gene handling is fully described in [Swift 2002b]. A duplicate is where two genes have the same value for the *lag*, *row* and *col* parts of the gene. This could occur during the *Shuffle* (resolved by retaining the gene that came from the fittest parent) and/or *Mutation* (resolved by mutating the gene until no duplicates exist) part of the algorithm or even during the creation of the initial population (handled in a similar manner to *Mutation*).

Fitness

As with a normal GA, Sparse-VARGA needs a suitable fitness function to evaluate candidate solutions to the problem. This fitness will rate a potential solution against a given dataset using some evaluation criteria. The fitness of a chromosome is determined by the sum of the magnitudes of the noise vectors computed by performing a series of one-step ahead forecasts from $t=T_0,..,T$ defined in equations 2 and 3.

$$\underline{\hat{\varepsilon}}(t) = \underline{x}(t) - \sum_{i=1}^{p} \hat{A}_i \cdot \underline{x}(t-i) \qquad (2)$$

$$\varepsilon = \sum_{t=T_0}^{T} \sum_{j=1}^{n} \left| \hat{\varepsilon}_j(t) \right| \qquad (3)$$

where $\hat{\varepsilon}(t)$ is the estimation of the noise vector, $\hat{A}_i$ is the estimation of the *ith* parameter matrix, ε is a scalar that represents the level of noise, and T_0 is defined as MAXORDER+1. Absolute value error rather than mean squared error is used because mean squared error can exaggerate the effects of outliers, especially when the time series is short [Armstrong 1992, Chatfield 1992]. All other variables are defined above. The model with the smallest ε value is deemed the most suitable model for forecasting since it is assumed that the best estimation for any unobserved noise vector is the zero vector.

Initial Population

In order to improve convergence, it would be desirable to start Sparse-VARGA from points in the search space which are better than randomly generated chromosomes. Seeding [Michalewicz 1996, Sharif 1998] is where the initial population for a genetic algorithm has some or all of its chromosomes set to some predefined representation, as opposed to an entirely random selection. Usually these seeds come from expert knowledge or some other fast approximate search method, for example Hill-Climbing. If the Yule-Walker equations can be solved for a given order for a given time series, then the results could be a good starting point for a VAR process specialised for short term forecasting. The Yule-Walker equations have been chosen since it is one of the standard methods for finding the parameters of a VAR process and has the least implementation problems (see the introduction). The *Noise Process* is also included because this proved to be a good benchmark to compare with any other model from the experiments detailed in [Swift 2002a].

As mentioned above, there are two choices for selecting the seeds. The first represents the noise model, and the second are the results of the Yule-Walker equations run for order 1,...,MAXORDER. Note that for the Yule-Walker seeds it is assumed that the value for MAXORDER is less than the intended population size. Another strategy for seeding is whether to pad out the seeds with zeros, i.e. genes where the *value* part is set to zero. This would enable *Gene Mutation* to affect parts of the solution that would not normally be available to it. Padding can be seen as giving mutation a boost within the early generations of the algorithm, generating many different genes for the *Shuffle* operator to divide between the population. Based on these seeding options, three sets of initial populations are generated using the following strategies:

Seeded Sparse-VARGA (SSV). Chromosomes are created by solving the *Yule-Walker Equations* for an order ranging from one to MAXORDER and then using the results as a seed. Each seed is padded out with zero where elements are not indicated, creating MAXORDER number of 100% dense matrices. The rest of the population (POPSIZE-MAXORDER) consists of 50% chance of a noise or random individual (within the constraints of table 2). Each noise seed has a 50% chance of being padded out with zeros as above.

Sparse-VARGA-Padding (SVP). Each individual has a 50% chance of being a noise seed or a randomly generated individual (within the constraints of table 2). Each chromosome has a 50% chance of being padded out with zero *genes* as above.

Sparse-VARGA-No-Padding (SVNP). Each individual has a 50% chance of being a noise seed or a randomly generated individual (within the constraints of table 2). No individual is padded out with zeros.

These are the three variants on seeding that will be evaluated for effectiveness. The algorithms for creating these seeds are described in [Swift 2002b]. When creating the initial population, special provision is made to ensure that there are no duplicate genes within a chromosome. For example, with a randomly created individual, a gene is only added if it is not already in the chromosome. Duplicate handling in the initial population is also described in [Swift 2002b].

Shuffle

Shuffle is an operator similar to the *Uniform Crossover* operator described in [Syswerda 1989]. Here all of the genes of both parents are placed into a single collection, which is then randomly shuffled. The first and the second gene element from this collection are then removed to create two single gene *chromosomes*, the starting point for two new children. The remaining genes in the collection are divided between these two new children by deciding randomly where each one goes (50% chance for each child). Figure 2 demonstrates this.

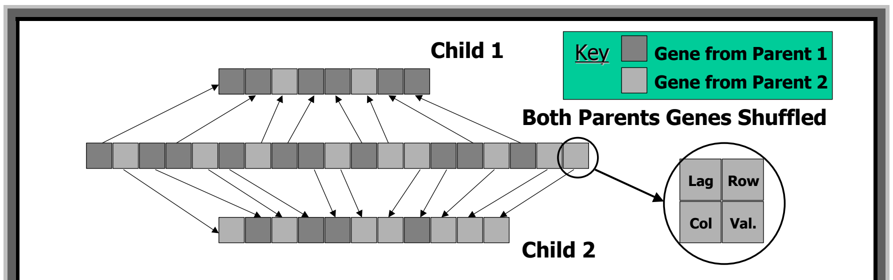

Figure 2: Sparse-VARGA Shuffle Operator

Note that the representation does not maintain a one to one mapping between phenotype and genotype (very desirable in genetic algorithm representations [Goldberg 1989, Holland1975]) since it has no sense of ordering. For example, given two *chromosomes* *x* and *y* where *y* is simply *x* in a different order, each would effectively be the same VAR process; this is the reason why the original version of *Uniform Crossover* (or any other standard crossover operator) will not work. However, this operator will effectively randomly divide the genes of two *chromosomes* between two new children, and when combined with the *Selection* operator, will result in children being produced that contain useful genes from both parents. As previously mentioned, this operator could result in duplicate genes. This could occur if the two parents have the same gene, i.e. one where the *lag*, *row*, and *col* part are the same. Note that each parent is assumed to have a unique set of genes before the operator is applied. During the *Shuffle* operation, before a gene is added to a child, a check is made to see if it is already contained in that child. If this is the case, the gene that came from the fittest parent is retained only, i.e. the *value* part of the gene is set to the corresponding *value* from the gene from the best parent.

Mutation

Mutation in the Sparse-VARGA algorithm is a two-stage process, i.e. *Gene Mutation* and *Size Mutation*. However there is a slight variation. A chromosome can only undergo either *Gene*

Mutation or *Size Mutation* with a 50% chance of either occurring. Note that only the parents mutate, and they will always mutate to create a new individual. It was found through initial experimentation that this produced better results than mutating just the children or mutating both the parents and children.

Size Mutation

This causes a chromosome to increase or decrease in size by the addition or deletion of a number of genes. A chromosome has a 50% chance of losing some genes (effectively setting specified parameters to zero), or gaining new genes (effectively setting zeros to values). Note that if a gene is removed then it is randomly selected from all of the genes within the chromosome. The size of a chromosome is restricted between one and some predefined upper limit, say MAXSIZE. Figure 3 shows this process (for each addition or deletion).

Definition 3. The function *UI*(MIN,MAX) is a uniformly distributed random number generator that returns an *integer* number in the interval [MIN,MAX].

CHROMESIZE is the number of genes in the chromosome, NSIZEMU and NSIZESTD are application dependent parameters, *Ceil* is a function that returns the nearest integer that is greater than the specified parameter, $N(\mu,\sigma)$ is a Gasssian/Normal distribution with mean μ and standard deviation σ and N_{Mute} is the number of *Size Mutations*. Deleting a gene from a chromosome will not result in any duplicates, but the addition of a random gene could do. To prevent this occurring, a random gene is repeatedly generated until it can be added to a chromosome without duplication.

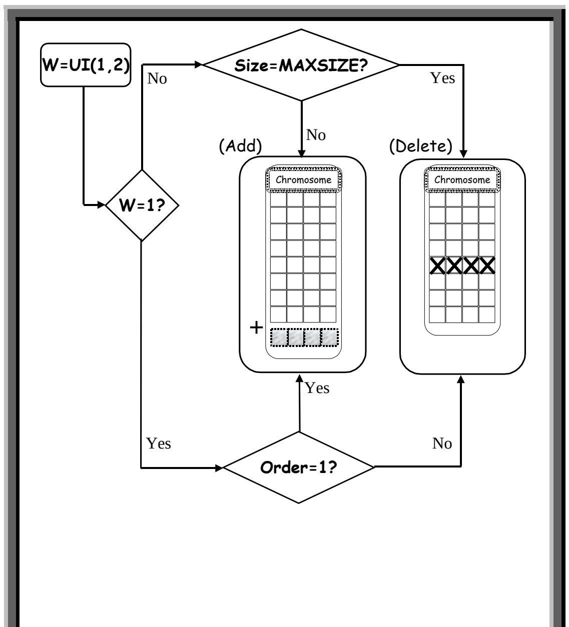

Figure 3: Sparse-VARGA Size Mutation

The number of additions or deletions is computed according to equation 6.

$$\mu = \text{CHROMESIZE} / \text{NSIZEMU} \tag{4}$$

$$\sigma = \text{CHROMESIZE} / \text{NSIZESTD} \tag{5}$$

$$N_{Mute} = Ceil\left(\left|N(\mu,\sigma)\right| + 1\right) \tag{6}$$

Gene Mutation

This mutates a single gene of an individual. The mutation only happens to the *value* part of the gene. Algorithm 2 describes this procedure.

Algorithm 2: Gene Element Mutation

```
1)    Input:  CHROME – a Chromosome, CHROMESIZE – the size of a Chromosome
              MAXGENE, MINGENE – Gene limits
2)    Set i = UI(1,CHROMESIZE), Set a = Gene i of CHROME, Set New_Value = N(a,σ)
3)    If New_Value > MAXGENE then Set New_Value = MAXGENE
4)    If New_Value < MINGENE then Set New_Value = MINGENE
5)    Set Gene i of CHROME to New_Value
6)    Output: CHROME, the mutated Chromosome
```

where σ is an application dependent parameter that is a standard deviation for the Normally distributed *Creep Mutation* [Goldberg 1990], and MINGENE and MAXGENE are the bounds an element of a VAR parameter matrix can take. *Gene Mutation* will not result in any duplicate genes since it only changes the *value* part of a gene. Note that *Gene Mutation* can only change a value for a position in the parameter matrices that a gene represents, which is why padding is used in the seeded population: giving many more potential targets for *Gene Mutation* to change.

Population Dynamics

The population grows through the application of genetic operators as in figure 4.

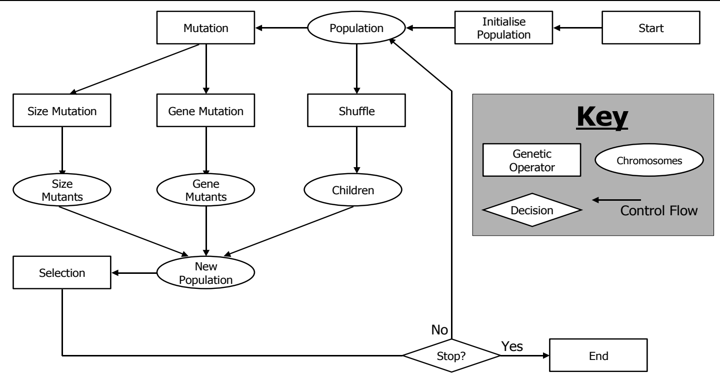

Figure 4: Sparse-VARGA Population Growth

The population will increase by a factor equal to 1+*ShuffleRate*. Note that the *Selection* operator reduces the population back to the size it was before the *Shuffle* and *Mutation* operators were applied.

One set of children is produced through the application of the *Shuffle* operator. The *Mutation* operators are applied, *Size* and *Gene*, to the old population consisting of the parents. Each parent has a chance of breeding (*ShuffleRate*). Therefore before *Selection*, the population will now consist of the parents, the mutated parents and children.

Selection

The *Selection* operator is the *ranked selection* described in [Baker 1985]. This is the same as the roulette wheel technique [Holland 1975], but the chromosome’s rank is used, rather than its fitness. This is because the task of finding a suitable VAR process is a minimisation problem; hence the roulette wheel would favour the worst in a population rather than the best. *Elitism* [Dejong 1975] is also used, which is the process of selecting a predefined number of the best *chromosomes* for survival, before the rest of the next population is selected.

Parameter	**Meaning**	**Value**
POPULATION	The size of the population	10
GENERATIONS	The number of generations the algorithm is run for (given the value for MAXCALLS)	~600
ELITISM	The number of the fittest individuals guaranteed survival	2
ShuffleRate	The chance of a chromosome becoming a parent	0.650
MINGENE	The minimum value a gene (*value*) can take	-1.250
MAXGENE	The maximum value a gene (*value*) can take	1.250
MAXORDER	The maximum possible order of a VAR process	5
MAXSIZE	The maximum number of genes a chromosome can have	n^2MAXORDER
NSIZEMU	The mean scaling constant in equation 4	20
NSIZESTD	The standard deviation scaling constant in equation 5	20
σ	The standard deviation for algorithm 2	0.400
MAXCALLS	The number of fitness calls allowed	10,000

Table 2: Sparse-VARGA Parameters

Parameters

Table 2 details the parameters of Sparse-VARGA. Note that GENERATIONS is an approximate figure, since all of the experiments were executed for a fixed number of fitness function calls.

Evaluation

The models found using the three Sparse-VARGA methods are compared with those produced by a conventional way of finding a VAR process, i.e. the solution of the *Yule-Walker* equations and a Hill-Climbing based method. All of the methods will be evaluated on three separate criteria. These are *Forecast Error*, *Weighted-Kappa* and *Complexity*. Forecast error is used since it is a direct way of indicating the accuracy of each method, i.e. how closely each one matches the original time series. Weighted-Kappa provides a measure of whether the predicted sequence follows the *direction of change* of the original. Direction of change refers to whether the time series is increasing, decreasing or staying the same as the previous observation. The Weighted-Kappa metric is described in [Swift et al. 2004]. Note that the closer the Weighted-Kappa value is to one, the better (see [Altman 1997]). An indication of complexity will put each model into perspective, for example if one method is 0.1% more accurate than another but 1000 times slower in executing, then the less accurate but faster model is the better choice in many real world applications.

The results are divided into three distinct sections. The first section (**Application Dependant Results**) discusses the results of the methods applied to the real world dataset, the second (**Application Independent Results**) discusses the performance of the Yule-Walker equations and considers *Complexity*. The third set of results uses synthetic data in order to show that Sparse-VARGA works well on both training and test data samples.

This rest of this section is organised as follows, firstly a description of the methods to be contrasted is presented, followed by a description of the application dataset used to evaluate the methods. The results then follow.

Methods to be Contrasted

In [Swift 1999a] S-Plus was used to solve the Yule-Walker equations. However, as noted above, there can be some problems with locating the order, and there can be some matrix inversion problems. To tackle these problems, a modified version of the Yule-Walker equations has been implemented external to S-Plus and order selection will be ignored. The Yule-Walker equations will be applied to the test data for all possible lags, and then the model with the lowest forecast error will be chosen. As with previous work, the noise model will also be implemented to put any results into context. Additionally a Hill-Climbing based search will be used to locate the parameters for a VAR(MAXORDER) process.

The Yule-Walker Equations

The method described by Lütkepohl in [Lütkepohl 1993] will be used. Essentially, a relationship between the parameter matrices and the auto-covariance function of a VAR process exists under certain circumstances. This relationship can be exploited and rearranged to produce a set of linear matrix equations called the Yule-Walker equations.

Hill-Climbing

The Hill-Climbing (HC) algorithm [Russell 1995] is a local search based method that iteratively improves a solution by making a single step from one point in the search space to an improved point. Usually, the starting point is randomly chosen, and then a sequence of one-step (usually random) changes is made. After each change, if the new point is better than the previous one, then the old one is forgotten, and any further changes are made to this new solution. The Hill-Climbing procedure for learning the VAR process is fully described in [Swift 2002b].

Test Data

A set of *Normal Tension Glaucoma Visual Field* data will be used to evaluate the Sparse-VARGA method, particularly in the aspects of *forecast accuracy* and *direction of change*. Glaucoma is the name given to a family of eye conditions [Hitchings 2000]. The common trait of these conditions is a functional abnormality in the optic nerve, leading to loss of visual field. The prediction of visual field deterioration in patients who are suffering from glaucoma plays an important role in the management, treatment and control of the diseases progress. For example, if the deterioration is slowing down, it might be appropriate to reduce the medication; or if the deterioration is speeding up, an increase in medication might be needed or surgery might be necessary.

Visual fields are where the retina is divided into a set of points, and the patient is tested to see how sensitive their eyesight is at each point. This level of sensitivity is usually a number between 0 and 60, which is measured through the use of a machine, taking several hours to examine all of the points in both eyes. A patient, once diagnosed with glaucoma or if they are a glaucoma suspect (e.g. ocular-hypertension), will undergo these tests approximately every 6 months. The rate of change of the patient’s sensitivity for parts of the eye, and as a whole, can be a very useful guide for clinicians who are treating these patients.

For normal tension glaucoma the *Central Threshold* 30-2 test is usually used which tests 76 points. The points can be grouped according to nerve fibre bundles where there are 16 different groups; the size of each group is listed as part of table 10. Current theory [Crabb et al. 1996/97, Heijl et al. 1988/89] states that deterioration of the visual field can be highly correlated if two points lie on the same nerve fibre bundle.

In [Swift 1999b] the visual field data were clearly demonstrated to be multivariate. Eighty-two patients' visual fields from their right eye have been made available, the length of which forms 82 rather short MTS (ranging from 10 to 44). In this paper all of the 16 groups (points within a nerve fibre bundle) will be considered to be a separate MTS for each patient; the number of variables in each MTS will be the number of points in each group.

Additionally each patient's visual field will be modelled as a single 76 variable MTS. This latter modelling will only be performed with the Yule-Walker equations, since there would be at least $76^2 = 5776$ parameters, which is far more than the number of data points available. Modelling the data as a single MTS should show whether any decomposition into smaller subsets is necessary or not.

It is worth noting that only the visual field data is available for each patient. No other information is known, apart from the fact that each one was undergoing treatment whilst the fields were being recorded. Therefore, the treatment and patient demographic cannot be factored into any models produced. Additionally, the results cannot be analysed by ophthalmic clinicians to verify the results, hence empirically testing the model results is the only way the experiments can be verified.

Application Dependent Results

In this section the results of running the following seven sets of experiments are examined:

i) The Yule-Walker (YW76) equations on the 76 points for each patient

ii) The Yule-Walker (YW16) equations for each nerve fibre bundle of each patient

iii) The noise model (NOISE) for each nerve fibre bundle of each patient

iv) *Seeded Sparse*-VARGA (SSV) for each nerve fibre bundle of each patient

v) *Sparse*-VARGA-*Padding* (SVP) for each nerve fibre bundle of each patient

vi) *Sparse*-VARGA-*No Padding* (SVNP) for each nerve fibre bundle of each patient

vii) Hill-Climbing for each nerve fibre bundle of each patient

Experiment sets 4-7 are stochastic, hence each will be run ten times, and then the results averaged. This gives six lots of 1312 (82 patients by 16 nerve fibre bundles) sets of results (experiment sets 2-7) and one set of 82 results (experiment set 1). These are presented as follows:

i) General comments about the results

ii) Results for experiments involving all methods over all groups and patients

iii) Results for methods 2-7 by group (16 comparisons). Note that the YW76 methods results cannot be considered by group since there is only one MTS

iv) Results for all methods by patient (82 comparisons)

Within the results section, the *Mean* will refer to the sample mean, the *Median* value will be the middle value when the result in question is sorted, *StDev.* is the sample standard deviation, *Min.* is the minimum value and *Max.* is the maximum value. The results are organised/summarised in several different ways in order to try and show if any set of experiments have any specific biases, e.g. whether a method performs exceptionally worse or better on a particular nerve fibre bundle or patient.

General Comments

It is worth noting the following, regarding the experiments and the presentation of the results:

i) With the YW16 results, 145 out of 1312 experiments were not solvable (11.05%), i.e. there were no solvable equations for all lags over a nerve fibre bundle for a patient. However 729 out of the 6560 YW16 runs (1312 × 5 lags) were not solvable (11.11%), which resulted in 33 out 82 patients series containing a non-solvable set of YW16 equations for some order.

ii) With the YW76 results, 165 out of 410 experiments were not solvable (40.24%), i.e. there were no solvable equations for all lags over all points for a patient. This is equivalent to 33 patients out of 82, which are exactly the same patients as those mentioned above; this will be discussed later.

iii) The noise model is an *n*VAR(0) process hence the forecast error (fitness) is the sum of the absolute values of each point from MAXORDER+1 to *T*.

iv) For both the forecast error (fitness) and Weighted-Kappa, standard summary statistics will be displayed. For methods 4-6, the size (number of genes) will be presented as an indication of the model’s sparseness.

v) The fitness will be scaled according to equation 7 for each set of results, so that the results are not biased against those cases with a longer time series or with a higher dimensionality. This would allow, for example, a 10 variable time series of length 25 to be compared for accuracy against a 5 variable time series of length 15.

$$Scaled\,Fitness = \frac{Fitness}{n(T - \text{MAXORDER})} \tag{7}$$

vi) For methods 4-7 the experiments were terminated after the evaluation of 10,000 fitness calls was exceeded (MAXCALLS = 10,000). This figure was chosen as it allows for a reasonable chance of convergence (to a good solutions) and is not too large as to make the total number of experiments impractical to carry out.

The rationale behind the selection of many of the parameters used in the experiments is as follows. MAXORDER needed to be smaller than the length of the shortest MTS, which was ten observations, hence a value of fives was selected as in was half the minimum length. A value of ±1.25 for the gene boundaries was selected since from many experiments with the Yule-Walker equations this was the range that nearly all of the parameter matrices elements fell between. MAXCALLS was selected to be 10,000 as from experimental results, the HC and Sparse-VARGA methods had almost always finished converging, and each experiment took a few minutes. This latter point is important since there are over 50,000 experiments, at this number of fitness calls, all of the experiments took a considerable amount of time to run. Allowing more fitness calls would mean that the experiments would take an unacceptable amount of time to complete.

Additionally, if the methods take only a few minutes to complete, then processing the whole of the visual field would be feasible within the time slot allocated to each patient's outpatient appointment, which is approximately 30 minutes. Finally, the values for parameters specific to Sparse-VARGA were selected for the following reasons. A *population* size of ten was chosen so that a sufficient number of *generations* would be iterated through, given the number of function calls specified (MAXCALLS). Based on the *population* size, the value for *elitism* was set to two – too high a value would interfere with the *ranked selection*. The *ShuffleRate* was set to 0.65 to allow a reasonable number of generations; higher values reduced the number of generations, preventing the algorithm from converging. A low value for *ShuffleRate* meant that the algorithm relied too much on *Mutation*, which had an adverse affect on the algorithms performance.

With *Mutation*, NSIZEMU and NSIZESTD were selected so that approximately the size of the chromosome would change by 5%. The rationale for this figure was that any larger a figure would result in too many mutations happening at once. Finally, the value for σ was chosen so that the Normal distribution covered reasonable range of the genes valid range (-1.25,1.25). Too large a value would mean that most of the mutations would be out of the range; whilst too small a value would not change the genes value enough.

Results by Experiment

Tables 3 to 6 show the results of all the experiments considered together. Table 3 provides summary statistics for the forecast error (fitness) for each experiment and table 4 displays the Weighted-Kappa metric. Table 5 shows the number of times each method performed the best with regards to forecast error, whilst table 6 displays the same for the Weighted-Kappa metric.

In table 3 the shaded cells represents the best performance for a given method using a given summary statistic. It can be clearly seen that the Seeded Sparse-VARGA (SSV) method is slightly better than the Hill-Climbing method: about 1.07% better in mean, and 2.05% better in median. The next best method is the *Yule-Walker Equations by nerve fibre bundle* (YW16). Here the mean and median error is about 50% more than with the HC method. The non-padded and then the padded versions of Sparse-VARGA (SVNP and SVP) are next in performance, followed next by the noise model and then the *Yule-Walker equations on all 76 points* (YW76). The standard deviations are smallest for the SSV and HC method (about 35% smaller than the next best – SVNP) indicating that these two methods produce a more stable (consistent) set of forecasts.

Method	**Fitness/Forecast Error**				
	Mean	**Median**	**StDev.**	**Min.**	**Max.**
YW76	3.212	4.592	1.713	0.915	9.789
YW16	0.849	0.738	0.806	0.000	5.641
NOISE	2.940	2.513	1.734	0.000	10.833
SSV	0.556	0.334	0.646	0.000	4.207
SVP	1.653	1.208	1.588	0.000	10.688
SVNP	1.406	1.163	1.032	0.000	6.022
HC	0.562	0.341	0.644	0.000	4.223

Table 3: Summary Statistics for the Seven Methods by Fitness

It is surprising that the YW76 method performs worse than the NOISE method. This perhaps suggests that modelling the visual fields as one large dimensional MTS is the wrong thing to do, and that splitting the variable up into sub-groups (maybe by nerve fibre bundles or by some other arrangement) will provide much better forecasts. In most cases (not YW76) the mean is larger than the median value, and the maximum value is very much larger than the mean (and the median). This would suggest that there are many small forecast errors, and a few very large ones. Note that the minimum value in table 3 is misleading. Some of the nerve fibre bundles for some of the patients are all zero, e.g. the blind spot (which consists of two points in their own group) for those patients with severe glaucoma. All of the methods will attain zero forecast error in this case by finding a model where the parameters are all zero. The YW76 method shows a different minimum as it is modelling all of the points together.

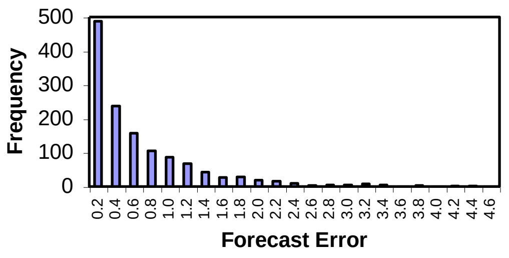

Figure 5: Forecast Error for the SSV Model

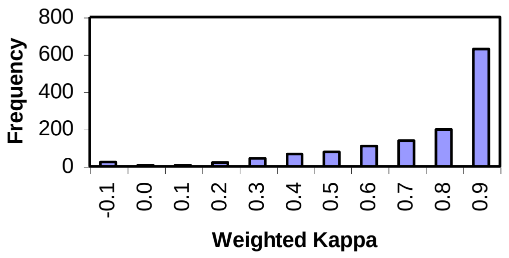

Figure 6: Weighted-Kappa for the SSV Model

Method	Weighted-Kappa				
	Mean	Median	StDev.	Min.	Max.
YW76	0.267	0.320	0.256	-0.028	0.748
YW16	0.686	0.781	0.313	-0.032	1.000
NOISE	0.021	0.000	0.142	0.000	1.000
SSV	0.800	0.886	0.229	-0.093	1.000
SVP	0.561	0.557	0.218	-0.222	1.000
SSNP	0.561	0.562	0.217	-0.200	1.000
HC	0.734	0.803	0.209	-0.089	1.000

Table 4: Summary Statistics for the Seven Methods by Weighted-Kappa

Figure 5 shows a histogram for the forecast error for the SSV method. This diagram clearly shows that many of the errors are less than 0.4. Table 4 contains the same information as table 3, but uses the Weighted-Kappa metric instead of forecast error (fitness). In table 4 the shaded cells represents the best performance for a given method using a given summary statistic.

It can be clearly seen that the Seeded Sparse-VARGA (SSV) methods is better than the Hill-Climbing method: about 8.99% better in mean, and 10.34% better in median. Figure 6 shows a histogram for the results for the SSV method. It can be clearly seen that the majority of the values are greater than 0.7. The next best performing method in terms of the Weighted-Kappa metric is the *Yule-Walker Equations by nerve fibre bundle* (YW16). Here the mean and median error is about 7% and 3% respectively more than with the HC method.

Method Ranking by Fitness	**Count**	**Percentage**
HC=YW16	1	0.076
HC=SSV=YW16	1	0.076
HC=SSV	2	0.152
SSV=YW16	7	0.534
HC=SSV=SVNP=SVP=NOISE	17	1.296
SVNP	24	1.829
YW16	43	3.277
SVP	88	6.707
HC	263	20.046
SSV	866	66.006

Table 5: Method Ranking by Fitness

The non-padded and then the padded versions of Sparse-VARGA (SVNP and SVP) are next in performance, followed next by the *Yule-Walker equations on all 76 points* (YW76) and then the noise model (NOISE). Here the NOISE method does the worst since every single point forecast made by this model is the same, i.e. no change. Note that the standard deviation for the NOISE method is the lowest (all of the others are roughly the same) since it consistently gets a very low value for this metric due to its poor forecasts.

Method Ranking by Weighted-Kappa	Count	Percentage
SSV=SVNP	1	0.076
HC=SSV=SVP	1	0.076
SSV=SVNP=SVP=YW16	1	0.076
SSV=SVP=YW16	1	0.076
SSV=SVNP=SVP	2	0.152
SSV=SVNP=SVP=NOISE=YW16	4	0.305
SSV=SVNP=SVP=NOISE	4	0.305
HC=SSV=SVNP=SVP=NOISE=YW16	5	0.381
NOISE	7	0.534
NOISE=YW16	8	0.610
HC=SSV=SVNP=SVP=NOISE	17	1.296
SVNP	36	2.744
SVP	39	2.973
HC	72	5.488
YW16	158	12.043
SSV=YW16	200	15.244
SSV	756	57.622

Table 6: Method Ranking by Weighted-Kappa

Tables 5 and 6 look at all of the methods except YW76 and display the frequencies of how many times each method had the best performance out of the 1312 experiments. Table 5 is with respect to the forecast error and table 6 is regarding the Weighted-Kappa metric. Almost two thirds of all of the experiments resulted in the Seeded Sparse-VARGA performing the best. The next best was the Hill-Climbing method, with a total number of better performances of about 20%. Surprisingly, the non-padded version of Sparse-VARGA (SVNP) was next in the rankings, with about 6.7% of the total number of experiments. The 17 cases where five of the methods drew correspond to 17 sets of MTS where all the visual field values are zero. With such a time series, the Yule-Walker equations will fail – see the section on **Application Independent Results** for a detailed discussion. The Weighted-Kappa results as indicated in table 6 again show that the SSV method comes out on top, with about 57% of the highest values. It also comes equal second with the YW16 method, with a total of about 15%. This means that the SSV method is either the best or equal best in about 72% of the cases.

It is easy to see how the YW16 method may better the HC, SVP and SVNP methods with respect to the Weighted-Kappa metric, even though it is a seed to some of these methods. This is due to a low forecast error not necessarily meaning that the change is in the right direction. Forecast error is computed as a deviation from the observed, with no consideration of sign. However in table 5 it can be clearly seen that the YW16 method has the best fitness (forecast error) in 43 cases (approximately 3.2%). This should be impossible since the YW16 method should always be equal or worse than the SSV, SVNP, SSP and HC methods because of seeding and elitism. Note that the HC method uses a form of elitism: there is a population of one, no crossover and a one-gene mutation producing one offspring. When these 43 cases are examined it can be seen that the difference in method fitness is 10^{-10} or lower; i.e. the Yule-Walker equations provided an almost perfect fit. With the four stochastic methods (that should have performed at least as well as the YW16 method), each is averaged over 10 runs. This combined with scaling the error as in equation 7 above, results in some cases having a small rounding error. In fact the results from these methods are all the same as expected, when they are compared to a very large number of decimal places.

Results by Group

In this section, the forecast error and Weighted-Kappa results will be examined by nerve fibre bundle. However, only the mean will be used as a summary statistic, because since there are 16 MTS groups, tables 3 to 6 and figures 5 and 6 would have to be repeated 16 times. In tables 7, 8 and 9, the shaded row represents the method with the best performance for the metric in question.

Group	Method					
	HC	SSV	SVNP	SVP	NOISE	YW16
0	1.333	1.317	1.614	1.851	3.207	1.885
1	0.431	0.422	1.250	1.413	2.708	0.578
2	0.411	0.399	1.213	1.445	2.727	0.623
3	0.761	0.752	1.306	1.557	2.718	0.904
4	0.458	0.453	1.332	1.563	2.885	0.751
5	0.402	0.410	1.597	1.837	3.095	0.948
6	0.428	0.422	1.631	1.873	3.395	0.963
7	0.567	0.561	1.546	1.849	3.442	0.832
8	0.667	0.655	1.391	1.650	3.030	0.872
9	0.731	0.721	1.517	1.765	3.058	0.909
10	0.384	0.371	1.372	1.655	2.933	0.627
11	0.314	0.307	1.409	1.673	2.851	0.670
12	0.418	0.430	1.473	1.734	2.809	0.881
13	0.435	0.428	1.328	1.595	2.904	0.677
14	0.744	0.738	1.329	1.577	2.741	0.910
15	0.512	0.510	1.187	1.417	2.530	0.666

Table 7: Mean Forecast Error by Group

In table 7 it can be clearly seen that the SSV method produces the lowest forecast error. However the SSV method is followed very closely by the HC method, which is better than SSV in 2 out of 16 groups. However the difference in forecast error in these two cases is very small. Table 8 splits the results in table 7 into groups and then displays the frequency that a method performed best or equal best (in terms of forecast error). The aim here is to show how many times a particular method outperforms the others, for example within table 7, the row HC=SSV is a count of the number of time the HC method and the SSV method equally outperform all of the other methods. It can then be seen that the method that comes top for all of the groups is the SSV method followed by HC, with SSV achieving over double the number of best forecasts than HC for all nerve fibre bundles. Overall the HC method only performs the best in under a third of the cases.

Methods	**Groups (Nerve fibre Bundles)**																**Totals**
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
HC	16	17	10	21	17	24	17	12	16	15	12	14	31	10	16	15	263
HC=SSV	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2
HC=SSV=SVNOPAD=SVPAD=NOISE	2	0	4	0	0	4	3	2	1	0	0	0	0	0	0	1	17
HC=SSV=YW16	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
HC=YW16	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
SSV	44	58	50	46	52	48	56	58	57	64	65	63	43	61	52	49	866
SSV=YW16	0	0	1	0	1	0	0	0	0	0	0	1	0	0	2	2	7
SVNOPAD	7	0	2	5	1	0	1	0	0	2	0	0	0	1	2	3	24
SVPAD	13	4	8	6	7	4	5	5	3	0	3	2	5	8	7	8	88
YW16	0	2	5	4	4	2	0	5	5	0	2	2	3	2	3	4	43
																	1312

Table 8: Method Forecast Error (Fitness) Rank by Group (Nerve fibre Bundle)

Whilst considering these tables, it is worth commenting about the previous versions of VARGA. Both versions were only run on nerve fibre bundle 12, and the mean forecast error was approximately (a slightly different measure was used in these papers) 2.2 for the version in [Swift 1999a] and 1.1 in [Swift 2002a], hence performing worse than the SV and HC methods.

Methods	**Groups (Nerve fibre Bundles)**																**Totals**
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
HC	2	2	3	5	5	13	8	3	1	1	0	3	17	4	4	1	72
HC=SSV=SVNOPAD=SVPAD=NOISE	2	0	4	0	0	4	3	2	1	0	0	0	0	0	0	1	17
HC=SSV=SVNOPAD=SVPAD=NOISE =YW16	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	1	5
HC=SSV=SVPAD	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
NOISE	2	0	1	1	2	1	0	0	0	0	0	0	0	0	0	0	7
NOISE=YW16	3	0	1	1	0	2	0	0	1	0	0	0	0	0	0	0	8
SSV	32	47	43	41	51	49	58	49	37	38	57	62	51	53	39	49	756
SSV=SVNOPAD	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
SSV=SVNOPAD=SVPAD	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2
SSV=SVNOPAD=SVPAD=NOISE	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	4
SSV=SVNOPAD=SVPAD=NOISE =YW16	0	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	4
SSV=SVNOPAD=SVPAD=YW16	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
SSV=SVPAD=YW16	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
SSV=YW16	3	16	12	17	11	6	5	14	20	24	12	11	7	10	18	14	200
SVNOPAD	13	2	2	5	3	0	1	0	0	2	0	0	1	2	2	3	36
SVPAD	10	1	5	2	1	2	2	2	3	1	0	0	0	2	6	2	39
YW16	13	14	9	7	6	4	5	12	18	15	11	5	6	10	12	11	158
																	1312

Table 9: Method Weighted-Kappa Rank by Group (Nerve fibre Bundle)

Table 9 is the same format as table 8, but the frequencies are given for the Weighted-Kappa metric. Within table 9 it can be seen that in all of the cases (groups), the SSV method performs the best (or equal best), i.e. for each group, it has the best Weighted-Kappa value out of the 82 patients. None of the other methods come near to its performance.

Group		**Max.**	**SSV**		**SVP**		**SVP**		**YW16**
ID	**n**		**Best**	**Average**	**Best**	**Average**	**Best**	**Average**	
0	2	20	15.3	15.0	11.4	9.8	11.2	9.4	19.1
1	4	80	74.0	74.2	25.5	19.4	24.9	18.6	63.4
2	4	80	67.2	67.2	23.0	18.1	22.5	17.2	62.6
3	3	45	37.0	36.9	18.6	14.7	18.3	14.2	41.8
4	5	125	98.3	98.3	26.2	19.8	25.7	19.1	81.9
5	9	405	330.2	330.4	43.2	33.1	39.9	29.3	155.9
6	7	245	204.7	204.8	34.4	26.2	33.1	24.5	128.7
7	4	80	73.7	74.0	25.6	19.8	25.1	19.1	61.5
8	3	45	42.9	42.9	21.7	16.9	21.3	16.3	41.6
9	3	45	43.3	43.3	20.8	16.1	20.7	15.8	41.1
10	4	80	77.2	77.6	26.0	19.6	25.7	19.1	64.2
11	7	245	236.5	237.2	35.3	25.7	34.8	25.0	119.4
12	9	405	360.9	360.2	38.2	27.3	37.9	26.7	163.1
13	5	125	104.7	105.1	28.0	20.9	27.4	20.1	83.2
14	3	45	39.1	39.1	19.8	15.5	19.6	15.1	41.0
15	4	80	71.9	72.1	24.5	18.8	24.1	18.2	64.2
Average:		100.0%	87.8%	87.8%	30.3%	23.6%	29.7%	22.8%	73.1%

Table 10: Method Model Size

Table 10 shows the average chromosome size for the methods. In this table the chromosome sizes for the SV, SVP, SVNP and YW16 methods are listed. *Group* is the nerve fibre bundle that the size pertains to, *Max.* is the maximum possible chromosome size, *Best* is the size of the fittest chromosome, averaged over all of the experiment for a particular group, and *Average* corresponds to the average of the populations average size over each generation. Note that the maximum chromosome size (*Max.*) is the size of the HC method. For the YW16 method, the size is computed by multiplying the average order by the number of variables in the MTS squared, n^2. Note that the noise model has a size of zero.

It can be seen that the best size for the Sparse-*VARGA* methods is usually smaller than the average size. The models in descending order of size are as follows: HC, YW16, SSV, SVP and SVNP. Note that technically the noise model is the smallest. The implications of the size will be discussed in later on.

Method	**Count of 1st Rank**	
	Forecast	**Weighted-Kappa**
HC	11 (13.5%)	1 (1.2%)
NOISE	0 (0.0%)	0 (0.0%)
SSV	69 (84.1%)	80 (97.6%)
SVNP	0 (0.0%)	0 (0.0%)
SVP	0 (0.0%)	0 (0.0%)
YW16	0 (0.0%)	0 (0.0%)
YW76	2 (2.4%)	1 (1.2%)

Table 11: Summary of Method Ranking by Patient

Results by Patient

The final analysis of the results is from a patient perspective. The results are averaged by patient, giving 82 sets. The column *Best* refers to which method provided the best average for a given patient. Any Yule-Walker equation estimate that could not be computed is given a penalising forecast error of 100.00 or a Weighted-Kappa of zero. Table 11 summarises these results, showing the number of times each method performs the best for each metric. It can be clearly seen that the SSV method performs best. However it is interesting to note that the YW76 method performs third best. This will be considered further in a later section.

Discussion

From the previous tables, one thing that stands out is that SSV is superior when rated on forecast accuracy and directional accuracy, although the HC method comes close in the former category. In the next section we will show that the SSV method has better computational complexity than the HC method.

Forecast Error (Fitness) Results. When looking at the forecast error results as a whole (table 3) it initially looks like there is not much difference between the HC and SSV methods. However it is only when these results are considered by looking at the frequencies of which performs best from a nerve fibre bundle (tables 7, 8 and 9) and a patient perspective (table 11), that it can be seen that the SSV method clearly performs better. This suggests that there are many low forecast error results, but a few very large error results for this method.

One unexpected result is that the YW76 method performs the best (when considered by patient with the other method) for 2 out of 82 patients (table 11). If all of the patients where the YW76 method could not provide an answer are ignored and the remaining patient's average point sensitivity (the average of all 76 points) is correlated with the point forecast error for the YW76 method using *Spearman's Rank Correlation Coefficient* [Spearman 1904], a value of –0.417 is obtained. This is significant to the 1% level, suggesting there is evidence to support that as the average point sensitivity increases, the YW76 point forecast error decreases. For the two cases where the YW76 method performs the best, the patients are ranked 13 and 23 out of the 82 in terms of average point sensitivity. This could be due to relationships existing between points in patients with better sensitivity that lie outside the nerve fibre bundle arrangements. The YW76 method could model these relationships whilst all of the other methods are restricted to modelling the visual field points according to the layout of each nerve fibre bundle.

Weighted-Kappa Results. The results for the SSV method are much better when comparing the directional agreement between observed and predicted deterioration using the Weighted-Kappa metric (tables 4 and 6). It is worth noting that this agreement is as important as forecast error, since the consequences of altering medication or performing surgery is quite severe. Most glaucoma clinicians make decisions based upon whether the visual fields are seen to be improving, deteriorating or remaining stable. Forecast error can give an indication of how accurate the forecasts are, but not if they were in the right direction of change.

Application Independent Results

In this section the results from performing complexity analysis of the methods and the performance issues arising from using the Yule-Walker equations are examined.

Complexity

No attempt to perform a formal analysis of algorithm efficiency will be made, but instead the focus will be on the most expensive part of all of the methods: the forecast error evaluation. The evaluation of the forecast error (fitness function) has a computation complexity in terms of arithmetic operations documented in table 12.

Method	**Complexity**	**Equation**
YW16, YW76	$(T - \text{MAXORDER}) \times \left(\text{O}(n) + \text{O}(n^2 lag) \right)$	(8)
SSV, SVP, SVNP	$(T - \text{MAXORDER}) \times \left(\text{O}(n) + \text{O}(\mathit{CHROMESIZE}) \right)$	(9)
HC	$(T - \text{MAXORDER}) \times \left(\text{O}(n) + \text{O}(n^2 \text{MAXORDER}) \right)$	(10)
NOISE	$(T - \text{MAXORDER}) \times \text{O}(n)$	(11)

Table 12: Forecast Error Computation Complexity by Method

In table 12, T is the length of the time series in question, MAXORDER is the maximum permissible order for a VAR process modelling the time series, O(·) is a function indicating the complexity of an operation, n is the number of variables in the time series, *lag* is the order of the VAR process being represented by a method and CHROMESIZE is the number of genes within a chromosome of Sparse-VARGA.

The term (T-MAXORDER) represents the number of sets of forecast error that is evaluated, O(n) represents the computations involved in taking the difference between observed and predicted error, and the terms concerning O(n^2…) involve the application of equation 3, i.e. computing the forecast.

Note that the Sparse-VARGA methods can have their fitness calculated in a lower computation time than the other methods because of the nature of their representation, i.e. sparse matrices (see [Swift 2002b]).

Table 12 puts all of the methods in perspective. Since the Yule-Walker equations (YW16 and YW76) and the noise model are used as seeds for the Sparse-*VARGA* based methods and the Hill-Climbing method, further analysis will only be done on the SSV, SVP, SVNP and HC methods. These four latter methods also iterate many times, whereas the three previous methods are deterministic. The relative complexity of the three Sparse-VARGA based methods will be compared with the Hill-Climbing based search. Since the term (T-MAXORDER) is common to all of the methods it can be ignored; also the term O(n) is less than any term involving O(n^2) and hence can also be ignored. This leaves only the term O(n^2MAXORDER) for the Hill-Climbing and the term O(CHROMESIZE) for the Sparse-VARGA based methods. If the average size figure from table 10 is used for CHROMESIZE then a graph can be drawn for the four methods for each nerve fibre bundle. Note that $1 \le \text{CHROMESIZE} \le n^2\text{MAXORDER}$ is always true. The Sparse-VARGA algorithm ensures there is at least one gene in a chromosome (but this may be zero valued) and that the *lag* part of any gene cannot exceed MAXORDER, therefore for an n-dimensional MTS, n^2MAXORDER is the maximum number of genes, i.e. a gene for each parameter of all the $n \times n$ parameter matrices.

Although the Sparse-VARGA family of applications appears to be less complex than the HC method, it would be interesting to see how they compare in practice. If the average chromosome length from table 10 is used for the value of CHROMESIZE, an indication of the relative complexity of the Sparse-VARGA and HC methods can be seen as in figure 7. In this figure, the HC method's size corresponds to the maximum size at all times, hence the relative complexity being equal to n^2MAXORDER, where n is dependent upon the dimensionality of each nerve fibre bundle. It is quite clear that the SVP and SVNP methods have a tendency to create small (sparse) models, and hence have a very fast and efficient forecast evaluation (less than 20% of the maximum). However the SSV method seems to have an average of about 87.8% the maximum. Note that the non-padded version of Sparse-VARGA (SVNP) produces slightly smaller models than the padded version (SVP), and also performs moderately worse (in terms of forecast error and Weighted-Kappa).

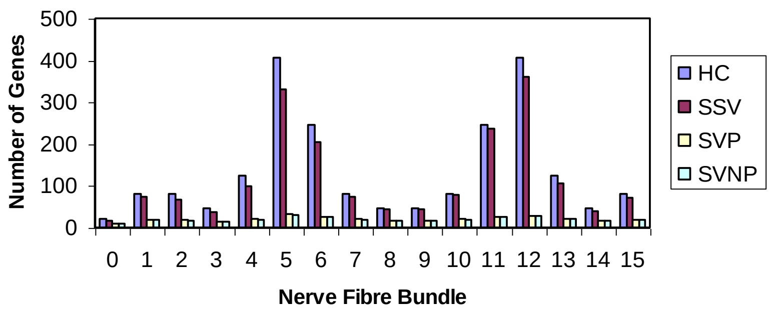

Figure 7: Relative Method Complexity

Finally, it is worth noting that the YW16 and YW76 methods produce models that are 100% populated, but the order is less than the maximum specified order. The SSV, SVP and SVNP methods produce models that are of the maximum order (they do not have to, but do so in most of the cases). Densely populated matrices mean that the Yule-Walker results are difficult to interpret, since there are too many parameters to visually inspect. This is not the case with the SVP and SVNP methods, since the parameter matrices (especially in higher dimensionality models) are highly sparse, thus relationships are easier to identify (this is true to some extent with the SSV method results). This could explain why the YW method can be improved upon by creating VAR subset models. These are models where some of the parameters of a VAR model are set to zero to improve the accuracy of the model. It has been acknowledged that the results of the statistical time series methods might not always return the desired results because techniques such as matrix inversion can be subject to inherent inaccuracies and biases, especially when the time series is short.

Performance of the Yule-Walker Equations

For short length MTS the Yule-Walker equations are often not solvable, for example in about 11% of the visual field cases (145 cases), i.e. the YW method could not find a solution for 145 multivariate time series over all permissible lags. For a matrix B to be invertible (a requirement for the Yule-Walker equations to be solvable):

i) The columns and rows of B are linearly independent.

ii) For a given vector b, $B\underline{x} = \underline{b}$ has a unique solution.

Note that other conditions not pertinent to this method are omitted; see [Stewart 1998] for a full list. Therefore the Yule-Walker equations produce either a set of equations that do not have a unique solution, or a matrix being inverted has two or more rows or columns that are linearly dependent. Note that with the visual field data, 17 out of the 145 cases are where all of the values in the time series are zero, hence it is not the case that the YW method's failures are due to the time series being completely zero, i.e. total lack of vision on part of the eye (a whole group). More investigation is needed to see why this is the case, but it is suggested that this is likely to happen when the time series is short and high dimensional. For example, with the visual field dataset, there are a large number of variables in nerve fibre bundle 5, which is where glaucoma usually originates. Hence there is a good chance that one or more of the variables will be mostly zero as the condition progresses (a blind spot developing). Hence a column of the intermediate matrices during the computation of the Yule-Walker equations could easily be zero. This will mean that condition (ii) above will be violated, since there will not be a unique solution (possibly none at all). The patients that the Yule-Walker equations fail on are those that have a lower average sensitivity. This is in cases where the condition is terminal, i.e. certain visual field points have a sensitivity of zero, indicating total blindness in part of the eye. The YW76 method fails whenever there is a failure with any associated YW16 method because a column of zeros in a nerve fibre bundle will be a column of zeros in the corresponding 76 variable MTS.

Further Evaluation Using Synthetic Datasets

One of the most challenging aspects of modelling short MTS is that there is often not enough data to be split between test and training sets. In this situation every single observation is needed towards accurate model building. This is particularly true with the visual field dataset, which varies in size between 10 and 44 observations. In order to evaluate Sparse-VARGA's performance on a test set of data, this section uses a novel method for generating stable synthetic VAR processes, thus allowing enough data to be generated so that Sparse-VARGA can be tested on out of sample data. Note that this technique will generate a stable VAR process of a given dimensionality and order, and that no other information is specified. Hence the specific details of the resultant process, e.g. the strength of the dependencies between variables in the process, are determined by chance.

This dataset consisted of nine VAR processes whose order and parameters are known, and have been used to generate MTS data. Each VAR process had a noise process associated with it, which consisted of the appropriate number of independent identically distributed uniformly distributed random numbers between ± 2.5. The process of constructing the parameter matrices for a synthetic VAR process is described in [Swift 2002b].

Evaluation

Table 13 describes the synthetic datasets. Each series was 100 time points in length. The first 10 are ignored (since the beginning *lag* observations are random noise), the next 50 were used for training, and the remaining 40 points were used for the test dataset. Seeded Sparse-VARGA was run ten times on each of the nine subsets, and the results averaged.

Name	2var5	3var5	4var4	5var3	6var2	7var5	8var4	9var3	10var2
Dimensionality	2	3	4	5	6	7	8	9	10
Order	5	5	4	3	2	5	4	3	2

Table 13: Synthetic Dataset Composition

Patient ID	**Training Data**				**Test Data**			
	Mean	**Median**	**StDev.**	**WK**	**Mean**	**Median**	**StDev.**	**WK**
2var5	1.262	0.955	1.150	0.892	3.010	2.400	2.484	0.943
3var5	1.695	1.314	1.542	0.882	1.994	1.877	1.356	0.560
4var4	1.072	0.886	0.953	0.869	1.894	1.554	1.543	0.914
5var3	1.382	1.066	1.294	0.958	1.533	1.278	1.146	0.824
6var2	1.825	1.276	1.946	0.716	1.883	1.615	1.385	1.000
7var5	1.623	1.117	1.731	0.974	4.469	3.834	3.473	0.929
8var4	1.218	0.785	1.331	0.878	2.513	2.031	2.046	1.000
9var3	1.875	1.165	2.080	0.850	2.336	1.884	1.997	0.735
10var2	1.719	0.997	1.979	0.800	2.706	2.183	2.189	0.849
Average:	1.519	1.062	1.556	0.869	2.482	2.073	1.958	0.862

Table 14: Results for Ten Patient Subset

Table 14 below is a short summary of the results of short-term forecasting after applying Seeded Sparse-VARGA to both test and training datasets. The columns *Mean, Median* and *StDev*. have their usual meaning and apply to the forecast error, and WK is the Weighted-Kappa metric applied to the directional accuracy of each point forecast. From table 14, it can be seen that the mean error on the training data lies in the range of (1.0, 1.9) and (1.5, 4.5) for the test data. The overall average forecast error for the training data was approximately 1.5 and approximately 2.5 for the test data. This is very promising when considering the magnitude of the noise was in the region of approximately 1.3. This means that the training error was only 15.4% larger than the theoretical best that could be achieved. The test error is less than twice the magnitude of the noise, which is reasonable considering the short length of the time series. In nearly all of the cases (for both test and training datasets), the median error is less than the mean error, suggesting that there might be some large magnitude forecast errors. This is supported by noting that the standard deviation is quite large in most cases, almost equal to the mean in many cases. Finally it is worth noting that the Weighted-Kappa score for the training and test datasets are approximately the same, with almost no change in overall average. In fact for two of the test datasets (shaded in grey) the best metric score is achieved.

Concluding Remarks

Modelling short multivariate time series is a challenging task, and there has been little work in the area. In this paper we have suggested a novel computational method that overcomes difficulties associated with our earlier attempts. We have shown how evolutionary computation techniques utilising sparse matrix representation can significantly extend the capabilities of traditional statistical methods for modelling short MTS. We have provided concrete evidence that useful models can be learned from short MTS using a combination of statistical time series modelling, sparse matrix computation, and intelligent search techniques. In particular, we have presented a method called Sparse-VARGA that improves upon the Yule-Walker method and works where this method cannot. It has been shown that by integrating the order selection process into the whole procedure, better short-term forecasts can be produced, and the resultant parameter matrices are a better aid to explaining relationships between variables.

Sparse-VARGA is a novel method in that it views a VAR(p) process not as a series of densely populated matrices, but as a collection of parameters that are spread over a series of zero matrices from order p=1 to some maximum order, p=MAXORDER. The advantage of this approach is that there is scope for handling much more complicated types of the VAR process family. For example, Sparse-VARGA can identify VAR subset models and models where some of the parameter matrices are zero, e.g. a VAR(3) process where only the first and third parameter matrices are populated. Additionally Sparse-VARGA introduces some new operators that exploit this representation. This approach has advantages over simply storing a VAR process as a series of matrices, as in [Swift 2002a]. That representation had the problem of having to store a vast number of parameters as the dimensionality of the time series increases, along with the previously mentioned order bias problems. The sparse matrix representation of Sparse-VARGA has the additional advantage of having a fitness function where the computational complexity is proportional to the number of genes in the chromosome.

The extensive results show that Sparse-VARGA performs better than the comparative methods with regards to *forecasting accuracy* and *direction of change*. In addition, the method has better computational complexity than the Hill-Climbing method. Therefore

Sparse-VARGA appears to be an appropriate method for modelling short length high dimensional multivariate time series, with good results on the visual field dataset. The results on synthetic data have been shown to have a low forecast error, which is consistent across both training and test datasets. There are many other datasets that are both short in length and high in dimensionality to which this approach could be applied. For example work could be carried out to see how the methods presented in this paper can be applied to the analysis of DNA microarray data, basically a short MTS dataset.

Finally, the method can be improved in the following areas:

i) Seeding techniques and strategies will be further studied to try to produce models with a smaller number of parameters.

ii) Attempts will be made to extending this work by developing spatio-temporal models [Pfeifer 1980a, Pfeifer 1980b]. Such models applied to visual field data has been looked into in [Tucker et al. 2005].

iii) In order to improve the effectiveness of the *Shuffle* operator, it could be adapted to mimic the Blend Crossover (BLX) operators' way of dealing with real numbers [Eshelman 1993].

iv) The *Mutation* operator could be improved by implementing an adaptive mutation rate such as in [Davis 1989], allowing a changeable mutation standard deviation.

v) Other techniques to improve performance such as *Niche Methods* [Mahfoud 1995] could also be looked into.

vi) A more rigorous complexity analysis could be conducted along with analysing and comparing the actual run times of all of the experiments.

Acknowledgements

We thank our research partners at Moorfields Eye Hospital and the Institute of Ophthalmology for their advice and the MTS visual field data. We are grateful to Dr Allan Tucker, Dr Steve Counsell and Dr Jason Crampton for their advice and assistance. We are also grateful to the reviewers for their constructive and helpful comments. This research was funded by Moorfields Eye Hospital, London; the Engineering and Physical Sciences Research Council, UK and the Biotechnology and Biological Sciences Research Council, UK.

References

[Akaike 1974] Akaike H (1974), A New Look at the Statistical Model Identification, IEEE Transactions on Automatic Control, AC-19(6), 716-723

[Altman 1997] Altman DG. (1997), Practical Statistics for Medical Research, Chapman and Hall

[Armstrong1992] Armstrong JS and Collopy F (1992), Error Measures For Generalizing About Forecasting Methods: Empirical Comparisons, International Journal of Forecasting, 8, 69-80

[Baker 1985] Baker JE (1985), Adaptive Selection Methods for Genetic Algorithms, Proceedings of the First International Conference on Genetic Algorithms, 101-111, Lawrence Erlbaum Associates

[Bearse 1998] Bearse PM and Bozdogan H (1998), Subset Selection in Vector Autoregressive Models Using the Genetic Algorithm with Informational Complexity as the Fitness Function, Systems Analysis, Modelling, and Simulation (SAMS), 31, 61-91

[Casdagli 1992] Casdagli M and Eubank S (1992), Nonlinear Modeling and Forecasting, Addison Wesley

[Chatfield 1995] Chatfield C (1995) Model Uncertainty, Data Mining and Statistical Inference (with discussion), Journal of the Royal Statistical Society, Series A, 158, 419-466

[Crabb et al. 1996/97]	Crabb D, Fitzke F, McNaught A and Hitchings R (1996/1997), A Profile of the Spatial Dependence of Pointwise Sensitivity Across The Glaucomatous Visual Field, Perimetry Update, 301-310
[Davis 1989]	Davis L (1989), Adapting operator probabilities in genetic algorithms, ICG89: Proceedings of the 3rd International Conference on Genetic Algorithms, 60-69, Morgan Kaufmann
[Dejong 1975]	De Jong KA (1975), An Analysis of the Behaviour of a Class of Genetic Adaptive Systems”. PhD Thesis: University of Michigan, Dissertational Abstracts International, 36:10, 5140B
[Eshelman 1993]	Eshelman LJ and Schaffer JD (1993), Real-Coded genetic Algorithms and Interval-Schemata, Editor: Whitley LD, Foundations of Genetic Algorithms 2, 187-202, Morgan Kaufmann
[Goldberg 1989]	Goldberg DE (1989), Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley
[Goldberg 1990]	Goldberg DE (1990), Real-Coded Genetic Algorithms, Virtual Alphabets, and Blocking, Technical Report no. 90001, University of Illinois at Urbana-Champaign
[Haley 1987]	Haley M J (editor) (1987), The Field Analyzer Primer, Allergan Humphrey
[Hand 1994]	Hand DJ (1994), Deconstructing Statistical Questions (with discussion), Journal of the Royal Statistical Society, Series A, 157, 317-356
[Heijl et al. 1988/89]	Heijl A, Lindgren A and Lindgren G (1988/1989), Inter-Point Correlations of Deviations of Threshold Values in Normal and Glaucomatous Visual Fields, Perimetry Update, 177-183
[Hitchings 2000]	Hitchings RA (2000), Glaucoma, BMJ Publishing Group
[Holden 1995]	Holden K (1995), Vector Autoregression Modelling and Forecasting, Journal of Forecasting: Special issues on the Vector Autoregressive Model, 14, 159-166
[Holland 1975]	Holland JH (1975), Adaptation in Natural and Artificial Systems, Ann Arbor, MI: The University of Michigan Press

[Kadous 1999] Kadous M (1999), Learning Comprehensive Descriptions of Multivariate Time Series, Proceedings of the International Conference on Machine Learning, 454-463

[Lockhart 2000] Lockhart B and Winzeler E (2000), Genomics, Gene Expression and DNA Arrays, Nature, 405, 827-836

[Lütkepohl 1993] Lütkepohl H (1993), Introduction to Multivariate Time Series Analysis. Springer-Verlag

[Mahfoud 1995] Mahfoud SW (1995), Niching methods for Genetic Algorithms, University of Illinois at Urbana-Champaign, Technical Report, no. 90001

[Meir 2000] Meir R (2000), Nonparametric Time Series Prediction Through Adaptive Model Selection, Machine Learning, 39:1, 5-34

[Michalewicz 1996] Michalewicz Z (1996), Genetic Algorithms + Data Structures = Evolution Programs, Springer, 3rd edition

[Oates et al. 1999] Oates T, Schmill M and Cohen P (1999), Efficient Mining of Statistical Dependencies, Proceedings of the 16th IJCAI, 794-799

[Pfeifer 1980a] Pfeifer EP and Deutsch SJ (1980), A Three-Stage Iterative Procedure for Space-Time Modeling, Technometrics, 22:1, 35-47

[Pfeifer 1980b] Pfeifer EP and Deutsch SJ (1980), Identification and Interpretation of First Order Space-Time ARMA Models, Technometrics, 22:3, 397-408

[Pole et al. 1994] Pole A, West M and Harrison PJ (1994), Applied Bayesian Forecasting and Time Series Analysis, Chapman-Hall

[Russell 1995] Russell S and Norvig P (1995), Artificial Intelligence, A Modern Approach, Prentice Hall, 111-112

[Shahar 1997] Shahar Y (1997), A Framework for Knowledge-Based Temporal Abstraction, Artificial Intelligence, 90, 79-133

[Sharif 1998] Sharif AM and Barrett AN (1998), Seeding a Genetic Population for Mesh Optimisation and Evaluation, Genetic Programming 1998 Conference: Late Breaking Papers, 195-200

[Snedecor 1967]	Snedecor G and Cochran W (1967), Statistical Methods, Iowa State University Press, 6th edition
[Spearman 1904]	Spearman C (1904), The Proof and Measurement of Association Between Two Things, The American Journal of Psychology, 15, 73-101
[Stewart 1998]	Stewart GW (1998), Matrix Algorithms Volume 1, Basic Decompositions, Philadelphia: Society for Industrial and Applied Mathematics
[Swift 1999a]	Swift S and Liu X (1999), Modelling and Forecasting of Glaucomatous Visual Fields Using Genetic Algorithms, GECCO99: Proceedings of the Genetic and Evolutionary Computation Conference, 1731-1737, Morgan Kaufmann
[Swift 2002a]	Swift S and Liu X (2002), Predicting Glaucomatous Visual Field Deterioration Through Short Multivariate Time Series Modelling, Artificial Intelligence in Medicine, 24(1), 5-24
[Swift 2002b]	Swift S (2002), The Modelling of Short High-Dimensional Multivariate Time Series, PhD Thesis: University of London, London, UK, 2002
[Swift et al. 1999b]	Swift S, Tucker A and Liu X (1999), Evolutionary Computation to Search for Strongly Correlated Variables in High-Dimensional Time-Series, Proceedings of Intelligent Data Analysis 99, 51-62, Springer-Verlag
[Swift et al. 2004]	Swift S, Tucker A, Liu X, Martin N, Orengo C and Kellam P (2004) Consensus Clustering and Functional Interpretation of Gene Expression Data, Genome Biology, 5(11), pp. R94.1-R94.16
[Syswerda 1989]	Syswerda G (1989), Uniform Crossover in Genetic Algorithms, Proceedings of the Third International Conference on Genetic Algorithms, 10-19, Morgan Kaufmann
[Tucker 1999]	Tucker A and Liu X (1999) Extending Evolutionary Programming to the Learning of Dynamic Bayesian Networks, GECCO99: Proceedings of the Genetic and Evolutionary Computation Conference, 923-929, Morgan Kaufmann
[Tucker et al. 2005]	Tucker A, Vinciotti V, Liu X Garway-Heath D (2005) A Spatio-Temporal Bayesian Network Classifier for Understanding Visual Field Deterioration, Artificial Intelligence in Medicine, 34(2), 163-177

[Weigend 1994] Weigend AS and Garshenfeld NA (1994) Time Series Prediction, Addison-Wesley

[Whittle 1984] Whittle P (1984) Prediction and Regulation, Basil Blackwell, 2nd edition

[Zlatev 1991] Zlatev Z (1991) Computational Methods for General Sparse Matrices, Kluwer Academic Publishers