*Journal of Computational Intelligence in Finance* **5:**6 (1997) 17-24

# Multivariate Embedding Methods: Forecasting High-Frequency Financial Data in the First INFFC

*Carol Alexander and Ian Giblin*
*School of Mathematical and Physical Sciences,*
*University of Sussex, Falmer,*
*Brighton, Sussex BN1 9QH.*
*United Kingdom*

**Abstract**

Forecasting software is described, where each point to be forecast is embedded in an m-dimensional library made from historic data. The approach is based on the well-known 'nearest neighbour' algorithm of Casdagli (1989) but there are important differences, including the facility for multivariate embedding, the use of predictor variables which may be different from the embedding variables, and the 'rolling library' which is of a constant size but is continuously updated as each successive point is forecast. In this way the univariate Casdagli algorithm has been developed into a more sophisticated 'pattern recognition' technique for short-term forecasting, whilst augmenting its original purpose of searching for evidence of chaos in time series.

Because each point to be forecast has its own parameter estimates a certain amount of variability between successive forecasts is to be expected. However it was interesting to find that forecasts made on the training data were in fact exceptionally smooth over certain periods so that for some time (not usually longer than a few days) all points fell within similar close point groups. On the other hand there were other, shorter periods (typically a few hours long) where forecasts became 'chaotic', because adjacent points fell into totally different areas of the library. Hence a by-product of our work for the INFFC has been to provide empirical evidence of the local stability results of Yao and Tong (1994).

## 1. Introduction

Univariate embedding methods for time series prediction have been well known for some time. Initially applied to the detection of chaotic attractors in such time series as fluid turbulence (Casdagli, 1992) or measles incidence (Sugihara and May, 1990), much recent research has focused on their ability to distinguish stochastic noise from deterministic chaotic dynamics (see for example Nychka et. al., 1992 and Smith, 1992). This has inspired applications of these methods to financial time series but the evidence in the literature, such as it is, claims both for and against the existence of chaotic systems underlying financial markets (for reviews see Alexander and Giblin, 1994 and LeBaron, 1994).

Our forecasting method for the first INFFC, which uses one of a class of *multivariate* embedding prediction methods, has been developed out of our work on the detection of deterministic chaos underlying noisy financial time series[1]. Prediction methods detect chaos when short-term forecasts are more accurate than long-term forecasts, so it is a straightforward generalisation of these algorithms to optimise their prediction accuracy over the short term, as we did. But why the generalisation to multivariate embeddings? Many financial time series have unit roots, so univariate prediction methods are difficult to develop and unlikely to be as accurate as multivariate methods based on common features in time series[2]. Common features analysis will not necessarily lead to any new insights into the relationships between variables, but it does

[1]Financial support from the ASARCO is gratefully acknowledged.

[2]This area of time series analysis, pioneered by Rob Engle (Engle and Granger, 1987, Engle and Koziki, 1993) has received enormous attention from academic econometricians and by comparison, applications to financial markets are relatively few (but see Alexander and Johnson (1992), Alexander (1994a, 1994b) and Alexander (1995)).

provide a rigorous modelling framework. We have, therefore, adhered to its principles whenever possible[3].

In contrast to common features, Technical Analysis (TA) is an area of financial time series analysis which has been and continues to be of widespread practical use in financial markets (see Brock et. al. (1992), Curcio and Goodhart (1992), LeBaron (1991)). Therefore, when it appears appropriate, we have tried to incorporate some of the most common TA strategies in our pre-processing of data. Also we have been mindful of certain simple technical trading rules in the construction of appropriate embedding spaces. For example, some of the usual 'stochastics' (i.e. moving averages) have been assumed, rather than arrived at through training the model.

## 2. Description of the model

We have developed a command based program which employs a variety of embedding methods for time-series forecasting. Certain features are common to all methods:

- the parameterization of the forecasting model is the same for all data points; however, the model parameters are estimated individually for each data point to be forecast.
- algorithms may require data pre-processing which includes normalisation of variables to have equivalent unconditional means and standard deviations;
- parameter estimation is based on a 'library' of historic data which is embedded in m-dimensional Euclidean space;

[3]In particular we have attempted to 'balance' the relationships modelled (see Granger and Terasvirta,1993) so, for example, only modelling stationary variables with other stationary variables.

- a relatively small subset of nearest neighbours called the 'close point group' is found in this embedding space and parameter estimates are based on this subset - see figure 1;
- the parameter estimates so obtained are then used to compute forecasts.

*2.1 The forecasting method*

The algorithm proposed by Casdagli (1992) is based on a univariate linear model of the form

$$x_{t+i} = a_0 + \sum_{j=1}^{m} a_j x_{t-k(j-1)} + e_t \qquad (1)$$

where m is the embedding dimension and k is a fixed positive integer. Each point to be forecast has its own parameter estimates $\hat{a}_0, \hat{a}_1, ....., \hat{a}_m$ which are obtained as follows:

- The point $x_t$ is embedded in m-space as $\mathbf{x}_t = (x_t, x_{t-k}, x_{t-2k}, ...., x_{t-(m-1)k})$ where the lag k is chosen as small as possible, but sufficiently large to minimise autocorrelation between successive points in this embedding;
- Amongst the points in the library, the n nearest neighbours to $\mathbf{x}_t$ are found;
- A parametric regression of the form (1) is estimated using these n nearest neighbours[4];
- The i-step ahead forecast of $x_t$ is then calculated as

$$\hat{x}_{t+i} = \hat{a}_0 + \sum_{j=1}^{m} \hat{a}_j x_{t-k(j-1)} \qquad (2)$$

[4]Note that $x_{t+i}$ must be known, so we must exclude any points near the end of the library if these are found to be amongst the nearest neighbours.

These are the *principles* which underlie the most important command in our script-based submission (taking the form 'forecast [steps ahead] Casdagli'), but there are important differences: firstly, the linear model (1) does not need to be univariate, and secondly neither does it need to correspond exactly to the embedding. In fact we allow the regressors in (1) to be any predetermined variables, and we also allow these to be different from the embedded variables. This is because the embedding and the forecasting model perform different roles in our analysis: the embedding space is used to detect similar patterns in historic data to the current pattern that we wish to forecast. In effect it is a sophisticated *regime selector* . Other nearest neighbour algorithms used in our software include the exponentially weighted[5] and barycentric co-ordinate methods[6]. These are non-parametric, non-linear methods which are, however, only appropriate for stationary data[7] .

*2.2 Embedding for pattern recognition.*

Howel Tong (1990) and Yao and Tong (1992) consider the local stability properties of multivariate time series, to determine whether there are regions where the different trajectories converge and others where they diverge. This idea has been further explored by LeBaron (1994), who investigates different regimes in the trading volume / volatility relationship. Our multivariate nearest neighbour algorithms extend this idea, by using the embedding to select a wide variety of regimes. For example, with the INFFC data set we may look for the degree of autocorrelation in

---

[5]Adapted from the simplex method of Sugihara and May, 1990.

[6]See Alexander and Giblin, 1994.

[7]They work by locating the point to be forecast in the close point group; the former by exponentially weighted distances from points in a close point group and the latter by the barycentric co-ordinates of the point within the smallest simplex which contains it. These library points (close point group, or simplex) are then projected forward i steps, and the same position relative to these new points is taken as the i step-ahead forecast.

returns; evidence of support or resistance levels in the price; different behaviour at different times of day; and so on. A full list of potential embedding variables, with reasons for including them follows:

- lagged returns - to pick out similarities in the autocorrelation structure of returns. In some periods there is substantial autocorrelation, in others much less. With lagged returns in our embedding, if our point to predict is in a highly autocorrelated regime, only highly autocorrelated returns will be used in its prediction;

- (smoothed) volume and range data - so that the points in the library used to forecast our point will lie in the same high/medium/low volume/volatility regimes;

- current price levels - by selecting historic points at similar price levels, information about historic support or resistance levels can be captured in the nearest neighbours used to forecast our current point;

- a deterministic time of day variable ('td'), taking the values 1/270 (10:31) to 1 (15:00). With this in the embedding close points are likely to be chosen from library points at similar times of day. This may be useful for the 120 minute forecasts since, for example, points towards the end of the day will fall into balls of points which are also at the end of the day, so this will pick up patterns in overnight variations. However, time of day as an embedding variable will have little effect upon the 1 day forecasts.

- transforms of 'stochastics' - technical analysts look at relationships between moving averages of different lengths to signal trading opportunities. A common indicator is the relationship between the 4-day, 9-day and 18-day stochastics (see Murphy, 1986) but for intra-day data it is often the 14-point vs 3-point moving average that indicates a trade. For example a 'buy' signal occurs when the 14-point crosses the 3-point from below. Hence we use the difference

between the 14-point and the 3-point moving averages of the close price as a potential embedding variable.

## 3. Implementation of the program

### *3.1 System architecture and operation*

The program has been entirely written in ANSI 'C' in order to provide compatibility with the platforms available to us. We have used an Acorn RiscPC and A5000 for the majority of software development and debugging, whilst Sun and Solbourne workstations were used for computationally intensive runs as well as the required timing tests[8].

Whilst the majority of 'core' functions had already been written in some form before this competition, it was necessary to incorporate a 'front end' which would allow flexibility in loading and processing data as well as more extensive testing without the need to re-compile the code. Originally a single Unix-style command line (with as many as twenty parameters) was used, but this has now been replaced with a dedicated command interpreter and prompt, allowing interactive use as well as automatic loading of successive command lines from 'scripts'. These scripts are simple in format, executed in a strictly linear fashion and have no facility for recursion or other features associated with more powerful interpreted languages. However, we have also written a variety of very short programs in Basic and C which themselves write scripts to use with the program. A mere ten lines of Basic are sufficient to produce a script which applies the forecasting algorithms 60 times (this figure is typical of our tests). By incorporating a simple record-keeping command into the interpreter, it was possible to automatically output the combined results of any number of tests across a wide range of parameters.

Each time series is stored as a separate array, these arrays only being initialised once the program is run. A single pointer is held for each series once the required memory has been allocated, and this system allows a very flexible implementation of multivariate embedding, by

[8]The suggested timing test over the second half of the training data took 2hrs (for both forecasts together) on a Sun Spark 5 workstation.

simply moving array pointers according to the choice of variable and lag. Four arrays of pointers allow us to specify the test and library data for both the embedding and predictor variables. The 'library' data sets are optional; where possible during testing we used the points immediately preceding our test set. These preceding points are of course unavailable for the initial points in each series, and a separate library is loaded and used when appropriate.

Our submission for the INFFC consists of the compiled program plus a script (of around 200 lines, of which perhaps 50 are comments) which we have developed to load 'unseen' data sets such as any interpolated section of the training data set. Although the program has been frequently re-compiled and altered, the use of this script has significantly simplified development and testing of the program. Several very specific commands have been added to the vocabulary of the interpreter in order to load test data and provide output files as required by this competition, as well as to account for unexpected peculiarities found in the training data, as explained in the next section.

*3.2 Dealing with data problems*

Some special features were necessary to cope with the long periods of zero trading volume in the INFFC data; a command of the form 'findbadpoints [series] m' was incorporated to do three things:

- it sets to zero the forecasted return corresponding to any points in the series which have not moved in the last m minutes[9] ; and
- it locates 'flat points', viz. points in the embedding space which coincide with the previous forecasted point. The 'findbadpoints' routine then short-cuts the forecasting procedure and sets the forecasts on 'flat points' to be the previous forecast; and

[9]We used the volume data for the series and m=120 or 270 respectively.

- it ignores all points in the embedding space which are at zero distance from the point to be forecast. Although these points represent, in principle, 'identical' points to our test, thus suggesting an optimal forecast, it was found during tests that sometimes the 'close point group' would consist *entirely* of points at zero distance, yielding a singular (hence useless) regression matrix.

Finally we note that since lags are necessary for our prediction algorithm, predictions of the first few points were set to the current close. Hence a forecast is provided for every point in the test data set, but the first two hundred or so point predictions are equivalent to a random walk .

**4. Training and Testing**

In this section we discuss data processing (pre-processing of inputs and post-processing of outputs), and describe the results of training and 'testing down' the embedding and regression variables.

The INFFC rules have placed substantial constraints on our choice of procedures to use in the competition. In particular, we have concentrated our training on only one of the embedding algorithms, described above. This algorithm is neither the fastest nor the slowest of the algorithms available, but was thought to be the most suitable for the competition since it requires less stationarity in the data generation processes to work effectively: It does not assume that relationships between nearby points in the embedding space will remain stable over time, an assumption which is central to the success of the other algorithms in our software. We have also used a constant library size, close point group size, embedding specification and choice of predictor variables for the entire data set. Yet although these parameters are constant, a different prediction equation for each point is obtained.

*4.1. Pre-processing*

We have based our 120- and 270- minute ahead forecasts on the 120- and 270-point returns series, viz.

$$x(i)_t = \ln(y_t) - \ln(y_{t-i}) \qquad (3)$$

where i = 120 or 270 and $y_t$ denotes the close price at time t. These returns forecasts are then translated into forecasts of the close using the inverse transform:

$$\hat{y}_{t+i} = \exp(\ln(y_t) + \hat{x}(i)_{t+i}) \qquad (4)$$

Although it is usual to assume that prices are generated by a random walk model, whilst financial returns may have stationary data generation processes, there is no evidence to suggest that the 120-minute and 1-day returns calculated from the INFFC training data are in fact stationary, hence the preference for the Casdagli algorithm that has already been mentioned.

Moving averages of different lengths were used to smooth the volume data. These moving averages were based on the simple technical trading rules described in section 2.2 and were used to see if the embedding of different TA signals helps improve efficiency of forecasts. Minimum RMSE searches over the length of moving average, holding constant the other parameters such as library and close point group sizes, and embedding and regression variables would have been appropriate. However it was not feasible to do an exhaustive search such as this, with all possible choices for these fixed parameters. Instead the limited specification tests performed pointed to smoothing the volume data with a 9-point moving average and/or a 27-point moving average.

*4.2 Choosing the optimal library and size of close point group.*

Forecasts on the INFFC training data turned out to be more accurate when we allowed the library of historic data to 'roll behind' the point to be forecast, rather than remain fixed at the beginning of the data set. This 'rolling library' facility has also increased the speed of computation

considerably, since the optimal library size can be fixed at lower level, and the searching for nearest neighbours is thus performed over a smaller set. We have therefore used a rolling library for specification searches over library and close point group sizes.

The perfect scenario, where surface plots of normalised RMSE for different group and library sizes is not yet fully automated in our software. Ideally we would have generated several surface plots of RMSEs versus library and close point group sizes, for different embedding and regression specifications. But figure 2 refers only to the embedding (i) below, which ex-post (i.e. *after* fixing library and group sizes) turned out to be optimal. On the basis of this we have selected an optimal library size of 2000 - 3000 and a close point group size of between 100 and 200 [10].

*4.3 Training the embedding*

With some surprize, we found that in our final specification tests, using lags of 120 and 240 in the autoregressive part of the embedding for 1-day returns was *more* efficient than using lags of 270 and 540. For example, the RMSE ratio to that of a random walk rose from 0.95961 (for lags of 120 and 240) to 0.98316 (for lags of 270 and 540)[11]. Hence this section on training the embedding only reports results using the lags of 120 and 240 for the 1-day returns.

Using the last 25,000 points in the training data, rolling libraries of 3000 points and optimal close point group sizes of 100, we have generated RMSE of forecasted returns summarised in table 1. The are so many permutations of potential embedding variables and regression variables that we've had to apply some selection in reporting results. To summarise the results which have *not* been reported, we have found that including any price level information in the embedding, such as close, open, or range (high-low) does not reduce RMSE of forecasts for

[10]Our final choice of these parameters for the competition were library size 2000 and close point group size 200. Although much of the training and testing used different values of these parameters, this did not affect the efficiency of our choices which were based on the *relative* magnitudes of RMSE.

[11]Over the second half of the training data, with a rolling library size 2000 and a close point group size of 50.

any regression specifications[12]. Had we used a larger library which is fixed (i.e. doesn't get dragged behind the point to be forecast), information on the levels may have been more useful in selecting the appropriate regime. But our choice of a relatively small 'rolling' library may in any case be proxying the use of levels or ranges in the embedding. Thus the results of table 1 are confined to searching over time-of-day and various moving averages of the volume data (the notation vn denotes an n-point moving average of the trading volume, td is the time-of-day dummy and x(i) are the i-period returns (i=120, 270))[13].

Minimising the RMSE implies using the embedding (i) for the 120-minute forecasts and (iv) for the 1-day forecasts. As expected, the time of day is only a significant variable for the 120-minute forecasts, but since the difference between the RMSEs of the 1-day forecasts in (i) and (iv) is so small, and since it is much quicker to use the same embedding for both 120-minute and 1-day forecasts, we decided to keep the time-of-day also in the embedding for the 1-day forecasts. Hence embedding (i) has been used for both.

[12]For example, adding a 9-point moving average of the range to the embedding (i) of table 1 increased the RMSEs but only very marginally: from .05598 to .055994 for the 120-minute forecasts, and from .082627 to .082634 for the 1-day forecasts. However adding the closing price to embedding (i) of table 1 increased the RMSEs more substantially - to .05847 for the 120-minute forecasts and to .086253 for the 1-day forecasts.

[13]An autoregressive forecasting regression has been used for these specification searches of the best embedding.

Table 1: RMSE of forecasts for different embeddings.

| Embedding | RMSE (120 min) x $10^2$ | RMSE (1 day) x $10^2$ |
|---|---|---|
| (i) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ $v27_t$ td | 5.5980 | 8.2627 |
| (ii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ td | 5.7367 | 8.6544 |
| (iii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v27_t$ td | 5.8268 | 8.7764 |
| (iv) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ $v27_t$ | 5.9022 | 8.2399 |
| (v) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ | 6.3801 | 9.0348 |
| (vi) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v27_t$ | 6.6947 | 9.1023 |
| (vii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ td | 7.2293 | 10.2843 |
| (viii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ | 8.0131 | 10.5031 |

*4.4 Training the regressors*

Training in our model does not just involve specification of an appropriate embedding. We also need to choose the most efficient regressors for the prediction equation (1). What are likely choices for the predictor variables in the construction of 120 minute and 1 day forecasts? Of course, all embedding variables are also potential regressors, since the fact that they define relevant patterns in the embedding space also means that they could be suitable predictors. So we employed a 'testing-down' procedure, starting with the optimal embedding specification (i) and

again looking at the RMSEs of 120-minute and 1-day forecasts for different predictor variables. The results are reported in table 2, where the same 25,000 points have been forecast as in table 1, and the embedding (i) of table 1 has been employed with a rolling library size of 2000, and we have used a close point group size of 50[14].

Table 2: RMSE of forecasts for different predictors.

| Predictors | RMSE (120 min) x $10^2$ | RMSE (1 day) x $10^2$ |
|---|---|---|
| (i) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ $v27_t$ td | 8.6984 | 11.0088 |
| (ii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ td | 8.1592 | 10.7104 |
| (iii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v27_t$ td | 8.5750 | 10.8076 |
| (iv) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ $v27_t$ | 7.9128 | 10.4808 |
| (v) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v9_t$ | 6.7852 | 9.7968 |
| (vi) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ $v27_t$ | 7.7230 | 10.1937 |
| (vii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ td | 7.8530 | 10.2812 |
| (viii) $x(i)_t$ $x(i)_{t-120}$ $x(i)_{t-240}$ | 5.5011 | 8.9126 |

[14]The results in table 2 (viii) are equivalent to the results in table 1 (i), the only difference being the choice of library size and close point group size. We can conclude that library sizes toward the lower end of their optimal range (2000 points) are better for the 120-minute predictions, but larger libraries (3000 points) are better for the 1-day predictions. However, we have decided in favour of a single script which forecasts 2-hrs and 1-day simultaneously, and therefore needs a single library size for both forecast horizons. This way, we can increase the close point group size to the upper end of its optimal range without overstepping the computational time constraint.

From table 2 we see that both 120-minute and 1-day forecasting equations were found to be more accurate without any time of day or volume variables. Neither of these variables are good indicators of the *sign* of the future return, although they may be better predictors of the magnitude, and we found that including these variables gave us some large predictions of the wrong sign which substantially increased the RMSE of our forecasts. It therefore turns out that an autoregressive model of returns is the most suitable predictor for both 120-minute and 1-day forecasts.

*4.4 Post-processing*

Even with simple autoregressive forecasting equations our forecasts of 120-minute and 1-day returns may need considerable smoothing in some places. This is because 'chaotic' periods were encountered where adjacent points fall into totally different close point groups and so forecasts were very different indeed for these adjacent points. Figure 3 illustrates 120-minute forecasts from part of the training data which are interesting because, after a short period (about an hour) of instability, they diverge significantly from the actual returns series[15]. On the other hand, we also found some very 'simple' patches in the data where successive points fall into very similar close point groups, and so the forecasts were very smooth.

Figures 4 and 5 show that these local stabilities and instabilities appear to be independent of the choices of embedding and predictor variables, and of the other parameters in the model. In Figure 4 we have predicted the 1-day returns using a long autoregression (lags of 270 and 540) in both the embedding and the prediction, a fixed library of 3000 points. In Figure 5 we have predicted the same set of returns, but added the smoothed volume data to the embedding and

[15]In figures 3 to 7 forecasts are plotted with the actual return which they forecast. That is, data point n on the horizontal axis records the actual return and the forecats (made either 120 minutes ago or 1 day ago) of that actual return.

used a rolling library of 2000 points. The close point group size was 50 in each. In both of these figures (and so independently of our parameter choices) an unstable regime at the end of the first day is followed by a period of very smooth forecasts which last the whole of the next day[16]. Sometimes there is a switch into a 'chaotic' period of forecasts over certain times of the day. Figure 6 illustrates 1-day forecasts during a period when only the mornings were particularly unstable.

It should be noted that many 'chaotic' periods have been found whilst examining our forecasts - and as illustrated above, these appear to be reasonably independent of our choices of embedding and predictor variables[17] - however no such features were apparent in the actual returns data. This is an interesting avenue for further research, but for the purposes of the competition we have simply smoothed our forecast returns by taking 9-point moving averages of all forecasts up to and including that point.

We have also 'clipped' forecasted returns so that they cannot lie more than $\alpha$ standard deviations away from the mean of the library of returns[18]. Clipping may not be necessary if the embedding and regression variables could be tailored more closely to the requirements of each individual point. But since the competition requires a single specification of parameters for the entire test data set, there are great differences in our results: in some regions of the training data forecasts are very accurate, in others the forecast errors are huge. But overall, using a data set of approximately half the size of the full training data, the RMSE ratio (to that of a random walk) was found to be 0.950536 for the 120-minute and 0.941270 for the 1-day forecasts.

---

[16]These data were taken from the latter part of the training data, where the time of close moved to 2:30pm and so 30 'flatpoints' occur at the end of each day. The flat periods make it easy to identify where one day closes and the next begins, and the actual coincide with our forecasts because the lags of 270 and 540 were used to generate these figures.

[17]Provided that, at a minimum, a returns autoregression is included in each.

[18]The choice of $\alpha$ was the subject of some 'trial-and-error' but we eventually settled on 1.645, for the sake of being normal!

There are bound to be points which are assigned into false regimes by an inappropriate embedding, and whose forecasts are therefore wildly off course. In figure 6 we see such a period in the training data, where the forecasts will be greatly improved after clipping. Having clipped and smoothed our returns forecasts they are now ready for integration back to levels using (4), and this concludes our description of the software submitted to the first INFFC.

## 5. Conclusions

The first INFFC has motivated us to adapt our programs, originally developed for the detection of 'chaos' in financial time series, for use in short-term forecasting. In the process we have developed a multivariate 'pattern recognition' or 'regime selection' methodology, which precedes and is distinct from the predictor model. We are very enthusiastic about the potential for this type of model in time-series forecasting.

One of the interesting by-products of our work on the INFFC data has been the discovery of short bursts of apparently 'chaotic' behaviour in the training data, where adjacent points fall into totally different areas of the embedding space and forecasts fluctuate wildly. The other side of the coin is the finding of longer periods of local stability where successive points fall into similar regions of the embedding space, and forecasts are very smooth indeed. However, most of the training data behaved as expected: generally our forecasts were far more variable than those one would expect from a neural network, because each point is forecast individually, and so part of the post-processing of our forecasts involves taking an historic moving average of previous forecasts.

Part of our problem is that the INFFC has been primarily aimed at neural networks, since the training and testing of the huge amount of data is so computationally intensive. In the longer-term we expect to develop more automated training facilities, but since we've had to do without

these for the competition many of the parameter choices have been made by 'trial-and-error' instead of comprehensive model diagnostics! Nevertheless we hope that our work provides a useful comparison of what can be achieved without the use of a neural network.

REFERENCES

Alexander, C.O. and A. Johnson "Are foreign exchange markets really efficient?" *Economics Letters*, 1992, **40** pp449-453

Alexander, C.O. "History Debunked" *RISK*, 1994, **7** No. 12 pp59-63.

Alexander, C.O. "Common volatility in the foreign exchange market" *Applied Financial Economics*, 1995, **5** No. 1 pp...

Alexander, C.O. and I. Giblin "Chaos in the system?" in *RISK***,** 1994, **7** no. 6 pp 71-76

Brock, W.A., J. Lakonishok and B. LeBaron "Simple technical trading rules and the stochastic properties of stock returns" *Jour. Finance*, 1992, **47** pp 1731-1764

Casdagli, M. 'Non-linear prediction of chaotic time series' *Physica D*, 1989, **35** pp335-356

Casdagli, M. 'Chaos and deterministic vs stochastic non-linear modelling' *JRSS series B* , 1992, **54** pp303-328.

Curcio, R. and C.A.E. Goodhart "When support/resistance levels are broken, can profits be made? Evidence from the foreign exchange market" *LSE financial markets report*, 1992.

Engle, R.F. and C.W.J. Granger "Co-integration and error correction: representation, estimation, and testing". *Econometrica*, 1987, **55**:2 pp 251-76.

Engle, R.F. and S. Kozicki "Testing for Common Features". *Journal of Business and Economic Statistics*, 1993, **11**, pp 369-95 (with discussions)

Hseih, D. "Chaos and nonlinear dynamics: applications to financial markets" *Jour. Finance* , 1991, **46** pp1839-1878

LeBaron, B "Chaos and Nonlinear Forecastability in Economics and Finance" in *Chaos and Forecasting* (ed H. Tong)1995 Vol 2 World Scientific

Murphy, J. *Technical Analysis of the Futures Markets* Prentice-Hall, 1986.

Nychka, D., S. Ellner, D. McCaffrey and A.R. Gallant "Finding chaos in noisy systems". *J.R.Statis.Soc.* **B** , 1992, **54** No. 2 pp 399-426.

Smith, R.L. 'Estimating dimension in noisy chaotic time series' *JRSS series B* , 1992, **54** pp329-352

Sugihara, G. and R.M. May 'Non-linear forecasting as a way of distinguishing chaos from measurement error in a data series' *Nature* 1990 **344** pp734-741.

Takens, F. 'Detecting strange attractors in fluid turbulence' *Dynamical systems and Turbulence* (eds. D. Rand and L-S Young) Spinger, 1981.

Tong, H. *Non-linear time series: a dynamical systems approach* OUP, 1990.

Yao, Q. and H. Tong "On prediction and chaos in stochastic systems" *presented to the Royal Society meeting on Chaos, London, 1994.*