Geum Yong Lee

This chapter explores several extensions to genetic programming for applications involving the forecasting of real world chaotic time series. We first used Genetic Symbolic Regression (GSR), which is the standard genetic programming technique applied to the forecasting problem in the same way that it is often applied to symbolic regression problems [Koza 1992, 1994]. We observed that the performance of GSR depends on the characteristics of the time series, and in particular that it worked better for deterministic time series than it did for stochastic or volatile time series. Taking a hint from this observation, an assumption was made in this study that the dynamics of a time series comprise a deterministic and a stochastic part. By subtracting the model built by GSR for the deterministic part from the original time series, the stochastic part would be obtained as a residual time series. This study noted the possibility that GSR could be used recursively to model the residual time series of rather stochastic dynamics, which may still comprise another deterministic and stochastic part. An algorithm called GRR (Genetic Recursive Regression) has been developed to apply GSR recursively to the sequence of residual time series of stochastic dynamics, giving birth to a sequence of sub-models for deterministic dynamics extractable at each recursive application. At each recursive application and after some termination conditions are met, the sub-models become the basis functions for a series-expansion type representation of a model. The numerical coefficients of the model are calculated by the least square method with respect to the predetermined region of the time series data set. When the region includes the latest data set, the model reflects the most recent changes in the dynamics of a time series, thus increasing the forecasting performance. This chapter shows how GRR has been successfully applied to many real world chaotic time series. The results are compared with those from other GSR-like methods and various soft-computing technologies such as neural networks. The results show that GRR saves much computational effort while achieving enhanced forecasting performance for several selected problems.

17.1 Problem Definition : Data Driven Model Building

The purpose of data driven model building in *n*-dimensional Euclidean space is to find the function $f : R^n \rightarrow R$ where the m data set $(R^n, R)_m$ is known. Rewriting the problem in terms of the time series analysis and forecasting literature [Casdagli 1993], we would like to find the function f in the following equation.

$$\begin{aligned} x_w^{(i)} &= f(\mathbf{x}_\mathbf{t}^{(\mathbf{i})}) \\ \mathbf{x}_\mathbf{t}^{(\mathbf{i})} &= \left(x_t, x_{t-\tau}, x_{t-2\tau}, \ldots, x_{t-(n-1)\tau}\right)^{(i)} \in R^n, and\ x_{t+T} \in R \end{aligned} \qquad (17.1)$$

where $x_{TIME}^{(i)}$ is the time series value of the *i*-th data set at time *TIME*, *t* is current time, T is future time (also called lead time or prediction horizon), τ is delay time (also called lag time or lag spacing), and $\mathbf{x}_\mathbf{t}^{(\mathbf{i})}$ is delay lag vector. $w = t + \mathrm{T}$ is forecast time.

Note that $\mathbf{x}_t^{(i)}$ is a point in the *n*-dimensional state-space R^n reconstructed from a scalar time series. Pseudo code for the general data driven model building process is

1. Obtain the scalar time series data : $x_0, \quad x_1, \quad x_2, \ \ldots$
2. Analyze the data to get information about reasonable values of T, τ , *n*
3. Prepare the *m* data set from the reconstructed *n*-dimensional state-space, R^n
4. Build the model from the *m* data set through a method, such as GSR.

It is not guaranteed to obtain reasonable values of T, τ , *n;* they depend on the characteristics of the data set. Algorithms or methods to determine T, τ , *n* may be another area of research. Also, it should be noted that R^n should be sufficiently dense (large *m*) to the extent that time series dynamics is clearly depicted in the space.

17.2 Genetic Symbolic Regression and Data Driven Model Building

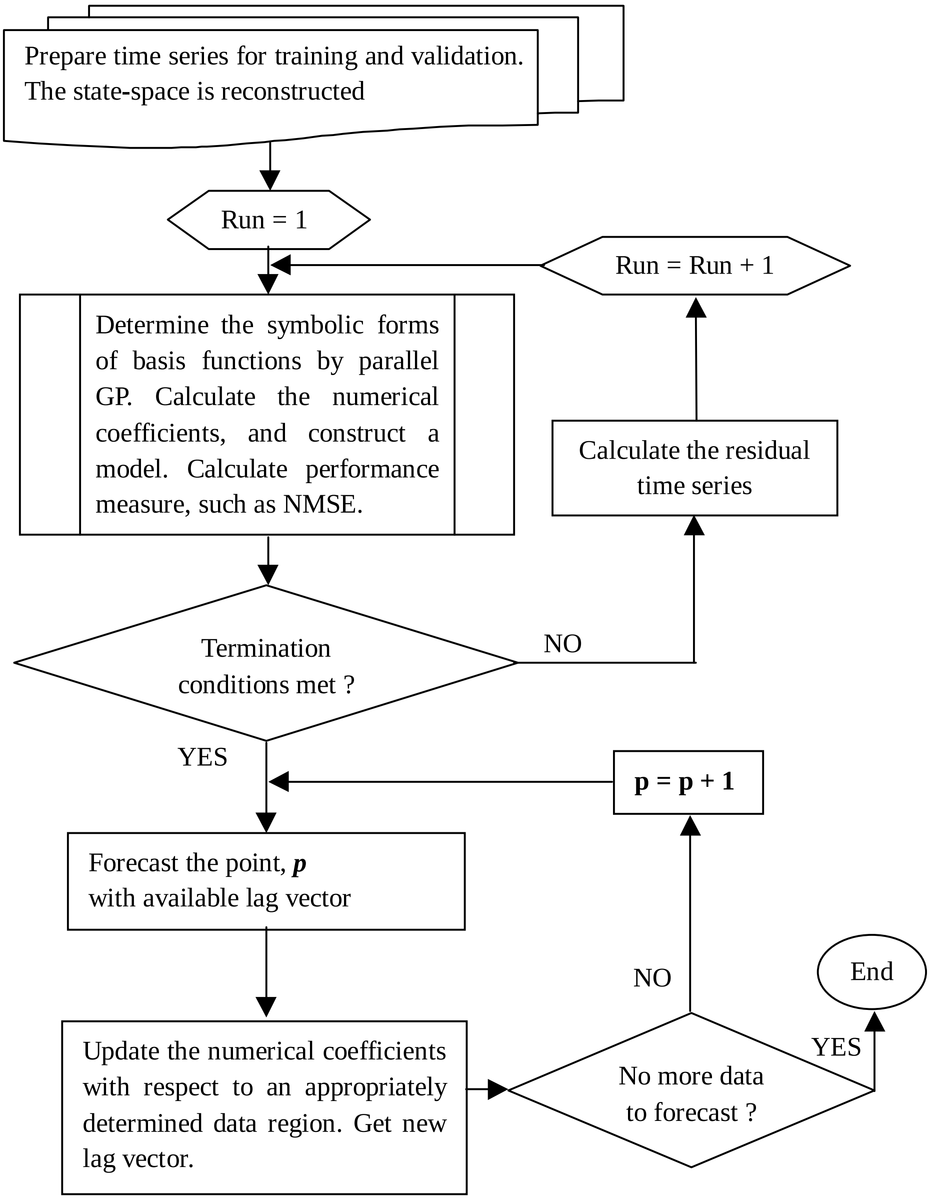
In this chapter, GSR is an ad hoc acronym for Genetic Symbolic Regression and refers to the standard genetic programming technique applied to a symbolic regression problem. Symbols may represent either complex concepts or simple values. GSR implements an elaborate set of symbolic operations designed to search possible combinations of symbolic elements, i. e. the ***symbol space***, based on the principle of natural selection.

Our problem of finding the functional relationship or model in Eq. (17.1) is none other than finding the appropriate symbolic form through which we can understand and forecast the dynamics carried by the series of data set. GSR helps us in solving the situation that we have no information about the shape and domain of the symbol space, and the meaningful symbolic forms should be found within the limited computational resources.

17.3 A New Algorithm; Genetic Recursive Regression (GRR)

The newly proposed GRR has five (5) major mechanisms that are different from the standard GSR. They are the recursive regression, the series-expansion type representation of a regression model, the multiple populations to efficiently get basis functions of the regression model, the real-time update of the numerical coefficients in the regression model and the extensive use of the derived terminal set.

Figure 17.1 shows the overall flow of the Genetic Recursive Regression. See section 17.3.1 for the concepts and formulation of the recursive regression. Section 17.3.2 addresses how to integrate a regression model based on several basis functions obtained by GSR. Section 17.3.3, 17.3.4 and 17.3.5 address the parallel computational architecture, the adaptation of a model to the latest data and the derived terminal set, respectively.



Prepare time series for training and validation. The state-space is reconstructed
Run = 1
Run = Run + 1
Determine the symbolic forms of basis functions by parallel GP. Calculate the numerical coefficients, and construct a model. Calculate performance measure, such as NMSE.
Calculate the residual time series
Termination conditions met ?
NO
YES
p = p + 1
Forecast the point, *p* with available lag vector
NO
End
YES
Update the numerical coefficients with respect to an appropriately determined data region. Get new lag vector.
No more data to forecast ?

Figure 17.1

The overall flow of the proposed method, GRR to construct a regression model and forecasting with it.

17.3.1 Recursive Regression

It may be practical to assume that real world time series is somewhat deterministic, and somewhat stochastic in its dynamics. If a data set is from a system of purely physical characteristics, e. g. the $NH_3 - FIR$ laser [Hübner 1993], it is usually more deterministic than stochastic. On the other hand, a data set from highly volatile economic system or physiological system is more stochastic than deterministic.

When we tried GSR for various data sets, it was relatively easy to model and forecast deterministic time series than stochastic ones. In this chapter, it was assumed that the governing system dynamics of a given time series is composed of deterministic and stochastic part. That is,

$$f = f_{deterministic} + f_{stochastic} \tag{17.2}$$

During study, it was confirmed that $f_{deterministic}$ is captured relatively easily at an early stage of GSR application. But, even with much increased computational efforts, it was very difficult to enhance the performance measure. If a data set is highly stochastic or volatile such as the foreign currency exchange rate, even the moderately performing $f_{deterministic}$ could not be obtained.

The solution was the recursive or the *zoom-in* regression. Let's see what this means in more detail. Recursion starts from zooming in on the difference or the residual time series $f - f_{deterministic} = f_{stochastic}$. The phrase 'zoom-in' comes from the fact that the modeling procedure is now applied to the residual time series of which order of magnitude is smaller than that of the original time series. The computational parameters remain unchanged from those used for obtaining $f_{deterministic}$. By the first recursive modeling procedure, we will have another pair of deterministic and stochastic part

$$f_{stochastic} = f^{(1)}_{deterministic} + f^{(1)}_{stochastic} \tag{17.3}$$

When the desirable level of performance is not reached with the still available computational resources, the modeling procedure is restarted with respect to the residual time series from Eq. (17.3). And the process goes on over and over again.

Now, let v be the number of the applied recursive modeling procedure. Then, the recursive model building procedure is given by

$$f^{(v)}_{stochastic} = f^{(v+1)}_{deterministic} + f^{(v+1)}_{stochastic}, \; v = 0,1,2,3,... \tag{17.4}$$

Note that Eq. (17.4) becomes Eq. (17.3) when v is 0. The left-hand-side of Eq. (17.3) is the first residual time series obtained by non-recursive modeling through GSR.

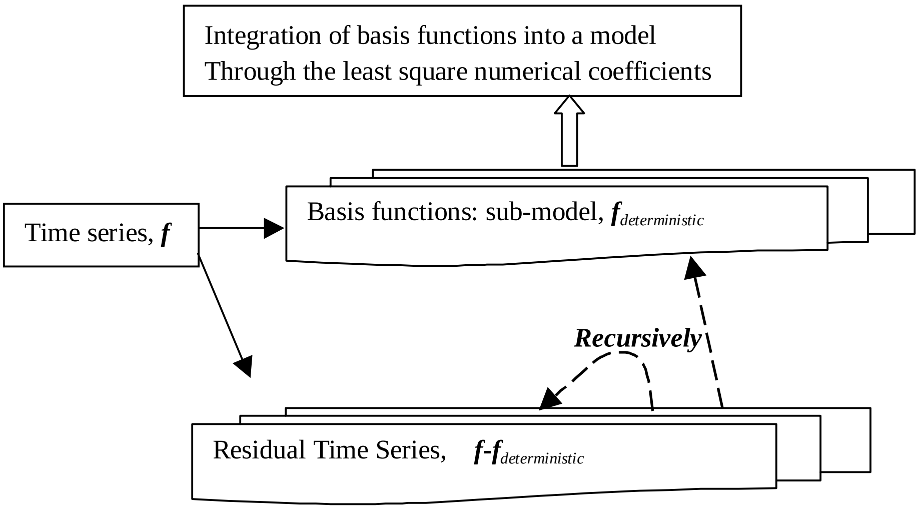

Integration of basis functions into a model
Through the least square numerical coefficients
Time series, *f*
Basis functions: sub-model, $f_{deterministic}$
Recursively
Residual Time Series, $f\text{-}f_{deterministic}$

Figure 17.2

The concept of the recursive regression.

Figure 17.2 shows the concept of the recursive regression. $f^{(0)}_{deterministic}$ and $f^{(v)}_{deterministic}$ become basis functions that are to be integrated into a final regression model. See section 17.3.2.

17.3.2 Representation of the Regression Model as a Series Expansion

Usually, the standard GSR produces one symbolic form as a regression model. But, GRR involves several symbolic forms to use in the series-expansion type representation of the regression model.

Now, let $f^{(0)}_{deterministic} = g_1(\mathbf{x})$ be the symbolic form from the first application of the modeling procedure. At this stage, our regression model is written as

$$\alpha_0 + \alpha_1 g_1(\mathbf{x}) \qquad (17.5)$$

where the numerical coefficients α_j are obtained by the least square method with respect to the training data set. If the second modeling procedure (= the first recursive regression) produces another symbolic form, say $f^{(1)}_{deterministic} = g_2(\mathbf{x})$, our regression model is now modified to

$$\alpha_0 + \alpha_1 g_1(\mathbf{x}) + \alpha_2 g_2(\mathbf{x}) \tag{17.6}$$

Since the numerical coefficients are re-calculated when there is any new symbolic form, the coefficient symbols in Eq. (17.5) and Eq. (17.6) do not necessarily have the same numerical values. The final representation of our model would be

$$f \cong \alpha_0 + \alpha_1 g_1(\mathbf{x}) + \alpha_2 g_2(\mathbf{x}) + ... = \sum_j \alpha_j g_j(\mathbf{x}) \tag{17.7}$$

The recursive modeling procedure can be considered as the gradual effort to find basis functions or sub-models $g_j(\mathbf{x})$ for the deterministic behavior remaining in the residual time series, $f_{stochastic}^{(v+1)} = f_{stochastic}^{(v)} - f_{deterministic}^{(v+1)}$.

17.3.3 Parallel Computational Architecture

Since each population can designate the best symbolic form as the model, a modeling procedure will produce only one basis function if it uses only a single population. However, GRR uses multiple populations to produce as many basis functions as there are populations undergoing evolution.

In fact, GRR is based on a parallel architecture to find multiple basis functions in one modeling procedure. Because of the limited computational resources to search the vast symbol space, we can not find a correct symbolic form for the dynamics of a given data set by applying only once the modeling procedure. There will be enormous variations in the attributes of the symbolic forms that are only partially or locally successful for identifying the time series dynamics.

Integration of the various locally successful symbolic forms is done as follows: let *P* the number of populations, then there will be *P* symbolic forms found for the first modeling procedure, that is, we have $g_{1,1}(\mathbf{x})$, $g_{1,2}(\mathbf{x})$, $g_{1,3}(\mathbf{x})$,…, $g_{1,P}(\mathbf{x})$. With these, the parallel version of Eq. (17.7) is

$$\alpha_0 + \sum_{p=1}^{P} \alpha_{1,p} g_{1,p}(\mathbf{x}) \tag{17.8}$$

And the final regression model can be rewritten as

$$f \cong \alpha_0 + \sum_{j=1}^{J} \sum_{p=1}^{P} \alpha_{j,p} g_{1,p}(\mathbf{x}) \tag{17.9}$$

In Eq. (17.8) and Eq. (17.9), the first subscripts stand for the modeling procedure, and the second subscripts stand for the number of the population.

There is a point to note for determining a basis function in Eq. (17.9). The best individual from a population replaces an existing individual of a special population called *super population* at each generation if and only if its fitness excels that of the individual being replaced. Otherwise, it is simply discarded. The number of the best individuals in the super population remains unchanged.

No individuals of the super population are engaged in the evolutionary processes of the *ordinary populations*. Only the individuals that survived the whole generations can become the basis functions in Eq. (17.9). In this context, the super population is different from something like the *multi-agent team* , e.g. [Luke 1996]; agents or team members are somehow engaged in the evolutionary processes. See section 17.4.2 for more detail.

17.3.4 Adaptive Update of the Numerical Coefficients.

Once the basis functions along with numerical coefficients are determined with respect to the training data set, the modeling procedure is over. An ideal model may be the one that forecasts data for any region of the given time series. However, usually our model is far from the ideal one for several reasons. For example, the limited number of data in the training region may not contain sufficient information to build an ideal model. Moreover, the dynamics of real world chaotic time series are generally largely time dependent.

Therefore, the forecasting performance of a model becomes poorer for a region that is distant from the training region. The simplest way to achieve forecasting performance in the remote region as good as those attainable for the training region may be to build a new model using the latest data region as a new training region. But the new model should be built in time to become a meaningful forecaster for the given time series.

This chapter explores more timesaving approach. The numerical coefficients are updated adaptively with respect to the newly available data set, so that as much latest information as possible is reflected in the model to enhance the forecasting performance. This may correspond to tuning up or adaptation of the existing model to the latest system dynamics.

17.3.5 Derived Terminal Set

An arbitrary function can be expressed as a series expansion with the basis orthogonal functions and the corresponding numerical coefficients. Taking a hint from this fact, we introduce a derived terminal set (DTS) and examine whether it can contribute to improving performance of a regression model when the model is expressed as a series expansion, like Eq. (17.9). DTS is obtained by applying an orthogonal function to a state-space coordinate variable x_j.

$$\boldsymbol{T} = \; [T_i(x_t), \;\; T_i(x_{t-\tau}), \;\; T_i(x_{t-2\tau}), \; \ldots, \; T_i(x_{t-(n-1)\tau})] \qquad (17.10)$$

where $T_i\,(x_j)$ is, in this chapter, $T_{i,j} = \cos[i \times \arccos(x_j)]$ or the Tschebyshev function of order *i* applied to the state-space coordinate variable x_j. We include DTS above in the terminal set along with the set of the normal state-space coordinate variables, $x_t, x_{t-\tau}, x_{t-2\tau}, \ldots, x_{t-(n-1)\tau}$ and a terminal which generates a random number between 0.0 and 1.0.

17.4 Implementation Issues

17.4.1 Fitness Assignment

Each symbolic form, i.e. individual in a population is assigned a value called *the fitness*, which measures how well the symbolic form fits the data set. In this chapter, the fitness value is the inverse of either NMSE (Normalized Mean Squared Error) or CV (Coefficient of Variation) calculated with respect to a predetermined subset, i.e. a region of the given time series. They are defined by

$$CV(\rho) = \frac{1}{\bar{x}}\left[(1/\rho)\sum_{i=1}^{\rho}\left(x^{(i)} - \tilde{x}^{(i)}\right)^2\right]^{\frac{1}{2}} \qquad (17.11)$$

$$NMSE(\rho) = \frac{\sum_i\left(x^{(i)} - \tilde{x}^{(i)}\right)^2}{\sum_i\left(x^{(i)} - \bar{x}\right)^2} \approx \frac{1}{\hat{\sigma}^2}\frac{1}{\rho}\sum_i\left(x^{(i)} - \tilde{x}^{(i)}\right)^2 = \frac{1}{\hat{\sigma}^2}MSE \qquad (17.12)$$

where $\tilde{x}^{(i)}$, $x^{(i)}$ are the predicted and the observed numerical values for the *i*-th datum. $\bar{x}$ and $\hat{\sigma}^2_\Omega$ denote the sample average and sample variance of the observed values in the predetermined subset of data [Weigend 1993]. ρ is the number of data points over which CV or NMSE is calculated. MSE stands for Mean Squared Error.

17.4.2 Super Population and Migration between Multiple Populations

By the evolutionary processes, we hope there is an overall increase in the performance or the fitness. But, also true is that good enough attributes (= structures and contents of symbolic forms) in one generation might be subject to destruction later through the blind application of the genetic symbolic operations. So, there is a need to keep the good enough attributes safely.

In addition to multiple populations undergoing evolution, GRR has a special-purpose population called the super population of which sole service is to store desirable attributes.

At the end of each generation, the best symbolic form is selected from each evolving populations. The size of the super population is equal to the number of such *ordinary* populations. Once the super population is full at a generation, any newly selected symbolic form after the generation is allowed to replace the existing one in the super population if and only if its fitness excels that of at least one individual in the super population.

No individuals at a generation in the super population are transferred to or *injected into* the ordinary populations; they only represent the best individuals that survived up to the generation. This means that they do not involve themselves in the evolutionary processes of the ordinary populations. Therefore, we should differentiate the super population from the concept of elitism [Goldberg 1989]. An elite individual or the multi-agent team member [Luke 1996] is somehow engaged in the evolutionary processes.

Obviously, we need to avoid any semantic replication in the super population. For example, $(+\, x_3\ (\sin\,(/\ x_1\ x_1)))$ is an example of semantic replications possible for $(+\, x_3\ (\sin 1.0)) = x_3$. Individuals of the super population after the last generation become the basis functions, Eq. (17.9). If there is any semantic replication among basis functions, it should be avoided before computing the numerical coefficients.

After operations to breed offspring are over, migrations occur between the ordinary popluations. In this chapter, the total number of migration operation is fixed to 1 % of the total number of individuals in all populations. Single migration is based on four random numbers. The first and second ones are to identify two populations between which a migration operation occurs. The third random number is compared with the migration probability, 0.02 here. The fourth random number is to identify an individual to replace. No migration to and from the super population is allowed.

17.4.3 Dealing with Absurd Attributes of a Symbolic Form

Symbolic forms sometimes contain mathematically or computationally absurd or nonsense attributes. Division-by-zero or negative values given to a function which requires only positive argument(s) results in the mathematical absurdity. Computational absurdity occurs when the attributes result in overflow or underflow for given computing systems.

For these cases, the numerical equivalent of the attributes is arbitrarily given, and the performance measures of the symbolic form are reduced by one hundred (100) times. This policy gives a definite penalty to the symbolic form for having the absurdity. By the reduced performance measures, their chances of being selected as parents to breed offspring become very small.

17.4.4 Division of the Data Set: Training, Validation and Prediction Regions

There are three regions in the time series data. That is, *T*, *V*, and *P* regions. The region *T* is called the “training region”, and comes first. Symbolic forms are constructed through the application of the evolutionary symbolic computation with respect to the region *T*.

Coming next is the *V* region or the “validation region”. When a regression model captures spurious information in the *T* region such as noise, the forecasting performance becomes low in the “prediction” or *P* region even if its performance in the *T* region was high. The problem is termed as *over-fitting*. See [Zhang 1993], [Smith 1993] for details. To prevent the over-fitting or over-training, the performance of any model constructed by the training data is evaluated with respect to the validation region.

When the validation performance deteriorates as compared to that in the former modeling procedure, the modeling procedure stops on the assumption that the model has started to capture spurious or excessive information such as noise or disturbance. This is the so-called *early stopping policy* [Geman 1992]. Also see [Weigend 1991].

The last region is called the “prediction” or *P* region. And, it is the region where the real effectiveness of the constructed model is manifested. Unless otherwise specified, the number of data points in *T*, *V*, and *P* region was 200, 100, 100 respectively for all time series that were applied in this chapter.

17.4.5 Termination Conditions

There may be two kinds of termination conditions. One is set by the limit on the computational resources; after the predetermined number of the generation is passed, the process of searching attributes of symbolic forms is terminated. The other one is natural; if the desired level of model performance, i.e. the performance criteria is achieved, there is no need to continue. For example, if NMSE or CV is 0.01 for a given data set, the model is sufficiently good and the modeling procedure stops. Deterioration in the validation performance also stops the procedure, see section 17.4.4.

17.4.6 Some Manipulations on the Raw Data

A DTS element $T_i\,(x_j)$ of Eq. (17.10) is defined on the interval $[-1,\ 1]$. Therefore, the raw interval or [*minimum*, *maximum*] of any state-space coordinate variable x_j should be modified to $[-1,\ 1]$. It is done by the following equation.

$$y_j = \ ax_j + b\,,\quad a = \frac{2}{\left(x_j^{n-\max} - x_j^{n-\min}\right)}\,,\quad b = -\left(1 + \frac{2x_j^{n-\min}}{\left(x_j^{n-\max} - x_j^{n-\min}\right)}\right) \qquad (17.13)$$

Then, $T_i\ (x_j)$ => $T_{i,j} = \cos[i \times \arccos(y_j)]$ = $\cos[i \times \arccos(ax_j + b)]$. The minimum and maximum values of the raw data, $x_j^{n-\min}$ and $x_j^{n-\max}$, should be determined for all regions of a data set. If they are determined only for the training region, values less than $x_j^{n-\min}$ or greater than $x_j^{n-\max}$ result in values less than –1 or values greater than 1, respectively, by Eq. (17.13). This situation causes mathematical absurdities. See section 17.4.3.

Let $x_j^{\min}$ and $x_j^{\max}$ be the minimum and the maximum of the variable x_j encountered in the training region. In this chapter, they are expanded by the following equations.

$$\begin{aligned} x_j^{n-\min} &= x_j^{\min} - \eta(x_j^{\max} - x_j^{\min}) = (1+\eta)x_j^{\min} - \eta x_j^{\max} \\ x_j^{n-\max} &= x_j^{\max} + \eta(x_j^{\max} - x_j^{\min}) = (1+\eta)x_j^{\max} - \eta x_j^{\min} \end{aligned} \qquad (17.14)$$

Therefore, the expanded effective interval [$x_j^{n-\min}$, $x_j^{n-\max}$] is 2η +1 times broader than the raw [$x_j^{\min}$, $x_j^{\max}$]. In this chapter, the expansion ratio is arbitrarily set to η =2.

17.5 Application to Real World Chaotic Time Series

17.5.1 Benchmarking

GRR was benchmarked with respect to three points of view. The effects of introducing DTS are not mentioned in this benchmarking, but discussed in the next section where GRR is applied to more stochastic time series. See section 17.5.2.2. We can see evident effects of DTS for stochastic time series. The three points of view are

1. How effective is the adaptive update of the numerical coefficients; for this, a new terminology “impact step” is introduced. The impact step γ is the number of data points between the data point to predict and the last data point of the region with respect to which the coefficients are updated. To predict x_t, the numerical coefficients in Eq. (17.7) or Eq. (17.9) are updated with data $x_{t-\gamma-200}$ ~ $x_{t-\gamma}$. The performance was observed with γ = 1, 2, 4, 6, 8, 12, 15, 20, 30, 40.
2. How effective is the parallel architecture; this was assessed by comparing the performances from single population with those from multiple populations.
3. How effective is the recursion; this was assessed by observing the performances with and without recursive regression.

17.5.1.1 Data and Computational Settings

For the benchmarking purpose, the time series data set generated by solving the Mackey-Glass equation was used, [Oakeley 1994]. The Mackey-Glass equation simulates the nonlinear dynamics of the human blood flow, and is written by

$$\frac{dx_t}{dt} = \frac{bx_{t-\Delta}}{1+\left(x_{t-\Delta}\right)^c} - ax_t \qquad \Rightarrow x_{t+1} = x_t + \frac{bx_{t-\Delta}}{1+\left(x_{t-\Delta}\right)^c} - ax_t \tag{17.15}$$

In Eq. (17.15), the constants are a = 0.1, b = 0.2, c = 10, and Δ = 30. To generate time series, the first 40 random seeds are prepared in the range [0.5, 1.5]. For the numerical values after the *40*-th datum, Eq. (17.15) was used.

GRR was benchmarked with three groups of computations. Each group has ten (10) computations, identified by ID's like ***c##*** where ## is the computation number. See Table 17.1 for details. The total number of symbolic forms allowed is 4500 in all computations, *c1 ~ c30,* for the sake of fair evaluation of the three points of view. Early termination could reduce the number. For the division of the data set, see section 17.4.4.

17.5.1.2 Benchmarking Results and Discussion

We applied GRR ten times for each computations *c1 ~ c30.* And the best model from a computation was taken for inter-comparisons between *c1 ~ c30.*

Effects of the Adaptive Update of the Numerical Coefficients

The allowed number of modeling procedures was 5; the first modeling procedure followed by four recursive modeling procedures. Note that all applications of GRR to the first group computations, *c1 ~ c10*, stopped only after the first recursion because the termination condition of NMSE (0.01 in the region *T*) had been met.

The performances of the best models in each computation are summarized in Table 17.2. Each numerical entry represents the NMSE value after the first recursion. First, observe the remarkable performance for every computation. The model was almost exact in capturing the dynamics carried by the time series data.

Table 17.1

Computational settings for benchmarking purpose, see section 17.5.1. There are three computation groups. For parameters that are not specified in this table, see Table 17.6.

	1st Group	2nd Group	3rd Group
Computation ID	*c1 ~ c10*	*c11 ~ c20*	*c21 ~ c30*
No. of Runs allowed	5	5	1
Population Size	15	30	30
No. of Populations	5	1	5
Generation Limit	12	30	30
Description	Recursion with multiple populations	Recursion with single population	No recursion with multiple populations

Table 17.2

NMSE values from the 1st group computations, see Table 17.1. γ is the impact step, section 17.5.1. In this table, ***NT*** = NMSE in Region *T*, ***NV*** = NMSE in Region *V*, ***NP*** = NMSE in Region *P*; Section 17.4.4

	c1	*c2*	*c3*	*c4*	*c5*	*c6*	*c7*	*c8*	*c9*	*c10*
γ	1	2	4	6	8	12	15	20	30	40
NT	0.00345	0.00767	0.00437	0.00345	0.00767	0.00437	0.01083	0.00586	0.00345	0.00767
NV	0.00526	0.01038	0.00401	0.00563	0.01100	0.00407	0.01378	0.01027	0.00493	0.01071
NP	0.00203	0.01060	0.00419	0.00228	0.01216	0.00450	0.00913	0.00544	0.00195	0.00693

Since the first group computations stopped only after the first recursive modeling procedure, the computational effort was only 1800 = the consumed modeling procedure 2 × the number of populations 5 × the population size 15 × the number of generation 12.

Although the performances vary with the computations: 0.00345 (*c1*, *c4*, *c9*), 0.00437 (*c3*, *c6*), 0.00586 (*c8*), 0.00767 (*c2*, *c5*, *c10*), and 0.01083 (*c7*), we can see that the better the performance in the training region, the better the performances in the validation and the prediction region.

Eq. (17.15) shows that the dictating variables are x_{t-1} and x_{t-30}. It is interesting to note that the impact step γ of 1 and 30 (*c1* and *c9*) resulted in the highest performances in all data regions. These observations with respect to the impact step γ were generally true for both the second group of computations (*c11* ~ *c20*) and the third group of computations (*c21* ~ *c30*).

Effects of the Parallel Architecture

The effectiveness of the parallel architecture or the multiple populations can be seen by comparing the performances between the first group computations (*c1* ~ *c10*) and the second group computations (*c11* ~ *c20*). Again, only the best models from each computations of the second group were taken for comparisons. Table 17.3 summarizes the comparison.

Except the row for γ, each entry represents a relative value between the two groups. Now, let NMSE (*c##*, *D*) represent the NMSE obtained in *c##* for the data region *D*. Then, the entry in the row *Ratio of* ***NT*** crossing the column *c11 vs. c1* is the NMSE (*c11*, *T*) divided by NMSE (*c1*, *T*).

Except the impact steps of 2, 8, 15, 20 and 40, the ratios of ***NT***, ***NV*** and ***NP*** are greater than 1.0. To take the computational efforts (CE) into account, the ratios of ***NT***, ***N*V** and ***NP*** should be multiplied by *Ratio of* **CE**. They become greater than 1.0 except the impact step of 15 and 40. This means that the NMSE values are larger in the second group computations than those in the first group computations.

Table 17.3

Performance comparisons between the 1st and 2nd group computations, see Table 17.1. Effects of the parallel computational architecture can be seen. CE (Computational Efforts) is the total number of symbolic forms evaluated to make a model.

2nd Group →	*c11*	*c12*	*c13*	*c14*	*c15*	*c16*	*c17*	*c18*	*c19*	*c20*
vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.
1st Group →	*c1*	*c2*	*c3*	*c4*	*c5*	*c6*	*c7*	*c8*	*c9*	*c10*
γ	1	2	4	6	8	12	15	20	30	40
Ratio of **CE**	2	2	2	2	2	2.5	1	2.5	2	1
Ratio of ***NT***	1.61	0.63	3.56	2.74	0.93	3.90	0.6	0.97	1.57	0.73
Ratio of ***NV***	1.80	0.80	5.13	2.36	0.85	4.60	0.88	0.43	2.00	0.83
Ratio of ***NP***	4.58	0.50	4.20	3.16	0.55	3.70	0.81	0.92	3.90	0.66

Effects of the Recursive Model Building

As discussed earlier, GRR allows multiple modeling procedures. The first modeling procedure is just a standard GSR. But, from the second modeling procedure, the modeling procedure can be considered as the recursive procedure. See Eq. (17.2) ~ Eq. (17.4), and Figure 17.2. To see how effective the recursive modeling procedure is, the model performances with and without recursion are compared in Table 17.4 between the first group and the third group computations.

The large values of the ratios of ***NT***, ***NV*** and ***NP*** in this table show that NMSE's are much smaller in the first group computations as compared to those in the third group computations. Since no recursive modeling procedure is allowed in the third group computations, these enhanced performances, i.e. smaller NMSE's must come from the recursive modeling procedures.

Table 17.4

Performance comparisons between the 1st and 3rd group computations, see Table 17.1. Effects of the recursive model building can be clearly seen.

3rd Group →	*c21*	*c22*	*c23*	*c24*	*c25*	*c26*	*c27*	*c28*	*c29*	*c30*
vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.	vs.
1st Group →	*c1*	*c2*	*c3*	*c4*	*c5*	*c6*	*c7*	*c8*	*c9*	*c10*
γ	1	2	4	6	8	12	15	20	30	40
Ratio of **CE**	2.5	2.5	2.5	2.5	2.5	2.5	2.5	2.5	2.5	2.5
Ratio of ***NT***	4.95	3.77	3.91	4.95	13.9	24.5	1.58	2.91	4.95	13.9
Ratio of ***NV***	3.28	2.98	4.49	3.24	11.3	29.7	1.26	1.59	3.34	9.90
Ratio of ***NP***	7.00	2.75	3.54	6.55	4.43	12.1	1.56	2.53	6.08	5.55

When we take the ratios of the computational efforts into account, the ratios of ***NT***, ***NV*** and ***NP*** become even greater; they should be multiplied by the entries in the corresponding columns crossing the row *Ratio of* **CE**. Also note that the recursive modeling procedure is much more effective than the parallel computational architecture; compare the foregoing two tables concerning them.

17.5.2 Real World Chaotic Time Series

17.5.2.1 Data and Computational Settings

The Complex Systems Summer School at the Santa Fe Institute planned a competition for time series analysis and prediction in the summer of 1990. Among the vast library of time series data, five representative data sets were selected and distributed through ftp for competition participants who used their own methods to predict designated hidden region. The data are available from `ftp.santafe.edu/pub/Time-Series.` [Weigend 1993] discusses the results and various methods used by successful participants of *Santa Fe Competition*.

ASHRAE (American Society for Heating, Refrigerating, and Air-conditioning Engineers) held a seminar on June 1993, Denver, Colorado to discuss and award the results of *ASHRAE Competition*.

For the competition, a vast range of time series data for weather and actual energy consumption such as electricity or hot water in a building is given to participants. See `ftp.cs.colorado.edu/pub/energy-shootout` for more details. Table 17.5 lists the data sources from Santa Fe and ASHRAE competitions.

In table 17.5, the data for intensity fluctuation of the far infra red NH_3 laser and for the surface brightness of a white dwarf star, PG1195 were generated from physics system, and they were analyzed to be rather deterministic or stationary. Other data are more stochastic or non-stationary. At the stochastic extreme is the currency exchange rate. [Weigend 1993] discusses the data characteristics in more detail.

For most time series data sets, the competitions required to use *the runaway extension* of the forecast data. The runaway extension means that the long-term forecasts should be based on the lag vectors available from the short-term forecasts by the same forecasters.

The performance measures from GRR are based on *the update extension* [Smith 1993] of the forecast data, in which the future forecast extension is free to use the true time series available up to the time that is T steps past from the data point to forecast.

Computational settings for this section are summarized in Table 17.6. Note that the crossover operation was allowed at any points in a tree. For mutation, a terminal symbol was simply replaced with another terminal symbol, and a function symbol was replaced with another function symbol that requires same number of arguments.

Table 17.5

This study uses time series data from Santa Fe [Weigend 1993] and ASHRAE competitions, section 17.5.2.1

Sources	Data	Description
ASHRAE Competition	Solar flare flux	Solar beam isolation flux data measured with respect to five different solar positions were given in the competition
	WBEC	Data for the whole energy-consumption data in a building (WBEC) and for the weather data outside the building were given
Santa Fe Competition	Laser intensity	Intensity fluctuation of the far infra red NH_3 laser. The data were analyzed to be deterministic since
	Heart rate of a human patient	Time series data for the heart rate, chest volume, blood oxygen concentration and EEG of a patient. Only the heart rate data were tackled in this study
	Currency exchange rate	Currency exchange rate between Swiss franc vs. US $. The data are highly non-stationary or stochastic.
	Particle position	Time-dependent position of a quantum particle in 4D potential well
	Star	Time series data observed for surface brightness of a white dwarf star, PG1195. The data are also deterministic

Table 17.6

Computational settings for genetic programming paradigm and time series modeling. See section 17.4 for other implementation issues of GRR. One run means either standard GSR or recursive GSR. If any termination condition, section 17.4.5, has been met, the actual number of run becomes smaller than that given here.

Settings for genetic programming paradigm	Number of populations	5
	Population size	30
	Generation limit	9
	Number of runs	5
	Function set	+,-,× ,/,*sin*,*cos*,*exp*,*log*,*expt*
	Terminal set	State-space coordinate variables, DTS, [illegible]
	Crossover fraction	0.8
	Depth of a tree	Initial 6, after-crossover 18
	Mutation fraction	0.1
	Reproduction fraction	0.1
	Migration probability	0.02
	Migrating individuals	1% of total individuals
Settings for time series modeling	Terminating NMSE	0.01
	Delay time	1
	Lag time	1
	Embedding dimension	1 or 4
	Number of data points	400
	Data region *T*	First 200 data
	Data region *V*	Next 100 data after region *T*
	Data region *P*	Last 100 data after region *V*

17.5.2.2 Results and Discussion

Depending on the characteristics of the time series data set, the computational efforts and the obtained performances were quite different. For example, the solar beam isolation flux and the fluctuations in the laser intensity were relatively easy to model and predict. That is, higher performance was possible with smaller computational efforts. The highly stochastic or volatile time series were very difficult.

Table 17.7 summarizes NMSE's obtained by GRR with the computational settings given in Table 17.6. The impact step was 1. For ASHRAE Competition, *Multivariate* means the usual time series modeling and forecasting problem, Eq. (17.1) for each time series, e.g. the solar beam isolation flux. This multivariate problem was not required in the ASHRAE Competition.

The competition required only the univariate analysis; see section 17.5.2.1 for details. *n* stands for the usual embedding dimension. The names of the time series data sets are reduced ad hoc for formatting purpose. See section 17.5.2.1 for details on them. We can see that the stochastic time series such as the human patient heart rate, the currency exchange rate, etc are much more difficult to tackle.

For stochastic data, the large embedding resulted in poorer performance while the performances generally improved for deterministic data. Simplistically assumed values of n, τ and T might have caused much more uncertainties for stochastic time series. We suspect that, with no correct information about n, τ and T available, a model based on smaller embedding dimension could be better for highly non-stationary dynamics.

Table 17.7

Summary of GRR application to real world chaotic time series data sets.

Time Series			***NT***	***NV***	***NP***
ASHRAE Competition (CV)	Solar	Multivariate	0.005	0.001	0.002
		Univariate	0.008	0.024	0.017
	WBEC	Multivariate	0.032	0.039	0.054
		Univariate	0.004	0.003	0.018
Santa Fe Competition (NMSE)	Laser	$n = 1$	0.007	0.018	0.015
		$n = 4$	0.0014	0.0027	0.0043
	Heart	$n = 1$	0.0654	0.1895	0.1651
		$n = 4$	0.1783	0.2587	0.3547
	Currency	$n = 1$	1.542	1.666	1.247
		$n = 4$	8.364	7.878	15.39
	Particle	$n = 1$	0.0233	0.0325	0.0756
		$n = 4$	0.6986	0.3544	0.1543
	Star	$n = 1$	0.0055	0.0075	0.0331
		$n = 4$	0.0016	0.0013	0.0017

Table 17.8

NMSE values with and without the derived terminal set, DTS. DTS has positive effects in particular on stochastic time series such as the heart rate and the currency exchange rate.

Time Series		NMSE (100) , Region *P*	
		Without DTS	With DTS
ASHRAE Competition	Solar	0.002	0.002
	WBEC	0.021	0.018
Santa Fe Competition	Laser (n = 4)	0.0042	0.0043
	Heart rate (n = 1)	8.021×10^{6}	0.1651
	Currency (n = 1)	35.887	1.247
	Particle (n = 1)	12.543	0.0756
	Star (n = 4)	0.0035	0.0017

Effects of the Derived Terminal Set (DTS)

To see the effects of DTS, GRR was applied to each of the time series used in section 17.5.1 and 17.5.2 with and without DTS. Table 17.8 shows selected results.

Observe that DTS has minimal effects for the deterministic time series. For the stochastic time series, DTS has positive effects, improving the performance measures. It may be temporarily safe to say that DTS does contribute to the model performances. The reasons why DTS has positive effects on the stochastic time series are not clear yet. Various kinds of DTS should be examined to have a conclusion.

17.5.3 Comparisons with Earlier Works

17.5.3.1 ASHRAE and Santa Fe Competitions

The model's forecasting performance recordings for the two time series data sets from each competition are compared with those of the best winners in the ASHRAE and Santa Fe Competitions, and summarized in the following table. The performance measures for comparisons were calculated for the same number of prediction data points.

Recall that GRR proposed here is based on the update extension while the very most participants used the runaway extension methods. Direct comparisons based on the runaway extension are topics for further study.

With the simplistic assumptions on the characteristics of time series and the small quantity (200) of data used to build a model, the runaway extension method is not applicable. As [Smith 1993] showed, the runaway extension is best successful when there is a sufficient data to build a dense state-space [Kailath 1980]. Coarsely reconstructed state-space, as was done in this study, results in the poor forecasts especially for stochastic time series. Table 17.9 is only for numerical comparisons between the best winners of the competitions and GRR.

Table 17.9

NMSE values for four time series data. Recall that the competitions required the runaway extension of the forecast data while GRR in this study was based on the update extension. See [Smith 1993].

	Time Series	Best Winner	This Study (GRR)
ASHRAE Competition	Solar Beam Isolation Flux	0.0240	0.002
	WBEC	0.14084	0.01804
Santa Fe Competition	Intensity of the laser light	0.023	0.00433
	Position of the quantum particle	0.086	0.0756

The performance measure for the first two time series was CV, and was NMSE for the last two times series. Numerically, it seems that GRR outperforms the best winners of the two competitions. However, it might be misleading at this stage if we say that GRR is superior to the methods used in the competitions, e. g. the highly sophisticated neural networks developed by Eric A. Wan [Wan 1993], the best winner of Santa Fe competition.

17.5.3.2 Mackey-Glass Equation

[Casdagli 1989], [Oakeley 1994] and [Iba 1994] also used the time series generated from Mackey-Glass equation, Eq. (17.15) to test their methods. The results by GRR were compared with those earlier works. See Table 17.10. Except the number of data points, computational settings were same as those used for section 17.5.2.

Since each of the earlier works used different performance measures, the performance measure NMSE of this study were converted to the corresponding performance measures. In Table 17.10, *GRR-c* is a variant of GRR. That is, the numerical coefficients obtained with respect to the region *T* are used to predict the time series. No adaptive update of the numerical coefficients is made.

Table 17.10

Comparisons of NMSE's between the works by Casdagli, Oakeley, and GRR. For the numerical values, See Table 17.4, p. 386 of [Oakeley 1994]. GRR-c stands for the constant-coefficient version of GRR. We can see that GRR-c and GRR worked better than earlier works. Comparing GRR-c and GRR reveals that the adaptive update of the numerical coefficients in Eq. (17.7) and (17.9) does improve the forecast performances.

Performance	Earlier works		This study	
Measures	[Casdagli 1989]	[Oakeley 1994]	GRR-c	GRR
NMSE(20)	0.0631	0.0311	0.0247	0.0187
NMSE(30)	0.1585		0.069	0.0085
NMSE(40)	0.316	0.158	0.181	0.0039
NMSE(50)	0.631	0.371	0.258	0.0025
NMSE(60)	0.990	0.6170	0.266	0.0046

See how much the performance was improved with GRR. Moreover, the computational efforts in the earlier works were much more than GRR. The computational efforts for GRR was only 5 (Number of populations) × 2 (Number of Modeling Procedure consumed; allowed was 5)× 30 (Population Size)× 9 (Generation Limit) = 2700, which is only fraction as compared with the earlier works. See Table 17.3, [Oakeley 1994].

[Iba 1994] used MSE (Mean Squared Error) as the performance measure. Δ is 17 for Iba. Table 17.11 compares MSE's with increasing computational efforts. Note that 6750 is the limit on the computational efforts set for GRR. Comparing MSE's for the testing data at 6750 (GRR-c) and 13890 [Iba 1994] reveals that GRR-c outperforms [Iba 1994] $0.01261/1.033\times10^{-4}=$ 122 times better. Moreover, if we consider the ratio of the computational efforts, 13890/6750 = 2.06, the figure goes up to 122× 2.06 = 250.

17.6 Conclusion and Issues Remaining

This chapter examined a new method to model chaotic time series through the application of evolutionary symbolic computation. And, if we use the update extension of the forecast data, the method GRR performed very well especially for deterministic dynamics or a time series that is deterministic.

The major originality of GRR lies in the recursive regression scheme through which multiple basis functions are derived. Based on the assumption that the dynamics of a time series comprise the stochastic part and the deterministic part, GRR has been quite successfully applied to many real world chaotic time series data sets of which modeling and forecasting were very difficult using the standard GSR.

It is interesting to note that a method known as *stochastic modeling* [Tong 1990] introduces a variety of noise terms for treating the stochastic dynamics of time series. Then, the modeling procedure is a systematic approach to minimize the noise terms or *errors*, they say, in a model. See [Tong 1990] for more details.

In the field of quantum mechanics, there is a theory or method called the *perturbation theory* to get solutions for very complex system equation that usually does not allow for an exact solution. The system for a time series might have such complex system equation.

Perturbation theory assumes that a solution to complex system equation is a sum of the unperturbed and the perturbed contribution. See [Rae 1992] and [Nayeh 1993] for more details. This is somewhat similar to our assumption here that a complex dynamics of real world chaotic time series comprise the deterministic part and the stochastic part, Eq. (17.2). It would be interesting if we try to interpret or improve GRR in the context of the various perturbation techniques to get the unperturbed *Hamiltonian* of a complex system.

Table 17.11

Comparison of the constant coefficient version of the proposed method, GRR-c with works by [Iba 1994]. For this table, the training region was the first 100 data of time series generated from Eq. (17.15) with $\Delta = 17$. The next 400 data were given for testing. Except the number of data, all other computational settings were the same as those given in Table 17.6.

Computational Efforts	Mean Squared Error			
	This Study (GRR-c)		[Iba 1994]	
	Training Data	Testing Data	Training Data	Testing Data
1350	2.632×10^{-3}	2.021×10^{-3}		
2700	2.935×10^{-4}	3.698×10^{-4}		
4050	2.931×10^{-4}	3.738×10^{-4}		
5400	9.932×10^{-5}	2.262×10^{-4}		
6750	4.219×10^{-5}	1.033×10^{-4}		
13980			0.01215	0.01261
104400			4.7×10^{-6}	5.06×10^{-6}

For tuning-up of a regression model, the numerical coefficients of the model were simply updated with respect to a predetermined number of data set of which last datum is γ, called the impact step, behind the datum to forecast. However, if there is plenty of data such that a sufficiently dense state-space is possible, the numerical coefficients should be updated with respect to *the nearest k neighbors* in the state-space [Casdagli 1993]. From the preliminary study, it was observed that the forecasting performance does depend on the number of the nearest neighbors. The dependency becomes more severe for stochastic time series than for deterministic time series.

Along with the numerical coefficients, the basis functions should also be updated. If sufficient computational resources are available, the basis functions could be updated with respect to newly available time series data. For an on-line or real-time forecasting of a time series, the update speed should be at least the one that can allow the new basis functions to be used in time. The update time of 6 hours is nonsense if the forecasting should be done in less than 1 hour or less.

Even if the basis functions are updated in a timely manner, it may still be a problem to update the regression model. We should have systematic selection algorithms for basis functions from the *old* and the newly updated basis functions.

GRR may be more powerful if it is combined with some kinds of time series characterization technologies. For example, detailed information about the lead time, the embedding dimension, and the delay time for a given time series is extremely important to have a successful model. Various chaos-qualifying technologies will also be very helpful. Fractal dimension is a good guide to reconstruct a state-space.

This study used the raw time series data. It is meaningful if we manipulate or represent the time series data differently. Normalization or appropriate data manipulation such as the first difference transformation [Chatfield 1989] might be helpful. The very most participants in the Santa Fe and ASHRAE competitions used the somehow manipulated data rather than the raw data. Also, we suspect that any normalization techniques to set limits on the minimum and maximum values of a model, e.g. between –1 and 1, should also be explored for the runaway extension.

For demonstrative comparisons between many time series of different characteristics, GRR was run in this study with only fixed computational parameters of genetic programming such as the crossover probability, the mutation probability, the migration probability between multiple populations, Table 17.6. The parents were selected only in proportion to the fitness values. Other selection policies were not examined. The initial and the after-crossover depth of trees were also fixed. Only the Tschebyshev function was used to make the derived terminal set. Detailed studies on effects of these parameters and policies are necessary. In addition, full-scale computational experiments are necessary to study the runaway extension of the forecast data with GRR.

Acknowledgments

I'd like to thank Professor John R. Koza of Stanford University for his encouragement. I would like to thank Professor Lee Spector of Hampshire College who fixed every spelling and grammatical bugs in my manuscript. Of course, any remaining errors are my responsibility.

Bibliography

Casdagli, M. C. (1989), “Nonlinear prediction of chaotic time series,” *Physics D*. 35: 335 – 356

Casdagli, M. C. (1993) and A. S. Weigend, “Exploring The Continuum Between Deterministic And Stochastic Modeling,” in *Time Series Prediction-Forecasting the Future and Understanding the Past*, A. S. Weigend, and N. A. Gershenfeld, Eds., SFI Studies in the Science of Complexity, Vol. XV, 347-366, Addison-Wesley Publishing Co.

Chatfield, C. (1989), *The Analysis of Time Series*, 4th ed. London : Chapman and Hall.

Geman, S. (1992) et al, “Neural Networks And The Bias / Variance Dilemma,” *Neural Computation*, **4**, 1-58.

Gershenfeld, N. A. (1993) and A. S. Weigend, “The Future Of Time Series : Learning and Understanding,” in *Time Series Prediction-Forecasting the Future and Understanding the Past*, A. S. Weigend, and N. A. Gershenfeld, Eds., SFI Studies in the Science of Complexity, Vol. XV, 1-70, Addison-Wesley Publishing Co.

Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Co.

Hübner, U (1993) et al, “Lorenz-Like Chaos In NH_3 Lasers (Data Set A),” in *Time Series Prediction-Forecasting the Future and Understanding the Past*, A. S. Weigend, and N. A. Gershenfeld, Eds., SFI Studies in the Science of Complexity, Vol. XV, 73-104, Addison-Wesley Publishing Co.

Iba, H. (1994) et al, “Genetic Programming Using A System Identification, ” *ETL-TR-94-11*, Electrotechnical Lab., Japan.

Kailath, T (1980), Linear Systems, Englewood Cliffs, NJ: Prentice Hall.

Koza, J. R. (1992), *Genetic Programming*, MIT Press.

Koza, J. R. (1994), *Genetic Programming* II, MIT Press.

Luke, S. (1996) and Spector L., “Evolving Teamwork and Coordination with Genetic Programming”, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 150-156, MIT Press, 28-31 July 1996.

Nayeh, A. H. (1993) and Nayfeh, A. H., *Introduction to Perturbation Techniques*, John Wiley & Sons.

Oakeley, H. (1994), “Two Scientific Application Of Genetic Programming: Stack Filters And Non-Linear Equation Fitting To Chaotic Data”, in *Advances in Genetic Programming*, Kenneth E. Kinnear Jr., Eds. MIT Press, 369-389.

Smith, L. A. (1993), “Does A Meeting In Santa Fe Imply Chaos ?,” in *Time Series Prediction-Forecasting the Future and Understanding the Past*, A. S. Weigend, and N. A. Gershenfeld, Eds., SFI Studies in the Science of Complexity, Vol. XV, 323-343, Addison-Wesley Publishing Co.

Rae, A. I. M. (1992), *Quantum Mechanics*, 3rd Edition, University of Birmingham, UK. IOP Publishing Ltd.

Tong, H. (1990), *Nonlinear Time Series Analysis: A Dynamical Systems Approach*. Oxford: Oxford University Press.

Wan, E. A. (1993), “Time Series Prediction by Using a Connectionist Network with Internal Delay Lines,” in *Time Series Prediction-Forecasting the Future and Understanding the Past*, A. S. Weigend, and N. A. Gershenfeld, Eds., SFI Studies in the Science of Complexity, Vol. XV, 195-217, Addison-Wesley Publishing Co.

Weigend, A. S. (1991) and D. E. Rumelhart, “The effective Dimension of the Space of Hidden Units.”, in *Proceedings of International Joint Conference on Neural Networks, Singapore*, 2069-2074. Piscataway, NY: IEEE Service Center.

Weigend, A. S. (1993) and N. A. Gershenfeld, Eds., Time *Series Prediction-Forecasting the Future and Understanding the Past*, SFI Studies in the Science of Complexity, Vol. XV, Addison Wesley Publishing Co

Zhang, X. (1993) and Jim Hutchinson, “Simple Architectures on Fast Machines : Practical Issues In Nonlinear Time Series Prediction,” in *Time Series Prediction-Forecasting the Future and Understanding the Past*, A. S. Weigend, and N. A. Gershenfeld, Eds., SFI Studies in the Science of Complexity, Vol. XV, 219-241, Addison-Wesley Publishing Co.