Gaussian Processes for Financial Time Series, a $C++$ Implementation

Sebastian Ferrando and Massimo Pascazi
Ryerson Polytechnic University
ferrando@acs.ryerson.ca

1. INTRODUCTION

This paper gives a brief overview of regression with Gaussian Processes (GP) and describes an implementation of GP models which can be of interest for empirical modelling of financial time series.

It is our intention to make available an implementation of GP based on a global optimization routine, specifically, Simulated Annealing (SA). The goal of finding a global maximum is a difficult one, nontheless getting traped in a local maximum of relatively poor quality is undesirable. Present solutions to this problem involve the use of priors and/or multiple runs of local optimization routines. The construction of the priors seems to be largely ad-hoc and not tailored to the specific problem. Moreover, it may be of interest to explore the likelihood itself. The implementation of SA is based on the one described in ([5]). For a description of Gaussian processes and their relation to neural networks see references ([6]), ([4]).

The program itself is a collection of classes that implements the various mathematical concepts that make up a Gaussian Process. There is a covariance matrix class, likelihood and prior classes, and classes for optimization and simulated annealing. The program, in short, optimizes the parameters that parameterize the covariance function through maximum likelihood estimation using simulated annealing and uses the optimized parameters to make predictions at test points.

2. MODELLING WITH GAUSSIAN PROCESSES, BASIC NOTIONS:

Elements of covariance functions will be denoted by $C_{p,r} = C(\vec{\mathbf{x}}^{(p)}, \vec{\mathbf{x}}^{(r)})$ where $\vec{\mathbf{x}}^{(p)}$ will stand for relevant variables used for modelling ("inputs") which are assumed to be known and not random. The stochastic process is denoted $\{T_{\vec{x}}\}$, i.e. a process indexed by vectors $\vec{x}$. Distributions are specified by conditioning on these variables. In general, they will be vector-like variables and the notation $\vec{\mathbf{x}}^{(p)} = \left(x_1^{(p)}, \ldots, x_m^{(p)}\right)$ will be used.

In our implementation of GP we have taken the mean of the GP to be zero, accordingly, our discussion will concentrate only on (non-sungular) covariance functions. Define the (auto) correlation function

$$\rho(\vec{\mathbf{x}}^{(p)}, \vec{\mathbf{x}}^{(r)}) = \frac{C_{p,r}}{\sqrt{C_{p,p}\, C_{r,r}}} \tag{1}$$

if the covariance function is stationary, i.e. $C_{p,r} = C(\vec{\mathbf{x}}^{(p)}, \vec{\mathbf{x}}^{(r)}) = C(\vec{\mathbf{x}}^{(p)} - \vec{\mathbf{x}}^{(r)})$, we have $\sigma^2 = C_{p,p} = C_{0,0}$. Given this invariance, the covariance can be treated as a function on the real numbers (or the given vector index set), this function is called the *autocovariance function* and denoted

$\gamma_{\vec{x}} = C(\vec{x})$. Similarly the *autocorrelation function* is defined as $\rho(\vec{x}) = \frac{\gamma_{\vec{x}}}{\gamma_{\vec{0}}}$. The value $\rho(\vec{x})$ may be interpreted as a measure of *linear correlation* between $T_{\vec{x}+\vec{x'}}$ and $T_{\vec{x'}}$.

A subclass of the stationary covariance functions is given by the situation when the autocorrelation function is isotropic, namely:

$$\rho(\vec{\mathbf{x}}^{(p)} - \vec{\mathbf{x}}^{(p)}) = \rho(||\vec{\mathbf{x}}^{(p)} - \vec{\mathbf{x}}^{(p)}||) = \tag{2}$$

where $||\ ||$ denotes the Euclidean norm in $\mathbf{R^m}$. More generally, the class of *anisotropic* correlation functions is given by

$$\rho(\vec{\mathbf{x}}^{(p)} - \vec{\mathbf{x}}^{(r)}) = \rho\left(\sqrt{\langle(\vec{\mathbf{x}}^{(p)} - \vec{\mathbf{x}}^{(r)}), K(\vec{\mathbf{x}}^{(p)} - \vec{\mathbf{x}}^{(r)})\rangle}\right) \tag{3}$$

where $\langle , \rangle$ is the Euclidean inner product and K is a positive definite matrix with respect to this inner product. Modelling with GP essentially entails to define consistent finite dimensional Gaussian distributions. As indicated in reference ([7]), a necessary and sufficient condition for this is the restriction that the covariance matrix is positive definite. The covariance matrices that we describe below all can be proven to imply well defined GP, some details for proving this can be found in ([7])p. 6.

3. LINEAR GAUSSIAN MODELS

The time series literature reveals that linear Gaussian Processes are the most common stochastic processes employed for prediction. The *autorregresive* (AR) *models* are of the form:

$$Zt = a_0 + \sum_{j=1}^{p} a_j Z_{t-j} + \epsilon_t$$

where the a_j are real constants ($a_p \neq 0$) and the ϵ_t are zero-mean uncorrelated random variables with a common variance. More general models are of the form:

$$Z_t = a_0 + \sum_{j=1}^{p} a_j Z_{t-j} + \sum_{j=0}^{q} b_j \epsilon_{t-j} \tag{4}$$

this is the so-called class of *autorregresive/moving averages* (ARMA). When $p = 0$ the model is called a *moving average* (MA). In practice, the noise ϵ_t is assumed to be i.i.d. and with distribution $N(0, \sigma^2)$. In this case $\{\epsilon_t\}$ is referred to as *Gaussian white noise*. Under appropriate conditions the process $\{Z_t\}$ is stationary, moreover if Z_0 has a $N(\mu_X, \sigma_Z^2)$ distribution, then the process $\{Z_t\}$ is a (linear) Gaussian Process. This class of models are called *linear Gaussian models*. It is clear that the present value of the time series is computed as a "linear filter" plus a noise term. It is not obvious, though, how to compute the parameters, what is the "best" one-step ahead prediction and what are the model implicit aasumptions about the correlation between the jumps $(Z_{n+1} - Z_n)$. This last point is essential for financial time series, given that in real data correlation among these forward jumps is expected.

The main advantage is that the theory is well understood and the computational time for the problem is well within the reach of modern computers. Also, over the years, a methodology

(known as Box and Jenkins), has been developed for forecasting time series. It may be thought as a first order approximation to the underlying process for the time series.

Suprisingly enough, explicit modelling with GP have been largely concentrated on linear GP,and have been, until recently, largely ignored by the time series community. The empirical work reported by Rasmussen ([8]) demostrates that Gaussian processes models have better predictive performance than several other nonparametric regression methods over a range of tasks with varying characteristics. Applications to the analysis and prediction of financial time series doesn't seem to be common among practitioners.

4. BASIC EQUATIONS

Assume we are given N data points $(\vec{\mathbf{x}}^{(1)}, t_{(1)}), \ldots, (\vec{\mathbf{x}}^{(N)}, t_{(N)})$ where $\vec{\mathbf{x}}^{(i)} = (x_1^{(i)}, \ldots, x_p^{(i)})$ is the $p - vector$ of inputs and $t_{(i)}$ is the associated target (which we assume to be a real number). The goal is to predict the target $t_{(N+1)}$ given $\vec{\mathbf{x}}^{(N+1)}$. This is done by computing the predictive distribution, which can be expressed only in terms of $x^{(N+1)}$ and the inputs and targets for the N given data points. We are interested in the joint distribution for the assumed Gaussian Processes (GP) underlying the data, i.e. we assume the existence of a GP $T_{\vec{\mathbf{x}}^{(i)}}$ and the $\{t_i\}$ are realizations of this GP. The joint distribution for the random variables $T_{\vec{\mathbf{x}}^{(i)}},\ i = 1, \ldots, N$, is a multivariate Gaussian characterized by its covariance matrix. For predictions we have the following formulas:

$$\mathbf{E}[T_{(N+1)}|t_1 \ldots, t_N] = \langle \vec{\mathbf{k}}, C^{-1}\vec{\mathbf{t}} \rangle \tag{5}$$

$$\mathbf{Var}[T_{N+1}|t_1, \ldots, t_N] = v - \langle \vec{\mathbf{k}}, C^{-1}\vec{\mathbf{k}} \rangle \tag{6}$$

where C is the N by N covariance matrix for the targets in the observed cases, $\vec{\mathbf{t}} = (t_1, \ldots, t_N)$ is the vector of known target values in these cases, $\vec{\mathbf{k}}$ is the vector of covariances between t_{N+1} and the N known targets, and v is the prior variance of t_{N+1}. In symbols

$$\begin{array}{rcl} k_p & = & C(\vec{\mathbf{x}}^{(N+1)}, \vec{\mathbf{x}}^{(p)}) \qquad , \quad p \ = \ 1, \ldots, n \\ v & = & C(\vec{\mathbf{x}}^{(N+1)}, \vec{\mathbf{x}}^{(N+1)}) \end{array}$$

In practice the covariance function is parametrized by a vector of parameters $\vec{\theta}$ which are estimated by maximizing the likelihood:

$$\ell\left(\vec{\theta}\right) = (2\pi)^{\frac{-N}{2}} (\det C(\vec{\theta}))^{\frac{-1}{2}} \exp\left(-1/2 \langle \vec{\mathbf{t}}, C^{-1}(\vec{\theta})\vec{\mathbf{t}} \rangle\right) \tag{7}$$

this quantity can be computed without explicit inversion of the matrix C. This is done by means of a recursive computation known as the *innovations algorithms* ([1]) which offers a faster algorithm than direct inversion of C, here we present a description of the algorithm (for proofs see ([1]). We need to introduce some notation, let $u_0 = C_{1,1}$ and define recursively for $1 \leq n \leq N-1$

$$\theta_{n,n-k} = u_k^{-1} \left(C_{n+1,k+1} - \sum_{j=0}^{k-1} \theta_{k,k-j}\theta_{n,n-j}u_j \right) \ k = 0, \ldots, n-1 \tag{8}$$

$$u_n = \left(C_{n+1,n+1} - \sum_{j=0}^{n-1} \theta_{n,n-j}^2 u_j \right) \tag{9}$$

these equations can be easily solved in the order u_0, $\theta_{1,1}$, $u_1, \theta_{2,2}\theta_{2,1}, u_2\theta_{3,3}\theta_{3,2}\theta_{3,1}, u_3, \ldots$. Next define the so-called one-step predictors by setting $\hat{t}_1 = 0$ and

$$\hat{t}_{n+1} = \sum_{j=1}^{n} \theta_{n,j} \left(t_{n+1-j} - \hat{t}_{n+1-j} \right). \tag{10}$$

Finally, the likelihood can be written as follows:

$$\ell\left(\vec{\theta}\right) = (2\pi)^{\frac{-N}{2}} (u_0 \ldots u_{N-1})^{\frac{-1}{2}} \exp\left(-1/2 \sum_{j=1}^{N} (t_j - \hat{t}_j)/u_{j-1} \right) \tag{11}$$

In practice it is convenient to minimize the minus log-Likelihood:

$$L = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log\det C - \frac{1}{2}\langle \vec{\mathbf{t}}, C(\vec{\theta})^{-1}\vec{\mathbf{t}} \rangle \tag{12}$$

Partial derivatives of minus log likelihood with respect parameters

$$\frac{\partial L}{\partial \theta_i} = \frac{1}{2} tr(\mathbf{C^{-1}} \frac{\partial \mathbf{C}}{\partial \theta_i}) + \frac{1}{2}\langle \vec{\mathbf{t}}, C^{-1} \frac{\partial \mathbf{C}}{\partial \theta_i} \mathbf{C^{-1}} \vec{\mathbf{t}} \rangle \tag{13}$$

5. COVARIANCES

In this section we indicate the covariance functions which have been implemented for modelling. We also indicate combinations of them, inputs used as well as constrains on the parameters.

AGARCH MODELS:

The formula for the AGARCH(p,q) model for the variance is

$$\sigma_n^2 = \omega + \sum_{i=1}^{p} \alpha_i (\phi(\vec{\mathbf{x}}^{(n-i)} - \xi) + \sum_{j=1}^{q} \beta_j \sigma_{n-j}^2, \tag{14}$$

where $\vec{\mathbf{x}}^{(n)}$ is a given time series of market prices, and ω, α_i, ξ and βj are the parameters of the model; they are determined using the procedure outlined below. The function ϕ will be specified later (see equation (21)).

The model for the standard deviation is the same, except the sigmas are not squared, i.e.,

$$\sigma_n = \omega + \sum_{i=1}^{p} \alpha_i \phi(\vec{\mathbf{x}}^{(n-i)} - \xi) + \sum_{j=1}^{q} \beta_j \sigma_{n-j} \tag{15}$$

The covariance is then:

$$C_{p,r} = \delta_{p,r}\sigma_p^2 \tag{16}$$

We refer to the models for the variance and standard deviation as "quadratic" and linear respectively. The basic constrains for the parameters in both models are:

$$\alpha_i > 0,\ \beta_j > 0,\ \omega > 0\ \ \xi > 0 \ \text{ and } \beta < 1 \tag{17}$$

Other constrains follow if we assume that $\mathbf{E}(x_i^{(n)}) = 0$ and $\mathbf{E}(x_i^{(n)}x_j^{(n)}) = \delta_{i,j}\sigma_n^2$. In our case, we are conditioning over the given inputs $\{x_i^{(n)}\}$ and no distributional assumptions are made. Actually, the usual assumption over the distribution of $\{x_i^{(n)}\}$ will take as away from the realm of GP.

In practice we will consider only the cases of $p = q = 1$ or $p = 0$ (this case is called the ARCH model) and q some small integer. Taking $\xi = 0$ in the above AGARCH model we obtain the GARCH model. A detailed reference for applications of ARCH-like models to financial data is ([2]).

The next covariance function was used by Rasmussen in his thesis ([8]),

$$C(\vec{\mathbf{x}}^{(p)}, \vec{\mathbf{x}}^{(q)}) = \theta_{m+1} \exp\left(-\frac{1}{2}\sum_{i=1}^{m}\theta_i(x_i^{(p)} - x_i^{(q)})^2\right) \tag{18}$$

this covariance can be easily seen to correspond to an anisotropic correlation function. Following Rasmussen we call this covariance the Automatic Relevance Determination (ARD) covariance. The ARD covariance can be combined, by addition, with the covariance of a linear process

$$a_0 + a_1\sum_{i=1}^{n} x_i^{(p)}x_i^{(q)} \tag{19}$$

this covariance is non-stationary. It can be seen that a well defined GP is defined thorough the following covariance function

$$C_1(\vec{\mathbf{x}}^{(p)}, \vec{\mathbf{x}}^{(q)}) = \sigma_p\sigma_q C(\vec{\mathbf{x}}^{(p)}, \vec{\mathbf{x}}^{(q)}) \tag{20}$$

where σ_n is given by equation (14) or (15) and C as in equation (18).

Selection of Inputs In general, the selection of inputs is problem dependent. In time series it is standard to use historical values as inputs i.e.:$\vec{\mathbf{x}}^{(p)} = \left(t_{(p-1)}, \ldots, t_{(p-m)}\right)$. This, of course, means that we are conditioning on the whole given path. For the function ϕ appearing in (14) we have taken:

$$\phi(\vec{\mathbf{x}}^{(n)} - \xi) = \left(x_1^{(n)} - \xi\right)^2 + \ldots + \left(x_q^{(n)} - \xi\right)^2 \tag{21}$$

where $q \leq m$ (usually $q = 1$).

Remarks:

1) The specification of the range for the parameters in the GARCH models above is too vague, further practical constrains have to be imposed to aid the optimization routine. Moreover, when adding covariances of different types, say $C^I(\vec{\theta})$ and $C^{II}(\vec{\theta})$, parameters $\vec{\theta}$ should be constrained in such a way that $C^I_{i,j}(\vec{\theta}) \in [a_{i,j}, b_{i,j}]$ and $C^{II}_{i,j}(\vec{\theta}) \in [a_{i,j}, b_{i,j}]$, for an apriori (problem dependent) interval $[a, b]$. Presently, our implementation handles constrains in an incomplete and unsatisfactory way.

2) We have not implemented the covariances of ARMA processes.

6. COMPUTATIONAL COST OF GAUSSIAN PROCESSES

The main computational burden when computing a Gaussian process is the calculation of the likelihood. Currently the program employs two methods to compute the likelihood, directly using matrix algebra, and indirectly using the innovations algorithm ([1]). Both the direct method and indirect method scale in the order of n^3 in computational time, where n is the number of inputs. This order of growth is just an estimate and the exact constant that scales this order is not known. It has been observed using the profiling program *gprof* that the direct method is on average 4 times slower than the indirect method. The order of growth for storage space when computing the likelihood is $n(n+1)/2$ since only one triangular covariance matrix is used in the matrix computations.

7. SIMULATED ANNEALING

The program for simulated annealing is based on Masters' ([5]) implementation, but we changed the way a new point is generated given a starting point. In Masters' code you supply a *temperature* range that is the standard deviation for the distribution that proposes the new points. In our implementation the temperature is a variable with an initial value of 1 and is reduced geometricaly to 0.01. We leave open the way a new point is generated, the user of annealing supplies a function that, given the current minimum and the new temperature value, returns a new point. In our implementation of this function we only return new points that are positive, and depending on what model we are training with, satisfy any other parameter constraints. Also, when the temperature is below 0.15, we use the parameter values as the standard deviation of the jumps. We have found this to be a useful way to adaptively set parameter-dependent standard deviations for the proposal distribution.

8. TESTS ON SIMULATED DATA AND EXAMPLE ON FINANCIAL DATA

Before proceeding to some illustrative examples, the different covariance models will be referred to as in Table 1. The first entry in Table 1, Model I, refers to the standard covariance function composed of an ARD term plus a linear term plus a term that estimates the variance of uncorrelated and identically distributed GP. The remaining models, Models II through IV, respectively add ARCH, GARCH, and AGARCH noise covariances to the ARD plus linear covariance. For practical reasons we have kept the number of training points small.

Our intention when presenting these numerical examples is to give an idea of the value of the relative error in the estimation of the likelihood given by our implementation. This has implications on the relevance of the use of a global optimization algorithm to estimate parameters. We do not report errors on test sets nor we perform predictions or tests on non-simulated data sets.

To illustrate interpolation with Gaussian processes the four models are used to generate a set of training data. The training data consists of 128 cases of inputs and targets. The training data is generated by arbitrarily fixing parameters for a covariance function, creating a dummy one dimensional input vector (the sequence 0 through 1 with increment $\frac{1}{128}$ is chosen), evaluating the covariance function with the fixed inputs at the fixed parameters, and sampling from this covariance by multiplying it's lower cholesky factor by a vector of normally distributed random numbers to obtain targets. Since we have chosen a covariance function, set parameters, and generated training data, we can reverse the process and recover an approximation to this particular covariance from the training data.

In Figure 1, Model I is used to generate the training data and interpolate. Table 2 shows the values where the parameters are fixed and the value of the likelihood at the fixed parameters. Model I and the generated training data are used to train, and the interpolating curve, seen in Figure 1, is obtained. In Table 2 the maximum likelihood parameters and the likelihood at these parameters (the minimum negative log parameters and likelihood are shown) are recorded with relative errors.

Figure 2 shows data generated using Model II. A heteroscedastic pattern of noisy targets can be seen near the left plot axis. This is caused by parameter β set at 0.95 which causes a slow decay of the noise variance initially set at 5.0 (see Table 3 for details and Figure 3 for a plot of the interpolating curve with error-bars). Tables 4 and 5 summarize results from generating data and training using Models III and IV, no figures are presented for these two simulations.

Next, we use the same four simulated data sets and train all the models (I, II, III, IV) on each of the sets. In this way we get an indication of the relevance of the estimates to distiguish among the models. Each subtable in Table 7 reports the achieved likelihoods for data generated from Models I through IV. The first subtable with heading Model I, for example, reports the achieved likelihoods for data generated as described in Table 2, but the training beeing done with each of the models.

The main purpose of the above examples was to test our implementation of simulated annealing. Also, To show the relevance of using simulated annealing we compared our implementation with that of Rasmussen. Figure 4 shows a smooth curve sampled from a Model I covariance. Table 6 reports the results when training with simulated annealing and conjugate gradient to interpolate this curve. Conjugate gradient stops short of the true likelihood and returns parameters that generate a straight line with very large error-bars.

Applications of the above models for financial data need to select the relevant inputs and its number (i.e. the dimensionality of the input vector). Moreover the magnitudes of the same entries for the different covariances making up the model should be comparable. Doing meaningful and informative test of Gaussian Processes for these data sets is out of the scope of this report. Nonetheless, our implementation of GP is readily applicable to time series data. Of course, the crucial point is that the input vector at time n should not depend on information available at later times. Figure 5 shows the raw data and interpolating curve for a foreign exchange (DEM/USD) for 500 data points. Figure 6 shows the error bars. We used Model IV with "dummy" (equally spaced) one dimensional input vectors.

Model I	ARD + linear + White Noise
Model II	ARD + linear + ARCH
Model III	ARD + linear + GARCH
Model IV	ARD + linear + AGARCH

Table 1: Model Types: each model number refers to a different covariance function.

Actual likelihood	4.703
Achieved likelihood	4.025
Relative error	0.1443

	w	v	a_0	a_1	σ
Actual parameters	100.0	10.0	20.0	20.0	0.2
Achieved parameters	106.423	6.987	23.386	53.154	0.238
Relative error	0.064	0.301	0.169	1.658	0.190

Table 2: Example Model I

Actual likelihood	3.143
Achieved likelihood	11.004
Relative error	2.502

	w	v	a_0	a_1	ω	β
Actual parameters	100.0	10.0	20.0	20.0	1e-4	0.95
Achieved parameters	106.062	7.754	102.147	49.293	0.002	0.950
Relative error	0.061	0.225	4.107	1.465	16.513	0.001

Table 3: Example Model II

Actual likelihood	36.780
Achieved likelihood	45.138
Relative error	0.227

	w	v	a_0	a_1	ω	α	β
Actual parameters	100.0	10.0	20.0	20.0	1e-4	0.05	0.92
Achieved parameters	60.344	28.445	36.771	72.133	0.042	0.013	0.887
Relative error	0.397	1.845	0.839	2.607	422.369	0.745	0.037

Table 4: Example Model III

Actual likelihood	-87.240
Achieved likelihood	-84.307
Relative error	0.034

	w	v	a_0	a_1	ω	α	ρ	β
Actual parameters	100.0	10.0	20.0	20.0	1e-4	0.5	0.5	0.5
Achieved parameters	83.977	42.021	79.625	177.175	0.001	0.858	0.508	0.141
Relative error	0.160	3.202	2.981	7.859	8.149	0.715	0.017	0.718

Table 5: Example Model IV

Actual likelihood	-507.388
Achieved likelihood by annealing	-507.366
Relative error	4.305e-05
Achieved likelihood by conjugate gradient	883.190
Relative error	2.74066

	w	v	a_0	a_1	σ
Actual parameters	500	500	1	1	0.001
Achieved parameters by annealing	509.559	477.206	28.408	143.682	0.001
Relative error	0.019	0.046	27.408	142.682	0.078
Achieved parameters by conjugate gradient	1.241	1.144	0.135	0.138	363.477
Relative error	0.998	0.998	0.865	0.862	363476

Table 6: Example comparison between the results obtained when training with simulated annealing and conjugate gradient.

Model I		
	Achieved likelihood	Relative error
Model I	4.02488	0.144259
Model II	4.80689	0.022005
Model III	110.858	22.56970
Model IV	6.28661	0.336614

Model II		
	Achieved likelihood	Relative error
Model I	84.0507	25.7447
Model II	11.0044	2.50156
Model III	95.2665	29.3135
Model IV	45.1952	13.3810

Model III		
	Achieved likelihood	Relative error
Model I	70.3538	0.912853
Model II	70.2092	0.908921
Model III	45.1379	0.227257
Model IV	45.7772	0.244637

Model IV		
	Achieved likelihood	Relative error
Model I	-43.4485	0.501963
Model II	-43.5729	0.500537
Model III	-35.4997	0.593078
Model IV	-84.3071	0.033613

Table 7:

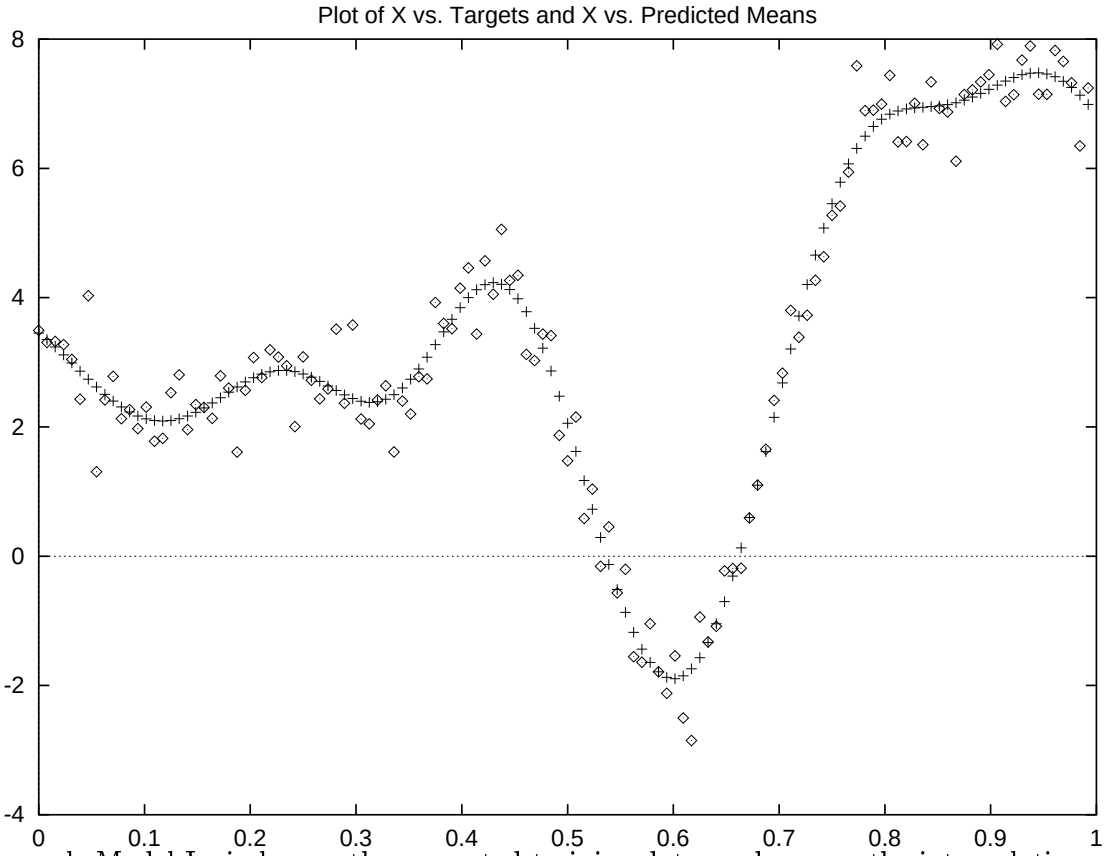

Fig. 1: Example Model I: circles are the generated training data, and crosses the interpolating curve.

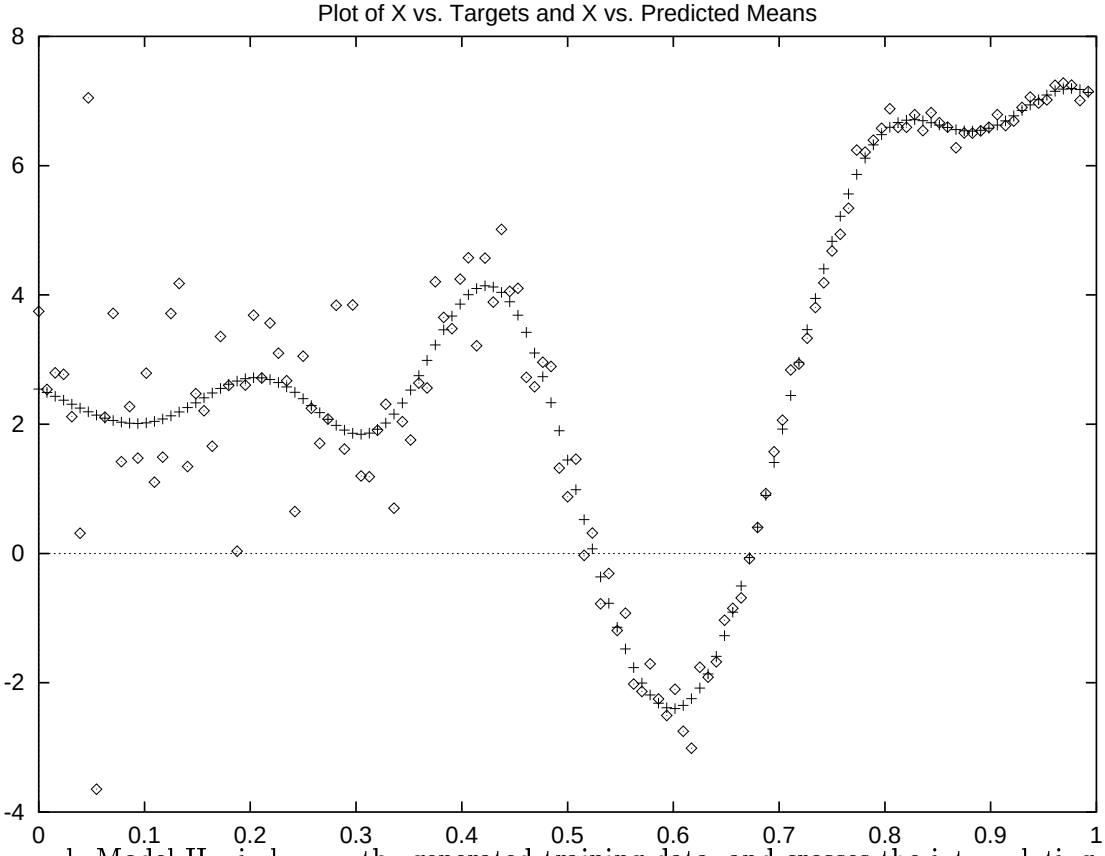

Fig. 2: Example Model II: circles are the generated training data, and crosses the interpolating curve.

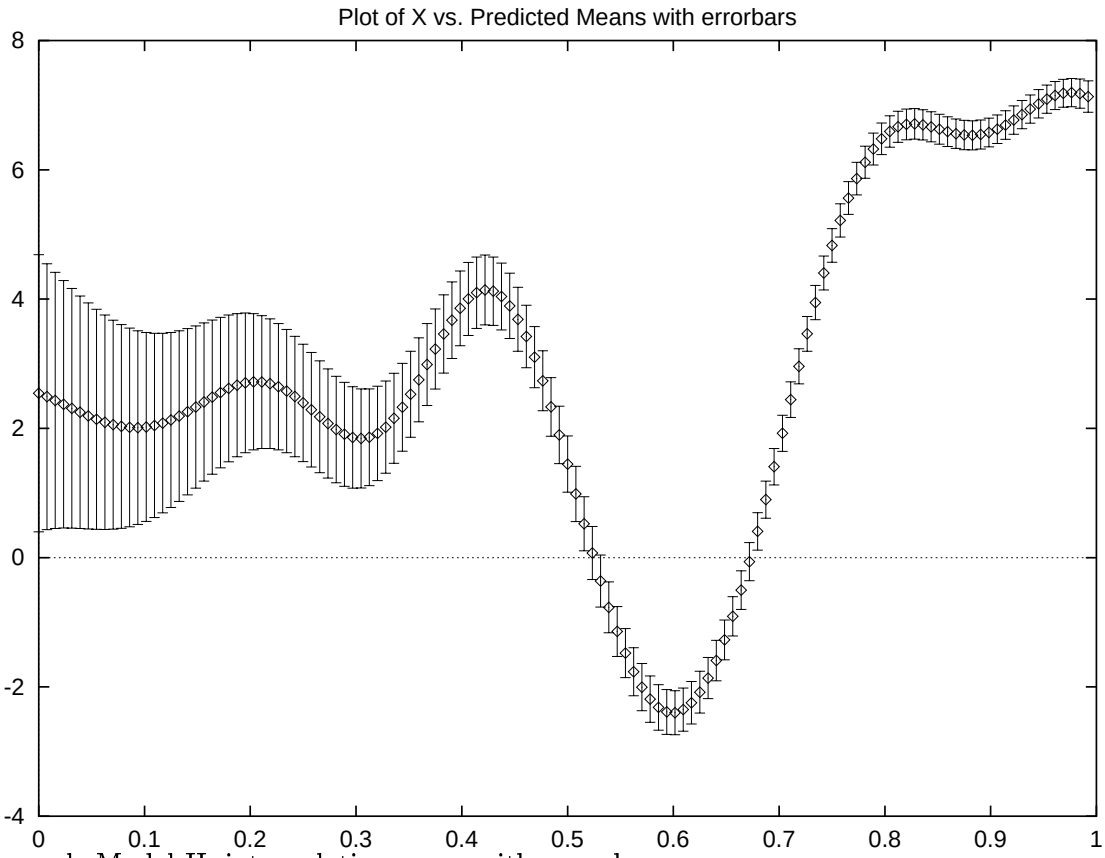

Fig. 3: Example Model II: interpolating curve with error-bars.

---

[1]P.J. Brockwell and R.A.Davis, *Time Series: Theory and Methods.* Springer-Verlag, New York, 1987.

[2]C. Gouriéoux, *ARCH Models and Financial Applications.* Springer Verlag 1997.

[3]M. Gibbs and D.J. MacKay, Efficient implementation of gaussian processes. Preprint, April 1997.

[4]D.J. MacKay, Gaussian processes, a replacement for supervised neural networks?. Preprint, April 1997.

[5]T. Masters, *Advanced Algorithms for Neural Networks.* John Wiley and Sons, New York.

[6]R.M. Neal, Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical report No. 9702, January 20, 1997.

[7]P. Abrahamsen, A review of Gaussian random fields and correlation functions, second edition. Report. *Norwegian Computing Center, Box 114 Blindern, N-0314 Oslo, Norway.* April 1997.

[8], C.E. Rasmussen, Evaluation of Gaussian processes and other methods for non-linear regression. PhD. Thesis, Univeristy of Toronto, 1996.

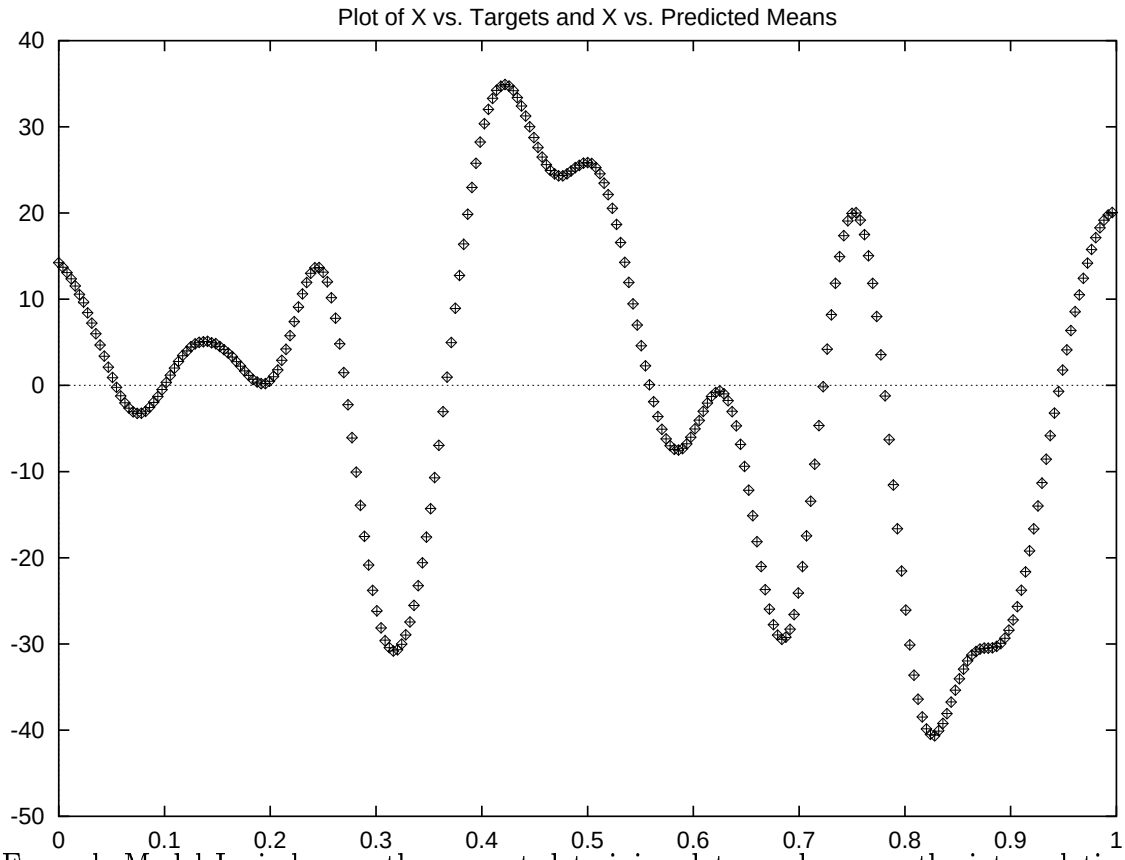

Fig. 4: Example Model I: circles are the generated training data, and crosses the interpolating curve obtained by training with simulated annealing.

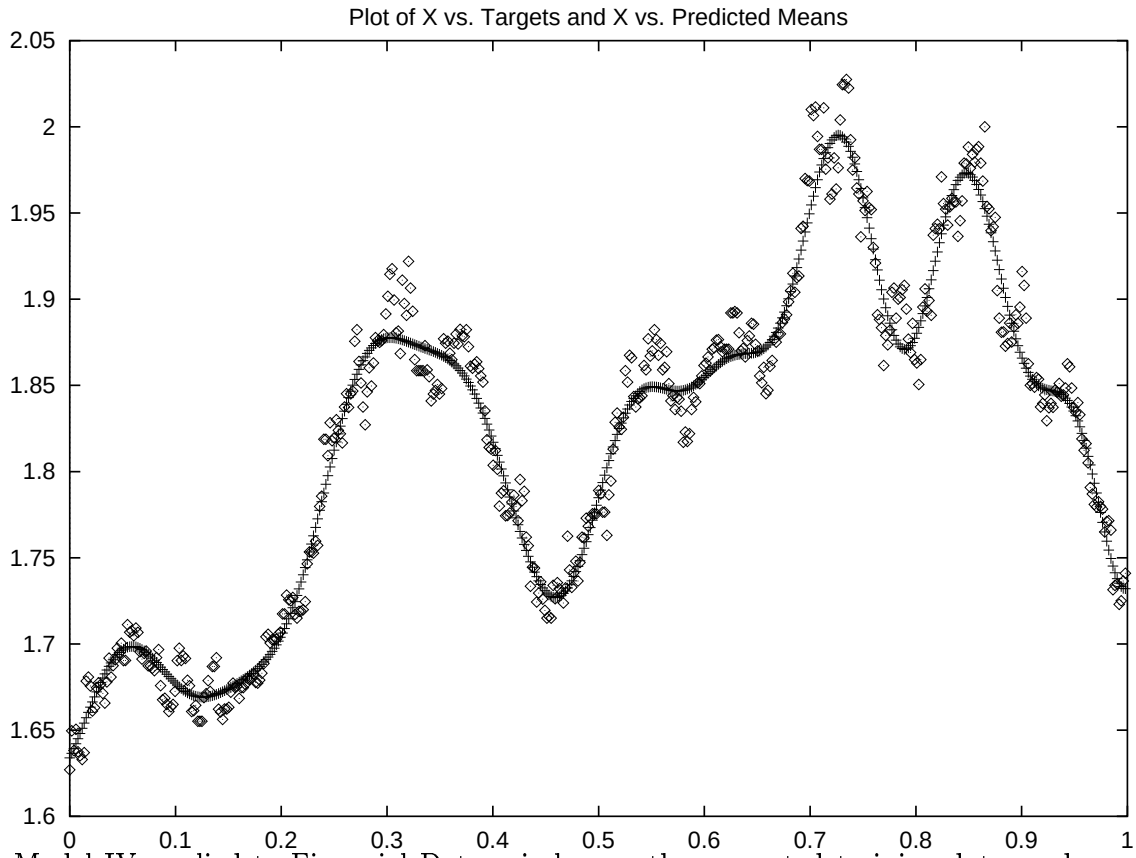

Fig. 5: Model IV applied to Financial Data: circles are the generated training data, and crosses the interpolating curve obtained by training with simulated annealing.

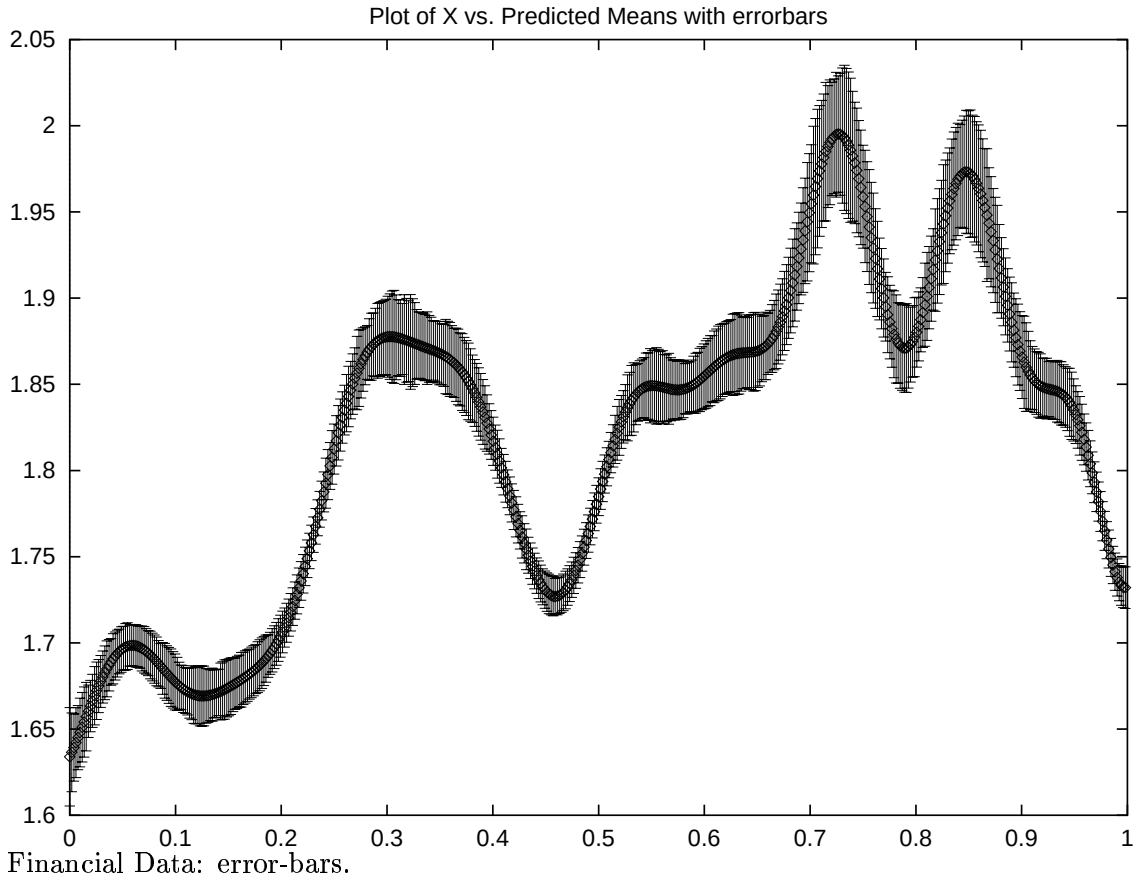

Fig. 6: Financial Data: error-bars.