First-order superdiagonal bilinear time series for tracking software reliability

Bouzaachane Khadija

Mohammadia Engineering School, Department of EGT, LERMA
Avenue Ibn Sina P.B. 765, Agdal Rabat, Morocco
Email: bouzaachane@netcourrier.com

Harti Mostafa

Math Info Department, Faculty of science Dhar Mahrez,
P.B. 1796, Atlas, Fez, Morocco
Email: mharti@fsdmfes.ac.ma

Benghabrit Youssef

ENSAM, Marjane II, Meknes Ismailia, P.B. 4024,
Ben M'Hamed, Meknes, Morocco
Email:you_benghabrit@yahoo.fr

Abstract

The research work of this paper is devoted to model and predict the inter-failure of software using one model of bilinear time series, this model can be related to existing software reliability growth model.

We propose the first-order bilinear time series model as a new model of software reliability. This model belongs to the models which are based on modeling inter-failure times of software. To show the strengths of our model, we compare its predictions with those of non-Gaussian Kalman filter model of Chen and Singpurwalla.

Keywords: Software Reliability, Inter-Failure Times, First-order Superdiagonal Bilinear Time Series Model, Non-Gaussian Kalman Filter Model.

1 Introduction

Software is an integral part of many critical and non-critical applications, such as, commercial avionics, banking and medical instrumentation. As, it permeates our modern society and continues to do so at an increasing pace in the future, the assurance of its reliability becomes an issue of critical concern.

Software reliability is defined as *the probability of failure-free operation for a specified period of time in a specified environment*(ANSI/IEEE (1991)). Software reliability is widely recognized as one of the most important aspects of software quality. In recent years an enormous multitude of software reliability models have been proposed for assessing the reliability of software, however, the assumptions made on many of them have not been theoretically or experimentally verified. Thus, models with few and simple assumptions are still needed.
This paper presents an important issue on software reliability: modeling and predicting inter-failure times of software by a particular case of bilinear time series models, which have not used for software reliability profit so far.

In reliability literature, we found several authors who were interested in time series models. Crow and Singpurwalla (1984) consider a Fourier series model for clustered failure data with cyclic behavior. Singpurwalla and Soyer(1992) propose an AR(1) model for log of the inter-failure times, they consider $T_i = \delta_i T_{i-1}^{\theta_i}$, for $i = 1, 2, 3, \ldots$, the coefficients θ_i are unknown and the error terms δ_i are added to account for the slight deviation from the relationship $T_i = T_{i-1}$, the coefficient θ_i represents the reliability growth decay for the i^{th} period. Chen and Singpurwalla(1994) study a non-Gaussian Kalman filter that is better suited for the skewed data often encountered in software reliability.

In this contribution, we propose the first-order superdiagonal bilinear model, and investigate its application to the inter-failure times data. To obtain a good predictions, we recall related work (Bouzaachane and al. (2006)), where we developed a new algorithm to estimate parameters of first-order superdiagonal bilinear model, by Maximum Likelihood method and using the widespread Kalman filter. The assumptions made by our approach were sample,moreover, the results obtained by numerical study are promising and prove the good performance of our model in contrast with on-Gaussian Kalman filter.

The remainder of this paper is organized as follows. In section 2 we present our bilinear model and we give a general overview of our estimating algorithm. In section 3 we shall describe briefly a non-Gaussian Kalman filter model. Section 4 is concerned with application of our model to real inter-failure data, and in order to highlight its advantage, we compare it with a non-Gaussian Kalman filter model. Finally, section 5 concludes.

2 General overview

The aims of this section are: (a) presenting the definition and some main properties of first-order superdiagonal bilinear model, (b) briefly describing the algorithm for estimating the parameters of first-order superdiagonal bilinear model.

2.1 First-order superdiagonal bilinear model

The first-order univariate superdiagonal bilinear model denoted BL(0,0,2,1) and defined by

$$X_t = e_t + b_{21} X_{t-2} e_{t-1}, \tag{1}$$

with b_{21} is constant coefficient of the model and e_t are independent and identically distributed (i.i.d)$N(0, \sigma^2)$ (i.e e_t Gaussian white noise).
Guegan (1981) established the necessary and sufficient condition for the existence of stationary process $\{X_t\}_{(t \in \mathbb{Z})}$, and stated the condition under what model (1) is invertible. The summary of these results follows.

Theorem 1 *(Guegan, 1981)*

If $b_{21}^2 \sigma^2 < 1$ *then there exists a unique strictly stationary process* X_t*, satisfying* (1)*, given by*

$$X_t = e_t + \sum_{j \geq 1} [b_{21}^j e_{t-2j}] \prod_{k=1}^{j} \{e_{t-2k+1}\}, \tag{2}$$

which converges on quadratic average.

Theorem 2 *(Guegan, 1981)*

If $b_{21}^2 \, \sigma^2 < \frac{1}{2}$*, then the model* (1) *is invertible on* b_{21}*.*

In this case

$$\begin{aligned} e_t^{b_{21}} = X_t - & \sum_{j=1}^{t-2} (-b_{21})^j b_{21} X_{t-j-2} X_{t-j-1} \prod_{l=1}^{j} X_{t-l-1} \\ & - b_{21} X_{t-2} X_{t-1}. \end{aligned} \tag{3}$$

In the next section, we will illustrate the important role of the expression (3) in our estimating algorithm.

2.2 Estimating Algorithm of parameters of the first-order superdiagonal bilinear model

Let $\theta = (\theta_1, \theta_2)$, where $\theta_1 = b_{21}, \theta_2 = \sigma^2$, denote the vector of unknown parameters, and $(x_1, \ldots, x_n)$ the observed data. In this study, we propose estimating θ by using Maximum Likelihood, and we assume that the process $\{X_t\}_{t=1}^n$ is Gaussian. So, the log-likelihood function of the process $\{X_t\}_{t=1}^n$ can be written as

$$L(x, \theta) = \ln \prod_{t=1}^{n} f(x_t | \mathcal{X}_{t-1}) = \sum_{t=1}^{n} \ln f(x_t | \mathcal{X}_{t-1}) \tag{4}$$

where $\mathcal{X}_t = (x_1, \ldots, x_t)$ is the set of observations available at time $t = 1, \ldots, n$, and $f(x_t | \mathcal{X}_{t-1})$ expressed normal density function of x_t given $\mathcal{X}_t$, with mean $\hat{x}_{t|t-1} = E[X_t | \mathcal{X}_{t-1}]$ and variance

$\hat{M}_{t|t-1} = E[(x_t - \hat{x}_{t|t-1})^2]$ (see Hamilton, 1994). The Maximum Likelihood estimates of θ, are thus given by maximizing

$$L(X;\theta) = -\frac{n}{2}\ln(2\pi) - \sum_{t=1}^{n}\ln(\hat{M}_{t|t-1}) - \sum_{t=1}^{n}\frac{(x_t - \hat{x}_{t|t-1})^2}{\hat{M}_{t|t-1}}. \quad (5)$$

Notice that $\hat{x}_{t|t-1}$ and $\hat{M}_{t|t-1}$ are calculated by the Kalman filter algorithm given in (Hamilton, 1994).
To apply the Kalman filter algorithm, we have to construct a convenient space-state representation of our BL(0,0,2,1) model (1). Our construction leads to the state form (6), with time-varying parameters, given by

$$\begin{cases} \xi_{t+1} = F_t\xi_t + v_{t+1} & \text{state equation} \\ X_t = H\xi_t & \text{observation equation} \end{cases} \quad (6)$$

where $t = 1, \ldots, n$, $\xi_t = [X_t, X_{t-1}, X_{t-2}]'$, $H = [1, 0, 0]$, $v_t = [e_t, 0, 0]'$ and $F_t = \begin{pmatrix} 0 & b_{21}e_t & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ is time-varying parameter (Hamilton, 1994).

It's worthwhile to mention that the conditions of stationarity and invertibility have an important role in our study. If the conditions are fulfilled we can express e_t by X_t (equation (3)), and so we are able to calculate F_t.

The Kalman filter recursively generates an optimal forecast $\hat{\xi}_{t+1|t} = E[\xi_{t+1}/X_1, ..., X_t]$ of the state vector ξ_{t+1}, with associated mean square error $P_{t+1|t} = E\{[\xi_{t+1} - E[\xi_{t+1}][\xi_{t+1} - E[\xi'_{t+1}\}$, $t = 1, \ldots, n$.
Given starting values $\hat{\xi}_{1|0}$ and $P_{1|0}$, the next step in Kalman filter algorithm is to calculate $\hat{\xi}_{2|1}$ and $P_{2|1}$. The calculations for $t = 1, \ldots, n$ have the same basic form, so we will describe them in general terms for t

Step 1: Calculate the forecasting $\hat{X}_{t|t-1}$ of the observation X_t, and the error $\hat{M}_{t|t-1}$ of this forecast.

Step 2: Updating the state vector $\hat{\xi}_{t|t}$. Compute $P_{t|t}$ the MSE of this updated projection.

Step 3: Calculate the forecasting $\hat{\xi}_{t+1|t}$, and the MSE $P_{t+1|t}$ of this forecast.

Now, our aim is to estimate b_{21} and σ^2 using the maximum likelihood method. We have also constructed the log-likelihood function using Kalman filter. It's straightforward to show that it is difficult to find the partial derivatives of $L(X;\theta)$, so, it is more natural to use a method which does not need derivatives. Therefore, we used the method of **Powell** (Himmelblau (1972)). Powell's method is a modification of a quadratically converge method proposed by Smith(Smith, 1962). It ensures convergence in a finite number of steps for positive quadratic function, by using a convenient conjugate directions. For non-quadratic function the method still valid.

Before describing the algorithm, it is worthwhile to provide a sub algorithm which tests if parameters fulfill the conditions of stationarity and invertibility, we will denote it by Test.

The second sub algorithm, which we must provide, concerns the compute of $L(X;\theta)$ by Kalman filter, we will denote it by KF. These two sub algorithm will be implemented in our global estimating algorithm.
Herein, we are interested in minimizing $l(X,\theta) = -L(X,\theta)$, which is clearly equivalent to maximizing $L(X,\theta)$.

Sub algorithm Test(θ)

If $(\theta_1^2 \times \theta_2 < \frac{1}{2})$
Then go to next.
Else return to the previous step and take the previous point as starting point. "Here we force our algorithm to chose another value of vector parameters".
End if

End Sub

θ_1 and θ_2 are the values of parameters, b_{21} and σ^2 respectively. In this test, we just considered the invertibility conditions, because $\theta_1^2 \times \theta_2 < \frac{1}{2}$ implies $\theta_1^2 \times \theta_2 < 1$, then the stationarity condition is verified.

Sub algorithm KF(θ)

Step 1 Given the starting condition $\hat{\xi}_{1|0}$ and $P_{1|0}$.

compute $\hat{X}_{1|0} = H\hat{\xi}_{1|0}$, $\hat{M}_{1|0} = HP_{1|0}H'$.

Step 2 For t=1 to n Do

Compute K_t, $\hat{\xi}_{t|t}$, $P_{t|t}$, F_t, $\hat{\xi}_{t+1|t}$, $P_{t+1|t}$.

Compute $\hat{X}_{t+1|t}$, $\hat{M}_{t+1|t}$.

End For

Step 3 som=$\frac{n}{2}\ln(2\pi)$

For t=1 to n Do

$som = som + \ln(\hat{M}_{t|t-1}) + \frac{(x_t - \hat{x}_{t|t-1})^2}{\hat{M}_{t|t-1}}$

End For

$l(X;\theta) = som.$

end Sub.

Now, we propose the global algorithm for parameters estimation, where we integrate all the sub algorithms describe above. We use the values of estimators obtained by moments method for BL(0,0,2,1) (Guegan, 1984) as initial values of the parameters b_{21} and σ^2 for our algorithm.

PMLKF algorithm

Step 1 : Let $\theta^{(0)}$ be an initial point(verify the test of invertibility) and let $d_1, \ldots, d_m$ the basis vectors (initial set of directions).

Step 2 : For k= 1 to m do

(m is a number of parameters)

Call sub algorithm KF($\theta^{(k)}$);

Step 3: Solve $\min_{\lambda} l(X; \theta^{(k-1)} + \lambda * d_k) = l(X; \theta^{(k-1)} + \lambda_k * d_k)$;

Step 4: call Test($\theta^{(k-1)} + \lambda_k * d_k$);

Step 5: Set $\theta^{(k)} \longleftarrow \theta^{(k-1)} + \lambda_k * d_k$;

End For

Step 6 : Compute :

- $l_0 = l(X; \theta^{(0)})$;
- $l_m = l(X; \theta^{(m)})$;
- $l_E = l(X; 2 * \theta^{(m)} - \theta^{(0)})$;
- $\Delta l = \max_{i=1,\ldots,m} \{l(X; \theta^{(i-1)}) - l(X; \theta^{(i)})\}$; let s be the index for which the maximum is attained (the high diminution of l obtained at step 3 in the direction d_s). Then we have two case:

Step 7 : If $((l_E \geq l_0)$ and/or $(l_0 - 2 * l_m + l_E)(l_0 - l_m - \Delta L)^2 \geq \frac{1}{2}\Delta l(l_0 - l_E)^2)$

Then keep the old set of directions for the next iteration and keep $\theta^{(m)}$ as the starting point.

Else

If $((l_E < l_0)$ and/or $(l_0 - 2 * l_m + l_E)(l_0 - l_m - \Delta l)^2 < \frac{1}{2}\Delta l(l_0 - l_E)^2)$

Then go to step 1 and take $\theta^{(m)} - \theta^{(0)}$ as new direction and the obtained point as a new starting point in the next iteration. In the other hand, the direction d_s is substituted by $\theta^{(m)} - \theta^{(0)} = d$. The new set of directions which we shall use through the next iterations was $(d_1, d_2, \ldots, d_{s-1}, d_{s+1}, \ldots, d_m, d)$.

Step 8 : the steps 1 to 7 are repeated until a termination criterion is satisfied.

The step 3 of our algorithm consists on minimizing a function $\varphi(\lambda) = l(X; \theta + \lambda * d)$ with one variable. To overcome this difficulty we chose the Brent's method which works faster and guarantees convergence in a reasonable number of steps (see Brent 1973).
To assess the performance of our estimate algorithm, and to verify that our method improves the estimations obtained by the moment method, we have conducted series of Monte Carlo simulation experiments from different models. The results of this study were detailed in (Bouzaachane and al. (2006)).

3 Non-Gaussian model

In this section, our goal, is to give a generalisation of the non-Gaussian model, the interested reader can find a detailed development of this model in Chen and Singpurwalla (1994).

Let T_n represent the observation at time n, θ_n the state of nature at n, and $D_n = \{T_1, \ldots, T_n\}$ the collection of observation until n. The notation "$T \sim \text{Gamma}(v, \theta)$" denotes the fact that T has a gamma distribution with shape parameter v and scale parameter θ. Similarly, "$T \sim \text{Beta}(v, \theta)$" denotes the fact that T has a beta distribution with parameters v and θ. A non-Gaussian model is defined as follows

the observation equation : $(T_n|\theta_n) \sim \text{Gamma}(w_n, \theta_n)$.

the system equation : $(C_n\theta_n/\theta_{n-1}|\theta_{n-1}) \sim \text{Beta}(\sigma_{n-1}, v_{n-1})$.

the initial information : $(\theta_0|D_0) \sim \text{Gamma}(\sigma_0 + v_0, u_0)$.

where C_n, w_n, v_n and σ_n are assumed to be known and non-negative; furthermore, they satisfy the condition

$$\sigma_{n-1} + w_n = \sigma_n + v_n, \qquad \text{for} \quad n = 2, 3, \ldots.$$

The results(Updating equations)

The posterior of θ_n : $(\theta_{n-1}|D_{n-1}) \sim \text{Gamma}(\sigma_{n-1} + v_{n-1}, u_{n-1})$.

The prior of θ_n : $(\theta_n|D_{n-1}) \sim \text{Gamma}(\sigma_{n-1}, C_n u_{n-1})$.

The 1-step forcast : $(T_n/C_n u_{n-1}|D_{n-1}) \sim \text{Pearson Type VI}(p = w_n, q = \sigma_{n-1})$.

The posterior of θ_n : $(\theta_n|D_n) \sim \text{Gamma}(\sigma_{n-1} + w_n, u_n)$, where $u_n = C_n u_{n-1} + T_n$.

In the next section, we apply the first-order superdiagonal bilinear model to real software data, and we compare it with non-Gaussian Kalman filter model.

4 Application to software reliability

In the recent years, many models have been developed to model and predict software reliability, and all the practitioners agree on the fact that no particular reliability growth model is superior in assessing software systems in any circumstances.

In this paper, our goal is to highlight the advantage of our bilinear model over the non-Gaussian Kalman filter model denoted ICD, by using simple criterion.
We are interested in modeling the process $\{T_i\}_{i=1}^{i=n}$, where T_n denotes the time between $(n-1)^{\text{th}}$ and the n^{th} failure, for $n \geq 1$. The data genereted from this process, have been detected from "system 40", and have been recorded in report of Musa (1979). They are presented in Figure 1.

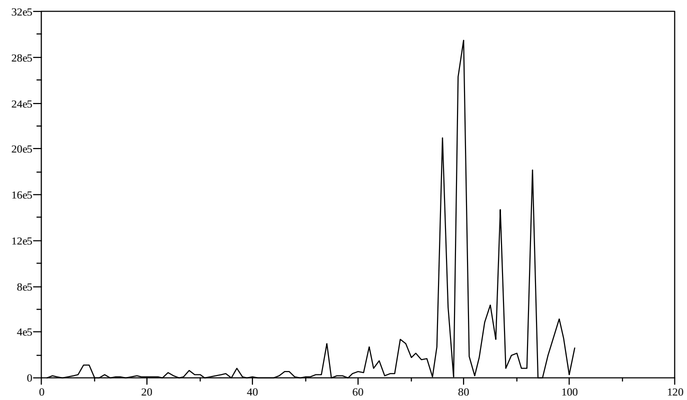

Figure 1: Graphical representation of "system 40" data

The application of our BL(0,0,2,1) model to data represented in Figure 1, leads to estimate the initial values of our estimating algorithm, and then, the parameters of this model. The estimation of the initial values obtained, didn't fulfill the conditions of stationarity and invertibility, which are necessary for our estimating algorithm. However, to overcome this problem, we apply the logarithm to inter-failure times T_n, that yields $X_n = log(T_n)$, the logarithms of the inter-failure times represented in Figure 2.

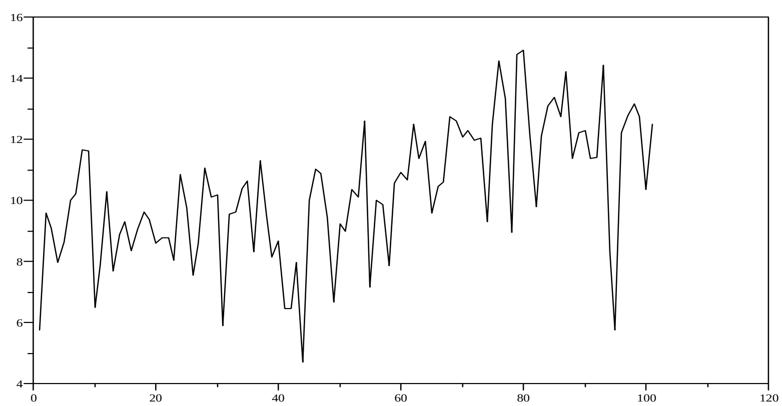

Figure 2: Graphical representation of logarithm of inter-failure times data

Fitting a first-order superdiagona bilinear model BL(0,0,2,1) to the data presented in Figure 2, gives the following estimates

$$b_{21} = 0.054177, \sigma^2 = 54.325463$$

Thus, to be able to compare our model with the ICD model, we apply the logarithm to $\hat{t}_n$, the predictions given by the ICD model. These predictions are given in Chen and Singpurwalla (1992).

The logarithms of inter-failure times data and the predictions obtained by both models are plotted in Figure 3. It shows that our model outperforms the ICD model. Moreover, this result is confirmed by the statistics criterion issued from the model application and given in table 1.

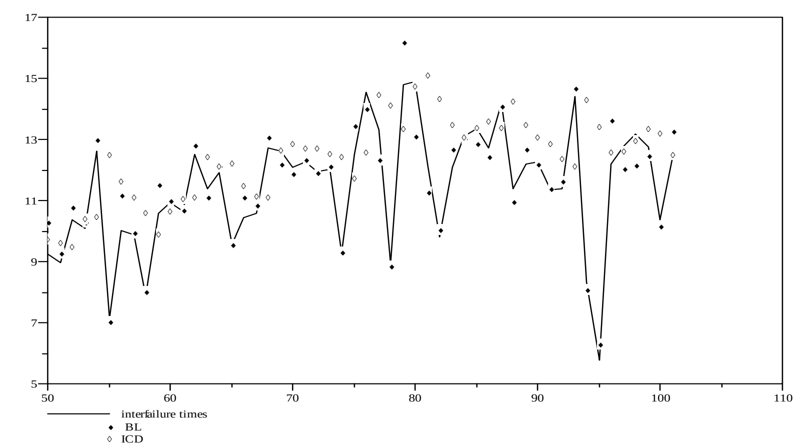

Figure 3: Graphical representation of logarithm of inter-failure times and predictions issued from BL(0,0,2,1) model and ICD model

Table 1: Comparison of values of criterion

Criterion	BL(0,0,2,1)	ICD
Mean error	-0.0861337	-0.9692787
Variance of errors(VarE)	0.3679005	4.1934257
MAPE	0.0390292%	0.1232053%
100×(VarE/Variance of data)	1.052579%	11.997569%

5 Conclusion

We have presented a new model enabling to predict software reliability. The obtained results support the superiority of the first-order superdiagonal bilinear model over a non-Gaussian Kalman filter model.
These results are promising and suggest a new research area to be integrated in software reliability.

References

Brent, R. B. (1973), Algorithms for Minimization without derivatives. *Prentice-Hall, Series in automatic computation.*

Bouzaachane, K., Harti, M. and Benghabrit, Y. (2006), Maximum likelihood estimation Algorithm of first-order superdiagonal bilinear time series *submitted to Interstat*

Chen, Y. and Singpurwalla, N. D. (1994), A non-Gaussian Kalman filter model for tracking software reliability. *Statistica Sinica 4, 535-548.*

Crow, L. H and Singpurwalla, N. D. (1984), An emprically developed Fourier series model for describing software failures. *IEEE Trans, R-33, 176-183.*

Granger, G. W. J. and Anderson, A. P. (1978), An introduction to bilinear time series analysis. *Vandehoeck and Ruprecht, Götting.*

Guegan, D. (1981), Etude d'un modèle non linéaire, le modèle superdiagonal d'ordre un. *C. R. Acad. Sci. Paris ser. 1 293. 95-98.*

Guegan, D. (1984), Tests de modèles non linéaires. *Proceeding of the third franco belgium meeting of statisticians, J.-P. Florens et al. Eds. FUSL, Brussels. 45-66.*

Guegan, D. (1986), Représentation l-markovienne et existence d'une représentation affine en l'état des modèles bilinéaires. *C.R.A.S. t. Serie I. N. 47. 289-292.*

Guegan, D. (1987), Different representation for bilinear Models. *C.N.R.S.Orsay, France. 389-408.*

Guegan, D. and Pham, D. T. (1989), A note on the estimation of the parameters of the diagonal bilinear model by the method of the least squares. *Scand. J. Statist. 16. 159-136.*

Hamilton, J. D. (1994), Time series analysis. *Princeton: Princeton University Press.*

Himmelblau, D. M. (1972), Applied nonlinear programming. *New York: McGraw-Hill Book Company.*

Hristova, D. (2005), Maximum likelihood estimation of a unit root bilinear model with an application to prices. *Studies in nonlinear dynamics and econometrics. volume 9, Issue 1.*

Minoux, M. (1983), Programmation mathématique, théorie et algorithmes:Tome1. *C.N.E.T et E.N.S.T.*

Musa, J. D. (1979), Software reliability data. *IEEE Comput. Soc. Repository.*

Musa, J. D. 1997 Introduction to software reliability engineering and testing, *IEEE .3-12.*

Pham, D. T. (1985), Bilinear Markovian representation and bilinear models. *Soch. Process and their appl. 20, 295-306.*

Powell, M. J. D. (1964), An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Comp.J.7, 155-162.*

Singpurwalla, N. D. and Soyer, R. (1992), Non-homogeneous autoregressive process for tracking (software) reliability growth, and their Bayesian analysis. *J. Roy. Statist. Soc. Ser. B 54, 145-146.*

AINSI/IEEE (1991), Standard glossary of software engineering terminology. *STD 729-1991.*