12-2021

NGSSEML: Non-Gaussian State Space with Exact Marginal Likelihood

Thiago R. Santos

Glaura C. Franco

Dani Gamerman

NGSSEML: Non-Gaussian State Space with Exact Marginal Likelihood

by Thiago R. Santos, Glaura C. Franco, Dani Gamerman

Abstract The number of packages/software for Gaussian State Space models has increased over recent decades. However, there are very few codes available for non-Gaussian State Space (NGSS) models due to analytical intractability that prevents exact calculations. One of the few tractable exceptions is the family of NGSS with exact marginal likelihood, named NGSSEML. In this work, we present the wide range of data formats and distributions handled by NGSSEML and a package in the R language to perform classical and Bayesian inference for them. Special functions for filtering, forecasting, and smoothing procedures and the exact calculation of the marginal likelihood function are provided. The methods implemented in the package are illustrated for count and volatility time series and some reliability/survival models, showing that the codes are easy to handle. Therefore, the NGSSEML family emerges as a simple and interesting option/alternative for modeling non-Gaussian time-varying structures commonly encountered in time series and reliability/survival studies.

Keywords: Bayesian, classical inference, reliability, smoothing, time series, software R

Introduction

Standard statistical software and packages in the State Space Model (SSM) context mainly implement Gaussian models, which are well developed in the literature. For example, the R-packages **StructTS** (Ripley, 2002), **dlm** (Petris, 2010), **dlmodeler** (Szymanski, 2014), **SSsimple** (Zes, 2019), and **MARSS** (Holmes et al., 2013) are a sample of them. The last introduces multivariate autoregressive state-space modeling or multivariate Gaussian state-space models. In spite of that, there is an increasing demand for methods or models to handle non-Gaussian and non-linear time series that can be applied to real data. In the SSM class, a general structure called Dynamic Generalized Linear Models (DGLM), proposed by West et al. (1985), attracted a great deal of interest due to its flexibility, allowing the observation distribution to belong to the exponential family. The book by Durbin and Koopman (2012) also analyzed non-Gaussian time series using importance sampling in SSM.

Many other researchers have devoted themselves, over the last decades, to auxiliary procedures to handle non-Gaussian State Space under the Bayesian approach (see, for example, Andrieu and Doucet (2002); Carvalho et al. (2010), among others). Some attempts to code NGSS include the **sspir** (Dethlefsen and Lundbye-Christensen, 2006), **pomp** (King et al., 2016), **KFAS** (Helske, 2016), **bssm** (Helske and Vihola, 2021), and **dynamichazard** (Christoffersen et al., 2021) packages in the R environment (Team, 2021) and SsfPack (Koopman et al., 1999) in the Ox software (Doornik, 1997). The SsfPack is the most consolidated package with several updated versions.

Nevertheless, even for simple structures in this class of models, the analytical tractability is easily lost, and, hence, approximations are required in order to perform inferential procedures. Thus, all packages mentioned above use intensive computational methods for making inferences for the model parameters, like sequential Monte Carlo methods or particle filters, importance sampling, particle Markov chain Monte Carlo (MCMC), etc.

In spite of this increasing number of models and packages to handle non-Gaussian SSM, a common feature shared by all of them is the need to use approximations to perform inferential procedures even in very simple cases. Gamerman et al. (2013) identified a subset of NGSS that allows for exact and analytical calculation of the marginal likelihood and the predictive and filtering distributions, called NGSSEML. This family unified several different models previously existing in the literature and included more cases. These exact computations are extremely easy to code and can be performed effortlessly and in a simple way. Applications to real data are presented, and the codes are made available by the authors in Ox language. Also, NGSSEML includes the works of Aktekin et al. (2013, 2018), de Pinho et al. (2016) for modeling count and volatility data, respectively, and was later extended by Santos et al. (2017) to include models commonly used in reliability and survival contexts.

Thus, in the face of the absence of consolidated software and packages to handle **NGSSEML**, the objective of this work is to unify the range of models encompassed and to introduce the R-package **NGSSEML**. The package illustrates how the NGSS models can be easily employed in non-Gaussian time series and reliability analysis using R under the Bayesian and classical perspectives. It also provides codes in the R environment for mainly formulating and specifying the NGSS models considered in Gamerman et al. (2013) and Santos et al. (2017). The main advantage of this package, which is designed for the specific class of NGSSEML, is that the models are implemented in a simple and flexible way. All other softwares listed above deal with state space models where the marginal likelihood can not be exactly obtained. The NGSSEML approach has the advantage of not needing to approximate the likelihood as others do, achieving that via an alternative evolution. This exact specification enables a computational gain, making the NGSSEML package an attractive option to handle non-Gaussian state space. Our approach to inference is to use (virtually) exact methods. Intensive computational methods are only required when the dimension of the unknown static parameter vector is high. In this case, the Adaptive Rejection Metropolis Sampling (ARMS) algorithm is used to approximate the marginal posterior of the static parameters (Petris, 2010). Applications to count data, volatility time series, and reliability models, such as the piecewise exponential model (PEM) and Weibull and Gamma software reliability models are presented.

The paper is organized as follows. The NGSSEML family is presented in Section 2, as well as the inferential and smoothing procedures for the latent states. Section 3 introduces the package functions. Section 4 shows illustrative examples of time series and reliability data. Finally, Section 5 presents the main conclusions and final remarks.

The model

Gamerman et al. (2013) introduced a rich class of non-Gaussian state space models with exact marginal likelihood, called here NGSSEML, based on the seminal work of Smith and Miller (1986). Inference free from approximations can be performed in the NGSSEML, and this is its main advantage compared with other NGSS procedures, such as the DGLM. Due to the model formulation, some components, such as seasonality and the effect of external covariates, are introduced as fixed effects.

A time series $\{y_t\}_{t=1}^n$ belongs to this class of models if its probability (density) function can be written as:

$$\begin{aligned} p(y_t|\mu_t, \mu_{t-1}, \ldots, \mu_1, \mathbf{Y}_{t-1}, \boldsymbol{\theta}) =& p(y_t|\mu_t, \mathbf{Y}_{t-1}, \boldsymbol{\theta}) \\ =& a(y_t, \boldsymbol{\theta}) \mu_t^{b(y_t, \boldsymbol{\theta})} \exp\left(-\mu_t c(y_t, \boldsymbol{\theta})\right), \end{aligned} \tag{1}$$

for $y_t \in S(\boldsymbol{\theta}) \subset \Re$ and $p(y_t|\mu_t, \boldsymbol{\theta}) = 0$, otherwise. Functions $a(\cdot)$, $b(\cdot)$, $c(\cdot)$, and $S(\cdot)$ are such that $p(y_t|\mu_t, \boldsymbol{\theta}) \geq 0$ and, therefore, $\mu_t > 0$, for all $t > 0$. μ_t is the state at time t and $\mathbf{Y}_t = \{Y_0, y_1, \ldots, y_t\}$ represents previously available information. It is also assumed that $\boldsymbol{\theta}$ varies in the q-dimensional parameter space Θ.

The state μ_t in Equation (1) is defined as $\mu_t = \lambda_t g(\boldsymbol{x}_t, \boldsymbol{\beta})$, where $\boldsymbol{\beta}$ are the regression coefficients (one of the components of $\boldsymbol{\theta}$), $\boldsymbol{x}_t$ is a covariate vector, and λ_t is the latent variable related to the dynamic level. The usual specification for the link function g is the logarithmic function given the positive nature of μ_t, but other link functions dictated by the application may also be used. Given λ_t, the dynamic level λ_{t+1} evolves according to the system equation

$$w \frac{\lambda_{t+1}}{\lambda_t} \mid \lambda_t, Y_t, \theta \sim \text{Beta}\left(w a_t, (1-w) a_t\right), \tag{2}$$

where a_t are values obtained in a sequential procedure and $0 < w < 1$. The parameter w is responsible for increasing the variance over time and plays a similar role to that of discount factors in the DGLM (West and Harrison, 1997). Finally, the initial prior distribution is given by $\lambda_0|Y_0 \sim \text{Gamma}(a_0, b_0)$. Thus, the full model specification is completed.

There is a wide range of distributions that belong to this class of models. It includes many commonly known discrete and continuous distributions such as Poisson, Gamma, and Normal (with static mean) but also includes many other distributions that are not so common and some reliability models. Table 1 provides the form of functions a, b, c, and S for some usual distributions in this family.

This family provides a collection of distributions for modeling a variety of real-time series and reliability data that are of practical importance, including software reliability. A detailed explanation and some illustrations can be found in de Pinho et al. (2016) and Santos et al. (2017).

Hereafter, following Theorem 1 in Gamerman et al. (2013), sequential inference for the NGSSEML will be presented as a basic result that includes the one-step-ahead predicted distributions and the filtering distribution of the latent states $\{\lambda_t\}_{t=1:n}$. If the model is defined as in (1) and (2), the following results can be obtained:

- One-step-ahead predicted (forecast) distribution of the states

$$\lambda_t | Y_{t-1}, \boldsymbol{\theta} \sim \text{Gamma}(w a_{t-1}, w b_{t-1}), \tag{3}$$

  where $0 < w < 1$.

	Models	θ	$a(y_t,\theta)$	$b(y_t,\theta)$	$c(y_t,\boldsymbol{\theta})$
Time Series	Poisson	w,β	$(y_t!)^{-1}$	y_t	1
	Gamma	w,χ,β	$y_t^{\chi-1}/\Gamma(\chi)$	χ	y_t
	Weibull	w,ν,β	$\nu(y_t)^{\nu-1}$	1	$(y_t)^{\nu}$
	Normal	w,μ,β	$(2\pi)^{-1/2}$	$1/2$	$(y_t-\mu)^2/2$
	Laplace	w,μ,β	$\frac{1}{\sqrt{2}}$	1	$\sqrt{2}\lvert y_t-\mu\rvert$
	Power Exponential (GED*)	w,ν,μ,β	$\frac{\nu}{2^{\frac{\nu+1}{\nu}}\Gamma(1/\nu)}$	$1/\nu$	$\frac{(y_t-\mu)^{\nu}}{2}$
	Generalized Gamma	w,ν,χ,β	$\nu y_t^{\nu\chi-1}/\Gamma(\chi)$	χ	y_t^{ν}
Relia-bility	PEM*	w	1	$\sum_{j=1}^{N}\chi_{ij}$	$\sum_{j=1}^{N}(t_{ij}-t_{i-1})$
	PH*	w,β	$\exp\left(\sum_{j\in R_t}\chi_{ij}x'_j\beta\right)$	$\sum_{j=1}^{N}\chi_{ij}$	$\sum_{j=1}^{N}(t_{ij}-t_{i-1})\exp(x'_{ij}\beta)$
	Weibull SR*	w,ν,β	$\nu y_t^{\nu-1}\exp(-\nu x_t\beta)$	1	$y_t^{\nu}\exp(-\nu x_t\beta)$
	Gamma SR*	w,α,β	$\frac{(y_t)^{\alpha-1}}{\Gamma(\alpha)\exp(\alpha x_t\beta)}$	α	$y_t\exp(-x_t\beta)$

Note: *PEM: Piecewise Exponential Model; PH: Proportional Hazards model; GED: Generalized Error Distribution; SR: Software Reliability.

Table 1: Special cases of the NGSSEML.

- Filtering distribution of the states

$$\lambda_t = \left(\mu_t[g(x_t^{'}\boldsymbol{\beta})]^{-1}\right)|Y_t,\boldsymbol{\theta} \sim \text{Gamma}(a_t,b_t), \tag{4}$$

where $a_t = wa_{t-1} + b(y_t,\boldsymbol{\theta})$ and $b_t = wb_{t-1} + c(y_t,\boldsymbol{\theta})g(x_t,\boldsymbol{\beta})$.

- One-step-ahead predictive density function of the observations

$$p(y_t|Y_{t-1},\theta) = \frac{\Gamma\left(b(y_t,\theta)+wa_{t-1}\right)a(y_t,\theta)[wb_{t-1}\left(g(x_t,\beta)\right)^{-1}]^{wa_{t-1}}}{\Gamma(a_{t-1})[c(y_t,\theta)+wb_{t-1}(g(x_t,\beta))^{-1}]^{b(y_t,\theta)+wa_{t-1}}}, \quad y_t \in S(\theta), \tag{5}$$

$\forall t \leq n$ where $\Gamma(\cdot)$ is the gamma function. The distribution of $\mu_t = \lambda_t g(x_t,\beta)$ can be easily obtained from the predictive and filtering distributions.

The analytical form of the predictive density function in (5) allows the exact computation of the marginal likelihood function

$$L(\theta) = L(\theta;Y_n) = \prod_{t=1}^{n}\left(p(y_t|Y_{t-1},\theta)\right). \tag{6}$$

Thus, classical and Bayesian inference for the model parameters are easily performed.

Classical inference for the static parameter vector θ can be performed through the maximum likelihood estimator (MLE) obtained by maximizing the marginal likelihood function with respect to θ. In general, since the MLE does not possess a closed-form expression, numerical maximization methods (such as the BFGS (Shanno, 1970)) are required to maximize the likelihood function.

Bayesian inference for the static parameters can be performed combining the marginal likelihood function and a prior of the static parameters, $p(\theta)$, thus obtaining the marginal posterior

$$p(\theta|Y_n) \propto L(\theta)p(\theta). \tag{7}$$

When the posterior distribution does not have a closed-form, computational methods can be used, such as the Markov chain Monte Carlo (MCMC), Adaptive Rejection Metropolis Sampling (ARMS) (Petris, 2010), or numerical integration/quadrature (Santos et al., 2017). In the latter, an inexpensive grid of points can be utilized due to the low dimensionality of θ in many special models and examples presented in this work. Table 2 shows the prior distributions for the static parameters used in each model of Table 1.

Regarding the latent state, $\boldsymbol{\lambda} = \{\lambda_t\}_{t=1:n}$, inference can be made using its predictive and filtering distributions as well as smoothing procedures with θ set as a fixed value, e.g., its respective MLE (Brockwell and Davis, 1996), sampled from its posterior distribution via MCMC output or directly. The latter approach is explained in the sequel. In order to obtain a sample of the latent state, the smoothing procedure, described in Theorem 2 of Gamerman et al. (2013), is required. It states that the distribution

	Models	θ	**Prior** ($p(\theta)$)
Time Series	Poisson	w, β	$w \sim \text{Beta}(a_w, b_w)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Gamma	w, χ, β	$w \sim \text{Beta}(a_w, b_w)$, $\chi \sim \text{Gamma}(a_\chi, b_\chi)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Weibull	w, ν, β	$w \sim \text{Beta}(a_w, b_w)$, $\nu \sim \text{Gamma}(a_\nu, b_\nu)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Normal	w, μ, β	$w \sim \text{Beta}(a_w, b_w)$, $\mu \sim \text{Normal}(m, v)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Laplace	w, μ, β	$w \sim \text{Beta}(a_w, b_w)$, $\mu \sim \text{Normal}(m, v)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Power Exponential (GED)	w, ν, μ, β	$w \sim \text{Beta}(a_w, b_w)$, $\nu \sim \text{Gamma}(a_\nu, b_\nu)$, $\mu \sim \text{Normal}(m, v)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Generalized Gamma	w, ν, χ, β	$w \sim \text{Beta}(a_w, b_w)$, $\nu \sim \text{Gamma}(a_\nu, b_\nu)$, $\chi \sim \text{Gamma}(a_\chi, b_\chi)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
Relia-bility	PEM*	w	$w \sim \text{Beta}(a_w, b_w)$
	PH*	w, β w, β	$w \sim \text{Beta}(a_w, b_w)$, $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Weibull SR	w, ν, β	$w \sim \text{Beta}(a_w, b_w)$, $\nu \sim \text{Gamma}(a_\nu, b_\nu)$ $\beta \sim \text{Normal}_q(m_{beta}, V_{beta})$
	Gamma SR	w, α, β	$w \sim \text{Beta}(a_w, b_w)$, $\alpha \sim \text{Gamma}(a_\alpha, b_\alpha)$ $\beta \sim \text{Normal}_q((m_{beta}, V_{beta})$

Note: *PEM: Piecewise Exponential Model; PH: Proportional Hazards model; GED: Generalized Error Distribution; SR: Software Reliability.

Table 2: Prior for the static parameters of the NGSSEML.

of $(\boldsymbol{\lambda}|\boldsymbol{\theta}, Y_n)$ is given by

$$p(\boldsymbol{\lambda}|\boldsymbol{\theta}, Y_n) = p(\lambda_n|\boldsymbol{\theta}, Y_n) \prod_{t=1}^{n-1} p\left(\lambda_t|\lambda_{t+1}, \boldsymbol{\theta}, Y_t\right),$$

where

$$\lambda_t - w\lambda_{t+1}|\lambda_{t+1}, \boldsymbol{\theta}, Y_t \sim \text{Gamma}\left((1-w)a_t, b_t\right), \forall t \geq 0. \tag{8}$$

In an analogous way, the marginal distribution of $(\boldsymbol{\lambda}|Y_n)$ is expressed as

$$p(\boldsymbol{\lambda}|Y_n) = \int p(\boldsymbol{\lambda}|\boldsymbol{\theta}, Y_n) p(\boldsymbol{\theta}|Y_n) d\boldsymbol{\theta}. \tag{9}$$

Once a sample $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(M)}$ is available, samples from the states are obtained retrospectively from the distribution of the states using the result in (8). More details can be found in Gamerman et al. (2013). A sample $\boldsymbol{\lambda}^{(1)}, \ldots, \boldsymbol{\lambda}^{(M)}$ of the marginal distribution in (9) is obtained in the following manner: i) sample $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(M)}$ from its marginal posterior $p(\boldsymbol{\theta}|Y_n)$. ii) sample $\boldsymbol{\lambda}^{(j)}$ from $p(\boldsymbol{\lambda}|\boldsymbol{\theta}^{(j)}, Y_n)$, $j = 1, \ldots, M$. If the dimensionality of the static parameters θ is low, the numerical calculation of $p(\theta|Y_n)$ using quadrature is indistinguishable from the exact calculation. If θ is discrete, it is an exact result. Thus, the numerical calculation of the marginal posterior distribution $p(\boldsymbol{\lambda}|Y_n)$ is a suitable approximation, whose quality depends on the approximation of $p(\theta|Y_n)$ and, hence, the dimensionality of θ and its nature.

Gamerman et al. (2013) used the Pearson and deviance residuals for model diagnostics and AIC, BIC, and DIC for model comparison in this family. It should be noted that, if MCMC methods are used, convergence checks using graphs like trace plot, autocorrelogram, etc., and tests like the Geweke and Gelman-Rubin tests are necessary.

Specification of the state space objects

The package **NGSSEML** can be downloaded and installed from GitHub or CRAN website and is then activated in R by `library("NGSSEML")`. Assuming that the data are available in the current environment, the NGSSEML can be set up using the formulas and family arguments presented in the next subsections.

Estimation of the static parameters

This subsection presents the estimation of the static parameters $\boldsymbol{\theta}$ under classical and Bayesian approaches. The classical procedure uses the BFGS algorithm, implemented in the "optim" function of the R package, to obtain the MLE. The Bayesian procedure uses numerical integration/quadrature with an inexpensive grid of points. The functions 'ngssm.mle' and 'ngssm.bayes' perform the classical and Bayesian inferences, respectively, shown in the previous section and utilize the function 'LikeF' (the marginal likelihood function) in Eq. (6), internally.

Basically, the functions of **NGSSEML** work with a very simple specification of a few arguments. We need to specify the `formula`, `data` (data frame) arguments, and `model`, as well as the function 'lm' for linear regression. For piecewise exponential models in reliability analysis, we cannot forget to include the `breaks` and `events` arguments.

The function 'ngssm.mle' performs the marginal likelihood estimation of the static parameters of the model using Eq. (6), and can be executed by the following command:

```
> ngssm.mle(formula, data, na.action="na.omit", pz = NULL, nBreaks = NULL,
+ model = "Poisson", StaPar = NULL, amp = FALSE, a0 = 0.01, b0 = 0.01,
+ ci = 0.95, LabelParTheta = NULL, verbose = TRUE, method = "BFGS",
+ hessian = TRUE, control = list(maxit = 30000, temp = 2000,
+ trace = FALSE, REPORT = 500))
```

The output of the model fit presents the maximum likelihood estimators, standard errors, Z statistics, and asymptotic confidence intervals of the model parameters. Furthermore, it allows the user to change the control settings, choose the optimizing algorithm, and set arguments as the hyperparameters a_0 and b_0, the confidence level, etc.

The function 'ngssm.bayes' performs the Bayesian estimation of the static parameters of the model using the marginal posterior in Eq. (7), and can be run by the following command:

```
> ngssm.bayes(formula, data, na.action = "na.omit", pz = NULL, nBreaks = NULL,
+ model = "Poisson", StaPar = NULL, amp = FALSE, a0 = 0.01, b0 = 0.01, prw = c(1, 1),
+ prnu = NULL, prchi = NULL, prmu = NULL, prbetamu = NULL, prbetasigma = NULL,
+ lower = NULL, upper = NULL, ci = 0.95, pointss = 10, nsamplex = 1000, mcmc = NULL,
+ postplot = FALSE, contourplot = FALSE, LabelParTheta = NULL, verbose = TRUE)
```

The model fit's output provides the Bayesian estimators, posterior mean and median, standard error, and percentile credibility intervals for NGSSEML parameters, and the posterior samples of the static parameters of the NGSSEML using multinomial sampling. The main functions of the model fit provide an object class "ngssm", associated with print, summary, and extract further information. Besides, the user can choose the number of grid points (the argument `pointss`) of the quadrature and samples (the argument `nsamplex`) and set the hyperparameters of prior distributions (the arguments `a0,b0,prw,prnu,prchi,prmu,prbetamu,prbetasigma`), the credibility level, etc. If `mcmc=TRUE` or the number of points in the grid is very high, the ARMS algorithm is executed instead of the quadrature method. Samples of the posterior distribution of the latent states using the smoothing procedure are calculated as described in Eq. (9).

Estimation of latent states

The filtering, smoothing, and plot functions for the NGSSEML are shown in this subsection. For these functions, the static parameters are estimated with the whole data.

The function 'FilteringF' computes the shape and scale parameters of the one-step-ahead forecast and filtering distributions of the latent states using the filters in Eq. 3 and 4, respectively, for several models. It can be called by the command below.

```
> FilteringF(formula, data, na.action = "na.omit", pz = NULL, nBreaks = NULL,
+ model = "Poisson", StaPar = NULL, a0 = 0.01, b0 = 0.01, amp = FALSE,
+ distl = "PRED", splot = FALSE)
```

The output presents the shape and scale parameters of the one-step-ahead forecast and filtering distributions of the latent states. Furthermore, it allows the user to modify the arguments of the function as the hyperparameters a_0 and b_0, etc.

The function 'SmoothingF' provides an exact sample of the smoothing distribution of the latent states in Eq. (9), and its command is

```
> SmoothingF(formula, data, na.action = "na.omit", pz = NULL, nBreaks = NULL,
+ model = "Poisson", StaPar = NULL, Type = "Cond", a0 = 0.01, b0 = 0.01,
+ amp = FALSE, samples = 1, ci = 0.95, splot = FALSE)
```

The function provides an object with an exact sample of the joint distribution of the states. The user can choose the arguments as the hyperparameters a_0 and b_0, type of the distribution of the latent states - margin/conditional on the static parameters, the confidence/credibility level, etc. If the number of samples is greater than 1, some summaries of the state samples are returned. If the argument `splot=TRUE`, a graph with the states' smoothed estimates is shown.

The function 'PlotF' builds graphs with smoothed/filtered estimates of the latent states and can be run by the following command:

```
> PlotF(formula, data, na.action = "na.omit", pz = NULL, nBreaks = NULL,
+ plotYt = TRUE, axisxdate = NULL, transf = 1, model = "Poisson", posts,
+ Proc = "Smooth", Type = "Marg", distl = "PRED", a0 = 0.01, b0 = 0.01,
+ ci = 0.95, startdate = NULL, enddate = NULL, Freq = NULL, ...)
```

The function returns a graph with smoothed or filtered estimates of the latent states. The user can change the arguments as the hyperparameters a_0 and b_0, the type of the distribution of the latent states - margin/conditional on the static parameters, the procedure between smoothing and filtering, the confidence/credibility level, etc. Other options related to graph edition, such as color, type line, labels, can be inserted into the function 'PlotF' using the argument ellipsis (...). To save space in this manuscript, they are not shown. We will try to provide more details in the examples.

Examples

In this section, we present four examples to illustrate the use of the proposed package for count, volatility, lifetime, and software reliability data. The count time series is the poliomyelitis data in the USA. The volatility series refers to the return data set from the Petrobras stock market. The lifetime data set (GTE) are the daily failure times of 125 telecommunication systems. The "SYS1" data set are times between 136 successive computer software failures. All examples are widely used in the literature of their respective areas. We present parameter estimation, the goodness of fit for the adjusted models and predictions. For the poliomyelitis data, we also provide comparisons with other softwares available in the literature.

As previously explained, this software is designed for the specific class of non-Gaussian state space models with exact marginal likelihood. The NGSSEML may seem analytically more complex in comparison with other procedures, which present in general simpler equations. Nevertheless, although simplest in form, these procedures frequently need more calculations and heavier computational effort to produce similar results to our method. Based on an alternative evolution, our approach requires less approximation, and thus, inference for static parameters is almost indistinguishable from the (unobtainable) exact form. Readers interested in the advantages and comparisons of the NGSSEML against alternatives should refer to Gamerman et al. (2013) and Santos et al. (2017).

Count data

A Poisson NGSSEML model is used to fit the monthly data of the number of poliomyelitis cases in the USA, shown in Figure 1, from 1970/01 to 1983/12 (168 observations). The polio data is a well-known example in the count time series literature and was first analyzed by Zeger (1988). The objective is to explain the number of poliomyelitis cases through covariates and make predictions. The covariates are the deterministic trend centered at 73, annual cosine and sine, and semiannual cosine and sine. The intercept is inserted in the model as a dynamic level. The confidence and credibility levels for the intervals are fixed at 0.95. The state initial condition is $\lambda_0|Y_0 \sim \text{Gamma}(a_0, b_0)$, where $a_0 = 0.3$ and $b_0 = 0.1$ that can be specified in the functions 'ngssm.mle' and 'ngssm.bayes' by the arguments $a0$ and $b0$. The static parameter vector in this example is $\boldsymbol{\theta} = (w, \beta_1, \beta_2, \beta_2, \beta_3, \beta_4, \beta_5)$, so it is necessary to specify a joint probability density for it, that is, a joint prior $p(\boldsymbol{\theta})$. Assuming that the static parameters are independent and based on the priors available for the package in Table 2, the marginal priors for its components are $w \sim \text{Beta}(1,1)$ and $\beta_j \sim N(0,10)$, for $j = 1,\ldots,4$. Parameter w controls the information loss over time, while parameters β are regression coefficients associated with the covariates for capturing trend and seasonal patterns of the count time series. The prior hyperparameters are chosen to set vague priors for the model parameters.

The MLE for the static parameters of the NGSSM in the poliomyelitis data can be obtained using the BFGS algorithm (default: `method = "BFGS"`) implemented in the optim function, which is the default of 'ngssm.mle' function. It is necessary to specify `data1 = data.frame(Ytm,Xtm)` in the argument data, the formula `Ytm ~ CosAnnual + SinAnnual + CosSemiAnnual + SinSemiAnnual`, and the model `model = "Poisson"`. The arguments `StaPar = c(0.79,-0.11,-0.49,0.18,-0.38)`, `a0 = 0.01`, `b0 = 0.01`, and `ci = 0.95` are optional. The name of the variables in the argument must coincide with the name in the data. The model fit is stored in the `fit` object. The code for classical inference is given below. We should note that a graph with the filtered estimates of the latent states is returned when the 'FilteringF' function is called.

```
> library(NGSSEML)
> data(Polio_data)
> Xtm = Polio_data[, 3:7]
> Xtm[,1] = (1:168-73)/168
> Ytm = Polio_data$y
> Ztm=NULL
## CosAnnual, SinAnnual, CosSemiAnnual, SinSemiAnnual
## LabelParTheta = c("w", "Beta1", "Beta2", "Beta3", "Beta4")
StaPar = c(0.79, -0.11, -0.49, 0.18, -0.38)
> a0 = 0.2
> b0 = 0.1
> ci = 0.95
> data1=data.frame(Ytm,Xtm)
## Fit
> fit = ngssm.mle(Ytm ~ CosAnnual + SinAnnual + CosSemiAnnual + SinSemiAnnual,
+ data = data1, model = model, pz = NULL, StaPar = StaPar,
+ a0 = a0, b0 = b0, ci = ci)
> esttheta = as.numeric(fit[,1])
> filt = FilteringF(Ytm ~ Trend + CosAnnual + SinAnnual + CosSemiAnnual
+ SinSemiAnnual, data = data1, StaPar = esttheta,
+ model = "Poisson", a0 = a0, b0 = b0,
+ distl = "FILTER", splot = TRUE)
> filtest = (filt[1,]/filt[2,])
```

The results are presented in Table 3, showing the MLE's and their respective 95% confidence intervals for w and $\beta's$. Bayesian estimation can be performed using quadrature rules with a grid of 6 points (`pointss = 6`), providing a sample of 2,000 draws (`nsamplex = 2000`) of the marginal posterior distribution of the static parameters. This can be performed without the use of intensive computational methods to approximate the posterior distribution as, is usually the case in other non-Gaussian state-space models. We should set the arguments in the function as the data frame `data1`, the formula `Ytm ~ CosAnnual + SinAnnual + CosSemiAnnual + SinSemiAnnual`, and the model `model = "Poisson"`. The marginal priors are: $w \sim U(0,1) \equiv \text{Beta}(1,1)$, and $\beta_j \sim N(0,10)$, for $j = 1,\ldots,4$, specified in the function 'ngssm.bayes' by the following arguments `prw = c(1,1)` (two shape parameters of a Beta distribution), `prbetamu = rep(0,4)`, and `prbetasigma = diag(10,4,4)` (two parameters of a multivariate Normal distribution). If `postplot = TRUE` and `contourplot = TRUE`, the graph of the marginal posterior and the contour plot are provided. If `verbose = TRUE`, the estimation results are shown; otherwise, they are omitted. The code is presented below.

```
> library(NGSSEML)
> data(Polio_data)
> Xtm=Polio_data[,3:7]
> Xtm[,1] = (1:168-73)/168
> Ytm = Polio_data$y
> Ztm = NULL
## CosAnnual, SinAnnual, CosSemiAnnual, SinSemiAnnual
## LabelParTheta=c("w", "Beta1", "Beta2", "Beta3", "Beta4")
> StaPar = c(0.79, -0.11, -0.49, 0.18, -0.38)
> a0 = 0.2
> b0 = 0.1
## points
```

```
> pointss = 6
## posterior sample
> nsamplex = 2000
## Cred. level
> ci = 0.95
> data1 = data.frame(Ytm, Xtm)
## Bayesian fit
> fitbayes = ngssm.bayes(Ytm ~ CosAnnual + SinAnnual +
 CosSemiAnnual + SinSemiAnnual,
+ data = data1, model = "Poisson", pz = NULL, StaPar = StaPar, a0 = a0, b0 = b0,
+ prw = c(1, 1), prbetamu = rep(0, 4), prbetasigma = diag(10, 4, 4), ci = ci,
+ pointss = pointss, nsamplex = nsamplex, postplot = TRUE, contourplot = TRUE,
+ verbose = TRUE)
```

The results of the Bayesian inference are also presented in Table 3, showing the posterior mean and the credibility intervals. The coefficient regression related to the determinist trend is not significant at the 5% significance level, so we decide to remove it from the model. Analyzing Table 3, we can see that both for β_2 and β_4 associated with the trigonometric annual and semiannual sine covariates, respectively, are negative and seem to be significant under both approaches at the same significance level. We keep the cosine covariates to preserve the trigonometric representation of seasonality. The magnitude and significance of the regression coefficients are similar to those obtained by other works in the count time series literature (Davis and Wu, 2009). Furthermore, the classical estimates are close to the Bayesian estimates. If the ARMS algorithm is used, we need to make a convergence check using graphs like trace plot, autocorrelogram, etc., and tests like the Geweke and Gelman-Rubin tests in the R-package **coda**.

The R codes for the smoothing function below are used to build the graphs presented in Figure 1, for the Polio data with the smoothed estimates for the observations and mean under the Bayesian approach. The first three arguments of the function 'PlotF' refer to the Bayesian model obtained with 'ngssm.bayes'. The commands `posts = posts`, `axisxdate = x`, `Proc = "Smooth"`, `Type = "Marg"`, `startdate = "1970/01/01"`, `enddate = "1983/12/31"`, and `Freq = "months"` indicate, respectively, the sample of the marginal posterior of the static parameters, the vector of monthly dates, the smoothed distribution for the states, the chosen distribution for the latent states, the start and end dates, and the frequency of the observations. The last arguments refer to the type of lines, colors, and legends in the graph. We can see that the smoothed mean trajectory in Figure 1 follows well the behavior of the time series.

```
> posts = fitbayes$samplepost
> x = seq(as.Date("1970/01/01"), as.Date("1983/12/31"), "months")
> PlotF(Ytm ~ CosAnnual + SinAnnual + CosSemiAnnual + SinSemiAnnual,
+ data = data1, model = "Poisson", axisxdate = x, Proc = "Smooth", Type = "Marg",
+ posts = posts, startdate = "1970/01/01", enddate = "1983/12/31",
+ Freq = "months", type = 'l', col = c("black", "blue","lightgrey"), xlab = "Months",
+ ylab = "The number of poliomyelitis cases", xlim = NULL, ylim = c(0, 15),
+ Lty = c(1, 2, 1), lwd = c(2, 2, 2), cex = 0.68)
```

θ	MLE	95% Conf. Int.	Posterior Mean	95% Cred. Int.
w	0.793	[0.711; 0.874]	0.783	[0.713; 0.843]
β_1	-0.116	[-0.324; 0.092]	-0.117	[-0.314; 0.015]
β_2	-0.488	[-0.717; -0.259]	-0.491	[-0.707; -0.344]
β_3	0.175	[-0.024; 0.374]	0.176	[-0.014; 0.301]
β_4	-0.385	[-0.577; -0.193]	-0.387	[-0.568; -0.265]

Table 3: Point and interval (level 95%) estimation for the Poisson model fitted to the polio count data.

Figure 2 shows the standardized Pearson residuals and its autocorrelogram for model diagnostic. The residuals are not correlated and distributed between -2 to 2, except the points indicated in the graph (a) that coincide with the largest values of the series in Figure 1. Further investigation of these points could be performed, and the inclusion of interventions may improve the model fit. The R code for the diagnostic analysis in this example is

```
> esttheta = as.numeric(fit[,1]); n = length(Ytm)
> filt = FilteringF(Ytm ~ CosAnnual + SinAnnual + CosSemiAnnual + SinSemiAnnual,
```

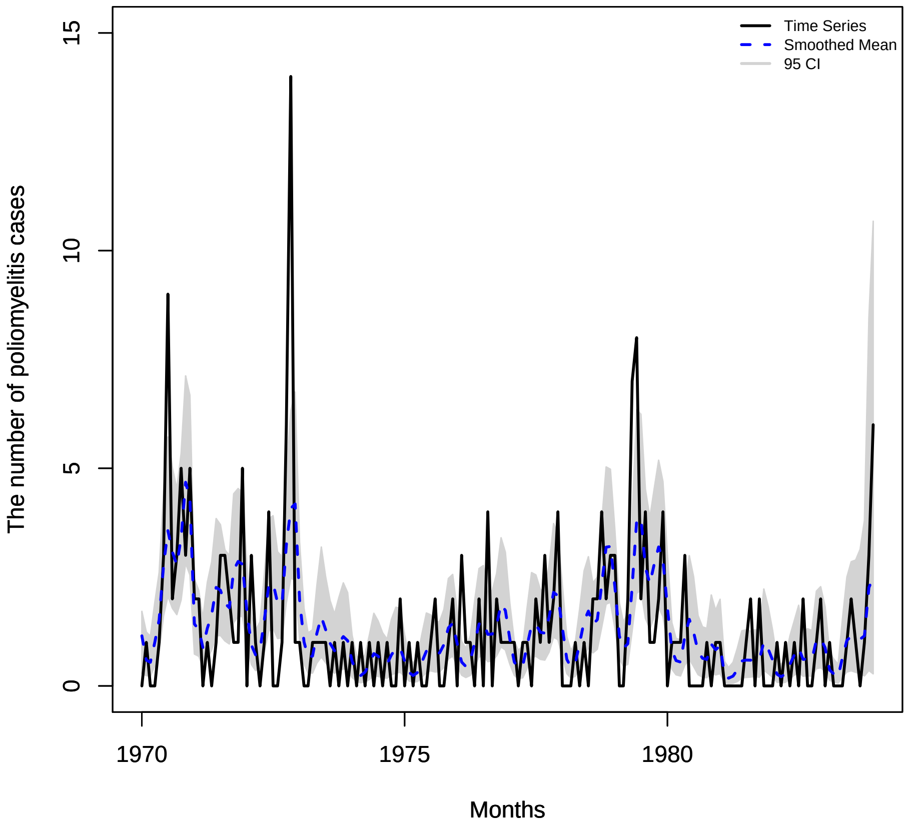

Figure 1: The polio data: The full line represents the polio monthly time series in the USA from 1970/01 to 1983/12 (168 observations). The dashed line and the grey area indicate the smoothed means and their respective 95% percentile credibility intervals using the smoothing procedure and the quadrature method.

```
+ data = data1, StaPar = esttheta, model = "Poisson", distl = "FILTER",
+ splot = TRUE)
> filtest = (filt[1,]/filt[2,])
> pred = FilteringF(Ytm ~CosAnnual + SinAnnual + CosSemiAnnual + SinSemiAnnual,
+ data = data1, StaPar = esttheta, model = "Poisson", a0 = a0, b0 = b0,
+ distl = "PRED")
> predonestepahead = (pred[1,]/pred[2,])
> residuals = (Ytm-predonestepahead)/sqrt(predonestepahead)
> summary(residuals)
> residuals = scale(residuals[1:n])
> par(mfrow = c(1, 2))
> x = seq(as.Date("1970/01/01"), as.Date("1983/12/31"), "months")
> plot(x[1:n], residuals, xlab = "Months", ylab = "Standardized Pearson residuals")
> points(x[7], residuals[7], col = "red", lwd = c(2))
> points(x[35], residuals[35], col = "red", lwd = c(2))
> points(x[74], residuals[74] ,col = "red", lwd = c(2))
> points(x[113], residuals[113], col = "red", lwd = c(2))
> title("(a)")
> acf(residuals, ci = 0.99, main = "(b)")
```

For comparison purposes, Table 4 displays some statistics for evaluating the in-sample performance of the fitted model using the KFAS, sspir and our software. The statistics are the square root of the mean square error SMSE $= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, the mean absolute error MAE $= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$, and the mean absolute percentage error, MAPE $= \frac{1}{n}\sum_{i=1}^{n}|(y_i - \hat{y}_i)/y_i|$, where $\hat{y}_i$, $i = 1, \ldots, n$, are the fitted values. The Poisson model with the logarithmic link function, level component, and covariates is specified

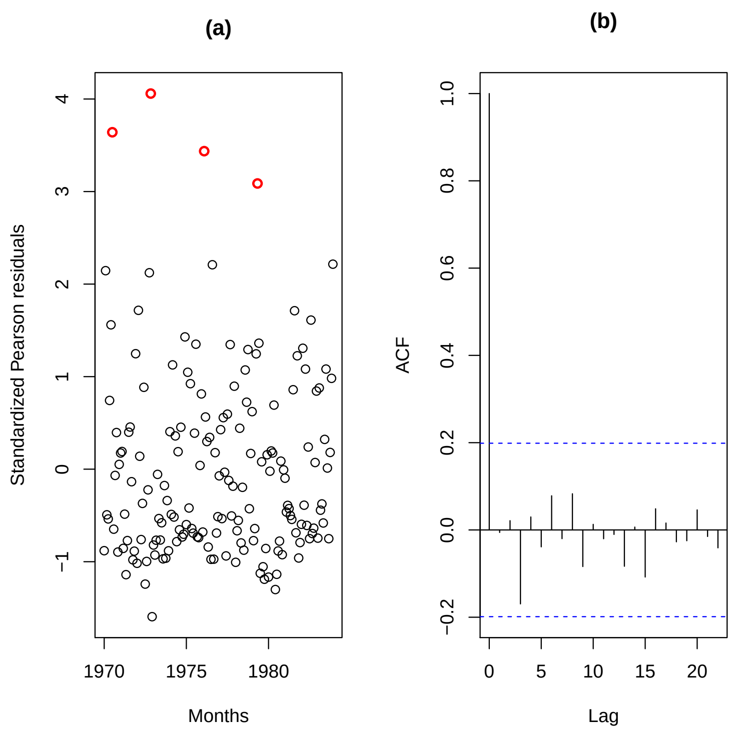

Figure 2: Polio data. (a) The full line represents the standardized Pearson residuals. The marked points are the largest values of the residuals; (b) The autocorrelogram of the standardized Pearson residuals. The marked points indicate the highest residual values.

for the three packages, as described in this subsection. The static parameters were fixed at the MLE and posterior mean for our package and the MLE for the other packages to get the smoothed/filtered mean. The results are similar for the KFAS and sspir packages, which use the maximum likelihood estimation via importance sampling methods. Our package provided the predicted values (the filtered estimates) using the classical and Bayesian approaches described in this paper and presented the best results for the SMSE, MAE, and MAPE. Convergence of the maximization algorithm is fast and stable because the likelihood is a well-behaved and exact function. The joint likelihood function (6) of our Poisson model is a product of the negative binomial distributions given by Eq. (5).

	SMSE	**MAE**	**MAPE**
KFAS*	1.776	1.179	0.638
NGSSEML*	1.280	0.890	0.468
NGSSEML**	1.264	0.880	0.462
sspir*	1.788	1.202	0.648

Note: *MLE; **The posterior mean.

Table 4: In sample SMSE, MAE, and MAPE for the Poisson model using the KFAS, NGSSEML, and sspir packages.

Volatility data

The second example refers to the daily return data from Petrobras stock market from 2000/01/06 to 2008/29/01 (1999 observations), which can be obtained on the website finance.yahoo.com/. The return at time t is defined as $y_t = \ln\left(\frac{P_t}{P_{t-1}}\right)$, where P_t is the daily closing spot price. Data collection irregularity due to holidays and weekends will be ignored. Figure 3 presents the time series plot of y_t. A distinctive feature of financial series is that they usually present non-constant variance or volatility. Thus, this series can be modeled using distributions with a heavy tail, such as GED, which does not belong to the exponential family. The aim is to estimate the daily volatility of Petrobras return time series.

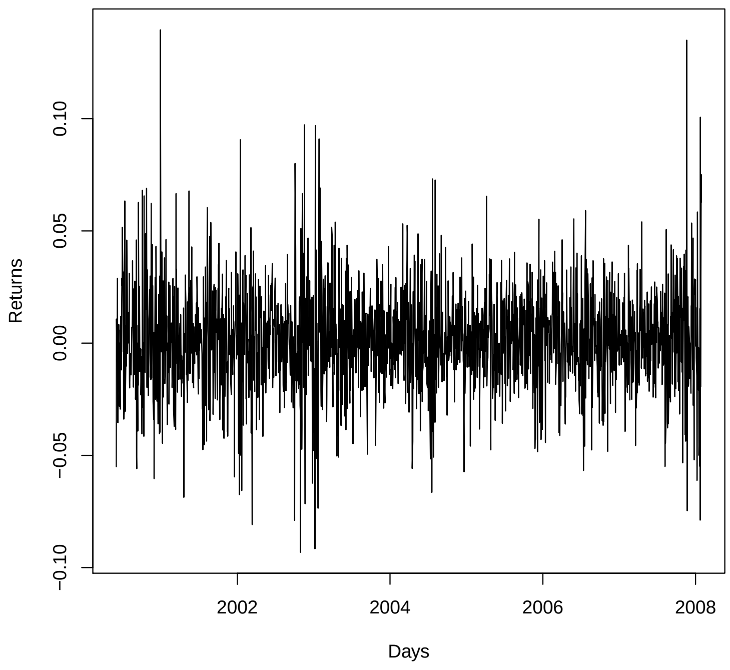

Figure 3: Petrobras return data: This time series is the Petrobras asset's daily return data in the Brazilian stock market from 2000/01/06 to 2008/29/01 (1999 observations).

The maximum likelihood estimation for the Petrobras return data is performed using the R code below with the following arguments in the function 'ngssm.mle': `Ytm~1`, `data = data.frame(Ytm)`, and `model = "GED"`. The state initial condition is $\lambda_0|Y_0 \sim \text{Gamma}(a_0, b_0)$, where $a_0 = b_0 = 0.01$ (the default values) that can be specified in the functions 'ngssm.mle' and 'ngssm.bayes' by the arguments $a0$ and $b0$.

```
> library(NGSSEML)
> data(Return_data)
> plot(Return_data[,2], Return_data[,1], type = 'l', xlab = "Days",
+ ylab = "Returns")
> Ytm = Return_data$Rt
> Xt = NULL
> Zt = NULL
> model = "GED"
## LabelParTheta = c("w", "nu")
> StaPar = c(0.9, 1)
> a0 = 0.01
> b0 = 0.01
> ci = 0.95
> fit = ngssm.mle(Ytm ~ 1, data=data.frame(Ytm), model = model, a0 = a0, b0 = b0,
+ ci = ci, verbose = FALSE)
```

Bayesian estimation for the Petrobras return data is obtained via quadrature with 15 points (argument `pointss=15`) and a sample of 1000 draws (`nsamplex=1000`) of the marginal posterior distribution. The state initial condition is $\lambda_0|Y_0 \sim \text{Gamma}(0.01, 0.01)$. Analogously to the previous example, the first arguments, which refer to the marginal prior distribution (see Table 2) for the static parameter vector $\boldsymbol{\theta} = (w, \nu)$ are $w \sim U(0,1) \equiv \text{Beta}(1,1)$ and $\nu \sim \text{Gamma}(0.01, 0.01)$, which can be specified by the arguments `prw = c(1,1)` (two shape parameters of a Beta distribution) and `prmu=rep(0.01,0.01)` (two parameters of a Gamma distribution with the mean equals to one). The difference here is that `model = "GED"`. Again, the prior hyperparameters are chosen to set vague priors for the model parameters. A summary of the estimates was contained in the objects `fitbayes` (Bayesian fit). The code is presented below.

```
> library(NGSSEML)
```

θ	MLE	95%Conf. Int.	Posterior Mean	95% Cred. Int.
w	0.949	[0.931; 0.966]	0.947	[0.928; 0.962]
ν	1.601	[1.458; 1.745]	1.606	[1.467; 1.743]

Table 5: Point and interval estimation for the GED model fitted to the Petrobras series.

```
> data(Return_data)
> Ytm = Return_data$Rt
> Date = Return_data$Date
> Xtm = NULL
> Ztm = NULL
## LabelParTheta = c("W", "nu")
> StaPar =c (0.9,1)
> a0 = 0.01
> b0 = 0.01
## points
> pointss = 15
## posterior sample size
> nsamplex = 1000
## Cred. level
> ci = 0.95
## Bayesian fit
> fitbayes = ngssm.bayes(Ytm ~ 1, data = data.frame(Ytm), model = "GED", pz = NULL,
+ StaPar = StaPar, a0 = a0, b0 = b0, prw = c(1, 1), prnu = c(0.01, 0.01), ci = ci,
+ pointss = pointss, nsamplex = nsamplex, postplot = TRUE, contourplot = TRUE,
+ verbose  =TRUE)
```

Table 5 presents the point and the interval estimates for the GED model fitted to the Petrobras return data under the classical and Bayesian approaches obtained by the above codes. Figure 4 shows the marginal posterior distributions and contour plot of the joint posterior distribution of the parameters w and ν, respectively, with `postplot = TRUE` and `contourplot = TRUE` in the code of the Bayesian approach. Parameter w controls the information loss over time, while r is the shape parameter of the GED model and controls its tails. If $\nu < 2$, the GED is a heavy-tailed distribution. It is the case of the Petrobras return data, as expected, since the estimate for ν is around 1.6 and the upper limit of the 95% intervals for ν is smaller than 2.

To illustrate the usage of the 'PlotF' function in this example, a detailed description of its arguments is given. The first three entries in the 'PlotF' function refer to the data and model used. As there are no explanatory variables to be inserted in the volatility, `pz = NULL`. A date vector for the x-axis was specified in the `axisxdate` argument. The time series was not inserted in the plot, thus `plotYt = FALSE`. A transformation of -0.5 was applied to the estimates of the latent states to get the volatility. The chosen distribution for the latent states was conditional on the static parameters and marginal (`Type = "Marg"`). A sample of the static parameters was set as `posts = fitbayes$samplepost`, and the latent states distribution was the smoothed (`Proc = "Smooth"`). The `started` date was set as `'2000-06-01'` and the `enddate` as `'2008-01-29'`. The frequency of data is daily (`Freq = "days"`). The remaining arguments refer to type of lines, colors, and legends in the graph. We can see that the peaks in Figure 5 correspond to periods of crisis known in the literature and are pointed out by the model.

```
## Smoothing
> set.seed(1000)
> posts = fitbayes$samplepost
> PlotF(Ytm ~ 1, data = data.frame(Ytm), model = "GED", pz = NULL,
+ axisxdate = Return_data$Date, plotYt = FALSE, transf = -0.5, Proc = "Smooth",
+ Type = "Marg", posts = posts, startdate = '2000-06-01', enddate = '2008-01-29',
+ Freq = "days", typeline = 'l', col = c("black", "blue", "lightgrey"),
+ xlab = "Days", ylab = expression(paste(hat(sigma)[t])), xlim = NULL,
+ ylim = c(0.02,0.10), lty = c(1, 2, 1), lwd = c(2, 2, 2), cex = 0.68)
```

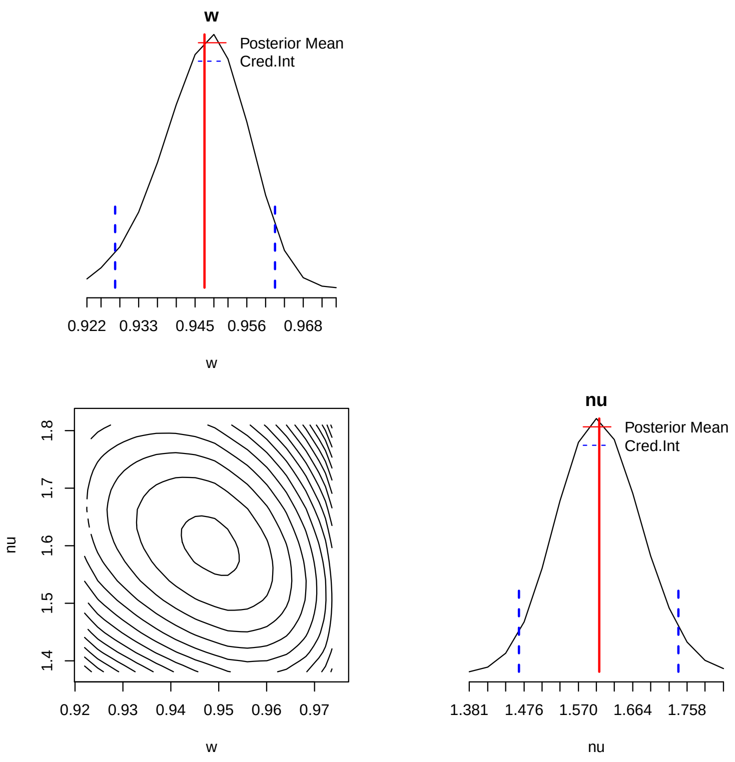

Figure 4: Petrobras return data: The marginal posterior distribution of the parameters w and ν and their respective contour plot, obtained by the quadratures.

Lifetime data

The third example refers to the daily failure times of 125 telecommunication systems installed by the GTE corporation in a pre-specified time period (Kim and Proschan, 1991). Seventeen failures were observed out of the 125 observations. The purpose is to estimate the failure rate for the GTE data.

We specify the formula `Ytm ~ Event` in the first argument of 'ngssm.mle' function. The data must contain columns with the times and events. We should also define the model (`model = "PEM"`) and obtain the breaks (one per interval) using the function 'GridP' with the times, events, and `nT = NULL` (the number of one interval per failure). The breaks are automatically inserted into the functions 'ngssm.mle' and 'ngssm.bayes' with `nBreaks = NULL`. The maximum likelihood estimation for the GTE data is performed using the R code below.

```
> library(NGSSEML)
> data(gte_data)
## Times
> Ytm = gte_data$V1
> Xtm = NULL
> Ztm = NULL
> amp = FALSE
## Event: failure, 1
> Event = gte_data$V2
> Break=NGSSEML:::GridP(Ytm, Event, nT = NULL)
## LabelParTheta = c("w")
> StaPar = c(0.73)
> a0 = 0.01
> b0 = 0.01
> ci = 0.95
> fit = ngssm.mle(Ytm ~ Event, data = data.frame(Ytm,Event), model = "PEM",
+  nBreaks = NULL, amp = amp, a0 = a0, b0 = b0, ci = ci, verbose = FALSE)
```

Bayesian estimation for the GTE data is built using quadrature with a grid of 50 points (`pointss = 50`) and a sample of 1000 draws (`nsamplex = 1000`). The state initial condition is $\lambda_0|Y_0 \sim \text{Gamma}(a_0, b_0)$, where $a_0 = b_0 = 0.01$ are specified in the functions by the arguments $a0$ and

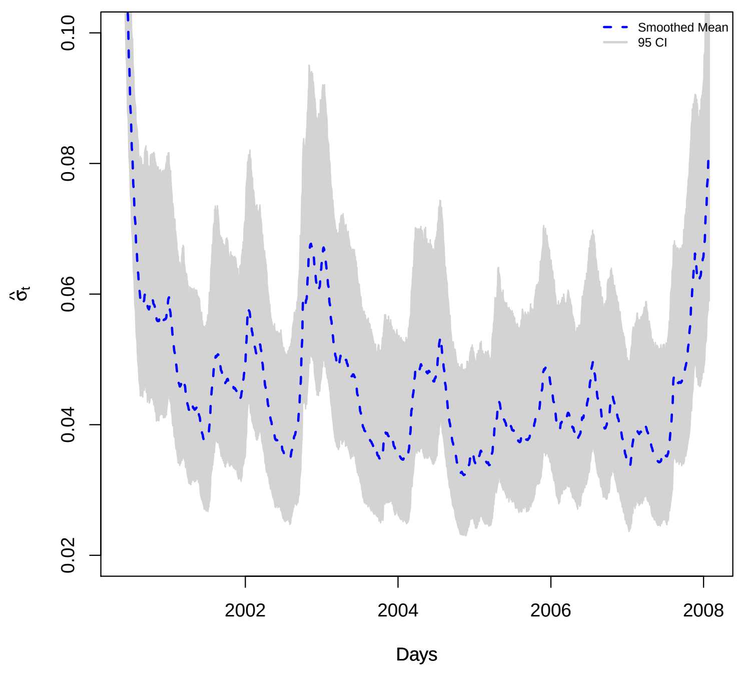

Figure 5: Petrobras return data: The dashed and dotted lines represent the smoothed estimate and the one-step-ahead prediction of the stochastic volatility ($\hat{\sigma}_t = \hat{\lambda}_t^{1/\hat{\nu}}$), obtained by the fit of the GED model under the Bayesian approach. The grey area indicates the 95% percetile credibility interval.

$b0$, and the prior is $w \sim U(0,1) \equiv \text{Beta}(1,1)$. Both priors are specified in the function 'ngssm.bayes' in a similar manner of the previous examples to get vague priors. The formula `Ytm ~ Event` contains the times and events, and the breaks are automatically inserted with `nBreaks = NULL`. The R code is below.

```
> library(NGSSEML)
> data(gte_data)
## Times
> Ytm = gte_data$V1
## Event: failure, 1
> Event = gte_data$V2
> Break = NGSSEML:::GridP(Ytm, Event, nT = NULL)
> Xtm = NULL
> Ztm = NULL
> amp = FALSE
## LabelParTheta = c("w")
> StaPar = c(0.5)
> lower = c(0.01)
> upper = c(0.99)
> a0 = 0.01
> b0 = 0.01
## points
> pointss = 50
## Number of samples from the posterior
> nsamplex = 1000
## Bayesian fit
> fitbayes = ngssm.bayes(Ytm ~ Event, data = data.frame(Ytm,Event), model = "PEM",
+ StaPar = StaPar, amp = amp, a0 = a0, b0 = b0, prw = c(1,1), pointss = pointss,
+ nsamplex = nsamplex, postplot = TRUE, contourplot = FALSE, verbose = TRUE)
```

Table 6 presents the point and the interval estimates for the PEM fitted to the GTE data under the classical and Bayesian approaches. Parameter w controls the information loss over time in the same manner as the previous application. If `pz = TRUE`, w becomes w_i for each interval, weighted by the interval width. These estimates are stored in the objects "fit" (classical fit) and "fitbayes" (Bayesian fit) of the previous codes.

The code below is used to build Figure 6, which shows the smoothed failure rate estimates under the Bayesian approach, with a 95% credibility interval (the grey area). As already mentioned, the first three entries in the 'PlotF' function refer to the data and model used. The date for the x-axis was specified in `axisxdate = Break[1:17]`. Once more, the chosen distribution for the latent states was conditional on the static parameters and marginal (`Type = "Marg"`), and the latent states distribution was the smoothed (`Proc = "Smooth"`). A sample of the static parameters was built as posts=fitbayes$samplepost. The remaining arguments refer to type of lines, colors, and legends in the graph. We can note that the failure rate is decreasing, as expected.

```
> posts = fitbayes$samplepost
> set.seed(1000)
> PlotF(Ytm ~ Event, data = data.frame(Ytm, Event), axisxdate = Break[1:17],
+ model = "PEM", Proc = "Smooth", Type = "Marg",  posts = posts, typeline = 's',
+ col = c("black", "blue", "lightgrey"),  xlab = "Time to Failure (Days)",
+ ylab = "Failure rate", xlim = c(0, 139), ylim = c(0, 0.008), lty = c(1, 2, 1),
+ lwd = c(2, 2, 2), cex = 0.68)
```

θ	MLE	95%Conf. Int.	Posterior Mean	95% Cred. Int.
w	0.773	[0.518; 1.000]	0.752	[0.503; 0.956]

Table 6: Point and interval (level 95%) estimation for the PEM fitted to the GTE data.

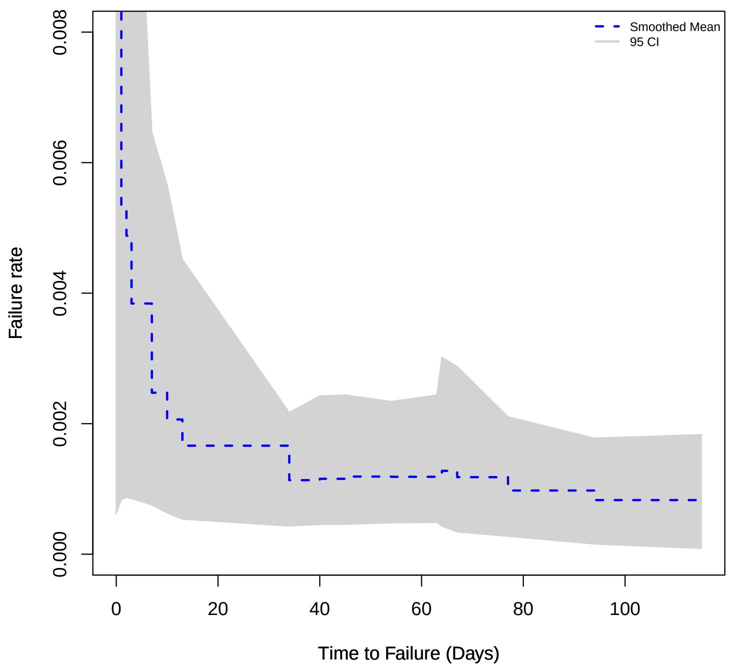

Figure 6: The full line indicates the smoothed estimate of the failure rate for the GTE data, and the grey area is its respective 95% credibility intervals, obtained by the quadratures.

Software reliability data

The Weibull software reliability model is used here for modeling the times between 136 successive computer software failures of "SYS1" data (Chang and Liu, 2009) and for estimating the mean time. A very small constant is added to failure times in order to avoid the case of a zero value in the likelihood function, but the value of the constant did not change the estimation results.

The state initial condition is $\lambda_0|Y_0 \sim \text{Gamma}(0.01, 0.01)$ in a similar way to the previous examples. The arguments of the 'ngssm.mle' that should be set are the formula `Ytm~Xtm`, `data.frame(Ytm,Xtm)`, and `model = "SRWeibull"`. An initial static parameter vector is optional (`StaPar = c(0.9,0.7,0.01)`). The maximum likelihood estimation for the "SYS1" data is performed using the R code below.

```
> library(NGSSEML)
> data(sys1_data)
> Ytm = sys1_data[,1]+0.00001
> Xtm = sys1_data[,2]
> Zt = NULL
> model = "SRWeibull"
## LabelParTheta = c("w", "alpha", "Beta1")
> StaPar = c(0.9, 0.7, 0.01)
> fit = ngssm.mle(Ytm ~ Xtm, data = data.frame(Ytm,Xtm), model = model,
+ StaPar = StaPar)
```

Bayesian estimation for the "SYS1" data is obtained via quadrature rules with a grid of 15 points (`pointss=15`) and a sample of 1000 draws of the marginal posterior distribution (`nsamplex=1000`). As in the chassical fit, the following arguments have to be inserted in the 'ngssm.bayes' function: `Ytm~Xtm`, `data = data.frame(Ytm,Xtm)`, and `model = "SRWeibull"`. An initial static parameter vector is optional (`StaPar = c(0.98,0.75,0.02)`. The marginal priors for the static parameter vector $\boldsymbol{\theta} = (w, \nu, \beta_1)$ are $w \sim \text{Beta}(1,1)$, $\nu \sim \text{Gamma}(0.01, 0.01)$ and $\beta_1 \sim N(0, 100)$ (vague priors) that are stated in the code via the arguments `prw = c(1,1)`, `prnu = c(0.1,0.1)`, `prbetamu = c(0)`, and `prbetasigma = diag(100,1,1)`, respectively.

```
> library(NGSSEML)
> data(sys1_data)
> Ytm = sys1_data[,1]+0.00001
> Xtm = sys1_data[,2]
> model = "SRWeibull"
## LabelParTheta = c("w", "nu", "Beta")
> StaPar = c(0.98,0.75,0.02)
## points
> pointss = 15
## Bayesian Fit:
> fitbayes = ngssm.bayes(Ytm ~ Xtm, data = data.frame(Ytm,Xtm), model = model,
+ pz = NULL, StaPar = StaPar, prw = c(1, 1), prnu = c(0.1, 0.1), prbetamu = c(0),
+ prbetasigma = diag(100, 1, 1), pointss = pointss, nsamplex = 1000, postplot = TRUE,
+ contourplot = TRUE, verbose = TRUE)
```

Table 7 presents the point and interval estimates for the Weibull SR model fitted to the "SYS1" data under the classical and Bayesian approaches. The estimate of parameter w is high (larger than 0.9), which means that the information loss over time is small. Besides, we have strong evidence from the data in favor of positive values for the regression coefficient β_1 and values lower than 1 for the shape parameter ν (i.e., decreasing failure rates).

$\boldsymbol{\theta}$	**MLE**	**95% Conf. Int.**	**Posterior Mean**	**95% Cred. Int.**
w	0.999	[0.996; 1.000]	0.969	[0.895; 0.994]
ν	0.753	[0.648; 0.857]	0.754	[0.655; 0.856]
β_1	0.023	[0.018; 0.029]	0.023	[0.015; 0.031]

Table 7: Point and interval (level 95%) estimation for the Weibull SR model fitted to the "SYS1" data.

The commands `postplot = TRUE` and `contourplot = TRUE` in the code of the Bayesian approach provide Figure 7 that shows the marginal posterior distributions and contour plot of the joint posterior distribution of the parameters w, ν, and β_1, respectively. Parameter w controls the information loss over time, while ν is the shape parameter of the Weibull SR model. The estimate of ν is lower than one, which means strong evidence in favor of decreasing failure rates. Analyzing the marginal posterior of β_1, we can observe that the number of previous failures seems to be relevant to the model.

The arguments `posts = posts`, `Proc = "Smooth"`, `Type = "Marg"`, `transf = 1/4`, and `model = "SRWeibull"` establish a sample of the marginal posterior of the static parameters, the smoothed distribution for the states, integrating the static parameters out, a transformation of the fourth root in the series, and the Weibull SR model are included in the code below to build Figure 8. Figure 8 provides transformed smoothed estimates for the mean of the data under the Weibull SR model. A transformation to reduce the scale of the data was necessary in order to obtain a better visualization of the graph. The fourth root was a suitable transformation in this case. As can be seen in Figure 8, the estimates follow well the behavior of the data.

```
> library(NGSSEML)
## Smoothing
> posts = fitbayes$samplepost
> set.seed(1000)
> PlotF(Ytm ~ Xtm, data = data.frame(Ytm, Xtm), plotYt = TRUE, transf = 1/4,
+ model = "SRWeibull", Proc = "Smooth", Type = "Marg", posts = posts, typeline = 'l',
+ col = c("black", "blue", "lightgrey"), xlab = "Number of Failures",
+ ylab = "Transformed Times", xlim = c(0, 139), ylim = c(-2, 10), lty = c(1, 2, 1),
+ lwd = c(1, 2, 1), cex = 0.68)
```

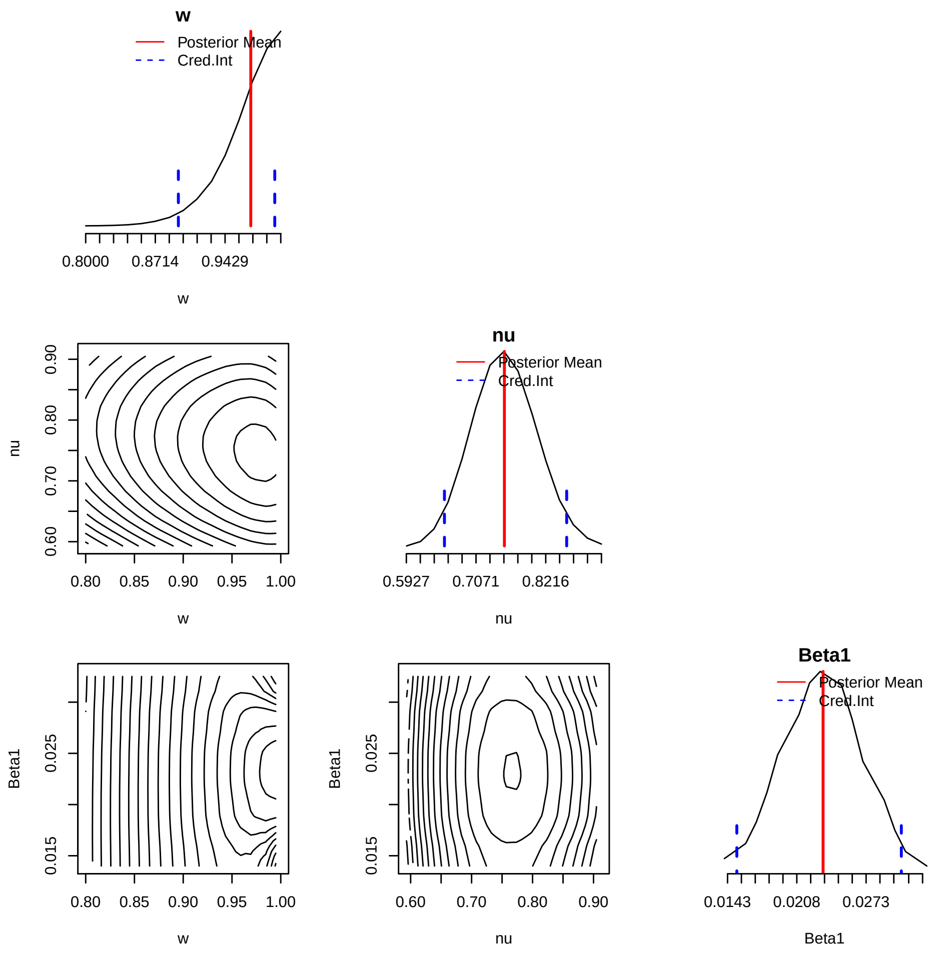

Figure 7: The full and dashed lines indicate, respectively, the fourth root of the series and its smoothed mean, obtained by the fit of the Weibull SR under the Bayesian perspective. The grey area indicates the 95% credibility intervals, obtained by the quadratures.

Conclusion

The package demonstrates how non-Gaussian state-space models (with exact marginal likelihood) can suitably be applied and employed in non-Gaussian time series and reliability analysis using R. The main contribution of the R-package NGSSEML is to provide an easy and fast code for classical and Bayesian estimations in non-Gaussian state space models with the exact marginal likelihood in the R programming language. Four important examples were presented for modeling time series and reliability data. The main advantages of the employed methodology are its analytical and

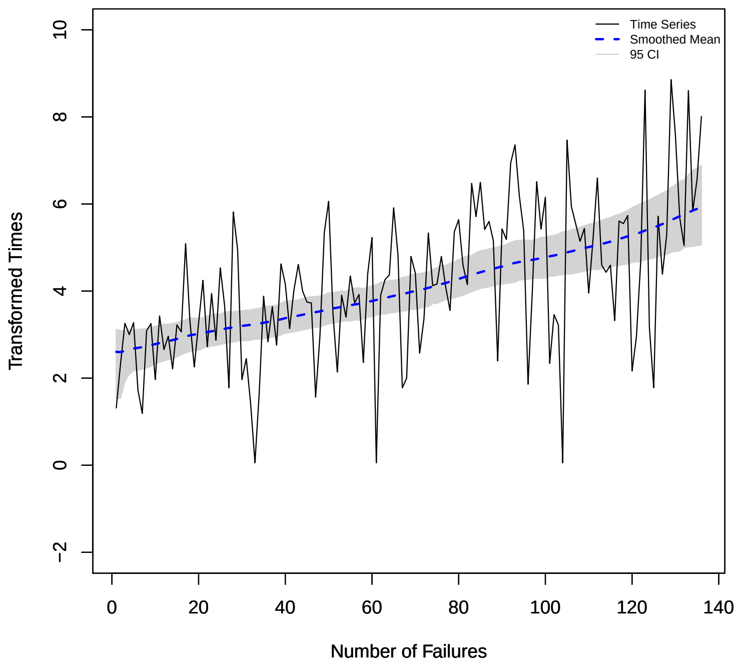

Figure 8: The full and dashed lines indicate, respectively, the fourth root of the series of the times between successive failures and the fourth root of the smoothed mean of the data, obtained by the fit of the SRWeibull under the Bayesian perspective using the quadratures. The grey area indicates the 95% credibility intervals.

computational simplicity, fast routines, and the ability to accommodate different types of distributions, even those which do not belong to the exponential family, combined with exact inference. Both classical and Bayesian approaches may be performed since the exact marginal likelihood for the static parameters is available. The package also provides R codes or functions for the prediction, filtering, and smoothing procedures of the latent states. Also, it is a platform where frequently used models, like the Normal and Poisson, can be included, as well as other volatility and reliability models, like the GED and Generalized Gamma.

Future work aims to increase the number of distributions (special cases) that can be specified for the observations in the package (mainly for time series data), to include dynamic linear models with the mean and variance varying over time that can conditionally be written in the NGSSEML form (Gamerman et al., 2013; Rego and dos Santos, 2020), and to introduce models for modeling multivariate time series data (Aktekin et al., 2018, 2020).

Acknowledgements

T. R. Santos was supported by CNPq-Brazil and FAPEMIG Foundation/Grant 25909. G. C. Franco was supported by CNPq-Brazil and FAPEMIG Foundation. D. Gamerman was supported by CNPq-Brazil and FAPERJ-Brazil. We thank the editor and reviewers for their constructive critique and positive review of our paper.

Bibliography

T. Aktekin, R. Soyer, and F. Xu. Assessment of mortgage default risk via bayesian state space models. *The Annals of Applied Statistics*, 7(3):1450–1473, Apr. 2013. URL `https://www.jstor.org/stable/23566480`. [p208]

T. Aktekin, N. Polson, and R. Soyer. Sequential bayesian analysis of multivariate count data. *Bayesian Analysis*, 13(2):385–409, June 2018. URL `https://doi.org/10.1214/17-BA1054`. [p208, 225]

T. Aktekin, N. Polson, and R. Soyer. A family of multivariate non-gaussian time series models. *Journal of Time Series Analysis*, 41(5):385–409, May 2020. URL `https://doi.org/10.1111/jtsa.12529`. [p225]

C. Andrieu and A. Doucet. Particle filtering for partially observed gaussian state space models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 64:827–836, Oct. 2002. URL https://doi.org/10.1111/1467-9868.00363. [p208]

P. J. Brockwell and R. A. Davis. *Introduction to Time Series and Forecasting*. Springer text in Statistics, New York, 1996. URL https://doi.org/10.1007/978-1-4757-2526-1. [p210]

C. Carvalho, M. Johannes, H. Lopes, and N. Polson. Particle learning and smoothing. *Statistical Science*, 25:88–106, Feb. 2010. URL https://doi.org/10.1214/10-STS325. [p208]

Y. Chang and C. Liu. A generalized jm model with applications to imperfect debugging in software reliability. *Applied Mathematical Modelling*, 33:3578–3588, Sept. 2009. URL https://doi.org/10.1016/j.apm.2008.11.018. [p222]

B. Christoffersen, A. Miller, A. Williams, B. developers, and R-core. *dynamichazard: Dynamic Hazard Models using State Space Models*. R-CRAN, Vienna, Austria, 2021. URL https://cran.r-project.org/web/packages/dynamichazard/index.html. [p208]

R. Davis and R. Wu. A negative binomial model for time series of counts. *Biometrika*, 96(3):735–749, Sept. 2009. URL https://doi.org/10.1093/biomet/asp029. [p215]

F. de Pinho, G. Franco, and R. Silva. Modeling volatility using state space models with heavy tailed distributions. *Mathematics and Computers in Simulation*, 139:108–127, Jan. 2016. URL https://doi.org/10.1016/j.matcom.2015.08.005. [p208, 209]

C. Dethlefsen and S. Lundbye-Christensen. Formulating state space models in r with focus on longitudinal regression models. *Journal of Statistical Software*, 16:1–15, Apr. 2006. URL https://doi.org/10.18637/jss.v016.i01. [p208]

J. Doornik. Ox: An object-oriented matrix language. *The Economic Journal*, 107(440):256–259, Jan. 1997. URL https://doi.org/10.1093/ej/107.440.256. [p208]

J. Durbin and S. J. Koopman. *Time Series Analysis by State Space Methods*. Oxford University Press, Oxford, Dec. 2012. URL https://doi.org/10.1093/acprof:oso/9780199641178.001.0001. [p208]

D. Gamerman, T. R. Santos, and G. C. Franco. A non-gaussian family of state-space models with exact marginal likelihood. *Journal of Time Series Analysis*, 34:625–645, Oct. 2013. URL https://doi.org/10.1111/jtsa.12039. [p208, 209, 210, 211, 213, 225]

J. Helske. Kfas: Exponential family state space models in r. *arXiv*, arXiv:1612.01907, 2016. [p208]

J. Helske and M. Vihola. *bssm: Bayesian Inference of Non-Linear and Non-Gaussian State Space Models*. R-CRAN, Vienna, Austria, 2021. URL https://cran.r-project.org/web/packages/bssm/index.html. [p208]

E. Holmes, E. Ward, M. Scheuerell, and K. Wills. *MARSS: Multivariate autoregressive state-space modeling*. R-CRAN, Vienna, Austria, 2013. URL https://cran.r-project.org/web/packages/MARSS/index.html. [p208]

J. Kim and R. Proschan. Piecewise exponential estimator of survivor function. *IEEE Transactions on Reliability*, 40:134–139, June 1991. URL https://doi.org/10.1109/24.87112. [p220]

A. King, D. Nguyen, and E. L. Ionides. Statistical inference for partially observed markov processes via the r package pomp. *Journal of Statistical Software*, 69(12):1–43, Mar. 2016. URL https://doi.org/10.18637/jss.v069.i12. [p208]

S. Koopman, N. Shephard, and J. Doornik. Statistical algorithms for models in state space using ssfpack 2.2. *The Econometrics Journal*, 2(1):107–160, June 1999. URL https://doi.org/10.1111/1368-423X.00023. [p208]

G. Petris. An r package for dynamic linear models. *Journal of Statistical Software*, 36(12):1–16, Oct. 2010. URL https://doi.org/10.18637/jss.v036.i12. [p208, 209, 210]

A. T. Rego and T. R. dos Santos. Non-gaussian stochastic volatility model with jumps via gibbs sampler. *Statistics and Its Interface*, 13:209–219, July 2020. URL https://dx.doi.org/10.4310/SII.2020.v13.n2.a6. [p225]

B. D. Ripley. Time series in r 1.5.0. *The Newsletter of the R Project*, 2(2):2–7, 2002. [p208]

T. Santos, D. Gamerman, and G. Franco. Reliability analysis via non-gaussian state-space models. *IEEE Transactions on Reliability*, 66:309–318, Mar. 2017. URL https://doi.org/10.1109/TR.2017.2670142. [p208, 209, 210, 213]

D. Shanno. Conditioning of quasi-newton methods for function minimization. *Mathematics of Computation*, 24:647–656, July 1970. URL https://doi.org/10.1090/S0025-5718-1970-0274029-X. [p210]

R. Smith and J. Miller. A non-gaussian state space model and application to prediction of records. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(1):79–88, Sept. 1986. URL https://doi.org/10.1111/j.2517-6161.1986.tb01392.x. [p209]

C. Szymanski. *Ldlmodeler: Generalized Dynamic Linear Modeler*. R-CRAN, Vienna, Austria, 2014. URL http://CRAN.R-project.org/package=dlmodeler. [p208]

R. C. Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. URL https://www.R-project.org/. [p208]

M. West and J. Harrison. *Bayesian Forecasting and Dynamic Models*. Springer, New York, May 1997. URL https://doi.org/10.1007/b98971. [p209]

M. West, P. Harrison, and H. Migon. Dynamic generalized linear models and bayesian forecasting (with discussion). *Journal of the American Statistical Association*, 73–97:80, Mar. 1985. URL https://doi.org/10.1080/01621459.1985.10477131. [p208]

S. Zeger. A regression model for time series of counts. *Biometrika*, 75:621–629, Dec. 1988. URL https://doi.org/10.2307/2336303. [p213]

D. Zes. *SSsimple: State Space Models*. R-CRAN, Vienna, Austria, 2019. URL https://cran.r-project.org/web/packages/SSsimple/index.html. [p208]

Thiago R. Santos
Universidade Federal de Minas Gerais
Brazil
ORCID: 0000-0003-2244-2718
thiagords@est.ufmg.br

Dani Gamerman
Universidade Federal do Rio de Janeiro
Brazil
dani@im.ufrj.br

Glaura C. Franco
Universidade Federal de Minas Gerais
Brazil
glaura@est.ufmg.br