Inference for Network Count Time Series with the R Package PNAR

by Mirko Armillotta, Michail Tsagris, and Konstantinos Fokianos

Abstract We introduce a new R package useful for inference about network count time series. Such data are frequently encountered in statistics and they are usually treated as multivariate time series. Their statistical analysis is based on linear or log-linear models. Nonlinear models, which have been applied successfully in several research areas, have been neglected from such applications mainly because of their computational complexity. We provide R users the flexibility to fit and study nonlinear network count time series models which include either a drift in the intercept or a regime switching mechanism. We develop several computational tools including estimation of various count Network Autoregressive models and fast computational algorithms for testing linearity in standard cases and when non-identifiable parameters hamper the analysis. Finally, we introduce a copula Poisson algorithm for simulating multivariate network count time series. We illustrate the methodology by modeling weekly number of influenza cases in Germany.

1 Introduction

Many data examples are frequently observed as multivariate counting processes recorded over a time-span and with known relations among observations (e.g epidemiological data with geographical distances between different areas). An important objective then is to study the effect of a known network to the observed data. This motivates a great amount of interest to network time series models; see Zhu et al. (2017) who developed continuous Network Autoregressive models (abbreviated as NAR). For these models, the observed variable Y, for the node i at time t, is denoted by $Y_{i,t}$, and it is assumed to depend on the past value of the variable for the node itself, say $Y_{i,t-1}$, and on the historical averages of its neighboring variables. The unknown parameters of the model are estimated by Least Squares estimation (LS). This work was advanced by Armillotta and Fokianos (2023a) who developed linear and log-linear Poisson Network Autoregression model (PNAR) for multivariate count distributed data. The joint dependence among different variables is specified by a copula construction (Fokianos et al., 2020, Sec. 2). In addition, Armillotta and Fokianos (2023a) have further established parametric estimation under the framework of quasi maximum likelihood inference (see Wedderburn (1974) Gourieroux et al. (1984)) and associated asymptotic theory when the network dimension increases. In the context of epidemiology, related applied work has been developed by Held et al. (2005) for the linear model only, and it was extended by Paul et al. (2008); Paul and Held (2011); Held and Paul (2012); Meyer and Held (2014) and Bracher and Held (2020).

The previous contributions impose linearity (or log-linearity) of the model, which can be a restrictive assumption for real world applications. For example, existence of different underlying states (e.g. exponentially expanding pandemic/ dying out pandemic) implies that different regime switching data generating processes should be applied and this fits the framework we consider. Recently, Armillotta and Fokianos (2023b) specified a general nonlinear Network Autoregressive model for both continuous and discrete-valued processes, establishing also related theory. In addition, the authors study testing procedures for examining linearity of NAR model against specific nonlinear alternatives by means of a quasi score test statistic. This methodology was developed with and without the presence of identifiable parameters under the null hypothesis.

Even though there exists sufficient statistical methodology for PNAR models, there has been a lack of up-to-date software for implementing their analyses. The aim of this work is to fill this gap by introducing the new package **PNAR** (Tsagris et al., 2023) and to demonstrate its usefulness for count network data analysis. Related R packages do not provide tools for estimating nonlinear models and applying associated testing procedures. The package **GNAR** (Leeming et al., 2023; Knight et al., 2020) studies Generalized NAR models (GNAR); this is a linear NAR model which takes into account the effect of several connection layers between the network nodes. This package deals with continuous-valued time series and does not contain tools for testing linearity. **PNAR** complements **GNAR** as it provides additional methodology for testing and inference about nonlinear discrete-valued network models.

Package **surveillance** (Höhle et al., 2022; Meyer et al., 2017) fits only linear models for spatial-temporal disease counts with Poisson or Negative Binomial distribution and with an autoregressive network effect. The package does accommodate various structural break-point tests but it does not contains functions for testing linearity and for log-linear model fitting. Moreover, standard errors of estimated parameters are computed by considering the quasi-likelihood as the true likelihood of the

model (Paul et al., 2008, Sec. 2.3). However, if the count time series are cross-sectional dependent, as it is usually the case, the likelihood function is misspecified and the obtained standard errors are not consistently estimated.

The **PNAR** package provides several advancements to the state of the art software: i) efficient estimation of (log-)linear models with proper robust standard errors accounting for possible model misspecification; ii) appropriate functions for testing linearity when a parameter is either identifiable of non-identifiable, under then null hypothesis, by providing appropriate p-value bounds and bootstrap approximations; iii) new algorithms for generating (log-)linear and nonlinear network count time series models.

The paper is organized as follows. The next section introduces linear and log-linear network time series autoregressive models for count data. Details about inference for unknown model parameters are provided. An application to estimation of weekly number of influenza A & B cases from two Southern German states is given and some further model aspects are discussed. Then we focus on non-linear models and associated testing theory. Results concerning score tests for testing linearity in NAR models are discussed and applied to influenza data. We also address the issue of computational speed. A short section discussing simulation of network count time series shows the usefulness of this methodology. The paper concludes with a short discussion.

2 Poisson network models

Consider a known network with N nodes, indexed by $i = 1, \ldots N$. The neighborhood structure of such a network is completely described by its adjacency matrix, say $A = (a_{ij}) \in \mathbb{R}^{N \times N}$ where $a_{ij} = 1$, if there is a directed edge from i to j, say $i \to j$, and 0 otherwise. Undirected graphs are allowed ($A = A'$), which means that the edge between two nodes, i and j, has no specific direction (say $i \sim j$). This is common in geographical and epidemic networks (e.g. district i shares a border with district j, patient i has a contact with patient j). Self-relationships are excluded i.e. $a_{ii} = 0$ for any $i = 1, \ldots, N$.

Let Y be a count variable measured on each node of the network ($i = 1, \ldots, N$), over a window of time ($t = 1, \ldots, T$). The data is a N-dimensional vector of time series $Y_t = (Y_{1,t}, \ldots, Y_{i,t}, \ldots Y_{N,t})'$, which is observed over the domain $t = 1, 2 \ldots, T$; in this way, a univariate time series is observed for each node, say $Y_{i,t}$, with corresponding conditional expectation $\lambda_{i,t}$. Denote by $\lambda_t = E(Y_t | \mathcal{F}_{t-1})$ with $\lambda_t = (\lambda_{1,t}, \ldots, \lambda_{i,t}, \ldots, \lambda_{N,t})'$ the conditional expectation vector of the counts with respect to their past history $\mathcal{F}_{t-1}$. The following linear autoregressive network model takes into account the known relations between nodes

$$Y_{i,t} | \mathcal{F}_{t-1} \sim Poisson(\lambda_{i,t}), \quad \lambda_{i,t} = \beta_0 + \beta_1 n_i^{-1} \sum_{j=1}^{N} a_{ij} Y_{j,t-1} + \beta_2 Y_{i,t-1}, \qquad (1)$$

where $n_i = \sum_{j \neq i} a_{ij}$ is the total number of connections starting from the node i, such that $i \to j$; called out-degree. We call (1) linear Poisson Network Autoregression of order 1, abbreviated by PNAR(1); (Armillotta and Fokianos, 2023a). From the left hand side equation of (1), we observe that the process $Y_{i,t}$ is assumed to be marginally Poisson but the joint process depends upon a copula function described in simulations at the end of the paper. Note that $\beta_0, \beta_1, \beta_2 > 0$ since the conditional mean of the Poisson is positive. Model (1) postulates that, for every single node i, the marginal conditional mean of the process is regressed on:

- the average count of the other nodes $j \neq i$ which have a connection with i; the parameter β_1 is called network effect, as it measures the average impact of node i's connections;

- the past count of the variable itself for i; the coefficient β_2 is called autoregressive effect because it provides an estimator for the impact of past count $Y_{i,t-1}$.

Model (1) implies that only nodes directly followed by the focal node i (i.e. $i \to j$), possibly, have an impact on its mean process of counts. It is a reasonable assumption in many applications; for example, in a social network the activity of node k, which satisfies $a_{ik} = 0$, does not affect node i. Hence, (1) measures the effect of a network to the observed multivariate count time series. Moreover, the model accommodates different types of network connectivity i.e. $a_{i,j}$ does not necessarily take the values 1-0 (connected-not connected). For example, $a_{i,j} = 1/d_{i,j}$ where $d_{i,j}$ is some measure of distance between node i and node j and $a_{i,i} = 0$. In this way the network effect becomes a spatial network component; see the last paragraph of Knight et al. (2020, p.3) for a discussion about a similar set of weights.

More generally, the counts $Y_{i,t}$ can be assumed to depend on the last p lagged values and q

covariates. Then consider the PNAR(p,q) model

$$\lambda_{i,t} = \beta_0 + \sum_{h=1}^{p} \beta_{1h} \left(n_i^{-1} \sum_{j=1}^{N} a_{ij} Y_{j,t-h} \right) + \sum_{h=1}^{p} \beta_{2h} Y_{i,t-h} + \sum_{l=1}^{q} \delta_l Z_{i,l} , \qquad (2)$$

where $\beta_0, \beta_{1h}, \beta_{2h} \geq 0$, for all $h = 1 \ldots, p$, $\delta_l \geq 0$, $l = 1, 2, \ldots, q$ and $Z_{i,l}$ are non-negative covariates measured for each node $i = 1, \ldots, N$. If $p = 1$ and $q = 0$ set $\beta_{11} = \beta_1$, $\beta_{21} = \beta_2$ to obtain (1). Model (2) is stationary if $\sum_{h=1}^{p} (\beta_{1h} + \beta_{2h}) < 1$ (Armillotta and Fokianos, 2023a).

The linear form of (2) offers a great advantage interpreting the parameters but it accommodates positive covariates. A real valued covariate enters (2) through suitable transformations that ensure positivity (e.g. include $\exp(Z)$ instead of directly Z). This restriction is bypassed by the log-linear model (Armillotta and Fokianos, 2023a):

$$\nu_{i,t} = \beta_0 + \sum_{h=1}^{p} \beta_{1h} \left(n_i^{-1} \sum_{j=1}^{N} a_{ij} \log(1 + Y_{j,t-h}) \right) + \sum_{h=1}^{p} \beta_{2h} \log(1 + Y_{i,t-h}) + \sum_{l=1}^{q} \delta_l Z_{i,l} , \qquad (3)$$

where $\nu_{i,t} = \log(\lambda_{i,t})$ and the observation are still marginally Poisson, $Y_{i,t}|\mathcal{F}_{t-1} \sim Poisson(\exp(\nu_{i,t}))$, for every $i = 1, \ldots, N$. Then the model parameters are real-valued since $\nu_{i,t} \in \mathbb{R}$ and the covariates can take any real values. The stationarity condition turns out to be $\sum_{h=1}^{p} (|\beta_{1h}| + |\beta_{2h}|) < 1$. Moreover, the interpretation of coefficients is similar to the case of linear model (1) but on the log-scale.

2.1 Inference

Model (2), or (3), depends on the m-dimensional vector of unknown parameters $\theta = (\beta_0, \beta_{11}, \ldots, \beta_{1p}, \beta_{21}, \ldots, \beta_{2p}, \delta_1, \ldots, \delta_q)'$, with $m = 1 + 2p + q$. We use of quasi-maximum likelihood methodology for estimation of θ; see Wedderburn (1974) and Gourieroux et al. (1984). The Quasi Maximum Likelihood Estimator (QMLE) is the vector of parameters $\hat{\theta}$ maximizing the function

$$l_T(\theta) = \sum_{t=1}^{T} \sum_{i=1}^{N} \left(Y_{i,t} \log \lambda_{i,t}(\theta) - \lambda_{i,t}(\theta) \right) , \qquad (4)$$

which is the so called pooled Poisson log-likelihood (up to a constant). Note that (4) is not necessarily the *true* log-likelihood of the process but it serves as an approximation. In particular, (4) is the log-likelihood function that would have been obtained if all time series were contemporaneously independent. However, the QMLE is not computed under the assumption of independence because (4) is simply a *working* log-likelihood function. The choice of maximizing (4) is justified for several reasons: i) full likelihood based on the joint process is complex (see the last section); ii) the optimization of (4) guarantees consistency and asymptotic normality of QMLE for the *true* parameter vector θ_0; iii) the QMLE is asymptotically equivalent to the MLE if the true probability mass function belongs to the linear exponential family (Gourieroux et al., 1984); iv) simplified computations entailing increased speed for estimation. Robustness of the QMLE in finite samples has been verified by Armillotta and Fokianos (2023a) through extensive simulation studies.

When considering the linear model (2), the score function is

$$S_T(\theta) = \sum_{t=1}^{T} \sum_{i=1}^{N} \left(\frac{Y_{i,t}}{\lambda_{i,t}(\theta)} - 1 \right) \frac{\partial \lambda_{i,t}(\theta)}{\partial \theta} = \sum_{t=1}^{T} s_t(\theta) . \qquad (5)$$

Define $\partial \lambda_t(\theta) / \partial \theta'$ the $N \times m$ matrix of derivatives, $D_t(\theta)$ the $N \times N$ diagonal matrix with elements equal to $\lambda_{i,t}(\theta)$, for $i = 1, \ldots, N$ and $\xi_t(\theta) = Y_t - \lambda_t(\theta)$ is the error sequence. Then, the empirical Hessian and conditional information matrices are given, respectively, by

$$H_T(\theta) = \sum_{t=1}^{T} \sum_{i=1}^{N} \frac{Y_{i,t}}{\lambda_{i,t}^2(\theta)} \frac{\partial \lambda_{i,t}(\theta)}{\partial \theta} \frac{\partial \lambda_{i,t}(\theta)}{\partial \theta'} , \quad B_T(\theta) = \sum_{t=1}^{T} \frac{\partial \lambda_t'(\theta)}{\partial \theta} D_t^{-1}(\theta) \Sigma_t(\theta) D_t^{-1}(\theta) \frac{\partial \lambda_t(\theta)}{\partial \theta'} , \quad (6)$$

where $\Sigma_t(\theta) = E(\xi_t(\theta) \xi_t'(\theta) \mid \mathcal{F}_{t-1})$ is the conditional covariance matrix evaluated at θ. Under suitable assumptions, Armillotta and Fokianos (2023a) proved that $\sqrt{NT}(\hat{\theta} - \theta_0) \xrightarrow{d} N(0, H^{-1}BH^{-1})$, when $N \rightarrow \infty$ and $T \rightarrow \infty$, where H and B are the theoretical limiting Hessian and information matrices, respectively, evaluated at the true value $\theta = \theta_0$. Then, a suitable estimator for the standard errors of θ is the square-rooted main diagonal of the empirical "sandwich" covariance matrix, i.e. $SE(\hat{\theta}) = \{ \text{diag} \left[H_T(\hat{\theta})^{-1} B_T(\hat{\theta}) H_T(\hat{\theta})^{-1} \right] \}^{1/2}$. Closely related works to ours have employed (4) for inference ; Paul et al. (2008) and Paul and Held (2011), among others. However, in such works (4) is

viewed as the true log-likelihood of the model and standard errors are computed by using the naive approach $SE_H(\hat{\theta}) = \{\text{diag}\left[H_T(\hat{\theta})^{-1}\right]\}^{1/2}$ which underestimates the real source of variation of the parameters when cross-section dependence among counts is present; see (6) which depends on the conditional covariance matrix of the process Y_t. The package **PNAR** returns robust standard errors as independence among counts is not assumed for their calculation. Similar theory holds for the log-linear model (3); details can be found in the aforementioned works.

2.2 Influenza data

To illustrate the use of **PNAR** we apply the methodology to the dataset `fluBYBW` from the **surveillance** package (Meyer et al., 2017). This dataset includes information about the weekly number of influenza A & B cases in the 140 districts of the two Southern German states Bavaria and Baden-Wuerttemberg, for the years 2001 to 2008 (416 time points). The response variable $Y = (Y_1, \ldots, Y_t, \ldots Y_T)'$ is then a 416×140 matrix of collective disease counts. Figure 1 illustrates the data in these two regions during 2007. We model these data by a linear PNAR model as we discuss next.
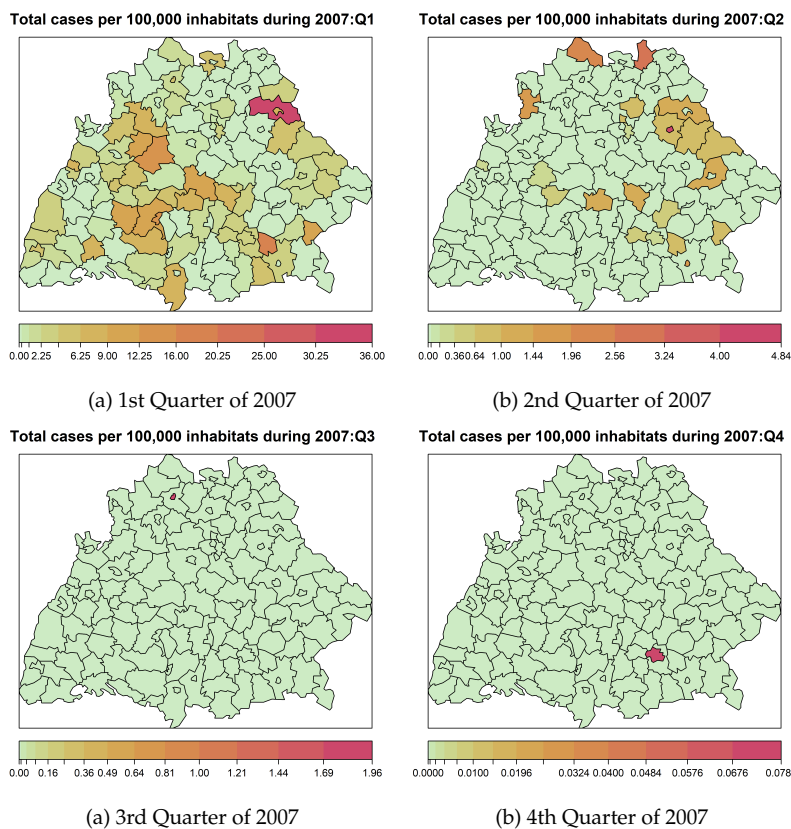

(a) 1st Quarter of 2007

(b) 2nd Quarter of 2007

(a) 3rd Quarter of 2007

(b) 4th Quarter of 2007

Figure 1: Quarterly flu cases in two Southern German states Bavaria and Baden-Wuerttemberg for 2007.

```
library(PNAR)
library(surveillance)
data(fluBYBW)
flu <- fluBYBW@observed
A_flu <- fluBYBW@neighbourhood
pop <- as.matrix(t(fluBYBW@populationFrac)[,1])
```

After loading **PNAR** we load **surveillance** for obtaining the 140×416 matrix of collective disease counts `flu`. The network adjacency matrix `A_flu` of dimension 140×140 has been obtained by linking two districts if they share (at least) a border. A covariate vector consisting of fraction of population in each district is introduced (pop). Model estimation for (2) when $p = 1$ and $p = 2$ are obtained below by using the function `lin_estimnarpq()` as follows:

```
est1.z <- lin_estimnarpq(y = flu, W = A_flu, p = 1, Z = pop)
```

```
est2.z <- lin_estimnarpq(y = flu, W = A_flu, p = 2, Z = pop)
```

Any type of non-negative matrix with zero main diagonal can be used as a valid adjacency matrix `W`. For instance consider weighted networks or inverse distance matrices, etc; see the last section for some alternatives when generating data. Optimization of (4) is implemented under the non-negativity constraint of coefficients satisfying the stationary condition. This is a nonlinear constrained optimization problem solved by means of a Sequential Quadratic programming (SQP) gradient-based algorithm (Kraft, 1994) of package **nloptr** (Ypma and Johnson, 2022). By default, the optimization is constrained in the stationary region; this can be removed by setting the option `uncons = TRUE` although this is not suggested because large sample properties of the estimators have been developed within the stationary region. The function `lin_estimnarpq()` has three additional features:

- `maxeval`: the maximum number of iterations for the optimization (the default is 100);
- `xtol_rel`: relative tolerance for the optimization termination condition (the default is 1e-8);
- `init`: starting value of the optimization (the default is NULL).

When `init = NULL` starting values are computed internally by ordinary LS using a lower barrier value of 0.01 because the regression coefficients cannot assume negative values. However, users can provide their own initial values through the argument `init`.

The function `lin_estimnarpq` returns as output a list consisting of the estimated coefficients, their associated standard errors, a z-test statistics with p-values, the score function evaluated at the optimum, the maximized log-likelihood, and the usual Akaike and Bayesian Information Criteria (AIC, BIC) accompanied by the Quasi IC (QIC) (Pan, 2001) which takes into account the fact that the log-likelihood (4) is a quasi log-likelihood; see Table 1 for the results, which are obtained in less than a second (see also Table 2). The score computed at the optimum values is of order 1e-5, on average, indicating successful convergence of the algorithm.

Table 1: Estimation of linear PNAR model (2) for $p = 1, 2$ and $Z = $ pop. Standard errors of coefficients are given in parentheses.

p	β_0	$\beta_{1,1}$	$\beta_{1,2}$	$\beta_{2,1}$	$\beta_{2,2}$	δ	AIC	BIC	QIC
1	0.0118 (0.0022)	0.2862 (0.0204)	-	0.6302 (0.0345)	-	2.0027 (0.4475)	-6041.20	-6025.08	-5886.16
2	0.0081 (0.0018)	0.2303 (0.0218)	0.0136 (0.006)	0.5459 (0.0379)	0.1445 (0.0183)	1.7609 (0.3998)	-7447.48	-7423.30	-7240.56

All the estimated coefficients are positive and significantly different from 0. The autoregressive effect $\beta_{2,h}$ shows higher magnitude with respect to the network effects $\beta_{1,h}$ since past counts of the same district are, in general, more informative than the neighboring cases. Both network and autoregressive parameters when $p = 1$ have a larger magnitude when compared to the corresponding coefficients at $p = 2$. This can be explained since influenza has an incubation period of only 1-4 days (with an average of 2 days) and a patient is still contagious for no more than 5-7 days after becoming sick[1] so the case counts at first lag are more informative than the ones at second lag which are still important. The population covariate is significant with positive effect. Standard errors are computed by using the sandwich estimator.

To select the model order p, for the PNAR model (2), **PNAR** includes a function for estimating model parameters for a range of lag values. By default $p \in \{1, 2, \ldots, 10\}$. This function returns the scatter plot of any IC (default is QIC) versus the lag order, for example

```
lin_ic_plot(y = flu, W = A_flu, p = 1:10, Z = pop, ic = "AIC")
```

Figure 2 shows the output of `lin_ic_plot()` for the case of AIC. Plots for the cases of BIC and QIC are similar and not shown. All information criteria point to the model with $p = 9$. However, from the corresponding estimation results reported in the Appendix, almost all β coefficients, which correspond to lags $p \geq 3$, are close to zero and non significant. Therefore, we decide to retain $p = 2$, for parsimony.

We compare estimation of standard errors after fitting the linear PNAR model to influenza data using **PNAR** and **surveillance** packages (covariate Z is excluded). The latter follows the standard error estimation according to the approach described in Paul et al. (2008) and Paul and Held (2011). The estimation of model (1) with **PNAR** gives

```
est1 <- lin_estimnarpq(y = flu, W = A_flu, p = 1)
summary(est1)
```
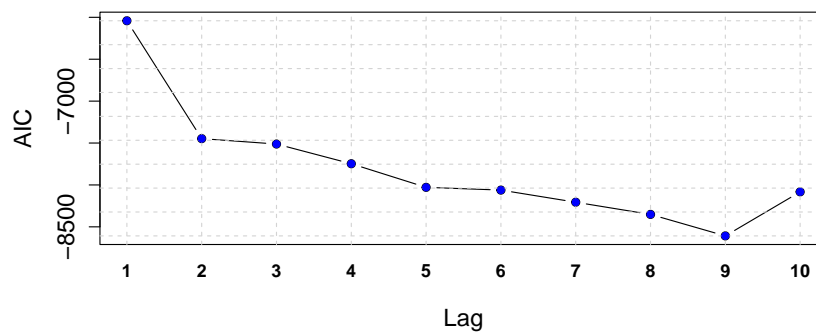
---

[1]https://www.cdc.gov/flu/about/disease/spread.htm

Figure 2: Scatter plot of AIC for PNAR(p) model versus p.

```
Coefficients:
         Estimate  Std. Error   z value      Pr(>|z|)
beta0  0.02460691 0.002722673  9.037777 1.598906e-19 ***
beta11 0.28952683 0.020393106 14.197289 9.522500e-46 ***
beta21 0.63082409 0.034462519 18.304642 7.598940e-75 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Using **surveillance**, the same model is fitted by the following lines of code:

```
stsObj <- fluBYBW
ni <- rowSums(neighbourhood(stsObj))
control <- list( ar = list(f = ~ 1),
ne = list(f = ~ 1, weights = neighbourhood(stsObj) == 1,
offset = matrix(1/ni, nrow(stsObj), ncol(stsObj), byrow=TRUE)),
end = list(f = ~ 1),
family = "Poisson", keep.terms = TRUE)
fit1 <- hhh4(stsObj, control)
coefSE <- coef(fit1, idx2Exp = TRUE, se = TRUE)
coefSE

            Estimate   Std. Error
exp(ar.1)  0.63082409 0.0066281675
exp(ne.1)  0.28952683 0.0052551614
exp(end.1) 0.02460691 0.0007619529
```

Comparing the above outputs, note that estimated coefficients are identical (subject to rounding errors) but standard errors, obtained by **surveillance** package, underestimate their value as it was explained at the end of the inference section. In addition, **PNAR** is more user-friendly for fitting (2) and it is almost twice as fast. Indeed, the average estimation time over 10 calls of the same function is 0.42 seconds in comparison to 0.22 by the **PNAR** package (see Table 2).

The usefulness of the **PNAR** package is not limited to epidemiological data. For example, the dataset crime, which is already built in the package, contains monthly number of burglaries within the census blocks on the south side of Chicago during 2010-2015 and includes a network matrix, called crime_W, connecting two blocks sharing a border. The documentation of the functions lin_estimnarpq() and log_lin_estimnarpq() provides an example of PNAR models applied to crime data.

3 Extending Linearity

In this section, we give some motivating examples of nonlinear models for time series of networks as introduced by Armillotta and Fokianos (2023b). In addition, we provide testing procedures for

testing the linearity assumption. For ease of presentation, denote by $X_{i,t} = n_i^{-1} \sum_{j=1}^N a_{ij} Y_{j,t}$ the average neighbor network mean. When a drift in the intercept term of (2) is introduced, the following nonlinear Intercept Drift model, ID-PNAR(p, q), is obtained

$$\lambda_{i,t} = \frac{\beta_0}{(1 + X_{i,t-d})^\gamma} + \sum_{h=1}^p \left(\beta_{1h} X_{i,t-h} + \beta_{2h} Y_{i,t-h} \right) + \sum_{l=1}^q \delta_l Z_{i,l} \,, \tag{7}$$

where $\gamma \geq 0$. Model (7) shares similar to a linear model when the parameter γ takes small values and for $\gamma = 0$ reduces to (2). Instead, when γ takes values away from zero, model (7) introduces a perturbation, deviating from the linear model, which depends on the network mean at lag $(t - d)$, where d is an additional delay parameter such that $d = 1, 2, \ldots, p$. Model (7) is directly applicable when the baseline effect, β_0, varies over time as a function of the network.

Another interesting class of nonlinear models are regime switching models, i.e. models allowing for the dynamics of the count process to depend on different regimes (e.g. exponentially expanding pandemic/ dying out pandemic). We give two such examples of models whose specification is based either on a smooth distortion or an abrupt transition. With the same notation as before, the Smooth Transition PNAR model, ST-PNAR(p, q) assumes a smooth transition between two regimes and it is defined by

$$\lambda_{i,t} = \beta_0 + \sum_{h=1}^p \left(\beta_{1h} X_{i,t-h} + \beta_{2h} Y_{i,t-h} + \alpha_h e^{-\gamma X_{i,t-d}^2} X_{i,t-h} \right) + \sum_{l=1}^q \delta_l Z_{i,l} \,, \tag{8}$$

where $\gamma \geq 0$ and $\alpha_h \geq 0$, for $h = 1, \ldots, p$. This models introduces a smooth regime switching behavior of the network effect making it possible to vary smoothly from $\beta_{1,h}$ to $\beta_{1,h} + \alpha_h$, as γ varies from large to small values. The additional delay parameter d determines the time of nonlinear transition that can be chosen. When $\alpha_h = 0$, for $h = 1, \ldots, p$ in (8), the linear PNAR model (2) is recovered. In some applications the transition between regimes may be abrupt (e.g. financial market crashes). For this reason, we consider the Threshold PNAR model, T-PNAR(p, q), which is defined by

$$\lambda_{i,t} = \beta_0 + \sum_{h=1}^p \left(\beta_{1h} X_{i,t-h} + \beta_{2h} Y_{i,t-h} + \left(\alpha_0 + \alpha_{1h} X_{i,t-h} + \alpha_{2h} Y_{i,t-h} \right) I \left(X_{i,t-d} \leq \gamma \right) \right) + \sum_{l=1}^q \delta_l Z_{i,l} \,, \tag{9}$$

where $I(\cdot)$ is the indicator function and $\gamma \geq 0$ is now threshold parameter. Moreover, $\alpha_0, \alpha_{1,h}, \alpha_{2,h} \geq 0$, for $h = 1, \ldots, p$. When $\alpha_0 = \alpha_{11} = \ldots = \alpha_{2p} = 0$, model (9) reduces to (2).

For all nonlinear models, estimation of the unknown parameters is based on QMLE, following the discussion in the inference section. Therefore, analogous conclusions about estimation of regression parameters, their standard errors and model selection apply to the case of nonlinear models (7)–(9). Further details can be found in Armillotta and Fokianos (2023b).

3.1 Standard implementation of testing linearity

Testing linearity against several specific alternatives offers guidance about the type of nonlinear model to be fitted. Moreover, in certain cases where the linear model is nested within a nonlinear model, some nonlinear parameters may be inconsistently estimated (see the case ST-PNAR and T-PNAR models below) so testing linearity prevents incorrect estimation.

Consider model (7) and the hypothesis testing problem $H_0 : \gamma = 0$ vs. $H_1 : \gamma > 0$ which is a hypothesis between the linear PNAR (2) null assumption versus ID-PNAR alternative model (7). Consider the vector of all the parameters of ID-PNAR model (7), $\theta = (\beta_0, \beta_{11}, \ldots, \beta_{2p}, \delta_1, \ldots, \delta_q, \gamma)'$. Define the partition of the parameters $\theta = (\theta^{(1)\prime}, \theta^{(2)\prime})'$, where $\theta^{(1)} = (\beta_0, \beta_{11}, \ldots, \beta_{2p}, \delta_1, \ldots, \delta_q)'$ is the sub-vector of parameters associated with the linear part of model. Let $\theta^{(2)}$ be the sub-vector of θ which corresponds to nonlinear parameters; for model (7), $\theta^{(2)} = \gamma$. Denote further by $S_T(\theta) = (S_T^{(1)\prime}(\theta), S_T^{(2)\prime}(\theta))'$ the corresponding partition of the quasi score function (5). We develop a quasi score test statistic based on the quasi log-likelihood (4). This is a convenient choice, because the score test requires estimation of model under the null hypothesis, i.e. under the linear model. Then the restricted estimator is denoted by $\tilde{\theta} = (\tilde{\beta}_0, \tilde{\beta}_{11}, \ldots, \tilde{\beta}_{2p}, \tilde{\delta}_1, \ldots, \tilde{\delta}_q)'$ and it is usually simpler to compute. The quasi score test statistic is given by

$$LM_T = S_T^{(2)\prime} (\tilde{\theta}) \, \Sigma_T^{-1} (\tilde{\theta}) \, S_T^{(2)} (\tilde{\theta}) \,, \tag{10}$$

with $\Sigma_T(\tilde{\theta}) = \tilde{J} H_T^{-1}(\tilde{\theta}) \tilde{J}' \left(\tilde{J} H_T^{-1}(\tilde{\theta}) B_T(\tilde{\theta}) H_T^{-1}(\tilde{\theta}) \tilde{J}' \right)^{-1} \tilde{J} H_T^{-1}(\tilde{\theta}) \tilde{J}'$, where $\tilde{J} = (O_{m_2 \times m_1}, I_{m_2})$, I_s is a $s \times s$ identity matrix and $O_{a \times b}$ is a $a \times b$ matrix of zeros. $\Sigma_T(\tilde{\theta})$ is a the estimator for the unknown covariance matrix $\Sigma = \text{Var}[S_T^{(2)}(\tilde{\theta})]$. It can be proved that the quasi score test (10) converges, asymp-

totically, to a $\chi^2_{m_2}$ distribution, where m_2 is the number of nonlinear parameter tested (Armillotta and Fokianos, 2023b). Then, in case of model (7), $m_2 = 1$ and we can compute the p-value of the test statistic (10) by $p = P(\chi^2_1 \geq M)$, where M is the observed value of the test statistic LM_T.

3.2 Non-standard implementation of testing linearity

Suppose now we wish to test linearity against (8). Then, considering the hypotheses $H_0 : \alpha_1 = \cdots = \alpha_p = 0$ versus $H_1 : \alpha_h \neq 0$ for some $h = 1, \ldots, p$, we note that this testing problem is non-standard because it is not possible to estimate the value of γ under H_0. Note that the parameter γ exists in the partition of the score function (5) related to nonlinear parameters, in particular in $\partial \lambda_{i,t}(\theta)/\partial \alpha_h$, and its associated covariance matrix. Hence, all relevant quantities for computing (10) are functions of γ; that is $S_T^{(2)}(\tilde{\theta}, \gamma)$, $\Sigma_T(\tilde{\theta}, \gamma)$ and $LM_T(\gamma)$. The model is then subject to non-identifiable parameter γ under the null. So if the true model is the linear PNAR model but a ST-PNAR is estimated instead the smoothing parameter γ will not be consistently estimated. Analogous conclusions hold for testing linearity against the T-PNAR model (9), where the threshold parameter γ is not identifiable under the null. When this issue arises, the standard theory does not apply and a chi-square type test is not suitable any more; see Davies (1987), Hansen (1996) and Armillotta and Fokianos (2023b), among others. It is clear that the value of the test changes by varying $\gamma \in \Gamma$, where Γ is some domain. A summary function of the test, computed under different values of γ, is then routinely employed in applications; a typical choice is $g_T = \sup_{\gamma \in \Gamma} LM_T(\gamma)$; see Armillotta and Fokianos (2023b) who established the convergence of g_T to $g = \sup_{\gamma \in \Gamma} LM(\gamma)$, where g is a function of a chi-square process, $LM(\gamma)$. The values of the latter asymptotic distribution cannot be tabulated, as they depends on unknown values of γ. Hence, we give methodology for computing p-values of such sup-type test statistic since they cannot be obtained otherwise.

Davies' bound. Since the space $\Gamma = [\gamma_L, \gamma_U]$ is usually assumed to be a closed interval, in practice, we take $\Gamma_F = (\gamma_L, \gamma_1, \ldots, \gamma_l, \gamma_U)$ i.e. a grid of values for the non-identifiable parameter γ, and the sup-test is obtained as the maximum of the tests $LM_T(\gamma)$ computed over Γ_F. Davies (1987) showed that that the p-value of the sup-test is approximately bounded by

$$P \left[\sup_{\gamma \in \Gamma_F} LM(\gamma) \geq M \right] \leq P(\chi^2_{m_2} \geq M) + V M^{\frac{1}{2}(m_2-1)} \frac{\exp(-\frac{M}{2}) 2^{-\frac{m_2}{2}}}{\Gamma(\frac{m_2}{2})}, \tag{11}$$

where M is the value of the sup-test statistic computed in the available sample, $\Gamma(\cdot)$ is the gamma function, and V is the approximated total variation

$$V = |LM_T^{\frac{1}{2}}(\gamma_1) - LM_T^{\frac{1}{2}}(\gamma_L) + |LM_T^{\frac{1}{2}}(\gamma_2) - LM_T^{\frac{1}{2}}(\gamma_1)| + \cdots + |LM_T^{\frac{1}{2}}(\gamma_U) - LM_T^{\frac{1}{2}}(\gamma_l)|.$$

Equation (11) shows how to approximate the p-values of the sup-type test in a straightforward way. Indeed, by adding to the tail probability of a chi-square distribution a correction term, which depends on the total variation of the process, we obtain the desired bound. This method is attractive for its simplicity and speed even when the dimension N of the network is large. However, the method approximate p-values with their bound (11) leading to a conservative test. In addition, (11) cannot be applied to the T-PNAR model (9), because the total variation requires differentiability of the asymptotic distribution $LM(\gamma)$ under the null hypothesis (Davies, 1987, p. 36), a condition that is not met for the case of T-PNAR models.

Bootstrapping the test statistic. Based on the previous arguments, we suggest an alternative p-value approximation of the test statistic employing stochastic permutations (Hansen, 1996; Armillotta and Fokianos, 2023b)-see Algorithm 1.

An approximation of the p-values is obtained from step 10 of Algorithm 1, where g_T is the value of the sup-test statistic computed on the available sample. When the number of bootstrap replications J is large enough, p_T^J provides a good approximation to the unknown p-values of the test. Then, the null hypothesis H_0 is rejected if p_T^J is smaller than a given significance level.

3.3 Revisiting the influenza data

We now apply the testing methodology described in the previous sections to influenza data. Consider testing linearity of the PNAR(2) model against the nonlinear ID-PNAR(2) (7) with $d = 1$. Analogous

---

Algorithm 1 Score bootstrap

1: Obtain the constrained QMLE of the linear model (2), say $\tilde{\theta}$
2: **for** $j = 1, \ldots, J$ **do**
3:     **for** $t = 1, \ldots, T$ **do**
4:         Generate $\nu_{t,j} \sim N(0,1)$
5:     **end for**
6:     Compute $S_T^{\nu_j}(\tilde{\theta}, \gamma) = \sum_{t=1}^{T} s_t(\tilde{\theta}, \gamma) \nu_{t,j}$
7:     Compute the test $LM_T^{\nu_j}(\gamma) = S_T^{\nu_j(2)\prime}(\tilde{\theta}, \gamma) \Sigma_T^{-1}(\tilde{\theta}, \gamma) S_T^{\nu_j(2)}(\tilde{\theta}, \gamma)$
8:     Optimize $LM_T^{\nu_j}(\gamma)$ for γ and take $g_T^j = \sup_{\gamma \in \Gamma} LM_T^{\nu_j}(\gamma)$
9: **end for**
10: Compute $p_T^J = J^{-1} \sum_{j=1}^{J} I(g_T^j \geq g_T)$

---

results have been obtained for $d = 2$ and therefore are omitted. The quasi score test (10) is computed by

```
id2.z <- score_test_nonlinpq_h0(b = est2.z$coefs[, 1], y = flu, W = A_flu,
p = 2, d = 1, Z = pop)
id2.z


        Linearity test against non-linear ID-PNAR(p) model

data:  coefficients of the PNAR(p, q)/time series data/order/lag/covariates/
chi-square-test statistic = 7.2318, df = 1, p-value = 0.007162
alternative hypothesis: True gamma parameter is greater than 0
```

where the first argument requires estimates under the null hypothesis $H_0 : \gamma = 0$ which have been obtained already. The rest of arguments follow previous syntax. For this testing problem the test is asymptotically chi-square distributed with 1 degree of freedom. The output lists the test statistic value and its corresponding p-value. There is strong indication to reject the linear model in favour of model (7).

Next consider testing linearity against the ST-PNAR(2) alternative (8). In this case, the test behaves in a non-standard way so we will be using the Davies' bound p-value (DV) for the sup-type test (11) and the bootstrap p-value approximation. First, the DV is computed by calling the following function

```
dv2.z <- score_test_stnarpq_DV(b = est2.z$coefs[, 1], y = flu, W = A_flu,
p = 2, d = 1, Z = pop)
```

Extreme values for the range of $\gamma \in \Gamma_F$ are computed internally in such a way that γ_U and γ_L are those values of γ where, on average, the smoothing function $\exp(-\gamma X_{i,t-d}^2)$ is equal to 0.1 and 0.9, respectively. In this way, during the optimization procedure, the extremes of the function domain are excluded. For more details see the **PNAR** manual (Tsagris et al., 2023). The user can specify different values for γ_L and γ_U using the arguments gama_L, gama_U and the number of grid values len (the default is 100). Results suggest again a deviation from linearity of the model, i.e.

```
dv2.z


        Test for linearity of PNAR(p) versus the non-linear ST-PNAR(p)

data:  coefficients of the PNAR(p, q)/time series data/order/lag/covariates
/lower gamma/upper gamma/length
chi-square-test statistic = 35.074, df = 2, p-value = 9.076e-08
alternative hypothesis: At least one coefficient of the non-linear component
is not zero
```

Next, we apply Algorithm 1 to compute the bootstrap p-values for the sup-type test statistic. In this case, the observed value of $\sup_{\gamma \in \Gamma} LM(\gamma)$ has to be computed. Initially, perform a global optimization of the $LM_T(\gamma)$ for the ST-PNAR(p) model, with respect to the nuisance scale parameter γ by using Brent's algorithm (Brent, 1973) in the interval [gama_L to gama_U], (see previous discussion for their computation). To ensure global optimality, the optimization is performed on runs at len-1 consecutive

equidistant sub-intervals and the global optimum is determined by the maximum over those sub-intervals. The default value for len is 10. Then using the function global_optimise_LM_stnarpq with the same arguments we obtain the optimal γ value and the corresponding value of the test statistic:

```
go1.z2 <- global_optimise_LM_stnarpq(b = est2.z$coefs[, 1], y = flu, W = A_flu,
p = 2, d = 1, Z = pop)
go1.z2$gama
[1] 8.387526
go1.z2$supLM
[1] 35.07402
```

This information is used as follows

```
boot1.z2 <- score_test_stnarpq_j(supLM = go1.z2$supLM, b = est2.z$coefs[, 1],
y = flu, W = A_flu, p = 2, d = 1, Z = pop,
J = 499, ncores = 7, seed = 1234)
```

which implements Algorithm 1 using $J = 499$ bootstrap replicates. The function uses a parallel processing option; the user can set the number of cores ncores (the default is no parallel). The seed for random number generation assures reproducibility of the results.

```
boot1.z2$pJ
[1] 0.002004008
boot1.z2$cpJ
[1] 0.004
```

The above output gives (among other information) p_T^J of step 10 of Algorithm 1 and an alternative corrected unbiased estimator for the p-value which is $cp_T^J = (J+1)^{-1} \left[\sum_{j=1}^{J} I(g_T^j \geq g_T) + 1 \right]$. Like in the case of DV p-value, linearity is rejected.

We work analogously for testing the PNAR(2) model versus the T-PNAR(2) model (9). Note that optimization of $LM_T(\gamma)$, in this case, is based on gama_L and gama_U which are obtained (by default) as the mean over $i = 1, \ldots, N$ of 20% and 80% quantiles of the empirical distribution of the network mean $X_{i,t}$ for $t = 1, \ldots, T$. In this way, during the optimization process, the indicator function $I(X_{i,t-d} \leq \gamma)$ avoids values close to 0 or 1. Alternatively, their value can be supplied by the user. The functions used are analogous to the functions used for the ST-PNAR(2).

```
tgo1.z2 <- global_optimise_LM_tnarpq(b = est2.z$coefs[, 1], y = flu, W = A_flu,
p = 2, d = 1, Z = pop)
tgo1.z2$gama
[1] 0.1257529
tgo1.z2$supLM
[1] 49.06505

tboot1.z2 <- score_test_tnarpq_j(supLM = tgo1.z2$supLM, b = est2.z$coefs[, 1],
y = flu, W = A_flu,  p = 2, d = 1, Z = pop,
J = 499, ncores = 7, seed = 1234)
tboot1.z2$pJ
[1] 0.3907816
tboot1.z2$cpJ
[1] 0.392
```

The test does not reject the null hypothesis of linearity, in the case of a threshold model. Overall, the analysis shows that the linear PNAR model may not be a suitable model to fit such epidemic data and nonlinear alternatives should be considered. In particular, evidence of a nonlinear drift in the intercept and of a regime switching mechanism is detected. In addition, it appears that a smooth regime switching mechanism might be more appropriate for the data.

3.4 Computational Speed

Package **PNAR** is quite efficient in terms of computational speed, especially for estimation problems. We have employed the **Rfast** and **Rfast2** packages (Papadakis et al., 2023a,b) wherever possible to ensure computational speed. We run each function 10 times and compute the average time required to be executed. We use a laptop computer equipped with Intel Core i7 processor (3.00GHz) and 16 GB of RAM. Results are given in Table 2. Estimation of linear PNAR model and standard testing (ID-PNAR model) is fast. Computations of p-values, in the case of non-identifiable parameters, requires several

evaluations of the test statistic on a grid of values for γ (Davies' bound) or global optimization of the test statistic (bootstrap). However both tasks are executed in a satisfactory amount of time. The bootstrap approximation Algorithm 1 runs slower but it still executed within satisfactory time limits. Computational speed, for both Davies' bound and bootstrap p-values, can be further increased by reducing the length of the γ grid for the former and the number of bootstrap replications for the latter. Increasing the number of cores will provide faster bootstrap approximated p-values.

Table 2: Average computation times (in seconds) for the functions called in the text.

PNAR Estimation				
p	1	2	4	9
No cov.	0.22	-	-	-
Cov.	0.39	0.57	0.89	1.26

Tests				
Models	χ_1^2	DV	Global opt.	Bootstrap
ID-PNAR	0.43	-	-	-
ST-PNAR	-	12.88	3.74	74.60
T-PNAR	-	-	1.30	75.20

4   Simulating network count time series

In this last section we present a further novel implementation of the **PNAR** package which can be used to simulate network count time series from linear and nonlinear models with multivariate copula Poisson distribution, as we explain next.

Equation (1) does not include information about the joint dependence structure of the PNAR(1) model. Following Fokianos et al. (2020) the joint multivariate distribution of the vector count time series Y_t is defined as $Y_t = N_t(\lambda_t)$ where, $\{N_t\}$ is a sequence of copula-Poisson processes, that is $N_t(\lambda_t)$ is a sequence of N-dimensional IID marginally Poisson count processes, with intensity 1, counting the number of events in the interval of time $[0, \lambda_{1,t}] \times \cdots \times [0, \lambda_{N,t}]$, and whose structure of dependence is modeled through a copula function $C(\rho)$ on their associated exponential waiting times random variables. The algorithm is described below for model (1).

Consider a network matrix A and a set of values $(\beta_0, \beta_1, \beta_2)'$ for model (1). Moreover, define a starting mean vector at time $t = 0$, say $\lambda_0 = (\lambda_{1,0}, \ldots, \lambda_{N,0})'$.

1. Let $U_l = (U_{1,l}, \ldots, U_{N,l})'$, for $l = 1, \ldots, K$ a sample from a N-dimensional copula $C(u_1, \ldots, u_N; \rho)$, where $U_{i,l}$ follows a Uniform(0,1) distribution, for $i = 1, \ldots, N$.

2. The transformation $E_{i,l} = -\log U_{i,l}/\lambda_{i,0}$ follows the exponential distribution with parameter $\lambda_{i,0}$, for $i = 1, \ldots, N$.

3. If $E_{i,1} > 1$, then $Y_{i,0} = 0$, otherwise $Y_{i,0} = \max\left\{k \in [1, K] : \sum_{l=1}^{k} X_{i,l} \leq 1\right\}$, by taking K large enough. Then, $Y_{i,0}|\lambda_0 \sim Poisson(\lambda_{i,0})$, for $i = 1, \ldots, N$. So, $Y_0 = (Y_{1,0}, \ldots, Y_{N,0})'$ is a set of (conditionally) marginal Poisson processes with mean λ_0.

4. By using the model (1), λ_1 is obtained.

5. Return back to step 1 to obtain Y_1, and so on.

In applications, choose K large , e.g. $K = 100$; its values generally depends on the magnitude of observed data. Moreover, the copula $C(\rho)$ depends on one or more unknown parameters, say ρ, which capture the contemporaneous correlation among the variables. The proposed algorithm ensures that all marginal distributions of $Y_{i,t}$ are univariate Poisson, conditionally to the past, as described in (1), while it introduces an arbitrary dependence among them in a flexible and general way by the copula construction through the parameter ρ. An analogous process is employed for generating log-linear and nonlinear count network models by suitable modifications.

Multivariate Poisson-type distributions have typically complicated form and their covariance matrix might not be appropriate (Fokianos et al., 2020); this inspired the adoption of this simulation methodology. Imposing a copula directly on Poisson marginals can lead to identifiability issues (Genest and Nešlehová, 2007). For further details see Fokianos et al. (2020), Armillotta and Fokianos (2023a) and the recent review in Fokianos (2022).

PNAR allows to generate multivariate count times from well-known network models like the Erdős-Rényi Model (Erdös and Rényi, 1959), with the function adja_gnp(), or the Stochastic Block Model (SBM) (Wang and Wong, 1987) with the function adja(). Such functions are based on the **igraph** package (Csardi and Nepusz, 2006); see Tsagris et al. (2023) for details.

```
set.seed(1234)

W_SBM <- adja(N = 10, K = 2, alpha = 0.7, directed = TRUE)

sim1 <- poisson.MODpq(b = c(0.2,0.2,0.4), W = W_SBM, p = 1, TT = 100, N = 10,
copula = "gaussian", corrtype = "equicorrelation", rho = 0.5)
sim1$y
```

The first function randomly generates an adjacency matrix from the directed SBM model with 10 nodes and 2 groups. The second function generates a 100×10 time series matrix object of network counts from the linear PNAR(1) model (1) where the joint dependence in the data generating process is modeled by a Gaussian copula with $\rho = 0.5$. The "equicorrelation" option generates a correlation matrix for the Gaussian copula where all the off-diagonal entries equal ρ. Another type of correlation matrix which can be used is the "toeplitz" option that returns a correlation matrix whose generic off-diagonal (i, j)-element is $\rho^{|i-j|}$. Moreover, other copula functions can also be chosen as the t or the Clayton copula. Some of the 10 simulated time series are plotted in Figure 3, for illustration. Analogous functions are provided for generating synthetic data from log-linear or nonlinear Poisson network model (Table 4 in the Appendix).
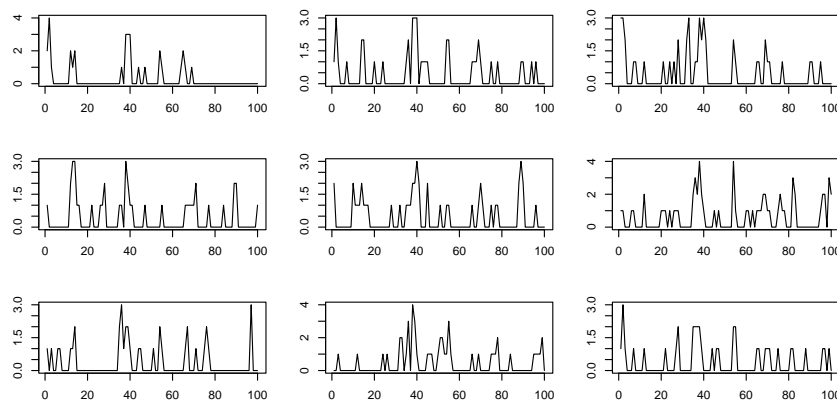

Figure 3: Simulated count time series from the linear PNAR(1) model.

5 Conclusion

There exists R software for fitting Network Autoregressive models (**GNAR**, **surveillance**). However, no published package includes functions for inference with nonlinear Network Autoregressive models. **PNAR** fills this gap by providing users tools for efficient estimation of (log-)linear models with proper robust standard errors, test statics and computational algorithms for testing model linearity and new simulation methodology for generating (log-)linear and nonlinear network count time series models. We showed that all these tasks are executed with a minimal computational effort.

There are a number of possible developments for **PNAR**. One possibility is to include several other nonlinear models and develop related linearity tests. In addition, developing a negative binomial quasi-likelihood estimation method offers more flexibility to model fitting. Alternative ways to compute p-values, for example employing different bootstrap approximation, may also be considered. Further simulation methods for generating network count time series are easily accommodated by suitable modification of the copula Poisson algorithm. All these extensions provide users with new set of tools for inference in the broad framework of multivariate discrete-valued time series models.

6 Appendix

Output from estimation of PNAR model (2) with lag $p = 9$ and population covariate:

```
est9.z <- lin_estimnarpq(y = flu, W = A_flu, p = 9, Z = pop)
summary(est9.z)

Coefficients:
          Estimate    Std. Error      z value      Pr(>|z|)
beta0  8.046948e-03 0.0018543147 4.339581e+00 1.427548e-05 ***
beta11 2.291637e-01 0.0222504112 1.029930e+01 7.097436e-25 ***
beta12 1.219436e-02 0.0058931749 2.069234e+00 3.852414e-02 *
beta13 2.880415e-08 0.0034601665 8.324498e-06 9.999934e-01
beta14 6.840207e-08 0.0047642557 1.435735e-05 9.999885e-01
beta15 4.994822e-08 0.0022529259 2.217038e-05 9.999823e-01
beta16 6.335964e-07 0.0023712114 2.672037e-04 9.997868e-01
beta17 1.096226e-06 0.0022836178 4.800390e-04 9.996170e-01
beta18 1.094862e-07 0.0018014328 6.077731e-05 9.999515e-01
beta19 1.230050e-07 0.0017010627 7.231070e-05 9.999423e-01
beta21 5.462136e-01 0.0389427608 1.402606e+01 1.079843e-44 ***
beta22 1.361396e-01 0.0179869627 7.568794e+00 3.767043e-14 ***
beta23 1.404481e-02 0.0053121667 2.643894e+00 8.195825e-03 **
beta24 1.764756e-08 0.0023065543 7.651051e-06 9.999939e-01
beta25 1.874096e-06 0.0016152031 1.160285e-03 9.990742e-01
beta26 2.036961e-05 0.0023094311 8.820187e-03 9.929626e-01
beta27 1.492721e-06 0.0019472654 7.665731e-04 9.993884e-01
beta28 6.414357e-08 0.0018167793 3.530620e-05 9.999718e-01
beta29 8.818047e-07 0.0009314276 9.467238e-04 9.992446e-01
delta1 1.412017e+00 0.3536615653 3.992566e+00 6.536216e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log-likelihood value:  4324.057
AIC: -8608.114 BIC: -8527.5 QIC: -8362.176
```

Table 3: A brief overview of the main functions in **PNAR**.

Function	Description
`lin_estimnarpq()`	Fitting the linear PNAR(p, q) model.
`log_lin_estimnarpq()`	Fitting the log-linear PNAR(p, q) model.
`score_test_nonlinpq_h0()`	Score test of linear PNAR versus ID-PNAR model.
`score_test_stnarpq_DV()`	Score test of linear PNAR versus ST-PNAR model by (11).
`global_optimise_LM_stnarpq()`	Maximize test statistic of ST-PNAR for nuisance parameters.
`score_test_stnarpq_j()`	Bootstrap score test of linear PNAR versus ST-PNAR model.
`global_optimise_LM_tnarpq()`	Maximize test statistic of T-PNAR for nuisance parameters.
`score_test_tnarpq_j()`	Bootstrap score test of linear PNAR versus T-PNAR model.

Table 4: A brief overview of functions simulating network count time series models in **PNAR**.

Function	Description
`poisson.MODpq()`	Generation from a linear PNAR(p) model with covariates.
`poisson.MODpq.log()`	Generation from a log-linear PNAR(p) model with covariates.
`poisson.MODpq.nonlin()`	Generation from a Intercept Drift PNAR(p) model with covariates.
`poisson.MODpq.stnar()`	Generation from a Smooth Transition PNAR(p) model with covariates.
`poisson.MODpq.tnar()`	Generation from a Threshold PNAR(p) model with covariates.

7 Acknowledgments

References

M. Armillotta and K. Fokianos. Count network autoregression. *Journal of Time Series Analysis*, jtsa.12728, 2023a. URL http://doi.org/10.1111/jtsa.12728. [p254, 255, 256, 264]

M. Armillotta and K. Fokianos. Nonlinear Network Autoregression. *The Annals of Statistics*, 51(6): 2526–2552, 2023b. URL https://doi.org/10.1214/23-AOS2345. [p254, 259, 260, 261]

J. Bracher and L. Held. Endemic-epidemic models with discrete-time serial interval distributions for infectious disease prediction. *International Journal of Forecasting*, 38:1221–1233, 2020. [p254]

R. Brent. *Algorithms for Minimization Without Derivatives*. Prentice-Hall, 1973. [p262]

G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006. URL https://igraph.org. [p265]

R. Davies. Hypothesis testing when a nuisance parameter is present only under the alternative. *Biometrika*, 74:33–43, 1987. [p261]

P. Erdös and A. Rényi. On random graphs I. *Publicationes Mathematicae Debrecen*, 6, 1959. [p265]

K. Fokianos. Multivariate count time series modelling. *To appear in Econometrics and Statistics*, 2022. [p264]

K. Fokianos, B. Støve, D. Tjøstheim, and P. Doukhan. Multivariate count autoregression. *Bernoulli*, 26: 471–499, 2020. [p254, 264]

C. Genest and J. Nešlehová. A primer on copulas for count data. *Astin Bulletin*, 37:475–515, 2007. [p264]

C. Gourieroux, A. Monfort, and A. Trognon. Pseudo maximum likelihood methods: Theory. *Econometrica*, 52:681–700, 1984. [p254, 256]

B. Hansen. Inference when a nuisance parameter is not identified under the null hypothesis. *Econometrica*, 64:413–430, 1996. [p261]

L. Held and M. Paul. Modeling seasonality in space-time infectious disease surveillance data. *Biometrical Journal*, 54:824–843, 2012. [p254]

L. Held, M. Höhle, and M. Hofmann. A statistical framework for the analysis of multivariate infectious disease surveillance counts. *Statistical Modelling*, 5:187–199, 2005. [p254]

M. Höhle, S. Meyer, and M. Paul. *surveillance: Temporal and Spatio-Temporal Modeling and Monitoring of Epidemic Phenomena*, 2022. URL https://CRAN.R-project.org/package=surveillance. R package version 1.21.1. [p254]

M. Knight, K. Leeming, G. Nason, and M. Nunes. Generalized network autoregressive processes and the GNAR package. *Journal of Statistical Software*, 96:1–36, 2020. URL https://www.jstatsoft.org/v096/i05. [p254, 255]

D. Kraft. Algorithm 733: TOMP–Fortran modules for optimal control calculations. *ACM Transactions on Mathematical Software (TOMS)*, 20:262–281, 1994. [p258]

K. Leeming, G. Nason, M. Nunes, and J. Wei. *GNAR: Methods for Fitting Network Time Series Models*, 2023. URL https://CRAN.R-project.org/package=GNAR. R package version 1.1.2. [p254]

S. Meyer and L. Held. Power-law models for infectious disease spread. *The Annals of Applied Statistics*, 8:1612–1639, 2014. [p254]

S. Meyer, L. Held, and M. Höhle. Spatio-temporal analysis of epidemic phenomena using the R package surveillance. *Journal of Statistical Software*, 77:1–55, 2017. [p254, 257]

W. Pan. Akaike's information criterion in generalized estimating equations. *Biometrics*, 57:120–125, 2001. [p258]

M. Papadakis, M. Tsagris, M. Dimitriadis, S. Fafalios, I. Tsamardinos, M. Fasiolo, G. Borboudakis, J. Burkardt, C. Zou, K. Lakiotaki, and C. Chatzipantsiou. *Rfast: A Collection of Efficient and Extremely Fast R Functions*, 2023a. URL https://CRAN.R-project.org/package=Rfast. R package version 2.0.8. [p263]

M. Papadakis, M. Tsagris, S. Fafalios, and M. Dimitriadis. *Rfast2: A Collection of Efficient and Extremely Fast R Functions II*, 2023b. URL https://CRAN.R-project.org/package=Rfast2. R package version 0.1.5.1. [p263]

M. Paul and L. Held. Predictive assessment of a non-linear random effects model for multivariate time series of infectious disease counts. *Statistics in Medicine*, 30:1118–1136, 2011. [p254, 256, 258]

M. Paul, L. Held, and A. M. Toschke. Multivariate modelling of infectious disease surveillance data. *Statistics in Medicine*, 27:6250–6267, 2008. [p254, 255, 256, 258]

M. Tsagris, M. Armillotta, and K. Fokianos. *PNAR: Poisson Network Autoregressive Models.*, 2023. URL https://CRAN.R-project.org/package=PNAR. R package version 1.6. [p254, 262, 265]

Y. J. Wang and G. Y. Wong. Stochastic blockmodels for directed graphs. *Journal of the American Statistical Association*, 82:8–19, 1987. [p265]

R. W. Wedderburn. Quasi-likelihood functions, generalized linear models, and the Gauss-Newton method. *Biometrika*, 61:439–447, 1974. [p254, 256]

J. Ypma and S. G. Johnson. *nloptr: R Interface to NLopt*, 2022. URL https://CRAN.R-project.org/package=nloptr. R package version 2.0.3. [p258]

X. Zhu, R. Pan, G. Li, Y. Liu, and H. Wang. Network vector autoregression. *The Annals of Statistics*, 45:1096–1123, 2017. [p254]

Mirko Armillotta
Vrije Universiteit Amsterdam
Department of Econometrics and Data Science
Netherlands
Tinbergen Institute
Netherlands
ORCiD: 0000-0002-0548-6957
m.armillotta@vu.nl

Michail Tsagris
University of Crete
Department of Economics
Greece
ORCiD: 0000-0002-2049-3063
mtsagris@uoc.gr

Konstantinos Fokianos
University of Cyprus
Department of Mathematics and Statistics
Cyprus
ORCiD: 0000-0002-0051-711X
fokianos@ucy.ac.cy