BANSHEE–A MATLAB toolbox for Non-Parametric Bayesian Networks

Paprotny, Dominik; Morales-Nápoles, Oswaldo; Worm, Daniël T.H.; Ragno, Elisa

DOI
10.1016/j.softx.2020.100588

Publication date
2020

Document Version
Final published version

Published in
SoftwareX

SoftwareX 12 (2020) 100588

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

BANSHEE–A MATLAB toolbox for Non-Parametric Bayesian Networks

Dominik Paprotny [a,*], Oswaldo Morales-Nápoles [b], Daniël T.H. Worm [c], Elisa Ragno [b]

[a] Helmholtz Centre Potsdam, GFZ German Research Centre for Geosciences, Section Hydrology, Telegrafenberg, 14473 Potsdam, Germany
[b] Delft University of Technology, Department of Hydraulic Engineering, Stevinweg 1, 2628CN Delft, The Netherlands
[c] TNO, Cyber Security & Robustness, Anna van Buerenplein 1, 2595DA The Hague, The Netherlands

ARTICLE INFO

Article history:
Received 3 June 2020
Received in revised form 14 September 2020
Accepted 14 September 2020

Keywords:
Copulas
Probabilistic models
Belief Nets

ABSTRACT

Bayesian Networks (BNs) are probabilistic, graphical models for representing complex dependency structures. They have many applications in science and engineering. Their particularly powerful variant – Non-Parametric BNs – are for the first time implemented as an open-access scriptable code, in the form of a MATLAB toolbox "BANSHEE".[1] The software allows for quantifying the BN, validating the underlying assumptions of the model, visualizing the network and its corresponding rank correlation matrix, and finally making inference with a BN based on existing or new evidence. We also include in the toolbox, and discuss in the paper, some applied BN models published in most recent scientific literature.

Code metadata

Current code version	*BANSHEE v1.0*
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX_2020_243
Code Ocean compute capsule	–
Legal Code License	*GNU General Public License*
Code versioning system used	*none*
Software code languages, tools, and services used	*MATLAB (including the Statistics and Machine Learning Toolbox)*
Compilation requirements, operating environments & dependencies	*MATLAB (including the Statistics and Machine Learning Toolbox)*
If available Link to developer documentation/manual	https://github.com/ElsevierSoftwareX/SOFTX_2020_243
Support email for questions	paprotny@gfz-potsdam.de

1. Motivation and significance

Bayesian Networks (BNs) are graphical, probabilistic models for representing high-dimensional and complex dependency structures [1–4]. A BN consists of a directed acyclic graph (DAG), in which nodes (representing random variables) are connected with arcs representing direct dependency between nodes. The direct predecessors of a node are called parents, and the direct successors are known as children. Each node with no parents has a marginal distribution specified, while each child node is associated with a conditional distribution. The strength of the dependency between nodes is informed by the conditional distributions in the BN [5,6].

BNs have been gaining popularity for several reasons. They are flexible and are able to present the dependence structure even for very large models. Different variants of BNs handle various data types, while the quantitative information needed to build a BN can be obtained both from data or through expert judgement [5,7–11]. Here, we present a MATLAB toolbox "BANSHEE" for a particular variant of BNs — non-parametric Bayesian Networks (NPBNs). This type of BNs was introduced by Kurowica and Cooke [12] and has a major advantage of using empirical

* Corresponding author.
E-mail addresses: paprotny@gfz-potsdam.de (D. Paprotny), o.moralesnapoles@tudelft.nl (O. Morales-Nápoles), daniel.worm@tno.nl (D.T.H. Worm), E.Ragno@tudelft.nl (E. Ragno).

[1] BANSHEE stands for 'Bayesian Networks in Scholarly Endeavours'. However, a banshee is also, in Irish folklore, a female spirit whose appearance is a warning about impending death. Bayesian Networks have been extensively used in risk analysis in fields ranging from aviation safety through natural hazards to building fire safety, hence they also warn against possible dangers, somewhat similarly to a banshee.

https://doi.org/10.1016/j.softx.2020.100588

(non-parametric) marginal distributions of continuous variables. No discretization assumptions or a particular parametric distribution is therefore required for defining the continuous variables (though parametric distributions may still be used), eliminating a significant source of inaccuracies in BN models. For modelling the dependency structure, copulas [13] are used in NPBNs.

The applications of NPBNs are numerous and diverse. One of the first applications was in engineering, with often very large models applied to improve earth dam safety [9,10], aviation safety [14] and infrastructure reliability [15]. Later, NPBNs were introduced in geosciences, in diverse subfields such as hydrology [16], geomorphology [17], seismology [18] and volcanology [19]. In social sciences, they were used in such remote applications as public health [20] and climate-change mitigation policies [21]. A comprehensive review of NPBN applications was provided by Hanea et al. [5] and an up-to-date list of many papers using the method is available online [22].

However, an important limiting factor in making new analyses with NPBNs is software availability. At present, there is only one dedicated software solution for NPBNs, in the form of a closed-source, proprietary package Uninet by LightTwist Software [22]. Our toolbox BANSHEE is the first implementation as a standalone, open-access code, which could allow researchers to provide transparent, reproducible results with NPBNs. BANSHEE itself originates from MATLAB scripts [23,24] that were also used by the authors to implement NPBNs in some recent publications [16,17,25]. While one of the main advantages of Uninet is its Graphical User interface (GUI) that makes it easier to construct and experiment with the model, BANSHEE allows quickly embedding an NPBN model into MATLAB scripts through a set of easy-to-use functions, including diagnostic tools for analysing the NPBN's underlying assumptions.

2. Software description

BANSHEE consists of a set of MATLAB functions. The software allows for quantifying the NPBN, analysing the underlying assumptions of the model, visualizing the network and its corresponding rank correlation matrix, and finally making inference with a NPBN based on existing or new evidence. Examples are provided as standalone scripts. Real-world example applications from literature are included to better explain the method and show the power of the toolbox.

2.1. Software architecture

The most important component of the toolbox is the code (`bn_rankcorr`) implementing the NPBN method described by Hanea et al. [5,26] and Kurowicka and Cooke [4]. As noted in the introduction, NPBNs make no assumptions on the marginal distributions (i.e. distributions of the nodes). The arcs are associated with one-parameter, conditional copulas [13]. Loosely, a bivariate copula, or simply a copula for the purposes of this paper, is a joint distribution with uniform margins in [0, 1]. Multivariate joint distributions can be written in terms of the univariate marginal distribution functions and a copula. For the bi-variate case one can write $H(x, y) = C(F_X(x), G_Y(y))$, where $H(x, y)$ is a joint distribution with marginal distributions F_X and G_Y. Therefore, C is a copula taking values from $I^2 = ([0, 1] \times [0, 1])$. For copulas with only one parameter there is a one-to-one relation between the parameter and Spearman's rank correlation coefficient (Eq. (3)). Copulas allow the investigation of probabilistic dependence separately from the effect of the one-dimensional margins and hence their importance in the NPBN framework.

The (conditional) copulas are assigned to arcs according to the (non-unique) ordering of the parent nodes. For a particular choice of copulas, dependency structure and a set of one-dimensional marginal distributions, the joint distribution of the NPBN is uniquely determined [26]. Any copula realizing all correlations in $[-1, 1]$ can be used, while the (conditional) independent copula realizing all (conditional) independence relationships encoded by the graph of the NPBN may be used. Here we implement the NPBNs with the Gaussian (normal) copula, which does not present tail dependence (or other asymmetries) between variables. The joint density of a BN with n variables is factorized as follows [5]:

$$f_{1,\ldots,n}(x_1, \ldots, x_n) = f_1(x_1) \prod_{i=2}^{n} f_{i|Pa(i)}\left(x_i | \boldsymbol{x}_{Pa(i)}\right) \tag{1}$$

where $Pa(i)$ is the set of parent nodes of X_i, with $i = 1, \ldots, n$. In case there are no parents, $f_{X_i|Pa(X_i)} = f_{X_i}$.

The BN's structure (directed acyclic graph — DAG) is typically expert knowledge-driven, therefore we do not include any automated way of deriving the DAG. The user defines the nodes and arcs, together with the ordering of the parent nodes as a cell array. An example DAG, as defined in BANSHEE, with three nodes and two arcs is as follows:

```
P{1} = []; % first node, no parents
P{2} = 3; % second node (third node is the parent node)
P{3} = 1; % third node (first node is the parent node)
```

where P{i} is the ith node of the DAG. Empirical distributions from the user's data are assigned to the nodes. The usual estimator of the cumulative probability distribution is applied here:

$$\hat{F}(x) = \frac{1}{n} \sum_{i=1}^{n} 1_{\{x_i \leq x\}} \tag{2}$$

where $(x_i, \ldots, x_n)$ are the samples of a random variable, $1_{\{x_i \leq x\}} = 1$ if $\{x_i \leq x\}$ and zero otherwise. Once the DAG is defined and the user's data are provided, the NPBN rank correlation matrix quantifying the dependency structure is estimated. As noted above, a Gaussian copula parametrized by Spearman's rank correlation is applied here to define the correlation between two connected nodes. Spearman's correlation is Pearson's product moment correlation coefficient computed with the ranks of the random variables. In such a specific case, the rank correlation of two random variables (nodes) X_i and X_j is as follows:

$$r\left(X_i, X_j\right) = 12 \int_0^1 \int_0^1 C_\theta(u, v)\, du dv - 3 \tag{3}$$

where u, v are the margins of one-parameter bivariate copula C_θ. The conditional Spearman's rank correlation of X_i and X_j given the random vector $\boldsymbol{Z} = \boldsymbol{z}$ is the Spearman's rank correlation calculated in the conditional distribution of X_i and X_j given the random vector $\boldsymbol{Z}_1 = \boldsymbol{z}_1, \ldots, \boldsymbol{Z_k} = \boldsymbol{z_k}$. For each variable X_i with m parents $Pa_1(X_i), \ldots, Pa_m(X_i)$ the arc $Pa_j(X_i) \rightarrow X_i$ is associated with the rank correlation:

$$\begin{cases} r\left(X_i, Pa_j(X_i)\right), & j = 1 \\ r\left(X_i, Pa_j(X_i) | Pa_1(X_i), \ldots, Pa_{j-1}(X_i)\right), & j = 2, \ldots, m \end{cases} \tag{4}$$

where the index j is in the non-unique sampling order. It is worth noticing that the m parent nodes of X_i in Eq. (4) can be permuted such that anyone can be the first parent, the second, and so on until the mth index. The resulting correlation matrices parametrizing the BN will in general differ according to the selected sampling order unless the BN is given by a complete (saturated) graph. Given a directed acyclic graph with n nodes described by invertible distribution functions and the conditional independence relationship between them modelled via the NPBN,

the specification in Eq. (4) guarantees that the joint distribution of the n variables (nodes) is uniquely determined. For details, see [5,26].

The BN rank correlation matrix is computed in BANSHEE using `bn_rankcorr` function, requiring a defined DAG (as a cell array) and an adequate number of variables (as a matrix). The function utilizes as default a matrix of data that forms a set of actual records, but includes an option to compute BN rank correlation without such data, which will be discussed in Section 3.

The assumptions underlying the BN quantification can be tested by measuring (1) the degree of agreement of the Gaussian copula with the data (Cramer–von Mises statistic M) and (2) the degree to which the chosen conditional independence statements implied by the BN agree with data (d-calibration score). In the first case, we use the sum of squared difference between the empirical and parametric copulas [27]. The Cramer–von Mises statistic M for a sample of length n is computed as follows:

$$M_n(\boldsymbol{u}) = n \sum_{|\boldsymbol{u}|} \left\{ C_{\hat{\theta}_n}(\boldsymbol{u}) - B(\boldsymbol{u}) \right\}^2, \boldsymbol{u} \in [0,1]^2 \quad (5)$$

where $B(\boldsymbol{u}) = \frac{1}{n}\sum_{i=1}^{n} 1(U_i \leq \boldsymbol{u})$ is the empirical copula and $C_{\hat{\theta}_n}(\boldsymbol{u})$ is a parametric copula with parameter $\hat{\theta}_n$ estimated from the sample. Notice that M is the sum of squared difference between the empirical copula and a particular parametric estimator. Hence a low value of M is desired over a high value. This observation may be used as a rule of thumb for assessing goodness of fit for copulas. Further details are given in [27].

The function `cvm_statistic` computes M for four copulas (Gaussian, Gumbel, Clayton, Frank)[2] and allows to visualize the results. The second test, called d-calibration score [11], consists in comparing the empirical correlation matrix (the data) with both the BN rank correlation matrix and the empirical normal rank correlation matrix (the model).Testing the distance between empirical and normal matrices informs whether the joint distribution of the variables defined by the user can be assumed to be Gaussian. This means that the Gaussian copula is a fair choice for modelling the bivariate dependence structure between variables. The distance between BN and normal matrices informs the user if the assumption of a joint normal distribution is valid for a particular non-saturated configuration of the NPBN. The distance between empirical and BN matrices provides information on whether the BN chosen is a fair model for the data. This is similar to the tests described by Hanea et al. [5]. The distance between matrices can be computed in different ways, and the user can choose between four methods [11,28] in the `gaussian_distance` function. The d-calibration score ranges between 0 (the two matrices are different) and 1 (the two matrices are similar).

Finally, inference can be made with the quantified BN using the function `inference`. The function computes the uncertainty distribution of nodes other than those that the user conditionalized in, i.e. provided specific values (posterior evidence). Algorithmically, the inference is done through sampling the NPBN; the process is described in more detail by Hanea et al. [26].

2.2. Software functionalities

The toolbox contains five MATLAB functions, two standalone MATLAB scripts with examples and three MATLAB functions implementing published NPBN models (Table 1). The functions are interconnected through a common input data structure. The user can build a NPBN following a few steps: (1) upload a dataset of interest (DATA); (2) define a DAG in PARENTCELL based on a prior knowledge of the dependence between the variables; (3) run the function `bn_rankcorr` to calculate the BN rank correlation matrix R; (4) run the function `inference`, given R and DATA as input, to obtain the joint distribution function of the variables and the conditional distribution of a variable given the remaining. For inference, the nodes to be conditionalized are specified by the user together with the values of those nodes. The user can also modify certain aspects of the calculation such as sampling size, interpolation methods and type of output. The BN dependency structure can be visualized with a separate `bn_visualize` function, which allows naming the variables e.g. for use in research publications. The function `bn_rankcorr` and the two diagnostic tools, `gaussian_distance` and `cvm_statistic`, all have an option to generate a plot.

All functions are collected in the example script, which apart from running the functions and generating all possible plots (Fig. 1) includes detailed descriptions of each step of the procedure. A second example, `example_udrm` implements a particular example of a model where the (conditional) correlations between variables are taken from an external source such as expert judgement (Section 3). The real-life example models are discussed in Section 4. All scripts are described in detail in the quick start guide included in the toolbox.

3. Illustrative examples

Example 1

The first script, `example`, is constructed to predict the level of personal safety (as indicated by variable "Safety") employing a default MATLAB dataset `cities`. It contains data on nine quality-of-life indicators in 329 cities in the United States. It should be noted that the data are transformed to ranks in the procedure, so there is no need for adjusting the input data e.g. through normalization or logarithmic transformation. The variables in the DAG (Fig. 1b) were identified by firstly creating an expert-knowledge derived DAG, as is common in Bayesian Network models. Then, `bn_rankcorr` function was applied to compute a BN rank correlation matrix for the defined DAG (Fig. 1b). The model was then iteratively modified to remove the least-correlated arcs between nodes, until only significant and theoretically explainable variable pairs remained. The final example model includes five nodes: four explanatory variables – "Climate", "Economics", "Recreation" and "Arts", and one variable of interest – "Safety".

The validity of the Gaussian copula assumption is then tested. The Cramer–von Mises statistic shows that the Gaussian copula achieves best fit for the majority of variable pairs according to `cvm_statistic` function (Fig. 1c). The d-calibration score (`gaussian_distance` function) gives a mixed picture: the d-calibration score of the empirical rank correlation matrix and the empirical normal matrix (vertical red line in Fig. 1d – left panel) falls outside the 90% confidence interval of the d-calibration score of the normal rank correlation matrix (red circles, Fig. 1d – left panel) estimated via bootstrapping. This means that the determinant of the empirical rank correlation matrix is different than the determinant of the normal empirical rank correlation matrix, and so the Gaussian copula might not be the best choice for modelling the bivariate dependences. However, the d-calibration score of the BN's rank correlation matrix and the empirical normal matrix is well within the 90% confidence interval of the d-calibration score of the empirical normal matrix (Fig. 1d – right panel). This second d-calibration score is more important, as it shows that the joint normal copula is valid for the particular (non-saturated) BN structure. It should be noted that the d-calibration test is

[2] Those one-parameter copulas are implemented in standard MATLAB functions. The user might expand the code with the two-parameter t copula that is also available in MATLAB, while other copulas require custom code (e.g. Plackett and Joe-Clayton copula functions from Andrew Patton's copula toolbox for MATLAB).

Table 1
Overview of function in the BANSHEE toolbox.

Code	Operation	Input	Optional inputs	Output
General functions				
inference	Conditionalizes a quantified NPBN, making inference based on evidence provided	NODES — definition of nodes to be conditionalized; VALUES — data for conditionalizing the nodes; R – NPBN rank correlation matrix (generated with 'bn_rankcorr'); DATA — quantification of the NPBN	OUTPUT — show the full uncertainty distribution in F, or only mean or median; SAMPLESIZE — number of samples drawn; INTERP — interpolation method used when conditionalizing the NPBN	F – uncertainty distribution (or mean/median depending on the option chosen) of the predictions
bn_rankcorr	Computes NPBN rank correlation matrix	PARENTCELL — definition of the BN structure; DATA — quantification of the NPBN; ISDATA — specifies type of input data	PLOT — show plot; NAMES — provide variable names for the plot	R
gaussian_distance	Computes d-calibration score for the rank correlation matrix	R; DATA	SAMPLESIZE; PLOT; TYPE — choose calculation method (Hellinger distance, Symmetric Kullback–Leibler divergence, Bhattacharyya distance, Abou Moustafa et al.'s "G" distance)	D_ERC, D_BNRC — d-calibration scores B_ERC, B_BNRC — 90% confidence intervals of the d-calibration of sampled random normal distributions
cvm_statistic	Computes copula goodness-of-fit for pairs of variables in the NPBN using Cramer–von Mises statistic.	DATA	PLOT; NAMES	M – results of the Cramer–von Mises statistic
bn_visualize	Visualizes the NPBN's structure	PARENTCELL; R	NAMES	(creates a plot with the NPBN structure)
Examples				
example	Runs all general functions with a default Matlab dataset	–	–	(creates plots and outputs from all functions)
example_udrm	Runs an example User-Defined Random Model with an actual research dataset from a Weigh-in-Motion experiment	–	–	(creates plots and outputs from selected functions)
example_hydro_simulation	The script applies the BN model for extreme river discharges with example real-life data	–	–	(creates plots and outputs from selected functions)
Real-world example models				
predict_floor_space	Estimates useful floor space area of residential buildings	VALUES	NODES; OUTPUT	FSA — building useful floor space area
predict_river_discharge	Estimates annual maximum of daily river discharges	VALUES	NODES; OUTPUT	QMAX — annual maximum of river discharge
predict_coast_erosion	Estimates storm-induced erosion of a cliff coast	VALUES	NODES; OUTPUT	SHORE, BEACH, FOOT, CLIFF, TOP — cliff erosion metrics

rather severe for large datasets [5]: the higher the number of variables the lower is the determinant, which makes the comparison between determinants numerically more difficult. Once the NPBN has been built, predictions of the level of safety in cities (or any other variable) given a combination of the remaining variables can be tested against observations, using the `inference` function.

Example 2

In a second example, `example_udrm`, a BN model without an actual dataset – User-Defined Random Model (UDRM) – is built. This example was created to allow implementation of BNs where the (conditional) correlations are obtained through structured expert judgement elicitation [8,11,29,30]. Indeed, BN models in engineering often require such a method of quantification due to lack of data [5,11]. The user should define the DAG, the marginal distribution at each node, and the (conditional) rank correlations on each arc. Based on these information, the BN rank correlation matrix is calculated via the `bn_rankcorr` function. The `example_udrm` script generates possible axle loads configurations from the defined BN model to estimate the probability of the vehicle weight accounting for (1) the type of vehicle (2-, 3-, 4-, 5- axle) and (2) the dependences existing between the axel loads of a single vehicle. Running the example will allow to visualize the correlation matrix and BN structure (Fig. S1). Moreover, the code contains a section in which the dependence between the axle loads is ignored (independent case). Fig S2 compares the probability of the vehicle weight estimated based on observations (blue dots), simulations from a dependent model (BN model — red dots), and simulations from an independent model (green dots). This plot highlights that considering the strong dependency between the load on different axels of vehicles is important in investigating maximum traffic load on bridges or other stretches

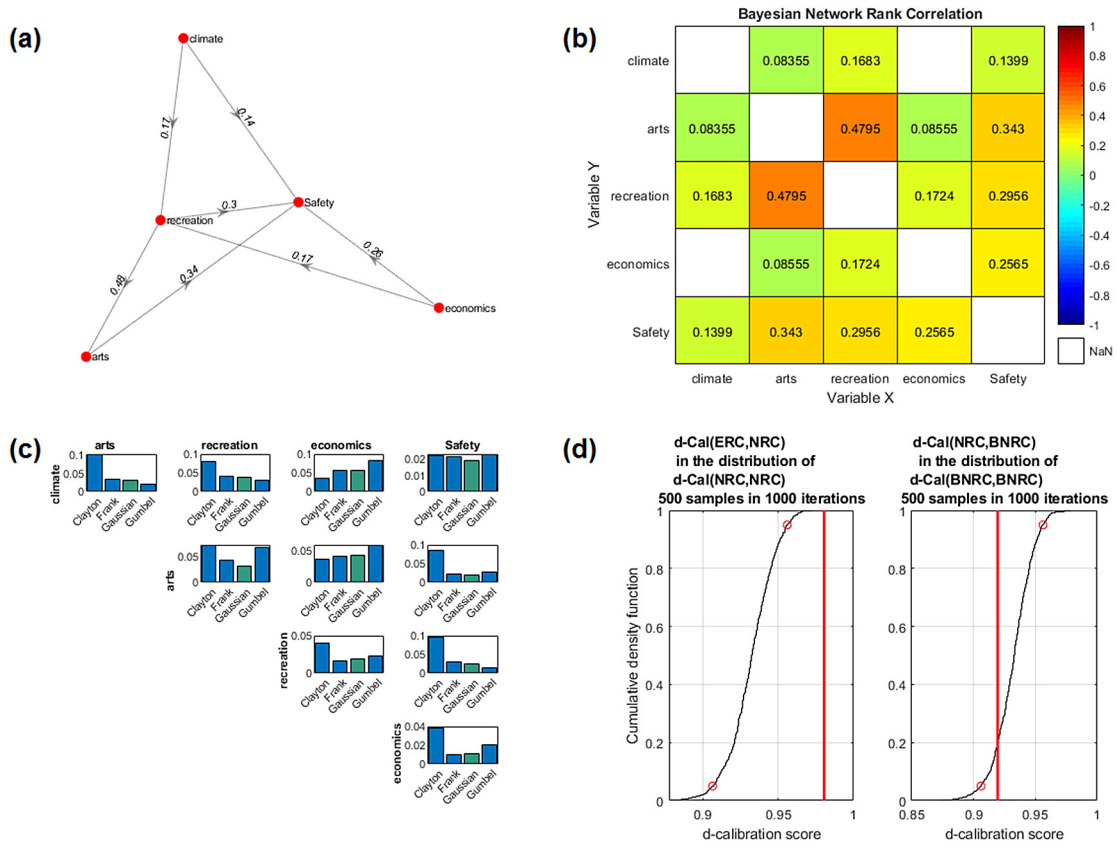

Fig. 1. Plots generated with the toolbox's functions and default Matlab dataset `cities`: (a) BN structure (`bn_visualize`); (b) NPBN rank correlation matrix (`bn_rankcorr`); (c) Cramer–von Mises statistic for all pairs of variables (`cvm_statistic`); (d) d-calibration score (`gaussian_distance`).

of roads, as opposed to considering them independent. It should be noted that this example uses real-life data on vehicle axle loads from Weigh-In-Motion traffic monitoring system in the Netherlands [15,31]. It is an simplified version of the full, 705-node model, which has been used to advise the Dutch Ministry of Infrastructure and Environment.

4. Impact

The toolbox includes three real-life example models from the authors' recent geoscientific applications (Fig. S3). Each model consists of a function script and a dataset containing the definition of the DAG, marginal distributions on the nodes, and a BN rank correlation matrix. The user only needs to specify which nodes are to be conditionalized and provide their data on the corresponding variables. The first model, `predict_river_discharge`, estimates the value of maximum annual discharge in European rivers [16]. This enables generating an annual time series of extreme discharge in locations where river gauge measurements are not available. Hydrological modelling of discharge on a European scale is complex and very time-consuming, but it has been shown that the NPBN model is an accurate and efficient substitute [16]. The model was applied for the whole of European river network using our MATLAB toolbox and served as input for pan-European flood hazard modelling [32]. The user can apply the model to any location and year providing e.g. the value of catchment size, maximum daily precipitation during the year or percentage of the catchment covered by lakes. Usage of the function is highlighted in the wrapper script `example_hydro_simulation`.

Another model, `predict_floor_space`, was conceived as a tool to substitute for missing information on the size of residential buildings [25]. Knowing how big is a house, especially its height and useful floor space area, is fundamental for estimating its exposure and vulnerability to natural hazards. As 3D models of cities are still scarce [33], this NPBN model was constructed based on OpenStreetMap building footprints combined with several pan-European raster datasets. The model was applied using this MATLAB toolbox to estimate exposure of residential buildings in several case studies of past floods [34]. As with the model for extreme discharges, only openly-available datasets are used as explanatory variables, hence the user can easily collect data to apply the model within Europe (they were not validated for other countries so far).

The final example is a model of storm-induced coastal erosion in Poland (`predict_coast_erosion`), which was created based on field observations of cliff retreat in Poland and Germany [17]. The model reproduces the complex dependency structure of the processes involved, as not only meteorological and hydrological factors impact the cliff and the beach below it, but erosion in one part of the profile triggers erosion in another part and so on. As with the other models, MATLAB was used extensively in data pre- and postprocessing, hence the implementation of NPBN code in the same language enabled a quick embedding of the BN model into the workflow.

5. Conclusions

BANSHEE is the first openly available tool for Non-Parametric Bayesian Networks. As the examples contained in the toolbox have highlighted, the actual and potential applications are numerous. We hope that our toolbox (1) will increase the popularity of NPBNs as a powerful statistical method, (2) will support researchers committed to sharing their code and data openly and

(3) will enhance implementation of NPBN models beyond science and into engineering and administrative practice.

BANSHEE is intended to be developed further, especially with improvements to analytical and visualization tools. New applicable BN models will be added as part of ongoing and future paper submissions, with novel BN flood damage models for the residential and commercial sectors expected to be added first [34,35]. Linking our toolbox with a MATLAB toolbox for structured expert judgement ANDURYL [36,37] is also envisioned.

CRediT authorship contribution statement

Dominik Paprotny: Conceptualization, Data curation, Software, Visualization, Writing - original draft, Writing - review & editing. **Oswaldo Morales-Nápoles:** Conceptualization, Methodology, Resources, Software, Visualization, Writing - original draft, Writing - review & editing. **Daniël T.H. Worm:** Methodology, Software, Writing - review & editing. **Elisa Ragno:** Software, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was partially supported by Climate-KIC through project "SAFERPLACES", Task ID TC2018B_4.7.3-SAFERPL_P430-1A KAVA2 4.7.3. Further funding was received from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 730381, as well as under the Marie Skłodowska-Curie Action grant agreement no. 707404. The authors thank Kai Schröter for useful comments on the documentation, and two anonymous referees for helpful comments on the entire work.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.softx.2020.100588.

References

[1] Cowell R, Dawid A, Lauritzen S, Spiegelhalter D. Probabilistic networks and expert systems statistics for engineering and information sciences. New York: Springer-Verlag; 1999.
[2] Fenton N, Neil M. Risk assessment and decision analysis with Bayesian networks. CRC Press; 2012.
[3] Koski T, Noble J. Bayesian networks: An introduction. Chichester, UK: Wiley; 2011.
[4] Kurowicka D, Cooke R. Uncertainty analysis with high dimensional dependence modelling. Chichester, UK: Wiley; 2006.
[5] Hanea A, Morales Nápoles O, Ababei D. Non-parametric Bayesian networks: Improving theory and reviewing applications. Reliab Eng Syst Saf 2015;144:265–84. http://dx.doi.org/10.1016/j.ress.2015.07.027.
[6] Marcot BG, Penman TD. Advances in Bayesian network modelling: Integration of modelling technologies. Environ Model Software 2019;111:386–93. http://dx.doi.org/10.1016/j.envsoft.2018.09.016.
[7] Hanea AM, Kurowicka D, Cooke RM, Ababei DA. Mining and visualising ordinal data with non-parametric continuous BBNs. Comput Stat Data An 2010;54:668–87. http://dx.doi.org/10.1016/j.csda.2008.09.032.
[8] Morales Nápoles O, Kurowicka D, Roelen A. Eliciting conditional and unconditional rank correlations from conditional probabilities. Reliab Eng Syst Saf 2008;93:699–710. http://dx.doi.org/10.1016/j.ress.2007.03.020.
[9] Delgado Hernández D-J, Morales Nápoles O, De León Escobedo D, Arteaga Arcos J-C. A continuous Bayesian network for earth dams' risk assessment: An application. Struct Infrastruct Eng 2014;10(2):225–38. http://dx.doi.org/10.1080/15732479.2012.731416.
[10] Morales Nápoles O, Delgado Hernández D-J, De León Escobedo D, Arteaga Arcos J-C. A continuous Bayesian network for earth dams' risk assessment: Methodology and quantification. Struct Infrastruct Eng 2014;10(5):589–603. http://dx.doi.org/10.1080/15732479.2012.731416.
[11] Morales-Nápoles O, Hanea AM, Worm DTH. Experimental results about the assessments of conditional rank correlations by experts: Example with air pollution estimates. In: Steenbergen RDJM, Van Gelder PHAJM, Miraglia S, Vrouwenvelder ACWM, editors. Safety, reliability and risk analysis: Beyond the horizon. London: Taylor & Francis; 2014, p. 1359–66.
[12] Kurowicka D, Cooke RM. Distribution-free continuous Bayesian belief nets. In: Wilson A, Keller-McNulty S, Armijo Y, Limnios N, editors. Modern statistical and mathematical methods in reliability. Singapore: World Scientific; 2005, p. 309–22.
[13] Joe H. Dependence modeling with copulas. London: Chapman & Hall/CRC; 2014.
[14] Ale BJM, Bellamy LJ, Cooper J, Ababei D, Kurowicka D, Morales Nápoles O, Spouge J. Analysis of the crash of TK 1951 using CATS. Reliab Eng Syst Saf 2010;95(5):469–77. http://dx.doi.org/10.1016/j.ress.2009.11.014.
[15] Morales Nápoles O, Steenbergen R. Analysis of axle and vehicle load properties through Bayesian networks based on weigh-in-motion data. Eng Syst Saf 2014;125:153–64. http://dx.doi.org/10.1016/j.ress.2014.01.018.
[16] Paprotny D, Morales-Nápoles O. Estimating extreme river discharges in europe through a Bayesian network. Hydrol Earth Syst Sci 2017;21:2615–36. http://dx.doi.org/10.5194/hess-21-2615-2017.
[17] Terefenko P, Paprotny D, Giza A, Morales-Nápoles O, Kubicki A, Walczakiewicz S. Monitoring cliff erosion with LiDAR surveys and Bayesian network-based data analysis. Remote Sens 2019;11:843. http://dx.doi.org/10.3390/rs11070843.
[18] Hincks T, Aspinall WA, Cooke RM, Gernon T. Oklahoma's induced seismicity strongly linked to wastewater injection depth. Science 2018;359:1251–5. http://dx.doi.org/10.1126/science.aap7911.
[19] Aspinall W, Woo G. Counterfactual analysis of runaway volcanic explosions. Front Earth Sci 2019;7:222. http://dx.doi.org/10.3389/feart.2019.00222.
[20] Gradowska PL, Cooke RM. Estimating expected value of information using Bayesian belief networks: A case study in fish consumption advisory. Environ Syst Decis 2014;34:88. http://dx.doi.org/10.1007/s10669-013-9471-4.
[21] Cooke RM, Wielicki B. Probabilistic reasoning about measurements of equilibrium climate sensitivity: Combining disparate lines of evidence. Clim Change 2018;151(3):541–54. http://dx.doi.org/10.1007/s10584-018-2315-y.
[22] LightTwist Software. Uninet. 2019, https://lighttwist-software.com/uninet/. [Accessed 27 April 2020].
[23] Morales Nápoles O, Worm D, van den Haak P, Hanea A, Courage W, Zouch M. Reader for course: Introduction to Bayesian networks, TNO-060-DTM-2012-01756. Delft. the Netherlands: TNO; 2012.
[24] Morales Nápoles O, Worm D, van den Haak P, Hanea A, Courage W, Miraglia S. Reader for course: Introduction to Bayesian networks, TNO-060-DTM-2013-01115. Delft. the Netherlands: TNO; 2013.
[25] Paprotny D, Kreibich H, Morales-Nápoles O, Terefenko P, Schröter K. Estimating exposure of residential assets to natural hazards in Europe using open data. Nat Hazards Earth Syst Sci 2020;20:323–43. http://dx.doi.org/10.5194/nhess-20-323-2020.
[26] Hanea AM, Kurowicka D, Cooke RM. Hybrid method for quantifying and analyzing Bayesian belief nets. Qual Reliab Eng Int 2006;22:709–29. http://dx.doi.org/10.1002/qre.808.
[27] Genest C, Rémillard B, Beaudoin D. Goodness-of-fit tests for copulas: A review and a power study. Insur Math Econ 2009;44:199–213. http://dx.doi.org/10.1016/j.insmatheco.2007.10.005.
[28] Abou Moustafa KT, De La Torre F, Ferrie FP. Designing a metric for the difference between Gaussian densities. In: Angeles J, Boulet B, Clark JJ, Kovecses J, Siddiqi K, editors. Brain, body and machine. Advances in intelligent and soft computing, Berlin: Springer; 2010.
[29] Cooke RM. Experts in uncertainty: Opinion and subjective probability in science. Oxford University Press; 1991, http://refhub.elsevier.com/S2352-7110(18)30060-8/sb1.
[30] Leontaris G, Morales-Nápoles O. ANDURIL - a MATLAB toolbox for analysis and decisions with uncertainty: Learning from expert judgments. SoftwareX 2018;7:313–7. http://dx.doi.org/10.1016/j.softx.2018.07.001.
[31] Morales Nápoles O, Steenbergen R. Large-scale hybrid Bayesian network for traffic load modeling from Weigh-in-Motion system data. J Bridge Eng 2015;20:04014059. http://dx.doi.org/10.1061/(ASCE)BE.1943-5592.0000636.
[32] Paprotny D, Morales Nápoles O, Jonkman SN. Efficient pan-European river flood hazard modelling through a combination of statistical and physical models. Nat Hazards Earth Syst Sci 2017;17:1267–83. http://dx.doi.org/10.5194/nhess-17-1267-2017.
[33] Schröter K, Lüdtke S, Redweik R, Meier J, Bochow M, Ross L, et al. Flood loss estimation using 3D city models and remote sensing data. Environ Modell Softw 2018;105:118–31. http://dx.doi.org/10.1016/j.envsoft.2018.03.032.

[34] Paprotny D, Kreibich H, Morales Nápoles O, Wagenaar D, Castellarin A, Carisi F, et al. A probabilistic approach to estimating residential losses from different flood types. Nat Hazards 2020. in review.

[35] Paprotny D, Kreibich H, Morales Nápoles O, Castellarin A, Carisi F. Schröter exposure and vulnerability estimation for modelling flood losses to commercial assets in Europe. Sci Total Environ 2020;737:140011. http://dx.doi.org/10.1016/j.scitotenv.2020.140011.

[36] 't Hart CMP, Leontaris G, O. Morales-Nápoles. Update (1.1) to ANDURIL — A MATLAB toolbox for analysis and decisions with uncertainty: Learning from expert judgments: ANDURYL. SoftwareX 2019;10:100295. http://dx.doi.org/10.1016/j.softx.2019.100295.

[37] Rongen G, 't Hart CMP, Leontaris G, O. Morales-Nápoles. Update (1.2) to ANDURIL and ANDURYL: Performance improvements and a graphical user interface. SoftwareX 2020;12:100497. http://dx.doi.org/10.1016/j.softx.2020.100497.