Evaluating Probabilistic Forecasts with scoringRules

Alexander Jordan
Heidelberg Institute for Theoretical Studies

Fabian Krüger
Heidelberg University

Sebastian Lerch
Heidelberg Institute for Theoretical Studies
Karlsruhe Institute of Technology

Abstract

Probabilistic forecasts in the form of probability distributions over future events have become popular in several fields including meteorology, hydrology, economics, and demography. In typical applications, many alternative statistical models and data sources can be used to produce probabilistic forecasts. Hence, evaluating and selecting among competing methods is an important task. The **scoringRules** package for R provides functionality for comparative evaluation of probabilistic models based on proper scoring rules, covering a wide range of situations in applied work. This paper discusses implementation and usage details, presents case studies from meteorology and economics, and points to the relevant background literature.

Keywords: comparative evaluation, ensemble forecasts, out-of-sample evaluation, predictive distributions, proper scoring rules, score calculation, R.

1. Introduction: Forecast evaluation

Forecasts are generally surrounded by uncertainty, and being able to quantify this uncertainty is key to good decision making. Accordingly, probabilistic forecasts in the form of predictive probability distributions over future quantities or events have become popular over the last decades in various fields including meteorology, climate science, hydrology, seismology, economics, finance, demography and political science. Important examples include the United Nation's probabilistic population forecasts (Raftery *et al.* 2014), inflation projections issued by the Bank of England (see, e.g., Clements 2004), or the now widespread use of probabilistic ensemble methods in meteorology (Gneiting and Raftery 2005; Leutbecher and Palmer 2008). For recent reviews see Gneiting and Katzfuss (2014) and Raftery (2016).

With the proliferation of probabilistic models arises the need for tools to evaluate the appropriateness of models and forecasts in a principled way. Various measures of forecast performance have been developed over the past decades to address this demand. Scoring rules are functions $S(F, y)$ that evaluate the accuracy of a forecast distribution F, given that an outcome y was observed. As such, they allow to compare alternative models, a crucial ability given the variety of theories, data sources and statistical specifications available in many situations. Conceptually, scoring rules can be thought of as error measures for distribution functions: While the squared error $\text{SE}(x, y) = (y - x)^2$ measures the performance of a point forecast x, a scoring rule $S(F, y)$ measures the performance of a distribution forecast F.

This paper introduces the R (R Core Team 2017) software package **scoringRules** (Jordan *et al.* 2017), which provides functions to compute scoring rules for a variety of distributions F that come up in applied work, and popular choices of S. Two main classes of probabilistic forecasts are parametric distributions and distributions that are not known analytically, but are indirectly described through a sample of simulation draws. For example, Bayesian forecasts produced via Markov chain Monte Carlo (MCMC) methods take the latter form. Hence, the **scoringRules** package provides a general framework for model evaluation that covers both classical (frequentist) and Bayesian forecasting methods.

The **scoringRules** package aims to be a comprehensive library for computing scoring rules. We offer implementations of several known (but not routinely applied) formulas, and implement some closed-form expressions that were previously unavailable. Whenever more than one implementation variant exists, we offer statistically principled default choices. The package contains the continuous ranked probability score (CRPS) and the logarithmic score, as well as the multivariate energy score and variogram score. All these scoring rules are proper, which means that forecasters have an incentive to state their true belief, see Section 2.

It is worth emphasizing that scoring rules are designed for *comparative* forecast evaluation. That is, one wants to know whether model A or model B provides better forecasts, in terms of a proper scoring rule. Comparative forecast evaluation is of interest either for choosing a specification for future use, or for comparing various scientific approaches. A distinct, complementary issue is to check the suitability of a given model via tools for *absolute* forecast evaluation (such as probability integral transforms, see, e.g., Gneiting and Katzfuss 2014). To retain focus, the **scoringRules** package does not cover absolute forecast evaluation.

Proper scoring rules are useful for model comparisons. In that sense, scoringRules is broadly related to all software packages which help users determine an appropriate model for the data at hand. Perhaps most fundamentally, the **stats** (R Core Team 2017) package provides the traditional Akaike and Bayes information criteria in order to select among linear models. Numerous add-on packages provide more recent or more specialized methods. The packages **caret** (Kuhn *et al.* 2017) and **forecast** (Hyndman and Khandakar 2008) provide cross-validation tools suitable for cross-sectional and time series data, respectively. The **loo** (Vehtari *et al.* 2016) package implements recent proposals to select among Bayesian models. In comparison to existing software for model comparisons, a key novelty of the **scoringRules** package is its extensive coverage of the CRPS. The latter scoring rule is attractive for both practical and theoretical reasons (Gneiting and Raftery 2007; Krüger *et al.* 2016). In the past, more widespread use of the CRPS has been hampered by computational challenges. By providing analytical formulas and efficient numerical implementations, the **scoringRules** package enables convenient usage of the CRPS in applied work.

To the best of our knowledge, **scoringRules** is the first R package designed as a library for computing proper scoring rules. However, a number of existing R packages include scoring rule computations for more specific empirical situations: The **ensembleBMA** (Fraley *et al.* 2015) and **ensembleMOS** (Yuen *et al.* 2017) packages contain formulas for the CRPS of a small subset of the distributions listed in Table 1 which are relevant for post-processing ensemble weather forecasts (Fraley *et al.* 2011), and can only be applied to specific data structures utilized in the packages. The **surveillance** (Meyer *et al.* 2017) package provides functions to compute the logarithmic score and other scoring rules for count data models in epidemiology. By contrast, the distributions contained in **scoringRules** are relevant in applications across disciplines and the score functions are generally applicable. Furthermore, the **verification** (Na-

tional Center for Atmospheric Research 2015) and **SpecsVerification** (Siegert 2015) packages contain implementations of the CRPS for simulated forecast distributions. Our contribution in that domain is twofold: First, we offer efficient implementations, which is especially relevant when the forecast distribution is a large MCMC sample. MCMC methods are popular across the disciplines, and many sophisticated R implementations are available (see, e.g., Kastner 2016; Carpenter *et al.* 2017, for recent examples). Second, we include various implementation options, and propose principled default settings based on our recent research (Krüger *et al.* 2016).

For programming languages other than R, implementations of proper scoring rules are sparse, and generally cover a much narrower functionality than the **scoringRules** package. For Python, the **properscoring** package (The Climate Corporation 2015) provides implementations of the CRPS for Gaussian distributions and for forecast distributions given by a discrete sample. Several institutionally supported software packages include tools to compute scoring rules, but typically require input in specific data formats and are tailored towards operational use at meteorological institutions. The Model Evaluation Tools[1] software (Developmental Testbed Center 2017) developed by the National Center for Atmospheric Research provides code to compute the CRPS based on a sample from the forecast distribution. However, note that a Gaussian approximation is applied which can be problematic if the underlying distribution is not Gaussian, see Krüger *et al.* (2016). The Ensemble Verification System[2] program (Brown *et al.* 2010) developed by the Hydrological Ensemble Prediction group of the US National Weather Service also provides an implementation of the CRPS for discrete samples. For a general overview of software for forecast evaluation in meteorology, see Pocernich (2012).

The remainder of this paper is organized as follows. Section 2 provides some theoretical background on scoring rules, and introduces the logarithmic score and the continuous ranked probability score. Section 3 gives an overview of the score calculation functionality in the **scoringRules** package and presents the implementation of univariate proper scoring rules. In Section 4, we give usage examples by application in case studies. In a meteorological example of accumulated precipitation forecasts, we cover the comparative evaluation of ensemble system output from numerical weather prediction models with parametric forecast distributions from statistical post-processing models. A second case study shows how using analytical information of a Bayesian time series model for the growth rate of the US economy's gross domestic product (GDP) can help in evaluating the model's simulation output. Definitions and details on the use of multivariate scoring rules are provided in Section 5. The paper closes with a discussion in Section 6.

2. Theoretical background

Probabilistic forecasts can be given in various forms. The most important cases are parametric distributions in the form of analytical cumulative distribution functions (CDFs) or probability density functions (PDFs), and forecasts that take the form of a simulated sample. The latter form is often used if the predictive distribution is not available analytically. Here, we give a brief overview of the theoretical background.

[1] available at http://www.dtcenter.org/met/users/index.php
[2] available at https://amazon.nws.noaa.gov/ohd/evs/evs.html

2.1. Proper scoring rules

Let Ω denote the set of possible values of the quantity of interest, Y, and let $\mathcal{F}$ denote a convex class of probability distributions on Ω. A *scoring rule* is a function

$$S : \mathcal{F} \times \Omega \longrightarrow \mathbb{R} \cup \{\infty\}$$

that assigns numerical values to pairs of forecasts $F \in \mathcal{F}$ and observations $y \in \Omega$. For now, we restrict our attention to univariate observations and set $\Omega = \mathbb{R}$ or subsets thereof, and identify probabilistic forecasts F with the associated CDF F or PDF f. In Section 5, we will consider multivariate scoring rules for which $\Omega = \mathbb{R}^d$.

We consider scoring rules to be negatively oriented, such that a lower score indicates a better forecast. For a *proper* scoring rule, the expected score is optimized if the true distribution of the observation is issued as a forecast, i.e., if

$$\mathsf{E}_{Y\sim G}\, S(G, Y) \leq \mathsf{E}_{Y\sim G}\, S(F, Y)$$

for all $F, G \in \mathcal{F}$. A scoring rule is further called *strictly proper* if equality holds only if $F = G$. Being proper is critically important for comparative evaluation and ranking of forecasts in that a proper scoring rule compels the forecaster to truthfully report what she thinks is the true distribution. See Gneiting and Raftery (2007) for a detailed review of the mathematical properties of proper scoring rules.

Popular examples of proper scoring rules for $\Omega = \mathbb{R}$ include the logarithmic score and the continuous ranked probability score. The *logarithmic score* (LogS; Good 1952) is defined as

$$\mathrm{LogS}(F, y) = -\log(f(y)),$$

where F admits a PDF f, and is a strictly proper scoring rule relative to the class of probability distributions with densities. The *continuous ranked probability score* (Matheson and Winkler 1976) is defined in terms of the predictive CDF F and is given by

$$\mathrm{CRPS}(F, y) = \int_{\mathbb{R}} (F(z) - \mathbb{1}\{y \leq z\})^2 \mathrm{d}z, \tag{1}$$

where $\mathbb{1}\{y \leq z\}$ denotes the indicator function which is 1 if $y \leq z$ and 0 otherwise. If the first moment of F is finite, the CRPS can be written as

$$\mathrm{CRPS}(F, y) = \mathsf{E}_F|X - y| - \frac{1}{2}\mathsf{E}_{F,F}|X - X'|,$$

where X and X' are independent random variables with distribution F, see Gneiting and Raftery (2007). The CRPS is a strictly proper scoring rule for the class of probability distributions with finite first moment. Closed-form expressions of the integral in equation (1) can be obtained for many parametric distributions and allow for exact and efficient computation of the CRPS. They are implemented in the **scoringRules** package for a range of parametric families, see Table 1 for an overview, and are provided in the "CRPS formulas" vignette.

2.2. Model assessment based on simulated forecast distributions

In various applications, the forecast distribution of interest F is not available in an analytic form, but only through a simulated sample $X_1, \ldots, X_m \sim F$. Examples include Bayesian

Distribution	**Family arg.**	CRPS	LogS	**Additional information**
Distributions for variables on the real line				
Laplace	`"lapl"`	✓	✓	
logistic	`"logis"`	✓	✓	
normal	`"norm"`	✓	✓	
mixture of normals	`"mixnorm"`	✓	✓	
Student's t	`"t"`	✓	✓	flex. location, scale
two-piece exponential	`"2pexp"`	✓	✓	
two-piece normal	`"2pnorm"`	✓	✓	
Distributions for non-negative variables				
exponential	`"exp"`	✓	✓	
gamma	`"gamma"`	✓	✓	
log-Laplace	`"llapl"`	✓	✓	
log-logistic	`"llogis"`	✓	✓	
log-normal	`"lnorm"`	✓	✓	
Distributions with flexible support and/or point masses				
beta	`"beta"`	✓	✓	flex. limits
uniform	`"unif"`	✓	✓	flex. limits
exponential	`"exp2"`		✓	flex. location, scale
	`"expM"`	✓		flex. location, scale, point mass
gen. extreme value	`"gev"`	✓	✓	
gen. Pareto	`"gpd"`	✓	✓	flex. point mass (CRPS only)
logistic	`"tlogis"`	✓	✓	truncated
	`"clogis"`	✓		censored
	`"gtclogis"`	✓		flex. point masses
normal	`"tnorm"`	✓	✓	truncated
	`"cnorm"`	✓		censored
	`"gtcnorm"`	✓		flex. point masses
Student's t	`"tt"`	✓	✓	truncated
	`"ct"`	✓		censored
	`"gtct"`	✓		flex. point masses
Distributions for discrete variables				
negative binomial	`"nbinom"`	✓	✓	
Poisson	`"pois"`	✓	✓	

Table 1: List of implemented parametric families for which CRPS and LogS can be computed via `crps()` and `logs()`. The character string is the corresponding value for the `family` argument.

forecasting applications where the sample is generated by a MCMC algorithm, or ensemble weather forecasting applications where the different sample values are generated by numerical weather prediction models with different model physics and/or initial conditions. In order to compute the value of a proper scoring rule, the simulated sample needs to be converted into a distribution with a closed-form expression. The implementation choices and default settings in the **scoringRules** package follow the findings of Krüger *et al.* (2016) who provide a systematic analysis of probabilistic forecasting based on MCMC output.

For the CRPS, the empirical CDF

$$\hat{F}_m(z) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}\{X_i \leq z\}$$

is a natural approximation of the predictive CDF. In this case, the CRPS reduces to

$$\mathrm{CRPS}(\hat{F}_m, y) = \frac{1}{m} \sum_{i=1}^{m} |X_i - y| - \frac{1}{2m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} |X_i - X_j| \tag{2}$$

which allows to compute the CRPS directly from the simulated sample, see Grimit *et al.* (2006). Implementations of equation (2) are rather inefficient with computational complexity $\mathcal{O}(m^2)$, and can be improved upon with representations using the order statistics $X_{(1)}, \ldots, X_{(m)}$, i.e., the sorted simulated sample, thus achieving an average $\mathcal{O}(m \log m)$ performance. In the **scoringRules** package, we use an algebraically equivalent representation of the CRPS based on the generalized quantile function (Laio and Tamea 2007), leading to

$$\mathrm{CRPS}(\hat{F}_m, y) = \frac{2}{m^2} \sum_{i=1}^{m} (X_{(i)} - y) \left(m \mathbb{1}\{y < X_{(i)}\} - i + \frac{1}{2} \right), \tag{3}$$

which Murphy (1970) reported in the context of the precursory, discrete version of the CRPS. We refer to Jordan (2016) for details.

In contrast to the CRPS, the computation of the LogS requires a predictive density. An estimator can be obtained with classical nonparametric kernel density estimation (KDE, e.g. Silverman 1986). However, this estimator is valid only under stringent theoretical assumptions, and can be fragile in practice. In an MCMC context, a *mixture-of-parameters* estimator which utilizes a simulated sample of parameter draws rather than draws from the posterior predictive distribution is a better and often much more efficient choice, see Krüger *et al.* (2016). This mixture-of-parameters estimator is specific to the model being used, but can often be implemented using functionality for parametric forecast distributions. We provide an example in Section 4.2.

3. Score calculation functionality

Table 2 gives an overview of the functions that are available for score calculations in the **scoringRules** package. We provide S3 generic functions for the CRPS and LogS,

```
crps(y, ...)
logs(y, ...)
```

and include S3 methods for the class '`numeric`':

S3 generics & methods	**Score functions**	
	Parametric (see Table 1)	Empirical
`crps`	`crps_norm`	`crps_sample`
`logs`	`...`	`logs_sample`
`crps.numeric`	`logs_norm`	`es_sample`
`logs.numeric`	`...`	`vs_sample`

Table 2: Overview of the score calculation functionality. The S3 methods for the class '`numeric`' are wrappers around the score functions for parametric forecasts, with stricter input checks. For the full list of implemented parametric distributions we refer to Table 1.

```
crps.numeric(y, family, ...)
logs.numeric(y, family, ...)
```

For these methods, the input for the argument `y` is a numeric vector of realized values and the input for the argument `family` is a string which specifies the parametric family. We refer to Table 1 for an overview of implemented parametric families. Depending on the chosen parametric family, the score functions expect further input in form of numeric parameter vectors. All numerical arguments should be of the same length, with the exception that vectors of length one will be recycled. The functions return a vector of score values. For example, the CRPS and LogS of a normal distribution can be computed as follows:

```
R> obs <- rnorm(10)
R> crps(obs, family = "normal", mean = c(1:10), sd = c(1:10))

 [1] 0.288 1.625 1.570 2.003 2.744 3.688 3.270 4.884 4.162 6.067

R> logs(obs, family = "normal", mean = c(1:10), sd = c(1:10))

 [1] 0.988 2.434 2.404 2.660 2.951 3.229 3.172 3.510 3.417 3.728
```

As another example, we could define functions that only depend on the argument `y` with fixed scalar parameters:

```
R> crps_y <- function(y) crps(y, family = "gamma", shape = 2, scale = 1.5)
R> logs_y <- function(y) logs(y, family = "gamma", shape = 2, scale = 1.5)
```

In Figure 1 we use these functions to illustrate the dependence between the score value and the observation in an example of a gamma distribution as forecast. The logarithmic score rapidly increases at the right-sided limit of 0, and the minimum score value is attained if the observation equals the predictive distribution's mode. By contrast, the CRPS is more symmetric around the minimum that is attained at the median value of the forecast distribution, particularly, it increases more slowly as the observation approaches 0. The methods `crps.numeric()` and `logs.numeric()` are wrappers for underlying worker functions that cover specific parametric families, for example:

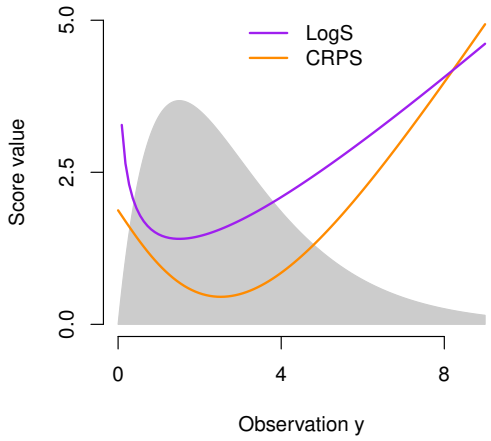

Figure 1: Values of LogS and CRPS as functions of the observation. The forecast distribution is given by a gamma distribution with a shape parameter of 2 and a scale parameter of 1.5. A scaled version of the forecast density is shown in gray.

```
crps_norm(y, mean = 0, sd = 1, location = mean, scale = sd)
logs_norm(y, mean = 0, sd = 1, location = mean, scale = sd)
```

Note that these functions mainly rely on the input checks of basic operators and functions of R. They are aimed towards expert users and package developers, e.g., for use in numerical optimization algorithms. The main differences to `crps.numeric()` and `logs.numeric()` lie in the handling of non-admissible input parameter values, i.e., the avoidance of error messages, and in using the base R recycling behavior for input arguments of different lengths.

Often forecast distributions can only be given as simulated samples, e.g., ensemble systems in weather prediction (Section 4.1) or MCMC output in econometrics (Section 4.2). We provide functions for both univariate and multivariate samples. The latter are discussed in Section 5, whereas the former are presented here:

```
crps_sample(y, dat, method = "edf", w = NULL, bw = NULL,
  num_int = FALSE, show_messages = TRUE)
logs_sample(y, dat, bw = NULL, show_messages = FALSE)
```

The input for `y` is a vector of observations, and the input for `dat` is a matrix with the number of rows matching the length of `y` and each row comprising one simulated sample, e.g., as in the following evaluation of a probabilistic forecast given by a random sample from a normal distribution with mean 2 and standard deviation 3:

```
R> obs_n <- c(0, 1, 2)
R> sample_nm <- matrix(rnorm(3e4, mean = 2, sd = 3), nrow = 3)
R> crps_sample(obs_n, dat = sample_nm)

[1] 1.216 0.833 0.710

R> logs_sample(obs_n, dat = sample_nm)
```

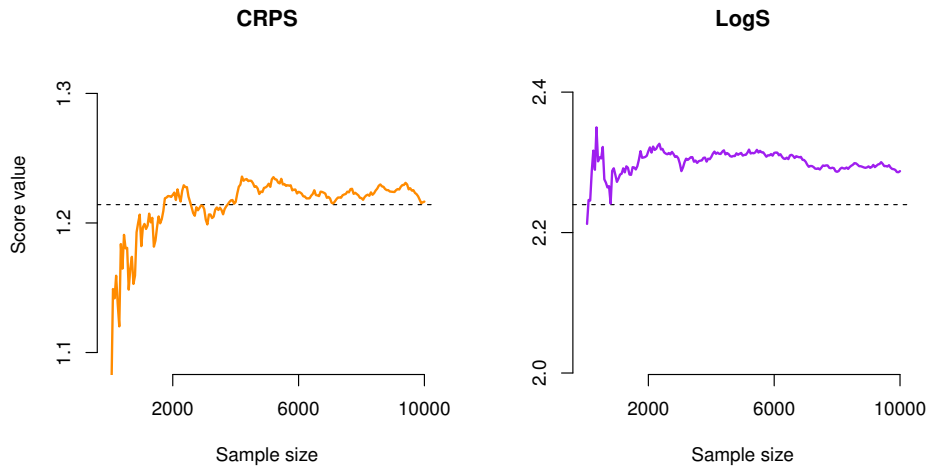

Figure 2: The scores of a Gaussian forecast distribution with mean 2 and standard deviation 3 when a value of 0 is observed, estimated from an independent sample from the predictive distribution, and shown as a function of the size of the (expanding) sample. The horizontal line represents the analytically calculated score.

```
[1] 2.29 2.10 2.04
```

When `y` has length 1 then `dat` may also be a vector. Random sampling from the forecast distribution can be seen as an option to approximate the values of the proper scoring rules. To empirically assess the quality of this approximation and to illustrate the use of the score functions, consider the following Gaussian toy example:

```
R> obs_1 <- obs_n[1]
R> sample_m <- sample_nm[1, ]
R> mgrid <- seq(from = 50, to = length(sample_m), by = 50)
R> crps_approx <- logs_approx <- numeric(length(mgrid))
R> for (i in seq_along(mgrid)) {
+    size <- mgrid[i]
+    crps_approx[i] <- crps_sample(obs_1, dat = sample_m[1:size])
+    logs_approx[i] <- logs_sample(obs_1, dat = sample_m[1:size])
+  }
```

The true CRPS and LogS values can be calculated using the `crps()` and `logs()` functions. Figure 2 graphically illustrates how the scores based on sampling approximations become more accurate as the sample size increases.

The `method` argument controls which approximation method is used in `crps_sample()`, with possible choices given by `"edf"` (empirical distribution function) and `"kde"` (kernel density estimation). The default choice `"edf"` corresponds to computing the approximation from equation (2), implemented as in equation (3). A vector or matrix of weights, matching the input for `dat`, can be passed to the argument `w` to calculate the CRPS for any distribution with a finite number of outcomes.

For kernel density estimation, i.e., the default in `logs_sample()` and the corresponding `method` in `crps_sample()`, we use a Gaussian kernel to estimate the predictive distribu-

tion. Kernel density estimation is an unusual choice in the case of the CRPS, but it is the only implemented option for evaluating the LogS of a simulated sample; this is because an estimated *density* is required for the LogS. The `bw` argument allows to manually select a bandwidth parameter for kernel density estimation; by default, the `bw.nrd()` function from the **stats** (R Core Team 2017) package is employed.

4. Usage examples

4.1. Probabilistic weather forecasting via ensemble post-processing

In numerical weather prediction (NWP), physical processes in the atmosphere are modeled through systems of partial differential equations that are solved numerically on three-dimensional grids. To account for major sources of uncertainty, weather forecasts are typically obtained from multiple runs of NWP models with varying initial conditions and model physics resulting in a set of deterministic predictions, called the 'forecast ensemble'. While ensemble predictions are an important step from deterministic to probabilistic forecasts, they tend to be biased and underdispersive (such that, empirically, the actual observation falls outside the range of the ensemble too frequently). Hence, ensembles require some form of statistical post-processing. Over the past decade, a variety of approaches to statistical post-processing has been proposed, including non-homogeneous regression (Gneiting *et al.* 2005) and Bayesian model averaging (Raftery *et al.* 2005).

Here we illustrate how to evaluate post-processed ensemble forecasts of precipitation, based on data and methods from the **crch** package (Messner *et al.* 2016). We model the conditional distribution of precipitation accumulation, $Y \geq 0$, given the ensemble forecasts $X_1, \ldots, X_m$ using censored non-homogeneous regression models of the form

$$\mathsf{P}(Y = 0 | X_1, \ldots, X_m) = F_\theta(0), \tag{4}$$

$$\mathsf{P}(Y \leq y | X_1, \ldots, X_m) = F_\theta(y), \text{ for } y > 0, \tag{5}$$

where F_θ is the CDF of a continuous parametric distribution with parameters θ. Equations (4) and (5) specify a mixed discrete-continuous forecast distribution for precipitation: There is a positive probability of observing no precipitation at all ($Y = 0$), however, if $Y > 0$, it can take many possible values y. In order to incorporate information from the raw forecast ensemble, we let θ be a function of $X_1, \ldots, X_m$, i.e., we use features of the raw ensemble to determine the parameters of the forecast distribution. Specifically, we consider different location-scale families F_θ and model the location parameter μ as a linear function of the ensemble mean $\bar{X} = \frac{1}{m} \sum_{i=1}^{m} X_i$,

$$\mu = a_0 + a_1 \bar{X},$$

and the scale parameter σ as linear function of the logarithm of the standard deviation s of the ensemble,

$$\log(\sigma) = b_0 + b_1 \log(s).$$

A logarithmic link function is used to ensure positivity of the scale parameter. The coefficients a_0, a_1, b_0, b_1 can be estimated using maximum likelihood approaches implemented in the **crch** package. The choice of a suitable parametric family F_θ is not obvious. Following Messner *et al.* (2016), we thus consider three alternative choices: the logistic, Gaussian, and Student's

t distributions. For details and further alternatives, see, e.g., Messner *et al.* (2014); Scheuerer (2014) and Scheuerer and Hamill (2015a).

The **crch** package contains a data set of ensemble forecasts and observations of precipitation for Innsbruck (Austria). The precipitation amounts are accumulated over 3 days, and the corresponding 11 member ensemble forecasts are forecasts of accumulated precipitation amount between 5 and 8 days ahead. Following Messner *et al.* (2016) we model the square root of precipitation amounts, and omit forecast cases where the ensemble has a standard deviation of zero. From Messner *et al.* (2016):

```
R> library(crch)
R> data(RainIbk)
R> RainIbk <- sqrt(RainIbk)
R> RainIbk$ensmean <- apply(RainIbk[,grep('^rainfc',names(RainIbk))], 1, mean)
R> RainIbk$enssd <- apply(RainIbk[,grep('^rainfc',names(RainIbk))], 1, sd)
R> RainIbk <- subset(RainIbk, enssd > 0)
```

We split the data into a training set from April 2001 until November 2004, and an out-of-sample (or test sample) evaluation period using the remainder of the available data set from January 2005 to September 2013.

```
R> data_train <- subset(RainIbk, as.Date(rownames(RainIbk)) <= "2004-11-30")
R> data_eval <- subset(RainIbk, as.Date(rownames(RainIbk)) >= "2005-01-01")
```

Then, we estimate the censored regression models that are based on the logistic, Student's t, and Gaussian distributions, and produce the parameters of the forecast distributions for the evaluation period using built-in functionality of the **crch** package. We only show the code for the Gaussian model since it can be adapted straightforwardly for the logistic and Student's t models.

```
R> CRCHgauss <- crch(rain ~ ensmean | log(enssd), data_train,
+    dist = "gaussian", left = 0)
R> gauss_mu <- predict(CRCHgauss, data_eval, type = "location")
R> gauss_sc <- predict(CRCHgauss, data_eval, type = "scale")
```

The raw ensemble of forecasts is a natural benchmark for comparison since interest commonly lies in quantifying the gains in forecast accuracy that result from post-processing:

```
R> ens_fc <- data_eval[, grep('^rainfc', names(RainIbk))]
```

Figure 3 shows the models' forecast distributions in three illustrative cases. To evaluate forecast performance in the entire out of sample period, we use the function `crps()` for the model outputs and the function `crps_sample()` to compute the CRPS of the raw ensemble. Note that we have to turn `ens_fc` into an object of class '`matrix`' manually.

```
R> obs <- data_eval$rain
R> gauss_crps <- crps(obs, family = "cnorm", location = gauss_mu,
+    scale = gauss_sc, lower = 0, upper = Inf)
R> ens_crps <- crps_sample(obs, dat = as.matrix(ens_fc))
```

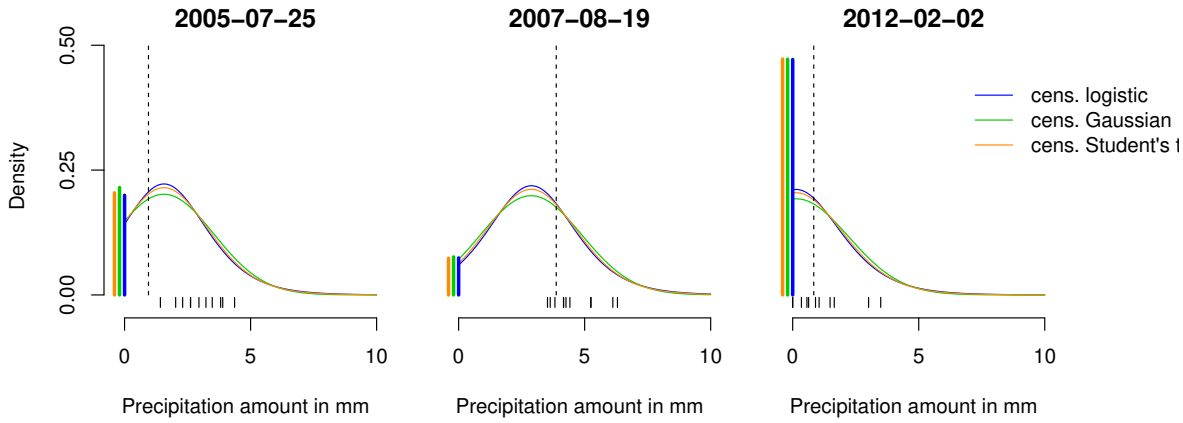

Figure 3: Illustration of the forecast distributions of the censored regression models for three illustrative 3-day accumulation periods (plot title indicates end of period). The predicted probabilities of zero precipitation are shown as solid thick vertical lines at 0, and the colored thin lines indicate the upper tail on the positive half axis of the forecast densities f_θ, c.f. equations (4) and (5). The raw ensemble forecasts are shown as short line segments at the bottom, and the realizing observation is indicated by the long dashed line.

The mean CRPS values indicate that all post-processing models substantially improve upon the raw ensemble forecasts. There are only small differences between the censored regression models, with the models based on the logistic and Student's t distributions slightly outperforming the model based on a normal distribution.

```
CRCHlogis CRCHgauss CRCHstud Ensemble
    0.875     0.876    0.875     1.32
```

4.2. Bayesian forecasts of US GDP growth rate

We next present a data example based on a Markov Switching autoregressive model for US GDP growth, which was first proposed by Hamilton (1989). We consider a variant of the model that incorporates time-varying heteroscedasticity, which is a salient feature of US GDP growth: For example, the series was much more volatile in the 1970s than in the 1990s. The model is estimated using Bayesian Markov chain Monte Carlo methods (Frühwirth-Schnatter 2006). Our implementation closely follows Krüger *et al.* (2016, Section 5), and uses the `ar_ms()` function, and the dataset `gdp`, included in the **scoringRules** package.

As a first step, we split the data into a training sample of observations containing the data before 2014's first quarter, and an evaluation period containing only the four quarters of 2014:

```
R> data(gdp, package = "scoringRules")
R> data_train <- subset(gdp, vint == "2014Q1")
R> data_eval <- subset(gdp, vint == "2015Q1" & grepl("2014", dt))
```

As is typical for MCMC-based analysis, the model's forecast distribution F_0 is not available as an analytical formula, but must be approximated in some way. Following Krüger *et al.* (2016), a generic MCMC algorithm to generate samples of the parameter vector θ and sample from the posterior predictive distribution proceeds as follows:

- fix $\theta_0 \in \Theta$
- for $i = 1, \ldots, m$,
  - draw $\theta_i \sim \mathcal{K}(\cdot|\theta_{i-1})$, where $\mathcal{K}$ is a transition kernel that is specific to the model under use
  - draw $X_i \sim F_c(\cdot|\theta_i)$, where F_c denotes the conditional distribution given the parameter values.

We use the function `ar_ms()` to fit the model and produce forecasts for the four quarters of 2014 based on the information available at the end of year 2013, i.e., a single prediction case where the forecast horizon extends from one to four quarters ahead. Here, the conditional distribution F_c is Gaussian, and we run the chain for 20 000 iterations.

```
R> h <- 4; m <- 20000
R> fc_params <- ar_ms(data_train$val, forecast_periods = h, n_rep = m)
```

This yields a simulated sample corresponding to $\{\theta_1, \ldots, \theta_m\}$, where we obtain matrices of parameters for the mean and standard deviation. We transpose these matrices to have the rows correspond to the observations, and columns represent the position in the Markov chain:

```
R> mu <- t(fc_params$fcMeans)
R> Sd <- t(fc_params$fcSds)
```

Next, we draw the sample of possible observations corresponding to $\{X_1, \ldots, X_m\}$ conditional on the Gaussian assumption and the available parameter information:

```
R> X <- matrix(rnorm(h * m, mean = mu, sd = Sd), nrow = h, ncol = m)
```

This gives rise to two competing estimators of the posterior predictive distribution F_0. The mixture-of-parameters estimator (MPE)

$$\hat{F}_m^{\mathrm{MP}}(z) = \frac{1}{m} \sum_{i=1}^{m} F_c(z|\theta_i), \tag{6}$$

builds on the simulated parameter values by mixing a series of Gaussian distributions uniformly, whereas the empirical CDF based approximation

$$\hat{F}_m^{\mathrm{ECDF}}(z) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}\{X_i \leq z\}$$

utilizes the simulated sample from the conditional distribution given the parameter values, instead of building on the simulated parameter values directly. A standard choice for a smoother approximation is to replace the indicator function with a Gaussian kernel, as in the `logs_sample()` function.

The two alternative estimators are illustrated in Figure 4: For each date, the histogram represents a simulated sample from the model's forecast distribution, and the black line

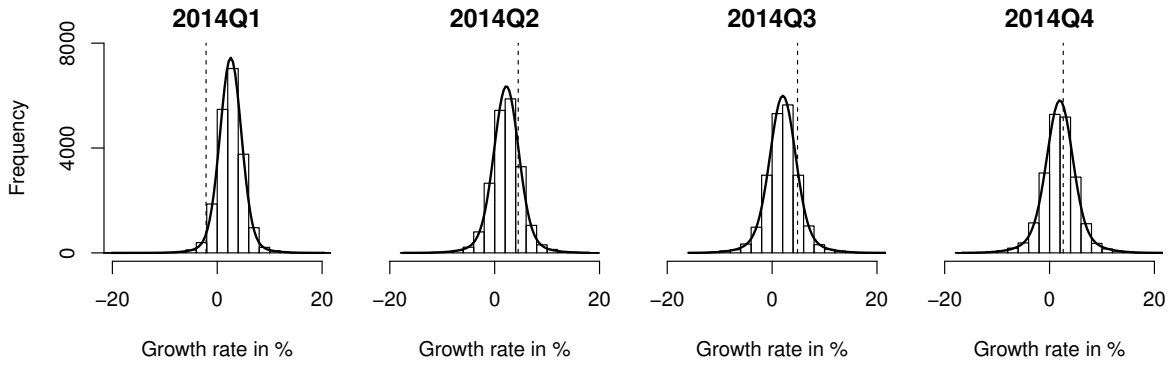

Figure 4: Forecast distributions for the growth rate of US GDP. The forecasts stem from a Bayesian time series model, as detailed in Krüger *et al.* (2016, Section 5). Histograms summarize simulated forecast draws at each date. Mixture-of-normals approximation to distribution shown in black; realizing observations shown by dashed line.

indicates the mixture-of-parameters estimator.[3] We can observe a distinct decrease in the forecast's certainty as the forecast horizon increases from one to four quarters ahead.

Finally, we evaluate the CRPS and LogS for the approximated forecast distributions described above. The mixture-of-parameters estimator $\hat{F}_m^{\mathrm{MP}}$ can be evaluated with the functions `crps()` and `logs()`, and $\hat{F}_m^{\mathrm{ECDF}}$ can be evaluated with the functions `crps_sample()` and `logs_sample()`:

```
R> obs <- data_eval$val
R> names(obs) <- data_eval$dt
R> w <- matrix(1/m, nrow = h, ncol = m)
R> crps_mpe <- crps(obs, "normal-mixture", m = mu, s = Sd, w = w)
R> logs_mpe <- logs(obs, "normal-mixture", m = mu, s = Sd, w = w)
R> crps_ecdf <- crps_sample(obs, X)
R> logs_kde <- logs_sample(obs, X)
R> print(cbind(crps_mpe, crps_ecdf, logs_mpe, logs_kde))
```

```
       crps_mpe crps_ecdf logs_mpe logs_kde
2014Q1    3.461     3.455     4.02     3.93
2014Q2    1.361     1.354     2.29     2.30
2014Q3    1.712     1.724     2.53     2.56
2014Q4    0.727     0.731     1.96     1.98
```

The score values are quite similar for both estimators, which seems natural given the large number of 20 000 MCMC draws. For the logarithmic score in particular, the MPE should be preferred over the KDE based estimator on theoretical grounds, see Krüger *et al.* (2016).

5. Multivariate scoring rules

[3]The algorithm and approximation methods just sketched are not idiosyncratic to our example, but arise whenever a Bayesian model is used for forecasting. For illustrative R implementations of other Bayesian models, see, e.g., the packages **bayesgarch** (Ardia and Hoogerheide 2010) and **stochvol** (Kastner 2016).

The basic concept of proper scoring rules can be extended to multivariate forecast distributions for which the support Ω is given by $\mathbb{R}^d, d \in \{2, 3, \ldots\}$. A variety of multivariate proper scoring rules has been proposed in the literature. Most of these scoring rules are defined for probabilistic forecasts given as samples from the forecast distributions.

Let $\mathbf{y} = (y^{(1)}, \ldots, y^{(d)}) \in \Omega = \mathbb{R}^d$, and let F denote a forecast distribution on $\mathbb{R}^d$ given through m discrete samples $\mathbf{X}_1, \ldots, \mathbf{X}_m$ from F with $\mathbf{X}_i = (X_i^{(1)}, \ldots, X_i^{(d)}) \in \mathbb{R}^d, i = 1, \ldots, m$. The **scoringRules** package provides implementations of the *energy score* (ES; Gneiting *et al.* 2008),

$$\mathrm{ES}(F, y) = \frac{1}{m} \sum_{i=1}^{m} \|\mathbf{X}_i - \mathbf{y}\| - \frac{1}{2m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} \|\mathbf{X}_i - \mathbf{X}_j\|,$$

where $\|\cdot\|$ denotes the Euclidean norm on $\mathbb{R}^d$, and the *variogram score of order p* (VS^p; Scheuerer and Hamill 2015b),

$$\mathrm{VS}^p(F, y) = \sum_{i=1}^{d} \sum_{j=1}^{d} w_{i,j} \left(\left| y^{(i)} - y^{(j)} \right|^p - \frac{1}{m} \sum_{k=1}^{m} \left| X_k^{(i)} - X_k^{(j)} \right|^p \right)^2.$$

In the definition of VS^p, $w_{i,j}$ is a non-negative weight that allows to emphasize or down-weight pairs of component combinations based on subjective expert decisions, and p is the order of the variogram score. Typical choices of p include 0.5 and 1.

ES and VS^p are implemented for multivariate forecast distributions given through simulated samples as functions

```
es_sample(y, dat)
vs_sample(y, dat, w = NULL, p = 0.5)
```

In both cases, the vector of observations, `y`, is required to be a vector of length d, and the corresponding forecasts, `dat`, have to be given as a $d \times m$ matrix, the columns of which are the simulated samples $\mathbf{X}_1, \ldots, \mathbf{X}_m$ from the multivariate forecast distribution.[4] Note that these functions are not vectorized, i.e., they can only evaluate a single multivariate forecast case and always return a single number.

In the following, we give a usage example of the multivariate scoring rules using the results from the economic case study in Section 4.2. Instead of evaluating the forecasts separately for each horizon (as we did before), we now jointly evaluate the forecast performance over the four forecast horizons based on the four-variate simulated sample.

```
R> es_sample(obs, dat = X)
R> vs_sample(obs, dat = X)

[1] 4.14
[1] 7.04
```

[4] In `vs_sample()` it is possible to specify a $d \times d$ matrix `w` of non-negative weights that allow to emphasize or down-weight pairs of component combinations based on subjective expert decisions. The entry in the i-th row and j-th column of `w` corresponds to the weight assigned to the combination of the i-th and j-th component. If no weights are specified, constant weights with $w_{i,j} = 1$ for all $i, j \in \{1, \ldots, d\}$ are used. For details and examples on choosing appropriate weights, see Scheuerer and Hamill (2015b).

While this simple example refers to a single forecast case and a single model, a typical empirical analysis would consider the average scores (across several forecast cases) of two or more models.

6. Summary and discussion

The **scoringRules** package enables computing of proper scoring rules for parametric and simulated forecast distributions. The package covers a wide range of situations prevalent in work on modeling and forecasting, and provides generally applicable and numerically efficient implementations based on recent theoretical considerations.

The main functions of the package – `crps()` and `logs()` – are S3 generics, for which we provide methods `crps.numeric()` and `logs.numeric()`. This allows for natural extensions by defining S3 methods for classes other than '`numeric`'. For example, consider a fitted model object of class '`crch`', obtained by the R package of the same name (Messner *et al.* 2016). An object of this class contains a detailed specification of the fitted model's forecast distribution (such as the parametric family of distributions and the values of the fitted parameters). This could be utilized to write a specific method that computes the CRPS of a fitted model object.

Apart from comparative evaluation, proper scoring rules also provide valuable tools for parameter estimation. In the general optimum score estimation framework of Gneiting and Raftery (2007), the parameters of a model's forecast distribution are determined by optimizing the average value of a proper scoring rule as a function of the parameters within a training set. Optimum score estimation based on the LogS corresponds to classical maximum likelihood estimation. The score functions to compute CRPS and LogS for parametric forecast distributions included in **scoringRules** (see Table 1), specifically the underlying worker functions, thus offer tools for the straightforward implementation of such optimum score estimation approaches. Functions to compute gradients and Hessian matrices of the CRPS can be leveraged by numerical optimization procedures such as `optim()`, and have been implemented for a subset of the parametric families covered by the package.

The choice of an appropriate proper scoring rule for model evaluation or parameter estimation is a non-trivial task. We have implemented the widely used LogS and CRPS along with the multivariate ES and VS. Possible future extension of the **scoringRules** package include the addition of novel proper scoring rules such as the Dawid-Sebastiani score (**?**) which has been partially implemented. Further, given the availability of appropriate analytical expressions, the list of covered parametric families can be extended as demand arises and time allows.

Acknowledgments

The work of Alexander Jordan and Fabian Krüger has been funded by the European Union Seventh Framework Programme under grant agreement 290976. Sebastian Lerch gratefully acknowledges support by Deutsche Forschungsgemeinschaft (DFG) through project C7 ("Statistical postprocessing and stochastic physics for ensemble predictions") within SFB/TRR 165 "Waves to Weather". The authors thank the Klaus Tschira Foundation for infrastructural support at the Heidelberg Institute for Theoretical Studies. Helpful comments by Tilmann Gneiting, Stephan Hemri, Jakob Messner and Achim Zeileis are gratefully acknowledged. We further thank Maximiliane Graeter for contributions to the implementation of the multivariate

scoring rules.

References

Ardia D, Hoogerheide LF (2010). "Bayesian Estimation of the GARCH(1, 1) Model with Student-t Innovations." *The R Journal*, **2**(2), 41–47.

Brown JD, Demargne J, Seo DJ, Liu Y (2010). "The Ensemble Verification System (EVS): A Software Tool for Verifying Ensemble Forecasts of Hydrometeorological and Hydrologic Variables at Discrete Locations." *Environmental Modelling & Software*, **25**(7), 854–872. doi:10.1016/j.envsoft.2010.01.009.

Carpenter B, Gelman A, Hoffman M, Lee D, Goodrich B, Betancourt M, Brubaker M, Guo J, Li P, Riddell A (2017). "Stan: A Probabilistic Programming Language." *Journal of Statistical Software*, **76**(1), 1–37. doi:10.18637/jss.v076.i01.

Clements MP (2004). "Evaluating the Bank of England Density Forecasts of Inflation." *The Economic Journal*, **114**(498), 844–866. doi:10.1111/j.1468-0297.2004.00246.x.

Developmental Testbed Center (2017). *Model Evaluation Tools Version 6.0 (METv6.0) User's Guide*. URL http://www.dtcenter.org/met/users/docs/users_guide/MET_Users_Guide_v6.0.pdf.

Fraley C, Raftery AE, Gneiting T, Sloughter JM, Berrocal VJ (2011). "Probabilistic Weather Forecasting in R." *The R Journal*, **3**(1), 55–63.

Fraley C, Raftery AE, Sloughter JM, Gneiting T (2015). ***ensembleBMA**: Probabilistic Forecasting using Ensembles and Bayesian Model Averaging*. R package version 5.1.1, URL http://CRAN.R-project.org/package=ensembleBMA.

Frühwirth-Schnatter S (2006). *Finite Mixture and Markov Switching Models*. Springer-Verlag, New York.

Gneiting T, Katzfuss M (2014). "Probabilistic Forecasting." *Annual Review of Statistics and Its Application*, **1**, 125–151. doi:10.1146/annurev-statistics-062713-085831.

Gneiting T, Raftery AE (2005). "Weather Forecasting with Ensemble Methods." *Science*, **310**(5746), 248–249. doi:10.1126/science.1115255.

Gneiting T, Raftery AE (2007). "Strictly Proper Scoring Rules, Prediction, and Estimation." *Journal of the American Statistical Association*, **102**(477), 359–378. doi:10.1198/016214506000001437.

Gneiting T, Raftery AE, Westveld III AH, Goldman T (2005). "Calibrated Probabilistic Forecasting Using Ensemble Model Output Statistics and Minimum CRPS Estimation." *Monthly Weather Review*, **133**(5), 1098–1118. doi:10.1175/MWR2904.1.

Gneiting T, Stanberry LI, Grimit EP, Held L, Johnson NA (2008). "Assessing Probabilistic Forecasts of Multivariate Quantities, with an Application to Ensemble Predictions of Surface Winds." *Test*, **17**, 211–235. doi:doi.org/10.1007/s11749-008-0114-x.

Good IJ (1952). "Rational Decisions." *Journal of the Royal Statistical Society B*, **14**(1), 107–114.

Grimit EP, Gneiting T, Berrocal VJ, Johnson NA (2006). "The Continuous Ranked Probability Score for Circular Variables and its Application to Mesoscale Forecast Ensemble Verification." *Quarterly Journal of the Royal Meteorological Society*, **132**(621C), 2925–2942. doi:10.1256/qj.05.235.

Hamilton JD (1989). "A New Approach to the Economic Analysis of Nonstationary Time Series and the Business Cycle." *Econometrica*, **57**(2), 357–384. doi:10.2307/1912559.

Hyndman RJ, Khandakar Y (2008). "Automatic Time Series Forecasting: The **forecast** Package for R." *Journal of Statistical Software*, **26**(3), 1–22. doi:10.18637/jss.v027.i03.

Jordan A (2016). "Facets of Forecast Evaluation." doi:10.5445/IR/1000063629. Ph.D. thesis, Karlsruhe Institute of Technology, available at https://publikationen.bibliothek.kit.edu/1000063629.

Jordan A, Krüger F, Lerch S (2017). ***scoringRules**: Scoring Rules for Parametric and Simulated Distribution Forecasts*. R package version 0.9.3, URL https://CRAN.R-project.org/package=scoringRules.

Kastner G (2016). "Dealing with Stochastic Volatility in Time Series Using the R Package **stochvol**." *Journal of Statistical Software*, **69**(5), 1–30. doi:10.18637/jss.v069.i05.

Krüger F, Lerch S, Thorarinsdottir TL, Gneiting T (2016). "Probabilistic Forecasting and Comparative Model Assessment Based on Markov Chain Monte Carlo Output." Working paper. Preprint available at https://arxiv.org/abs/1608.06802.

Kuhn M, Wing J, Weston S, Williams A, Keefer C, Engelhardt A, Cooper T, Mayer Z, Kenkel B, the R Core Team, Benesty M, Lescarbeau R, Ziem A, Scrucca L, Tang Y, Candan C, Hunt T (2017). ***caret**: Classification and Regression Training*. R package version 6.0-77, URL https://CRAN.R-project.org/package=caret.

Laio F, Tamea S (2007). "Verification Tools for Probabilistic Forecasts of Continuous Hydrological Variables." *Hydrology and Earth System Sciences Discussions*, **11**(4), 1267–1277. doi:10.5194/hess-11-1267-2007.

Leutbecher M, Palmer TN (2008). "Ensemble Forecasting." *Journal of Computational Physics*, **227**(7), 3515–3539. doi:10.1016/j.jcp.2007.02.014.

Matheson JE, Winkler RL (1976). "Scoring Rules for Continuous Probability Distributions." *Management Science*, **22**(10), 1087–1096. doi:10.1287/mnsc.22.10.1087.

Messner JW, Mayr GJ, Wilks DS, Zeileis A (2014). "Extending Extended Logistic Regression: Extended Versus Separate Versus Ordered Versus Censored." *Monthly Weather Review*, **142**(8), 3003–3014. doi:10.1175/MWR-D-13-00355.1.

Messner JW, Mayr GJ, Zeileis A (2016). "Heteroscedastic Censored and Truncated Regression with **crch**." *The R Journal*, **17**(11), 173–181.

Meyer S, Held L, Höhle M (2017). "Spatio-Temporal Analysis of Epidemic Phenomena Using the R Package **surveillance**." *Journal of Statistical Software*, **77**(11), 1–55. doi:10.18637/jss.v077.i11.

Murphy AH (1970). "The Ranked Probability Score and the Probability Score: A Comparison." *Monthly Weather Review*, **98**(12), 917–924.

National Center for Atmospheric Research (2015). ***verification**: Weather Forecast Verification Utilities*. R package version 1.42, URL http://CRAN.R-project.org/package=verification.

Pocernich M (2012). "Verification Software." In IT Jolliffe, DB Stephenson (eds.), *Forecast Verification: A Practitioner's Guide in Atmospheric Science*, 2 edition, pp. 231–240. John Wiley & Sons.

Raftery AE (2016). "Use and Communication of Probabilistic Forecasts." *Statistical Analysis and Data Mining: The ASA Data Science Journal*, **9**(6), 397–410. doi:10.1002/sam.11302.

Raftery AE, Alkema L, Gerland P (2014). "Bayesian Population Projections for the United Nations." *Statistical Science*, **29**(1), 58–68. doi:10.1214/13-STS419.

Raftery AE, Gneiting T, Balabdaoui F, Polakowski M (2005). "Using Bayesian Model Averaging to Calibrate Forecast Ensembles." *Monthly Weather Review*, **133**(5), 1155–1174. doi:10.1175/MWR2906.1.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Scheuerer M (2014). "Probabilistic Quantitative Precipitation Forecasting Using Ensemble Model Output Statistics." *Quarterly Journal of the Royal Meteorological Society*, **140**(680), 1086–1096. doi:10.1002/qj.2183.

Scheuerer M, Hamill TM (2015a). "Statistical Postprocessing of Ensemble Precipitation Forecasts by Fitting Censored, Shifted Gamma Distributions." *Monthly Weather Review*, **143**(11), 4578–4596. doi:10.1175/MWR-D-15-0061.1.

Scheuerer M, Hamill TM (2015b). "Variogram-based Proper Scoring Rules for Probabilistic Forecasts of Multivariate Quantities." *Monthly Weather Review*, **143**(4), 1321–1334. doi:10.1175/MWR-D-14-00269.1.

Siegert S (2015). ***SpecsVerification**: Forecast Verification Routines for the SPECS FP7 Project*. R package version 0.4-1, URL http://CRAN.R-project.org/package=SpecsVerification.

Silverman BW (1986). *Density Estimation for Statistics and Data Analysis*. CRC press, Boca Raton.

The Climate Corporation (2015). ***properscoring**: Proper Scoring Rules in Python*. Python package version 0.1, URL https://pypi.python.org/pypi/properscoring.

Vehtari A, Gelman A, Gabry J (2016). ***loo**: Efficient Leave-one-out Cross-validation and WAIC for Bayesian Models*. R package version 1.1.0, URL https://CRAN.R-project.org/package=loo.

Yuen RA, Baran S, Fraley C, Gneiting T, Lerch S, Scheuerer M, Thorarinsdottir TL (2017). ***ensembleMOS**: Ensemble Model Output Statistics*. R package version 0.8, URL http://CRAN.R-project.org/package=ensembleMOS.

Affiliation:

Alexander Jordan
Heidelberg Institute for Theoretical Studies
HITS gGmbH
Schloss-Wolfsbrunnenweg 35
69118 Heidelberg, Germany
E-Mail: alexander.jordan@h-its.org

Fabian Krüger
Alfred-Weber-Institute for Economics
Heidelberg University
Bergheimer Str. 58
69115 Heidelberg, Germany
E-Mail: fabian.krueger@awi.uni-heidelberg.de
URL: https://sites.google.com/site/fk83research/home

Sebastian Lerch
Heidelberg Institute for Theoretical Studies
HITS gGmbH
Schloss-Wolfsbrunnenweg 35
69118 Heidelberg, Germany
E-Mail: sebastian.lerch@h-its.org
URL: https://sites.google.com/site/sebastianlerch/
and
Institute for Stochastics
Karlsruhe Institute of Technology