Journal of Statistical Software

August 2019, Volume 90, Issue 12. doi: 10.18637/jss.v090.i12

Evaluating Probabilistic Forecasts with scoringRules

Alexander Jordan
University of Bern

Fabian Krüger
Heidelberg University

Sebastian Lerch
Karlsruhe Institute of Technology
Heidelberg Institute for
Theoretical Studies

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 computation, 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, Alkema, and Gerland 2014), inflation projections issued by the Bank of England (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 func-

tions $S(F, y)$ 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 2019) software package **scoringRules** (Jordan, Krüger, and Lerch 2019), 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 of 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. Here, the focus typically lies in diagnostics, e.g., whether the predictive distributions match the observations statistically (see probability integral transform histogram, e.g., in Gneiting and Katzfuss 2014). To retain focus, the **scoringRules** package does not cover absolute forecast evaluation.

Comparative forecast evaluation shares key aspects with out-of-sample model comparison. 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 2019) package provides the traditional Akaike and Bayes information criteria to select among linear models in in-sample evaluation. The packages **caret** (Kuhn 2008) and **forecast** (Hyndman and Khandakar 2008) provide cross-validation tools suitable for cross-sectional and time series data, respectively. The **loo** (Vehtari, Gabry, Yao, and Gelman 2019) package implements recent proposals to select among Bayesian models. In contrast to existing software, a key novelty of the **scoringRules** package is its extensive coverage of the CRPS. This scoring rule is attractive for both practical and theoretical reasons (Gneiting and Raftery 2007; Krüger, Lerch, Thorarinsdottir, and Gneiting 2016), yet more widespread use has been hampered by computational challenges. In providing analytical formulas and efficient numerical implementations, we hope to enable convenient use 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 for many types of forecast distributions. However, a number

of existing R packages include scoring rule computations for more specific empirical situations: The **ensembleBMA** (Fraley *et al.* 2018) and **ensembleMOS** (Yuen *et al.* 2018) 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, Raftery, Gneiting, Sloughter, and Berrocal 2011), and can only be applied to specific data structures utilized in the packages. The **surveillance** (Meyer, Held, and Höhle 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. The **scoring** (Merkle and Steyvers 2013) package focuses on discrete (categorical) outcomes, for which it offers a large number of proper scoring rules. It is thus complementary to **scoringRules** which supports a wide range of forecast distributions while focusing on a smaller number of scoring rules. Furthermore, the **verification** (NCAR Research Applications Laboratory 2015) and **SpecsVerification** (Siegert 2017) packages contain implementations of the CRPS for simulated forecast distributions. Our contribution in that domain is twofold: First, we offer efficient implementations to deal with predictive distributions given as large samples. MCMC methods are popular across the disciplines, and many sophisticated R implementations are available (e.g., Kastner 2016; Carpenter *et al.* 2017). Second, we include various implementation options, and propose principled default settings based on 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 score computation functionality than the **scoringRules** package. The **properscoring** (The Climate Corporation 2015) package for the Python (Python Software Foundation 2017) language 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 software (**MET**, Developmental Testbed Center 2018) 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 (**EVS**, Brown, Demargne, Seo, and Liu 2010) 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 computation 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 compare ensemble system output from numerical weather prediction models to 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. This paper is used as the basis for a package vignette that is kept up to date should corrections or extensions be necessary.

2. Theoretical background

Probabilistic forecasts usually fit one of two categories, parametric distributions or simulated samples. The former type is represented by its cumulative distribution function (CDF) or its probability density function (PDF), whereas the latter is often used if the predictive distribution is not available analytically. Here, we give a brief overview of the theoretical background for comparative forecast evaluation in both cases.

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 $F = G$. Using a proper scoring rule is critical for comparative evaluation, i.e., the ranking of forecasts. In practice, the lowest average score over multiple forecast cases among competing forecasters indicates the best predictive performance, and in this setup, proper scoring rules compel forecasters to truthfully report what they think is the true distribution. See Gneiting and Raftery (2007) for a more detailed review of the mathematical properties.

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 one if $y \leq z$ and zero otherwise. If the first moment of F is finite, the CRPS can be written as

$$\mathrm{CRPS}(F, y) = \mathsf{E}_F|X_1 - y| - \frac{1}{2}\mathsf{E}_{F,F}|X_1 - X_2|,$$

where X_1 and X_2 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

Distribution	Family	CRPS	LogS	Additional parameters
Distributions for variables on the real line				
Laplace	`"lapl"`	✓	✓	
Logistic	`"logis"`	✓	✓	
Normal	`"norm"`	✓	✓	
Mixture of normals	`"mixnorm"`	✓	✓	
Student's t	`"t"`	✓	✓	
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"`	✓	✓	limits
Uniform	`"unif"`	✓	✓	limits, point masses
Exponential	`"exp2"`		✓	location, scale
	`"expM"`	✓		location, scale, point mass
Gen. extreme value	`"gev"`	✓	✓	
Gen. Pareto	`"gpd"`	✓	✓	point mass (CRPS only)
Logistic	`"tlogis"`	✓	✓	limits (truncation)
	`"clogis"`	✓		limits (censoring)
	`"gtclogis"`	✓		limits, point masses
Normal	`"tnorm"`	✓	✓	limits (truncation)
	`"cnorm"`	✓		limits (censoring)
	`"gtcnorm"`	✓		limits, point masses
Student's t	`"tt"`	✓	✓	limits (truncation)
	`"ct"`	✓		limits (censoring)
	`"gtct"`	✓		limits, point masses
Distributions for discrete variables				
Binomial	`"binom"`	✓	✓	
Hypergeometric	`"hyper"`	✓	✓	
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. The CRPS formulas are given in Appendix A.

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 Appendix A.

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 $X_1, \dots, X_m \sim F$. Examples include Bayesian 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 an estimated distribution (say, $\hat{F}_m(z)$) that can be evaluated at any point $z \in \mathbb{R}$. 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 one to compute the CRPS directly from the simulated sample, see Grimit, Gneiting, Berrocal, and Johnson (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 $X_{(1)}, \dots, X_{(m)}$, 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 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: If the outcome falls into the tails of the simulated forecast distribution, the estimated score may be highly sensitive to the choice of the bandwidth tuning parameter. In an MCMC context, a mixture-of-parameters estimator that utilizes a simulated sample of parameter values – 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, in that one needs to know the analytic form of

the forecast distribution conditional on the parameter draws. If such knowledge is available, the mixture-of-parameters estimator can be implemented using functionality for parametric forecast distributions. We provide an example in Section 4.2. This example features a conditionally Gaussian forecast distribution, a typical case in applications.

3. Package design and functionality

The main functionality of **scoringRules** is the computation of scores. The essential functions for this purpose follow the naming convention `[score]_[suffix]()`, where the two maturest choices for `[score]` are `crps` and `logs`. Regarding the `[suffix]`, we aim for analogy to the usual `d/p/q/r` functions for parametric families of distributions, both in terms of naming convention and usage details, e.g.,

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

Based on these computation functions, package developers may write S3 methods that hook into the respective S3 generic functions, currently limited to

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

As the implementation of additional scoring rules matures, this list will be extended. We reserve methods for the class '`numeric`', e.g.,

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

which are pedantic wrappers for the corresponding `[score]_[family]()` functions, but with meaningful error messages, making the '`numeric`' class methods more suitable for interactive use as opposed to numerical score optimization or package integration. Table 1 gives a list of the implemented parametric families, as does the '`numeric`' class method documentation for the respective score, e.g., see `?crps.numeric`.

Echoing the distinction in Section 2 between parametric and empirical predictive distributions, we note that computation functions following the naming scheme `[score]_sample()`, see Sections 3.2 and 5, have a special status and cannot be called via the '`numeric`' class method.

3.1. Parametric predictive distributions

When the predictive distribution comes from a parametric family, we have two options to perform the score computation and get the resulting vector of score values, i.e., via the score generics or the computation function. For a Gaussian distribution:

```
R> library("scoringRules")
R> obs <- rnorm(10)
R> crps(obs, "norm", 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> crps_norm(obs, 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
```

The results are identical, except when the much stricter input checks of the '`numeric`' class method are violated, e.g.,

```
R> crps(obs, "norm", mean = c(1:10), sd = c(1:9, -5))

Error in checkInput(input): Parameter 'sd' contains non-positive values.

R> crps_norm(obs, mean = c(1:10), sd = c(1:9, -5))

Warning in pnorm(y, sd = scale): NaNs produced

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

This helps in detecting manual errors during interactive use, or in debugging automated model fitting and evaluation. Other restrictions in the '`numeric`' class method include the necessity to pass input to all arguments, i.e., default values in the computation functions are ignored, and that all numerical arguments should be vectors of the same length, with the exception that vectors of length one will be recycled. In contrast, the computation functions aim to closely obey standard R conventions.

In package development, we expect predominant use of the computation functions, where developers will handle regularity checks themselves. As a trivial example, we define functions that only depend on the argument `y` and compute scores for a fixed predictive gamma distribution:

```
R> crps_y <- function(y) crps_gamma(y, shape = 2, scale = 1.5)
R> logs_y <- function(y) logs_gamma(y, shape = 2, scale = 1.5)
```

In Figure 1 these functions are used 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 zero, 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 zero.

3.2. Simulated predictive distributions

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)
```

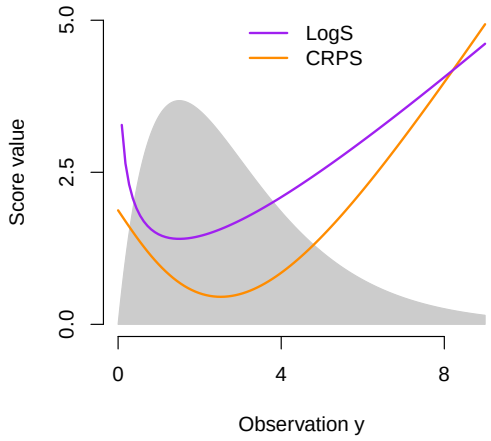

Figure 1: Score values for a gamma distribution with `shape = 2` and `scale = 1.5`, dependent on the observation (functions `crps_y()` and `logs_y()` defined in the text). A scaled version of the predictive density is shown in gray.

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., for a Gaussian predictive distribution:

```
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)

[1] 2.29 2.10 2.04
```

When `y` has length one 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, where we examine the performance of forecasts given as samples of size up to 5 000. The approximation experiment is independently repeated 500 times:

```
R> R <- 500
R> M <- 5e3
R> mgrid <- exp(seq(log(50), log(M), length.out = 51))
R> crps_approx <- matrix(NA, nrow = R, ncol = length(mgrid))
R> logs_approx <- matrix(NA, nrow = R, ncol = length(mgrid))
R> obs_1 <- 2
R> for (r in 1:R) {
+    sample_M <- rnorm(M, mean = 2, sd = 3)
```

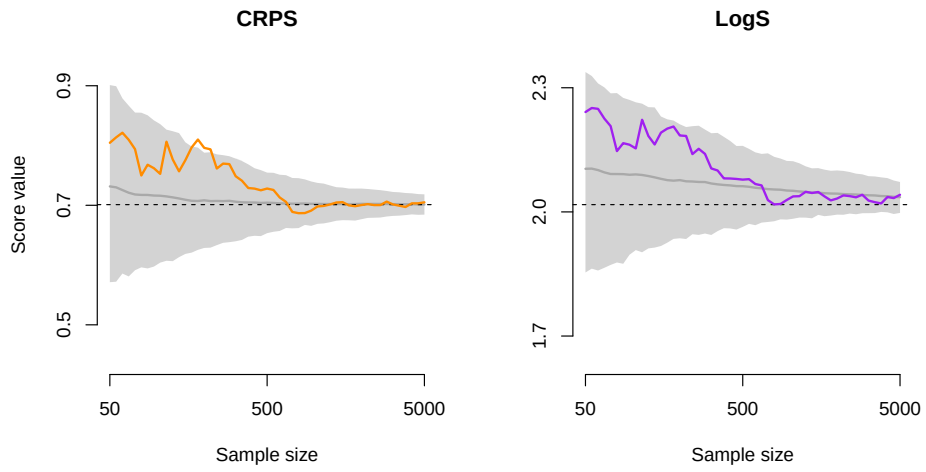

Figure 2: The scores of a Gaussian forecast distribution with mean 2 and standard deviation 3 when a value of 0 is observed, estimated from an independent sample from the predictive distribution, and shown as a function of the size of the (expanding) sample. The horizontal dashed line represents the analytically calculated score. The gray area indicates empirical 90% confidence intervals for each sample size computed from 500 independent replications of the simulation experiment, and the gray line shows the corresponding mean value over all repetitions.

```
+    for (i in seq_along(mgrid)) {
+      m <- mgrid[i]
+      crps_approx[r, i] <- crps_sample(obs_1, dat = sample_M[1:m])
+      logs_approx[r, i] <- logs_sample(obs_1, dat = sample_M[1:m])
+    }
+  }
```

In this case, the true CRPS and LogS values can be computed 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 compute 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 distribution. 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. In particular, the empirical distribution function is not applicable to LogS because an estimated density is required. The `bw` argument allows one to manually select a bandwidth parameter for kernel density estimation; by default, the `bw.nrd()` function from the **stats** (R Core Team 2019) 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, Raftery, Westveld III, and Goldman 2005) and Bayesian model averaging (Raftery, Gneiting, Balabdaoui, and Polakowski 2005).

Here we illustrate how to evaluate post-processed ensemble forecasts of precipitation, based on data and methods from the **crch** (Messner, Mayr, and Zeileis 2016) package for R. 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 a 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, Mayr, Wilks, and Zeileis (2014); Scheuerer (2014) and Scheuerer and Hamill (2015a).

The **crch** package contains a data set `RainIbk` of precipitation for Innsbruck, Austria. It comprises ensemble forecasts `rainfc.1`, ..., `rainfc.11` and observations `rain` for 4971 cases from January 2000 to September 2013. The precipitation amounts are accumulated over

three days, and the corresponding 11 member ensemble consists of accumulated precipitation amount forecasts between five and eight 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", package = "crch")
R> RainIbk <- sqrt(RainIbk)
R> ensfc <- RainIbk[, grep('^rainfc', names(RainIbk))]
R> RainIbk$ensmean <- apply(ensfc, 1, mean)
R> RainIbk$enssd <- apply(ensfc, 1, sd)
R> RainIbk <- subset(RainIbk, enssd > 0)
```

We split the data into a training set until November 2004, and an out-of-sample evaluation period (or test sample) from January 2005.

```
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))
```

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.

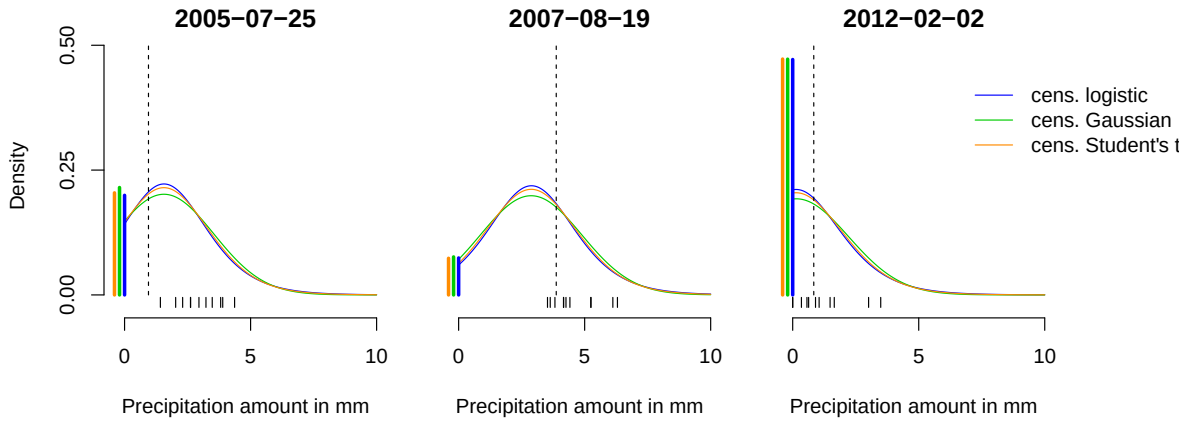

Figure 3: Illustration of the forecast distributions of the censored regression models for three illustrative 3-day accumulation periods (plot title indicates end of period). The predicted probabilities of zero precipitation are shown as solid thick vertical lines at 0, and the colored thin lines indicate 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.

```
R> scores <- data.frame(CRCHlogis = logis_crps, CRCHgauss = gauss_crps,
+    CRCHstud = stud_crps, Ensemble = ens_crps)
R> sapply(scores, mean)

CRCHlogis CRCHgauss  CRCHstud  Ensemble
    0.875     0.876     0.875     1.321
```

4.2. Bayesian forecasts of US GDP growth rate

We next present a representative example from economics, where the predictive distribution is given as a simulated sample. Hamilton (1989) first proposed the Markov switching autoregressive model to allow regime-dependent modeling, i.e., to capture different recurring time-series characteristics. We consider the following simple variant of the model:

$$\begin{aligned} Y_t &= c_0 + c_1\, Y_{t-1} + \varepsilon_t, \\ \varepsilon_t &\sim \mathcal{N}(0, \sigma^2_{s_t}), \end{aligned}$$

where Y_t is the observed GDP growth rate of quarter t, and $s_t \in \{1, 2\}$ is a latent state variable that represents two regimes in the residual variance $\sigma^2_{s_t}$. Since s_t evolves according to a first-order Markov chain, the specification allows for the possibility that periods of high (or low) volatility cluster over time. The latter is a salient feature of the US GDP growth rate: 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 (e.g., Frühwirth-Schnatter 2006). Our implementation closely follows Krüger *et al.* (2016, Section 5), and uses the `ar_ms()` function, and the data set `gdp`, included in the **scoringRules** package.

The data set `gdp` comprises US GDP growth observations `val`, and the corresponding quarters `dt` and vintages `vint`. Measuring economic variables is challenging, hence records tend to be

revised and each quarter has its own time-series, or vintage, of past observations. As a result, the data set for 271 quarters from 1947 to 2014 contains 33616 records.

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, using the most recent vintage in both cases:

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

This function call 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)
```

We consider two competing estimators of the posterior predictive distribution F_0. The mixture-of-parameters estimator (MPE)

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

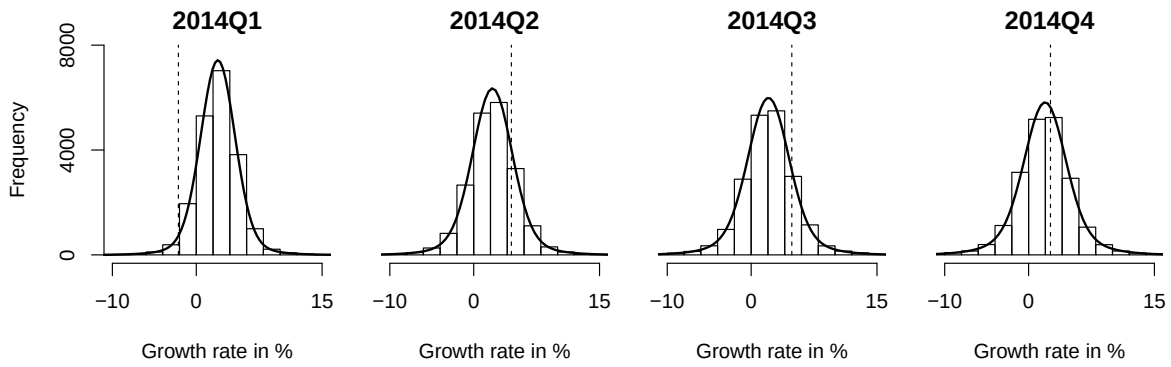

Figure 4: Forecast distributions for the growth rate of US GDP. The forecasts stem from a Bayesian time series model, as detailed in Krüger *et al.* (2016, Section 5). Histograms summarize simulated forecast draws at each date. Mixture-of-normals approximation to distribution shown in black; realizing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

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 indicates the mixture-of-parameters estimator. 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 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    3.457     3.468     4.01     3.97
2014Q2    1.358     1.362     2.29     2.28
```

```
2014Q3    1.700     1.690     2.52      2.54
2014Q4    0.724     0.729     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).

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).

4.3. Parameter estimation with scoring rules

Apart from comparative forecast evaluation, proper scoring rules also provide useful tools for parameter estimation. In the optimum score estimation framework of Gneiting and Raftery (2007, Section 9.1), the parameters of a model's forecast distribution are determined by optimizing the value of a proper scoring rule, on average over a training sample. 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) thus offer tools for the straightforward implementation of such optimum score estimation approaches. Specifically, the computation functions of the form `[score]_[family]()` entail little overhead in terms of input checks and are thus well suited for use in numerical optimization procedures such as `optim()`. Furthermore, functions to compute gradients and Hessian matrices of the CRPS have been implemented for a subset of parametric families, and can be supplied to assist numerical optimizers. Such functions are available for the `"logis"`, `"norm"` and `"t"` families and truncated and censored versions thereof (`"clogis"`, `"tlogis"`, `"cnorm"`, `"tnorm"`, `"ct"`, `"tt"`). The corresponding computation functions follow the naming scheme `gradcrps_[family]()` and `hesscrps_[family]()`. However, we emphasize that implementing minimum CRPS or LogS estimation approaches is possible for all parametric families listed in Table 1, even if analytical gradient and Hessian functions are not supplied but are instead approximated numerically by `optim()`.

Gebetsberger, Messner, Mayr, and Zeileis (2018) provide a detailed comparison of maximum likelihood and minimum CRPS estimation in the context of non-homogeneous regression models for post-processing ensemble weather forecasts. Here we illustrate the use for minimum CRPS estimation in a simple simulation example. Consider an independent sample $y_1, \ldots, y_n$ from a normal distribution with mean μ and standard deviation σ. The analytical maximum likelihood estimates $\hat{\mu}_{\mathrm{ML}}$ and $\hat{\sigma}_{\mathrm{ML}}$ are given by

$$\hat{\mu}_{\mathrm{ML}} = \frac{1}{n}\sum_{i=1}^{n} y_i \quad \text{and} \quad \hat{\sigma}_{\mathrm{ML}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\mu}_{\mathrm{ML}})^2}.$$

To determine the corresponding estimates by numerically minimizing the CRPS define wrapper functions which compute the mean CRPS and corresponding gradient for a vector of training data `y_train` and distribution parameters `param`.

```
R> meancrps <- function(y_train, param) mean(crps_norm(y = y_train,
+    mean = param[1], sd = param[2]))
```

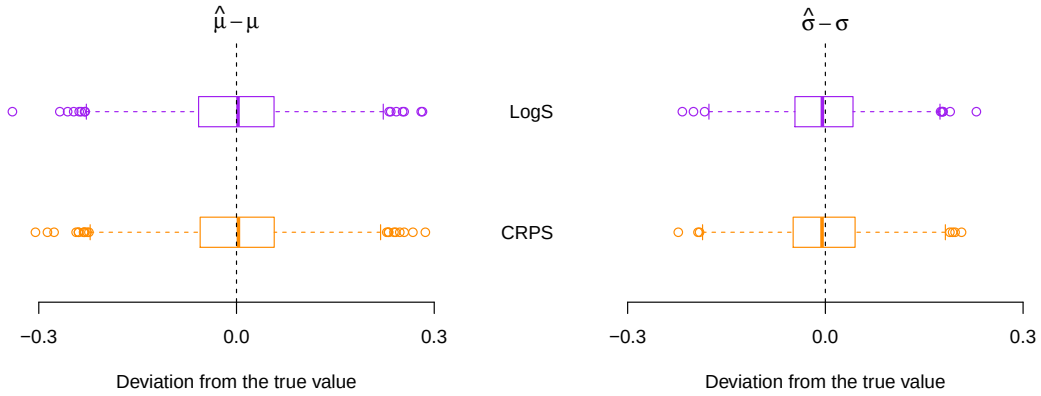

Figure 5: Boxplots of deviations from the true parameter values for estimates obtained via minimum CRPS and minimum LogS (i.e., maximum likelihood) estimation based on 1 000 independent samples of size 500 from a normal distribution with mean $\mu = -1$ and standard deviation $\sigma = 2$.

```
R> grad_meancrps <- function(y_train, param) apply(gradcrps_norm(y_train,
+    param[1], param[2]), 2, mean)
```

These functions can then be passed to `optim()`, for example, mean and standard deviation of a normal distribution with true values -1 and 2 can be estimated as illustrated in the following. The estimation with sample size 500 is repeated 1 000 times.

```
R> R <- 1000
R> n <- 500
R> mu_true <- -1
R> sigma_true <- 2
R> estimates_ml <- matrix(NA, nrow = R, ncol = 2)
R> estimates_crps <- matrix(NA, nrow = R, ncol = 2)
R> for (r in 1:R) {
+    dat <- rnorm(n, mu_true, sigma_true)
+    estimates_crps[r, ] <- optim(par = c(1, 1), fn = meancrps,
+      gr = grad_meancrps, method = "BFGS", y_train = dat)$par
+    estimates_ml[r, ] <- c(mean(dat), sd(dat) * sqrt((n - 1) / n))
+  }
```

Figure 5 compares minimum CRPS and minimum LogS (i.e., maximum likelihood) parameter estimates. The differences to the true values show very similar distributions and illustrate the consistency of general optimum score estimates (Gneiting and Raftery 2007, Equation 59). For the standard deviation parameter σ, the difference between estimate and true value exhibits slightly less variability for the maximum likelihood method.

5. Multivariate scoring rules

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. For example, the univariate LogS allows for a straightforward generalization towards multivariate forecast distributions. However, parametric modeling and forecasting of multivariate observations is challenging, and when sampling is a feasible alternative we encounter the same, even exacerbated, problems in kernel density estimation as for univariate samples. As another example, the univariate CRPS can also be generalized to multivariate forecast distributions, and one such generalization is discussed in this chapter, the energy score. Finding closed form expressions for parametric distributions is even more involved than for the univariate CRPS, but the robustness in the evaluation of sample forecasts is retained. We refer to Gneiting, Stanberry, Grimit, Held, and Johnson (2008) and Scheuerer and Hamill (2015b) for a detailed discussion of multivariate proper scoring rules and limit our attention to the case where probabilistic forecasts are given as samples from the forecast distributions.

Let $\mathbf{y} = (y^{(1)}, \dots, y^{(d)}) \in \Omega = \mathbb{R}^d$, and let F denote a forecast distribution on $\mathbb{R}^d$ given through m discrete samples $\mathbf{X}_1, \dots, \mathbf{X}_m$ from F with $\mathbf{X}_i = (X_i^{(1)}, \dots, X_i^{(d)}) \in \mathbb{R}^d, i = 1, \dots, m$. 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 one 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)
```

These functions can only evaluate a single multivariate forecast case and always return a single number to simplify use and documentation, see Appendix B for an example on how to use `apply()` functions or `for` loops to sequentially apply them to multiple forecast cases. The observation input for `y` is required to be a vector of length d, and the corresponding forecast input for `dat` has to be given as a $d \times m$ matrix, the columns of which are the simulated samples $\mathbf{X}_1, \dots, \mathbf{X}_m$ from the multivariate forecast distribution. In `vs_sample()` it is possible to specify a $d \times d$ matrix for `w` of non-negative weights as described in the text. 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 $i, j \in \{1, \dots, d\}$ are used. For details and examples on choosing appropriate weights, see Scheuerer and Hamill (2015b).

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> names(obs) <- NULL
R> es_sample(obs, dat = X)

[1] 4.13

R> vs_sample(obs, dat = X)

[1] 7.05
```

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 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()`. The package can be extended naturally 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 information could be utilized to write a specific method that computes the CRPS of a fitted model object.

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^p. Possible future extension of the **scoringRules** package include the addition of novel proper scoring rules such as the Dawid-Sebastiani score (Dawid and Sebastiani 1999) 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, and two referees for constructive comments on an earlier version of the manuscript.

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. doi:10.32614/rj-2010-014.

Baran S, Lerch S (2015). "Log-Normal Distribution Based Ensemble Model Output Statistics Models for Probabilistic Wind-Speed Forecasting." *Quarterly Journal of the Royal Meteorological Society*, **141**(691), 2289–2299. doi:10.1002/qj.2521.

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 MD, 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), 1–32. 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.

Dawid AP, Sebastiani P (1999). "Coherent Dispersion Criteria for Optimal Experimental Design." *The Annals of Statistics*, **27**(1), 65–81.

Developmental Testbed Center (2018). ***MET**: Version 7.0 Model Evaluation Tools Users Guide*. Available at http://www.dtcenter.org/met/users/docs/overview.php.

Fraley C, Raftery AE, Gneiting T, Sloughter JM, Berrocal VJ (2011). "Probabilistic Weather Forecasting in R." *The R Journal*, **3**(1), 55–63. doi:10.32614/rj-2011-009.

Fraley C, Raftery AE, Sloughter JM, Gneiting T, University of Washington (2018). ***ensembleBMA**: Probabilistic Forecasting Using Ensembles and Bayesian Model Averaging*. R package version 5.1.5, URL https://CRAN.R-project.org/package=ensembleBMA.

Friederichs P, Thorarinsdottir TL (2012). "Forecast Verification for Extreme Value Distributions with an Application to Probabilistic Peak Wind Prediction." *Environmetrics*, **23**(7), 579–594. doi:10.1002/env.2176.

Frühwirth-Schnatter S (2006). *Finite Mixture and Markov Switching Models*. Springer-Verlag, New York. doi:10.1007/978-0-387-35768-3.

Gebetsberger M, Messner JW, Mayr GJ, Zeileis A (2018). "Estimation Methods for Nonhomogeneous Regression Models: Minimum Continuous Ranked Probability Score versus Maximum Likelihood." *Monthly Weather Review*, **146**(12), 4323–4338. doi:10.1175/MWR-D-17-0364.1.

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, Larson K, Westrick K, Genton MG, Aldrich E (2006). "Calibrated Probabilistic Forecasting at the Stateline Wind Energy Center." *Journal of the American Statistical Association*, **101**(475), 968–979. doi:10.1198/016214506000000456.

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**(2), 211–235. doi:10.1007/s11749-008-0114-x.

Gneiting T, Thorarinsdottir TL (2010). "Predicting Inflation: Professional Experts Versus No-Change Forecasts." *arXiv 1010.2318*, arXiv.org E-Print Archive. URL http://arxiv.org/abs/1010.2318.

Good IJ (1952). "Rational Decisions." *Journal of the Royal Statistical Society B*, **14**(1), 107–114. doi:10.1111/j.2517-6161.1952.tb00104.x.

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*, **27**(3), 1–22. doi:10.18637/jss.v027.i03.

Jordan A (2016). *Facets of Forecast Evaluation*. Ph.D. thesis, Karlsruhe Institute of Technology. URL https://publikationen.bibliothek.kit.edu/1000063629.

Jordan A, Krüger F, Lerch S (2019). ***scoringRules**: Scoring Rules for Parametric and Simulated Distribution Forecasts*. R package version 1.0.0, 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." *arXiv 1608.06802*, arXiv.org E-Print Archive. URL http://arxiv.org/abs/1608.06802.

Kuhn M (2008). "Building Predictive Models in R Using the **caret** Package." *Journal of Statistical Software*, **28**(5), 1–26. doi:10.18637/jss.v028.i05.

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.

Merkle EC, Steyvers M (2013). "Choosing a Strictly Proper Scoring Rule." *Decision Analysis*, **10**(4), 292–304. doi:10.1287/deca.2013.0280.

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*, **8**(1), 173–181. doi:10.32614/rj-2016-012.

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.

Möller D, Scheuerer M (2015). "Probabilistic Wind Speed Forecasting on a Grid Based on Ensemble Model Output Statistics." *Annals of Applied Statistics*, **9**(3), 1328–1349.

Murphy AH (1970). "The Ranked Probability Score and the Probability Score: A Comparison." *Monthly Weather Review*, **98**(12), 917–924. doi:10.1175/1520-0493(1970)098<0917:trpsat>2.3.co;2.

NCAR Research Applications Laboratory (2015). ***verification**: Weather Forecast Verification Utilities*. R package version 1.42, URL https://CRAN.R-project.org/package=verification.

Pocernich M (2012). "Appendix: Verification Software." In IT Jolliffe, DB Stephenson (eds.), *Forecast Verification: A Practitioner's Guide in Atmospheric Science*, 2nd edition, pp. 231–240. John Wiley & Sons, Chichester. doi:10.1002/9781119960003.app1.

Python Software Foundation (2017). *Python Software, Version 3.6.4*. Beaverton. URL https://www.python.org/.

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 (2019). *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 (2017). ***SpecsVerification**: Forecast Verification Routines for the SPECS FP7 Project*. R package version 0.5-2, URL https://CRAN.R-project.org/package=SpecsVerification.

Silverman BW (1986). *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London.

Taillardat M, Mestre O, Zamo M, Naveau P (2016). "Calibrated Ensemble Forecasts Using Quantile Regression Forests and Ensemble Model Output Statistics." *Monthly Weather Review*, **144**(6), 2375–2393. doi:10.1175/mwr-d-15-0260.1.

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

Thorarinsdottir TL, Gneiting T (2010). "Probabilistic Forecasts of Wind Speed: Ensemble Model Output Statistics by Using Heteroscedastic Censored Regression." *Journal of the Royal Statistical Society A*, **173**(2), 371–388. doi:10.1111/j.1467-985x.2009.00616.x.

Vehtari A, Gabry J, Yao Y, Gelman A (2019). ***loo**: Efficient Leave-One-Out Cross-Validation and WAIC for Bayesian Models*. R package version 2.1.0, URL https://CRAN.R-project.org/package=loo.

Wei W, Held L (2014). "Calibration Tests for Count Data." *TEST*, **23**(4), 787–805. doi:10.1007/s11749-014-0380-8.

Yuen RA, Baran S, Fraley C, Gneiting T, Lerch S, Scheuerer M, Thorarinsdottir TL (2018). ***ensembleMOS**: Ensemble Model Output Statistics*. R package version 0.8.2, URL https://CRAN.R-project.org/package=ensembleMOS.

A. Formulas for the CRPS

A.1. Notation

γ	Euler-Mascheroni constant
$\lfloor x \rfloor$	Floor function
$\text{sgn}(x)$	Sign function
$\text{Ei}(x)$	Exponential integral
$\varphi(x)$	Standard Gaussian density function
$\Phi(x)$	Standard Gaussian distribution function
$\Gamma(a)$	Gamma function
$\Gamma_l(a, x)$	Lower incomplete gamma function
$\Gamma_u(a, x)$	Upper incomplete gamma function
$B(a, b)$	Beta function
$I(a, b, x)$	Regularized incomplete beta function
$I_m(x)$	Modified Bessel function of the first kind
${}_2F_1(a, b; c; x)$	Hypergeometric function

A.2. Distributions for variables on the real line

Laplace distribution

The function `crps_lapl()` computes the CRPS for the standard distribution, and generalizes via `location` parameter $\mu \in \mathbb{R}$ and `scale` parameter $\sigma > 0$,

$$\begin{aligned} \text{CRPS}(F, y) &= |y| + \exp(-|y|) - \frac{3}{4}, \\ \text{CRPS}(F_{\mu,\sigma}, y) &= \sigma \, \text{CRPS}\left(F, \tfrac{y-\mu}{\sigma}\right). \end{aligned}$$

The CDFs are given by $F_{\mu,\sigma}(x) = F\left(\frac{x-\mu}{\sigma}\right)$ and

$$F(x) = \begin{cases} \frac{1}{2}\exp(x), & x < 0, \\ 1 - \frac{1}{2}\exp(-x), & x \geq 0. \end{cases}$$

Logistic distribution

The function `crps_logis()` computes the CRPS for the standard distribution, and generalizes via `location` parameter $\mu \in \mathbb{R}$ and `scale` parameter $\sigma > 0$,

$$\begin{aligned} \text{CRPS}(F, y) &= y - 2\log(F(y)) - 1, \\ \text{CRPS}(F_{\mu,\sigma}, y) &= \sigma \, \text{CRPS}\left(F, \tfrac{y-\mu}{\sigma}\right). \end{aligned}$$

The CDFs are given by $F_{\mu,\sigma}(x) = F\left(\frac{x-\mu}{\sigma}\right)$ and $F(x) = (1 + \exp(-x))^{-1}$.

Normal distribution

The function `crps_norm()` computes the CRPS for the standard distribution, and generalizes

via `mean` parameter $\mu \in \mathbb{R}$ and `sd` parameter $\sigma > 0$, or alternatively, `location` and `scale`,

$$\begin{aligned}\mathrm{CRPS}(\Phi, y) &= y\,(2\Phi(y) - 1) + 2\varphi(y) - \frac{1}{\sqrt{\pi}},\\ \mathrm{CRPS}(F_{\mu,\sigma}, y) &= \sigma\,\mathrm{CRPS}\left(\Phi, \tfrac{y-\mu}{\sigma}\right).\end{aligned}$$

The CDFs are given by Φ and $F_{\mu,\sigma}(x) = \Phi\left(\frac{x-\mu}{\sigma}\right)$. Derived by Gneiting *et al.* (2005).

Mixture of normal distributions

The function `crps_mixnorm()` computes the CRPS for a mixture of normal distributions with mean parameters $\mu_1, \ldots, \mu_M \in \mathbb{R}$ comprising `m`, scale parameters $\sigma_1, \ldots, \sigma_M > 0$ comprising `s`, and (automatically rescaled) weight parameters $\omega_1, \ldots, \omega_M > 0$ comprising `w`,

$$\mathrm{CRPS}(F, y) = \sum_{i=1}^{M} \omega_i A\left(y - \mu_i, \sigma_i^2\right) - \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M} \omega_i \omega_j A\left(\mu_i - \mu_j, \sigma_i^2 + \sigma_j^2\right).$$

The CDF is $F(x) = \sum_{i=1}^{M} \omega_i \Phi\left(\frac{x-\mu_i}{\sigma_i}\right)$, and $A\left(\mu, \sigma^2\right) = \mu\left(2\Phi\left(\frac{\mu}{\sigma}\right) - 1\right) + 2\sigma\varphi\left(\frac{\mu}{\sigma}\right)$. Derived by Grimit *et al.* (2006).

Student's t distribution

The function `crps_t()` computes the CRPS for Student's t distribution with `df` parameter $\nu > 1$, and generalizes via `location` parameter $\mu \in \mathbb{R}$ and `scale` parameter $\sigma > 0$,

$$\begin{aligned}\mathrm{CRPS}(F_\nu, y) &= y\Big(2F_\nu(y) - 1\Big) + 2f_\nu(y)\left(\frac{\nu + y^2}{\nu - 1}\right) - \frac{2\sqrt{\nu}}{\nu - 1}\frac{B(\frac{1}{2}, \nu - \frac{1}{2})}{B(\frac{1}{2}, \frac{\nu}{2})^2},\\ \mathrm{CRPS}(F_{\nu,\mu,\sigma}, y) &= \sigma\,\mathrm{CRPS}\left(F_\nu, \tfrac{y-\mu}{\sigma}\right).\end{aligned}$$

The CDFs and PDF are given by $F_{\nu,\mu,\sigma}(x) = F_\nu\left(\frac{x-\mu}{\sigma}\right)$ and

$$\begin{aligned}F_\nu(x) &= \frac{1}{2} + \frac{x\ {}_2F_1(\frac{1}{2}, \frac{\nu+1}{2}; \frac{3}{2}; -\frac{x^2}{\nu})}{\sqrt{\nu}B(\frac{1}{2}, \frac{\nu}{2})},\\ f_\nu(x) &= \frac{1}{\sqrt{\nu}B(\frac{1}{2}, \frac{\nu}{2})}\left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}.\end{aligned}$$

Two-piece exponential distribution

The function `crps_2pexp()` computes the CRPS for the two-piece exponential distribution with `scale1` and `scale2` parameters $\sigma_1, \sigma_2 > 0$, and generalizes via `location` parameter $\mu \in \mathbb{R}$,

$$\begin{aligned}\mathrm{CRPS}(F_{\sigma_1,\sigma_2}, y) &= \begin{cases} |y| + \frac{2\sigma_1^2}{\sigma_1+\sigma_2}\exp\left(-\left|\frac{y}{\sigma_1}\right|\right) - \frac{2\sigma_1^2}{\sigma_1+\sigma_2} + \frac{\sigma_1^3+\sigma_2^3}{2(\sigma_1+\sigma_2)^2}, & y < 0,\\ |y| + \frac{2\sigma_2^2}{\sigma_1+\sigma_2}\exp\left(-\left|\frac{y}{\sigma_2}\right|\right) - \frac{2\sigma_2^2}{\sigma_1+\sigma_2} + \frac{\sigma_1^3+\sigma_2^3}{2(\sigma_1+\sigma_2)^2}, & y \geq 0,\end{cases}\\ \mathrm{CRPS}(F_{\mu,\sigma_1,\sigma_2}, y) &= \mathrm{CRPS}(F_{\sigma_1,\sigma_2}, y - \mu).\end{aligned}$$

The CDFs are given by $F_{\mu,\sigma_1,\sigma_2}(x) = F_{\sigma_1,\sigma_2}(x-\mu)$ and

$$F_{\sigma_1,\sigma_2}(x) = \begin{cases} \frac{\sigma_1}{\sigma_1+\sigma_2} \exp\left(\frac{x}{\sigma_1}\right), & x < 0, \\ 1 - \frac{\sigma_2}{\sigma_1+\sigma_2} \exp\left(-\frac{x}{\sigma_2}\right), & x \geq 0. \end{cases}$$

Two-piece normal distribution

The function `crps_2pnorm()` computes the CRPS for the two-piece exponential distribution with `scale1` and `scale2` parameters $\sigma_1, \sigma_2 > 0$, and generalizes via `location` parameter $\mu \in \mathbb{R}$,

$$\begin{aligned} \mathrm{CRPS}(F_{\sigma_1,\sigma_2}, y) &= \sigma_1 \, \mathrm{CRPS}\left(F_{-\infty,0}^{0,\sigma_2/(\sigma_1+\sigma_2)}, \tfrac{\min(0,y)}{\sigma_1}\right) \\ &\quad + \sigma_2 \, \mathrm{CRPS}\left(F_{0,\sigma_1/(\sigma_1+\sigma_2)}^{\infty,0}, \tfrac{\max(0,y)}{\sigma_2}\right), \\ \mathrm{CRPS}(F_{\mu,\sigma_1,\sigma_2}, y) &= \mathrm{CRPS}(F_{\sigma_1,\sigma_2}, y-\mu), \end{aligned}$$

where $F_{l,L}^{u,U}$ is the CDF of the generalized truncated/censored normal distribution as in Section A.4.7. The CDFs for the two-piece normal distribution are given by

$$\begin{aligned} F_{\sigma_1,\sigma_2}(x) &= \begin{cases} \frac{2\sigma_1}{\sigma_1+\sigma_2} \Phi\left(\frac{x}{\sigma_1}\right), & x < 0, \\ \frac{\sigma_1-\sigma_2}{\sigma_1+\sigma_2} + \frac{2\sigma_2}{\sigma_1+\sigma_2} \Phi\left(\frac{x}{\sigma_2}\right), & x \geq 0, \end{cases} \\ F_{\mu,\sigma_1,\sigma_2}(x) &= F_{\sigma_1,\sigma_2}(x-\mu). \end{aligned}$$

Gneiting and Thorarinsdottir (2010) give an explicit CRPS formula.

A.3. Distributions for non-negative variables

Exponential distribution

The function `crps_exp()` computes the CRPS for the exponential distribution with `rate` parameter $\lambda > 0$,

$$\mathrm{CRPS}(F_\lambda, y) = |y| - \frac{2F_\lambda(y)}{\lambda} + \frac{1}{2\lambda}.$$

The CDF is given by

$$F_\lambda(x) = \begin{cases} 1 - \exp(-\lambda x), & x \geq 0, \\ 0, & x < 0. \end{cases}$$

Gamma distribution

The function `crps_gamma()` computes the CRPS for the gamma distribution with `shape` parameter $\alpha > 0$ and `rate` parameter $\beta > 0$, or alternatively `scale = 1/rate`,

$$\mathrm{CRPS}(F_{\alpha,\beta}, y) = y\,(2F_{\alpha,\beta}(y) - 1) - \frac{\alpha}{\beta}\,(2F_{\alpha+1,\beta}(y) - 1) - \frac{1}{\beta B\left(\frac{1}{2}, \alpha\right)}.$$

The CDF is given by

$$F_{\alpha,\beta}(x) = \begin{cases} \frac{\Gamma_l(\alpha,\beta x)}{\Gamma(\alpha)}, & x \geq 0, \\ 0, & x < 0. \end{cases}$$

Derived by Möller and Scheuerer (2015).

Log-Laplace distribution

The function `crps_llapl()` computes the CRPS for the log-Laplace distribution with `locationlog` parameter $\mu \in \mathbb{R}$ and `scalelog` parameter $\sigma \in (0, 1)$,

$$\mathrm{CRPS}(F_{\mu,\sigma}, y) = y\,(2F_{\mu,\sigma}(y) - 1) + \exp(\mu)\left(\tfrac{\sigma}{4-\sigma^2} + A(y)\right).$$

The CDF and otherwise required functions are given by

$$F_{\mu,\sigma}(x) = \begin{cases} 0, & x \leq 0, \\ \frac{1}{2}\exp\left(\frac{\log x - \mu}{\sigma}\right), & 0 < x < \exp(\mu), \\ 1 - \frac{1}{2}\exp\left(-\frac{\log x - \mu}{\sigma}\right), & x \geq \exp(\mu), \end{cases}$$

$$A(x) = \begin{cases} \frac{1}{1+\sigma}\left(1 - (2F_{\mu,\sigma}(x))^{1+\sigma}\right), & x < \exp(\mu), \\ -\frac{1}{1-\sigma}\left(1 - (2(1 - F_{\mu,\sigma}(x)))^{1-\sigma}\right), & y \geq \exp(\mu). \end{cases}$$

Log-logistic distribution

The function `crps_llogis()` computes the CRPS for the log-logistic distribution with `locationlog` parameter $\mu \in \mathbb{R}$ and `scalelog` parameter $\sigma \in (0, 1)$,

$$\begin{aligned}\mathrm{CRPS}(F_{\mu,\sigma}, y) = {} & y\,(2F_{\mu,\sigma}(y) - 1) \\ & - \exp(\mu)B(1+\sigma, 1-\sigma)\,(2\,I(1+\sigma, 1-\sigma, F_{\mu,\sigma}(y)) + \sigma - 1).\end{aligned}$$

The CDF is given by

$$F_{\mu,\sigma}(x) = \begin{cases} 0, & x \leq 0, \\ \left(1 + \exp\left(-\frac{\log x - \mu}{\sigma}\right)\right)^{-1}, & x > 0. \end{cases}$$

Taillardat, Mestre, Zamo, and Naveau (2016) give an alternative CRPS formula.

Log-normal distribution

The function `crps_lnorm()` computes the CRPS for the log-logistic distribution with `locationlog` parameter $\mu \in \mathbb{R}$ and `scalelog` parameter $\sigma > 0$,

$$\mathrm{CRPS}(F_{\mu,\sigma}, y) = y\,(2F_{\mu,\sigma}(y) - 1) - 2\exp(\mu + \sigma^2/2)\left(\Phi\left(\tfrac{\log y - \mu - \sigma^2}{\sigma}\right) + \Phi\left(\tfrac{\sigma}{\sqrt{2}}\right) - 1\right).$$

The CDF is given by

$$F_{\mu,\sigma}(x) = \begin{cases} 0, & x \leq 0, \\ \Phi\left(\frac{\log x - \mu}{\sigma}\right), & x > 0. \end{cases}$$

Derived by Baran and Lerch (2015).

A.4. Distribution with flexible support and/or point masses

Beta distribution

The function `crps_beta()` computes the CRPS for the beta distribution with `shape1` and `shape2` parameters $\alpha, \beta > 0$, and generalizes via `lower` and `upper` parameters $l, u \in \mathbb{R}$, $l < u$,

$$\begin{aligned}
\mathrm{CRPS}(F_{\alpha,\beta}, y) &= y(2F_{\alpha,\beta}(y) - 1) + \frac{\alpha}{\alpha+\beta}\left(1 - 2F_{\alpha+1,\beta}(y) - \frac{2B(2\alpha, 2\beta)}{\alpha B(\alpha,\beta)^2}\right), \\
\mathrm{CRPS}(F^u_{l,\alpha,\beta}, y) &= (u-l)\,\mathrm{CRPS}\left(F_{\alpha,\beta}, \tfrac{y-l}{u-l}\right).
\end{aligned}$$

The CDFs are given by $F^u_{l,\alpha,\beta}(x) = F_{\alpha,\beta}\left(\frac{x-l}{u-l}\right)$ and

$$F_{\alpha,\beta}(x) = \begin{cases} 0, & x < 0, \\ I(\alpha,\beta,x), & 0 \le x < 1, \\ 1, & x \ge 1. \end{cases}$$

Taillardat *et al.* (2016) give an equivalent expression.

Continuous uniform distribution

The function `crps_unif()` computes the CRPS for the continuous uniform distribution on the unit interval, and generalizes via `min` and `max` parameters $l, u \in \mathbb{R}$, $l < u$, and by allowing point masses in the boundaries, i.e., `lmass` and `umass` parameters $L, U \ge 0$, $L + U < 1$,

$$\begin{aligned}
\mathrm{CRPS}(F, y) &= |y - F(y)| + F(y)^2 - F(y) + \frac{1}{3}, \\
\mathrm{CRPS}(F^U_L, y) &= |y - F(y)| + F(y)^2(1 - L - U) - F(y)(1 - 2L) \\
&\quad + \frac{(1-L-U)^2}{3} + (1-L)U, \\
\mathrm{CRPS}(F^{u,U}_{l,L}, y) &= (u-l)\,\mathrm{CRPS}\left(F^U_L, \tfrac{y-l}{u-l}\right).
\end{aligned}$$

The CDFs are given by $F^{u,U}_{l,L}(x) = F^U_L\left(\frac{x-l}{u-l}\right)$ and

$$F(x) = \begin{cases} 0, & x < 0, \\ x, & 0 \le x < 1, \\ 1, & x \ge 1, \end{cases}$$

$$F^U_L(x) = \begin{cases} 0, & x < 0, \\ L + (1-L-U)x, & 0 \le x < 1, \\ 1, & x \ge 1. \end{cases}$$

Exponential distribution with point mass

The function `crps_expM()` computes the CRPS for the standard exponential distribution, and generalizes via `location` parameter $\mu \in \mathbb{R}$ and `scale` parameter $\sigma > 0$, and by allowing

a point mass in the boundary, i.e., a `mass` parameter $M \in [0, 1]$,

$$\begin{aligned}\mathrm{CRPS}(F_M, y) &= |y| - 2(1-M)F(y) + \frac{(1-M)^2}{2},\\ \mathrm{CRPS}(F_{M,\mu,\sigma}, y) &= \sigma\,\mathrm{CRPS}\left(F_M, \tfrac{y-\mu}{\sigma}\right).\end{aligned}$$

The CDFs are given by $F_{M,\mu,\sigma}(x) = F_M\left(\frac{x-\mu}{\sigma}\right)$ and

$$\begin{aligned}F(x) &= \begin{cases}1 - \exp(-x), & x \geq 0,\\ 0, & x < 0,\end{cases}\\ F_M(x) &= \begin{cases}M + (1-M)F(x), & x \geq 0,\\ 0, & x < 0.\end{cases}\end{aligned}$$

Generalized extreme value distribution

The function `crps_gev()` computes the CRPS for the generalized extreme value distribution with `shape` parameter $\xi < 1$, and generalizes via `location` parameter $\mu \in \mathbb{R}$ and `scale` parameter $\sigma > 0$,

$$\begin{aligned}\mathrm{CRPS}(F_\xi, y) &= \begin{cases}-y - 2\,\mathrm{Ei}(\log F_\xi(y)) + \gamma - \log 2, & \xi = 0,\\ y\,(2F_\xi(y) - 1) - 2G_\xi(y) - \frac{1-\left(2-2^\xi\right)\Gamma(1-\xi)}{\xi}, & \xi \neq 0,\end{cases}\\ \mathrm{CRPS}(F_{\xi,\mu,\sigma}, y) &= \sigma\,\mathrm{CRPS}\left(F_\xi, \tfrac{y-\mu}{\sigma}\right).\end{aligned}$$

The CDFs and otherwise required functions are given by $F_{\xi,\mu,\sigma}(x) = F_\xi\left(\frac{x-\mu}{\sigma}\right)$ and

$$\begin{aligned}&\text{for } \xi = 0: \quad F_\xi(x) = \exp\left(-\exp(-x)\right)\\ &\text{for } \xi > 0: \quad F_\xi(x) = \begin{cases}0, & x \leq -\frac{1}{\xi},\\ \exp\left(-(1+\xi x)^{-1/\xi}\right), & x > -\frac{1}{\xi},\end{cases}\\ &\qquad\qquad G_\xi(x) = \begin{cases}0, & x \leq -\frac{1}{\xi},\\ -\frac{F_\xi(x)}{\xi} + \frac{\Gamma_u(1-\xi, -\log F_\xi(x))}{\xi}, & x > -\frac{1}{\xi},\end{cases}\\ &\text{for } \xi < 0: \quad F_\xi(x) = \begin{cases}\exp\left(-(1+\xi x)^{-1/\xi}\right), & x < -\frac{1}{\xi},\\ 1, & x \geq -\frac{1}{\xi},\end{cases}\\ &\qquad\qquad G_\xi(x) = \begin{cases}-\frac{F_\xi(x)}{\xi} + \frac{\Gamma_u(1-\xi, -\log F_\xi(x))}{\xi}, & x < -\frac{1}{\xi},\\ -\frac{1}{\xi} + \frac{\Gamma(1-\xi)}{\xi}, & x \geq -\frac{1}{\xi}.\end{cases}\end{aligned}$$

Friederichs and Thorarinsdottir (2012) give an equivalent expression.

Generalized Pareto distribution with point mass

The function `crps_gpd()` computes the CRPS for the generalized extreme value distribution with `shape` parameter $\xi < 1$, and generalizes via `location` parameter $\mu \in \mathbb{R}$ and `scale`

parameter $\sigma > 0$, and by allowing a point mass in the lower boundary, i.e., a `mass` parameter $M \in [0, 1]$,

$$\begin{aligned}
\mathrm{CRPS}(F_{M,\xi}, y) &= |y| - \frac{2(1-M)}{1-\xi}\left(1 - (1 - F_\xi(y))^{1-\xi}\right) + \frac{(1-M)^2}{2-\xi}, \\
\mathrm{CRPS}(F_{M,\xi,\mu,\sigma}, y) &= \sigma\,\mathrm{CRPS}\left(F_{M,\xi}, \tfrac{y-\mu}{\sigma}\right).
\end{aligned}$$

The CDFs are given by $F_{M,\xi,\mu,\sigma}(x) = F_{M,\xi}\left(\frac{x-\mu}{\sigma}\right)$ and

$$\begin{aligned}
F_{M,\xi}(x) &= \begin{cases} M + (1-M)F_\xi(x), & x \geq 0, \\ 0, & x < 0, \end{cases} \\
\text{for } \xi = 0: \quad F_\xi(x) &= \begin{cases} 0, & x < 0, \\ 1 - \exp(-x), & x \geq 0, \end{cases} \\
\text{for } \xi > 0: \quad F_\xi(x) &= \begin{cases} 0, & x < 0, \\ 1 - (1 + \xi x)^{-1/\xi}, & x \geq 0, \end{cases} \\
\text{for } \xi < 0: \quad F_\xi(x) &= \begin{cases} 0, & x < 0, \\ 1 - (1 + \xi x)^{-1/\xi}, & 0 \leq x < |\xi|^{-1}, \\ 1, & x \geq |\xi|^{-1}. \end{cases}
\end{aligned}$$

Friederichs and Thorarinsdottir (2012) give a CRPS formula for the generalized Pareto distribution without a point mass.

Generalized truncated/censored logistic distribution

The function `crps_gtclogis()` computes the CRPS for the generalized truncated/censored logistic distribution with `location` parameter $\mu \in \mathbb{R}$, `scale` parameter $\sigma > 0$, `lower` and `upper` boundary parameters $l, u \in \mathbb{R}$, $l < u$, and by allowing point masses in the boundaries, i.e., `lmass` and `umass` parameters $L, U \geq 0$, $L + U < 1$,

$$\begin{aligned}
\mathrm{CRPS}\left(F_{l,L}^{u,U}, y\right) &= |y - z| + uU^2 - lL^2 \\
&\quad - \left(\frac{1-L-U}{F(u) - F(l)}\right) z \left(\frac{(1-2L)F(u) + (1-2U)F(l)}{1-L-U}\right) \\
&\quad - \left(\frac{1-L-U}{F(u) - F(l)}\right) (2\log F(-z) - 2G(u)U - 2G(l)L) \\
&\quad - \left(\frac{1-L-U}{F(u) - F(l)}\right)^2 (H(u) - H(l)), \\
\mathrm{CRPS}(F_{l,L,\mu,\sigma}^{u,U}, y) &= \sigma\,\mathrm{CRPS}\left(F_{(l-\mu)/\sigma,L}^{(u-\mu)/\sigma,U}, \tfrac{y-\mu}{\sigma}\right),
\end{aligned}$$

The CDFs are given by $F(x) = (1 + \exp(-x))^{-1}$ and

$$\begin{aligned}
F_{l,L}^{u,U}(x) &= \begin{cases} 0, & x < l, \\ \frac{1-L-U}{F(u)-F(l)}F(x) - \frac{1-L-U}{F(u)-F(l)}F(l) + L, & l \leq x < u, \\ 1, & x \geq u, \end{cases} \\
F_{l,L,\mu,\sigma}^{u,U}(x) &= F_{(l-\mu)/\sigma,L}^{(u-\mu)/\sigma,U}\left(\tfrac{x-\mu}{\sigma}\right).
\end{aligned}$$

Otherwise required functions are given by $G(x) = xF(x) + \log F(-x)$ and

$$z = \begin{cases} l, & y < l, \\ y, & l \leq y < u, \\ u, & y \geq u, \end{cases}$$
$$H(x) = F(x) - xF(x)^2 + (1 - 2F(x)) \log F(-x).$$

The function `crps_clogis()` computes the CRPS for the special case when the tail probabilities collapse into the respective boundary,

$$\mathrm{CRPS}\left(F_l^u, y\right) = |y - z| + z + \log\left(\frac{F(-l)F(u)}{F(z)^2}\right) - F(u) + F(l),$$

where the CDF is given by

$$F_l^u(x) = \begin{cases} 0, & x < l, \\ F(x), & l \leq x < u, \\ 1, & x \geq u. \end{cases}$$

The function `crps_tlogis()` computes the CRPS for the special case when $L = U = 0$, where the CDF is given by

$$F_l^u(x) = \begin{cases} 0, & x < l, \\ \frac{F(x)-F(l)}{F(u)-F(l)}, & l \leq x < u, \\ 1, & x \geq u. \end{cases}$$

Taillardat *et al.* (2016) give a formula for left-censoring at zero. Möller and Scheuerer (2015) give a formula for left-truncating at zero.

Generalized truncated/censored normal distribution

The function `crps_gtcnorm()` computes the CRPS for the generalized truncated/censored normal distribution with `location` parameter $\mu \in \mathbb{R}$, `scale` parameter $\sigma > 0$, `lower` and `upper` boundary parameters $l, u \in \mathbb{R}$, $l < u$, and by allowing point masses in the boundaries, i.e., `lmass` and `umass` parameters $L, U \geq 0$, $L + U < 1$,

$$\begin{aligned} \mathrm{CRPS}\left(F_{l,L}^{u,U}, y\right) &= |y - z| + uU^2 - lL^2 \\ &\quad + \left(\frac{1 - L - U}{\Phi(u) - \Phi(l)}\right) z \left(2\Phi(z) - \frac{(1 - 2L)\Phi(u) + (1 - 2U)\Phi(l)}{1 - L - U}\right) \\ &\quad + \left(\frac{1 - L - U}{\Phi(u) - \Phi(l)}\right) (2\varphi(z) - 2\varphi(u)U - 2\varphi(l)L) \\ &\quad - \left(\frac{1 - L - U}{\Phi(u) - \Phi(l)}\right)^2 \left(\frac{1}{\sqrt{\pi}}\right) \left(\Phi\left(u\sqrt{2}\right) - \Phi\left(l\sqrt{2}\right)\right), \\ \mathrm{CRPS}(F_{l,L,\mu,\sigma}^{u,U}, y) &= \sigma\, \mathrm{CRPS}\left(F_{(l-\mu)/\sigma,L}^{(u-\mu)/\sigma,U}, \tfrac{y-\mu}{\sigma}\right). \end{aligned}$$

The CDFs and otherwise required functions are given by

$$F_{l,L}^{u,U}(x) = \begin{cases} 0, & x < l, \\ \frac{1-L-U}{\Phi(u)-\Phi(l)}\Phi(x) - \frac{1-L-U}{\Phi(u)-\Phi(l)}\Phi(l) + L, & l \leq x < u, \\ 1, & x \geq u, \end{cases}$$

$$F_{l,L,\mu,\sigma}^{u,U}(x) = F_{(l-\mu)/\sigma,L}^{(u-\mu)/\sigma,U}\left(\tfrac{x-\mu}{\sigma}\right),$$

$$z = \begin{cases} l, & y < l, \\ y, & l \leq y < u, \\ u, & y \geq u. \end{cases}$$

The function `crps_cnorm()` computes the CRPS for the special case when the tail probabilities collapse into the respective boundary, where the CDF is given by

$$F_l^u(x) = \begin{cases} 0, & x < l, \\ \Phi(x), & l \leq x < u, \\ 1, & x \geq u. \end{cases}$$

The function `crps_tnorm()` computes the CRPS for the special case when $L = U = 0$, where the CDF is given by

$$F_l^u(x) = \begin{cases} 0, & x < l, \\ \frac{F(x)-F(l)}{F(u)-F(l)}, & l \leq x < u, \\ 1, & x \geq u. \end{cases}$$

Thorarinsdottir and Gneiting (2010) give a formula for left-censoring at zero. Gneiting, Larson, Westrick, Genton, and Aldrich (2006) give a formula for left-truncating at zero.

Generalized truncated/censored Student's t distribution

The function `crps_gtct()` computes the CRPS for the generalized truncated/censored Student's t distribution with `df` parameter $\nu > 1$, `location` parameter $\mu \in \mathbb{R}$, `scale` parameter $\sigma > 0$, `lower` and `upper` boundary parameters $l, u \in \mathbb{R}$, $l < u$, and by allowing point masses in the boundaries, i.e., `lmass` and `umass` parameters $L, U \geq 0$, $L + U < 1$,

$$\begin{aligned} \mathrm{CRPS}\left(F_{l,L,\nu}^{u,U}, y\right) &= |y - z| + uU^2 - lL^2 \\ &\quad + \left(\frac{1-L-U}{F_\nu(u) - F_\nu(l)}\right) z \left(2F_\nu(z) - \frac{(1-2L)F_\nu(u) + (1-2U)F_\nu(l)}{1-L-U}\right) \\ &\quad - \left(\frac{1-L-U}{F_\nu(u) - F_\nu(l)}\right) \left(2G_\nu(z) - 2G_\nu(u)U - 2G_\nu(l)L\right) \\ &\quad - \left(\frac{1-L-U}{F_\nu(u) - F_\nu(l)}\right)^2 \bar{B}_\nu\left(H_\nu(u) - H_\nu(l)\right), \\ \mathrm{CRPS}(F_{l,L,\nu,\mu,\sigma}^{u,U}, y) &= \sigma\, \mathrm{CRPS}\left(F_{(l-\mu)/\sigma,L,\nu}^{(u-\mu)/\sigma,U}, \tfrac{y-\mu}{\sigma}\right). \end{aligned}$$

The CDFs are given by

$$F_\nu(x) = \frac{1}{2} + \frac{x\ {}_2F_1\left(\frac{1}{2}, \frac{\nu+1}{2}; \frac{3}{2}; -\frac{x^2}{\nu}\right)}{\sqrt{\nu}B\left(\frac{1}{2}, \frac{\nu}{2}\right)},$$

$$F^{u,U}_{l,L,\nu}(x) = \begin{cases} 0, & x < l, \\ \frac{1-L-U}{F(u)-F(l)}F(z) - \frac{1-L-U}{F(u)-F(l)}F(l) + L, & l \le x < u, \\ 1, & x \ge u, \end{cases}$$

$$F^{u,U}_{l,L,\nu,\mu,\sigma}(x) = F^{\frac{u-\mu}{\sigma},U}_{\frac{l-\mu}{\sigma},L,\nu}\left(\frac{x-\mu}{\sigma}\right).$$

Otherwise required functions are given by

$$z = \begin{cases} l, & y < l, \\ y, & l \le y < u, \\ u, & y \ge u, \end{cases}$$

$$f_\nu(x) = \frac{1}{\sqrt{\nu}B\left(\frac{1}{2}, \frac{\nu}{2}\right)}\left(1 + \frac{x^2}{\nu}\right)^{-(\nu+1)/2},$$

$$G_\nu(x) = -\left(\frac{\nu + x^2}{\nu - 1}\right) f_\nu(x),$$

$$H_\nu(x) = \frac{1}{2} + \frac{1}{2}\operatorname{sgn}(x)\, I\left(\frac{1}{2}, \nu - \frac{1}{2}, \frac{x^2}{\nu + x^2}\right),$$

$$\bar{B}_\nu = \left(\frac{2\sqrt{\nu}}{\nu - 1}\right)\frac{B\left(\frac{1}{2}, \nu - \frac{1}{2}\right)}{B\left(\frac{1}{2}, \frac{\nu}{2}\right)^2}.$$

The function `crps_ct()` computes the CRPS for the special case when the tail probabilities collapse into the respective boundary, where the CDF is given by

$$F^u_{l,\nu}(x) = \begin{cases} 0, & x < l, \\ F_\nu(x), & l \le x < u, \\ 1, & x \ge u. \end{cases}$$

The function `crps_tt()` computes the CRPS for the special case when $L = U = 0$, where the CDF is given by

$$F^u_{l,\nu}(x) = \begin{cases} 0, & x < l, \\ \frac{F_\nu(x) - F_\nu(l)}{F_\nu(u) - F_\nu(l)}, & l \le x < u, \\ 1, & x \ge u, \end{cases}$$

A.5. Distribution for discrete variables

Binomial distribution

The function `crps_binom()` computes the CRPS for the binomial distribution with `size`

parameter $n = 0, 1, 2, \ldots,$, and `prob` parameter $p \in [0, 1]$,

$$\mathrm{CRPS}(F_{n,p}, y) = 2 \sum_{x=0}^{n} f_{n,p}(x) \left(\mathbb{1}\{y < x\} - F_{n,p}(x) + f_{n,p}(x)/2 \right) (x - y).$$

The CDF and probability mass function are given by

$$F_{n,p}(x) = \begin{cases} I\left(n - \lfloor x \rfloor, \lfloor x \rfloor + 1, 1 - p\right), & x \geq 0, \\ 0, & x < 0, \end{cases}$$

$$f_{n,p}(x) = \begin{cases} \binom{n}{x} p^x (1-p)^{n-x}, & x = 0, 1, \ldots, n, \\ 0, & \text{otherwise.} \end{cases}$$

Hypergeometric distribution

The function `crps_hyper()` computes the CRPS for the hypergeometric distribution with two population parameters, the number $m = 0, 1, \ldots,$ of entities with the relevant feature and the number $n = 0, 1, \ldots,$ of entities without that feature, and a parameter for the size $k = 0, \ldots, m + n$ of the sample to be drawn,

$$\mathrm{CRPS}(F_{m,n,k}, y) = 2 \sum_{x=0}^{n} f_{m,n,k}(x) \left(\mathbb{1}\{y < x\} - F_{m,n,k}(x) + f_{m,n,k}(x)/2 \right) (x - y).$$

The CDF and probability mass function are given by

$$F_{m,n,k}(x) = \begin{cases} \sum_{i=0}^{\lfloor x \rfloor} f_{m,n,k}(i), & x \geq 0, \\ 0, & x < 0, \end{cases}$$

$$f_{m,n,k}(x) = \begin{cases} \frac{\binom{m}{x}\binom{n}{k-x}}{\binom{m+n}{k}}, & x = \max\{0, k-n\}, \ldots, \min\{k, m\}, \\ 0, & \text{otherwise.} \end{cases}$$

Negative binomial distribution

The function `crps_nbinom()` computes the CRPS for the negative binomial distribution with `size` parameter $n > 0$, and `prob` parameter $p \in (0, 1]$ or alternatively a non-negative mean parameter given to `mu`,

$$\begin{aligned} \mathrm{CRPS}(F_{n,p}, y) = {} & y \left(2F_{n,p}(y) - 1\right) \\ & - \frac{n(1-p)}{p^2} \left(p \left(2F_{n+1,p}(y-1) - 1\right) + {}_2F_1\left(n+1, \tfrac{1}{2}; 2; -\tfrac{4(1-p)}{p^2}\right) \right). \end{aligned}$$

The CDF and probability mass function are given by

$$F_{n,p}(x) = \begin{cases} I\left(n, \lfloor x + 1 \rfloor, p\right), & x \geq 0, \\ 0, & x < 0, \end{cases}$$

$$f_{n,p}(x) = \begin{cases} \frac{\Gamma(x+n)}{\Gamma(n)x!} p^n (1-p)^x, & x = 0, 1, 2, \ldots, \\ 0, & \text{otherwise.} \end{cases}$$

Derived by Wei and Held (2014).

Poisson distribution

The function `crps_pois()` computes the CRPS for the Poisson distribution with mean parameter $\lambda > 0$ given to `lambda`,

$$\mathrm{CRPS}(F_\lambda, y) = (y - \lambda)\,(2F_\lambda(y) - 1) + 2\lambda f_\lambda\,(\lfloor y \rfloor) - \lambda \exp(-2\lambda)\,(I_0(2\lambda) + I_1(2\lambda))\,.$$

The CDF and probability mass function are given by

$$F_\lambda(x) = \begin{cases} \frac{\Gamma_u(\lfloor x+1 \rfloor, \lambda)}{\Gamma(\lfloor x+1 \rfloor)}, & x \geq 0, \\ 0, & x < 0, \end{cases}$$

$$f_\lambda(x) = \begin{cases} \frac{\lambda^x}{x!} e^{-\lambda}, & x = 0, 1, 2, \ldots, \\ 0, & \text{otherwise}, \end{cases}$$

Derived by Wei and Held (2014).

B. Computation of multivariate scores for multiple forecasts

As noted in Section 5 the computation functions for multivariate scoring rules are defined for single forecast cases only. Here, we demonstrate how `apply` functions can be used to compute ES and VS^p for multiple forecast cases. The simulation example is based on the function documentation of `es_sample()` and `vs_sample()`.

The observation is generated as a sample from a multivariate normal distribution in $\mathbb{R}^{10}$ with mean vector $\boldsymbol{\mu} = (0, \ldots, 0)$ and covariance matrix $\boldsymbol{\Sigma}$ with $\boldsymbol{\Sigma}_{i,j} = 1$ if $i = j$ and $\boldsymbol{\Sigma}_{i,j} = c = 0.2$ if $i \neq j$ for all $i, j = 1, \ldots, 10$.

```
R> d <- 10
R> mu <- rep(0, d)
R> Sigma <- diag(d)
R> Sigma[!diag(d)] <- 0.2
```

The multivariate forecasts are given by 50 random samples from a corresponding multivariate normal distribution with mean vector $\boldsymbol{\mu}^f = (1, \ldots, 1)$ and covariance matrix $\boldsymbol{\Sigma}^f$ which is defined as $\boldsymbol{\Sigma}$, but with $c = 0.1$.

```
R> m <- 50
R> mu_f <- rep(1, d)
R> Sigma_f <- diag(d)
R> Sigma_f[!diag(d)] <- 0.1
```

The simulation process is independently repeated 1 000 times. To illustrate two potential data structures, observations and forecasts are saved as list elements in an outer list where the index corresponds to the forecast case, and as 2- and 3-dimensional arrays where the last dimension indicates the forecast case.

```
R> n <- 1000
R> fc_obs_list <- vector("list", n)
R> obs_array <- matrix(NA, nrow = d, ncol = n)
R> fc_array <- array(NA, dim = c(d, m, n))
R> for (fc_case in 1:n) {
+    obs_tmp <- drop(mu + rnorm(d) %*% chol(Sigma))
+    fc_tmp <- replicate(m, drop(mu_f + rnorm(d) %*% chol(Sigma_f)))
+    fc_obs_list[[fc_case]] <- list(obs = obs_tmp, fc_sample = fc_tmp)
+    obs_array[, fc_case] <- obs_tmp
+    fc_array[, , fc_case] <- fc_tmp
+  }
```

Given the data structures of forecasts and observations, all 1 000 forecast cases can be evaluated sequentially using the `sapply()` function (or, alternatively, a `for` loop) along the list elements or along the last array dimension.

```
R> es_vec_list <- sapply(fc_obs_list, function(x) es_sample(y = x$obs,
+    dat = x$fc_sample))
R> es_vec_array <- sapply(1:n, function(i) es_sample(y = obs_array[, i],
+    dat = fc_array[, , i]))
R> head(cbind(es_vec_list, es_vec_array))

     es_vec_list es_vec_array
[1,]        2.44         2.44
[2,]        2.68         2.68
[3,]        2.56         2.56
[4,]        1.85         1.85
[5,]        3.83         3.83
[6,]        3.04         3.04
```

Affiliation:

Alexander Jordan
University of Bern
Institute of Mathematical Statistics and Actuarial Science
Alpeneggstrasse 22
3012 Bern, Switzerland
E-Mail: alexander.jordan@stat.unibe.ch

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

Sebastian Lerch
Karlsruhe Institute of Technology
Institute for Stochastics
Englerstr. 2
76131 Karlsruhe, Germany
and
Heidelberg Institute for Theoretical Studies
E-Mail: sebastian.lerch@kit.edu
URL: https://sites.google.com/site/sebastianlerch/

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
August 2019, Volume 90, Issue 12 *Submitted:* 2017-11-08
doi:10.18637/jss.v090.i12 *Accepted:* 2018-08-12