Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

Evaluating Covid-19 Short-Term Forecasts using scoringutils in R

Nikos I. Bosse
London School of Hygiene and Tropical Medicine

Second Author
Plus Affiliation

Abstract

Forecasts play an important role in a variety of fields. Its role in informing public policy has attracted increased attention from the general public with the emergence of the Covid-19 pandemic. Evaluating forecasts, however, is not trivial in practice. Even though scoring methods such as proper scoring rules have been widely studied in the past, there is a lack of software implementation that allows a forecaster to conveniently evaluate their forecasts. In this paper we introduce **scoringutils**, an R package that facilitates automated forecast evaluation. It gives the user access to a wide range of scoring metrics for various types of forecasts as well as a variety of ways to visualise the evaluation. We give an overview of the evaluation process and the metrics implemented in **scoringutils** and show a full evaluation of a set of short-term forecasts of public health related targets made by SPI-M during the 2020 Covid-19 epidemic in the United Kingdom.

Keywords: JSS, style guide, comma-separated, not capitalized, R.

1. Introduction

Good forecasts are of great interest to decision makers in various fields like finance (), weather predictions or infectious disease modeling (Funk et al. 2020). An integral part of assessing and improving their usefulness is forecast evaluation. For decades, researchers therefore have developed and refined an arsenal of techniques not only to forecast, but also to evaluate these forecasts (see e.g. Bracher et al. (2020), Funk et al. (2019), Gneiting, Balabdaoui, and Raftery (2007), and Gneiting and Raftery (2007)). Yet even with this rich body of research available, implementing a complete forecast evaluation in R is not trivial. We therefore present the **scoringutils** package. The goal of the scoringuitls package is to facilitate the evaluation process and to allow even inexperienced users to perform a thorough evaluation of their forecasts. In this paper we give a quick introduction of the fundamental ideas behind

forecast evaluation, explain the evaluation metrics implemented in **scoringutils** and present a full example evaluation of Covid-19 related short-term forecasts in the UK (Funk et al. 2020).

1.1. Forecast types and forecast formats

In its most general sense, a forecast is the forecaster's stated belief about the future (Gneiting and Raftery 2007) that can come in many different forms. Quantitative forecasts are either point forecasts or probabilistic in nature and can make statements about continuous, discrete or binary outcome variables. Point forecasts only give one single number for the most likely outcome, but do not quantify the forecaster's uncertainty. This limits their usefulness, as a very certain forecast may, for example, warrant a very different course of actions than does a very uncertain one. Probabilistic forecasts, in contrast, by definition provide a full predictive distribution. This makes them much more useful in any applied setting, as we learn about the forecaster's uncertainty and their belief about all aspects of the underlying data-generating distribution (including e.g. skewness or the width of its tails). Probabilistic forecasts are therefore the focus of this paper as well as the **scoringutils** package.

The predictive distribution of a probabilistic forecast can be represented in different ways with implications for the appropriate evaluation approach. For most forecasting problems, predictive distributions are not readily available in a closed form (and the **scoringutils** package therefore does not support scoring them directly). Instead, predictive distributions are often represented by a set of quantiles or predictive samples. Predictive samples require a lot of storage space and also come with a loss of precision that is especially pronounced in the tails of the predictive distribution, where quite a lot of samples are needed to accurately characterise the distribution. For that reason, often quantiles or central prediction intervals are reported instead [citation FORECAST HUBS]. For binary prediction targets, common in many classification problems, a probabilistic forecasts is represented by the probability that an outcome will come true. DO I NEED TO TALK ABOUT MULTINOMIAL CLASSIFICATION PROBLEMS? Table 2 summarises the different forecast types and formats. The general forecasting paradigm Gneiting, Balabdaoui, and Raftery (2007), however, that guides the evaluation process is the same irrespective of the reporting format, even though specific scoring metrics differ.

Forecast type	**Target type**	**Representation of the predictive distribution**
Point forecast	continuous integer binary	one single number for the predicted outcome
Probabilistic forecast	continuous integer	predictive samples quantiles closed analytical form
	binary	binary probabilities

Table 2: Ssummary of the different forceast types and forecast formats

1.2. The forecasting paradigm

Any forecaster should aim to minimise the difference between the predictive distribution F and the unknown true data-generating distribution G (Gneiting, Balabdaoui, and Raftery 2007). For an ideal forecast, we therefore have

$$F = G.$$

As we don't know the true data-generating distribution, we cannot assess the difference between the two distributions directly. Gneiting, Balabdaoui, and Raftery (2007) instead suggest to focus on two central aspects of the predictive distribution, calibration and sharpness. Calibration refers to the statistical consistency between the predictive distribution and the observations. A well calibrated forecast does not systematically deviate from the observed values. For an in-depth discussion of different ways in which a forecast can be miscalibrated, we refer to Gneiting, Balabdaoui, and Raftery (2007). Sharpness is a feature of the forecast only and describes how concentrated the predictive distribution is, i.e. how precise the forecasts are. The general forecasting paradigm states that we should maximise sharpness of the predictive distribution subject to calibration. A model that made very precise forecasts would be at best useless if the forecasts were wrong most of the time. On the other hand, a model may be well calibrated, but not sharp enough to be useful. Take a weather forecast that would assign 30 percent rain probability for every single day. It may be well calibrated over the course of a year (it would be marginally calibrated according to Gneiting, Balabdaoui, and Raftery (2007)), but we would of course prefer a more precise forecast. Figure 1 illustrates the concepts of calibration and sharpness.

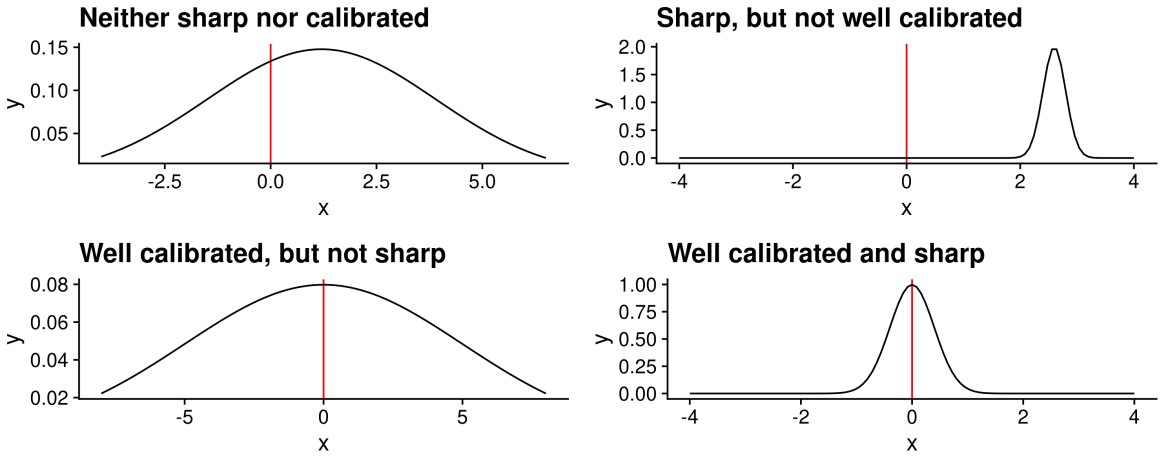


Figure 1: Schematic illustration of calibration and sharpness. True value are represented in red, the predictive distribution is shown in black

2. Scoring metrics implemented in scoringutils

Some of the metrics in **scoringuitls** focus only on sharpness or on calibration. Others, called proper scoring rules, combine both aspects into a single number. The former can be helpful

to learn about specific model aspects and improve them, the latter are especially useful to assess and rank predictive performance of a forecaster. The following gives an introduction to how these metrics can be used to evaluate forecasts. Table 4 shows an overview of the metrics implemented in **scoringutils**. Table 5 in the Appendix gives a thorough explanation of all the metrics.

2.1. Proper scoring rules

Proper scoring rules (Gneiting and Raftery 2007) jointly assess sharpness and calibration and assign a single numeric value to a forecast. A scoring rule is proper if a perfect forecaster (the predictive distribution equals the data-generating distribution) receives the lowest score on average. This makes sure that a forecaster evaluated by a proper scoring rule is always incentivised to state their best estimate.

For sample-based forecasts, the (continuous) ranked probability score (crps) [CITATION], the log score (logs) [CITATION], and the Dawid-Sebastiani-score (dss) [CITATION] are available in **scoringutils**. These are implemented as wrappers around functions from the **scoringRules** package (which also has closed-form versions of the scoring rules available). They are in principle applicable to continuous as well as integer forecasts. The **scoringRules** implementation of the log score, however, requires a kernel density estimation that may be inappropriate for integer values (see also Table 4) and is therefore not available for discrete predictions in **scoringutils**. For forecasts in an interval or quantile format, the weighted interval score (wis) (Bracher et al. 2020) is available. For an increasing number of equally-spaced prediction intervals, the wis converges to the crps and therefore has very similar properties. Binary forecasts can be scored using the Brier score (bs) [CITATION].

When scoring forecasts in a sample-based format, the choice is usually between the log score and the crps. The dss is much less commonly used. It is easier to compute, but apart from that does not have immediate advantages over the former two. Crps and log score differ in two important aspects: the first is sensitivity to distance Winkler et al. (1996), the second is how harshly far-off predictions are punished.

The crps is a so-called global scoring rule, which means that the entire predictive distribution is taken into account when scoring a single forecast. The log score, on the other hand is local. The resulting score does not depend on the overall distance between the observed value and the distribution, but only on the probality density assigend to the actual outcome. Imagine two forecasters, A and B, who forecast the number of points scored in a basketball game. If both forecasters assigned the same probability to the true outcome (100 points), but A assigned higher probability to extreme outcomes far away from the actually observed outcome, then A will receive a worse score than B. The log score, in contrast, is a local scoring rule that only scores the probability assigned to the actual outcome and ignores the rest of the predictive distribution. Judged by the log score, A and B would receive exactly the same score. Sensitivity to distance (taking the entire predictive distribution into account) is arguably an advantage in most settings that involve decision making. Forecaster A's prediction that assigns high probability to results far away from the observed value is arguably less useful than B's forecast that assigns higher probability to values closer to it (the probability assigned to the actual outcome being equal for both forecasts). The log score is only implicitly sensitive to distance if we assume that values close to the observed value are actually more likely to occur. It may, however, be more appropriate for inferential purposes

(see Winkler et al. (1996)).

A second important difference is how forecasts are treated that deviate strongly from the observered outcome. The crps can be thought of as a generalisation of the absolute error to a predictive distribution. It therefore scales linearly with the distance between forecast distribution and true value. The log score, however, is the log of the predictive density evaluated at the observed value. It can therfore quickly go to negative infinity if the probability assigned to the observed outcome is close to zero. The crps is therefore considered more stable than the log score. The behaviour of the dss is in between the two. Whether or not harsh punishment of bad predictions is desirable or not depends of course on the setting. Bracher et al. (2020) exemplify that in practice there may indeed be substantial differences between how the crps and log score judge the same forecast.

For quantile forecasts the wis is the best available option and can be understood as an approximation to the crps. One additional benefit of the wis is that it can easily be decomposed into three additive components: an uncertainty penalty (sharpness) for the width of a prediction interval and penalties for over- and underprediction (if a value falls outside of a prediction interval). This can be very helpful in diagnosing model problems. It may therefore even be useful to convert samples into quantiles and use the wis instead of the crps to make use of this decomposition for the purpose of model diagnostics. Both crps and wis, as generalisations of the absolute value, suffer from the fact that overall scores depend on the order of magnitude of the quantity we try to forecast. It can therefore be difficult to compare forecasts for very different targets. One possibility to address this is the use of pairwise comparisons introduced later. In general, this is much less of a problem for the log score and the dss. IS IT ONE AT ALL?

2.2. Evaluating calibration and sharpness independently

In addition to the proper scoring rules outlined above, **scoringutils** makes numerous metrics available to evaluate calibration and sharpness independently. This is especially helpful for model diagnostics.

Assessing calibration

Several strategies have been proposed to detect systematic deviations of the predictive distributions from the observations (see e.g. Funk et al. (2019); Gneiting, Balabdaoui, and Raftery (2007); Gneiting and Raftery (2007)). Using **scoringutils**, we can look at three different aspects of calibration: bias, empirical coverage, and the probability integral transform (PIT).

Bias, i.e. systematic over- or underprediction, is a very common form of miscalibration which therefore deserves separate attention. The bias metric (with slightly different versions for the various forecast types and formats) captures a general tendency to over- and underpredict that is bound to be between minus one (underpredicton) and one (overprediction), where zero is ideal. It is derived by looking at how much of the probability mass of the predictive distribution is below or above the true observed value. For quantile forecasts we have second alternative approach available to assess over- and underprediction - by simply looking at the corresponding components of the weighted interval score. What is different between the over- and underprediction components and bias as described above is its sensitivity to outliers. The former are derived from absolute differences, while the latter is bound and rather captures a

general tendency to be biased.

Another way to look at calibration (precisely: probabilistic calibration in Gneiting, Balabdaoui, and Raftery (2007)) is to compare the proportion of observed values covered by different parts of the predictive distribution with the nominal coverage implied by the CDF of the distribution. This is most easily understood in the context of quantile forecasts, but can easily be transferred to sample-based continuous and integer forecasts as well. To assess empirical coverage at a certain interval range, we simply measure the proportion of true observed values that fall into corresponding range of the predictive distribution. If the 0.05, 0.25, 0.75, and 0.95 quantiles are given, then 50% of the true values should fall between the 0.25 and 0.75 quantiles and 90% should fall between the 0.05 and 0.95 quantiles. We can calculate and plot these values to inspect how well different parts of the forecast distribution are calibrated. To get an even more precise picture, we can also look at the percentage of true values below every single quantile of the predictive distribution. This allows to diagnose issues in the lower and upper tails of the prediction intervals separately. A similar way to visualise the same information is a PIT histogram. In order to conveniently assess deviations between the predictive distribution and the true data-generating distribution we can transform the observed values using the probability integral transformation (PIT) (Dawid 1984) (see more details in Table 5). If both distributions are equal, the transformed values will follow a uniform distribution. A histogram of the transformed values can help to diagnose systematic differences between the predictions and the observed values. Figure 2 exemplifies the characteristic shape of certain systematic deviations of the predicitve distribution from the true data-generating distribution. In the PIT histograms, bias leads to a triangular shape, overdispersion results in a hump shaped form and underdispersion in a U-shape. ADD INTERPRETATION FOR QUANTILE AND INTERVAL COVERAGE PLOTS HERE.

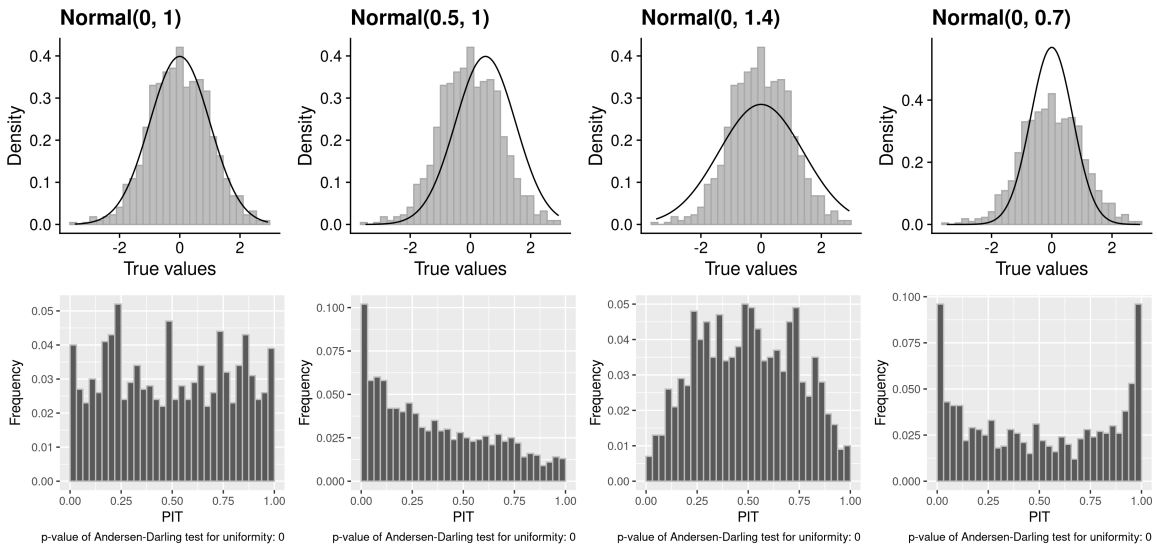


Figure 2: Calibration plots for different forecast distributions. I plan to add interval and quantile coverage plots, but these still have to be made.

Assessing sharpness

Sharpness is the ability to produce narrow forecasts. It does not depend on the actual observations and is a quality of the forecast only **??**. Sharpness is therefore only useful subject to calibration, as exemplified above in Figure **??**. We may be willing to trade off a little calibration for a lot more sharpness, but usually not much. For sample-based forecasts, **scoringutils** calculates sharpness as the normalised median absolute deviation about the median (MADN) [funkAssessingPerformanceRealtime2019] (for details see Table **??**). For quantile forecasts, we take the sharpness component of the wis which corresponds to a weighted average of the individual interval widths.

WHAT WOULD BE REALLY COOL IS TO DETERMINE A WAY TO FIND THE OPTIMAL SHARPNESS OF A FORECAST. AS IS, I FEEL THIS PARAGRAPH IS SOMEWHAT USELESS.

2.3. Pairwise comparisons

If what we care about is to determine which model performs best, pairwise comparisons between models are a suitable approach [CITATION CRAMER et al.]. In turn, each pair of models is evaluated based on the targets that both models have predicted. The mean score by one model is divided by the mean score of the other model to obtain the mean score ratio (see Table **??**, a measure of relative performance. To obtain an overall relative skill score for a model, we take the geomatric mean of all mean score ratios that involve that model (omitting comparisons where there is no overlapping set of forecasts). This gives us an indicator of performance relative to all other models. The orientation depends on the score used. For the proper scoring rules described above, smaller is better and a relative skill score smaller than 1 indicates that a model is performing better than the average model. We can obtain a scaled relative skill score by dividing a model's relative skill by the relative skill of a baseline model. A scaled relative skill smaller than one then means that the model in question performed better than the baseline.

It is in principle possible to obtain p-values that help determine whether two models perform significantly differently. **scoringutils** allows to compute these using eitab:score-tablether the Wilcoxon rank sum test or a permutation test. In practice, this is slightly complicated by the fact that both tests assume independent observations. In reality, however, forecasts by a model may be correlated across time or another dimension (e.g. if a forecaster has a bad day, they will likely perform badly across different targets for a given forecast date). P-values may therefore be too quick to suggest significant differences where there aren't any. One way to mitigate this is to aggregate observations over a category where one suspects correlation. A test that is performed on aggregate scores will likely be more conservative.

3. Evaluating UK short-term forecasts

3.1. The data

To illustrate the evaluation process with **scoringutils** we use short-term predictions of four different Covid-19 related targets in the UK made between March 31 and July 13 2020. Forecasts were produced by six groups in the UK, and submitted to the Scientific Pandemic

Influenza Group on Modelling (SPI-M). The forecasts aimed to assess the likely future burden the UK healthcare system would face from the Covid-19 pandemic. Predictions were then aggregated and used to inform UK government health policy through the Strategic Advisory Group of Experts (SAGE). The data, as well as the individual forecast models are discussed in more depth in [FUNK ET AL].

- timing (weekly?) and number of forecast dates - the four targets - the models. - is the set complete?

We first need to obtain the data by installing and loading the **covid19.forecasts.uk** using the following commands:

```
R> # install and load data pacakge from external repository
R> # remotes::install_github("sbfnk/covid19.forecasts.uk")
R>
R> # load packages
R> library(covid19.forecasts.uk)
R> library(dplyr)
R> library(scoringutils)
R> # load truth data
R> data(covid_uk_data)
R> # head(covid_uk_data)
R>
R> # load forecasts
R> data(uk_forecasts)
R> # head(uk_forecasts)
```

Let us take a first look a the data:

```
R> glimpse(covid_uk_data)

Rows: 4,174
Columns: 6
$ geography  <fct> London, London, London, London, London, London, L...
$ value_type <fct> hospital_inc, hospital_inc, hospital_inc, hospita...
$ value_desc <fct> Hospital admissions, Hospital admissions, Hospita...
$ truncation <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ value_date <date> 2020-03-20, 2020-03-21, 2020-03-22, 2020-03-23, ...
$ value      <dbl> 18, 232, 280, 241, 313, 353, 516, 637, 677, 546, ...

R> glimpse(uk_forecasts)

Rows: 1,254,513
Columns: 8
$ model         <fct> EpiSoon, EpiSoon, EpiSoon, EpiSoon, EpiSoon, E...
$ geography     <chr> "East of England", "East of England", "East of...
$ value_type    <fct> hospital_inc, hospital_inc, hospital_inc, hosp...
$ creation_date <date> 2020-03-31, 2020-03-31, 2020-03-31, 2020-03-3...
$ value_date    <date> 2020-04-01, 2020-04-02, 2020-04-03, 2020-04-0...
```

```
$ quantile      <dbl> 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05...
$ value         <dbl> 70, 64, 53, 42, 30, 19, 4, 0, 0, 0, 0, 0, 0, 0...
$ value_desc    <fct> Hospital admissions, Hospital admissions, Hosp...
```

To bring the forecasts into the format needed for the evaluation, some minor changes need to be made to the data. The names of the columns that hold the forecasts and the true observed values need to be changed to `prediction` and `true_value`. While we could also proceed with separate data sets for the evaluation, we merge the two data sets in order to remove all instances where the forecasts, but not the true observations were made public. The **scoringutils** package provides a function that attempts to merge the data sets in a sensible way.

```
R> uk_forecasts <- rename(uk_forecasts, prediction = value)
R> covid_uk_data <-rename(covid_uk_data, true_value = value)
R> combined <- merge_pred_and_obs(uk_forecasts, covid_uk_data)
```

Before we start with scoring the forecasts, it makes sense to start the evaluation process by visualising the data. To get a feeling for how complete the data set is, we can run the following code to obtain a heatmap with the number of available forecasts: Missing forecasts can have

```
R> show_avail_forecasts(combined,
+                        x = "creation_date",
+                        show_numbers = FALSE,
+                        legend_position = "bottom",
+                        facet_formula = ~ value_desc)
```

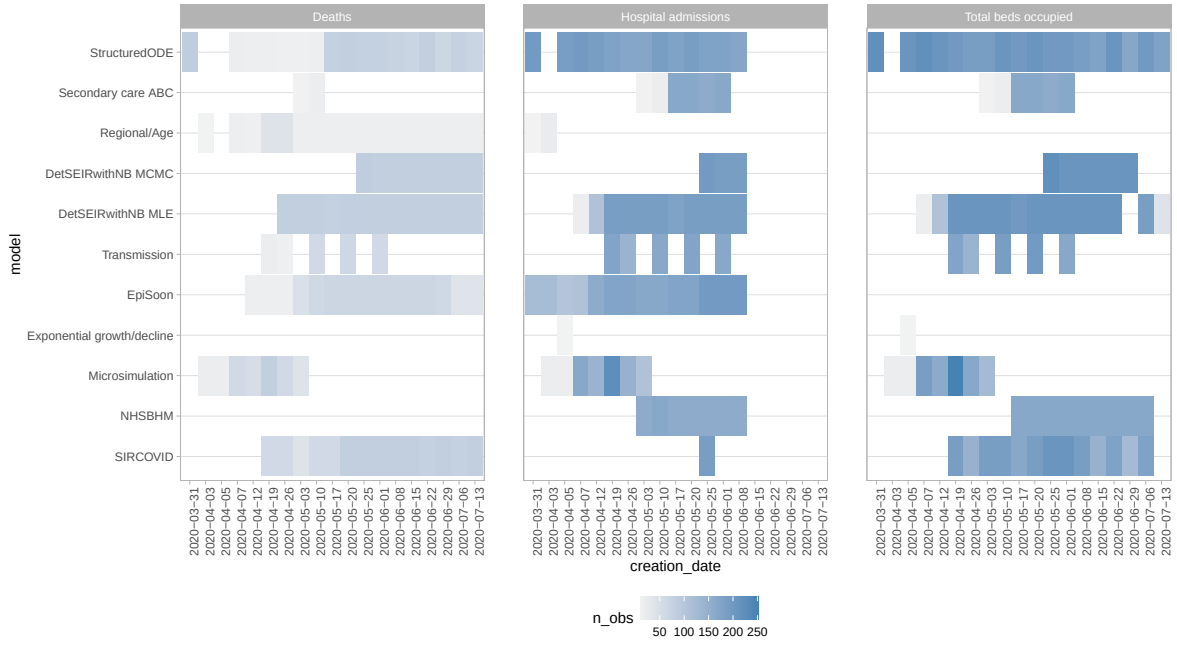


Figure 3: Overview of the number of forecasts available

a large impact on the forecast evaluation, if forecasts are not missing at random, but instead missingness correlates with performance. By default, the function treats a set of different quantiles or samples as one forecast. However, the user can specify manually which elements to treat as one forecast and which categories to sum over to count the number of available forecasts.

Forecasts can be visualised using the `plot_predictions()` function. The function accepts either a single combined data set or two separate truth data sets that can be merged together. Forecasts and observed values can be filtered independently, to show for example only predictions from a certain model or a specific number of weeks of true data before the forecast. Conditions to filter on need to be provided as a list of strings, where each of the strings represents an expression that can be evaluated to filter the data. To obtain, for example, a specific forecast from a model we are interest in, we can call: The output is shown in Figure

```
R> locations <- 'c("Scotland", "Wales", "Northern Ireland", "England")'
R> plot_predictions(truth = covid_uk_data,
+                   forecasts = uk_forecasts,
+                   filter_both = list(paste("geography %in%", locations, collapse = "")),
+                   filter_truth = list('value_date <= "2020-06-22"',
+                                       'value_date > "2020-06-01"'),
+                   filter_forecasts = list('model == "SIRCOVID"',
+                                           'creation_date == "2020-06-22"'),
+                   x = "value_date",
+                   facet_formula = geography ~ value_type) +
+    ggplot2::theme(legend.position = "bottom")
```

Figure 4: Short-term forecasts made by the SIRCOVID model on June 22 2020.

4.

3.2. Scoring forecasts with `eval_forecasts()`

A full evaluation of all forecasts based on observed values can be performed using the function `eval_forecasts()`. This requires a `data.frame` or similar which has at least a column called "prediction" and one called "true_value". Depending on the exact input format, additional columns like "sample", "quantile", or "range" and "boundary" are needed. Additional columns may be present to indicate a grouping of forecasts, for example forecasts made in different locations or over different forecast horizons. We will not discuss all possible input formats here, but instead refer to the example data for each format that is provided with the package. Where possible, **scoringutils** also provides functionality to transform between various formats, e.g. from a sample based format to a quantile format. The `eval_forecasts()` function automatically recognises the prediction type and input format, applies the appropriate scoring metrics and aggregates results as desired by the user. Internally, operations are handled using **data.table** to allow for fast and efficient computation.

As a first start for the evaluation of UK short-term forecasts, it makes sense to look at the scores achieved by every model separate for all prediction targets. This can be achieved by calling

```
R> scores <- eval_forecasts(combined,
+                           summarise_by = c("model", "value_desc"))
R> glimpse(scores)

Rows: 29
Columns: 9
$ model              <fct> EpiSoon, StructuredODE, Microsimulation, ...
$ value_desc         <fct> Hospital admissions, Hospital admissions,...
$ interval_score     <dbl> 38.56510, 61.75914, 100.71914, 35.44876, ...
$ sharpness          <dbl> 22.886153, 6.133974, 33.985553, 10.365082...
$ underprediction    <dbl> 3.66782766, 24.82255866, 33.12350598, 8.5...
$ overprediction     <dbl> 12.0111222, 30.8026045, 33.6100814, 16.55...
$ coverage_deviation <dbl> 0.034531405, -0.268749094, -0.095652174, ...
$ bias               <dbl> 0.26194170, -0.27368253, -0.39217045, -0....
$ aem                <dbl> 45.96333, 73.38096, 138.14395, 53.22563, ...
```

If a more detailed analysis is desired, the level of aggregation can of course be changed to show for example separate scores for the different locations as well. This can be achieved using the `summarise_by` argument. To additionally stratify by location, we could specify `summarise_by = c("mode", "value_type", "geography")`. If we wanted to have one score per quantile or one per prediction interval range, we could specify something like `summarise_by = c("model", "quantile")` or `summarise_by = c("model", "quantile", "range")`. This can be useful if we, for example, want to analyse what proportion of true values are covered by certain interval ranges, or if we want to analyse the accuracy of the tails of the forecasts. When aggregating, `eval_forecasts()` takes the mean according to the group defined in `summarise_by`. In the above example, if `summarise_by = c("model", "value_type")`, then scores would be averaged over all creation dates, forecast horizons (as represented by the the value dates), locations and quantiles to yield one score per model and forecast target. In addition to the mean, we can also obtain the standard deviation of the scores over which we average, as well as any desired quantile, by specifying `sd = TRUE` and for example `quantiles = c(0.5)` for the median.

The user must, however, still exercise some caution when aggregating scores, as many of the metrics are absolute and scale with the magnitude of the quantity to forecast. Looking at one score per model (i.e. specifying `summarise_by = c("model")`) may not be so useful in this instance, as overall aggregate scores would be dominated by hospital admissions, while errors on death forecasts would have little influence.

In the above example, we did not have to explicitly specify the `by` argument, but this may be necessary if additional columns are present in the data that do not indicate a grouping of forecasts. The `by` argument must then be used to denote the unit of a single forecast. In the above example, the unit of a single forecast would be `by = c("model", "geography", "value_type", "creation_date", "value_date", "value_desc")`. Quantiles should not be included, as several quantiles make up one forecast (and similarly for samples). If we had additional columns that do not serve to group forecasts (like for example the number of inhabitants in a certain location over time), these should also not be included. By default, if `by = NULL`, `eval_forecasts()` will automatically use all present columns to determine the unit of a single forecast.

3.3. Pairwise comparisons

Pairwise comparisons between models [CITATION] can be obtained in two different ways. First, relative skill scores based on pairwise comparisons are by default returned from `eval_forecasts()`. These will be computed separately for the categories defined in the `summarise_by` argument (excluding the category 'model'). Alternatively, a set of scores can be post-processed using the separate function `pairwise_comparison()`. This approach is to be used for visualisation and if p-values for the pairwise comparisons are needed, as those are not returned from `eval_forecasts()`. Usually, one would compute scores without specifying a `summarise_by` argument, but sometimes it may be sensible to average over certain scores, for example for predictions generated at a certain date. This allows to reduce the correlation between observations that enter the computation of p-values, which in turn makes the test less liberal. Using the function `plot_pairwise_comparison()` we can visualise the mean score ratios between all models as well as the

```
R> # unsummarised scores
R> unsummarised_scores <- eval_forecasts(combined)
R> pairwise <- pairwise_comparison(unsummarised_scores,
+                                  summarise_by = "value_desc")
```

The result is a `data.table` with different scores and metrics in a tidy format that can easily be used for further manipulation and plotting.

4. Visualisation and interpretation of evaluation results

4.1. Visualising aggregate scores and rankings

A good starting point for an evaluation is the following score table that visusalises the scores we produced above. We can facet the table to account for the different forecast targets:

The most informative metric in terms of model ranking is the relative_skill. However, interpretation is not always straightforward and has to be done carefully. We can see that performance varied quite a bit across different metrics, where some models did well on one target, but poorly on another. Especially the Exponential growth/decline model stands out as it received the lowest relative skill score for hospital admissions, but the highest for the total number of beds occupied. Looking back at Figure 3, we see that the model has only submitted very few forecasts over all. It may therefore be sensible to require all models to have submitted forecasts for at least 50% of all forecast targets in order to enter the pairwise comparisons. For similar reasons, the interval score may be deceiving if looked at in isolation. As can be seen, the DetSEIRwithNB MLE model received a lower relative skill score, but a higher interval score than the DetSEIRwithNB MCMC model. This, again, can be explained by the fact that they forecasted different targets. The interval score, as an absolute metric, is highly influenced by the absolute value of the quantity that is forecasted. For the same reason, one should be careful when summarising interval scores from different locations or forecast targets, as the average score will be dominated by outliers as well as differences in the absolute level. Assuming a large enough set of available overlapping forecasts, the relative skill score is more robust. It therefore is reasonable to assume that the DetSEIRwithNB MLE

```
R> score_table(scores, y = "model", facet_formula = ~ value_desc)
```

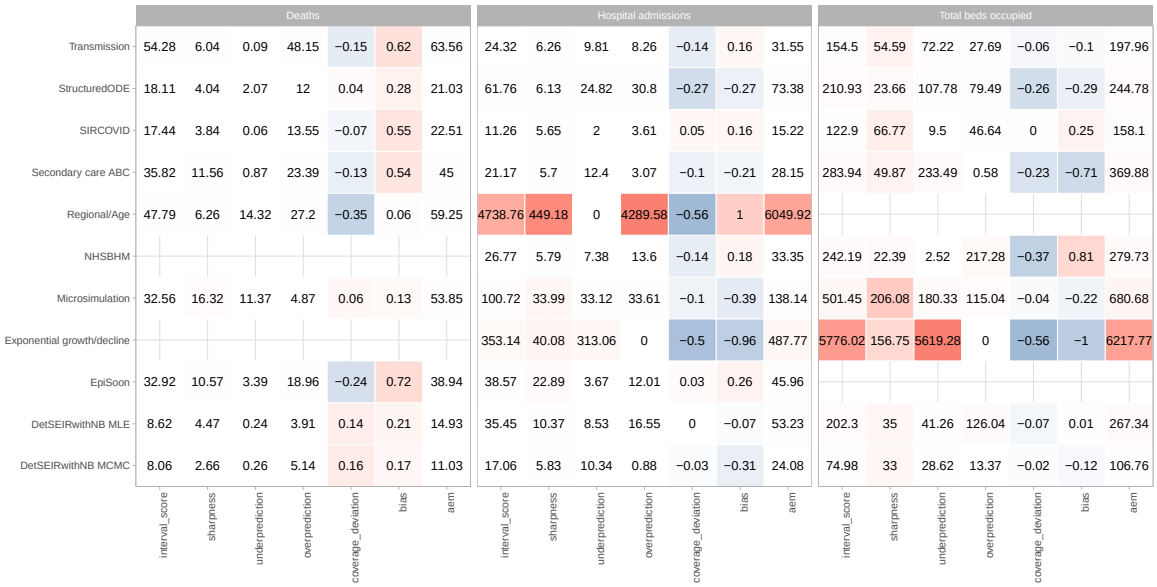


Figure 5: Coloured table to visualise the computed scores

forecasted quantities with a higher absolute value, but tended to perform worse than the DetSEIRwithNB MCMC model as far as we can tell based on the set of all pariwise comparisons. This can be confirmed for the direct comparison between the two by looking at the mean score ratios from the pairwise comparisons. These can be obtained by calling

we can also look at p-values in Figure 7 PROBABLY REMOVE THAT FROM THE PAPER

In terms of actually understanding *why* one model performs well or badly, the other metrics shown in Figure 5 provide additional insight. We turn to them in the following.

4.2. Visual model diagnostics

For forecasts in an interval format, looking at the components of the weighted interval score separately is a natural next step. We can see in Figure 8 that the majority of penalties come from over-and underprediction, instead of the sharpness component. We also see that most models tended to either over- or underpredict actual numbers.

We can have a closer look at calibration using the functions `interval_coverage()` and `quantile_coverage()`. The interval coverage plot shows the proportion of all true values that fall within all the different prediction intervals. This gives a visual impression of probabilistic calibration **??**. Ideally, x percent of true values should be covered by the x%-prediction intervals, resulting in a 45° line. Areas shaded in green indicate that the model is covering more true values than it actually should, while areas in white indicate that the model fails to cover the desired proportion of true values with its prediction intervals. The majority of the models were too confident in their predictions, while some showed showed good calibration.

```
R> plot_pairwise_comparison(pairwise) +
+    ggplot2::facet_wrap(~ value_desc, scales = "free_x")
```

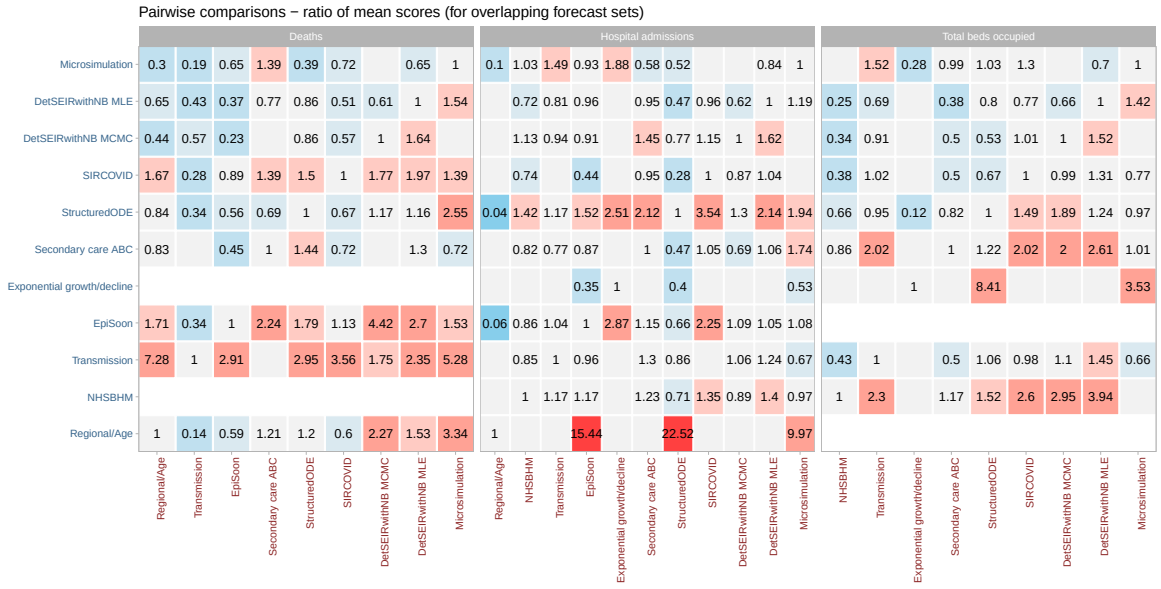


Figure 6: Ratios of mean scores based on overlapping forecast sets. If a tile is blue, then the model on the y-axis performed better. If it is red, the model on the x-axis performed better in direct comparison.

The quantile coverage plot shows the proportion of all true values below certain predictive quantiles. While this plot is slightly harder to interpret, it also includes information about bias as and allows to separate the lower and upper boundaries of the prediction intervals. We can see, for example, that the Exponential growth/decline model was consistently biased downwards. Figure 9

DO I WANT TO INCLUDE PIT PLOTS AS WELL? I GUESS? NEED TO LOOK AT THE IMPLEMENTATION FOR QUANTILE FORECASTS

Look at e.g. bias by location? Figure 10

WHAT IS NEEDED HERE IS A BIT OF THINKING WITH REGARDS TO WHAT VISUALISATION I WANT TO SHOW AND IN HOW MUCH DETAIL I WANT TO ANALYSE THE MODELS.

.

5. Summary and discussion

COMING SOON.

```
R> plot_pairwise_comparison(pairwise, type = "pval") +
+    ggplot2::facet_wrap(~ value_desc, scales = "free_x")
```

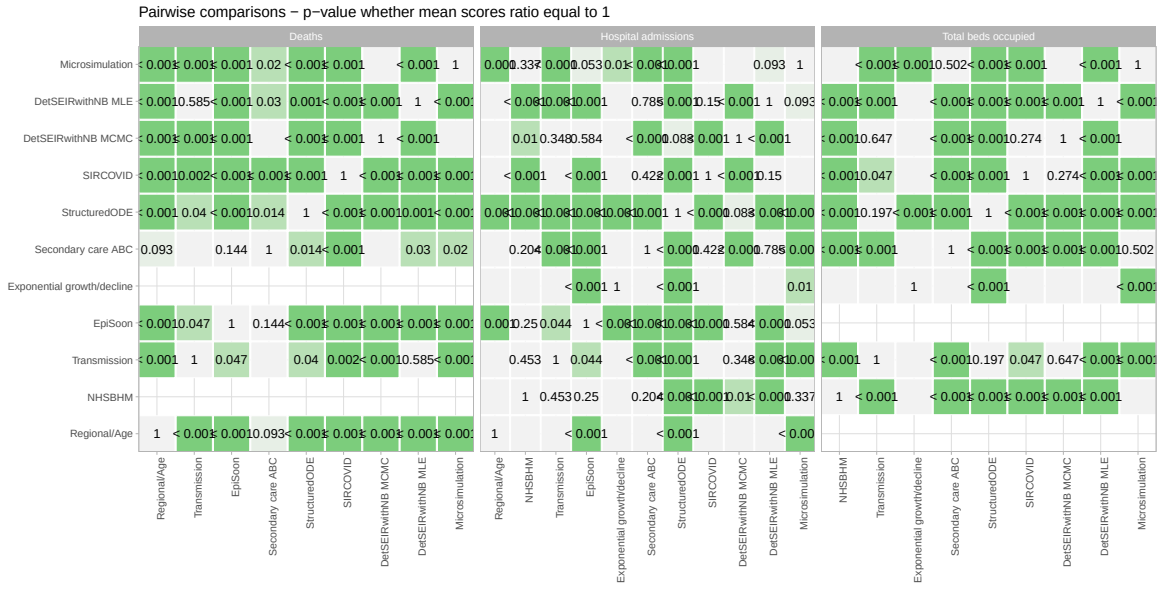


Figure 7: XXXX

Acknowledgments

References

Bracher J, Ray EL, Gneiting T, Reich NG (2020). "Evaluating epidemic forecasts in an interval format." *arXiv:2005.12881 [q-bio, stat]*. ArXiv: 2005.12881, URL http://arxiv.org/abs/2005.12881.

Dawid AP (1984). "Present Position and Potential Developments: Some Personal Views Statistical Theory the Prequential Approach." *Journal of the Royal Statistical Society: Series A (General)*, **147**(2), 278–290. ISSN 2397-2327. doi:10.2307/2981683.

Funk S, Abbott S, Atkins BD, Baguelin M, Baillie JK, Birrell P, Blake J, Bosse NI, Burton J, Carruthers J, Davies NG, Angelis DD, Dyson L, Edmunds WJ, Eggo RM, Ferguson NM, Gaythorpe K, Gorsich E, Guyver-Fletcher G, Hellewell J, Hill EM, Holmes A, House TA, Jewell C, Jit M, Jombart T, Joshi I, Keeling MJ, Kendall E, Knock ES, Kucharski AJ, Lythgoe KA, Meakin SR, Munday JD, Openshaw PJM, Overton CE, Pagani F, Pearson J, Perez-Guzman PN, Pellis L, Scarabel F, Semple MG, Sherratt K, Tang M, Tildesley MJ, Leeuwen EV, Whittles LK (2020). "Short-term forecasts to inform the response to the Covid-19 epidemic in the UK." *medRxiv*, p. 2020.11.11.20220962. doi:10.1101/2020.11.11.20220962. Publisher: Cold Spring Harbor Laboratory Press, URL https://www.medrxiv.org/content/10.1101/2020.11.11.20220962v1.

```
R> wis_components(scores,
+                 facet_formula = ~ value_desc,
+                 scales = "free_x") +
+    ggplot2::coord_flip() +
+    ggplot2::theme(legend.position = "bottom")
```

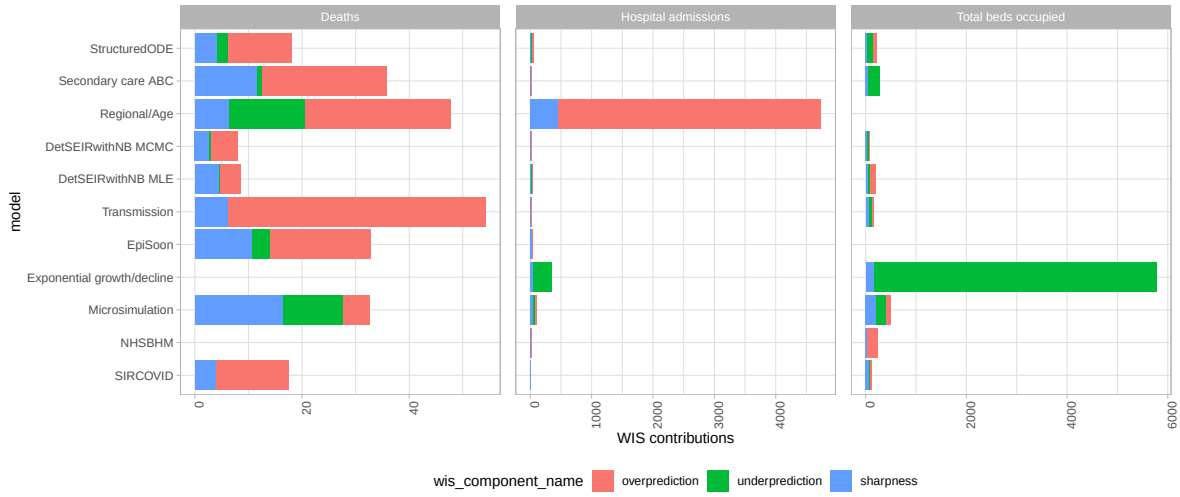


Figure 8: p-values and ratios together.

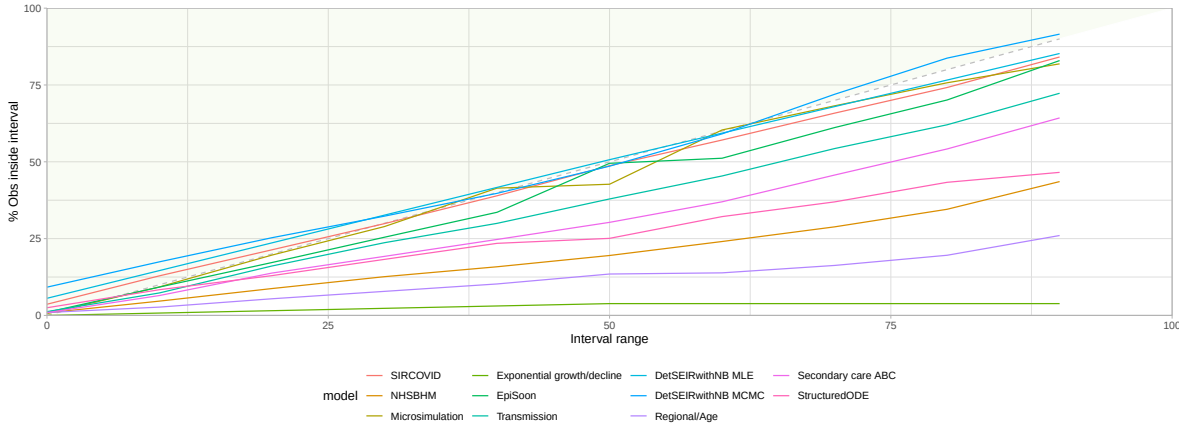
Funk S, Camacho A, Kucharski AJ, Lowe R, Eggo RM, Edmunds WJ (2019). "Assessing the Performance of Real-Time Epidemic Forecasts: A Case Study of Ebola in the Western Area Region of Sierra Leone, 2014-15." *PLOS Computational Biology*, **15**(2), e1006785. ISSN 1553-7358. doi:10.1371/journal.pcbi.1006785.

Gneiting T, Balabdaoui F, Raftery AE (2007). "Probabilistic forecasts, calibration and sharpness." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **69**(2), 243–268. ISSN 1467-9868. doi:10.1111/j.1467-9868.2007.00587.x. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2007.00587.x.

Gneiting T, Raftery AE (2007). "Strictly Proper Scoring Rules, Prediction, and Estimation." *Journal of the American Statistical Association*, **102**(477), 359–378. ISSN 0162-1459, 1537-274X. doi:10.1198/016214506000001437. URL http://www.tandfonline.com/doi/abs/10.1198/016214506000001437.

Winkler RL, Muñoz J, Cervera JL, Bernardo JM, Blattenberger G, Kadane JB, Lindley DV, Murphy AH, Oliver RM, Ríos-Insua D (1996). "Scoring Rules and the Evaluation of Probabilities." *Test*, **5**(1), 1–60. ISSN 1863-8260. doi:10.1007/BF02562681.

```
R> cov_scores <- eval_forecasts(combined,
+                                summarise_by = c("model",
+                                                 "range", "quantile"))
R> scoringutils::interval_coverage(cov_scores)
R> scoringutils::quantile_coverage(cov_scores)
```




```
R> cov_scores <- eval_forecasts(combined,
+                                summarise_by = c("model",
+                                                 "range", "quantile"))
R> scoringutils::quantile_coverage(cov_scores)
```

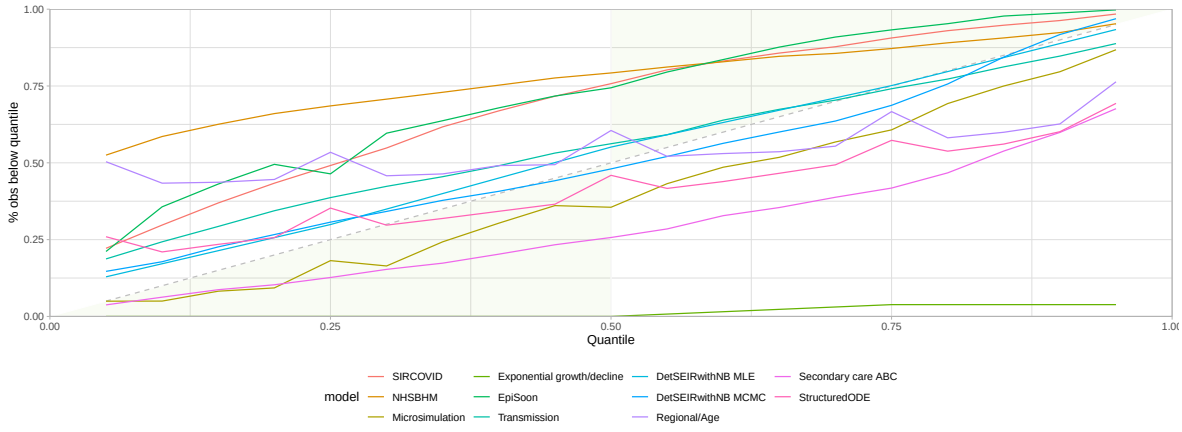


Figure 9: quantile and interval coverage

A. Appendix section

```
R> scores <- eval_forecasts(combined,
+                           summarise_by = c("model",
+                                            "value_desc",
+                                            "geography"),
+                           compute_relative_skill = FALSE)
R> scoringutils::score_heatmap(scores, metric = "bias",
+                              x = "geography", facet_formula = ~ value_desc,
+                              scale = "free_x")
```

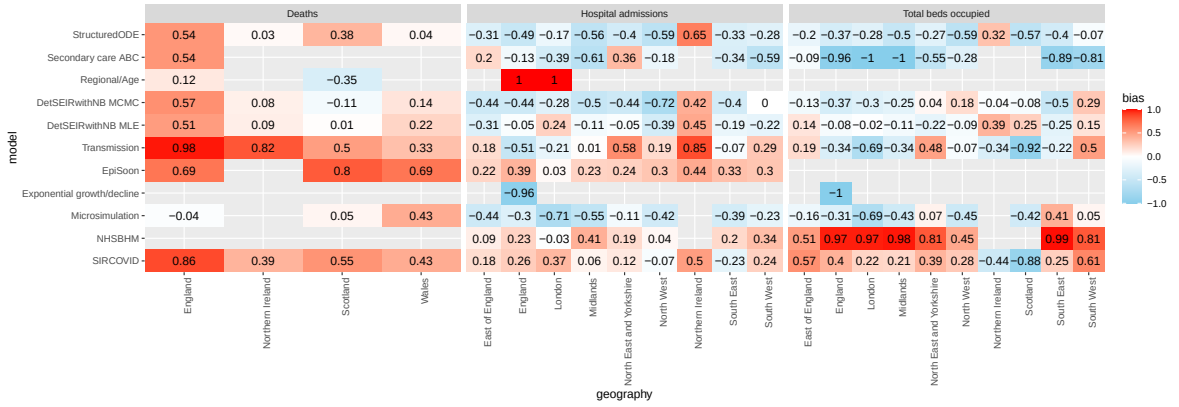


Figure 10: bias by location

Affiliation:

Nikos Bosse
Centre for Mathematical Modelling of Infectious Diseases
London School of Hygiene and Tropical Medicine
Keppel Street
London WC1E 7HT
E-mail: nikos.bosse@lshtm.ac.uk

Metric	Target types	Forecast formats	Properties
CRPS (Continuous) ranked probability score	continuous, integer	closed-form, samples (approximation)	proper scoring rule, global, stable handling of outliers
Log score	continuous, (integer not in scoringutils)	closed-form, samples (approximation)	proper scoring rule, local, unstable for outliers
WIS (Weighted) interval score	continuous, integer	quantile or interval predictions	proper scoring rule, global, stable handling of outliers, converges to crps
DSS Dawid-Sebastiani score	continuous, integer	closed-form, samples (approximation)	proper scoring rule, somewhat global, somewhat stable handling of outliers
Brier score	binary	binary probabilities	proper scoring rule
Interval coverage	continuous, integer	interval forecasts (needs matching quantiles)	measure for calibration
Quantile coverage	continuous, integer	quantile or interval forecasts	measure for calibration
Probability integral transform (PIT)	continuous, integer, quantile	closed-form, samples, quantile or interval forecasts	assesses calibration
Sharpness	continuous, integer	closed-form, samples, quantile or interval forecasts	measures sharpness, slightly different depending on forecast format
Bias	continuous, integer, quantile	closed-form, samples, quantile or interval forecasts	captures tendency to over-or underpredict (aspect of calibration)
Mean score ratio	depends on score	depends on score	compares performance of two models
Relative skill	depends on scored	depends on score	Ranks models based on pairwise comparisons

Table 4: Summary table of scores available in scoringutils

Table 5: Explanation of all the scores