Proper scoring rules for prediction assessment

Finn Lindgren

January 30, 2026

Summary

These notes introduce the concept of *proper scoring rules* for assessing probabilistic predictions/forecasts. These methods can be used to expand model comparisons beyond point predictions in a systematic way, by also taking into account the estimated uncertainty in the predictions.

A generalised regression example is used for illustration, including R code for generating the model estimates, predictions, scores, and illustrative figures, followed by an overview of the most practically relevant aspects of the theory of proper scoring rules. The theory sections are mostly inspired by Gneiting and Raftery, *Strictly proper scoring rules, prediction, and estimation* (JASA, 2007).

The notes are based on lectures developed for the Edinburgh course on *Statistical Computing*, ca 2020, and is work in progress.

Contents

1 Predictive distributions

We are interested in estimating a parameter vector $\boldsymbol{\theta} \in \mathbb{R}^p$, given observed values $\mathcal{Y}_{\text{obs}} = \{y_i, i = 1, \ldots, n\}$, for some likelihood model $L(\boldsymbol{\theta}; \mathcal{Y}_{\text{obs}})$, and then use the estimated model to *forecast* or *predict* the values of some *test data* $\mathcal{Y}_{\text{test}}$. By producing not only point predictions but also quantifying the prediction uncertainty due to inherent randomness and parameter estimation error uncertainty, e.g. in the form of a full predictive distribution or just a prediction variance, we can then compare different models and estimation methods in terms of how good their assessment *scores* are.

To simplify the notation for tracking the parameter estimation uncertainty, after conditioning on the observed data, we will treat the unknown parameter vector $\boldsymbol{\theta}$ as a random vector which is approximately

$$\boldsymbol{\theta} \sim \text{N}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_\theta) \tag{1}$$

for some point estimate $\widehat{\boldsymbol{\theta}}$ and covariance matrix $\boldsymbol{\Sigma}_\theta$. This way, once we have found $\widehat{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}_\theta$, the model assessment only relies on this approximation, in combination with the conditional distributions for the observations, given the parameter values.

Remark 1. *In classical* frequentist statistics*, this is a slight abuse of notation, since the true $\boldsymbol{\theta}$ value, $\boldsymbol{\theta}_{true}$, is not random in that setting. However, likelihood theory can be used to show that, for large observation samples, the maximum likelihood estimation error is approximately Normal,*

$$\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}_{true} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_\theta) \tag{2}$$

where $\boldsymbol{\Sigma}_\theta^{-1} = H(\widehat{\boldsymbol{\theta}})$, the Hessian of the negative log-likelihood, is a plug-in estimate of the error covariance matrix.

On the other hand, in Bayesian statistics *treating the unknown value as a random variable conditionally on the observations is directly the valid approach, where the Bernstein-von Mises theorem shows that the asymptotic behaviour of the posterior distribution for $\boldsymbol{\theta}$ is Gaussian, under mild assumption on the observation mechanism and prior density for $\boldsymbol{\theta}$. The general results discussed in these notes hold for general Bayesian posterior predictions, but some of the examples rely on the Normal approximation for simplicity and computational tractability.*

Given the distribution properties of an test observation y given $\boldsymbol{\theta}$, e.g. as a probability density $f_{y|\boldsymbol{\theta}}(\cdot)$, we can write the full predictive/forecast density as

$$f_y(y) = \int_{\mathbb{R}} f_{y|\boldsymbol{\theta}}(y) f_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}, \tag{3}$$

where $f_{\boldsymbol{\theta}}(\boldsymbol{\theta})$ is the density of the parameter error uncertainty model, $\text{N}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_\theta)$. In general, this integral might be difficult to evaluate, but there are special cases where the answer is known.

We will in general identify the notation F with both the abstract *predictive distribution* and the concrete *predictive cumulative distribution function*, $F(x) = \text{P}(y \leq x) = \int_{-\infty}^{x} f_y(y) \, \mathrm{d}y$.

1.1 General notation and predictive moments

Working with the full predictive distribution is often difficult, so we here focus on Normal/Gaussian predictive distributions, or predictions that only involve the 1st and 2nd order moments of the predictive distribution.

Using the expectation *tower property* (also known as the *law of total expectation*), $\text{E}_A(A) = \text{E}_A[\text{E}_{A|B}(A)]$, we can write

$$\mu_F = \text{E}_F(y) = \text{E}_\theta[\text{E}_{y|\theta}(y)] \tag{4}$$

$$\sigma_F^2 = \text{Var}_F(y) = \text{E}_\theta[\text{Var}_{y|\theta}(y)] + \text{Var}_\theta[\text{E}_{y|\theta}(y)] \tag{5}$$

where the second line is the tower property for variances, also known as the *law of total variance*, $\text{Var}_A(A) = \text{E}_A[\text{Var}_{A|B}(A)] + \text{Var}_A[\text{E}_{A|B}(A)]$. These identities often provide practical solutions to finding the forcast/prediction mean and standard deviations, μ_F and σ_F.

1.2 Example: non-constant variance

In basic linear regression models with additive Gaussian noise, for simplicity the noise variance is often chosen to be the same for all observations. This leads to useful theoretical properties, but in practice, this may not be a realistic assumption. This section considers an extended model, where the logarithm of the variance is also allowed to follow a linear model.

1.2.1 General model definition

We define a model where $(y|\boldsymbol{\theta})$ are independent and Normal/Gaussian, and the expectation and log-variance are linear in $\boldsymbol{\theta}$,

$$\mathrm{E}_{y|\theta}(y) = \boldsymbol{z}_E^\top \boldsymbol{\theta} \tag{6}$$

$$\log[\mathrm{Var}_{y|\theta}(y)] = \boldsymbol{z}_V^\top \boldsymbol{\theta} \tag{7}$$

The $\boldsymbol{z}_E$ and $\boldsymbol{z}_V$ vectors can be stacked as rows of model matrices $\boldsymbol{Z}_E$ and $\boldsymbol{Z}_V$, so that the expectation of a vector of observations can be written $\boldsymbol{Z}_E\boldsymbol{\theta}$.

Combining the conditional moments for $(y|\boldsymbol{\theta})$ with the uncertainty model for $\boldsymbol{\theta}$, we obtain

$$\mathrm{E}_F(y) = \boldsymbol{z}_E^\top \widehat{\boldsymbol{\theta}} \tag{8}$$

$$\mathrm{Var}_F(y) = \mathrm{E}_\theta\left[\exp\left(\boldsymbol{Z}_V\boldsymbol{\theta}\right)\right] + \mathrm{Var}_\theta\left(\boldsymbol{z}_E^\top\boldsymbol{\theta}\right) \tag{9}$$

The second term of the variance is

$$\mathrm{Var}_\theta\left(\boldsymbol{z}_E^\top\boldsymbol{\theta}\right) = \mathrm{Cov}_\theta\left(\boldsymbol{z}_E^\top\boldsymbol{\theta}, \boldsymbol{z}_E^\top\boldsymbol{\theta}\right) \tag{10}$$

$$= \boldsymbol{z}_E^\top \mathrm{Cov}_\theta\left(\boldsymbol{\theta},\boldsymbol{\theta}\right)\boldsymbol{z}_E \tag{11}$$

$$= \boldsymbol{z}_E^\top \boldsymbol{\Sigma}_\theta \boldsymbol{z}_E. \tag{12}$$

The first term of the variance is more difficult. We will use the known result (either from the *log-Normal distribution* or the *moment generating function* for the Normal distribution) that if $x \sim \mathrm{N}(\mu, \sigma^2)$, then $\mathrm{E}(\mathrm{e}^x) = \mathrm{e}^{\mu+\sigma^2/2}$:

$$\boldsymbol{z}_V^\top\boldsymbol{\theta} \sim \mathrm{N}(\boldsymbol{z}_V^\top\widehat{\boldsymbol{\theta}}, \boldsymbol{z}_V^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_V) \tag{13}$$

$$\mathrm{E}_\theta[\exp(\boldsymbol{z}_V^\top\boldsymbol{\theta})] = \exp\left(\boldsymbol{z}_V^\top\widehat{\boldsymbol{\theta}} + \boldsymbol{z}_V^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_V/2\right). \tag{14}$$

Combining the results, we get the predictive variance as

$$\sigma_V^2 = \mathrm{Var}_F(y) = \exp\left(\boldsymbol{z}_V^\top\widehat{\boldsymbol{\theta}} + \boldsymbol{z}_V^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_V/2\right) + \boldsymbol{z}_E^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_E. \tag{15}$$

1.2.2 Example model definition

We implement a simple version of the general model, by requiring each θ_i parameter to be used for either the expectation or the log-variance, but not both. We also use a single covariate, in the code called `x`, so that every observation is a pair (x_i, y_i), and $\mathrm{E}_{y|\theta}(y_i) = \theta_1 + x_i\theta_2$ and $\log[\mathrm{Var}_{y|\theta}(y_i)] = \theta_3 + x_i\theta_4$.

First, define a function that constructs the $\boldsymbol{Z}$ matrices, for three possible model choices:

```r
model_Z <- function(x, model) {
  model <- match.arg(model, c("ConstEV", "ConstV", "Full"))
  Z0 <- model.matrix(~ 1 + x)
  if (model == "ConstEV") {
    # Constant mean and variance
    list(ZE = cbind(Z0[, 1], 0), ZV = cbind(0, Z0[, 1]))
  } else if (model == "ConstV") {
    # Linear mean, constant variance
    list(ZE = cbind(Z0, 0), ZV = cbind(Z0 * 0, Z0[, 1]))
  } else {
    # Linear mean, log-linear variance
    list(ZE = cbind(Z0, Z0 * 0), ZV = cbind(Z0 * 0, Z0))
  }
}
```

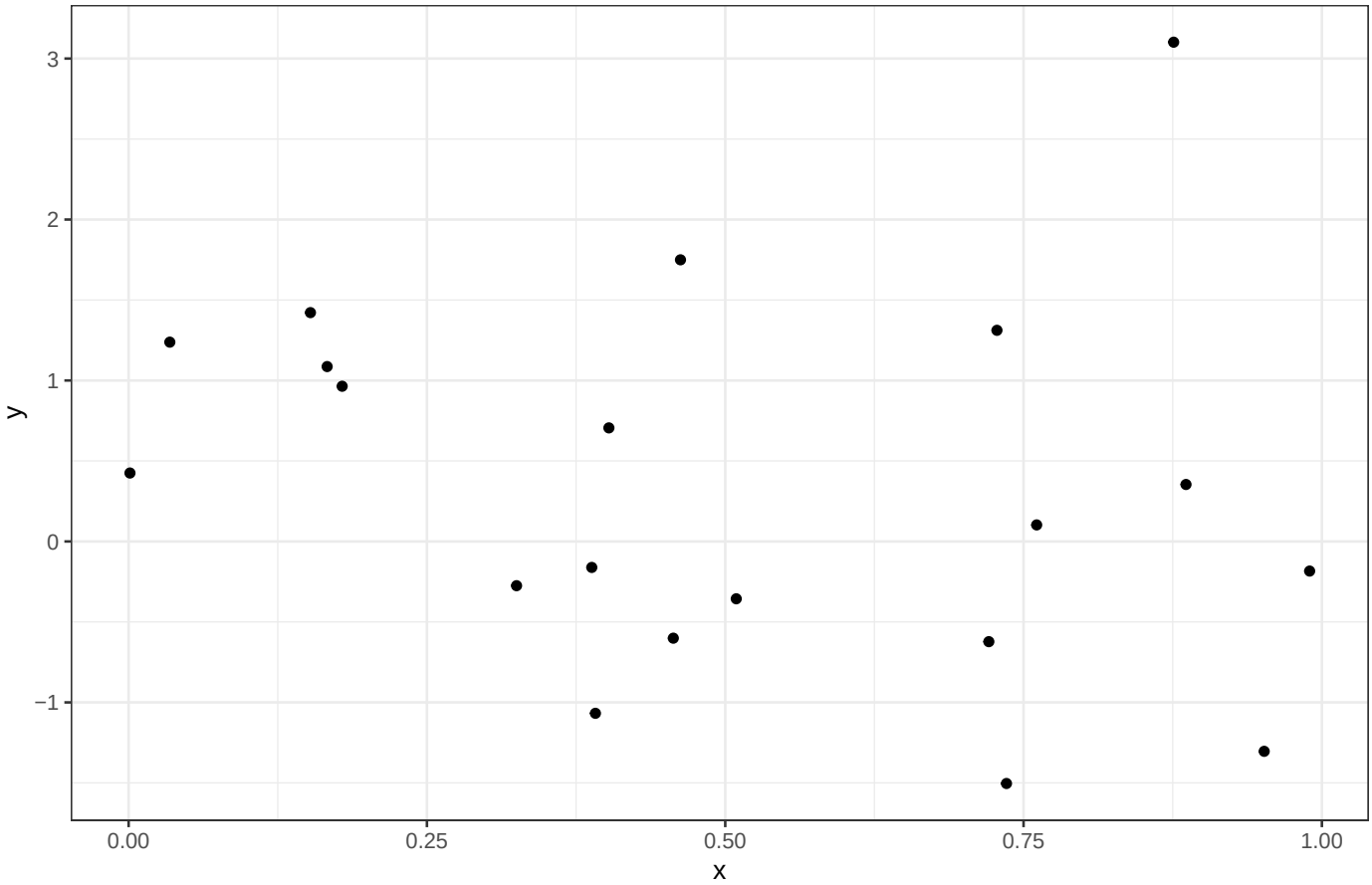
We will write the rest of the code so that we essentially only need to change the definition of `model_Z()` to run a different model of the same general class.

Then, a function that implements the general version of the negative log-likelihood, using a list of the type generated by `model_Z()` to known what the model is.

```r
neg_log_lik <- function(theta, Z, y) {
  -sum(dnorm(y, mean = Z$ZE %*% theta, sd = exp(Z$ZV %*% theta)^0.5, log = TRUE))
}
```

In order to have something to test, we generate a synthetic data sample:

```r
model_simulate <- function(x, model, theta) {
  Z <- model_Z(x, model = model)
  data.frame(x = x, y = rnorm(n = length(x), mean = Z$ZE %*% theta, sd = exp(Z$ZV %*% theta)^0.5))
}
n <- 20
x_obs <- runif(n)
Z_obs <- model_Z(x_obs, model = "Full")
theta_true <- c(1, -2, -2, 4)
data_obs <- model_simulate(x = x_obs, model = "Full", theta = theta_true)
ggplot(data_obs) + geom_point(aes(x, y))
```



Treating the simulated data as our observed sample, we estimate the parameter vector $\boldsymbol{\theta}$ using `optim()`:

```
model_estimate <- function(model, data) {
  Z <- model_Z(data$x, model)
  opt <- optim(rep(0, ncol(Z$ZE)), fn = neg_log_lik, Z = Z, y = data$y,
               method = "BFGS", hessian = TRUE)
  theta_hat <- opt$par
  Sigma_theta <- solve(opt$hessian)
  list(theta = theta_hat, Sigma_theta = Sigma_theta, model = model)
}

fit <- list(
  True = list(theta = theta_true, Sigma_theta = NULL, model = "Full"),
  Const = model_estimate("ConstEV", data_obs),
  Linear = model_estimate("ConstV", data_obs),
  Full = model_estimate("Full", data_obs)
)
```

Next, we define a function with behaviour similar to `predict()`, that we'll use to compute predictive distributions and predic-

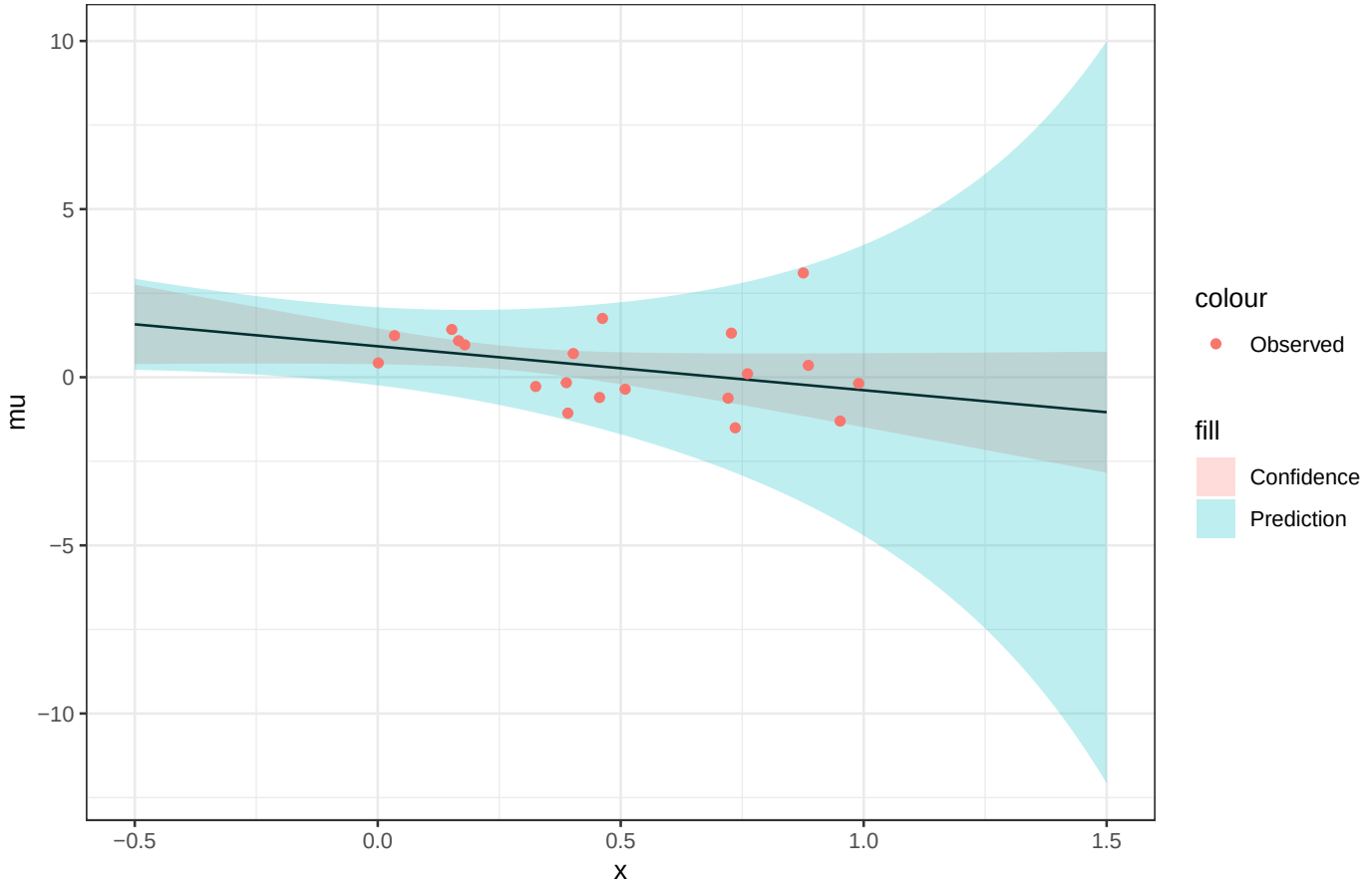
tion intervals:

```
# Value: data.frame with columns (mu, sigma, lwr, upr)
model_predict <- function(fit,
                          data,
                          type = c("expectation", "log-variance", "observation"),
                          alpha = 0.05, df = Inf,
                          nonlinear.correction = TRUE) {
  type <- match.arg(type)
  Z <- model_Z(data$x, model = fit$model)
  fit_E <- Z$ZE %*% fit$theta
  fit_V <- Z$ZV %*% fit$theta
  if (is.null(fit$Sigma_theta)) {
    ZE_var <- 0
    ZV_var <- 0
  } else {
    ZE_var <- rowSums(Z$ZE * (Z$ZE %*% fit$Sigma_theta))
    ZV_var <- rowSums(Z$ZV * (Z$ZV %*% fit$Sigma_theta))
  }
  if (type == "expectation") {
    mu_fit <- fit_E
    sigma <- ZE_var^0.5
  } else if (type == "log-variance") {
    mu_fit <- fit_V
    sigma <- ZV_var^0.5
  } else if (type == "observation") { ## observation predictions
    mu_fit <- fit_E
    sigma <- (exp(fit_V + ZV_var / 2 * nonlinear.correction) + ZE_var)^0.5
  }
  q <- qt(1 - alpha / 2, df = df)
  lwr <- mu_fit - q * sigma
  upr <- mu_fit + q * sigma
  tmp <- data.frame(mu = mu_fit, sigma, lwr, upr)
  tmp
}
```

Now, let's plot the estimates and predictions!

```
x_plot <- data.frame(x = seq(-0.5, 1.5, length=100))
conf_plot <-
  cbind(x_plot,
        model_predict(fit$Full, x_plot, type = "expectation"))
pred_plot <-
  cbind(x_plot,
        model_predict(fit$Full, x_plot, type = "observation"))
pl <- ggplot() +
  geom_ribbon(data = conf_plot,
              aes(x, ymin = lwr, ymax = upr, fill = "Confidence"),
              alpha = 0.25) +
  geom_line(data = conf_plot, aes(x, mu), col = "black") +
```

```
  geom_ribbon(data = pred_plot,
              aes(x, ymin = lwr, ymax = upr, fill = "Prediction"),
              alpha = 0.25) +
  geom_point(aes(x, y, col = "Observed"), data = data_obs)
pl
```



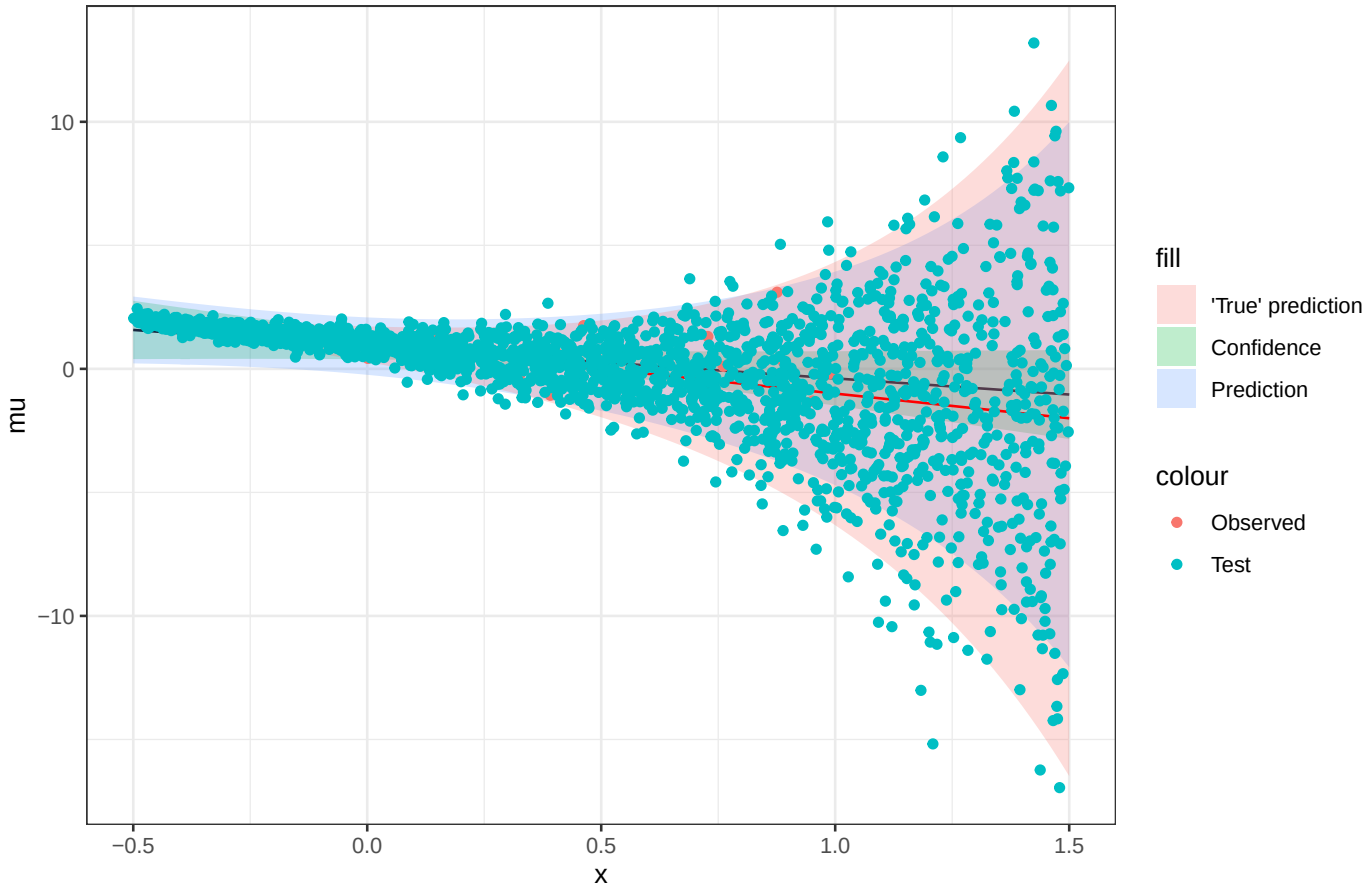
1.2.3 Towards model assessment with test data

We want to assess how good the estimated model is at predicting unseen data; i.e. data that wasn't used when estimating the model parameters. We simulate some new test data from the true model (in real data, this would have been a held-out part of the raw observed data):

```
N_test <- 2000
data_test <- model_simulate(x = runif(N_test, -0.5, 1.5),
                            model = "Full", theta = theta_true)
```

Let's do a visual inspection:

```
true_plot <-
  cbind(x_plot,
        model_predict(fit$True, x_plot, type = "observation"))
pl +
  geom_ribbon(data = true_plot,
              aes(x, ymin = lwr, ymax = upr, fill = "'True' prediction"),
              alpha = 0.25) +
    geom_line(data = true_plot,
              aes(x, mu),
              col = "red") +
  geom_point(aes(x, y, col = "Test"), data = data_test)
```



We see that the estimated model (black) is close to the true model (red), and that the nonlinear uncertainty contribution from the exponential variance gives an important contribution.

The next step is to introduce more formal and quantifiable assessment techniques in the form of *proper scoring rules*.

2 Scoring rules

We want to assess how *far away* from the truth our forecast/prediction distributions are. To do this, we might consider a number of quantities that measure the discrepancy in different ways.

2.1 Common scoring rules

2.1.1 Scoring rules for continuous outcomes

- Squared Error (SE):

$$S_{\text{SE}}(F, y) = (y - \widehat{y}_F)^2 \tag{16}$$

  where $\widehat{y}_F$ is a point estimate under F, e.g. the expectation μ_F.

- Absolute Error (AE):

$$S_{\text{AE}}(F, y) = |y - \widehat{y}_F| \tag{17}$$

  where $\widehat{y}_F$ is a point estimate under F, e.g. the predictive median $F^{-1}(1/2)$.

- Logarithmic/Ignorance score (LOG/IGN):

$$S_{\text{LOG}}(F, y) = -\log f(y) \tag{18}$$

  where $f(\cdot)$ is the predictive density.

- Dawid-Sebastiani score (DS):

$$S_{\text{DS}}(F, y) = \frac{(y - \mu_F)^2}{\sigma_F^2} + \log(\sigma_F^2) \tag{19}$$

  Note: If F is Normal, then $S_{\text{DS}}(F, y) = 2S_{\text{LOG}}(F, y) - \log(2\pi)$

- Continuous Ranked Probability Score (CRPS):

$$S_{\text{CRPS}}(F, y) = \int_{\mathbb{R}} [F(x) - \mathbb{I}(y \leq x)]^2 \, \mathrm{d}x \tag{20}$$

  This can be seen as a generalisation of AE that also cares about other quantiles than the median.

These scores are defined to be *negatively oriented*, meaning that the *lower the score, the better*.

For discrete variables, other scores are also used, such as the Brier score for binary outcomes. We will discuss the binary prediction case further in Section 4.

2.2 Defining scores in R

Thinking back at the output from the example `model_predict()`, we can define R functions that mimic the $S(F, y)$ notation. For SE and DS we define `score_se()` and `score_ds()`:

```r
# Input:
#   pred : data.frame with (at least) a column "mu"
#   y : data vector
score_se <- function(pred, y) {
  (y - pred$mu)^2
}
# Input:
#   pred : data.frame with (at least) columns "mu" and "sigma"
#   y : data vector
score_ds <- function(pred, y) {
  ((y - pred$mu) / pred$sigma)^2 + 2 * log(pred$sigma)
}
```

2.3 Evaluating scores

We can now evaluate the scores for the example. We include the scores for a simplistic model that assumes that all the observations have a common expectation and variance, $(y_i|\boldsymbol{\theta}) \sim \mathrm{N}(\beta_0, \sigma_\epsilon^2)$ (for brevity, we ignore some of the parameter uncertainty when constructing the prediction from this model, in `pred0`).

```r
data <- rbind(
  cbind(Data = "Est", data_obs),
  cbind(Data = "Test", data_test)
)
pred <- lapply(names(fit),
               function(x) {
                 pred_data <- model_predict(fit[[x]], data, type = "observation")
                 cbind(
                   data,
                   pred_data,
                   Model = x
                 )
               })
pred <- do.call(rbind, pred)

scores <- rbind(
  pred |> mutate(ScoreType = "SE", Score = score_se(data.frame(mu, sigma), y)),
  pred |> mutate(ScoreType = "DS", Score = score_ds(data.frame(mu, sigma), y))
)
score_summaries <- scores |>
  group_by(Model, Data, ScoreType) |>
  summarise(Score = mean(Score), .groups = "drop")
```

```r
score_summaries |>
  select(Model, Data, ScoreType, Score) |>
  pivot_wider(names_from = Model, values_from = Score) |>
  arrange(ScoreType) |>
  kableExtra::kbl(digits = 3)
```

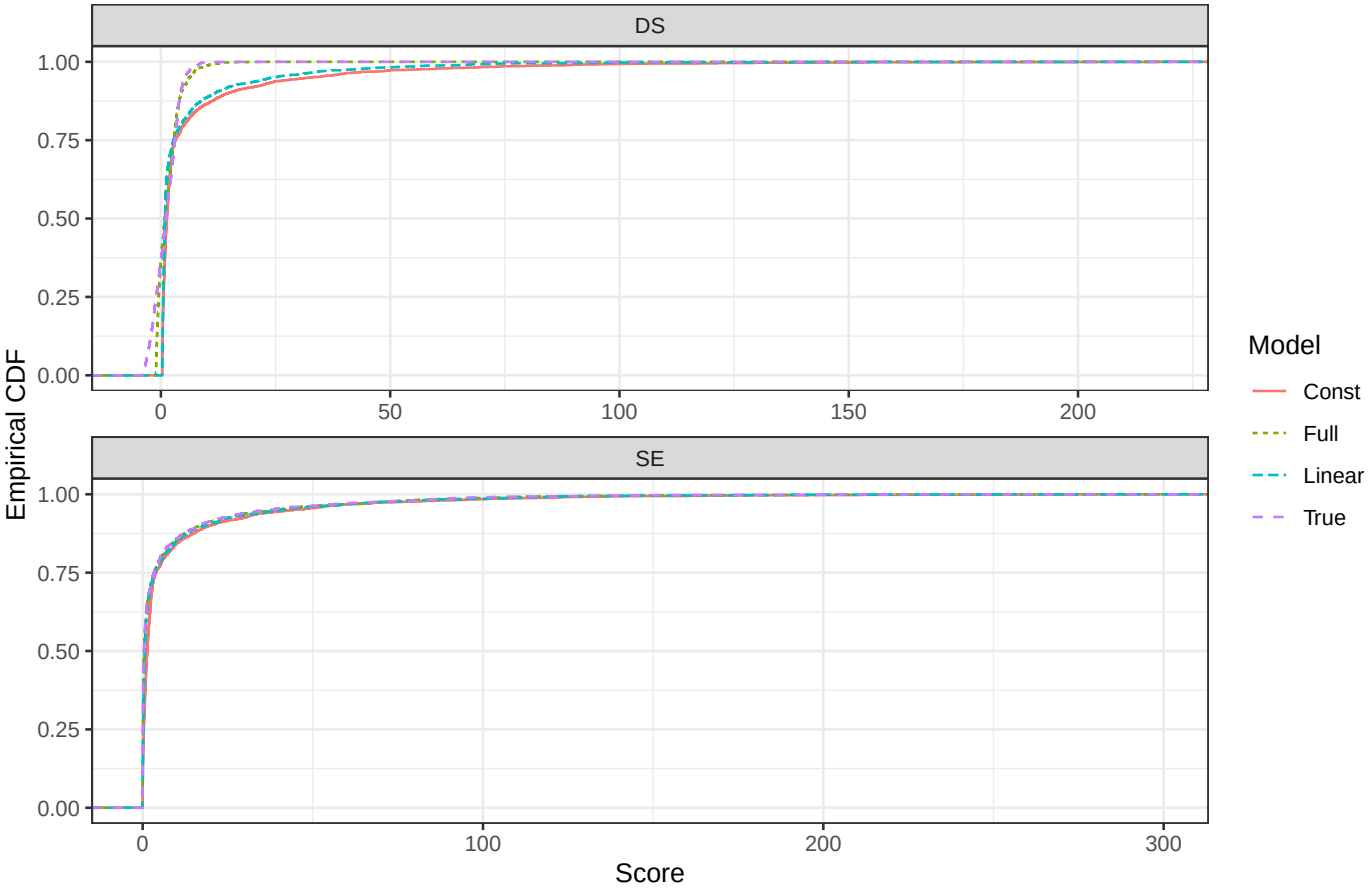
Data	ScoreType	Const	Full	Linear	True
Est	DS	1.226	0.943	1.183	1.002
Test	DS	6.352	1.510	4.821	1.082
Est	SE	1.248	1.216	1.189	1.430
Test	SE	8.291	6.990	7.386	6.684

We see that the SE appears to be less sensitive to model mis-specification than DS. We also see that the scores for the true model are generally *worse* than for the estimated full model when applied to the observed data. This is because the estimated models are adapted to the random values that happened to be observed. For the test data, the mis-specified models (”Const” and ”Linear”) are clearly worse than the full estimated models as well as the true model.

In general, one wants to use many observations for parameter estimation, but at the same time have enough test observations for reliable model assessment.

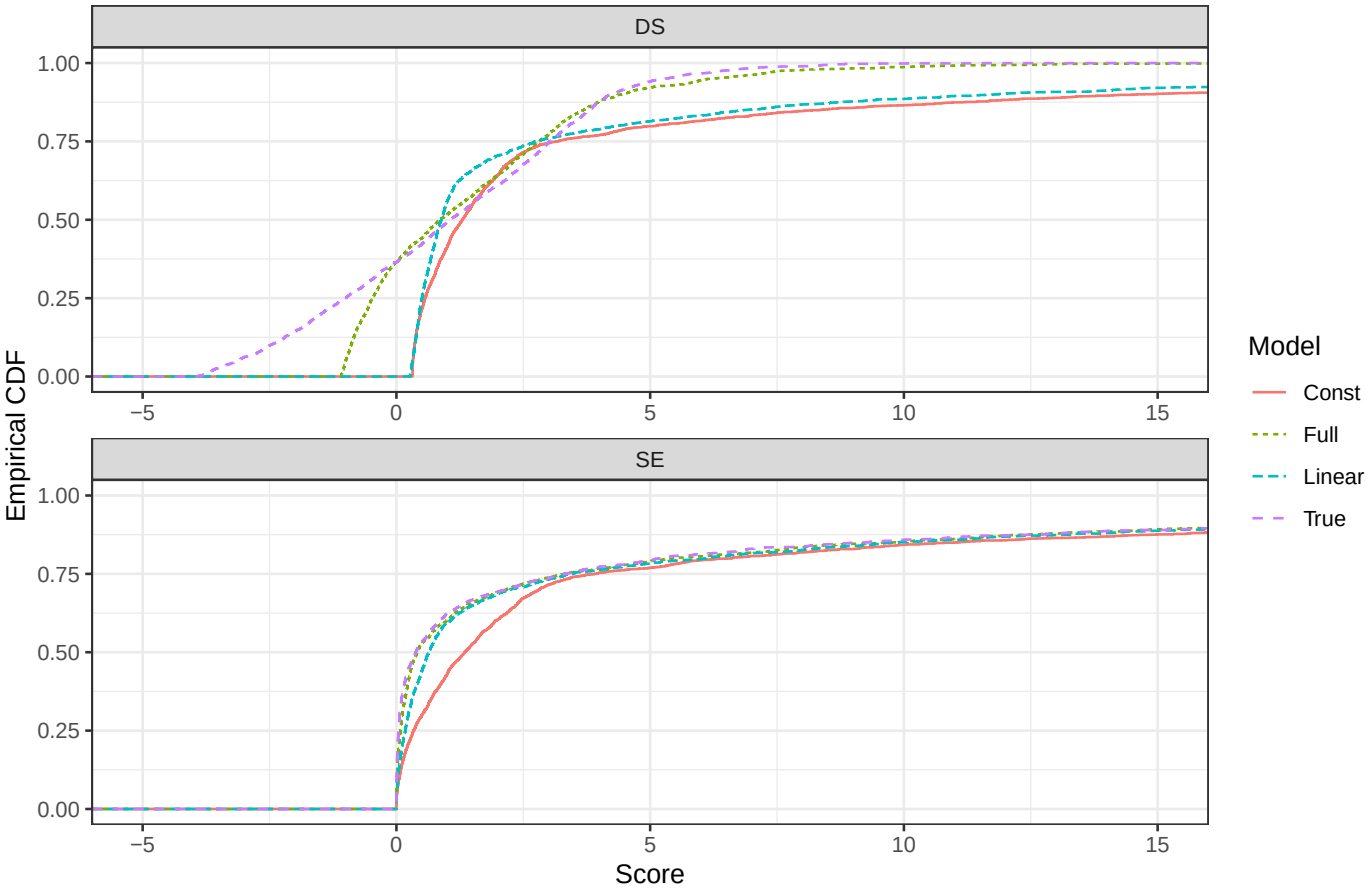
We can also plot the individual scores, to get an overview of the contributions to the average scores:

```
ggplot(scores |> filter(Data == "Test")) +
  stat_ecdf(aes(x = Score, col = Model, linetype = Model)) +
  xlab("Score") + ylab("Empirical CDF") +
  facet_wrap(~ ScoreType, scales = "free_x", ncol = 1)
```



In particular for the naive reference model, a few observations are acting as outliers, contributing most of the score penalty. Let's draw the same plot but limit the Score axis values:

```
ggplot(scores |> filter(Data == "Test")) +
  stat_ecdf(aes(x = Score, col = Model, linetype = Model)) +
  xlab("Score") + ylab("Empirical CDF") +
  facet_wrap(~ ScoreType, scales = "free_x", ncol = 1) +
  coord_cartesian(xlim = c(-5, 15))
```

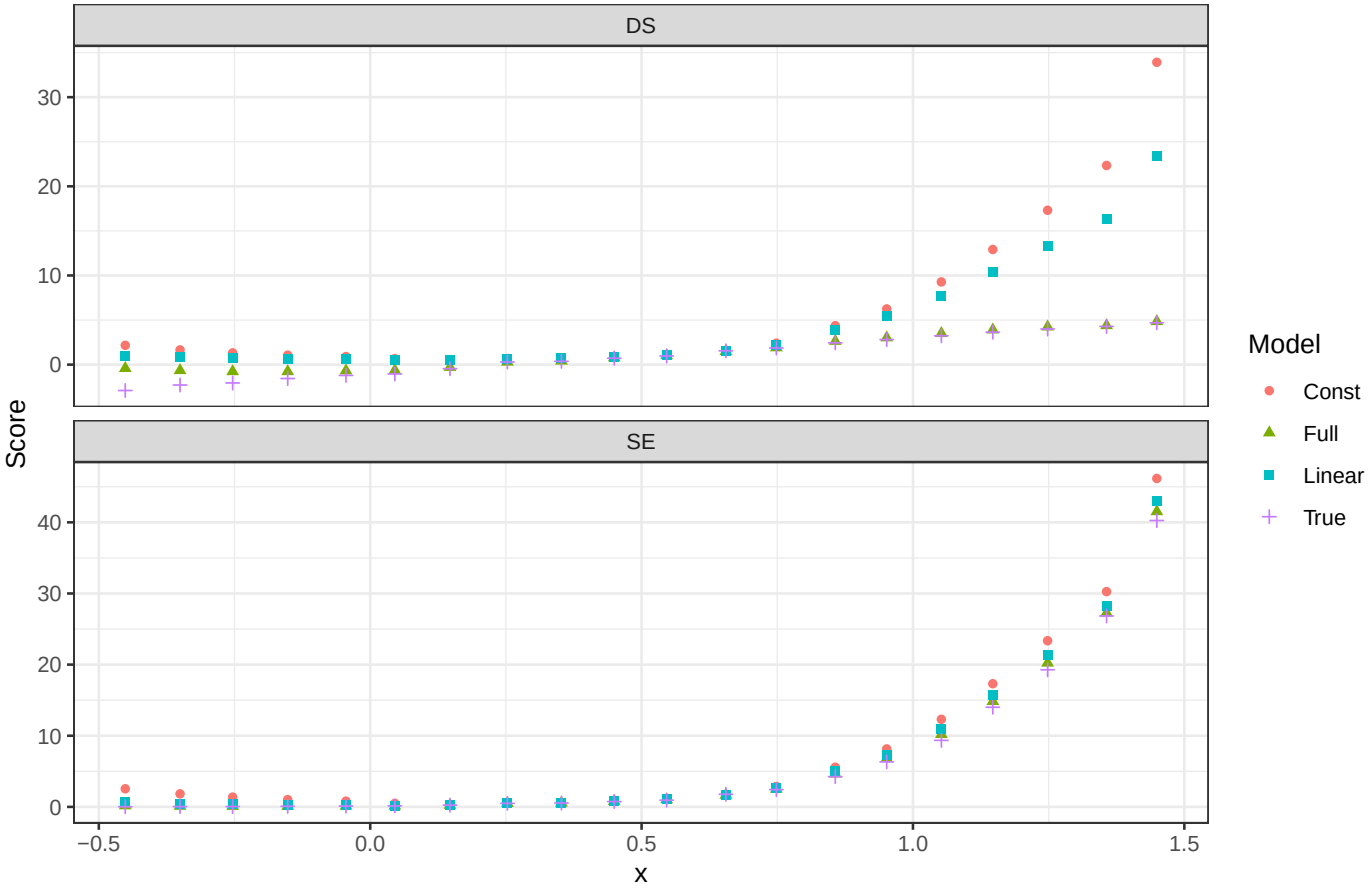


These CDF plots show a consistent pattern for SE, but the picture is more complicated for DS. It's clear that the "True" predictions generally do better. However, the aggregated CDFs are implictly based on the assumption that the observations are *exchangeable*, which isn't true, since the distribution of y depends on x. To make matters worse, even the residuals are non-exchangeable, as the variance also depends on x.

To alleviate this issue, we might compare the scores as functions of the regression input, x:

```
ggplot(scores |>
         filter(Data == "Test") |>
         group_by(ScoreType, Model, floor(x*10)) |>
```

```
        summarise(x = mean(x),
                  Score = mean(Score),
                  .groups = "drop")) +
  geom_point(aes(x, Score, col = Model, shape = Model)) +
  facet_wrap( ~ ScoreType, ncol = 1, scales = "free_y")
```

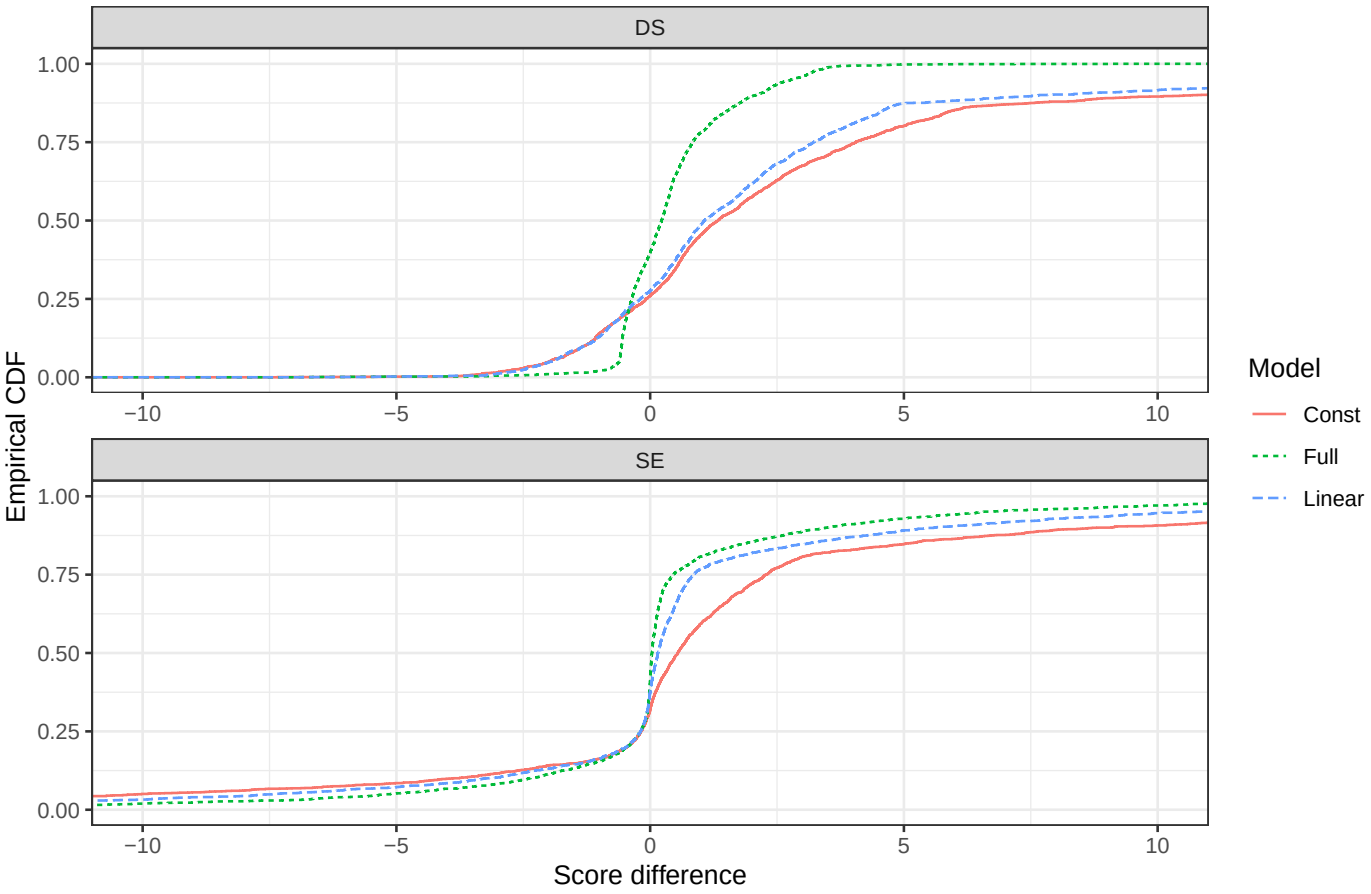


This shows clearly how the models systematically deviate from the true model, and that the DS score more clearly provides information about what is going wrong for the mis-specified models.

In the next section, we will look more closely at the properties of scores, and provide a formal framework for comparing prediction scores that is valid despite the issues with non-exchangeable observations.

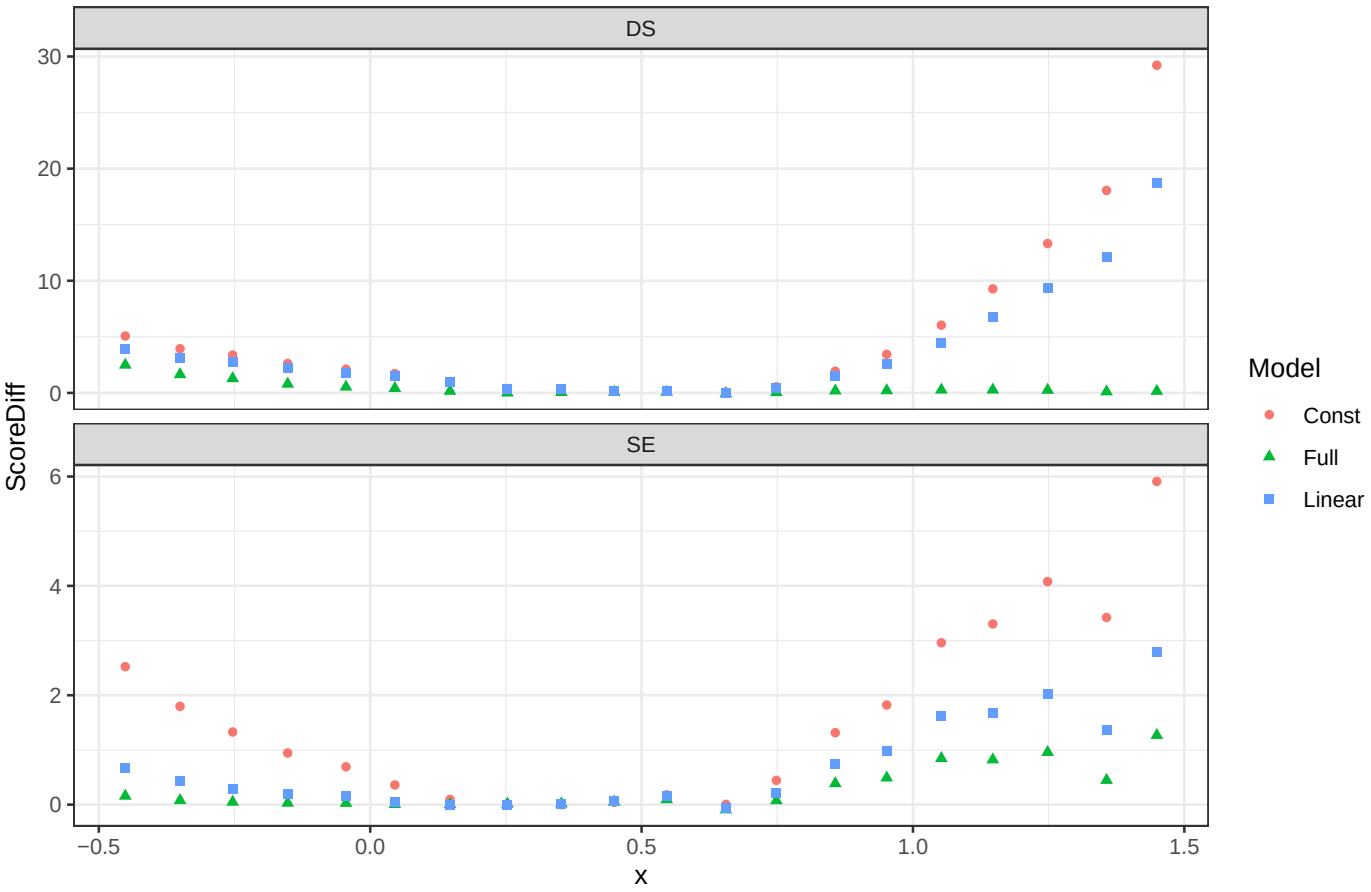
One important aspect is that the score values for a single test data point is dependent across each model in a comparison. To take this into account, it is often more useful to consider pairwise score differences between models, instead of the raw score values. Here, we illustrate the score differences between each estimated model and the true model prediction scores. The plots of score difference CDFs still need to be interpreted with care, and note that for real world problems we do not have access to the ”true” model. However, by taking pairwise score differences to a reference model, we reduce the interpretational issues that arise from non-exchangeability, due to the properties of the scores that will be explored in the next section.

```
score_diffs <- scores |> filter(Data == "Test", Model != "True") |>
        left_join(scores |> filter(Data == "Test", Model == "True"),
                  by = c("x", "y", "ScoreType"),
                  suffix = c("", ".True"))
score_diffs |>
  ggplot() +
  stat_ecdf(aes(x = Score - Score.True, col = Model, linetype = Model)) +
  xlab("Score difference") + ylab("Empirical CDF") +
  facet_wrap(~ ScoreType, scales = "free_x", ncol = 1) +
  coord_cartesian(xlim = c(-10, 10))
```




```
score_diffs |>
  group_by(ScoreType, Model, floor(x * 10)) |>
  summarise(
    x = mean(x),
    ScoreDiff = mean(Score - Score.True),
    .groups = "drop"
```

```
) |>
ggplot() +
geom_point(aes(x, ScoreDiff, col = Model, shape = Model)) +
facet_wrap(~ ScoreType, ncol = 1, scales = "free_y")
```



3 Score expectations and proper scoring rules

What functions of the predictive distributions are useful scores? In order to compare models in coherent, systematic ways, we need to ensure the scores we use have some fundamental properties. We also need to ensure that aggregated measures, such as score averages, do not break the basic requirements.

We want to reward accurate (unbiased) and precise (small variance) predictions, but not at the expense of understating true uncertainty. One important concept, developed for weather forecasting, is *calibrated probability statements*. For example, if it rains, on average, 70% of the days for which a rain forecast probability of 70% was given, and similarly for every other forecast, the forecast method is said to generate calibrated probability forecasts. The goal of forecasting then becomes to minimise uncertainty, subject to calibration.

First, we define the expectation of a score of a single observation y as

$$S(F, G) = \mathrm{E}_{y\sim G}[S(F, y)], \tag{21}$$

where F denotes the prediction model, and G is a data generating model. A negatively oriented score is *proper* if it fulfils

$$S(F, G) \geq S(G, G). \tag{22}$$

The practical interpretation of this is that a proper score does not reward cheating; stating a lower (or higher) forecast/prediction uncertainty will not, on average, give a better score than a model based on the true data generating model. A proper score that has equality of the expectations *only* when F and G are the same, $F(\cdot) \equiv G(\cdot)$, is said to be *strictly proper*. For non-strict proper scores, there is typically a few properties of the predictive distribution that give equality, e.g. the expectation or median. We might then say that such a scores are *proper score targeting the predictive expectation* and *proper score targeting the predictive median*, to make it clear what prediction properties the score is capable of distinguishing between.

Remark 2. *Note that there may be philosophical objections to the notion of a "true" model, but the concept is still useful as a theoretical construct encapsulating the concept of "properly calibrated uncertainty, based on the available observations". This "calibrated optimal oracle" is all we need for a coherent prediction assessment framework.*

Let's revisit our previously defined scores and check if they are proper!

3.1 SE

$$\begin{aligned}
S_{\text{SE}}(F, G) &= \mathrm{E}_{y\sim G}[S_{\text{SE}}(F, y)] && (23)\\
&= \mathrm{E}_{y\sim G}[(y - \mu_F)^2] && (24)\\
&= \mathrm{E}_{y\sim G}[(y - \mu_G + \mu_G - \mu_F)^2] && (25)\\
&= \mathrm{E}_{y\sim G}[(y - \mu_G)^2 + 2(y - \mu_G)(\mu_G - \mu_F) + (\mu_G - \mu_F)^2] && (26)\\
&= \mathrm{E}_{y\sim G}[(y - \mu_G)^2] + 2(\mu_G - \mu_F)\mathrm{E}_{y\sim G}[y - \mu_G] + (\mu_G - \mu_F)^2 && (27)\\
&= \sigma_G^2 + (\mu_G - \mu_F)^2 && (28)
\end{aligned}$$

This is minimised when $\mu_F = \mu_G$. Therefore $S_{\text{SE}}(F, G) \geq S_{\text{SE}}(G, G) = \sigma_G^2$, so the score is proper, targeting the expectation. It is not strictly proper, since there are many different distributions with expectation μ_G.

3.2 AE

For notational convenience, write $m_F = F^{-1}(1/2)$ for the predictive median under F.

$$S_{\text{AE}}(F, G) = \mathrm{E}_{y\sim G}[S_{\text{AE}}(F, y)] = \mathrm{E}_{y\sim G}\left[|y - m_F|\right] = \int_{-\infty}^{\infty} |y - m_F| \, \mathrm{d}G(y) \tag{29}$$

We then take the derivative with respect to m_F and change order between derivative and integration:

$$\begin{aligned}
\frac{\partial}{\partial m_F} S_{\text{AE}}(F, G) &= \int_{-\infty}^{\infty} \operatorname{sign}(m_F - y) \, \mathrm{d}G(y) && (30)\\
&= \int_{-\infty}^{m_F} \mathrm{d}G(y) - \int_{m_F}^{\infty} \mathrm{d}G(y) && (31)\\
&= G(m_F) - [1 - G(m_F)] = 2G(m_F) - 1. && (32)
\end{aligned}$$

This is equal to zero if and only if $G(m_F) = 1/2$, i.e. when $m_F = G^{-1}(1/2)$, the predictive median under G. Since the derivative expression is increasing in m_F, this is a (unique) minimum. Therefore $S_{\text{AE}}(F, G) \geq S_{\text{AE}}(G, G) = \sigma_G^2$, so the score is proper, targeting the median. It is not strictly proper, since there are many different distributions with median $G^{-1}(1/2)$.

Remark 3. *It is noteworthy that although the AE score targets the predictive* median*, many applications of it in the applied literature use the predictive* mean *(expectation) when evaluating the score. This means that even under G, the score might not be minimised, and some other model might get a lower average score by having a mean closer to the median of G. Since the prediction mean and median are often close to each other, this is unlikely to be a major problem, but it does illustrate that care should be taken when applying scores in practice, to avoid unnecessary pitfalls.*

3.3 LOG

The Logarithmic score is strictly proper. Showing this can be done in different ways, such as taking a functional derivative of the expectation integral, but one can also show it using a connection to the *Kullback-Leibler divergence*, $D_{\text{KL}}(g||f)$, that measures the difference between two densities:

$$S_{\text{LOG}}(F, G) - S_{\text{LOG}}(G, G) = \int g(y) \log \left[\frac{g(y)}{f(y)} \right] \, \mathrm{d}y \tag{33}$$

where the integral is the definition of $D_{\text{KL}}(g||f)$, which can be shown to be non-negative, with equality if and only if $f(y) \equiv g(y)$.

3.4 DS

$$\begin{aligned} S_{\text{DS}}(F, G) &= \mathrm{E}_{y \sim G}[S_{\text{DS}}(F, y)] && (34) \\ &= \frac{\mathrm{E}_{y \sim G}[(y - \mu_F)^2]}{\sigma_F^2} + \log(\sigma_F^2) && (35) \\ &= \frac{\sigma_G^2 + (\mu_G - \mu_F)^2}{\sigma_F^2} + \log(\sigma_F^2) && (36) \end{aligned}$$

This is minimised when $\mu_F = \mu_G$ and $\sigma_F = \sigma_G$. Therefore $S_{\text{DS}}(F, G) \geq S_{\text{DS}}(G, G) = 1 + \log(\sigma_G^2)$, so the score is proper, targeting the expectation and standard deviation (or equivalently, the variance). It is not strictly proper, since there are many different distributions with expectation μ_G and standard deviation σ_G.

3.5 CRPS

By definition, we have a relationship between the cdf $G(\cdot)$ and the expectation of an indicator function:

$$\mathrm{E}_{y \sim G}[\mathbb{I}(y \leq x)] = \mathrm{P}(y \leq x) = G(x). \tag{37}$$

By expanding the square in the integrand for the CRPS definition, and changing order between expectation and integration, we can rewrite the CRPS expectation as follows:

$$S_{\text{CRPS}}(F, G) = \mathrm{E}_{y\sim G}\left\{\int_{\mathbb{R}} [F(x) - \mathbb{I}(y \le x)]^2 \, \mathrm{d}x\right\} \tag{38}$$

$$= \mathrm{E}_{y\sim G}\left\{\int_{\mathbb{R}} \left[F(x)^2 - 2F(x)\mathbb{I}(y \le x) + \mathbb{I}(y \le x)\right] \, \mathrm{d}x\right\} \tag{39}$$

$$= \int_{\mathbb{R}} \left[F(x)^2 - 2F(x)G(x) + G(x)\right] \, \mathrm{d}x \tag{40}$$

$$= \int_{\mathbb{R}} \left[F(x)^2 - 2F(x)G(x) + G(x)^2 + G(x) - G(x)^2\right] \, \mathrm{d}x \tag{41}$$

$$= \int_{\mathbb{R}} [F(x) - G(x)]^2 \, \mathrm{d}x + \int_{\mathbb{R}} G(x)\,[1 - G(x)] \, \mathrm{d}x \tag{42}$$

This is minimised when $F(\cdot) \equiv G(\cdot)$, so the score is proper. Furthermore, $S_{\text{CRPS}}(F, G) = S_{\text{CRPS}}(G, G)$ *only* when $F(\cdot) \equiv G(\cdot)$, so the score is strictly proper.

3.6 Improper scores

Scores that are not proper leads to risks of rewarding bad prediction models. One such example is an early attempt for improving the SE score by adding the prediction variance as a penalty term, $(y - \mu_F)^2 + \sigma_F^2$. At first glance, this might look appealing, as it has the form of Bias^2 + Variance. However, this score is not proper, as the expectation under G is $\sigma_G^2 + (\mu_G - \mu_F)^2 + \sigma_F^2$, which is minimised when $\mu_F = \mu_G$ and $\sigma_F = 0$, so that the score rewards underestimating the prediction uncertainty. The "appealing" interpretation is incorrect, as it *assumes* that F is the true distribution, i.e. G!

Fortunately, the DS score is a simple proper version of a generalised SE score, and just as easy to calculate, so there is no practical reason to use the improper score

4 Classification scores

For models where the target is an binary (0/1) indicator or categorical classification labels $(1, \ldots, K)$, the most common scores are the Brier and the Logarithmic scores. In the binary case, let $z \in \{0, 1\}$ be a binary outcome, and let p_F be the predictive probability for $z = 1$ under model F. For categorical outcomes, let $z \in \{1, \ldots, K\}$ be the observed class label, and let $\mathrm{P}_F(Z = k)$ be the predictive probability for class k under model F. Then the scores are then defined as

$$S_{\text{Brier}}(F, z) = (z - p_F)^2 \quad \text{(Binary case, } z \in \{0, 1\}) \tag{43}$$

$$S_{\text{Brier}}(F, z) = \sum_{k=1}^{K} \left[\mathbb{I}(z = k) - \mathrm{P}_F(Z = k)\right]^2 = 1 - 2P_F(Z = z) + \sum_{k=1}^{K} \mathrm{P}_F(Z = k)^2 \quad \text{(Categorical case, } z \in \{1, \ldots, K\}) \tag{44}$$

$$S_{\text{LOG}}(F, z) = -z \log(p_F) - (1 - z) \log(1 - p_F), \quad \text{(Binary case, } z \in \{0, 1\}) \tag{45}$$

$$S_{\text{LOG}}(F, z) = -\log \mathrm{P}_F(Z = z) \quad \text{(Categorical case, } z \in \{1, \ldots, K\}) \tag{46}$$

where the Logarithmic scores are the (negated) log-likelihood of a Bernoulli distribution with success probability p_F, and a Multinomial distribution with class probabilities $\mathrm{P}_F(Z = k)$, respectively. Both the Brier and Logarithmic scores are strictly proper scoring rules for binary and categorical predictions.

Remark 4. *The binary and categorical Brier scores as defined above differs by a factor 2 between the binary score and the $K = 2$ case for the categorical score. Proper scores preserve their properness when shifted, and when scaled by positive constants, so the differently scaled definitions are just a matter of tradition and cosmetical style preferences.*

Remark 5. *There is an interesting connection between the binary Brier score and CRPS. Let F be the predictive CDF for y, and let $z_x = \mathbb{I}(y \leq x)$ be the binary indicator of whether y is less than or equal to x, so that $F(x)$ is the probability for the event $z_x = 1$. Then $S_{Brier}(F, y) = [z_x - F(x)]^2 = [F(x) - \mathbb{I}(y \leq x)]^2$. This is exactly the integrand in the definition of CRPS, taken over all possible values x. The effect of this is that CRPS can be interpreted as an integrated Brier score over all possible thresholds x.*

5 Aggregated scores and differences

When we take the average of n predictive scores for a model, by applying only proper scores, the properness property is preserved.

We define *average* or *mean* scores under two scenarios:

1. A single forcast/prediction distribution F, with many independent observations y_i
2. A collection of forcast/prediction distributions $\{F_i\}$, each predicting a single observation from the collection $\{y_i\}$, i.e. we have a collection of prediction/obserrvation pairs $\{(F_i, y_i)\}$.

For case 1, the average score is

$$\overline{S}(F, \{y_i\}) = \frac{1}{n} \sum_{i=1}^{n} S(F, y_i). \tag{47}$$

For case 2, the average score is

$$\overline{S}(\{(F_i, y_i)\}) = \frac{1}{n} \sum_{i=1}^{n} S(F_i, y_i). \tag{48}$$

For example, the commonly used *Mean Squared Error* (MSE) can be defined as $\text{MSE} = \overline{S}_{\text{SE}}(\{(F_i, y_i)\})$. This also means that MSE is a proper scoring rule. It is easy to see that the aggregated scores are proper:

$$\begin{aligned}
\overline{S}(\{(F_i, G_i)\}) &= \mathrm{E}_{\{y_i\} \sim G\}} \left[\frac{1}{n} \sum_{i=1}^{n} S(F_i, y_i) \right] && (49)\\
&= \frac{1}{n} \sum_{i=1}^{n} \mathrm{E}_{y_i \sim G_i}[S(F_i, y_i)] && (50)\\
&= \frac{1}{n} \sum_{i=1}^{n} S(F_i, G_i) && (51)\\
&\geq \frac{1}{n} \sum_{i=1}^{n} S(G_i, G_i) && (52)\\
&= \overline{S}(\{(G_i, G_i)\}). && (53)
\end{aligned}$$

As case 1 can be seen as a special case of case 2 (by letting $F_i \equiv F$ for all i), the properness property is preserved for both cases.

This means that even when the individual scores themselves are random variables with different distributions for each observation, the aggregated score at least fulfils the basic properness requirement.

The main remaining step is to deal with the dependence of the the scores for different models. We can do that by taking pairwise score differences between models:

$$S^{\Delta}(F_i^A, F_i^B, y_i) = S^{\Delta}(F_i^A, y_i) - S^{\Delta}(F_i^B, y_i) \tag{54}$$

$$\overline{S}^{\Delta}(\{F_i^A, F_i^B, y_i\}) = \overline{S}^{\Delta}(\{F_i^A, y_i\}) - \overline{S}^{\Delta}(\{F_i^B, y_i\}) \tag{55}$$

Although the difference between the score averages is the same as the average of the pairwise score differences, the latter formulation provides an easy route to assessing the *uncertainty in the score differences*. We can treat the pairwise score differences as a single collection of random variables taking from a joint collection, and estimate the overall average score difference variance:

$$\text{Var}\left[\overline{S}^{\Delta}(\{F_i^A, F_i^B, y_i\})\right] \approx \frac{1}{n^2}\sum_{i=1}^{n} \text{Var}\left[S^{\Delta}(F_i^A, F_i^B, y_i)\right] \tag{56}$$

$$\approx \frac{1}{n(n-1)}\sum_{i=1}^{n}\left(S^{\Delta}(F_i^A, F_i^B, y_i) - \overline{S}^{\Delta}(\{F_i^A, F_i^B, y_i\})\right)^2 \tag{57}$$

Another approach is to estimate the variability of the aggregated differences with bootstrap resampling methods, which can also be used to formulate *permutation tests* for aggregated score differences.

Remark 6. *We saw before that* not *aggregating the scores or their differences may provide more nuanced insights into model performance. This is in particular true for spatial models, where one model might be better than another model in one region, but worse in another. Such structured differences are masked when taking averages over all observations. However, often some kind of overall comparison is needed, and then the pairwise difference aggregation formulation provides methods for assessing the variability of the aggregated score differences.*