Prediction and model assessment (MATH10093: L03)

- Lab 3, 30/1: RMarkdown (easy PDFs including R code and figures), scoring rules
- Coursework A: Wed 6/2 (w4) – Wed 27/2 (w6)
- Theme: Numerical model estimation and predictive assessment
- Counts towards 50% of the unit grade.
- Individual electronic (PDF) handin on Learn.
- Coursework A from last year (2018) is available, including solutions and detailed marking scheme

Lecure and lab in week 4.
No lecture in week 5.
There will be a few tutors available in the week 5 lab session to anwer general questions.
A half-way feedback form will be available in Lab 4.

Prediction and proper scoring rules

- When using numerical estimation methods subject to both numerical precision errors, random data collection variation, and methodological approximations, it's useful to be able to assess the end result in a way that's not tied to a particular method or model.
- One approach: *split* the data into *observations for estimation* and *test* data: $\mathcal{Y}_{\text{obs}}$ and $\mathcal{Y}_{\text{test}}$.
- Estimate the model parameters using the *estimation* data $\mathcal{Y}_{\text{obs}}$.
- Assess how good the estimated model is at predicting the values of the *test* data $\mathcal{Y}_{\text{test}}$.
- The most common assessment is to compare *point predictions* with their corresponding actual value in the test data:
  Squared error $= (y_{\text{test}} - \widehat{y})^2$.
- To assess methodology, it's often useful to use *simulated* data, so that we know the true model. If the method is able to come close to the true values, we are more confident that it will also work on data where we don't know the true model.

More extensive written notes on prediction and proper scoring rules are available on Learn.

Simple estimation/test assessment

Consider the model $y_i \sim \mathsf{N}(\theta_1, \theta_2^2)$, independent y_i, $i = 1, \ldots, n$.

```
## Simulate data:
N <- 100
Y <- rnorm(N, mean = 10, sd = 2)

## Split the data, 75 for estimation, 25 for testing:
n <- c(75, 25)
obs <- sample(rep(c(TRUE, FALSE), c(75, 25)), size = N, replace = FALSE)  # Split randomly
y_obs <- Y[obs] # Extract the observations to be used for estimation
y_test <- Y[!obs] # Extract the observations to be used for testing

## Estimate the paramters:
theta1_hat <- mean(y_obs)
theta2_hat <- sd(y_obs)

## Test:
mean( (y_test - theta1_hat)^2 ) ## Average squared error

## [1] 5.113702

mean( (y_test - 0)^2 ) ## Average squared error for a bad model estimate

## [1] 133.0396
```

Forecasting and prediction

- The term *forecasting* typically means doing a *prediction* of future events or values, e.g. for weather forecasts
- Based on some statistical model and observed data, we typically construct a *point estimate* that is our best guess of the future value. Ideally, we also compute some measure of *uncertainty* about how large we expect the error to be, i.e. the difference between the point estimate and the true future value.
- In statistical terminology, this process is called *prediction*, and we seek useful *predictive distributions* that encode our knowledge from a statistical model and previously observed data into a representative distribution of possible future data values.
- Note: In Bayesian statistics, *prediction* (distributions and prediction intervals) can apply to *any* quantity that has not yet been observed. In frequentist statistics, fixed but unknown parameter values are instead associated with *confidence intervals*, and *prediction intervals* are reserved for observable random quantities.
- We will gloss over the differences between frequentist and Bayesian approaches, and focus on prediction of observable data values.

Prediction distributions

- From asymptotic likelihood theory it is known that, approximately, $\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}_{\text{true}} \sim \mathsf{N}(\mathbf{0}, \boldsymbol{\Sigma}_\theta)$, where $\boldsymbol{\Sigma}_\theta^{-1}$ can be estimated by $H(\widehat{\boldsymbol{\theta}})$ (log-likelihood Hessian from Lecture 2)
- With a slight abuse of (frequentist) notation, we represent the estimation uncertainty by a distribution of *potential* parameter values: $\boldsymbol{\theta} \sim \mathsf{N}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_\theta)$, with density written as $f_\theta(\boldsymbol{\theta})$
- The conditional density for the observable values y is written as $f_{y|\theta}(y)$
- The *predictive density* $f_y(\cdot)$ is then obtained by integrating over $\boldsymbol{\theta}$:

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

- We will identify the predictive distribution with the CDF F, $F(x) = \mathsf{P}_{y\sim F}(y \leq x) = \int_{-\infty}^{x} f_y(y) \, \mathrm{d}y$.
- Often, we only consider the predictive expectation and variance:

$$\mu_F = \mathsf{E}_{y\sim F}(y)$$
$$\sigma_F^2 = \mathsf{Var}_{y\sim F}(y)$$

Example: Generalisation of Lab 2

- Let $y \sim \mathsf{N}\left[\boldsymbol{z}_E^\top \boldsymbol{\theta}, \exp(\boldsymbol{z}_V^\top \boldsymbol{\theta})\right]$, i.e. the expectation has a linear model, and the variance has a log-linear model, where the same parameters could potentially influence both the expectation and the variance.
- With the $\boldsymbol{z}_{\cdot}$ vectors for each observation stored as rows in two matrices $\boldsymbol{Z}_E$ and $\boldsymbol{Z}_V$, the vector of observation expectations can be written as $\mathsf{E}_{y|\theta}(\boldsymbol{y}) = \boldsymbol{Z}_E\boldsymbol{\theta}$, and similarly for the log-variances.

Example: Take the special case $\boldsymbol{z}_{E_i}^\top = \begin{bmatrix}1 & x_i & 0 & 0\end{bmatrix}$ and $\boldsymbol{z}_{V_i}^\top = \begin{bmatrix}0 & 0 & 1 & x_i\end{bmatrix}$, i.e. the parameters for expectation and log-variance are not directly coupled, and we have an intercept and a single covariate for both of the submodels.

```
model_Z <- function(x) {
  Z0 <- model.matrix(~ 1 + x)
  list(ZE = cbind(Z0, Z0 * 0), ZV = cbind(Z0 * 0, Z0))
}
```

This takes a vector of covariate values x_i, one for each observation, as input, and return a list, with $\boldsymbol{Z}_E$ and $\boldsymbol{Z}_V$ as named elements.

Example: Predictive distribution

- Using numerical optimisation on the negative loglikelihood,

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

  provides $\widehat{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}_\theta$.

- The *tower property* ($\mathsf{E}_A(A) = \mathsf{E}_B[\mathsf{E}_{A|B}(A)]$) gives

$$\mathsf{E}_F(y) = \boldsymbol{z}_E^\top \widehat{\boldsymbol{\theta}}, \qquad \mathsf{Var}_F(y) = \mathsf{E}_\theta\left[\exp\left(\boldsymbol{Z}_V\boldsymbol{\theta}\right)\right] + \mathsf{Var}_\theta\left(\boldsymbol{z}_E^\top\boldsymbol{\theta}\right).$$

  The second term of the variance is

$$\mathsf{Var}_\theta\left(\boldsymbol{z}_E^\top\boldsymbol{\theta}\right) = \mathsf{Cov}_\theta\left(\boldsymbol{z}_E^\top\boldsymbol{\theta}, \boldsymbol{z}_E^\top\boldsymbol{\theta}\right) = \boldsymbol{z}_E^\top \mathsf{Cov}_\theta\left(\boldsymbol{\theta},\boldsymbol{\theta}\right)\boldsymbol{z}_E = \boldsymbol{z}_E^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_E.$$

  For the first term, we use a result that says that if $x \sim \mathsf{N}(\mu, \sigma^2)$, then $\mathsf{E}(\mathrm{e}^x) = \mathrm{e}^{\mu+\sigma^2/2}$:

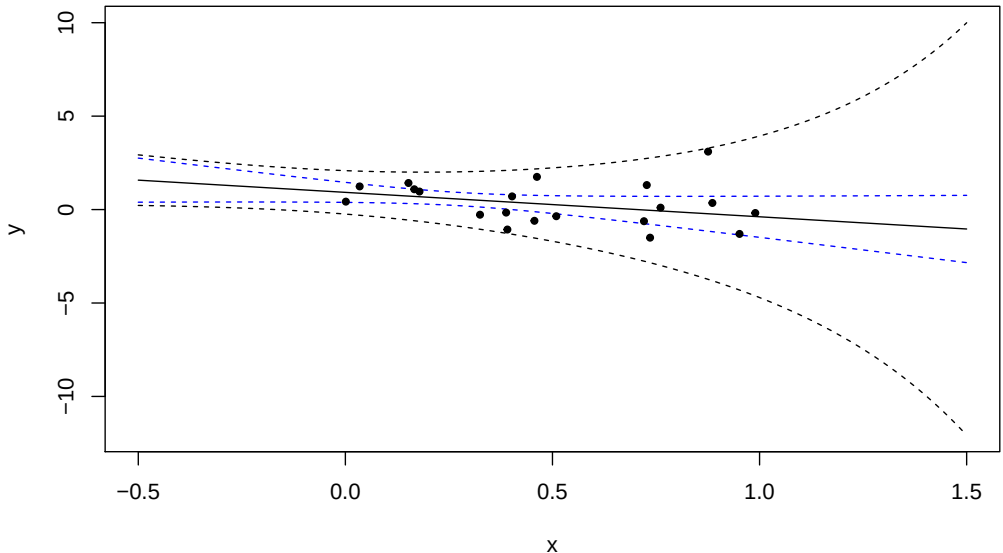
$$\mathsf{E}_\theta[\exp(\boldsymbol{z}_V^\top\boldsymbol{\theta})] = \exp\left(\boldsymbol{z}_V^\top\widehat{\boldsymbol{\theta}} + \boldsymbol{z}_V^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_V/2\right).$$

  Combining the results, we get the predictive variance as

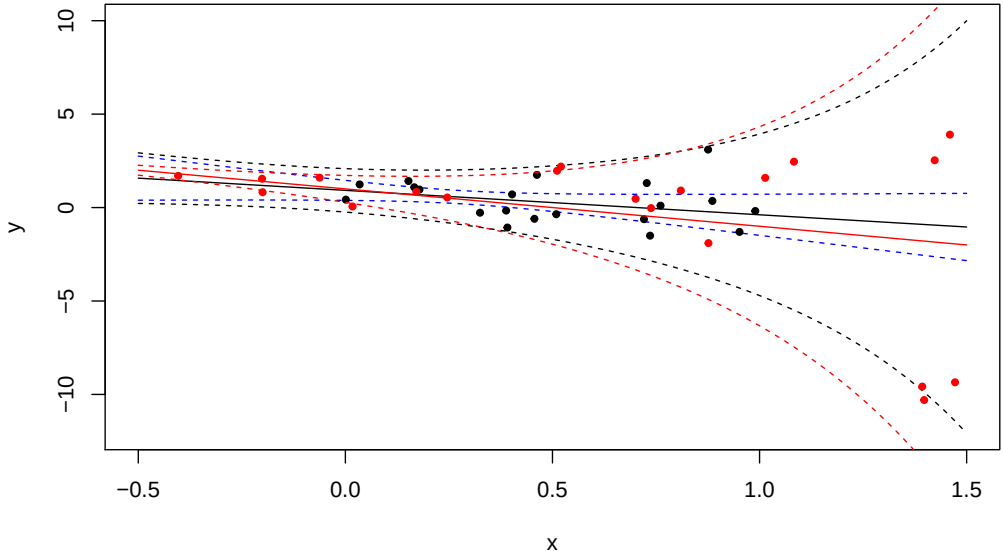
$$\sigma_V^2 = \mathsf{Var}_F(y) = \exp\left(\boldsymbol{z}_V^\top\widehat{\boldsymbol{\theta}} + \boldsymbol{z}_V^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_V/2\right) + \boldsymbol{z}_E^\top\boldsymbol{\Sigma}_\theta\boldsymbol{z}_E.$$

```r
# Value: data.frame with columns (mu, sigma, lwr, upr)
model_predict <- function(theta, data, Sigma_theta = NULL,
                          type = c("expectation", "log-variance", "observation"),
                          alpha = 0.05, df = Inf,
                          nonlinear.correction = TRUE) {
  type <- match.arg(type)
  Z <- model_Z(data) ## Note: Will use model_Z() defined in the global workspace!
  fit_E <- Z$ZE %*% theta
  fit_V <- Z$ZV %*% theta
  if (is.null(Sigma_theta)) {
    ZE_var <- ZV_var <- 0
  } else {
    ZE_var <- rowSums(Z$ZE * (Z$ZE %*% Sigma_theta))
    ZV_var <- rowSums(Z$ZV * (Z$ZV %*% Sigma_theta))
  }
  if (type == "expectation") { ## confidence interval
    ...
  } else if (type == "observation") { ## observation predictions
    fit <- fit_E
    sigma <- (exp(fit_V + ZV_var / 2 * nonlinear.correction) + ZE_var)^0.5
  }
  q <- qt(1 - alpha / 2, df = df) ## If the user wants a t-quantile instead of Normal.
  lwr <- fit - q * sigma
  upr <- fit + q * sigma
  data.frame(mu = fit, sigma, lwr, upr)
}
```

Estimation data, point estimates, confidence and prediction intervals



Add the true model&predictions, and some test data



Scores

- We want to quantify how well our predictions represent the test data.
- Squared Error (SE): $S_{\text{SE}}(F, y) = (y - \widehat{y}_F)^2$,
  where $\widehat{y}_F$ is a point estimate under F, e.g. the expectation μ_F.
- Logarithmic/Ignorance score (LOG/IGN): $S_{\text{LOG}}(F, y) = -\log f(y)$,
  where $f(\cdot)$ is the predictive density.
- Dawid-Sebastiani (DS): $S_{\text{DS}}(F, y) = \frac{(y-\mu_F)^2}{\sigma_F^2} + \log(\sigma_F^2)$.
- The scores defined here are *negatively oriented*, meaning that the *lower the score, the better*.

```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)
}
```

Example

We evaluate the SE and DS scores for the true model, a naive simplistic model, and the estimated full model.

```
## For the estimation data:
opred_true <- model_predict(theta_true, x_obs, type = "observation")
opred0 <- data.frame(mu = mean(y_obs), sigma = sd(y_obs))
opred1 <- model_predict(theta_hat, x_obs, Sigma_theta = Sigma_theta, type = "observation")

## For the test data:
pred_true <- model_predict(theta_true, x_test, type = "observation")
pred0 <- data.frame(mu = mean(y_obs), sigma = sd(y_obs))
pred1 <- model_predict(theta_hat, x_test, Sigma_theta = Sigma_theta, type = "observation")
```

Example

```
## SE for observed and test data
rbind(
  c(mean(score_se(opred_true, y_obs)),
    mean(score_se(opred0, y_obs)),
    mean(score_se(opred1, y_obs))),
  c(mean(score_se(pred_true, y_test)),
    mean(score_se(pred0, y_test)),
    mean(score_se(pred1, y_test)))
)
```

```
##           [,1]      [,2]      [,3]
## [1,]  1.429616  1.247714  1.215824
## [2,] 13.800697 17.275713 14.711117
```

Example

```
## DS for observed and test data
rbind(
  c(mean(score_ds(opred_true, y_obs)),
    mean(score_ds(opred0, y_obs)),
    mean(score_ds(opred1, y_obs))),
  c(mean(score_ds(pred_true, y_test)),
    mean(score_ds(pred0, y_test)),
    mean(score_ds(pred1, y_test)))
)

##          [,1]      [,2]      [,3]
## [1,] 1.002367  1.222606 0.9429584
## [2,] 2.202256 13.426209 2.0991537
```

Score expectations and proper scoring rules

- What functions of the predictive distributions are useful scores?
- We want to reward accurate (unbiased) and precise (small variance) predictions, but not at the expense of understating true uncertainty.
- First, we define the expectation of a score under a true distribution G s

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

Proper scores/scoring rules

A negatively oriented score is *proper* if it fulfils

$$S(F, G) \geq S(G, G).$$

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*.

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 stating the truth.

$$
\begin{aligned}
S_{\mathsf{SE}}(F,G) &= \mathsf{E}_{y\sim G}[S_{\mathsf{SE}}(F,y)] = \mathsf{E}_{y\sim G}[(y-\mu_F)^2] = \mathsf{E}_{y\sim G}[(y-\mu_G+\mu_G-\mu_F)^2] \\
&= \mathsf{E}_{y\sim G}[(y-\mu_G)^2 + 2(y-\mu_G)(\mu_G-\mu_F) + (\mu_G-\mu_F)^2] \\
&= \mathsf{E}_{y\sim G}[(y-\mu_G)^2] + 2(\mu_G-\mu_F)\mathsf{E}_{y\sim G}[y-\mu_G] + (\mu_G-\mu_F)^2 \\
&= \sigma_G^2 + (\mu_G-\mu_F)^2
\end{aligned}
$$

This is minimised when $\mu_F = \mu_G$. Therefore $S_{\mathsf{SE}}(F,G) \geq S_{\mathsf{SE}}(G,G) = \sigma_G^2$, so the score is proper. Is it strictly proper?

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

This is minimised when $\mu_F = \mu_G$ and $\sigma_F = \sigma_G$. Therefore $S_{\mathsf{DS}}(F,G) \geq S_{\mathsf{DS}}(G,G) = 1 + \log(\sigma_G^2)$, so the score is proper. Is it strictly proper?