Statistical computing MATH10093
Computer lab 3
Solutions

Finn Lindgren

29/1/2020

Summary

In this lab session you will experiment with RMarkdown, and explore model assessment methods using predictive scores. You will not hand in anything, but you should keep your code script file for later use.

1. Download all the R files for Lab 3; `lab03code.R`, `RMdemo.Rmd`, and `RMtemplate.Rmd`.

   If you're running on your own computer, make sure you have the `rmarkdown` package installed; If `library(rmarkdown)` gives an error, run `install.packages("rmarkdown")`

2. Make sure that `Tools`→`Global Options`→`Sweave`→`Weave` is set to `knitr`.

3. Open `RMdemo.Rmd` in RStudio and press `Ctrl-Shift-K` to generate a PDF version.

   There is also an button labeled `Knit` at the top of the editor window that can be pressed; when the mouse hovers over it, it will say `Knit the current document`.

   A third option is `File`→`Knit Document`.

   The R function call `rmarkdown::render('RMdemo.Rmd')` should also work, or by providing the full pathname of the file.

4. Open `lab03code.R` and read through the code. This is also a kind of RMarkdown document, so you can generate and read a pdf version if you prefer.

   The code near the end of the document generates synthetic data for a model that links the calculations from a CAD program to the actual amounts of material required to 3D-print a collection of objects. The code uses different model parameters for each material colour, reflecting the varying chemical and physical properties of different materials. The aim of the lab is to experiment on this data with a model that ignores the colour information.

5. Make a copy of the `RMtemplate.Rmd` file and open it. Note that it contains a commented line that would automatically include the code from `CWAcode.R`. Make the following changes:

   (a) Change the to `source` code line to use the `lab03code.R` file instead of `CWAcode.R`.

   (b) Change the document title to something more appropriate for the lab

   (c) Change the document author to your own name

6. Copy the commented plotting code from the end of `lab03code.R` (and *only* that code) into a new code chunk in your new document, and generate a `pdf` report. Do not copy code such as function definitions that you're just calling from your own code. The `source` function will load all the code from `lab03code.R` and unless stated otherwise, you should *not* change or copy it to your own file.

7. Use the estimation and prediction code from Lecture 3 estimate, predict, and check, for the simple model where $y =$`actual` and $x =$`cad` in the simple Lecture 3 model.

   First, split the data into estimation and testing parts:

```
obs <- sample(rep(c(TRUE, FALSE), c(0.75, 0.25) * nrow(printer_data)),
              size = nrow(printer_data),
              replace = FALSE)
data_obs <- printer_data[obs, ]
data_test <- printer_data[!obs, ]
```

   Then, define a new `model_Z` function that takes a `data.frame` parameter `data` instead of a vector `x`, and uses the `cad` column of the `data.frame` in the model matrix construction.

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

   This model class isn't the true model for this data, so we don't have a `theta_true` value to test against. Instead, use the constant-variance model `theta_ref = c(0, 1, 6, 0)` as a reference model.

   Now follow the recipe from the lecture and longform notes. The main steps are

   (a) Estimate $\boldsymbol{\theta}$ and obtain $\boldsymbol{\Sigma}_\theta$.

(b) Compute predictions using `model_predict()` to plot prediction intervals as functions of `cad`. Remember that you need to supply a `data.frame` with a column `cad` to predict in this model. See previous labs for how to plot prediction intervals using `geom_ribbon`. To be able to distinguish between the models, use `geom_ribbon(..., fill = ...)` to specify a *fill* colour, and use the same colour for the prediction curve.

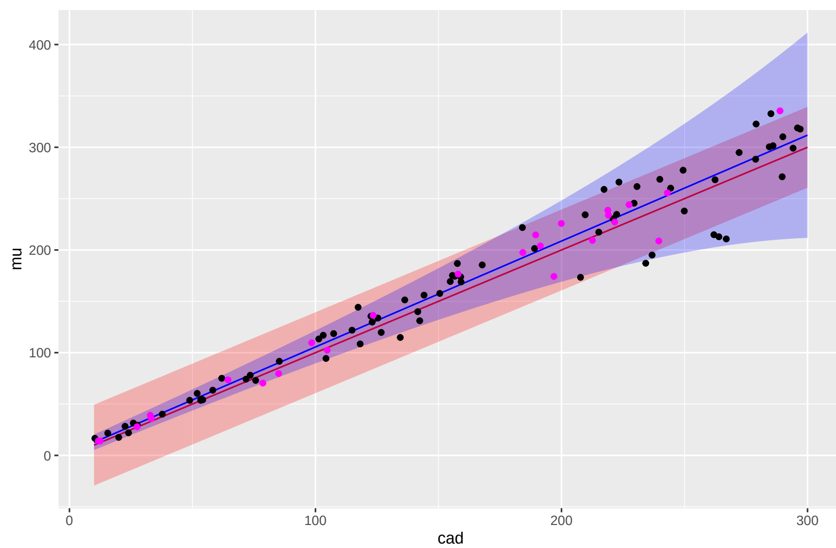
(c) Predict for the estimation data and test data separately.

(d) Compute and compare scores using `score_se()` and `score_ds()`.

Note: This lab question has a close similarities to parts of the upcoming coursework; take the opportunity to aks questions about the lab and the code in the Prediction and Proper Scoring Rules notes (which you should feel free to liberally copy into your lab file, for the parts that you need that are not in the `lab03code.R` file). Remember that the lab solutions are also available.

```
Z_obs <- model_Z(data_obs)
opt <- optim(rep(0, 4),
             fn = neg_log_lik,
             Z = Z_obs, y = data_obs$actual,
             method = "BFGS", hessian = TRUE)
theta_hat <- opt$par
Sigma_theta <- solve(opt$hessian)

## Prediction intervals:

theta_ref <- c(0, 1, 6, 0)
x_plot <- data.frame(cad = seq(10, 300, length=100))
pred_plot_ref <-
  cbind(x_plot,
        model_predict(theta_ref, x_plot, type = "observation"))
pred_plot_hat <-
  cbind(x_plot,
        model_predict(theta_hat, x_plot,
                      Sigma_theta = Sigma_theta, type = "observation"))
ggplot() +
  geom_ribbon(data = pred_plot_ref,
              aes(cad, ymin = lwr, ymax = upr),
              alpha = 0.25, fill = "red") +
  geom_line(data = pred_plot_ref, aes(cad, mu), col = "red") +
  geom_ribbon(data = pred_plot_hat,
              aes(cad, ymin = lwr, ymax = upr),
              alpha = 0.25, fill = "blue") +
  geom_line(data = pred_plot_hat, aes(cad, mu), col = "blue") +
  geom_point(data = data_obs, aes(cad, actual)) +
  geom_point(data = data_test, aes(cad, actual), col = "magenta")
```

```r
## Test scores:

obs_pred_ref <- model_predict(theta_ref, data_obs,
                              type = "observation")
obs_pred_hat <- model_predict(theta_hat, data_obs,
                              Sigma_theta = Sigma_theta,
                              type = "observation")
test_pred_ref <- model_predict(theta_ref, data_test,
                               type = "observation")
test_pred_hat <- model_predict(theta_hat, data_test,
                               Sigma_theta = Sigma_theta,
                               type = "observation")

## SE for observed and test data
rbind(
  obs = c(ref = mean(score_se(obs_pred_ref, data_obs$actual)),
          hat = mean(score_se(obs_pred_hat, data_obs$actual))),
  test = c(ref = mean(score_se(test_pred_ref, data_test$actual)),
           hat = mean(score_se(test_pred_hat, data_test$actual)))
)

##           ref      hat
## obs  466.6786 426.3953
## test 289.4991 217.8406

## DS for observed and test data
rbind(
  obs = c(ref = mean(score_ds(obs_pred_ref, data_obs$actual)),
          hat = mean(score_ds(obs_pred_hat, data_obs$actual))),
  test = c(ref = mean(score_ds(test_pred_ref, data_test$actual)),
           hat = mean(score_ds(test_pred_hat, data_test$actual)))
)

##           ref      hat
## obs  7.156781 6.285741
## test 6.717597 5.758875

# As should be expected, the model that knows about the variable
# variance scores better (lower) than the constant variance model.
```


400
300
200
100
0
mu
0
100
200
300
cad