

UNIVERSITY OF TECHNOLOGY SYDNEY
School of Mathematical and Physical Sciences
37458 Advanced Bayesian Methods

LABORATORY 4

Due time and date: 9:55am, Friday 14th June, 2019.

Submission method: E-mail message to Professor Wand (matt.wand@uts.edu.au).

Goals: The goals of Laboratory 4 are:

- Demonstrate how the Bayesian inference engine can handle complications such as data subject to missingness or measurement error and heteroscedasticity (non-constant variance) in regression analysis.
- Show that linear mixed models for grouped data can also be fit in R using non-Bayesian approaches. Even though the theme of 37458 is advanced Bayesian methods it is worth knowing about non-Bayesian (or *frequentist*) software for analysis of grouped data.

1. Downloading Required Files

- (a) Choice of web browser if using Windows: the use of Mozilla Firefox or Google Chrome rather than Microsoft Explorer is recommended for all laboratories in 37458 Advanced Bayesian Methods since the latter web-browser has a tendency to change filenames (for example, Microsoft Explorer might add a `.txt` extension) which can cause problems later in this laboratory.
- (b) Choice of editor if using Windows: the use of WordPad rather than Notepad is recommended for all laboratories in 37458 Advanced Bayesian Methods since Notepad has a tendency to corrupt text files that do not have an extension that matches particular types defined by Microsoft Corporation.
- (c) Decide on a folder in which you will store the files for this laboratory. It is best if you put them all in the same folder. A simple option is to put them on the Desktop folder, but please take note of where you store them for later.
- (d) Open Mozilla Firefox or Google Chrome and go to the web-page:
`matt-wand.utsacademics.info/37458.html`
- (e) Scroll down to the heading **Files for Laboratory 4** and download each of the files underneath this heading into the same folder. There are 6 files in total with names:
 - `summMCMC.r`
 - `SLRmeasErr.Rs`
 - `npRegMis.Rs`
 - `lidar.txt`
 - `meanAndVarNonparRegn.Rs`
 - `loadRatsData.Rs`
 - `SydneyRealEstateSubData.txt`

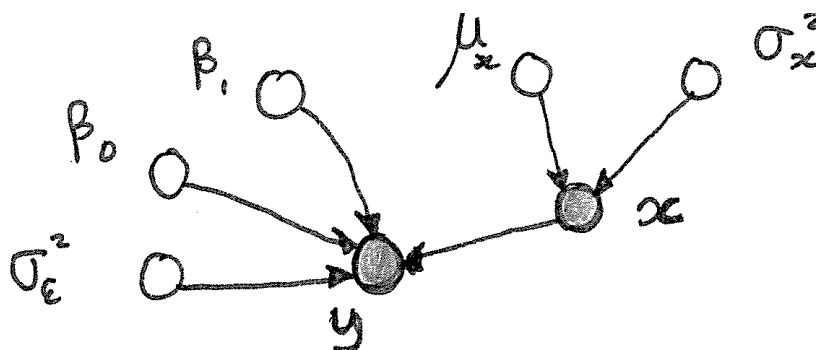
- (f) The remainder of this laboratory assumes that these the filenames are **exactly** as listed here. If the downloading process corrupts the filenames in any way then you need to change the filenames to be these.
- (g) Check to see if `rstan` is installed on the computer that you are using. If not then you will need to install `rstan` for this laboratory. The instructions are on the subject web-site under the heading **Advice About Laptop Preparation For Computer Laboratories**.

2. Simple Linear Regression with Measurement Error

Standard simple linear regression involves estimation of the parameters in

$$y_i | \beta_0, \beta_1, \sigma_\varepsilon^2 \sim N(\beta_0 + \beta_1 x_i, \sigma_\varepsilon^2), \quad 1 \leq i \leq n,$$

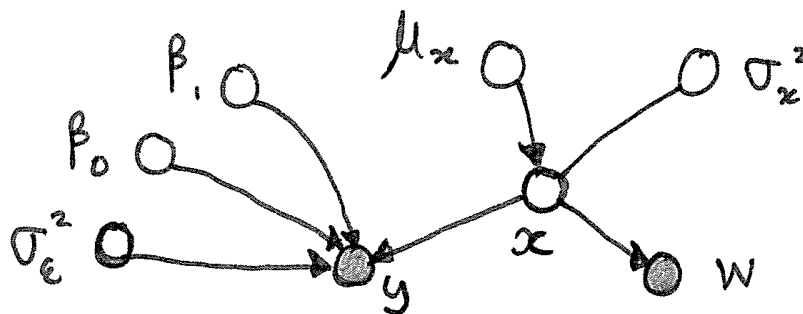
based on the data (x_i, y_i) . Suppose that it also reasonable to assume that $x_i \stackrel{\text{ind.}}{\sim} N(\mu_x, \sigma_x^2)$ and estimation of μ_x and σ_x^2 is of interest. The relevant directed acyclic graph is:



However, suppose that x_i is too expensive to observe and instead we observe a surrogate variable w_i that is related to x_i according to

$$w_i | x_i \stackrel{\text{ind.}}{\sim} N(x_i, \sigma_w^2)$$

for some $\sigma_w > 0$ (assumed known here). This is one of the simplest *measurement error models*. The directed acyclic graph for this simple linear regression measurement error model is:



It is important to incorporate the fact that a contaminated version of x (i.e. w) is observed rather than x to make sure that β_0 , β_1 and σ_ε^2 are estimated correctly.

- (a) Download the script `SLRmeasErr.Rs` from the subject web-site.
(<http://matt-wand.utsacademics.info/37458.html>).
- (b) Issue the command:

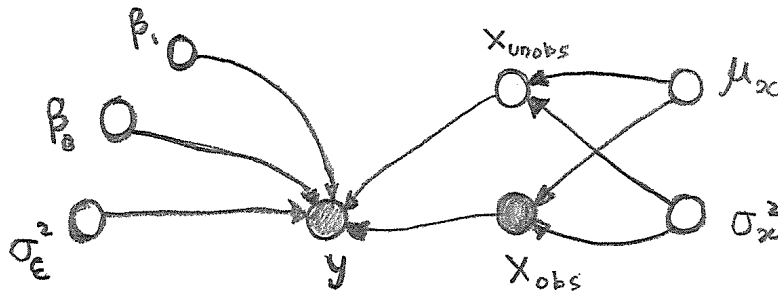
source ("SLRmeasErr.Rs")

to fit this model in Stan and R. The script also produces graphs that show the importance of recognising the measurement error and how the correct parameters are well-estimated after taking the measurement error into account properly.

3. Penalized Spline Nonparametric Regression with Incomplete Data

- (a) Download the script `npRegMis.Rs` from the subject web-site.
- (b) This script performs penalized spline-based nonparametric regression for data (x_i, y_i) , $1 \leq i \leq 300$, data. The problem is that 20% of the x_i values are missing. The Stan code takes this into account, but needs to:
 - Incorporate a model for the x_i data. Here we use $x_i \stackrel{\text{ind.}}{\sim} N(\mu_x, \sigma_x^2)$.
 - Construct the spline basis functions inside Stan. The O'Sullivan spline basis functions used in complete data analyses are difficult to implement in Stan, so instead we use the simple truncated line basis functions $z_k(x) = (x - \kappa_k)_+$ for knots $\kappa_1, \dots, \kappa_K$. Here $a_+ = \max(a, 0)$.

The relevant directed acyclic graph is:



Also note that the data in `npRegMis.Rs` are simulated, so we are pretending the data are missing during the fitting. Later we can see how well the Bayesian inference engine infers the missing x_i values.

- (c) Issue the command `source ("npRegMis.Rs")` to run the script. It may take a few minutes to run, during which you could e.g. check out the latest tweets from the United States president, peruse the more recent cricket World Cup scorecards or find out what a typical day in the life of Kylie Jenner involves. The last set of plots may show some interesting bimodal posterior density functions. This is due to the periodicity of the signal, with more than one x value matching a given y value.

4. Penalized Spline Nonparametric Regression with Variance Function Estimation The simplest nonparametric regression model:

$$y_i \stackrel{\text{ind.}}{\sim} N(f(x_i), \sigma_\epsilon^2), \quad 1 \leq i \leq n \quad (1)$$

assumes that the variance of the y_i values is constant for all $1 \leq i \leq n$. However, this assumption, known as the *homoscedasticity* assumption, is not always reasonable. Instead it is common for the data to be *heteroscedastic* – meaning that the variance is not constant but also a function to be estimated. This involves extension of (1) to the nonparametric mean and variance function model:

$$y_i \stackrel{\text{ind.}}{\sim} N(f(x_i), g(x_i)), \quad 1 \leq i \leq n. \quad (2)$$

- (a) Read in a data set involving a light detection and ranging (LIDAR) experiment via the command:

```
lidar <- read.table("lidar.txt",header=TRUE)
```

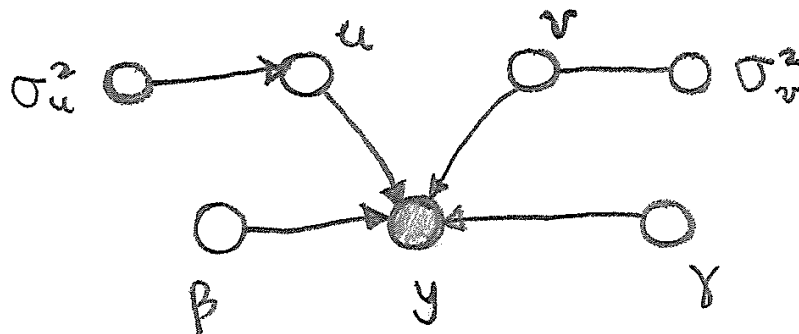
- (b) `plot(lidar,col="blue")`
`fit <- smooth.spline(lidar)`
`points(lidar[,1],fitted(fit),col="red")`
`plot(lidar[,1],residuals(fit),col="blue")`
`abline(h=0,col="red")`

These plots show that after accounting for the mean via a smoothing spline fit there is strong heteroscedasticity in the residuals.

- (c) The script `meanAndVarNonparRegn.Rs` extends the Bayesian penalized spline model to fit a Bayesian penalized spline version of the mean and variance function model (2) to the LIDAR data. The matrix algebraic form of the model is:

$$y | \beta, u, \gamma, v \stackrel{\text{ind.}}{\sim} N(\mathbf{X}\beta + \mathbf{Z}_u \mathbf{u}, \exp(\mathbf{X}\gamma + \mathbf{Z}_v \mathbf{v}))$$

and the relevant directed acyclic graph is:



- (d) Type `source("meanAndVarNonparRegn.Rs")` to fit this more elaborate model to the LIDAR data. The script produces graphics for both the mean and standard deviation (square root of the variance) functions.

5. Linear Mixed Model Analyses with Non-Bayesian R Software

In 37458 *Advanced Bayesian Methods* the overarching theme is Bayesian inference, with fitting usually achieved using Markov chain Monte Carlo-based inference engines such as Stan. However, many of the models considered in 37458 *Advanced Bayesian Methods* have non-Bayesian, or *frequentist* versions and corresponding software. In this section of Laboratory 4 we revisit the rats and bacteria data sets and analyse them using the `lme4` package.

- (a) `install.packages("lme4")` to install the `lme4` package.
- (b) `source("loadRatsData.Rs")` to load and plot the rats data.
- (c) `library(lme4)`
`allData <- data.frame(weeks,weight,idnum,weekssq=weeks^2)`
`fitModel1 <- lmer(weight ~ weeks + (1|idnum),data=allData)`
`fitModel2 <- lmer(weight ~ weeks + (weeks|idnum),data=allData)`
`fitModel3 <- lmer(weight ~ weeks + weekssq + (weeks|idnum),`
`data=allData)`

- (d) Issues the following commands to obtain residual plots of each of the three fitted models:

```
plot (fitModel1)
plot (fitModel2)
plot (fitModel3)
```

The last residual plot is the most pleasing in that it is not showing any strong (“bow tie” or “arch-shaped”) patterns – suggesting that the third model is the most appropriate for the rats data.

- (e) `anova (fitModel1, fitModel2, fitModel3)`

This command leads to a so-called *analysis of variance* comparison between the three models, with chi-squared tests used to formalise the comparison. Notice the small p -values in the last column, which show strong statistical significance for the Model 2 versus Model 1 and Model 3 versus Model 2 comparisons.

Next we revisit the Northern Territory bacteria data, which has the additional complication of the response variable being binary.

- (f) Read in and assign the data using

```
bacteria <- read.table("bacteria.txt", header=TRUE)
```

Fit a random intercept and slope (for the effect of week) using:

```
fitRanIntAndSlp <- glmer(bacteriaIsPresent ~ week + treatIsDrug
                        + treatIsDrugPlus + (week|idnum),
                        data=bacteria, family=binomial)
```

Assess properties of the fit including possible significance of predictors using:

```
print(summary(fitRanIntAndSlp))
```

It is seen that `treatIsDrug` has borderline significance. However, note that the estimated correlation between the intercept and slope is exactly 1.00 which indicates a troublesome maximum likelihood boundary-on-parameter-space problem.

Given issues with the previous fit, we now try just having a random intercept for each person in the study:

```
fitRanIntOnly <- glmer(bacteriaIsPresent ~ week + treatIsDrug
                      + treatIsDrugPlus + (1|idnum),
                      data=bacteria, family=binomial)
print(summary(fitRanIntOnly))
```

The correlation problem is overcome with this simpler model. Moreover, `treatIsDrug` is significant at the 5% level.

- (g) `anova (fitRanIntOnly, fitRanIntAndSlp)`

This time the p -value for the chi-squared test is high, which indicates that the random intercept and slope model does not offer a significant improvement over the random intercept model.

The default methodology used by `glmer()` is called *Laplace's method* and is known to give inaccurate p -values for binary response data. Issue the commands:

```

fitHighAccur <- glmer(bacteriaIsPresent ~ week + treatIsDrug
                      + treatIsDrugPlus + (1|idnum),
                      data=bacteria, family=binomial,
                      nAGQ=100)

print(summary(fitHighAccur))

```

This does a more accurate fit using so-called adaptive Gauss-Hermite quadrature. Notice that the p-value for `treatIsDrug` is somewhat higher than the previous fit, but is still significant at the 5% level.

6. Geoadditive model analyses with non-Bayesian R software

The Sydney real estate data set within the R package `HRW` is a relatively big data set corresponding to sales of 37,676 houses in Sydney in or around the year 2001. For each of the 37,676 houses, apart from sale price there are 38 other variables such as distance to the General Post Office in Sydney's central business district. The size of the data means that Bayesian analyses involving Markov chain Monte Carlo inference engines such as `Stan` can be very slow – potentially requiring hours to run on a standard desktop or laptop computer, depending on the complexity of the model. In this part of Laboratory 4 we will do a relatively quick non-Bayesian geoadditive model analysis using the `mgcv` package. A geoadditive model is an extension of a generalized additive model in which a geographical effect is incorporated using bivariate penalized splines.

To keep the analysis quick, here we work with a 20% random sub-sample of the houses and only four predictors apart from longitude and latitude. These predictors have abbreviations as follows:

`lotSize` — lot size in square meters but with some imputation.

`income` — average weekly income of the suburb in which the house is located.

`distToCoastline` — distance from house to the nearest coastline location (kilometres).

`distToGPO` — distance from the house to the General Post Office in Sydney's central business district (kilometres).

Now issue the following commands:

```

library(mgcv)
SydneyRealEstateSubData <- read.table("SydneyRealEstateSubData.txt",
                                     header=TRUE)
fit <- gam(logSalePrice ~ s(longitude, latitude, k=100)
           + s(lotSize, k=25)
           + s(income, k=25)
           + s(distToCoastline, k=25)
           + s(distToGPO, k=25),
           data=SydneyRealEstateSubData)
plot(fit, pages=1, se=FALSE, scheme=2)

```

Submission component of Laboratory 4

This is quite informal and can be done in half a page or less. As a resident of Sydney, try to explain some of the patterns apparent in the `distToCoastline` and `distToGPO` plots.