UNIVERSITY OF TECHNOLOGY SYDNEY

School of Mathematical and Physical Sciences

**37457 Advanced Bayesian Methods**

# LABORATORY 4

**Due time and date:** 9:55am, Wednesday 6th November, 2024.

**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 37457 is advanced Bayesian methods it is worth knowing about non-Bayesian (or *frequentist*) software for analysis of grouped data.

1. **Downloading Required Files**

   (a) 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.

   (b) Open a web browser and go to the web-page:
   `matt-p-wand.net/37457.html`

   (c) Scroll down to the heading **Files for Laboratory 4** and download each of the files underneath this heading into the same folder. There are 7 files in total with names:

   - `SLRmeasErr.R`
   - `npRegMis.R`
   - `lidar.txt`
   - `meanAndVarNonparRegn.R`
   - `loadRatsData.R`
   - `SydneyRealEstateSubData.txt`

   (d) 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.

   (e) 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.

   (f) Make sure that the package HRW is installed on the computer that you are using. A quick check is to issue the command `library(HRW)` and see whether or not an error is produced. If the package is not present then the command:

   ```
   install.packages("HRW")
   ```

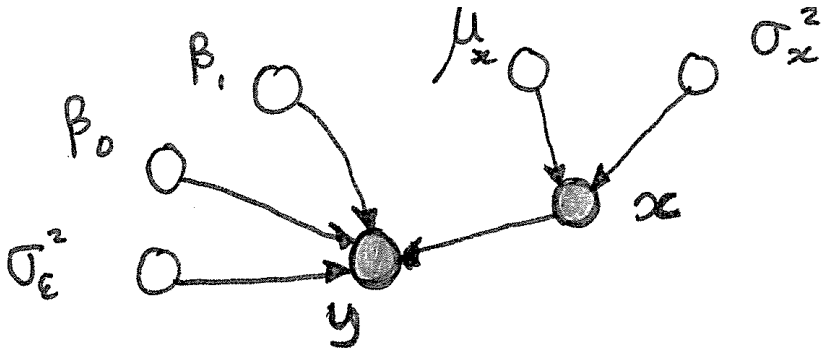
   is required for this laboratory.

2. **Simple Linear Regression with Measurement Error**

   Standard simple linear regression involves estimation of the parameters in

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

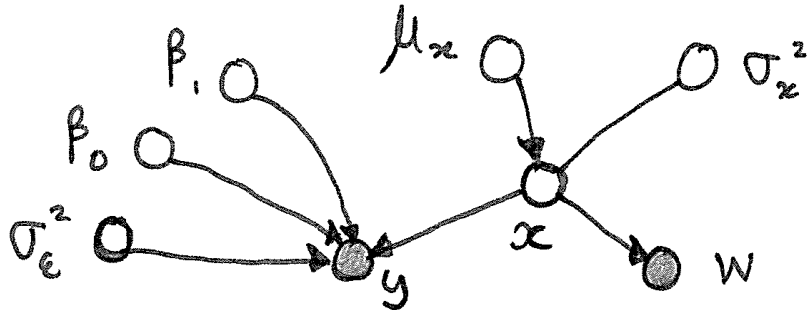
   based on the data $(x_i, y_i)$. Suppose that it also reasonable to assume that $x_i \overset{\text{ind.}}{\sim} N(\mu_x, \sigma_x^2)$ and estimation of $\mu_x$ and $\sigma_x^2$ is of interest. The relevant directed acyclic graph is:




   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

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

   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 $\beta_0$, $\beta_1$ and $\sigma_\varepsilon^2$ are estimated correctly.

   (a) Download the script `SLRmeasErr.R` from the subject web-site.

   (`http://matt-p-wand.net/37457.html`).

   (b) Issue the command:

   `source("SLRmeasErr.R")`

   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.

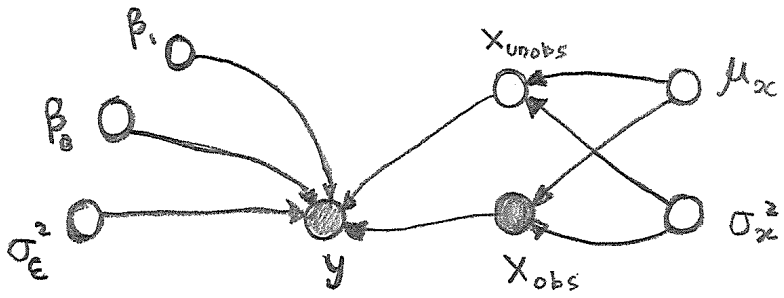
3. **Penalized Spline Nonparametric Regression with Incomplete Data**

   (a) Download the script `npRegMis.R` 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 $x_i \overset{\text{ind.}}{\sim} N(\mu_x, \sigma_x^2)$.
- 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 $\kappa_1, \ldots, \kappa_K$. Here $a_+ = \max(a, 0)$.

The relevant directed acyclic graph is:




Also note that the data in `npRegMis.R` 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.R")` to run the script. It may take a few minutes to run, during which you could e.g. check out the latest U.S. election news, watch some of the best goals of 2024 or study the demise of Bennifer 2.0. The last set of plots may show some interesting bimodal posterior density functions. This is due to the periodicity of the signal, with more that one $x$ value matching a given $y$ value.

4. **Penalized Spline Nonparametric Regression with Variance Function Estimation**

The simplest nonparametric regression model:

$$y_i \overset{\text{ind.}}{\sim} N\big(f(x_i), \sigma_\varepsilon^2\big), \quad 1 \leq i \leq n \tag{1}$$

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:

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

(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)
```

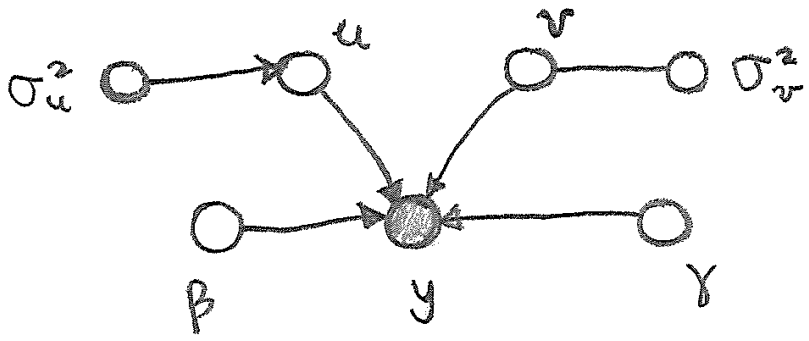
```
points(lidar[,1],fitted(fit),col="red")
plot(lidar[,1],residuals(fit),col="blue")
abline(h=0,col="red")
```

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.R` 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:

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

and the relevant directed acyclic graph is:




(d) Type `source("meanAndVarNonparRegn.R")` 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 37457 *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 37457 *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.R")` 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 1 comparisons.

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. Please e-mail your submission to the lecturer (`matt.wand@uts.edu.au`).