APTS Week 1: Statistical Computing Assessment

The work provided here is intended to take up to half a week to complete. Students should talk to their supervisors to find out whether or not their department requires this work as part of any formal accreditation process. Departments decide on the appropriate level of assessment locally, and may choose to drop some (or indeed all) of the parts, accordingly. In order to avoid undermining institutions' local assessment procedures the module lecturer will not respond to enquiries from students about this assignment (unless they spot errors in this problem sheet).

Q1. Consider the linear mixed model for a response vector $\mathbf{y}$:

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{Z}\mathbf{b} + \epsilon, \quad \mathbf{b} \sim N(\mathbf{0}, \mathbf{I}\sigma_b^2), \quad \epsilon \sim N(\mathbf{0}, \mathbf{I}\sigma^2).$$

where $\mathbf{X}$ and $\mathbf{Z}$ are (fixed) model matrices, β, σ_b^2 and σ^2 are parameters, and $\mathbf{b}$ and ϵ are independent.

To simulate data from this model, run the following code:

```
set.seed(10)
n <- 100;n.b <- 10;n.beta <- 5
## X and Z are fixed in the model, not random. Random numbers
## used only to generate arbitrary examples, here....
X <- cbind(1,matrix(runif(n*n.beta-n),n,n.beta-1))
Z <- matrix(runif(n*n.b),n,n.b)
beta <- rep(1,n.beta)
b <- rnorm(n.b)
y <- X%*%beta + Z%*%b + rnorm(n)
```

You'll use the data, $\mathbf{y}$, simulated here, along with the corresponding $\mathbf{X}$ and $\mathbf{Z}$, to experiment with *fitting* linear mixed models (so from now on pretend that you don't know what values β, σ_b and σ had). Next, you should attempt the following questions:

Q1.a) With pencil and paper, find the (marginal) expectation, $\boldsymbol{\mu}$, and covariance matrix, $\mathbf{V}$, of $\mathbf{y}$. State the (marginal) distribution of $\mathbf{y}$. Here we marginalise (integrate) over the random effects, $\mathbf{b}$, and the errors ϵ, but still condition on everything else.

Q1.b) Write an R function that evaluates the log likelihood of the parameters β, σ_b^2 and σ^2 given data $\mathbf{y}$. To maximise the log likelihood of the model using unconstrained methods, it is better to use a parameterization that guarantees positive variances. Hence, makes sure that the function accepts the unconstrained parameter vector $\boldsymbol{\theta}^T = (\beta^T, \rho_b, \rho)$ as input, where $\rho = \log(\sigma)$ and $\rho_b = \log(\sigma_b)$.

Q1.c) Use `optim` to maximise your likelihood (note that `optim` minimizes by default; see the documentation for how to do maximisation, and for how to choose optimisation method).

Q1.d) In fact, using general purpose optimisation methods to find the optimising β is a bit wasteful. Given the variance parameters, closed form expressions for the β maximising the likelihood are available, and might as well be used. Then it is only necessary to use general methods for the variance parameters. The likelihood considered only as a function of the variance parameters, with the corresponding MLEs of β *plugged in* is termed a *profile likelihood*. Show that, given the variance parameters, the log-likelihood is maximised by the β minimising

$$(\mathbf{y} - \mathbf{X}\beta)^{\mathsf{T}}\mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\beta) = \|\mathbf{R}^{-\mathsf{T}}(\mathbf{y} - \mathbf{X}\beta)\|^2$$

where $\mathbf{R}^{\mathsf{T}}\mathbf{R} = \mathbf{V}$. ($\|\mathbf{x}\|^2 = \mathbf{x}^{\mathsf{T}}\mathbf{x}$ here.) Hence, produce a 'profile log likelihood' function equivalent to your previous log likelihood function. Your function should accept a vector of variance parameters as its first argument, and should return the corresponding profile log likelihood value. You might want to return the corresponding β values as an attribute of the return value, e.g.

```r
logLikp <- function(theta,y,X,Z) {
  #
  # YOUR CODE HERE
  #
  attr(ll,"beta") <- beta
  ll
}
```

Q1.e) Use `optim` to maximise your profiled log likelihood function, and confirm that you get near identical parameter estimates to those you obtained before.

Q2. Derive the composite Newton–Cotes quadrature rule with $k = 5$, implement it and test it on a suitable example. If using the code in the lecture notes, this would involve extending the definition of `newton.cotes`. You can use a closed or open scheme.

Q3. Let $U : \mathbb{R}^2 \to \mathbb{R}$ be defined as

$$U(x) = 15x_1^2 + 10x_1x_2 + 10x_2^2 - 110x_1 - 20x_2 + \sin(x_1 + x_2).$$

Define the probability density function

$$p(x) = \frac{1}{Z} \exp(-U(x)),$$

where

$$Z = \int_{\mathbb{R}^2} \exp(-U(x)) \mathrm{d}x.$$

Approximate Z using importance sampling.

Hint: consider approximating the important terms in U using a quadratic form.