Introduction to Bayesian Inference: Practical Exercises

You will be using WinBUGS 1.4.1 for these practicals which you should have already installed on your PC.

All the data and other files you will need for the practicals were provided in a zip file, which you should have unzipped and saved in the directory `C:\data`.

If you didn't receive this zip archive, and/or don't have WinBUGS installed on your PC, please ask as we have a copy on a USB memory stick that you can install.

Solutions for all the exercises are also provided in the directory `C:\data`.

Scripts have also been prepared which can be used to run most of the models rather than using the menu click-and-point interface. You are advised to start by using the menu interface for the first practical, so that you understand the steps involved in setting up and running a model in WinBUGS. However, feel free to use the scripts to run models for the other practicals (or write your own scripts to do this) if you prefer.

Further detailed instructions on using `WinBUGS` can be found in the handout *Hints on using WinBUGS*, and in the on-line WinBUGS User Manual (see the Help menu in `WinBUGS`).

Notes on scripts in WinBUGS 1.4.1

The standard way to control a WinBUGS model run is using the click-and-point menu interface. However, there is also a script language which enables all the menu options to be listed in a text file and run in batch mode. Scripts to run all of the models in the practical exercises are included in the data zip file you were provided with. Using scripts is generally much quicker than clicking-and-pointing. However, we recommend you start with the click-and-point approach so that you understand how WinBUGS works. The click-and-point approach is also better for debugging a model.

The section *Batch mode: Scripts* of the on-line User Manual in WinBUGS gives details of the script language.

To run a script, click on the window containing the script file to 'focus' it, and then select Script from the Model menu.

Notes on scripts:

- The model code, data and each set of initial values called by the script have to be stored in separate files
- Once the script has finished executing, the analysis is still 'live' and you can continue use the WinBUGS menus interactively in the usual way.
- The script language currently has very limited error handling, so check that your model compiles correctly and that the data and initial values can all be loaded OK using the usual WinBUGS menu/GUI interface, before setting up a script to carry out a full analysis.

Notes on DIC tool in WinBUGS 1.4.1

- Remember that you should run sufficient burn-in iterations to allow the simulations to converge *before* you set the DIC tool, since you cannot discard burn-in values from the DIC calculations after the monitor has been set.
- There is an FAQ about DIC on the BUGS web site: `http://www.mrc-bsu.cam.ac.uk/bugs/winbugs/dicpage.shtml`

There are also a couple of small bugs in the DIC tool in WinBUGS 1.4.1:

- The dialog box doesn't function properly, so if you close the DIC tool window, you cannot open it again within the current model run! **So, keep the DIC tool open.**

These bugs will be fixed in the next 'patch' to WinBUGS 1.4!

Notes on BUGS language

- A list of distributions and their syntax in the BUGS language is given in the on-line Help: `Help: User Manual: Distributions`.
- A list of functions and their syntax in the BUGS language is given in the on-line Help: `Help: User Manual: Model Specification: Logical Nodes`.
- Details of data formats accepted by WinBUGS are given in the on-line Help: `Help: User Manual: Model Specification: formatting of data`.
- Syntax for writing 'loops' and indexing vectors and arrays in the BUGS language is given in the on-line Help: `Help: User Manual: Model Specification: Arrays and indexing` and `Help: User Manual: Model Specification: repeated structures`.

Practical Class 1: Introduction to Monte Carlo using WinBUGS

A. Coin example

1. Start WinBUGS
2. Open `coins.odc` from the `C:\data` directory: this program will simulate throws of 10 balanced coins and record which give 8 or more heads.
3. First try running using the interactive interface: if necessary, full instructions are given in *Hints on Using WinBUGS*.
   - Open up the Model : Specification window.
   - Highlight `model` click on check model (this should also work without highlighting, provided `coins.odc` is the open window).
   - There is no data to load, so just click compile.
   - There is no real need to provide initial values as this is a Monte Carlo forward sampling from a known distribution, but the program still requires some initial values to be given. Click gen inits to generate some.
   - Open up the Model : Update window and generate 1000 iterations.
   - Then open up the Inference : Samples window, type `Y` in the node window and then click set. This sets the monitor. Repeat for `P8`. Type `*` in the node window to indicate all monitored quantities.
   - Click trace to generate traces of the simulated values.
   - Then do another 1000 updates. stats then gives summary statistics.
4. *Using the script language*: `WinBUGS` version 1.4 includes a facility for running models in batch mode using a script. The script to run the first model for the coins example in `WinBUGS` is in file `coins-script.odc` in the `C:\data` subdirectory. Open this file and look at the commands to make sure you understand them (see section *Batch mode: Scripts* of the on-line User Manual in `WinBUGS` for more details). To run the script, click on the window containing the script file to focus it, and then select Script from the Model menu.

   See notes on scripts at the beginning of the Practical Exercise sheet for further details.
5. By editing the model file, find the probability that a clinical trial with 30 subjects, each with probability 0.7 of response, will show 15 or fewer responses.

B. Drug example from lecture 1

1. Open `drug-MC.odc` and carry out a `WinBUGS` run for this model, obtaining the results shown in the lectures (slides 1-25 to 1-26). You should be able to run it using the previous instructions (question A) and the short list given in the lectures (slide 1-19). Otherwise full details are given in **Running a model in** `WinBUGS` of the hints handout. If stuck, a script `drug-MC-script.odc` is available.

2. Edit the model code to specify a Uniform(0, 1) prior on the response rate `theta`, and re-run the analysis. (Note: the syntax for the uniform prior in WinBUGS is `dunif(a, b)` where `a` and `b` are the lower and upper bounds. The values of `a` and `b` can either be specified in the data file, or directly in the BUGS code (e.g. `a <- 1`), or just replace `a` and `b` by their values in the `dunif` statement.)

    - Plot the predictive distribution for the number of successes.
    - What is now the predictive probability that 15 or more patients will experience a positive response out of 20 new patients affected?

C. Power example

1. Run this interactively using the menus from `predpower.odc`, and then from the script in `predpower-script.odc`. Check you get the answers shown in the lecture (slide 1-31).

2. What if the prior SD of sigma were reduced to 0.1?

D. Optional: Writing your own code

1. Generate 10000 observations from a standard t distribution from 4 degrees of freedom [WinBUGS code, `y ~ dt(0,1,4)`], and plot the density. Would you consider this a heavy -tailed distribution compared to the normal?

2. Find by simulation the expectation of the cube of a normal random variable with mean 1 and standard deviation 2 (remember the WinBUGS parameterisation of the normal is in terms of the precision = 1/variance), and y^3 is written as `pow(y,3)`.

Practical Class 2: Conjugate Bayesian inference using WinBUGS

A. Drug example from lecture 2

The `WinBUGS` code for fitting the beta-binomial model (slide 2-19) to the drug data can be found in file `drug-model.odc`. The data are in file `drug-data.odc` and 2 sets of initial values are in files `drug-in1.odc` and `drug-in2.odc` respectively.

1. Open these files and carry out a `WinBUGS` run for this model

   (a) Check and compile the model, following the instructions in section **Running a model in `WinBUGS`** of the hints handout.

   (b) Before you start updating, set *sample* monitors for the drug response rate, `theta`, the predicted number of positive responses in 40 future patients, `r.pred`, and the indicator of whether 25 or more positive responses are predicted, `P.crit` (See section **Monitoring parameter values** of the hints handout).

   (c) Run 10000 iterations (or more or less if you wish) to obtain samples from the posterior distribution of the model parameters.

   (d) Produce summary statistics for all the variables you have monitored.

   - Provide a point and interval estimate for the treatment response rate
   - What is the probability that 25 or more patients will experience a positive response out of 40 new patients to be administered the drug?

   (e) Produce kernel density plots of the posterior distribution for `theta` and the predictive distribution for `r.pred`

   A script to run this model in `WinBUGS` is in file `drug-script.odc` in the `C:\data` subdirectory, if you wish to use it.

2. Compare the model code (and data) with the code for the Monte Carlo analysis of the drug data that you carried out in practical 1. Make sure you understand the difference between the two analyses.

3. Edit the model code to specify a Uniform(0, 1) prior on the response rate `theta` (or equivalently, a Beta(1, 1) prior), and re-run the analysis.

   - How is the posterior estimate of `theta` affected?
   - How is the probability that 25 or more patients will experience a positive response out of 40 new patients affected?

B. THM data from Lecture 2

From scratch, implement the THM example of inference on the mean of a Normal distribution, slides 2-29 to 2-33. (Note — you will need to specify values for both the THM observations, rather than just inputting their mean as the data, so just enter both values as 130).

1. First implement the model with the informative prior specified in the lecture notes (slide 2-30). The syntax for the normal distribution in WinBUGS is `dnorm(mu, tau)` where `mu` is the mean and `tau` is the *precision* (= 1/variance)).

2. Now try using a non-informative prior on the unknown mean THM concentration (assume either a uniform prior with a wide range, or a normal prior with a large variance (small precision))

3. For each model, include statements in your WinBUGS code to:

   (a) obtain a sample from the predictive distribution of a future THM concentration in the zone

   (b) calculate the probability that the zone *mean* THM concentration exceeds 130 μg/l

   (c) calculate the probability that a future THM concentration measured in the zone exceeds 130 μg/l

C. Flying bombs (optional)

From scratch, implement the flying bomb analysis of Poisson data described at the end of Lecture 2.

Practical Class 3: Fitting Bayesian linear regression models in WinBUGS

A. Union density data

The data, some initial values and a basic model file assuming vague priors for each of the regression coefficients are provided in files `union-dat.odc`, `union-in1.odc`, `union-in2.odc` and `union-model.odc`. Note that the data for this example are in the 'rectangular array' data format, rather than the 'list' format. The 20 rows correspond to countries, and the columns give each of the four variables (response and 3 predictors). To load these data into `WinBUGS`, highlight the first row containing the column names and click on `Load data` at the appropriate stage (just clicking on `Load data` without highlighting the first row should also work, provided the data file is the focused window).

1. Open these files and carry out a `WinBUGS` run for this model to get the parameter estimates shown on slide 4-29. You will need to carry out checks for convergence and discard any burn-in samples before making any posterior inference. To do this, you should run 2 separate chains and use the Gelman-Rubin convergence diagnostic (slides 3-11 to 3-15) as follows:

   (a) Check and compile the model as before (see the instructions in section **Running a model in `WinBUGS`** of the hints handout), but remember to set the number of chains to be 2 (the default is 1) **before** you click *compile* (step 9 on the handout), and to load both sets of initial values (step 12 on the handout).

   (b) Set samples monitors on appropriate parameters (see section **Monitoring parameter values** of the hints handout).

   (c) Run 1000 iterations (updates), then look at history plots and autocorrelation plots of the sample traces and calculate the Gelman and Rubin convergence diagnostic (GR diag) for the parameters you have monitored.

   - Do the simulations look like they have converged?
   - If not, carry out some more updates and check again.

   (See section **Checking convergence** of the hints handout).

   (d) Once you are happy with convergence, carry out a further 5000 iterations (or more if you wish) to obtain samples from the posterior distribution of the model parameters.

   (e) Produce posterior summary statistics for the variables you have monitored, remembering to discard any burn-in samples. Check the Monte Carlo standard error of each variable to assess the accuracy of your estimates (See section **Obtaining summary statistics of the posterior distribution** of the hints handout, and slide 3-17 for a discussion of the MC error). How does the MC error change if you run more iterations? (Alternatively, look at how the MC error changes if you base your posterior summaries on fewer iterations — you can specify which iterations to summarise by setting the `beg` and `end` options in the *samples* tool in `WinBUGS`).

(f) Produce a box plot of the posterior distribution of the regression coefficients as shown on slide 4-30 (select 'box plot' from the 'compare' tool on the 'inference' menu; again, remember to discard any burn-in iterations by setting the `beg` option in this tool window). Also produce autocorrelation plots and bivariate scatter plots of the regression coefficients (for the latter, select 'scatter plot' from the 'correlations' tool on the 'inference' menu).

2. Edit the code to fit the same model, but with 'uncentred' covariates. How does this affect the convergence of the MCMC simulations and how does it affect the MC error? (Have a look at the history plots, GR diagnostics, autocorrelations, bivariate scatter plots and posterior summary statistics of the regression coefficients)

3. Edit the priors on the regression coefficients to implement (i) Stephens' informative prior (slide 4-33) and (ii) Wallerstein's informative prior (slide 4-32).

   - Note, if you wish to produce boxplots comparing the posterior distributions of each regression coefficient under the 3 priors, you will need to run all 3 models in the same code. This involves replicating the data. See the solutions file for details.

4. Western and Jackman (1994) carry out a series of sensitivity analyses to investigate how the posteriors of the regression coefficients depend on the priors. Their analysis involved gradually increasing the prior variance while keeping the prior mean fixed (at the informative values).

   (a) Edit your code for each of the informative prior models so that the prior variance on the relevant coefficient is multiplied by 10 (i.e. prior precision is divided by 10).

   (b) How are the conclusions regarding the effects of industrial concentration and labour force size affected under these new priors?

Practical Class 4: Fitting Bayesian GLMs and non-linear regression models in WinBUGS

A. Dugongs

The data, one set of initial values and a basic model file are provided in `dugongs-dat.odc`, `dugongs-in1.odc` and `dugongs-model.odc`.

1. Create a second set of initial values, and run the model to get the parameter estimates and model fit plot shown on slide 5-9

2. Change the prior to being uniform on $\log(\sigma)$: is there any influence? (Note — you must always specify initial values for the stochastic parameters, i.e. the parameters assigned prior distributions, so you will need to edit the initial values files here as well as the model code. Also remember that you cannot specify a function of parameters on the left side of a distribution in the BUGS language, so you will need to create a new variable equal to $\log(\sigma)$ and assign this new variable an appropriate prior. Since BUGS does not allow the same variable to appear more than once on the left side of a statement, you will need to specify this new variable using the following format:

   ```
   function.of.sigma <- new.variable
   ```

   rather than

   ```
   new.variable <- function.of.sigma
   ```

   (See solutions if you are confused!)

3. Check the posterior distribution for γ: do you think the uniform prior on this parameter is very influential?

B. Beetles

The data, some initial values and a basic model file are provided in `beetles-data.odc`, `beetles-in1.odc`, `beetles-in2.odc` and `beetles-model.odc`

1. Run the model to get the parameter estimates and plot shown on slide 5-3, and the history plot in 5-14

2. Try 'uncentering' the covariate and check the influence on the convergence of `beta`

3. Try different link functions: complementary log-log [$\log(-\log(1-p))$] and probit [$\Phi(p)$]. These can be very sensitive to initial values! Try the two different ways of implementing the probit model as described in the lecture notes (slide 5-5).

Practical Class 5: Predictions, model checking and model comparison

Predictions and model checking

A. Dugongs

1. Run the dugongs model from practical 4 and include statements to calculate:
   - standardised residuals
   - p-value for each residual
   - predictive distribution for each dugong, `Y.pred[i]`
   - Bayesian p-value calculating posterior probability that `Y[i]` > `Y.pred[i]`

   You can either edit the model code in `dugongs-model.odc` to add in the relevant statements to calculate residuals and p-values (and the corresponding script to set appropriate monitors), or use the pre-prepared code and script in `dugongs-check-model.odc` and `dugongs-check-script.odc`.

2. Try reproducing the box plots of standardised residuals and the 'QQ plot' of residual p-values against order-statistics shown in the lecture notes (slide 6-10), plus the plot showing the predictive fit of the model (slide 6-6). Also produce a 'QQ plot' of the predictive p-values. Compare the p-values calculated using the two methods. Box plots and 'caterpillar' plots (for the 'QQ plot') can be produced in Winbugs using the appropriate options from the `Inference-Compare` menu. Produce plots of residuals versus covariates and fitted values as well (use the `scatterplot` option from the `Inference-Compare` menu).

3. Edit the model code to include statements to predict dugong length at ages 35 and 40. Try producing the plot of these predictions similar to that shown on slide 6-5.

4. The file `dugongs-outl-dat.odc` contains a modified version of the data with one observation changed to be an outlier. Repeat the above analyses using this new data file (edit the scripts accordingly).

5. How could you modify the model specification to accommodate the outlier in the above analysis? How do the diagnostics change when you do this? Note: the standard deviation of a t distribution with precision parameter τ and v degrees of freedom is $\sqrt{\frac{v}{(v-2)\tau}}$.

6. Try fitting a (rather inappropriate) linear regression model
   ```
   mu[i] <- alpha + beta*x[i]
   ```
   and use DIC to compare the fit of this model to the non-linear model. (See notes on using DIC tool at start of practical exercise sheets). Note that there will be no `gamma` parameter in the linear model, but you can just leave it in the code as a prior distribution to avoid having to edit the initial values to remove the `gamma`'s.

B. Union density (optional if you have time)

1. Try editing the code for the union density example from Practical 3 to include calculation of standardised and/or predictive residuals and their associated p-values.

C. Beetles (optional if you have time)

1. Use DIC to compare the 3 alternative link functions for the beetles data from Practical 4.

Practical Class 6: Hierarchical models

A. THM data

A simulated set of data (slightly different from that used in the lecture notes) for the THM example can be found in file `thm-dat.odc`. The code for fitting the basic hierarchical model and some initial values are in files `thm-model.odc`, `thm-in1.odc` and `thm-in2.odc`

1. Run the basic model, setting monitors on the zone mean THM concentrations `theta` and on the VPC, random effects mean and variance and the residual error variance. Also monitor the DIC. Produce box plots of the posterior distribution of the zone mean THM levels, and make a note of the DIC.

2. Edit the model code to allow the residual error variance to be zone specific, and assume a hierarchical prior distribution for the logarithms of these variances (see the N-of-1 example in the lecture notes, slides 7-55 to 7-60, or have a look at the solutions file). Think about how you should calculate the VPC for this model. Remember to edit the initial values files as appropriate.

3. Run the above model, produce box plots of the posterior distributions of the zone mean THM levels and the within zone residual variances, and report summary statistics for the VPC that you have calculated. Compare the DIC for this model with that of the original model — which model is preferred?

4. The file `thm-x-dat.odc` contains an additional covariate for each water zone, indicating the (standardised) average residence time (time water remains in the supply pipes between the source and the tap) for the tap water supply in each zone. Longer residence times often lead to greater fluctuations in THM levels in tap water. Edit your code to replace the random effects model for the zone-specific variances with a model that allows these variances to depend on residence time. Re-run the model and compare its fit with the previous models using DIC. Which model is preferred? Is there evidence of an association between variability in THM levels and residence time in these data?

B. Modelling longitudinal data: Repeated measurements of CD4 counts in HIV-infected patients

This example uses (simulated) data from a clinical trial comparing two alternative treatments for HIV-infected individuals. 80 patients with HIV infection were randomly assigned to one of 2 treatment groups (`drug` = 0 (didanosine, ddI) and `drug` = 1 (zalcitabine, ddC)). CD4 counts were recorded at study entry (time `t` = 0) and again at 2, 6 and 12 months. An indicator of whether the patient had already been diagnosed with AIDS at study entry was also recorded (`AIDS` = 1 if patient diagnosed with AIDS, and 0 otherwise). The data can be found in files `CD4-dat.odc` and `CD4-time-dat.odc`.

1. Specify a non-hierarchical (i.e. fixed effects) linear regression model to examine the effect of drug treatment, AIDS diagnosis at study entry, and time since entry to clinical trial on CD4 count (the model is similar to the non-hierarchical model for the Hepatitis data discussed in Lecture 7, slides 7-36 to 7-43).

(a) Run the model and monitor the slope and intercept parameters, the regression coefficients for the effects of treatment and AIDS, and the residual error variance. You should also set a monitor on the matrix of parameters corresponding to the 'fitted values' (i.e. the mean of the normal likelihood you have specified for the CD4 counts) for patients 1 to 10 (e.g. if you have called the mean `mu` then set a monitor on `mu[1:10, ]`). You will need the samples of these fitted values to produce the model fit plots (see below). After convergence, monitor the DIC as well.

(b) Produce summary statistics of the monitored variables, and note the DIC.

(c) Produce 'model fit' plots to show the 2.5%, 50% and 97.5% quantiles of the fitted regression line for each of patients 1 to 10 (select 'model fit' from the 'compare' tool on the 'Inference' menu).

2. Modify your code for the previous model to include a random intercept and a random slope (i.e. time coefficient) for each patient. Treat the coefficients for the effects of drug treatment and AIDS as fixed (i.e. not random effects) as before. Assume independent prior distributions for the random intercepts, and for the random slopes. Remember that you will also need to give hyperprior distributions for the parameters of these random effects distributions (i.e. for the random effects means and variances). To start with, you should use a *non hierarchically centered* parameterisation for this model (i.e. the random effects should have mean zero, with the overall means included as separate coefficients in the linear predictor — see solutions if you are not sure). You will also need to specify initial values for any additional parameters included in your model. Note that to save you typing in initial values for all 160 random effects, just enter and load initial values for the population level parameters (i.e. the overall intercept and slope, plus the other regression coefficients and the precision parameters) and then select the `gen inits` option from the model specification menu — this will cause `WinBUGS` to automatically generate initial values for the random effects from their prior distributions.

   (a) Run the model and monitor on the same parameters as before, plus the patient-specific slope and intercept parameters (random effects) for patients 1 to 10 (don't monitor all 80 patients as storing the sampled values for all the random effects will quickly start to 'clog-up' the computer's memory), and the standard deviations of the random effects. Monitor the DIC as well.

   (b) Produce summary statistics and plots as before, and compare your results with those from the non-hierarchical model. Also produce history and autocorrelation plots (and keep these for comparison with the hierarchical centered model in the next part). Compare the DIC values to choose the most appropriate model.

3. Edit the model code to fit a hierarchically centered version of the model (Hint: you will need to centre both the random intercept and the random slope about their overall means). Run the model and compare the history and autocorrelation plots with those from the un-centered model above. Does hierarchical centering make much difference in this model?

4. Repeat the above analysis, but specify a correlated (bivariate normal) prior distribution on the random slope and intercept parameters for each patient.