4. A FIRST SESSION IN WINBUGS: THE “MODEL OF THE MEAN”

4.1. INTRODUCTION

To familiarize ourselves with WinBUGS and key features of a Bayesian analysis in practice (such as prior, likelihood, MCMC settings, initial values, updates, convergence etc.), we will first run an analysis directly within WinBUGS of what is perhaps the simplest of all models – the “model of the mean”. That is, we estimate the mean of a normal population from a sample of measurements taken in that population. Our first example will deal with body mass of male peregrines (Fig. 2-1).

Fig. 4-1: Male peregrine falcon (*Falco peregrinus*) wintering in the French Mediterranean, Sète, 2008 (Photo J.-M. Delaunay).

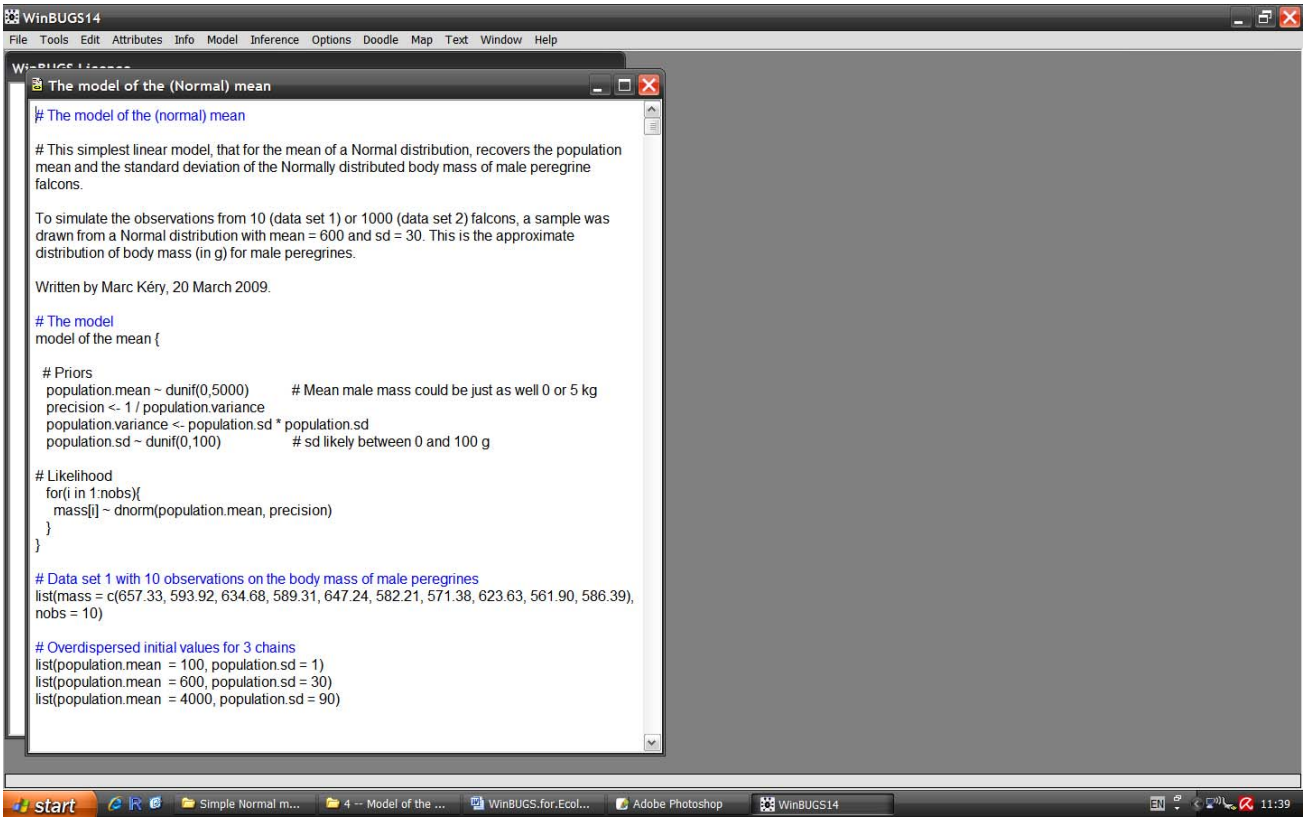
As an aside, this example emphasizes that even something that most people would not think of as a model – taking a simple average – in fact already *implies* a model that may be more or less adequate in any given case and is thus not as innocuous as many would think. For instance, using a

simple average to express the central tendency of a skewed population is not very useful and the "model of the mean" is then not adequate. Or taking an average over nonindependent measurements, such as diameter of each of several flowers measured within each of a sample of plants, is also not such a good idea since the "model of the mean" underlying that average does not account for dependencies among measurements (as would, for instance, a mixed model). Finally, the computation and usual interpretation of a standard error for the population mean (as `SD(x)/sqrt(n(x))`) is implicitly based on a "model of the mean" where each measurement is independent and the population distribution reasonably close to a normal or at least symmetric.

4.2. SETTING UP THE ANALYSIS

To open WinBUGS, we click on its bug-like icon. WinBUGS uses files in its own ODC format. To create an ODC file, you can either modify an existing ODC file (e.g., one of the examples contained in the WinBUGS help: go to Help > Examples vol I and II) or write a program in a text editor such as Word, save it as a text file (with ending `.txt`) and read it into WinBUGS and save it as an ODC document. Within WinBUGS, these files can be edited in content and format.

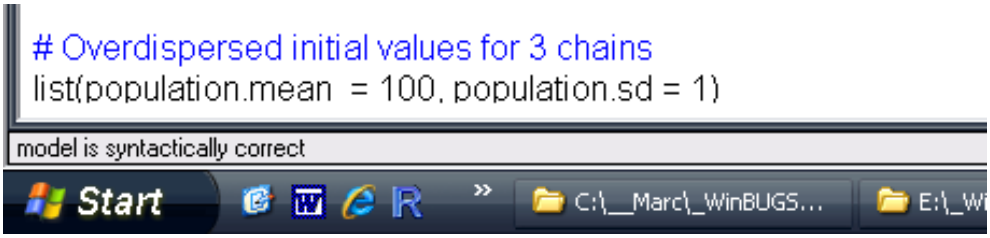
Here, we open the file called "`The model of the (normal) mean.odc`" (from the book website) via File > Open. We see something like this:



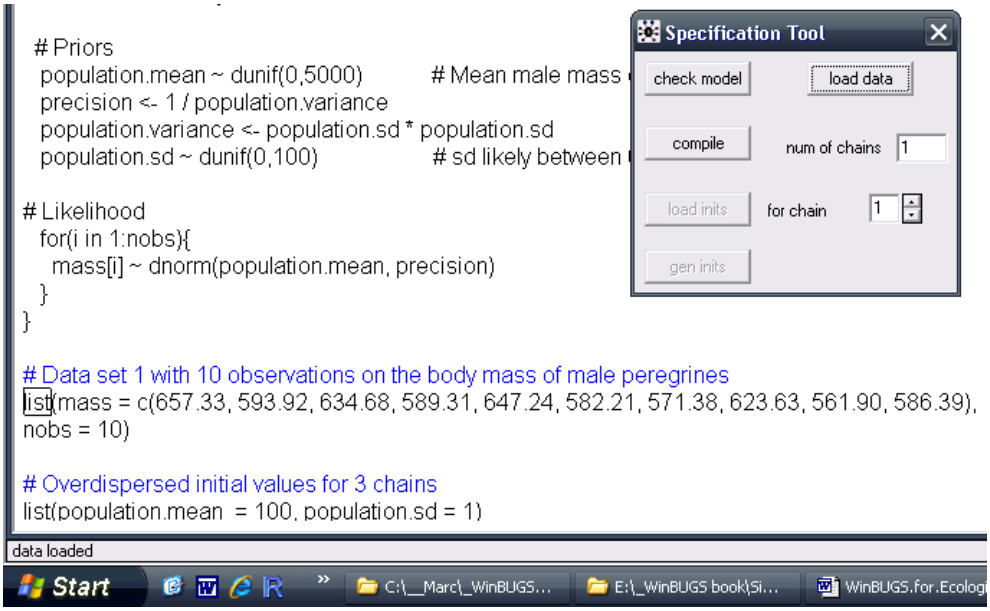
Note that we have two data sets available on the body mass of male peregrines. Both were created by using program R to produce random normal numbers with specified values for the mean and the standard deviation. One data set contains 10 observations and the other contains 1000 (scroll down to see them). Male peregrines in Western Europe weigh on average about 600 g and Monneret (2006) gives a range of 500–680 g. The assumption of a normal distribution of body mass therefore implies a standard deviation of about 30 g.

To run a Bayesian analysis of the "model of the mean" for one of these data sets, we have to tell WinBUGS what the model is, what data to use, provide initial values from where to start the Markov chains and set some MCMC features, e.g., say how many Markov chains we want, how many draws from the posterior distribution we would like to have, how many draws WinBUGS should discard as a burnin at the start and perhaps by how much we want to thin the resulting chain to save disk space and reduce autocorrelation among repeated draws.

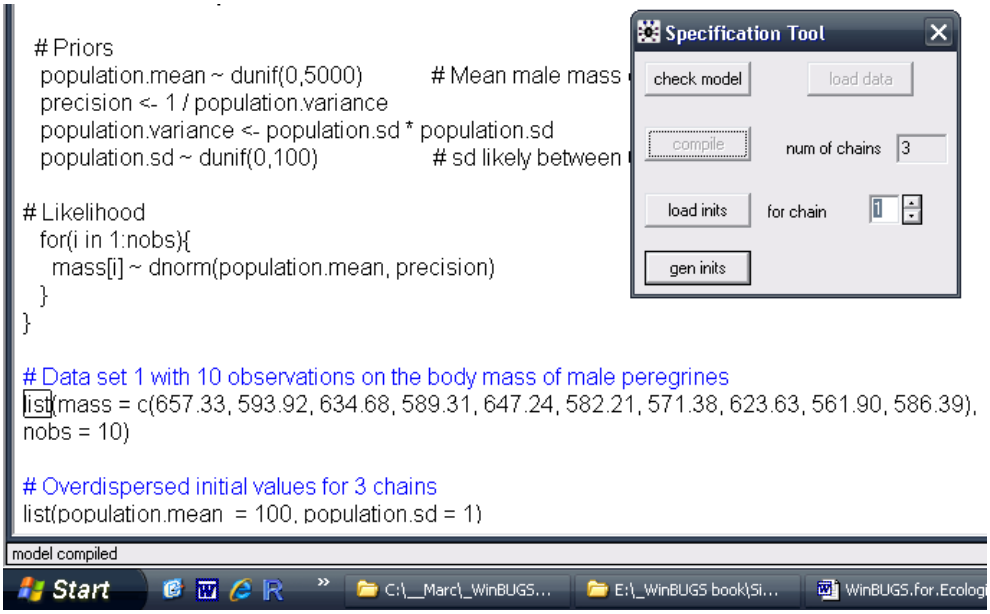
First, we have to tell WinBUGS what the model is. We select Model > Specification which causes the model specification tool window to pop up. Then we put the cursor somewhere on the word "model" or indeed anywhere within the model definition in the ODC document and press "check model", whereupon, if the model is syntactically correct, WinBUGS responds in the bottom left corner:



Next we load the data. We start with data set number 1, the ten measurements, mark the entire word "list" in the data statement and press "load data", whereupon WinBUGS responds in the bottom left corner "data loaded".

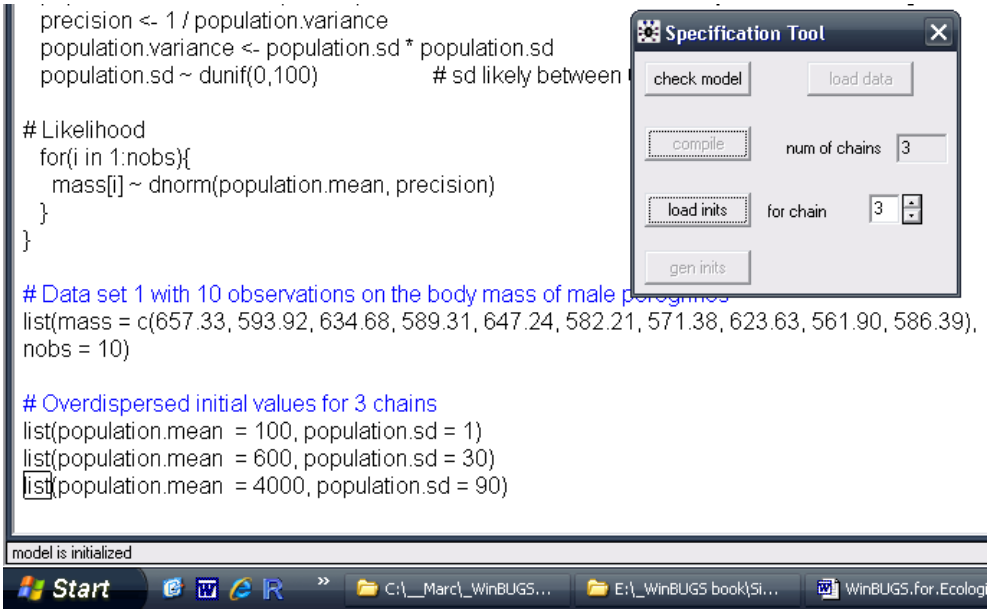


Then we select the number of parallel Markov chains that WinBUGS should "prepare" (i.e., compile), here, 3. This needs to be done *before the "compile" button is pressed* ! Then press the "compile" button.

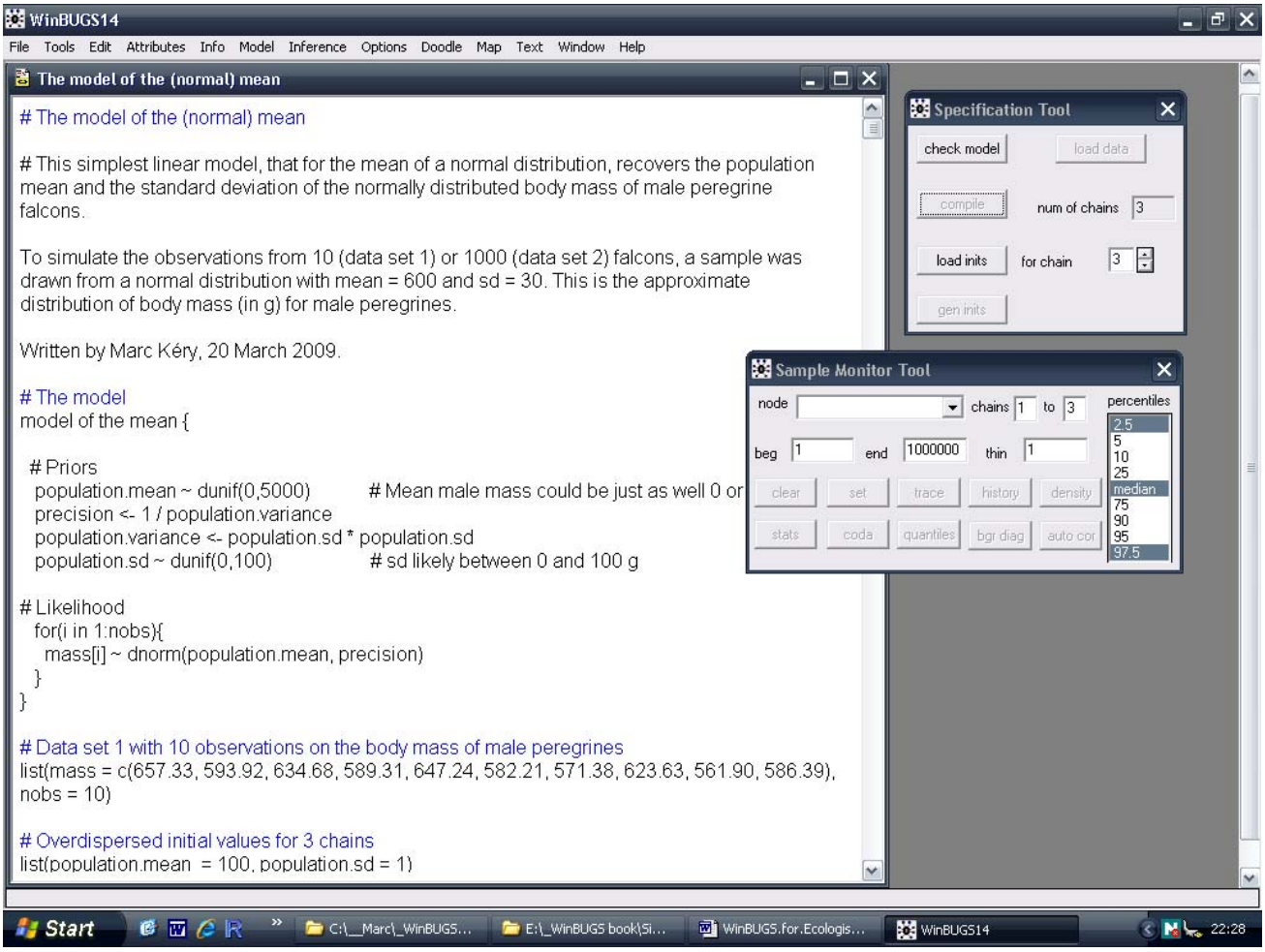


WinBUGS replies (in the bottom left corner) that the model is compiled. We still need to provide the starting values. One after another, mark the word "list" in that part of the program where the initial values are defined and then press "load inits". After you have done this once for each chain, the model is initialized and ready to run.

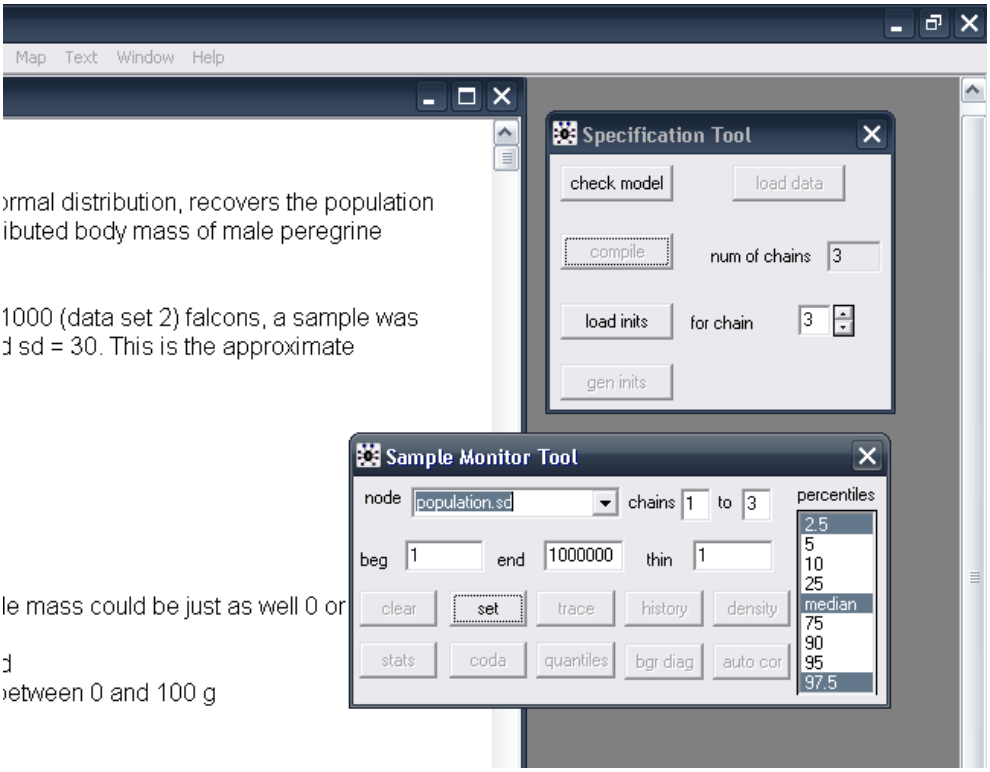
It is possible to let WinBUGS select the initial values herself by instead pressing the “gen inits” button. This works often, and may be necessary for some parameters in more complex models, but in general it is preferable to explicitly specify inits for as many parameters as possible.



Next, we choose Inference > Samples to tell WinBUGS what kind of inference we want for which parameter, which incidentally is called a *node* by WinBUGS. This makes the Sample Monitor Tool pop up.

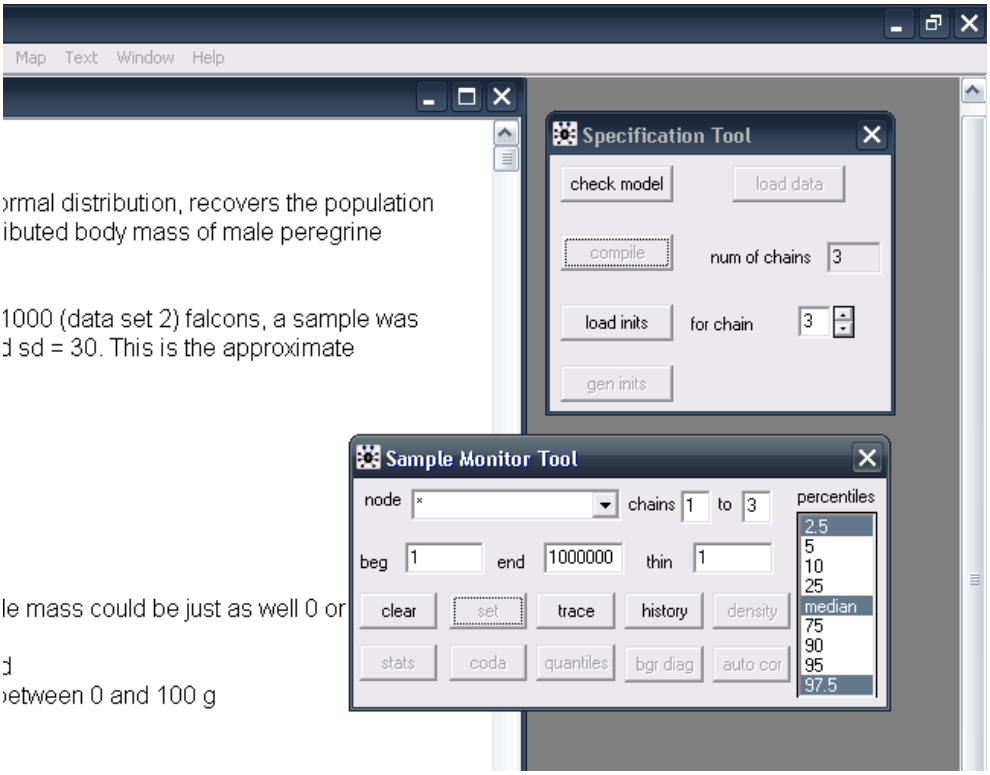


We write into the blank space next to the word “node” the names of those parameters whose posterior distributions we want to estimate by sampling them and press the “set” button. This button only becomes black, i.e., activated, once the correct name of a parameter from the previously defined model has been entered.

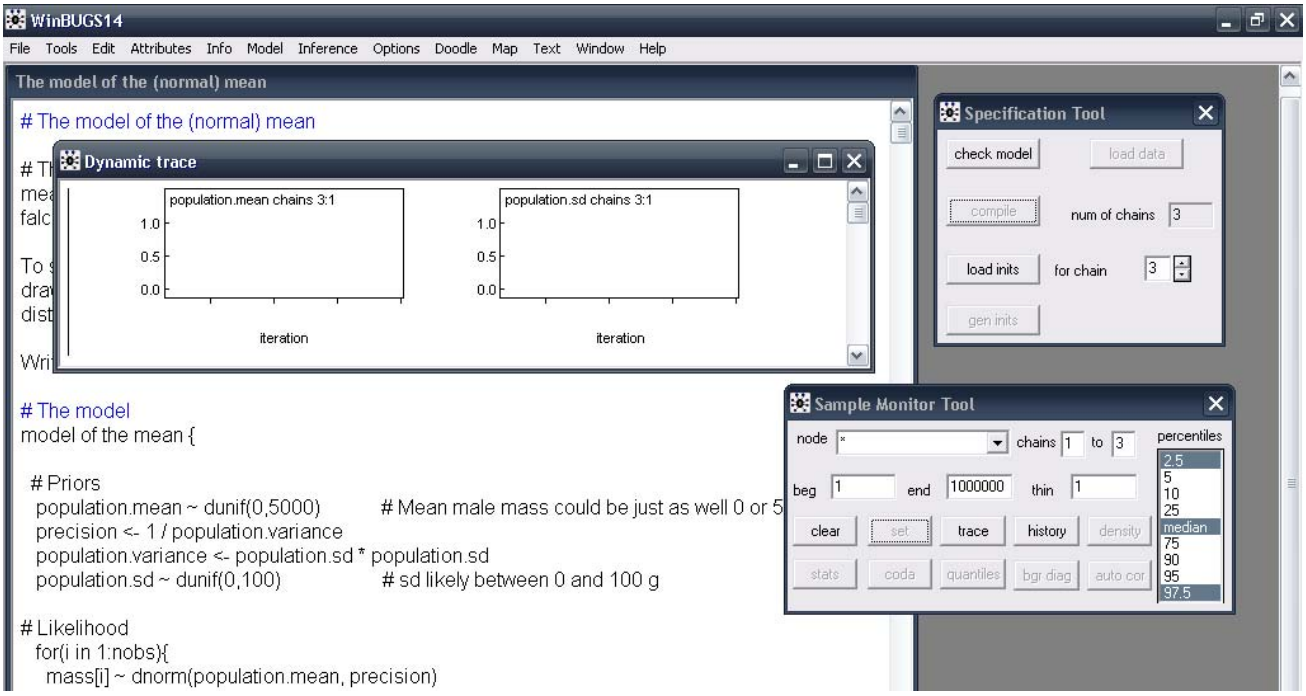


For models with many parameters, this can be tedious, because we have to enter manually all of their names (correctly !). Moreover, every time we want to rerun the model we have to repeat this whole procedure.

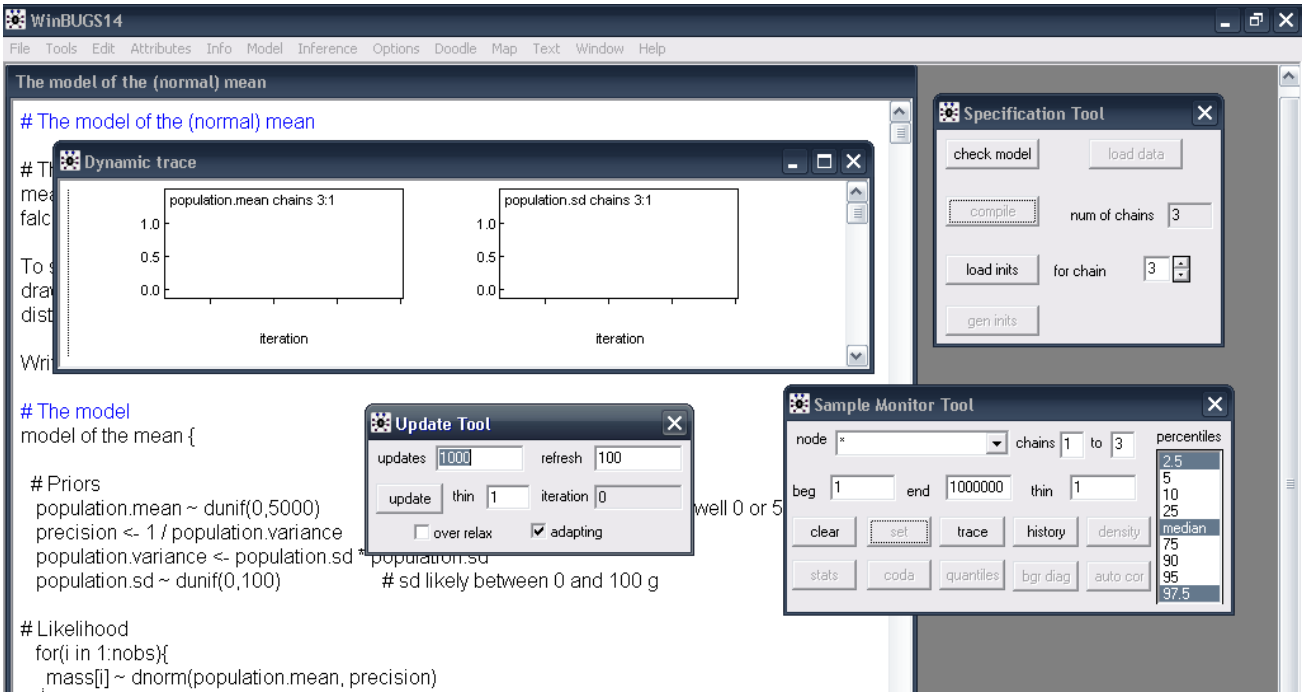
Once we have entered all parameters of whose Markov chains we want to keep track, i.e., for which we want to get estimates, we type an asterisk into the blank space to the right of the word “node”. This tells WinBUGS that it should save the simulated draws from the posterior distributions of all the parameters whose names we have entered previously. On typing the asterisk, a few further buttons become activated, among them “trace” and “history”.



We select “trace”, which will provide us with a moving time-series plot of the posterior draws for each selected parameter. (The button “history” produces a static graph of all draws up to the particular draw at which the history button is pressed, if this is done while WinBUGS is updating the model, or alternatively, of all draws, when it is done once WinBUGS is done with the run.) Another window pops up that has an empty graph for every selected parameter. There are several other settings in this window that could be changed, perhaps only at later stages of the analysis. An important one is the “beg” and the “end” setting. Together, they define which draws should be used when summarizing the inference about each parameter for instance by pressing the “density” or the “stats” buttons (see later). Changing the “beg” setting to, say, 100, specifies a burnin of 100. This means that the first 100 draws from each Markov chain are discarded as not representative of the stationary distribution of the chain (i.e., the posterior distribution of the parameters in the model).

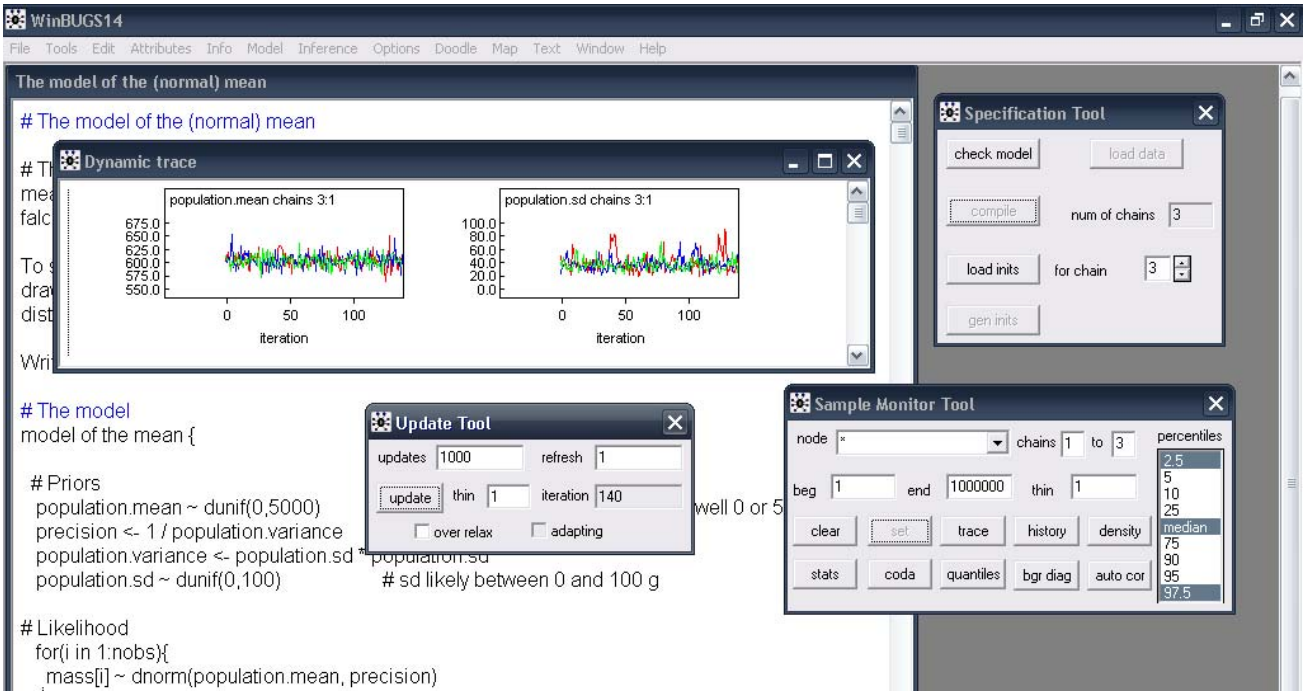


Now we are ready to start our simulation, i.e., start the Markov chains at their specified initial values and let the chains evolve for the requested number of times. To do this, we select Model > Update which makes a third window pop open, the “Update tool”.



4.3. STARTING THE GIBBS SAMPLER

We choose the default 1000 updates, but change the “refresh” setting to 1 or 10, which gives a more smoothly moving dynamic trace of the Markov chains. Then we press the “update” button and off we go. One continuous stream for each chain (here, 3) appears and represents the sampled values for each monitored parameter. Also, note that WinBUGS is explaining what it is doing at the bottom left corner, i.e., WinBUGS says that it is updating (=drawing samples from the posterior distribution).



Once the specified number of iterations (=updates, =draws) have been completed, WinBUGS stops and says that (on my laptop) the “updates took 62 s”.

4.4. SUMMARIZING THE RESULTS

From each Markov chain, we now have a sample of 1000 random draws from the joint posterior distribution of the two parameters in the model. For inference about the mass of male peregrines, we can summarize these samples numerically or we can graph them, either in one dimension (for each parameter singly) or in two, to look at correlations among two parameters. If we wanted more samples, i.e., longer Markov chains, we could just press the “update” button again and would get another 1000 draws added. For now, let’s be happy with a sample of 1000 (which is by far enough for this simple model and the small data set) and proceed with the inference about the parameters.

First we should check whether the Markov chains have indeed reached a stable equilibrium distribution, i.e., have *converged*. This can be done visually or by inspecting the Brooks-Gelman-Rubin (BGR) diagnostic statistic that WinBUGS displays on pressing the “bgr diag” button in the “Sample

monitor tool". The BGR statistic (graphed by the red line) is an ANOVA-type diagnostic that compares within- and among-chain variance. Values around 1 indicate convergence, with 1.1 considered as acceptable limit by Gelman and Hill (2007). Numerical summaries of the BGR statistic for successive sections of the chains can be obtained by first double-clicking on a BGR plot and then holding the CTRL key and clicking on the plot once again.

According to this diagnostic, the chains converge to a stationary distribution almost instantly; thus, we may well use all 3*1000 iterations for inference about the parameters. In contrast, for more complex models, nonzero burnin lengths are always needed. For instance, for some complex models, some statisticians may use chains one million iterations long and routinely discard the first half of every chain as a burnin.

Once we are satisfied that we have a valid sample from the posterior distribution of the parameters of the model of the mean and based on the observed ten values of male peregrine body mass, we can use these draws to make an inference. Inference means to draw a probabilistic conclusion about the population from which these ten peregrines came. Since we have created both this population and the samples from it, we know of course that the true population mean and standard deviations are 600 and 30 g, respectively.

We close some windows and press the "history", "density", "stats" and the "auto cor" buttons, which produces the following graphical display of information (after some rearranging of the pop-up windows):

Visual inspection of the time series plot produced by "history" again suggests that the Markov chains have converged. Then, we obtain a kernel-smoothed histogram estimate of the posterior distributions of both parameters. The posterior of the mean looks symmetrical, while that for the standard deviation is fairly skewed. The node statistics give the formal parameter estimates. As a Bayesian point estimate, typically the posterior mean or the posterior median (or sometimes also the mode) is reported, while the posterior standard deviation is used as a standard error of the parameter estimate. The range between the 2.5th and 97.5th percentiles represents a 95% Bayesian confidence interval and is called a credible interval.

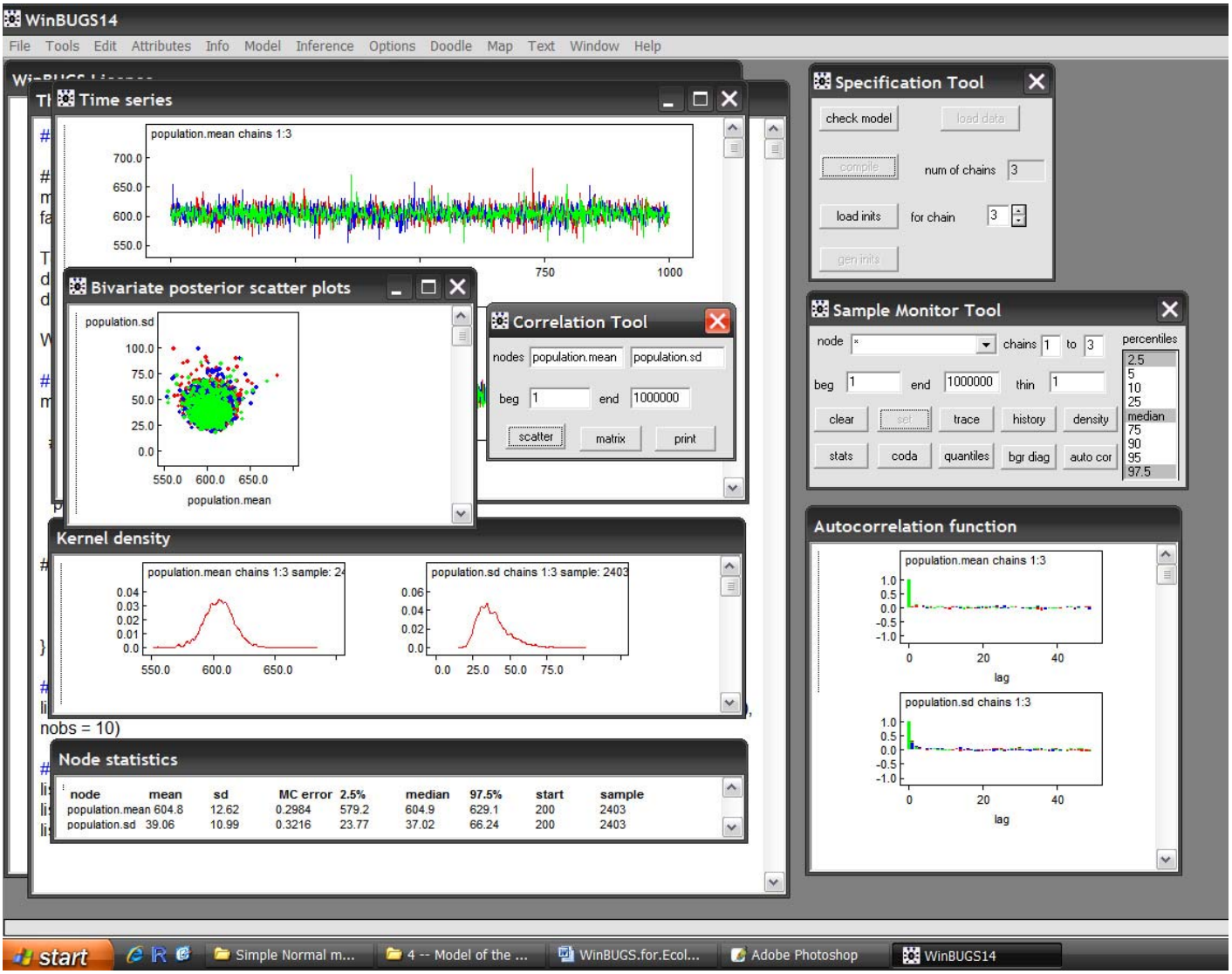
The autocorrelation function is depicted last. For this simple model, the chains are hardly autocorrelated at all. This is good, as our sample contains more information about the parameters than when successive draws are correlated. In more complex models we will frequently find considerable autocorrelation between consecutive draws in a chain, and then it may be useful to thin the chains to get more approximately independent draws, i.e., more "concentrated" information. Note that a very high autocorrelation may also mean that a parameter is not identified in the model, i.e., not estimable, or that there are other structural problems with a model.

You should note two things about Bayesian credible intervals: First, though in practice most non-statisticians will not make a distinction between a Bayesian credible interval and a frequentist confidence interval, and indeed, for large samples sizes and vague priors, the two will typically be very similar numerically, there is in fact a major conceptual difference between them (see Chapter 2). Second, there are different ways in which the posterior distribution of a parameter may be summarized by a Bayesian credible interval. One is the highest posterior density interval (HPDI), which denotes the limits of the narrowest segment of the posterior distribution containing 95% of its total mass (which is 1 by definition). HPDIs are not computed by WinBUGS (nor by R2WinBUGS, see later), but can be obtained using the function `HPDinterval()` in the R package CODA.

One should also keep an eye on the MC error under node statistics. The MC error quantifies the variability in the estimates that is due to Markov chain variability, i.e., the sampling error in the simulation-based solution for Bayes rule for our problem. MC error should be small. According to one rule of thumb it should be <5 % of the posterior standard deviation for a parameter.

One possible summary from our Bayesian analysis of the model of the mean adopted for the body mass of male peregrines would be that mean body mass is estimated at 604.8 g (SE 12.62, 95% CI 579.2–629.1 g) and that the standard deviation of the body mass of male peregrines is 39.06 g (SE 10.99, 95% CI 23.77–66.24). (Note how variance parameters are typically estimated much less precisely than are means.)

There is other summary information from WinBUGS that we can examine; for instance, Inference > Compare or Inference > Correlation allows us to plot parameter estimates or to see how they are related across two or more parameters. Let’s have a look at whether the estimates of the mean body mass and its standard deviation are correlated: Choose Inference > Correlation, type the names of the two parameters (nodes) and we see that the estimates of the two parameters are not correlated at all.



4.5. SUMMARY

We have conducted our first Bayesian analysis of the simplest statistical model, the “model of the mean”. We see that WinBUGS is a very powerful sofware to fit models in a Bayesian mode of inference using MCMC and that a lot can be achieved using simple click and point techniques.

Exercises

1. Save your file under another name, e.g., `test.odc`. Then try out a few things that can go wrong.

- create a typo in the model and WinBUGS complains “made use of undefined node”.
- select an extremely wide range for the priors and see whether WinBUGS gets caught in a trap.
- select initial values outside of the range of a (uniform) prior and WinBUGS will crash.
- add data to the data set (e.g., `silly.dat = c(0,1,2)`) and WinBUGS will say “undefined variable”.
- try out other typos in model, inits or data, e.g., an o (‘oh’) instead of a zero.
- See how WinBUGS deals with missing responses: turn the second element of the mass vector (in the smaller data set, but it doesn’t really matter) into a missing value (NA). On loading the inits, you will now see that WinBUGS says (in its bottom left corner) that for every chain, there are still uninitialized variables. By this, it means the 2nd element of mass, as we shall see. Continue and request a sample for both population.mean and mass in the Sample Monitor Tool. Request the Trace to be displayed and you will see that we get one for the node population.mean and another one for mass[2]. We see that this missing value is automatically imputed (estimated). The imputed value in this example is the same as the population mean, except that its uncertainty is much larger.