An Introduction to Using WinBUGS for Cost-Effectiveness Analyses in Health Economics

Dr. Christian Asseburg

Centre for Health Economics
University of York, UK

`ca505@york.ac.uk`

Practical 1

Getting started in OpenBUGS / WinBUGS

Practical 1: Getting started in OpenBUGS

- Brief comparison WinBUGS / OpenBUGS
- Practical
  - Opening OpenBUGS
  - Entering a model and data
  - Some error messages
  - Starting the sampler
  - Checking sampling performance
  - Retrieving the posterior summaries

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

WinBUGS / OpenBUGS

- **WinBUGS** was developed at the MRC Biostatistics unit in Cambridge. Free download, but registration required for a licence. No fee and no warranty.
- **OpenBUGS** is the current development of WinBUGS after its source code was released to the public. Download is free, no registration, GNU GPL licence.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

WinBUGS / OpenBUGS

- There are **no major differences** between the latest WinBUGS (1.4.1) and OpenBUGS (2.2.0) releases.
- Minor differences include:
  - OpenBUGS is occasionally a bit slower
  - WinBUGS requires Microsoft Windows OS
  - OpenBUGS error messages are sometimes more informative
- The examples in these slides use OpenBUGS.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

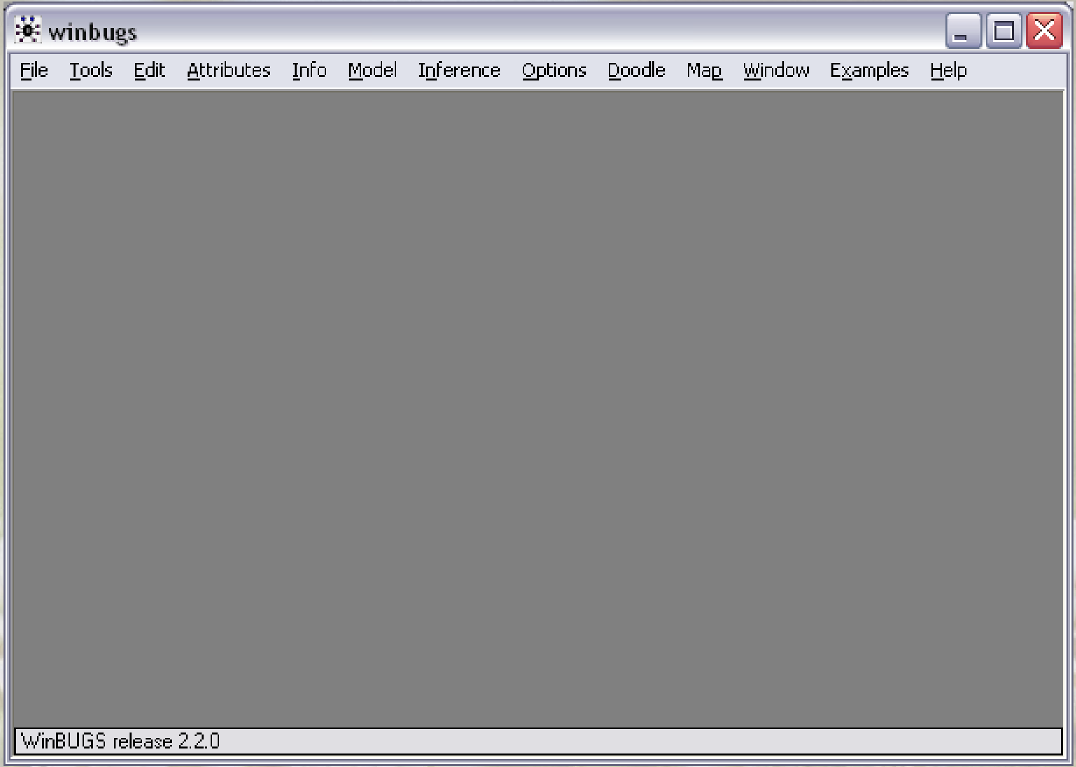
Practical 1: Target

- Start OpenBUGS
- Code the example from health economics from the earlier presentation
- Run the sampler and obtain posterior summaries

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

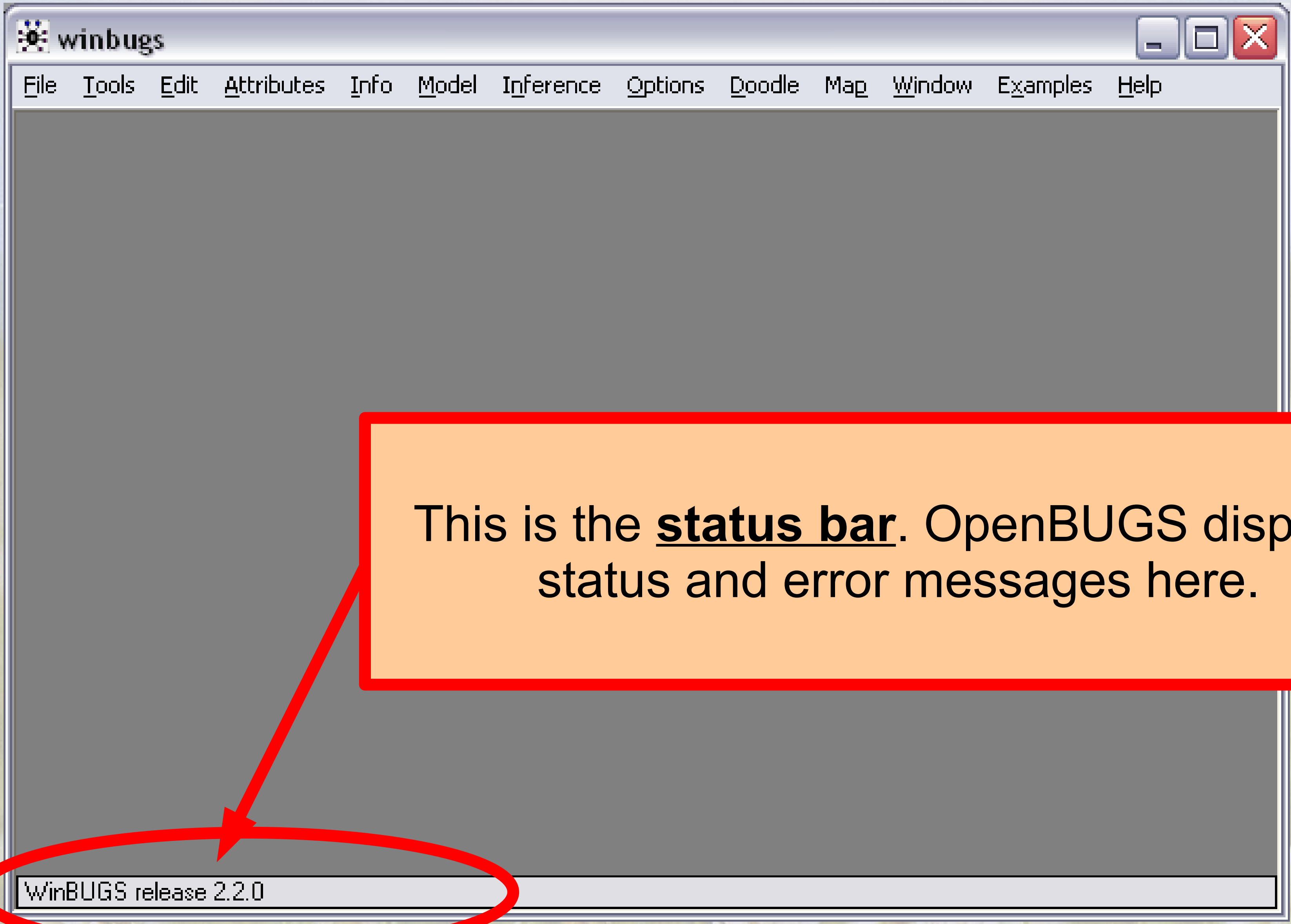
Starting OpenBUGS

- You can download OpenBUGS from **http://mathstat.helsinki.fi/openbugs/**
- Start the program.


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Window Examples Help
WinBUGS release 2.2.0

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS

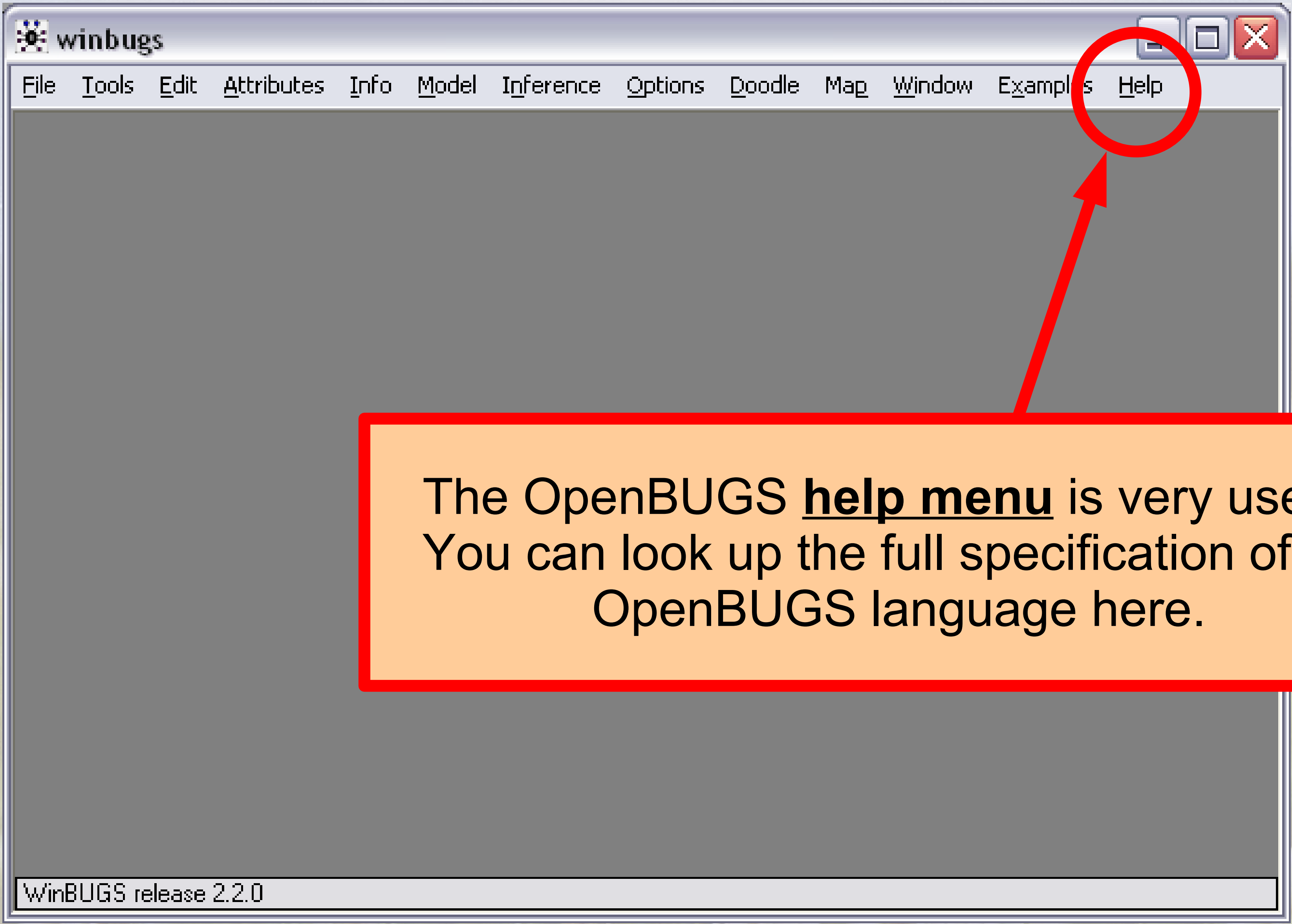


This is the **status bar**. OpenBUGS displays status and error messages here.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS

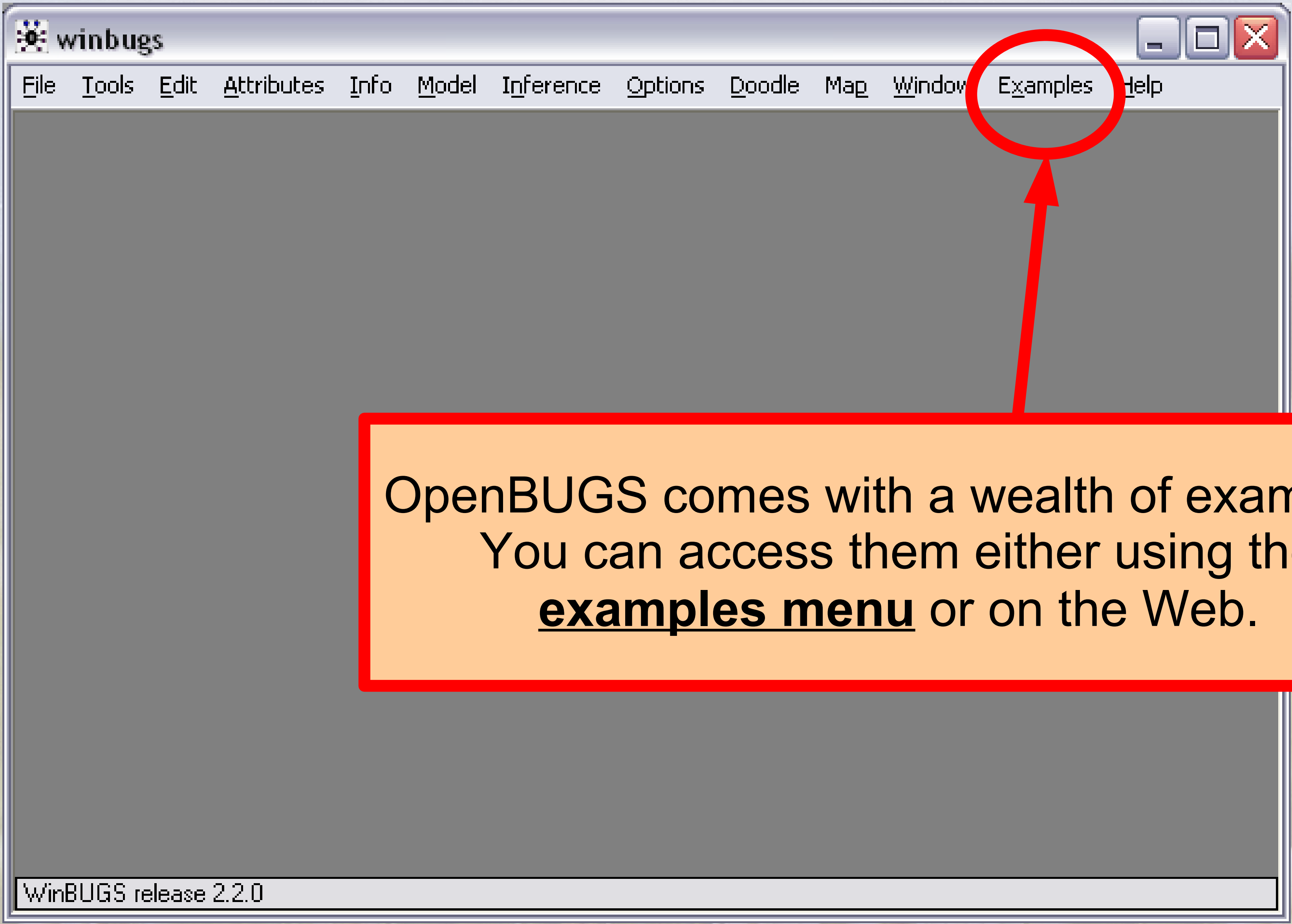

winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Window Examples Help
WinBUGS release 2.2.0

The OpenBUGS **help menu** is very useful. You can look up the full specification of the OpenBUGS language here.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS

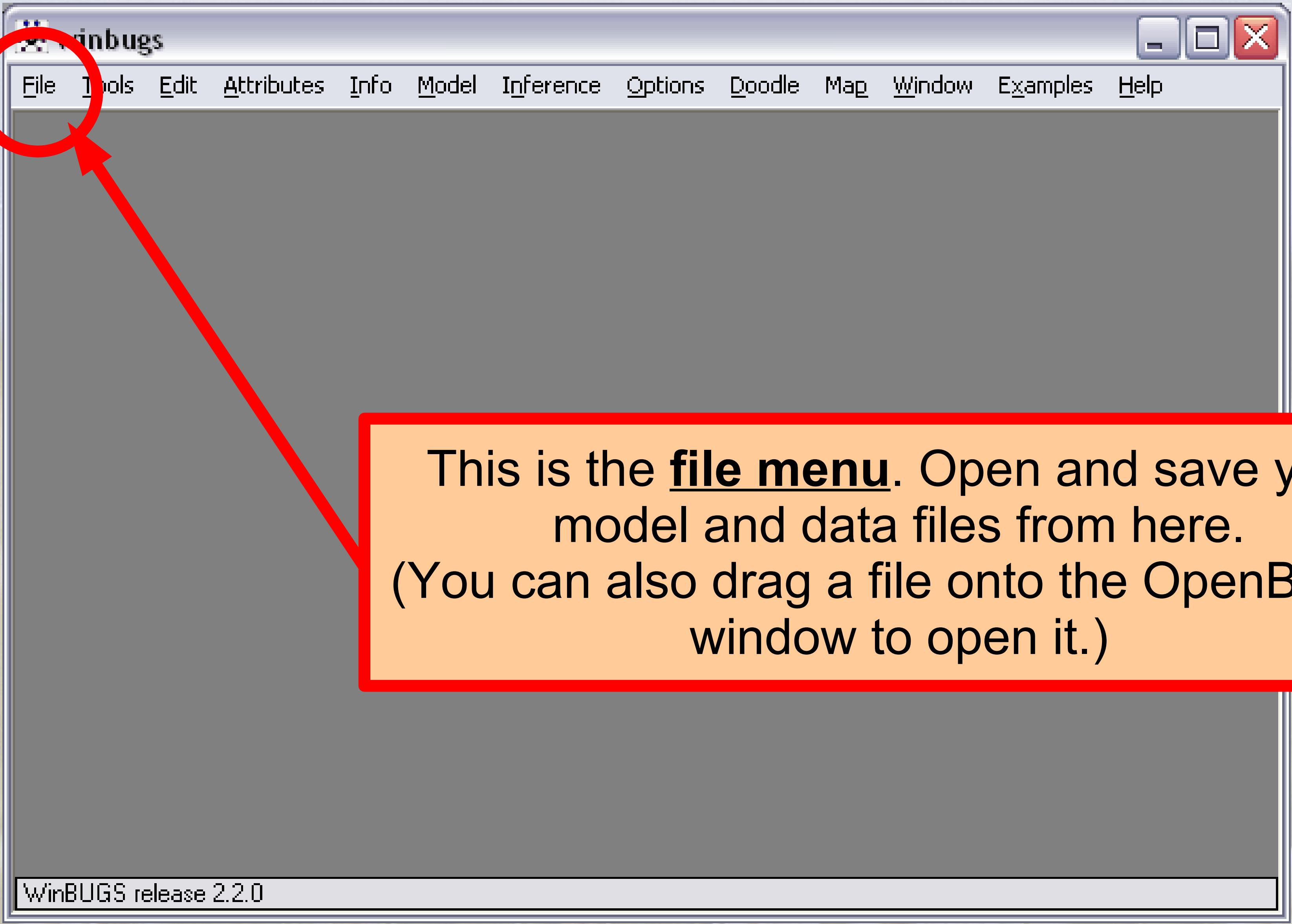

winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Window Examples Help
WinBUGS release 2.2.0

OpenBUGS comes with a wealth of examples. You can access them either using the **examples menu** or on the Web.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Centre For Health Economics

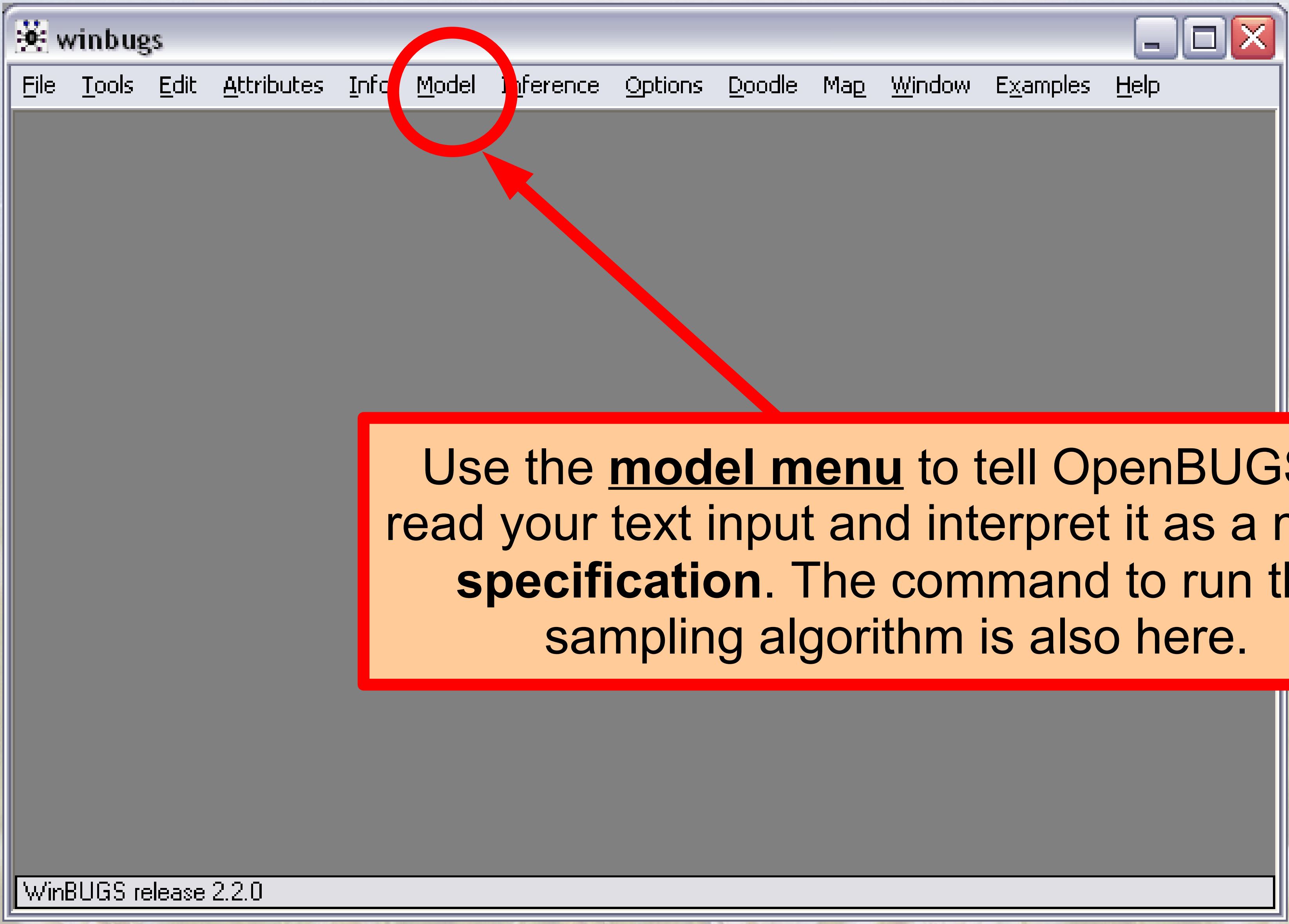
Practical 1: Getting started in OpenBUGS



This is the **file menu**. Open and save your model and data files from here.
(You can also drag a file onto the OpenBUGS window to open it.)

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

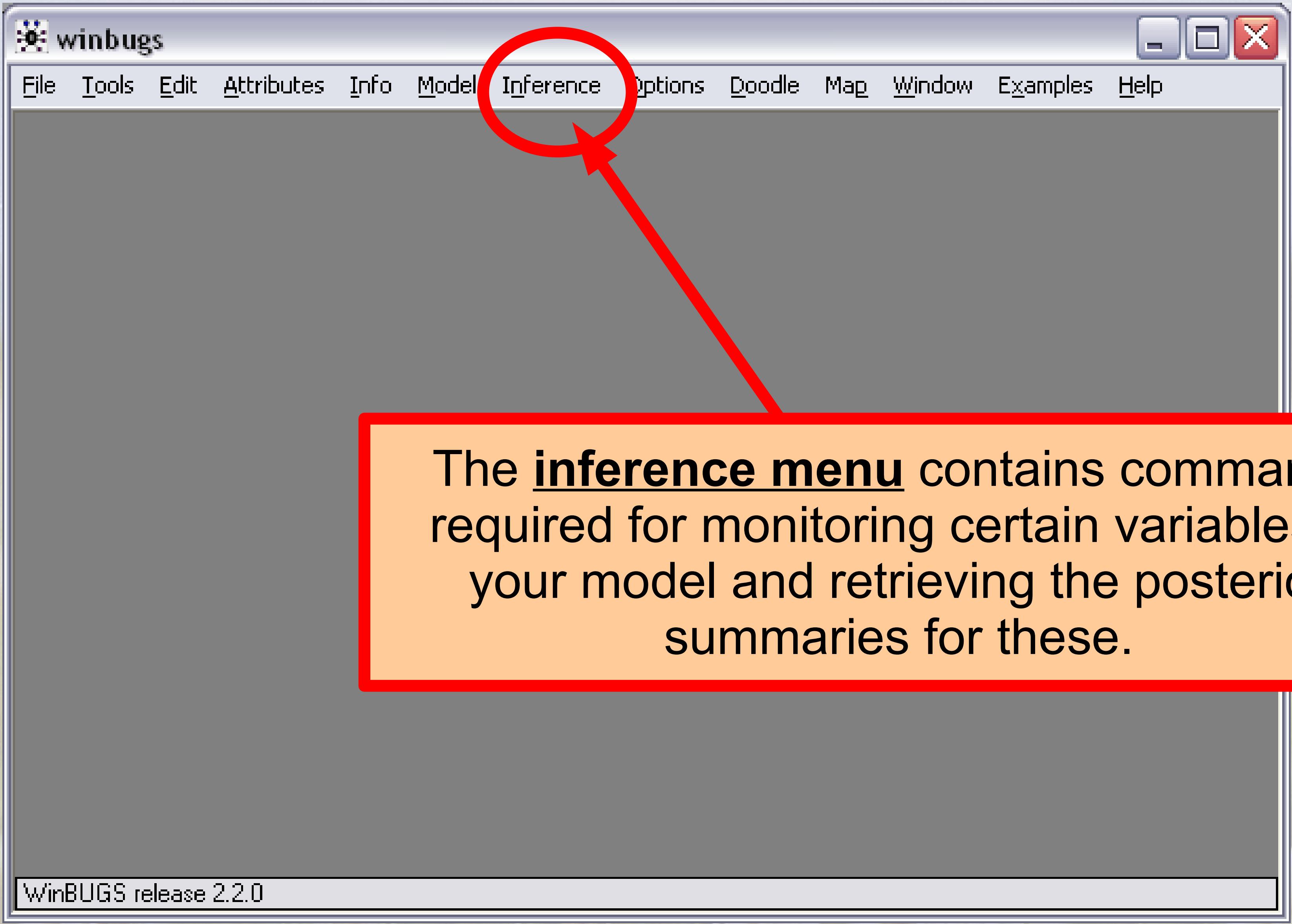
Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Window Examples Help
WinBUGS release 2.2.0

Use the **model menu** to tell OpenBUGS to read your text input and interpret it as a model **specification**. The command to run the sampling algorithm is also here.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

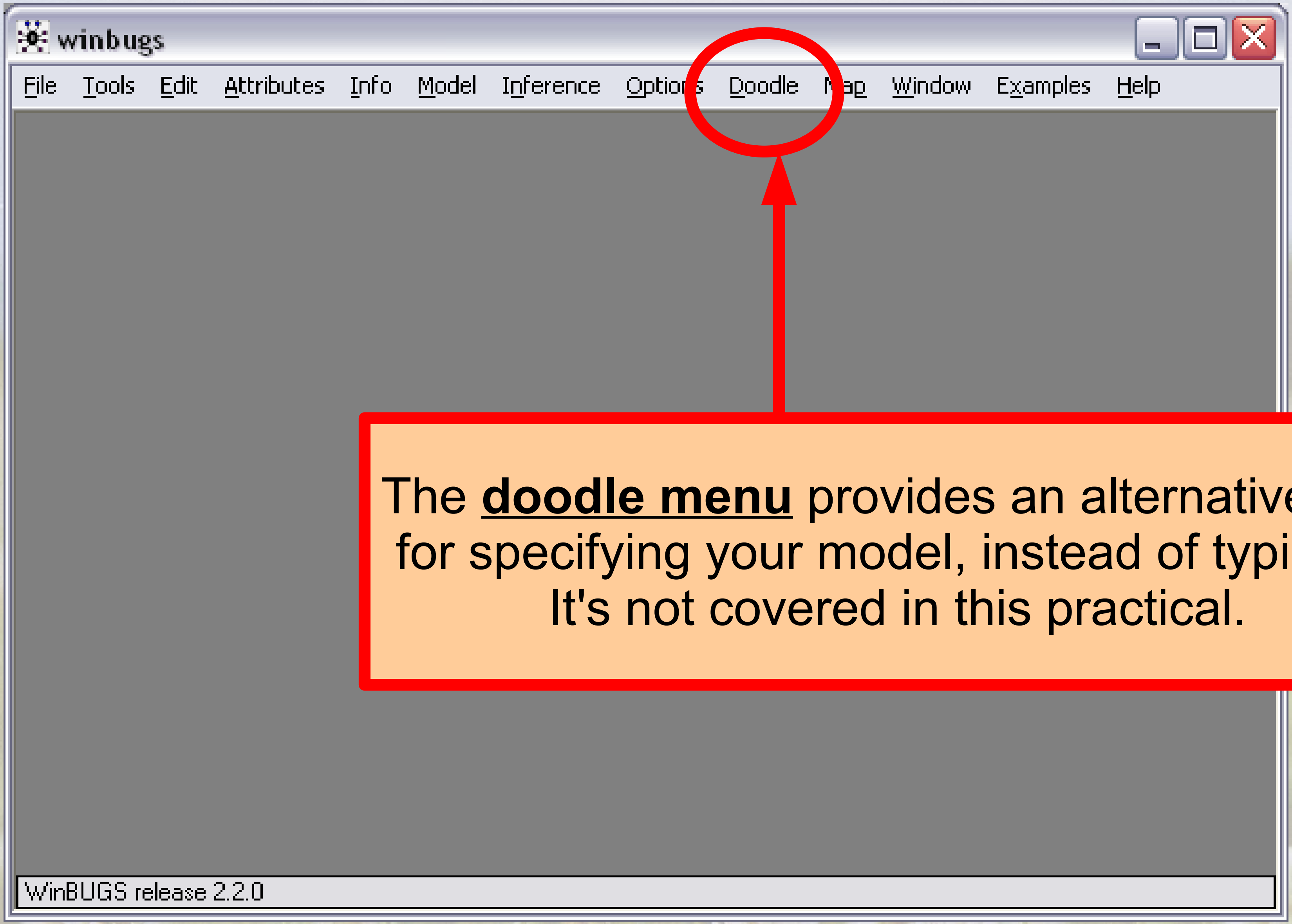
Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Window Examples Help
WinBUGS release 2.2.0

The **inference menu** contains commands required for monitoring certain variables in your model and retrieving the posterior summaries for these.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

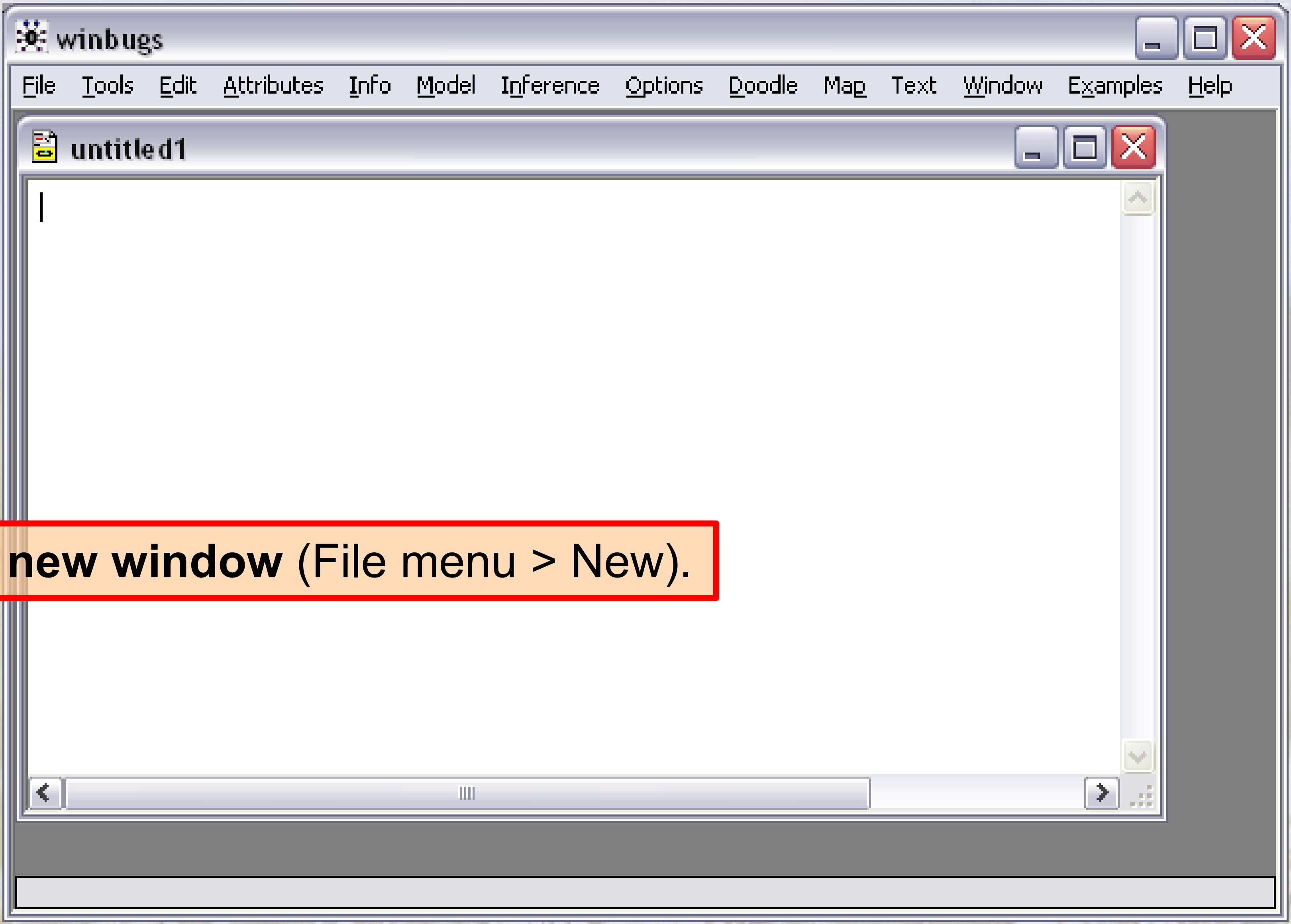
Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Window Examples Help
WinBUGS release 2.2.0

The **doodle menu** provides an alternative way for specifying your model, instead of typing it. It's not covered in this practical.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS

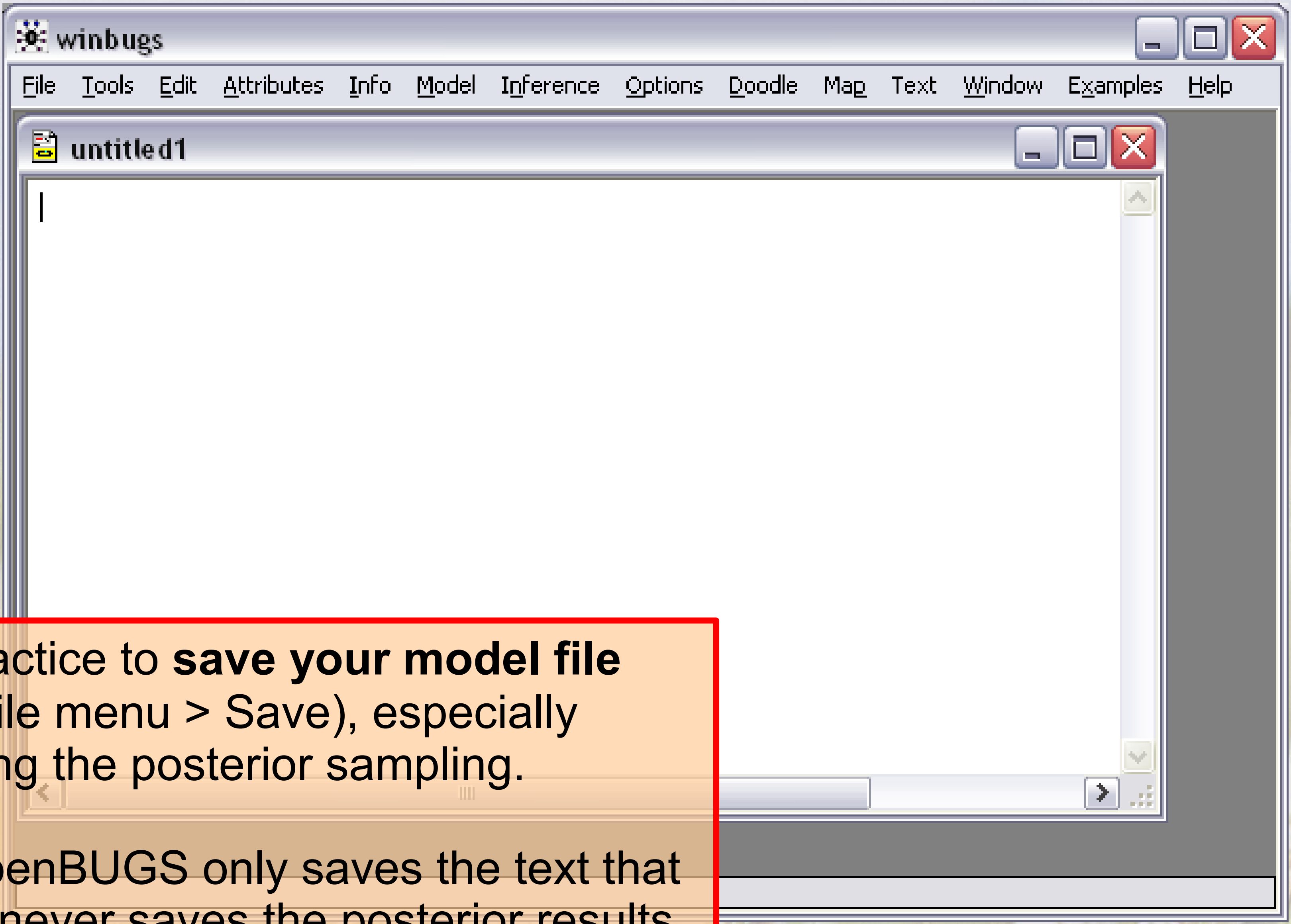

winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
untitled1

Let's open a **new window** (File menu > New).

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
untitled1

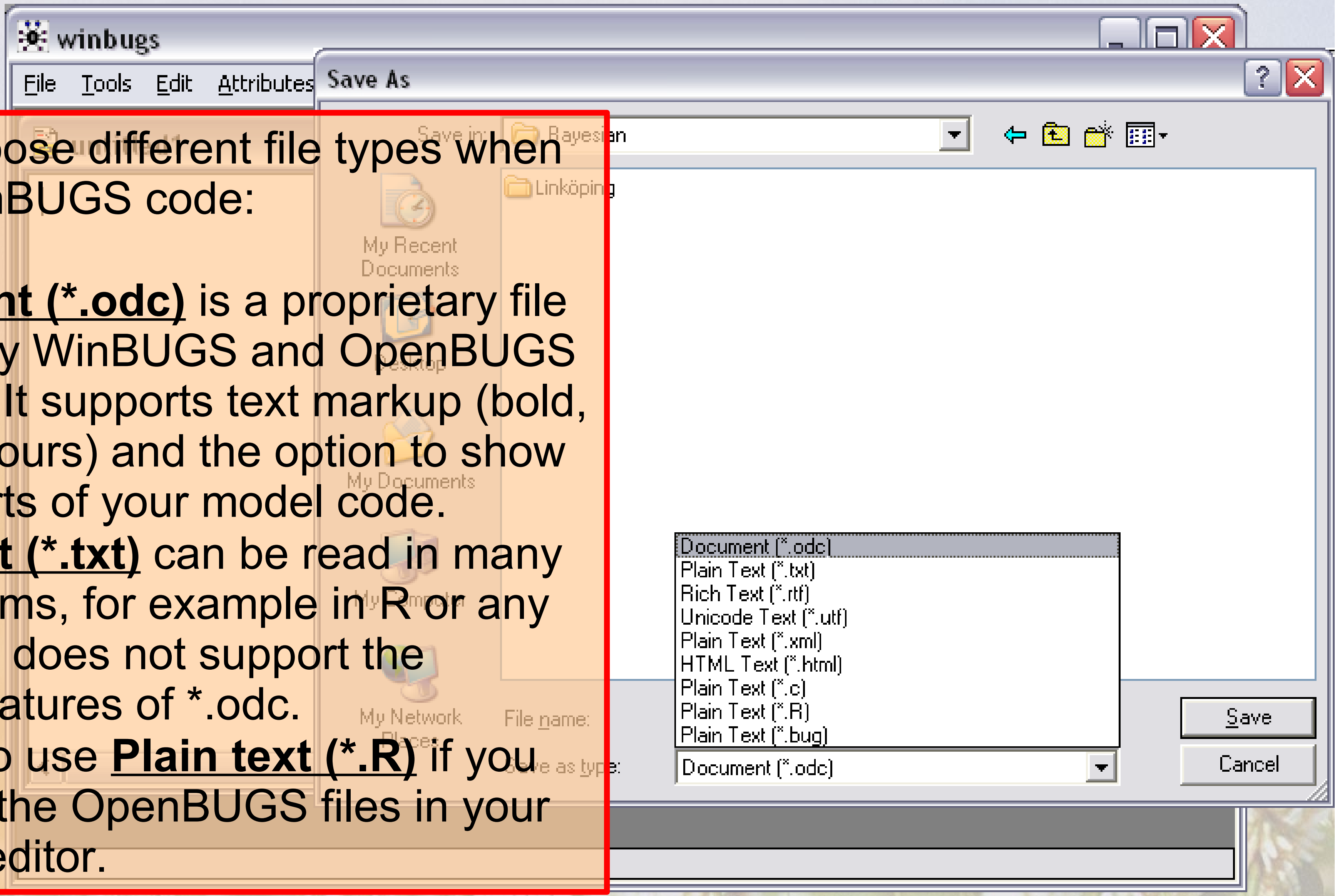
It is good practice to **save your model file regularly** (File menu > Save), especially before starting the posterior sampling.

Note that OpenBUGS only saves the text that you enter, it never saves the posterior results.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



Document (*.odc)
Plain Text (*.txt)
Rich Text (*.rtf)
Unicode Text (*.utf)
Plain Text (*.xml)
HTML Text (*.html)
Plain Text (*.c)
Plain Text (*.R)
Plain Text (*.bug)

Save as type: Document (*.odc)

You can choose different file types when saving OpenBUGS code:

➔ **Document (*.odc)** is a proprietary file type that only WinBUGS and OpenBUGS understand. It supports text markup (bold, fontface, colours) and the option to show and hide parts of your model code.

➔ **Plain text (*.txt)** can be read in many other programs, for example in R or any text editor. It does not support the advanced features of *.odc.

You can also use **Plain text (*.R)** if you want to edit the OpenBUGS files in your favourite R editor.

Dr Christian Asseburg
University of York, UK   ca505@york.ac.uk
CHE Centre For Health Economics

Syntax of the OpenBUGS language

Writing OpenBUGS code is similar to writing R code.

`<-`	Define the value of the LHS by a calculation
`~`	Define the value of the LHS by a probability distribution
`x[5]`	Refer to a vector (or array) element

```
for (i in 1:20) {
             ...
             }
```

Write a "for" loop like this.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Syntax of the OpenBUGS language

Note on the use of brackets:

( ) round brackets

- group mathematical terms in an equation – e.g. **`x*(a+b)`**
- submit arguments to a function – e.g. **`exp(log(q))`**
- are used in special cases like the “for” loop.

[ ] square brackets index a vector or an array.

{ } curly brackets group statements into blocks.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Syntax of the OpenBUGS language

Important differences to R syntax:

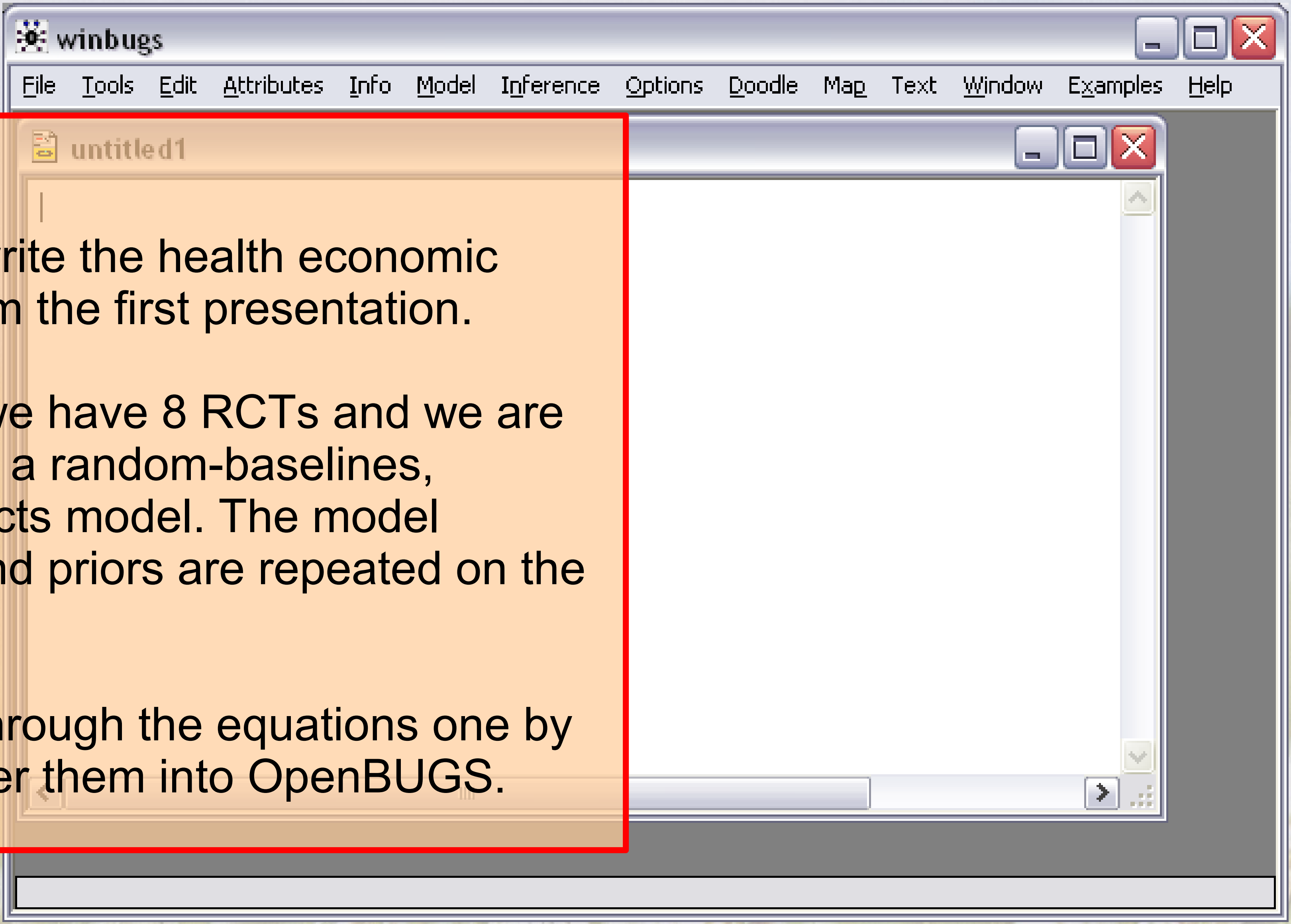
In OpenBUGS the **order of the statements is irrelevant**. Code is not executed from top to bottom.

In OpenBUGS you do **not need to define an array or vector** before you use it. It suffices to indicate on the LHS of an equation that you are assigning to a particular element of an array, e.g. **`q[5,3] <- a*b`**

In OpenBUGS if you want to **refer to an entire array**, you must put empty square brackets behind the name, e.g. **`q[,]`**

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
untitled1

Ready?

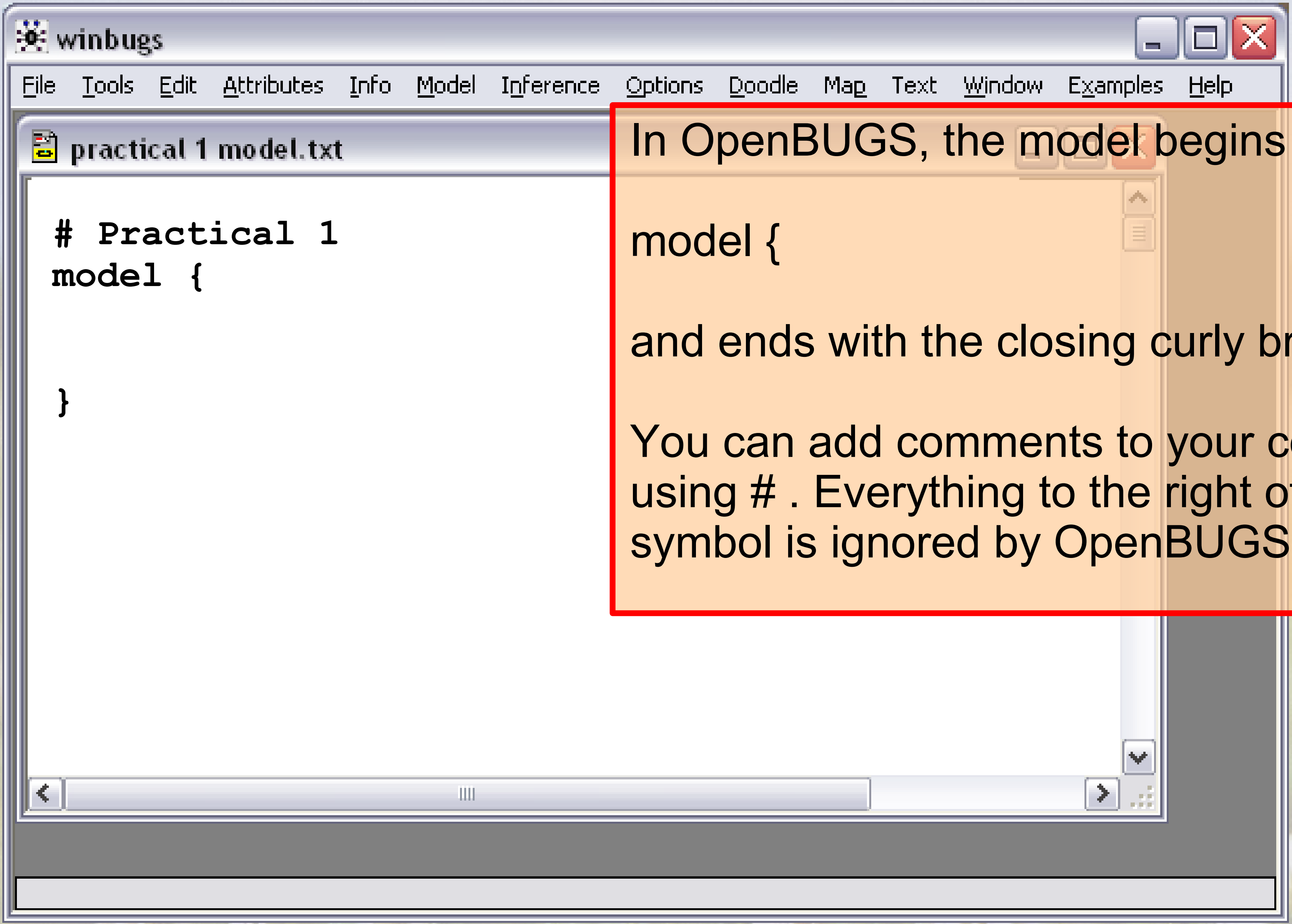
Let's try to write the health economic example from the first presentation.

Recall that we have 8 RCTs and we are working with a random-baselines, random-effects model. The model equations and priors are repeated on the next slides.

We will go through the equations one by one and enter them into OpenBUGS.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
# Practical 1
model {


}
```

In OpenBUGS, the model begins with

model {

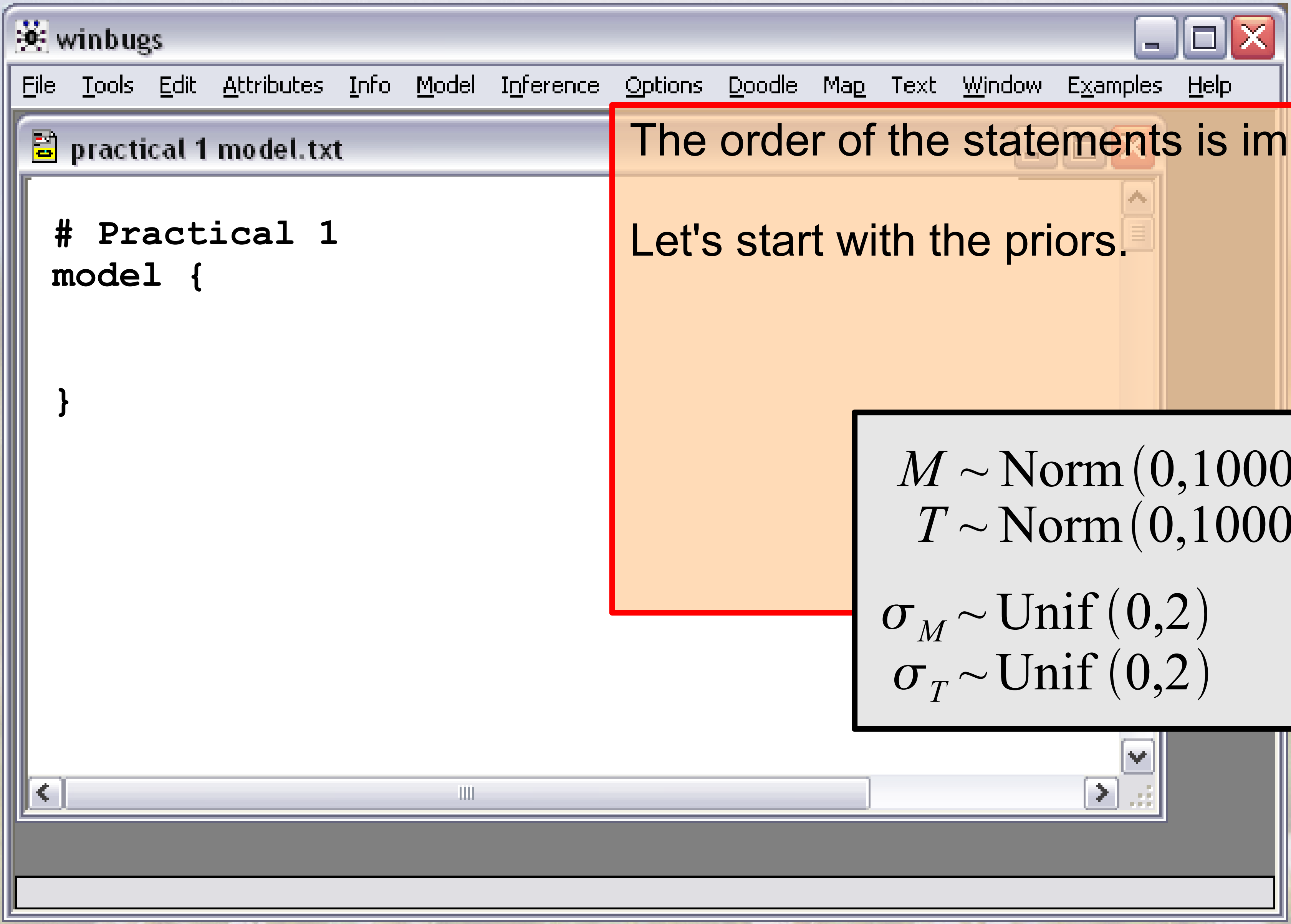
and ends with the closing curly bracket } .

You can add comments to your code using # . Everything to the right of a # symbol is ignored by OpenBUGS.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
# Practical 1
model {


}
```

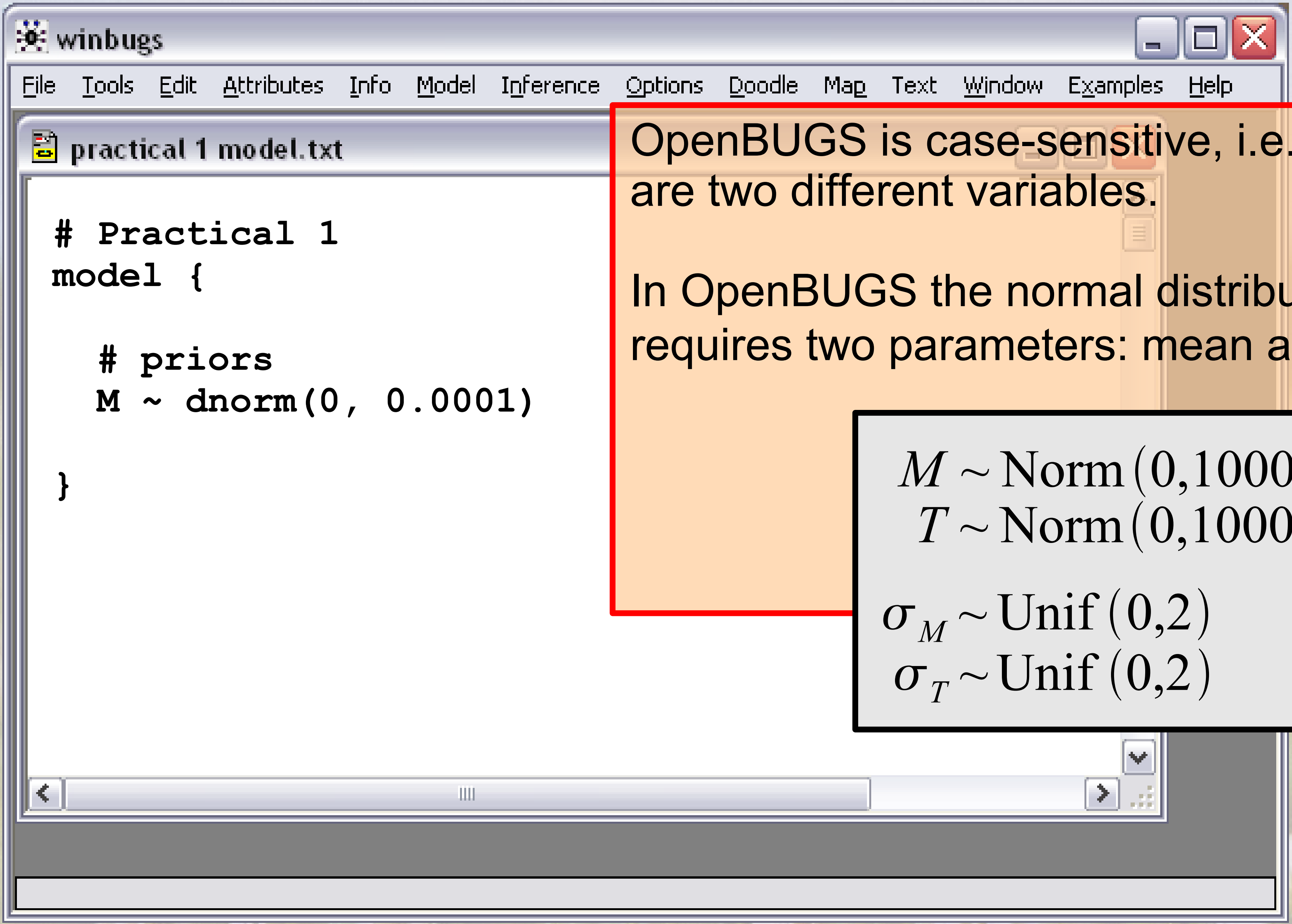
The order of the statements is immaterial.

Let's start with the priors.

$$M \sim \text{Norm}(0,10000)$$
$$T \sim \text{Norm}(0,10000)$$
$$\sigma_M \sim \text{Unif}(0,2)$$
$$\sigma_T \sim \text{Unif}(0,2)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
# Practical 1
model {

   # priors
   M ~ dnorm(0, 0.0001)

}
```

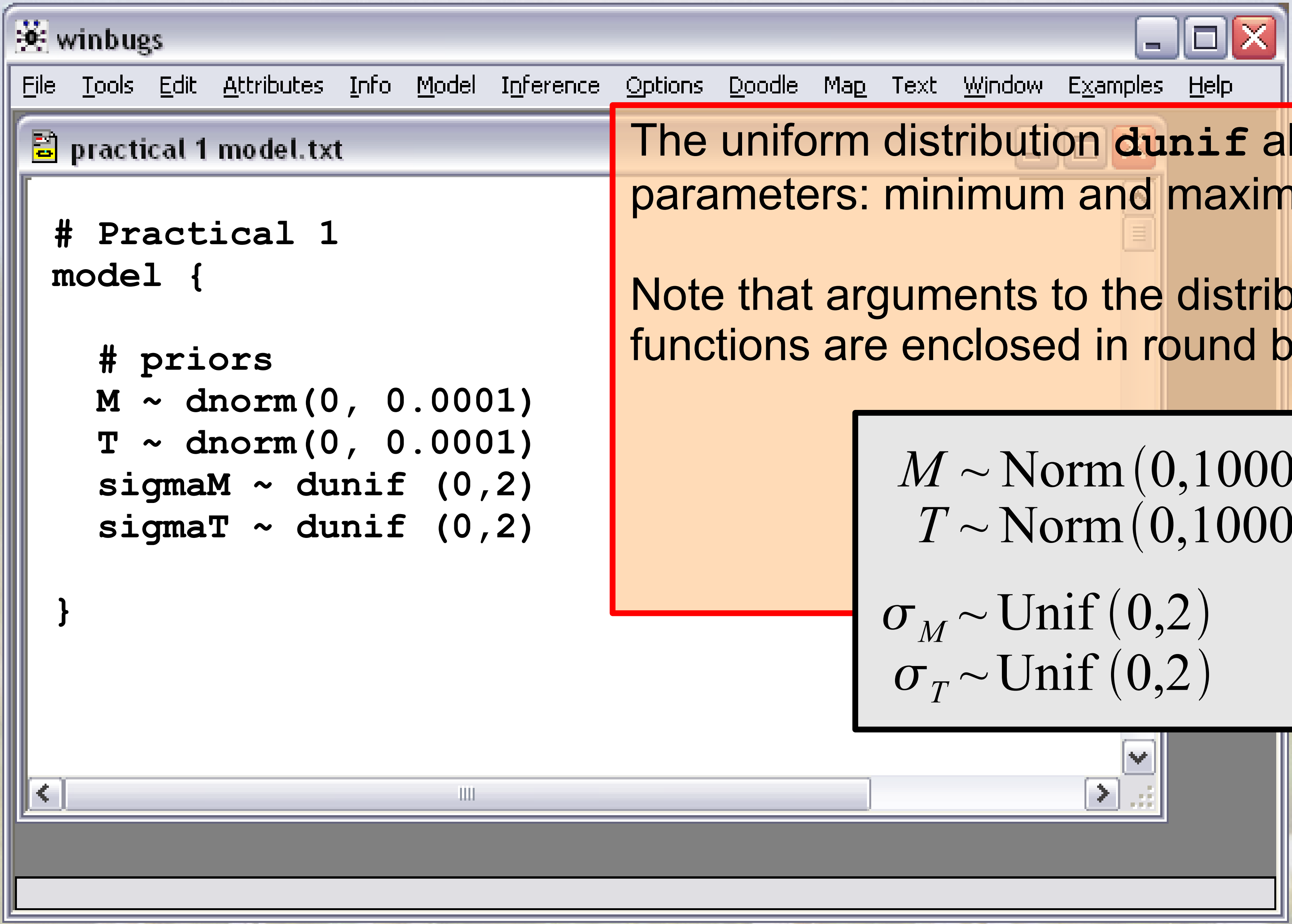
OpenBUGS is case-sensitive, i.e. *M* and *m* are two different variables.

In OpenBUGS the normal distribution **`dnorm`** requires two parameters: mean and <u>precision</u>.

$$M \sim \text{Norm}(0,10000)$$
$$T \sim \text{Norm}(0,10000)$$
$$\sigma_M \sim \text{Unif}(0,2)$$
$$\sigma_T \sim \text{Unif}(0,2)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
# Practical 1
model {

  # priors
  M ~ dnorm(0, 0.0001)
  T ~ dnorm(0, 0.0001)
  sigmaM ~ dunif (0,2)
  sigmaT ~ dunif (0,2)

}
```

The uniform distribution **`dunif`** also takes two parameters: minimum and maximum.

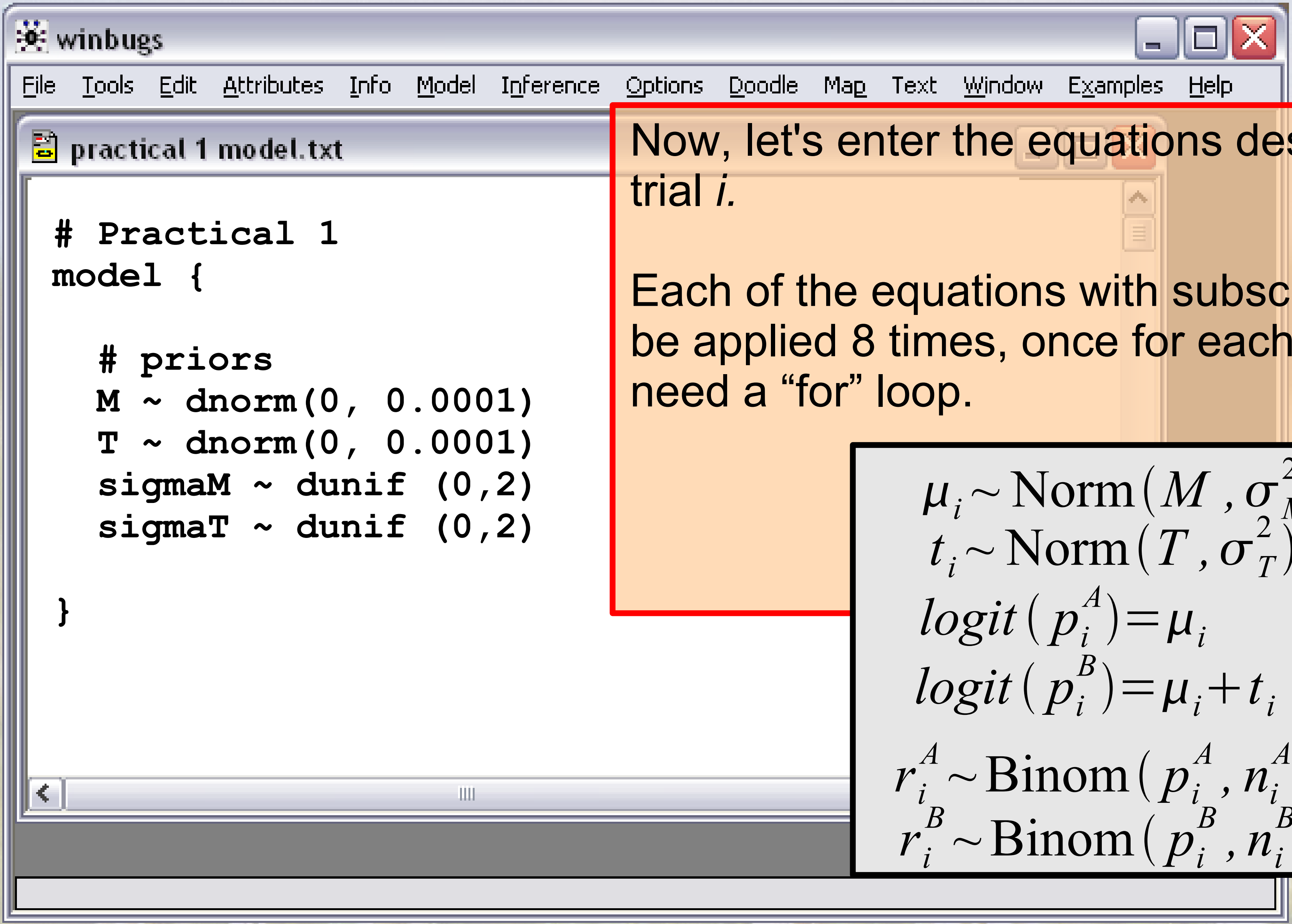
Note that arguments to the distribution functions are enclosed in round brackets ( ) .

$$M \sim \mathrm{Norm}(0,10000)$$
$$T \sim \mathrm{Norm}(0,10000)$$
$$\sigma_M \sim \mathrm{Unif}(0,2)$$
$$\sigma_T \sim \mathrm{Unif}(0,2)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
# Practical 1
model {

  # priors
  M ~ dnorm(0, 0.0001)
  T ~ dnorm(0, 0.0001)
  sigmaM ~ dunif (0,2)
  sigmaT ~ dunif (0,2)

}
```

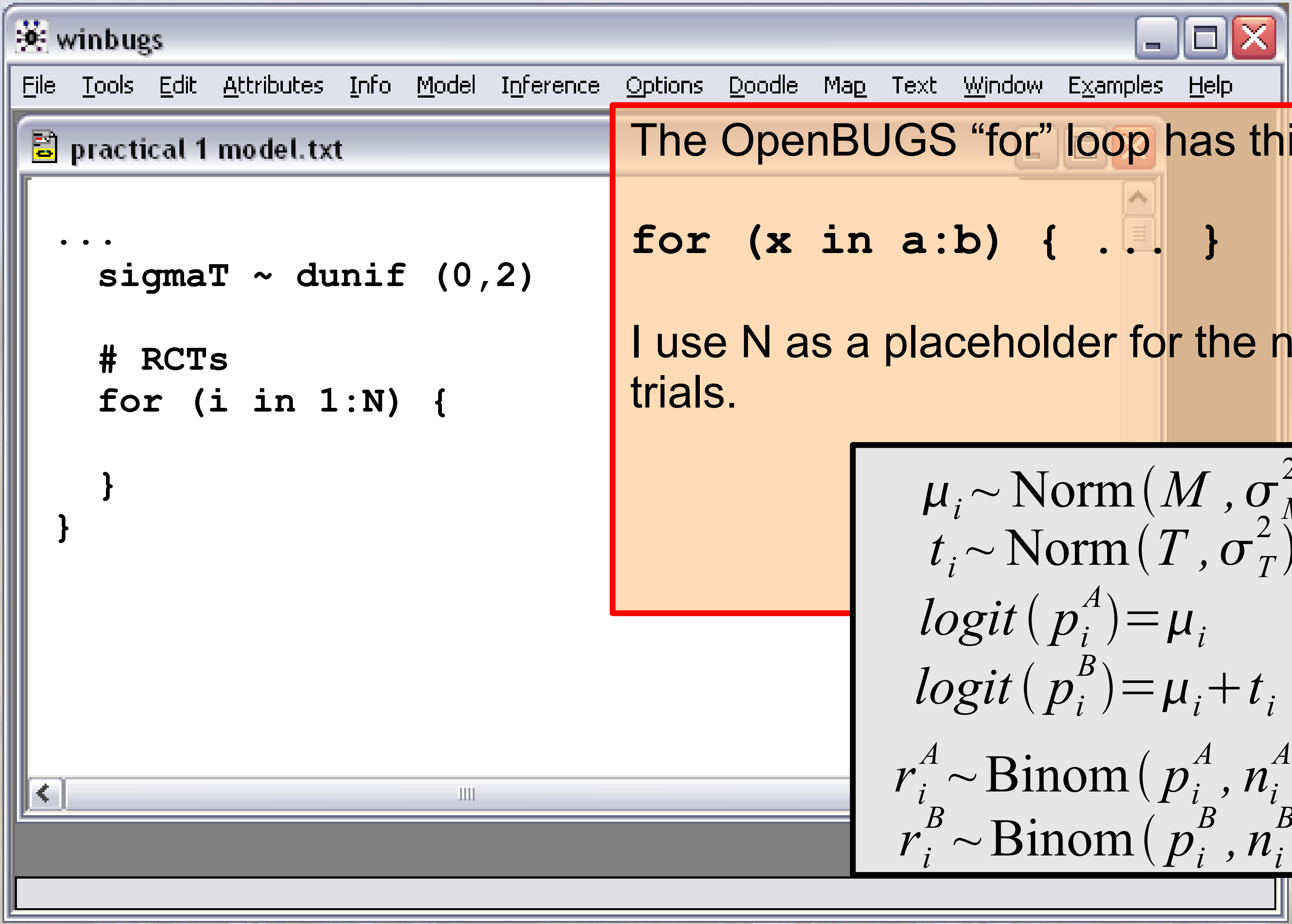
Now, let's enter the equations describing each trial *i.*

Each of the equations with subscript *i* has to be applied 8 times, once for each trial. We need a “for” loop.

$$\mu_i \sim \text{Norm}(M, \sigma_M^2)$$
$$t_i \sim \text{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \text{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \text{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
...
  sigmaT ~ dunif (0,2)

  # RCTs
  for (i in 1:N) {

  }
}
```

The OpenBUGS “for” loop has this syntax:

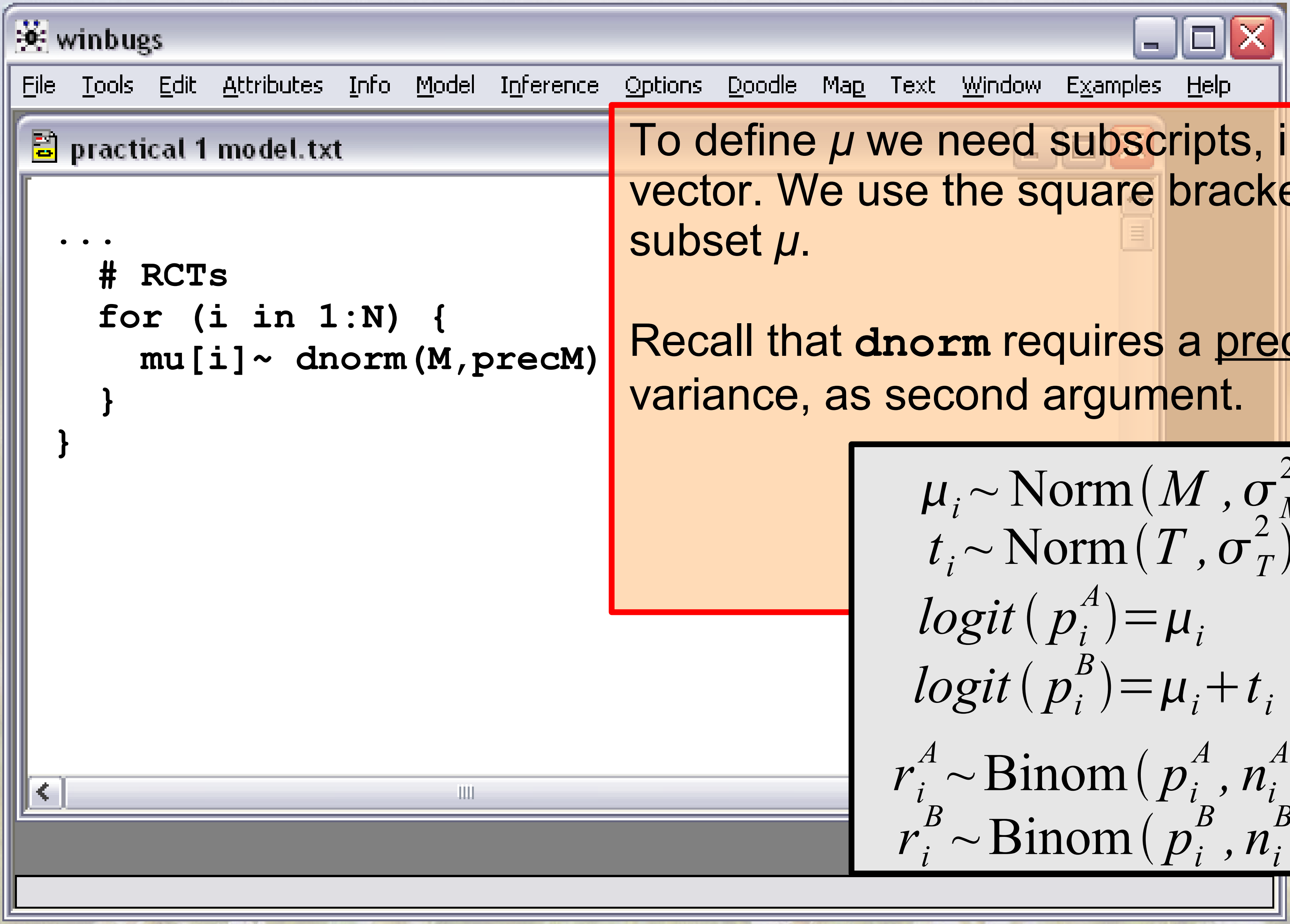
```
for (x in a:b) { ... }
```

I use N as a placeholder for the number of trials.

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
  }
}
```

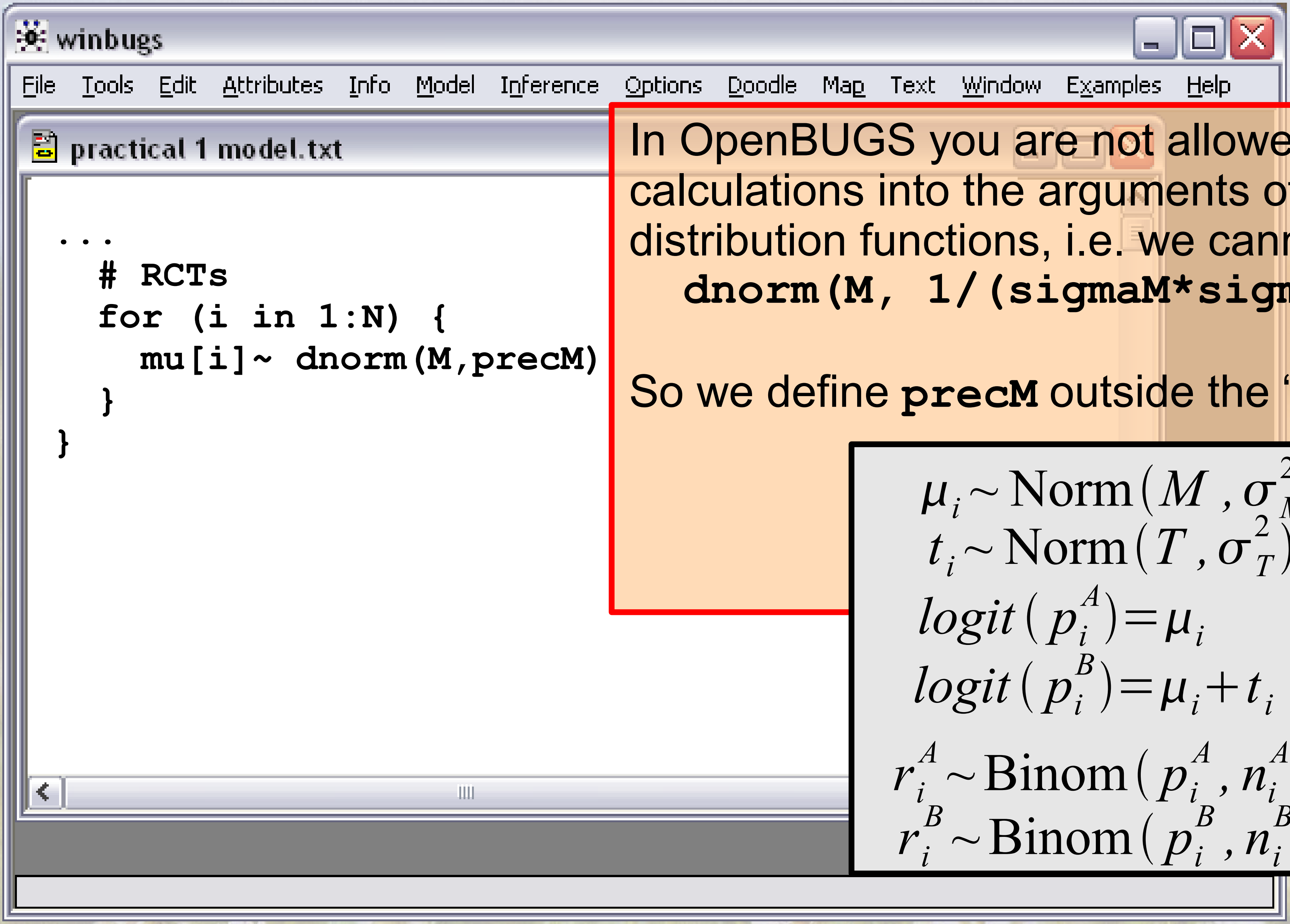
To define *μ* we need subscripts, i.e. *μ* is a vector. We use the square brackets [ ] to subset *μ*.

Recall that **`dnorm`** requires a precision, not a variance, as second argument.

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
  }
}
```

In OpenBUGS you are not allowed to write calculations into the arguments of the distribution functions, i.e. we cannot write

`dnorm(M, 1/(sigmaM*sigmaM) )`

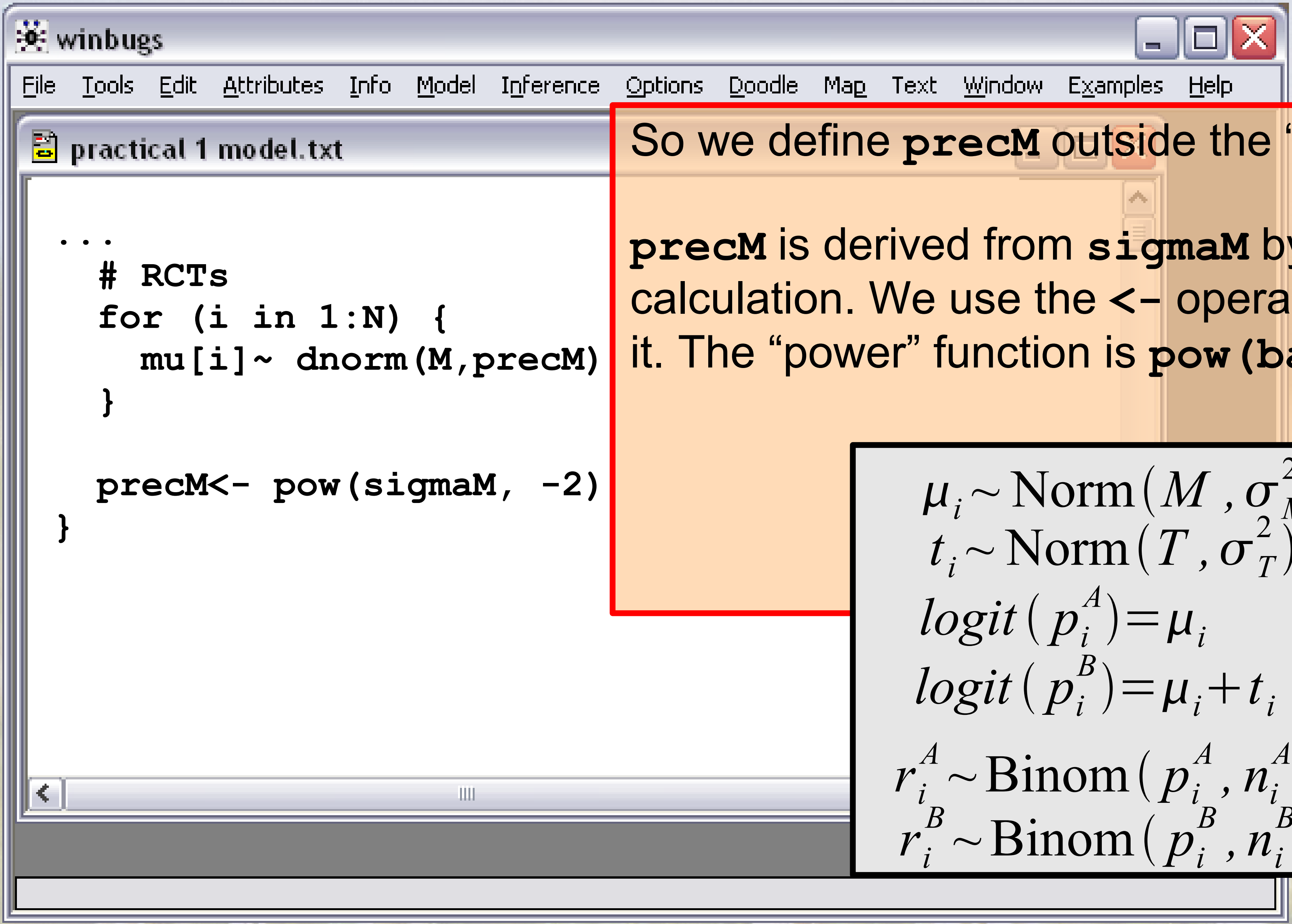
So we define **`precM`** outside the "for" loop.

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
  }

  precM<- pow(sigmaM, -2)
}
```

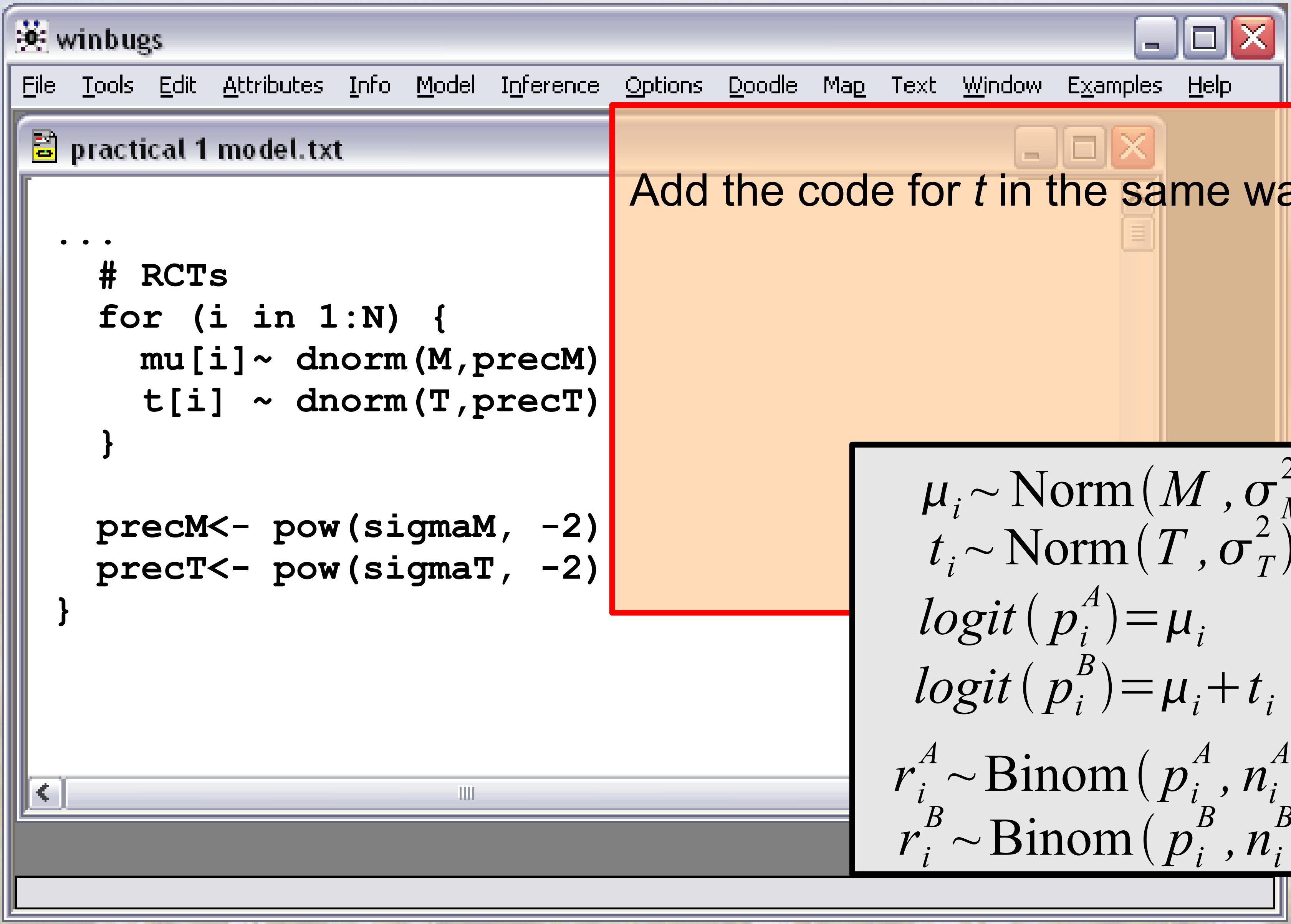
So we define **`precM`** outside the "for" loop.

`precM` is derived from **`sigmaM`** by simple calculation. We use the **`<-`** operator to define it. The "power" function is **`pow(base,exp)`**.

$$\mu_i \sim \text{Norm}(M, \sigma_M^2)$$
$$t_i \sim \text{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \text{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \text{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

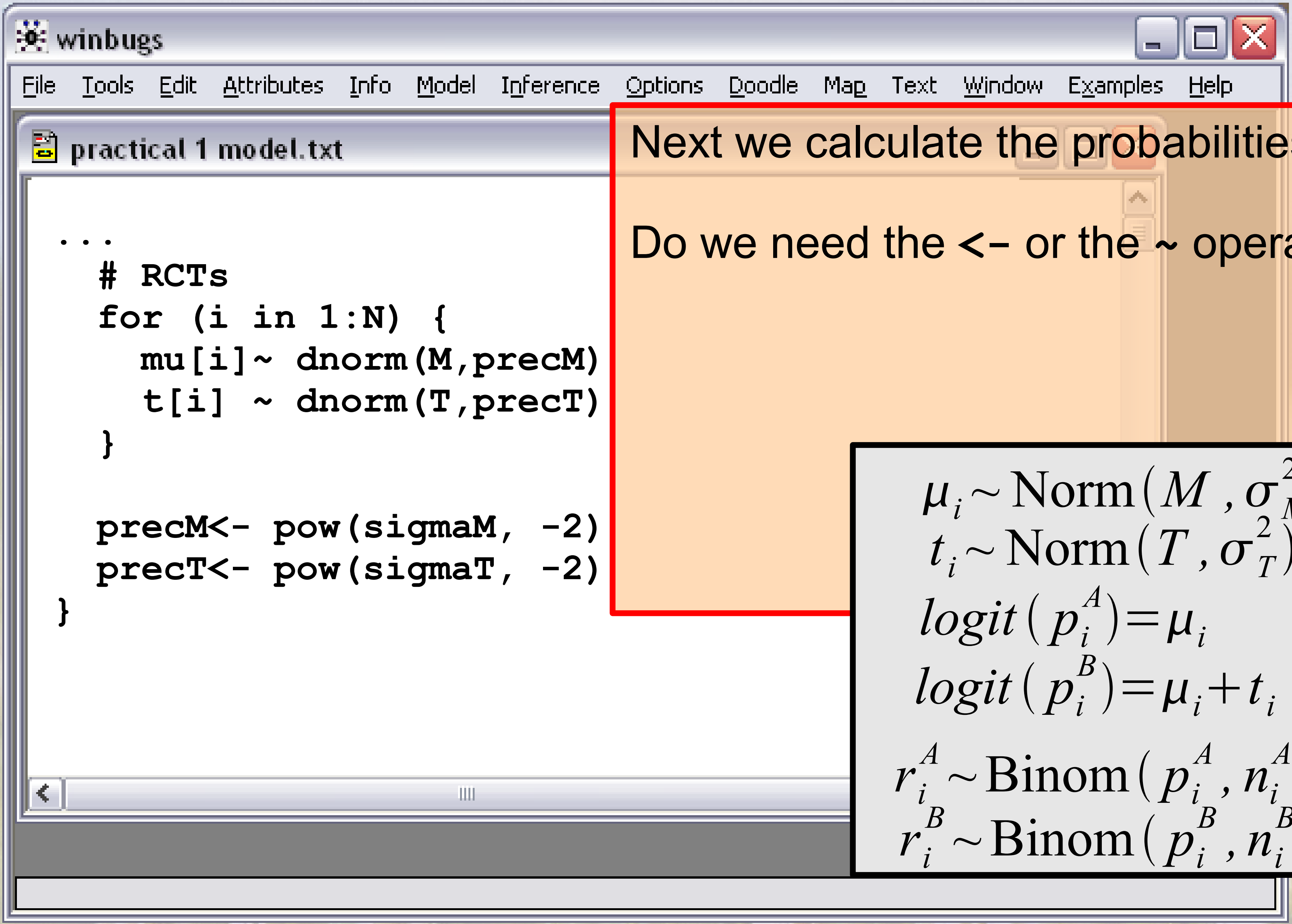
```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
  }

  precM<- pow(sigmaM, -2)
  precT<- pow(sigmaT, -2)
}
```

Add the code for *t* in the same way.

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
  }

  precM<- pow(sigmaM, -2)
  precT<- pow(sigmaT, -2)
}
```

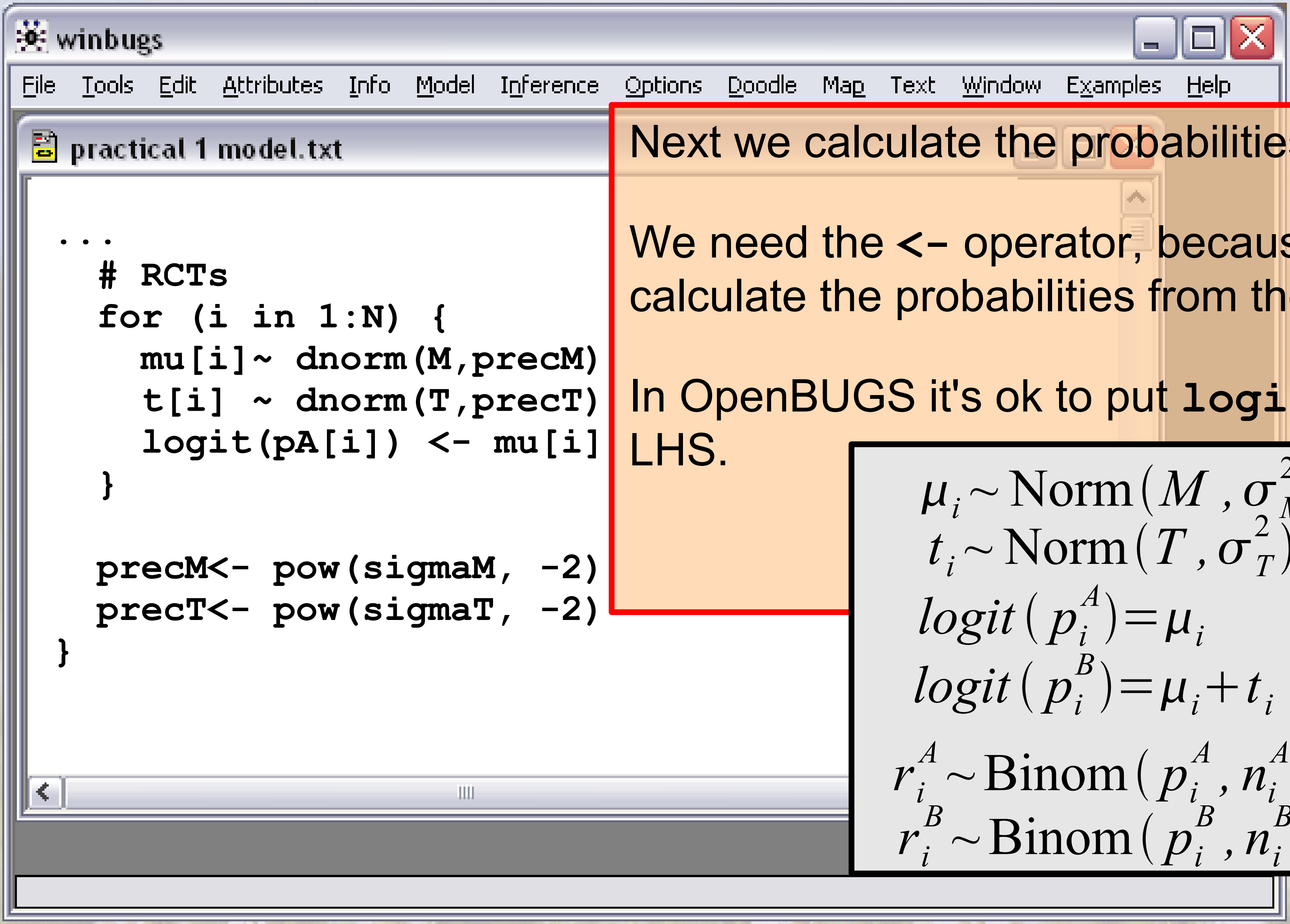
Next we calculate the probabilities p^A and p^B.

Do we need the **<-** or the **~** operator?

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
    logit(pA[i]) <- mu[i]
  }

  precM<- pow(sigmaM, -2)
  precT<- pow(sigmaT, -2)
}
```

Next we calculate the probabilities p^A and p^B.

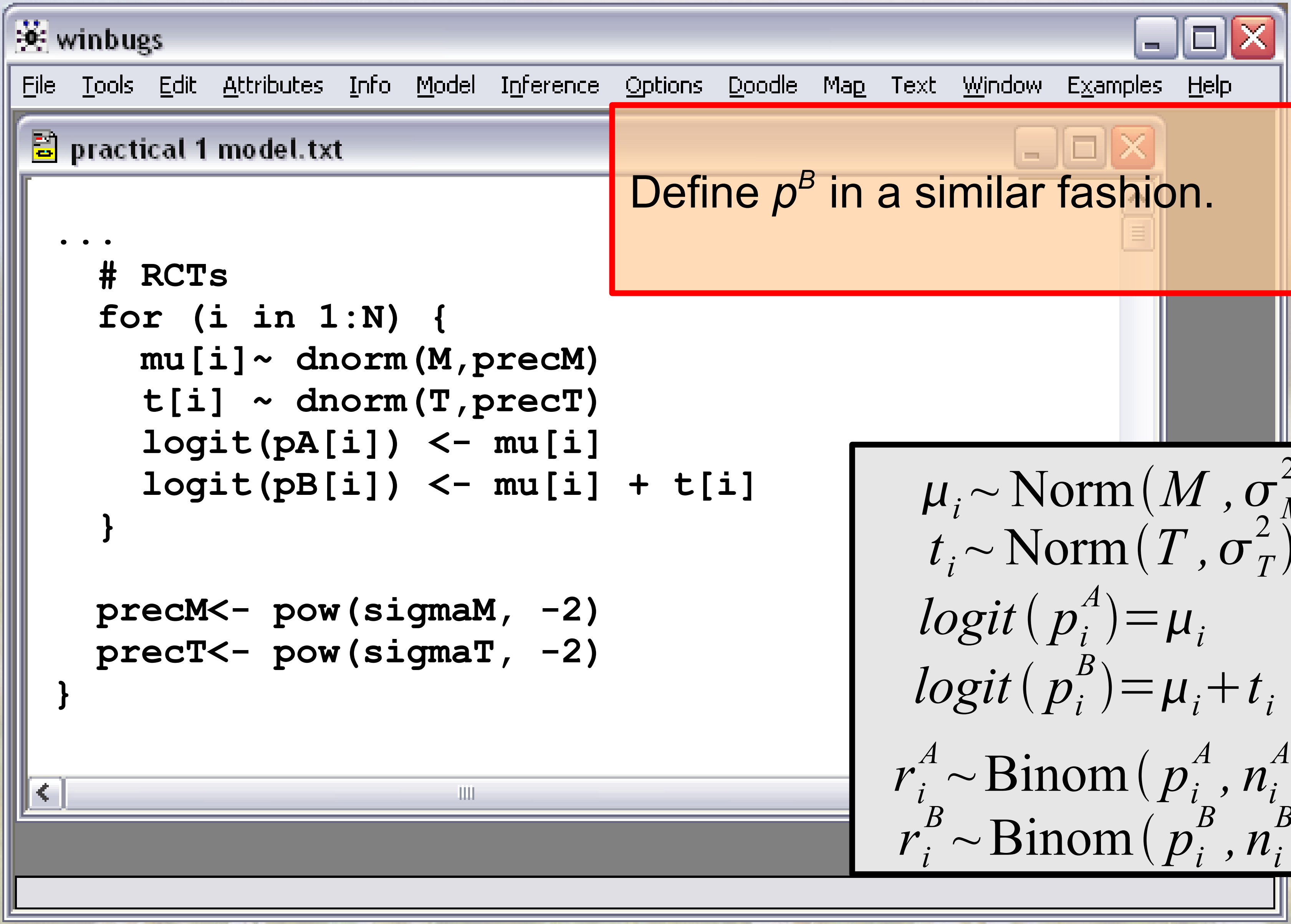
We need the **<-** operator, because we can calculate the probabilities from the *μ* and *t*.

In OpenBUGS it's ok to put **logit** on the LHS.

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

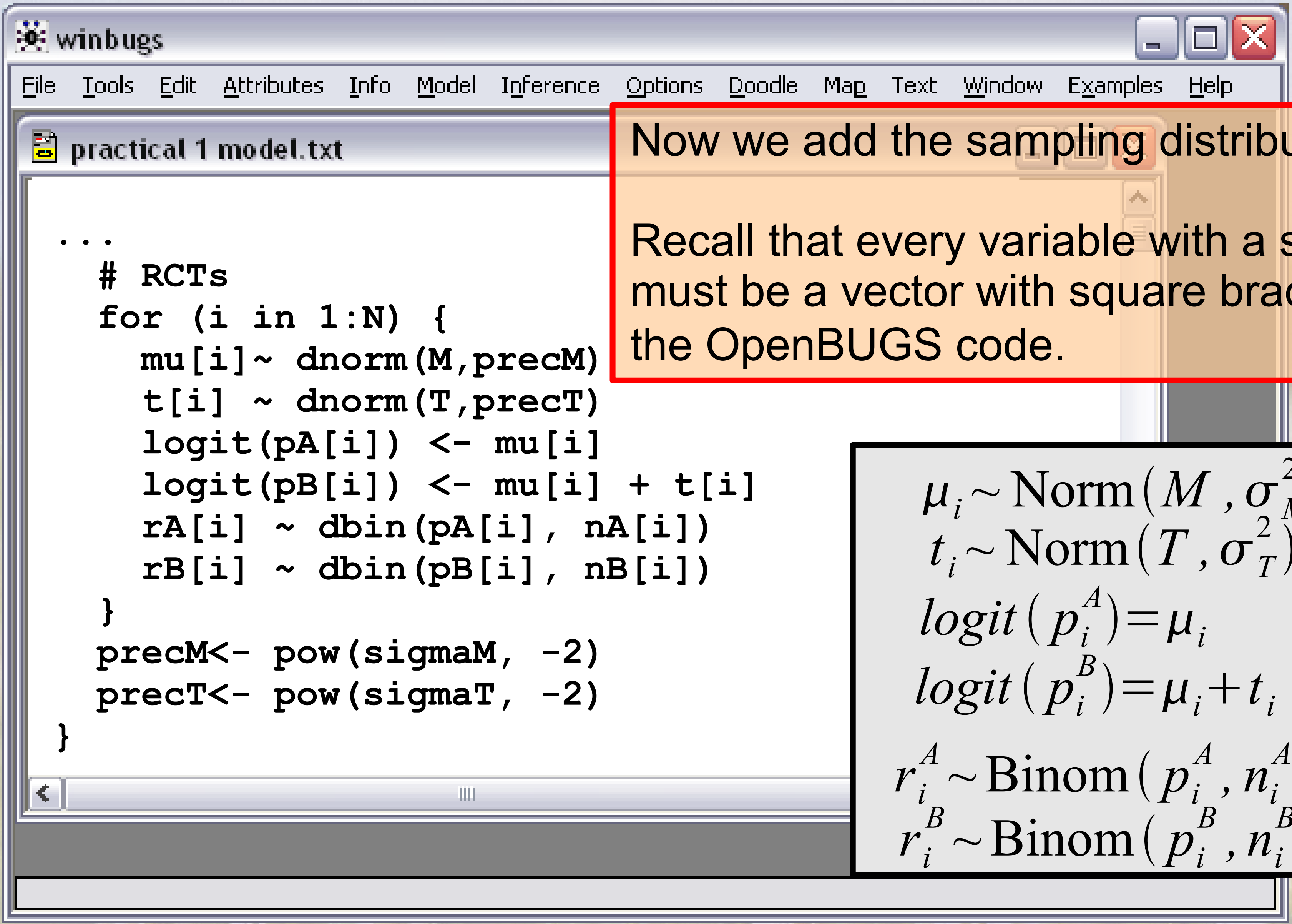
```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
    logit(pA[i]) <- mu[i]
    logit(pB[i]) <- mu[i] + t[i]
  }

  precM<- pow(sigmaM, -2)
  precT<- pow(sigmaT, -2)
}
```

Define p^B in a similar fashion.

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
...
  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
    logit(pA[i]) <- mu[i]
    logit(pB[i]) <- mu[i] + t[i]
    rA[i] ~ dbin(pA[i], nA[i])
    rB[i] ~ dbin(pB[i], nB[i])
  }
  precM<- pow(sigmaM, -2)
  precT<- pow(sigmaT, -2)
}
```

Now we add the sampling distributions.

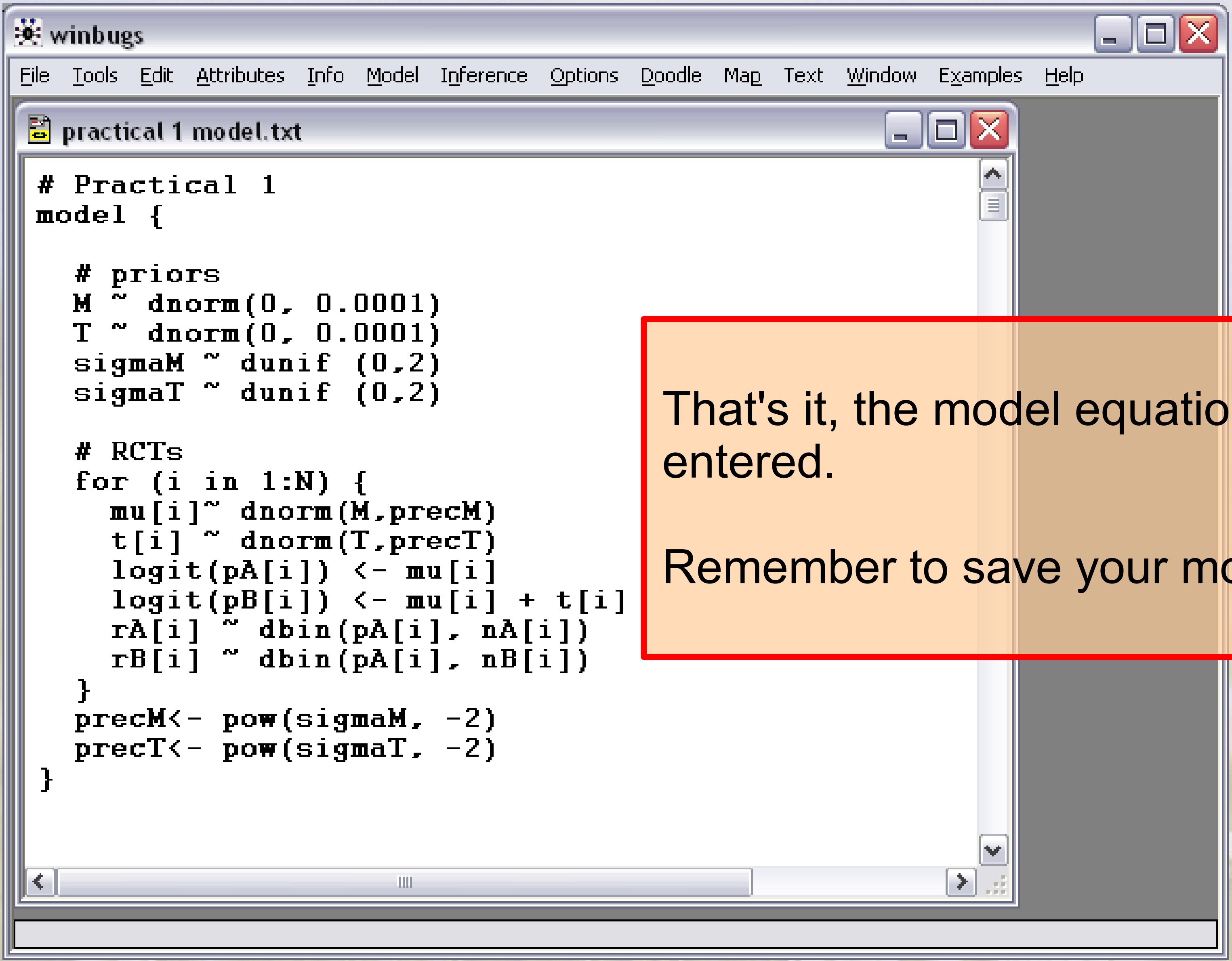
Recall that every variable with a subscript *i* must be a vector with square brackets **[i]** in the OpenBUGS code.

$$\mu_i \sim \mathrm{Norm}(M, \sigma_M^2)$$
$$t_i \sim \mathrm{Norm}(T, \sigma_T^2)$$
$$logit(p_i^A) = \mu_i$$
$$logit(p_i^B) = \mu_i + t_i$$
$$r_i^A \sim \mathrm{Binom}(p_i^A, n_i^A)$$
$$r_i^B \sim \mathrm{Binom}(p_i^B, n_i^B)$$

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
# Practical 1
model {

  # priors
  M ~ dnorm(0, 0.0001)
  T ~ dnorm(0, 0.0001)
  sigmaM ~ dunif (0,2)
  sigmaT ~ dunif (0,2)

  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
    logit(pA[i]) <- mu[i]
    logit(pB[i]) <- mu[i] + t[i]
    rA[i] ~ dbin(pA[i], nA[i])
    rB[i] ~ dbin(pA[i], nB[i])
  }
  precM<- pow(sigmaM, -2)
  precT<- pow(sigmaT, -2)
}
```

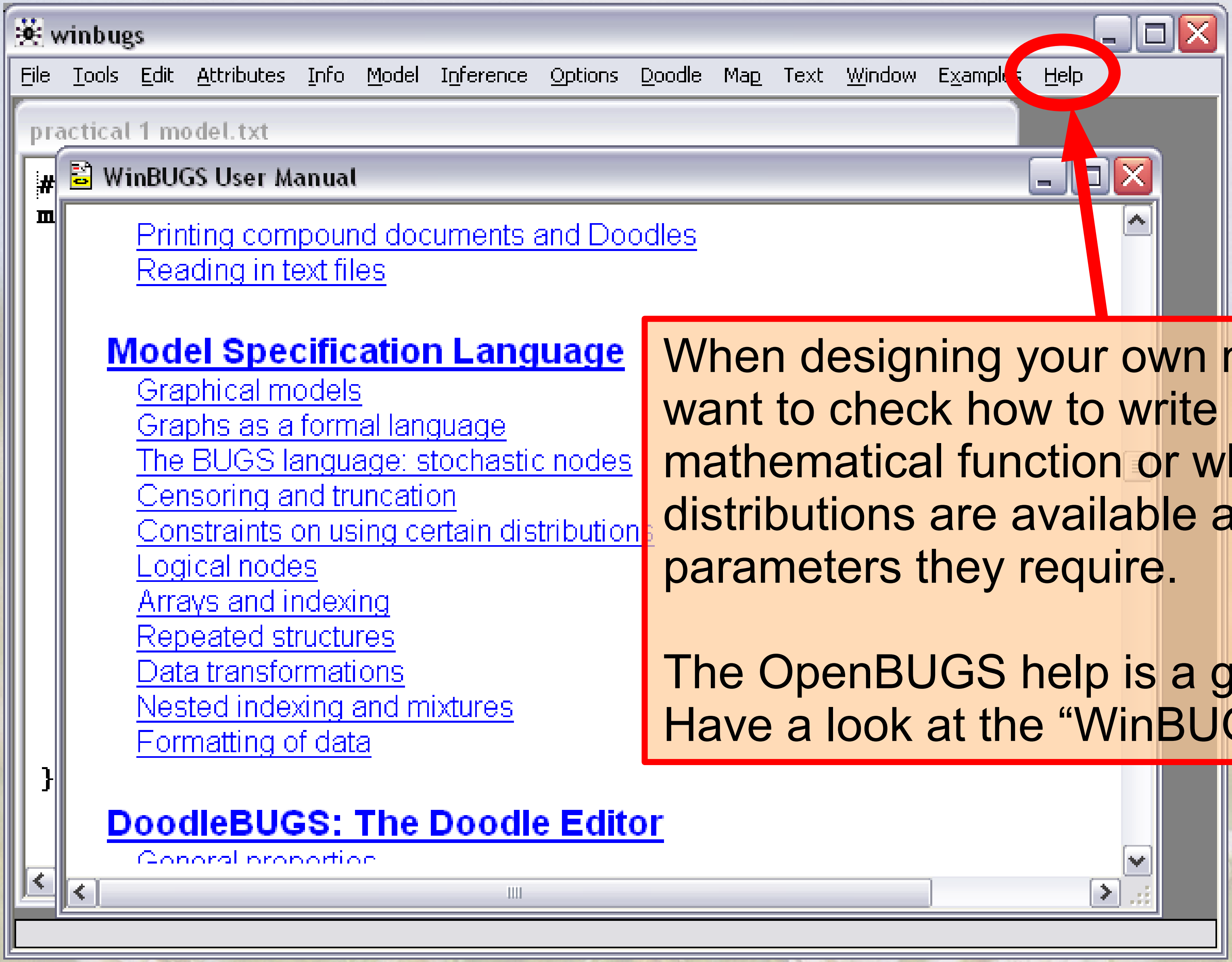
That's it, the model equations have been entered.

Remember to save your model!

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt
WinBUGS User Manual

Printing compound documents and Doodles
Reading in text files

Model Specification Language
Graphical models
Graphs as a formal language
The BUGS language: stochastic nodes
Censoring and truncation
Constraints on using certain distributions
Logical nodes
Arrays and indexing
Repeated structures
Data transformations
Nested indexing and mixtures
Formatting of data

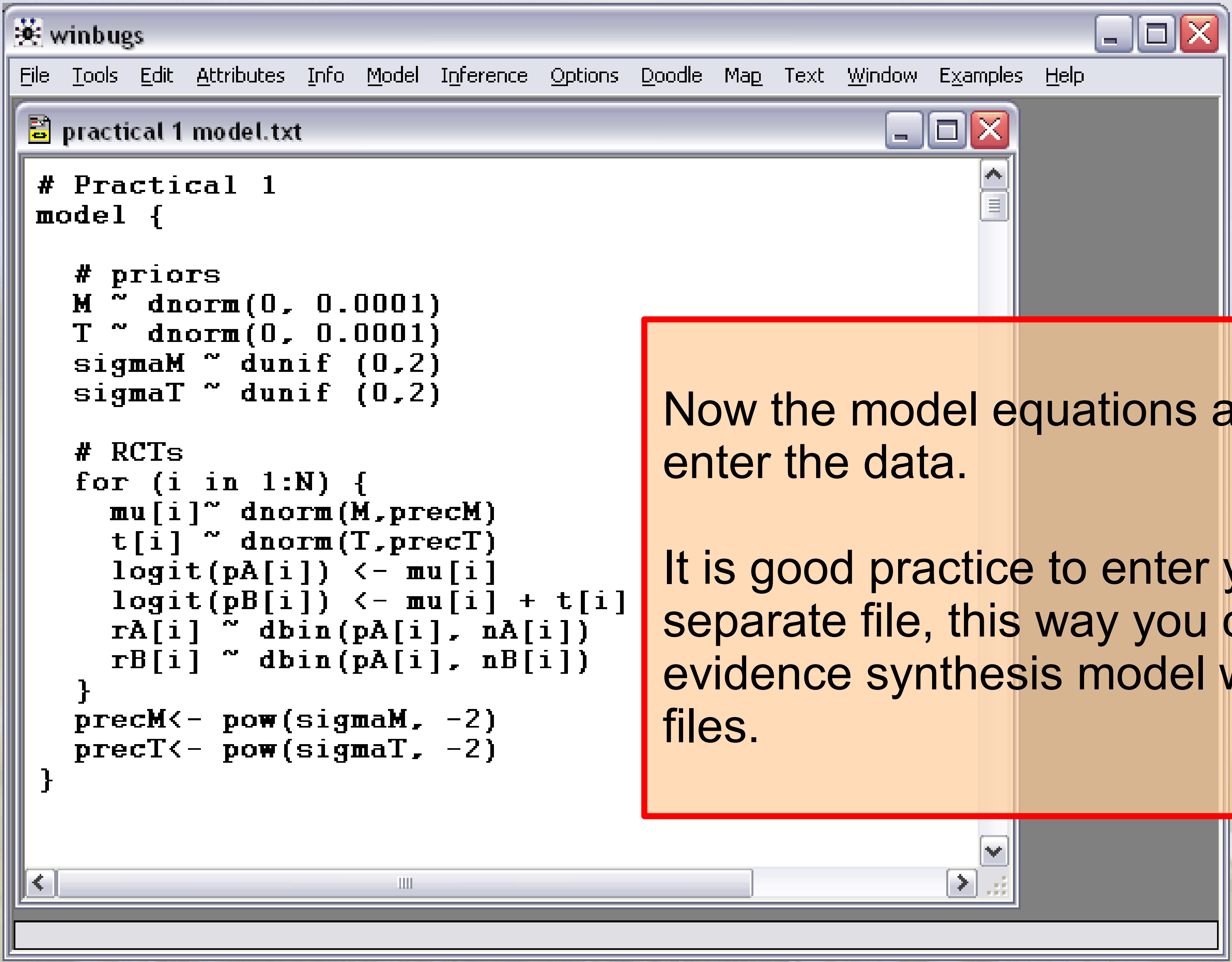
DoodleBUGS: The Doodle Editor

When designing your own models, you may want to check how to write a particular mathematical function or which probability distributions are available and what parameters they require.

The OpenBUGS help is a great place to start. Have a look at the "WinBUGS User Manual"!

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt

```
# Practical 1
model {

   # priors
   M ~ dnorm(0, 0.0001)
   T ~ dnorm(0, 0.0001)
   sigmaM ~ dunif (0,2)
   sigmaT ~ dunif (0,2)

   # RCTs
   for (i in 1:N) {
     mu[i]~ dnorm(M,precM)
     t[i] ~ dnorm(T,precT)
     logit(pA[i]) <- mu[i]
     logit(pB[i]) <- mu[i] + t[i]
     rA[i] ~ dbin(pA[i], nA[i])
     rB[i] ~ dbin(pA[i], nB[i])
   }
   precM<- pow(sigmaM, -2)
   precT<- pow(sigmaT, -2)
}
```

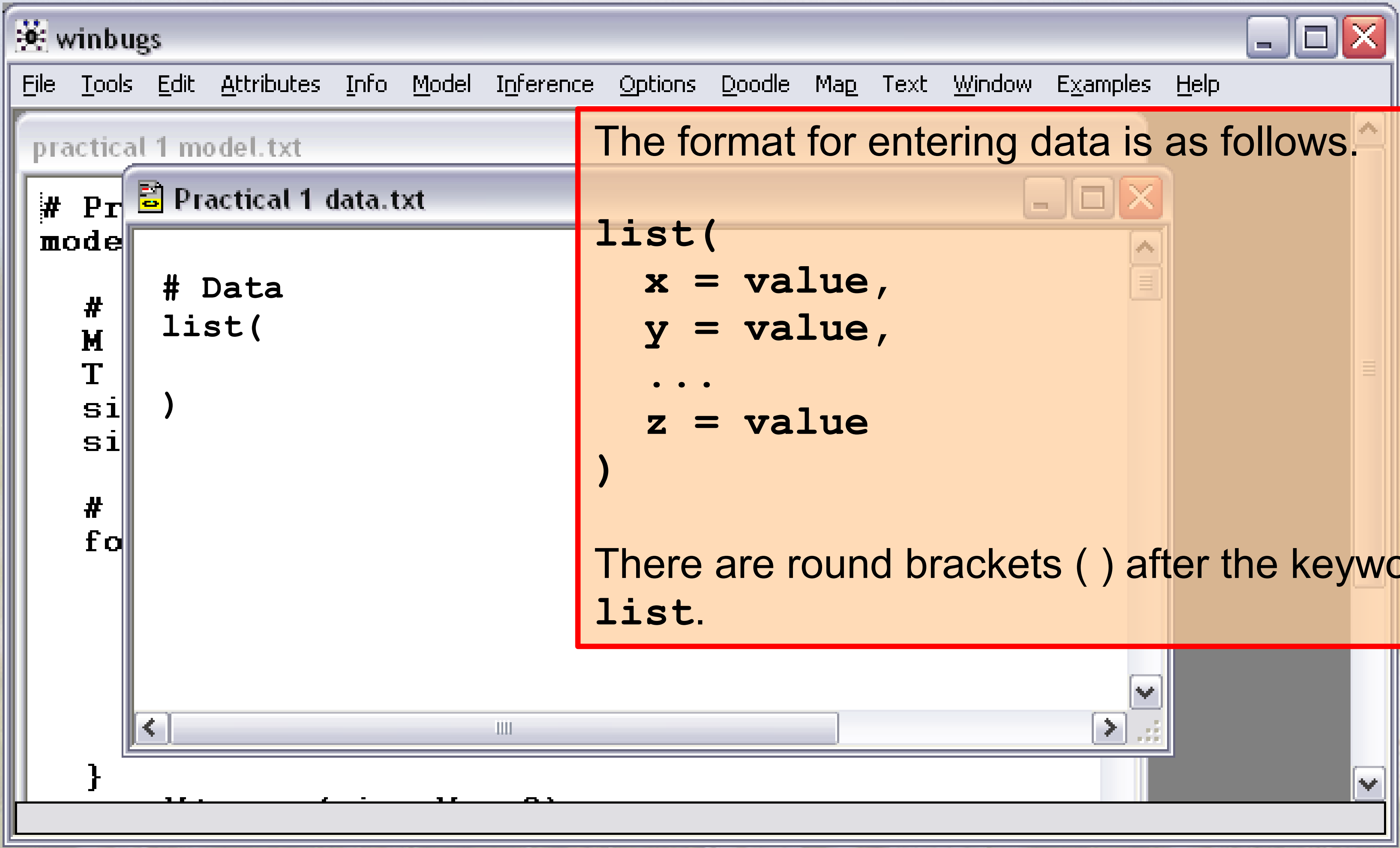
Now the model equations are done, next we enter the data.

It is good practice to enter your data into a separate file, this way you can reuse one evidence synthesis model with different data files.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt
Practical 1 data.txt

```
# Data
list(

)
```

The format for entering data is as follows.

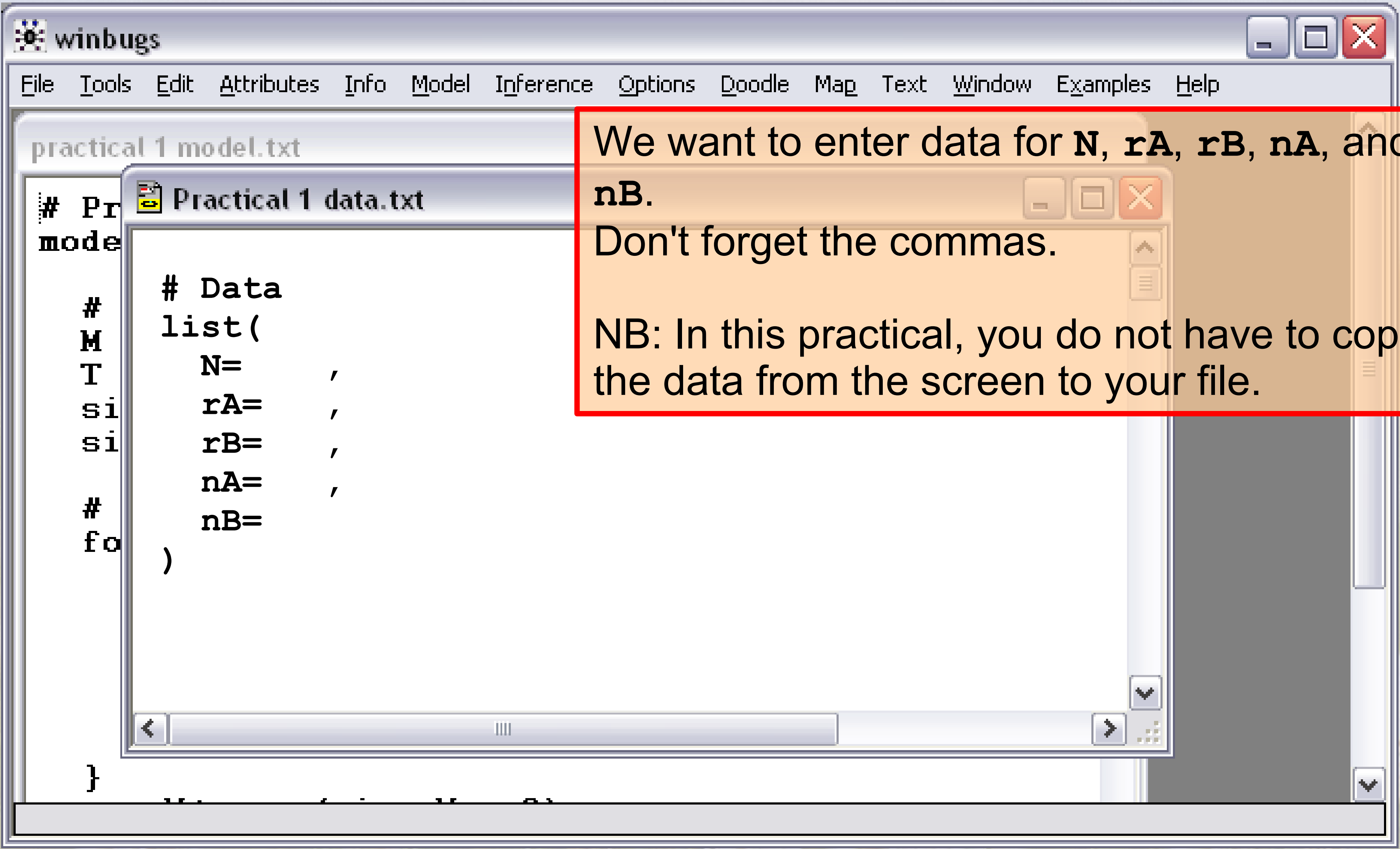
```
list(
   x = value,
   y = value,
   ...
   z = value
)
```

There are round brackets ( ) after the keyword **`list`**.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt
Practical 1 data.txt

```
# Data
list(
   N=     ,
   rA=    ,
   rB=    ,
   nA=    ,
   nB=
)
```

We want to enter data for **`N`**, **`rA`**, **`rB`**, **`nA`**, and **`nB`**.
Don't forget the commas.

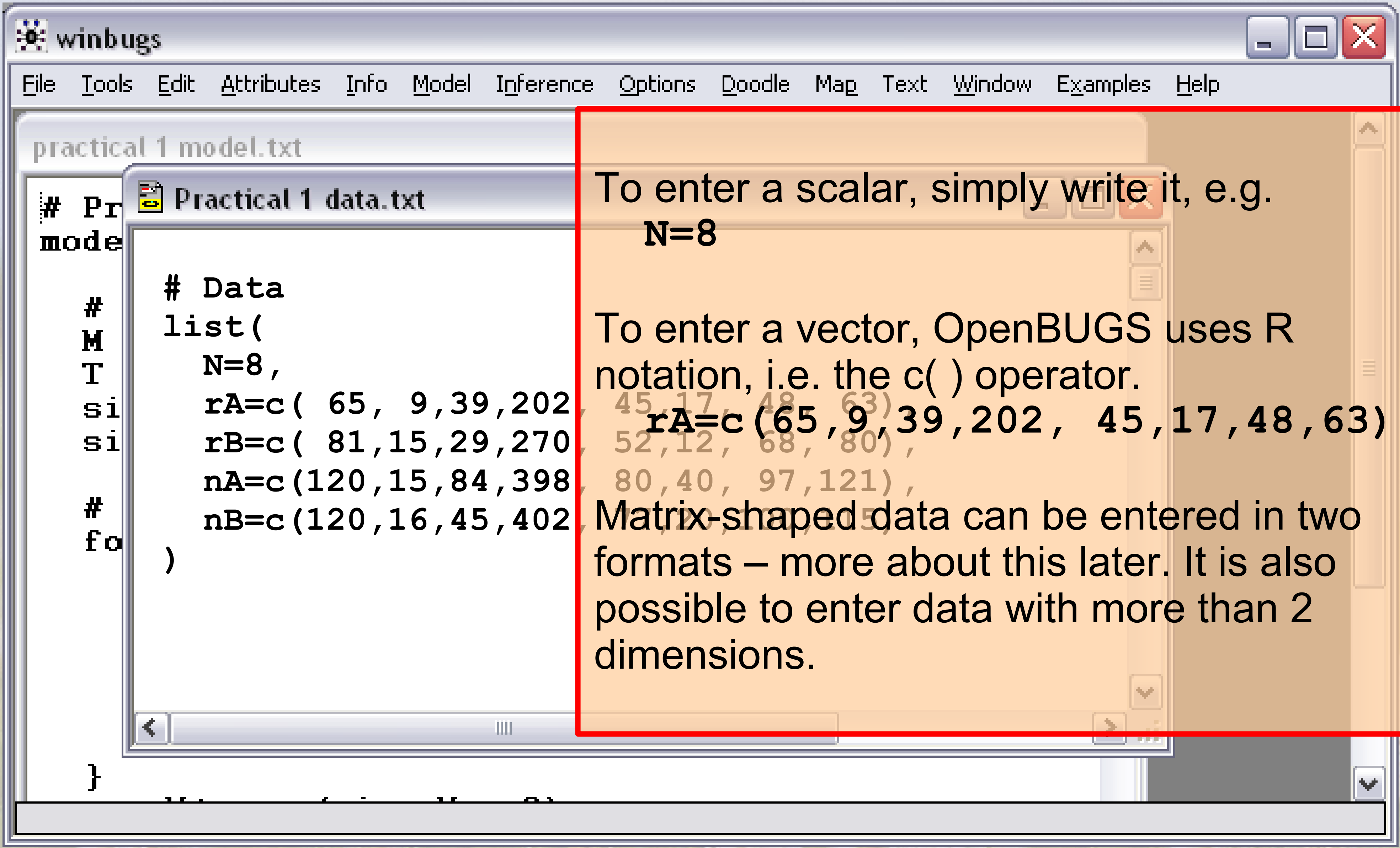
NB: In this practical, you do not have to copy the data from the screen to your file.

2007-03-12, Linköping

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt
Practical 1 data.txt

```
# Data
list(
  N=8,
  rA=c( 65, 9,39,202, 45,17, 48, 63),
  rB=c( 81,15,29,270, 52,12, 68, 80),
  nA=c(120,15,84,398, 80,40, 97,121),
  nB=c(120,16,45,402
)
```

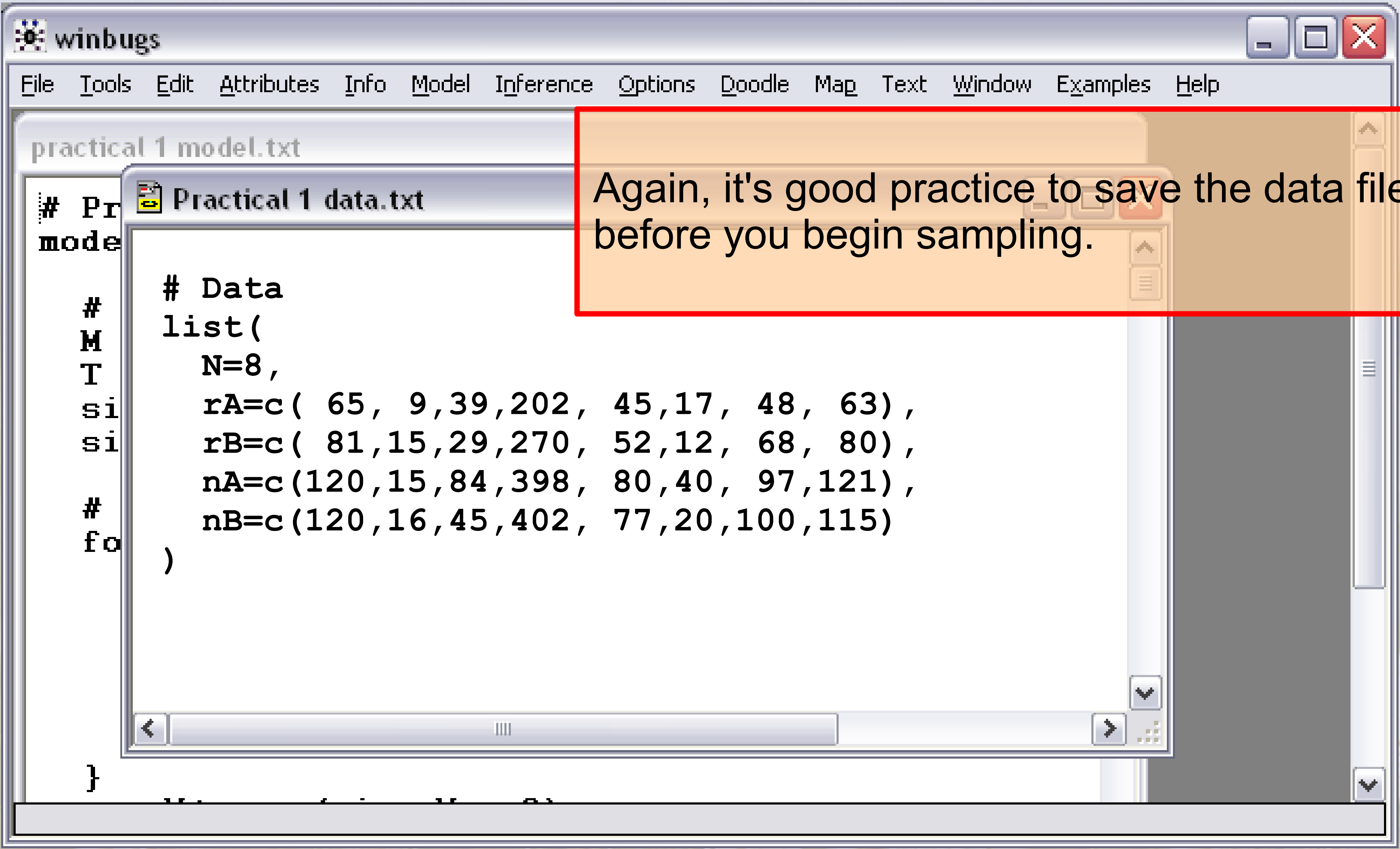
To enter a scalar, simply write it, e.g.
`N=8`

To enter a vector, OpenBUGS uses R notation, i.e. the c( ) operator.
`rA=c(65,9,39,202, 45,17,48,63)`

Matrix-shaped data can be entered in two formats – more about this later. It is also possible to enter data with more than 2 dimensions.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS


winbugs
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Examples Help
practical 1 model.txt
Practical 1 data.txt

```
# Data
list(
  N=8,
  rA=c( 65, 9,39,202, 45,17, 48, 63),
  rB=c( 81,15,29,270, 52,12, 68, 80),
  nA=c(120,15,84,398, 80,40, 97,121),
  nB=c(120,16,45,402, 77,20,100,115)
)
```

Again, it's good practice to save the data file before you begin sampling.

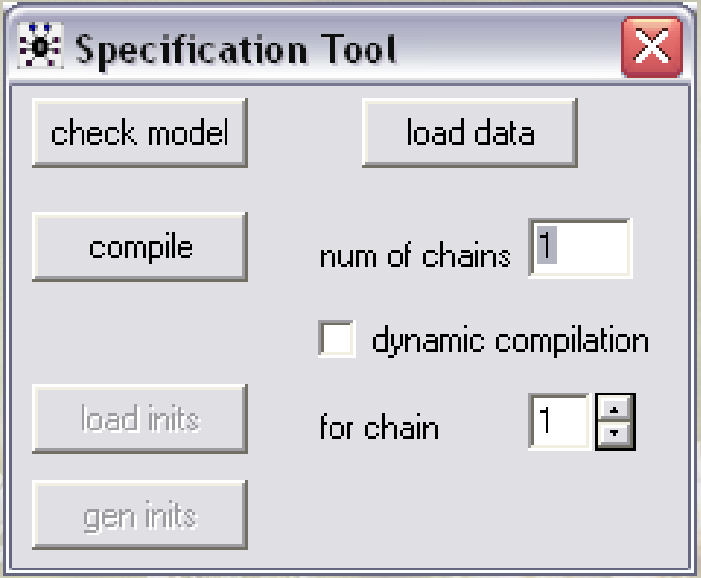
Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Overview of model specification

- When you have entered the model and the data, open the "Specification Tool" on the "Model" menu.

1. "Check" the model
2. "Load" the data
3. "Compile" the model
4. Load or generate initial values



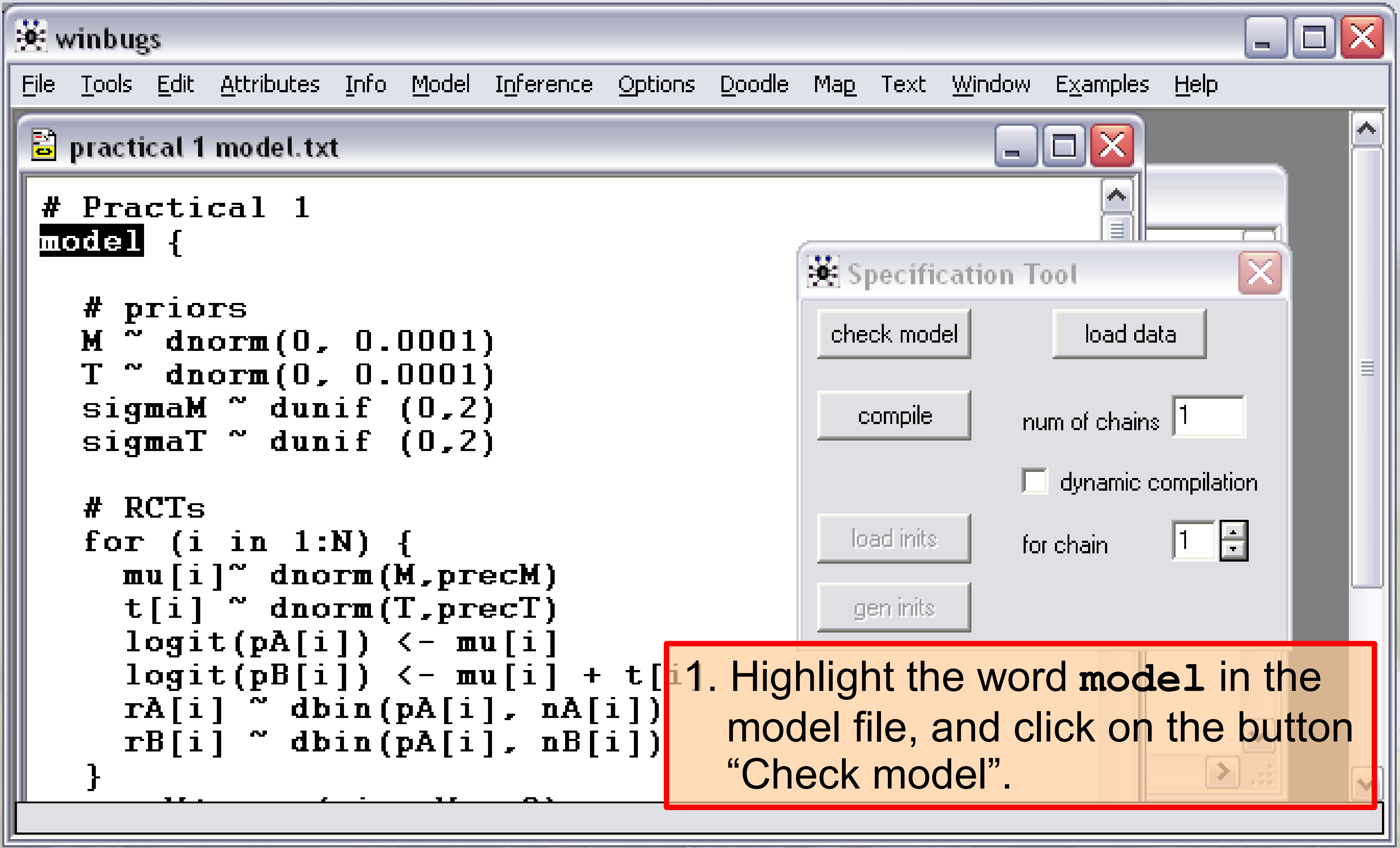
num of chains 1

☐ dynamic compilation

for chain 1

Dr Christian Asseburg
University of York, UK
ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS

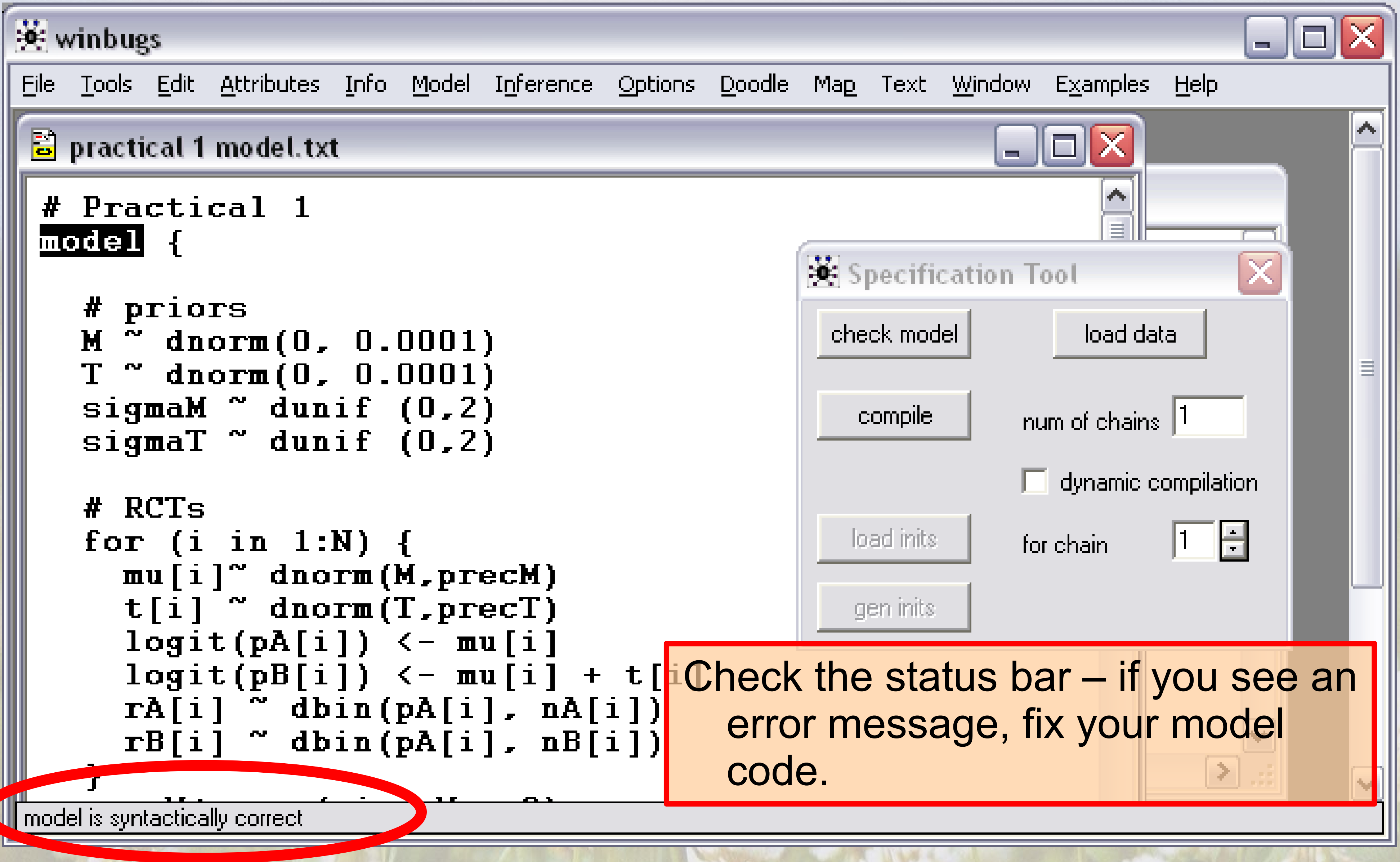


```
# Practical 1
model {

  # priors
  M ~ dnorm(0, 0.0001)
  T ~ dnorm(0, 0.0001)
  sigmaM ~ dunif (0,2)
  sigmaT ~ dunif (0,2)

  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
    logit(pA[i]) <- mu[i]
    logit(pB[i]) <- mu[i] + t[i
    rA[i] ~ dbin(pA[i], nA[i])
    rB[i] ~ dbin(pA[i], nB[i])
  }
```

1. Highlight the word **`model`** in the model file, and click on the button "Check model".

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS



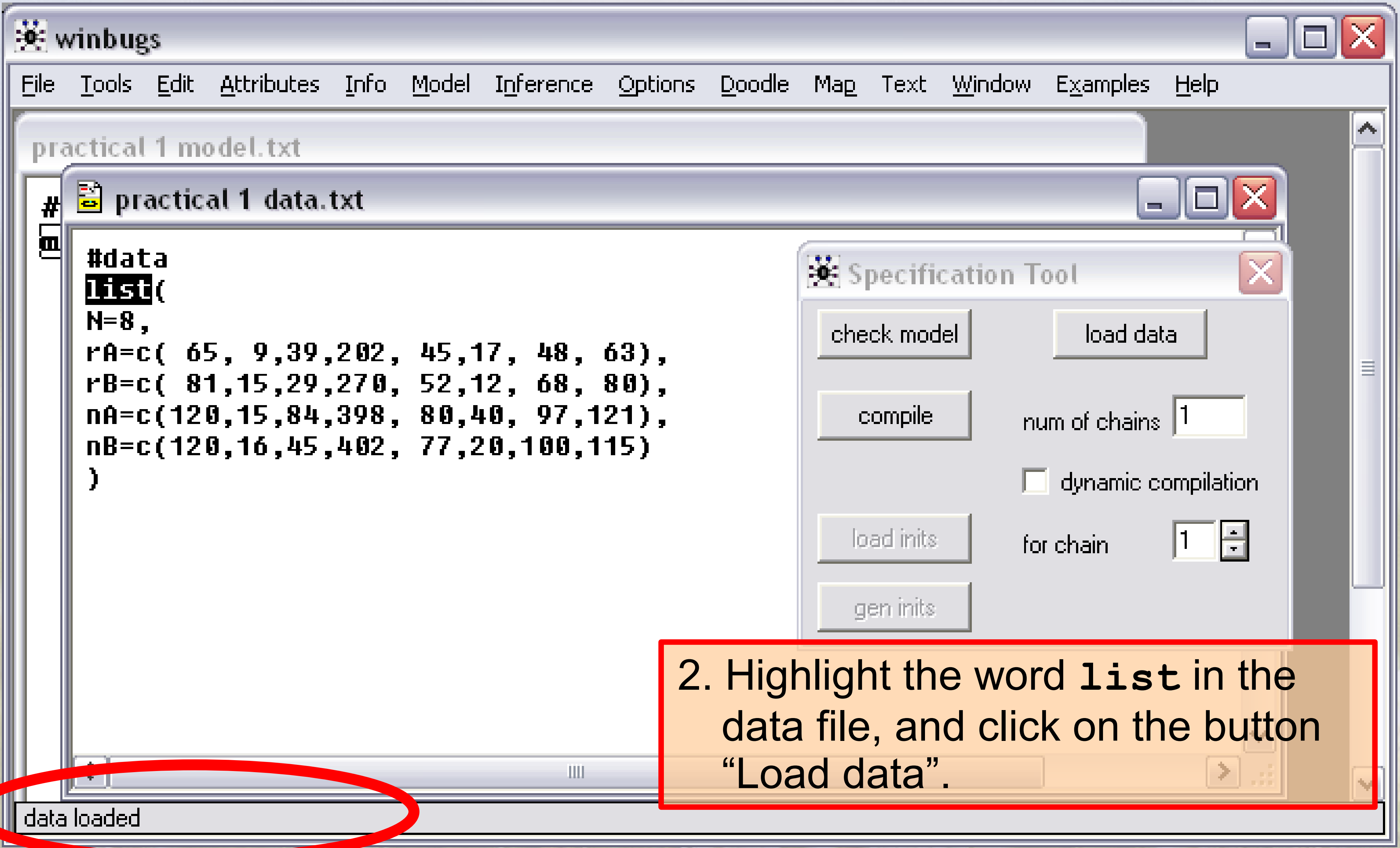
```
# Practical 1
model {

  # priors
  M ~ dnorm(0, 0.0001)
  T ~ dnorm(0, 0.0001)
  sigmaM ~ dunif (0,2)
  sigmaT ~ dunif (0,2)

  # RCTs
  for (i in 1:N) {
    mu[i]~ dnorm(M,precM)
    t[i] ~ dnorm(T,precT)
    logit(pA[i]) <- mu[i]
    logit(pB[i]) <- mu[i] + t[i
    rA[i] ~ dbin(pA[i], nA[i])
    rB[i] ~ dbin(pA[i], nB[i])
  }
```

```
model is syntactically correct
```

Check the status bar – if you see an error message, fix your model code.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS



```
#data
list(
N=8,
rA=c( 65, 9,39,202, 45,17, 48, 63),
rB=c( 81,15,29,270, 52,12, 68, 80),
nA=c(120,15,84,398, 80,40, 97,121),
nB=c(120,16,45,402, 77,20,100,115)
)
```

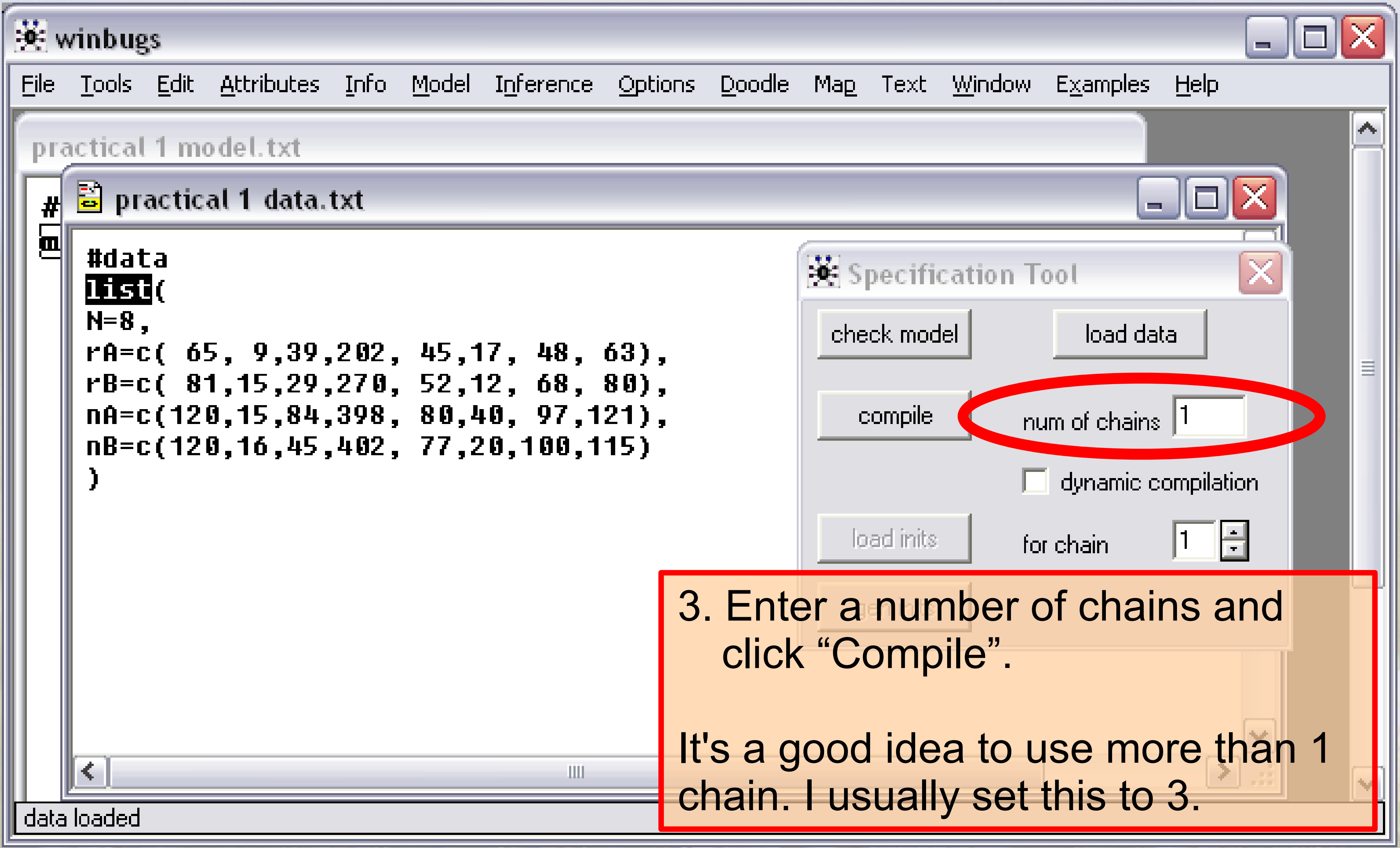
data loaded

2. Highlight the word `list` in the data file, and click on the button "Load data".

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



```
#data
list(
N=8,
rA=c( 65, 9,39,202, 45,17, 48, 63),
rB=c( 81,15,29,270, 52,12, 68, 80),
nA=c(120,15,84,398, 80,40, 97,121),
nB=c(120,16,45,402, 77,20,100,115)
)
```

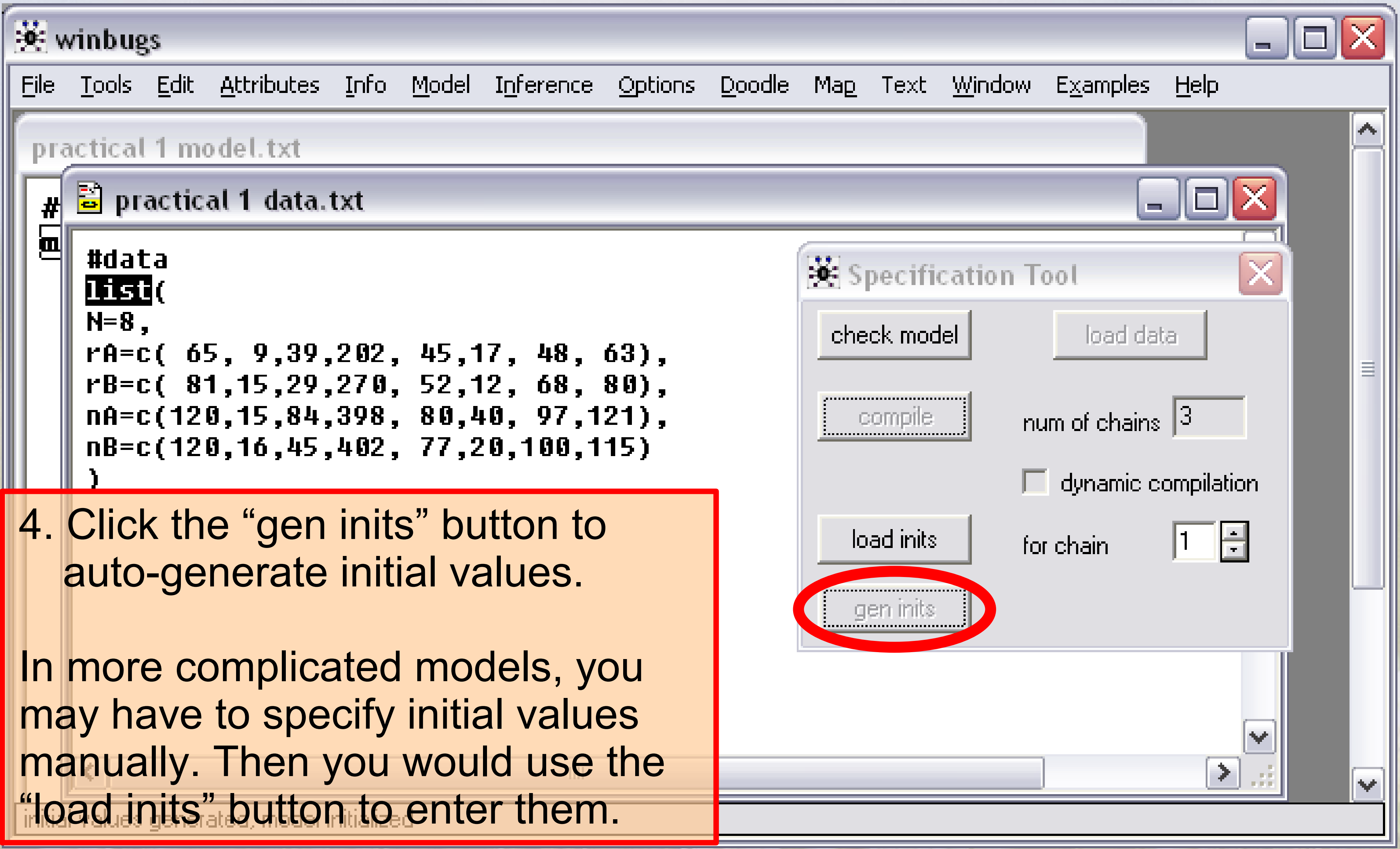
3. Enter a number of chains and click "Compile".

It's a good idea to use more than 1 chain. I usually set this to 3.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



```
#data
list(
N=8,
rA=c( 65, 9,39,202, 45,17, 48, 63),
rB=c( 81,15,29,270, 52,12, 68, 80),
nA=c(120,15,84,398, 80,40, 97,121),
nB=c(120,16,45,402, 77,20,100,115)
)
```

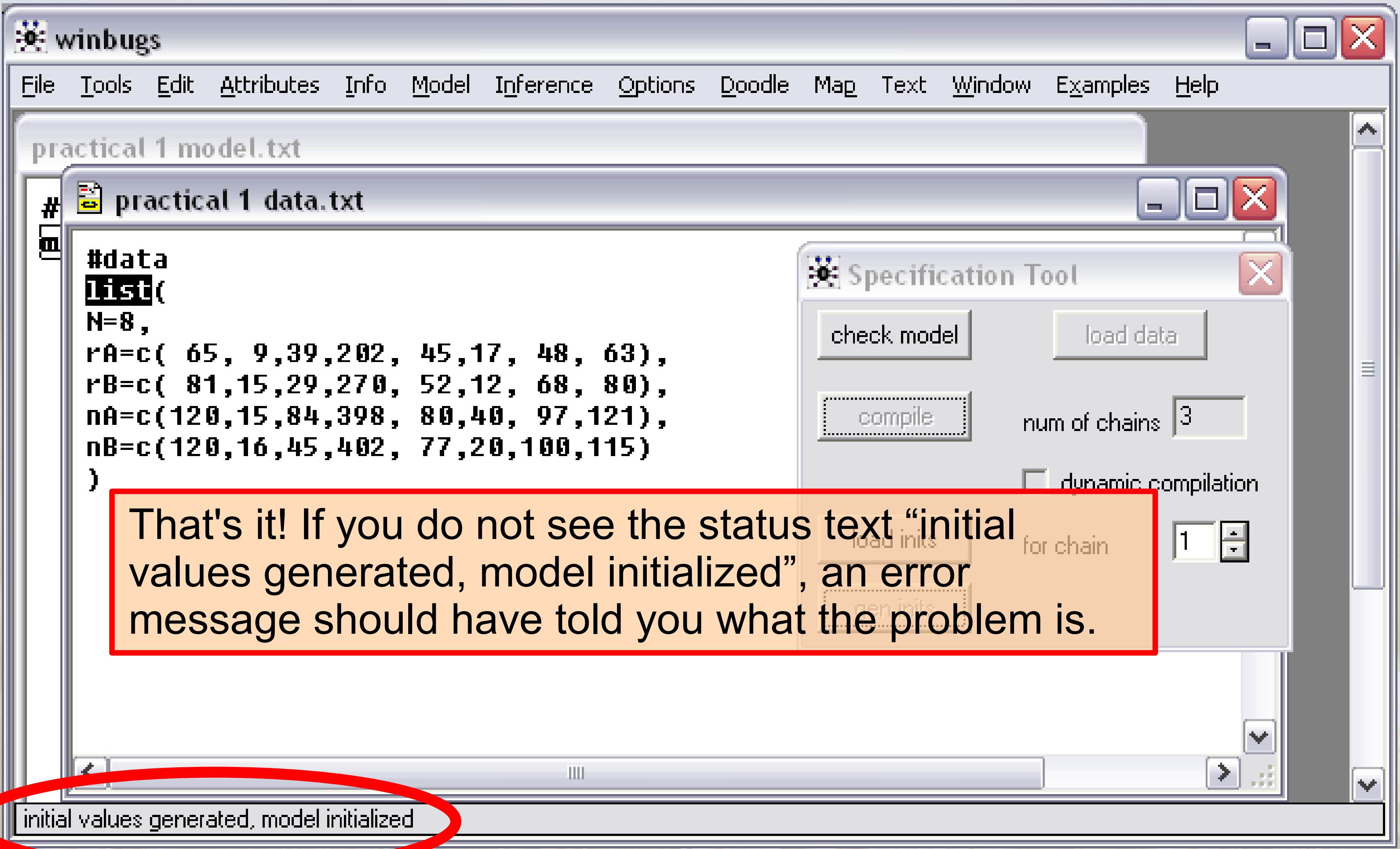
4. Click the “gen inits” button to auto-generate initial values.

In more complicated models, you may have to specify initial values manually. Then you would use the “load inits” button to enter them.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



```
#data
list(
N=8,
rA=c( 65, 9,39,202, 45,17, 48, 63),
rB=c( 81,15,29,270, 52,12, 68, 80),
nA=c(120,15,84,398, 80,40, 97,121),
nB=c(120,16,45,402, 77,20,100,115)
)
```

```
initial values generated, model initialized
```

That's it! If you do not see the status text “initial values generated, model initialized”, an error message should have told you what the problem is.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Common errors

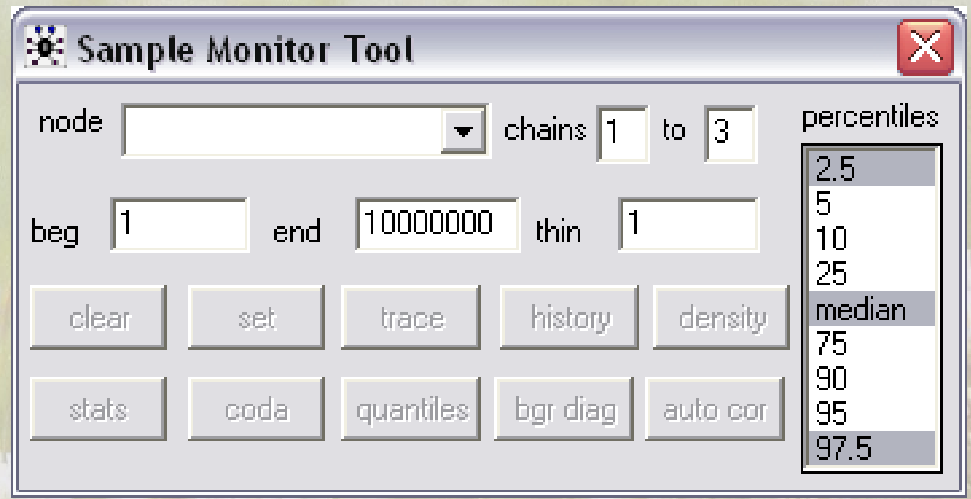
- After "Check model"
  - Syntax errors like commas and ( ) and [ ] and { }
  - Using the same variable name sometimes as scalar and sometimes as array
- After "Load data"
  - Loading data for variables that are defined by `<-`
  - Loading data of the wrong shape for a vector or array
- After "Compile"
  - Defining a variable twice or forgetting to specify some required data
  - Leaving out the correct [ ] indices for variables that are defined inside a "for" loop

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Overview of the sampling steps

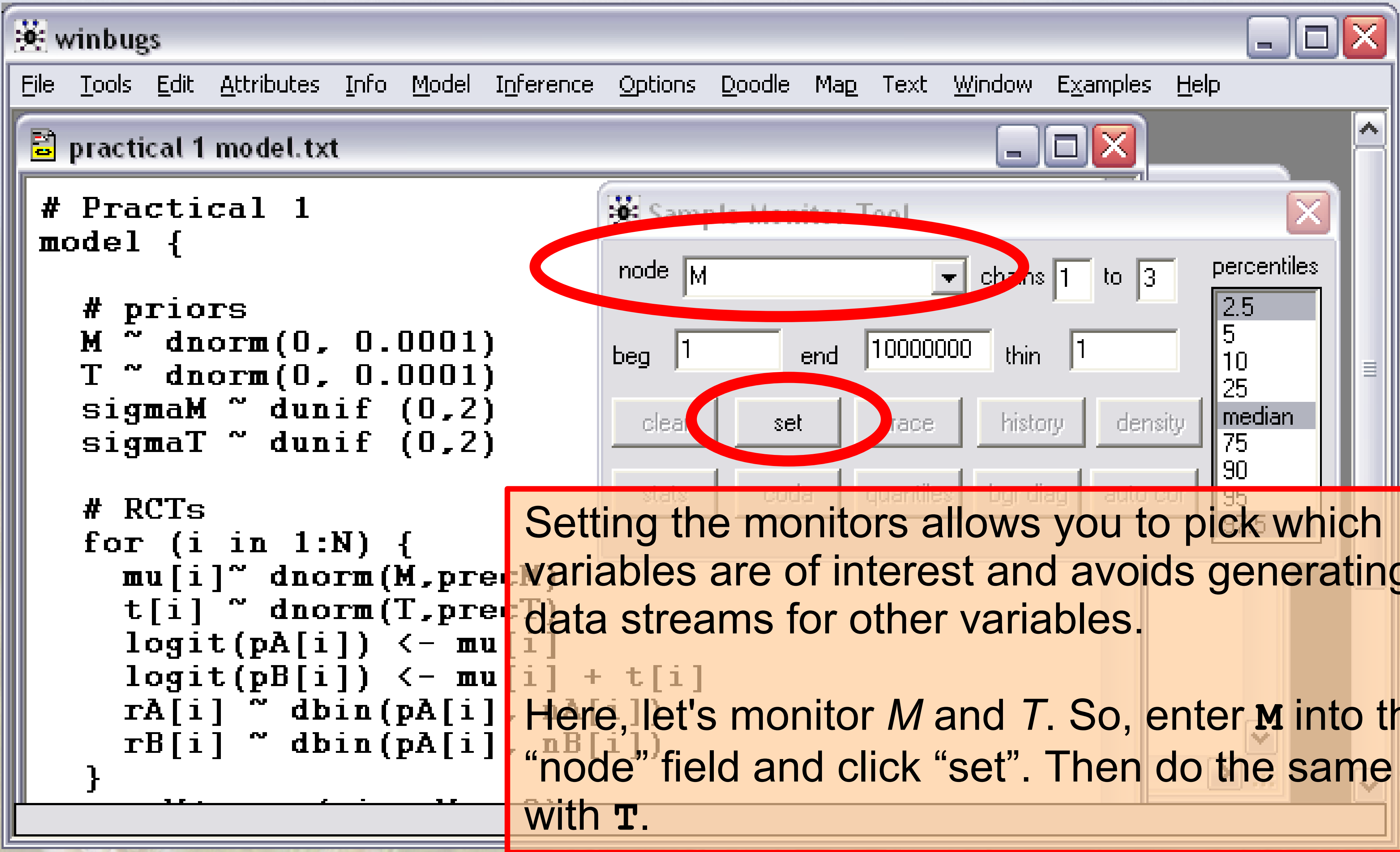
- After compiling and initialising your model, open the "Sample Monitor Tool" on the "Inference" menu

1. Set monitors for the parameters of interest
2. Using the "Update tool", simulate draws from the posterior
3. Check for issues like burn-in and convergence
4. Retrieve sample summary or histograms, or export draws to another program



Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



```
# Practical 1
model {

   # priors
   M ~ dnorm(0, 0.0001)
   T ~ dnorm(0, 0.0001)
   sigmaM ~ dunif (0,2)
   sigmaT ~ dunif (0,2)

   # RCTs
   for (i in 1:N) {
      mu[i]~ dnorm(M,precM)
      t[i] ~ dnorm(T,precT)
      logit(pA[i]) <- mu[i]
      logit(pB[i]) <- mu[i] + t[i]
      rA[i] ~ dbin(pA[i],nA[i])
      rB[i] ~ dbin(pA[i],nB[i])
   }
```

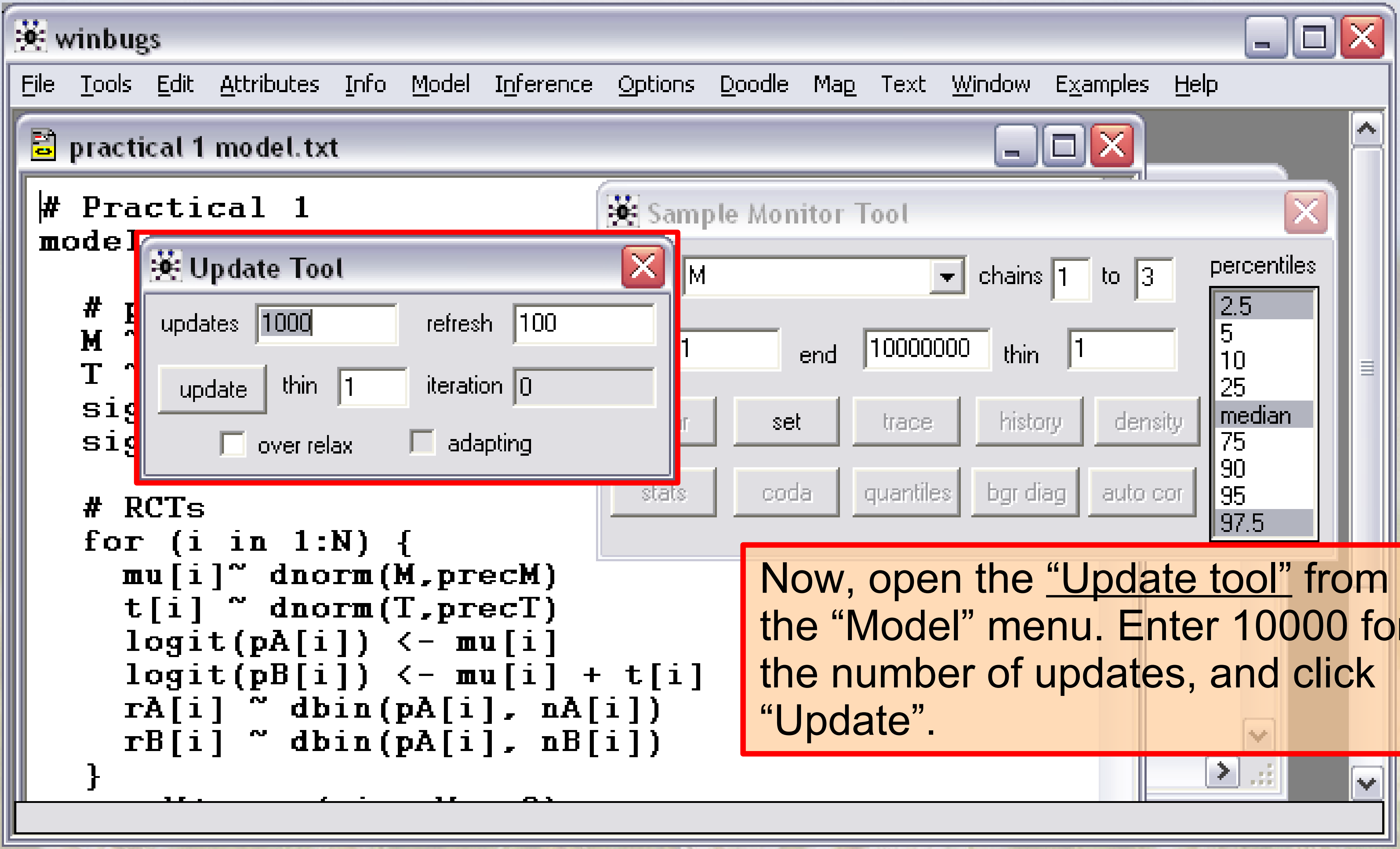
Setting the monitors allows you to pick which variables are of interest and avoids generating data streams for other variables.

Here, let's monitor *M* and *T*. So, enter **`M`** into the "node" field and click "set". Then do the same with **`T`**.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



```
# Practical 1
model

    # RCTs
    for (i in 1:N) {
      mu[i]~ dnorm(M,precM)
      t[i] ~ dnorm(T,precT)
      logit(pA[i]) <- mu[i]
      logit(pB[i]) <- mu[i] + t[i]
      rA[i] ~ dbin(pA[i], nA[i])
      rB[i] ~ dbin(pA[i], nB[i])
    }
```

Now, open the "Update tool" from the "Model" menu. Enter 10000 for the number of updates, and click "Update".

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Common errors during “Update”

If you get error messages while OpenBUGS is generating draws from the posterior, these are usually displayed in the form of cryptic “trap” messages. It can be difficult to figure out what went wrong.

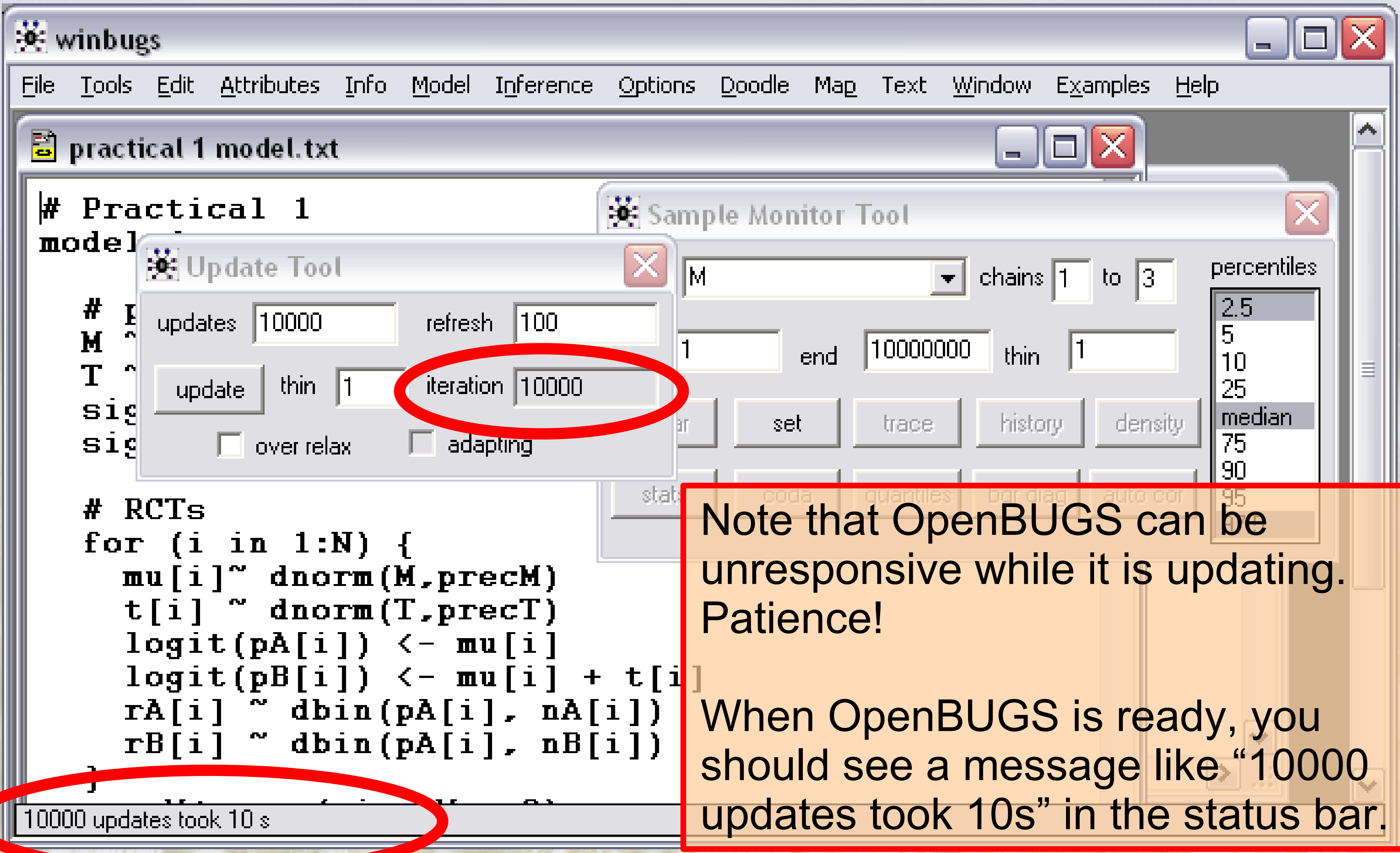
The most common problems are unsuitable initial values, a model that leaves part of posterior parameter space undefined, or a model that is too complex for OpenBUGS.

Hopefully you do not run into any of these during this first practical!

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



```
# Practical 1
model {

   # P
   M ~
   T ~
   sig
   sig

   # RCTs
   for (i in 1:N) {
      mu[i]~ dnorm(M,precM)
      t[i] ~ dnorm(T,precT)
      logit(pA[i]) <- mu[i]
      logit(pB[i]) <- mu[i] + t[i]
      rA[i] ~ dbin(pA[i], nA[i])
      rB[i] ~ dbin(pA[i], nB[i])
   }
```

Update Tool: updates 10000, refresh 100, thin 1, iteration 10000

Sample Monitor Tool: M, chains 1 to 3, end 10000000, thin 1

10000 updates took 10 s

Note that OpenBUGS can be unresponsive while it is updating. Patience!

When OpenBUGS is ready, you should see a message like “10000 updates took 10s” in the status bar.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Some basic checks

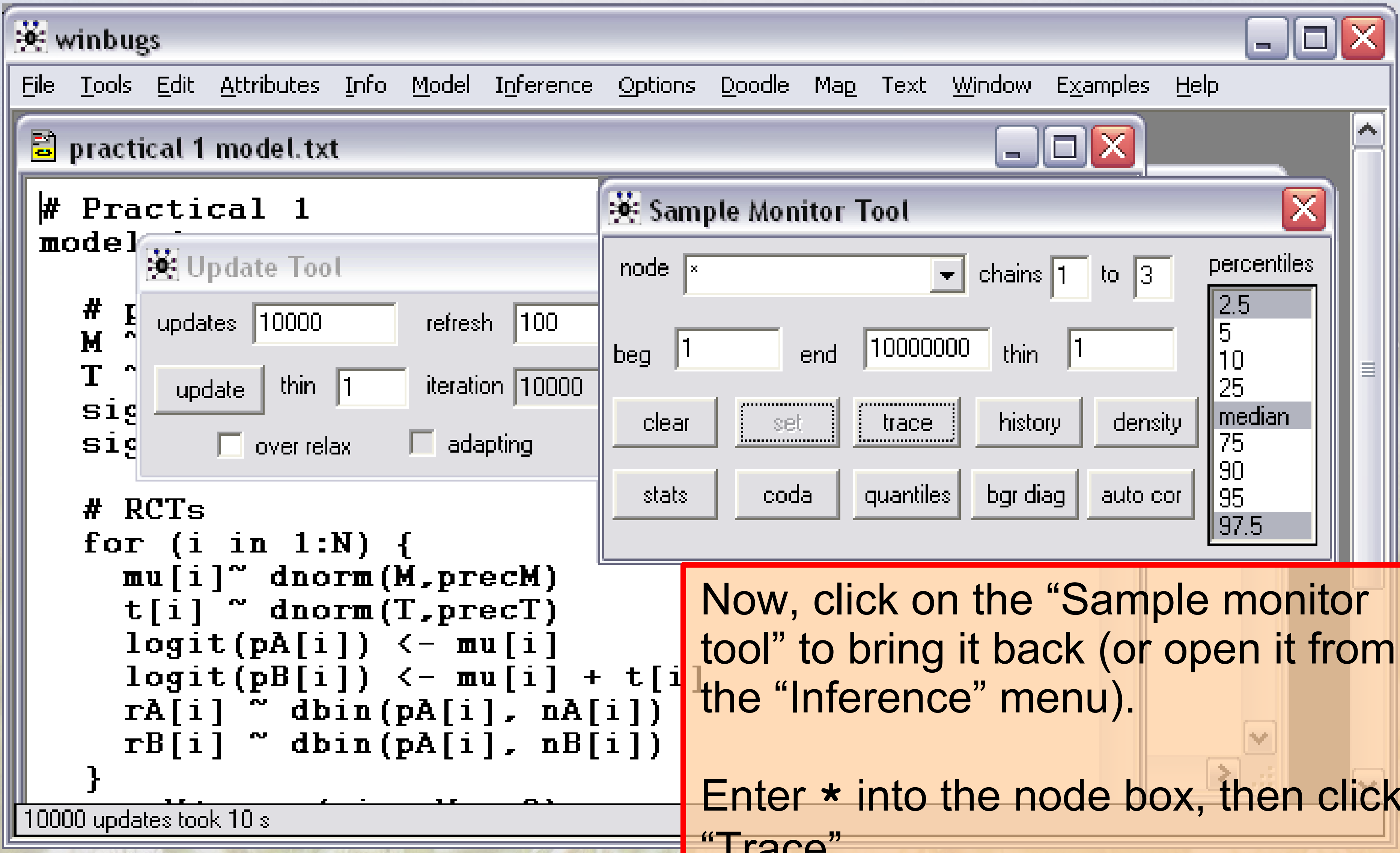
Sampling Bayesian posteriors through a numerical method like MCMC (e.g. with OpenBUGS) can go wrong if the sampler does not explore model space well.

In this practical we will check 3 diagnostics. You should always check them.

Trace	The simultaneous chains should wiggle in the same area in posterior model space.
BGR	The red line should be very close to 1 at the RHS.
Auto-correlation	Most of the bars in these graphs should be small.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



```
# Practical 1
model {

   # P
   M ~
   T ~
   sig
   sig

   # RCTs
   for (i in 1:N) {
     mu[i]~ dnorm(M,precM)
     t[i] ~ dnorm(T,precT)
     logit(pA[i]) <- mu[i]
     logit(pB[i]) <- mu[i] + t[i]
     rA[i] ~ dbin(pA[i], nA[i])
     rB[i] ~ dbin(pA[i], nB[i])
   }
```

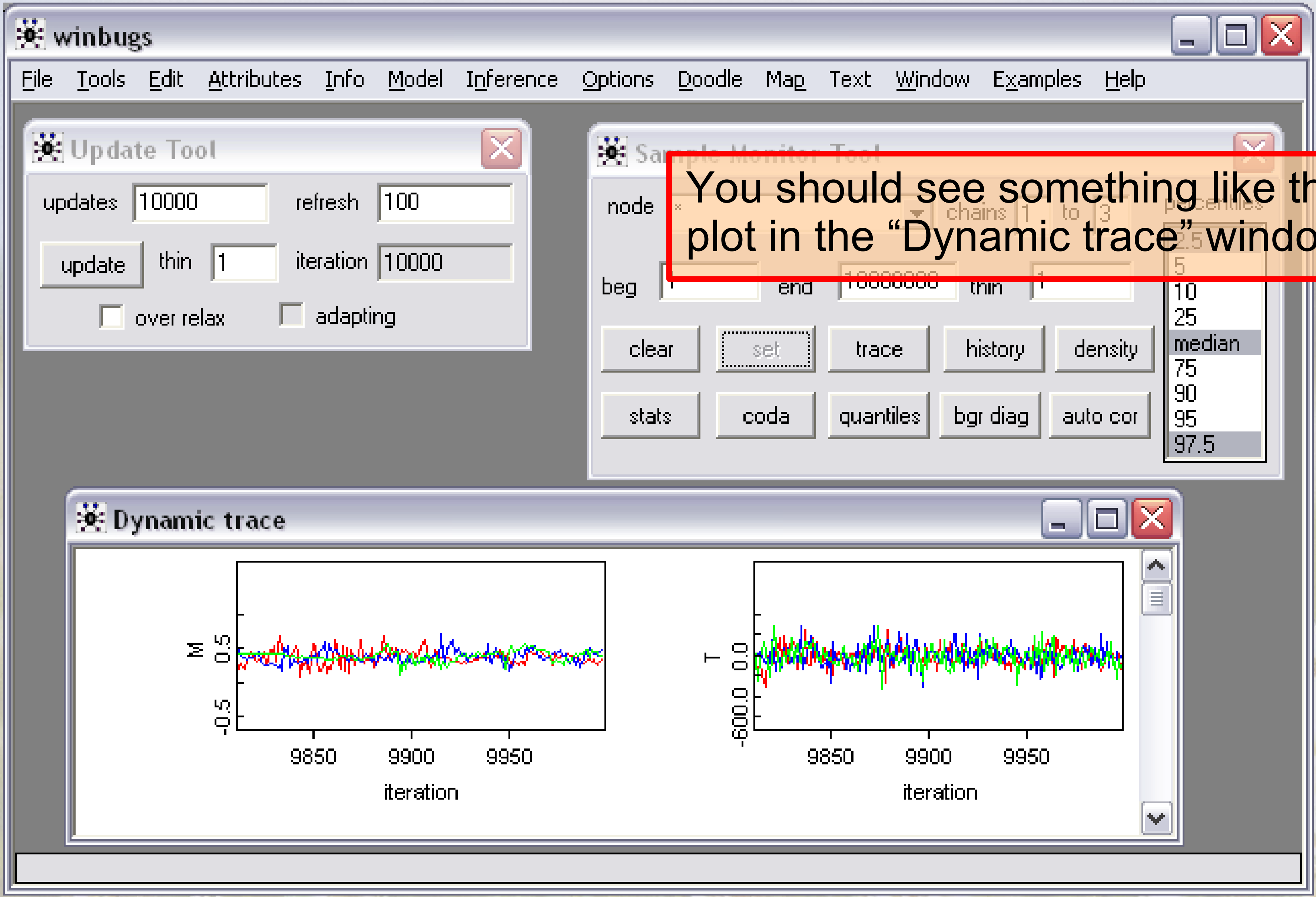
Now, click on the “Sample monitor tool” to bring it back (or open it from the “Inference” menu).

Enter * into the node box, then click “Trace”.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



updates 10000
refresh 100
thin 1
iteration 10000
beg 1
end 10000000
thin 1

You should see something like this plot in the "Dynamic trace" window.

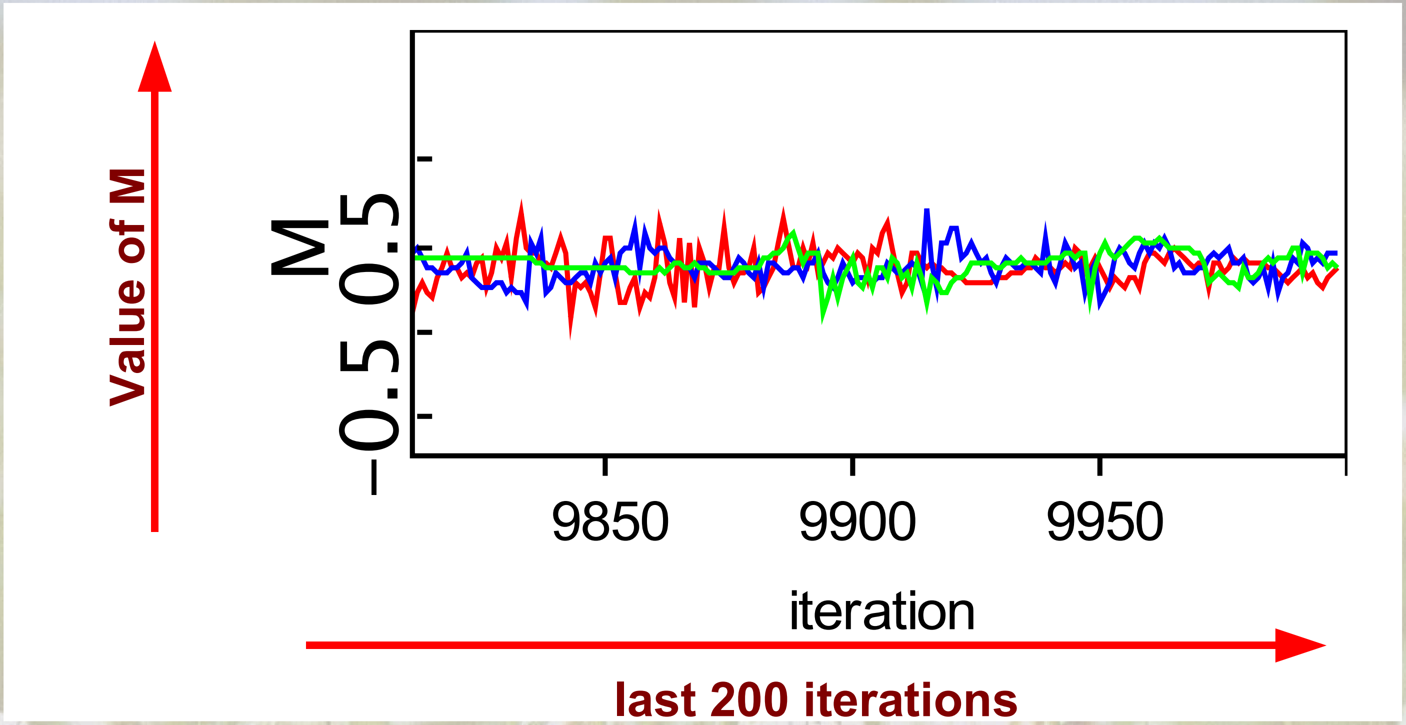
Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS

You should see something like this plot in the "Dynamic trace" window. Each colour represents one of the chains in the posterior sample.

Check the trace to see that all chains "wiggle" and that they overlap well. The plot below looks quite good.



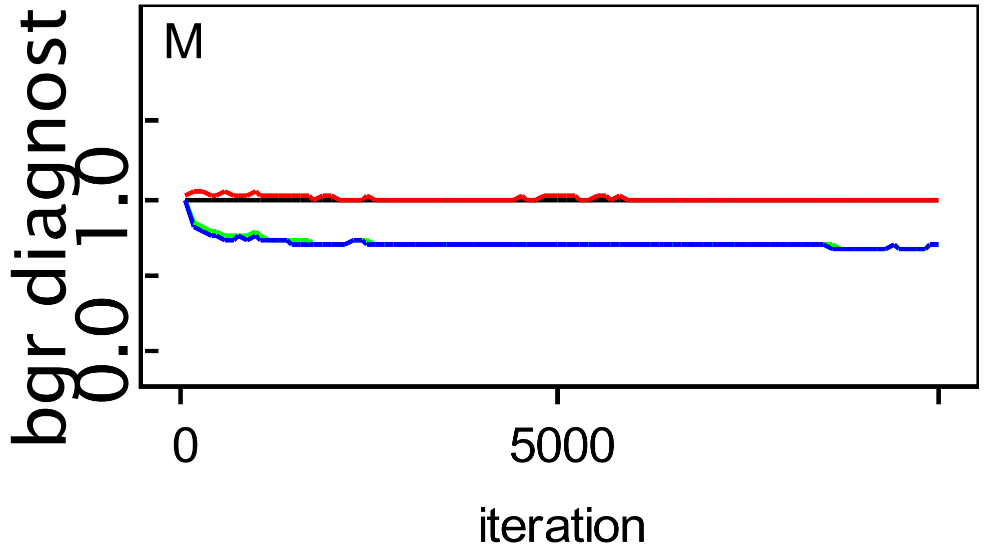
Dr Christian Asseburg
University of York, UK
ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS

Now create the BGR plot for all monitors `*`.

Note that the colours in this plot do not represent the three different chains. There are always three colours (red, green, blue), the red line matters in this test.

The red line should be very close to 1.0 on the right-hand side. This plot looks excellent.

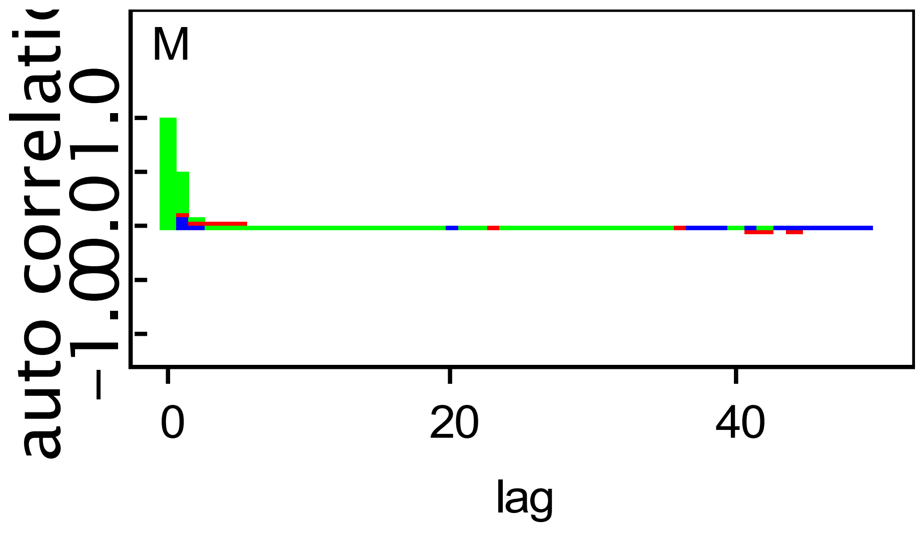


Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS

Third, click on the "auto cor" button to look at the within-chain autocorrelation. (The colours correspond to the individual chains here.)

Ideally, autocorrelation should be noticeable only for a lag of 1, indicating that the chain moves randomly from one iteration to the next.



In this case the sampler on the "green" chain performed slightly less well than the others.

But autocorrelation is not apparent in any chain for lags above 3. This plot looks ok.

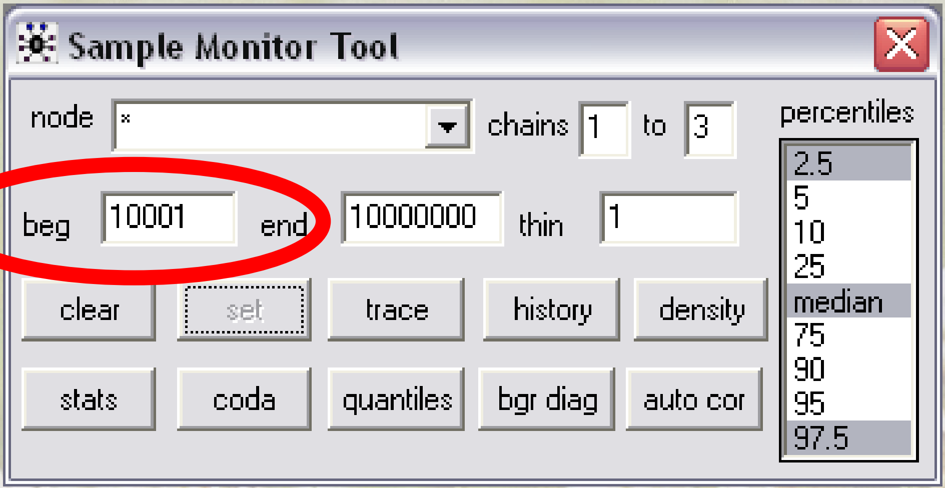
Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Burn-in and convergence

We have looked at 3 basic diagnostics and the posterior sampler appears to be sampling well after 10,000 iterations. There was no sign that the initial values were still influencing the output.

So if we delete these first 10,000 draws as "**burn-in**", we should get good draws from now on.

Write `10,001` into the ***beg*** field on the "Sample monitor tool". Then use the "Updater" to generate the 10,000 samples that we will use.


Sample Monitor Tool

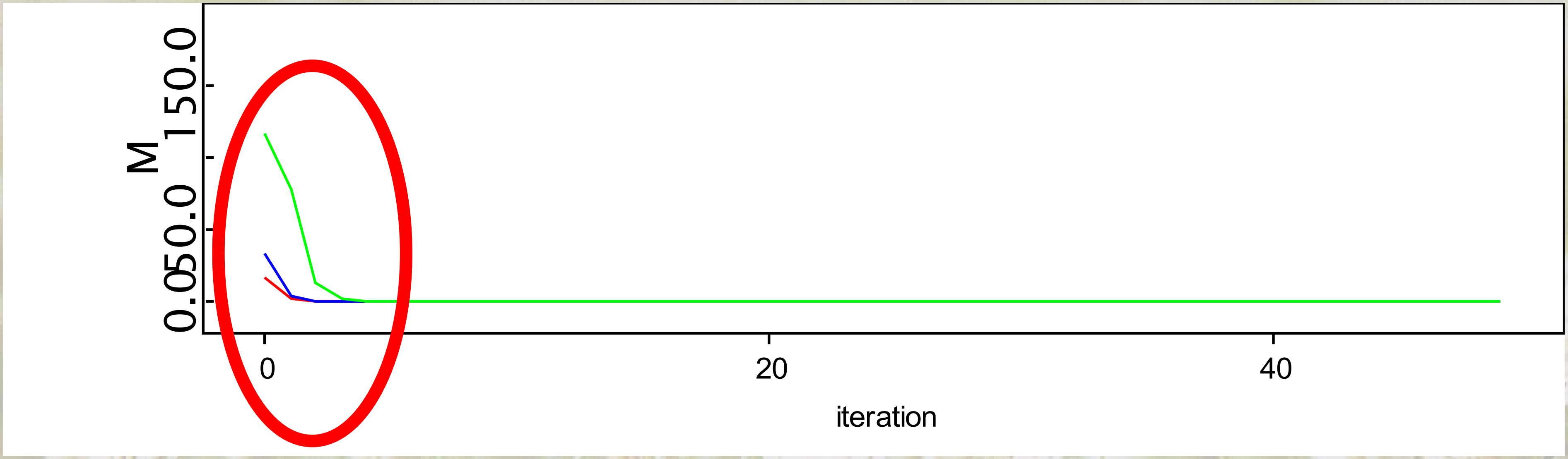
node *
chains 1 to 3
beg 10001 end 10000000 thin 1
percentiles: 2.5, 5, 10, 25, median, 75, 90, 95, 97.5

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics

How many draws to discard?

In this simple example, it is probably a waste of computing time to throw away 10,000 draws for burn-in. Actually it took around 5 iterations to move away from the initial values.

But we checked the statistics after 10,000 draws so we are safe only if we discard all of these for burn-in.



Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

How many draws to generate?

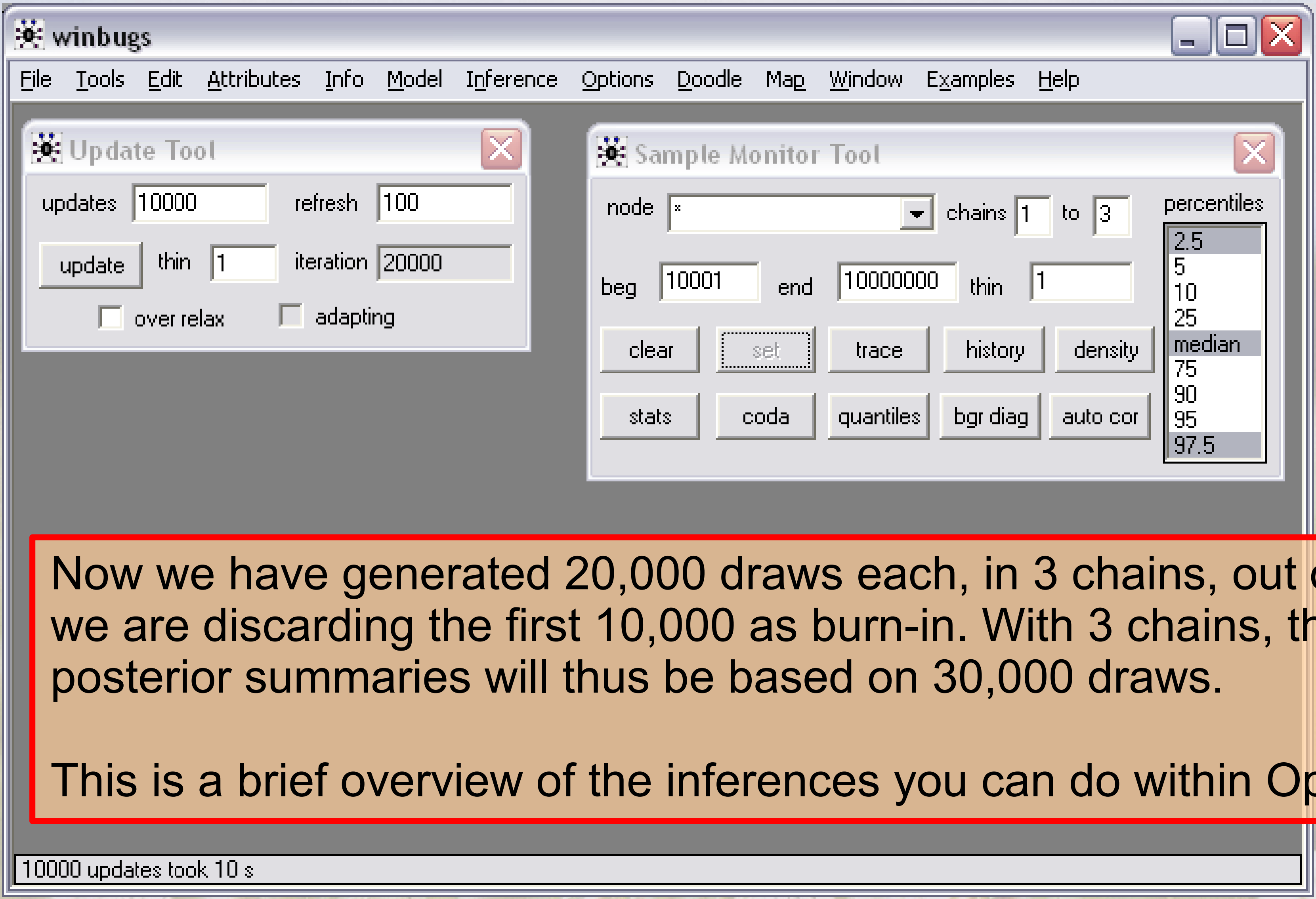
This depends on your application. People often use 10,000 draws but there is no magic number.

When you generate more draws, the resulting means, standard deviations, credibility intervals etc. will be more accurate. Histograms and other graphs will appear smoother.

Running 3 chains, you actually end up with 3*10,000 draws.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS



updates 10000 refresh 100
thin 1 iteration 20000
over relax adapting

node * chains 1 to 3
beg 10001 end 10000000 thin 1

10000 updates took 10 s

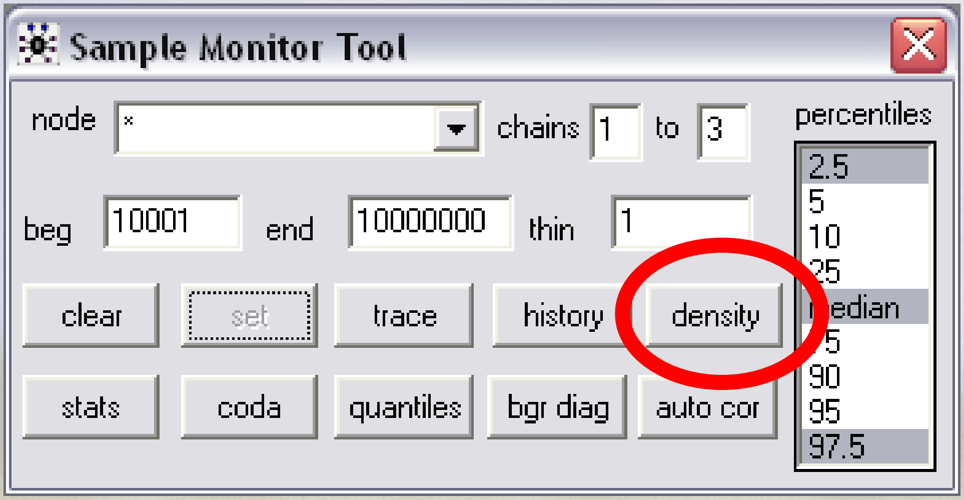
Now we have generated 20,000 draws each, in 3 chains, out of which we are discarding the first 10,000 as burn-in. With 3 chains, the posterior summaries will thus be based on 30,000 draws.

This is a brief overview of the inferences you can do within OpenBUGS.

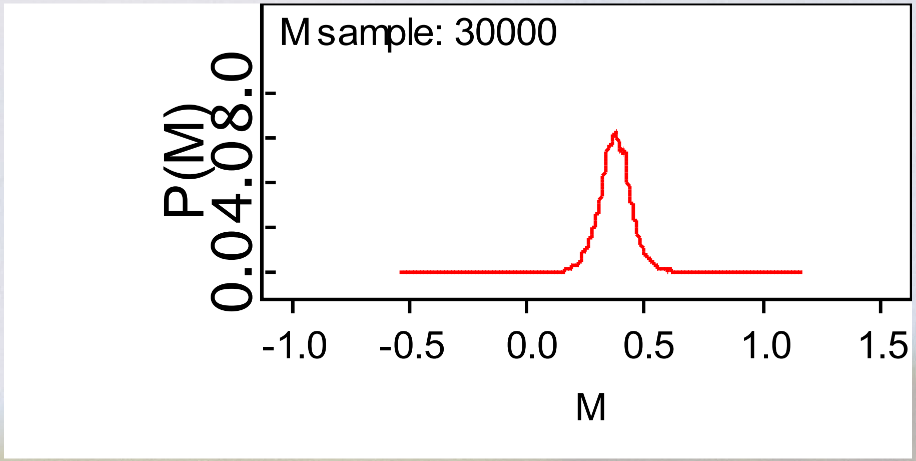
Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



node: *, chains 1 to 3, beg 10001, end 10000000, thin 1

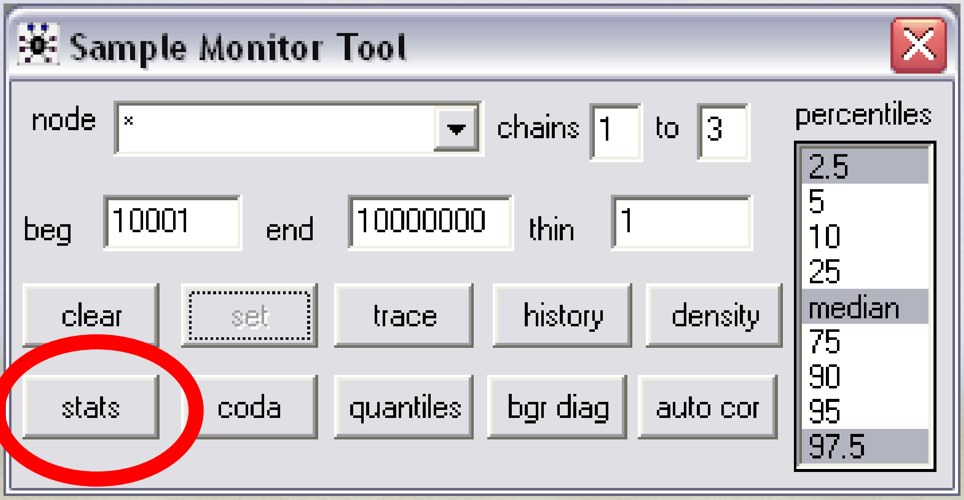


The **"Density"** button shows partial posterior histograms, i.e. the posterior probability distribution for an individual parameter regardless of the values of other parameters.

As with all posterior summaries, if higher accuracy is desired, generate more draws. You should generate more draws if the posterior density looks "rugged" or has unexplainable spikes.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS



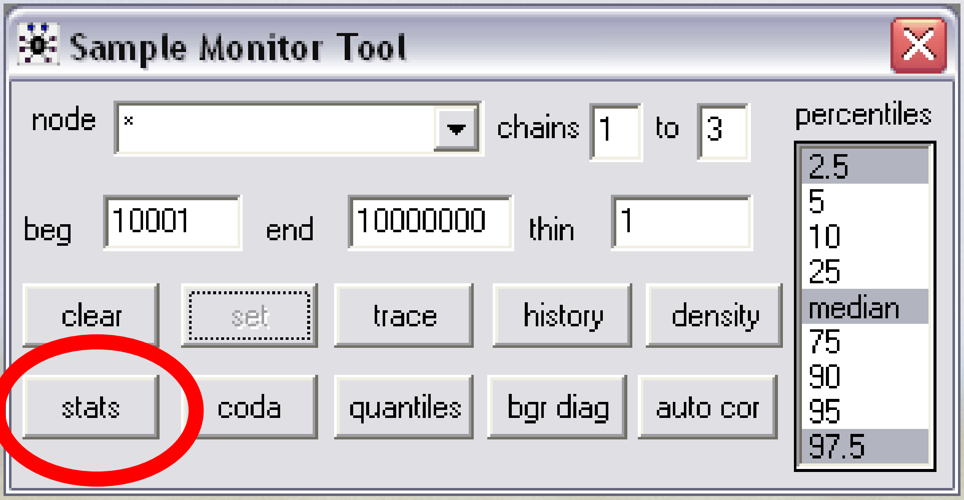
The “**Stats**” button creates a table with posterior mean, standard deviation, MC error, 95%-credibility interval and median.

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
M	0.04836	0.08293	0.002087	-0.1076	0.04576	0.2129	10001	30000
T	0.7127	0.1249	0.00317	0.481	0.7105	0.9601	10001	30000

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



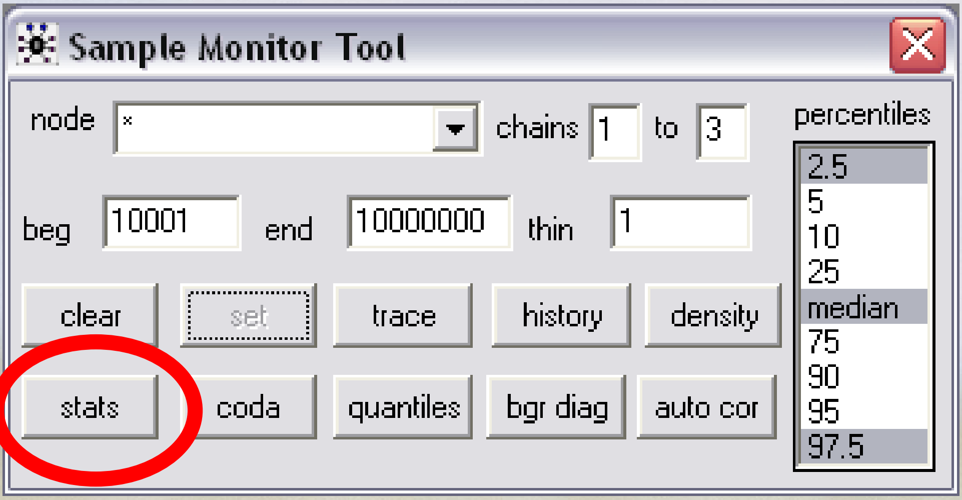
The posterior mean and standard deviation.
Note that you can compare these to the prior mean and standard deviation – this gives you an idea of the information content of your likelihood function, compared to the prior.

The **"Stats"** button creates a table with posterior mean, standard deviation, MC error, 95%-credibility interval and median.

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
M	0.04836	0.08293	0.002087	-0.1076	0.04576	0.2129	10001	30000
T	0.7127	0.1249	0.00317	0.481	0.7105	0.9601	10001	30000

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS



node *
chains 1 to 3
beg 10001 end 10000000 thin 1

The MC error is an estimate of how much of the variation in the posterior sample is due to the noise generated in the sampler.

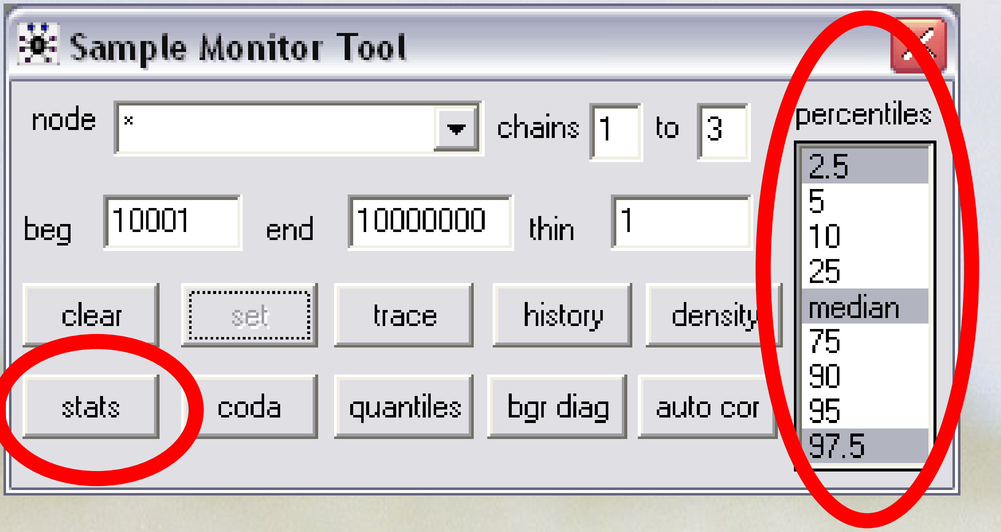
Its value should be very small relative to the sd.

The "**Stats**" button creates a table with posterior mean, standard deviation, MC error, 95%-credibility interval and median.

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
M	0.04836	0.08293	0.002087	-0.1076	0.04576	0.2129	10001	30000
T	0.7127	0.1249	0.00317	0.481	0.7105	0.9601	10001	30000

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS



The posterior median and 95% credibility interval

To change the percentiles you can use the controls in the "Sample Monitor Tool" before you click on the "Stats" button.

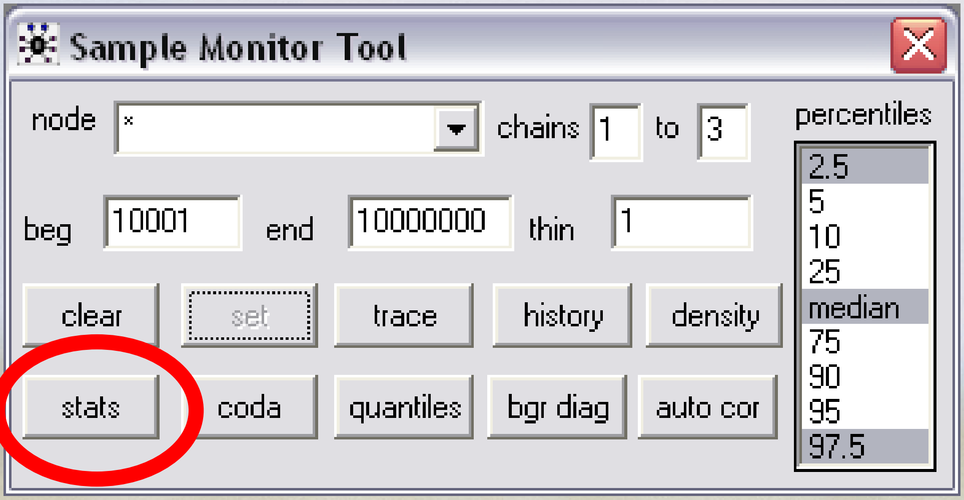
The **"Stats"** button creates a table with posterior mean, standard deviation, MC error, 95%-credibility interval and median.

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
M	0.04836	0.08293	0.002087	-0.1076	0.04576	0.2129	10001	30000
T	0.7127	0.1249	0.00317	0.481	0.7105	0.9601	10001	30000

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

CHE Centre For Health Economics

Practical 1: Getting started in OpenBUGS



Some information on the number of draws that went into this table.

The “**Stats**” button creates a table with posterior mean, standard deviation, MC error, 95%-credibility interval and median.

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
M	0.04836	0.08293	0.002087	-0.1076	0.04576	0.2129	10001	30000
T	0.7127	0.1249	0.00317	0.481	0.7105	0.9601	10001	30000

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

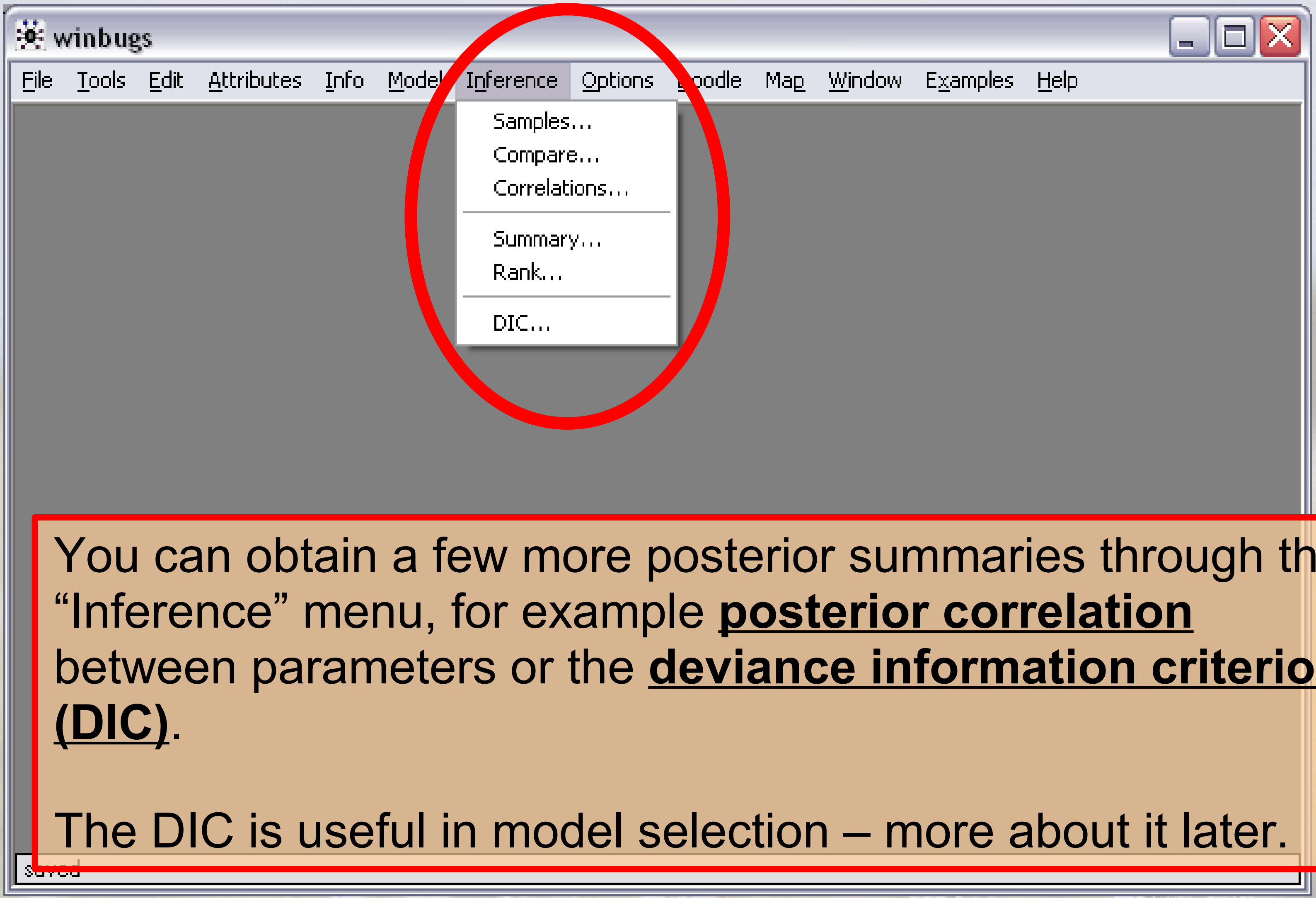
CHE Centre For Health Economics

Some comments

- It is instructive to compare the posterior means and standard deviations to your priors – to see whether, through the likelihood function, the data have contributed information to all parameters.
- In this case we notice no abnomalies in the posterior densities or summaries. If for example you had found signs of a bimodal posterior distribution or unreasonably wide credible intervals, you should rethink your model design.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Practical 1: Getting started in OpenBUGS



You can obtain a few more posterior summaries through the "Inference" menu, for example **posterior correlation** between parameters or the **deviance information criterion (DIC)**.

The DIC is useful in model selection – more about it later.

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk

Summary

- Now you have entered and run your first OpenBUGS model. You know how to
  - code a model in OpenBUGS language
  - load data
  - run the numerical sampling algorithm
  - check for convergence and discard burn-in iterations
  - produce a few useful posterior summaries
- If you want to practice more examples on your own, check out the **“Examples”** menu!

Dr Christian Asseburg
University of York, UK ca505@york.ac.uk
CHE Centre For Health Economics