R与WinBUGS

---贝叶斯统计分析

汤银才
(yctang@stat.ecnu.edu.cn)

(华东师范大学 金融与统计学院)
Outline

1. Statistics and Statistical Inference
2. Statistical Softwares (packages or toolboxes)
3. What is R and why R?
4. Bayesian Inference and BUGS
5. WinBUGS and related packages
6. A start example using DoodleBUGS
7. Case Study --- Gamma-Poisson hierarchical model
8. R2WinBUGS: WinBUGS run in R
9. Summary
10. References
Statistics is the science that relates data to specific questions of interest. This includes devising
- methods to gather data relevant to the question,
- methods to summarize and display the data to shed light on the question,
- methods that enable us to draw answers to the question that are supported by the data.

Different from other disciplines: data almost always contain uncertainty.

Statistics is a science about uncertainty.
● **Statistical inference** gives us **methods and tools** for data analysis.
  - there is a **probability model** explaining how the uncertainty gets into the data.
  - **data** are generated in accordance with some unknown probability distribution
  - By analyzing the data, we attempt
    - to learn about the unknown distribution,
    - To make some inferences about certain properties of the distribution, and
    - to determine the relative likelihood that each possible distribution is actually the correct one.
Some well-known statistics/math softwares

- SAS
- SPSS
- STATA
- Minitab
- Matlab (a toolbox)
- S-Plus

All of them are NOT free
- Free statistics/math softwares
  - Maxima vs. maple/mathematica
  - Octave/scilab vs. Matlab
  - R vs. S-plus

- My points of view:
  Free does not always mean “not safe or useful”
  - Maxima, Scilab and R are popular and reliable softwares!
  - Linux are better than Windows
  - (La)TeX are more popular and better than word/Power Point.
What is R and why R?

R first appeared in 1996, when the statistics professors Robert Gentleman, left, and Ross Ihaka released the code as a free software package (under GNU General Public License).
R is a **language** and **environment**

R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It includes

- an effective data handling and storage facility,
- a suite of operators for calculations on arrays/matrices,
- a large, coherent, integrated collection of intermediate **tools for data analysis**, 
- **graphical facilities** for data analysis and display
- a well-developed, **simple and effective programming language** which includes conditionals, loops, user-defined recursive functions and input and output facilities.

*From* [http://www.r-project.org/about.html](http://www.r-project.org/about.html)
Why R?

- Data Analysts Captivated by R’s Power (Ashlee Vance, The New York Times, Jan 6, 2009) give the answer:
  - R is an open-source program, and its popularity reflects a shift in the type of software used inside corporations.
  - … statisticians, engineers and scientists without computer programming skills find it easy to use.
  - They can improve the software’s code or write variations for specific tasks.
  - “The great beauty of R is that you can modify it to do all sorts of things,” said Hal Varian, chief economist at Google.

Now R is surpassing what Mr. Chambers had imagined possible with S.

“R has really become the second language for people coming out of grad school now, and there’s an amazing amount of code being written for it,” said Max Kuhn, associate director of nonclinical statistics at Pfizer.

while SAS plays down R’s corporate appeal, companies like Google and Pfizer say they use the software for just about anything they can

“R is a real demonstration of the power of collaboration, and I don’t think you could construct something like this any other way,” Mr. Ihaka said. “We could have chosen to be commercial, and we would have sold five copies of the software.”
Bayesian Inference/Modeling

- **Main Approaches to Statistics**
  - Frequentist/classical approach, which is based on:
    - **Parameters**, the numerical characteristics of the population, are **fixed but unknown** constants.
    - Probabilities are always interpreted as long run relative frequency.
    - Statistical procedures are judged by how well they perform in the long run over an infinite number of hypothetical repetitions of the experiment.
- Bayesain approach, the ideas of which are:
  - Parameters are considered as random.
  - Probability statements about parameters must be interpreted as "degree of belief." The prior distribution must be subjective.
  - We revise our beliefs about parameters after getting the data by using Bayes' theorem.
  - The posterior distribution comes from two sources: the prior distribution and the observed data.

- The Bayesian approach forms the basis of science -- learning process: As we get more data, we add to our store of information by multiplying it by our current posterior distribution.
Notations:

- $D = \{y_1, y_2, \ldots, y_n\}$: data from sampling distribution $f(x \mid \theta)$

- $L(\theta \mid D) = \prod_{i=1}^{n} f(y_i \mid \theta)$: Likelihood---sampling information

- $\theta$: parameter of interest

- $\pi(\theta)$: prior distribution of $\theta$ --- prior information

- $\pi(\theta \mid D) \propto \pi(\theta)L(\theta \mid D)$: posterior distribution
Prior info+sampling info=Posterior info

Bayes Theorem:

\[ \pi(\theta | x) = \frac{\pi(\theta) f(x | \theta)}{\int \pi(\theta) f(x | \theta) d\theta} \propto \pi(\theta) f(x | \theta) \]

Fomulation:

\[ \begin{cases} 
\theta \sim \pi(\theta) \\
y_1, y_2, \ldots, y_n \ iid \sim f(y | \theta) 
\end{cases} \Rightarrow \theta | x \sim \pi(\theta | x) \]

Bayesian Model Specification:

1) prior distribution (often conjugate)
2) sampling distribution
Examples:

1) Beta-Binomial model for proportion:
\[
\begin{align*}
\theta &\sim \text{Beta}(\alpha, \beta) \\
y | \theta &\sim \text{Bin}(n, \theta)
\end{align*}
\]
\[
\Rightarrow \theta | y \sim \text{Beta}(\alpha + y, n - y + \beta)
\]

2) Normal-normal model for mean (given variance)
\[
\begin{align*}
\theta &\sim \mathcal{N}(\mu, \tau^2) \\
y_1, \ldots, y_n | \theta &\sim \mathcal{N}(\theta, \sigma^2)
\end{align*}
\]
\[
\Rightarrow \theta | y \sim \mathcal{N}(\mu_n, \sigma_n^2), \text{ where}
\]
\[
\mu_n = \frac{1}{\sigma^2 / n + \tau^2} \bar{y} + \frac{1}{\tau^2} \mu,
\]
\[
1/\sigma_n^2 = \frac{1}{\sigma^2 / n} + \frac{1}{\tau^2}
\]
What is BUGS/WinBUGS?

Logo of BUGS
What is BUGS?

- BUGS (Bayesian inference Using Gibbs Sampling) is a popular software for analyzing complex statistical models using Markov Chain Monte Carlo (MCMC) methods.

- It uses
  - Gibbs sampling
  - Metropolis algorithm to generate a Markov chain by sampling from full conditional distributions.
- **History**([http://www.mrc-bsu.cam.ac.uk/bugs/](http://www.mrc-bsu.cam.ac.uk/bugs/))
  - a command-line interface. Not further developed since 1996.
  - Originate at the MRC BIOSTATISTICAL UNIT in Cambridge
  - Later developed into **WinBUGS**, jointly with the Imperial College School of Medicine at St Mary’s, London.
  - **OpenBUGS**, completely revised version of WinBugs, developed in the University of Helsinki, Finland.
  - Add-on or interfaces: **GeoBUGS**, **PKBUGS**,......
BUGS

- Classic BUGS
- WinBUGS (Windows Version)
- OpenBUGS (Windows Version)
- GeoBUGS (spatial models)
- PKBUGS (pharmacokinetic modeling)
- DoodleBUGS (graphical modeling)
What is WinBUGS?

- **WinBUGS**, a windows program with an option of a graphical user interface, the standard ‘point-and-click’ windows interface, and on-line monitoring and convergence diagnostics. It also supports Batch-mode running (version 1.4).

- **GeoBUGS**, an add-on to WinBUGS that fits spatial models and produces a range of maps as output.

- **PKBUGS**, an efficient and user-friendly interface for specifying complex population pharmacokinetic pharmacokinetic and pharmacodynamic (PK/PD) models within WinBUGS software.
Why use WinBUGS?

- Its **ability** to fit complex statistical models using MCMC methods.
- Its **flexibility** to program, two different ways to specify model
  - DoodleBUGS: *Direct graphics*
  - *BUGS language*
- **Free** to download: http://www.mrc-bsu.cam.ac.uk/bugs.
- A lot of **examples** in WinBUGS
- A lot of **online resources**
  http://www.mrc-bsu.cam.ac.uk/bugs/weblinks/webresource.shtml
Related packages

- Package needs to be downloaded
  - WinBUGS14.exe

- Potential useful packages
  - CODA (Convergence Diagnostic and Output Analysis)
  - BOA (Bayesian Output Analysis)
  - JAGS (Just Another Gibbs Sampler)---for analysis of Bayesian hierarchical models.
  - R2WinBUGS---A package for Running WinBUGS from R
Working with BUGS

Prepare Data
- Editor Spread sheet

BUGS Analysis
- WinBUGS

Output Analysis
- R/Splus STATA
How to use WinBUGS

- **Step 1**: Specify the model to run
- **Step 2**: Load the data and initial values for a specified number of chains
- **Step 3**: Run WinBugs to get the Markov chains and save the results for the parameters for later use.
unemployed proportion $p$ in the population?

- $p$ is in $[0, 1]$, where 0 means no one is unemployed and 1 means everyone is.
- prior information of $p$ (from newspaper reports, economic theory, previous surveys, etc.) can be used.
- choose a beta prior distribution restricted between 0.1 and 0.6.
- Data: $n = 14$ people were surveyed and $r = 4$ of them were unemployed.
Model Specification in WinBUGS

- Method 1: directly write a WinBUGS document (in BUGS language)
- Method 2: Draw a directed graph in DoodleBUGS and then transfer into a WinBUGS document.
  - Start WinBUGS
  - Select “Doodle” from menu bar
  - Select “New…”
  - Press “ok”
  - You have a window to “Doodle” in.
  - Creating a node
    - **Mouse click** in Doodle Window
- A node has a name and type along with other characteristics and parameters depending on its type.

- Delete a node:
  - highlight it
  - CTRL + Del
Node Types
- Stochastic
- Logical
- Constant (rectangle)

Create node $p$, click on and type/choose
- Name: $p$
- Type: stochastic
- Density: $\text{dbeta}$
- $A:1; b:1$ (beta parameters)
- lower bound: $0.1$; upper bound: $0.6$
WinBUGS document

- Two ways to transfer into a WinBUGS document
  1. Menu bar > Doodle > Write Code.
     - highlight the code > Click on Attributes > 16 point.
     - ~ is read “distributed as”

```plaintext
model;
{
  p ~ dbeta(1,1)( 0.1, 0.6)
}
```
2. Select/ Copy/Paste your doodle into WinBUGS new document (still a doodle!)

- Menu bar > File > New
- Menu bar > Edit > Copy
- Menu bar > Edit > Paste
Running BUGS

- Menu bar > model > Specification…
- use WinBUGS to look at samples from our prior

- **Check Model**
  - Select the Doodle (note the hairy boarder)
  - Menu bar - Model - Check model
  - Note the message in bottom left hand corner

- **Compiling the Model**
  - Menu bar - Model - Compile
  - Bottom left hand corner
- Load Initial Values
  - Menu bar - Model - Gen inits
  - Bottom left hand side

  ![initial values generated]

- Before produce result, select Options > Output options > Select the log radio button.

![Output options window]
- Update the Model
  - Menu bar - Model – Update
  - 1000 MCMC updates to be carried out.
  - Note:
    - Model has been updated
    - MCMC run did not store any data---useful for “burn-in”
    - Store values by “monitoring” them to
      - Draw inferences
      - Monitor MCMC run
- Monitoring Nodes
  - Monitoring p, our parameter of interest
  - Menu bar - Inference - Samples...
  - Type name of node “p” to monitor
  - Press “set”
- Update & Monitor
  - Update model again
  - 1000 values “monitored” of the MCMC run for p
WinBUGS output--1

- **Summary Statistics**
  - Select “p” from the Sample Monitor Tool
  - Press “stats” (Sample Monitor Tool)

- **MCMC Time Series**
  - Press “History” in Sample Monitor Tool

- **Kernel Density**
  - Press “Density” in the Sample Monitor Tool
Output results in log file
Inference: combining prior with data

- For unemployment example, \( n = 14 \) people were surveyed and \( r = 4 \) of them were unemployed.

- Bayesian Model: Beta – Binomial

\[
\begin{align*}
&\left\{ \begin{array}{l}
p \sim \text{Beta}(a,b) \\
r \mid p \sim \text{Bin}(N, p)
\end{array} \right.
\end{align*}
\]

- \( N=14 \) (order), \( a=1, b=1 \). Change \( p \) to be restricted between 0.2 and 0.45.
- New doodle/directed graph
  - Create a stochastic node $p$ in (0.2, 0.45) (done!)
  - Add a node $N$
    - name: $N$
    - type: constant
  - Add a node $r$
    - Name: $r$
    - type: stochastic
    - density: $\text{dbin}\text{(binomial)}$, with proportion $p$ and order $N$. 
  - add links between the nodes:
    - click on node $r$ (highlighted),
    - With the Ctrl key held, click on node $p$ and node $N$. 
- Two relationships in a noodle
  - Single arrows: stochastic dependencies/statistical relationship (is distributed as)
  - Hollow arrows: logical relationship

Beta-binomial doodle/directed graph
- To see the code
  - Select Doodle > Write Code
- To enter the data, in the model code window
  list(N=14, r=4)
Running BUGS again!

- Select Model > Specification..., do
  
  - WinBUGS calculate the appropriate likelihood function and prior for $p$, and combine them into a posterior distribution. (straightforward for conjugate)

- Select Model > Update..., change updates to 5000
  
  - WinBUGS generates updated samples of $p$ (from the initial) by combining the prior information on $p$ and the new information on $p$ given by the data $r$ and $N$. 
WinBUGS output--2

(N=14, r=4)

Node statistics

<table>
<thead>
<tr>
<th>node</th>
<th>mean</th>
<th>sd</th>
<th>MC error</th>
<th>2.5%</th>
<th>median</th>
<th>97.5%</th>
<th>start</th>
<th>sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>0.3142</td>
<td>0.06715</td>
<td>7.423E-4</td>
<td>0.2069</td>
<td>0.3099</td>
<td>0.4391</td>
<td>5001</td>
<td>5000</td>
</tr>
</tbody>
</table>

Kernel density

p sample: 5000

(N=140, r=40)

Node statistics

<table>
<thead>
<tr>
<th>node</th>
<th>mean</th>
<th>sd</th>
<th>MC error</th>
<th>2.5%</th>
<th>median</th>
<th>97.5%</th>
<th>start</th>
<th>sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>0.289</td>
<td>0.03756</td>
<td>5.556E-4</td>
<td>0.2198</td>
<td>0.2874</td>
<td>0.3671</td>
<td>5001</td>
<td>5000</td>
</tr>
</tbody>
</table>

Kernel density

p sample: 5000

N=14, r=4

N=140, r=40
2nd example: two props comparison

- Two surveys are taken:
  - N1=14, r1=4
  - N2=12, r2=5

- Question: What is the evidence that the underlying rates for the two surveys is really different?

- Naïve thinking: 4/14=29%, 5/12=42% -> the difference appears to be large.

- Bayesian analysis using WinBUGS: calculate a posterior for the difference or ratio
Create two sets of nodes as in 1st example
create two logical nodes

1. **type:** logical  
   **name:** ratio  
   **link:** identity  
   **value:** p2/p1

2. **type:** logical  
   **name:** difference  
   **link:** identity  
   **value:** p2-p1.
Add the links to finish the model.
corresponding code for the two proportion model: Select Doodle > Write Code

```r
model;
{
  p1 ~ dbeta(1,1)( 0.2,0.45)
  r1 ~ dbin(p1,N1)
  r2 ~ dbin(p2,N2)
  p2 ~ dbeta(1,1)( 0.2,0.45)
  ratio <- p2 / p1
  difference <- p2 - p1
}
```
- Summary: node connectors:

<table>
<thead>
<tr>
<th>connector</th>
<th>Solid arrow</th>
<th>Hollow arrow</th>
</tr>
</thead>
<tbody>
<tr>
<td>meaning</td>
<td>Stochastic/statistical</td>
<td>Logical/deterministic</td>
</tr>
<tr>
<td>relationship</td>
<td>relationship</td>
<td></td>
</tr>
<tr>
<td>Sign in BUGS</td>
<td>&lt;-</td>
<td>~</td>
</tr>
<tr>
<td>language</td>
<td>diff&lt;-p2-p1</td>
<td>r~dbin(p,N)</td>
</tr>
<tr>
<td>example</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Add the data before analyzing the model
  - In the model window type:
    list(N1=14, r1=4, N2=12, r2=5)
Running BUGS again!

- Select Model > Specification..., do
- Sample Monitor Tool to set ratio and difference
- Select Model > Update..., change updates to 5000

check model → load data → compile → gen inits
WinBUGS output--3

### Node statistics

<table>
<thead>
<tr>
<th>node</th>
<th>mean</th>
<th>sd</th>
<th>MC error</th>
<th>2.5%</th>
<th>median</th>
<th>97.5%</th>
<th>start</th>
<th>sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>ratio</td>
<td>1.158</td>
<td>0.3359</td>
<td>0.004827</td>
<td>0.6129</td>
<td>1.116</td>
<td>1.915</td>
<td>1001</td>
<td>5000</td>
</tr>
<tr>
<td>difference</td>
<td>0.03322</td>
<td>0.09322</td>
<td>0.00134</td>
<td>-0.1555</td>
<td>0.03619</td>
<td>0.2017</td>
<td>1001</td>
<td>5000</td>
</tr>
</tbody>
</table>

### Kernel density

- **Ratio sample:** 5000
- **Difference sample:** 5000
The sample rates of unemployment are **NOT significantly different**. Reasons:

- The area under the difference curve to the right of zero is not much larger than the area to the left of zero. Also the area to the right of one under the ratio curve is not much larger than the area to the left of one.
- the mean difference is close to zero and the mean ratio is close to one
- 95% credible interval for difference (-0.16, 0.20)
- 95% credible interval for ratio (0.61, 1.92)
More complex models with plates

- **Plates**
  - Allow more complex structure, e.g.,
    - Repeated measures
    - Hierarchical models

- **Creating a plate**
  - CTRL + mouse click in Doodle Window
  - Deleting a plate: CTRL + Del
Case Study: pumps data

<table>
<thead>
<tr>
<th>PUMP</th>
<th>tᵢ</th>
<th>xᵢ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>94.5</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>15.7</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>62.9</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>126</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>5.24</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>31.4</td>
<td>19</td>
</tr>
<tr>
<td>7</td>
<td>1.05</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1.05</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>2.1</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>10.5</td>
<td>22</td>
</tr>
</tbody>
</table>

- tᵢ: the length of operation time of the pump (in 1000s of hours).
- xᵢ: the number of failures
Gamma-Poisson hierarchical model

for \( i = 1, 2, \ldots, 10 \)

\[
x_i \sim \text{Poisson}(t_i \theta_i)
\]

\[
\theta_i \sim \text{Gamma}(\alpha, \beta)
\]

\[
\alpha \sim \text{Exponential}(1.0)
\]

\[
\beta \sim \text{Gamma}(0.1, 1.0)
\]

Paramters of interest: \( \theta_i, i = 1, 2, \ldots, 10 \)
Model specification through DoodleBUGS

- **Nodes**
  - Constants, denoted by rectangles
  - Stochastic nodes, denoted by ellipses
  - Deterministic nodes, logical function of other nodes

- **Edges**
  - Directed links
    - Solid arrow: stochastic dependence
    - Hollow arrow: logical function
  - Undirected links, dashed line
    - Representing an upper or lower bound

- **Plates**, repeated parts of the graph
Stochastic node
Constant node
Deterministic node
Plate
Stochastic relationship
logical relationship
Model specification through BUGS language

model
{
  for (i in 1 : N) {
    theta[i] ~ dgamma(alpha, beta)
    lambda[i] <- theta[i] * t[i]
    x[i] ~ dpois(lambda[i])
  }
  alpha ~ dexp(1)
  beta ~ dgamma(0.1, 1.0)
}
Format data and specify initial values

- Data format
  - R/S-Plus format using list()
    \[
    \text{list}(t = c(94.3, 15.7, 62.9, 126, 5.24, 31.4, 1.05, 1.05, 2.1, 10.5),
    \text{x} = c(5, 1, 5, 14, 3, 19, 1, 1, 4, 22),
    N = 10)
    \]
  - Rectangular format

- Initial values
  \[
  \text{list}(\alpha = 1, \beta = 1)
  \]
  \[
  \text{list}(\alpha = 10, \beta = 10)
  \]
- Data in rectangular format
  - Data are usually stored in the same **odc** file with END sign
  - To conserve space, hide/fold data in rectangular format by
    - **Tools > Create Fold**
    - **Edit menu > select > copy > paste between two arrows**
    - A name can be inserted in between
  - Load the data by highlight the first row
  - Repeat the procedure for multiple sets of data

<table>
<thead>
<tr>
<th>t</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>94.3</td>
<td>5</td>
</tr>
<tr>
<td>15.7</td>
<td>1</td>
</tr>
<tr>
<td>62.9</td>
<td>5</td>
</tr>
<tr>
<td>126</td>
<td>14</td>
</tr>
<tr>
<td>5.24</td>
<td>3</td>
</tr>
<tr>
<td>31.4</td>
<td>19</td>
</tr>
<tr>
<td>1.05</td>
<td>1</td>
</tr>
<tr>
<td>1.05</td>
<td>1</td>
</tr>
<tr>
<td>2.1</td>
<td>4</td>
</tr>
<tr>
<td>10.5</td>
<td>22</td>
</tr>
</tbody>
</table>

END
```r
model;
{
  for( i in 1 : N ){
    x[i] ~ dpois(lambda[i])
    lambda[i] <- theta[i] * t[i]
    theta[i] ~ dgamma(alpha,beta)
  }
  alpha ~ dexp(1)
  beta ~ dgamma(1,1)
}

data
list(N = 10)
Data(t,x)

Initial values
list(alpha = 1, beta = 1)
list(alpha = 10, beta = 10)
```
Anything missing?

- What kind of output can WinBUGS give us?
- What kind of sampling method does WinBUGS implement?
Output of WinBUGS

- **Samples at Inference Menu**
  - **trace**: plots the variable value against iteration number.
  - **history**: plots out a complete trace for the variable.
  - **density**: smoothed kernel density estimate for continuous variable or a histogram for discrete variable.
  - **auto cor**: auto correlation, up to lag-50
  - **stats**: Summary statistics, pooling over the chains selected.
  - **coda**: output the monitored values to CODA or BOA
quantiles: plots out the running mean with running 95% confidence intervals against iteration number.


- **Green**: the width of the central 80% interval of the pooled runs
- **Blue**: the average width of the 80% intervals within the individual runs
- **Red**: their ratio $R (= \text{pooled} / \text{within})$.
- Convergence: $R$ to 1, and with convergence of both the pooled and within interval widths to stability.
• Compare at Inference Menu

Box Plot

Caterpillar Plot
• **Rank at Inference Menu**
  - **stats**: the distribution of the ranks of each component of the variable.
  - **histogram**: the empirical distribution of the simulated rank for each component

• **DIC at Inference Menu**
  - **Deviance Information Criterion**
sampling methods in WinBUGS

Sampling methods are used in the following hierarchies

• **Continuous target distribution**
  - Conjugate: Direct sampling using standard algorithms
  - Log-concave: Derivative-free adaptive rejection sampling
  - Non-log-concave (restricted range): Slice sampling
  - Non-log-concave (unrestricted range): Metropolis

• **Discrete target distribution**
  - Finite upper bound: Inversion
  - Shifted Poisson: Direct sampling using standard algorithm
**R2WinBUGS**: WinBUGS run in R

- It is a R package that provides the tools/functions to call WinBUGS directly.
- It automatically writes the data and scripts in a format readable by WinBUGS for processing in batch mode.
- Then it is possible to read resulting data into R to give
  - summary statistics
  - graphical summary
  - convergence diagnostics
  - ……
How to install R2WinBUGS

Suppose that the internet connection is available.

- Start R

- First way: at R command prompt type

  ```
  install.packages("R2WinBUGS")
  ```

- Second way: install from R menu “packages”
How to use R2WinBUGS

We recommend to use R2WinEdt as a script editor and use school data as an example---to fit the hierarchical normal model.

- Step 1: Start R
- Step 2: Start WinBUGS
- Step 3: load R2WinBUGS from R by typing
  ```
  library("R2WinBUGS") # or Before using bugs()
  load RWinEdt from R by typing
  library("RWinEdt")
  ```

Remark 1: We use the school data in Section 5.5 of "Bayesian Data Analysis(Gelman, 2004)" to illustrate
- Step 4: **Model specification** --- establish the **Bugs** model file: `schools.bug`
Remark 2: In Bugs, the normal distribution is parameterized by its precision (=1/variance)

Remark 3: For convenience, use $\tau$ for precisions and $\sigma$ for standard deviations.
- Step 5: **data specification** (school.dat)

<table>
<thead>
<tr>
<th>school estimate</th>
<th>sd</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>28.39</td>
</tr>
<tr>
<td>B</td>
<td>7.94</td>
</tr>
<tr>
<td>C</td>
<td>-2.75</td>
</tr>
<tr>
<td>D</td>
<td>6.82</td>
</tr>
<tr>
<td>E</td>
<td>-0.64</td>
</tr>
<tr>
<td>F</td>
<td>0.63</td>
</tr>
<tr>
<td>G</td>
<td>18.01</td>
</tr>
<tr>
<td>H</td>
<td>12.16</td>
</tr>
</tbody>
</table>
- Step 6: Data read, starting values and parameters specification (R_school_full.R)

```r
schools <- read.table("c:/bugs.R/schools.dat", head=T)
J <- nrow(schools)
y <- schools$estimate
sigma.y <- schools$sd
data <- list("J","y","sigma.y")

inits <- function()
  list(theta=rnorm(J, 0, 100),
       mu.theta=rnorm(1, 0, 100),
       sigma.theta=runif(1, 0, 100))

parameters <- c("theta", "mu.theta", "sigma.theta")
```
Remark 5: Bugs does not require all parameters to be initialized, but it is a good idea to do so.

Starting values can be constructed
- randomly (as in the example above)
- with simple initial values
- from crude estimates
Step 7: Run Bugs (MCMC simulation) with given number of chains for given number of iterations.

```r
schools.sim <- bugs(data, inits, parameters, model.file="c:/bugs.R/schools.bug", n.chains=3, n.iter=1000, bugs.directory = "E:/WinBUGS14/")
```

Remark 6:
- `n.chains=` specifies the number of chains
- `n.iter=` specifies the number of iterations

Remark 7: While Bugs is running, it opens a new window and freezes R.
Step 8: Show the results

- 8.1 Numerical output/summaries of the simulations for the parameters
  
  ```r
  print(schools.sim)
  schools.sim$summary
  ```

- 8.2 Graphical summaries of the inferences and convergence.
  
  ```r
  plot(schools.sim)
  ```

Remark 8:

- n.eff: rough measure of effective sample size
- Rhat: potential scale reduction factor
- pD: effective number of parameters
- DIC: estimate of expected predictive error
> print(schools.sim)

Inference for Bugs model at "c:/bugsR/schools.txt", fit using WinBUGS,
3 chains, each with 1000 iterations (first 500 discarded)
n.sims = 1500 iterations saved

               mean   sd 2.5%  25%  50%  75% 97.5% Rhat n.eff
theta[1]  10.6  7.5 -0.7  5.5  9.4 14.1 28.9    1  170
theta[2]   7.7  5.9 -3.6  3.8  7.7 11.3 19.6    1  400
theta[3]   6.4  6.6 -9.2  2.8  6.6 10.0 18.8    1 1500
theta[4]   7.3  5.9 -4.4  3.5  7.4 11.0 19.6    1  450
theta[5]   5.6  5.7 -7.5  2.5  5.9  9.2 16.2    1 1100
theta[6]   6.1  6.2 -7.1  2.7  6.1  9.9 17.4    1  650
theta[7]   9.6  6.4 -0.1  5.0  8.9 13.3 24.9    1  340
theta[8]   8.4  7.1 -4.4  3.8  8.2 12.2 23.5    1 1000
mu.theta   7.8  4.5  0.0   4.6  7.8 10.7 16.6    1  280
sigma.theta 5.5  5.2  0.2  1.7  4.2  7.8 18.0    1  110
deviance  60.2  1.8  57.1  59.2 60.0 61.2 64.4    1  280

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)
pD = 1.6 and DIC = 61.9
DIC is an estimate of expected predictive error (lower deviance is better).
Bugs model at "c:/bugsR/schools.txt", fit using WinBUGS, 3 chains, each with 1000 iterations (first 500 discarded)

80% interval for each chain
-10 0 10 20 30

R-hat
1 1.5 2+

medians and 80% intervals

theta
-10 0 10 20 30

mu.theta
-10 0 10 20 30

sigma.theta
-10 0 10 20 30

deviance
58 59 60 61 62 63 64
Summary

- **BUGS is a power tool**
  - Bayesian Analysis
  - Simulation Tool
- **Model specification in WinBUGS**
  - Graphical Models
    - DoodleBUGS: Direct graphics
    - Simple representation of model
  - BUGS language
- **WinBUGS run in R (R2WinBUGS)**
  also: in S-plus, Stata, SAS,…
A Reminder

- WinBUGS is Easy to use!

But

- MCMC sampling can be dangerous in WinBUGS!
References

[1] Petra Kuhnert, The Ecology Centre, UQ., An Introduction to WinBUGS 1.4


……