Professor William Browne School of Veterinary Science and Centre for Multilevel Modelling (& many co-workers!)

# The STAT-JR package and it's potential use with social network models









# Summary

- Background to STAT-JR package
- Some screen shots of the program features
- Multiple membership models for spatial models/ social networks
- STAT-JR named in memory of Jon Rasbash whose ideas started project.



## The E-STAT project and STAT-JR

- STAT-JR developed jointly by LEMMA II and E-STAT ESRC nodes
- Consists of a set of components many of which we have an alpha version for which contains:
- Templates for model fitting, data manipulation, input and output controlled via a web browser interface.
- Currently model fitting for 90% of the models that MLwiN can fit in MCMC plus some it can't including greatly sped up REALCOM templates
- Some interoperability with MLwiN, WinBUGS, R, Stata and SPSS (written by Camille Szmaragd)



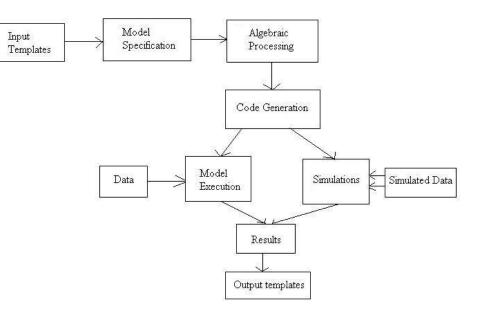
4

Jon identified 3 groups of users:

- Novice practitioners who want to use statistical software that is user friendly and maybe tailored to their discipline
- Advanced practitioners who are the experts in their fields and also want to develop tools for the novice practitioners
- Algorithm Developers who want their algorithms used by practitioners.
- See <u>http://www.cmm.bristol.ac.uk/research/NCESS-</u>
   <u>EStat/news.shtml</u> for details of Advanced User's guide for
   STAT-JR

### STAT-JR component based approach

Below is an early diagram of how we envisioned the system. Here you will see boxes representing components some of which are built into the STAT-JR system. The system is written in Python with currently a VB.net algebra processing system. A team of coders (currently me, Chris, Danius, Camille and Bruce) work together on the system.





#### Templates

Consist of a set of code sections for advanced users to write.

For a model template it consists of at least:

- an invars method which specifies inputs and types
- An outbug method that creates (BUGS like) model code for the algebra system
- An (optional) outlatex method can be used for outputting LaTeX code for the model.
- Other optional functions required for more complex templates

#### **Regression 1 Example**

```
from EStat.Templating import *
from mako.template import Template as
MakoTemplate
import re
```

```
class Regression1(Template):
```

'A model template for fitting 1 level Normal multiple regression model in E-STAT only. To be used in documentation.'

```
tags = [ 'model' , '1-Level' ]
```

```
invars = '''
y = DataVector('response: ')
```

```
tau = ParamScalar()
sigma = ParamScalar()
x = DataMatrix('explanatory variables: ')
beta = ParamVector()
beta.ncols = len(x)
""
```

```
outbug = ""
model{
  for (i in 1:length(${y})) {
     ${y}[i] ~ dnorm(mu[i], tau)
     mu[i] <- ${mmult(x, 'beta', 'i')}
}</pre>
```

```
}
```

```
# Priors
% for i in range(0, x.ncols()):
beta${i} ~ dflat()
% endfor
tau ~ dgamma(0.001000, 0.001000)
sigma <- 1 / sqrt(tau)</pre>
```

```
....
```

outlatex = r'''
\begin{aligned}
\mbox{\${y}}\_i & \sim \mbox{N}(\mu\_i, \sigma^2) \\
\mu\_i & =
\${mmulttex(x, r'\beta', 'i')} \\
%for i in range(0, len(x)):
\beta\_\${i} & \propto 1 \\
%endfor
\tau & \sim \Gamma (0.001,0.001) \\
\sigma^2 & = 1 / \tau
University of
\end{aligned}

```
....
```

#### Invars function

invars = '''
y = DataVector('response: ')
tau = ParamScalar()
sigma = ParamScalar()
x = DataMatrix('explanatory variables: ')
beta = ParamVector()
beta.ncols = len(x)
'''

....



### An example of STAT-JR – setting up a model

Mozilla Firefox		
le Edit View History Bookmarks Too		
	http://localhost:8080/run/	☆~ <mark>る</mark> - Goode
	atest Headlines 📄 Customize Links 📄 Windows Marke	
http://localhost:8080/run/	+	
		Ctat ID Domonatrator
		Stat-JR Demonstrator
Template: Regression1 Ch	hange Dataset: tutorial Change View Sur	mmary
Configuration		
Comgaration		Start again
response:	normexam	
explanatory variables:	cons,standirt	
Name of output results:	out	
Random Seed:	1	
length of burnin:	1000	
number of iterations:	5000	
thinning:	1	
		Next
		Set
		Inputs: {\/; 'normexam', \/; 'cons,standirt'}

University of BRISTOL

### An example of STAT-JR – setting up a model

Mozilla Firefox     Bodimarks     Tools     telp       Image: Started States Velocities     Cutowice Links     Mining Started     Latest Holdines     Cutowice Links     Win	dows Marketplace	
Configuration		nstrator
response:	normexam	
explanatory variables:	cons,standlrt	
Name of output results:	out	
Random Seed:	1	
length of burnin:	1000	
number of iterations:	5000	:lirt'}
thinning:	1	



#### Equations for model and model code

Decilla Firefox	
jie Edit View Higtory Bookmarks Iools Help	JAN MAKIN LEP TH
🔇 💽 🗸 🏠 🗋 http://127.0.0.1:8080/run/	😭 👻 🖓 🗧 amsterdam multi level conference 🖉 🔎
🗖 Most Visited 📄 Getting Started 脑 Latest Headlines 📲 Customize Links 📄 Windows Marketplace	
http://127.0.0.1:8080/run/ +	
thinning: 1	
Equation rendering	
$\mathrm{normexam}_i \sim \mathrm{N}(\mu_i, \sigma^2)$	
$\mu_i = eta_0 \mathrm{cons}_i + eta_1 \mathrm{standlrt}_i + u_{\mathrm{school}_{[i]}}$	
$u_{ ext{school}[i]}  ext{ }  ext{N}(0, \sigma_u^2)$	
$\beta_0 \propto 1$	
$\beta_1 \propto 1$	
$ au \sim \Gamma(0.001, 0.001)$	
$\sigma^2 = 1/ au$	
$ au_u \sim \Gamma(0.001, 0.001)$	
$\sigma_u^2=1/ au_u$	
Model	
<pre>model (    for (i in 1:length (normexam)) (         for (i in 1:length (normexam)) (</pre>	
normexam[i] ~ dnorm(mu[i], tau) mu[i] <- cons[i] * beta0 + standlrt[i] * beta1 + u[school[i]] * cons[i]	
, for (j in 1:length(u)) (	
) u[j]~ dnorm(0, tau_u)	
# Priors	
beta0 ~ dflat() beta1 ~ dflat()	
tau ~ dgamma(0.001000, 0.001000) sigma <- 1 / sqrt(tau)	
tau u ~ dgamma(0.001000, 0.001000)	
sigma_u <- 1 / sqrt(tau_u)	
Run     Simulate     Selection     Simulate     Specify starting Values     Code     JS	Set
	n
Inputs: {y': 'normexam', 'L2ID': 'school', 'D': 'Normal', 'x': 'cons.standlit'	<u>}                                    </u>

Note Equations use MATHJAX and so underlying LaTeX can be copied and paste. The model code is based around the WinBUGS language with some variation. This is a more complex template for 2 level models.



#### Equations for model and model code

$$\begin{split} \operatorname{normexam}_{i} &\sim \operatorname{N}(\mu_{i}, \sigma^{2}) \\ \mu_{i} &= \beta_{0} \mathrm{cons}_{i} + \beta_{1} \mathrm{standlrt}_{i} + u_{\mathrm{school}[i]} \\ u_{\mathrm{school}[i]} &\sim \operatorname{N}(0, \sigma_{u}^{2}) \\ \beta_{0} &\propto 1 \\ \beta_{1} &\propto 1 \\ \tau &\sim \Gamma(0.001, 0.001) \\ \sigma^{2} &= 1/\tau \\ \tau_{u} &\sim \Gamma(0.001, 0.001) \\ \sigma^{2}_{u} &= 1/\tau_{u} \end{split}$$

Note Equations use MATHJAX and so underlying LaTeX can be copied and paste. The model code is based around the WinBUGS language with some variation. This is a more complex template for 2 level models.



## Outbug function

```
outbug = '''
model{
  for (i in 1:length(${y})) {
     ${y}[i] ~ dnorm(mu[i], tau)
     mu[i] <- ${mmult(x, 'beta', 'i')}
  }</pre>
```

```
# Priors
% for i in range(0, x.ncols()):
beta${i} ~ dflat()
% endfor
tau ~ dgamma(0.001000, 0.001000)
sigma <- 1 / sqrt(tau)</pre>
```

```
111
```

}



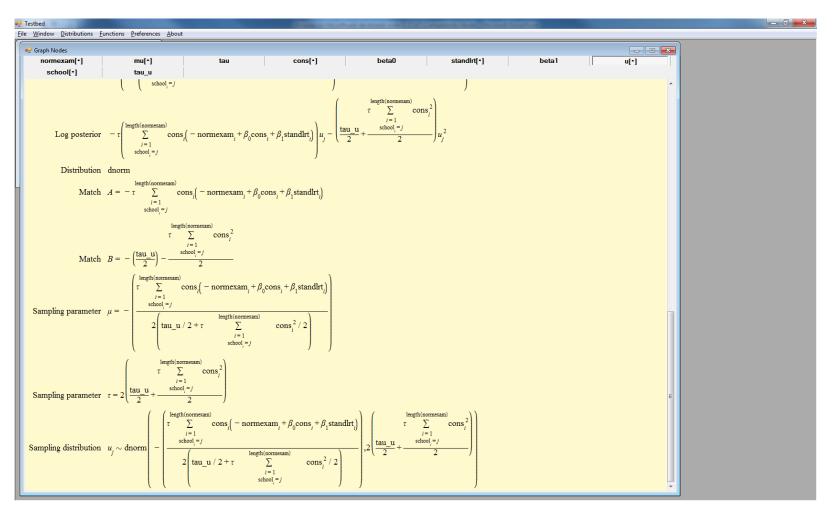
#### Model code in detail

```
model {
 for (i in 1:length(normexam)) {
   normexam[i] ~ dnorm(mu[i], tau)
   mu[i] <- cons[i] * beta0 + standlrt[i] * beta1 + u[school[i]] * cons[i]
  }
 for (j in 1:length(u)) {
   u[j] \sim dnorm(0, tau u)
 }
# Priors
 beta0 \sim dflat()
 beta1 ~ dflat()
 tau ~ dgamma(0.001000, 0.001000)
 tau u ~ dgamma(0.001000, 0.001000)
}
```

For this template the code is, aside from the length function, standard WinBUGS model code.

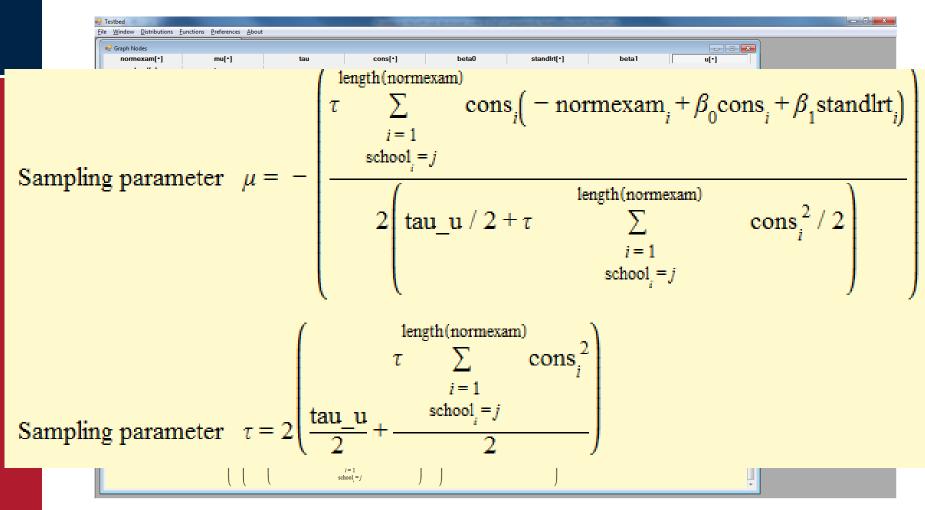


#### Bruce's (Demo) algebra system step for parameter u



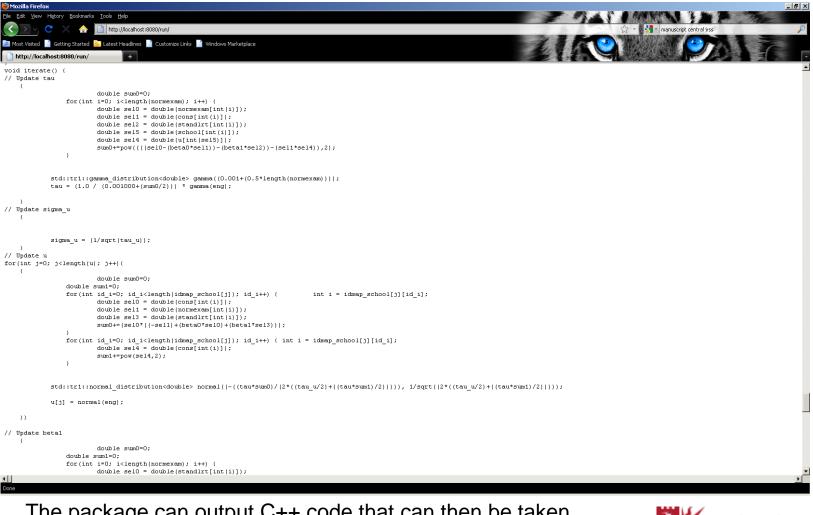


#### Bruce's (Demo) algebra system step for parameter u





#### Output of generated C++ code



The package can output C++ code that can then be taken away by software developers and modified.



#### Output of generated C++ code

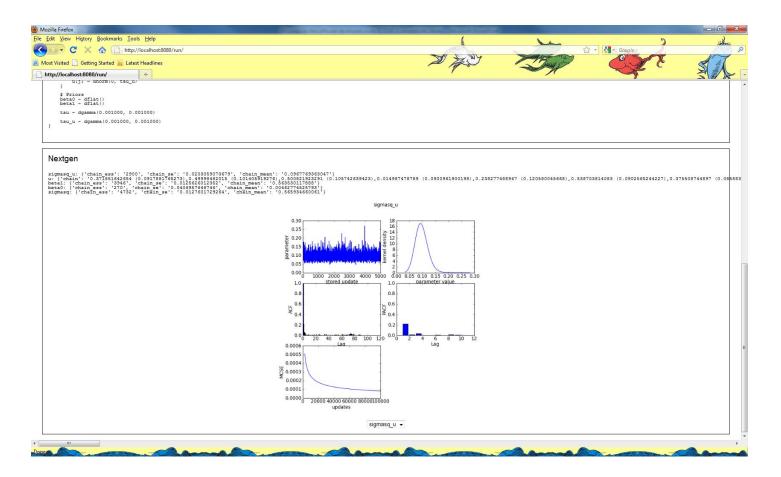
```
_ 8 ×
                  Hi<u>s</u>tory <u>B</u>ookmarks <u>T</u>ools <u>H</u>elp
                       http://localhost:8080/run/
// Update u
for(int j=0; j < length(u); j++){
                             double sum0=0;
                   double sum1=0;
                   for(int id i=0; id i<length(idmap school[j]); id i++) {</pre>
                                                                                                 int i = idmap school[j][id i];
                             double sel0 = double(cons[int(i)]);
                             double sel1 = double(normexam[int(i)]);
                             double sel3 = double(standlrt[int(i)]);
                             sum0+=(sel0*((-sel1)+(beta0*sel0)+(beta1*sel3)));
                   }
                   for(int id i=0; id i<length(idmap_school[j]); id_i++) { int i = idmap_school[j][id_i];</pre>
                             double sel4 = double(cons[int(i)]);
                             sum1+=pow(sel4,2);
                   3
              std::tr1::normal distribution<double> normal((-((tau*sum0)/(2*((tau u/2)+((tau*sum1)/2)))), 1/sqrt(
              u[j] = normal(eng);
    } }
            // Update beta1
                         double sum0=0;
                     double sum1=0;
                     for(int i=0; i<length(normexam); i++) {
                         double sel0 = double(standlrt[int(i)]);
               The package can output C++ code that can then be taken
```

away by software developers and modified.



18

### Output from the E-STAT engine



Here the six-way plot functionality is in part taken over to STAT-JR after the model has run. In fact graphs for all parameters are calculated and stored as picture files so can be easily viewed quickly.



### Output from the E-STAT engine

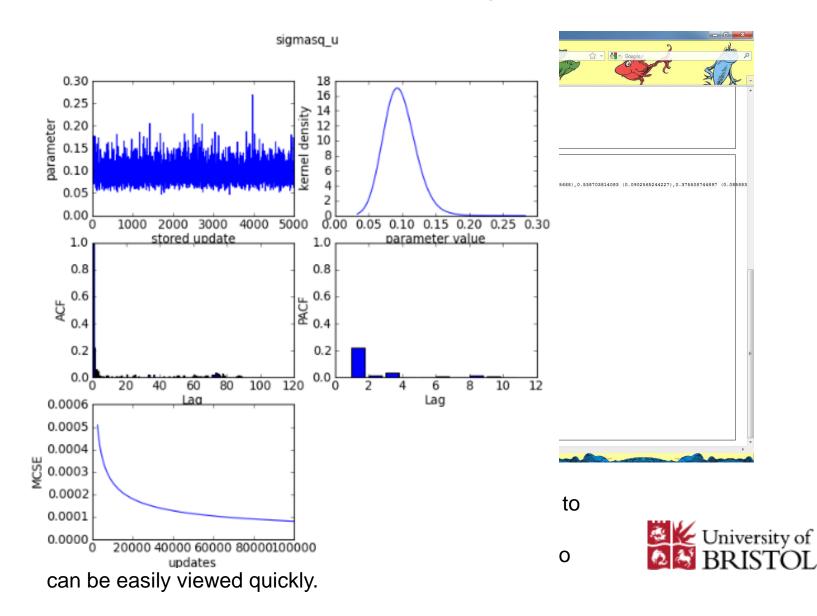
Edit View History Bookmarks Tools Help					
		A	Le	2	where .
C X 🟠 http://localhost:8080/run/		JUN 2 VOV	Stand Friday	3 - Google	A Company
lost Visited 📄 Getting Started <u>a</u> Latest Headlines		2 AS	The way	C	
http://localhost:8080/run/		/	- A		31.1.1.
u[j] ~ unorm(0, tau_u) }					·
<pre># Priors beta0 ~ dflat()</pre>					
betal ~ dflat()					
tau ~ dgamma(0.001000, 0.001000) tau u ~ dgamma(0.001000, 0.001000)					
}					
Nextgen					
sigmasg_u: {'chain_ess': '2900', 'chain_se': '0.0203059073	679', 'chain_mean': '0.0967769363047'}				
<pre>u: ('chāin': '0.371851642654 (0.0917881765273),0.499984820 betal: ('chāin_ess': '3946', 'chain_se': '0.0406557646746', igmasq: ('chāin_ess': '370', 'chāin_se': '0.0406557646746', sigmasq: ('chāīn_ess': '4732', 'chāin_se': '0.042760172926</pre>	<pre>15 (0.101405919276),0.500821923291 (0.105742639423),0.1 , 'chain_mean': '0.56353011788'} 'chain_mean': '0.00462374535793'\</pre>	014987478789 (0.0900961900198),0.23827	/48894/ (0.120580045668),0.538	703814083 (0.0902565244227),0.37	5508744897 (0.085883
sigmasq: {'chaīn_ess': '4732', 'chāin_se': '0.012760172926	4', 'chāin_mean': '0.565934660061'}				
	sigm	nasq_u			
1: {.cusiu_ess.: .5000.	', 'chain_se': '0.020 0.0917881765273),0.49	130220/36/2.'	cnain_mean.	: 0.0967765	363047.}
	chain_se': '0.0125626				
	hain se': '0.04069576	546746', 'chair	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	nain se': '0.04069576 'chain_se': '0.01276	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	nain_se': '0.04069576 'chain_se': '0.01276	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	nain_se': '0.04069576 'chain_se': '0.01276	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	nain_se': '0.04069576 'chain_se': '0.01276 %	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276 <sup>2000</sup> 20 40 60 100 120	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276 0.00000 0.00000 0.00000 0.000000	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276 0,000000	546746', 'chair 501729264', 'ch	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276	546746', 'chair 501729264', 'ch a a a a a a a a a a a a a a a a a a a	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276 0,000000	546746', 'chair 501729264', 'ch a a a a a a a a a a a a a a a a a a a	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276 "0.0 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 60000 800001000 updates	546746', 'chair 501729264', 'ch 000 2 4 6 8 10 12	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276 "0.0 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 60000 800001000 updates	546746', 'chair 501729264', 'ch 000 2 4 6 8 10 12	mean': '0.	004627745257	793'}
'chain ess': '270', 'ch	hain_se': '0.04069576 'chain_se': '0.01276 "0.0 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 5000 5000000	546746', 'chair 501729264', 'ch a a a a a a a a a a a a a a a a a a a	mean': '0.	004627745257	793'}

Here the six-way plot functionality is in part taken over to STAT-JR after the model has run. In fact graphs for all parameters are calculated and stored as picture files so can be easily viewed quickly.



sigma u: { beta: beta( sigma

#### Output from the E-STAT engine



#### Interoperability with WinBUGS

Mozilla Firefox Edit Vjew Hętory Bookmariis Tools Help C Tools Help Most Visited C Getting Started Latest Headines Customize Links Vince http://localhost:8060/run/	Luves Marketplace
	Stat-JR Demonstrator
Template: 2LevelMod Change Dataset: tutorial Change Vie	<u>w Summary</u>
Configuration	
response:	Start again normexam
Level 2 ID:	school
specify distribution:	Normal
explanatory variables:	cons,standirt
Name of output results:	out
Is estimation method by MCMC:	Yes
Choose estimation engine - eSTAT, WinBUGS, MLwiN:	WinBUGS
number of chains:	3
Random Seed:	1
length of burnin:	1000
number of iterations:	5000
thinning:	1
Equation rendering	$\operatorname{normexam}_i \sim \operatorname{N}(\mu_i, \sigma^2)$
	$\mu_i = eta_0 \mathrm{cons}_i + eta_1 \mathrm{standlrt}_i + u_{\mathrm{school}[i]}$
	$u_{ ext{school}[i]} \sim \mathrm{N}(0, \sigma_u^2)$
ne	$\theta \propto 1$

Interoperability in the user interface is obtained via a few extra inputs. In fact in the template code user written functions are required for all packages apart from WinBUGS. The transfer of data between packages is however generic.



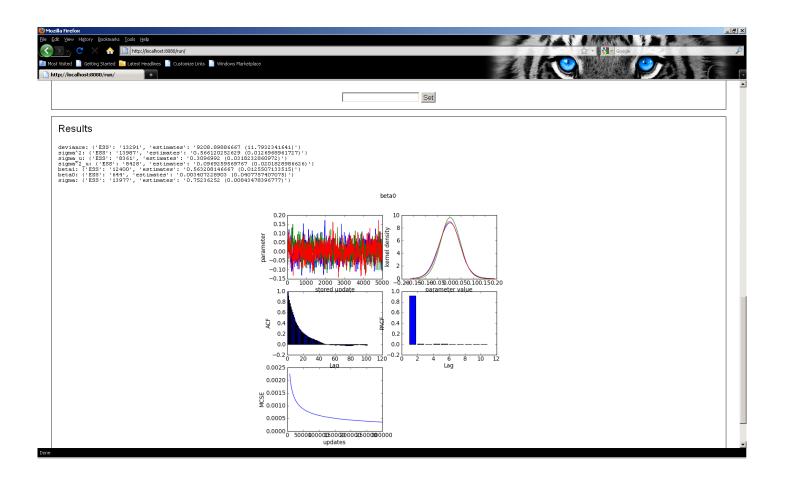
#### Interoperability with WinBUGS

🎱 Mozilla Firefox	response:	normexam	_ 8 ×
Ele Edit Yew Higory Bookmarks Iools Help  C   C   Must Visited  G Getting Started  Latest Headines  http://localhost:8080/run/  +	Level 2 ID:	school	
	specify distribution:	Normal	<u> </u>
Template: 2LevelMod <u>Change</u> Dataset:	explanatory variables:	cons,standirt	
Configuration	Name of output results:	out	
s exp	Is estimation method by MCMC:	Yes	
Nam Is estimation - Choose estimation engine - eSTAT, V	Choose estimation engine - eSTAT, WinBUGS, MLwiN:	WinBUGS	
	number of chains:	3	
nu	Random Seed:	1	
Equation rendering	length of burnin:	1000	
	number of iterations:	5000	
Done	thinning:	1	-1

Interoperability in the user interface is obtained via a few extra inputs. In fact in the template code user written functions are required for all packages apart from WinBUGS. The transfer of data between packages is however generic.



## Output from WinBUGS with multiple chains

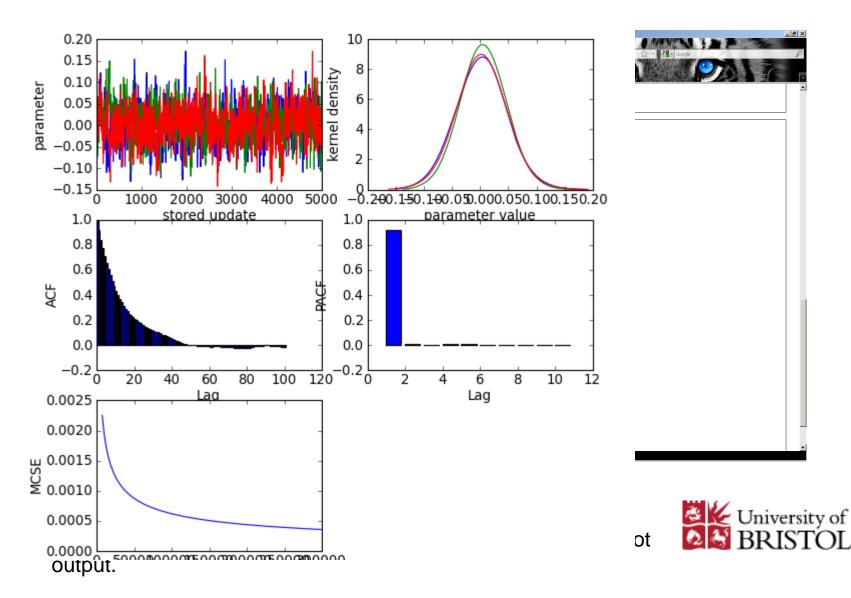


STAT-JR generates appropriate files and then fires up WinBUGS. Multiple Chains are superimposed in the sixway plot output.



#### Output from WinBUGS with multiple chains





## **Multiple Membership Models**

- Example is Scottish lip cancer data
- Response is Poisson (number of cases)
- Use as offset expected cases based on population size, makeup
- One predictor percaff percentage in agriculture, farming, fishing.
- Use the template MultipleMembershipNLev to allow both own random effect and neighbour random effects
- Template will allow fitting in STAT-JR engine, WinBUGS or MLwiN.



#### Inputs for model

👻 Mozilla Firefox			a) 🗙
Ejle Edit View Higtory Bookmarks Tools Help	A STREET, STRE	LESS WAR	
🔇 👻 🔁 📩 http://localhost:8080/run/	ये <b>- 3</b> - 9	ioogle	P
🔯 Most Visited 📄 Getting Started 🔊 Latest Headlines 📄 Free Hotmail 👕 Suggested Sites 📄 Web Slice Gallery			
http://localhost:8080/run/	Carlos and the		
			1.0

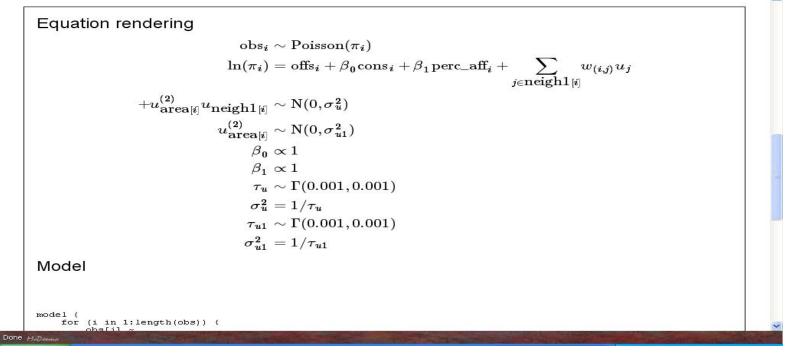
Template: MultipleMembershipNLev Change Dataset: lips Change View Summary

Configuration	Start again
response:	obs
Number of Classifications:	1
Classification 1:	area
MM IDs:	neigh1,neigh2,neigh3,neigh4,neigh5,neigh6,neigh7,neigh8,neigh9,neigh10,neigh11
MM weights:	weight1,weight2,weight3,weight4,weight5,weight6,weight7,weight8,weight9,weight10,weight11
specify distribution:	Poisson
Is there an offset?:	Yes
offset:	offs
explanatory variables:	cons,perc_aff
Name of output results:	out
	Next



#### LaTeX for Model

🕹 Mozilla Firefox	1	_ ) 🗗	$\mathbf{X}$
Elle Edit View Higtory Bookmarks Tools Help			
🔨 🖂 🔻 🗟 http://localhost:8080/run/	🖒 - 🎦 - Google		P
💌 Most Visited 📄 Getting Started 🔊 Latest Headlines 📄 Free Hotmail 📄 Suggested Sites 📄 Web Site Gallery		1/1	
http://localhost:8080/run/		a series	
			~





#### Model Code

Averalla Firefox Ele Edit View History Bookmarks Tools Help	2 🔀
🕥 🔄 🧟 🖒 http://localhost:8080/run/	2
Most Visited Getting Started Latest Headlines Free Hotmail Suggested Sites Web Slice Gellery           http://localhost:8080/run/	
<pre>model {     for (i in 1:length(obs)) {         dos[1] ~         dpois(p[i])         ln(p[i]) &lt;-         offs[1] +         cons[i] * beta0 + perc_aff[i] * beta1 + ul[area[i]] * cons[i] + weight[1] * umm[neigh1[1]] + weight2[i] * umm[neigh2[1]] + weight3[1] * umm[neigh4[1]] + weight4[1] * umm[neigh4[1]] + weight6[1] * umm[neigh6[1]] + weight6[1] * umm[neigh6[1]] + weight9[i] * umm[neigh10[1]] + weight10[i] * umm[neigh10[1]] + weight11[i] * umm[neigh10[1]] + weight11[i] * umm[neigh10[1]]     for (i1 in 1:length(umm)) {         uum[i] ~ dnorm(0, tau_u)         }         for (j in 1:length(umm)) {             uum[i] ~ dnorm(0, tau_u)         }         }     } }</pre>	
<pre># Priors beta0 ~ dflat() beta1 ~ dflat()  tau u1 ~ dgamma(0.001000, 0.001000) sigma2_u1 &lt;- 1 / tau_u1</pre>	
tau_u ~ dgamma(0.001000, 0.001000) sigma2_u <- 1/tau_u )	
Run     Simulate     Selection     Simulate     Specify starting Values     Code     JS     Save imputed every.	~



## The E-STAT project – still to come

We have lots of work to do:

- Parallel processing.
- E-books.
- Optimising code generation.
- Improving algebra system.
- Suites of templates for missing data and social network models.
- Interoperability with SAS and hooking up more templates for other packages.

