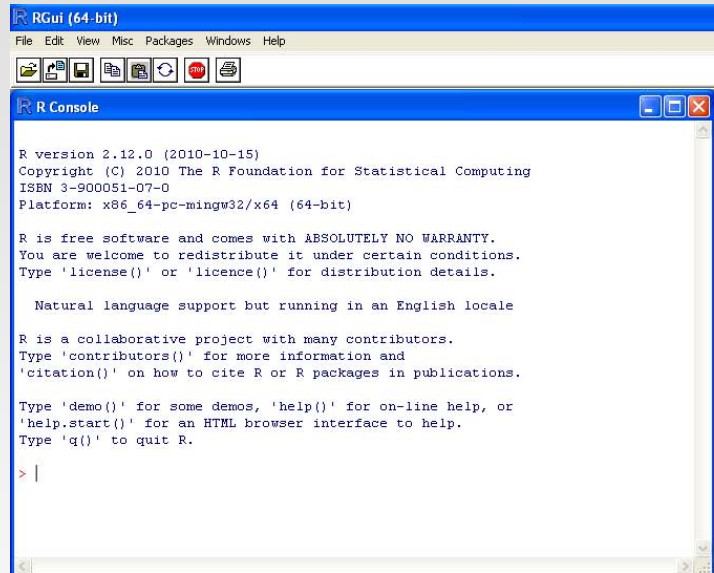


Session 4: Practical Demonstration

Outline

- Look at how you can fit the IE and convolution models from the lectures
- Introduce software package R
 - ◆ Simulate data using ecoreg function
 - ◆ Fit MLE of IE model using ecoreg function
 - ◆ Fit convolution model using function RxCEcollnf
- Introduce program WinBUGS
 - ◆ Fit IE and HRR models
 - ◆ Demonstration of WinBUGS
- Summary of the different packages and functions

Introduction to R



The screenshot shows the R GUI (64-bit) window with the R Console open. The console displays the following text:

```
R version 2.12.0 (2010-10-15)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-pc-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```


R is a software package for data manipulation, analysis and graphical display

Very flexible

Lots of inbuilt functions

User can write own functions

R for Windows can be downloaded free at
<http://cran.us.r-project.org/bin/windows>

The Windows taskbar at the bottom shows the Start button and several open applications: Inbox - Mcr..., Lecture 1: G..., 2 Windows..., WinBUG..., RGui (64-bit), RGui (64-bit), 2 Microsoft..., Practical de..., and R-intro.pdf. The search bar on the right contains the text "Search Desktop".

Simulate aggregate and survey data

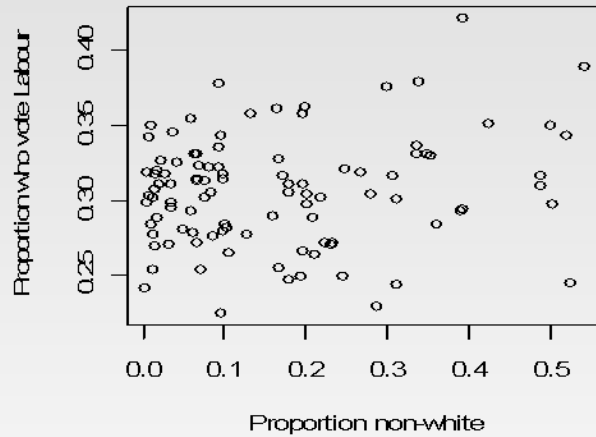
- Assume that an individual either votes Labour or does not vote Labour with a probability that depends only the individual's
 - ◆ Ethnicity, odds ratio = 1.5 non-white/white
 - ◆ job type, odds ratio = 0.6 for non-manual/manual
 - ◆ and smoking status, odds ratio = 2 for smoking/non-smoking
- Assume 100 areas, 10,000 people in each area, and survey is a random sample of 20 individuals from each area
- Probability a white, manual, non-smoker votes Labour = 0.3
- Probability a non-white, manual, non-smoker votes Labour = 0.39

Simulate aggregate and survey data

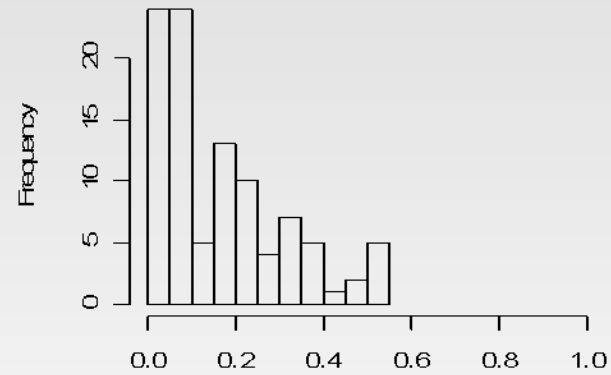
- Use the package *ecoreg* to simulate some voting data.
- R code
 - ◆ `ng <- 100 # number of areas`
 - ◆ `N <- rep(10000, ng) # number of people in each area`
 - ◆ `nonwhite <- rbeta(ng, 1, 5);`
 - ◆ `nonmanual <- runif(ng, 0, 1)`
 - ◆ `smoke <- runif(ng, 0, 0.5)`
 - ◆ `sim <- sim.eco(N, binary = ~ nonwhite + nonmanual +
smoke, mu = log(0.3/0.7), alpha = log(c(1.5, 0.6, 2)),
isam = 20)`

Simulated aggregate data

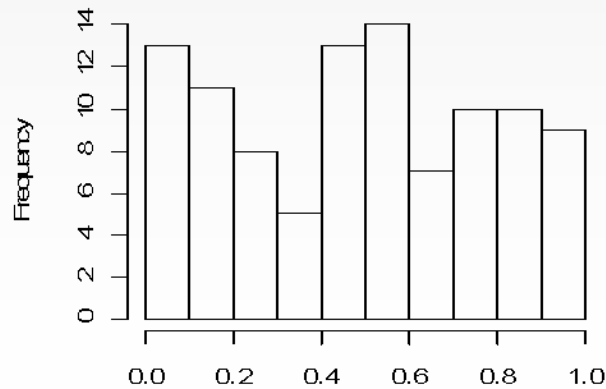
Plot of non-white versus Labour voting



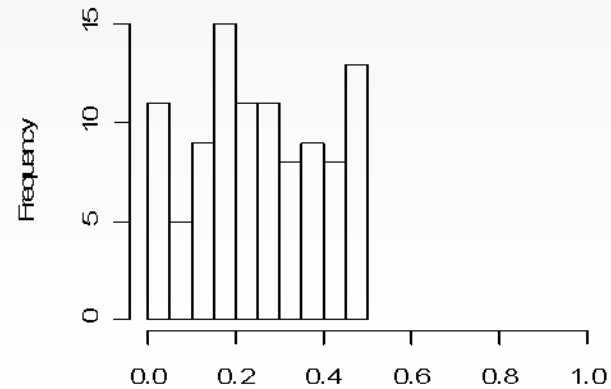
Proportion who are non-white



Proportion who are non-manual



Proportion who smoke



Simulated individual data

- For individual survey data, we only want to keep about 1/3 of the generated data, i.e. We are assuming we have individual data from a random sample of the areas, with each area included with probability 1/3. For this dataset, 32 areas (640 individuals) are included
- Contingency table for individual data (32 areas, with 640 individuals), gives an odds ratio of 1.55.

	Vote Labour	Don't vote Labour	
Non-white	46	78	124
White	142	374	516
	188	452	640

R package “ecoreg”

- Fits a maximum likelihood estimation of the HRR model in Jackson, Best and Richardson (2006), or the convolution model of Wakefield (2004)
- Estimates an underlying individual-level logistic regression model using
 - ◆ Only individual data
 - ◆ Only aggregate data (IE model)
 - ◆ Or individual and aggregate data together (HRR model)
- Can include any number of covariates
- Covariates can be
 - ◆ Individual-level covariates
 - ◆ binary or categorical – expressed as proportions over the group
 - ◆ continuous – assumed normally distributed and expressed as within-area means and optional covariances
 - ◆ Contextual (group-level)

Data format for the *ecoreg* package

- Individual data: dataframe with one line per individual, e.g.

y	group	nonwhite	nonmanual	smoke
0	2	0	0	1
0	2	0	1	1
0	2	1	1	0
0	2	1	1	1
1	2	1	1	0
1	2	0	1	0

- Aggregate data: dataframe with one line per area, covariates are proportions, e.g.

y	N	\bar{X}_i nonwhite	nonmanual	smoke
2942	10000	0.39	0.71	0.17
2719	10000	0.23	0.82	0.25
2971	10000	0.50	0.92	0.27

Analysis with ecoREG

- Fit the integrated ecological model with random intercepts, using both individual and aggregate data

The model

$$Y_i \sim \text{Binomial}(p_i, N_i)$$

$$p_i = p_i^W (1 - \bar{X}_i) + p_i^N \bar{X}_i$$

$$p_i^W = \text{expit}(\alpha_i)$$

$$p_i^N = \text{expit}(\alpha_i + \beta)$$

R code

eco(cbind(y, N) ~ 1, binary = ~ nonwhite,

ifformula = y ~ nonwhite,

random = TRUE, random effect

group = 1:100, igroup = group,

data = aggeco, idata = indeco,

model = "marginal")

Formula for individual data

Area identifier for the random effects

Individual-level covariate

Contextual covariate

\bar{X}_i

Analysis with ecoreg

The model

$$Y_i \sim \text{Binomial}(p_i, N_i)$$

$$p_i = p_i^W (1 - \bar{X}_i) + p_i^N \bar{X}_i$$

$$p_i^W = \text{expit}(\alpha_i)$$

$$p_i^N = \text{expit}(\alpha_i + \beta)$$

Output from R

Aggregate-level odds ratios:

	OR	l95	u95
(Intercept)	0.415924	0.4117755	0.4201143

Mean probability for non-whites

Individual-level odds ratios:

	OR	l95	u95
nonwhite	1.369087	1.322954	1.416828

Odds ratio, $\exp(\beta)$

Random effect standard deviation

	estimate	l95	u95
sigma	0.1587761	0.1540287	0.1636698

Estimate of random effect variance

-2 x log-likelihood: 2351.896

R package "RxCeCollnf"

- Fits the hierarchical model of Greiner and Quinn (2009) (or Wakefield (2004) for 2x2 tables) to ecological data in which the underlying contingency tables can have any number of rows or columns
- Convolution of independent binomials for each row in the 2x2 table

$$Y_i \sim \sum_{\substack{\text{admissible} \\ \text{values of } Y_i^W}} \text{Binomial}(Y_i^W; p_i^W, N_i(1 - \bar{X}_i)) \times \text{Binomial}(Y_i - Y_i^W; p_i^W, N_i\bar{X}_i)$$

$$\text{logit } p_i^W \sim N(\mu^W, \Sigma^W); \quad \text{logit } p_i^N \sim N(\mu^N, \Sigma^N);$$

- Estimates functions of the convolution likelihood using
 - ◆ Only aggregate data
 - ◆ Or individual (survey data) and aggregate data together
- Can only include one discrete individual-level covariate

Data format for the *RxCcollnf* package

- Aggregate data: dataframe with one line per area (i.e. one line per table), entries are row and column totals (not proportions), e.g.

lab	nonlab	nonwhite	white
2942	7058	3909	6091
2719	7281	2328	7672
2971	7029	5014	4986

- Individual data: dataframe

- ◆ In same format as aggregate data, i.e. summed up
- ◆ Contains same number of rows as aggregate data, and in same order
- ◆ Areas with no survey data contain zeros
- ◆ Must have $R * C$ columns (one column for each cell of the contingency table)
- ◆ Entries are cell totals of each contingency table
- ◆ Column names must be in specific format

KK.nonwhite.lab	KK.white.lab	KK.nonwhite.nonlab	KK.white.nonlab
0	0	0	0
1	5	3	1
4	9	4	3

RxCeCollnf - Tune

- Need to call function Tune first
 - ◆ This tunes the MCMC algorithm used to fit the model
 - ◆ To sample from the posterior, algorithm uses a Metropolis-Hastings step with a multivariate t_4 proposal distribution
 - ◆ Function Tune tunes the MCMC algorithm to achieve acceptance ratios of between 0.2 and 0.5 for the t_4 proposal
 - ◆ Can either specify values of the hyper-priors or use default values
 - ◆ Returns vector called “rhos” which should be fed into Analyze

Aggregate data only, R code – Tune

- `tune.agg <- Tune("lab, nonlab ~ nonwhite, white",
 data=aggquinn)`
- Ordering of names in function is important
 - ◆ LHS of ~
 - ◆ These are the column totals
 - ◆ Assumes last column are abstainers, so for a 2x2 table some of the returned values are of no use
 - ◆ RHS of ~
 - ◆ Assumes final column is the reference category
- Can also specify
 - ◆ `num.runs` – number of times the tuning algorithm will be implemented, default = 12
 - ◆ `num.iters` – number of iterations in each run of the tuning algorithm, default = 10,000
- Returns `tune.agg$rhos` to use with [Analyze](#)

Aggregate data only, R code – Analyze

- `Analyze` returns samples from the posterior distribution as an mcmc object

```
chain1.agg <- Analyze("lab, nonlab ~ nonwhite, white",  
rho.vec = tune.agg$rhos,  
data = aggquinn,  
num.iters = 1000000,  
burnin = 500000,  
save.every = 50,  
debug = 1)
```

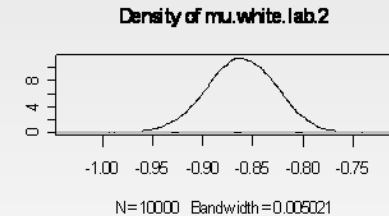
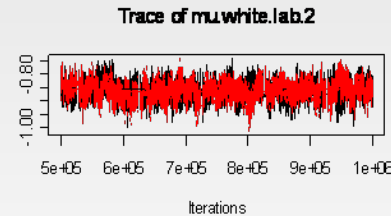
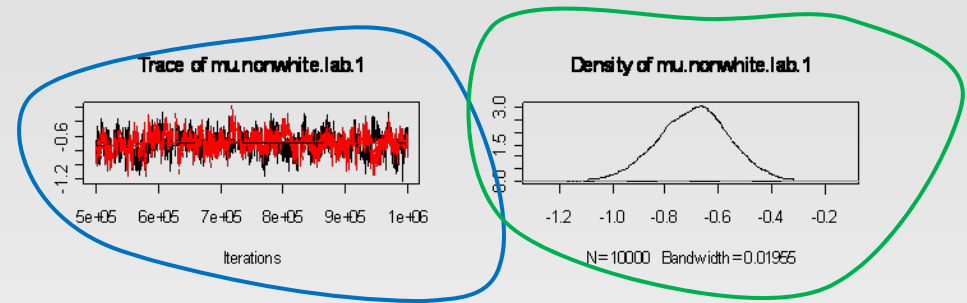
- Run at least 2 chains

Output from RxCEcolInf

- Analyze returns an object of class `mcmc`
 - ◆ `agg.mcmc<- mcmc.list(chain1.agg, chain2.agg)`
- Main things of interest
 - ◆ **Lambda** – fraction of each races voters supporting a particular candidate
 - ◆ **Turnout** – proportion of each race voting
 - ◆ **Gamma** – fraction that each race contributes to the voting electorate
 - ◆ **Beta** – fraction of each race that supports a particular candidate
- For a 2x2 table, only interested in **beta**

Trace plots

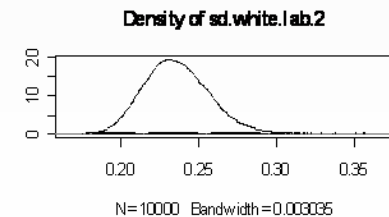
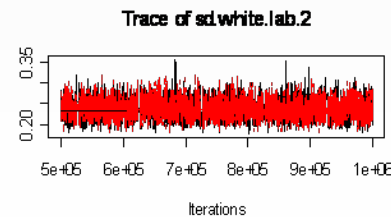
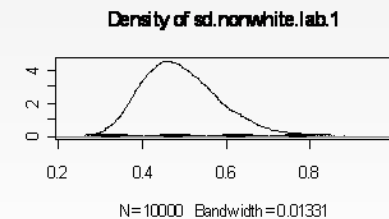
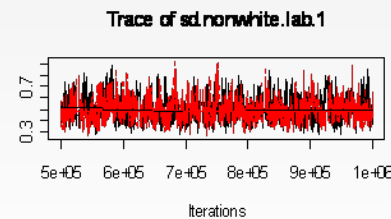
- `plot(agg.mcmc[,1:4])`



Use R code

`dimnames(agg.mcmc[[1]])[[2]]`

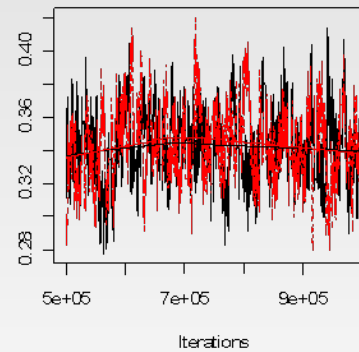
to give column names to see which are of interest



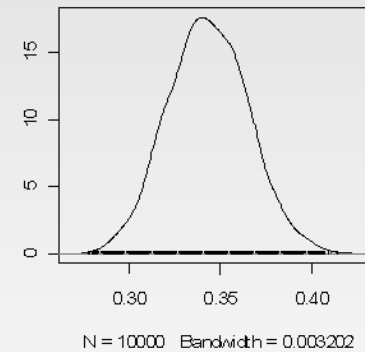
Trace plots

- `plot(agg.mcmc[,16:17])`

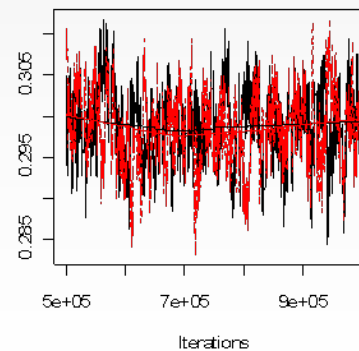
Trace of BETA.nonwhite.lab



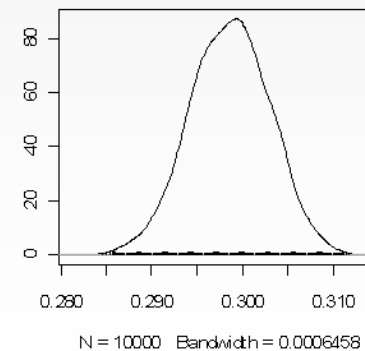
Density of BETA.nonwhite.lab



Trace of BETA.white.lab



Density of BETA.white.lab



Calculating odds ratios and probabilities

- `beta1 <- c(agg.mcmc["BETA.nonwhite.lab"][[1]],
agg.mcmc["BETA.nonwhite.lab"][[2]])`
- `beta2 <- c(agg.mcmc["BETA.white.lab"][[1]],
agg.mcmc["BETA.white.lab"][[2]])`
- `or <- beta1 * (1 - beta2) / ((1 - beta1) * beta2)`
- `round(mean(or),2); round(quantile(or, probs=c(0.025, 0.975)),2)`
- `round(mean(beta1),2); round(quantile(beta1, probs=c(0.025, 0.975)),2)`
- `round(mean(beta2),2); round(quantile(beta2, probs=c(0.025, 0.975)),2)`
- OR – 1.23 (0.97, 1.55)
- Probability a non-white votes Labour – 0.34 (0.30, 0.39)
- Probability a white votes Labour – 0.30 (0.29, 0.31)

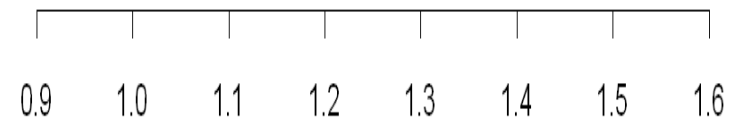
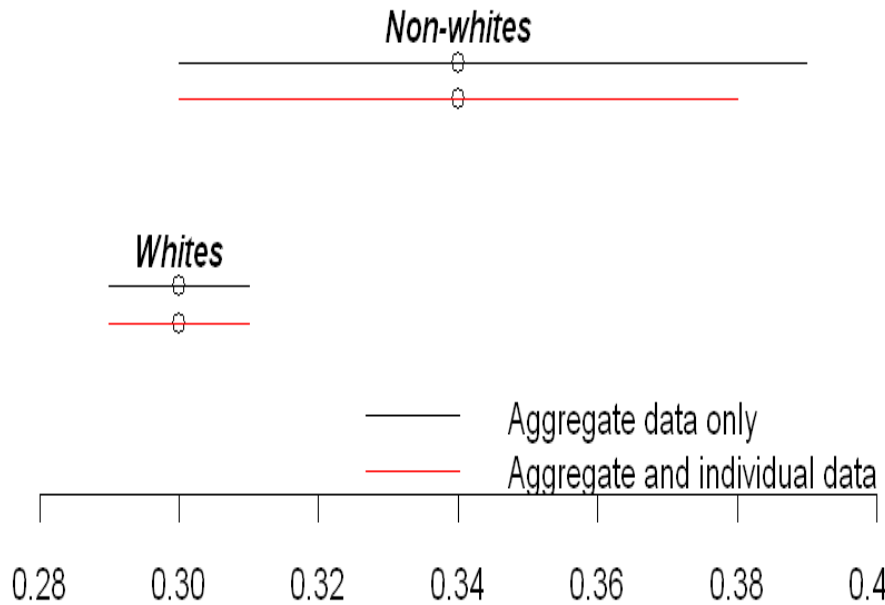
Now also include individual-level survey data

- `tune.comb <- TuneWithExitPoll("lab, nonlab ~ nonwhite, white",
data = aggquinn, exitpoll = indquinn)`
- `chain1.comb <- AnalyzeWithExitPoll("lab, nonlab ~ nonwhite, white",
data = aggquinn, exitpoll = indquinn,
rho.vec = tune.comb$rhos,
num.iters = 1000000,
burnin = 500000,
save.every = 50,
debug = 1)`
- Post analysis commands as for aggregate only analysis
- OR – 1.22 (0.98, 1.49)
- Probability a non-white votes Labour – 0.34 (0.30, 0.38)
- Probability a white votes Labour – 0.30 (0.29, 0.31)

Comparison of aggregate and hybrid estimates using RxCEcolInf

Probability of voting Labour

Odds ratio



Notes on RxCEcolInf

- Inclusion of survey data
 - ◆ Assumes that the survey is a simple random sample
 - ◆ Future implementations will allow incorporation of more complicated sampling schemes
- Inclusion of additional individual level covariates
 - ◆ As long as the full cross-classification of covariates is known, the contingency table simply has more rows
- Inclusion of a contextual covariate
 - ◆ Although R package cannot include a contextual covariate, it is possible to do so via a regression on the mean log odds probabilities, this is an implementation issue not a modelling issue

WinBUGS

- WinBUGS (Bayesian Inference Using Gibbs Sampling) is a computer program for the Bayesian analysis of complex statistical models using Markov Chain Monte Carlo (MCMC) methods
- Developed initially at the MRC Biostatistics Unit in Cambridge, then jointly with Imperial College
- User specifies the model (likelihood and prior)
- WinBUGS generates samples from the posterior distribution
 - ◆ Check convergence of posterior distributions
 - ◆ Make inferences and obtain parameter estimates
- Available free from
<http://www.mrc-bsu.cam.ac.uk/bugs/welcome.shtml>

WinBUGS – Data format

- Individual data: text file with one line per individual
- Aggregate data: text file with one line per area, covariates are proportions
- Can also specify data in list format

```

iy[] group[] inonwhite[]
0 2 0
0 2 0
.
.
.
0 98 0
0 98 0
END
  
```

```

y[] N[] nonwhite[]
2942 10000 0.390917701650597
2719 10000 0.232773058036124
.
.
.
3023 10000 0.218001180641006
2774 10000 0.127572515215463
END
  
```

```

list(nareas = 100, Nsubjects = 460)
  
```

HRR model, using WinBugs

```

model {
  for (i in 1:Nareas {
    y[i] ~ dbin(p[i], N[i])
    p[i] <- pw[i] * (1 - nonwhite[i]) + pn[i] * nonwhite[i]
    logit(pw[i]) <- alpha[i] # pw[i] = marginal prob. for individual who is white
    logit(pn[i]) <- alpha[i] + beta } # pnw[i] = marg. prob. for non-white
  }

```

```

for(i in 1:Nsubjects) {
  iy[i] ~ dbern(ip[i])
  logit(ip[i]) <- alpha[group[i]] + beta*inonwhite[i]
}

```

```

for(i in 1:Nareas) {
  alpha[i] ~ dnorm(alpha0, tau)
}

```

Priors

```

beta ~ dnorm(0, 0.43)
alpha0 ~ dnorm(0, 0.35)

tau ~ dgamma(0.5, 0.0015)
sigmasq <- 1/tau

```

}

Blue writing for analysis with aggregate data only
 Green writing for analysis with individual data only
 Blue and Green to include both levels of data

```

rr <- exp(beta)
logit(probN) <- alpha0 + beta
logit(probW) <- alpha0

```

Integrated Ecological (IE) model

Jackson et al (2006, 2008)

- Derived from an underlying individual-level model

$$y_{ij} \sim \text{Bernoulli}(p_{ij})$$

where $p_{ij} = p_{ij}(x)$ is a function of x (white/non-white), e.g.

$$\text{logit } p_{ij}(x) = \alpha_i + \beta x_{ij} \quad \Rightarrow \quad p_{ij}(x) = \text{expit}(\alpha_i + \beta x_{ij})$$

- Individual-level model is **averaged over population in area i** to obtain model at aggregate level

$$Y_i \sim \text{Binomial}(p_i, N_i); \quad p_i = \int p_{ij}(x) f_i(x) dx$$

where $f_i(x)$ is the distribution of x in area i

Integrated Ecological (IE) model for binary x

- For **a single binary x** , the integral $\int p_{ij}(x) f_i(x) dx$ is just the weighted sum over $x=0$ and $x=1$

$$\begin{aligned} p_i &= p_{ij}(x=0) \Pr_i(x=0) + p_{ij}(x=1) \Pr_i(x=1) \\ &= p_i^W (1 - \bar{X}_i) + p_i^N \bar{X}_i \end{aligned}$$

- Suppose we assume the individual-level model

$$\text{logit } p_{ij} = \alpha_i + \beta x_{ij}$$

- Then $\text{logit } p_{ij}(x=0) = \alpha_i \quad \Rightarrow p_i^W = \text{expit}(\alpha_i)$
- $\text{logit } p_{ij}(x=1) = \alpha_i + \beta \quad \Rightarrow p_i^N = \text{expit}(\alpha_i + \beta)$

Initial values

- To start the MCMC algorithm, you need initial values for all unknown quantities (parameters)
- These can either be
 - ◆ Specified by the user
 - ◆ Generated by WinBUGS
 - ◆ Mixture of user specified and WinBUGS generated
- E.g. `list(alpha = 0, beta = 0, tau = 0.1)`

WinBUGS demo – model specification

To specify a model, select

Model > Specification

to bring up the
Specification Tool
 dialog box

```

WinBUGS14
File Tools Edit Attributes Info Model Inference Options Doodle Map Text Window Help

model1.txt
model {
  for(i in 1:Nareas) {

    y[i] ~ dbin(p[i], N[i])

    p[i] <- pw[i] * (1 - nonwhite[i]) + pn[i] * nonwhite[i]
    logit(pw[i]) <- alpha[i]
    logit(pn[i]) <- alpha[i] + beta
  }

  # pn[i] = marginal probability for individual who is non-white
  # pw[i] = marginal probability for individual who is white

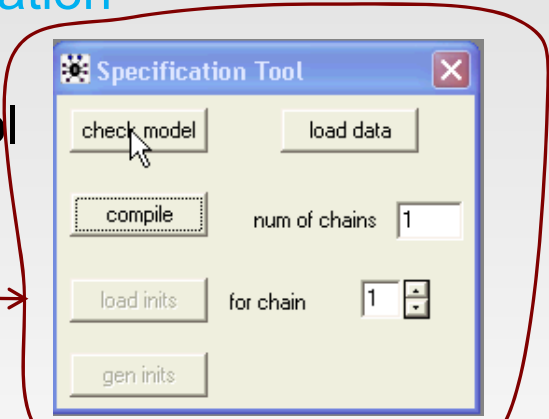
  for(i in 1:Nsubjects) {
    iy[i] ~ dbern(ip[i])
    logit(ip[i]) <- alpha[group[i]] + beta*inonwhite[i]
  }

  for(i in 1:Nareas) { alpha[i] ~ dnorm(alpha0, tau) }

  ## Priors
  beta ~ dnorm(0, 0.43)
  alpha0 ~ dnorm(0, 35)
  tau ~ dgamma(0.5, 0.0015)
  sigmasq <- 1/tau

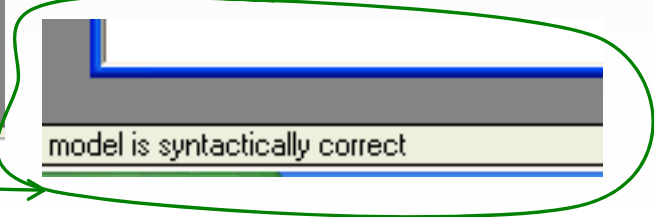
  rr <- exp(beta)
  logit(probNW) <- alpha0 + beta
  logit(probW) <- alpha0
}

# Things to monitor
# alpha0
# beta
# sigmasq
# rr
# probNW
# probW
    
```



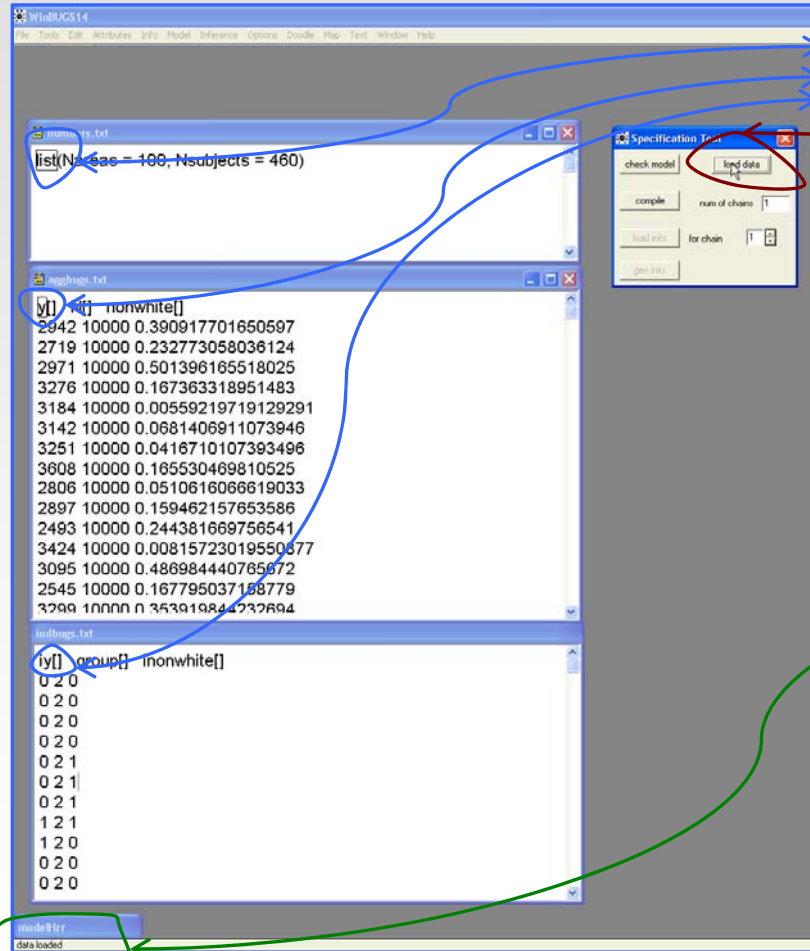
Highlight **“model”** in the model file and click **“check model”** in the dialog box

In the bottom left corner of the main window you should see



model is syntactically correct

WinBUGS demo – loading data



In data file, highlight “list” or first data value

Click “load data” in Specification Tool dialog box

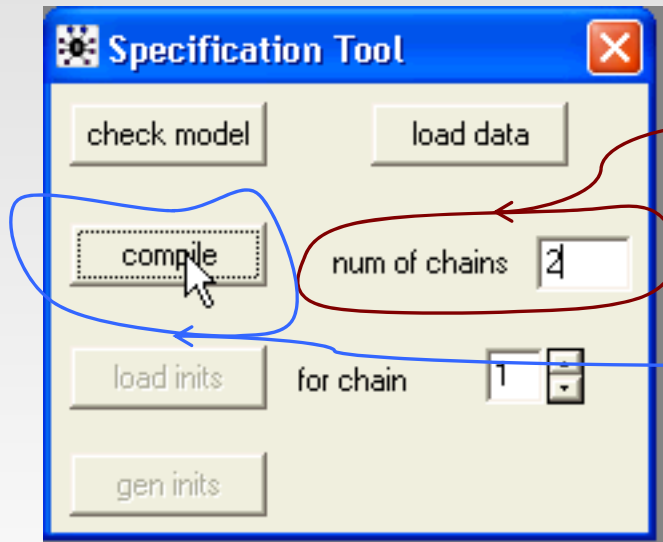
Repeat for as many data files as you have

In the bottom left corner of the main window you should see

data loaded

data loaded

WinBUGS demo – compile model



To compile the model,

In the Specification Tool dialog box, change “**num of chains**” to the number of chains you want to run. This should be at least 2.

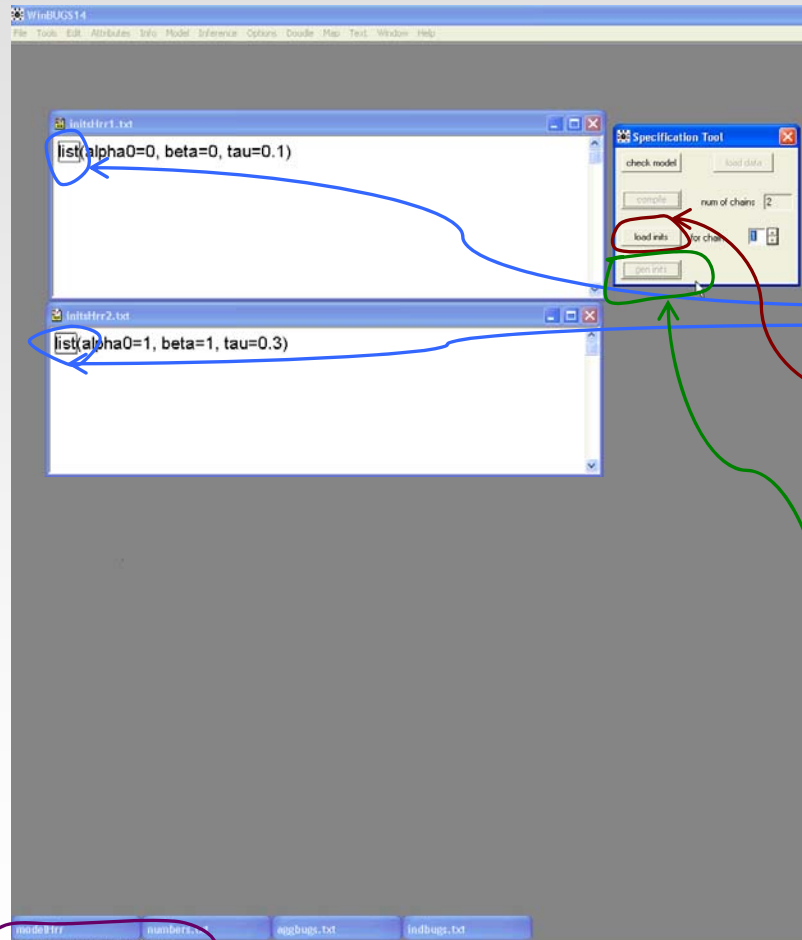
Click “**compile**” in the Specification Tool dialog box

In the bottom left corner of the main window you should see



model compiled

WinBUGS demo – initial values



You need to load initial values for all unknown quantities in the model (e.g. parameters and missing values), and for all chains

Highlight “list” in the initial value data file

In the Specification Tool dialog box, click “load inits”

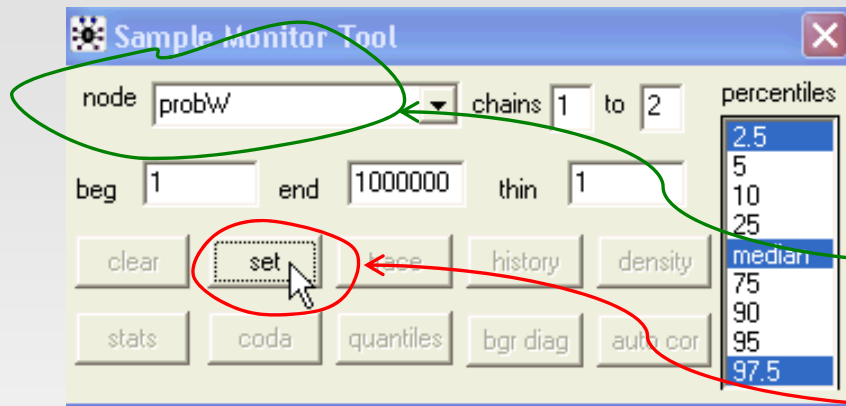
Repeat for all chains

Or you can generate initial values, click “gen inits” in the Specification Tool dialog box

In the bottom left corner of the main window you should see

initial values generated, model initialized

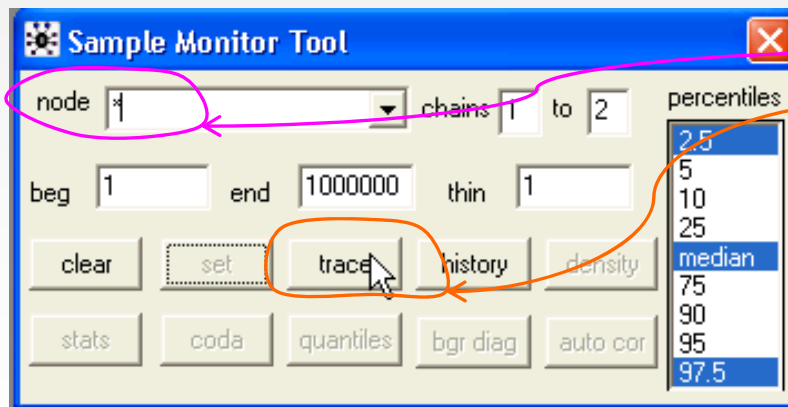
WinBUGS demo – monitor parameters



Select **Inference > Samples**, this brings up the **Sample Monitor Tool** dialog box

Type the name of any parameters you want to monitor in the “**node**” box

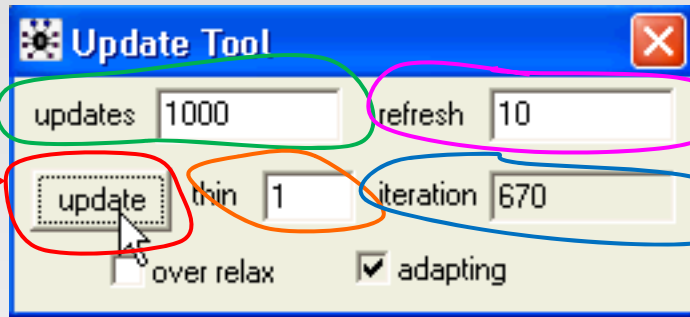
and click “**set**”



When you have set all the parameters you are interested in, type “*****” in the “**node**” box, and click “**trace**”

You will now be able to see a trace plot, which is a plot of the variable value against iteration number. The trace is dynamic, being redrawn each time the screen is redrawn.

WinBUGS demo - update



The model is now ready to run

Select **Model > Update**, to bring up the **Update Tool** dialog box

Type the number of iterations you want to run in the “**update**” box

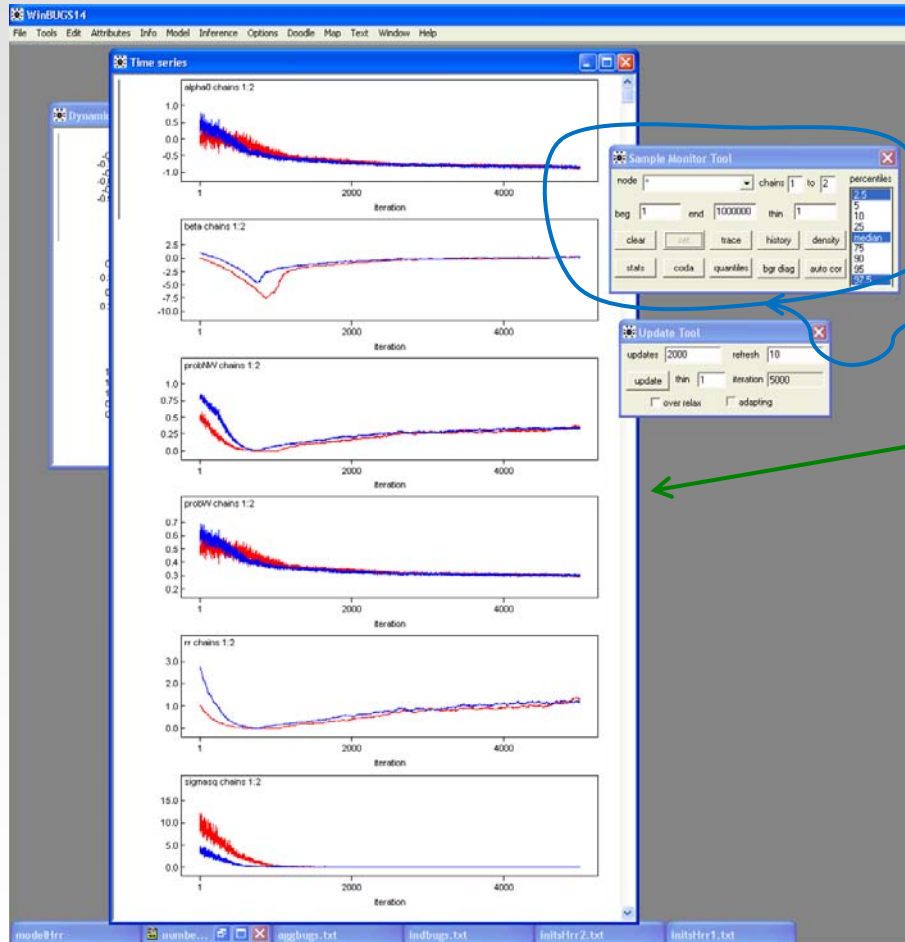
In the “**refresh**” box, type the number of updates between redrawing the screen, the number you want here will depend on how slow your model is

In the “**thin**” box, type the number you want to thin by, samples from every k th iteration will be stored, where k is the number you entered

and click “**update**”

Your model is now running. The number of iterations stored is shown in the “**iteration**” box, this number updates until the run is complete

WinBUGS demo – history plots

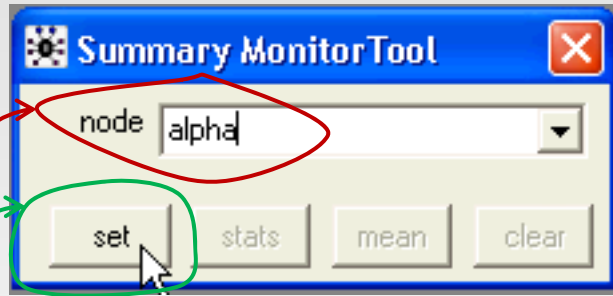


Once the model has been running for a while you can look at history plots to check for convergence.

In the **Sample Monitor Tool** dialog box, type “*” in the “**node**” box and click on “**history**”. You will see this plot.

You can also click the “**bgr diag**” box to look at plots of the Gelman-Rubin statistic, as modified by Brooks and Gelman (1998)

WinBUGS demo – Summary Monitor Tool



If you are satisfied that your model has converged, you can set the Summary Monitor Tool.

Select [Inference > Summary](#) to bring up the Summary Monitor Tool.

→ Enter the variable names of interest in the “**node**” box

Running means, standard deviations and quantiles will be calculated. The commands in this dialog are less powerful and general than those in the Sample Monitor Tool, but they require much less storage

→ Click on “**set**”, running means will now be calculated

WinBUGS demo - results

Once your model has finished running, you can look at various plots of the samples and calculate summary statistics.

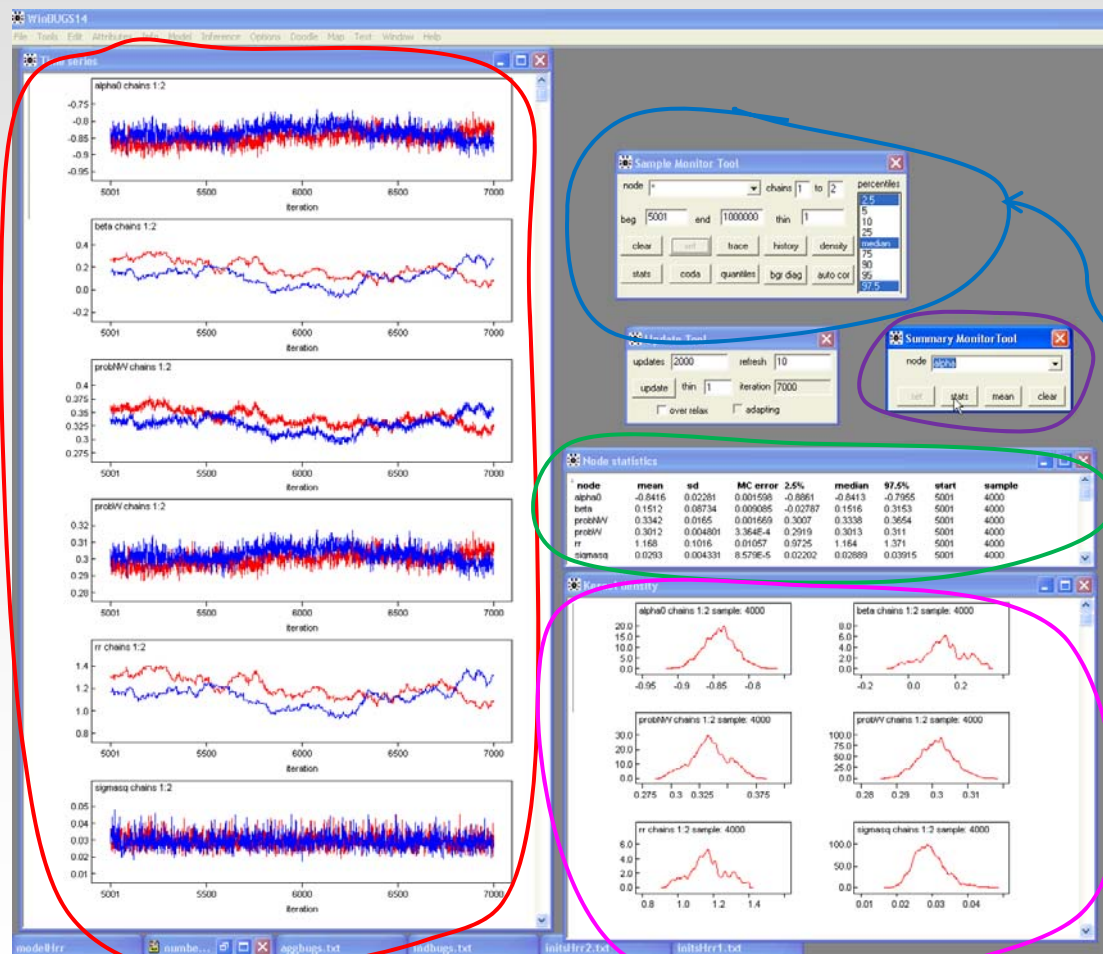
In the **Sample Monitor Tool** dialog box, click on

history

stats

density

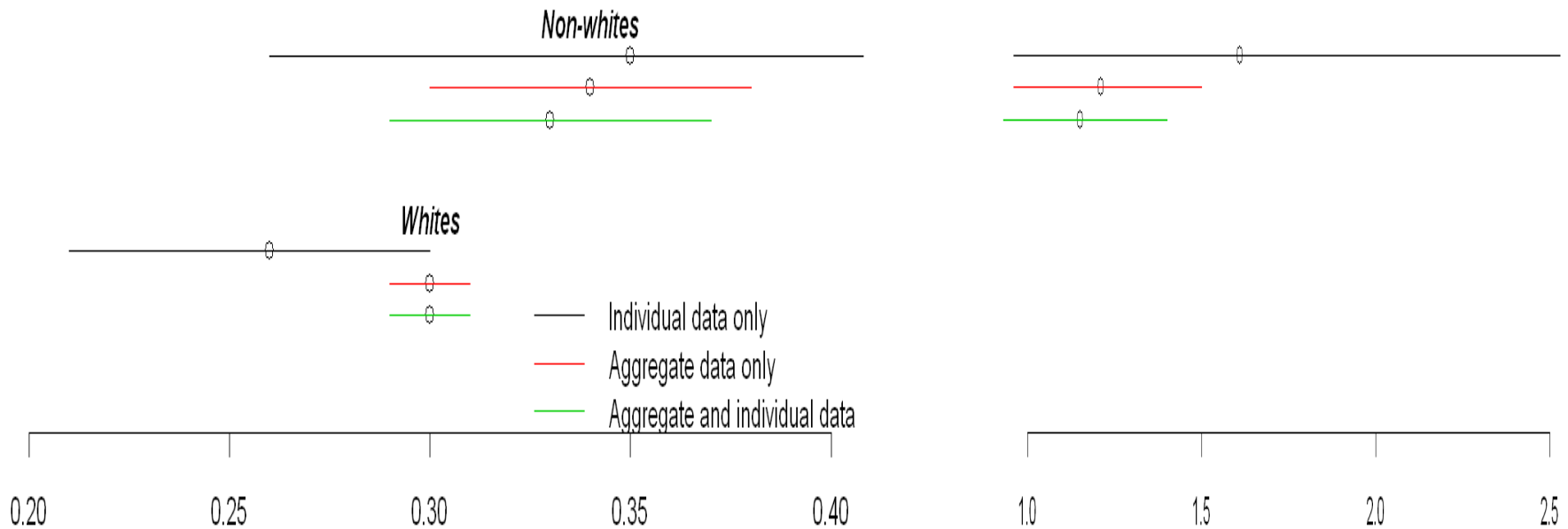
Or look at running means using the **Summary Monitor Tool** dialog box



Comparison of estimates of IE model

Probability of voting Labour

Odds ratio



Flexibility of modelling assumptions in WinBUGS

- Random intercepts model

$\text{logit}(\text{pw}[i]) \leftarrow \alpha[i]$

$\text{logit}(\text{pn}[i]) \leftarrow \alpha[i] + \beta$

Aggregate model

$\text{logit}(\text{ip}[i]) \leftarrow \alpha[\text{group}[i]] + \beta * \text{inonwhite}[i]$

Individual model

- Random slopes model

$\text{logit}(\text{pw}[i]) \leftarrow \alpha[i]$

$\text{logit}(\text{pn}[i]) \leftarrow \alpha[i] + \beta[i]$

$\text{logit}(\text{ip}[i]) \leftarrow \alpha[\text{group}[i]] + \beta[\text{group}[i]] * \text{inonwhite}[i]$

Flexibility of modelling assumptions in WinBUGS

- Survey design issues

- ◆ Non-response bias – different intercept for individual level model

$$\text{logit}(ip[i]) \leftarrow \text{delta} + \text{alpha}[\text{group}[i]] + \text{beta} * \text{inonwhite}[i]$$

- ◆ Cluster sampling

$$\text{logit}(ip[i]) \leftarrow \text{alpha}[\text{group}[i]] + \text{beta} * \text{inonwhite}[i] + \text{ward}[i]$$

- Spatial random effect

$$\text{alpha}[i] = U[i] + S[i]$$

$$U[i] \sim N(\text{alpha}0, \text{tau}.U)$$

$$S[1:N\text{areas}] \sim \text{car.normal}(\text{adj}[], \text{weights}[], \text{num}[], \text{tau}.S)$$

Flexibility of modelling assumptions in WinBUGS

■ Additional covariates

- ◆ Include a contextual effect to account for aggregation bias

$$\text{logit}(pw[i]) \leftarrow \alpha[i] + \text{gamma} * \text{nonwhite}[i]$$

$$\text{logit}(pn[i]) \leftarrow \alpha[i] + \text{beta} + \text{gamma} * \text{nonwhite}[i]$$

- ◆ Include another categorical individual-level covariate, for instance social class (defined as manual or non-manual)

- ◆ Now we need to know the full cross-classification of covariates, or at least a reasonable estimate of it

$$p[i] \leftarrow \text{phi00}[i]*p00[i] + \text{phi01}[i]*p01[i] + \text{phi10}[i]*p10[i] + \text{phi11}[i]*p11[i]$$

$$\text{logit}(p00[i]) \leftarrow \alpha[i]$$

$$\text{logit}(p01[i]) \leftarrow \alpha[i] + \text{gamma}$$

$$\text{logit}(p10[i]) \leftarrow \alpha[i] + \text{beta}$$

$$\text{logit}(p11[i]) \leftarrow \alpha[i] + \text{beta} + \text{gamma}$$

Probability of being non-white and a manual worker

Probability of a non-white manual worker voting Labour

Summary

	Random intercept	Random slopes	Calculate probabilities	Calculate odds ratios
Ecoreg	Y	N	N	Y
RxCeCollnf	N	Y	Y	Y
WinBUGS	Y	Y	Y	Y

	Another categorical variable	Another continuous variable	Contextual effects	Bayesian
Ecoreg	Y	Y	Y	N
RxCeCollnf	Y	N	N	Y
WinBUGS	Y	Y	Y	Y