2016 SISG MODULE 17: Bayesian Statistics for Genetics
Lecture 10: Imputation and Model Comparison

Jon Wakefield

Departments of Statistics and Biostatistics
University of Washington

Outline

Methods for Imputation

Model Comparison

Conclusions

Introduction

- In this lecture we consider three topics.
- First, we consider methods for imputation of missing genotypes.
- We describe a number of the more common Bayesian approaches to this problem.
- Second, we will briefly review a number of procedures to carry out model comparison.

Motivation for Imputation

- Imputation is the prediction of missing genotypes.
- Imputation is used in both GWAS and in fine-mapping studies.
- The technique is becoming increasingly popular since it can:
  - Increase power in GWAS.
  - Facilitate meta-analysis in which it is required to combine information from different panels which have different sets of SNPs. In this way power can be increased.
  - Fine-map causal variants, see Figure 1. Imputed SNPs that show large associations can be better candidates for replication studies.
- The key idea in the approaches we describe is the use of data on haplotypes from a relevant population to build a prior model for the missing data, basically the models leverage linkage disequilibrium.

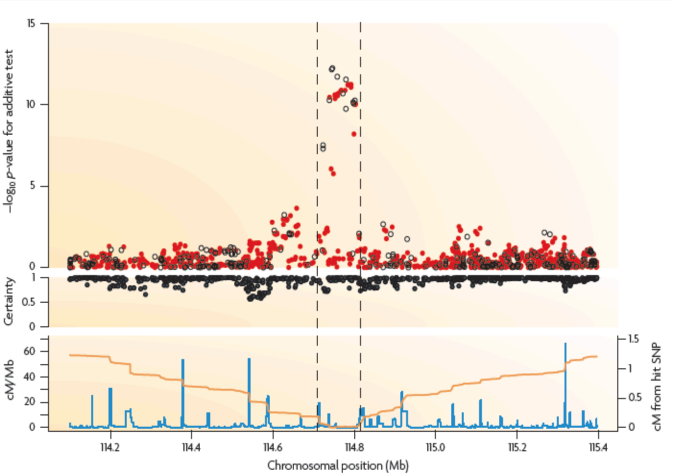
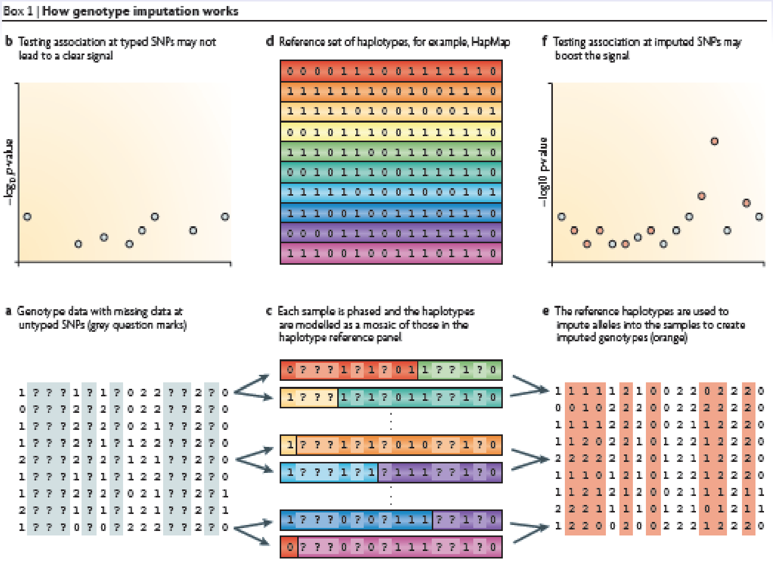


Figure 1 : Imputation for the TCF7L2 gene, from Marchini *et al.* (2007). Imputed SNP signals are in red and observed SNPs in black.



Box 1 | How genotype imputation works

b Testing association at typed SNPs may not lead to a clear signal

d Reference set of haplotypes, for example, HapMap

```
0 0 0 0 1 1 1 0 0 1 1 1 1 1 1 0
1 1 1 1 1 1 1 0 0 1 0 0 1 1 1 0
1 1 1 1 1 0 1 0 0 1 0 0 0 1 0 1
0 0 1 0 1 1 1 0 0 1 1 1 1 1 1 0
1 1 1 0 1 1 0 0 1 1 1 0 1 1 1 0
0 0 1 0 1 1 1 0 0 1 1 1 1 1 1 0
1 1 1 1 1 0 1 0 0 1 0 0 0 1 0 1
1 1 1 0 0 1 0 0 1 1 1 0 1 1 1 0
0 0 0 0 1 1 1 0 0 1 1 1 1 1 1 0
1 1 1 0 0 1 0 0 1 1 1 0 1 1 1 0
```

f Testing association at imputed SNPs may boost the signal

a Genotype data with missing data at untyped SNPs (grey question marks)

```
1 ? ? ? 1 ? 1 ? 0 2 2 ? ? 2 ? 0
0 ? ? ? 2 ? 2 ? 0 2 2 ? ? 2 ? 0
1 ? ? ? 2 ? 2 ? 0 2 1 ? ? 2 ? 0
1 ? ? ? 2 ? 1 ? 1 2 2 ? ? 2 ? 0
2 ? ? ? 2 ? 2 ? 1 2 1 ? ? 2 ? 0
1 ? ? ? 1 ? 1 ? 1 2 2 ? ? 2 ? 0
1 ? ? ? 2 ? 2 ? 0 2 1 ? ? 2 ? 1
2 ? ? ? 1 ? 1 ? 1 2 1 ? ? 2 ? 1
1 ? ? ? 0 ? 0 ? 2 2 2 ? ? 2 ? 0
```

c Each sample is phased and the haplotypes are modelled as a mosaic of those in the haplotype reference panel

```
0 ? ? ? 1 ? 1 ? 0 1 1 ? ? 1 ? 0
1 ? ? ? 1 ? 1 ? 0 1 1 ? ? 1 ? 0
        ⋮
1 ? ? ? 1 ? 1 ? 0 1 0 ? ? 1 ? 0
1 ? ? ? 1 ? 1 ? 1 1 1 ? ? 1 ? 0
        ⋮
1 ? ? ? 0 ? 0 ? 1 1 1 ? ? 1 ? 0
0 ? ? ? 0 ? 0 ? 1 1 1 ? ? 1 ? 0
```

e The reference haplotypes are used to impute alleles into the samples to create imputed genotypes (orange)

```
1 1 1 1 1 2 1 0 0 2 2 0 2 2 2 0
0 0 1 0 2 2 2 0 0 2 2 2 2 2 2 0
1 1 1 1 2 2 2 0 0 2 1 1 2 2 2 0
1 1 2 0 2 2 1 0 1 2 2 1 2 2 2 0
2 2 2 2 2 1 2 0 1 2 1 1 2 2 2 0
1 1 1 0 1 2 1 0 1 2 2 1 2 2 2 0
1 1 2 1 2 1 2 0 0 2 1 1 1 2 1 1
2 2 2 1 1 1 1 0 1 2 1 0 1 2 1 1
1 2 2 0 0 2 0 0 2 2 2 1 2 2 2 0
```

Figure 2 : Imputation overview from Marchini and Howie (2010).

The Statistical Framework

- Suppose we wish to estimate the association between a phenotype and m genetic markers in n individuals.
- Let G_{ij} represent the genotype of individual i at SNP j with G_{ij} unobserved for some SNPs.
- We consider diallelic SNPs so that G_{ij} can take the value 0, 1 or 2 depending on whether the pair of constituent SNPs are $\{0, 0\}$, $\{0, 1\}$, $\{1, 0\}$ or $\{1, 1\}$.
- If G_{ij} is observed then for SNP j we simply model

$$p(y_i | G_{ij})$$

- For example, if the phenotype y_i is continuous, we might assume a normal model:

$$\mathrm{E}[Y_i] = \beta_0 + \beta_1 G_{ij},$$

  and if y_i is binary, a logistic model is an obvious candidate:

$$\frac{p_i}{1 - p_i} = \exp(\beta_0 + \beta_1 G_{ij})$$

  where p_i is the probability of disease for individual i.

The Statistical Framework

- Let $\mathbf{H} = (\mathbf{H}_1, ..., \mathbf{H}_N)$ represent haplotype information at m SNPs in a relevant reference-panel, with N distinct haplotypes.
- Let $\mathbf{G}_i$ be the observed genotype information for individual i.
- If G_{ij} is unobserved then for SNP j we have the model

$$p(y_i|\mathbf{H}, \mathbf{G}_i) = \sum_{k=0}^{2} p(y_i|G_{ij} = k) \times \Pr(G_{ij} = k|\mathbf{H}, \mathbf{G}_i)$$

- The big question is how to obtain the predictive distribution

$$\Pr(G_{ij} = k|\mathbf{H}, \mathbf{G}_i).$$

- A common approach is to take as prior a Hidden Markov Model (HMM).
- We digress to discuss HMMs.

Hidden Markov Models

- Example: Poisson Time Series A common problem is how to model count data over time. A Poisson model is the obvious choice but how to introduce:
  1. overdispersion and
  2. dependence over time.
- Consider the model:

  Stage 1: $Y_t|\lambda_t \sim \text{Poisson}(\lambda_t),\ t = 1, 2, \ldots$

  Stage 2: $\lambda_t|Z_t \sim_{iid} \begin{cases} \lambda_0 & \text{if } Z_t = 0 \\ \lambda_1 & \text{if } Z_t = 1 \end{cases}$

  Stage 3: $Z_t|p \sim_{iid} \text{Bernoulli}(p)$.
- An alternative model replaces *Stage 3* with a (first-order) Markov chain model, i.e, $\Pr(Z_t|Z_1, \ldots, Z_{t-1}) = \Pr(Z_t|Z_{t-1})$:

$$\begin{aligned} \Pr(Z_t = 0|Z_{t-1} = 0) &= p_0 \\ \Pr(Z_t = 1|Z_{t-1} = 1) &= p_1 \end{aligned}$$

- Z_t is an unobserved (hidden) state.
- As an example we consider the number of major earthquakes (magnitude 7 and above) for the years 1990–2006.
- We illustrate the fit of this model with two or three underlying states.

Example: Earthquake Data

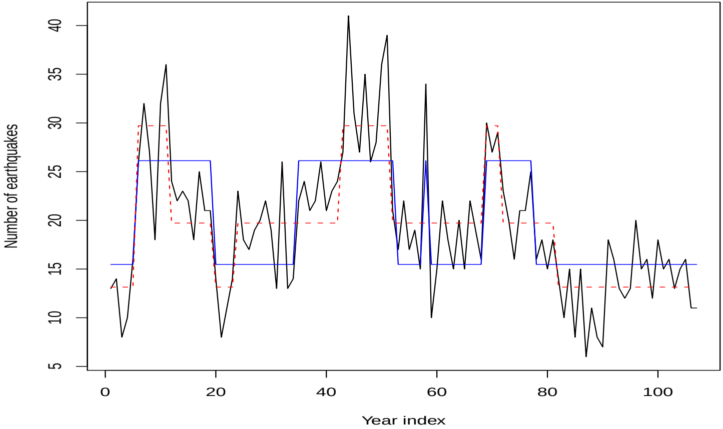


Figure 3 : The earthquake data along with the underlying states for the two and three state HMMs, in blue and red, respectively.

IMPUTE v1

- Marchini *et al.* (2007) consider a HMM for the vector of genotypes for individual i:

$$\Pr(\mathbf{G}_i|\mathbf{H},\theta,\rho) = \sum_{\mathbf{Z}_i=(\mathbf{Z}_i^{(1)},\mathbf{Z}_i^{(2)})} \Pr(\mathbf{G}_i|\mathbf{Z}_i,\theta) \times \Pr(\mathbf{Z}_i|\mathbf{H},\rho)$$

  where $\mathbf{Z}_i^{(1)} = \{Z_{i1}^{(1)}, ..., Z_{iJ}^{(1)})$ and $\mathbf{Z}_i^{(2)} = \{Z_{i1}^{(2)}, ..., Z_{iJ}^{(2)})$.
- The $(\mathbf{Z}_i^{(1)}, \mathbf{Z}_i^{(2)})$ are the pair of haplotypes for SNP j from the reference panel that are copied to form the genotype vector. These are the hidden states.
- The term $\Pr(\mathbf{Z}_i|\mathbf{H},\rho)$ models how the pair of copied haplotypes for individual i changes along the sequence. This probability changes according to a Markov chain with the switching of states depending on the fine-scale recombination rate ρ.
- The term $\Pr(\mathbf{G}_i|\mathbf{Z},\theta)$ allows the observed genotypes to differ from the pair of copied haplotypes through mutation; the mutation parameter is θ.
- IMPUTE v2 (Howie *et al.*, 2009) is a more flexible version that alternates between phasing and haploid imputation.

fastPHASE and BIMBAM

- We describe the model of Scheet and Stephens (2006).
- A Hidden Markov Model (HMM) is used to determine $\Pr(G_{ij} = k | \boldsymbol{\alpha}, \boldsymbol{\theta}, r)$.
- The basic idea is that haplotypes tend to cluster into groups of similar haplotypes; suppose there are K clusters.
- The unobserved hidden state is the haplotype cluster from which this SNP arose from. Each cluster has an associated set of allele frequencies $\boldsymbol{\theta}_{kj}$.
- With K underlying states we have, for SNP j, α_{kj} being the probability of arising from haplotype k, with

$$\sum_{k=1}^{K} \alpha_{kj} = 1.$$

- The model is

$$\Pr(\mathbf{G}_i | \boldsymbol{\alpha}, \boldsymbol{\theta}, r) = \sum_{\mathbf{Z}} \Pr(\mathbf{G}_i | \mathbf{Z}_i, \boldsymbol{\theta}) \times \Pr(\mathbf{Z}_i | \boldsymbol{\alpha}, r)$$

  with Z_{ij} the haplotype of origin for individual i and SNP j.
- A Markov chain is constructed for Z_{ij} with the strength of dependence being based on the recombination rate r at a given location.
- Given $Z_{ij} = k$, the genotype assigned depends on the allele frequencies of the k-th haplotype at the j-th SNP.

Use in Association Studies

- The simplest approach to using imputed SNPs is to substitute $\widehat{G}_{ij}$ (a number between 0 and 2) into the phenotype association model.
- A set of probabilities $\Pr(G_{ij} = k|\mathbf{G}, \mathbf{H})$ for $k = 0, 1, 2$ are produced and these may be used to average over the uncertainty in the phenotype model.
- Within `BIMBAM` the unknown genotype is sampled from its posterior distribution, within an MCMC framework.
- Other approaches:
  - `MACH`: similar methodology to `IMPUTE` (Li *et al.*, 2010).
  - `Beagle`: uses a graphical model for haplotypes (Browning and Browning, 2009).

Practical Issues

- One may attempt to match the haplotype panel (e.g. from HapMAP 2) with the study individuals.
- An alternative approach is to use all available haplotypes, and assigning equal prior probabilities to each.
- Many studies, for example Huang *et al.* (2009), have examined SNP imputation accuracy in different populations.

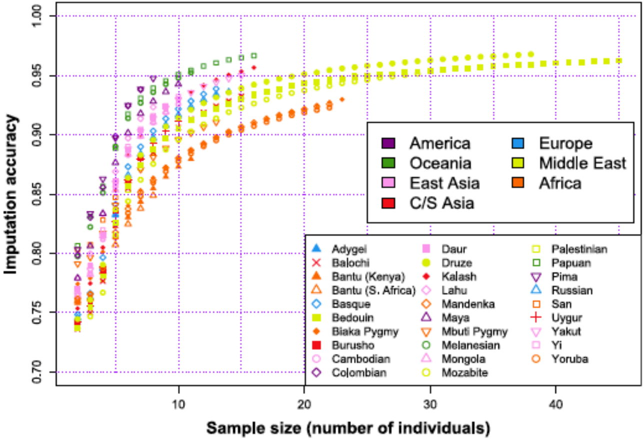


Figure 4 : Imputation accuracy as a function of sample size, from Huang *et al.* (2009).

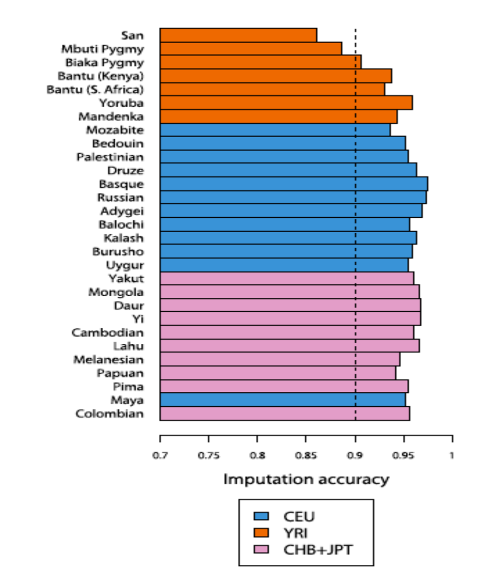


Figure 5 : Imputation accuracy for different populations with a reference-panel of 120 haplotypes. From Huang *et al.* (2009).

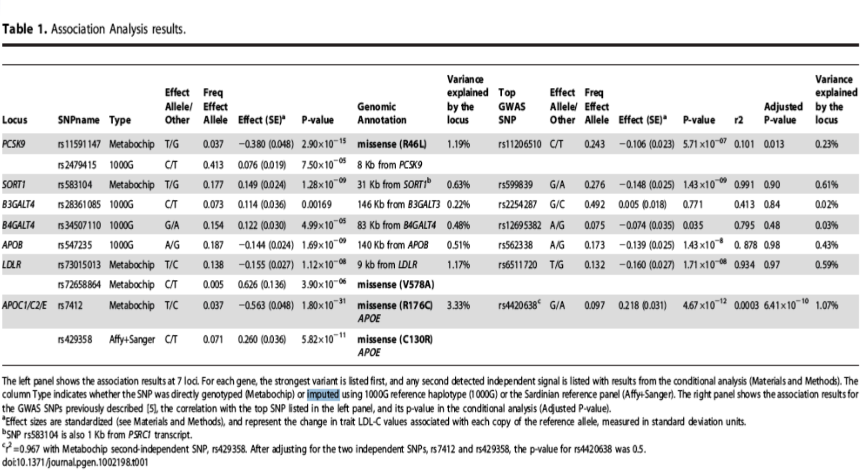


Table 1. Association Analysis results.

Locus	SNPname	Type	Effect Allele/ Other	Freq Effect Allele	Effect (SE)[a]	P-value	Genomic Annotation	Variance explained by the locus	Top GWAS SNP	Effect Allele/ Other	Freq Effect Allele	Effect (SE)[a]	P-value	r2	Adjusted P-value	Variance explained by the locus
PCSK9	rs11591147	Metabochip	T/G	0.037	−0.380 (0.048)	2.90×10^{-15}	**missense (R46L)**	1.19%	rs11206510	C/T	0.243	−0.106 (0.023)	5.71×10^{-07}	0.101	0.013	0.23%
	rs2479415	1000G	C/T	0.413	0.076 (0.019)	7.50×10^{-05}	8 Kb from *PCSK9*									
SORT1	rs583104	Metabochip	T/G	0.177	0.149 (0.024)	1.28×10^{-09}	31 Kb from *SORT1*[b]	0.63%	rs599839	G/A	0.276	−0.148 (0.025)	1.43×10^{-09}	0.991	0.90	0.61%
B3GALT4	rs28361085	1000G	C/T	0.073	0.114 (0.036)	0.00169	146 Kb from *B3GALT3*	0.22%	rs2254287	G/C	0.492	0.005 (0.018)	0.771	0.413	0.84	0.02%
B4GALT4	rs34507110	1000G	G/A	0.154	0.122 (0.030)	4.99×10^{-05}	83 Kb from *B4GALT4*	0.48%	rs12695382	A/G	0.075	−0.074 (0.035)	0.035	0.795	0.48	0.03%
APOB	rs547235	1000G	A/G	0.187	−0.144 (0.024)	1.69×10^{-09}	140 Kb from *APOB*	0.51%	rs562338	A/G	0.173	−0.139 (0.025)	1.43×10^{-8}	0. 878	0.98	0.43%
LDLR	rs73015013	Metabochip	T/C	0.138	−0.155 (0.027)	1.12×10^{-08}	9 kb from *LDLR*	1.17%	rs6511720	T/G	0.132	−0.160 (0.027)	1.71×10^{-08}	0.934	0.97	0.59%
	rs72658864	Metabochip	C/T	0.005	0.626 (0.136)	3.90×10^{-06}	**missense (V578A)**									
APOC1/C2/E	rs7412	Metabochip	T/C	0.037	−0.563 (0.048)	1.80×10^{-31}	**missense (R176C)** *APOE*	3.33%	rs4420638[c]	G/A	0.097	0.218 (0.031)	4.67×10^{-12}	0.0003	6.41×10^{-10}	1.07%
	rs429358	Affy+Sanger	C/T	0.071	0.260 (0.036)	5.82×10^{-11}	**missense (C130R)** *APOE*									

The left panel shows the association results at 7 loci. For each gene, the strongest variant is listed first, and any second detected independent signal is listed with results from the conditional analysis (Materials and Methods). The column Type indicates whether the SNP was directly genotyped (Metabochip) or imputed using 1000G reference haplotype (1000G) or the Sardinian reference panel (Affy+Sanger). The right panel shows the association results for the GWAS SNPs previously described [5], the correlation with the top SNP listed in the left panel, and its p-value in the conditional analysis (Adjusted P-value).

[a] Effect sizes are standardized (see Materials and Methods), and represent the change in trait LDL-C values associated with each copy of the reference allele, measured in standard deviation units.

[b] SNP rs583104 is also 1 Kb from *PSRC1* transcript.

[c] r^2=0.967 with Metabochip second-independent SNP, rs429358. After adjusting for the two independent SNPs, rs7412 and rs429358, the p-value for rs4420638 was 0.5.

doi:10.1371/journal.pgen.1002198.t001

Figure 6 : Example from Sanna *et al.* (2011). Imputation carried out using the `MACH` software.

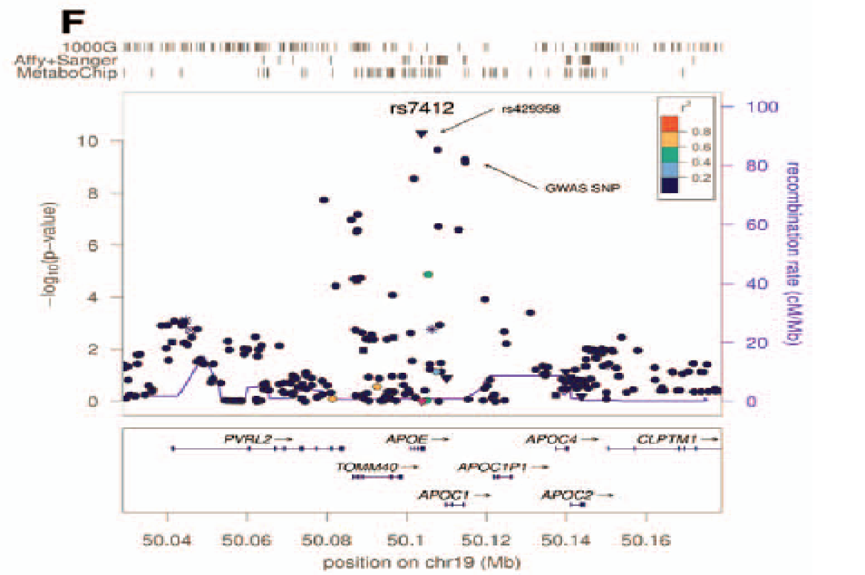


Figure 7 : Example from Sanna *et al.* (2011).

Model Comparison

- Markov chain Monte Carlo in particular has allowed the fitting of more and more complex models, often hierarchical in nature with layers of random effects.
- The search for a method to find the “best” of a set of candidate models has also grown.
- Let $p(\mathbf{y}|\boldsymbol{\theta})$ represent a generic likelihood for $\mathbf{y} = [y_1, \ldots, y_n]$ and let

  $$D(\boldsymbol{\theta}) = -2\log[p(\mathbf{y}|\boldsymbol{\theta})]$$

  represent the deviance.
- For example, in an iid normal$(\mu_i(\boldsymbol{\theta}), \sigma^2)$ normal the deviance is

  $$\frac{1}{\sigma^2}\sum_{i=1}^{n}[y_i - \mu_i(\boldsymbol{\theta})]^2.$$

- Frequentist model comparison for nested models is often carried out using likelihood ratio statistics, which corresponds to the comparison of deviances in generalized linear models (GLMs), see for example McCullagh and Nelder (1989).

Model Comparison: AIC

- One approach to model comparison is based on a model's ability to make good predictions.
- Such an objective, and predicting the actual observed data, leads to Akaike's an information criterion (AIC), derived in Akaike (1973).
- In AIC one tries to estimate the (Kullback-Leibler) distance between the true distribution of the data, and the modeled distribution of the data.
- AIC is given by

  $$\mathrm{AIC} = -2\log[p(y|\widehat{\boldsymbol{\theta}})] + 2k$$

  where $\widehat{\theta}$ is the MLE and k is the number of parameters in the model, i.e. the size of $\boldsymbol{\theta}$.
- Small values of the AIC are favored, since they suggest low prediction error.
- The penalty term $2k$ penalizes the double use of the data.
- In general for prediction: overly complex models are penalized since redundant parameters "use up" information in the data.

Model Comparison: BIC

- Another approach is based on trying to identify the “true” model.
- Schwarz (1978) developed the Bayesian Information Criterion (BIC) which is given by

$$\text{BIC} = -2\log[p(y|\widehat{\boldsymbol{\theta}})] + k\log n.$$

- BIC approximates $-2\log p(\mathbf{y}|\boldsymbol{\theta})$ under a certain unit information prior (Kass and Wasserman, 1995).
- BIC is consistent[1] for finding the true model, if that model lies in the set being compared.
- AIC is not consistent for finding the true model, but recall is intended for prediction.

[1] meaning the BIC hones in on the true model as the sample size increases

Model Comparison: DIC

- Spiegelhalter *et al.* (2002) introduced what has proved to be a very popular model comparison statistic, the deviance information criterion (DIC).
- To define the DIC, define an "effective number of parameters as

$$
\begin{aligned}
p_D &= E_{\theta|y}\{-2\log[p(\mathbf{y}|\boldsymbol{\theta})]\} + 2\log[p(\mathbf{y}|\overline{\boldsymbol{\theta}})] \\
&= \overline{D} + D(\overline{\boldsymbol{\theta}})
\end{aligned}
$$

  where $\overline{\boldsymbol{\theta}} = E[\boldsymbol{\theta}|\mathbf{y}]$ is the posterior mean, $D(\overline{\boldsymbol{\theta}})$ is the deviance evaluated at the posterior mean and $\overline{D} = E[D|\mathbf{y}]$.
- Hence, p_D is the

  posterior mean deviance − deviance of posterior means.

- The DIC is given by

$$
\begin{aligned}
\text{DIC} &= D(\overline{\boldsymbol{\theta}}) + 2p_D \\
&= \overline{D} + p_D,
\end{aligned}
$$

  so that we have a measure of goodness of fit + complexity.
- DIC is straightforward to evaluate using MCMC or INLA.

Model Comparison: DIC

DIC has been heavily criticized (Spiegelhalter *et al.*, 2014):

- p_D is not invariant to parameterization.
- DIC is not consistent for choosing the correct model.
- DIC has a weak theoretical justification and is not universally applicable.
- DIC has been shown to under penalize complex models (Plummer, 2008; Ando, 2007).
- See Spiegelhalter *et al.* (2014) for an interesting discussion of the history of DIC, including a summary of attempts to improve DIC.
- According to Google Scholar, as of June 20th, 2014, Spiegelhalter *et al.* (2002) has 5251 citations...

Model Comparison: CPO

- Another approach based on prediction uses the conditional predictive ordinate (CPO).
- Let
  $$\mathbf{y}_{-i} = [y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n]$$
  represent the vector of data with the i-th observation removed.
- The idea is to predict the density ordinate of the left-out observation, based on those that remain.
- Specifically, the CPO for observation i is defined as:

$$\begin{aligned} \text{CPO}_i &= p(y_i|\mathbf{y}_{-i}) \\ &= \int p(y_i|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{y}_{-i})\, d\boldsymbol{\theta} \\ &= E_{\theta|y_{-i}}\left[p(y_i|\boldsymbol{\theta})\right] \end{aligned}$$

Model Comparison: CPO

- The CPOs can be used to look at local fit, or one can define an overall score for each model:

$$\log(\text{CPO}) = \sum_{i=1}^{n} \log \text{CPO}_i.$$

- Good models will have relatively high values of log (CPO).
- See Held *et al.* (2010) for a discussion of shortcuts for estimation (i.e. avoidance of fitting the model n times) using MCMC and INLA.

Model Comparison: Illustration, Childhood Mortality in Tanzania

- We illustrate the use of CPO and DIC in a study of estimating childhood (under 5) mortality in regions of Tanzania.
- The data are collected via a series of 8 surveys in 21 regions covering the period 1980–2009.
- Let q_{its} be the childhood mortality in area i, at time point t from survey s.
- Based on the surveys we can obtain weighted (Horvitz-Thompson) estimators $\widehat{q}_{its}$ with associated asymptotic variances V_{its}.
- We summarize the data via logit estimates

$$y_{its} = \log \left(\frac{\widehat{q}_{its}}{1 - \widehat{q}_{its}} \right).$$

- Let

$$\phi_{its} = \log \left(\frac{q_{its}}{1 - q_{its}} \right)$$

represent the logit of the childhood mortality.

Model Comparison: Illustration, Childhood Mortality in Tanzania

We have a three-stage hierarchical model:

- Stage 1: Likelihood:

$$y_{its} | \phi_{its} \sim \text{normal}(\phi_{its}, V_{its}).$$

and we compare the following six models:

$$\begin{aligned}
\text{Model 1: } \phi_{its} &= \mu + \alpha_t + \gamma_t + \theta_i + \eta_i + \delta_{it} \\
\text{Model 2: } \phi_{its} &= \mu + \alpha_t + \gamma_t + \theta_i + \eta_i + \delta_{it} + \nu_s \\
\text{Model 3: } \phi_{its} &= \mu + \alpha_t + \gamma_t + \theta_i + \eta_i + \delta_{it} + \nu_s + \nu_{is} \\
\text{Model 4: } \phi_{its} &= \mu + \alpha_t + \gamma_t + \theta_i + \eta_i + \delta_{it} + \nu_s + \nu_{ts} \\
\text{Model 5: } \phi_{its} &= \mu + \alpha_t + \gamma_t + \theta_i + \eta_i + \delta_{it} + \nu_s + \nu_{ts} + \nu_{is} \\
\text{Model 6: } \phi_{its} &= \mu + \alpha_t + \gamma_t + \theta_i + \eta_i + \delta_{it} + \nu_s + \nu_{ts} + \nu_{is} + \nu_{its}
\end{aligned}$$

where α_t, θ_i, δ_{it} are independent random effects for time, area and the interaction, γ_t and η_i are random effects that carry out local smoothing in time and space and $\nu_s, \nu_{ts}, \nu_{is}, \nu_{its}$ are independent random effects to reflect survey effects.

- Stage 2: Normal random effects Distributions.
- Stage 3: Hyperpriors on μ and the random effects variances.

Model Comparison: Illustration, Childhood Mortality in Tanzania

Table 1 : Model comparison statistics for 6 models for the Tanzania data; "best" in red.

Model	No. Parameters	p_D	$\bar{D}$	DIC	log(CPO)
2	181	75	409	484	-295
2	189	81	382	463	-288
3	313	120	219	339	-193
4	223	91	364	454	-282
5	347	128	202	330	-182
6	920	149	185	334	-184

- Notice how much smaller the effective number of parameters is, when compared with the total number of parameteres; this is because of the shrinkage/penalization of the random effects distributions.
- Both CPO and DIC suggest that model 5 is the best:

$$\text{Model 5: } \phi_{its} = \mu + \alpha_t + \gamma_t + \theta_i + \eta_i + \delta_{it} + \nu_s + \nu_{ts} + \nu_{is}$$

- So survey effects vary across time and across areas (different teams sent out).

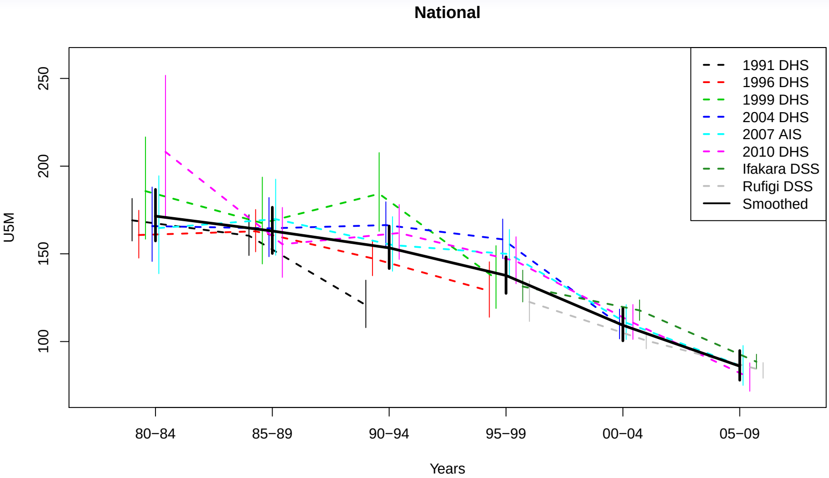


Figure 8 : Smoothed estimates of national under 5 mortality in Tanzania (solid line) per 1000 births, different surveys denoted with dashed lines and vertical lines represent 95% interval estimates.

Conclusions

- Hierarchical models allow complex dependencies within data to be modeled.
- Prior specification for variance components is not straightforward, and sensitivity analysis is a good idea.
- No universally agreed upon approach to carrying out model comparison.]item The Widely Applicable Information Criteria (WAIC) is growing in popularity (Watanabe, 2013; Gelman *et al.*, 2014).

References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In P. B.N. and C. F., editors, *Second International Symposium on Information Theory*, pages 267–281. Akademia Kiadó, Budapest.

Ando, T. (2007). Bayesian predictive information criterion for the evaluation of hierarchical bayesian and empirical bayes models. *Biometrika*, **94**, 443–458.

Browning, B. and Browning, S. (2009). A unified approach to genotype imputation and haplotype-based inference for large data sets of trios and unrelated individuals. *The American Journal of Human Genetics*, **84**, 1084–1097.

Gelman, A., Hwang, J., and Vehtari, A. (2014). Understanding predictive information criteria for bayesian models. *Statistics and Computing*, **24**, 997–1016.

Held, L., Schrödle, B., and Rue, H. (2010). Posterior and cross-validatory predictive checks: A comparison of MCMC and INLA. In T. Kneib and G. Tutz, editors, *Statistical Modeling and Regression Structures – Festschrift in Honour of Ludwig Fahrmeir*, pages 91–110. Physica-Verlag.

Howie, B., Donnelly, P., and Marchini, J. (2009). A flexible and accurate genotype imputation method for the next generation of genome-wide association studies. *PLoS Genetics*, **5**, e1000529.

Huang, L., Yun, L., Singleton, A., Hardy, J., Abecasis, G., Rosenberg, N., and Scheet, P. (2009). Genotype-imputation accuracy across worldwide human populations. *The American Journal of Human Genetics*, **84**, 235–250.

Kass, R. and Wasserman, L. (1995). A reference Bayesian test for nested hypotheses and its relationship to the schwarz criterion. *Journal of the American Statistical Association*, **90**, 928–934.

Li, Y., Willer, C., Fing, J., Scheet, P., and Abecasis, G. (2010). Mach: using sequence and genotype data to estimate haplotypes and unobserved genotypes. *Genetic Epidemiology*, **34**, 816–834.

Marchini, J. and Howie, B. (2010). Genotype imputation for genome-wide association studies. *Nature Reviews Genetics*, **11**, 499–511.

Marchini, J., Howie, B., Myers, S., McVean, G., and Donnelly, P. (2007). A new multipoint method for genome-wide association studies by imputation of genotypes. *Nature Genetics*, **39**, 906–913.

McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models, Second Edition*. Chapman and Hall, London.

Plummer, M. (2008). Penalized loss functions for Bayesian model comparison. *Biostatistics*, **9**, 523–539.

Sanna, S., Li, B., and Mulas, A. (2011). Fine mapping of five loci associated with low density lipoprotein choesterol fetects variants that double the explained heritability. *PLoS Genetics*, **7**, 1002198.

Scheet, P. and Stephens, M. (2006). A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase. *The American Journal of Human Genetics*, **78**, 629–644.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, **6**, 461–464.

Spiegelhalter, D., Best, N., Carlin, B., and Linde, A. V. D. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B*, **64**, 583–639.

Spiegelhalter, D., Best, N., Carlin, B., and Linde, A. V. D. (2014). The deviance information criterion: 12 years on (with discussion). *Journal of the Royal Statistical Society: Series B*, **64**, 485–493.

Watanabe, S. (2013). A widely applicable Bayesian information criterion. *Journal of Machine Learning Research*, **14**, 867–897.

Bayesian Statistics for Genetics
Lecture 10: Software

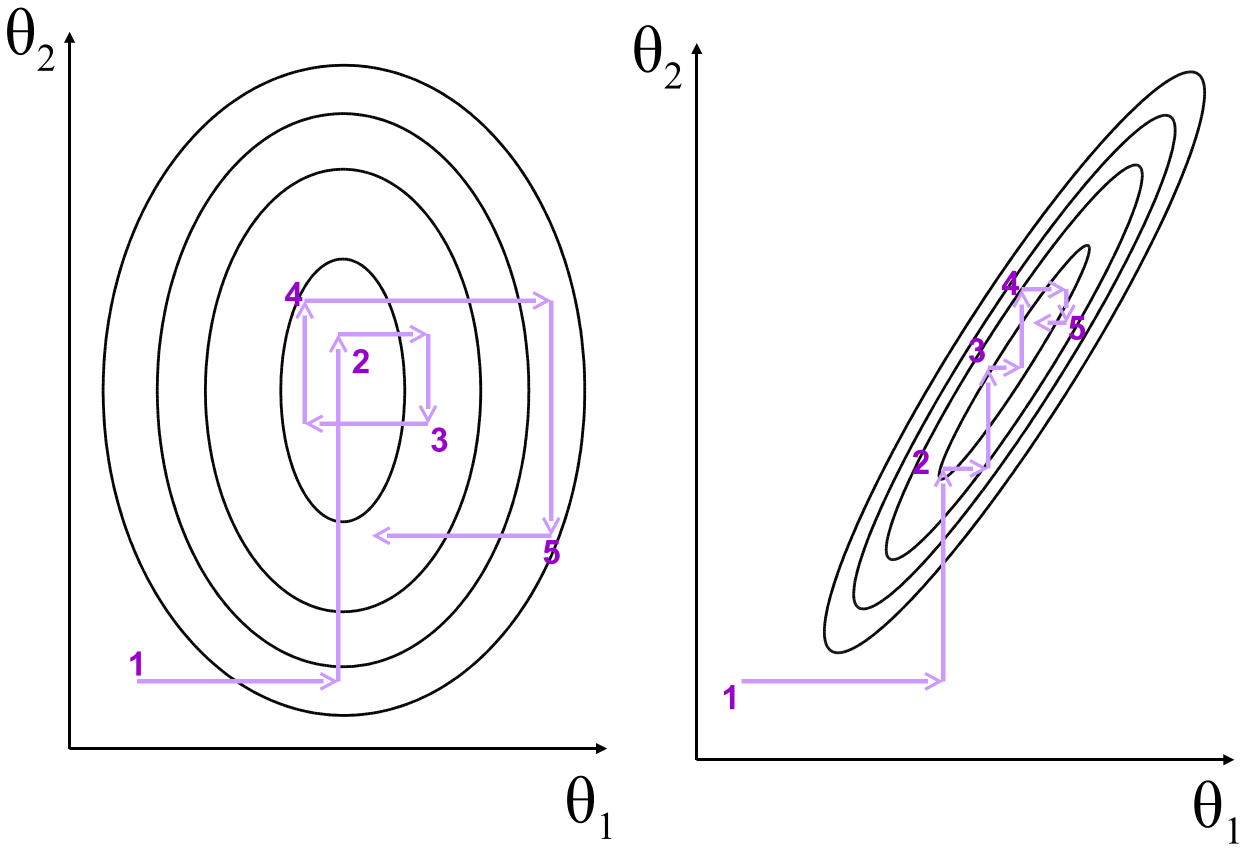
Ken Rice

UW Dept of Biostatistics

July, 2016

Off-the-shelf MCMC

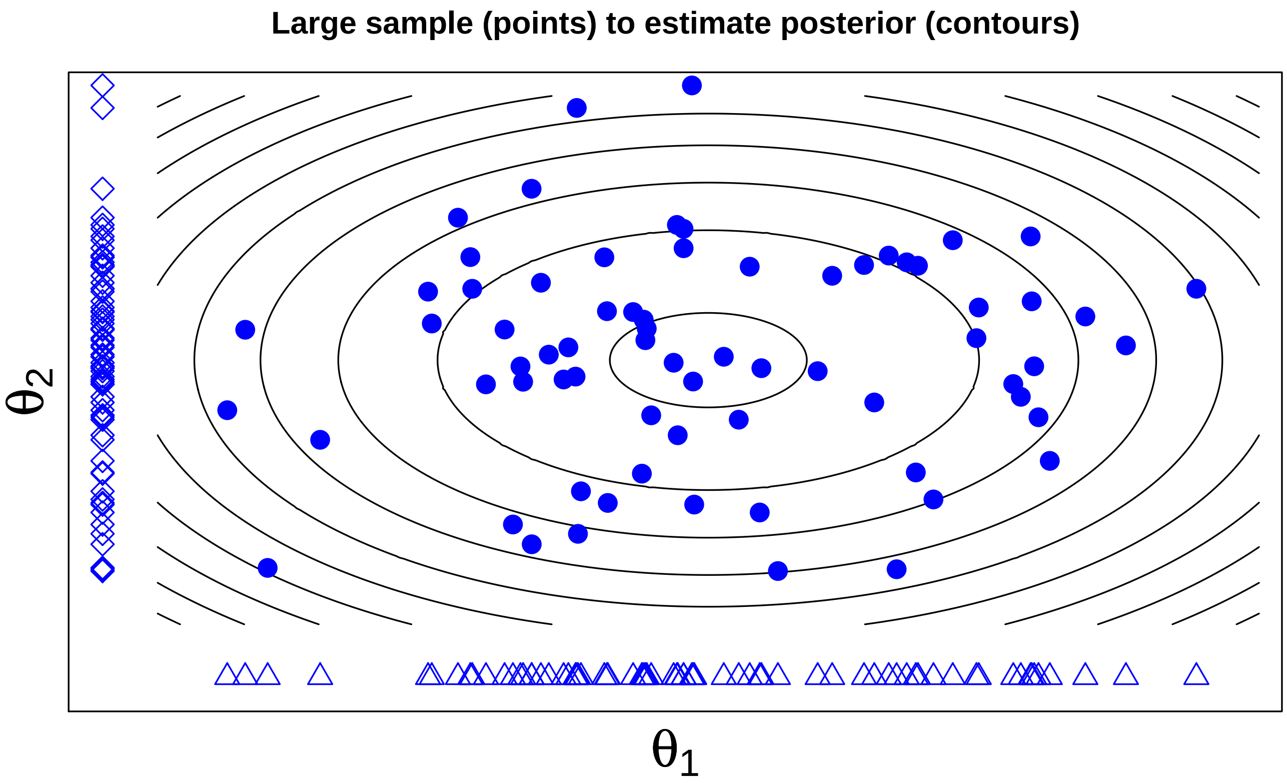
Recall the big picture(s) of Markov Chain Monte Carlo;



These show the *Gibbs Sampler*.

Off-the-shelf MCMC

Recall the big picture(s) of Markov Chain Monte Carlo;



However we get there, we want a large sample from the posterior distribution.

Off-the-shelf MCMC

Markov Chain Monte Carlo (MCMC) is the general term for sampling methods that use Markov Chain processes* to 'explore' the parameter space; the (many) random process values form our approximation of the posterior.

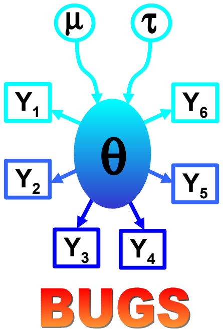
But in many settings this 'walking around' is mundane; once we specify the model and priors, the process of getting samples from the posterior **can** be done with no original thought – i.e. we can get a computer to do it.

Some example of this labor-saving approach;

- WinBUGS (next)
- ... or JAGS, OpenBUGS, NIMBLE and Stan
- INLA – a competitor to MCMC

The R Task Views on Genetics and Bayesian methods may also have specialized software; see also Bioconductor

WinBUGS



Started in 1989, the **B**ayesian analysis **U**sing **G**ibbs **S**ampling (BUGS) project has developed software where users specify only model and prior – everything else is internal. WinBUGS is the most comprehensive version.

- The model/prior syntax is very similar to `R`
- ... with some annoying wrinkles – variance/precision, also column major ordering in matrices
- Can be ‘called’ from `R` – see e.g. `R2WinBUGS`, but you still need to code the model

Child cancers 'not caused by Sellafield'

Before we try it on genetic data, a tiny example $(n = 1, Y = 4)$;

$$\begin{aligned} Y|\theta &\sim \text{Pois}\,(E\exp(\theta)) \\ \theta &\sim N(0, 1.797^2) \\ E &= 0.25 \end{aligned}$$

WinBUGS

One (sane) way to code this in the BUGS language;

```
model{
   Y~dpois(lambda)            ...Poisson distribution, like R
   lambda <- E*exp(theta)     ...syntax follows R
   E <- 0.25                  ...constants could go in data
   theta~dnorm(m,tau)         ...prior for θ
   m <- 0
   tau <- 1/v                 tau = precision NOT variance!
   v <- 1.797*1.797
 }                            ...finish the model

#data
list(Y=4)                     Easiest way to input data
#inits
list(theta=0)                 Same list format; or use gen.inits
```

WinBUGS

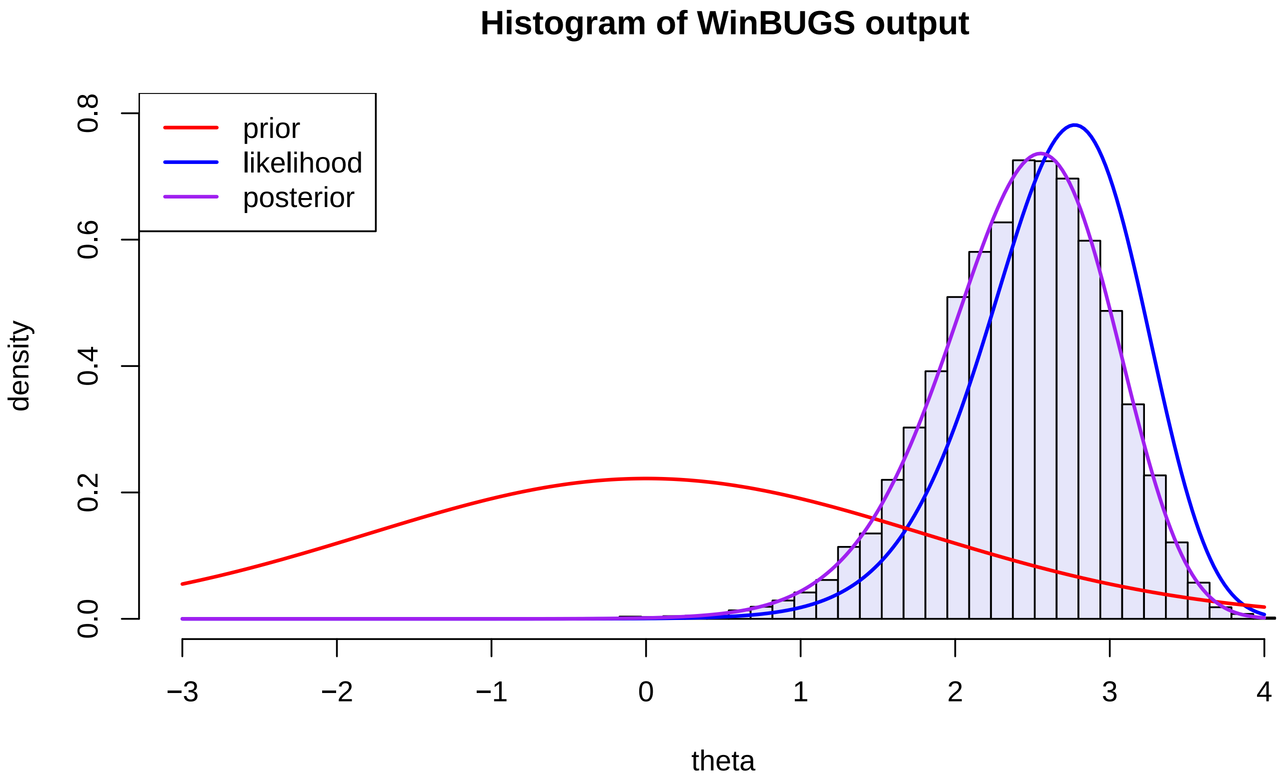
Notes on all this; (not a substitute for reading the manual!)

- This should look somewhat familiar, from the models we have been writing out. In particular '$\sim$' is used to denote distributions of data *and* parameters
- All 'nodes' appear **once** on the LHS; hard work is done on RHS
- No formulae allowed when specifying distributions
- Data nodes *must* have distributions. Non-data nodes *must* have priors – it's easy to forget these
- Write vector math 'by hand'; `beta0 + beta1*x1 + ...`
- This language can't do everything; BUGS does not allow e.g.

```
Y <- U + V
U~dnorm(meanu,tauu); V~dt(meanv,tauv,k)
#data
list(Y=...)
```

WinBUGS

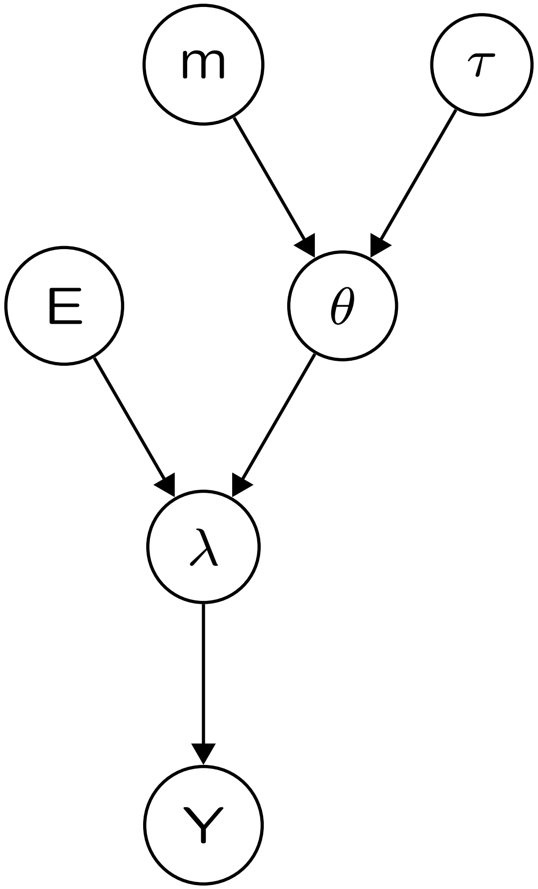
From 10,000 iterations, how do we do? (Note 'MC error' estimates Monte Carlo error in the posterior mean)



node	mean	sd	MC error	2.5%	median	97.5%
`theta`	2.422	0.5608	0.005246	1.229	2.466	3.388

WinBUGS

Under the hood, here's how WinBUGS 'thinks';



- It's a DAG; arrows represent stochastic relationships (not causality)
- Some texts use square nodes for observed variables (Y, here)
- To do a Gibbs update, we need to know/work out the distribution of a node conditional on **only** its parents, children, and its children's other parents*.

* *This set is a node's 'Markov blanket'. The idea saves a lot of effort, and is particularly useful when fitting random effects models.*

WinBUGS: HWE example

A multinomial example, with a default prior;

$$
\begin{aligned}
\mathbf{Y} &\sim \text{Multinomial}(n, \boldsymbol{\theta}) \\
\text{where } \boldsymbol{\theta} &= (p^2, 2p(1-p), (1-p)^2) \\
p &\sim \text{Beta}(0.5, 0.5).
\end{aligned}
$$

A typical way to code it in “the BUGS language”;

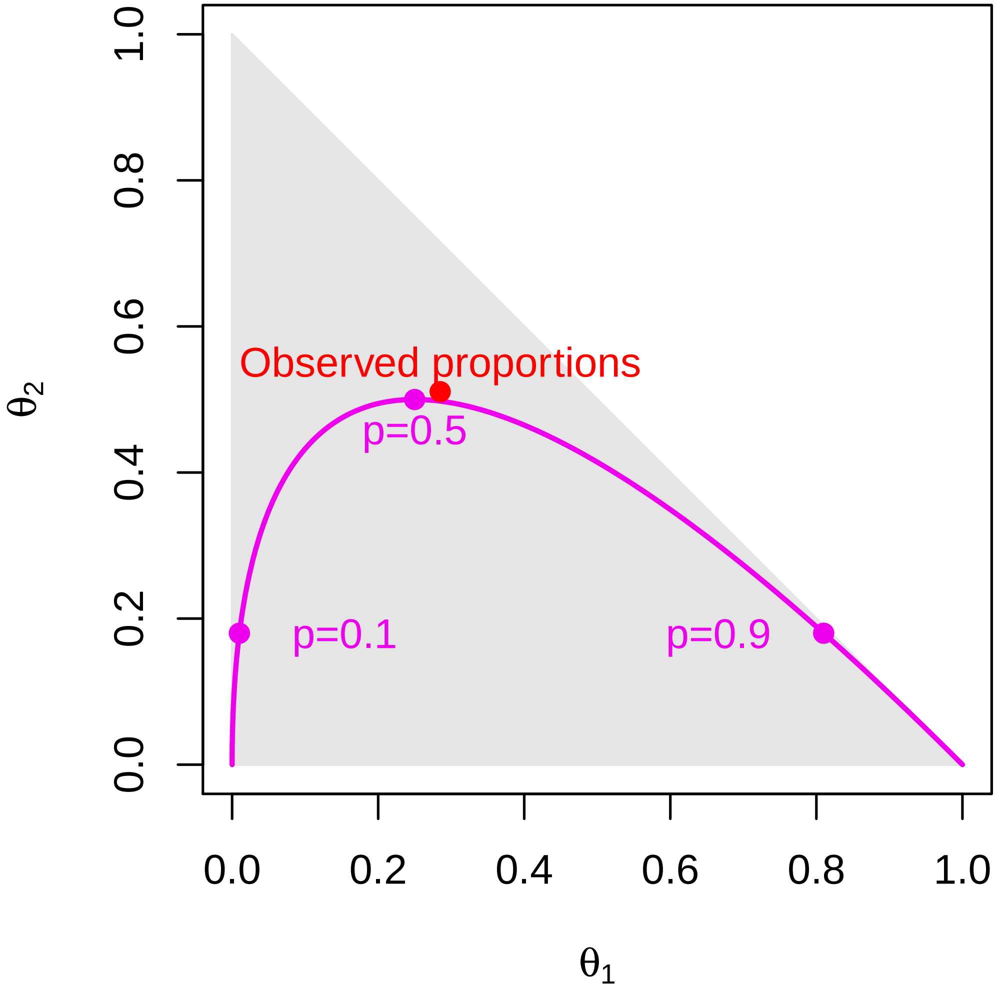
```
model{
   y[1:3] ~ dmulti(theta[], n)
   theta[1] <- p*p
   theta[2] <- 2*p*(1-p)
   theta[3] <- (1-p)*(1-p)
   p ~ dbeta(0.5, 0.5)
}
```

WinBUGS: HWE example

We have $n = 186$, and $\mathbf{Y} = (53, 95, 38)$.

We will run 3 chains, starting at $p = 0.5, 0.1$ and 0.9.

In WinBUGS, input these by highlighting `list` objects:

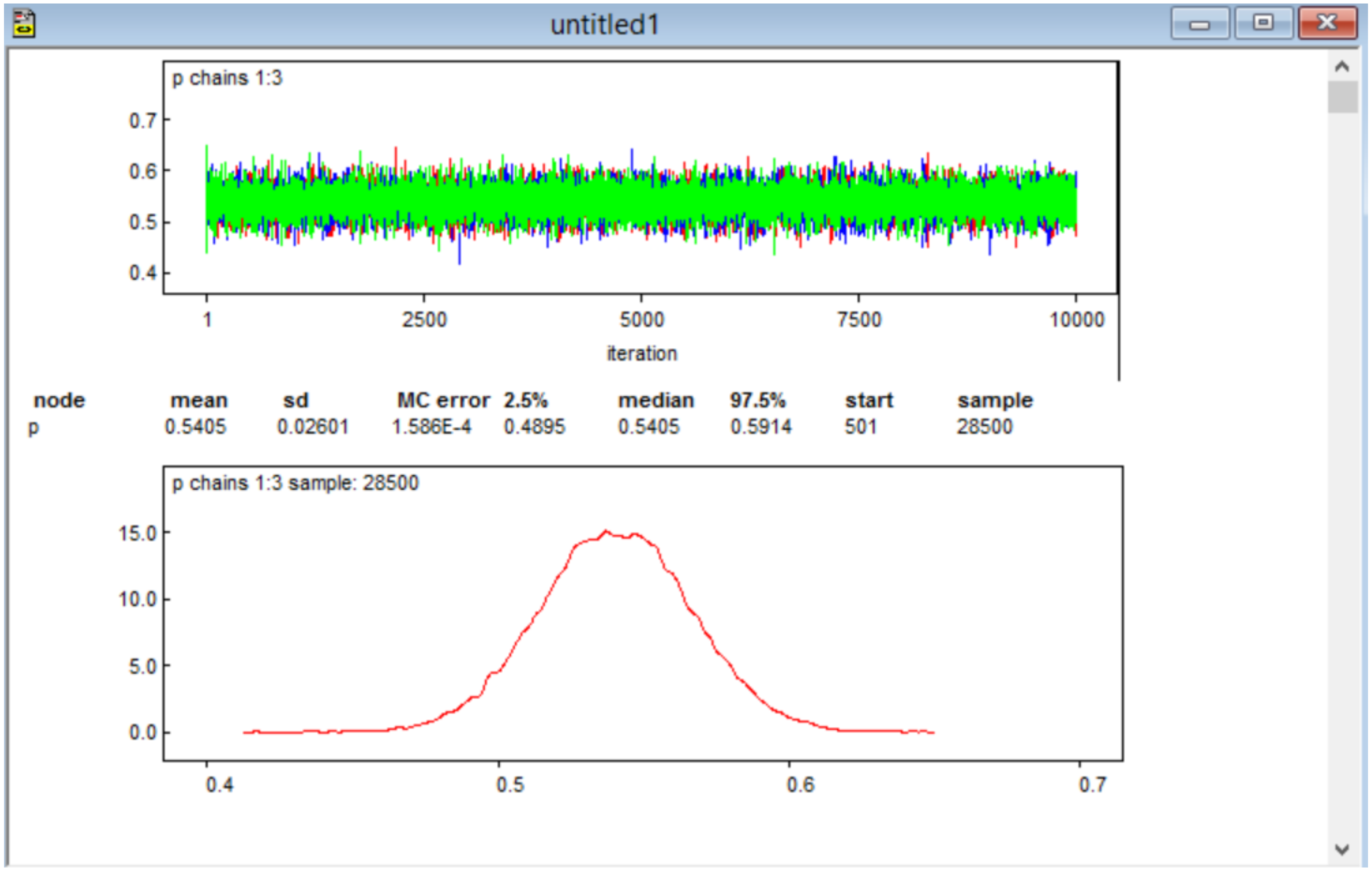


```
# Data
list(y=c(53,95,38),n=186)
```

```
# Initial values
list(p=0.5)
list(p=0.1)
list(p=0.9)
```

WinBUGS: HWE example

WinBUGS unlovely but functional in-house output;



```
node      mean     sd        MC error  2.5%     median   97.5%    start   sample
p         0.5405   0.02601   1.586E-4  0.4895   0.5405   0.5914   501     28500
```

The posterior has 95% support for $p \in (0.49, 0.59)$, the posterior mean = posterior median = 0.54. Use `coda` to get the chain(s).

WinBUGS: less pointy-clicky

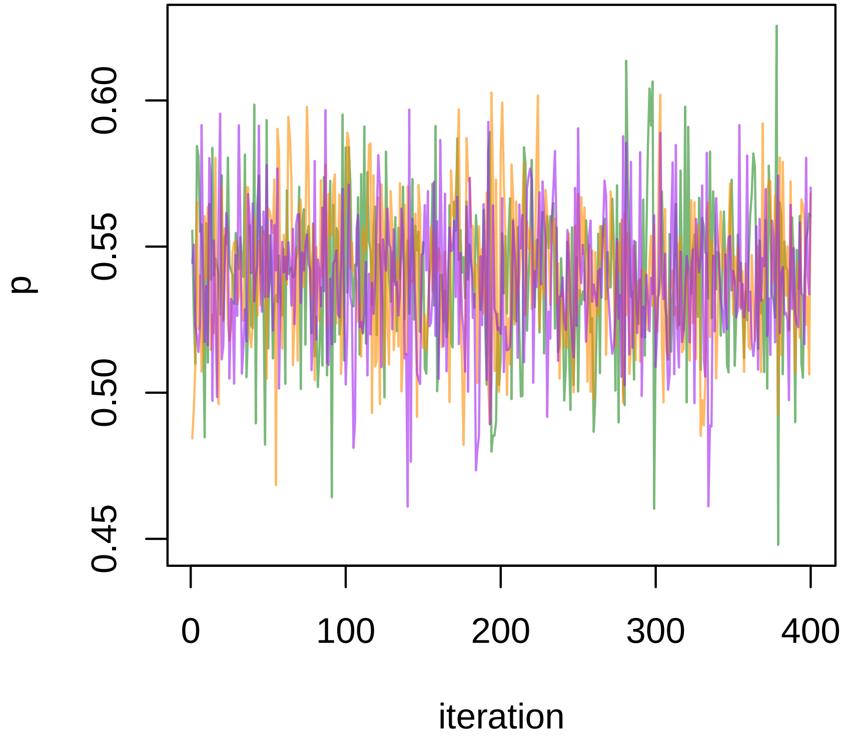
Apart from coming up with the model, everything can be automated – using R's `R2WinBUGS` package;

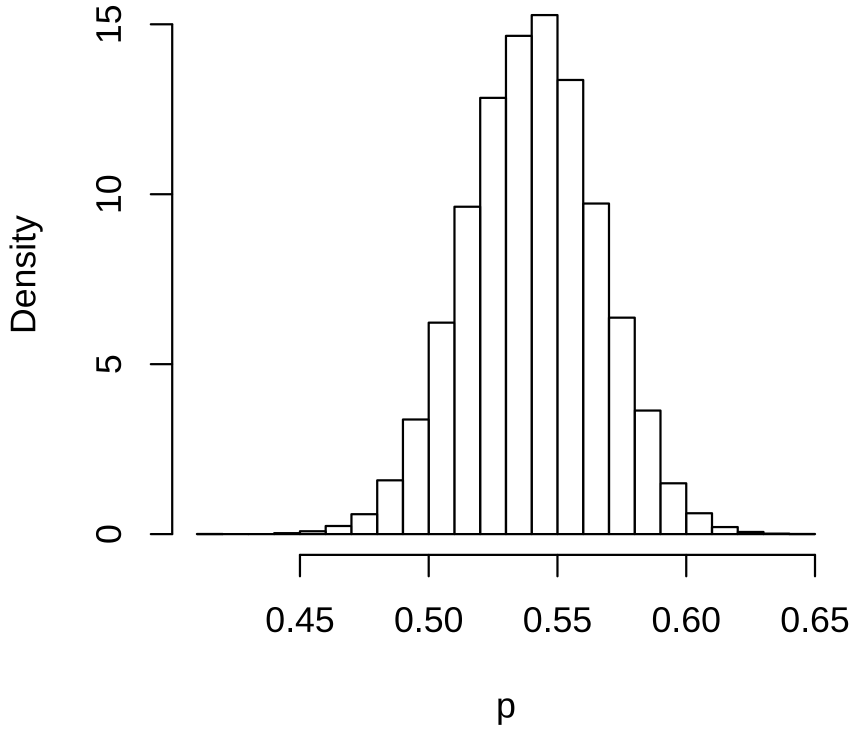
```
library("R2WinBUGS")
hweout <- bugs(data=list(y=c(53,95,38),n=186),
   inits=list(p=0.5, p=0.1, p=0.9),
               parameters.to.save=c("p"),
                  model.file="hweprog.txt",
                  bugs.directory = "C:/Program Files/WinBUGS14",
                  n.chains=3, n.iter=10000,
                  n.burnin=500, n.thin=1, DIC=FALSE)
```

- Model code in a separate file (`hweprog.txt`)
- Specify the data and initial values as R structures
- Tell R where to find WinBUGS
- The output is stored in `hweout`, an R object – no need to go via `coda`
- When debugging, pointy-clicky WinBUGS is still useful
- See next slide for less-clunky graphics

WinBUGS: less pointy-clicky

```
> print(hweout, digits=3)
Inference for Bugs model at "hweprog.txt", fit using WinBUGS,
 3 chains, each with 10000 iterations (first 500 discarded)
 n.sims = 28500 iterations saved
    mean       sd     2.5%       50%         97.5%      Rhat     n.eff
   0.540    0.026    0.490     0.541         0.590     1.001 28000.000
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).
```





WinBUGS: how it works (roughly)

- As well as the Markov blanket idea, WinBUGS uses what it knows about conjugacy to substitute closed form integrals in the calculations, where it can. (e.g. using inverse-gamma priors on Normal variances)
- Otherwise, it chooses from a hierarchy of sampling methods – though these are not cutting-edge
- Because of its generality, and the complexity of turning a model into a sampling scheme, don't expect too much help from the error messages
- Even when the MCMC is working correctly, it is possible you may be fitting a ridiculous, unhelpful model. WinBUGS' authors assume you take responsibility for that

Also, while Gibbs-style sampling works well in many situations, for some problems it's not a good choice. If unsure, check the literature to see what's been tried already.

WinBUGS: other engines

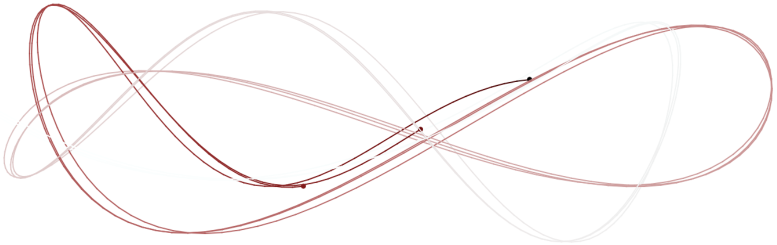
WinBUGS is no longer updated — but it's pointy-clicky interface remains a good place to get started. The BUGS language, describing models, is now used in JAGS, NIMBLE and OpenBUGS. Here's `rjags` using the **exact** same model file we just saw;

```
> library("rjags")
> jags1 <- jags.model("hweprog.txt", data=list(y=c(53,95,38),n=186) )
> update(jags1, 10000)
> summary( coda.samples(jags1, "p", n.iter=10000) )
Iterations = 11001:21000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000
1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:
          Mean             SD       Naive SE Time-series SE
     0.5398583      0.0258055      0.0002581      0.0003308
2. Quantiles for each variable:
  2.5%    25%    50%    75%  97.5%
0.4890 0.5225 0.5398 0.5576 0.5895
```

JAGS uses C, so is easier to extend than WinBUGS.

Stan

Stan is similar to BUGS, WinBUGS, JAGS etc – but new & improved;



- Coded in C++, for faster updating, it runs the *No U-Turn Sampler* – cleverer than WinBUGS' routines
- The `rstan` package lets you run chains from R, just like we did with `R2WinBUGS`
- Some modeling limitations – no discrete parameters – but becoming popular; works well with some models where WinBUGS would struggle
- Basically the same modeling language as WinBUGS – but Stan allows R-style vectorization
- Requires declarations (like C++) – unlike WinBUGS, or R – so models require a bit more typing...

Stan: example

A Stan model for the HWE example

```
data {
    int y[3];
}
parameters {
   real<lower=0,upper=1> p;
}
transformed parameters {
   simplex[3] theta;
   theta[1] = p*p;
   theta[2] = 2*p*(1-p);
   theta[3] = (1-p)*(1-p);
}
model {
   p~beta(0.5, 0.5);
   y~multinomial(theta);
}
```

- More typing than BUGS!
- But experienced programmers will be used to this overhead

Stan: example

With the model stored in `HWEexample.stan` (a text file) the rest follows as before;

```
> library("rstan")
> stan1 <- stan(file = "HWEexample.stan", data = list(y=c(53,95,38)),
+ iter = 10000, chains = 1)
> print(stan1)
Inference for Stan model: HWEexample.
1 chains, each with iter=10000; warmup=5000; thin=1;
post-warmup draws per chain=5000, total post-warmup draws=5000.
            mean se_mean   sd    2.5%     25%     50%     75%   97.5% n_eff
p           0.54    0.00 0.03    0.48    0.52    0.54    0.56    0.60  5000
theta[1]    0.29    0.00 0.03    0.23    0.27    0.29    0.31    0.36  5000
theta[2]    0.49    0.00 0.01    0.48    0.49    0.50    0.50    0.50  4541
theta[3]    0.21    0.00 0.03    0.16    0.19    0.21    0.23    0.27  5000
lp__     -192.17    0.02 0.87 -194.71 -192.44 -191.81 -191.57 -191.49  2762

Samples were drawn using NUTS(diag_e) at Tue Jul 26 14:13:31 2016.
```

- Iterations in the `stan1` object can be used for other summaries, graphs, etc
- `lp__` is the log likelihood, used in (some) measures of model fit

INLA

We've seen various examples of Bayesian analysis using Integrated Nested Laplace Approximation (INLA). For a (wide) class of models known as *Gaussian Markov Random Fields*, it gives a very accurate approximation of the posterior by 'adding up' a series of Normals.

- This approximation is not stochastic – not a Monte Carlo method
- Even with high-dimensional parameters, where MCMC works less well/badly, INLA can be practical
- INLA is so fast that e.g. 'leave-one-out' & bootstrap methods are practical – and can scale to GWAS-size analyses
- Fits most regression models – but not everything, unlike MCMC
- Non-standard posterior summaries require more work than manipulating MCMC's posterior sample

INLA

The `inla` package in R has syntax modeled on R's `glm()` function. And with some data reshaping, our HWE example **is** a GLM;

```
> y <- c(53,95,38) # 2,1,0 copies of allele with frequency "p"
> n <- 186
> longdata <- data.frame(y=rep(2:0, y), ni=rep(2, n) )
> # non-Bayesian estimate of log(p)/(1-log(p)) i.e. log odds
> glm1 <- glm( cbind(y,ni-y) ~ 1, data=longdata, family="binomial" )
> expit <- function(x){exp(x)/(1+exp(x) )}
> expit(coef(glm1))
(Intercept)
  0.5403226
> expit(confint(glm1))
    2.5 %    97.5 %
0.4895317 0.5905604
> inla1 <- inla( y~1, family="binomial", data=longdata, Ntrials=rep(2,n) )
> summary(inla1)$fixed
              mean    sd 0.025quant 0.5quant 0.975quant   mode kld
(Intercept) 0.1616 0.104    -0.0422   0.1615     0.3661 0.1612   0
> expit(summary(inla1)$fixed[,3:5]) # posterior of "p"
0.025quant   0.5quant 0.975quant
 0.4894516  0.5402875  0.5905163
```

For non-default priors, see the examples on the course site.