

Toy Example 1

Taken from “On some difficulties with a posterior probability approximation technique”.

<http://hal.archives-ouvertes.fr/hal-00212326/en/>

M1: $y \sim U(0, \theta)$, $\theta \sim E(1)$

M2: $y \sim E(\theta)$, $\theta \sim E(1)$

For values $y=0.2$ and $y=0.9$

This requires use of the ones trick for model 1 as the upper limit of a uniform is unknown. Application of the method in the StatMeth paper (e.g. see equation 4) gives

$$\Pr(m=1|y)=0.642$$

for $y=0.2$ and

$$\Pr(m=1|y)=0.486$$

for $y=0.9$ using the last 99K of a 100K single chain run. A tricky issue is approximating the posterior for θ_1 given it is truncated below by y . As the Stat Meth paper makes clear, an approximate posterior for θ_1 is needed to obtain the relevant pseudo-prior, denoted g_1 in equation (4) of the paper This involves sampling shifted θ^*_1 values, namely

$$\theta^*_1 = \theta_1 - y$$

and deriving the (truncated) gamma parameters for g_1 from the mean and sd of the θ^*_1 .

The code is as follows

```
model { z[1] <- 1; z[1] ~ dbern(p[1]); p[1] <- 1/th[1];
  th[1] ~ dexp(1) I(x[1],); th1 <- th[1]-x[1]
  x[2] ~ dexp(th[2]); th[2] ~ dexp(1)
# Likelihood terms for models
  LL[1] <- log(1/th[1]); LL[2] <- log(th[2])-th[2]*x[2]
# prior ordinates for models
  pr.ord[1] <- -th[1]; pr.ord[2]<- -th[2]
# pseudo-prior ordinates for models
  g.ord[1] <- (r[1]-1)*log(th[1]-x[1])-s[1]*(th[1]-x[1])+r[1]*log(s[1])-loggam(r[1])
```

```

g.ord[2] <- (r[2]-1)*log(th[2])-s[2]*th[2]+r[2]*log(s[2])-loggam(r[2])
# Combining likelihoods, priors and importance samples
L[1]<-LL[1]+ pr.ord[1]-g.ord[1]+log(0.5);
L[2]<-LL[2]+ pr.ord[2]-g.ord[2]+log(0.5)
# Scaling against largest likelihood
maxL<-ranked(L[,2]); SL[1]<- L[1]-maxL; SL[2]<- L[2]-maxL
expSL[1] <- exp(SL[1]); expSL[2]<- exp(SL[2]);
# model weights
w[1]<- expSL[1]/sum(expSL[]); w[2]<-expSL[2]/sum(expSL[])

```

Inits for both models

```
list(th=c(2,2))
```

Data for x=0.2 is

```
list(x=c(0.2,0.2),r=c(0.631,2),s=c(1.356,1.2))
```

Data for x=0.9 is

```
list(x=c(0.9,0.9),r=c(0.83,2),s=c(1.27,1.9))
```

Results x=0.2

node	mean	sd	MC error 2.5%	median	97.5%	start	sample
w[1]	0.6423	0.04354	1.687E-4	0.5225	0.6526	0.6743	1000 99001
w[2]	0.3577	0.04354	1.687E-4	0.3257	0.3474	0.4775	1000 99001

Results x=0.9

node	mean	sd	MC error 2.5%	median	97.5%	start	sample
w[1]	0.4856	0.02257	8.213E-5	0.4209	0.4914	0.5009	1000 99001
w[2]	0.5144	0.02257	8.213E-5	0.4991	0.5086	0.5791	1000 99001

Toy Example 2

Consider comparing two beta-binomial models, namely for a single observation $x=1$,

$$M_1: x \sim \text{Bin}(N=10, p); p \sim \text{Beta}(1, 1)$$

and

$$M_2: x \sim \text{Bin}(N=10, p); p \sim \text{Beta}(30, 30)$$

This can be solved analytically to give $P(M_1|x)=0.8634$. In particular, integrating p out from the Binomial-Beta mixture model $x \sim \text{Bin}(N, p)$, $p \sim \text{Beta}(a, b)$ gives a Beta-Binomial distribution $x \sim \text{Beta-Bin}(N, a, b)$ with density

$$p(x|N, a, b) = \frac{\text{Beta}(x+a, N-x+b)}{\text{Beta}(a, b)} \frac{N!}{(N-x)!x!}$$

which can be evaluated at $p(x|10, 1, 1)$ and $p(x|10, 30, 30)$. The posterior densities of p (providing importance sample corrections g_1 and g_2 under the two models) are obtainable analytically as $\text{Be}(2, 10)$ and $\text{Be}(1, 39)$. Alternatively an initial run provides posterior means (sd) for p , denoted $m_j(s_j)$, under the two models of 0.1656 (0.1025), 0.4427 (0.0586). One may then use the fact that beta density $\text{Be}(\alpha, \beta)$ parameters α and β are obtainable as

$$\alpha = -m(m^2 - m + s^2)/s^2, \beta = (m - 1)(m^2 - m + s^2)/s^2.$$

Using the latter results provides beta importance samples g_j with parameters (2,10.1) and (31.3,39.5). A subsequent MCMC run of 100 thousand iterations obtains likelihoods, prior ordinates and importance sample ordinates at each iteration, giving iteration specific weights, and a posterior mean weight for model 1 of 0.8634, with Monte Carlo SE of 7.8E-6. A Winbugs code for this (with the mean of `w[1:2]` being the relevant output) is

```

model {x[1]~dbin(p[1],10); p[1]~dbeta(1,1)
       x[2]~dbin(p[2],10); p[2]~dbeta(30,30)
# Likelihood terms for models
LL[1]<-logfact(10)-logfact(10-x[1])-logfact(x[1])+x[1]*log(p[1])+(10-x[1])*log(1-
p[1])
LL[2]<-logfact(10)-logfact(10-x[2])-logfact(x[2])+x[2]*log(p[2])+(10-x[2])*log(1-
p[2])
# prior ordinates for models
LP[1]<-loggam(2)-loggam(1)-loggam(1)+0*log(p[1])+0*log(1-p[1])
LP[2]<-loggam(60)-loggam(30)-loggam(30)+29*log(p[2])+29*log(1-p[2])
# importance sample ordinates for models
IMP[1]<-loggam(12.1)-loggam(2)-loggam(10.1)+1*log(p[1])+9.1*log(1-p[1])
IMP[2]<-loggam(70.8)-loggam(31.3)-loggam(39.5)+30.3*log(p[2])+38.5*log(1-
p[2])
# Combining likelihoods, priors and importance samples
L[1]<-LL[1]+LP[1]+log(0.5)-IMP[1]; L[2]<-LL[2]+LP[2]+log(0.5)-IMP[2]
# Scaling with the largest likelihood, and protecting for underflow
maxL<-ranked(L[,2]); SL[1]<-max(L[1]-maxL,-700); SL[2]<-max(L[2]-maxL,-
700)
# exponentiating and normalizing

```

```
expSL[1]<- exp(SL[1]); expSL[2]<- exp(SL[2]);  
# model weights  
w[1]<-expSL[1]/sum(expSL[]); w[2]<-expSL[2]/sum(expSL[])}
```

with the same data for both models, namely `list(x=c(1,1))`.