ORIE 678 — D. Ruppert
R2WinBUGS example: penalized spline

R Program:

```
library(R2WinBUGS)
#   Generate data
z = seq(from=0,to=2*pi,by=.025)
y = sin(2*z) + rnorm(N,0,1)

#  Construct bases
N=length(z)
Nknots=8
degree=2
X = matrix(0,nrow=N,ncol=(1+degree))
Z  =matrix(0,nrow=N,ncol=Nknots)
knots = seq(from=0,to=2*pi,length.out=Nknots+2)
knots = knots[2:(Nknots+1)]
for(i in 1:N)
{
for(j in 1:(degree+1))
{
X[i,j] = z[i]^(j-1)
}
for(j in 1:Nknots)
{
Z[i,j] = sign(z[i]-knots[j])*((z[i]-knots[j])^degree)
}
}

#   Run BUGS
data=list("y","X","Z","N","degree","Nknots")
inits=function(){list(beta=c(0,1,-.1),b=rep(.1,Nknots),taue=.3,taub=.1)}
Pspline.sim = bugs(data,inits,model.file="Pspline.bug",
parameters=c("mu","beta","b","taue","taub"),
n.chains = 2,n.iter=50000,n.burnin=5000,n.thin=50,
bugs.directory="c:/Program Files/WinBUGS14/",codaPkg=FALSE)

print(Pspline.sim,digit=3)
pdf("Pspline1.pdf")
plot(Pspline.sim)

mm=Pspline.sim$sims.array[,1,]
mu=mm[,1:(length(z))]
mumean = apply(mu,MARGIN=2,FUN=mean)
musd = apply(mu,MARGIN=2,FUN=sd)

pdf("Pspline2.pdf")
plot(z,y,type="p",col="red",pch="o",cex=1.1,ylim=c(-4,6.5))
lines(z,mumean,typ="l",col="black",lwd=4)
```

```
lines(z,mumean+2*musd,type="l",col="blue",lwd=2,lty=2)
lines(z,mumean-2*musd,type="l",col="blue",lwd=2,lty=2)
lines(z,sin(2*z),type="l",col="green",lwd=4,lty=3)
legend(1.6,6.4,c("data","estimate","lower CL","upper CL","true curve"),
col=c("red","black","blue","blue","green"),lwd=c(0,4,2,2,4),
lty=c(0,1,2,2,3),pch=c("o",-1,-1,-1,-1),cex=1.1)
graphics.off()

# Save synthetic data
data2=cbind(z,y)
#write.csv(data2,"PsplineData.csv",row.names=FALSE)
```

Bugs program:

```
model{
for(i in 1:N)
{
y[i] ~ dnorm(mu[i],taue)
mu[i] <- inprod(beta[],X[i,]) + inprod(b[],Z[i,])
}
for(j in 1:(degree+1))
{
beta[j] ~ dnorm(0,.0001)
}
for(j in 1:Nknots)
{
b[j] ~ dnorm(0,taub)
}
taub ~ dgamma(.1,.00001)
taue ~ dgamma(0.1,0.01)
}
```

Output:

```
Inference for Bugs model at "Pspline.bug", fit using winbugs,
 2 chains, each with 50000 iterations (first 5000 discarded), n.thin = 50
 n.sims = 1800 iterations saved
            mean    sd    2.5%     25%     50%     75%   97.5%  Rhat n.eff
mu[1]      0.050 0.321  -0.584  -0.168   0.055   0.266   0.649 1.104    20
mu[2]      0.119 0.305  -0.478  -0.087   0.122   0.325   0.702 1.082    25
mu[3]      0.186 0.290  -0.375  -0.013   0.187   0.381   0.750 1.062    32
mu[4]      0.250 0.275  -0.292   0.062   0.249   0.439   0.792 1.045    43
mu[5]      0.311 0.262  -0.202   0.129   0.309   0.492   0.827 1.031    62
mu[6]      0.370 0.250  -0.119   0.200   0.363   0.541   0.863 1.020   100
mu[7]      0.426 0.239  -0.028   0.265   0.421   0.589   0.895 1.012   200
mu[8]      0.479 0.228   0.044   0.326   0.474   0.631   0.930 1.006   590
mu[9]      0.530 0.219   0.110   0.383   0.523   0.675   0.965 1.003  1800
mu[10]     0.578 0.210   0.176   0.435   0.572   0.716   0.996 1.003  1800
```

mu[11]	0.624	0.202	0.238	0.486	0.618	0.756	1.024	1.006	420
mu[12]	0.667	0.195	0.294	0.535	0.660	0.795	1.049	1.011	160
mu[13]	0.707	0.189	0.343	0.577	0.705	0.832	1.075	1.014	120
mu[14]	0.745	0.184	0.386	0.619	0.743	0.866	1.109	1.024	67
mu[15]	0.780	0.179	0.428	0.661	0.776	0.895	1.132	1.038	45
mu[16]	0.813	0.175	0.467	0.697	0.809	0.927	1.158	1.053	34
mu[17]	0.842	0.171	0.506	0.732	0.839	0.954	1.182	1.069	27
mu[18]	0.870	0.168	0.534	0.759	0.864	0.981	1.205	1.085	23
mu[19]	0.894	0.165	0.563	0.785	0.891	1.002	1.225	1.099	21
mu[20]	0.916	0.163	0.591	0.806	0.913	1.022	1.246	1.111	19
mu[21]	0.936	0.161	0.613	0.827	0.933	1.041	1.261	1.118	18
mu[22]	0.952	0.159	0.632	0.846	0.949	1.056	1.277	1.122	17
mu[23]	0.967	0.157	0.653	0.861	0.964	1.070	1.286	1.121	18

.
.
.

mu[243]	-0.161	0.212	-0.581	-0.304	-0.158	-0.024	0.251	1.008	200
mu[244]	-0.120	0.221	-0.555	-0.270	-0.120	0.025	0.312	1.007	240
mu[245]	-0.079	0.229	-0.530	-0.235	-0.081	0.071	0.376	1.006	290
mu[246]	-0.037	0.238	-0.511	-0.201	-0.040	0.118	0.430	1.005	370
mu[247]	0.006	0.247	-0.485	-0.166	0.003	0.166	0.494	1.004	480
mu[248]	0.050	0.256	-0.458	-0.128	0.045	0.216	0.561	1.003	660
mu[249]	0.095	0.266	-0.430	-0.091	0.092	0.269	0.629	1.002	990
mu[250]	0.141	0.276	-0.401	-0.053	0.137	0.323	0.695	1.001	1600
mu[251]	0.188	0.286	-0.371	-0.018	0.184	0.378	0.763	1.001	1800
mu[252]	0.236	0.297	-0.336	0.021	0.232	0.432	0.830	1.001	1800
beta[1]	9.329	4.707	0.681	5.695	9.799	12.422	18.370	1.578	6
beta[2]	-2.486	1.447	-5.409	-3.419	-2.557	-1.415	0.319	1.327	8
beta[3]	-0.661	0.272	-1.224	-0.859	-0.648	-0.464	-0.180	1.846	4
b[1]	0.296	0.429	-0.386	-0.040	0.268	0.564	1.309	2.461	3
b[2]	1.171	0.401	-0.018	1.008	1.225	1.416	1.834	1.212	13
b[3]	0.674	0.474	-0.209	0.355	0.647	0.963	1.750	1.032	53
b[4]	-1.270	0.590	-2.493	-1.623	-1.244	-0.897	-0.051	1.012	140
b[5]	-0.734	0.536	-1.846	-1.090	-0.734	-0.378	0.303	1.015	310
b[6]	0.535	0.412	-0.276	0.281	0.528	0.831	1.264	1.005	350
b[7]	1.434	0.400	0.757	1.155	1.399	1.669	2.339	1.144	18
b[8]	-0.670	0.314	-1.378	-0.866	-0.649	-0.449	-0.135	1.372	8
taue	0.985	0.089	0.822	0.923	0.983	1.039	1.173	1.000	1800
taub	1.060	0.604	0.253	0.616	0.926	1.364	2.575	1.002	1200
deviance	720.645	4.736	714.000	717.300	719.800	723.400	731.902	1.043	61

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).

pD = 8.6 and DIC = 729.2 (using the rule, pD = Dbar-Dhat)
DIC is an estimate of expected predictive error (lower deviance is better).

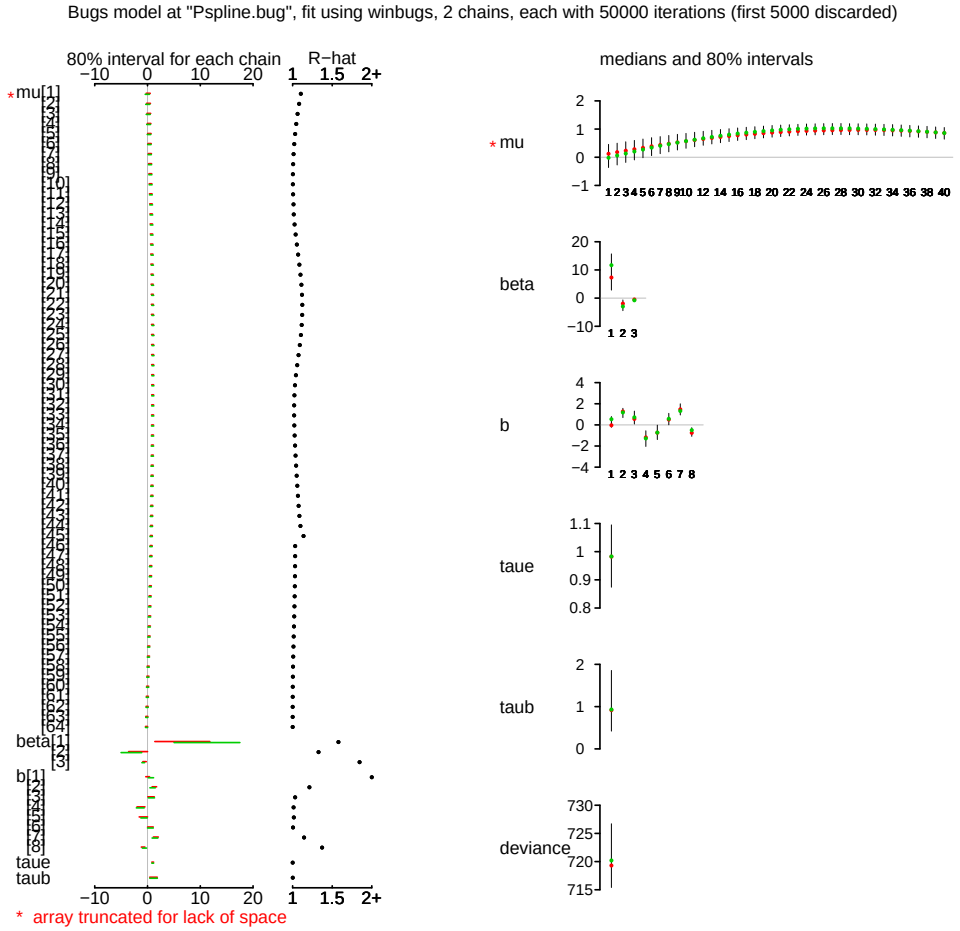

Figure 1: Plot of BUGS object.

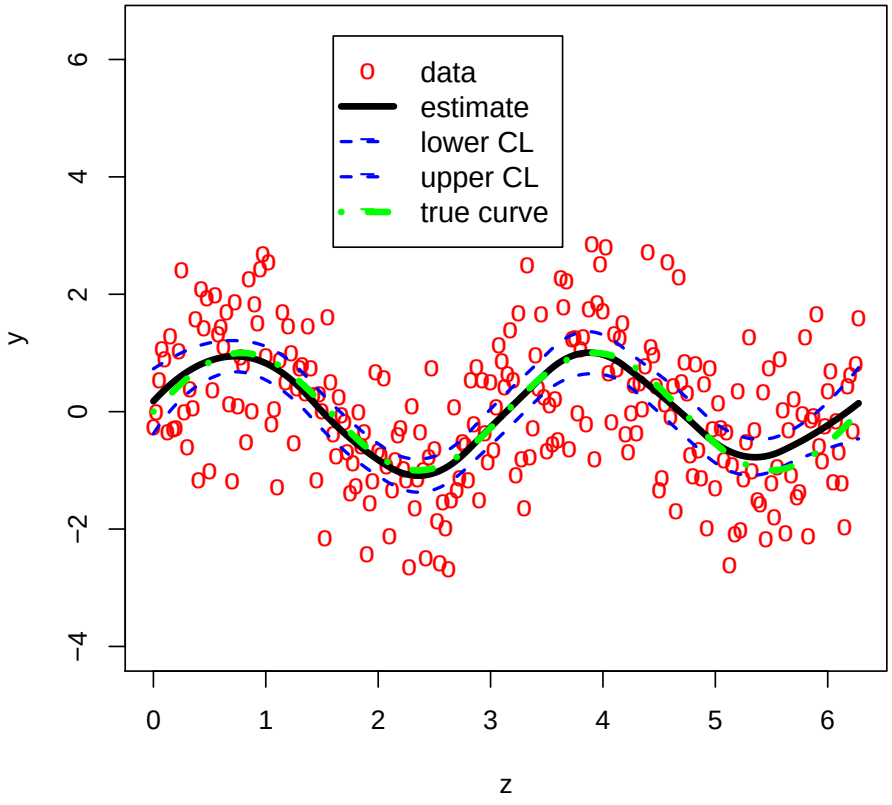

Figure 2: Data, estimator, and true regression function.