```
#
# Script to run the CCL Model on data from the CAS Loss Reserve Database
# To run the LCL model, set prior for rho ~ dunif(-.00001,00001) in JAGS script
# by Glenn Meyers
#
rm(list = ls())      # clear workspace"
#setwd("Your Working Directory")
#
# user inputs
#
insurer.data="comauto_pos (3).csv"
#insurer.data="ppauto_pos .csv"
#insurer.data="wkcomp_pos.csv"
#insurer.data="othliab_pos.csv"
#insurer.data="prodliab_pos.csv"
#insurer.data="medmal_pos.csv"
grpcode="14885"
losstype="incloss"  #"incloss" if incurred loss or "cpdloss" if paid loss
outfile="CCL Method Outputs.csv"
nburn=10000
#
# JAGS script
#
modelString = "model {
  mu[1]<-alpha[w[1]]+beta[d[1]]
  logloss[1]~dnorm(mu[1],1/sig2[1])
  for (i in 2:length(w)){
    mu[i]<-alpha[w[i]]+beta[d[i]]+rho*(logloss[i-1]-mu[i-1])*wne1[i]
    logloss[i]~dnorm(mu[i],1/sig2[i])
  }
#
# set up sig2
#
  for (i in 1:length(w)){
    sig2[i]<-sigd2[d[i]]
  }
  for (j in 1:10){
    sigd2[j]<-sum(a[j:10])
  }
  for (k in 1:10){
    a[k]~dunif(0.000001,1)
  }
#
# specify priors
```

```
#
  for (i in 1:numlev){
    alpha[i]~dnorm(log(premium[i])+logelr,.1)
    }
  logelr~dunif(-1.5,0.5)
#
  for (i in 1:9){
    beta[i]~dunif(-5,5)
    }
  beta[10]<-0
  rho~dunif(-1,1)
# rho~dunif(-.00001,.00001) # Use for LCL model
}"
#
# get data
#
a=read.csv(insurer.data)
#
# function to get Schedule P triangle data given ins group and line of business
#
ins.line.data=function(grpcode){
  b=subset(a,a$GRCODE==grpcode)
  name=b$GRNAME
  grpcode=b$GRCODE
  w=b$AccidentYear
  d=b$DevelopmentLag
  cum_incloss=b[,6]
  cum_pdloss=b[,7]
  bulk_loss=b[,8]
  dir_premium=b[,9]
  ced_premium=b[,10]
  net_premium=b[,11]
  single=b[,12]
  posted_reserve97=b[,13]
  # get incremental paid losses - assume data is sorted by ay and lag
  inc_pdloss=numeric(0)
  for (i in unique(w)){
    s=(w==i)
    pl=c(0,cum_pdloss[s])
    ndev=length(pl)-1
    il=rep(0,ndev)
    for (j in 1:ndev){
      il[j]=pl[j+1]-pl[j]
    }
    inc_pdloss=c(inc_pdloss,il)
  }
  data.out=data.frame(grpcode,w,d,net_premium,dir_premium,ced_premium,
                    cum_pdloss,cum_incloss,bulk_loss,inc_pdloss,single,posted_reserve97)
  return(data.out)
}
#
# read and aggregate the insurer data and
# set up training and test data frames
#
cdata=ins.line.data(grpcode)
set.seed(12345)
w=cdata$w-1987
d=cdata$d
#
# sort the data in order of d, then w within d
#
o1=100*d+w
o=order(o1)
w=w[o]
d=d[o]
premium=cdata$net_premium[o]
cpdloss=cdata$cum_pdloss[o]
cpdloss=pmax(cpdloss,1)
incloss=cdata$cum_incloss[o]-cdata$bulk_loss[o]
incloss=pmax(incloss,1)
wne1=ifelse(w==1,0,1)
adata=data.frame(grpcode,w,d,premium,cpdloss,incloss,wne1)
rdata=subset(adata,(adata$w+adata$d)<12)
numw=length(unique(rdata$w))
rdata=subset(adata,(adata$w+adata$d)<12)
if(losstype=="incloss") rloss=rdata$incloss else rloss=rdata$cpdloss
if(losstype=="incloss") aloss=adata$incloss else aloss=adata$cpdloss
#
# Initialize JAGS model
#
inits1=list(.RNG.name= "base::Wichmann-Hill",
          .RNG.seed= 12341)
inits2=list(.RNG.name= "base::Marsaglia-Multicarry",
          .RNG.seed= 12342)
inits3=list(.RNG.name= "base::Super-Duper",
          .RNG.seed= 12343)
inits4=list(.RNG.name= "base::Mersenne-Twister",
          .RNG.seed= 12344)
```

```
data.for.jags=list(premium= premium[1:10],
                   logloss = log(rloss),
                   numlev  = numw,
                   w       = rdata$w,
                   wne1    = rdata$wne1,
                   d       = rdata$d)
#
# run the model
#
library(runjags)
library(coda)
#
# call to run.jags #
nthin=2
maxpsrf=2
while (maxpsrf>1.05){
  nthin=nthin*2
  print(paste("nthin =",nthin))
  jagout=run.jags(model=modelString,monitor=c("alpha","beta[1:9]","sigd2","rho"),
                  data=data.for.jags,n.chains=4,method="parallel",
                  inits=list(inits1,inits2,inits3,inits4),thin=nthin,silent.jags=F,
                  plots=TRUE,burnin=nburn,sample=2500,psrf.target=1.05)
  gelman=gelman.diag(jagout)
  maxpsrf=max(gelman$psrf[,1])
  print(paste("maxpsrf =",maxpsrf))
}
# Print summary of model parameters
print(jagout[[1]][, 1:2])
#print(jagout$timetaken)
#
# optional diagnostics
#
#crosscorr.plot(jagout$mcmc)
#traceplot(jagout$mcmc)
#print(gelman.diag(jagout))
#gelman.plot(jagout)
print(jagout)
#print(summary(jagout)[[1]][,1:2])
#
# extract information from jags output to process in R
#
b=as.matrix(jagout$mcmc)
alpha=b[,1:10]
beta=cbind(b[,11:19],rep(0,dim(b)[1]))
```

```
rho=b[,20]
sigd2=b[,21:30]
#
# simulate loss statistics by accident year reflecting total risk for the JAGS model
#
set.seed(12345)
Premium = subset(rdata, rdata$d == 1)$premium
at.wd10 = matrix(0, dim(b)[1], 10)
ss.wd10 = rep(0, 10)
ms.wd10 = rep(0, 10)
mu.wd11mw = matrix(0, dim(b)[1], 10)
mu.wd10 = matrix(0, dim(b)[1], 10)
at.wd10[,1] = rep(rloss[55], dim(b)[1])
mu.wd10[,1] = alpha[,1] + beta[,10]
for (w in 2:10){
  mu.wd10[,w] = alpha[,w] + beta[,10] + rho * (log(at.wd10[,w-1]) - mu.wd10[,w-1])
  for (i in 1:dim(b)[1]){
    at.wd10[i,w] = rlnorm(1, mu.wd10[i,w], sqrt(abs(sigd2)))
  }
}
ms.wd10[1] = mean(at.wd10[,1])
for (w in 2:10){
  ms.wd10[w] = mean(at.wd10[,w])
  ss.wd10[w] = sd(at.wd10[,w])
}
Pred.CCL = rowSums(at.wd10)
ms.td10 = mean(Pred.CCL)
ss.td10 = sd(Pred.CCL)
CCL.Estimate = round(ms.wd10)
CCL.S.E. = round(ss.wd10)
CCL.CV = round(CCL.S.E. / CCL.Estimate, 4)
act = sum(subset(aloss, adata$d == 10)[1:10])
pct.CCL = sum(Pred.CCL <= act) / length(Pred.CCL) * 100

#
# put CCL accident year statistics into a data frame
#
W=c(1:10,"Total")
CCL.Estimate=c(CCL.Estimate,round(ms.td10))
CCL.S.E.=c(CCL.S.E.,round(ss.td10))
CCL.CV=c(CCL.CV,round(ss.td10/ms.td10,4))
Premium=c(Premium,sum(Premium))
Outcome=subset(aloss,adata$d==10)
Outcome=c(Outcome,sum(Outcome))
```

```
Group=rep(grpcode,11)
CCL.Pct=c(rep(NA,10),pct.CCL)
risk=data.frame(W,Premium,CCL.Estimate,CCL.S.E.,CCL.CV,Outcome,CCL.Pct)
print(risk)
write.csv(risk,file=outfile,row.names=F)
par(mfrow=c(1,1))
hist(rho,main="",xlim=c(-1,1),xlab=expression(rho))
hist(Pred.CCL,main="Predictive Distribution of Outcomes",xlab="")
```