Coursework: Fitting model with JAGS

By Ida Johanne Austad
Module: Statistics and Machine Learning 2
University of Manchester, Spring 2019

Description of chosen problem and data

The chosen problem for this coursework is the WINBugs example on "Pumps: conjugate gamma-Poisson hierarchical model", originally an example taken from a paper from 1993 by George et al. named "Conjugate Likelihood Distributions" [1,2]. The problem is that of modelling pump failure, and the accompanied data is for ten pumps for which failures has been recorded over some time period.

Description of variables:
N = Number of power plant pumps
t = Length of operation time in thousands of hours
x = Number of failures

The data is presented in the table below:

Pump	t_i	x_i
1	94.5	5
2	15.7	1
3	62.9	5
4	126	14
5	5.24	3
6	31.4	19
7	1.05	1
8	1.05	1
9	2.1	4
10	10.5	22

It is suggested that the number of failures x_i can be assumed to follow a Poisson distribution with failure rate θ_i for pump i:

$$\mathrm{x_i} \sim \mathrm{Poisson}(\theta_i t_i)\,, \qquad i = 1, \ldots, 10$$

The Poisson distribution is a common model of choice when the goal is to model how many of an event will occur within a time frame. This is because it is a discrete distribution which expresses the probability of a fixed number of events occurring within a time interval [3, 4]. Overall, assuming this distribution seems sensible.

The failure rate θ_i can itself be assumed to follow a Gamma distribution with hyperparameters α and β:

$$\theta_i \sim \mathrm{Gamma}\ (\alpha, \beta), \qquad i = 1, \ldots, 10$$

Where (for instance)

$$\alpha \sim Exponential(1.0)$$
$$\beta \sim Gamma(0.1,1)$$

It was convenient to choose the Gamma distribution for θ_i as it is the Poisson distribution's conjugate prior, making the GIBBS sampling process easier [4, 5]. We choose a gamma distribution with its own *hyperparameters* for θ_i as it is reasonable to believe, under the Bayesian approach, that there is also uncertainty in the parameter θ_i for each x_i . Thus, we assume that each pump has its own failure rate so that for each observation, there may be natural variation in θ_i . Here, α and β are called hyperparameters as they can influence the data indirectly via θ_i. Overall, this means that we end up with what is known as a hierarchical model, as we have two levels of uncertainty; firstly θ_i and secondly α and β [6]. The choice of prior distributions for α and β will be discussed later.

The purpose of this coursework is to apply GIBBS sampling using the JAGS (Just Another Gibbs Sampler) library to estimate the value of the hyperparameters α and β . We choose only to evaluate the results for α and β, as including the θ_i values for all i's would create a very lengthy report. The original motivation in the paper by George et al. for applying the GIBBS sampler to this problem was that they found that no matter which prior distributions are chosen for the hyperparameters, it led to a non-standard posterior for α .

The following sections will present the code used to create the models and the modelling choices taken along the way. Lastly the results from running two slightly different models will be interpreted and compared.

Preparing the data

```
# Load libraries
library( "rjags" )
library( "coda" )
library("mcmcplots")

# Prepare data
pump.data <- list(t = c(94.3, 15.7, 62.9, 126, 5.24, 31.4, 1.05, 1.05, 2.1,
                        10.5),
                  x = c(5, 1, 5, 14, 3, 19, 1, 1, 4, 22),
                  N = 10)
t <- pump.data$t
x <- pump.data$x
N <- pump.data$N
```

Model with uninformed prior distributions for α and β

```
model {
            for (i in 1:N) {
                  theta[i] ~ dgamma(alpha, beta)
                  lambda[i] <- theta[i] * t[i]
                  x[i] ~ dpois(lambda[i])
            }
            alpha ~ dnorm(0, 0.01)
            beta ~ dnorm(0, 0.01)
}
```

Model with informed prior distributions for α and β

```
model {
            for (i in 1:N) {
                  theta[i] ~ dgamma(alpha, beta)
                  lambda[i] <- theta[i] * t[i]
                  x[i] ~ dpois(lambda[i])
            }
            alpha ~ dexp(1)
            beta ~ dgamma(0.1, 1.0)
}
```

Choice of prior distributions for α and β

There are several schools of thought on how to select prior distributions, whereof one of these is informative versus uninformative. The purpose of choosing a certain prior is to include information or beliefs one may have about the likely location of a (hyper)parameter. One may choose an informative prior distribution if one has certain information or beliefs about the distribution of the (hyper)parameters. On the other hand one may choose an uninformative prior if one has belief in the representativeness of collected data or does not want to make (strong) assumptions about the distribution [7, 8]. In this coursework both strategies will be tested and compared. The uninformed strategy makes use of a normal distribution

and the informed strategy makes use of the Exponential and Gamma distribution for α and β , the same as was used in the paper where the example was taken from. This is based on the assumption that the George and his colleagues had information to believe that these were appropriate in the example used in *Conjugate Likelihood Distributions* [3].

Initialize JAGS object

```
jags.obj <- jags.model(
  "pump_model.jags",
  data=list(
    't' = t,
    'x' = x,
    'N' = N),
  inits = function() {
    list(
      alpha = 1,
      beta = 1,
      theta = rep(0.1, 10))
  },
  n.chains = 5
  n.adapt = 1000
)
```

Choice of model specification

Number of chains
As a sample in a MCMC sampling process is dependent in the previous drawn value, it is desirable to run multiple chains to evaluate convergence [9]. Here the choice was 5 chains.

Choice of initial values for α and β
The initial values specify the starting values of the hyperparameters. It is recommended to choose initial values which are likely to lie under the posterior distribution, as this will allow the chains to converge faster [9]. In this case we had to choose ten initial values for each θ_i, and one for α and β . The example code above shows the same initial values as used by George et. al., but other values for α and β will also be tested.

Length of adaptation phase
The adaptive phase in JAGS is used to allow the sampler to optimize certain values, such as the step size [9]. Here the adaptive phase was specified to be 1000. The samples collected in this phase are discarded.

Sample data

```
# Do burnin
n.burnin <- 500
update(jags.obj, n.burnin)

# Sample
n.samples <- 5000
my.samples <- coda.samples(
jags.obj, c("alpha","beta"), n.iter=n.samples #only investigate alpha and beta
)
```

Choice of burn-in and number of iterations per chain

Although an adaptation phase has already been included in the model, some suggest that a burn-in phase should always be included as well – as the adaptation phase is non-Markovian and does as such not substitute a bur- in phase. Still, it is suggested that a relatively short burn-in may be sufficient if one has a sufficiently large adaptation phase [10]. Several lengths of burn-in phases were tested, keeping the length of adaptation still, and no significant differences were recorded.

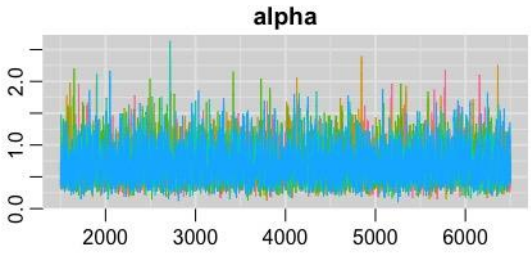
The number of iterations specifies the length of the chains and one should choose the number of iterations so that the results indicate convergence [9]. In this coursework several alternatives where tested, but the code above shows 5000 which was sufficient under all other variations tested.

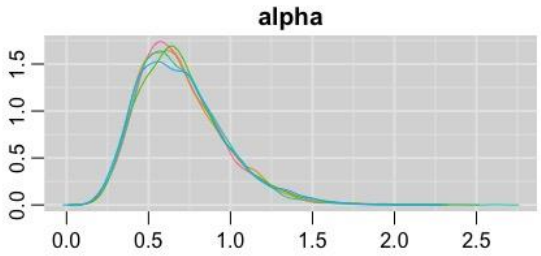
Sample results and diagnostics code

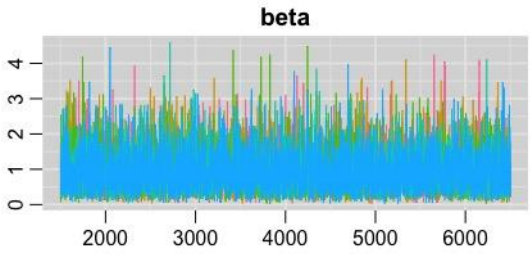
```
# Look at sample
head( as.matrix(my.samples, chain=TRUE, iters=TRUE) )

# Summarize posterior samples
summary( my.samples )

# Traceplot
traplot(my.samples,c("alpha","beta"))

# Posterior density distribution
denplot(my.samples,c("alpha","beta"))

# Make a plot that addresses convergence
quartz()
gelman.plot( my.samples )

# Autocorrelation plot
autocorr.plot(my.samples)
```

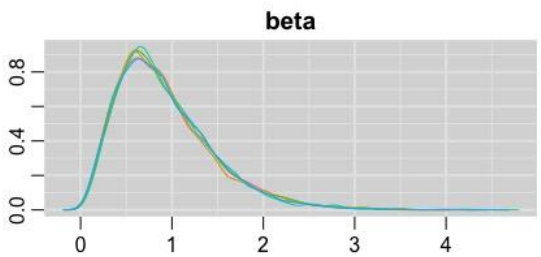
Results output - Model with informed priors

```
> head( as.matrix(my.samples, chain=TRUE, iters=TRUE) )
      CHAIN ITER  alpha       beta
[1,]  1 1501 0.5700619 0.9077394
[2,]  1 1502 0.7762593 1.8005028
[3,]  1 1503 0.9390023 1.3642014
[4,]  1 1504 0.8402246 1.0556078
[5,]  1 1505 0.7465948 0.9006835
[6,]  1 1506 0.5983581 0.4947241

> summary( my.samples )
Iterations = 1501:6500
Thinning interval = 1
Number of chains = 5
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

      Mean  SD Naive SE Time-series SE
alpha 0.6984 0.2677 0.001693        0.003702
beta  0.9264 0.5347 0.003382        0.006851

2. Quantiles for each variable:

      2.5%  25%   50%   75% 97.5%
alpha 0.293 0.5047 0.6598 0.8482 1.325
beta  0.191 0.5416 0.8223 1.2062 2.236

> traplot(my.samples,c("alpha","beta"))
> denplot(my.samples,c("alpha","beta"))
```

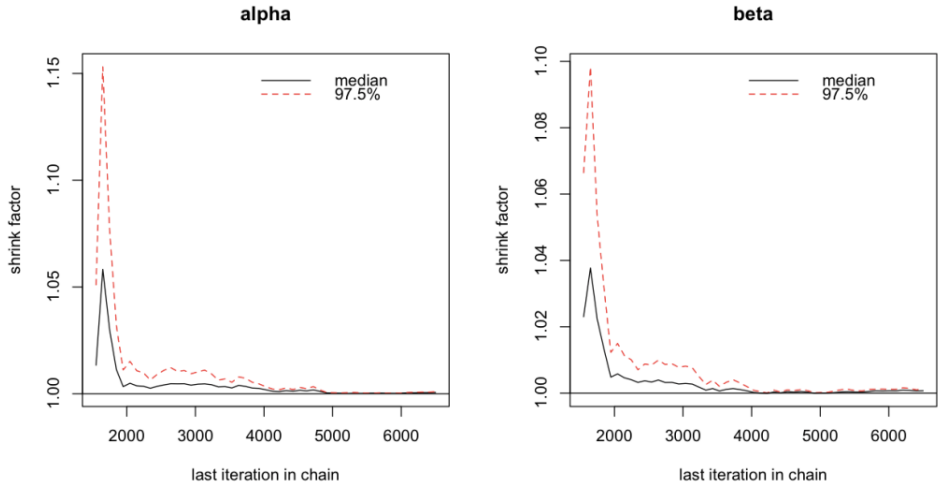




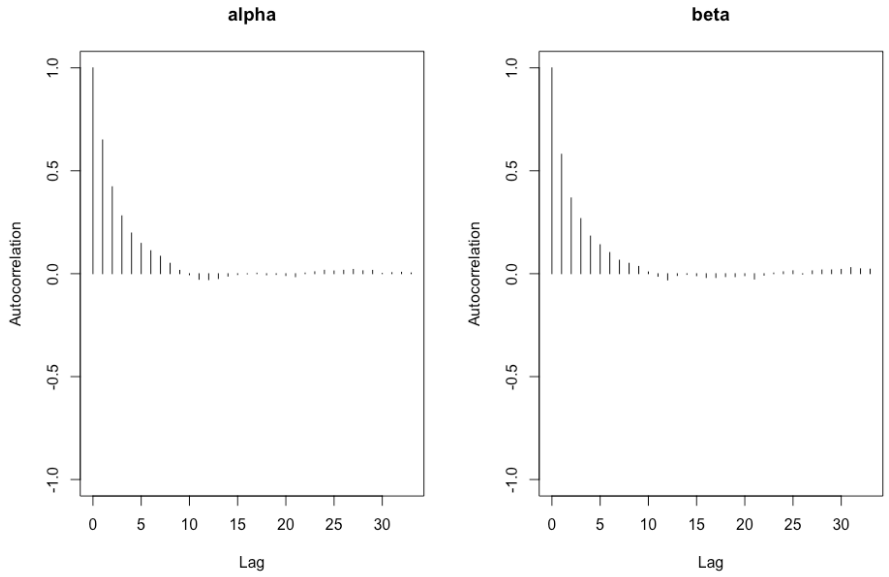




```
> quartz()
> gelman.plot( my.samples )
```

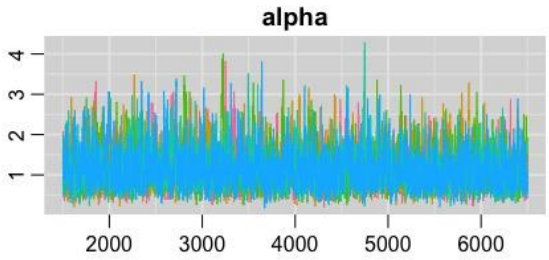


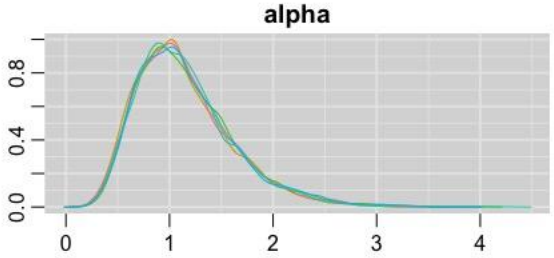

```
> autocorr.plot(my.samples)
```

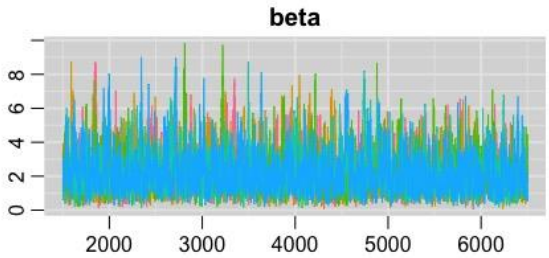


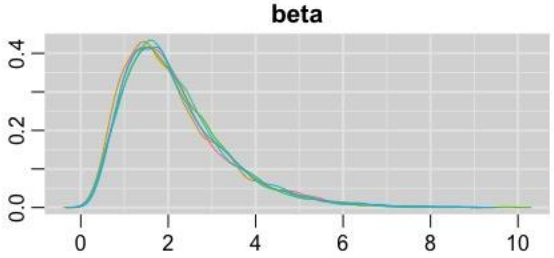
Results Output - Model with uninformed priors

```
> head( as.matrix(my.samples, chain=TRUE, iters=TRUE) )
      CHAIN ITER  alpha       beta
[1,]  1 1501 2.046275 3.982103
[2,]  1 1502 1.962290 2.672654
[3,]  1 1503 2.077879 3.475554
[4,]  1 1504 1.523599 3.800458
[5,]  1 1505 1.787954 3.700117
[6,]  1 1506 1.634829 2.857922

> summary( my.samples )

Iterations = 1501:6500
Thinning interval = 1
Number of chains = 5
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

      Mean  SD Naive SE Time-series SE
alpha 1.151 0.4757 0.003009         0.009303
beta  2.151 1.2011 0.007596         0.026260

2. Quantiles for each variable:

      2.5%  25%   50%   75% 97.5%
alpha 0.4577 0.81 1.071 1.408 2.301
beta  0.5504 1.30 1.902 2.729 5.222

> traplot(my.samples,c("alpha","beta"))
> denplot(my.samples,c("alpha","beta"))
```

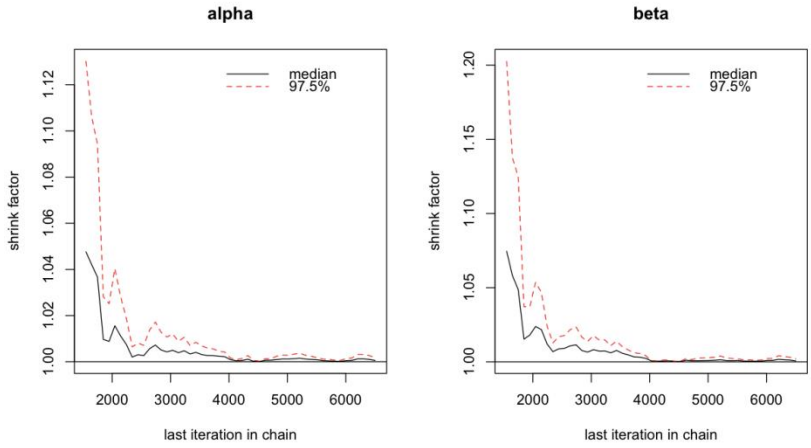




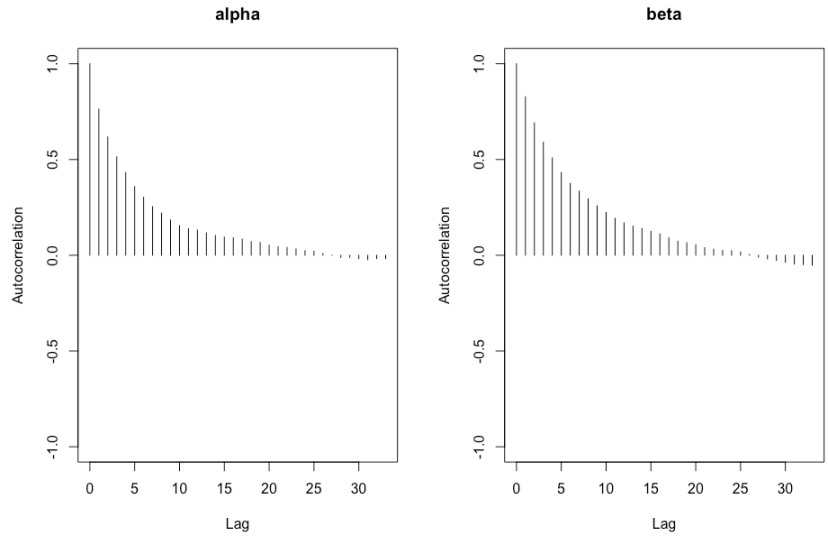




```
> quartz()
> gelman.plot( my.samples )
```




```
> autocorr.plot(my.samples)
```



Interpretation and comparison of results and diagnostics

In the following section the results from the two models will be interpreted and compared. Some of the diagnostics tools are overlapping in terms of the insights they may provide, such as assessing convergence. Still, they have all been included as it is interesting to understand how they can be applied and used.

Summary statistics

From the summary statistics on the two fitted models one can see that the mean for α and β for the informed and the uninformed model are different, respectively 0.293 and 0.191 versus 1.151 and 2.151. The results from the informed model is very similar to the results reported by George et al. [1,2]. This is expected as the model using the informed prior distributions is very similar to theirs, the only difference being 5000 instead of 10000 iterations. The reason for this difference between the two models may be that by choosing the uninformative prior in one of the models one allows the available data to have a larger effect on the posterior distribution of the hyperparameters. Moreover, one can see that the empirical sample standard deviation of the hyperparameters from the two model samples are different for α and β, respectively 0.2677 and 0.5347 versus 0.4757 and 1.2011. As such there was a larger standard variation in the sample of the uninformed model compared to the informed model.

Trace plot

Trace plots are commonly used to assessing the convergence of the chains. In the plots presented here each chain has its own color. To validate that the chains have stabilized one should see a stable trajectory, also known as a hairy (flat) caterpillar [11]. For both the models fitted here this seems to be the case; both have a stable consistent trajectory. The difference between them is the width of their chains' trajectories; the width is much larger in the uninformed case compared to the informed case. This was reflected in the sample standard deviation as well. The sample values for α and β in the uninformed case reach as high as 4 and 9 respectively, while in the informed case they are no higher than approximately 0.25 and 0.45.

Density plot

The density plots represent the distribution of the hyperparameters that are being explored during the sampling, where each chain has its own color [11]. It provides a visual presentation of the means and standard deviation given by the statistics summary for both models. One can see that the distribution for both hyperparameters α and β are much narrower for the informed model, and the mean is lower.

Gelman plot

The Gelman plot tests if there is significant difference between the chains and the variance between the chains. It is also a convergence diagnostic tool, and complements the traceplot. It is considered good if the functions stabilize at a shrink factor level below 1.1 [12]. This is the case for both the models here, and supports the understanding we got from looking at the trace plots.

Autocorrelation
The lag-k autocorrelation is the correlation between a sample and sample k before it. One wants to see a decreasing autocorrelation as k increases, i.e. meaning that the samples can be considered independent [12]. This is the case in both the models implemented here. The difference is that the autocorrelation drops to 0 after 10 k for the informative model, while the takes about 25 for the uninformed model.

Conclusion
Overall, the two models fitted in this coursework both gave converging chains. Whether one was better than another cannot be stated for certain, only additional data on which we could compare the models' aptness would enable evaluate them further and potentially update our beliefs (priors). Still, one could assume that George et al. [2] had more insights than what was attained for the purpose of this coursework, in which case the informed model may produce the most reliable results.

References
[1] WINBugs Pump example (n.d.) Pumps: conjugate gamma-Poisson hierarchical model. Available at: http://www.openbugs.net/Examples/Pumps.html [Accessed 25 April 2019].

[2] George, E., Makov, U., & Smith, A. (1993). Conjugate Likelihood Distributions. Scandinavian Journal of Statistics, 20(2), 147-156. Retrieved from http://www.jstor.org/stable/4616270

[3] Statistics How. (n.d.) "Poisson Distribution / Poisson Curve: Simple Definition." Available at www.statisticshowto.datasciencecentral.com/poisson-distribution/. [Accessed 28 April 2019].

[4] Medium. (2019). Analytical Bayesian Inference with Conjugate Priors. [online] Available at: https://medium.com/paper-club/analytical-bayesian-inference-with-conjugate-priors-4a1d75ca799b [Accessed 28 April 2019].

[5] Math.stackexchange.com. (2019) What prior to use given a Poisson likelihood? [online]
Available at:
https://math.stackexchange.com/questions/1223036/what-prior-to-use-given-a-poisson-likelihood [Accessed 28 April 2019].

[6] ICEPaC. (n.d.) Hierarchical Models. Available at http://www.columbia.edu/~cjd11/charles_dimaggio/DIRE/styled-4/styled-11/code-8/#hierarchical-models [Accessed 28 April 2019].

[7] ST440/540 Applied Bayesian Statistics at NC State University (2018). One-parameter models - Selecting priors. Available at https://www4.stat.ncsu.edu/~reich/ABA/notes/Priors.pdf [Accessed 29 April 2019].

[8] Stats.stackexchange.com. (2018) Prior comparison: Uninformative vs informative. Available at https://stats.stackexchange.com/questions/327149/prior-comparison-uninformative-vs-informative [Accessed 29 April 2019].

[9] Erler, N. (2019). MCMC Settings. Available at https://cran.r-project.org/web/packages/JointAI/vignettes/MCMCsettings.html [Accessed 28 April 2019].

[10] Stackoverflow.com (2016) How to interpret some syntax in jags? Available at https://stackoverflow.com/questions/38701100/how-to-interpret-some-syntax-n-adapt-update-in-jags/38875637 [Accessed 28 April 2019].

[11] Sbfnk.github.io. (2019). Practical session: MCMC diagnostics. [online] Available at: http://sbfnk.github.io/mfiidd/mcmc_diagnostics.html [Accessed 1 May 2019].

[12] Hartig, F. (2019). MCMC chain analysis and convergence diagnostics with coda in R. [online] theoretical ecology. Available at: https://theoreticalecology.wordpress.com/2011/12/09/mcmc-chain-analysis-and-convergence-diagnostics-with-coda-in-r/ [Accessed 1 May 2019].