HOMEWORK1

Ganna Fagerberg / Statistical Methods

Question 1

1. Write a program that reproduces Figures 1, 3 and 5 following Examples 1 and 3 on pages 168, 169 and 172 of Casella & George (1992) paper.

A. Reproduce Figure 1 and Figure 3

The information we dispose for the analysis is the joint density of the two random variables, X and Y (see expression (2.5) in Casella & George (1992) paper). From this joint density, we can derive the full conditional distributions for each variable: $f(x|\, y) \sim Bin(n, y)$, and $f(y|\, x) \sim Beta(x + a,\ n - x + b)$.

We are interested in sampling from the marginal distribution of the X variable, $f(x)$. The proposed method is to sample by means of the Gibbs sampling method. This is feasible since the full conditional distributions of both X and Y variables are available.

In this particular case, the distribution of $f(x)$ can be obtained analytically – it is recognized as the beta-binomial distribution. Thus, we can compare the effectiveness of the Gibbs sampler by sampling directly from the Beta-binomial distribution (Figure 1) and by calculating its exact probabilities (Figure 3).

To recreate Figure 1 and Figure 3, I resort to the algorithms described on p. 168 and p. 169 of Casella *o*f Casella & George (1992) paper respectively.

The detailed R codes for the assignment can be found in the attachment GFagerberg_Statistical methods_codes (Question 1).

Figure 1

Figure 1 shows two histograms based on samples of the size 500, both obtained by sampling from the Beta-Binomial distribution with parameters $n = 16$, $a = 2$, $b = 4$. The sampling was approached by two methods: direct sampling from the Beta-Binomial distribution and the Gibbs sampling method based on a sequence of size 10.

<u>The Gibbs sampling algorithm</u>:

1. Set the values for the number of iterations, M, and for the length of a Gibbs sequence, k: $M = 500$, $k = 10$.
2. Set the values of the Beta-Binomial parameters: $n = 16$, $a = 2$, $b = 4$.
3. Start iteration 1, $j = 1$, (of 500) and produce a Gibbs sequence of length $k = 10$ by following the steps outlined below.
4. Choose appropriate starting values for X and Y, e.g., $Y^{(1)} \sim Beta(a, b)$, $X^{(1)} \sim Bin(n, y)$;
   - draw $X^{(2)}$ from $Bin(n, y^{(1)})$
   - draw $Y^{(2)}$ from $Beta\left(x^{(2)} + a, n - x^{(2)} + b\right)$
   - repeat until $x^{(10)}$ and $y^{(10)}$ are sampled.
5. Repeat until get 500 Gibbs sequences of length 10.
6. Extract the final values from each sequence, that is $X_j^{(10)}$, $i = 1, \ldots, 500$. This should yield an approximate iid sample of 500 observations from $f(x)$.

Ganna Fagerberg / Statistical Methods

The direct sampling algorithm from Beta-Binomial

We recognize that a Beta-Binomial has a hierarchical structure where the probability parameter of the Binomial distribution is modelled as coming from the Beta distribution, $Beta(a, b)$. Thus, to simulate values from the Beta-binomial distribution, I first simulate probabilities from the Beta distribution and then use these probabilities to generate values from the Binomial distribution as follows.

1. Generate sample probabilities, $prob.beta$, from $Beta(a, b)$.
2. Generate 500 values from the Binomial distribution: $Bin(n, prob.beta)$.

R codes

```
# Gibbs sampling method

for (j in 1:M){ #sampling loop for M
    set.seed(2012*j)
   Y[1,j]=rbeta(1,a,b)  # start values
   X[1,j]=rbinom(1,n,Y[1,j])

  for (i in 2:k) { #sampling loop for
k

    X[i,j]=rbinom(1,n,T[i-1,j])
    T[i,j]=rbeta(1,a+X[i,j],n-X[i,j]+b
) }}

# Direct sampling
prob.Beta <- rbeta(n,a,b)
bin.out <- rbinom(M,n,prob.Beta)
```

Figure 1

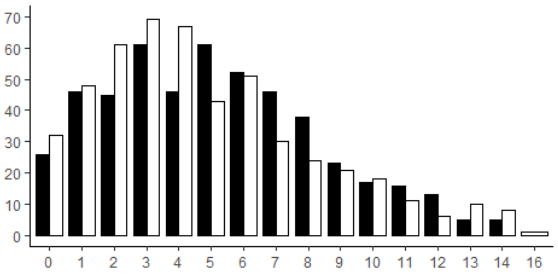


Figure 1. Comparison of Two Histograms of Samples of Size m = 500 From the Beta-Binomial Distribution With n = 16, a = 2, and b = 4. The black histogram sample was obtained using Gibbs sampling with k = 10. The white histogram sample was generated directly from the beta-binomial distribution.

Figure 3

The Figure 3 shows two probability histograms of the Beta-Binomial distribution with parameters $n = 16$, $a = 2$, $b = 4$ constructed (i) by the Gibbs method as described in the algorithm below, and (ii) by resorting to the exact probabilities of the Beta-Binomial distribution.

The algorithm behind the Gibbs sampler:

1. Set the values of M, k, n, a, b as in the previous example.
2. Extract the values of $theta = Y[10,]$ simulated in the previous example, that is the final values of Y extracted from each sequence, $Y_j^{(10)}$, $j = 1, \ldots, 500$.

3. Use $theta$ to generate the binomial density for a given range of X values, i.e., for $X \in [0,16]$ as according to expression (2.11) in Casella & George (1992) paper.

The algorithm behind computing the exact probabilities of the Beta-Binomial distribution:

1. Set up the marginal distribution of X.

2. Compute the probabilities for the possible set of X values, i.e. for $X \in [0,16]$.

R codes

```
# Gibbs sampling method
Theta<-T[10, ]
x=c(0:16)prob.X<- sapply(x, function(x){m
ean(dbinom(x,n,Theta))})

# Exact probabilities
Gam_fun<- function (x, n, a, b) {
result <- choose(n,x)*(gamma(a+b)/
gamma(a)*gamma(b)))*
((gamma(x+a)*gamma(n-x+b)/gamma(a+b+n)))}
exact_res<- Gam_fun(x,n,a,b)
```

Figure 3

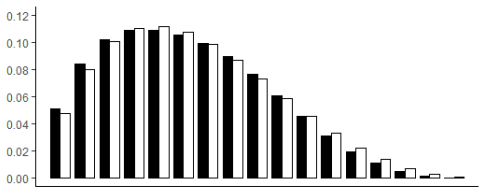


Figure 3. Comparison of Two Probability Histograms of the Beta-Binomial Distribution With n = 16, ct = 2, and f3 = 4. The black histogram represents estimates of the marginal distribution of using Equation (2.11), based on a sample of Size m = 500 from the pair of conditional distributions in (2.6). The Gibbs sequence had length k = 10. The white histogram represents the exact beta-binomial probabilities

Figure 5.

In this example, we dispose of the joint density of the three random variables, X, Y, N, as given in expression (4.8) (see Casella & George (1992) paper). N is now a realization from a Poisson process. The full conditional distributions of the three variables are given in expression (4.9) (see Casella & George (1992) paper).

We are interested in computing the marginal probabilities of the X variable using equation (2.11) (see Casella & George (1992) paper). To compute the characteristics of $f(x)$, we follow the same algorithm as for two random variables, with the caveat that we extend it to the case of three random variables. The algorithm is outlined in Casella & George (1992) paper, on p. 172 (see expression (4.6)).

To be able to reproduce the algorithm, first, I make following transformations.

Given expression the full conditional density for N is proportional to

$$f(n\,|x,y) \propto e^{-(1-y)\lambda}\frac{((1-y)\lambda)^{n-x}}{(n-x)}, n = x, x+1, \ldots,$$

I introduce a new variable Z so that

$$Z = N - X$$

Then, the distribution of Z is Poisson, $Z{\sim}Po(\lambda(1-Y))$. Now, we can sample iteratively from

$$X{\sim}Bin(n,y)$$

$$Y{\sim}Beta(x+a, n-x+b)$$

$$N = X + Z, Z{\sim}Po(\lambda(1-Y))$$

The algorithm behind the Gibbs sampler

1. Set the values of $M = 500, k = 10, \lambda = 16, a = 2, b = 4$
2. Set a Gibbs sequence of length $k = 10$ and start iteration 1, $j = 1$, (of 500), according to following steps:

- hoose the appropriate starting values for X, Y, Z, e.g., $N^{(1)} \sim Poi(\lambda)$, $Y^{(1)} \sim Beta(a, b)$, $X^{(1)} \sim Bin(n^{(1)}, y^{(1)})$,
- draw $X^{(2)}$ from $Bin(n^{(1)}, y^{(1)})$,
- draw $Y^{(2)}$ from $Beta(a + x^{(2)}, n - x^{(2)} + b)$
- draw $N^{(2)} = X^{(2)} + Z \sim Poi((\lambda(1 - Y^{(2)}))$

3. Repeat until we get 500 Gibbs sequences of length 10.
4. Extract the values of $theta = Y[10,]$ and $N = N[10,]$ simulated in the previous step, that is the final values of Y extracted from each sequence, $Y_j^{10}, j = 1, \ldots, 500$ and the final values of N extracted from each sequence, $N_j^{10}, j = 1, \ldots, 500$.
5. Use $theta$ and N to generate the binomial density as according to formula (2.11) in Casella paper a range of X values, i.e., for $X \in [0,20]$.

R codes

```
# Gibbs sampling method
lambda=16 # parameter of a poisson
X=Y=N=array(0,dim=c(k,M))
for (j in 1:M){
  set.seed(2012*j)
  #set initial values
  N[1,j] <- rpois(1,lambda)
  T[1,j] <- rbeta(1,a,b)
  X[1,j] <- rbinom(1, N[1,j], Y[1,j])
for (i in 2:k){ #sampling loop
X[i,j]=rbinom(1,N[i-1,j],Y[i-1,j])
Y[i,j]=rbeta(1,a+X[i,j],N[i-1,j]-X[i,j]+b)
N[i,j]=X[i,j]+rpois(1, lambda=(lambda*(1-Y[i,j
]))))}}


T.vec <- T[10,]; N.vec<-N[10,]
grid=seq(0, 20, by=1)

prob.X_pois = as.data.frame(cbind(grid,sapply(
grid,function(grid){mean(dbinom(grid,N.vec,T.v
ec))})))
```

Figure 3

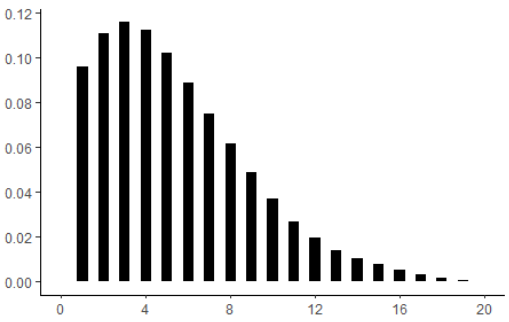


Figure 5. Estimates of Probabilities of the Marginal Distribution of X Using Equation (2. 11), Based on a Sample of Size m = 500 From the Three Conditional Distributions in (4.9) With A = 16, a= 2, and:1 = 4. The Gibbs sequences had length k =10.

2. Analyze these figures using your own words and make conclusions about effectiveness of the Gibbs sampler. In particular, discuss right tails of figures 3 and 5: compare them, explain.

Figure1. It seems that the Gibbs sampler does a relatively good job in sampling as compared to direct sampling from the Beta-Binomial distribution.

Figure 3. As advised in Casella & George (1992) paper, I have used the Y values computed for Figure 1 to more accurately evaluate the distribution of X. Indeed, the empirical distribution of X values obtained by this method seems to closely follow the exact beta-binomial distribution, as can be seen in Figure 3. It confirms the conclusion made by Casella & George (1992) that calculating the distribution of X conditional on the simulated Y values "...carry more information about $f(x)$ than $x_1, .., x_m$ alone, and will yield better estimates." (p.169).

Figure 3 and Figure 5. The right tail of the distribution in Figure 5 is longer than that of Figure 3. This is because to reproduce Figure 5, we additionally assumed that N is a random variable by contrast from the model for Figure 3 where we assumed a fixed N. X is essentially modeled hierarchically, as

coming from a binomial process with parameters Y and N that are also modelled as coming from a Beta and a Poisson processes respectively. Since N is modelled as coming from a Poisson process with a constant mean parameter $\lambda = 16$, the resulting distribution will naturally be more variable than when is N is assumed fixed and equal 16.

3. Implement more advanced versions of Gibbs sampler than suggested in the paper: use burn-in, test for convergence, etc.

There are several approaches to generating Gibbs samples. Another approach described in Casella & George (1992) is to generate a long Gibbs chain/sequence of length M, discard a portion of observations simulated in the beginning of the process up to $j's$ observation, i.e., all observations for which $j < M$ (this is the so-called burn-in period). The empirical distribution of the remaining data is expected to converge towards the target distribution, $f(x)$, despite the fact the observations will still exhibit some dependent structure.

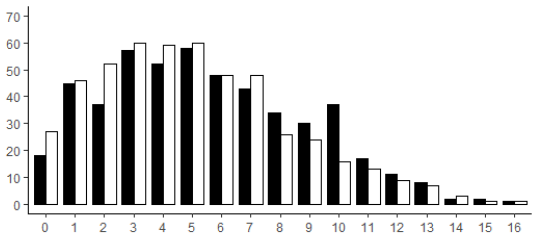
It is generally *a priori* unknown how many observations must be generated in order for the chain to converge, how large the burn-in period should be, or whether the chain would at all converge after some M draws. Therefore, after each simulation, the chain should be tested for convergence. These can be done in several ways, both by visual inspection of the diagnostics plots and by applying the statistical tests.

For this example, I will reproduce Figure1 (p. 168) by generating a long Gibbs chain. Thus, I am interested in simulating 500 observations from a Beta-binomial distribution with the parameters $n = 16$, $a = 2, b = 4$. I will resort to a conservative approach and discard the first half of the simulated observations. Therefore, the length of simulation chain is set to 1000. Next, I compare the characteristics of the obtained sample with the sample obtained by directly sampling from the beta-binomial distribution (as for Figure 1). The two histograms are given below. The two histograms seem to be quite similar, and it seems that the chosen Gibbs approach to simulating data works satisfactory.

R codes

```
M = 1000 #take larger number of iterations
burn.in= 1:500 #
X=T=array(0,dim=c(M,1))
# Perform Gibbs iterations
for (i in 2:M) {
  T[1]=rbeta(1,a,b)
  X[1]=rbinom(1, n, T[1])
  X[i] =rbinom(1,size=n,prob=T[i-1])
  T[i] = rbeta(1,a+X[i],b+n-X[i])
}
# Discard burn-in
X.out = X[-burn.in]
T.out = T[-burn.in]
```

Histograms



Comparison of Two Histograms of the Beta-Binomial Distribution With n = 16, a = 2, and b = 14 based on a sample of 500. The black histogram represents estimates of the marginal distribution based on a Gibbs sample of length 1000. The white histogram represents direct sampling from the beta-binomial dostribution

The diagnostics of the generated sample

As said, every MCMC sequence require checking for convergence to a stationary distribution. There are several convergence criteria. I will check the convergence by means of (i) trace plots, (ii) running average plots, (iii) autocorrelation (ACF) plots and (iv) by means of Gelman and Rubin diagnostics.

Trace plots. Trace plots allow to investigate how well the chain is mixing, i.e., how well the chain is moving around the parameter space. For example, the chain is said to mix badly if it hovers in certain

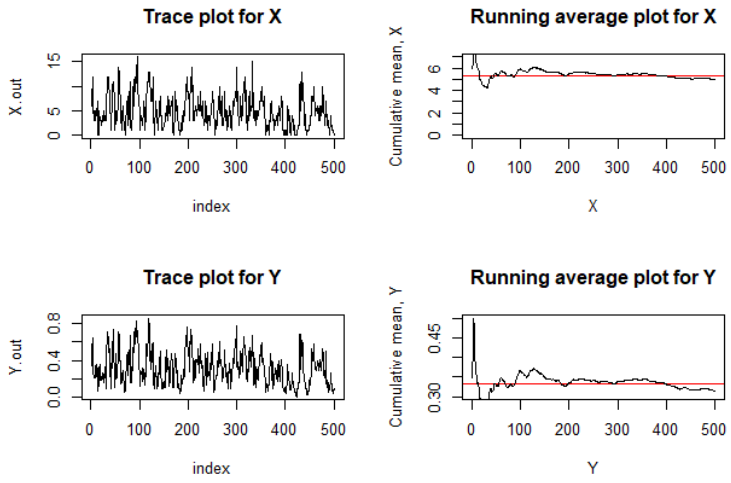
areas of the parameter space. The mixing behavior of all parameters should be investigated. The trace plots for X and Y, along with the codes are given below. I will check the mixing behavior of the chain after I have discarded the first 500 observations. The trace plots for two parameters are given in Table 1.

It seems that our chain is mixing well, although it is difficult for me to draw any definitive conclusions when I check the behavior of the chain in the region of iteration steps 400-500. This might be an indication that our chain is stuck or mixing slowly.

Running average plot. Another way of visually inspecting the chain for convergence is by means of running average plots. It is a plot of the iteration against the mean of the draws up at each iteration and shows how fast the empirical mean convergences towards the true mean. The running average plots for two parameters are also given in Table 1.

The true (theoretical) mean of the $Beta - Binomial\ (16, 2, 4\ )$ is 5.33, of the $Beta(\ 2, 4\ )$ it is 0.33. These means are shown by the red lines in the running average plots for X and Y respetively. I think that these plots might indicate that we should try to produce a longer chain (more than 1000 observations) and investigate the properties of the distribution afterwards.

Table 1. Trace plots and running average plots for X and Y

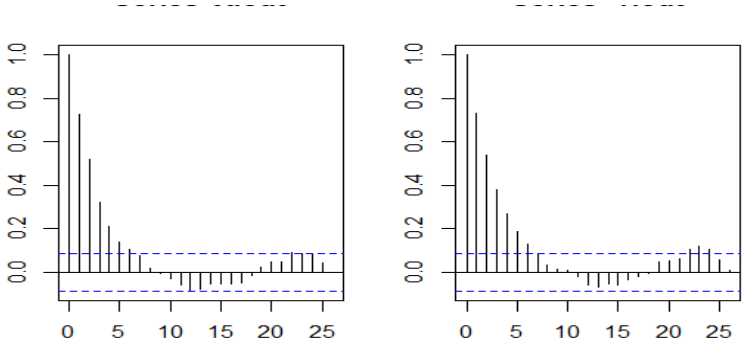


ACF plots

The ACF plots are used to assess the autocorrelations between the draws at different lags. Generally, the sample will show some degree of dependence, though a good sample will show a fast decaying autocorrelation patter. An autocorrelation pattern that shows a persistent correlation pattern may ndicate, e.g., that we need to run the sequence for a longer period, or that a thinning of a sequence at a certain lag may remedy for the problem.

Figure 1 shows ACF plots for X and Y variables. The autocorrelation patterns for both variables seem to decay reasonably fast. Some persistent patterns might be discerned after lag 6, though they are hardly significant. Though the ACF plots are not indicative of major problems with autocorrelation, an investigation of the behavior of a larger sequence and potentially thinning can be recommended.

Figure 1. ACF plots for X and Y



Gelman and Rubin sequence diagnostics

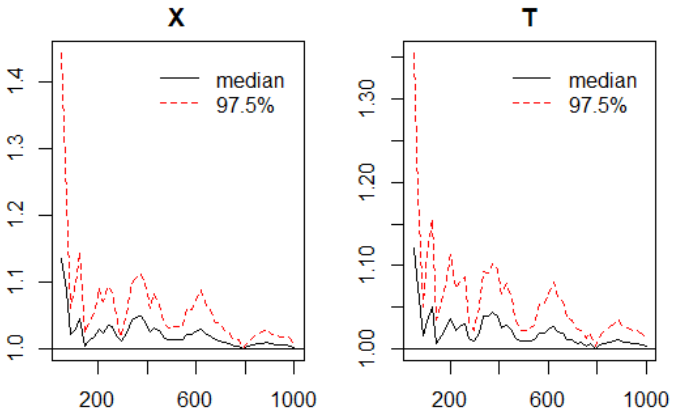
The Gelman and Rubin diagnostics methods is a popular approach for checking the convergence of an MCMC sequence. The diagnostics serves as an indication of whether the burn-in period or an MCMC run-length should be prolonged. To run the method, several chains (minimum two) of equal length are generated from different (over-dispersed) starting values. The within-chain variance and the between-chain variance are then computed as according to the formulas given in Casella & Robert (Introducing Monte Carlo methods with R, 2010, p.253) and subsequently compared. When the between-chain variance is considerably larger than the within-chain variance, it serves as an indication that that the sequence should be run for a longer period. The test statistics which is used to this end is the scale reduction factor. If it is equal to 1, the between-chain and within-chain variances are deemed as equal, and the chain is deemed to converge. If it is larger than 1 (usually, larger than 1.1 or 1.2), than it might indicate that the chains should be run for longer.

I will create four chains with different starting values to apply the test upon. The different starting values will be accounted by different seed values for each chain. I will simulate the starting values for the Y variable both from the uniform and the beta distribution. The obtained scale reduction statistic for X and Y parameters is 1 (with an upper CI of 1.01). The results of the test for the two parameters signify convergence of the chains.

We can also check how fast the convergence occurs visually by means of Gelman and Rubin scale reduction plots. The plots show how the scale reduction factors for the parameters change at different iteration steps (see Table 2).

We can see that the shrink factor seems to tend toward 1 for both parameters for larger iterations. However, its behavior does not seem to be stable, e,g., after step 500 (the projected burn-in period). This might indicate that the burn-in should be made longer, as well as the number of iterations. We could, for example, generate 1500 observations, or try applying a suitable thinning approach to collect 500 observations.

Table 2. Gelman and Rubin scale reduction plots



Ganna Fagerberg / Statistical Methods

Question 2

1. Discuss the concept of conjugate priors

If the form of a posterior distribution has the same form as a prior distribution, such a prior is called conjugate given its likelihood function. Conjugate priors have convenient mathematical properties (the posteriors can be derived analytically), but there exists only a handful of conjugate priors to some standard likelihoods (e.g., Bernoulli, normal likelihoods). The conjugate priors are easy to work with analytically, they often provide a good approximation to reality, and are easy to interpret. However, being convenient approximations of reality, they may nevertheless fail to incorporate full and more complex prior information (Gelman *et al.*, Bayesian data analysis, third edition, 2014).

2. Describe clearly the data set and discuss how you would model it.

The data we dispose of are the number of bomb hits in London WW2. The data is modeled by the Poisson distribution by dividing London into 576 regions. Totally, there are 537 hits in 576 regions. The average number of hits per region is thus $537/576 = 0.9323$.

If we denote the number of hits by Y and the number of regions by N, we can present the model as

$$y_i \sim Poi(\theta), i = 1, \dots, N,$$

where θ is the mean rate parameter, i.e., it stands for the average number events that occurs in a fixed interval of (here) space, and y_i are iid.

The aim of the analysis is to infer θ in the framework of the Bayesian analysis, that is we are interested in deriving the posterior distribution for θ. To this end, a $Gamma$ with varying shape, a, and rates, b, are suggested as a prior for θ, $\theta \sim Gamma(a, b)$. Given that a Beta distribution is conjugate to the Poisson likelihood, the resulting posterior distribution is conveniently proportional to the Gamma distribution, i.e. $\theta | data \sim Gamma(a + Y, b + N)$.

The parameters of the prior Gamma distribution tried for the analyses are following: a and b can take on values $ca.\ 0,\ 10,$ and 100. The combination of these parameters gives nine different priors.

The detailed R codes for the assignment can be found in the attachment GFagerberg_Statistical methods_codes (Question 2).

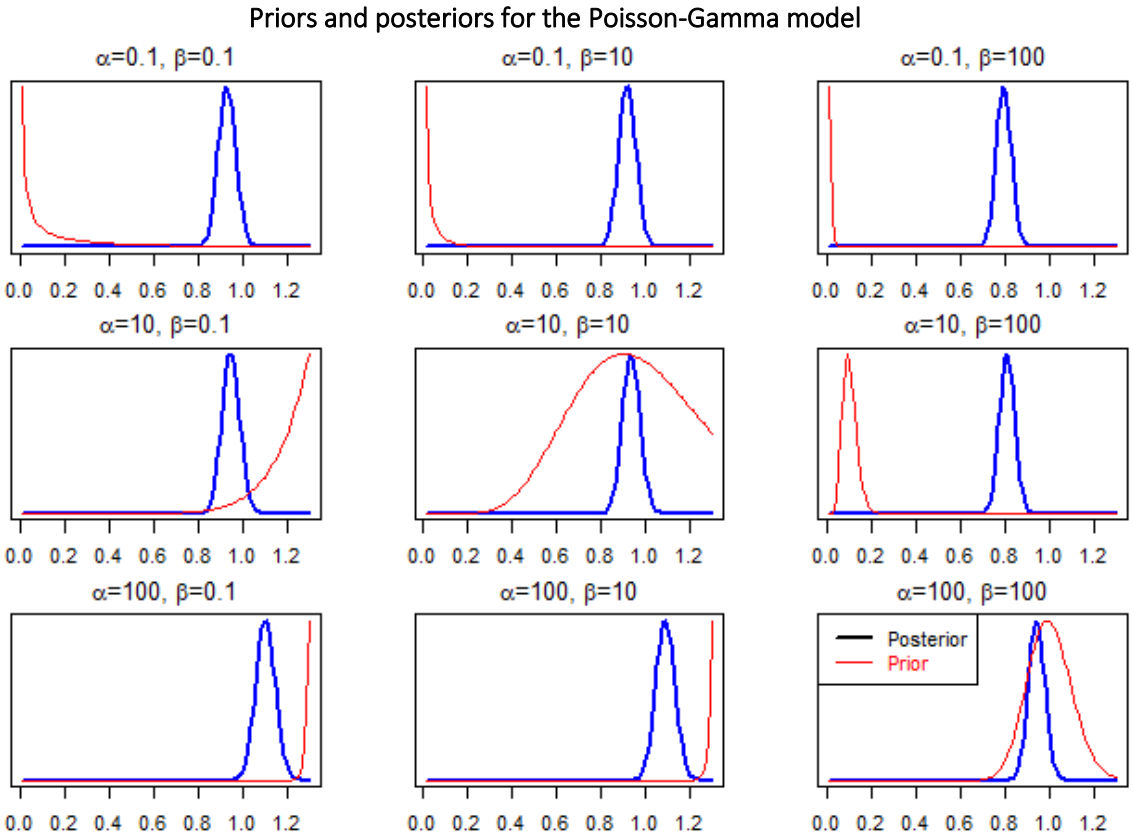
3. *Reproduce Figure on page 7 /L2/Poisson_Example from the lecture notes that exemplifies nine different priors. Attach your program and its output. Comment extensively your code.*

R codes

```
gamma_fun<-function(shape, rate){
  # set seed for reproducibility
  set.seed(1979)
  # generate data from the Poisson as according the observed distribution
  y = rpois(576,0.9373)
  # set a grid for a range of theta values
  grid = seq(0.01,1.3,0.01)
  #compute densities of the prior given grid
  d.prior=dgamma(grid, shape, rate)
  # compute densities of the conjugate posterior
  d.post=dgamma(grid, shape+537, rate+576)
  # compute normalising constant
  const=max(d.post)/max(d.prior)
  # update the prior
```

```
  d.prior=const*d.prior
  # parameters of the graphs
  par(mar = c(2, 2, 2, 2))
  # posterior density plot
  p=plot(grid, d.post,  type="l", lty = 1, lwd=2,  xlim=c(0.01, 1.3), yaxt='n',
         xlab="", ylab="", col="blue")
  # graph plots in the same box
  par(new=TRUE)
  # prior density plot
  p=plot(grid, d.prior, type = "l", pch=15,  xlab="", ylab="", xlim=c(0.01, 1.3),
         axes=FALSE, col="red", main=substitute(paste(alpha,"=",shape,", " , beta,"=",rat
e))))}
```

Mow we can use the *gamma_fun* for the nine combination of the parameters to reproduce the figure on p. 7 of the Poisson example.



4. Provide visualization/discussion on credible intervals reported on page 8. Make clear conclusions what method is preferred for the credible intervals and why.

Here we are presented with three types of Bayesian intervals - an approximate 95% credible interval, exact 95% equal-tail interval (assuming a=b=0), and exact high posterior density interval, (assuming a=b=0).

(i) Approximate 95% credible interval. This method is easy to implement and implies that the posterior can be approximated with a normal form distribution. As in a frequentist setting, we use the quantiles of a normal distribution to construct a symmetric interval around the distribution's mean.

Thus, this method implies that we dispose of a large number of observations for the analysis, and that the posterior of interest is unimodal and approximately symmetric.

I simulated 100,000 observations from the Gamma posterior in the bomb hits study assuming small values for a and b. The 95% approximate interval is $[0.8535, 1.0112]$.

(ii) Exact 95% equal-tail interval (assuming a=b=0). This standard central interval is derived from a posterior probability distribution by using distribution's $\frac{\alpha}{2}$ and $1-\frac{\alpha}{2}$ quantiles. However, if the posterior is skewed or extends to the boundary of its parameter space, an exact interval may not be a sensible choice.

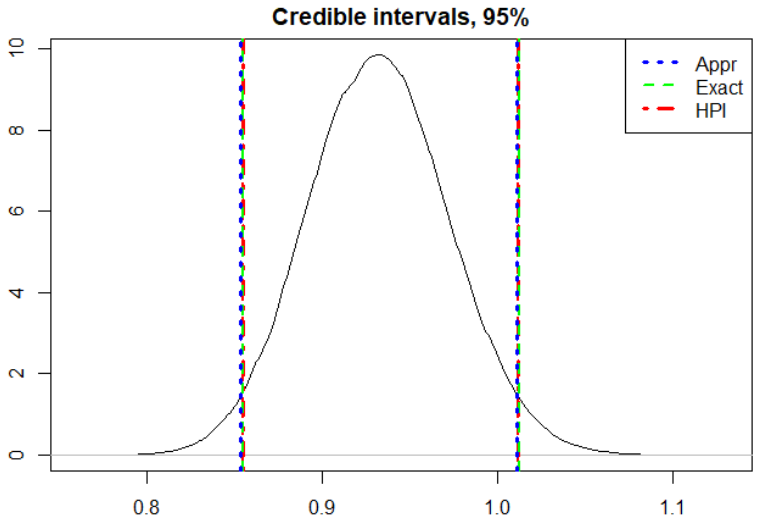
The exact 95% interval for the Gamma posterior in the bomb hits study assuming $a = b = 0$ is $[0.8551, 1.0128]$.

(iii) *Exact high posterior density interval (HPI)*. HPI is a $(1-\alpha)$ credible interval that enclose the $(1-\alpha)$ posterior probability with the largest posterior density. The computed HPI interval is $[0.8548, 1.0123]$.

Generally, HPI is preferred over the alternatives mentioned above for skewed, multimodal or non-standard posterior distributions. However, when posterior probability distributions are unimodal and symmetric, all three intervals usually lead to identical results.

This is true for the bomb hits analysis case. The posterior distribution depicted in Figure 2 is unimodal, symmetric, and we can see that all three intervals – approximate, exact and HPI - coincide. In this particular case, with relatively large number of observations, any of the intervals can be used.

Figure 2. Visualization of the three types of intervals



Ganna Fagerberg / Statistical Methods

Question 3

1. *Your task is to reproduce Figure 1 on page 5, using any programming tool, except python. You are expected to streamline Algorithm1 (p.2) to the problem, clearly indicate what proposal q(.;.) you use (formula (7), p.5) and modify it in the way that you get different levels of acceptance*.

The aim of the assignment is to infer the correlation parameter, ρ, between the two variables, X and Y in the framework of the Bayesian analysis. The two variables are assumed to follow the bivariate normal distribution. For model simplicity, the variables are standardized, i.e., the means and variances of X and Y and equal to 0 and 1 respectively. The variance-covariance (here correlation matrix) is of the following form

$$\Sigma = \begin{matrix} 1 & \rho \\ \rho & 1 \end{matrix}$$

The likelihood function is given as in expression (4) of the "Bayesian Inference: Metropolis-Hastings Sampling" paper (2014). The prior assigned to p is the analogue of the Jeffrey's prior for the bivariate distribution, $\frac{1}{|\Sigma|^{\frac{3}{2}}} = \frac{1}{(1-\rho^2)^{\frac{3}{2}}}$. Under the Bayesian rule, the posterior density for ρ , $p(\rho)$, is proportional to expression (6) in "Bayesian Inference: Metropolis-Hastings Sampling" paper (2014).

This density is not of a known form, so the Gibbs sampler is not a straightforward option here. We use The Metropolis-Hastings (MH) method to sample from this distribution as according to the algorithm described on p. 2 of the paper mentioned above. The MH method requires that we choose a suitable proposal or jumping distribution so that the acceptance rate is satisfactory.

To reproduce Figure 1 (p. 5), I will use the proposal distribution, $q(\rho)$ that is suggested by the authors of the paper. This is a uniform distribution centered around the current value of ρ with an overall width of 0.14:

$$\rho_{i+1} = \rho_i + Uni(-half.length, half.length),$$

Where the $half.length$ is set to 0.7.

When the proposals are of a symmetric form, this is known as a random walk Metropolis sampling.

Whether or not the candidate sample from a proposal distribution is accepted or not, is regulated by the MH acceptance ratio, R:

$$R = \frac{p(\rho^{(i)})q(\rho^{(i-1)}|\rho^{(i)})}{p(\rho^{(i-1)})(\rho^{(i)}|\rho^{(i-1)})}$$

Note that for the symmetrical distributions, as in this case, it simplifies to the ratio of the posterior distributions,

$$R = \frac{p(\rho^{(i)})}{p(\rho^{(i-1)})}$$

To check how effective the MH method is for estimating the parameter ρ, we simulate two random variables, each of size $N = 1000$, from the bivariate distribution with a positive correlation, ρ, of 0.4 between the variables.

The codes for the function that employs the MH algorithm to sample from the posterior is given below. The detailed R codes for the assignment can be found in the attachment GF_Statistical methods_codes (Question 3).

R codes

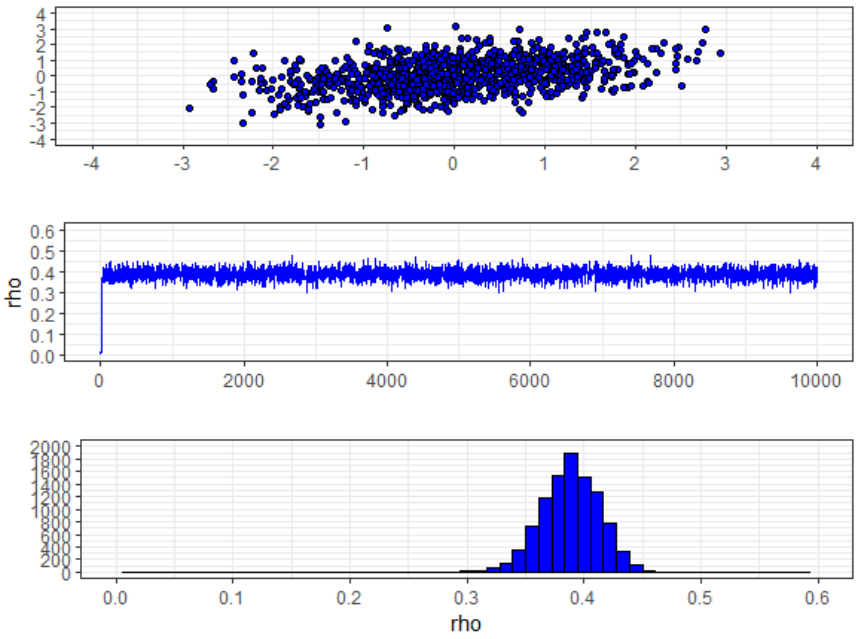
```
MH.fun = function(M, rho, half.int,  verbose = TRUE) {

    set.seed(2009)
    # create the function that evaluates the log of the post density
    logpost = function(rho){
    if(rho>-1 & rho <1) return(-3./2*log(1.-rho**2) - N*log((1.-rho**2)**(1/2)) -
                         sum(1./(2.*(1.-rho**2))*(x**2-2.*rho*x*y+y**2)))
    else return(-Inf)}

    accepted_number=0 # start counter for the accepted samples
    chain_rho = vector("numeric", M) # vector to store the samples
    rho=0 # starting value

    #start the loop
    for (i in 1:M) {
    # draw a sample from the proposal distr (Equation 7)
    rho_candidate=rho + runif(1,-half.int, half.int)

    # Compute the acceptance probability (Eq. 8 and Eq. 6) (in log domain)
    num=logpost(rho_candidate); den=logpost(rho)

    accept=num-den; accept=min(0,accept) # 0 as we are operating in log domain
    accept=exp(accept) # transform to original scale

    # Accept rho_candidate with probability accept
    if (runif(1,0,1) < accept) {
      rho=rho_candidate
      accepted_number=accepted_number+1
    }
     chain_rho[i]=rho}

  if(verbose) cat("acceptance rate:", ((accepted_number/M)), "\n")
  list(chain_rho, acceptance_rate=accepted_number/M)}
```

Setting the starting value of ρ to 0, we get the acceptance rate of 0.50. For symmetric proposal distribution the acceptance rate is recommended to lie in the interval ca. $(0.30 - 0.60)$. Now, we have all the data to reproduce the figure 1 on p. 5 (see Figure 3).

To get a different level of acceptance, I proceed by modifying the width of the uniform interval by changing the values for $half.length$ parameter from 0.05 to 0.5 $(by\ 0.02)$., so that we get 23 different values for the $half.length$ of the interval. I use the same function to produce the acceptance rate as for reproducing Figure 1, by slightly modifying the output of the function to produce the mean of the simulation sample, the standard error and the acceptance rate for each value of the $half.length$.

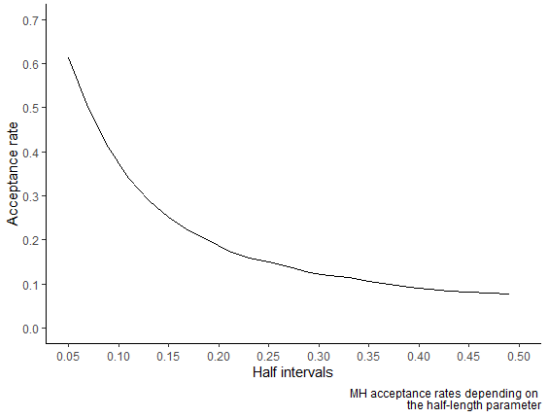
Note that in those cases when the simulated ρ values lie outside the parameter space, that is when they are larger than 1 or smaller than -1, the log-posterior function is programmed to return (-inf). It means that the algorithm will have a low acceptance rate if the simulated ρ often lies outside its boundaries.

Figure 3. Reproduction of Figure 1 on p. 5



The acceptance rates for various values of $half.length$ are shown in Figure 4. The acceptance rates for the values in the range $(0.07, 0.11)$ are satisfactory. For 0.07 the rate is ca. 0.50, for 0.11 it is ca. 0.34. The estimated mean of ρ and its standard error are similar for all the values of $halh.length$, with the mean of ca. 0.43, and the standard error of ca. 0.03.

Figure 4. The acceptance rates for the uniform symmetrical proposal



2. *You are expected to use uniform, normal and at least one non-symmetric proposal and discuss your findings in detail. Discuss these differences and try to find optimal proposal.*

Uniform symmetrical proposal distribution. I used the uniform proposal symmetrical distribution to illustrate the previous case.

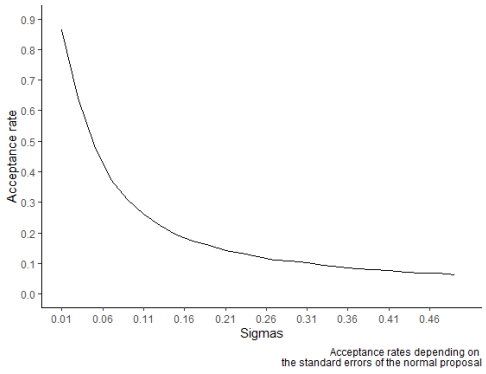
Normal symmetrical proposal distribution. I will use another common random walk Metropolis sampling method based on the normal proposal distribution centered around the current value of the parameter, i.e.

$$\rho_{i+1} = \rho_i + N(0, sigma)$$

I check the acceptance rate of the MH sampling method depending on the standard error, $sigma$, of the normal proposal. I simulated 25 values of sigma in the range between $0.01\ and\ 0.5$ (by a step of 0.02). In those cases when the simulated ρ lies outside its parameter space, that is when it is larger than 1 or smaller than -1, the log-posterior function is programmed to return (-inf).

The acceptance rates for various values of $sigma$ are shown in Figure 5. The acceptance rates for the values in the range $(0.05, 0.09)$ are deemed satisfactory. For 0.05 the rate is ca. 0.49, for 0.09 it is ca. 0.31. The estimated mean of ρ and its standard error are similar for all the values of $sigma$, with the mean of ca. 0.43, and the standard error of ca. 0.03.

Figure 5. The acceptance rates for the normal symmetrical proposal



Normal non-symmetrical proposal distribution

Here I will use the independent non-symmetrical proposal distribution, that is the proposal that foresees that the candidate values are drawn independently of the past values. Ideally, the proposal distribution should resemble the target distribution and cover it in the tails. In this case, the likelihood is high that the samples we thus obtain are approximately independent draws from the target distribution. By contrast, if the proposal distribution differs from the target distribution, the likelihood is high that the chain will be stuck at particular points for long periods of time. This may lead to low acceptance probabilities and highly correlated samples.

The proposal distribution I use here is the independent normal proposal distribution centered around zero

$$\rho_{i+1} = N(0, sd = sigma)$$

Note that since the proposal is independent, the acceptance rate we use in this case is

$R = \frac{p(\rho^{(i)})q(\rho^{(i-1)}|\rho^{(i)})}{p(\rho^{(i-1)})(\rho^{(i)}|\rho^{(i-1)})}$. The values for $sigma$ vary from 0.05 to 0.2.

The overall results of the simulation are non-satisfactory. The acceptance rate is less than 1% for all the values of sigma. The proposal does not cover well the target distribution.

Discuss these differences and try to find optimal proposal

There exists a variety of proposals for the Metropolis–Hastings algorithm. I have considered here a generic Metropolis–Hastings algorithm random walk algorithm and the independent Metropolis–Hastings algorithm. The independent MH requires a high acceptance rate, and usually the performance of the independent samplers, among other things, can be compared by the acceptance rate. A low acceptance rate for the MH may indicate that the samples are highly correlated. The performance of the random walk MH by contrast cannot be measured by the high levels of the acceptance rate alone. The high acceptance rate may indicate that the chain is moving too slowly on the surface of the target distribution. A low acceptance rate is less of an issue for the random walk MH. However, it may signal that the chain misses some parts of the distribution and require a larger number of simulations.

I have not managed to find a good independent proposal for the target distribution. The uniform and the normal symmetrical distributions seem to provide satisfactory results. The simulated samples should also be checked for convergence.

3. Can you extend this analysis to the three streams of observations and model their correlation?

We extend the previous analysis with three streams of observations and assume the following. The three random variables, X, Y and Z, follow the multivariate normal distribution; the means of all variables are equal to 0, their respective variances are equal to 1, i.e., the variables are standardized. We assume also that there are positively correlated. We assume also that the correlation between the variables is positive, as in the previous case is equal to 0.4. That the variance-covariance matrix is equal to

$$\Sigma = \begin{matrix} 1 & 0.4 & 0.4 \\ 0.4 & 1 & 0.4 \\ 0.4 & 0.4 & 1 \end{matrix}$$

The prior for the correlation parameter is Jeffreys prior's analogues for multivariate normal distribution and in the case of three normal variables it is equal to

$$\frac{1}{|\Sigma|^2} = \frac{1}{\left((2\rho+1)(\rho-1)^2\right)^2}$$

The updated posterior distribution given the multivariate normal likelihood for three streams of observations is

$$L = \frac{1}{(2\rho+1)(\rho-1)^2} \prod_i^N \frac{1}{(2\pi)^{\frac{3}{2}}\sqrt{((2\rho+1)(\rho-1)^2)}} \exp\left(-\frac{1}{2(2\rho+1)(\rho-1)^2}((1+\rho)\left(x_i^2+y_i^2+z_i^2\right)-2\rho(x_i y_i + x_i z_i + y_i z_i)\right),$$

given that $\rho \neq 1$ and $\rho \neq -\frac{1}{2}$ (otherwise the inverse matrix of Σ does not exist).

The proposal distribution is symmetric uniform centered around the current value of ρ

$$\rho_{i+1} = \rho_i + Uni(-0.07, 0.07)$$

The codes are analogous as those employed for producing Figure 1 on p. 5 but extended to the case of three variables.

The acceptance rate is deemed satisfactory, ca. 0.40; the estimated mean is ca. 0.39, with a standard error of ca. 0.02.

The trace plot for the sample as well as the ACF plot does not seem to show any deviant behavior (see Figure 6). However, upon closer inspection of the ACF lot some thinning might be recommended.

Figure 6. Trace plot and ACF plot for ρ