Giosuè Migliorini 3145491

Andrea Raminelli 3036220

Table of contents

1. Introduction

In this report we describe the implementation of two MCMC algorithms: Metropolis Hasting and Gibbs Sampler with auxiliary variables. Both are applied to compute a sample from the posterior distribution of the coefficients of a probit model. We implemented the model both with conjugate and with flat priors. We also test the algorithms changing the parameters to check their reliability. In the Gibbs Sampler we introduce the short chains. This mechanism makes the chain restart at the initial value after a given number of iterations. In the end we compare the results of the two algorithms.

As can be seen in the figure, in order to conduct the analyses we simulated a dataset of k = 3 parameters fixed by us at β = [1, 1, 1]. We considered 200 observations simulating the independent variables X and computing Ys according to the logic of the probit model. We fixed the seed in order to maintain the same observations for all our trials. The results are commented, with the diagnostics relying on the trace plot of the Markov Chain and on the convergence of the sample mean to the true value where the posterior was centered.

```
k = 3 #number of variables
n = 200 #number of observations
X = np.random.normal(size=(n, k))
true_beta = np.array([1,1,1])
p_true = stats.multivariate_normal.cdf(X @ true_beta)
Y = stats.bernoulli.rvs(p_true)
print(true_beta)
```

```
[1 1 1]
```

2. Metropolis Hasting

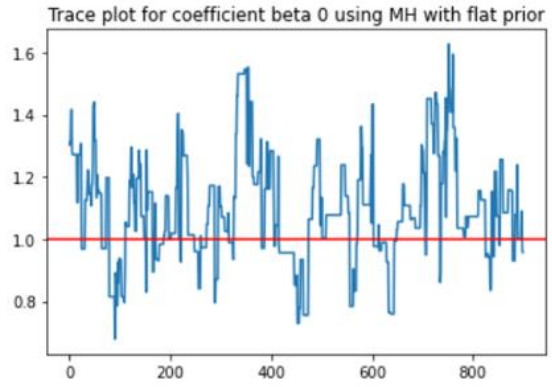
For the Metropolis algorithm first we computed the posterior distribution in levels using as the likelihood the Bernoulli distribution of the Ys. In the case of a conjugate prior we introduced a multivariate normal whose parameters must be defined in the function. Then we computed the Fisher Information, necessary to compute the variance of our proposal. In order to compute the FI we computed the weights using the method for Generalized Linear Models. Indeed, we used weights *W* equal to the reciprocal of the variance of our Bernoulli distribution evaluated for each proposal at β_{t-1}, and the first derivative of the distribution function of our normal (i.e. the probability function of a normal distribution) squared. The Fisher information matrix is then equal to $X^T W X$.

After that we computed the proposal as a multivariate normal distribution centred at β and with covariance matrix given by the inverse of Fisher information evaluated at the current update we had just calculated. Right after we computed the acceptance probability and compared it to a uniform in order to select the β. We only considered the observations after the 100th term. In the end we computed the acceptance ratio and showed the trace plots highlighting the true values.

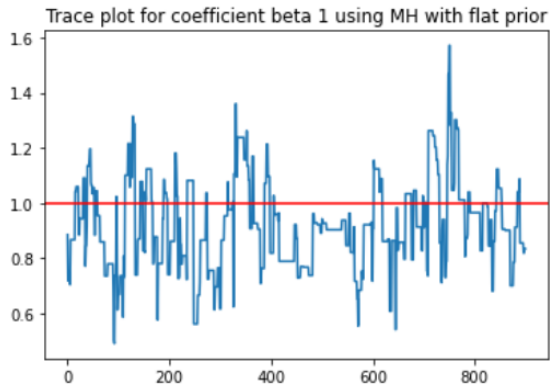
2.1 Flat priors

When implementing the algorithm we faced the choice of the variance parameters, and we set it in order to get an acceptance as near as possible to 0.234 according to the suggestion Robert, Gelman, Gilks(). The results converge and the high variance makes it possible for the chain to explore widely the sample space, although often this leads to rejection of the proposed βs.

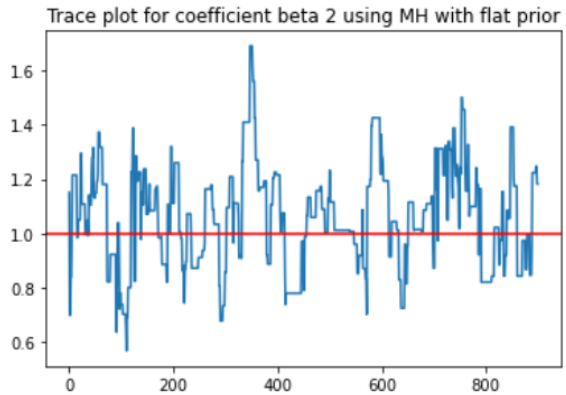
```
Acceptance ratio: 25.1%
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.09553043366673108
```




```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: -0.07989065234118065
```

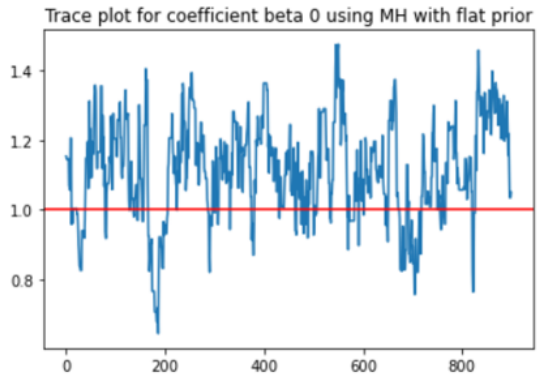



```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.0628426573945049
```

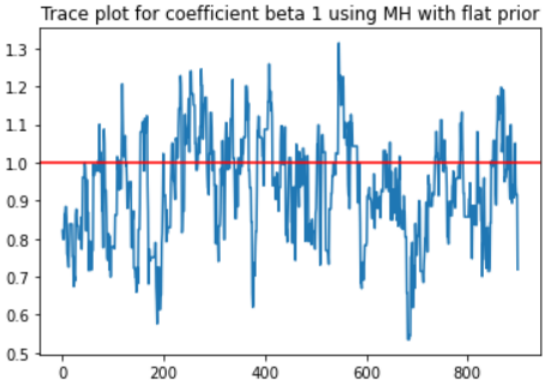


For the sake of completeness, we include the results for a lower variance. We set the variance parameter to 5 and the results are the following. They don't seem to differ much from the previous trial.

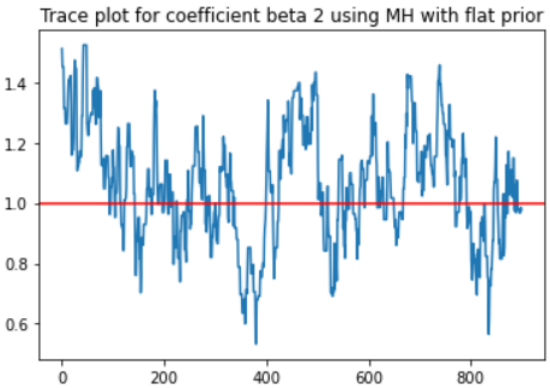
```
Acceptance ratio: 59.6%
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.11601340718000852
```




```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: -0.06983441284208536
```



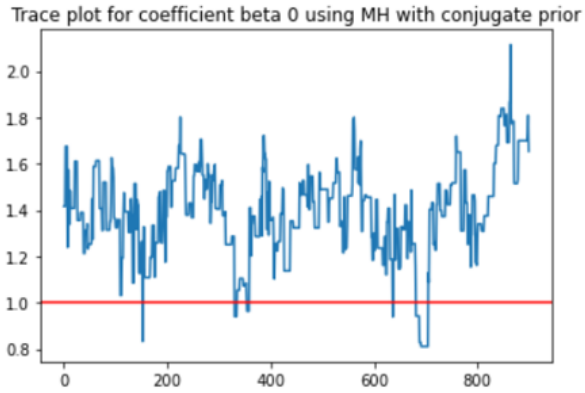

```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.060383673361552814
```



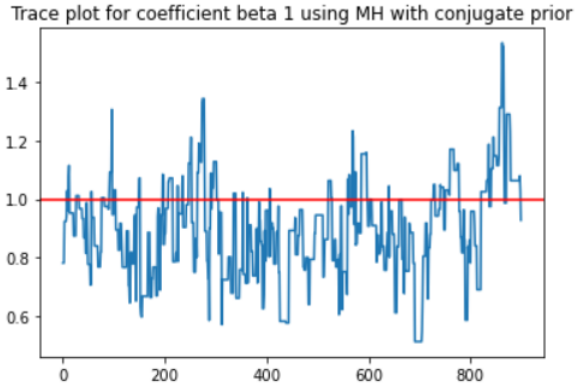
2.2 Conjugate priors

To test if the variant of the algorithm considering conjugate priors worked properly, we included a set of informative normal priors centred at β0 = [10, -10, 10] and with variance equal to the identity matrix. We chose values so divergent from the true parameters to see how much the results would have shifted, and the results were in line with our expectations. Indeed, the first and the last parameters are shifted upwards and the second is shifted downwards.

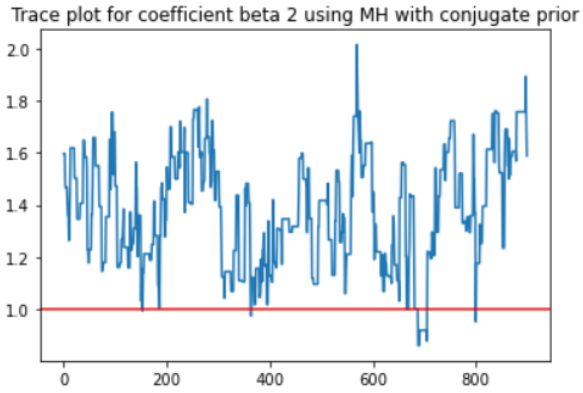
```
Acceptance ratio: 30.7%
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.3926588784122411
```




```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: -0.10428792309638335
```



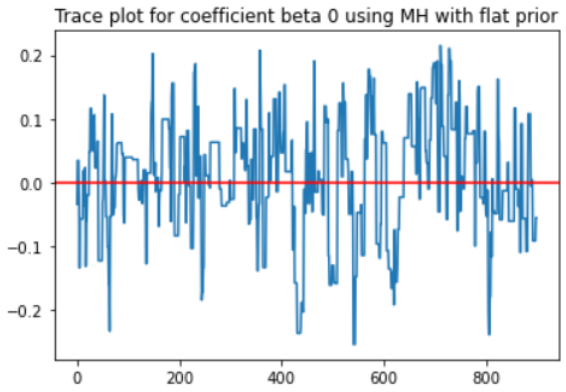

```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.38106172097373925
```



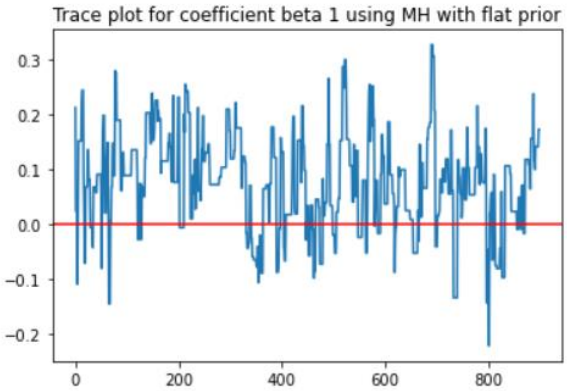
2.3 Trial with different parameters

We ran the algorithm with a different choice of the true parameters. The parameters we chose are β= [0, 0, 0]. We had to set a lower variance parameter to maintain the same acceptance rate, and the results are the same of the first trial.

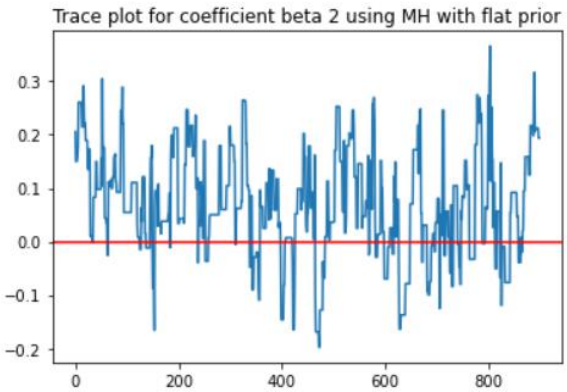
```
Acceptance ratio: 32.3%
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.009327029388852858
```




```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.07725530780241086
```




```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.07162972855739197
```



3 Gibbs Sampling

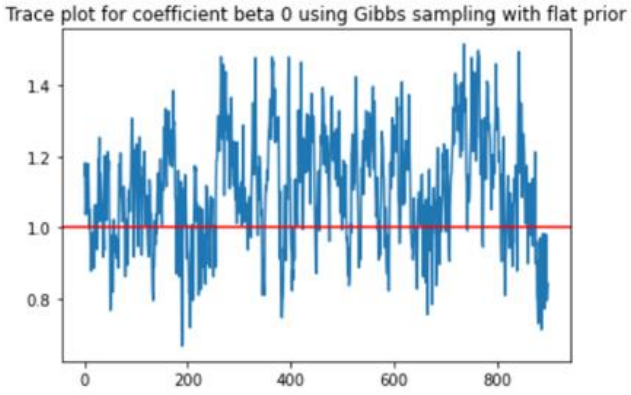
In order to implement the algorithm, we proceeded as follows. First, we derived the full conditionals computing the expected value and the covariance matrix both in the noninformative prior case and in the informative prior one. Then we computed the full conditional of the truncated latent variable Z. After that we ran the algorithm both updating the full conditional of β by conditioning on the new Z and updating Z conditioning on the new β. Also here we considered only the observations after the 100th term, we computed the acceptance ratio and showed the trace plots highlighting the true values.

3.1 Flat priors

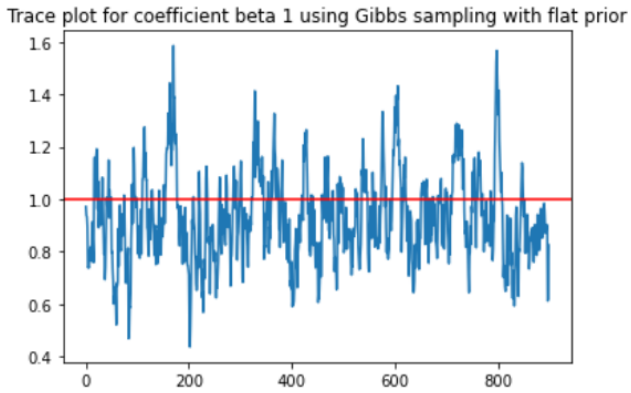
In our first trial of the Gibbs sampling algorithm we kept the priors flat and ran the chain for 1000 iterations and using as starting values the least squares estimates. The results converge to the true values of the parameters and the sample

averages seem to converge with a slight error that vanishes as the number of iterations increase.

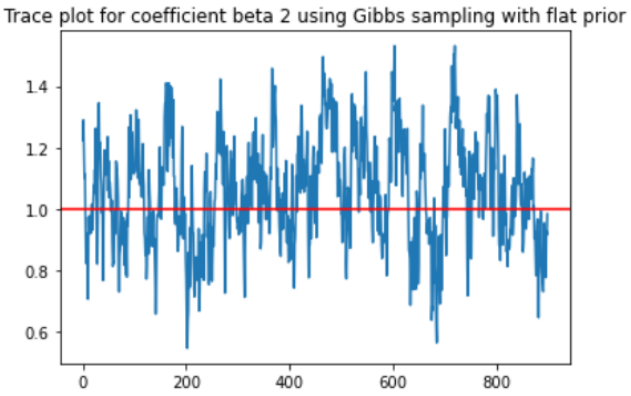
```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.10883056562571602
```




```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: -0.0706160420893891
```



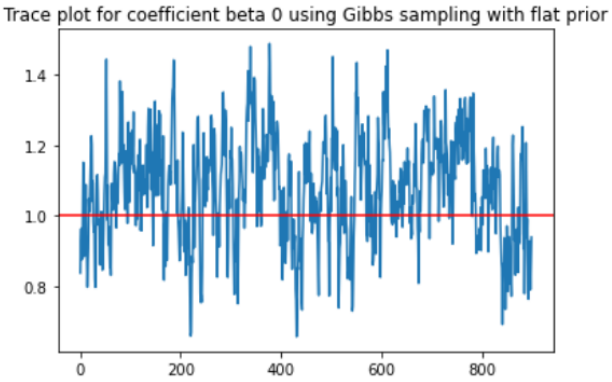

```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.05860722967566456
```



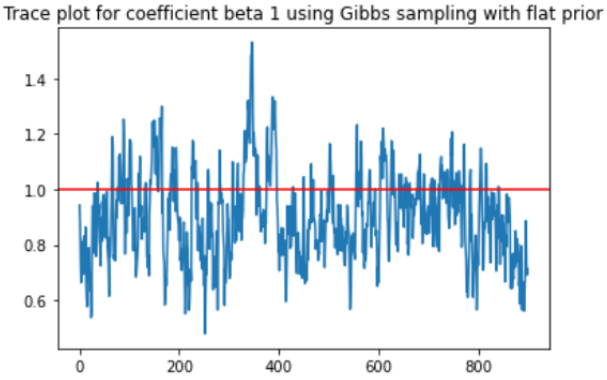
3.2 Flat priors, short chains

In this trial we modified slightly the algorithm and we changed our approach in terms of length of the chain. Instead of running a long one, we introduced a new variable 'nchain' that can be set to any value and makes the chain restart at the initial value after a given number of iterations (in the example below the iterations are 1000 and we split them in 4 chains, where the first *t* / nchain* trials are excluded). The results don't seem to diverge from the long chain version of the algorithm.

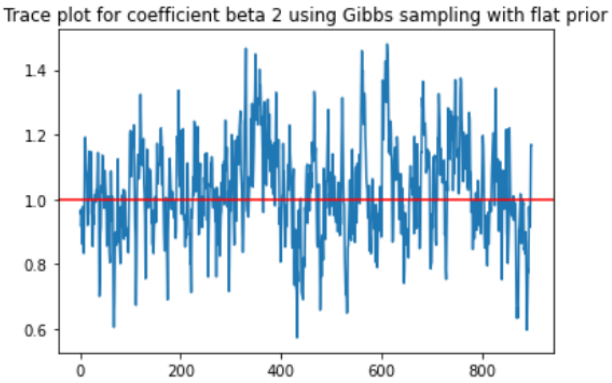
Convergence of averages (sample mean - true value) running 4 chains for 250 iterations each: 0.07906860212602074



Convergence of averages (sample mean - true value) running 4 chains for 250 iterations each: -0.09722431677599908



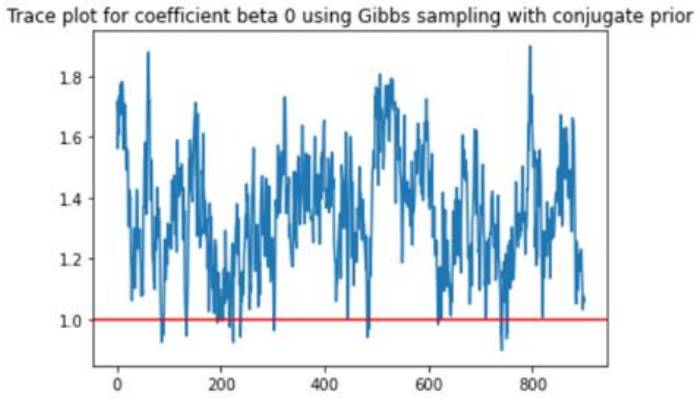
Convergence of averages (sample mean - true value) running 4 chains for 250 iterations each: 0.02841547364601804



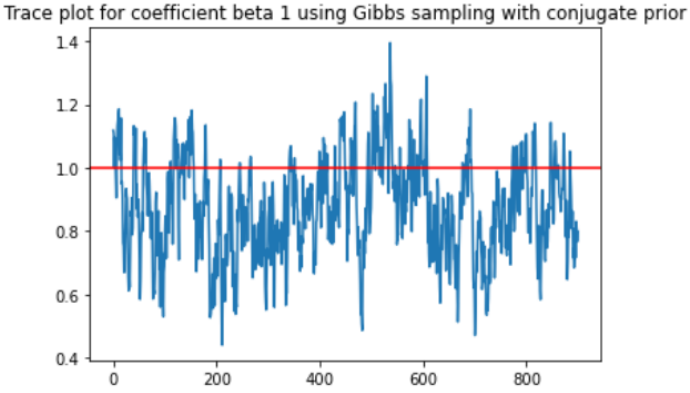
3.3 Conjugate priors

Just like in the Metropolis example, to test the algorithm we included a set of informative normal priors centred at β0 = [10, -10, 10] and with variance equal to the identity matrix. We chose values so divergent from the true parameters to see how much the results would have shifted, and the results were in line with our expectations. Indeed, the first and the last parameters are shifted upwards and the second is shifted downwards.

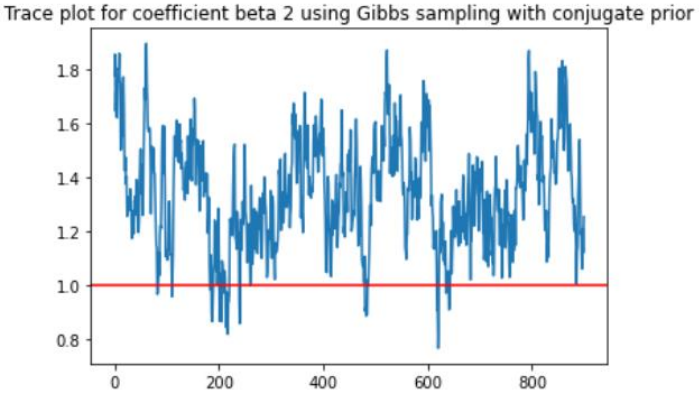
```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.3469409418826708
```




```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: -0.1258325474870401
```



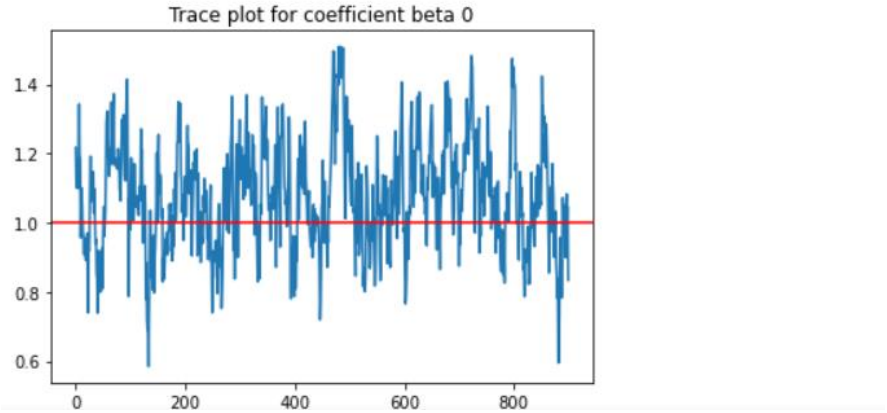

```
Convergence of averages (sample mean - true value) running the chain for 1000 iterations: 0.3414566578835738
```



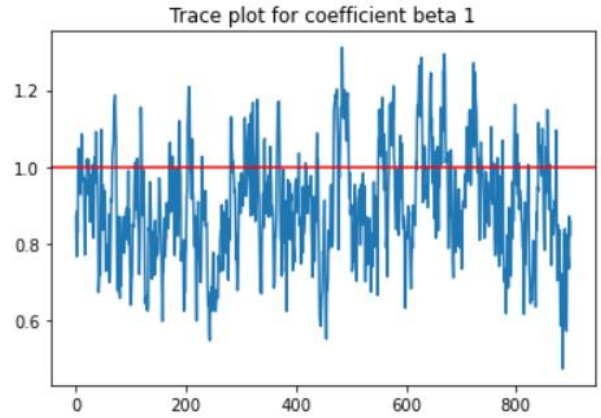
3.4 Trial with different parameters

Setting β= [0, 0, 0] the results are still converging.

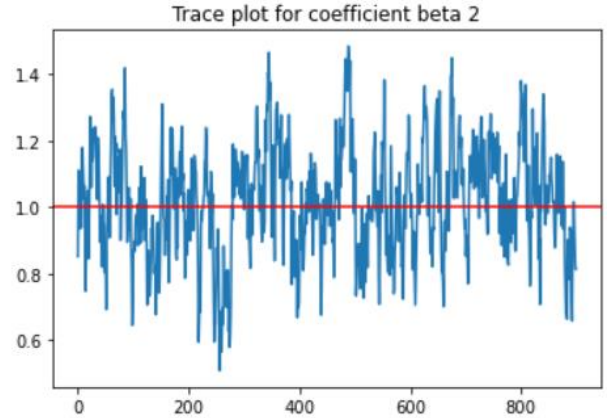
```
Convergence of averages (sample mean - true value): 0.0781212831267104
```




```
Convergence of averages (sample mean - true value): -0.10280160877782485
```




```
Convergence of averages (sample mean - true value): 0.020429877915317363
```



4 Comparison of results of the algorithms

Starting with the same observed data we ran both the algorithms, and the results didn't seem to differ much in terms of convergence to the true parameters. However, we noticed an important difference in terms of computational efficiency. While this might be due specifically to the code we used, we tried to reduce the computational complexity for both to the minimum but the difference is still very noticeable. While the convergence is similar, the Metropolis often gets stuck if we set the optimal acceptance ratio. This leads to several repetitions of the same sample from the posterior of beta, while with the Gibbs sampler the sample looks closer to IID.

5 Derivation of full conditionals

Let's have N latent variables Z_i, where each one is independent and have a normal distribution $N(X_i^T\beta, 1)$ and $Z = X\beta + \epsilon$ where $X = (x_1^T, \dots, x_N^T)$ *and* ϵ
is distributed as $N(0, I)$, where I is the identity matrix.

We have that $Y_i = 1$ if $Z_i > 0$ and $Y_i = 0$ if $Z_1 \leq 0$. Y is deterministic conditional on the sign of the stochastic auxiliary variable Z_i.

The Y_i are independent Bernoulli random variables with $p_i = P(Y_i = 1) = \Phi(X_i^T * \beta)$.

The joint posterior distribution of the latent variable Z_i and of the model parameter β given the data $Y(y_1, \dots, y_N)$ is:

$$\pi(\beta, Z \mid Y) \propto p(\beta)\, p(Z \mid \beta, X)\, p(Y|Z) = \pi(\beta)\, \Pi_{i=1}^N\, p(Z_i \mid \beta, X_i)\, p(Y_i|Z_i)$$

$$p(Z_i|\beta, X_i) = N(Z_i|X_i^T\beta, 1) \;\; and \;\; p(Y_i|Z_i) = 1_{(Y_i=1)}1_{(Z_i>0)} + 1_{(Y_i=0)}1_{(Z_i\leq 0)}.$$

Now let's compute the full conditionals:

$p(\beta|Z, X) \propto \pi(\beta) * \prod_{i=1}^N \; N(Zi|XiT * B, 1)$

that is the posterior density for the normal linear regression model (y for the regression parameter in the normal linear model $Z = X * \beta + \epsilon$).

So using standard normal results: If we have a flat prior for β, $\pi(\beta) = 1$ and $\beta|Y, Z$ is distributed as $N(U, V)$ where
$U = (X^TX)^{-1}X^TZ \; and \; V = (X^TX)^{-1}$.

If we have conjugate prior $\pi(\beta) = N(\beta; \mu_{prior}, var_{prior})$, then $\beta|Y, Z$ is distributed as $N(\hat{\mu}, \hat{V})$ where
$\hat{V} = (var_{prior}^{-1} + X^TX)^{-1} \; and \; \hat{\mu} = \hat{V} \cdot (var_{prior}^{-1}\mu_{prior} + X^T * Z)$.
The full conditional of $Z_i|Y, B$ is a truncated normal distribution distributed as a $N(X_i^T\beta, 1)$ truncated at the left by 0 if $Y = 1$ or truncated at the right by 0 if $Y = 0$.