A Bayesian Analysis of Spotify Data

Nathaniel Maxwell, Jessie Bierschenk

01 May, 2021

Introduction

Music-making is often thought of as an artform—a subjective expression that falls into a specific "genre" according to its musical attributes. Beginning in the 1960s, pop music had been dominated by the verse-chorus form where "the verse sets the scene, the pre-chorus builds tension, and the chorus reaches a climax," with the cycle predictably repeating itself [5]. This musical formula dominated the industry; in fact, "music theorist Jay Summach has found that by the end of the 1960s, 42 percent of hit songs used verse-chorus form. By the end of the 1980s, that figure had doubled to 84 percent" [5]. With the advent of the 21st Century, however, the digitization of music production paired with the introduction of streaming platforms has warped the fundamental structure of songs. On popular media platforms, only snapshots of songs reach the ears of the public: five-second memes, 15-second TikToks, or 30-second ads. The limitless access to songs on streaming platforms has changed the landscape of song-making—"the gist of it: songwriters now get to the good stuff sooner" [2]. This phenomenon exists as increasing accessibility of songs results in decreasing revenue for artists. "Artists are paid per play—provided the listener stays tuned for at least 30 seconds. Each stream earns a tiny fraction of a cent. And just 13% of that goes to the songwriter, says David Israelite of the National Music Publishers Association" [Economist]. In turn, for an artist to make a decent living, their songs need millions of plays. For many musicians, the art of composing/performing/marketing a new song is an arduous process. Even after all the work has been completed and a song is ready to be played to the public, the biggest uncertainty still awaits: How will the song be received? Will it become a hit? Will it be a song that everyone skips over, or never becomes popular? The purpose of this analysis is to investigate which characteristics of a song (such as tempo, duration, mode, acousticness, etc.) would make it more "likeable," less likely to be skipped, or more popular. Of course, music taste is a very subjective matter, and thus, there will be quite a bit of uncertainty around any variables that are deemed important/unimportant. What one person likes; another person may dislike. Therefore, looking at such musical characteristics through a Bayesian lens will help to quantify the uncertainty surrounding any of our findings. Through this analysis we hope to provide some conclusions that an aspiring musician (or even a well-established musician) can have at their disposal when creating new music. These findings beg the question: so what are the features that make a song popular or appealing to a listener? Answers that would be valuable for any musician seeking success in today's music industry.

Pre-Analysis

Data

In our initial data set, we take a sample of 89,393 songs from 186,000 released by Spotify that tracks features of the song and if it was skipped or not. The songs within the data set were not confined to any prerequisite of genre or form; we wanted to see if any variables could impact the popularity of a song. Because this data set was not confined to an individual "taste" nor genre, limitations of this data were that song features may not be determined as significant due to the broadness of the data collected. With this limitation, we also analyzed data that pertained to one specific individual that recorded his tastes for 2,000 songs. For this second dataset, the size was not as big because it presented the opinions of a single individual. George McIntire assembled this data when exploring the explanation behind his varying taste in music. He created two playlists—each with 1,000 songs—one with songs he liked and the other with songs he did not like [4]. In order to minimize bias, he had to make the "BAD" playlist equally as diverse as the "GOOD" playlist by putting in songs of different form rather than an entire album of a single artist he did not like. Limitations of this data can be attributed to the source being a single individual as well as the possibility that "taste" in music can not be fully associated with the variables in the dataset.

Two datasets were utilized during this analysis.

1. The first dataset consists of 83,939 observations on Spotify of whether or not a track was skipped by users. In total, 65,417 different tracks were included in the dataset. Each track has the following characteristics:

   (a) Release Year (Year the song was released)

   (b) Duration (length of song in seconds)

   (c) US Popularity Estimate (A popularity rating of song, on a scale 1-100)

   (d) Acousticness (A confidence measure from 0-1 on whether the track is acoustic, where values near 1 represent high confidence that the track is acoustic)

   (e) Beat Strength (The strength of the beat from 0-1, where 1 represents a very strong sense of beat)

   (f) Bounciness (A rating of the bounciness from 0-1, where 1 represents a strong sense of bounciness)

   (g) Danceability (A rating from 0-1 of how suitable the track is for dancing, where values near 1 represent high suitability)

   (h) Energy (A rating from 0-1 representing a perceptual measure of intensity and activity, where values near 1 represent high energy)

   (i) Instrumentalness (A rating from 0-1 that predicts whether a track has no vocals, where values close to 1 represent high confidence that there are no vocals)

   (j) Mode (Predicts whether or not a song is major or minor)

   (k) Speechiness (A rating from 0-1 that detects the presence of spoken words in a track, with values near 1 representing an exclusively speech-like track)

   (l) Tempo (The estimated tempo of the track in Beats Per Minute (BPM))

   (m) Valence (A rating from 0-1 that represents the positivity of the song, with 1 representing high positivity)

   (n) Skipped (Denotes whether or not that particular track was skipped or played the entire way through)

   Note: in order to try to obtain tracks most representative of new music, only the following tracks were kept:

   (a) Tracks from 2010-present

   (b) Tracks with a speechiness value <= 0.4 (filters out tracks that are mostly spoken, such as podcasts and ebooks)

   (c) Tracks with an instrumentalness value <= 0.6 (filters out tracks that contain no vocals)

   (d) Tracks with a duration <= 360 seconds (given that the average new song is 3-5 minutes, a cutoff of 6 minutes seemed appropriate)

2. The second dataset consisted of 2017 songs compiled by a single person, where a portion of the songs are songs that he likes, and the other portion are songs that he dislikes. This dataset includes similar variables as the first dataset, including:

(a) Acousticness

(b) Danceability

(c) Duration

(d) Energy

(e) Instrumentalness

(f) Key (The particular grouping of chords and notes in a song)

(g) Liveness (rating from 0-1 of whether the track was performed live, with 1 representing high confidence the track was performed live)

(h) Loudness (Overall loudness of the track in decibles (dB))

(i) Mode

(j) Speechiness

(k) Tempo

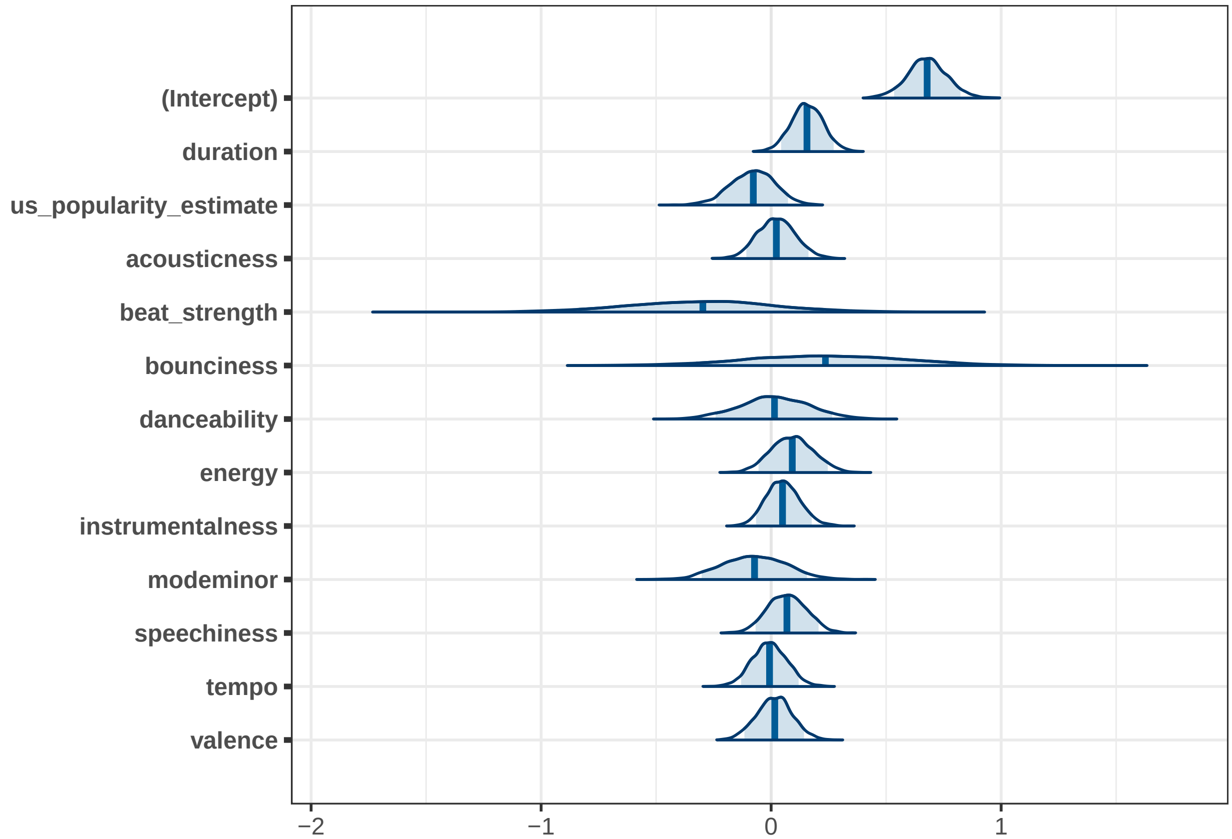
(l) Time Signature (The way in which beats of the song are organized)

(m) Valence

Model Selection

Our response variable, $\mathbf{y}$, will be modeled by a $Bernoulli(\theta)$ distribution, where 1 means the track was skipped. To obtain the variable θ, we will use the logit link, where $logit(\theta) = \eta$, and $\eta = \mathbf{x}^T\boldsymbol{\beta}$, where $\mathbf{x}$ is the covariate space for $\mathbf{Y}$. In other words, we are creating a logistic regression model where θ is obtained from a linear model. For the first dataset, we wanted to estimate the values of the coefficients $\boldsymbol{\beta}$ for each of the variables to find out how they impact whether or not a track is skipped. We are assuming little knowledge about each variable's effect, so we propose a weakly informative prior for $[\boldsymbol{\beta}]$: Using recommendations from Gelman, Jakulin, Pittau, and Su, we use a cauchy(0,2.5) prior for each scaled variable. We centered all the variables and then scaled them so they had the same standard deviation so that no variable could have a disproportionate effect on the outcome.Using the rstanarm package, Rstudio will compute the posterior and draw MCMC samples from the posterior distribution $[\boldsymbol{\beta}|\mathbf{Y}, \mathbf{X}]$.

Posterior Estimates




```
##                             5%   95%
## (Intercept)              0.535 0.823
## duration                 0.043 0.272
## us_popularity_estimate -0.239 0.073
## acousticness            -0.108 0.162
## beat_strength           -0.836 0.227
## bounciness              -0.324 0.804
## danceability            -0.242 0.266
## energy                  -0.055 0.246
## instrumentalness        -0.065 0.177
## modeminor               -0.302 0.156
## speechiness             -0.068 0.206
## tempo                   -0.130 0.120
## valence                 -0.116 0.143
```

```
##
## Computed from 4000 by 1000 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -654.9 10.2
## p_loo        13.2  0.8
## looic      1309.8 20.3
## ------
```

```
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
##
## Computed from 4000 by 1000 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -648.5  9.3
## p_loo        1.0  0.0
## looic     1296.9 18.6
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
##            elpd_diff se_diff
## posterior0  0.0       0.0
## posterior1 -6.5       3.7
```

After running the rstanarm function and including all of the variables, we see that there is only variable whose 90% confidence interval does not include 0. That variable is duration, and furthermore, when calculating the 'leave-one-out' cross-validation information criterion (looic), we see that this model actually has a *higher* value than the looic of a baseline model with no predictors. In other words, our model is worse at predicting whether or not a song is skipped than if someone randomly guessed! Therefore, we will drop all variables that were not deemed significant at a 90% confidence interval (included 0 in their posterior interval), and rerun the model. In this case, 'duration' is the only variable remaining.

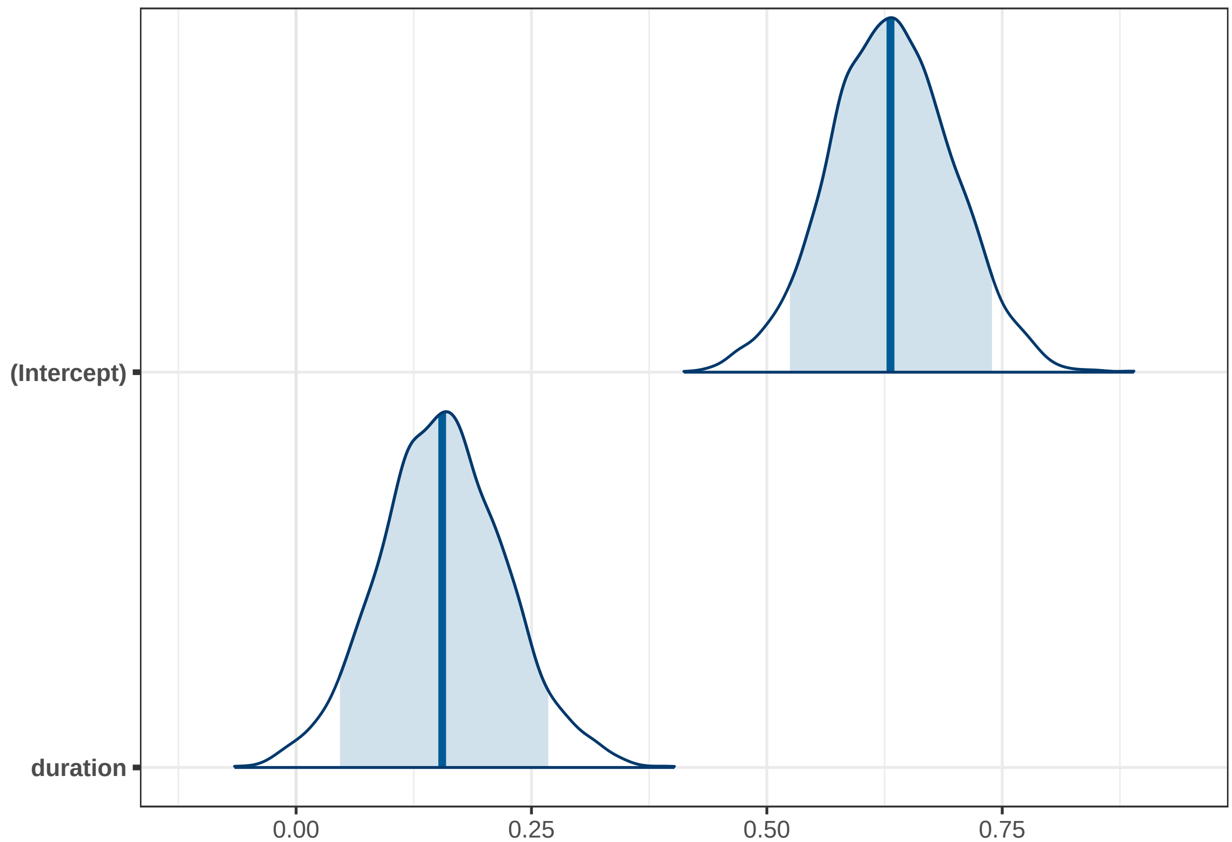
```
posterior2 <- stan_glm(skipped ~ duration, data = Track_features_a,
                 family = binomial(link = "logit"),
                 prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                 seed = seed,
                 refresh = 0)
(loo2 <- loo(posterior2, save_psis = TRUE))
```

```
##
## Computed from 4000 by 1000 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -646.7  9.6
## p_loo        2.0  0.1
## looic     1293.4 19.3
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
rstanarm::loo_compare(loo0, loo2)
```

```
##            elpd_diff se_diff
## posterior2  0.0       0.0
## posterior0 -1.8       2.4
```

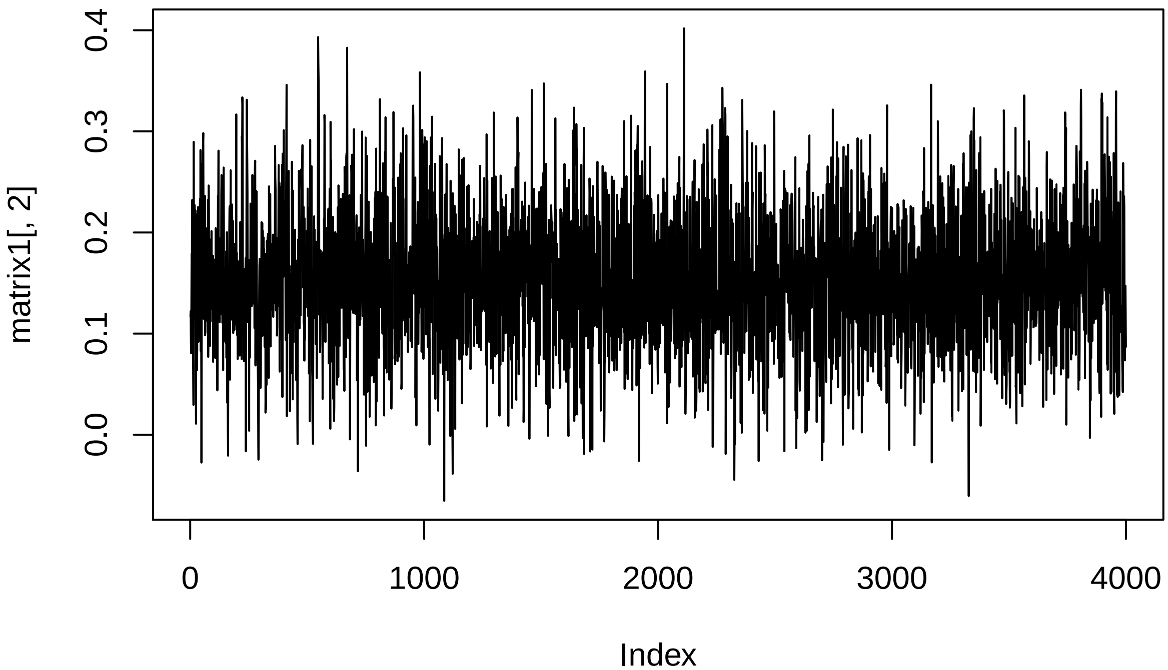
```
mcmc_areas(as.matrix(posterior2), prob = 0.90, prob_outer = 1)
```



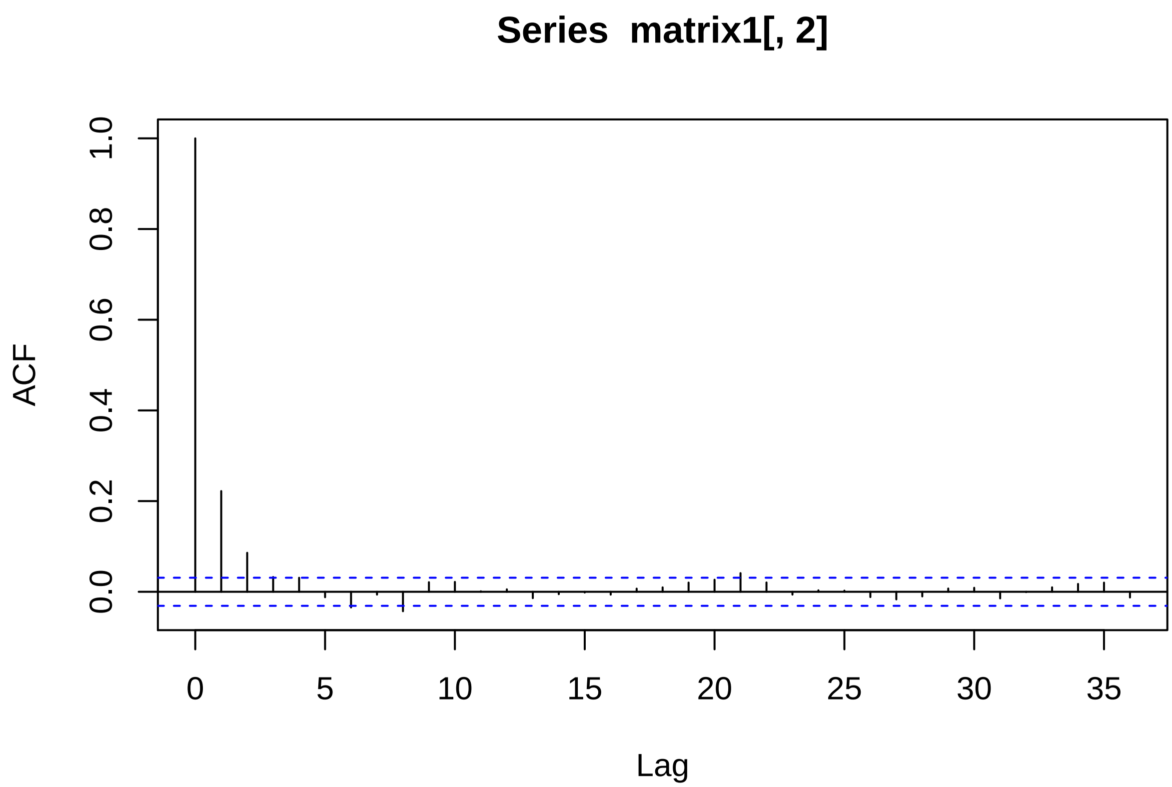

```
round(posterior_interval(posterior2, prob = 0.90), 3)
```

```
##                5%   95%
## (Intercept) 0.524 0.739
## duration    0.047 0.268
```

```
matrix1<-as.matrix(posterior2)
plot(matrix1[,2], type="l")
```




```
acf(matrix1[,2])
```

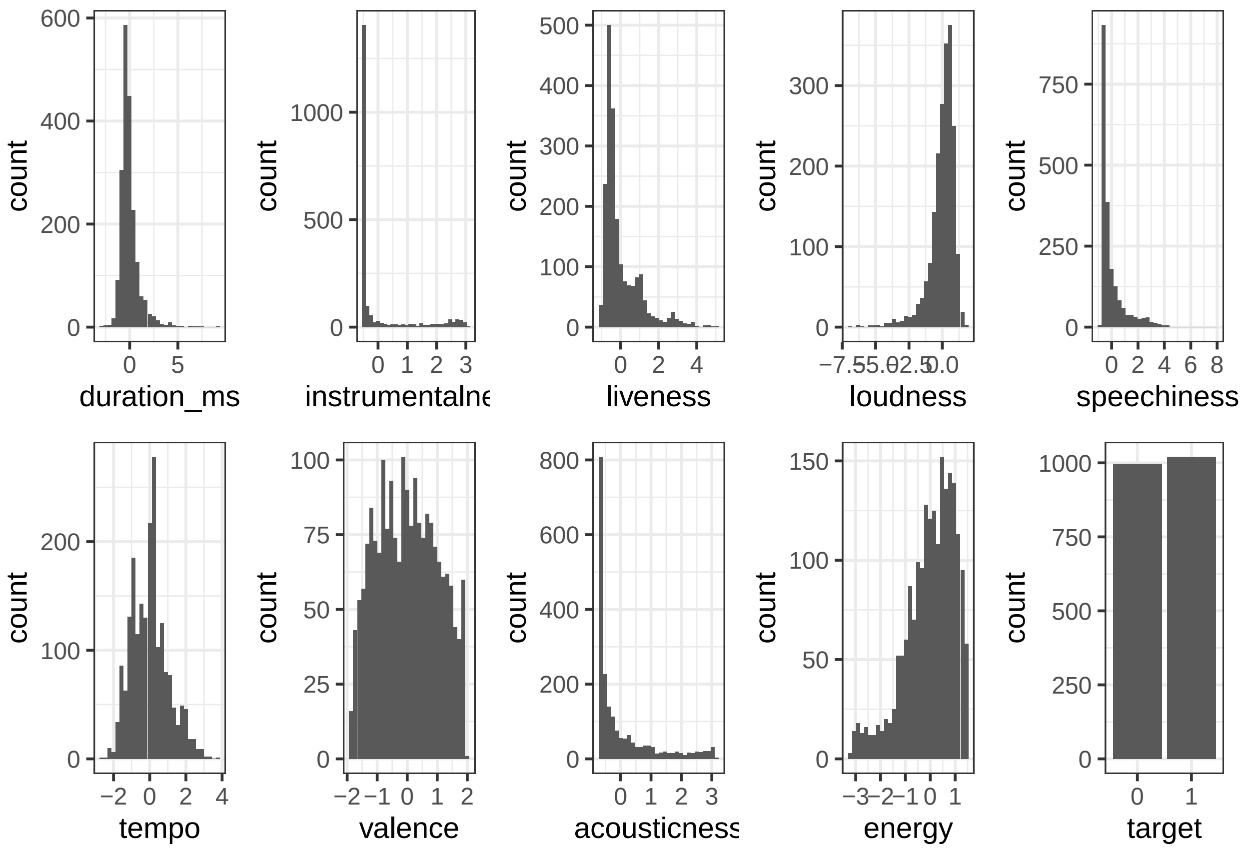


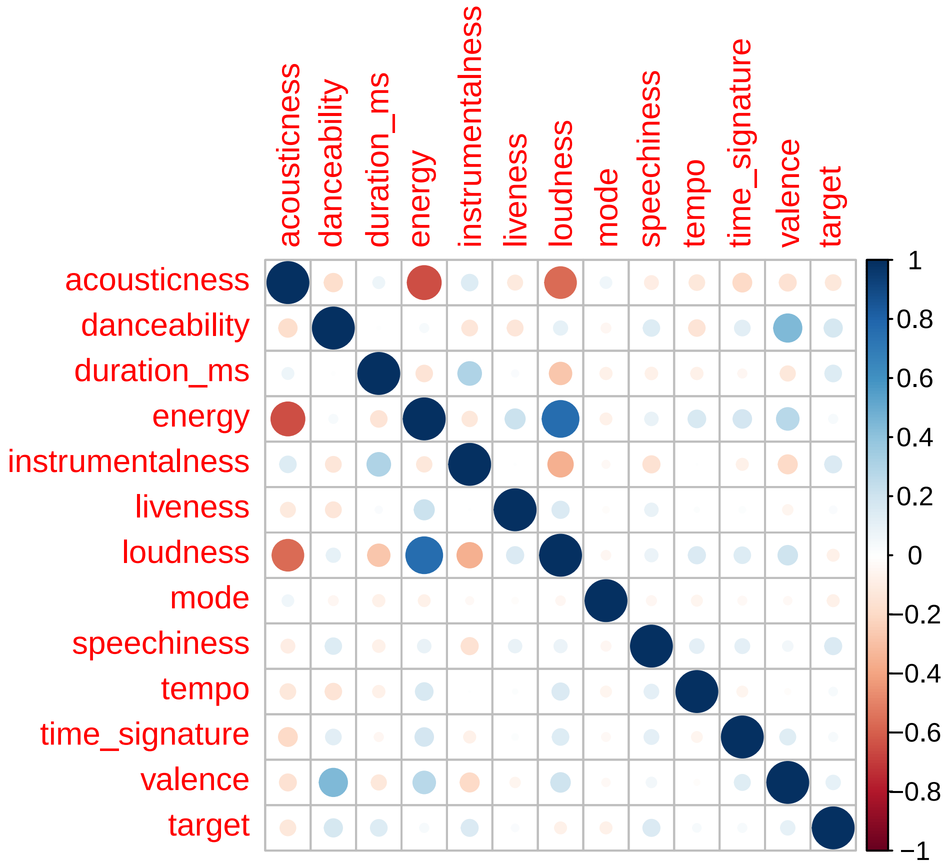
This model proved to be better, but not by much. To calculate our posterior predictive accuracy, we used the following method. If the posterior probability of a track being skipped is greater or equal to 0.5, then we would predict that observation to have a value of 1 (and similarly for less than 0.5). For each observation, we can compare the posterior prediction to the actual observed value. The proportion of times we correctly predict an individual (i.e. [prediction = 0 and observation = 0] or [prediction = 1 and observation = 1]) is our classification accuracy. In our case, the posterior classification accuracy is 0.65. While we would want to also calculate the estimated accuracy on unseen data, since our number of observations is so large, we would expect the value to be the same when using a LOOCV, and this is indeed true: the value is still 0.65.

```
## [1] 0.65
```

```
## [1] 0.65
```

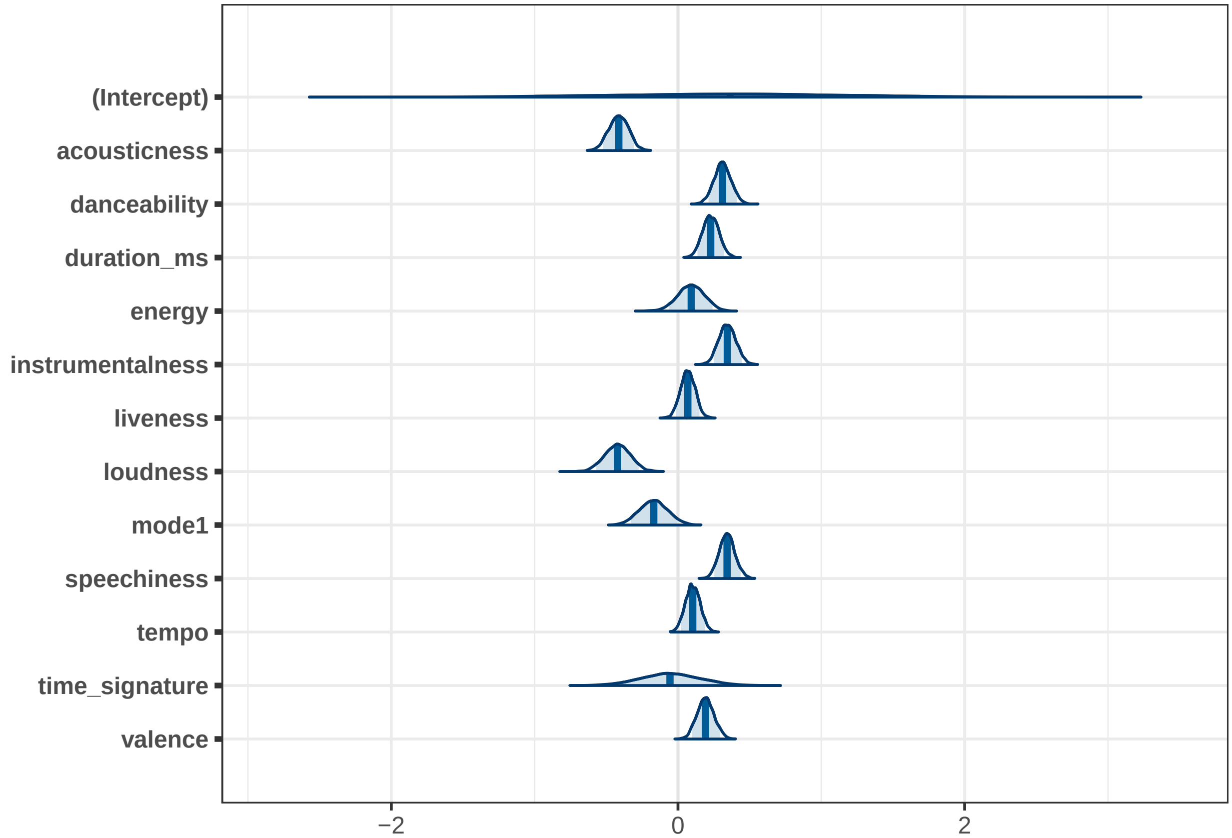
###New Data


count
duration_ms
instrumentalne
liveness
loudness
speechiness
tempo
valence
acousticness
energy
target




```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      target ~ .
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 2017
##  predictors:   15
##
## Estimates:
##                    mean   sd   10%   50%   90%
## (Intercept)      -0.2    1.3 -1.8  -0.2   1.3
## acousticness     -0.4    0.1 -0.5  -0.4  -0.3
## danceability      0.3    0.1  0.2   0.3   0.4
## duration_ms       0.2    0.1  0.2   0.2   0.3
## energy            0.1    0.1  0.0   0.1   0.2
## instrumentalness  0.3    0.1  0.3   0.3   0.4
## liveness          0.1    0.1  0.0   0.1   0.1
## loudness         -0.4    0.1 -0.5  -0.4  -0.3
## mode1            -0.2    0.1 -0.3  -0.2  -0.1
## speechiness       0.3    0.1  0.3   0.3   0.4
## tempo             0.1    0.1  0.0   0.1   0.2
## time_signature3   0.3    1.3 -1.1   0.3   1.9
## time_signature4   0.4    1.3 -1.1   0.3   1.9
```

```
## time_signature5   0.1    1.3 -1.5   0.0   1.7
## valence           0.2    0.1  0.1   0.2   0.3
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.5    0.0  0.5   0.5   0.5
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##                  mcse Rhat n_eff
## (Intercept)      0.0  1.0  1602
## acousticness     0.0  1.0  4283
## danceability     0.0  1.0  3487
## duration_ms      0.0  1.0  4696
## energy           0.0  1.0  2629
## instrumentalness 0.0  1.0  4320
## liveness         0.0  1.0  5343
## loudness         0.0  1.0  3439
## mode1            0.0  1.0  5382
## speechiness      0.0  1.0  4854
## tempo            0.0  1.0  5100
## time_signature3  0.0  1.0  1621
## time_signature4  0.0  1.0  1603
## time_signature5  0.0  1.0  1601
## valence          0.0  1.0  3983
## mean_PPD         0.0  1.0  4699
## log-posterior    0.1  1.0  1813
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```




```
##      (Intercept)     acousticness     danceability      duration_ms
##            0.363           -0.413            0.311            0.228
##           energy instrumentalness         liveness         loudness
##            0.092            0.344            0.069           -0.422
##            mode1      speechiness            tempo   time_signature
##           -0.169            0.343            0.103           -0.056
##          valence
##            0.192
```

```
##                      5%    95%
## (Intercept)      -0.997  1.687
## acousticness     -0.524 -0.305
## danceability      0.213  0.411
## duration_ms       0.136  0.323
## energy           -0.064  0.242
## instrumentalness  0.249  0.444
## liveness         -0.016  0.150
## loudness         -0.572 -0.274
## mode1            -0.331 -0.005
## speechiness       0.254  0.438
## tempo             0.019  0.188
## time_signature   -0.391  0.292
## valence           0.096  0.294
```

```r
(loo3 <- loo(posterior3, save_psis = TRUE))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1268.5 16.3
## p_loo        15.2  0.5
## looic      2537.0 32.6
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     2016  100.0%  2653
##  (0.5, 0.7]   (ok)          1    0.0%  1430
##    (0.7, 1]   (bad)         0    0.0%  <NA>
##    (1, Inf)   (very bad)    0    0.0%  <NA>
##
## All Pareto k estimates are ok (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

#Model Selection

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo  -1398.9 0.5
## p_loo         1.0 0.0
## looic      2797.9 1.0
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
##            elpd_diff se_diff
## posterior3    0.0       0.0
## posterior4 -130.4      16.3
```

```r
posterior4.1 <- stan_glm(target ~ danceability+ duration_ms+ energy+ instrumentalness+ liveness+ loudne
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.2 <- stan_glm(target ~ acousticness+ duration_ms+ energy+ instrumentalness+ liveness+ loudne
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.3 <- stan_glm(target ~ acousticness+ danceability+ energy+ instrumentalness+ liveness+ loudn
```

```
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.4 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ instrumentalness+ liveness+ l
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.5 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ liveness+ loudness+ m
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.6 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ instrumentalness+ lou
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.7 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ instrumentalness+ liv
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.8 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ instrumentalness+ liv
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.9 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ instrumentalness+ liv
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.10 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ instrumentalness+ li
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.11 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ instrumentalness+ li
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior4.full <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ energy+ instrumentalness+
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
```

```
(loo4.1 <- loo(posterior4.1, save_psis = T))
```

```
##
```

```
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1284.2 15.4
## p_loo        11.9  0.4
## looic      2568.4 30.8
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.2 <- loo(posterior4.2, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1279.6 15.5
## p_loo        12.1  0.4
## looic      2559.1 30.9
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.3 <- loo(posterior4.3, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1274.2 15.8
## p_loo        12.0  0.4
## looic      2548.5 31.6
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.4 <- loo(posterior4.4, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1266.1 16.2
## p_loo        12.0  0.4
## looic      2532.2 32.4
## ------
```

```
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.5 <- loo(posterior4.5, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1283.4 15.5
## p_loo        11.9  0.4
## looic      2566.9 31.0
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.6 <- loo(posterior4.6, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1266.3 16.2
## p_loo        11.8  0.4
## looic      2532.6 32.3
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.7 <- loo(posterior4.7, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1277.5 15.5
## p_loo        11.7  0.4
## looic      2555.0 31.0
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.8 <- loo(posterior4.8, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1266.9 16.1
## p_loo        12.0  0.4
## looic      2533.9 32.2
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.9 <- loo(posterior4.9, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1287.7 14.9
## p_loo        11.8  0.4
## looic      2575.4 29.9
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.10 <- loo(posterior4.10, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1267.7 16.1
## p_loo        11.9  0.4
## looic      2535.3 32.3
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo4.11 <- loo(posterior4.11, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
```

```
## elpd_loo  -1271.1 16.2
## p_loo        12.2  0.4
## looic      2542.2 32.3
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
(loo4.full <- loo(posterior4.full, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1266.7 16.3
## p_loo        13.1  0.4
## looic      2533.4 32.5
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
rstanarm::loo_compare(loo4.1, loo4.2, loo4.3, loo4.4, loo4.5, loo4.6, loo4.7, loo4.8, loo4.9, loo4.10, l
```

```
##                 elpd_diff se_diff
## posterior4.4      0.0       0.0
## posterior4.6     -0.2       1.8
## posterior4.full  -0.6       1.0
## posterior4.8     -0.8       2.0
## posterior4.10    -1.5       2.3
## posterior4.11    -5.0       3.7
## posterior4.3     -8.1       4.5
## posterior4.7    -11.4       4.5
## posterior4.2    -13.4       5.6
## posterior4.5    -17.3       6.1
## posterior4.1    -18.1       6.9
## posterior4.9    -21.6       6.9
```

```
posterior5.1 <- stan_glm(target ~ danceability+ duration_ms+ instrumentalness+ liveness+ loudness+ mode
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.2 <- stan_glm(target ~ acousticness+ duration_ms+  instrumentalness+ liveness+ loudness+ mod
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.3 <- stan_glm(target ~ acousticness+ danceability+  instrumentalness+ liveness+ loudness+ mo
                family = binomial(link = "logit"),
```

```
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)

posterior5.4 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ liveness+ loudness+ mode+ spe
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.5 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ instrumentalness+ loudness+ m
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.6 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+  instrumentalness+ liveness+m
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.7 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+  instrumentalness+ liveness+
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.8 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+  instrumentalness+ liveness+
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.9 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+  instrumentalness+ liveness+
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.10 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+  instrumentalness+ liveness
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
posterior5.full <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ instrumentalness+ livenes
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)
```

```
(loo5.1 <- loo(posterior5.1, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1290.5 14.9
## p_loo        10.8  0.4
```

```
## looic      2581.1 29.9
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.2 <- loo(posterior5.2, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1278.4 15.5
## p_loo        10.9  0.4
## looic      2556.9 30.9
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.3 <- loo(posterior5.3, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1273.6 15.7
## p_loo        10.6  0.3
## looic      2547.2 31.4
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.4 <- loo(posterior5.4, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1285.8 15.3
## p_loo        10.9  0.4
## looic      2571.7 30.6
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.5 <- loo(posterior5.5, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1266.2 16.1
## p_loo        11.0  0.4
## looic      2532.4 32.2
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.6 <- loo(posterior5.6, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1280.2 15.3
## p_loo        10.5  0.4
## looic      2560.3 30.6
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.7 <- loo(posterior5.7, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1266.4 16.0
## p_loo        10.9  0.4
## looic      2532.9 32.1
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.8 <- loo(posterior5.8, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
```

```
## elpd_loo  -1287.6 14.8
## p_loo        10.6  0.3
## looic      2575.2 29.6
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.9 <- loo(posterior5.9, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1267.3 16.1
## p_loo        11.0  0.4
## looic      2534.6 32.1
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.10 <- loo(posterior5.10, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1272.0 16.1
## p_loo        10.7  0.4
## looic      2544.1 32.1
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
(loo5.full <- loo(posterior5.full, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1266.1 16.2
## p_loo        12.0  0.4
## looic      2532.2 32.4
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
rstanarm::loo_compare(loo5.1, loo5.2, loo5.3, loo5.4, loo5.5, loo5.6, loo5.7, loo5.8, loo5.9, loo5.10, l
```

```
##                  elpd_diff se_diff
## posterior5.full    0.0       0.0
## posterior5.5      -0.1       1.6
## posterior5.7      -0.3       1.7
## posterior5.9      -1.2       2.2
## posterior5.10     -5.9       3.8
## posterior5.3      -7.5       4.3
## posterior5.2     -12.3       5.6
## posterior5.6     -14.0       5.7
## posterior5.4     -19.7       6.2
## posterior5.8     -21.5       6.8
## posterior5.1     -24.4       7.5
```

```r
posterior5.interaction <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ instrumentalness+ l
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)

(loo5.interaction <- loo(posterior5.interaction, save_psis = T))
```

```
##
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1238.2 17.4
## p_loo        15.7  0.9
## looic      2476.3 34.9
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```r
rstanarm::loo_compare(loo5.interaction, loo5.full)
```

```
##                         elpd_diff se_diff
## posterior5.interaction    0.0       0.0
## posterior5.full         -27.9       8.3
```

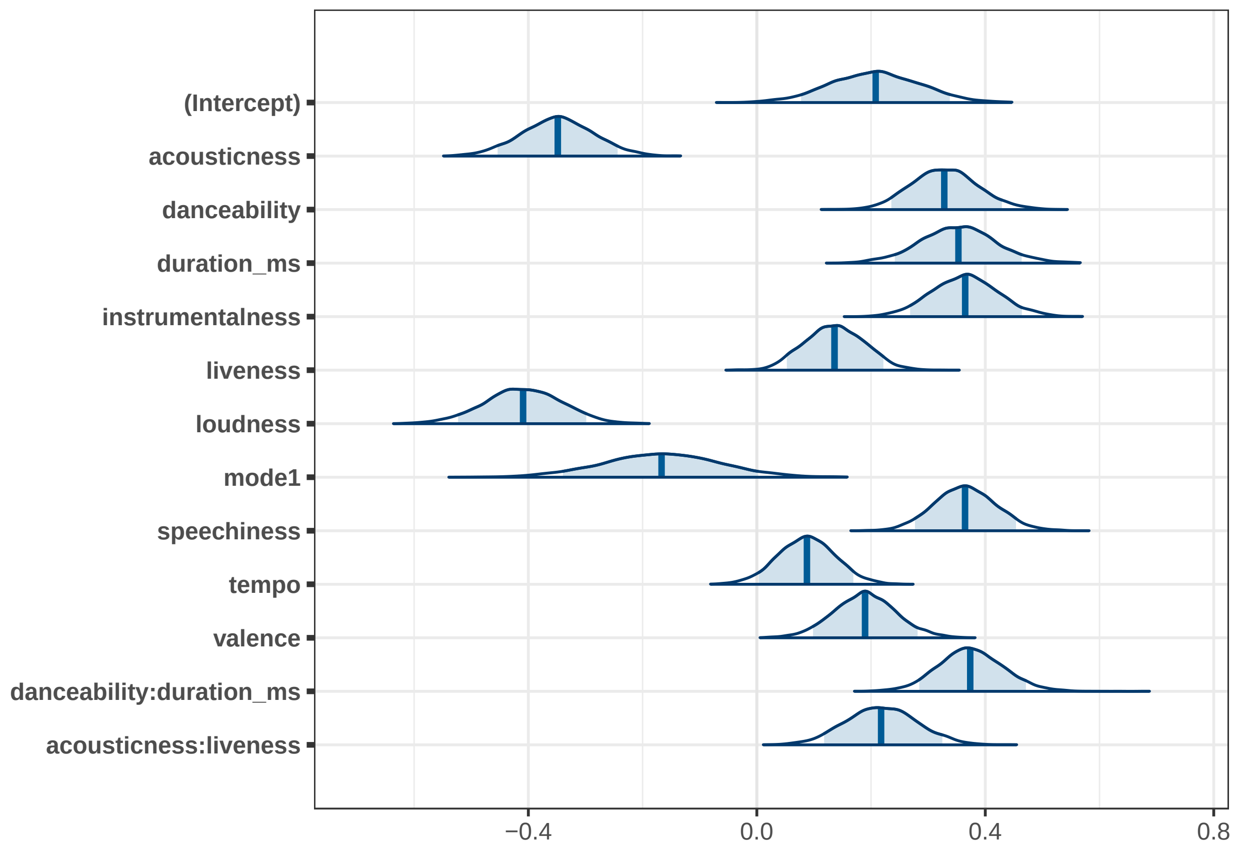
```r
posterior5.interaction2 <- stan_glm(target ~ acousticness+ danceability+ duration_ms+ instrumentalness+
                family = binomial(link = "logit"),
                prior = cauchy(0,2.5), prior_intercept = cauchy(0,2.5),
                seed = seed,
                refresh = 0)

(loo5.interaction2 <- loo(posterior5.interaction2, save_psis = T))
```

```
##
```

```
## Computed from 4000 by 2017 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -1237.3 17.4
## p_loo        14.8  0.9
## looic      2474.6 34.9
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
rstanarm::loo_compare(loo5.interaction, loo5.interaction2)
```

```
##                         elpd_diff se_diff
## posterior5.interaction2  0.0       0.0
## posterior5.interaction  -0.9       0.1
```




```
##              (Intercept)             acousticness             danceability
##                    0.208                   -0.348                    0.328
##              duration_ms         instrumentalness                 liveness
##                    0.353                    0.365                    0.136
##                 loudness                    mode1              speechiness
##                   -0.409                   -0.167                    0.364
```

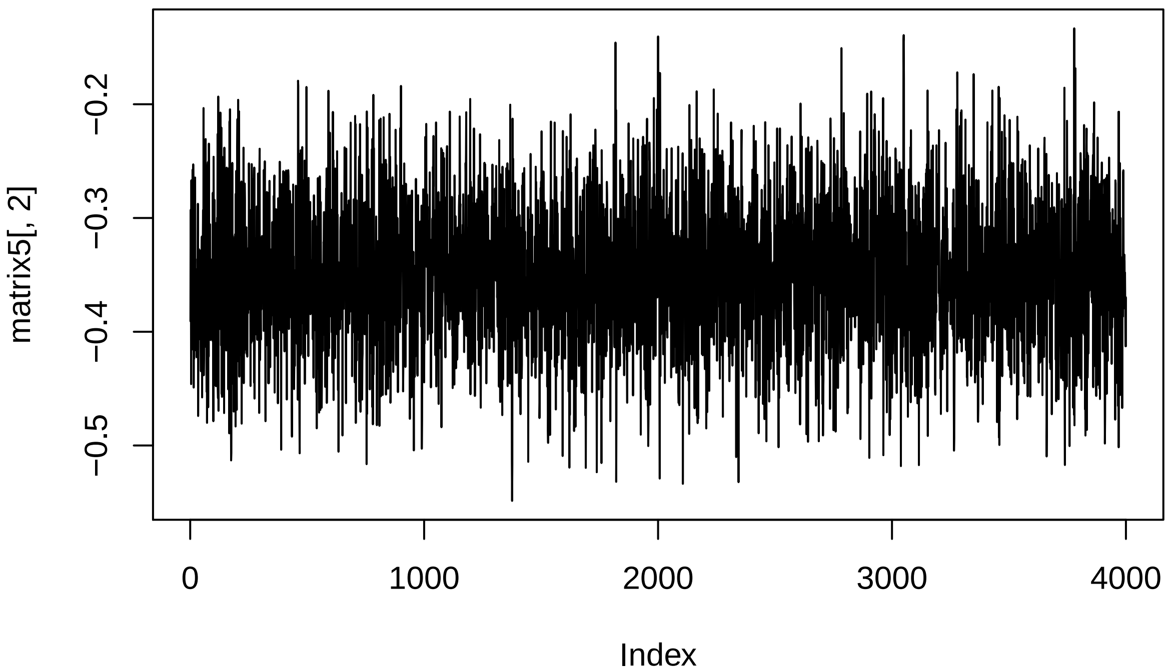
```
##                   tempo                  valence danceability:duration_ms
##                   0.088                    0.189                    0.374
##    acousticness:liveness
##                   0.218
```

```
##                              5%    95%
## (Intercept)               0.078  0.338
## acousticness             -0.454 -0.244
## danceability              0.236  0.429
## duration_ms               0.242  0.464
## instrumentalness          0.268  0.464
## liveness                  0.053  0.222
## loudness                 -0.523 -0.299
## mode1                    -0.340  0.002
## speechiness               0.277  0.454
## tempo                     0.003  0.169
## valence                   0.098  0.282
## danceability:duration_ms  0.284  0.471
## acousticness:liveness     0.118  0.325
```

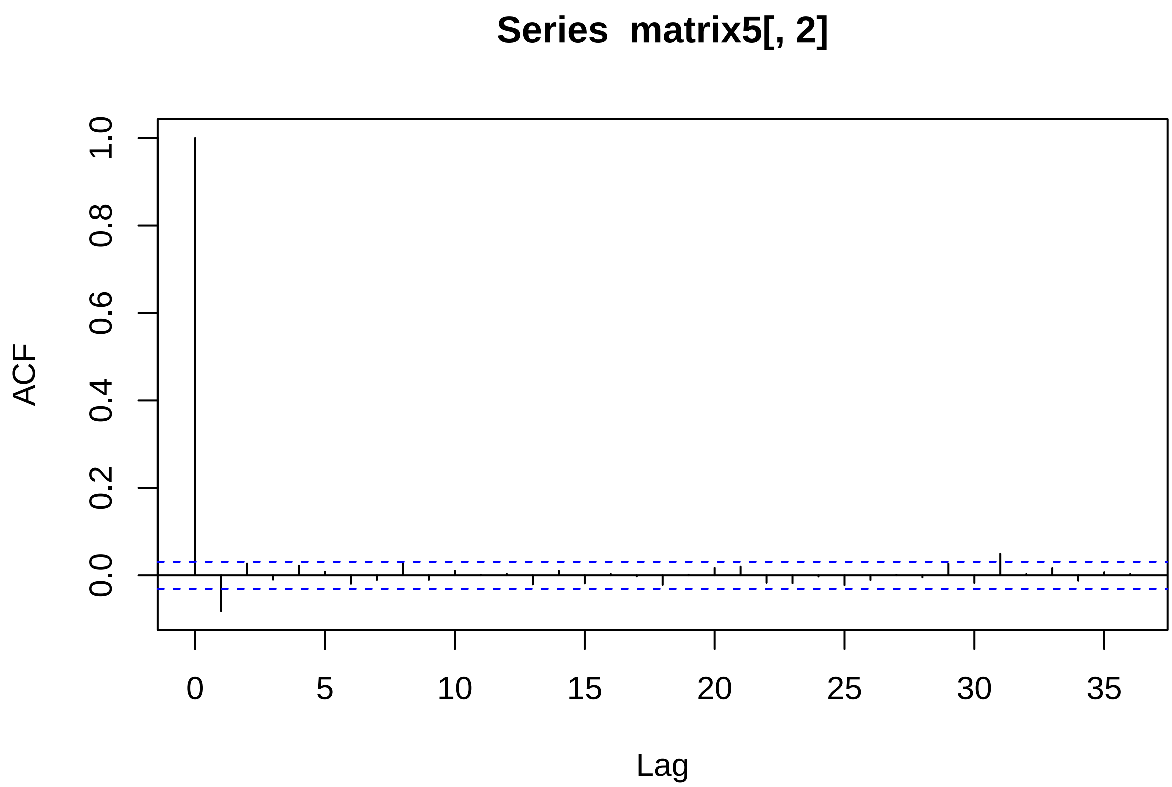
To calculate our posterior predictive accuracy, we used the following method. If the posterior probability of the song being liked for an particular song is greater or equal to 0.5, then we would predict that that song to have a value of 1 (and similarly for less than 0.5). For each observation, we can compare the posterior prediction to the actual observed value. The proportion of times we correctly predict an individual (i.e. [prediction = 0 and observation = 0] or [prediction = 1 and observation = 1]) is our classification accuracy.

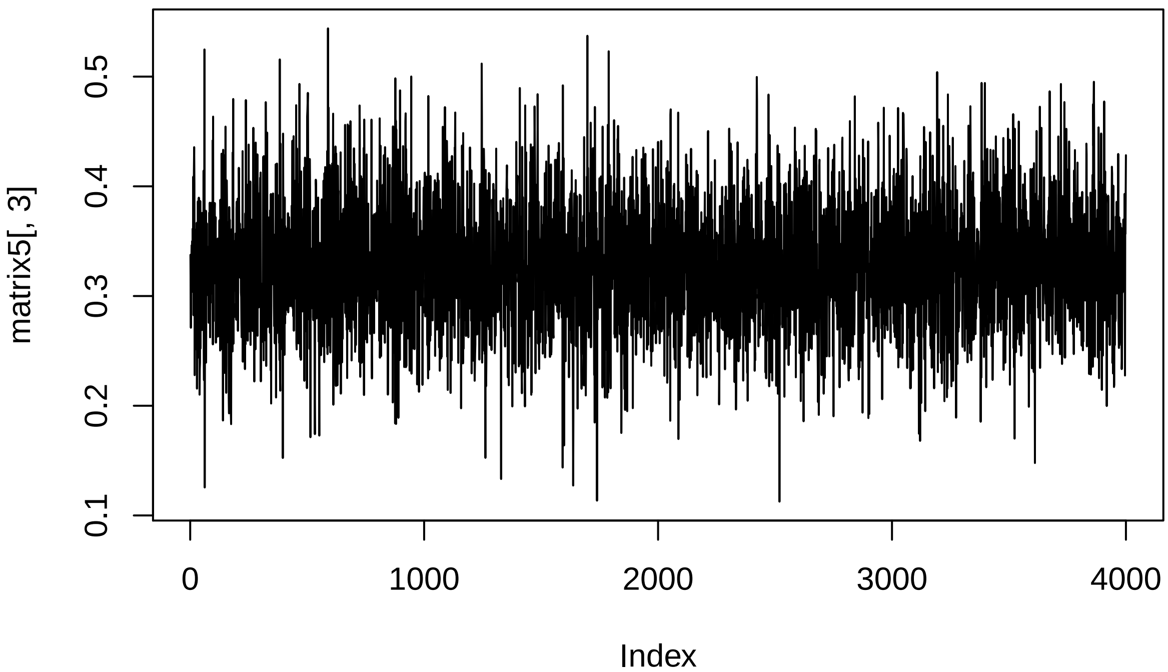
```
## [1] 0.688
```

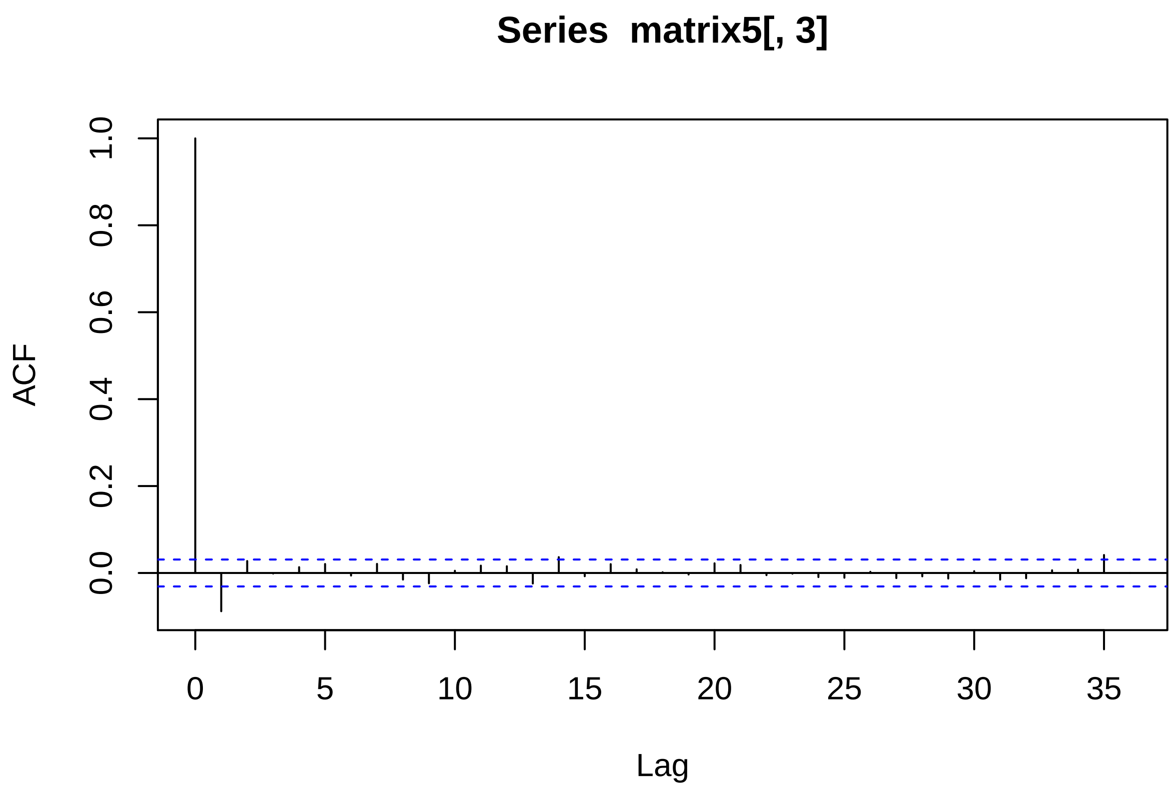
```
## [1] 0.683
```

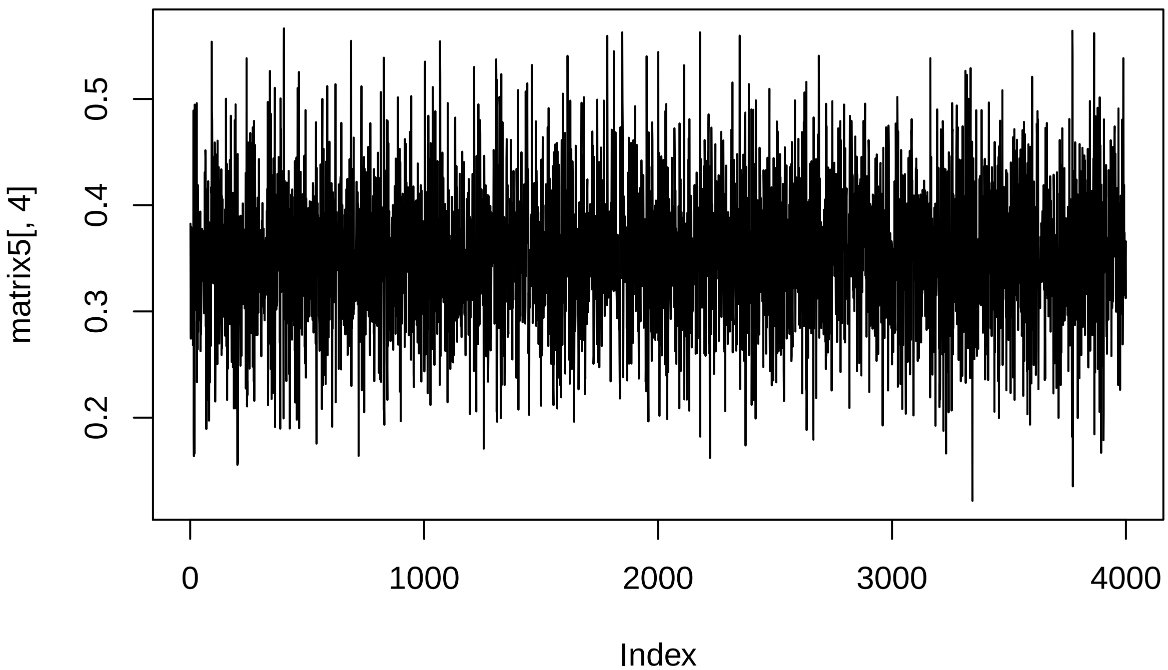


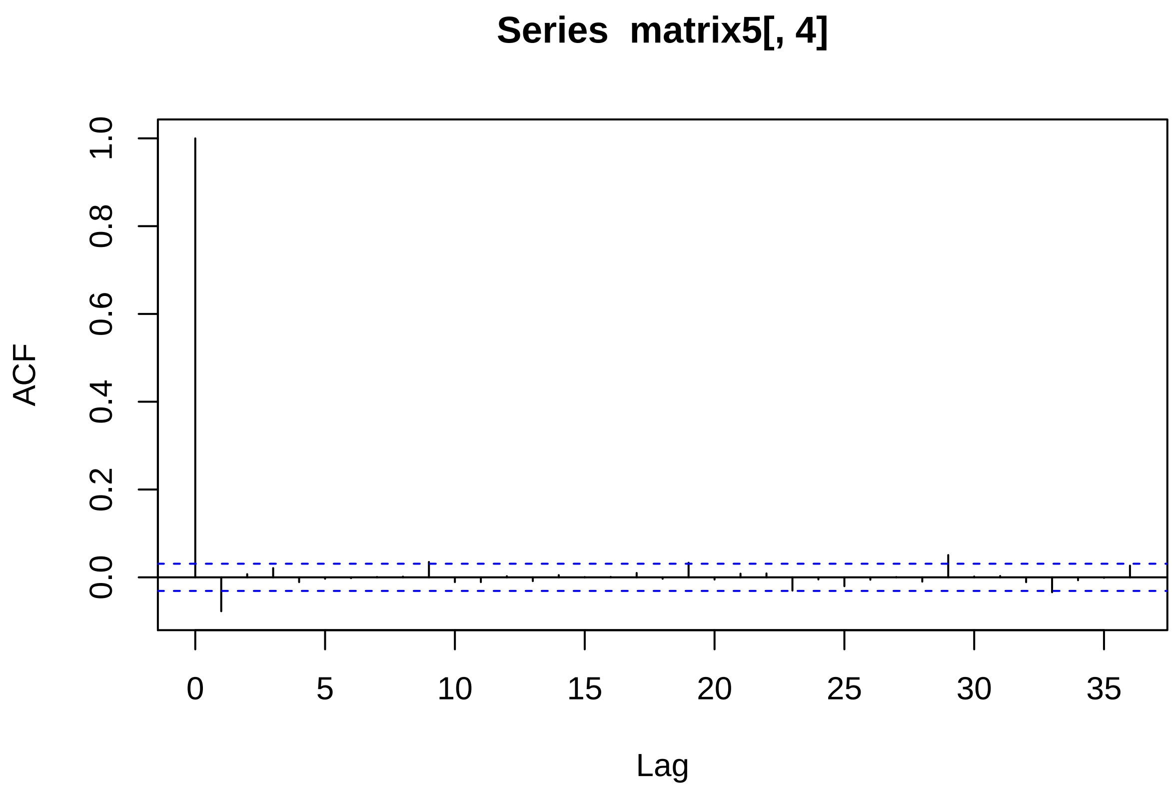
−0.2
−0.3
−0.4
−0.5
matrix5[, 2]
0
1000
2000
3000
4000
Index

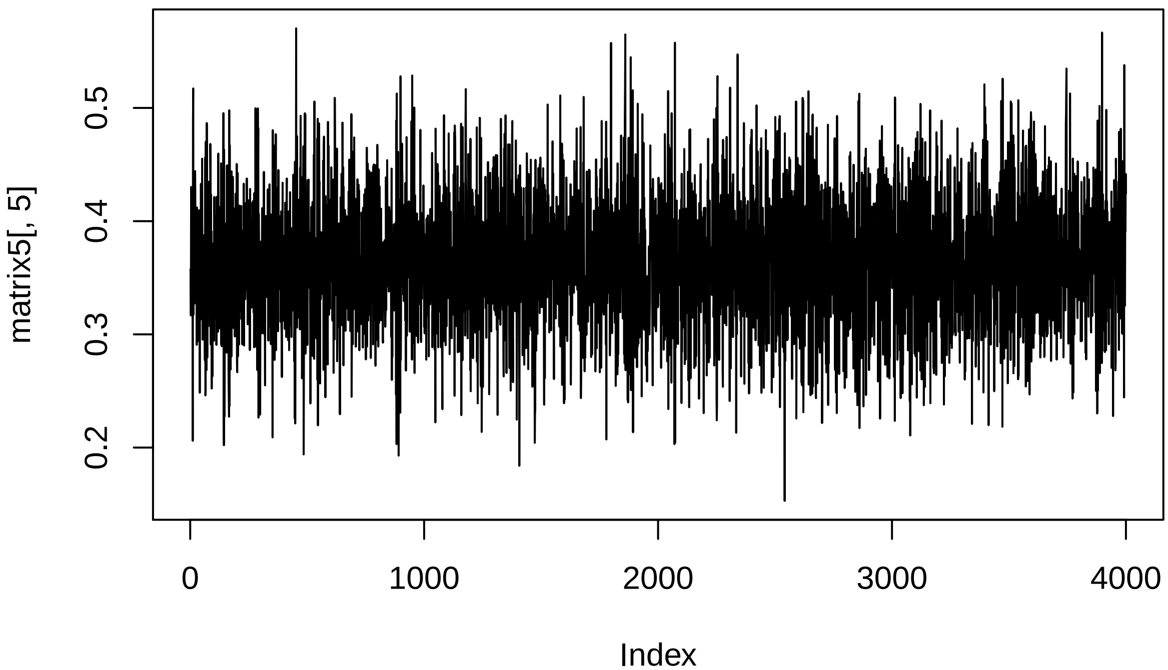

Series matrix5[, 2]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag

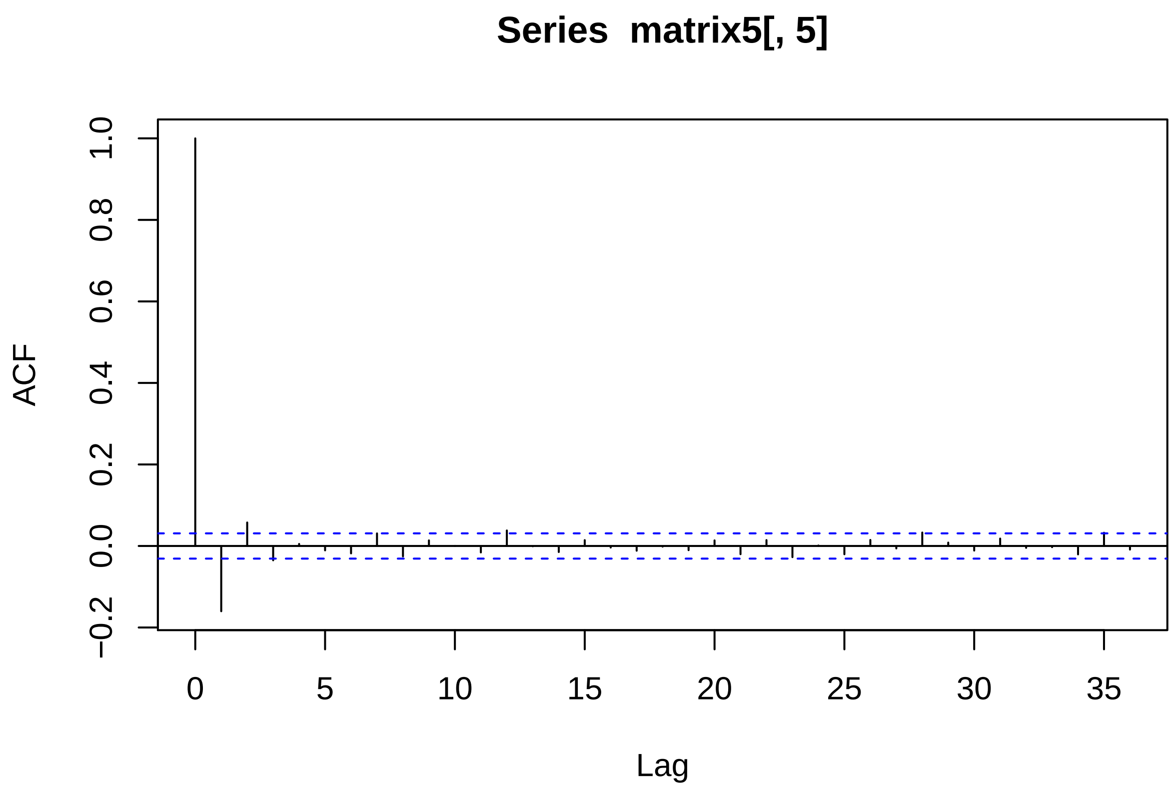

matrix5[, 3]
0.1
0.2
0.3
0.4
0.5
0
1000
2000
3000
4000
Index

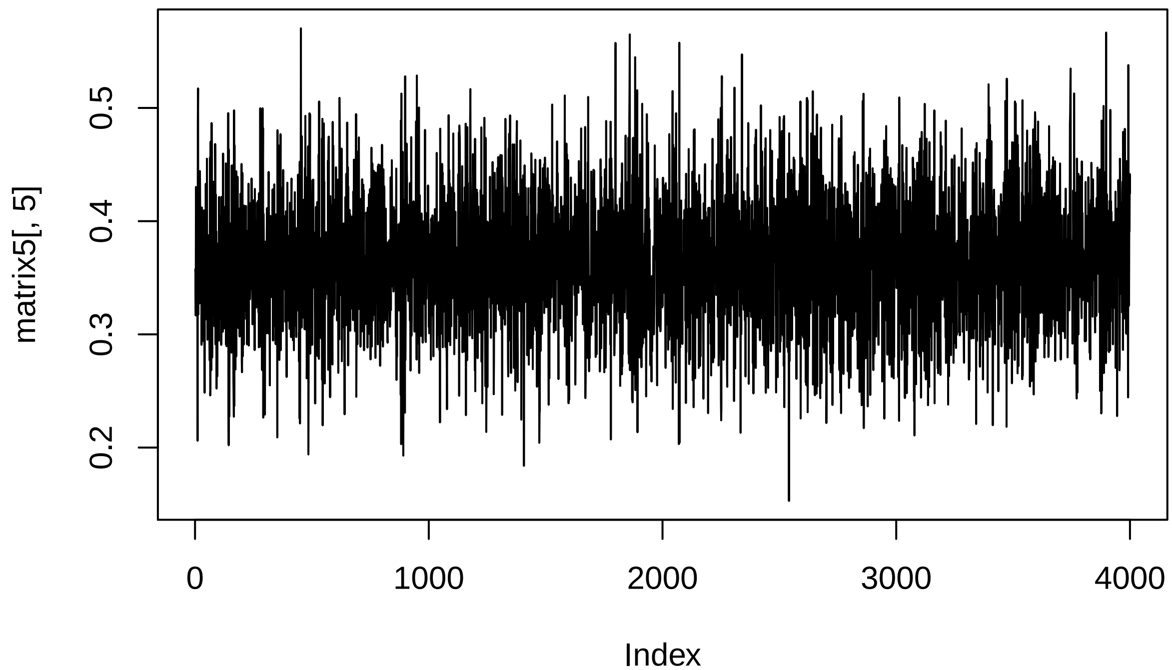

Series matrix5[, 3]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag

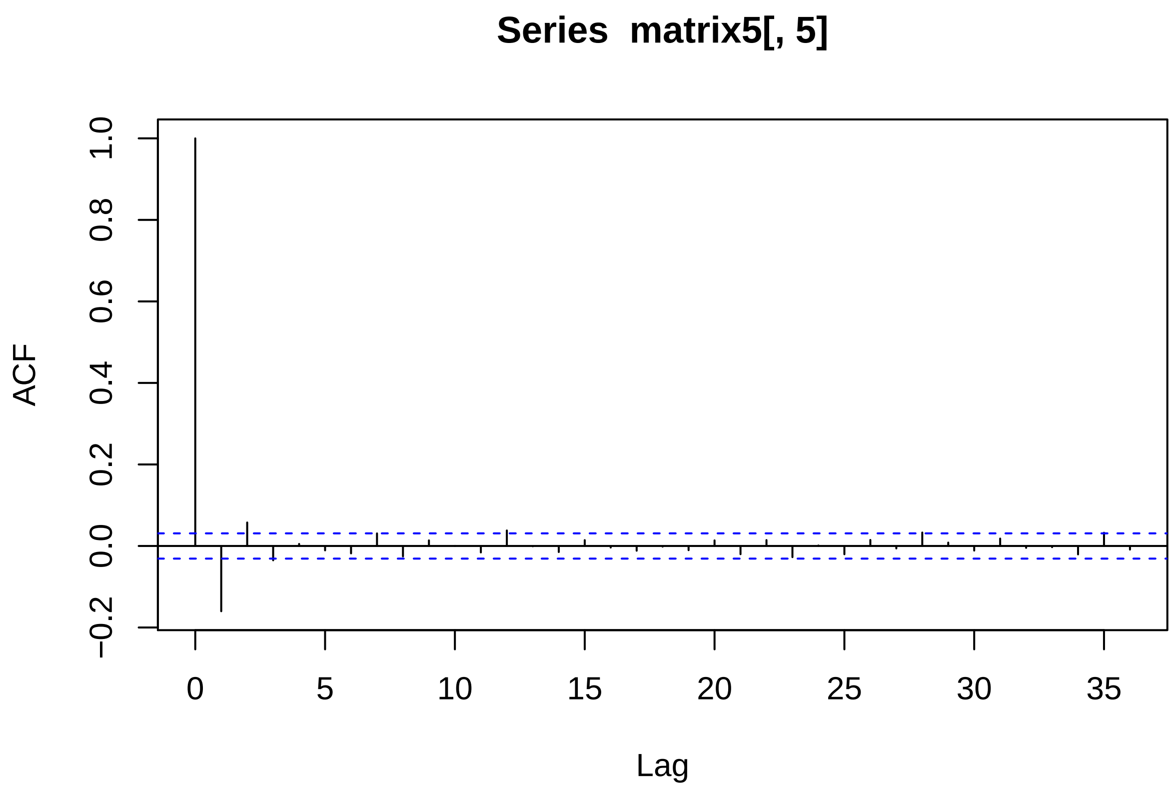

matrix5[, 4]
0.2
0.3
0.4
0.5
0
1000
2000
3000
4000
Index

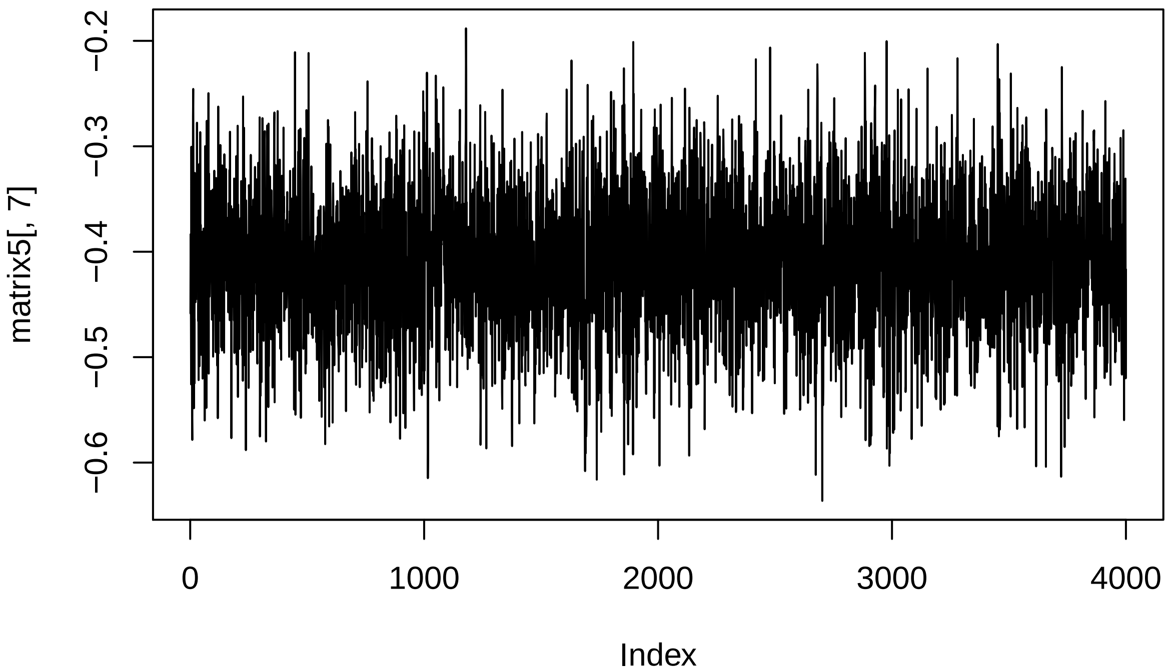

Series matrix5[, 4]
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
Lag


matrix5[, 5]
0.2
0.3
0.4
0.5
0
1000
2000
3000
4000
Index

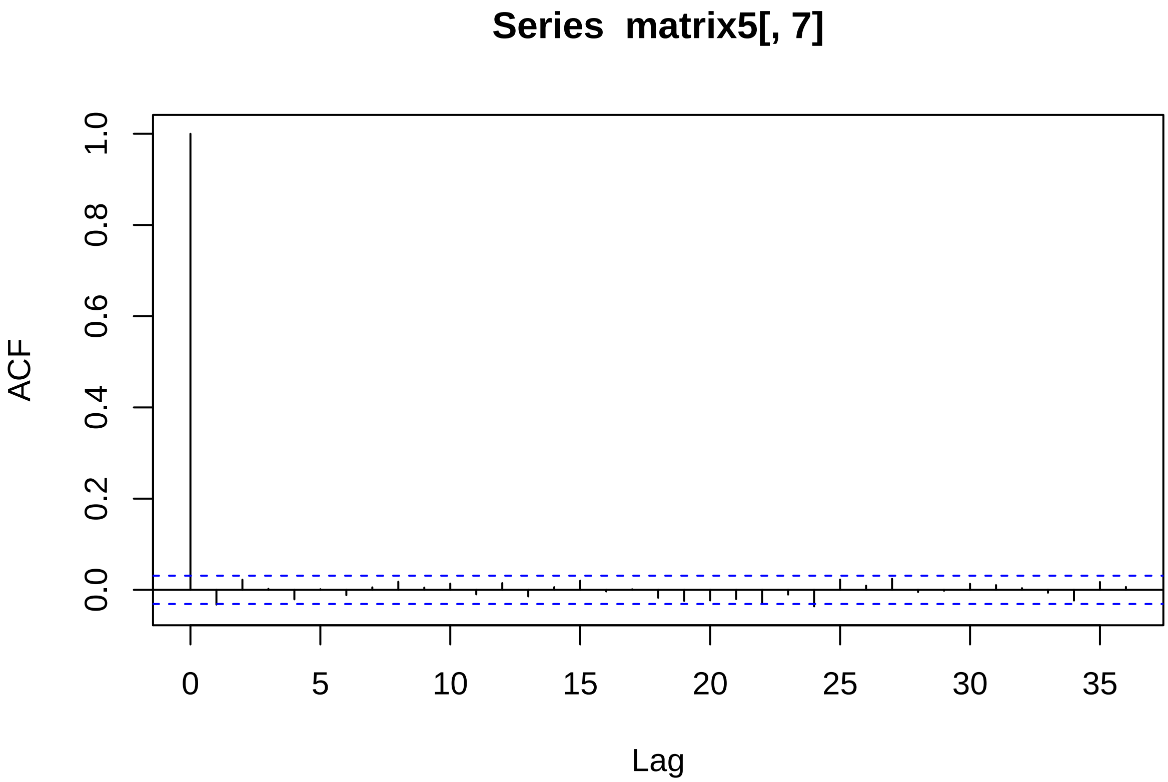

Series matrix5[, 5]
ACF
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag


matrix5[, 5]
0.2
0.3
0.4
0.5
0
1000
2000
3000
4000
Index

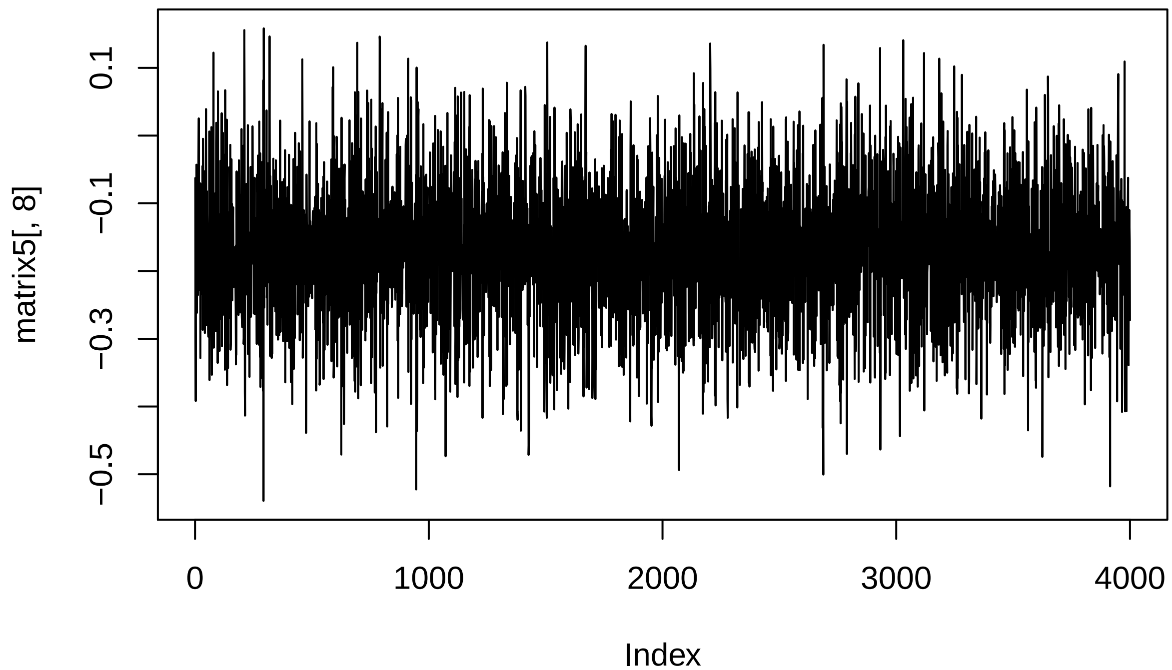

Series matrix5[, 5]
ACF
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
5
10
15
20
25
30
35
Lag

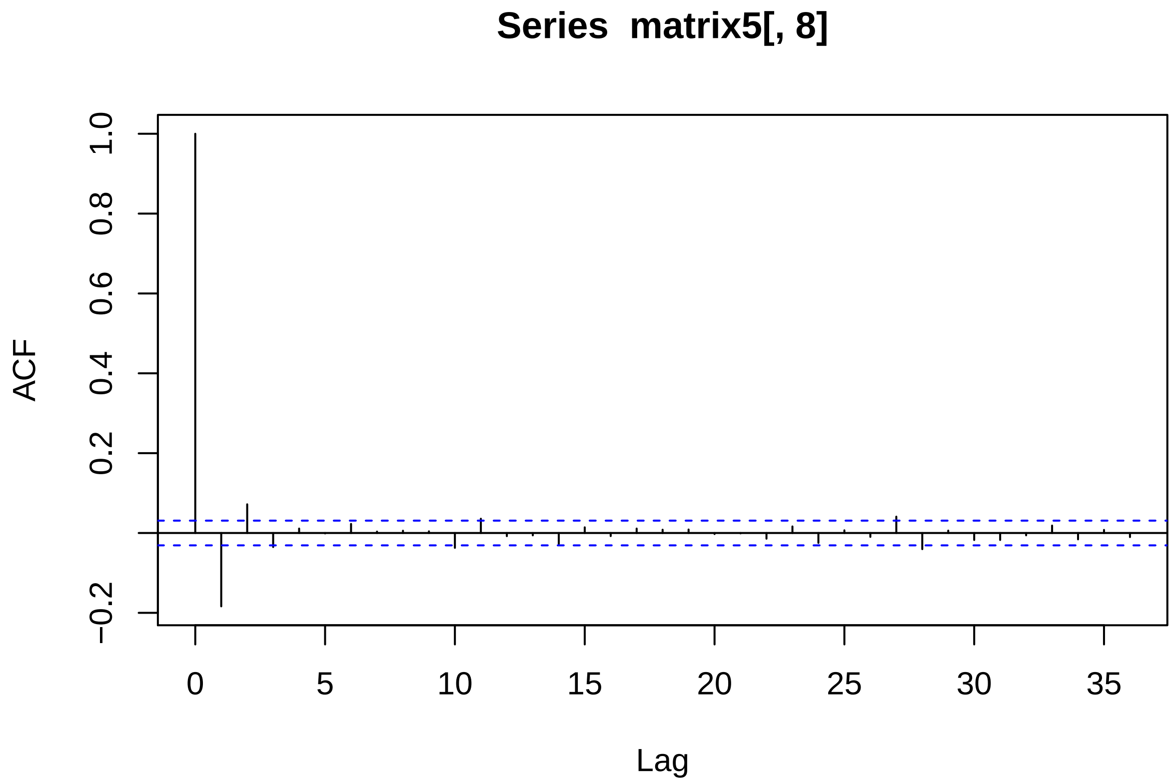


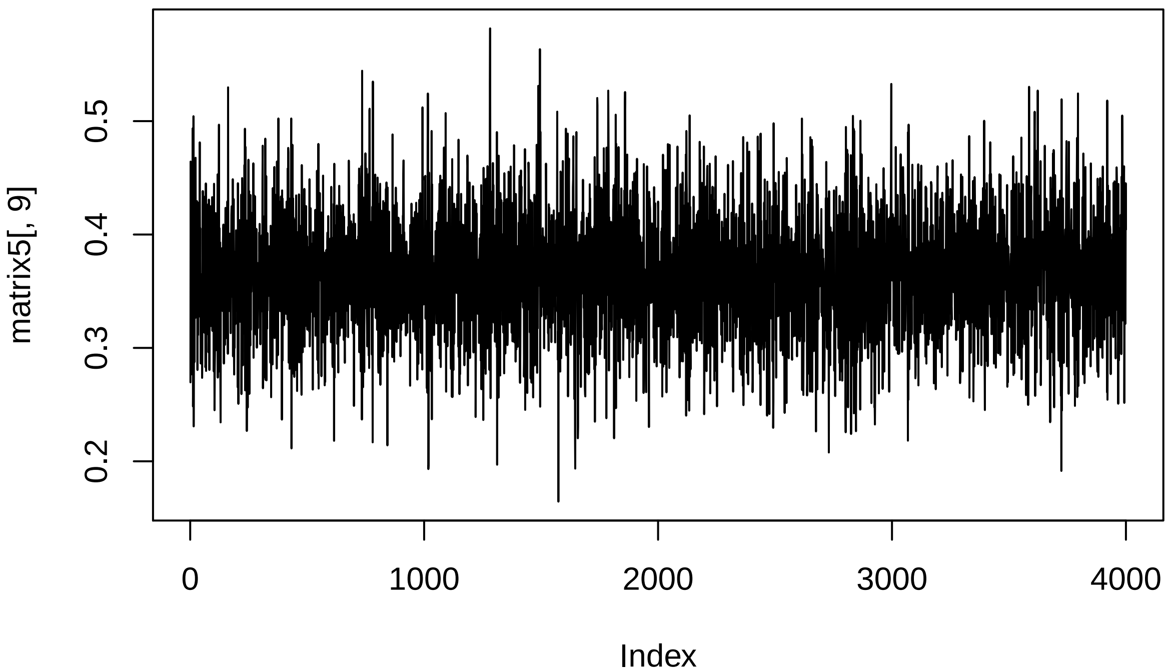
−0.2
−0.3
−0.4
−0.5
−0.6
matrix5[, 7]
0
1000
2000
3000
4000
Index

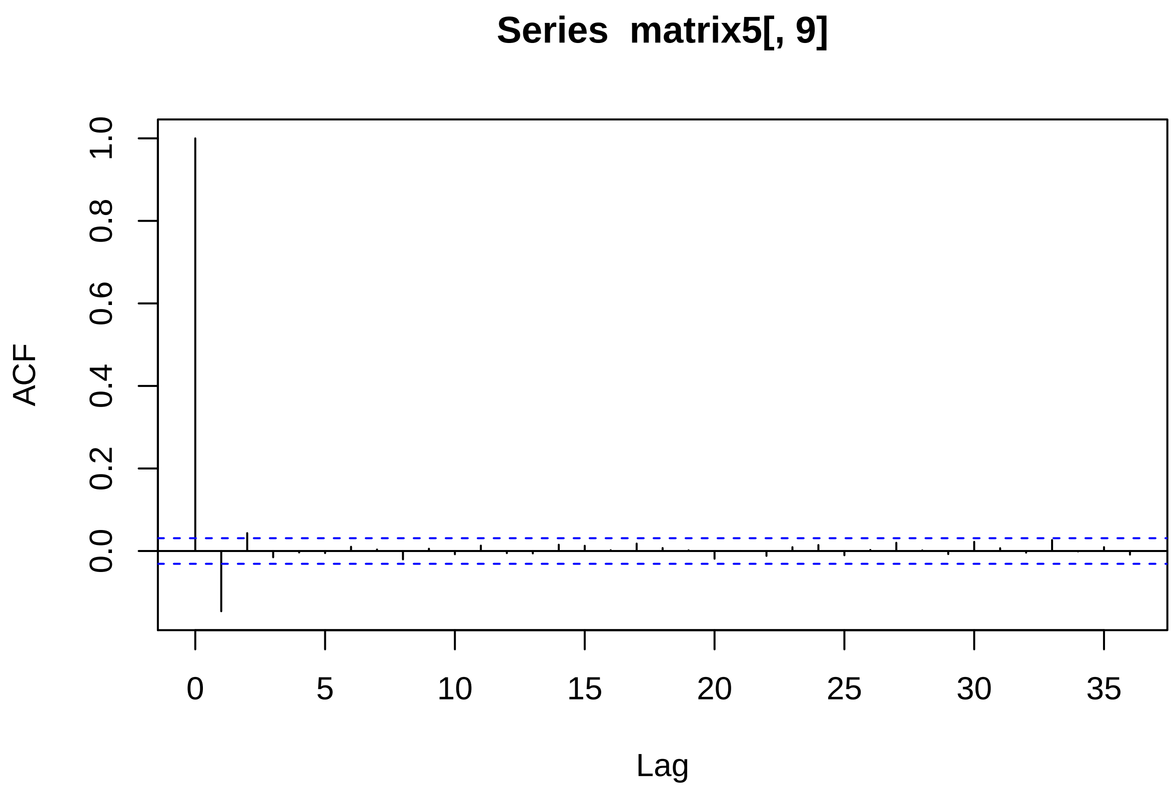


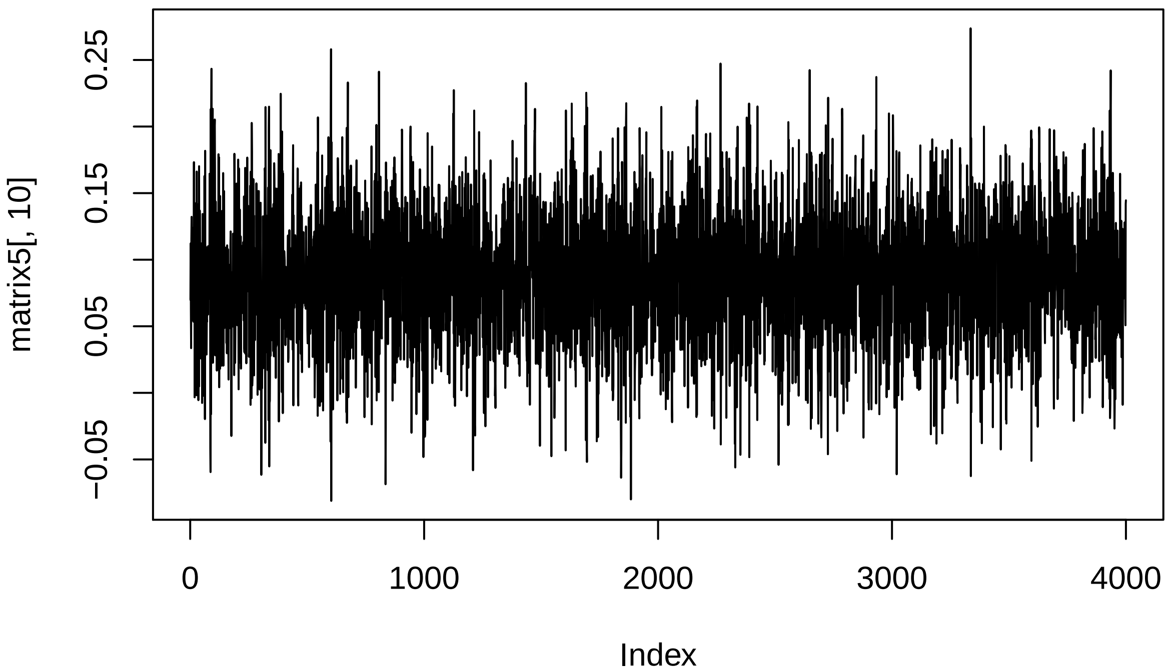
Series matrix5[, 7]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag

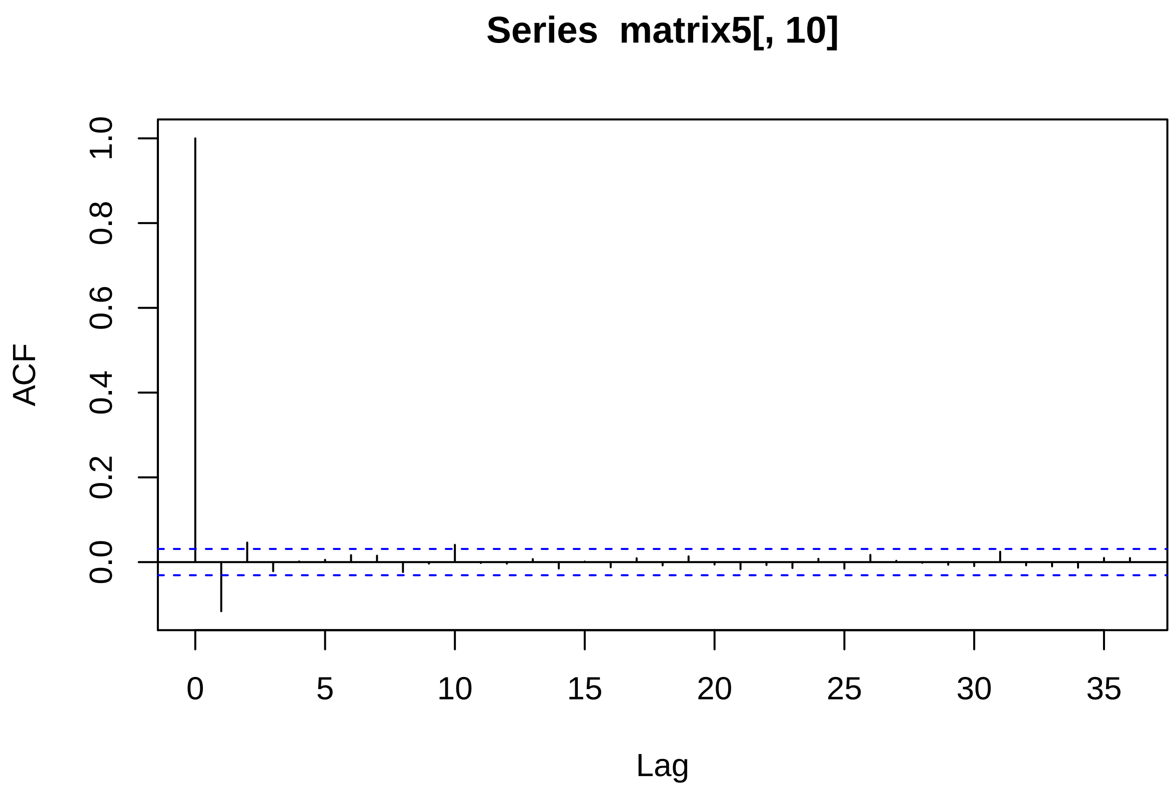

0.1
−0.1
−0.3
−0.5
matrix5[, 8]
0
1000
2000
3000
4000
Index

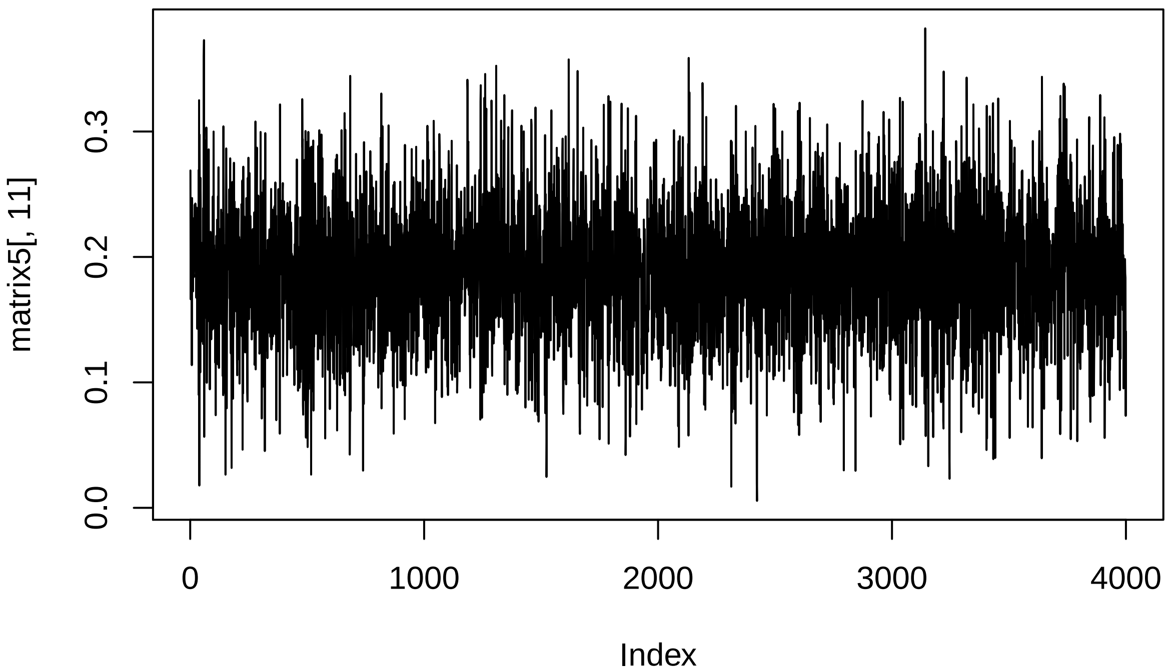

Series matrix5[, 8]
ACF
1.0
0.8
0.6
0.4
0.2
−0.2
0
5
10
15
20
25
30
35
Lag

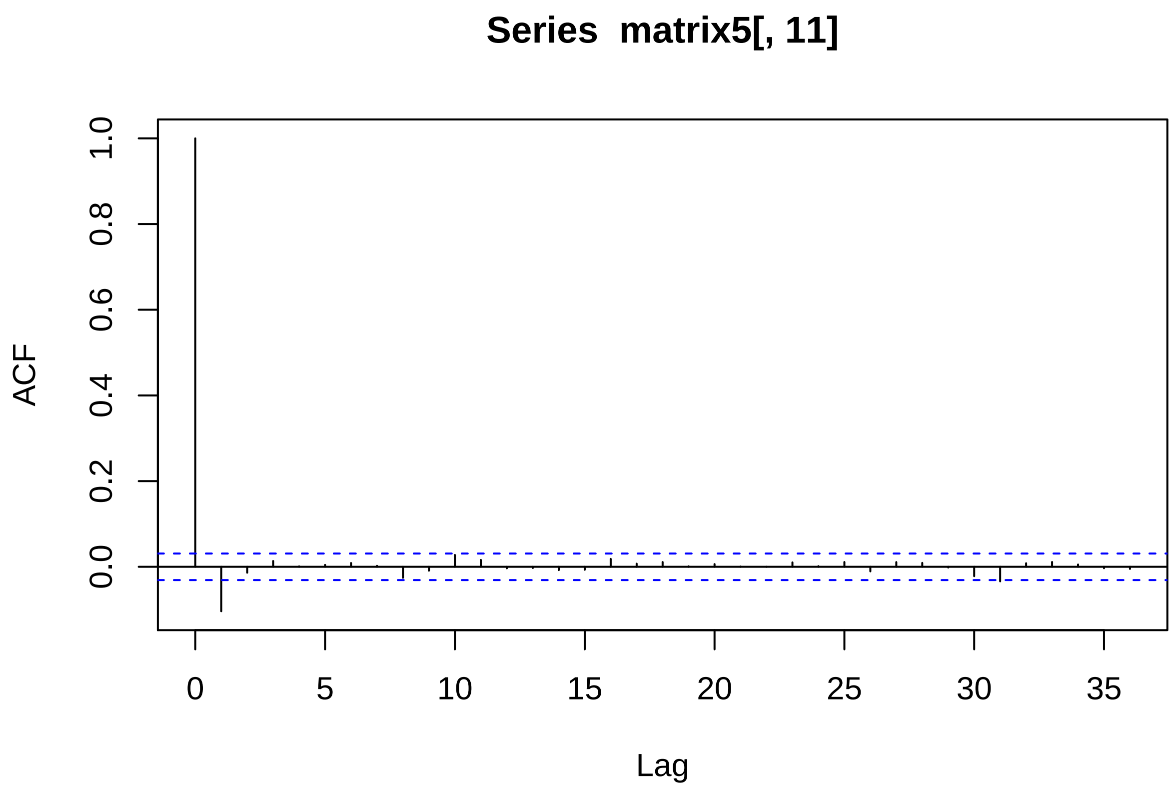

matrix5[, 9]
0.2
0.3
0.4
0.5
0
1000
2000
3000
4000
Index

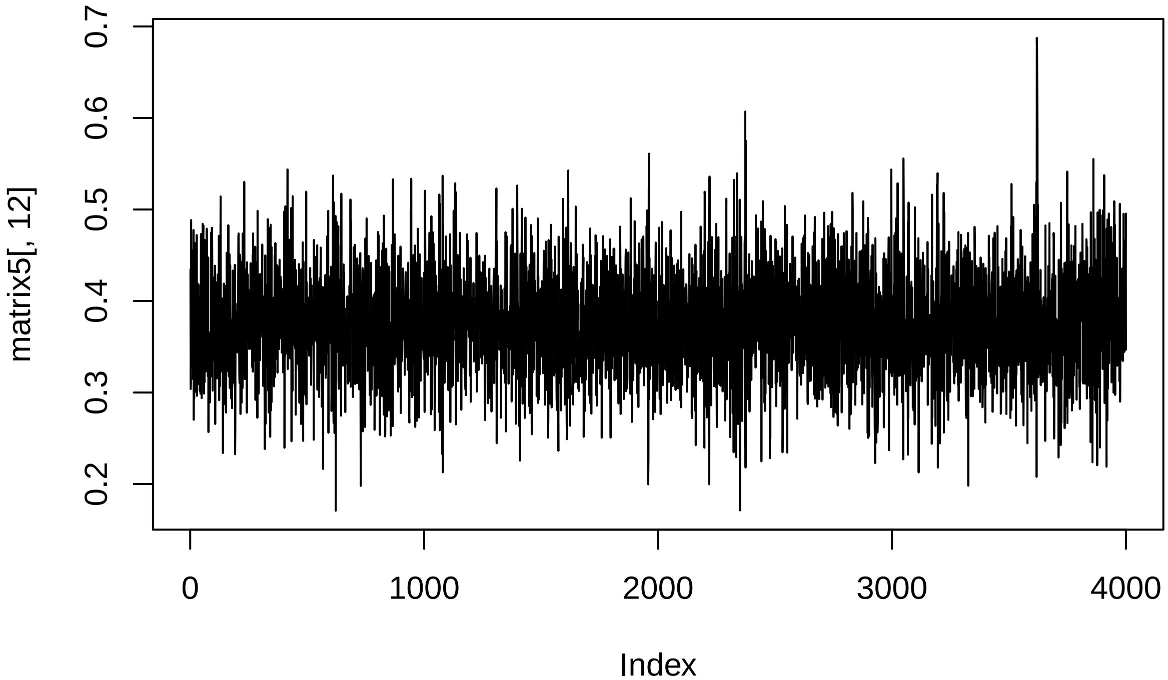

Series matrix5[, 9]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag

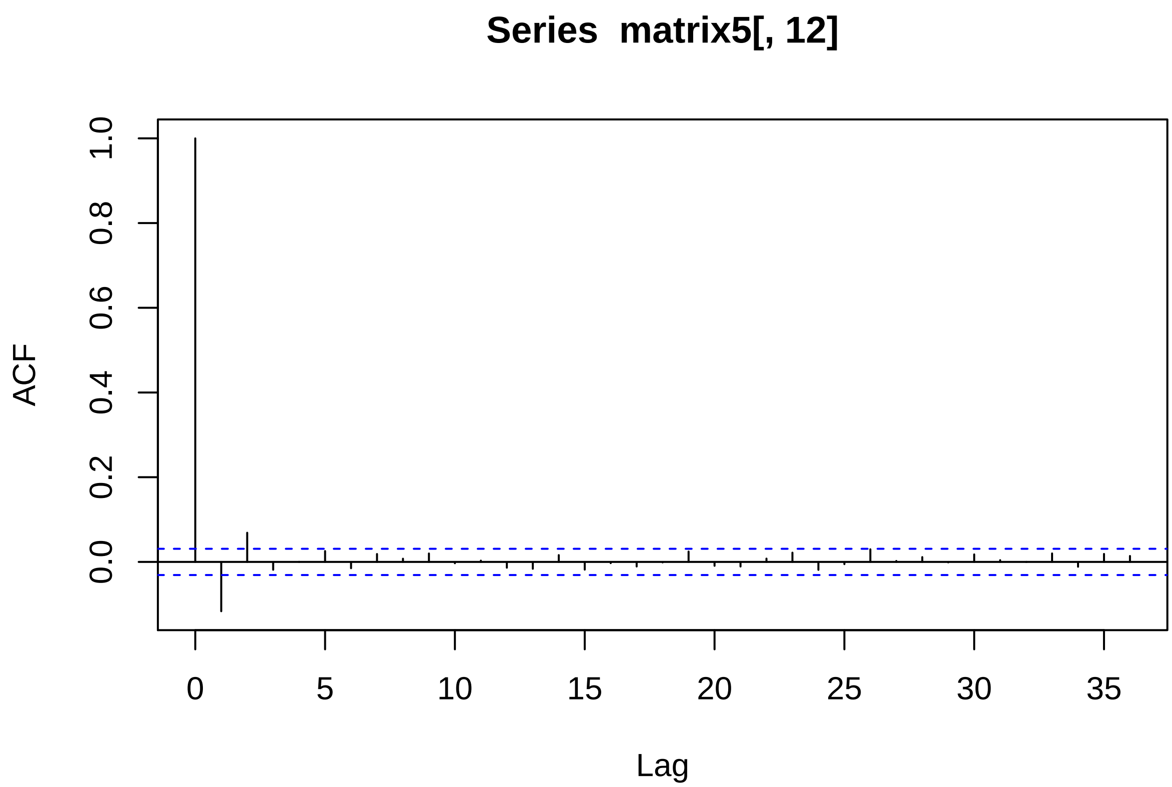

0.25
0.15
0.05
−0.05
matrix5[, 10]
0
1000
2000
3000
4000
Index


Series matrix5[, 10]
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
Lag

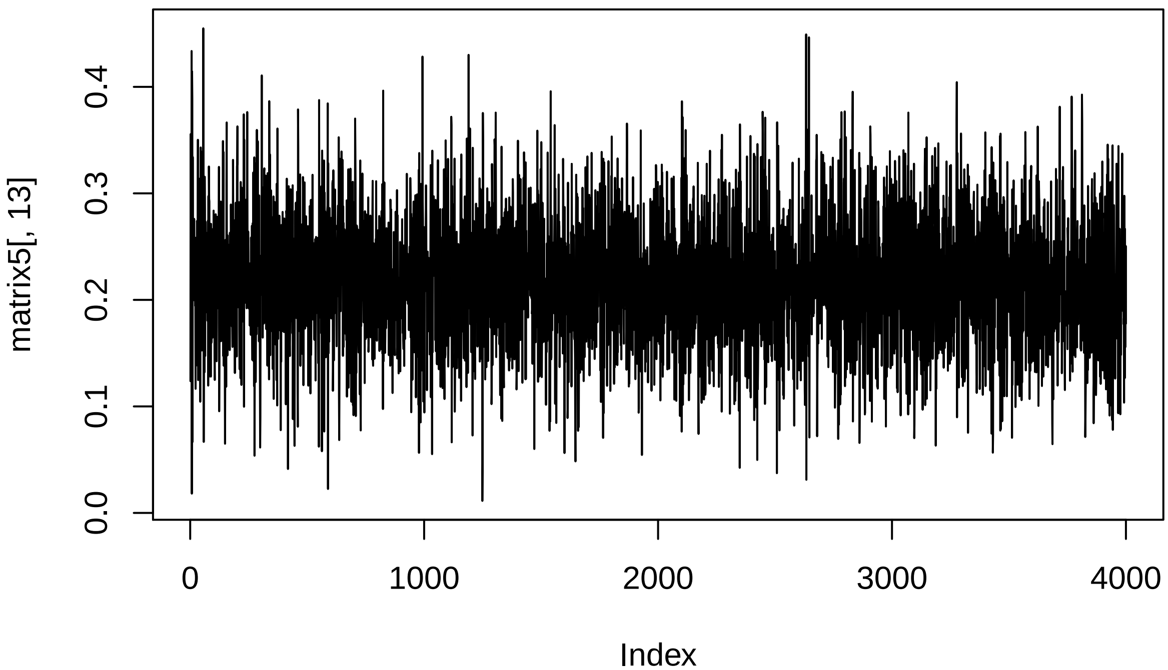

0.0
0.1
0.2
0.3
matrix5[, 11]
0
1000
2000
3000
4000
Index

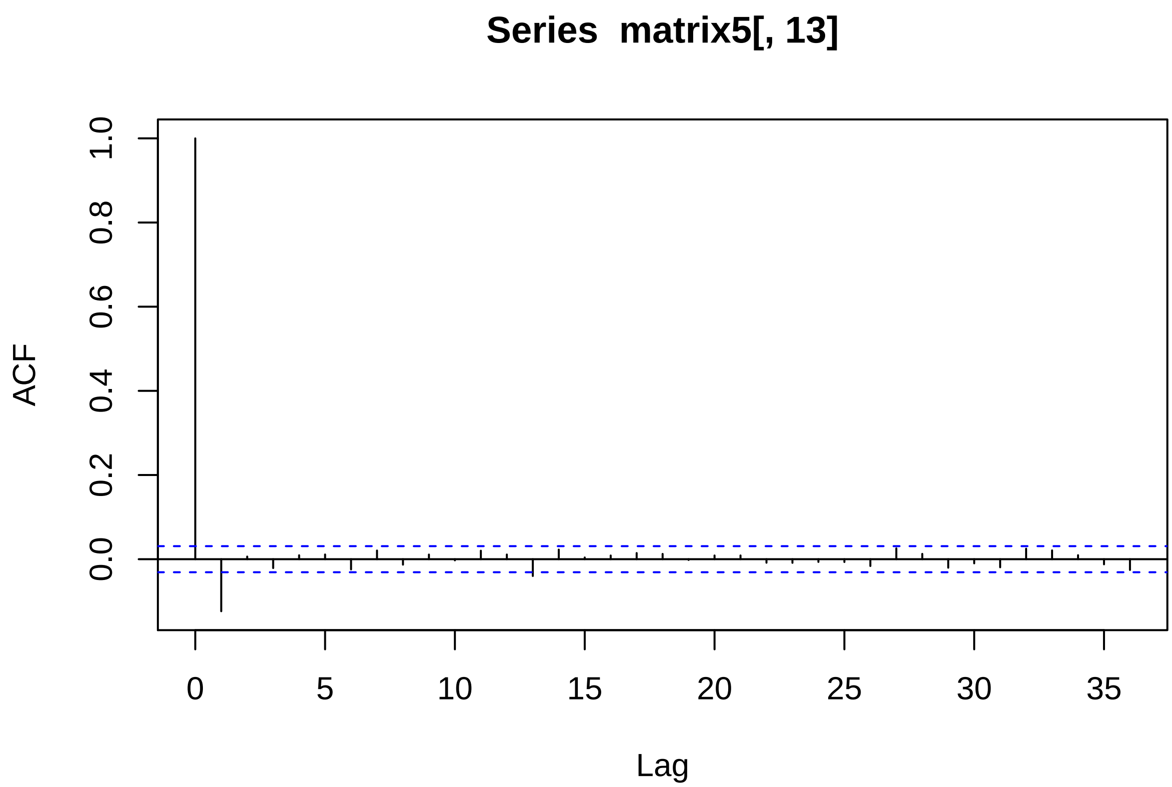

Series matrix5[, 11]
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
Lag


matrix5[, 12]
0.2
0.3
0.4
0.5
0.6
0.7
0
1000
2000
3000
4000
Index



Series matrix5[, 12]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag


matrix5[, 13]
0.0
0.1
0.2
0.3
0.4
0
1000
2000
3000
4000
Index



[1] Betancourt, Michael. How the Shape of a Weakly Informative Prior Affects Inferences, mc-stan.org/users/documentation/case-studies/weakly_informative_shapes.html.

[2] “The Economics of Streaming Is Changing Pop Songs.” The Economist, The Economist Newspaper, 5 Oct. 2019, www.economist.com/finance-and-economics/2019/10/05/the-economics-of-streaming-is-changing-pop-songs.

[3] Gander, Kashmira. “Perfect Pitch: Why Some People Might Have Rare Musical Skill Possessed by Bach and Mozart.” Newsweek, Newsweek, 22 Feb. 2019, www.newsweek.com/perfect-pitch-why-rare-musical-skill-bach-mozart-1326380.

[4] McIntire, George. “A Machine Learning Deep Dive into My Spotify Data.” Open Data Science - Your News Source for AI, Machine Learning & More, 5 Apr. 2018, opendatascience.com/a-machine-learning-deep-dive-into-my-spotify-data/.

[5] Sloan, Nate, and Charlie Harding. “The Culture Warped Pop, for Good.” The New York Times, The New York Times, 14 Mar. 2021, www.nytimes.com/interactive/2021/03/14/opinion/pop-music-songwriting.html?auth=login-google1tap&login=google1tap.

[6] Vehtari, Aki, et al. “Bayesian Logistic Regression with Rstanarm.” Github, 4 Dec. 2019, avehtari.github.io/modelselection/diabetes.html.