Bayesian Inference: Google Stock Data

Connor Kennedy, Rebecca Netson,

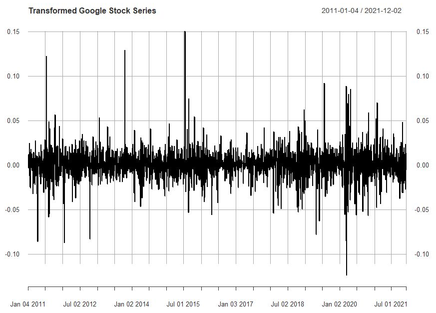
Michaela LaCasse, Alexis Doyle

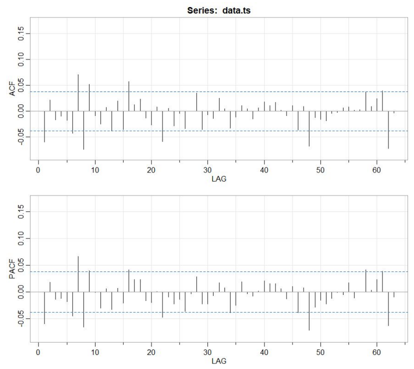
Fall 2021

Introduction

- Google Stock Data from 'tidyquant' package in R
  - Closing Price from 2011 to 2021
  - Closing price is calculated by dividing the total product by the total number of shares traded during the 30 minutes
  - Closing price is the last price at which a security traded during the regular trading day
  - Securitys closing price is the standard benchmark used by investors to track it's performance over time
- Exploratory Analysis:
- Increasing trend from 2011 - 2021
- Possible cyclical trend
- No clear seasonality

Exploratory Analysis of Google Stock Data





- Differencing the log of closing price data to remove the trend at lag 1
- ACF and PACF are shown
  - ACF (Autocorrelation Function): correlation between points separated by various time lags
  - PACF (Partial Autocorrelation): partial correlation between the series and lags of itself
  - For AR(1) the ACF exponentially decreases to 0 as the lag h increases
- Augmented Dicky-Fuller Test to assess stationarity, results shown for linear model with no drift and linear trend with respect to time,
  - ADF = 57, p-value = 0.01 (Stationary)

Autoregressive Model AR(1)

Assumptions:

1. Residuals approximate white noise
2. An AR(1) autoregressive process is one in which the current value is based on the immediately preceding value
3. Stationarity can be assessed by checking the autocorrelation function and whether the process depends on lag 1
    a. A stationary time series is one whose properties do not depend on the time at which the series is observed

General Form

$$y_t = \phi y_{t-1} + \epsilon_t$$
$$\epsilon_t \overset{iid}{\sim} N(0, \sigma^2), \; t = 2, \ldots, T$$
$$[y_t | y_{t-1}, \phi, \sigma^2] \overset{iid}{\sim} N(\phi y_{t-1}, \sigma^2), \; t = 2, \ldots, T$$

Where ϵ_t is white noise, ϕ is a real-value constant and $|\phi| < 1$ following causality criteria. The general form for y_t can be re-written as a linear combination of white noise, i.e:

$$y_t = \sum_{j=0}^{\infty} \psi^j \epsilon_{t-j},$$

where $\psi_j = \phi^j$ and $\psi_1 = \phi$.

Bayesian Approach

The unknown parameters of y_t are ϕ and σ^2. The conditional probability is the likelihood of an outcome occurring, based on a previous outcome occurring, denoted as $L(\phi, \sigma)$. Since we are considering a stationary AR(1) model, we have the following:

$$E(y_1) = \frac{0}{1-\phi} = 0$$

$$Var(y_1) = \frac{\sigma^2}{1-\phi^2}$$

Assuming that $y_t|y_{t-1}$ follows a normal distribution, we thus have:

$$[y_1|\phi, \sigma^2] \sim N(0, \frac{\sigma^2}{1-\phi^2})$$

$$f(y_1|\sigma^2) = \frac{1}{\sqrt{2\pi\frac{\sigma^2}{1-\phi^2}}} e^{-\frac{y_1^2}{2\sigma^2/1-\phi^2}}$$

Likelihood and Priors

The likelihood may be written as:

$$L(\phi, \sigma) = f(y_1|\phi, \sigma^2) \prod_{t=2}^{T} f(y_t|y_{t-1}, \phi, \sigma^2)$$

$$L(\phi, \sigma^2) = \frac{1}{\sqrt{2\pi \frac{\sigma^2}{1-\phi^2}}} e^{-\frac{y_1^2}{2\sigma^2/1-\phi^2}} \prod_{t=2}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_t - \phi y_{t-1})^2}{2\sigma^2}}$$

Analogous to Bayesian inference for unknown variance in linear regression, the Gamma prior for $\frac{1}{\sigma^2}$ is a prior to the AR(1) likelihood with parameters α, β, given as:

$$[\frac{1}{\sigma^2}] \sim Gamma(\alpha, \beta)$$

The prior for ϕ is uniform:

$$[\phi] \sim Uniform(-1, 1)$$

Simulation Study

- We conducted a simulation study to assess the performance of our Bayesian estimation method
- We simulated 1,000 Monte Carlo samples for each of twelve possible combinations of φ and σ

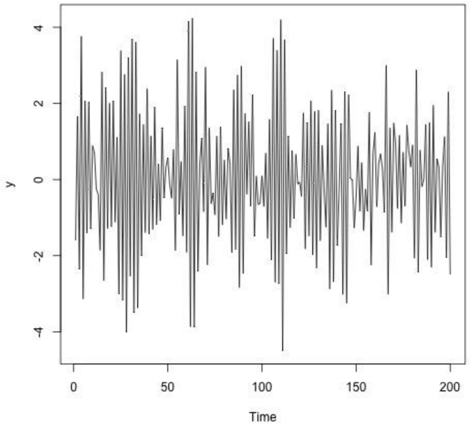
Table 1: Parameter values used to simulate data

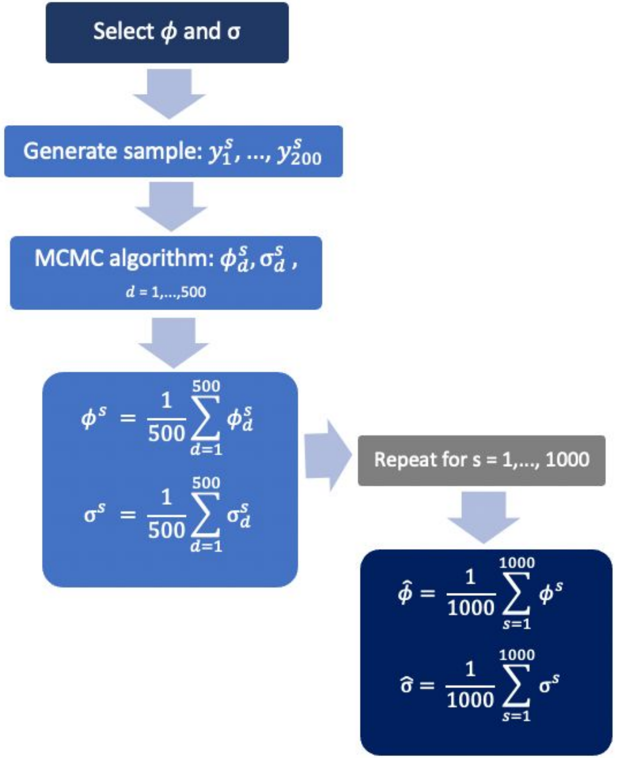
ϕ	σ
-0.92	1
-0.55	5
0.12	10
0.78	

$$y_t = \phi y_{t-1} + \epsilon_t, \text{ where } \epsilon_t \overset{iid}{\sim} N(0, \sigma^2) \text{ for } t = 2, \ldots, 200$$

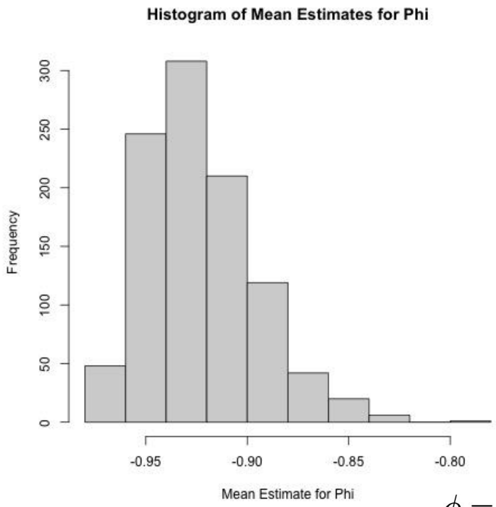
$$[y_1 | \phi, \sigma^2] \sim N(0, \tfrac{\sigma^2}{1-\phi^2})$$

Simulated time series when $\phi = -0.92$ and $\sigma = 1$

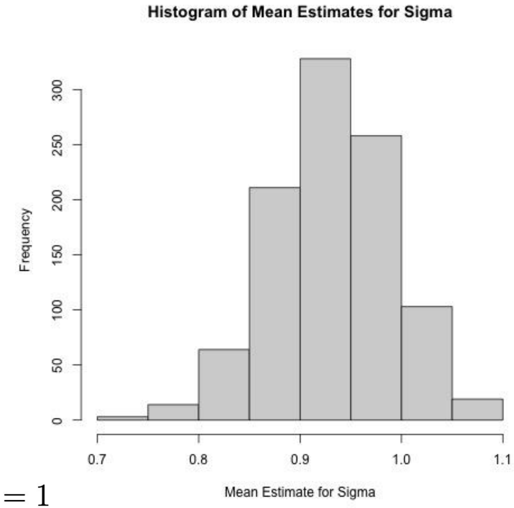




- We fit a Bayesian structural model to each of the simulated samples
- The model uses a Monte Carlo Markov Chain (MCMC) algorithm to estimate φ and σ
  - Model fit is based on 500 MCMC draws from the posterior distributions
  - We take the mean of the estimates for φ and σ to obtain point estimates for each Monte Carlo sample



$\phi = -0.92$ and $\sigma = 1$



- We take the mean of the point estimates across the 1,000 samples and assess performance

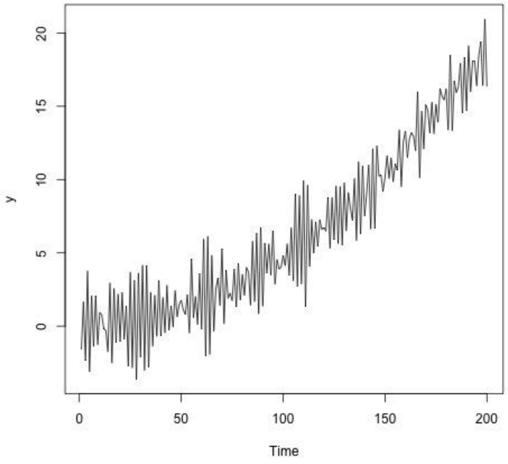
Parameter estimates obtained under different settings

Parameter setting:		$\phi = -0.92$	$\phi = -0.55$	$\phi = 0.12$	$\phi = 0.78$
$\sigma = 1$	Mean $\hat{\phi}$	-0.9228	-0.5712	0.5051	0.8472
	Bias($\hat{\phi}$)	-0.0028	-0.0212	0.3851	0.0672
	MSE($\hat{\phi}$)	0.0007	0.0028	0.1488	0.0059
$\sigma = 5$	Mean $\hat{\phi}$	-0.9566	-0.5667	0.3399	0.8181
	Bias($\hat{\phi}$)	-0.0366	-0.0167	0.2199	0.0391
	MSE($\hat{\phi}$)	0.0014	0.0020	0.0986	0.0025
$\sigma = 10$	Mean $\hat{\phi}$	-0.9108	-0.6546	0.2449	0.8384
	Bias($\hat{\phi}$)	-0.0092	-0.1046	0.1249	0.0584
	MSE($\hat{\phi}$)	0.0027	0.0130	0.0494	0.0034
$\sigma = 1$	Mean $\hat{\sigma}$	0.9319	1.0172	0.2775	0.9462
	Bias($\hat{\sigma}$)	-0.0681	0.0172	-0.7225	-0.0538
	MSE($\hat{\sigma}$)	0.0081	0.0010	0.5254	0.0044
$\sigma = 5$	Mean $\hat{\sigma}$	4.8603	4.3433	2.9021	4.8668
	Bias($\hat{\sigma}$)	-0.1397	-0.6567	-2.0979	-0.1332
	MSE($\hat{\sigma}$)	0.0543	0.6457	4.9122	0.0410
$\sigma = 10$	Mean $\hat{\sigma}$	9.3603	7.8678	5.1550	8.3486
	Bias($\hat{\sigma}$)	-0.6397	-2.1322	-4.8450	-1.6514
	MSE($\hat{\sigma}$)	1.1362	5.8419	23.4740	2.8736

- Bias and MSE for estimates of φ and are small, except when φ = 0.12
- When φ is close to zero, y_t is largely determined by ε_t
  - Fitting procedure finds structure in the noise
  - Bias and MSE decrease as σ increases
- Bias and MSE of estimates for σ increase as the true value of σ increases
- Bias and MSE of estimates for σ are the largest when φ = 0.12

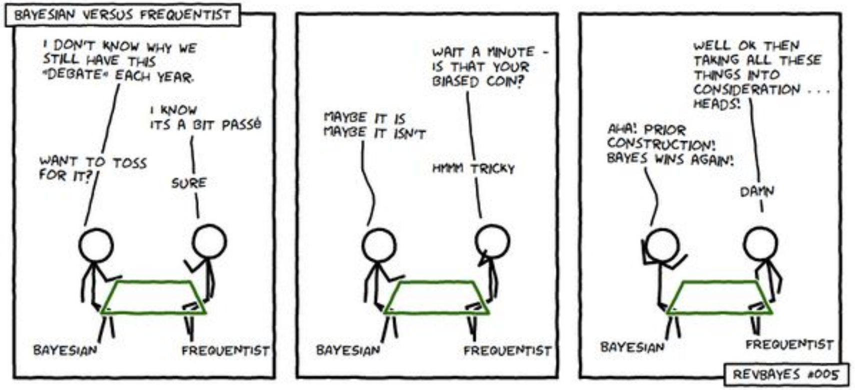
- Simulation results suggest that BSTS modeling approach yields accurate estimates for φ when the true value is not close to zero
- Estimates for σ are less accurate when the true value of σ is large
- Interpretations are based on simulations where the data were generated from an AR(1) process
- What if the underlying data are not generated from an AR(1) process...

$$y_t = \alpha t^2 + \phi y_{t-1} + \epsilon_t, \text{ where } \epsilon_t \overset{iid}{\sim} N(0, \sigma^2) \text{ for } t = 2, \ldots, T \text{ and } 0 < \alpha < 1$$



Fitting to Google Stock Data

- We will compare frequentist and bayesian time series models
- Frequentist
  - Arima(1,1,0)
  - Differenced first-order auto regressive model
- Bayesian
  - Structural model with ‘bsts’
  - Auto regressive local level term
- ‘GOOGL’ daily closing price from January 1, 2011 to June 30, 2021
  - Train model on 2011-2020
    - (10 years)
  - Validate with 2021
    - (6 months)

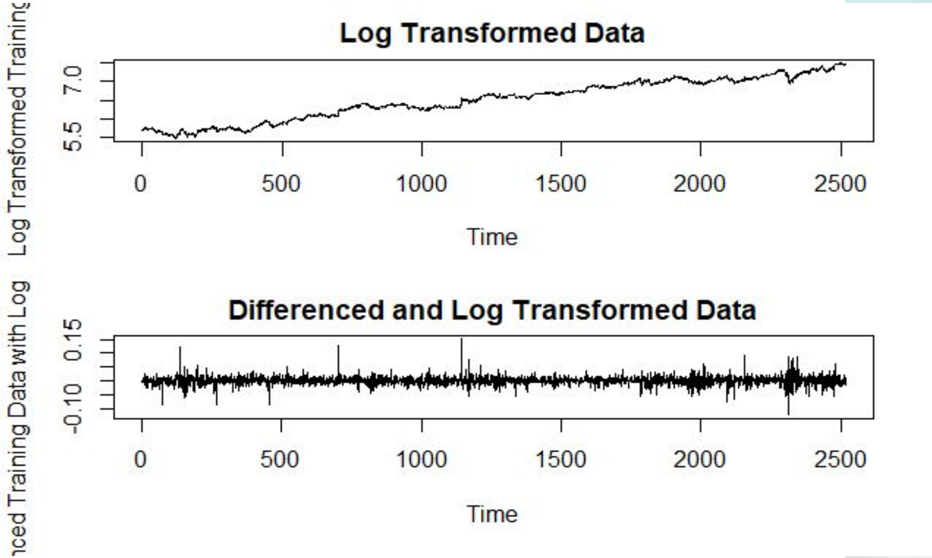


Frequentist Time Series

- This method was done as a comparison to the Bayesian method to see which is better based on overall MAPE value.
- Steps that were followed:
  - Split data into training and validation
  - Plot raw data for training to see if stationary
  - If not stationary, transform training data (using just log, just differencing, or by doing a log transformation and then differencing the data)
  - Look at ACF/PACF to decide what model the data would use
  - Fit that model using arima() function in R
  - Extract the fitted values of the model using predict() function in R
  - Reverse the log transformation for the predicted values to get an accurate MAPE value
  - Finally, find the MAPE value using 'mean(abs(actual-fitted)/actual)'

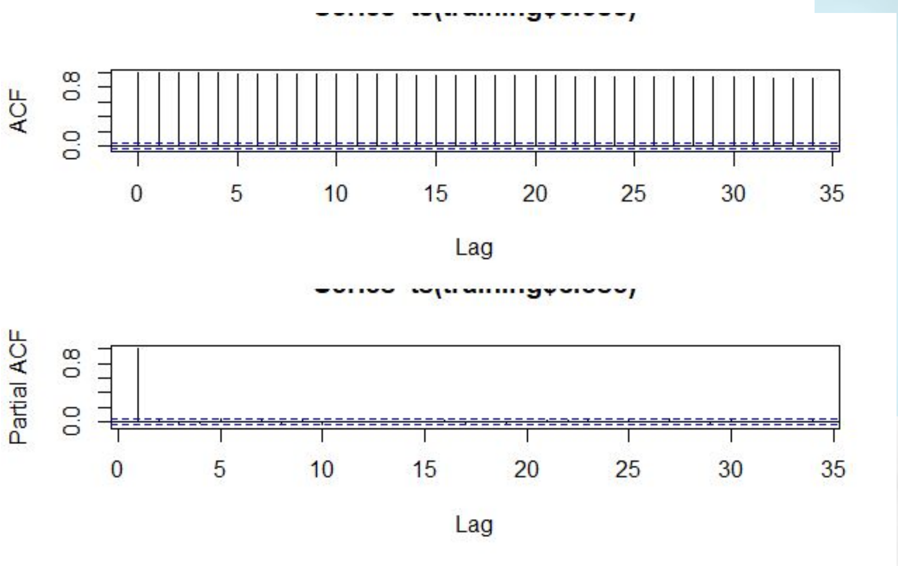
Process to Transform Data

- Raw data for training data set looked like the plot in the introduction
- We wanted to look at the data using the log transformation first since we wanted to make the variance constant across time.
- The plot still showed an upward trend, so we took the difference of it too to make it stationary.
- Transforming the data isn't always necessary



ACF/PACF of Training Data

- Look at ACF and PACF to determine possible models for the data
- ACF shows lags that slowly go down until they reach 0
- PACF has a cutoff after the first lag
- These show characteristics of an AR(1) model, so that is the model we will move forward with

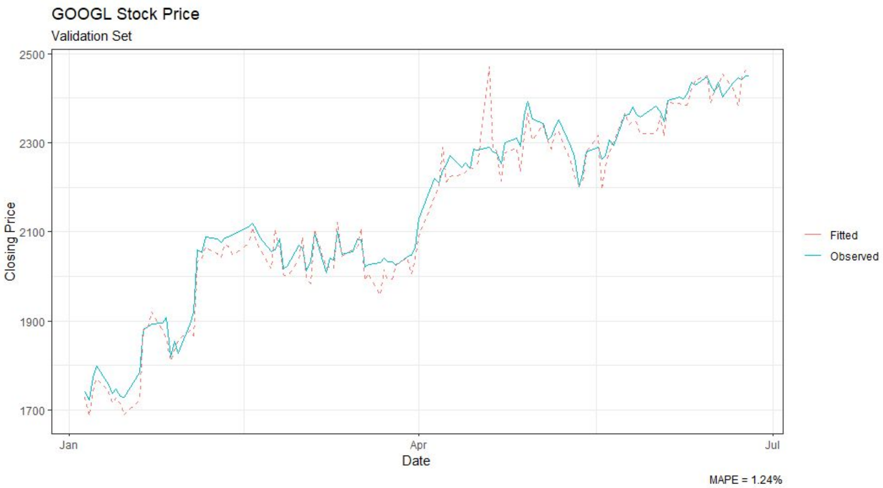


Overview of Rest of the Process

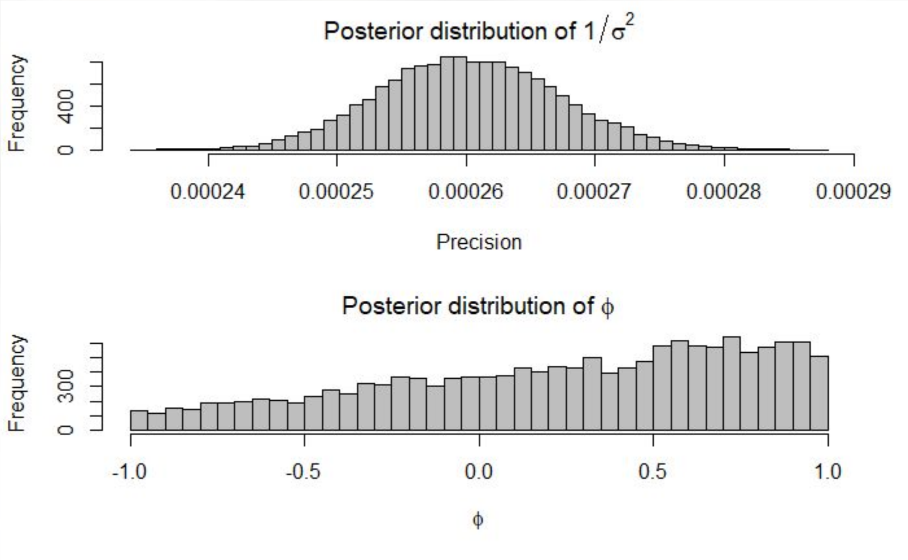
- Difference transformation was used solely to determine possible models and was, then, taken out to fit since the arima() function can do that for us
- The model was fit using arima(1,1,0) as said before.
  - First 1 representing AR(1) model
  - Second 1 representing the first-order difference of the data
- Fitted values were extracted for fitting purposes using predict() function
- Undid log transformation on fitted values using exp() to get accurate MAPE value.
- MAPE value was 1.8%

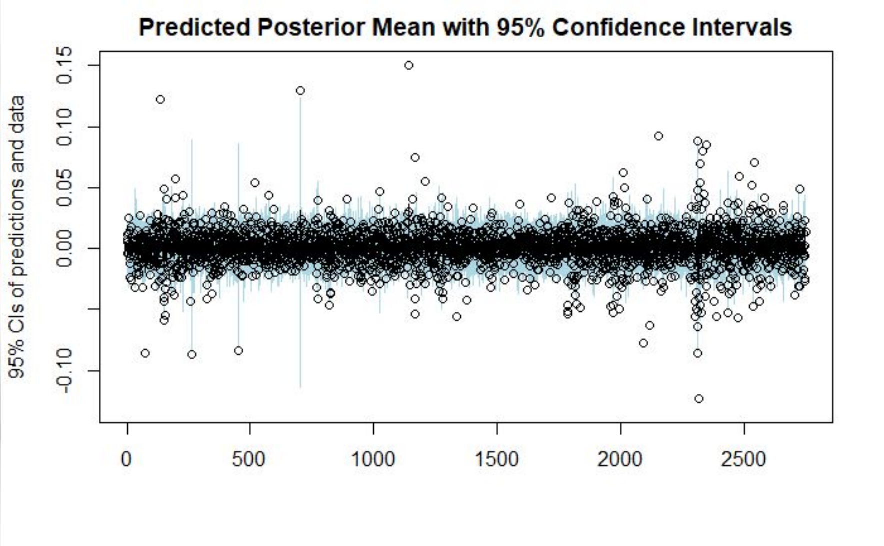
Bayesian Time Series

- State Specification
  - Auto regressive local level term
- Fit model with 1000 MCMC iterations
  - Burn first 32 for better starting value
- Use MCMC to find mean of parameters
  - $\phi = 0.99$
  - $\sigma = 0.02$
- Predict values using validation set
  - MAPE = 1.2%



Simulation using JAGS (Just Another Gibbs Sampler)





Sources

Scott, S. L. (2021, July 02). Bsts: Bayesian Structural Time Series (Version 0.9.7) [Program documentation]. Retrieved from https://cran.r-project.org/web/packages/bsts/index.html

Alphabet Inc. (GOOG) Stock Historical Prices & Data. (2021, December 17). Retrieved from https://finance.yahoo.com/quote/GOOG/history?p=GOOG

Dancho, M., & Vaughan, D. (2021, March 05). Tidyquant: Tidy Quantitative Financial Analysis (Version 1.0.3) [Program documentation]. Retrieved from https://cran.r-project.org/web/packages/tidyquant/index.html