Time Series Forecasting of Store Sales: ARIMA, RNN, LSTM, and GRU Time Series Modeling

Shuo Li, Kai Li, Jun Hyuk Suh

Applied Mathematics and Statistics, Stony Brook University

April 25, 2022

Introduction

In this project, we will be using a store sales dataset (source: `https://www.kaggle.com/c/store-sales-time-series-forecasting`) from Kaggle to perform time series forecasting using classical autoregressive integrated moving average (ARIMA) modeling, recurrent neural networks (RNN), long short-term networks (LSTM), and gated recurrent units (GRU).

Data

The dataset includes dates, store, item information, promotions, and unit sales from Corporation Favorita, a large Ecuadorian-based grocery retailer. The train data covers sales information from January 1, 2013 to August 15, 2017. The test data provides dates, store, item information and promotions from August 16, 2017 to August 31, 2017 to predict grocery sales.

The objective of this project is to present classical and machine learning methods in time series analysis. However, practical solutions and future strategies in management require careful interpretations from the analysis and predictions so that relevant parties can utilize the modeling results for improvements. For grocery stores, more accurate forecasting can decrease food waste related to overstocking and improve customer satisfaction.

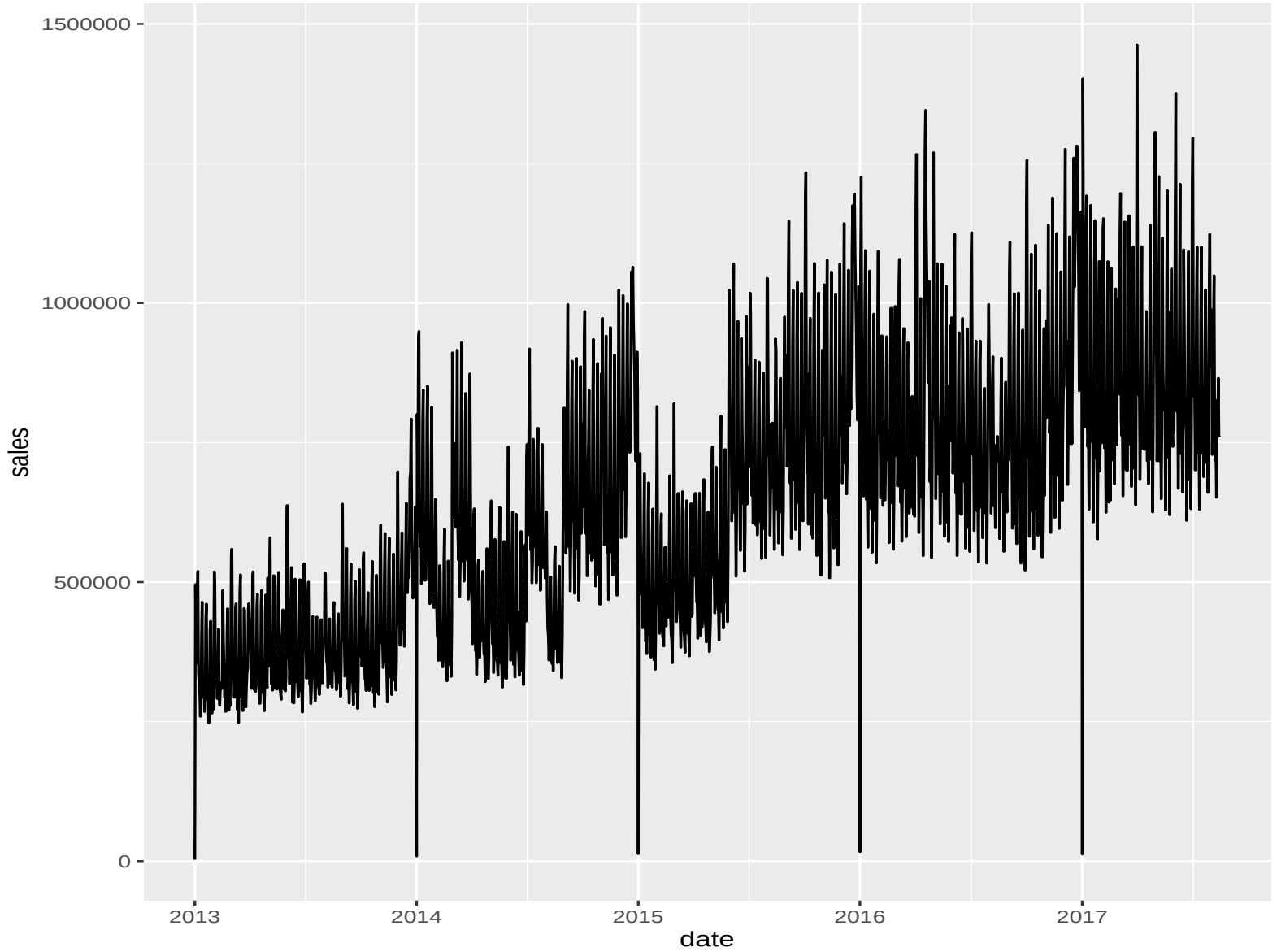
More about the Data

For this project, we will be modeling and forecasting the aggregated sales of all stores of all items on a given day instead of individual items sold in individual stores. We have 1684 days of aggregated sales in total for training. The summary statistics for the aggregated sales are given in the table below.

Table: Summary Statistics

Name	Obs	Mean	Std. Dev.	Min	Max
sales	1684	637556	234410.2	2512	1463084

Data Visualization



ARIMA

Autoregressive integrated moving average (ARIMA) models are commonly used in time series modeling, denoted ARIMA(p,d,q). A special case of ARIMA is the autoregressive moving average (ARMA) model, where ARMA provides a stationary stochastic process, denoted ARMA(p,q). In an ARMA model, AR stands for autoregression and MA stands for moving average. The AR part involves regressing the variable on its own lagged (i.e., past) values, and p is the order (number of time lags) of the AR model. The MA part involves modeling the error term as a linear combination of error terms occurring contemporaneously and at various times in the past, and q is the order of the MA model.

ARIMA

If data show evidence of non-stationarity in the sense of mean, an initial differencing step (corresponding to the "integrated" part of the ARIMA model) can be applied one or more times to eliminate the non-stationarity of the mean function (i.e., the trend). d is the degree of differencing.

Further information about ARMA and ARIMA:

1. `https://en.wikipedia.org/wiki/Autoregressive-moving-average_model`
2. `https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average`

Stationarity

There are existing hypothesis testing and visualization methods to help us determine the potential ARIMA models. Though we will be using R to fit the best ARIMA model according to some information criterion, we want to check if the preliminary methods match our expectations. Before building an ARIMA model, the first step is to see if the series is stationary. Dickey–Fuller test can be used to reach the objective.

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ots
## Dickey-Fuller = -7.1184, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary
```

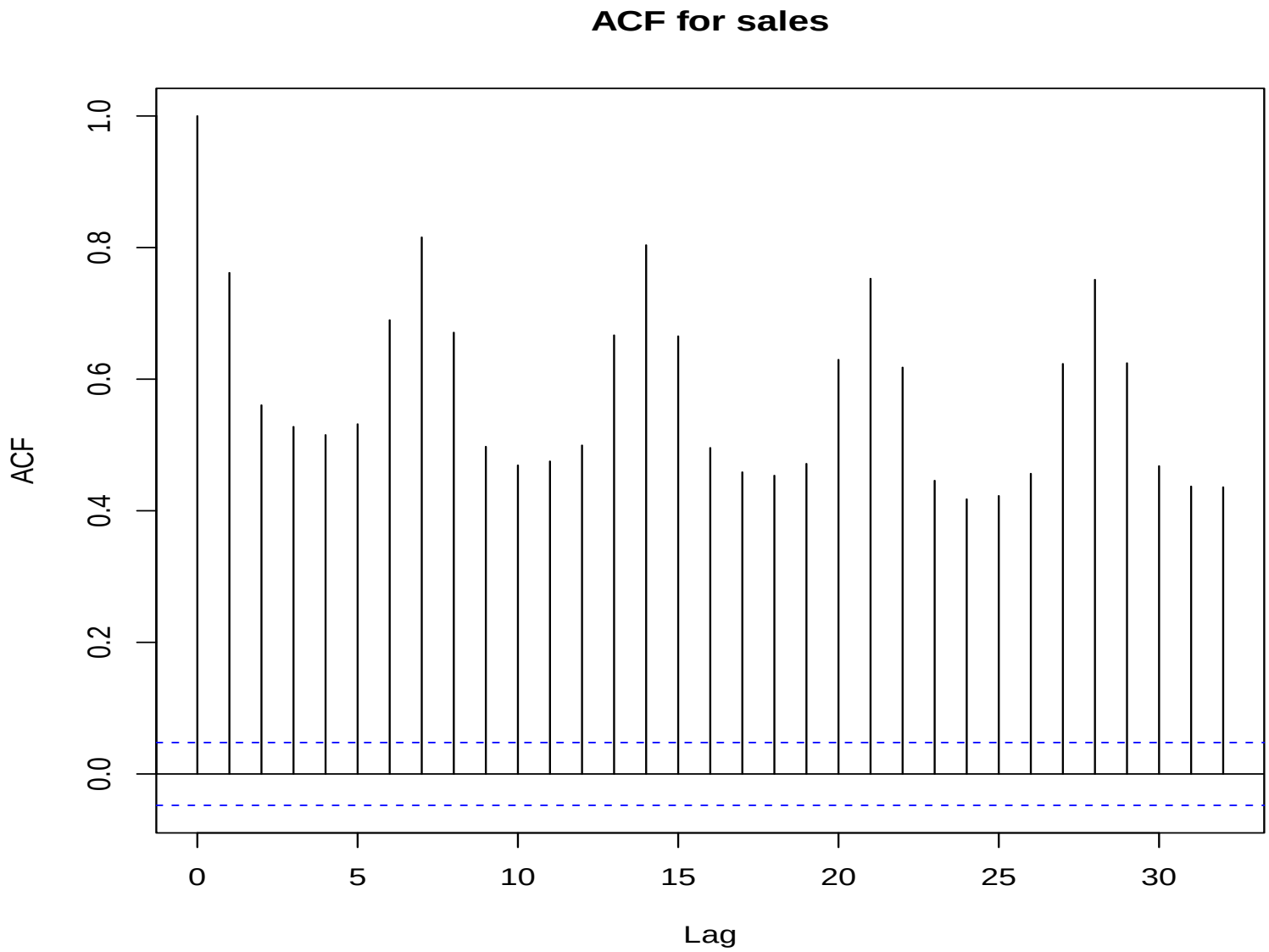
The p-value for the test statistic is less than 0.01, which indicates that the series is stationary. In other words, we should expect the differencing order d to be 0.

ACF and PACF

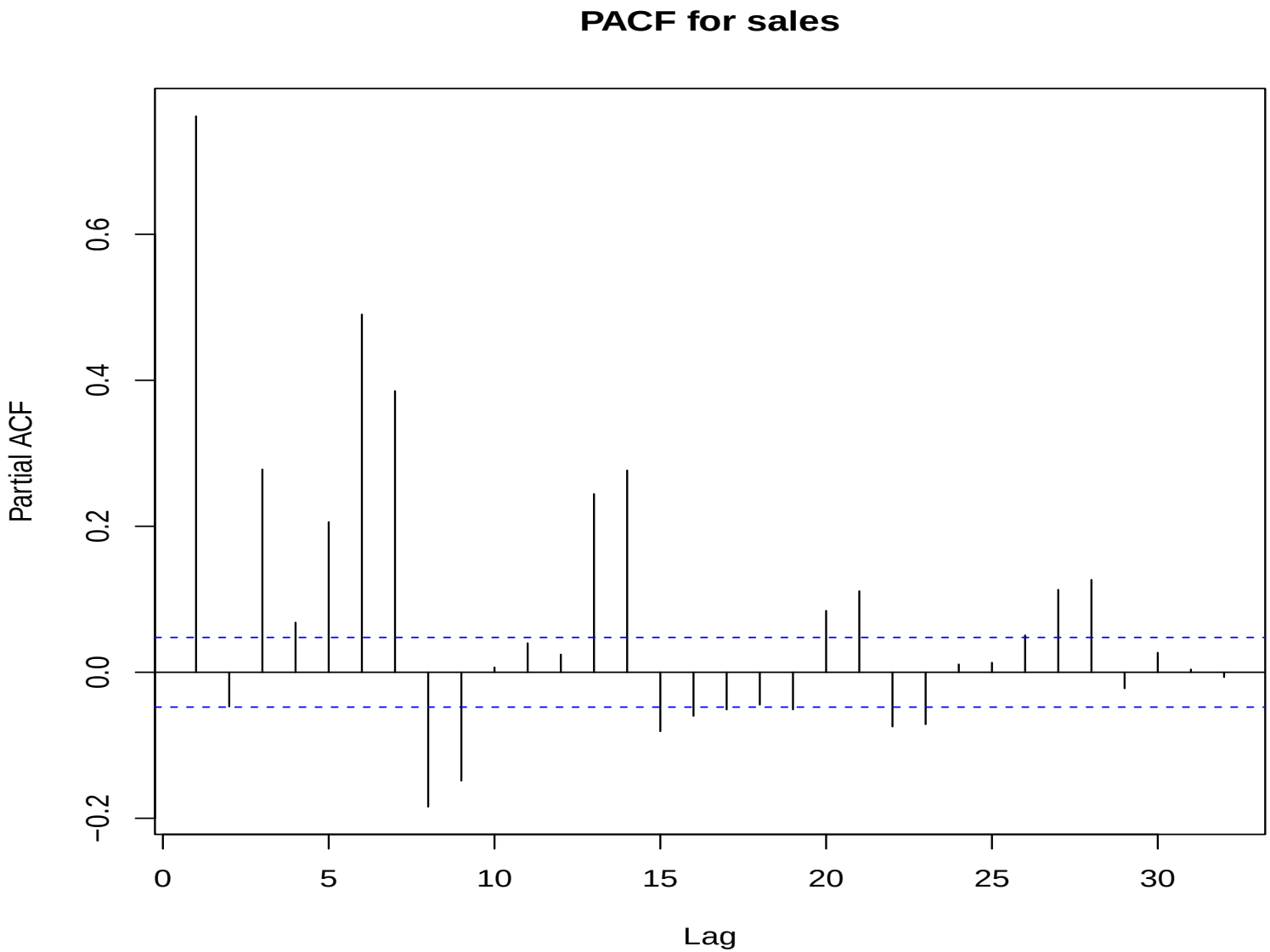
Next, we want to plot the autocorrelation function (ACF) plot and the partial autocorrelation function (PACF) plot to identify whether the process is an AR, an MA process, or a combination of both. Moreover, we may find the optimal parameters or orders of the ARMA process based on the two plots.

More specifically, ACF is an (complete) autocorrelation function which gives us values of autocorrelation of series with its lagged values. PACF is a partial (not complete) autocorrelation function that finds correlation of the residuals with the next lag value.

ACF



PACF



ACF and PACF

The ACF shows a gradually decreasing trend, while the PACF cuts immediately after one lag, three lags, etc. Thus, the graphs suggest that an AR model would be appropriate for the time series. The degree p may be 1, 3, etc. q would be 0. That is, we may have several combinations of orders p, d and q for the ARIMA model such as ARIMA(1,0,0).

Fitting the Best ARIMA Model

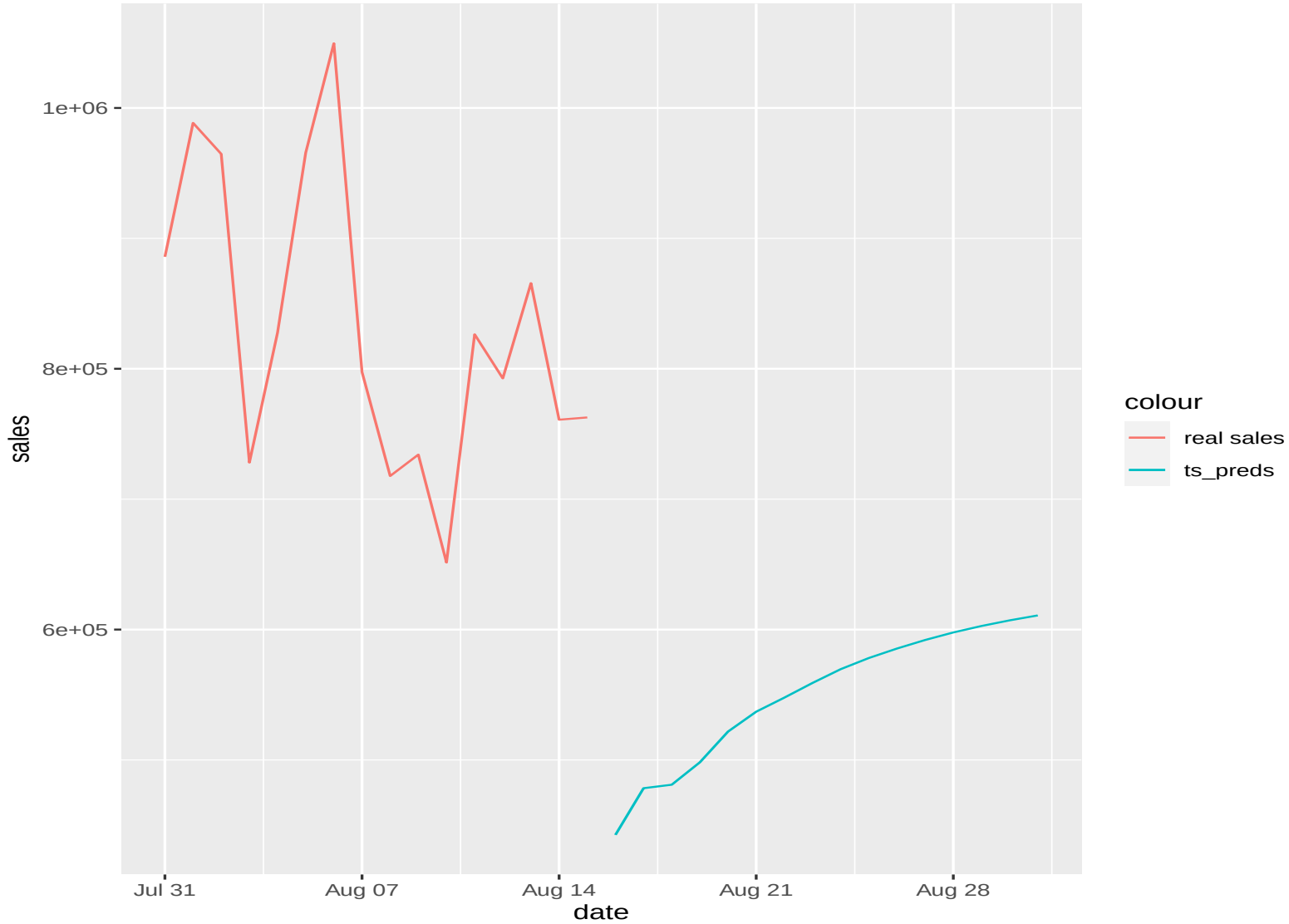
Now we want to see if the best ARIMA model coincides with our observations from the basic analysis. In empirical studies, we normally restrict the orders in an ARIMA model and search for the best model given the constraints. In this project, we will set the maximum order of p to be 3. We don't have to specify the maximum orders for d and q given our previous results. Furthermore, we will be using the Bayesian information criterion (BIC) to evaluate the model estimations. We have the final ARIMA model results on the next slide.

Fitting the Best ARIMA Model

```
## Series: ots
## ARIMA(3,0,0) with non-zero mean
##
## Coefficients:
##          ar1      ar2     ar3       mean
##       0.8200  -0.2808  0.2847  636705.12
## s.e.  0.0234   0.0301  0.0234   20034.97
##
## sigma^2 = 2.111e+10:  log likelihood = -22458.33
## AIC=44926.67   AICc=44926.7   BIC=44953.83
##
## Training set error measures:
##                     ME     RMSE      MAE      MPE     MAPE
## Training set 53.55204 145135.6 111865.9 -46.128 59.52955 0.
```

Prediction

Finally, we want to use the best ARIMA model to build a 16-day prediction from August 16, 2017 to August 31, 2017.



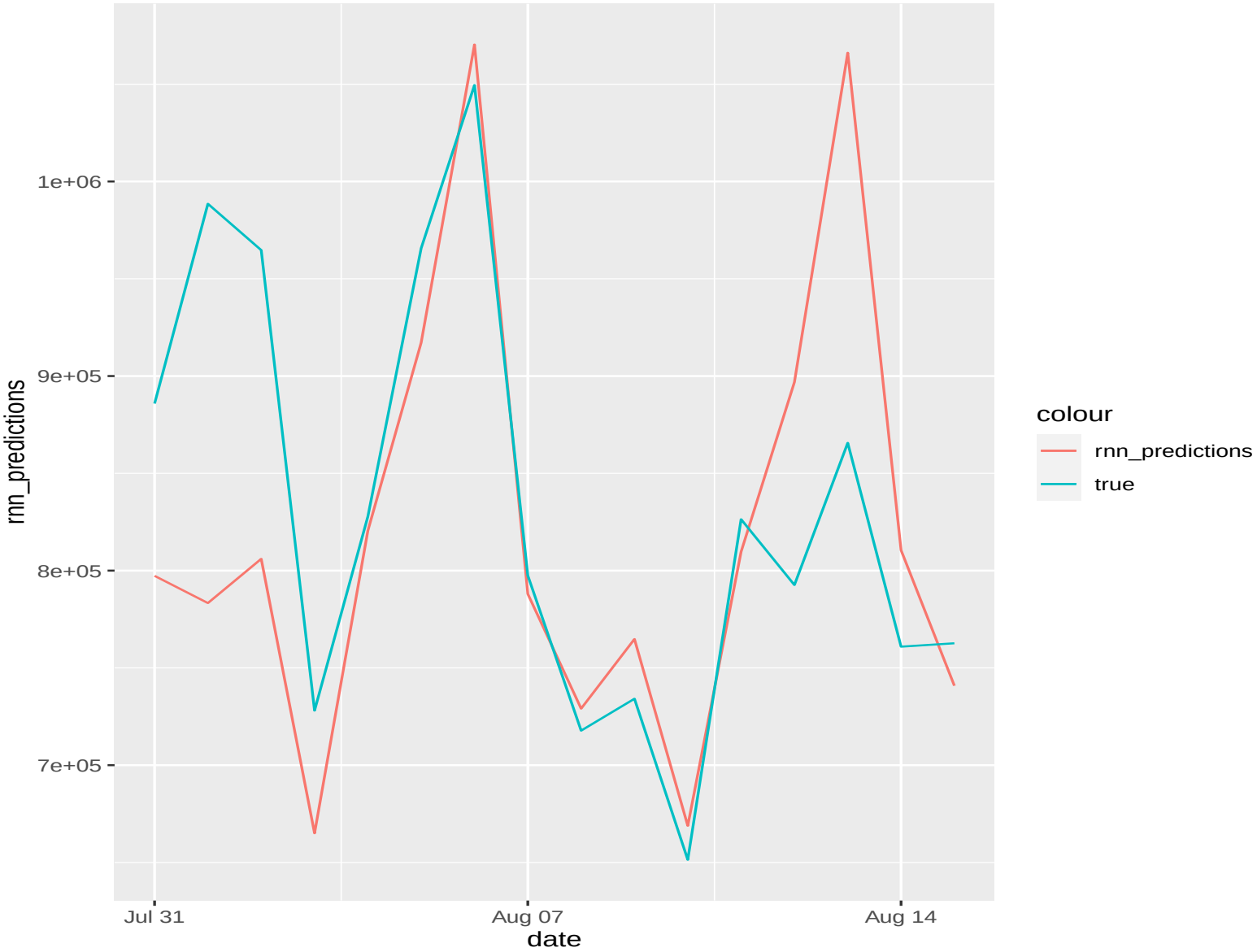
Machine Learning Methods in Time Series Analysis

After fitting the data with an ARIMA model, we are also eager to use machine learning methods to predict store sales. First, we normalize the sales data due to the nature of the variable using z-score normalization. Then, we separate the 1684 days in total into a training dataset with the first 1652 days of sales record and a testing dataset with the last 32 days of sales. Now we can apply various machine learning methods to train our data.

RNN

RNN stands for "recurrent neural network". In an RNN, prediction depends not only on features of the current time point, but also on the time points that came before the current time point. As a result, RNNs are most useful for modelling patterns in sequences of data where the next item depends on previous items in the sequence.

RNN


1e+06
9e+05
8e+05
7e+05
rnn_predictions
Jul 31
Aug 07
Aug 14
date
colour
rnn_predictions
true

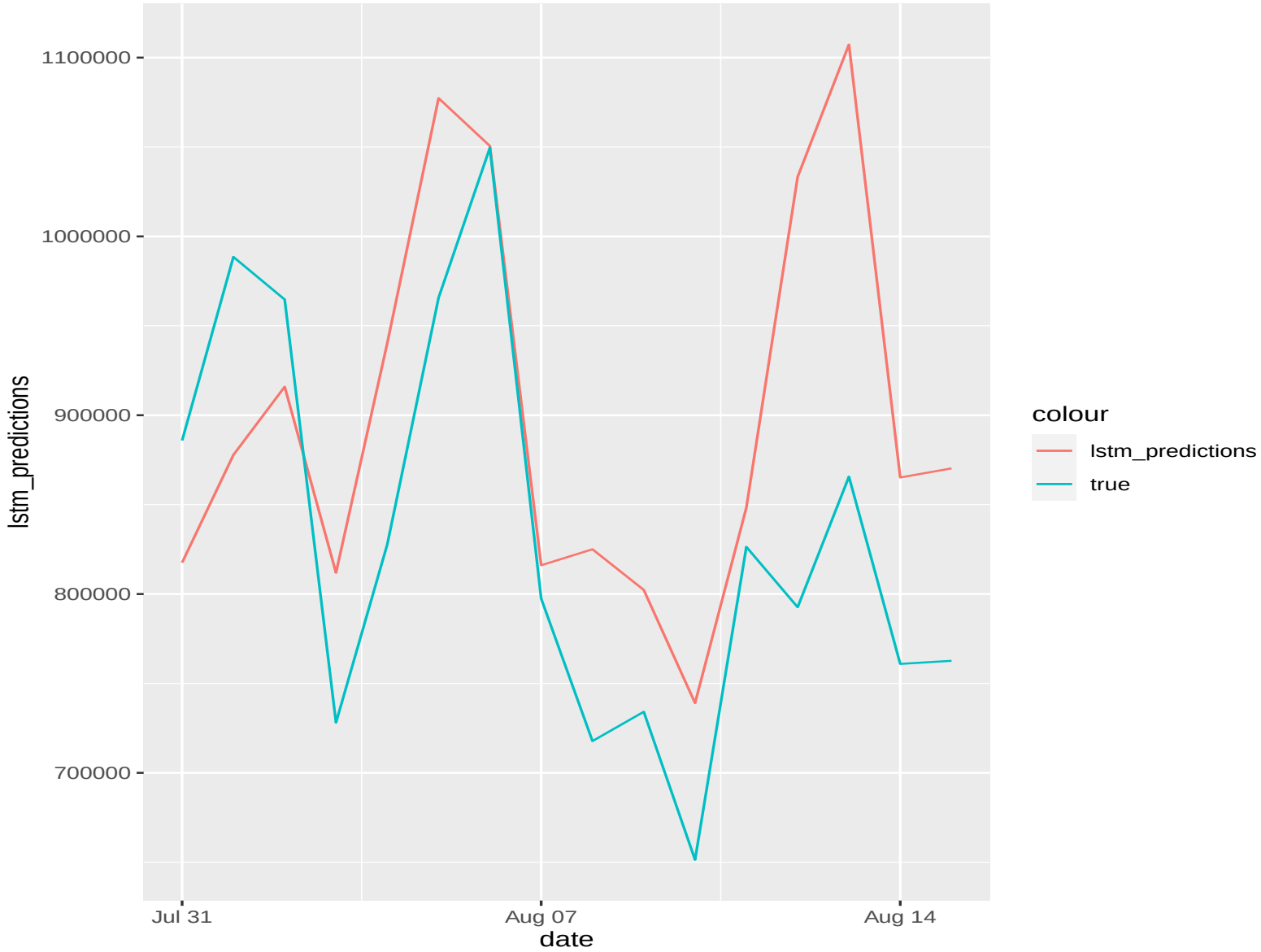
LSTM

Even though RNNs are quite powerful, they suffer from vanishing gradient problems which hinder them from using long-term information. For example, they are good for storing memory that is 3 to 4 instances of past iterations, but RNNs cannot provide good results for larger numbers of instances. Therefore, a better variation of RNNs is called the long short-term networks (LSTM).

LSTM

Long short-term memory units (or blocks) are building units for layers of a recurrent neural network (RNN). An RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate, and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals and hence the word "memory" in LSTM. LSTMs are useful in time series prediction tasks involving autocorrelation, because of their ability to maintain state and pattern recognition over the length of the series.

LSTM


1100000
1000000
900000
800000
700000
lstm_predictions
Jul 31
Aug 07
Aug 14
date
colour
lstm_predictions
true

GRU

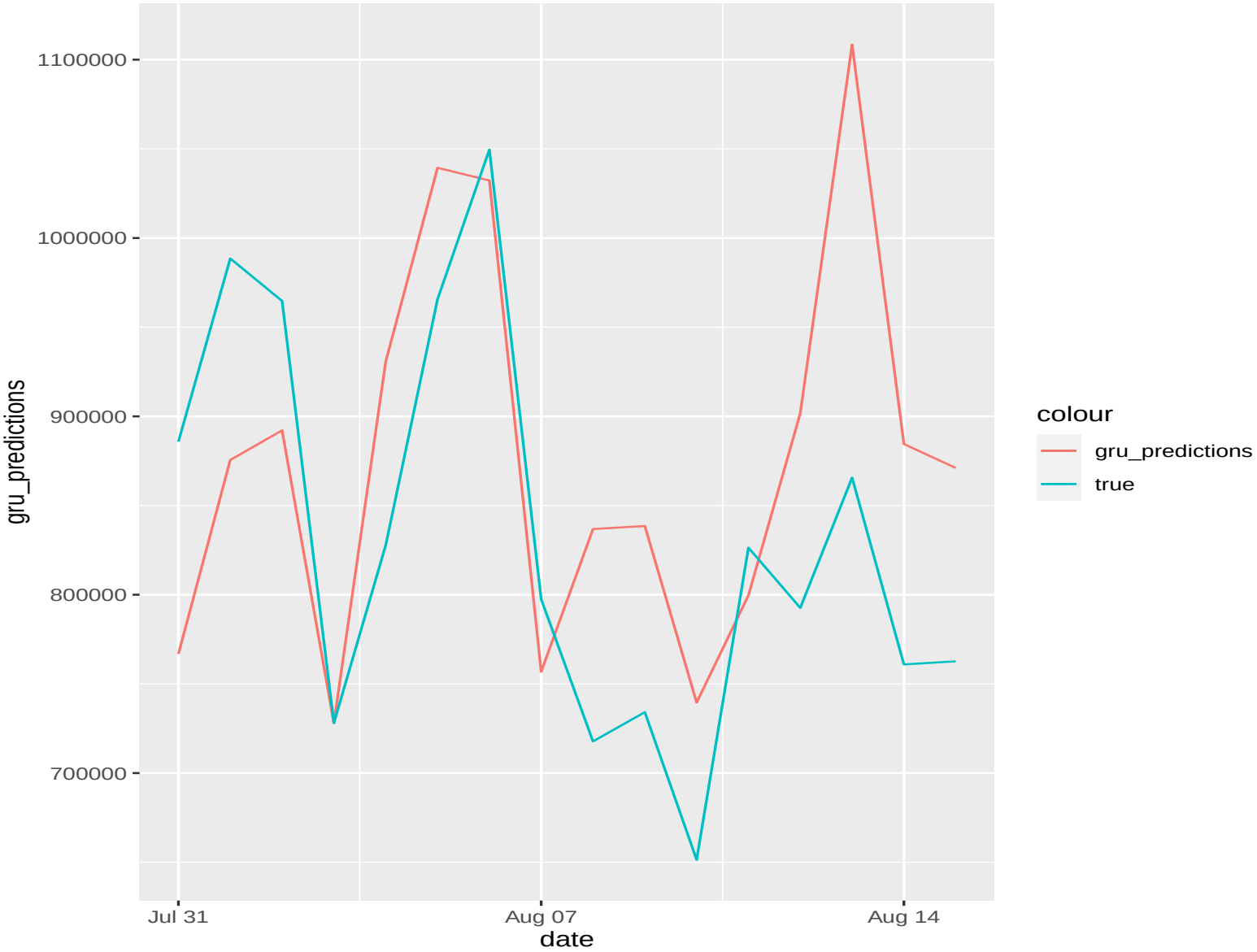
LSTM is not the only kind of unit that has taken the world of Deep Learning by a storm. We also have gated recurrent units (GRU). GRU is similar to the LSTM networks, and in particular GRU is a newer version of RNN. However, there are some differences between them:

1. GRU doesn't contain a cell state.
2. GRU uses its hidden states to transport information.
3. GRU contains only 2 gates, namely the reset and the update gate.
4. GRU is faster than LSTM.
5. GRU has fewer tensor operations that makes it faster.

GRU

Simply put, a GRU unit does not have to use a memory unit to control the flow of information like an LSTM unit. It can directly make use of all hidden states without any control. GRUs have fewer parameters and thus may train faster or need less data to generalize. But, with large datasets, LSTM with higher expressiveness may lead to better results.

GRU


1100000
1000000
900000
800000
700000
gru_predictions
Jul 31
Aug 07
Aug 14
date
colour
gru_predictions
true

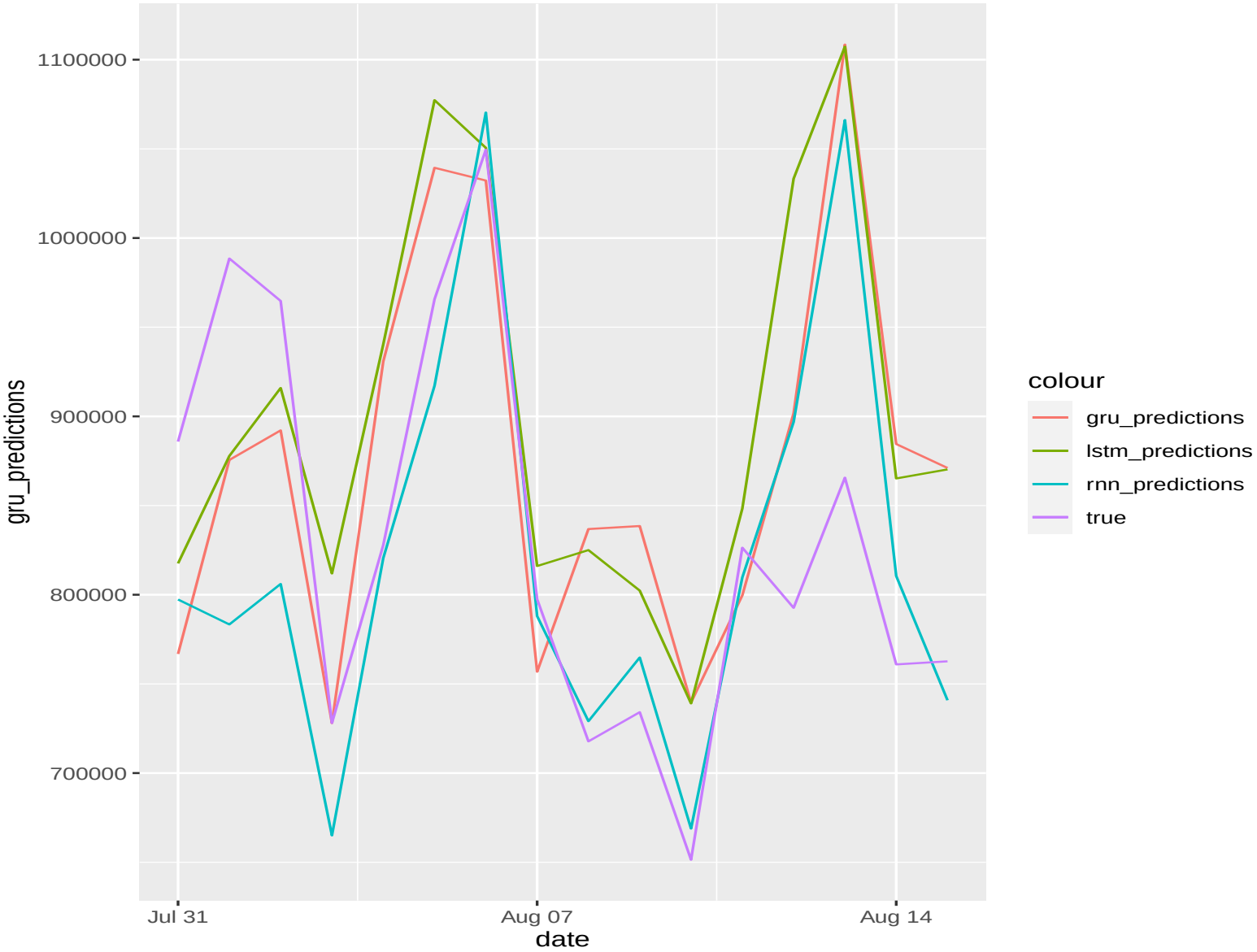
Model Performance Comparison

After a series of training, we come to the model performance comparison section. We would like to know which of the three machine learning models provide the best fit in the training data. Numerically, we use the root mean square error (RMSE) as our performance metric. The RMSEs are shown in the table below. The RNN model has the smallest RMSE. Next, we further analyze the fittings through a plot combining the three models.

Table: RMSE Comparison Table

Metric	**RNN**	**LSTM**	**GRU**
RMSE	89213.54	95040.73	101219.8

Model Performance Comparison



Model Prediction Comparison

Finally, our goal is to provide a prediction for the days of aggregated sales from August 16, 2017 to August 31, 2017. The prediction results are shown and compared in the next slide.

Model Prediction Comparison