Forecasting issues

Forecast Padawan 2

November 17, 2016

The goal of this experiment is to design the best model to forcaste the number of issue in the per day in the comming two weeks. We think that this could help Open Source organisation to manage there human ressources.

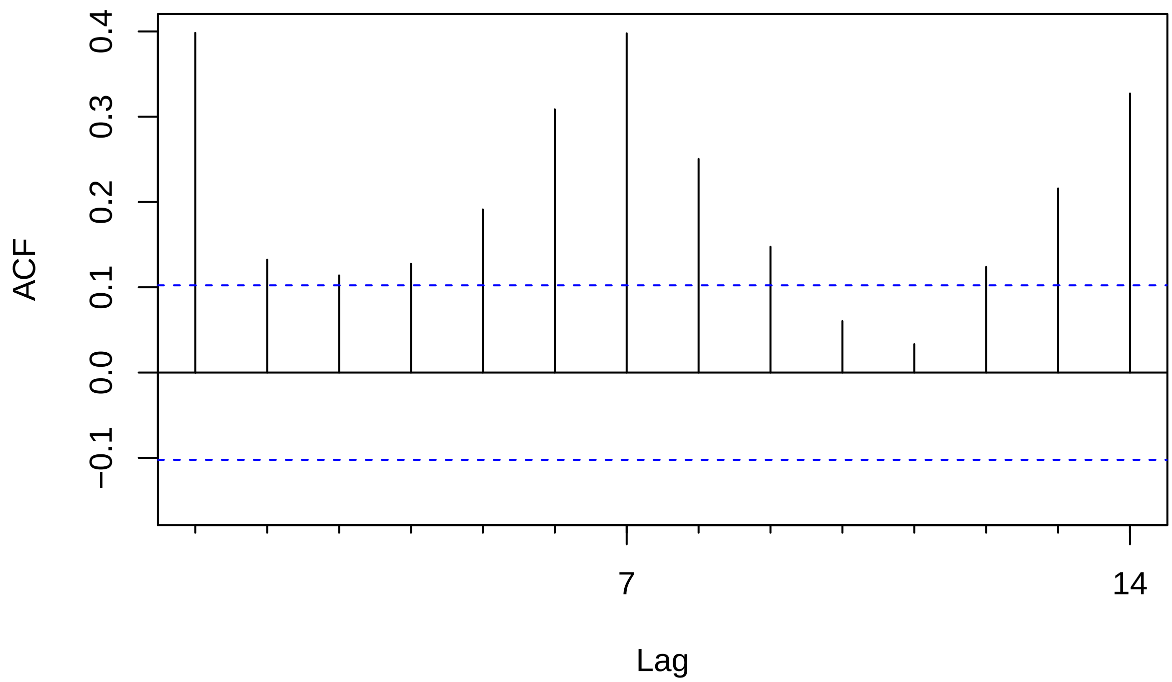
Load the data

```
#install.packages('forecast')

library('forecast')
library(knitr)
#load the data frame
issues.csv <- read.csv("issues/julialang_julia.csv")
commits.csv <- read.csv("commits/julialang_julia.csv")

issues.csv$date = as.POSIXlt(as.Date(issues.csv$date,format='%m/%d/%Y'))
commits.csv$date = as.POSIXlt(as.Date(commits.csv$date,format='%m/%d/%Y'))
```

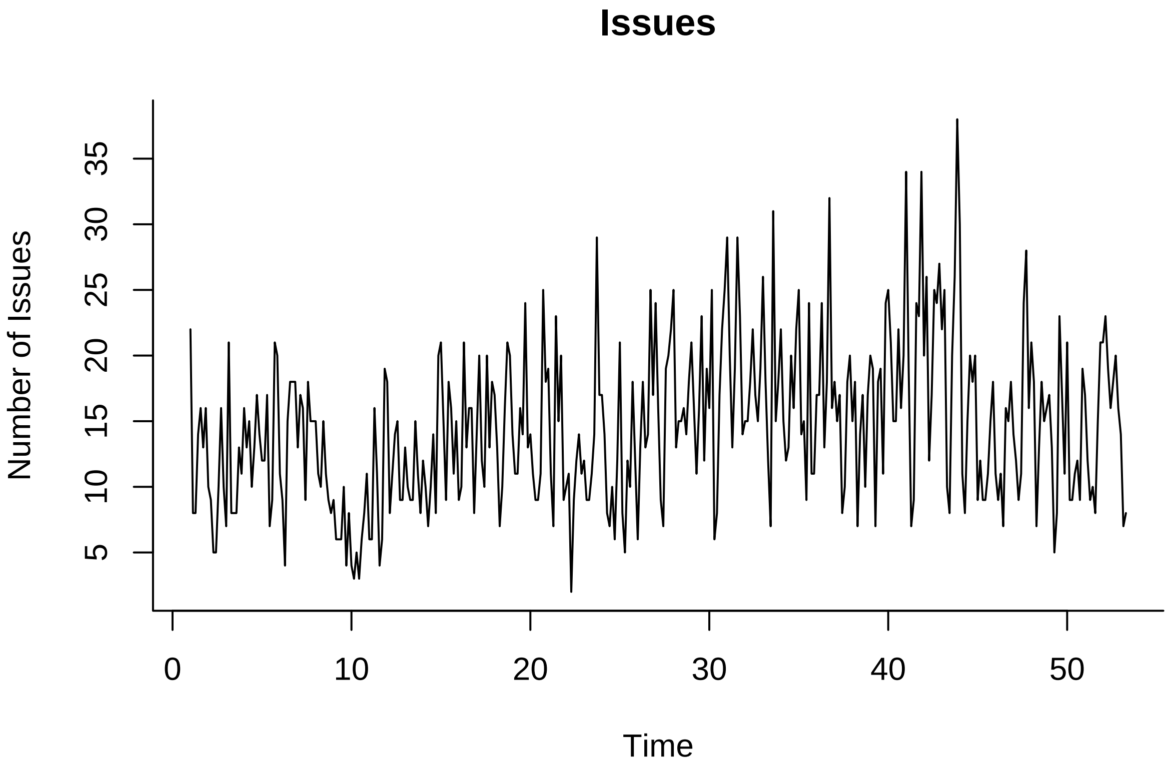
keep the last 12 months

```
to_date <- issues.csv$date[length(issues.csv$date)]
from_date <- to_date
from_date$year <- from_date$year - 1

issues.csv <- subset(issues.csv, date <= to_date & date >= from_date)
commits.csv <- subset(commits.csv, date <= to_date & date >= from_date)
```

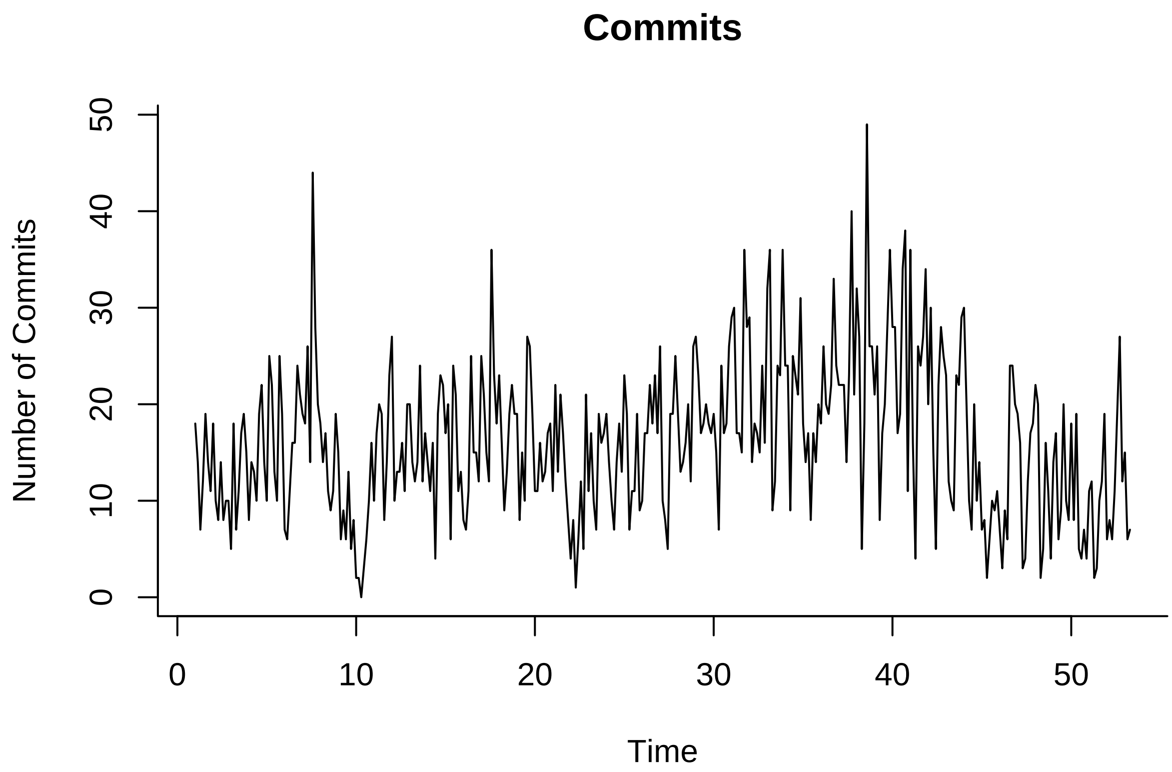
```
#loading issues and commits into a ts object
issues.ts <- ts(issues.csv$number_of_issues, frequency = 7)

Acf(issues.ts, lag.max = 14, main = "")
```




```
commits.ts <- ts(commits.csv$number_of_commits, frequency = 7)
plot(issues.ts, main = 'Issues', bty = 'l', ylab = 'Number of Issues')
```




```r
plot(commits.ts, main = 'Commits', bty = 'l', ylab = 'Number of Commits')
```




```
time <- time(issues.ts)

n.sample <- 14
n.valid <- 21


separate.train.test <- function(timeserie, n.valid) {
  time <- time(timeserie)
  n.train <- length(timeserie) - n.valid
  results = list()
  results$train.ts <- window(timeserie, start=time[1], end=time[n.train])
  results$valid.ts <- window(timeserie, start=time[n.train+1], end=time[n.train+n.valid])
  return(results)
}

all.issues <- sapply(0:(n.sample - 1), function(i) return(separate.train.test(window(issues.ts,start=ti

issues  <- separate.train.test(issues.ts, n.valid)
commits  <- separate.train.test(commits.ts, n.valid)
```

```
mean.all.accuracy <- function(all.forecast) {
  Reduce("+", all.forecast['summary',])/length(all.forecast['summary',])
}

plot.all.residuals <- function(all.forecast) {
```

```
  plot(1, type="l", main="Residuals", xlim=c(48.5, 53.3), ylim=c(-30, 30), xlab = 'Week', ylab = 'Errors
  sapply(1:n.sample, function(i) lines(all.forecast['residual',i]$residual))
  return(NULL)
}

plot.all.pred <- function(all.forecast) {
  plot(issues.ts, main="Prediction",xlim=c(30, 53.3), xlab = 'Week', ylab = 'Number of Issues')
  if (class(all.forecast['pred',1]$pred) == "forecast") {
    sapply(1:n.sample, function(i) lines(all.forecast['pred',i]$pred$mean, col=rgb(0, 0, 1, 0.5)))
  } else {
    sapply(1:n.sample, function(i) lines(all.forecast['pred',i]$pred, col=rgb(0, 0, 1, 0.5)))
  }
  return(NULL)
}

hist.all.residuals <- function(all.forecast) {
  residuals <- sapply(1:n.sample, function(i) as.numeric(all.forecast['residual',i]$residual))
  boxplot(residuals)
  hist(residuals)
  quantile(residuals,c(0.975,0.95,0.05,0.025))
}
```

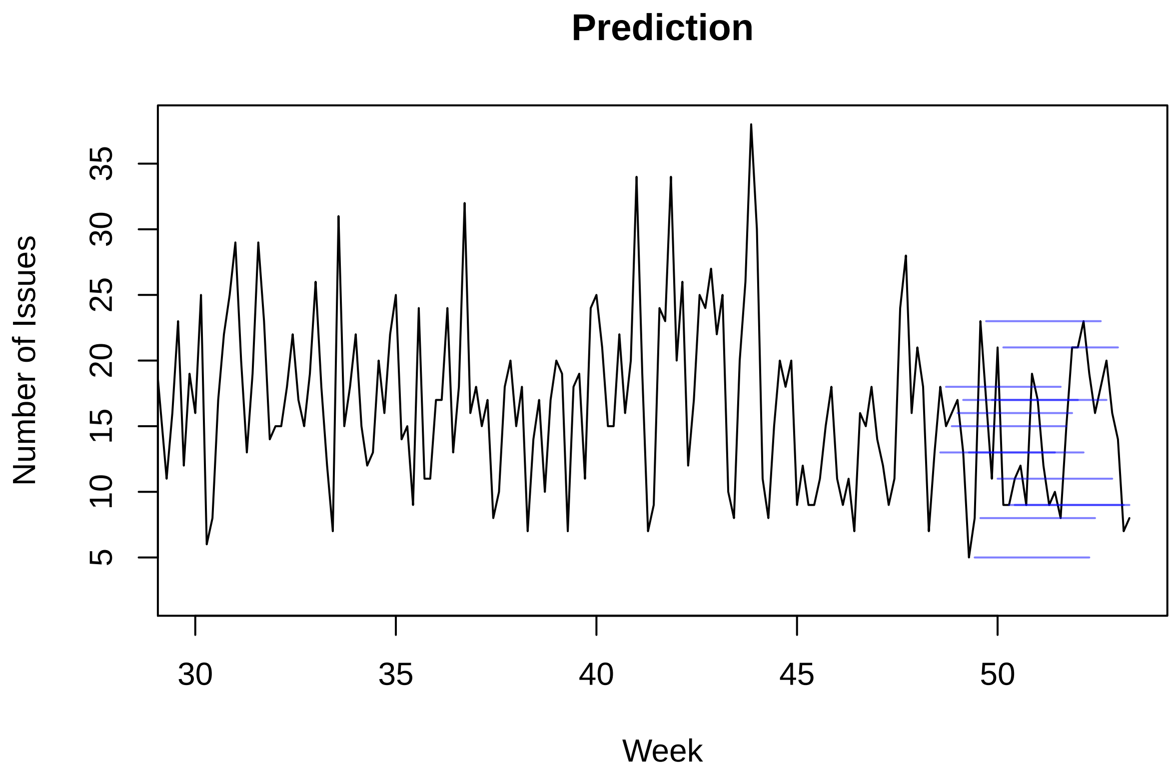
Naive Forecast

Naive

```
naive.forecast <- function(sample) {
  results = list()
  results$valid <- sample$valid.ts
  results$pred <- naive(sample$train.ts, h=n.valid)
  results$residual <- sample$valid.ts - results$pred$mean
  results$summary <- accuracy(results$pred, sample$valid.ts)

  return(results)
}

all.naive.forecast <- sapply(1:n.sample, function(i) return(naive.forecast(all.issues[,i])))

kable(mean.all.accuracy(all.naive.forecast))
```

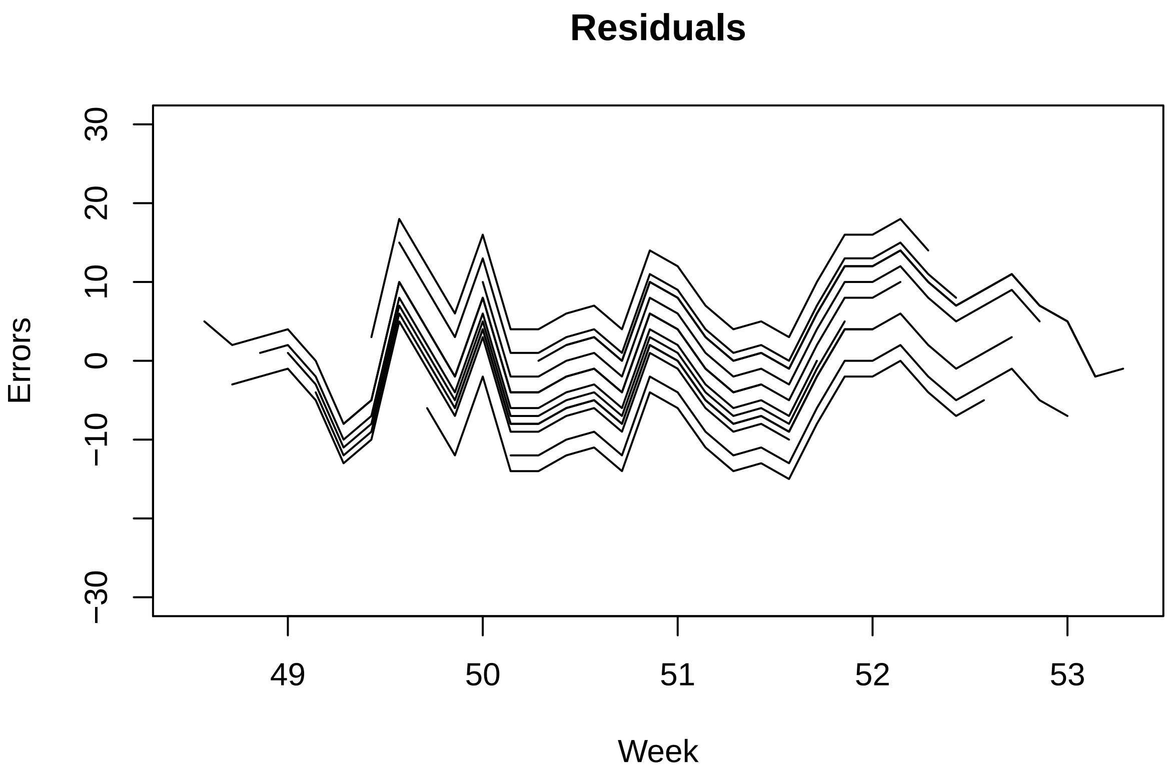
	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	-0.0238173	6.690828	5.236218	-12.86230	41.78262	1.002643	-0.2787088	NA
Test set	0.1394558	6.978572	5.914966	-14.08305	47.98660	1.132777	0.3294768	1.229262

```
plot.all.pred(all.naive.forecast)
```



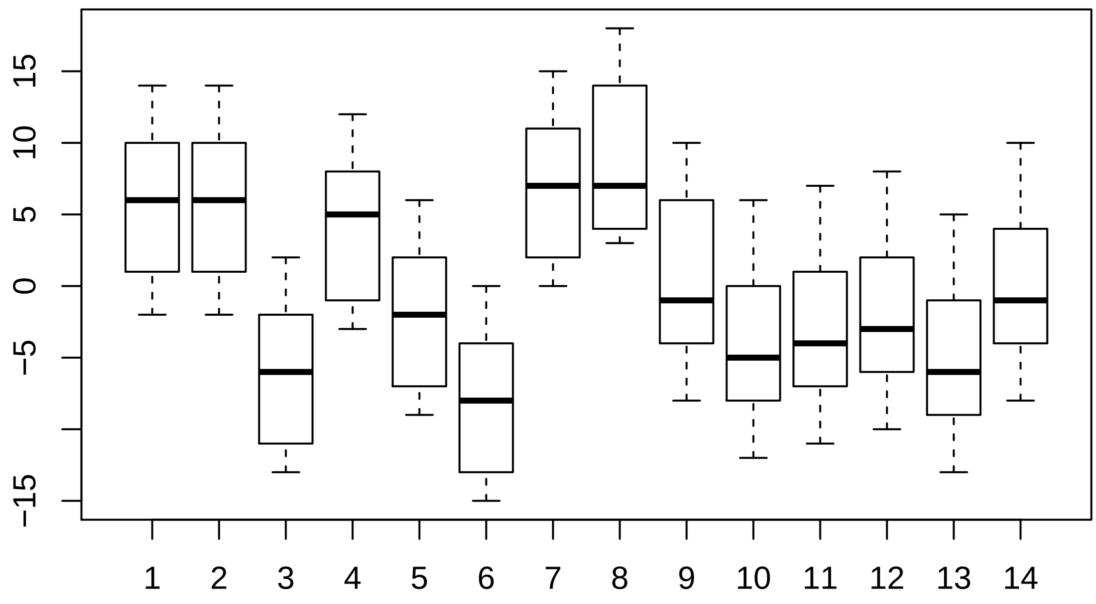

```
## NULL
```

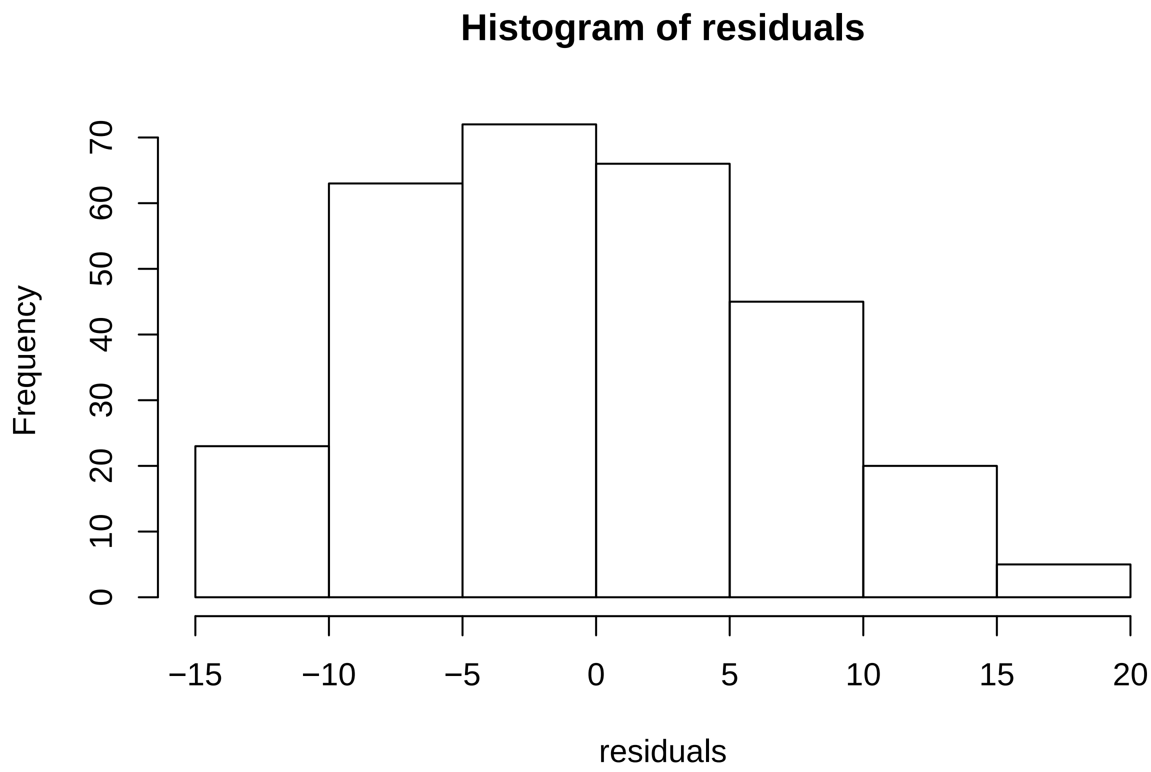
```
plot.all.residuals(all.naive.forecast)
```




```
## NULL
```

```r
hist.all.residuals(all.naive.forecast)
```


15
10
5
0
−5
−15
1
2
3
4
5
6
7
8
9
10
11
12
13
14



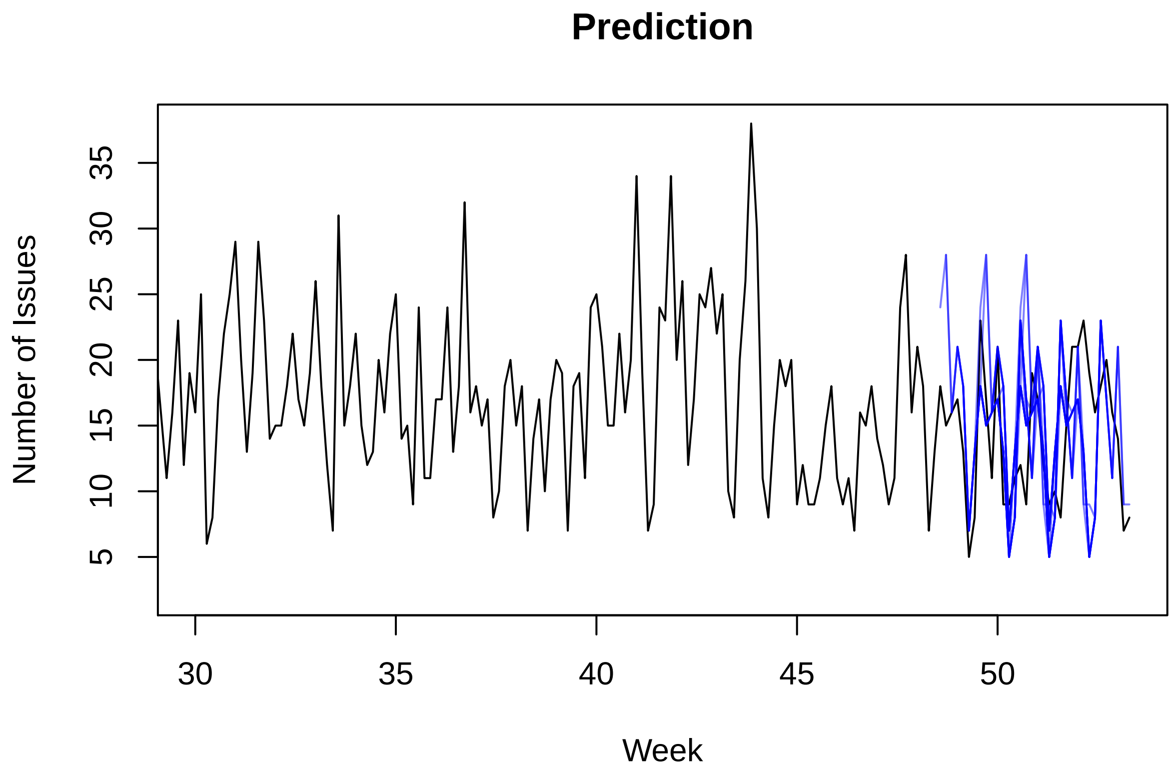

```
##   97.5%     95%      5%    2.5%
##  14.000  12.000 -11.350 -12.675
```

Seasonal Naive

```
snaive.forecast <- function(sample) {
  results = list()
  results$valid <- sample$valid.ts
  results$pred <- snaive(sample$train.ts, h=n.valid)
  results$residual <- sample$valid.ts - results$pred$mean
  results$summary <- accuracy(results$pred, sample$valid.ts)

  return(results)
}

all.snaive.forecast <- sapply(1:n.sample, function(i) return(snaive.forecast(all.issues[,i])))

kable(mean.all.accuracy(all.snaive.forecast))
```

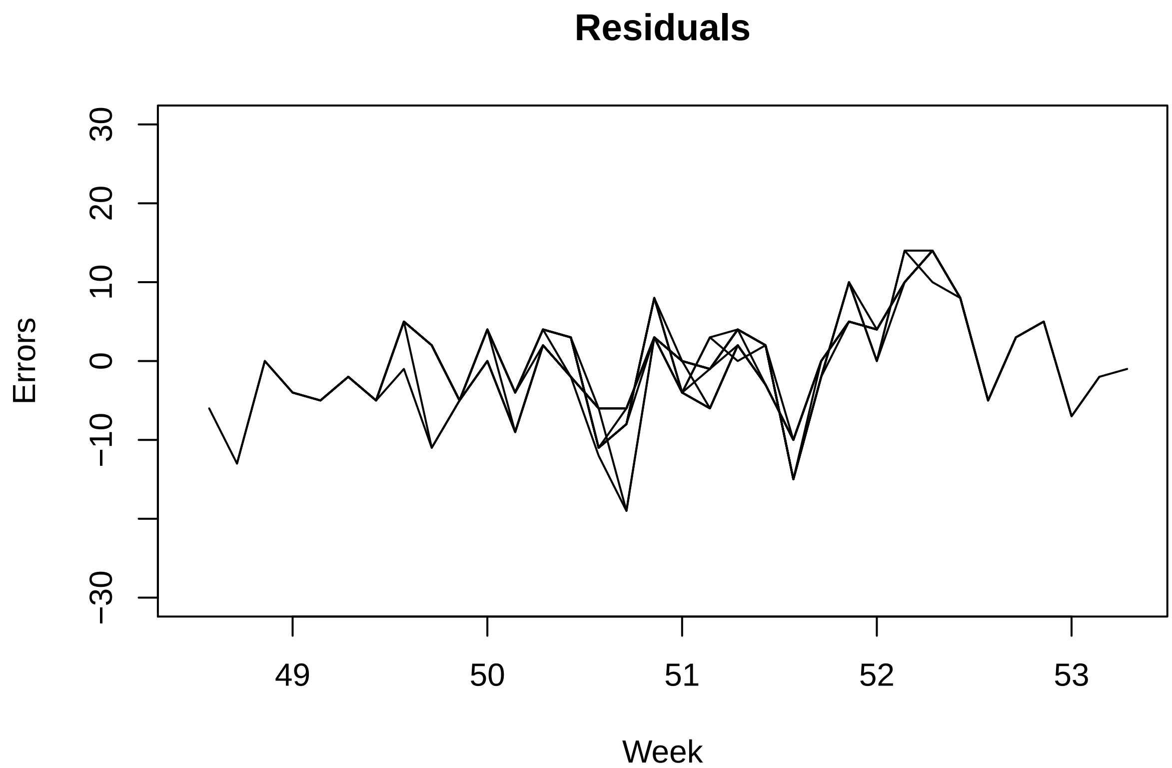
	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	0.0170509	6.586269	5.222487	-12.85417	42.34831	1.0000000	0.1758323	NA
Test set	-0.5850340	6.251959	5.074830	-14.17575	41.37955	0.9720907	0.0345263	1.056669

```
plot.all.pred(all.snaive.forecast)
```



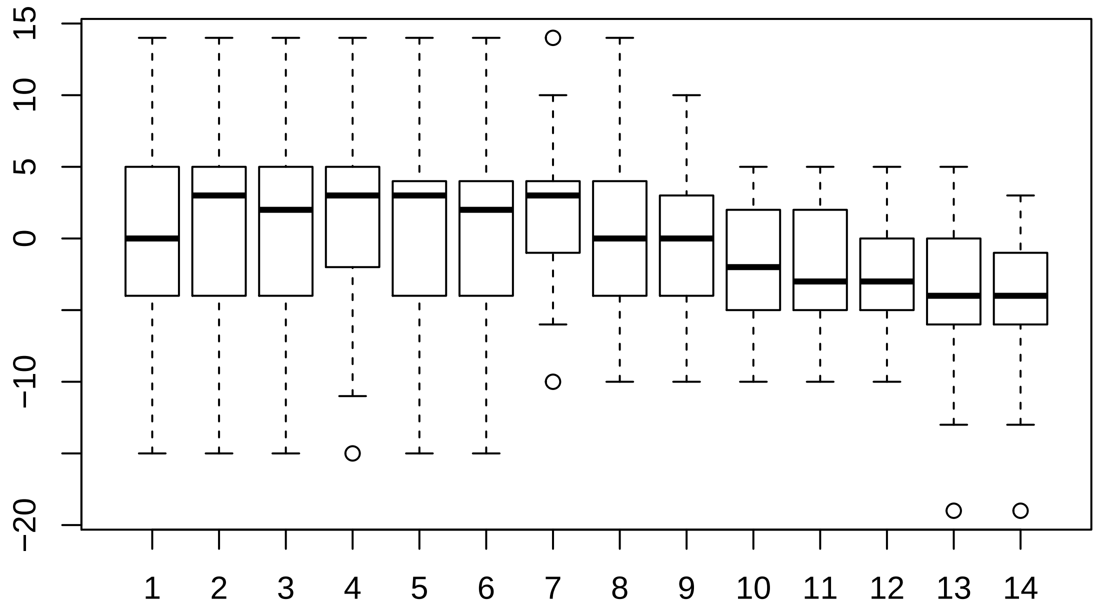

```
## NULL
```

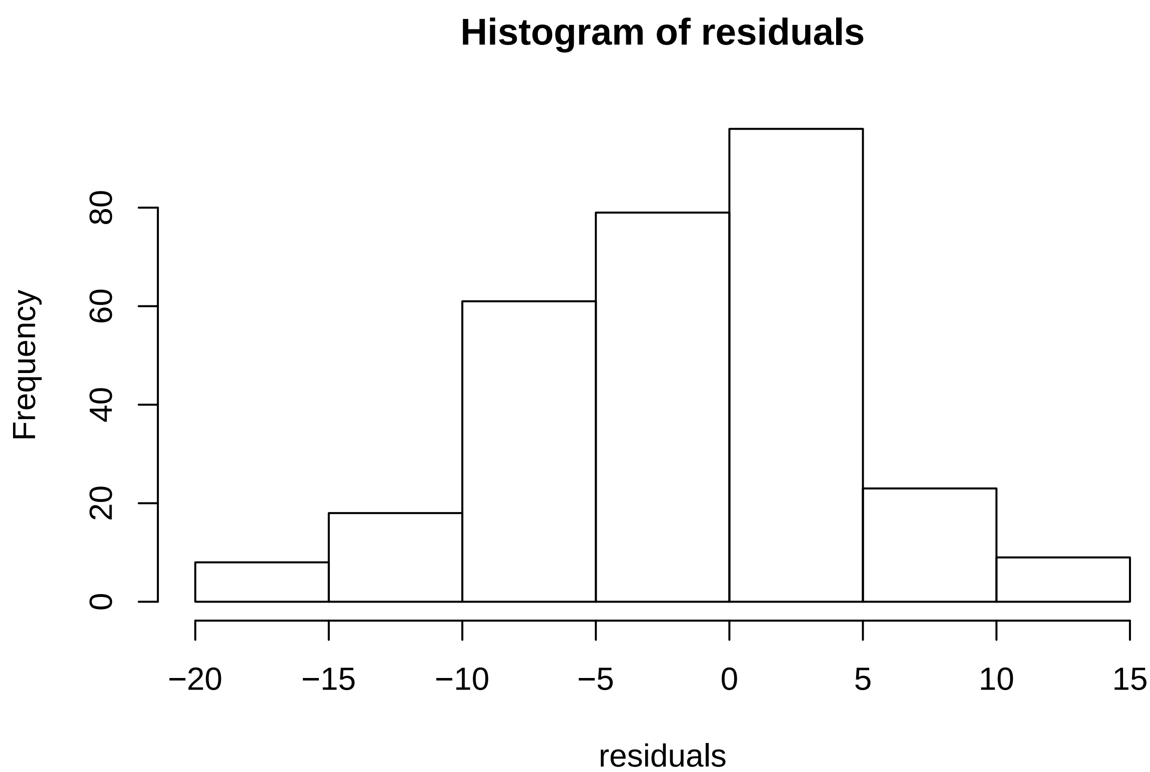
```
plot.all.residuals(all.snaive.forecast)
```




```
## NULL
```

```
hist.all.residuals(all.snaive.forecast)
```


15
10
5
0
−10
−20
1
2
3
4
5
6
7
8
9
10
11
12
13
14




```
##  97.5%    95%     5%   2.5%
##  14.00  10.00 -11.00 -14.35
```

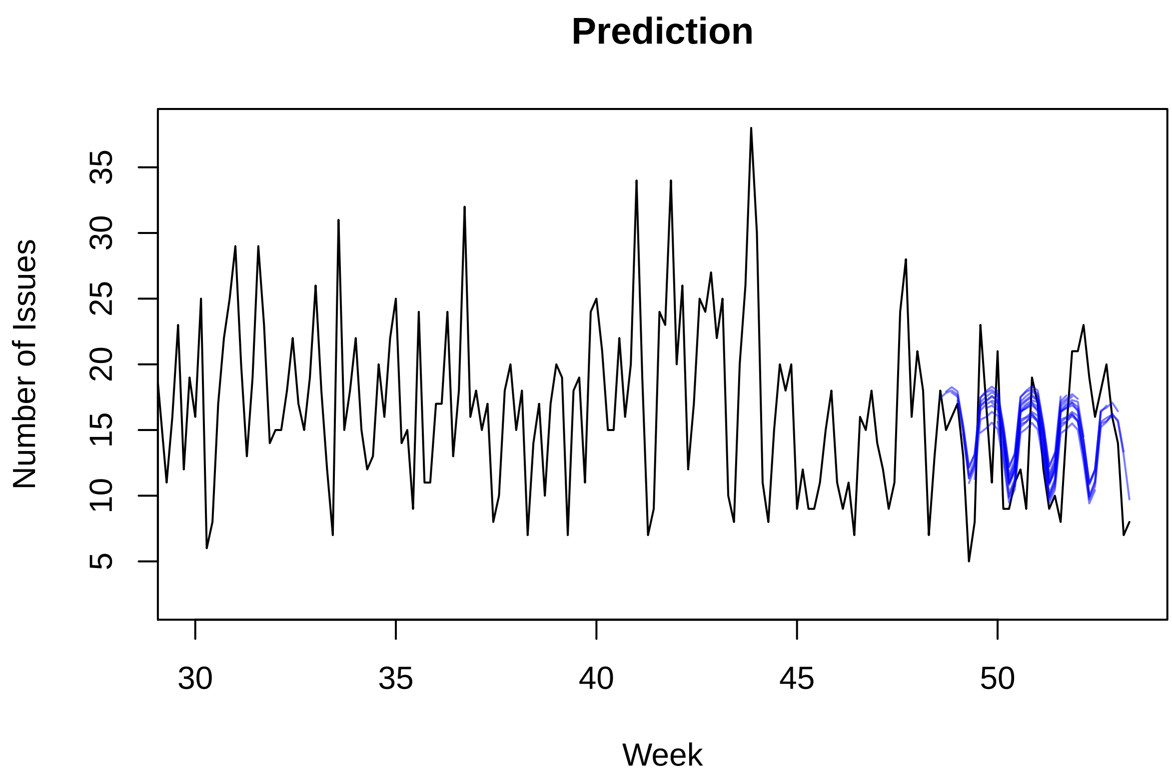
Smoothing

Holt Winter

```
hw.forecast <- function(sample) {
  results = list()
  results$valid <- sample$valid.ts
  results$model <- ets(sample$train.ts, model = "ZAA")
  results$pred <- forecast(results$model, h=n.valid)
  results$residual <- sample$valid.ts - results$pred$mean
  results$summary <- accuracy(results$pred, sample$valid.ts)

  return(results)
}

all.hw.forecast <- sapply(1:n.sample, function(i) return(hw.forecast(all.issues[,i])))

kable(mean.all.accuracy(all.hw.forecast))
```

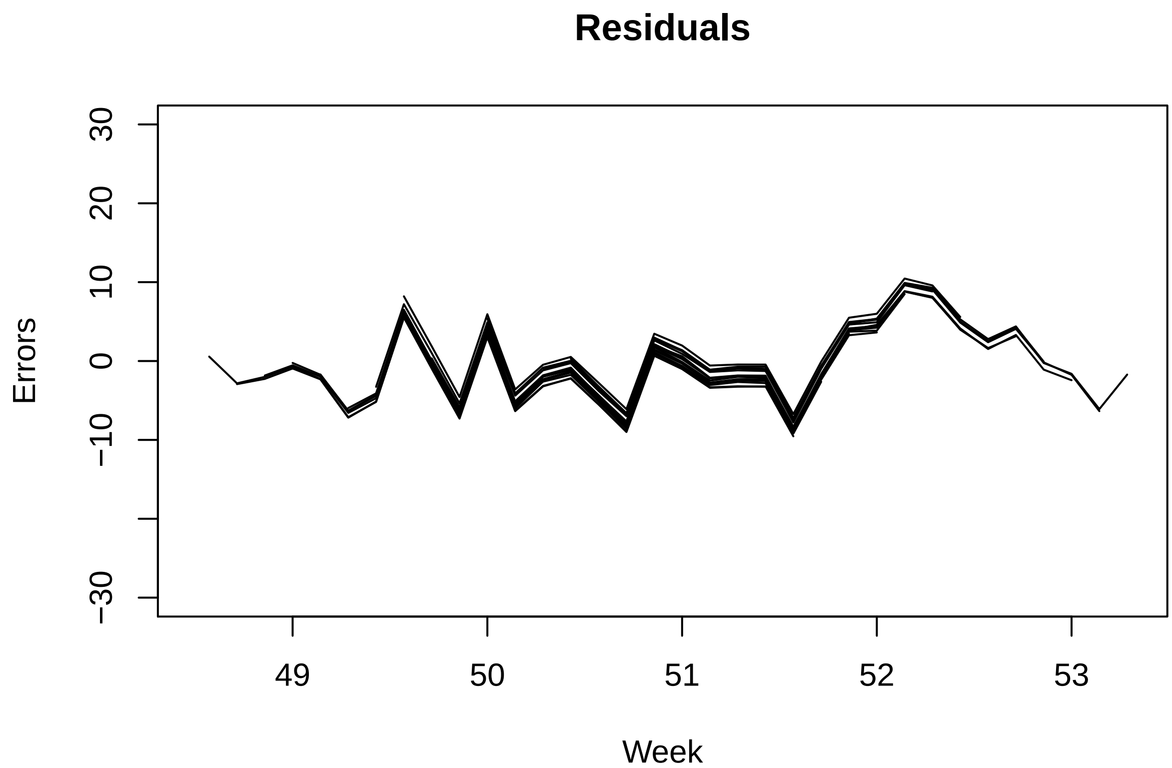
	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	-0.0271828	4.986111	3.882105	-13.30617	32.63851	0.7433471	0.0909490	NA
Test set	-0.7128420	4.629126	3.784447	-17.11348	32.23754	0.7246322	0.2126386	0.7492468

```
plot.all.pred(all.hw.forecast)
```



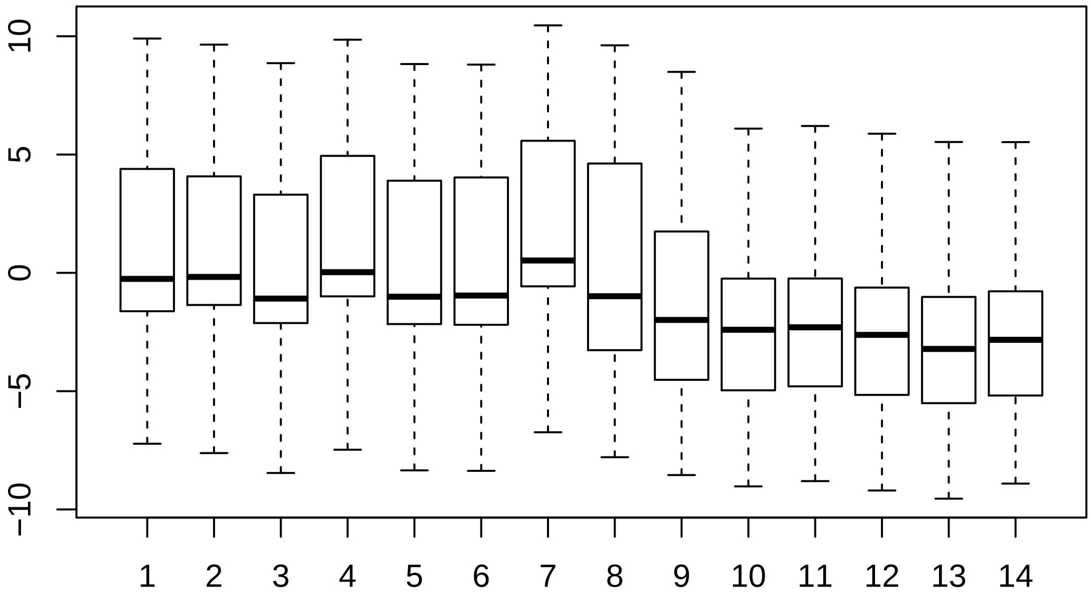

```
## NULL
```

```
plot.all.residuals(all.hw.forecast)
```

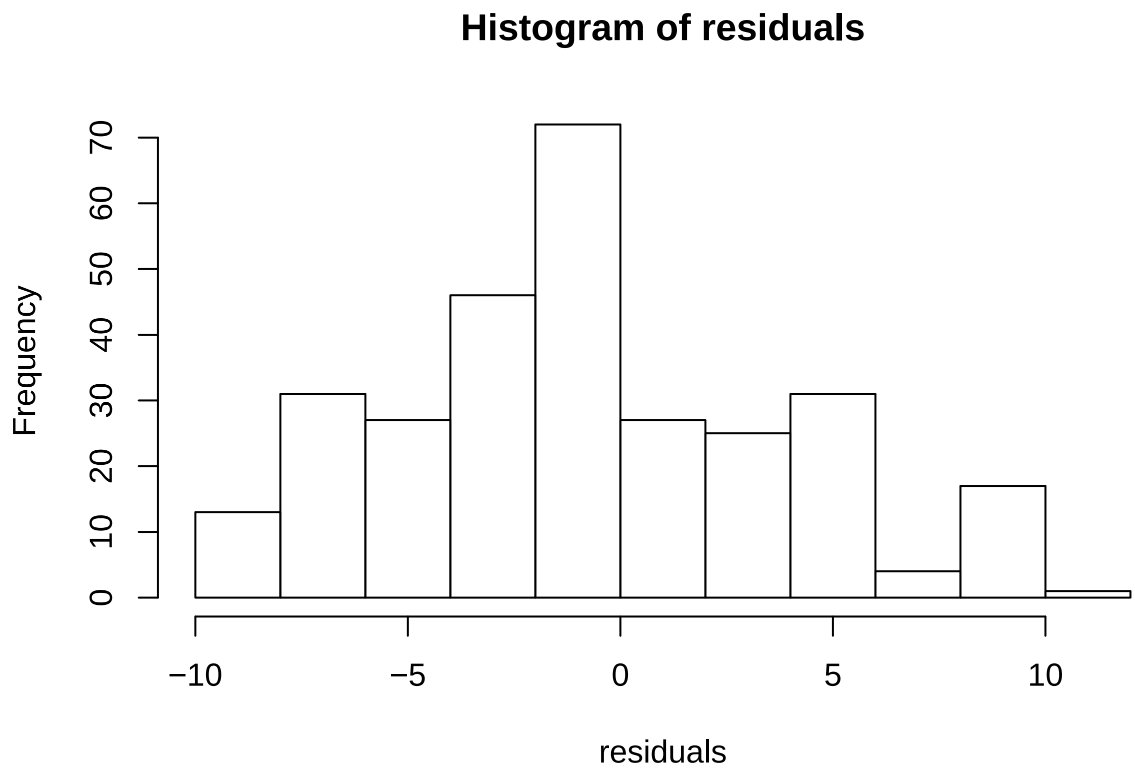



```
## NULL
```

```r
hist.all.residuals(all.hw.forecast)
```



10
5
0
−5
−10
1
2
3
4
5
6
7
8
9
10
11
12
13
14




```
##     97.5%       95%        5%      2.5%
##  9.093400  8.171099 -7.805061 -8.517816
```

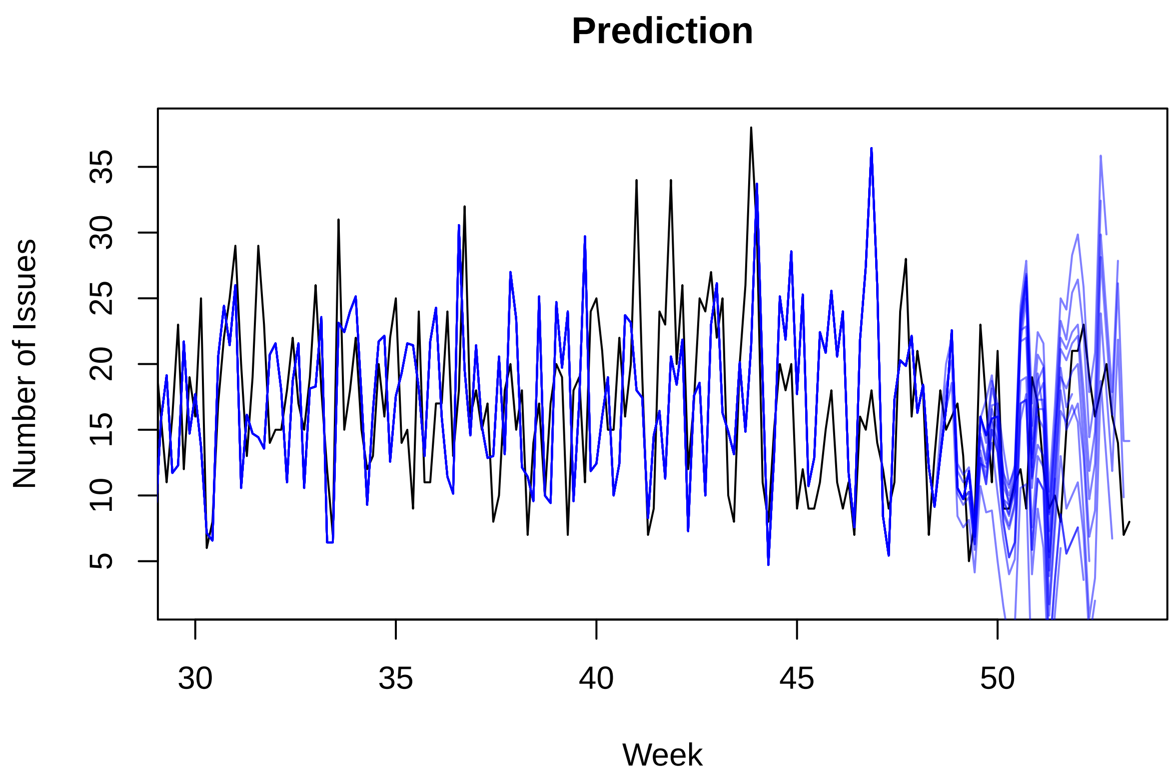
Double differencing

```
ma.dd.forecast <- function(sample) {
  train.issues.d1 <- diff(sample$train.ts, lag = 1)
  train.issues.d1.d7 <- diff(train.issues.d1, lag = 7)

  ma.trailing <- rollmean(train.issues.d1.d7, k = 7, align = "right")
  last.ma <- tail(ma.trailing, 1)
  ma.trailing.pred <- ts(c(ma.trailing, rep(last.ma, n.valid)), start=c(3, 1), frequency = 7)

  ma.dd.pred.d1 <- train.issues.d1
  ma.dd.pred <- sample$train.ts

  for(i in 1:(n.valid/7)) {
    ma.dd.pred.d1 <- ma.trailing.pred + lag(ma.dd.pred.d1,k = -7)
    ma.dd.pred <- ma.dd.pred.d1 + lag(ma.dd.pred,k = -8)
  }

  results = list()
  results$valid <- sample$valid.ts
```

```
  results$pred <- ma.dd.pred
  results$residual <- sample$valid.ts - results$pred
  results$summary <- accuracy(results$pred, sample$valid.ts)

  return(results)
}

all.ma.dd.forecast <- sapply(1:n.sample, function(i) return(ma.dd.forecast(all.issues[,i])))

kable(mean.all.accuracy(all.ma.dd.forecast))
```

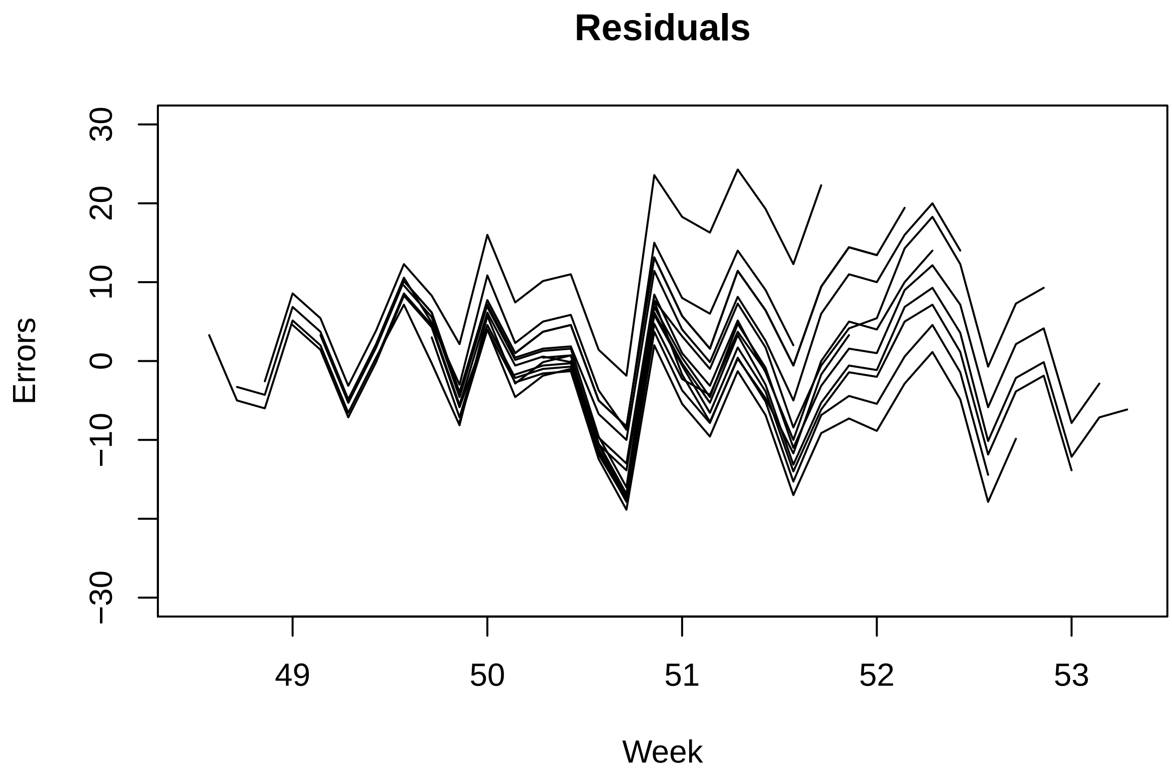
	ME	RMSE	MAE	MPE	MAPE	ACF1	Theil's U
Test set	0.7784257	8.033107	6.441205	-3.438831	50.79643	0.1573949	1.378406

```
plot.all.pred(all.ma.dd.forecast)
```



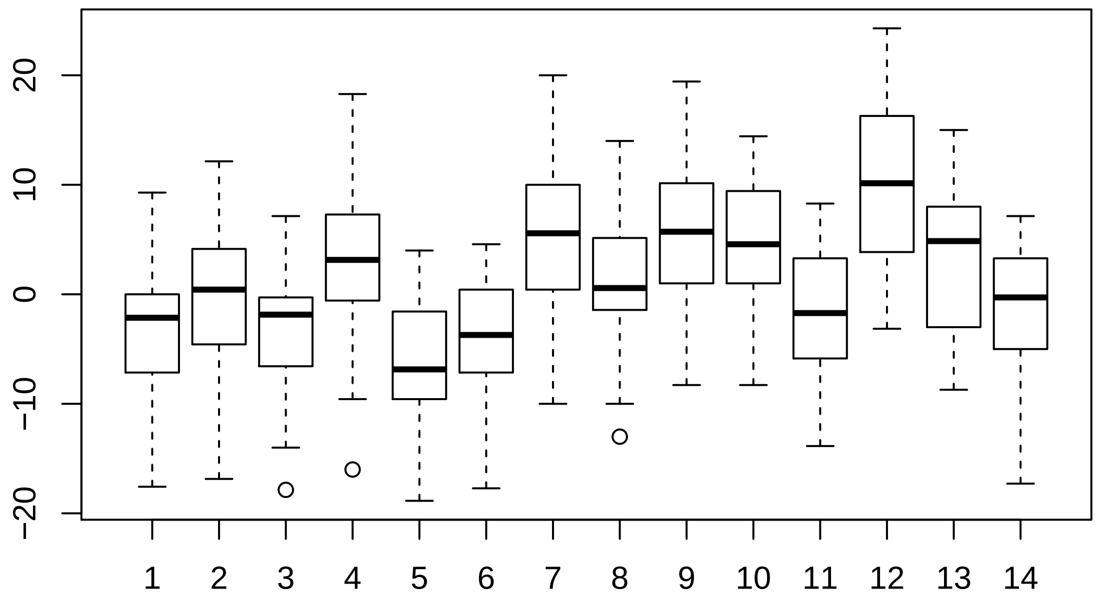

```
## NULL
```

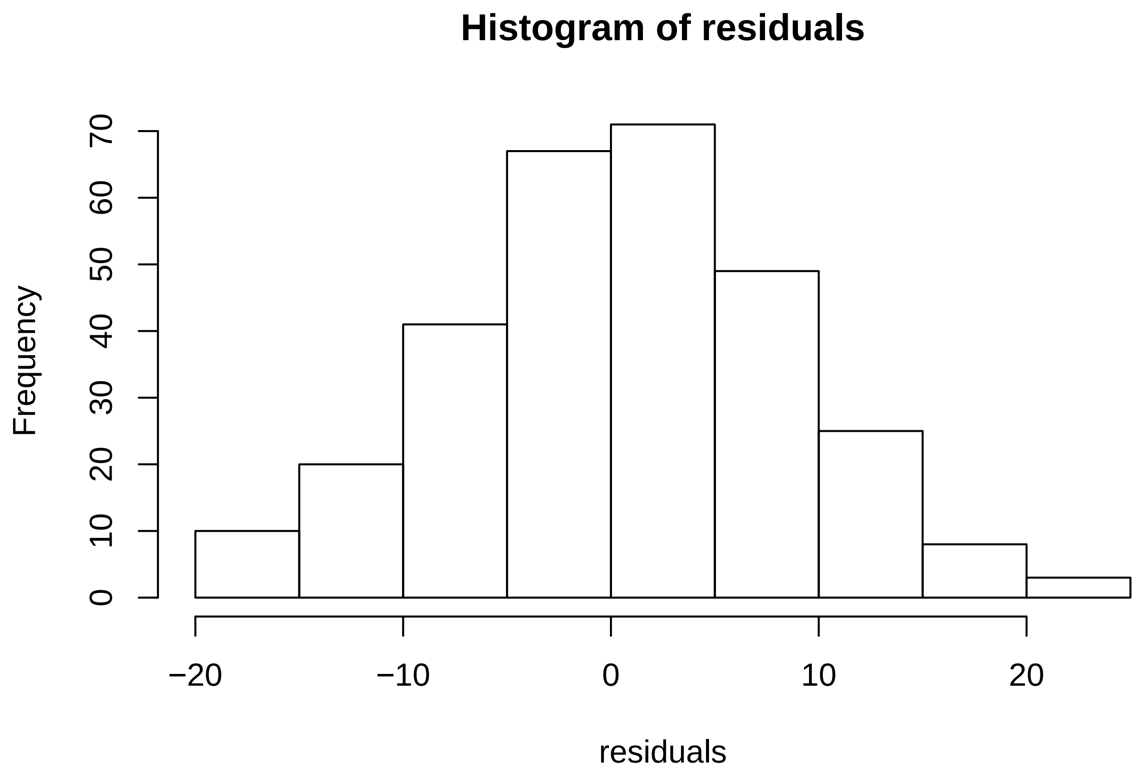
```
plot.all.residuals(all.ma.dd.forecast)
```




```
## NULL
```

```r
hist.all.residuals(all.ma.dd.forecast)
```


20
10
0
−10
−20
1
2
3
4
5
6
7
8
9
10
11
12
13
14




```
##      97.5%        95%         5%       2.5%
##   17.63571   14.10000 -13.05000 -16.57857
```

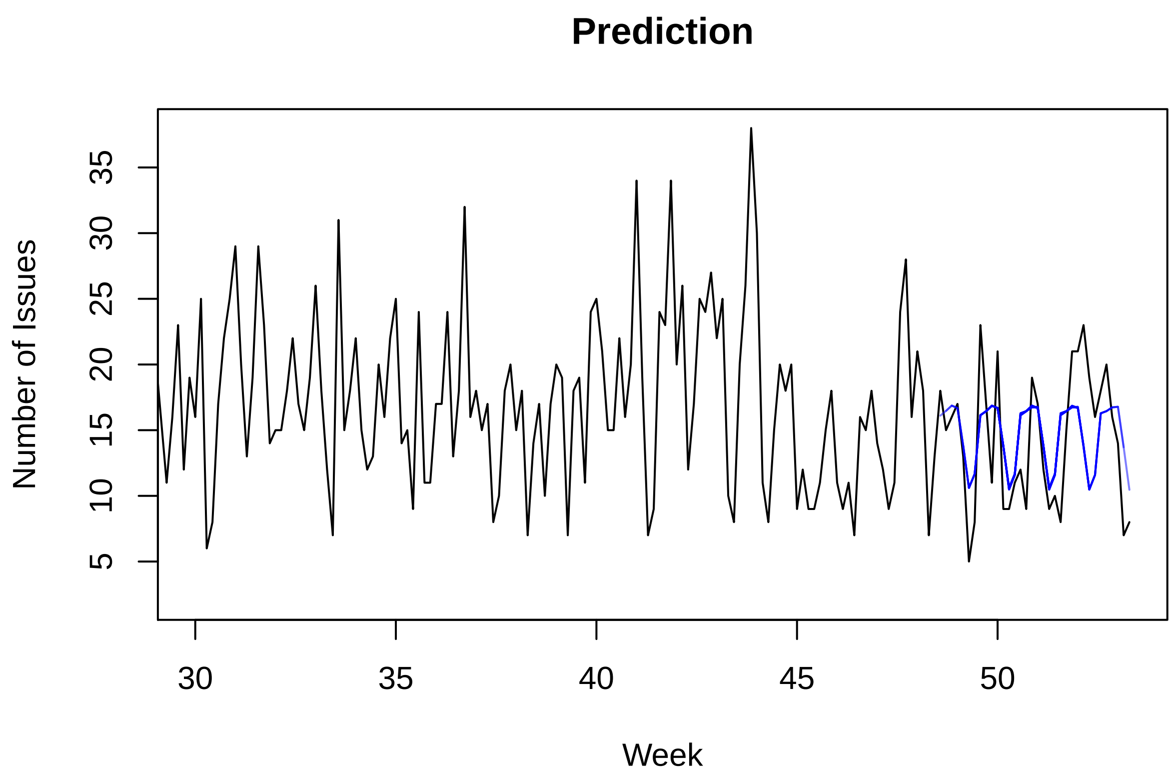
Regression

Linear additive regression

```
regr.add.forecast <- function(sample) {
  results = list()
  results$valid <- sample$valid.ts
  results$model <- tslm(sample$train.ts ~ season)
  results$pred <- forecast(results$model, h=n.valid)
  results$residual <- sample$valid.ts - results$pred$mean
  results$summary <- accuracy(results$pred, sample$valid.ts)

  return(results)
}

all.regr.add.forecast <- sapply(1:n.sample, function(i) return(regr.add.forecast(all.issues[,i])))

kable(mean.all.accuracy(all.regr.add.forecast))
```

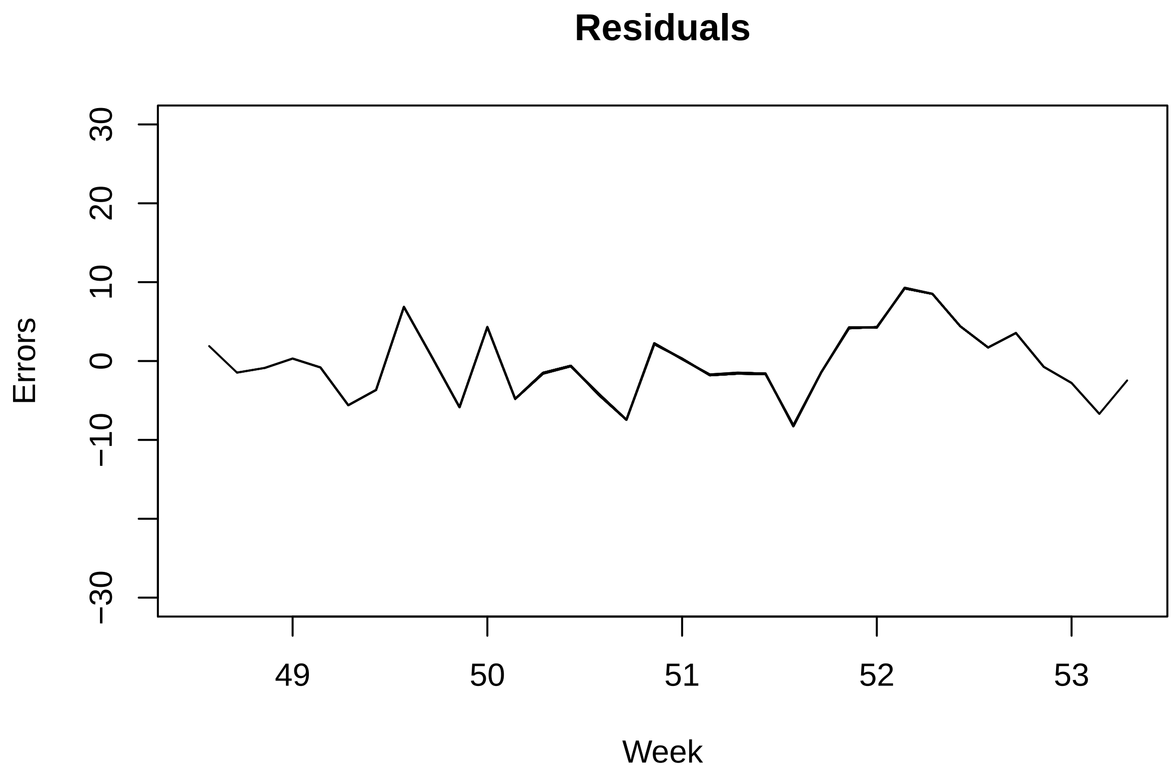
	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	0.0000000	5.588445	4.398678	-19.19098	39.53468	0.8422578	0.3783599	NA
Test set	-0.5230361	4.388761	3.542433	-15.07922	29.89041	0.6784011	0.2107481	0.7257861

```
plot.all.pred(all.regr.add.forecast)
```



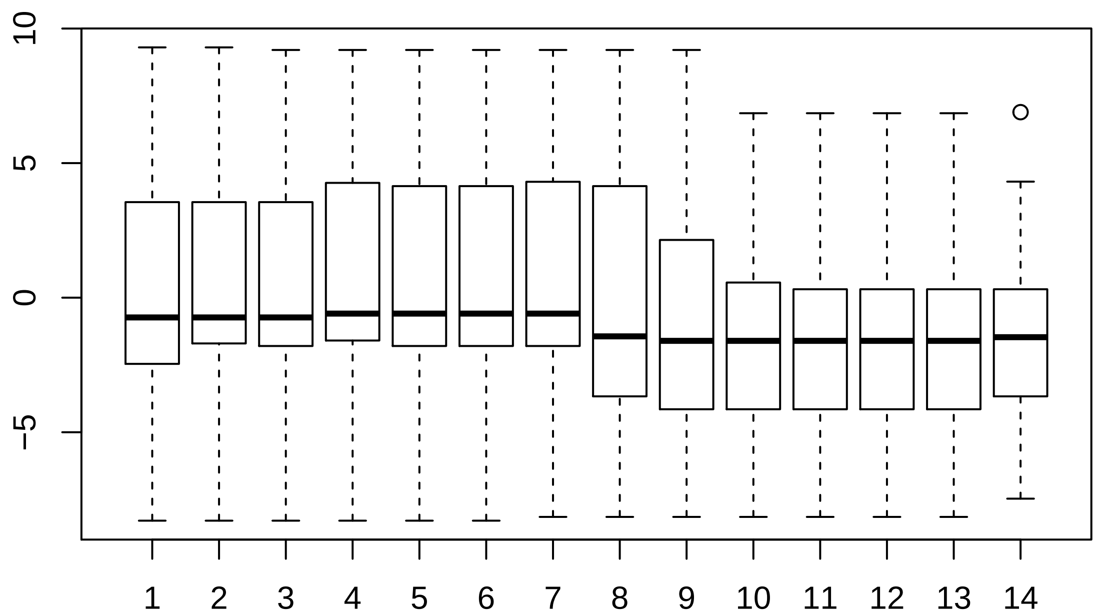

```
## NULL
```

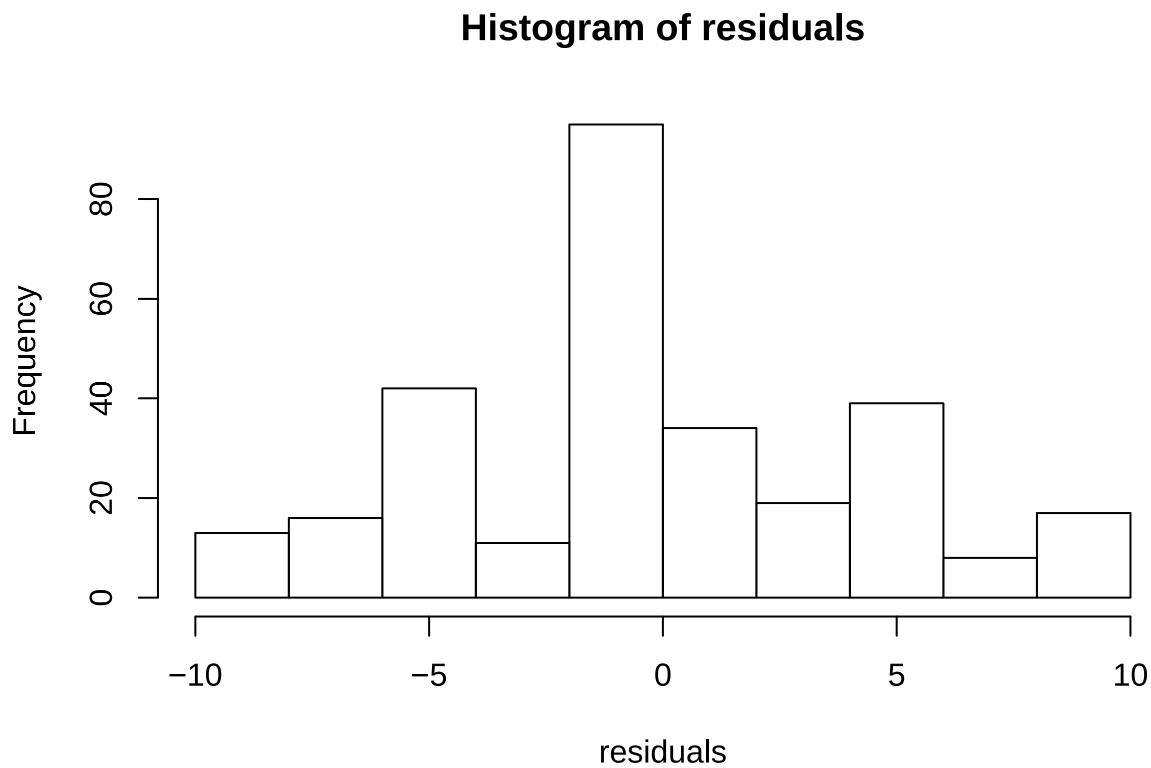
```
plot.all.residuals(all.regr.add.forecast)
```




```
## NULL
```

```
hist.all.residuals(all.regr.add.forecast)
```


10
5
0
−5
1
2
3
4
5
6
7
8
9
10
11
12
13
14



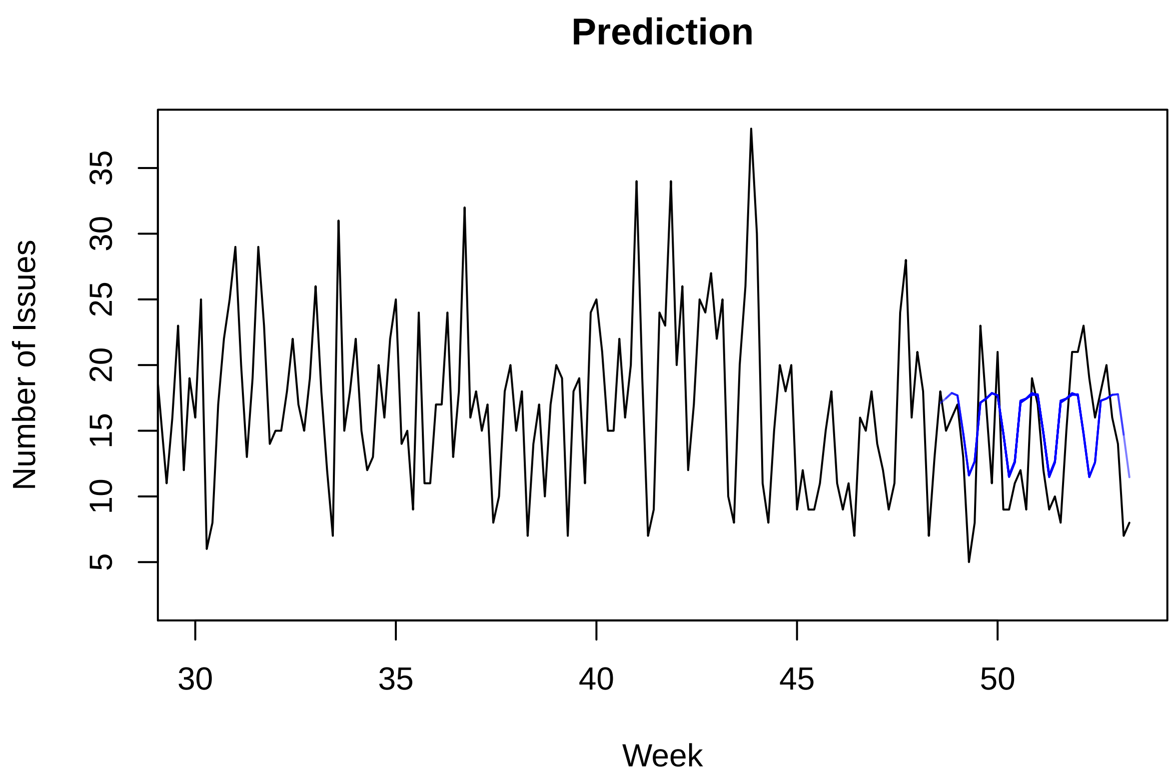

```
##     97.5%       95%        5%      2.5%
##  9.204082  8.510204 -7.455667 -8.145833
```

linear multiplicative regression

```
regr.mult.forecast <- function(sample) {
  results = list()
  results$valid <- sample$valid.ts
  results$model <- tslm(sample$train.ts ~ season, lambda = 1)
  results$pred <- forecast(results$model, h=n.valid)
  results$residual <- sample$valid.ts - results$pred$mean
  results$summary <- accuracy(results$pred, sample$valid.ts)

  return(results)
}

all.regr.mult.forecast <- sapply(1:n.sample, function(i) return(regr.mult.forecast(all.issues[,i])))

kable(mean.all.accuracy(all.regr.mult.forecast))
```

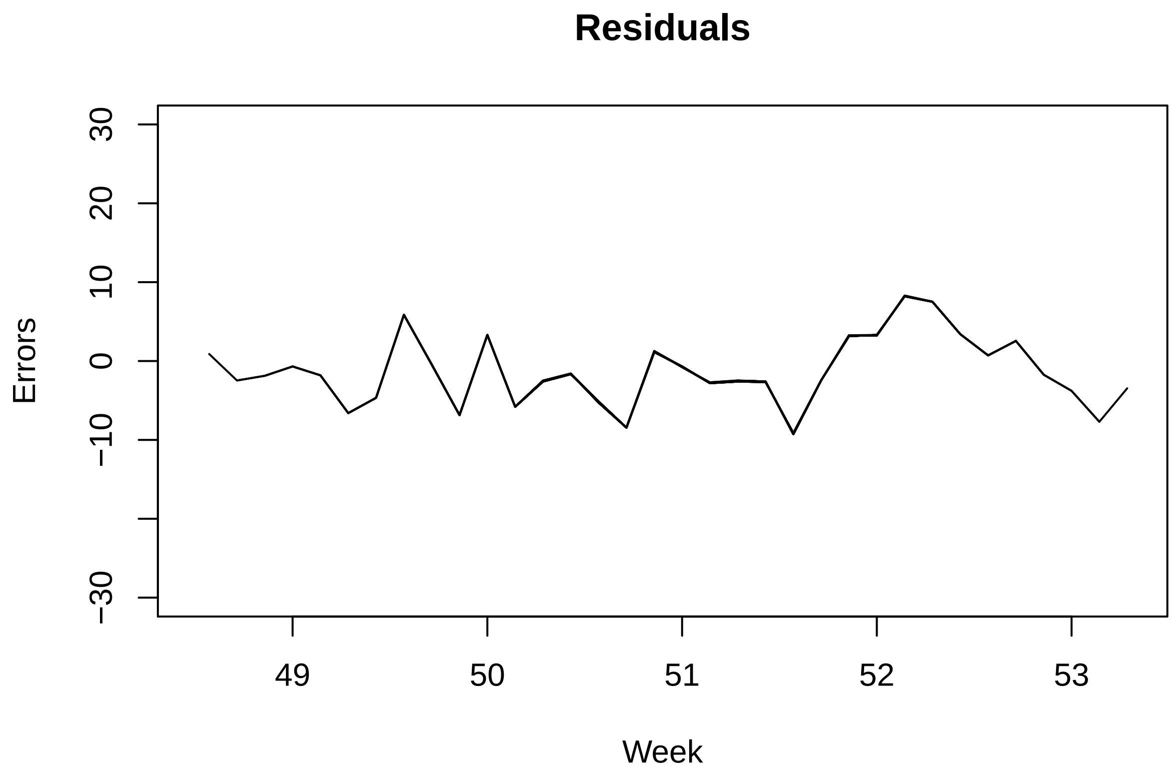
	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	0.000000	5.588445	4.398678	-15.59050	34.98281	0.8422578	0.3783599	NA
Test set	-1.523036	4.625507	3.860092	-23.23226	34.57178	0.7391587	0.2107481	0.7953356

```r
plot.all.pred(all.regr.mult.forecast)
```



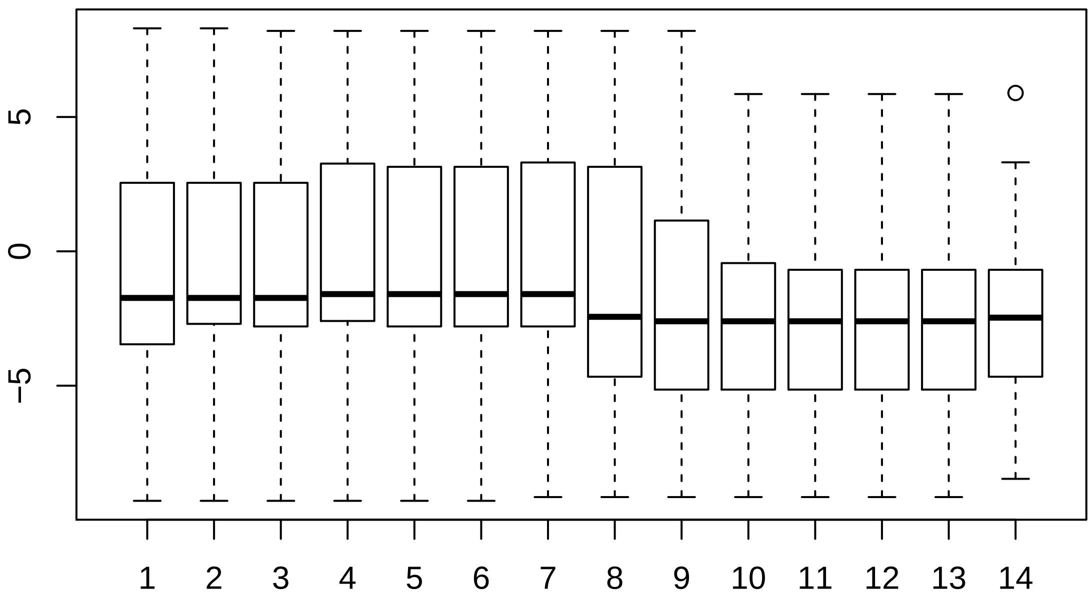

```
## NULL
```

```r
plot.all.residuals(all.regr.mult.forecast)
```

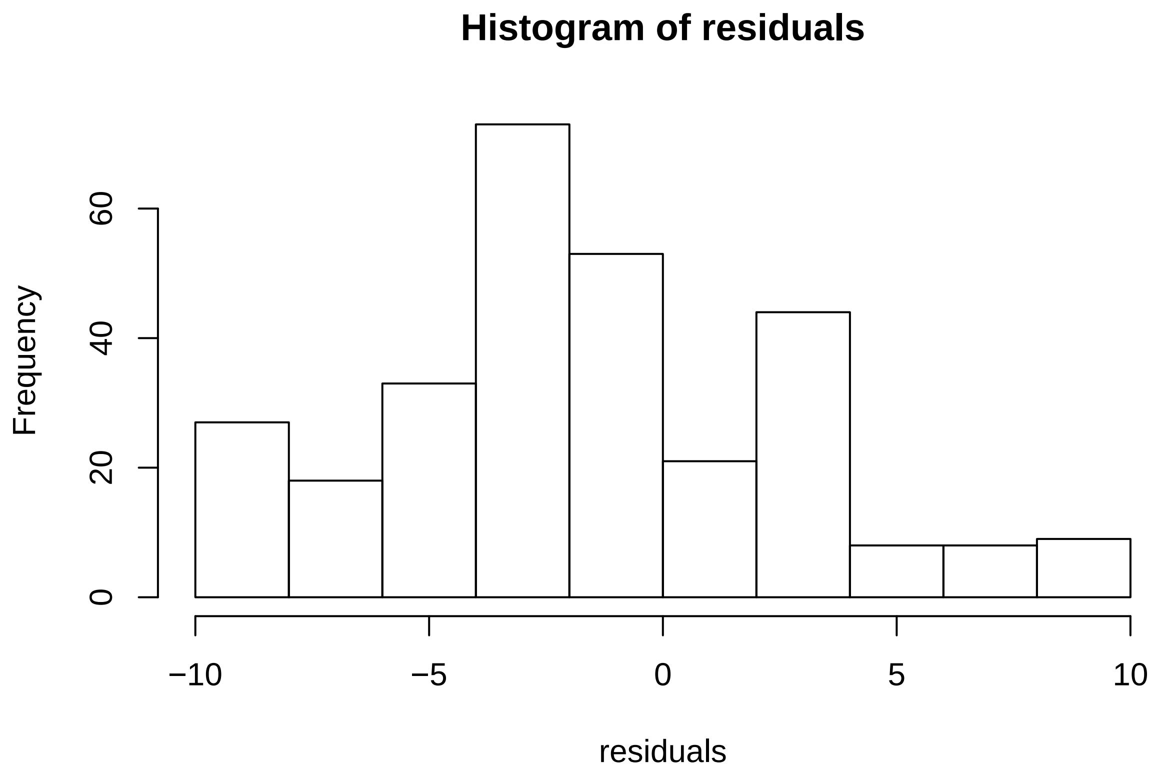



```
## NULL
```

```
hist.all.residuals(all.regr.mult.forecast)
```



5
0
−5
1
2
3
4
5
6
7
8
9
10
11
12
13
14



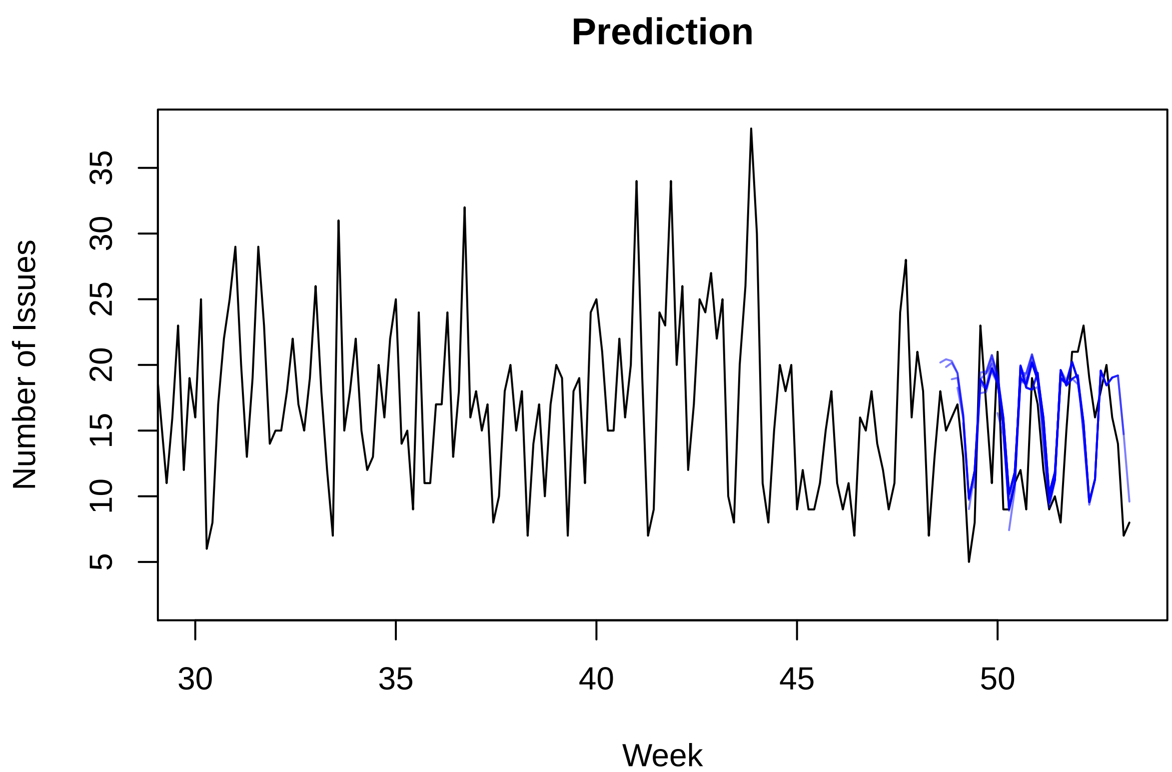

```
##     97.5%       95%        5%      2.5%
##  8.204082  7.510204 -8.455667 -9.145833
```

Arima

```r
arima.forecast <- function(sample) {
  results = list()
  results$valid <- sample$valid.ts
  results$model <- Arima(sample$train.ts, order=c(1,0,0), seasonal=c(1,1,1))
  results$pred <- forecast(results$model, h=n.valid)
  results$residual <- sample$valid.ts - results$pred$mean
  results$summary <- accuracy(results$pred, sample$valid.ts)

  return(results)
}

all.arima.forecast <- sapply(1:n.sample, function(i) return(arima.forecast(all.issues[,i])))

kable(mean.all.accuracy(all.arima.forecast))
```

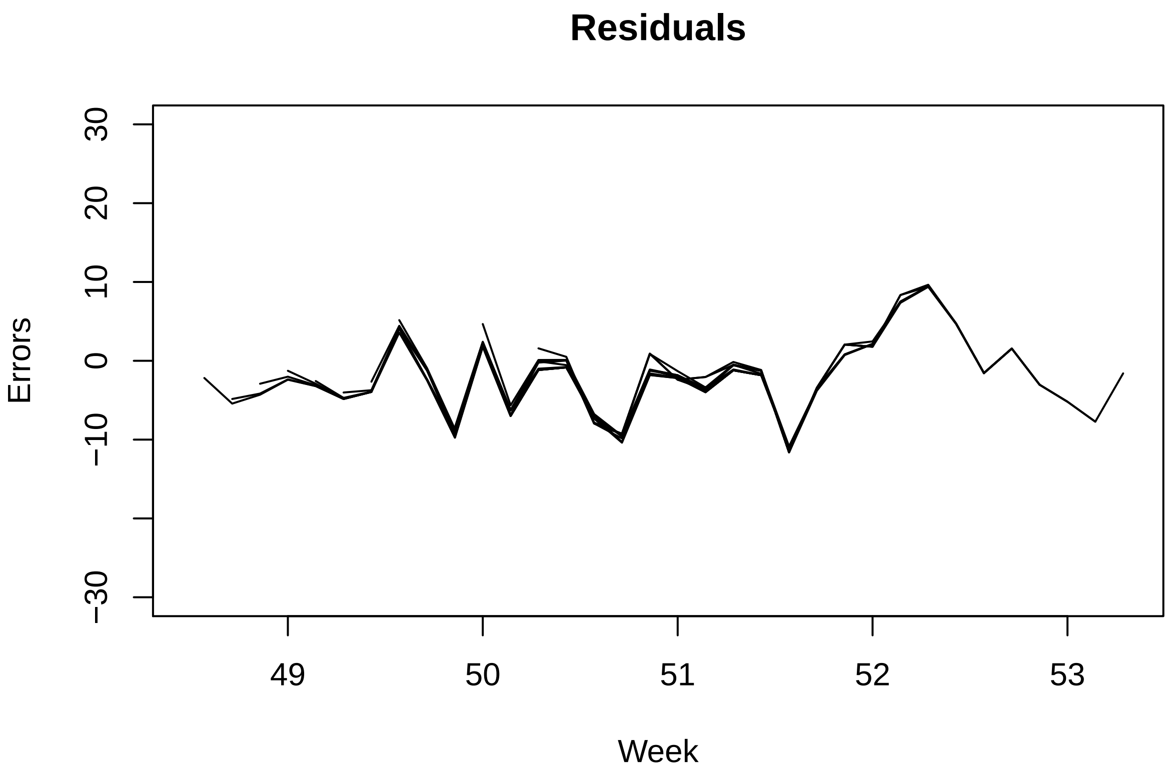
	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	0.4264553	5.083153	3.969447	-11.30950	33.11801	0.7600731	-0.0067769	NA
Test set	-2.0917555	5.169223	4.034230	-25.90281	35.57975	0.7724997	0.1558786	0.8695223

```r
plot.all.pred(all.arima.forecast)
```



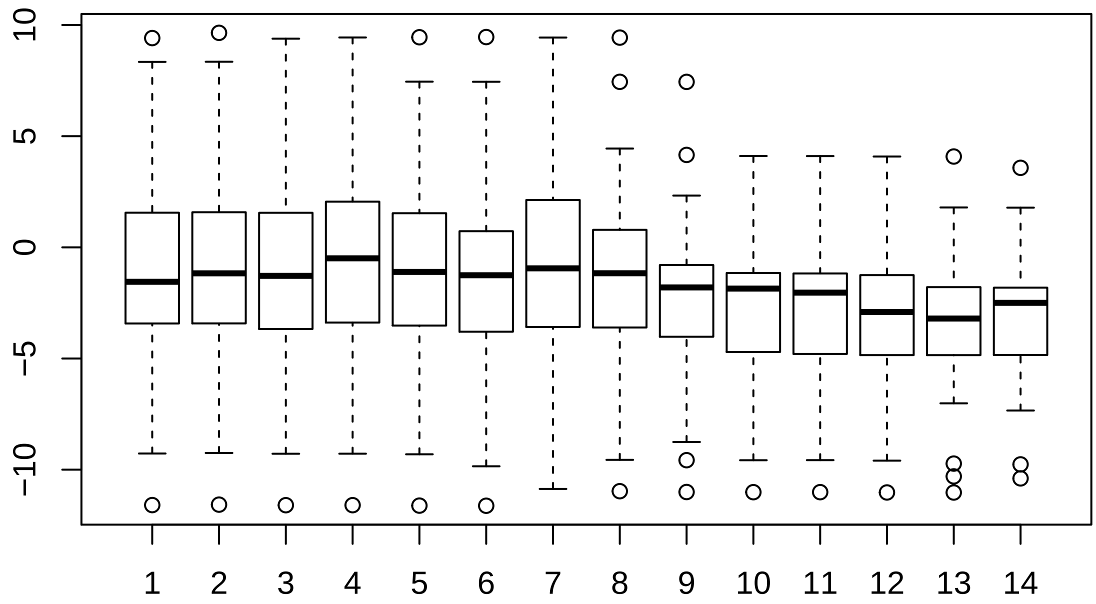

```
## NULL
```

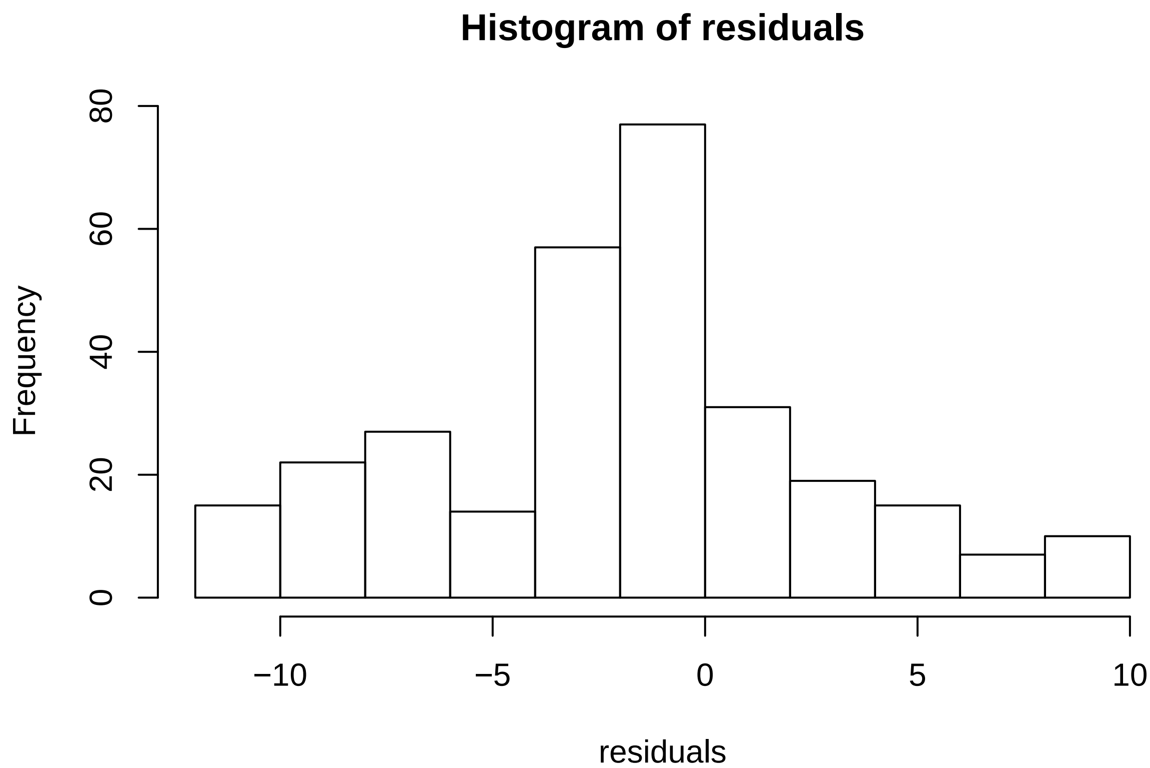
```r
plot.all.residuals(all.arima.forecast)
```




```
## NULL
```

```
hist.all.residuals(all.arima.forecast)
```


10
5
0
−5
−10
1
2
3
4
5
6
7
8
9
10
11
12
13
14




```
##      97.5%        95%          5%       2.5%
##   9.048025   7.443001 -10.006371 -11.020427
```