Experiments with Simulated Data

Sofia, Sai, Alexander, Lucy

3/14/2020

```
library(tidyverse)
```

```
## -- Attaching packages -------------------------------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ----------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(CausalImpact)
```

```
## Loading required package: bsts

## Loading required package: BoomSpikeSlab

## Loading required package: Boom

## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

##
## Attaching package: 'Boom'

## The following object is masked from 'package:stats':
##
##     rWishart

##
## Attaching package: 'BoomSpikeSlab'

## The following object is masked from 'package:stats':
##
##     knots

## Loading required package: zoo

##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: xts
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
##
## Attaching package: 'bsts'
```

```
## The following object is masked from 'package:BoomSpikeSlab':
##
##     SuggestBurn
```

```
library(bsts)
set.seed(27)
```

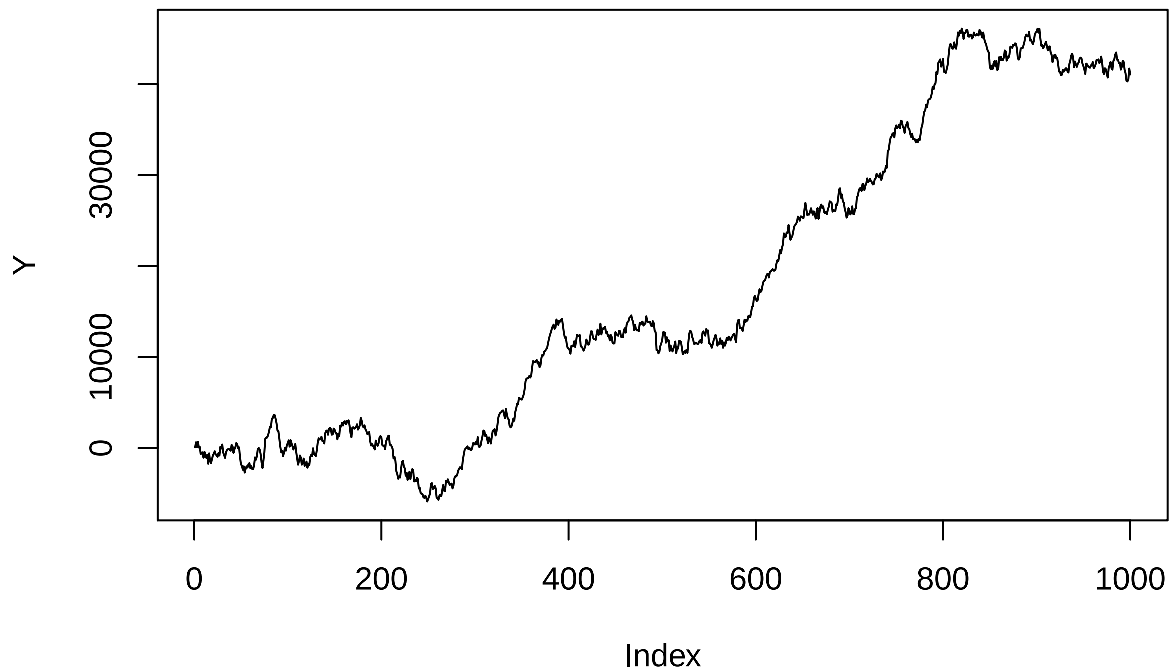
Plan

- Generate data
- Change BSTSM values for the Gaussian to see how they impact the performance.
- Create BSTSM with different state components.

Genarating Time Series Data

```
M_t = 67
V_t = 5
Y= NULL

for (i in 1:1000){
  Y[i] <- M_t +  rnorm(n= 1, mean = 0, sd = 5)
  M_t  <- M_t + V_t + rnorm(n= 1, mean = 0, sd = 500)
  V_t  <- V_t + rnorm(n= 1, mean = 0.08, sd = 1)
}
plot(Y, type = "l")
```



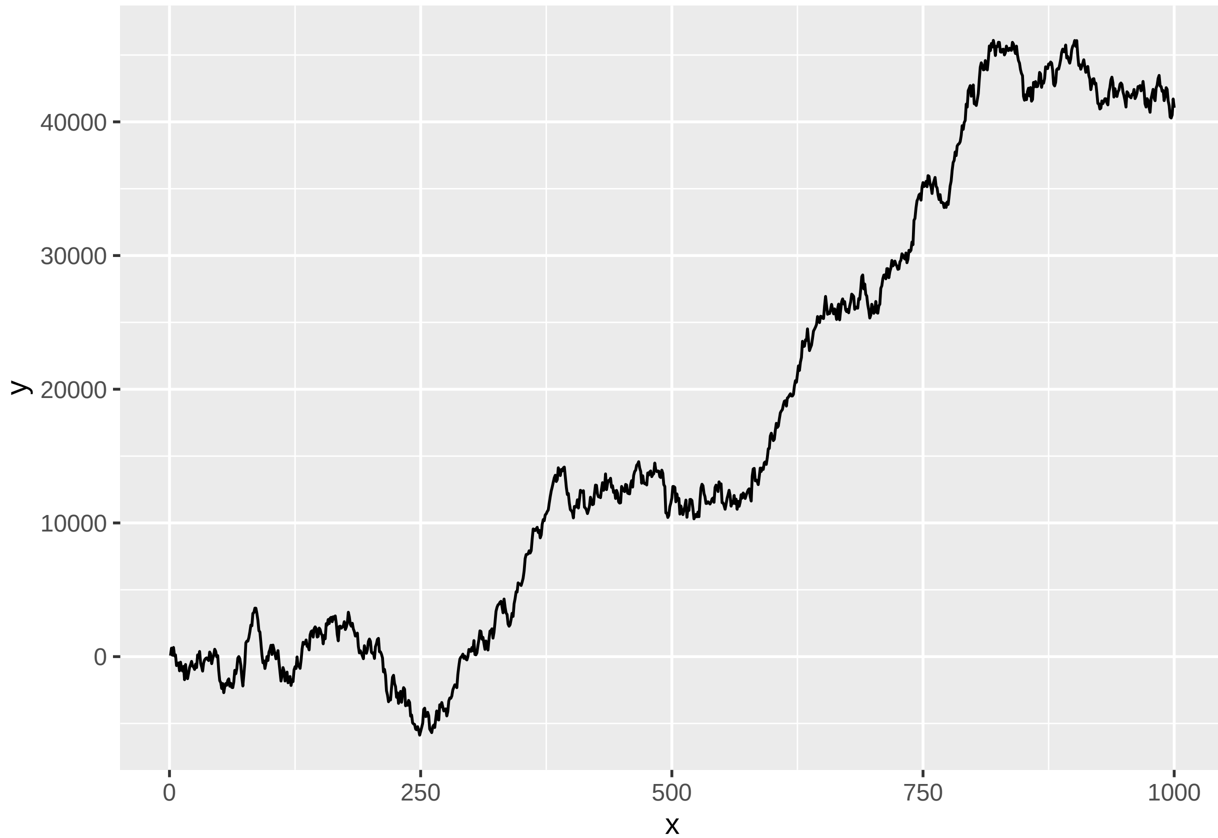

```
data_lin_trend<- data_frame(cov = Y + rnorm(1000, 0, 0.01), Y)
```

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

```
data_lin_trend$Y[800:1000] <- data_lin_trend$Y[800:1000]+20000

plot(data_lin_trend$Y, type = "l")
```


```r
data <- data_frame(y = Y) %>% mutate(x = 1:1000)
ggplot(data, aes(x = x, y = y))+
  geom_line()
```

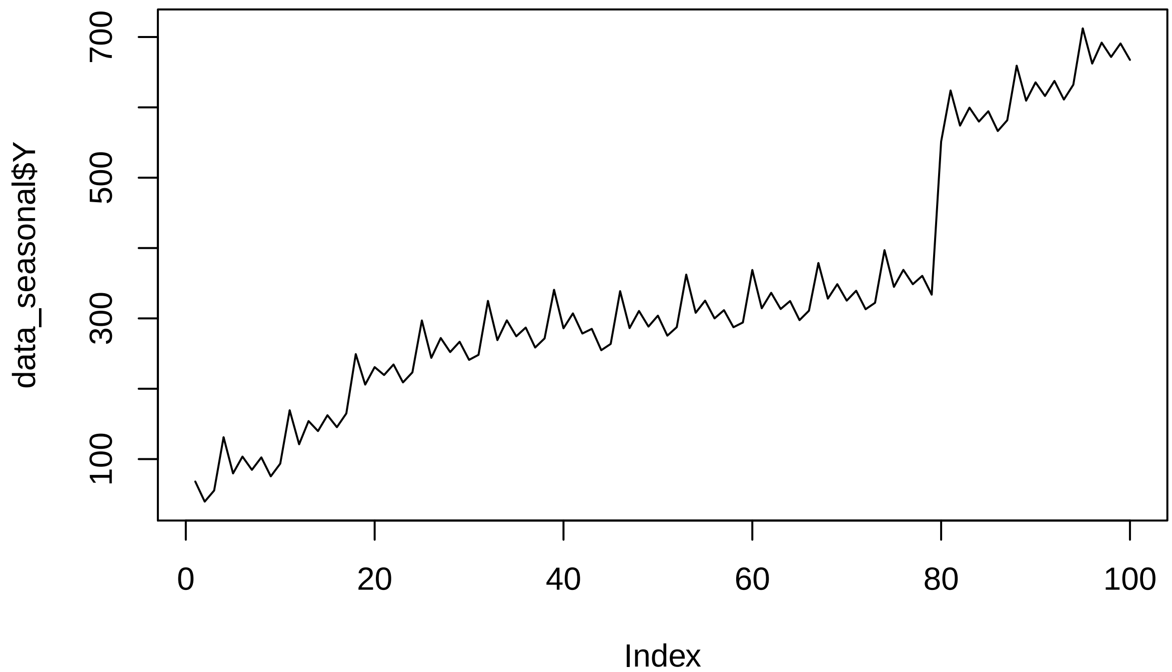


With seasonality

```
M_t = 67
V_t = 5
G_t = c(-10,15,-5, 50, -20, -30,0)
Y= NULL
gamma_t = G_t[7]
dummy1 = NULL
dummy2 = NULL
dummy3 = NULL
for (i in 1:100){
  current_season <- length(G_t)- i%%length(G_t)

  Y[i] <- M_t +  rnorm(n= 1, mean = 0, sd = 1) + gamma_t
  M_t  <- M_t + V_t + rnorm(n= 1, mean = 0, sd = 1)
  V_t  <- V_t + rnorm(n= 1, mean = 0.08, sd = 1)
  gamma_t <- G_t[current_season] + rnorm(1,0,1)
  dummy1[i] <- i
  dummy2[i] <- sin(i)
  dummy3[i] <- rnorm(1,0,1)
}
plot(Y, type = "l")
```


```
data_seasonal<- data_frame(cov = Y + rnorm(100, 0, 0.01), Y, dummy1,dummy2,dummy3)
data_seasonal$Y[80:100] <- data_seasonal$Y[80:100] + 200
plot(data_seasonal$Y, type = "l")
```

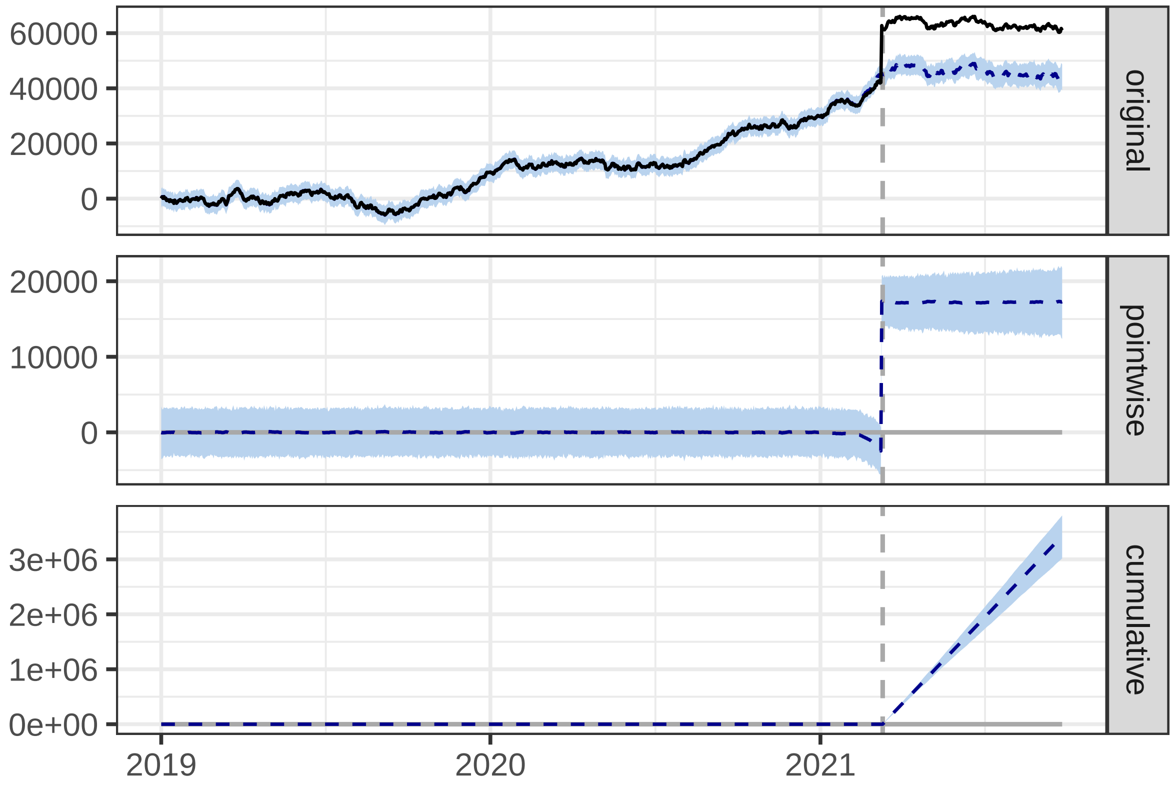


Running Causal impact on data with trend

```
pre.period <- as.Date(c("2019-01-01", "2021-03-11"))
post.period <- as.Date(c("2021-03-12", "2021-09-27"))
time.points <- seq.Date(as.Date("2019-01-01"), by = 1, length.out = 1000)
data <- zoo(cbind(data_lin_trend$Y,data_lin_trend$cov
                  ), time.points)
impact <- CausalImpact(data, pre.period, post.period)
```

```
## Warning in FormatInputPrePostPeriod(pre.period, post.period, data): Setting
## post.period[2] to end of data: 2021-09-26
```

```
plot(impact)
```

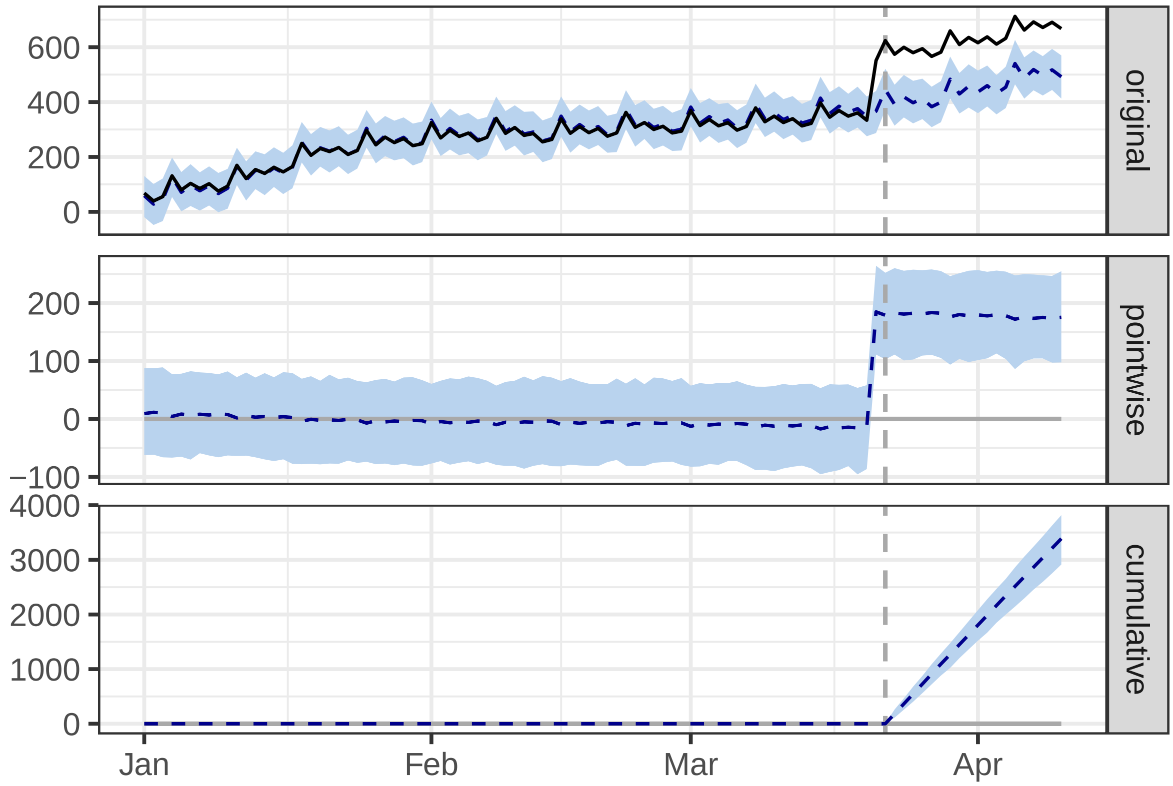



```
# Running Causal impact on data with seasonality
```

```
pre.period <- as.Date(c("2019-01-01", "2019-03-22"))
post.period <- as.Date(c("2019-03-23", "2019-04-11"))
time.points <- seq.Date(as.Date("2019-01-01"), by = 1, length.out = 100)
data <- zoo(cbind(data_seasonal$Y,data_seasonal$cov
                  ), time.points)
impact <- CausalImpact(data, pre.period, post.period)
```

```
## Warning in FormatInputPrePostPeriod(pre.period, post.period, data): Setting
## post.period[2] to end of data: 2019-04-10
```

```
plot(impact)
```



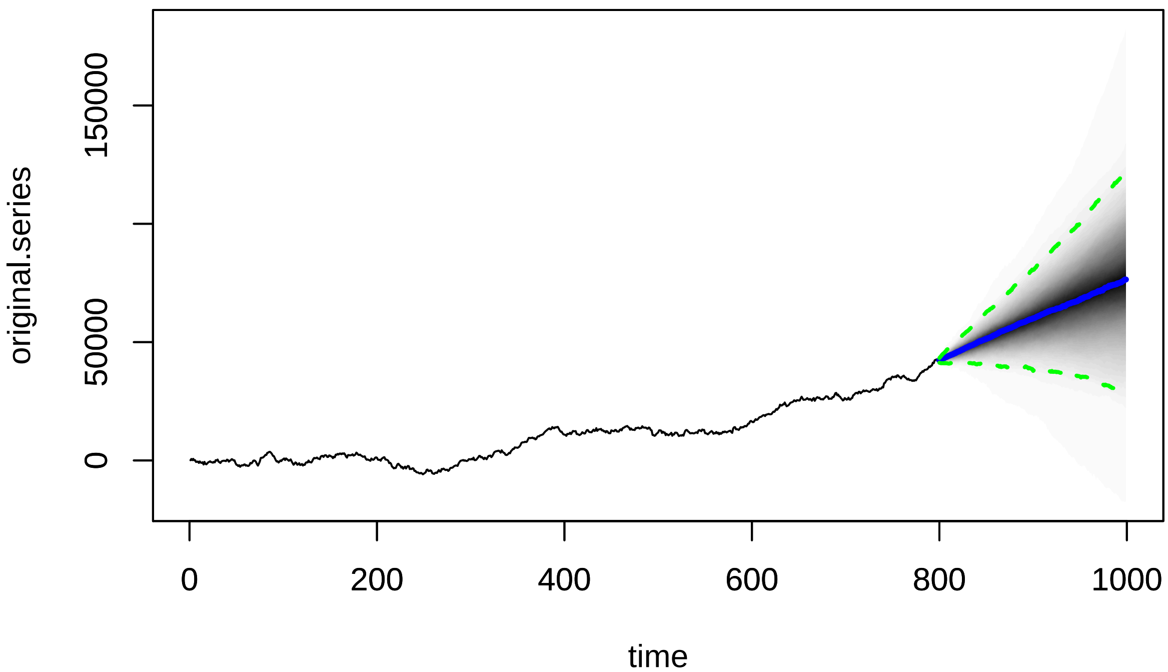
Linear Model without regression and seasionality. Plot is drawn to predict values for 20 future time points. Created Pre-intervention period by slicing the data frame before the intervention(800) COMMENT - Without regression the prediction is not accurate. Confidence intervals are pretty wide

```
data_pre_intervention_lin = data_lin_trend[1:799,]
linear_only_state <- AddLocalLinearTrend(list(), data_pre_intervention_lin$Y)
model_linear <- bsts(data_pre_intervention_lin$Y,
               state.specification = linear_only_state,
               niter = 1000) #### LINEAR MODEL WITHOUT REGRESSION AND SEASIONALITY
```

```
## =-=-=-=-= Iteration 0 Thu Mar 19 16:06:08 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 100 Thu Mar 19 16:06:09 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 200 Thu Mar 19 16:06:10 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 300 Thu Mar 19 16:06:11 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 400 Thu Mar 19 16:06:11 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 500 Thu Mar 19 16:06:12 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 600 Thu Mar 19 16:06:14 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 700 Thu Mar 19 16:06:14 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 800 Thu Mar 19 16:06:15 2020
```

```
##  =-=-=-=-=
## =-=-=-=-= Iteration 900 Thu Mar 19 16:06:16 2020
##  =-=-=-=-=
```

```
pred_linear <- predict(model_linear, horizon = 200)
plot(pred_linear, plot.original = 800)
```



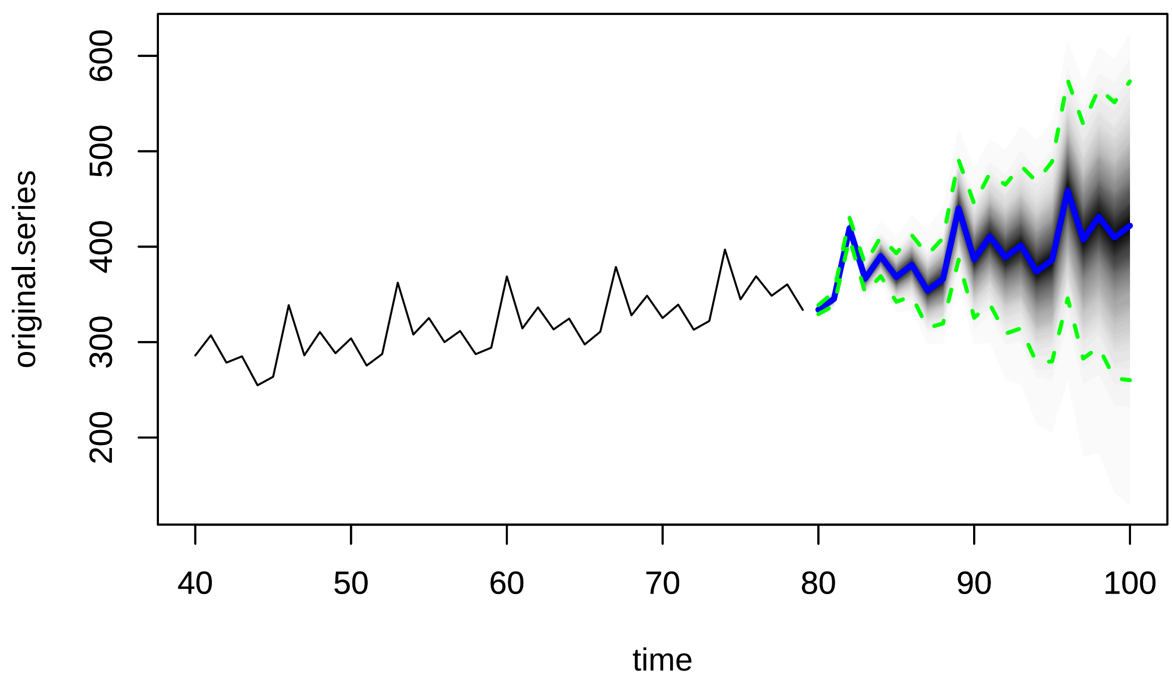
Linear model with seasionality and regression Plot is drawn to predict values for 20 future time points. Created Pre-intervention period by slicing the data frame before the intervention(80) COMMENT- Prediction when modelled properly seems to be close to true values, small confidence intervals

```
data_pre_seasonal = data_seasonal[1:79,]
linear_only_state <- AddLocalLinearTrend(list(), data_pre_seasonal$Y)
linear_seasonal_state <- AddSeasonal(linear_only_state, data_pre_seasonal$Y, nseasons = 7)
model_lin_season_regr <- bsts(Y ~ .,
               state.specification = linear_seasonal_state,
               niter = 1000,
               data = data_pre_seasonal,
               expected.model.size = 1) #### LINEAR MODEL WITH REGRESSION AND SEASIONALITY
```

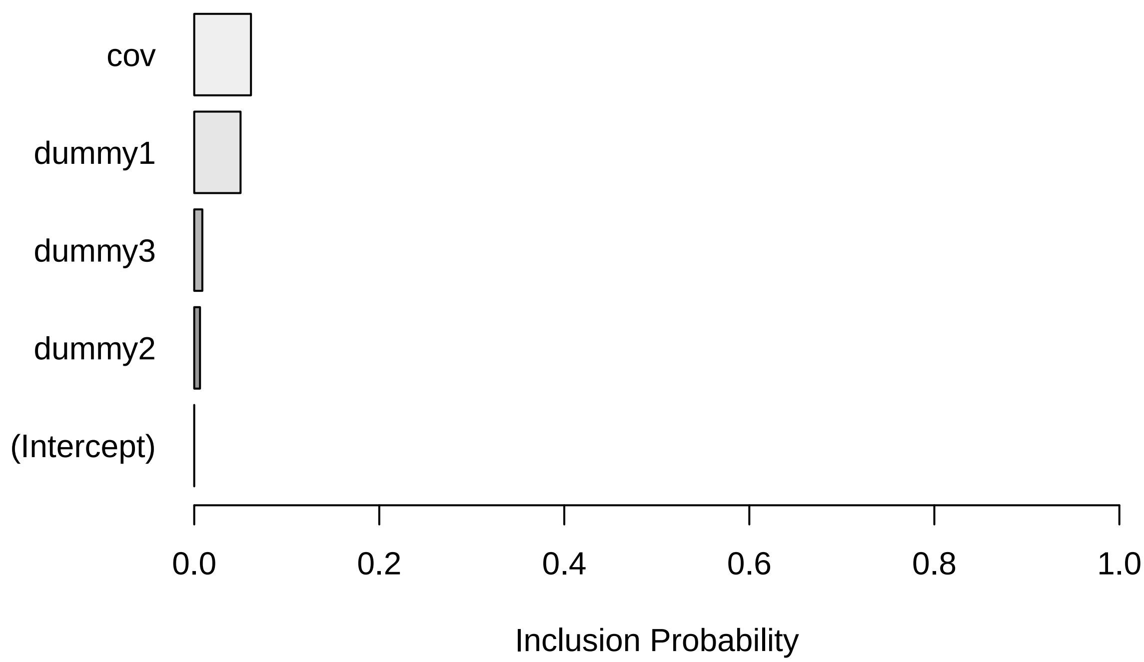
```
## =-=-=-=-= Iteration 0 Thu Mar 19 16:06:18 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 100 Thu Mar 19 16:06:18 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 200 Thu Mar 19 16:06:18 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 300 Thu Mar 19 16:06:18 2020
##  =-=-=-=-=
```

```
## =-=-=-=-= Iteration 400 Thu Mar 19 16:06:18 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 500 Thu Mar 19 16:06:18 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 600 Thu Mar 19 16:06:19 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 700 Thu Mar 19 16:06:19 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 800 Thu Mar 19 16:06:19 2020
##  =-=-=-=-=
## =-=-=-=-= Iteration 900 Thu Mar 19 16:06:19 2020
##  =-=-=-=-=
```

```
pred_linear_seasonal_regr <- predict(model_lin_season_regr, horizon = 20,newdata = data_seasonal[80:100
plot(pred_linear_seasonal_regr, plot.original = 40)
```




```
plot(model_lin_season_regr,"coef")
```



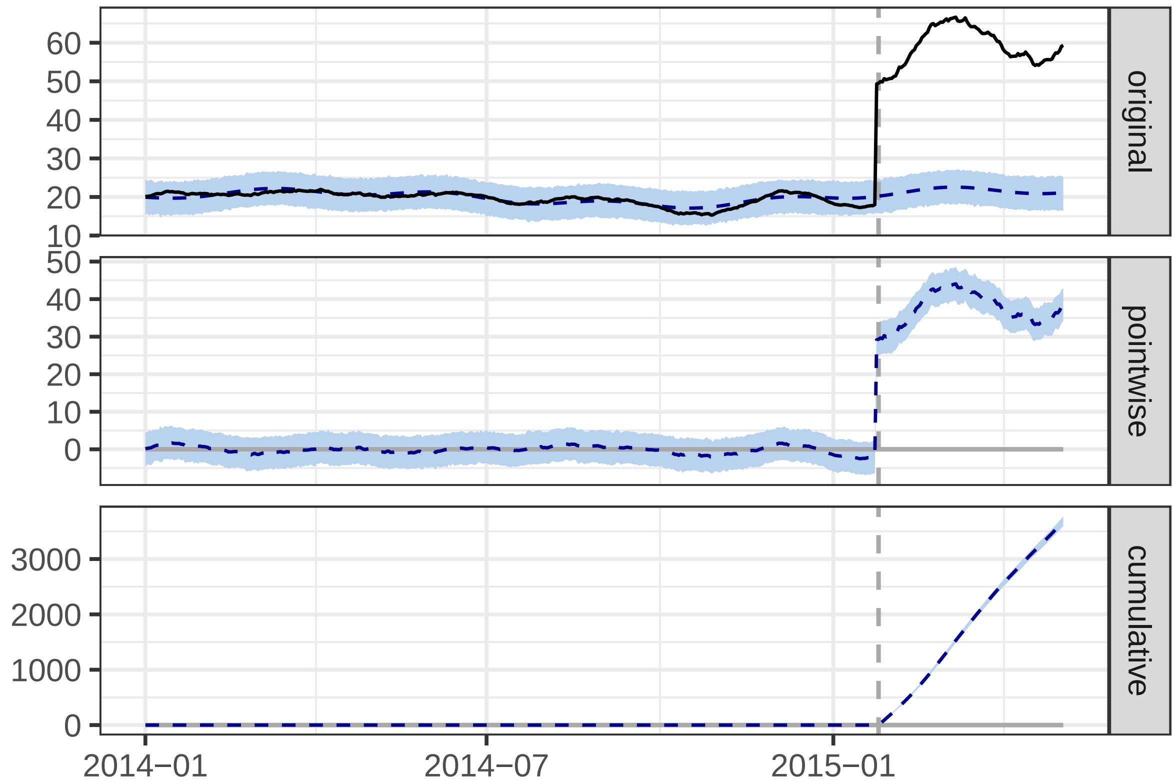
Reproducing Theoritical Simulation Created two regression coefficients,two sinusoidal covariates and a local level Initialized covariates to 1 and level to 0 as explained in the paper Intervention occured at 389 days later by multiplying it with e

```
data = NULL
linear_values = NULL
linear_values[1] = 0
beta1 = NULL
beta1[1] = 1
beta2 = NULL
beta2[1] = 1
cov1 = NULL
cov2 = NULL
cov1[1] = sin(pi/45)
cov2[1] = sin(pi/180)
for(i in 2:488){
  linear_values[i] = rnorm(n = 1,mean = linear_values[i-1],sd = 0.1)
  beta1[i] = rnorm(n = 1,mean = beta1[i-1],sd= 0.1)
  beta2[i] = rnorm(n = 1,mean = beta2[i-1],sd= 0.1)
  cov1[i] = sin((pi*i)/45)
  cov2[i] = sin((pi*i)/180)
}
output_true =20 + linear_values + rnorm(n = 1, mean = 0, sd = 0.1) + (cov1 * beta1) + (cov2 * beta2)
output_intervention = output_true
output_intervention[389:488] = output_intervention[389:488]*exp(1)
time.points <- seq.Date(as.Date("2014-01-01"), by = 1, length.out = 488)
pre.period <- as.Date(c("2014-01-01", "2015-01-25"))
```

```r
post.period <- as.Date(c("2015-01-26", "2015-05-04"))
data <- zoo(cbind(output_intervention, cov1,cov2), time.points)
#pre.period = c(1,389)
#post.period = c(390,488)
#data = cbind(output_intervention,cov1,cov2)
impact <- CausalImpact(data, pre.period, post.period)
```

```
## Warning in FormatInputPrePostPeriod(pre.period, post.period, data): Setting
## post.period[2] to end of data: 2015-05-03
```

```r
plot(impact)
```




```r
matplot(output_true,main = "True Output",col = "green",type = "l")
```


True Output
output_true
16
18
20
22
24
0
100
200
300
400
500