

Figures: Trend Analysis

Murray Stokely

2025-01-23

Contents

1	Introduction	1
2	Weekly Pattern Analysis	2
2.1	Figure 5	5
3	Sensitivity Analysis of Trend and Period for Commitment Savings	5
3.1	Table 3	6
3.2	Figure 6	9

1 Introduction

This file includes figures and analysis for Section 3.3.1 Trend in Setting Optimal Commitment Level for Periodic Demand.

We start by aggregating the different regions and SKUs together to a single timeseries of total VM demand, and normalize this to a 100 unit peak over that time window.

```
data <- read_parquet("../hourly_normalized.parquet")
dim(data)
```

```
## [1] 524832      4
```

```
head(data)
```

```
## # A tibble: 6 x 4
```

	USAGE_HOUR	REGION_NUM	INSTANCE_TYPE	NORM_USAGE
	<dtm>	<dbl>	<chr>	<dbl>
## 1	2021-02-01 00:00:00	2	A	172
## 2	2021-02-01 00:00:00	4	F	1
## 3	2021-02-01 00:00:00	1	I	9
## 4	2021-02-01 00:00:00	4	I	8
## 5	2021-02-01 00:00:00	3	I	6
## 6	2021-02-01 00:00:00	1	A	169

```
range(data$USAGE_HOUR)
```

```
## [1] "2021-02-01 00:00:00 UTC" "2024-01-31 23:00:00 UTC"
```

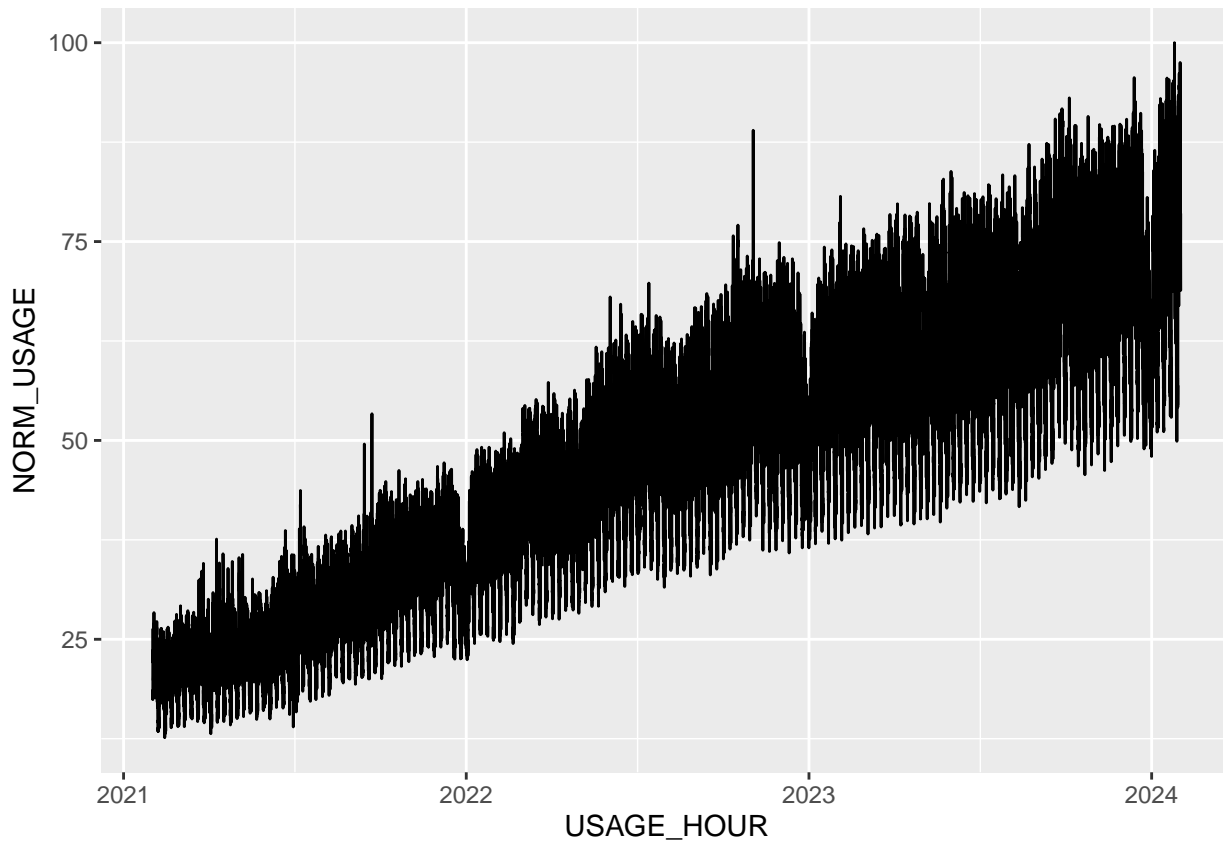
```
data.all <- data %>% group_by(USAGE_HOUR) %>% summarise(across(NORM_USAGE, sum))
data.all$NORM_USAGE = 100*data.all$NORM_USAGE / max(data.all$NORM_USAGE)
head(data.all)
```

```
## # A tibble: 6 x 2
```

	USAGE_HOUR	NORM_USAGE
--	------------	------------

```
##      <dtm>                <dbl>
## 1 2021-02-01 00:00:00      18.4
## 2 2021-02-01 01:00:00      18.6
## 3 2021-02-01 02:00:00      17.6
## 4 2021-02-01 03:00:00      17.4
## 5 2021-02-01 04:00:00      19.7
## 6 2021-02-01 05:00:00      20.4
```

```
ggplot(data.all, aes(x=USAGE_HOUR, y=NORM_USAGE)) + geom_line()
```



2 Weekly Pattern Analysis

We trim the start and end of our timeseries to align our dataset with Sunday through Saturday weeks so we can split it up in 7 day chunks and look at the distribution of weekly patterns.

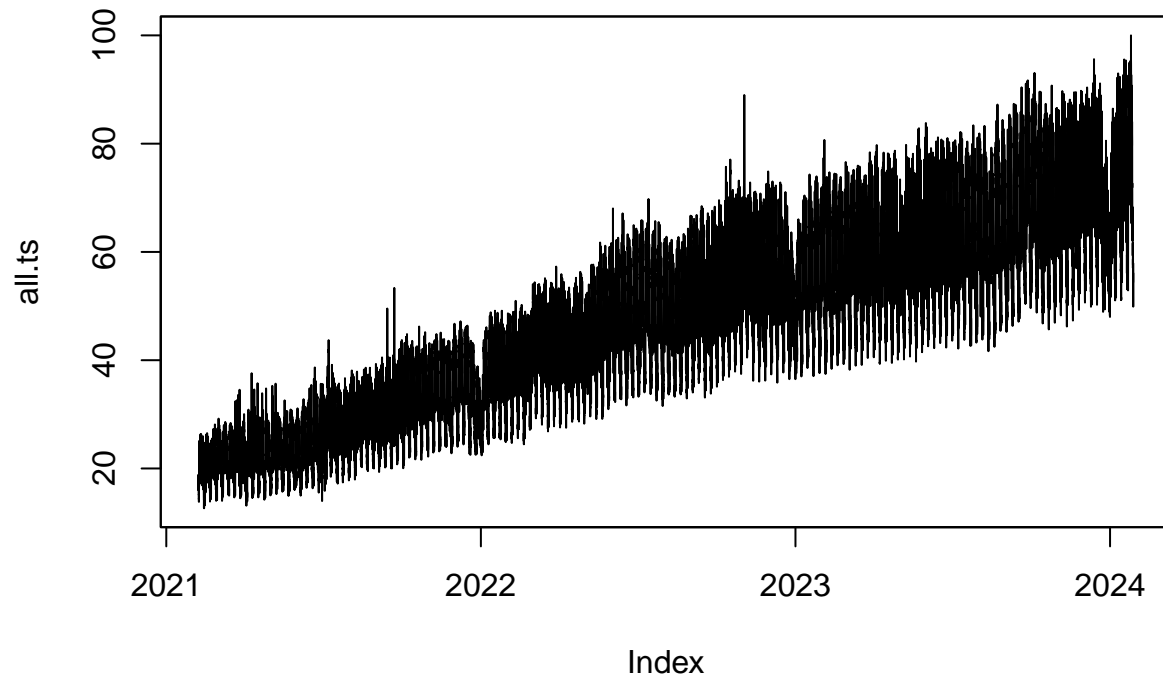
We then generate timeseries of the maximum, minimum, and mean of each day.

```
all <- data.all %>% subset(USAGE_HOUR >= as.POSIXct("2021-02-07", tz="UTC")) %>%
  subset(USAGE_HOUR < as.POSIXct("2024-01-28", tz="UTC"))
range(all$USAGE_HOUR)
```

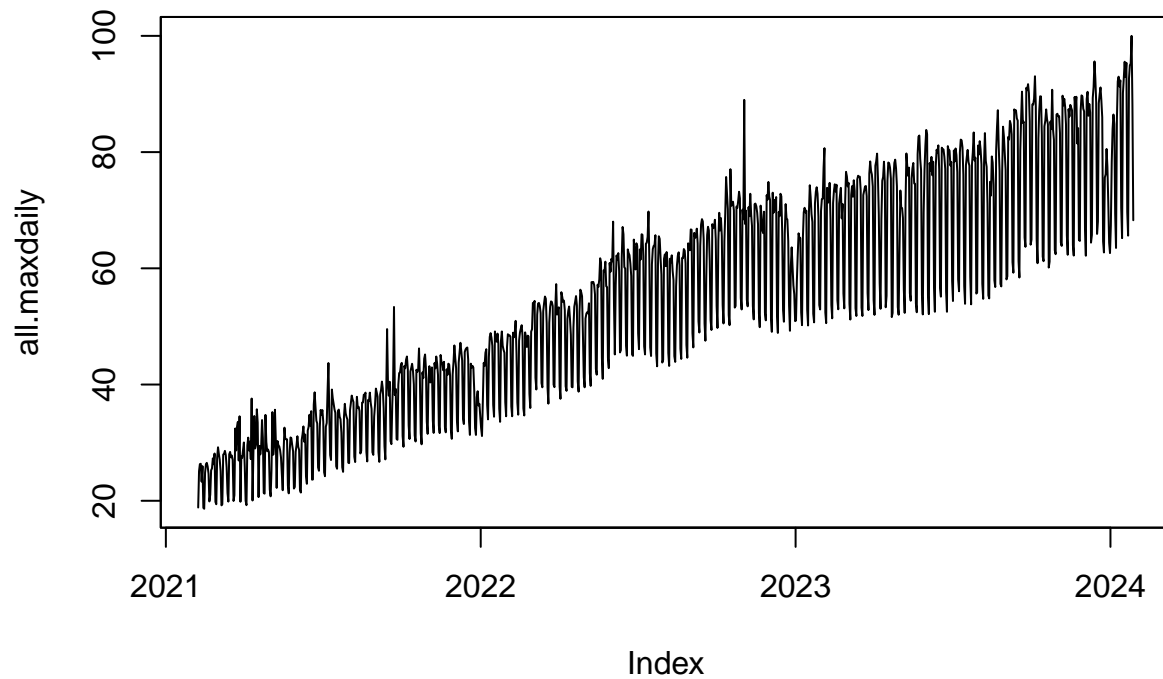
```
## [1] "2021-02-07 00:00:00 UTC" "2024-01-27 23:00:00 UTC"
```

```
all.ts <- zoo(all$NORM_USAGE, all$USAGE_HOUR)
# Compute the maximumes over each day.
all.maxdaily <- rollapply(all.ts, width=24, by=24, FUN=max)
all.meandaily <- rollapply(all.ts, width=24, by=24, FUN=mean)
all.mindaily <- rollapply(all.ts, width=24, by=24, FUN=min)
```

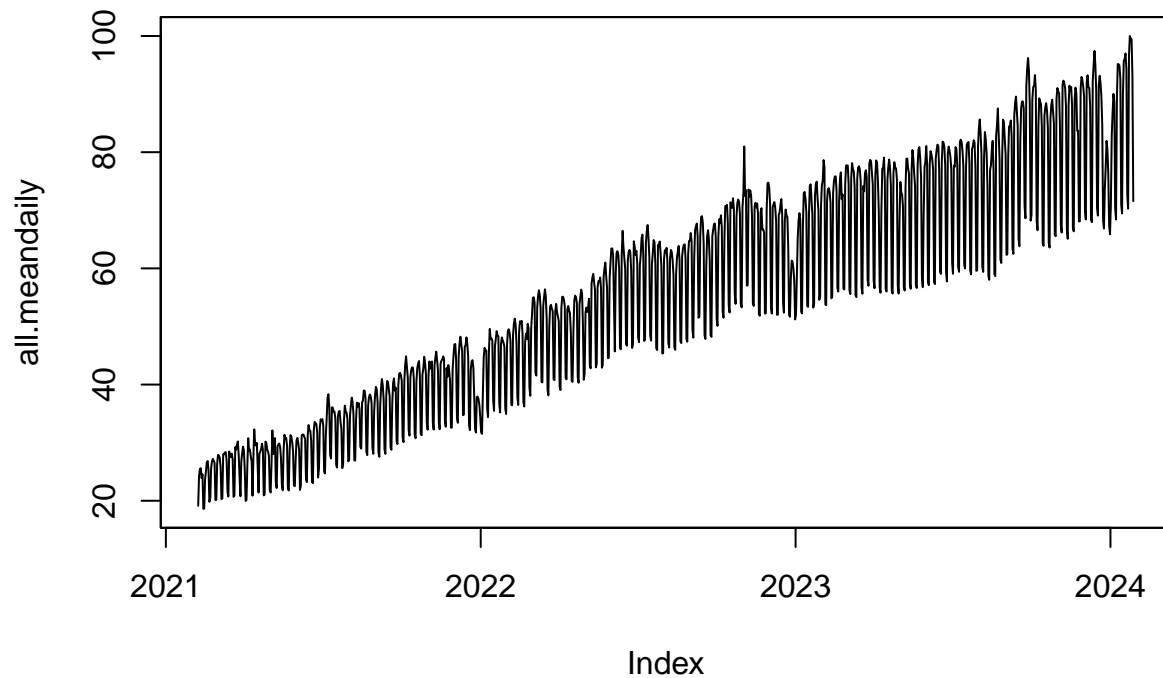
```
plot(all.ts)
```



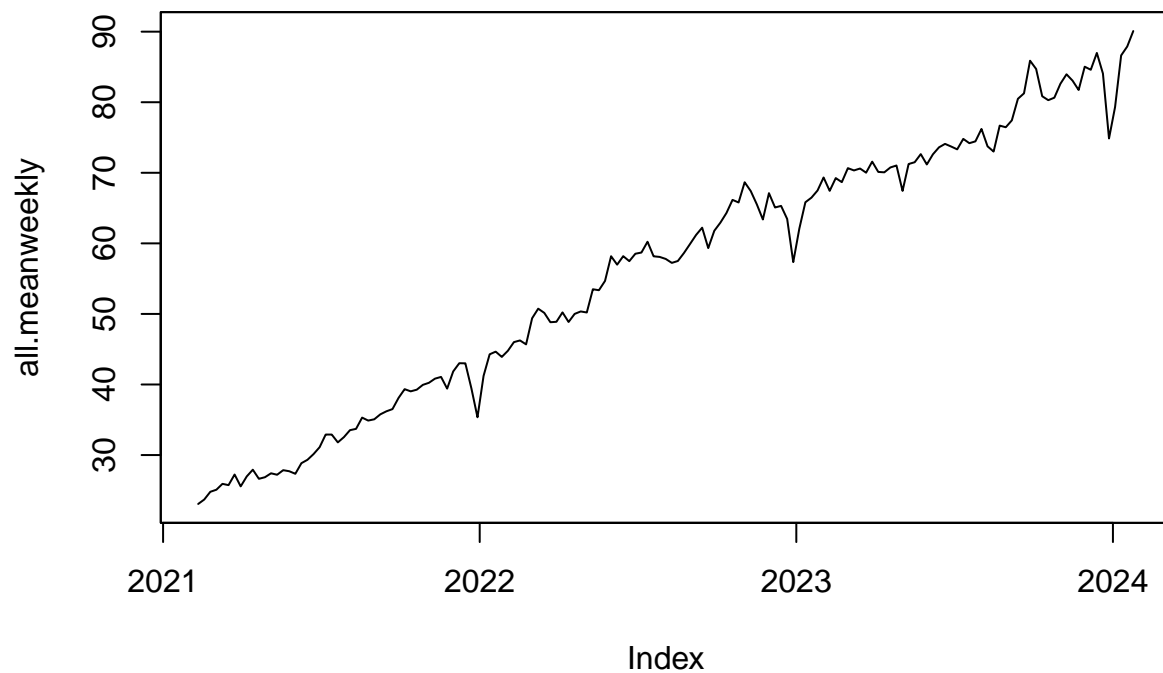
```
plot(all.maxdaily)
```



```
all.meandaily <- 100*all.meandaily/(max(all.meandaily))  
plot(all.meandaily)
```



```
all.meanweekly <- rollapply(all.meandaily, width=7, by=7, FUN=mean)
plot(all.meanweekly)
```



```
growthmultiple.3yr <- max(range(all.meanweekly)) / min(range(all.meanweekly))
growthrate.3yr <- growthmultiple.3yr ^ (1/3)
print(growthmultiple.3yr)
```

```
## [1] 3.904665
```

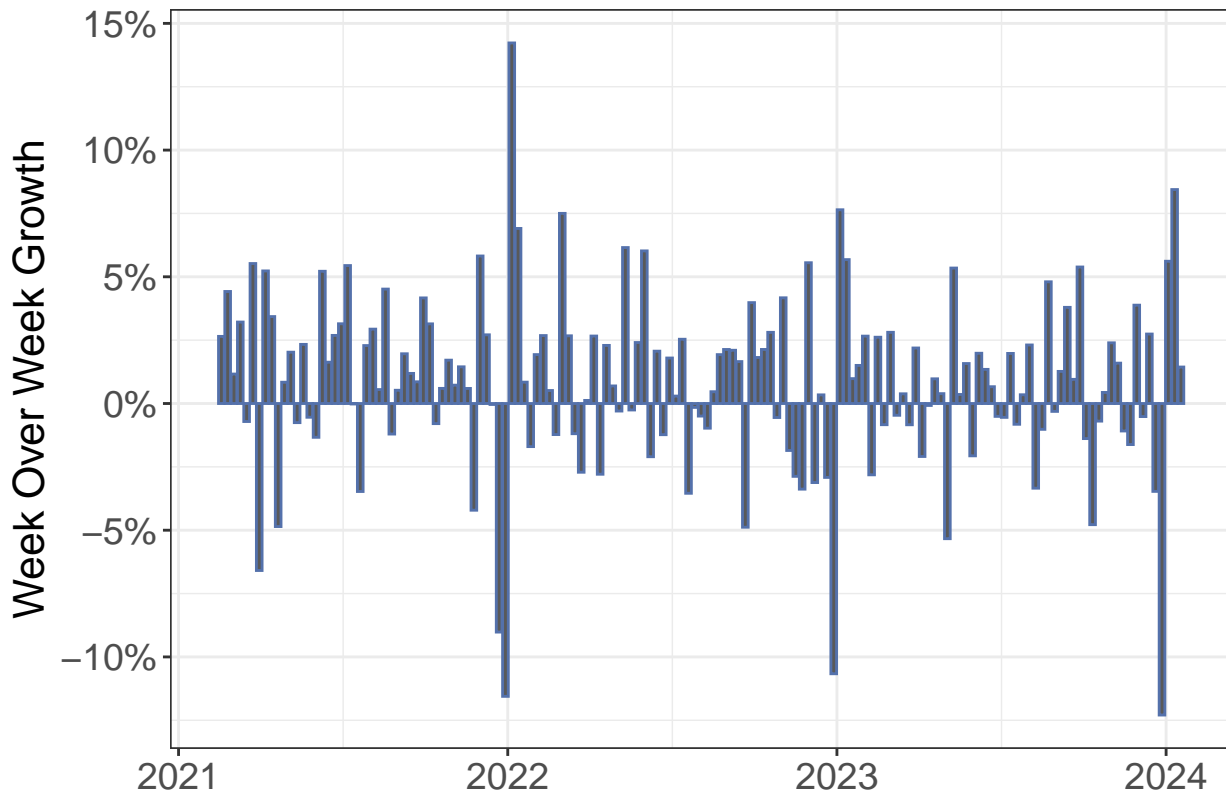
```
print(growthrate.3yr)
```

```
## [1] 1.574688
```

2.1 Figure 5

Week over week growth timeseries.

```
library(scales)
growth.rate <- diff(all.meanweekly) / head(all.meanweekly, -1)
growth.rate.df <- data.frame(time=time(growth.rate), value=as.numeric(growth.rate))
ggplot(growth.rate.df, aes(x=time, y=value)) + geom_bar(stat="identity", color="#5471AB") + theme_bw()
```



How many weeks is the trend negative?

```
length(which(diff(all.meanweekly) < 0 )) / length(diff(all.meanweekly))
```

```
## [1] 0.3701299
```

3 Sensitivity Analysis of Trend and Period for Commitment Savings

First lets pick a week of data to start with:

```
data.subset <- data.all %>%
  subset(USAGE_HOUR >= as.POSIXct("2024-01-21", tz="UTC")) %>%
  subset(USAGE_HOUR < as.POSIXct("2024-01-28", tz="UTC"))
range(data.subset$USAGE_HOUR)
```

```
## [1] "2024-01-21 00:00:00 UTC" "2024-01-27 23:00:00 UTC"
```

```
dim(data.subset)
```

```
## [1] 168 2
```

Needed helper functions from optimization.Rmd

```
data.subset.orig <- data.subset
```

3.1 Table 3

We now do a sensitivity analysis on the costs for different annual trend and number of weeks.

```
data.subset <- data.subset.orig
#data.subset$NORM_USAGE <- data.subset$NORM_USAGE * 10

ExtendWeek <- function(df, weeks=1, trend=0.05) {
  daily.trend <- (1+trend)^(1/365)
  df.tmp <- df
  for (w in seq(weeks)) {
    df.new <- df
    df.new$USAGE_HOUR <- df.new$USAGE_HOUR + (w*7*86400)
    df.new$NORM_USAGE <- df.new$NORM_USAGE * daily.trend^(rep(seq(((w-1)*7)+1,((w-1)*7)+7),each=24))
    #df.new$DEMAND <- df.new$DEMAND * daily.trend^(rep(seq(((w-1)*7)+1,((w-1)*7)+7),each=24))
    df.tmp <- rbind(df.tmp, df.new) %>% arrange(USAGE_HOUR)
  }
  return(df.tmp)
}

# 100 steps was not enough to generate full table accurately.
steps <- 30000

for (annual.trend in c(.1, .25, .5, .75, 1)) {
  # Extend our dataset n weeks into future at specified annual trend
  df.week.1 <- ExtendWeek(data.subset, weeks=1, trend=annual.trend)
  df.week.2 <- ExtendWeek(data.subset, weeks=2, trend=annual.trend)
  df.week.4 <- ExtendWeek(data.subset, weeks=4, trend=annual.trend)
  df.week.8 <- ExtendWeek(data.subset, weeks=8, trend=annual.trend)

  # Filter out the actuals and only consider the future.
  df.forecast.1 <- df.week.1 %>% subset(USAGE_HOUR > max(data.subset$USAGE_HOUR))
  df.forecast.2 <- df.week.2 %>% subset(USAGE_HOUR > max(data.subset$USAGE_HOUR))
  df.forecast.4 <- df.week.4 %>% subset(USAGE_HOUR > max(data.subset$USAGE_HOUR))
  df.forecast.8 <- df.week.8 %>% subset(USAGE_HOUR > max(data.subset$USAGE_HOUR))

  # Find lowest cost SP on the actuals
  lowest.actuals <- findMinSPLevel.2(data.subset, steps)

  # Compute lowest cost on the forecast
  lowest.forecast.1 <- findMinSPLevel.2(df.forecast.1, steps)
  lowest.forecast.2 <- findMinSPLevel.2(df.forecast.2, steps)
  lowest.forecast.4 <- findMinSPLevel.2(df.forecast.4, steps)
  lowest.forecast.8 <- findMinSPLevel.2(df.forecast.8, steps)

  # Compute cost of the week1 forecast with our min SP level found from actuals
  cost.base.1 <- FindCost(df.forecast.1, lowest.actuals)
  cost.base.2 <- FindCost(df.forecast.2, lowest.actuals)
  cost.base.4 <- FindCost(df.forecast.4, lowest.actuals)
  cost.base.8 <- FindCost(df.forecast.8, lowest.actuals)
  cost.forecast.1 <- FindCost(df.forecast.1, lowest.forecast.1)
  cost.forecast.2 <- FindCost(df.forecast.2, lowest.forecast.2)
}
```

```

cost.forecast.4 <- FindCost(df.forecast.4, lowest.forecast.4)
cost.forecast.8 <- FindCost(df.forecast.8, lowest.forecast.8)

print(paste0(" Annual Trend: ", annual.trend,
  sprintf(" CostBase:1 %0.7f", cost.base.1), sprintf(" Cost1Wk: %0.7f", cost.forecast.1),
  sprintf(" CostBase:2 %0.7f", cost.base.2), sprintf(" Cost2Wk: %0.7f", cost.forecast.2),
  sprintf(" CostBase:4 %0.7f", cost.base.4), sprintf(" Cost4Wk: %0.7f", cost.forecast.4),
  sprintf(" CostBase:8 %0.7f", cost.base.8), sprintf(" Cost8Wk: %0.7f", cost.forecast.8)))
print(paste0(sprintf(" CostDelta.1: %0.7f", cost.base.1 - cost.forecast.1),
  sprintf(" CostDelta.2: %0.7f", cost.base.2 - cost.forecast.2),
  sprintf(" CostDelta.4: %0.7f", cost.base.4 - cost.forecast.4),
  sprintf(" CostDelta.8: %0.7f", cost.base.8 - cost.forecast.8)))

print("")
}

## [1] " Annual Trend: 0.1 CostBase:1 14494.5175470 Cost1Wk: 14494.4153616 CostBase:2 29093.9118974 Co
## [1] " CostDelta.1: 0.1021854 CostDelta.2: 0.1021859 CostDelta.4: 0.1021858 CostDelta.8: 3.8253154"
## [1] ""
## [1] " Annual Trend: 0.25 CostBase:1 14515.0953006 Cost1Wk: 14514.8558493 CostBase:2 29170.9914238 Co
## [1] " CostDelta.1: 0.2394513 CostDelta.2: 0.2394511 CostDelta.4: 3.5393320 CostDelta.8: 32.8397205"
## [1] ""
## [1] " Annual Trend: 0.5 CostBase:1 14544.5000878 Cost1Wk: 14544.0776849 CostBase:2 29282.9114239 Co
## [1] " CostDelta.1: 0.4224029 CostDelta.2: 1.9681066 CostDelta.4: 14.0702544 CostDelta.8: 141.3641222
## [1] ""
## [1] " Annual Trend: 0.75 CostBase:1 14569.4128531 Cost1Wk: 14568.8907534 CostBase:2 29378.9036382 Co
## [1] " CostDelta.1: 0.5220997 CostDelta.2: 4.0283987 CostDelta.4: 30.5086578 CostDelta.8: 283.2709930
## [1] ""
## [1] " Annual Trend: 1 CostBase:1 14591.0313988 Cost1Wk: 14590.4228362 CostBase:2 29463.1189561 Cost2
## [1] " CostDelta.1: 0.6085625 CostDelta.2: 6.1130679 CostDelta.4: 51.8426906 CostDelta.8: 446.4012950
## [1] ""

for (weeks in c(1, 2, 4,8)) { #} c(1,2,4,8)) {
  for (annual.trend in c(.1, .25, .5, .75, 1)) {
    # Extend our dataset n weeks into future at specified annual trend
    df.tmp <- ExtendWeek(data.subset, weeks=weeks, trend=annual.trend)
    # Find lowest cost on the actuals
    lowest.actuals <- findMinSPLevel.2(data.subset, steps)
    # Filter out the actuals and only consider the future.
    df.forecast <- df.tmp %>% subset(USAGE_HOUR > max(data.subset$USAGE_HOUR))
    # Compute lowest cost on the forecast
    lowest.forecast <- findMinSPLevel.2(df.forecast, steps)
    # Compute cost of the forecast with our min SP level found from actuals
    cost.base <- FindCost(df.forecast, lowest.actuals)
    # Compute cost of the forecast with our min SP level found from forecast
    cost.forecast <- FindCost(df.forecast, lowest.forecast)
    #
    extra.cost <- ((1000000*cost.base) / cost.forecast) - 1000000
    #extra.cost <- ((cost.1 / cost.forecast) * 1000000) - 1000000
    print(paste0(" Weeks: ", weeks, sprintf(" MaxVal: %0.1f", max(df.tmp$NORM_USAGE)), sprintf(" Trend:
      sprintf(" LowActuals: %0.3f", lowest.actuals),
      sprintf(" LowForecast: %0.7f", lowest.forecast),
      sprintf(" BaseCost: %0.3f", cost.base),
      sprintf(" Cost2: %0.3f", cost.forecast),

```

```

        sprintf(" Extra Cost: %0.4f", extra.cost),
        sprintf(" CostDelta: %0.7f", (cost.base - cost.forecast)),
        #sprintf(" CostDeltaPct: %0.4f", 100*(cost.base - cost.forecast)/cost.base),
        sprintf(" CostPM: %0.4f", (1000000/cost.base) * (cost.base-cost.forecast)))
    }
    print("")
}

```

```

## [1] " Weeks: 1 MaxVal: 100.1 Trend: 0.10 LowActuals: 78.215 LowForecast: 78.3176196 BaseCost: 14494.
## [1] " Weeks: 1 MaxVal: 100.3 Trend: 0.25 LowActuals: 78.215 LowForecast: 78.4548855 BaseCost: 14515.
## [1] " Weeks: 1 MaxVal: 100.6 Trend: 0.50 LowActuals: 78.215 LowForecast: 78.6378371 BaseCost: 14544.
## [1] " Weeks: 1 MaxVal: 100.8 Trend: 0.75 LowActuals: 78.215 LowForecast: 78.7375339 BaseCost: 14569.
## [1] " Weeks: 1 MaxVal: 101.0 Trend: 1.00 LowActuals: 78.215 LowForecast: 78.8239977 BaseCost: 14591.
## [1] ""
## [1] " Weeks: 2 MaxVal: 100.3 Trend: 0.10 LowActuals: 78.215 LowForecast: 78.3176205 BaseCost: 29093.
## [1] " Weeks: 2 MaxVal: 100.7 Trend: 0.25 LowActuals: 78.215 LowForecast: 78.4548853 BaseCost: 29170.
## [1] " Weeks: 2 MaxVal: 101.3 Trend: 0.50 LowActuals: 78.215 LowForecast: 78.6378367 BaseCost: 29282.
## [1] " Weeks: 2 MaxVal: 101.9 Trend: 0.75 LowActuals: 78.215 LowForecast: 78.8217093 BaseCost: 29378.
## [1] " Weeks: 2 MaxVal: 102.3 Trend: 1.00 LowActuals: 78.215 LowForecast: 79.0934808 BaseCost: 29463.
## [1] ""
## [1] " Weeks: 4 MaxVal: 100.7 Trend: 0.10 LowActuals: 78.215 LowForecast: 78.3176200 BaseCost: 58372.
## [1] " Weeks: 4 MaxVal: 101.6 Trend: 0.25 LowActuals: 78.215 LowForecast: 78.7251534 BaseCost: 58675.
## [1] " Weeks: 4 MaxVal: 102.9 Trend: 0.50 LowActuals: 78.215 LowForecast: 79.2517125 BaseCost: 59120.
## [1] " Weeks: 4 MaxVal: 104.1 Trend: 0.75 LowActuals: 78.215 LowForecast: 79.6677866 BaseCost: 59508.
## [1] " Weeks: 4 MaxVal: 105.1 Trend: 1.00 LowActuals: 78.215 LowForecast: 80.0207646 BaseCost: 59855.
## [1] ""
## [1] " Weeks: 8 MaxVal: 101.4 Trend: 0.10 LowActuals: 78.215 LowForecast: 78.5368608 BaseCost: 117257
## [1] " Weeks: 8 MaxVal: 103.4 Trend: 0.25 LowActuals: 78.215 LowForecast: 79.3202076 BaseCost: 118483
## [1] " Weeks: 8 MaxVal: 106.2 Trend: 0.50 LowActuals: 78.215 LowForecast: 80.2564811 BaseCost: 120346
## [1] " Weeks: 8 MaxVal: 108.6 Trend: 0.75 LowActuals: 78.215 LowForecast: 81.2114315 BaseCost: 122027
## [1] " Weeks: 8 MaxVal: 110.8 Trend: 1.00 LowActuals: 78.215 LowForecast: 81.8779822 BaseCost: 123562
## [1] ""

```

```

df.tmp <- ExtendWeek(data.subset, weeks=1, trend=0.05)
lowest.1 <- findMinSPLevel(data.subset, 100)
df.forecast <- df.tmp %>% subset(USAGE_HOUR > max(data.subset$USAGE_HOUR))
lowest.forecast <- findMinSPLevel(df.forecast, 100)
cost.1 <- FindCost(df.forecast, lowest.1)
cost.forecast <- FindCost(df.forecast, lowest.forecast)
cost.1

```

```
## [1] 14487.02
```

```
cost.forecast
```

```
## [1] 14487.01
```

3.1.1 Pull in PlotBoxes from optimization.Rmd

```

PlotBoxes <- function(df, sp.level=NULL, title=FALSE, ylim=NULL) {
  days <- c("S", "M", "T", "W", "R", "F", "S")
  p <- ggplot(df) +
    geom_rect_pattern(aes(xmin=xmin, xmax=xmax, ymin=ymin, ymax=ymax,
      fill=Pricing, pattern=Pricing),
      colour=NA, pattern_size=0.25,

```



```

        pattern_spacing=0.02) +
  theme_bw() +
  theme(axis.text=element_text(size=15),
        axis.title=element_text(size=15),
        legend.text = element_text(size=15),
        legend.title = element_blank(),
        legend.position="bottom",
        strip.text=element_text(size=15)) +
  ylab("Cost") + xlab("") +
  scale_pattern_manual(values=c("none", "none", "stripe")) +
  scale_x_continuous(breaks=seq(from=as.POSIXct("2024-06-02 12:00:00"),
                                to=as.POSIXct("2024-06-15 12:00:00"),
                                length.out=14),
                    labels=rep(days,2)) +
  scale_fill_manual(values=c("orange", "darkgreen", "red"))
if (title) {
  if (!is.null(sp.level)) {
    cost.premium <- TotalCostPremium(df)
    p <- p + ggtitle(paste0("c=", round(sp.level, 1), " ", "C(c)=", round(cost.premium, 0)))
  }
}
if (!is.null(ylim)) {
  p <- p+ylim(ylim)
}
return(p)
}

```

3.2 Figure 6

We create a new data frame from our existing week extended out 8 weeks with 100% annual trend and compare the SP level that would be set on the initial data vs the 8-week forecast.

```

df.tmp<-ExtendWeek(data.subset, weeks=8, trend=1)
lowest.1 <- findMinSPLevel(data.subset, 100)
df.forecast <- df.tmp %>% subset(USAGE_HOUR > max(data.subset$USAGE_HOUR))
lowest.forecast <- findMinSPLevel(df.forecast, 100)
cost.1 <- FindCost(df.forecast, lowest.1)
cost.forecast <- FindCost(df.forecast, lowest.forecast)
cost.1

## [1] 123557.4

cost.forecast

## [1] 123118.4

PlotBest.2 <- function(df, sp.level, annotated.level) {
  df.ann <- AnnotateSPRILevel(df, sp.level)
  df.bboxes <- GenerateBoxes(df.ann)
  brks <- seq(from=min(df.bboxes$xmin), to=max(df.bboxes$xmax), by=(60*60*24*7))
  p <- PlotBoxes(df.bboxes)
  p <- p +
    scale_x_continuous(breaks=brks, labels=1:length(brks)) +
    geom_hline(aes(yintercept=sp.level,
                  linetype="Min Cost Commitment for 1-week"),
              color="black") +

```

```

geom_hline(aes(yintercept=annotated.level,
               linetype="Min Cost Commitment for 8-weeks"),
           color="red") +
scale_linetype_manual(name="Commitment Level",
                      values=c("dashed", "dashed"),
                      guide=guide_legend(
                        position="top",
                        direction="vertical",
                        override.aes = list(color=c("black", "red")))) +

  xlab("Week")
return(p)
}

p <- PlotBest.2(df.forecast, lowest.1, lowest.forecast)

## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
p

```

