Modelling and pricing weather derivatives (NY)

Mengyu Huang

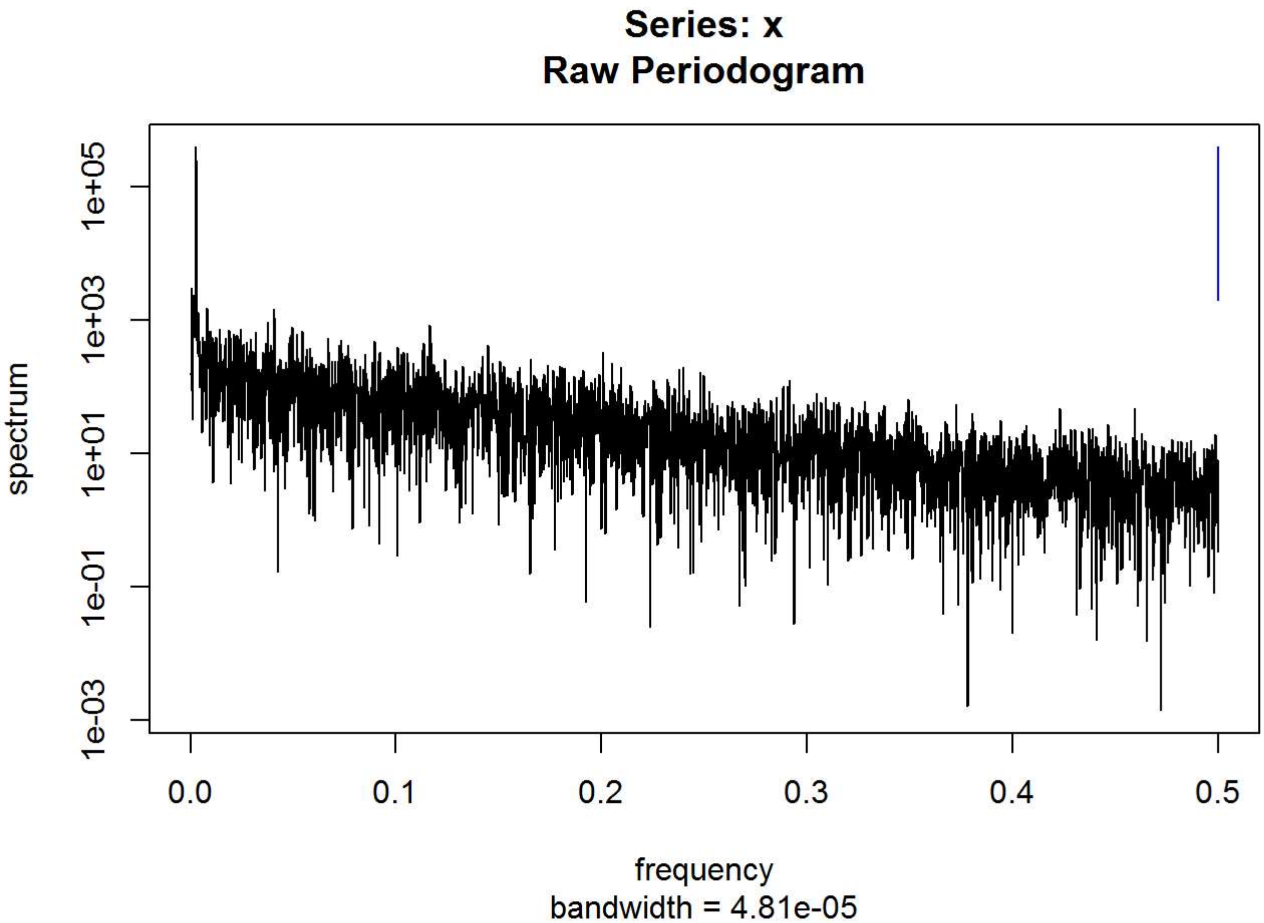
April 14, 2017

First part Modelling for New York temperature 1. Read in the data

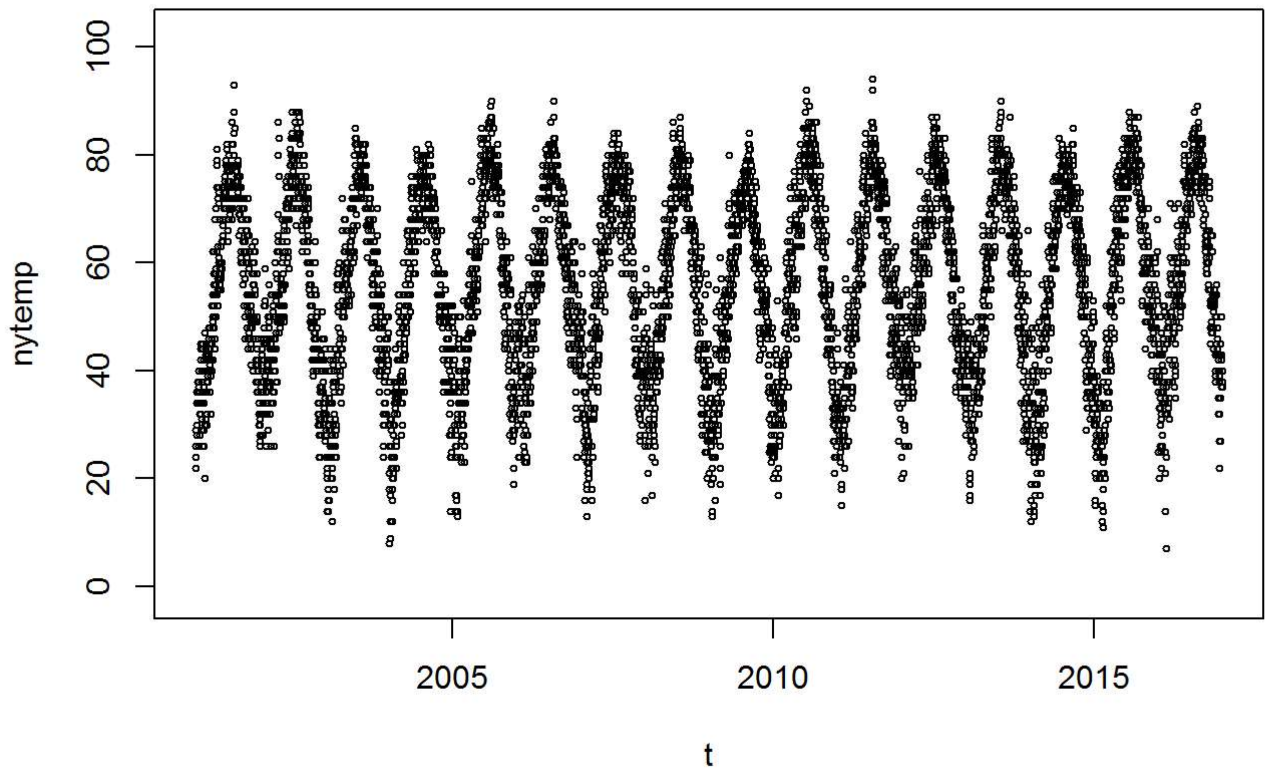
```
setwd("C:/programming projects-for github/R projects/Temperature Modelling and Weather Derivatives Pricin
g")

ny <- read.csv("nyc_daily_2001_2016.csv")
colnames(ny)[1]<-"date"



ny$date<-as.Date(ny$date,"%m/%d/%Y")

nytemp<-as.numeric(as.character(ny$avg))
```

```
## Warning: NAs introduced by coercion
```

```
t<-ny$date

# linear interpolation for NA values
nytemp[4046] = nytemp[4045]+(nytemp[4048]-nytemp[4045])/3
nytemp[4047] = nytemp[4045]+(nytemp[4048]-nytemp[4045])*2/3

# calculate per
ssp <- spectrum(nytemp)
```




```
per <- 1/ssp$freq[ssp$spec==max(ssp$spec)]


# plot to check the difference
rg <- diff(range(nytemp, na.rm = TRUE), na.rm=TRUE)
plot(nytemp~t,ylim=c(min(nytemp,na.rm=TRUE)-0.1*rg,max(nytemp,na.rm=TRUE)+0.1*rg),cex=0.5)
```



2. run linear/ non-linear fitting

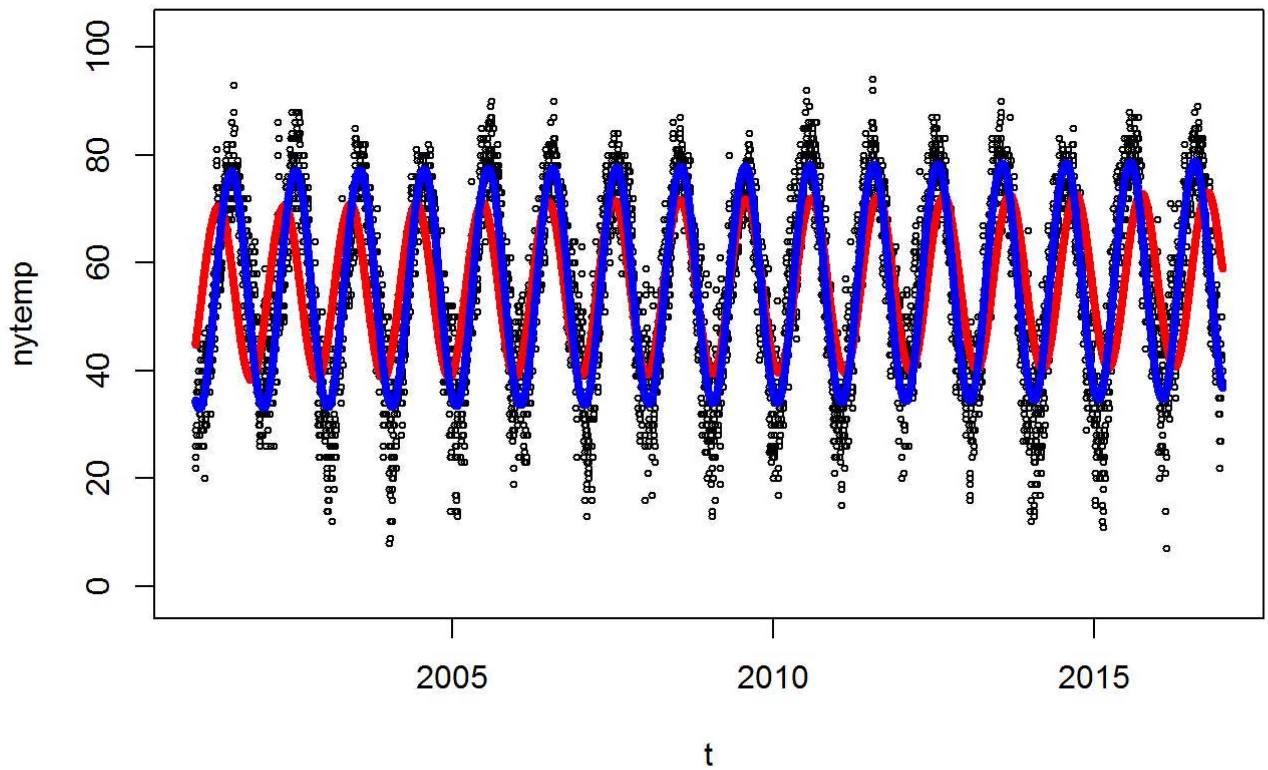
```
# including the trend term t
# fit linear model
dates = 1:5844
reslm2 <- lm(nytemp ~ dates+sin(2*pi/per*dates)+cos(2*pi/per*dates))#+sin(4*pi/per*t)+cos(4*pi/per*t))
summary(reslm2)
```

```
##
## Call:
## lm(formula = nytemp ~ dates + sin(2 * pi/per * dates) + cos(2 *
##     pi/per * dates))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -40.769  -9.296   0.837   9.410  37.324
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             54.2310076  0.3378848 160.501  < 2e-16 ***
## dates                    0.0004689  0.0001001   4.683 2.89e-06 ***
## sin(2 * pi/per * dates) 12.8961512  0.2394656  53.854  < 2e-16 ***
## cos(2 * pi/per * dates) -9.6925227  0.2383546 -40.664  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.91 on 5840 degrees of freedom
## Multiple R-squared:  0.4388, Adjusted R-squared:  0.4385
## F-statistic:  1522 on 3 and 5840 DF,  p-value: < 2.2e-16
```

```
t_mean<-function(t){
  t_mean <- 54.2310 + 0.0004689*t +12.8961*sin(2*pi/per * t) - 9.6925*cos(2*pi/per * t)
  return (t_mean)
}

fitmean <- t_mean(dates)
plot(nytemp~t,ylim=c(min(nytemp,na.rm=TRUE)-0.1*rg,max(nytemp,na.rm=TRUE)+0.1*rg),cex=0.5)
lines(fitmean~t,col='red',lwd=4)    # solid red line is periodic with second harmonic


# fit non-linear model
W = 2*pi/per
reslm3 <- nls(nytemp ~ cons+A*dates+B*sin(W*dates)+C*cos(W*dates), start=list(cons=54.2312, A=0.0004681,
B=12.900,W=W,C=-9.6935))#+sin(4*pi/per*t)+cos(4*pi/per*t))
summary(reslm3)
```

```
##
## Formula: nytemp ~ cons + A * dates + B * sin(W * dates) + C * cos(W *
##     dates)
##
## Parameters:
##        Estimate Std. Error  t value Pr(>|t|)
## cons  5.492e+01  1.939e-01  283.177  < 2e-16 ***
## A     3.336e-04  5.748e-05    5.803 6.85e-09 ***
## B    -8.459e+00  2.594e-01  -32.611  < 2e-16 ***
## W     1.720e-02  3.700e-06 4649.471  < 2e-16 ***
## C    -2.028e+01  1.659e-01 -122.230  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.404 on 5839 degrees of freedom
##
## Number of iterations to convergence: 5
## Achieved convergence tolerance: 1.314e-06
```

```
# the fitted model could be written into: T = 54.92 + 0.0003318*t -8.461*sin(0.0172 * t) - 20.029*cos(0.0
172 * t) + sigma1 * W(t)
t_mean2<-function(t){
  t_mean2 <- 54.92 + 0.0003336*t -8.459*sin(0.0172 * t) - 20.28*cos(0.0172 * t)
  return (t_mean2)
}
fitmean2 <- t_mean2(dates)
lines(fitmean2~t,col='blue',lwd=4)
```



3. Run Monte Carlo simulation

```
## RUN SIMULATION !!!
v = sd(diff(nytemp))

# number of simulation paths
N = 1000

# change the simulation period you want

orit = as.Date("2001-01-01",format="%Y-%m-%d")
startt = as.Date("2016-03-01",format="%Y-%m-%d")
finalt = as.Date("2016-03-31",format="%Y-%m-%d")
# set the step to be 1
dt = 1
time1 = as.numeric(startt-orit)+1
time2= as.numeric(finalt-orit)+1


df<- data.frame(matrix(NA, nrow = time2-time1+3, ncol= N))

df[1,]= t_mean2(time1)
#HDD
df[time2-time1+2,] = max(65-t_mean2(time1),0)
#CDD
df[time2-time1+3,] = max(t_mean2(time1)-65,0)


for(a in 1:N){
  for(b in 1: (time2-time1)){
      dw = rnorm(1)*1;
      temp = df[b,a]
      df[b+1,a] = temp+(t_mean2(b+time1)-t_mean2(b+time1-1))+v*dw
      # HDD
      df[time2-time1+2,a] = df[time2-time1+2,a] + max(65-df[b+1,a],0)
      # CDD
      df[time2-time1+3,a] = df[time2-time1+3,a] + max(df[b+1,a]-65,0)
  }
}

finalsim<- rowMeans(df[1:(time2-time1+1),])
# seasonal HDD Nov to Mar
finalHDD <- rowMeans(df[time2-time1+2,])
# seasonal CDD May to Sep
finalCDD<- rowMeans(df[time2-time1+3,])

as.numeric(finalHDD)
```

```
## [1] 754.3689
```

```
as.numeric(finalCDD)
```

```
## [1] 69.33543
```