# project4

### *tammy*

### *Monday, October 20, 2014*

```
setwd("E:/courseworks/Infrastructure planning/project4")
library(knitr)
library(lubridate)
library(plyr)
```

```
##
## Attaching package: 'plyr'
##
## The following object is masked from 'package:lubridate':
##
##     here
```

```
library(ggplot2)
```

```
#Part A


# Load in data files
demand.mw <- read.csv("demand.csv",stringsAsFactors = FALSE) # Regional demands
gen.info <- read.csv("gen.info.csv") # Generator information
data <- read.csv("wind.sites.csv") # Wind site info
```

```
sites.wind.power.per <- read.csv("sites.wind.power.per.csv",stringsAsFactors = FALSE) # Hourly wind power p
er turbine at each site
trans.limits <- read.csv("trans.limits.csv") # Transmission limits between regions
```

```
## Warning: incomplete final line found by readTableHeader on
## 'trans.limits.csv'
```

```
# define a few objects that will make things easier to generalize later
n.sites <- nrow(data)
n.hrs <- nrow(demand.mw)
n.regions <- nrow(gen.info)
n.hrs  #8760
```

```
## [1] 8760
```

```
n.sites   #66
```

```
## [1] 66
```

```
nrow(sites.wind.power.per)
```

```
## [1] 8760
```

```
#1. Calculate the baseload generation for each region and the state as a whole [MWh]
```

```
west.base.gen<-sum(gen.info[1,2:4])*1000
east.base.gen<-sum(gen.info[2,2:4])*1000
donwstate.base.gen<-sum(gen.info[3,2:4])*1000
whole.base.gen<-sum(west.base.gen,east.base.gen,donwstate.base.gen)
```

```
#2.max capacity of other generation[MW]!!!! Per hour!!! The sun of import and nonbase generation
ImportLimits<-read.csv("import.limits.csv",header=TRUE)
```

```
## Warning: incomplete final line found by readTableHeader on
## 'import.limits.csv'
```

```
MaxCapOtherWest<-gen.info[1,5]+sum(ImportLimits[1,2:5])
MaxCapOtherEast<-gen.info[2,5]+sum(ImportLimits[2,2:5])
MaxCapOtherDownstate<-gen.info[3,5]+sum(ImportLimits[3,2:5])
MaxCapOtherWhole<-sum(MaxCapOtherWest,MaxCapOtherEast,MaxCapOtherDownstate)
```

```
#3.uncutailed capacity factor
#per means per turbine
data<-read.csv("wind.sites.csv")
sites.wind.power.per<-read.csv("sites.wind.power.per.csv", header=TRUE)
data$TatalAnnPer<-colSums(sites.wind.power.per[,2:67])
n<-nrow(sites.wind.power.per)
data$UncurtailedCapFactor<-data$TatalAnnPer/(3*n)
```

```
#4.Annual Potential wind-generated electricity, create a table of top 10UCF
data$TotalAnnPotential<-data$TatalAnnPer*data$n.t.max
```

```
Order_max <- order(data$TotalAnnPotential, decreasing = TRUE)
data.top10<-data$TotalAnnPotential[Order_max[1:10]]
DataTop10<-data.frame(Sites=1:10,Potentialoutput=data.top10,MaxTurbineNo=data$n.t.max[Order_max[1:10]])
print(DataTop10)
```

```
##    Sites Potentialoutput MaxTurbineNo
## 1      1         4448163          542
## 2      2         3467674          412
## 3      3         3370828          357
## 4      4         2833303          331
## 5      5         2109830          257
## 6      6         2001361          215
## 7      7         1698449          203
## 8      8         1648955          188
## 9      9         1604801          184
## 10    10         1391342          171
```

```
#5 Plot the annual demand profile for each region and the state as a whole
demand.mw <- read.csv("demand.csv",stringsAsFactors = FALSE, header=TRUE)
demand.mw$date.time <- as.POSIXct(demand.mw$date.time,tz="UTC",format="%m/%d/%y %H:%M")
demand.mw$whole<-demand.mw$west+demand.mw$east+demand.mw$downstate
sum(is.na(demand.mw))
```

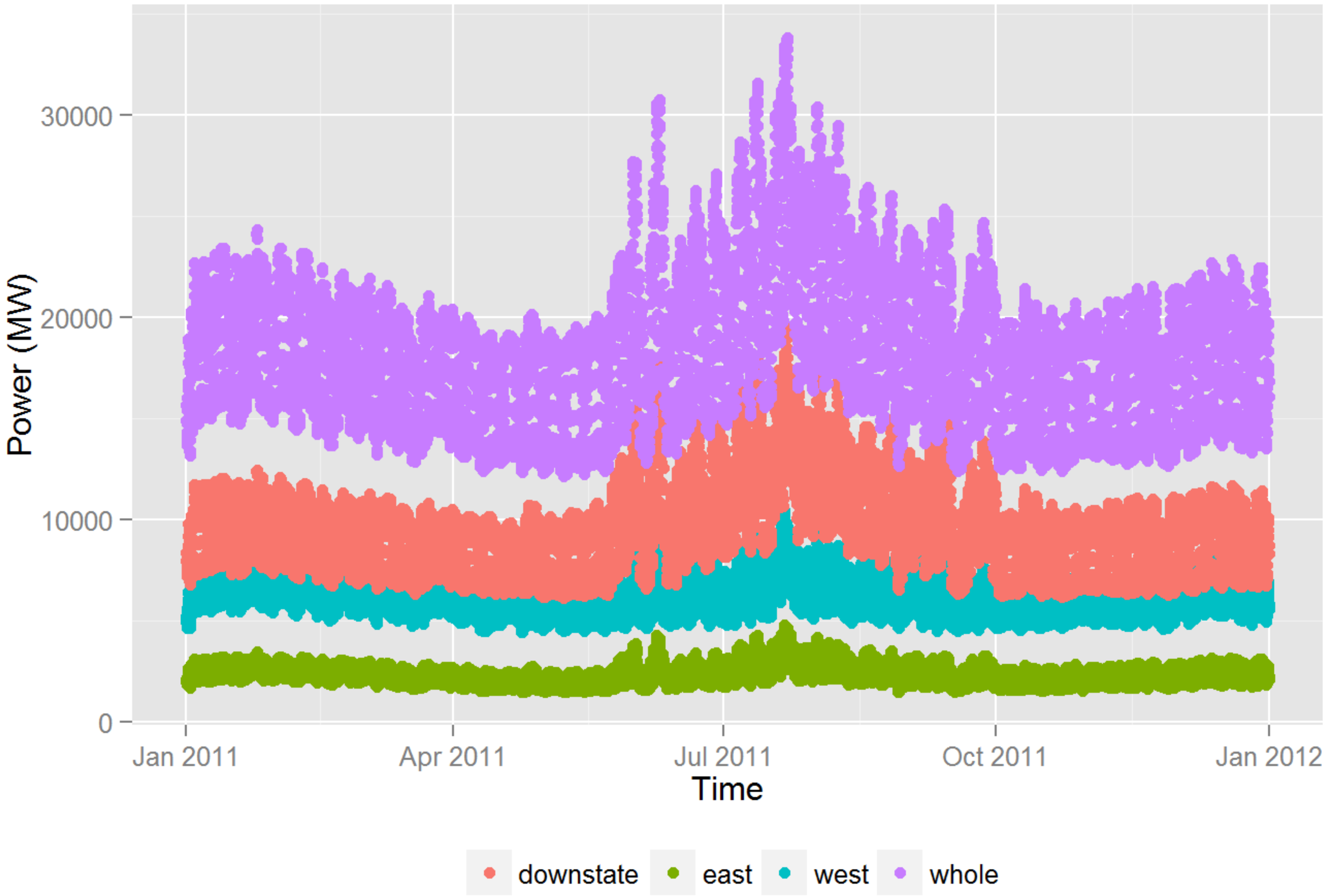
```
## [1] 0
```

```
#check and replace missing values
```

```r
demand.mw[,2:ncol(demand.mw)] <- rapply(demand.mw[,2:ncol(demand.mw)], f=function(x) ifelse(is.nan(x),NA,x)
, how="replace")
sum(is.na(demand.mw))
```

```
## [1] 0
```

```r
ggplot() + geom_point(data=demand.mw, aes(x=date.time, y=west, color="west")) +geom_point(data=demand.mw, a
es(x=date.time, y=east, color="east")) + geom_point(data=demand.mw,aes(x=date.time, y=downstate, color="dow
nstate")) + geom_point(data=demand.mw, aes(x=date.time, y=whole, color="whole")) +xlab("Time") +ylab("Power
(MW)") +theme(legend.position="bottom",legend.title=element_blank(),axis.title.x=element_text(),axis.title.
y=element_text())
```




```
#PartB
###Case1A
#rearrange the order of data by capacity factor from max to min
```

```
order_CF<-order(data$UncurtailedCapFactor,decreasing=TRUE)
data<-(data[order_CF,])
data$order<-c(1:66)
#View(data)
#calculate the accumulative max number of turbines for each site
#n<-data$order[2:66]
data$n.t.max.accu[1]<-50
for (n in 2:66){
  data$n.t.max.accu[n]<-data$n.t.max[n]+data$n.t.max.accu[n-1]
}
```

```
#total annual wind-generated electricity per turbine for each site
#hourly wind generated electricity
sites.wind.power.per$HourlyWIndGenElec<-rowSums(sites.wind.power.per[,2:67])
#calculate the uncurtailed capacity factor of the overall installation, when installation is at3MW, 3GW, 6GW, 9GW, 15GW

counter <- c(1,1000,2000,3000,4000,5000)
n.t.last = NULL
TotalWindGenElec= NULL
CF= NULL
for (i in 1:length(counter)){
  j<-1
  while (data$n.t.max.accu[j] <= counter[i]){
    j <- j+1
  }
  n.t.last[i] <- data$n.t.max[j]-(data$n.t.max.accu[j]-counter[i])
  #total potential wind-generated electricity for every capacity
```

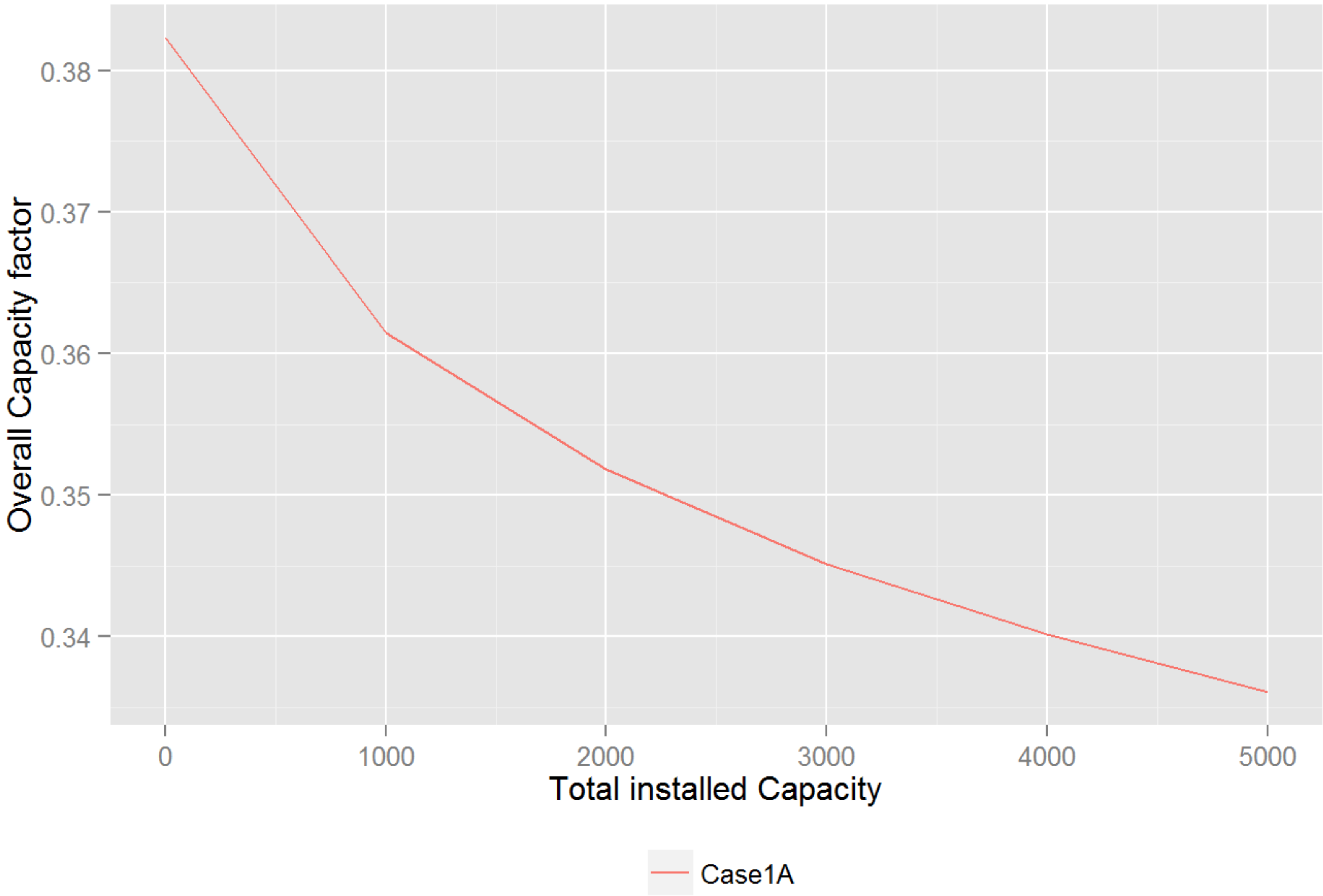
```
 TotalWindGenElec[i]<-sum(data$TotalAnnPotential[1:j-1])+data$TatalAnnPer[j]*(n.t.last[i])
#overall capacity factor
CF[i]<-((sum(data$TotalAnnPotential[1:j-1]))+data$TatalAnnPer[j]*(n.t.last[i]))/(3*8760*counter[i])
}
TotalWindGenElec
```

```
## [1]    10048  9498670 18493407 27210710 35755744 44160322
```

```
CF
```

```
## [1] 0.3823 0.3614 0.3519 0.3451 0.3401 0.3361
```

```
#set up a data frame of Case1A
Case1A<-data.frame(counter[1:6],TotalWindGenElec,CF)
#View(Case1A)
#plot the overall capacity factor vs. total installed capacity
ggplot() + geom_line(data=Case1A, aes(x=counter[1:6], y=CF, color="Case1A")) +xlab("Total installed Capacit
y") +ylab("Overall Capacity factor") +theme(legend.position="bottom",legend.title=element_blank(),axis.titl
e.x=element_text(),axis.title.y=element_text())
```




```
#Determine the placement of wind turbines that produces the highest total wind-generated electricity for th
e year
max<-max(TotalWindGenElec)
```

```
Order_max2<- order(Case1A$TotalWindGenElec, decreasing = TRUE)
#Create a new data frame as the TotalWindGenElec decrease
Case1A2<-Case1A[Order_max2,]
Site.MaxTotalWindGenElec<-Case1A2[1,]
print(Site.MaxTotalWindGenElec)
```

```
##   counter.1.6. TotalWindGenElec     CF
## 6         5000         44160322 0.3361
```

```
#a data frame containing turbines distribution at each site for 9GW
i<-3000
j<-1
  while (data$n.t.max.accu[j] <= i){
    j <- j+1
  }
n.t.last <- data$n.t.max[j]-(data$n.t.max.accu[j]-i)
n.t.max<-data$n.t.max
n.t.3000<-data.frame(data$site,data$n.t.max)
n.t.3000[j,2]<-n.t.last
n.t.3000[j+1:66,2]<-0
colnames(n.t.3000)<-c("SiteNo","NumberofTurbine")
TurbineDistCase1A<-n.t.3000
```

```
#Get the original order of TurbineDistCase1A
OrderO<-order(data$site)
TurbineDistCase1A<-TurbineDistCase1A[OrderO,]
```

```
###Case1B, Consider the actual demand
#rearrange the order of sites.wind.power.per by capacity factor from max to min
sites.wind.power.per <- read.csv("sites.wind.power.per.csv",stringsAsFactors = FALSE)
#View(sites.wind.power.per)
#Because there are only 66 rows in "data", so we have to remove the dateTime column in order to use the ori
ginal order of capacity factor to rearrange sites.wind.power .per in the order of uncurtailed capacity fact
or
sites.wind.power.per<-sites.wind.power.per[,2:67]
#View(sites.wind.power.per)
#order of the sites
order<-data$site
sites.wind.power.per<-sites.wind.power.per[,order]
#View(sites.wind.power.per)

#Add date.time again
for.date.time<-read.csv("sites.wind.power.per.csv",stringsAsFactors = FALSE)
date.time<-for.date.time$date.time
sites.wind.power.per$date.time<-date.time
```

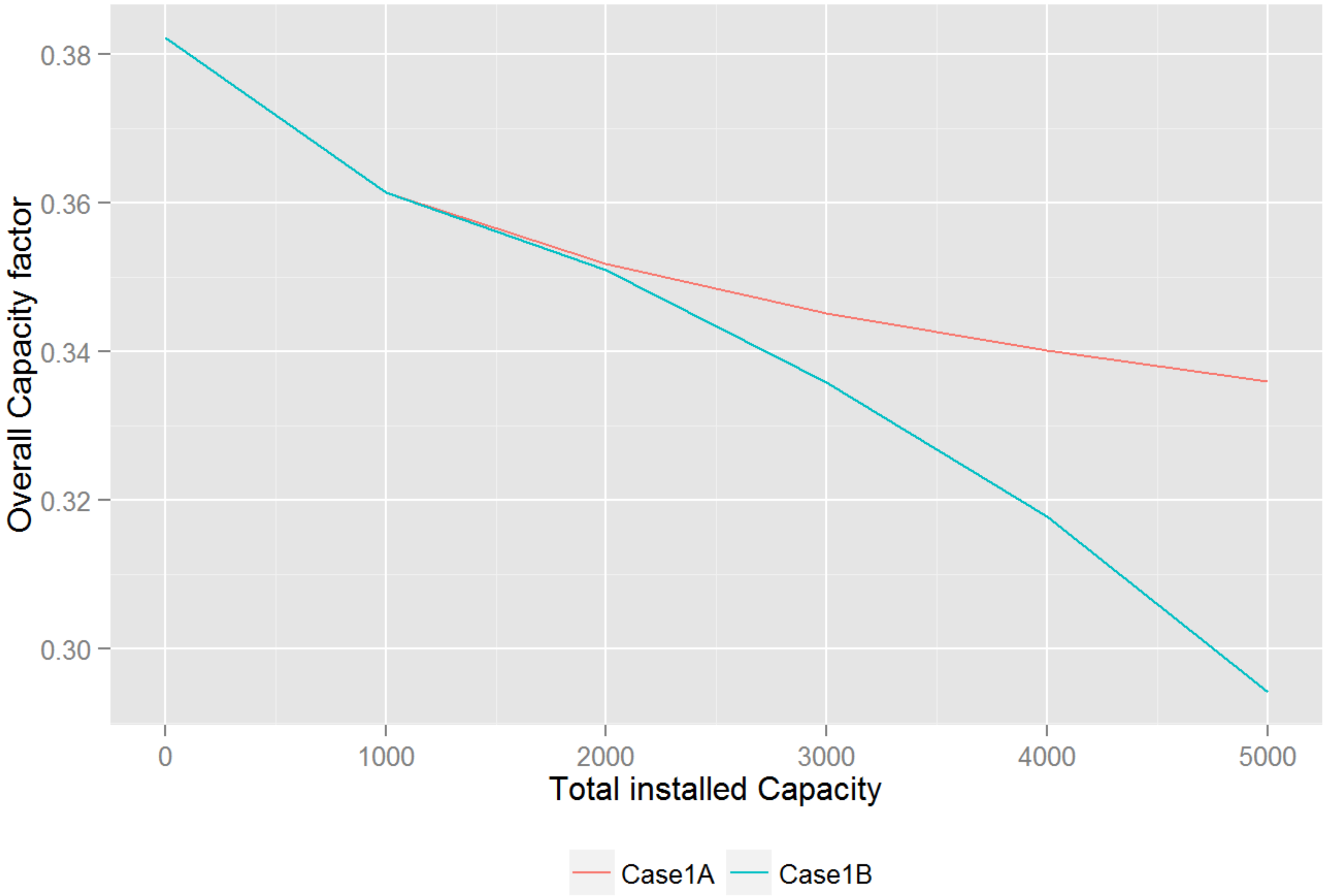
```
#View(demand.mw)
#calculate the new CF for 1B
#first, calculate hrly baseload, use the whole devide by 8760, same for every hour
hrly.baseload<-whole.base.gen/n.hrs
#Then find the hrly demand,8760 items.
hrly.demand<-demand.mw$whole

counter <- c(1,1000,2000,3000,4000,5000)
```

```
CFb=NULL
for (i in 1:length(counter)){
 j <- 1
  while (data$n.t.max.accu[j] <= counter[i]){
    j <- j+1
  }
  n.t.last[i] <- data$n.t.max[j]-(data$n.t.max.accu[j]-counter[i])

##create a data frame for Case1B
##Case1B<-data.frame(hrly.Wind.Pot,demand.mw$whole,hrly.baseload)
#multiply the n.t.max by the hourly potential power
sites.wind.power<-sites.wind.power.per
for(m in 1:66){
  sites.wind.power[,m]<-sites.wind.power.per[,m]*data$n.t.max[m]
}
#Create a dataframe Case1B
SitesWithNtmax<-rowSums(sites.wind.power[,1:j-1])
LastSite<-n.t.last[i]*sites.wind.power.per[,j]
Case1B<-data.frame(SitesWithNtmax,LastSite)
hrly.Wind.Pot<-Case1B$SitesWithNtmax+Case1B$LastSite

Case1B$hrly.Wind.Pot<-hrly.Wind.Pot

hrly.Wind.Util<-pmin(hrly.Wind.Pot,hrly.demand-hrly.baseload)

Total.Wind.Util<-sum(hrly.Wind.Util)

#overall capacity factor
```

```
CFb[i]<-Total.Wind.Util/(3*8760*counter[i])
}
CFb
```

```
## [1] 0.3823 0.3614 0.3510 0.3359 0.3179 0.2942
```

```
#plot the overall capacity factor vs. total installed capacity for both case1A&B
ggplot() + geom_line(data=Case1A, aes(x=counter[1:6], y=CF, color="Case1A")) + geom_line(data=Case1B, aes(x
=counter[1:6], y=CFb, color="Case1B")) +xlab("Total installed Capacity") +ylab("Overall Capacity factor") +
theme(legend.position="bottom",legend.title=element_blank(),axis.title.x=element_text(),axis.title.y=elemen
t_text())
```




```
#a data frame containing turbines distribution at each site for 9GW
i<-3000
j<-1
```

```r
  while (data$n.t.max.accu[j] <= i){
    j <- j+1
  }
n.t.last <- data$n.t.max[j]-(data$n.t.max.accu[j]-i)
n.t.max<-data$n.t.max
n.t.3000<-data.frame(data$site,data$n.t.max)
n.t.3000[j,2]<-n.t.last
n.t.3000[j+1:66,2]<-0
colnames(n.t.3000)<-c("SiteNo","NumberofTurbine")
TurbineDistCase1B<-n.t.3000
```

```r
#Get the original order of TurbineDistCase1A
OrderO<-order(data$site)
TurbineDistCase1B<-TurbineDistCase1B[OrderO,]
```

```r
#Case2
library(lpSolveAPI)
##Identifying the number of the variables, constraints and bounded variables
##n of variable= number of turbines at each site+ hrly generation from other nonbase sources
n.variables<-n.sites + n.hrs
##number of constraints =1(total number turbines constraint)+energy balances???
n.cons<-1+n.hrs
##numbers of bounded variables=number of turbines at each site+hrly generation from nonbase source
n.bounds<-n.sites+n.hrs
```

```r
#hourly net demand
hrly.net.demand<-hrly.demand-hrly.baseload
```

```
#The sites.wind.power.per has been rearranged by uncurtailed CF, we need the original order.
sites.wind.power.per.original<-read.csv("sites.wind.power.per.csv")
#Create a for loop to evaluate different levels of wind penetration,
#Create a capacity vector,that is capacities.mw, which is counter in my case.
cf.wind<-NULL
turbine.distr.mat.2<-matrix(0,66,6)
counter <- c(1,1000,2000,3000,4000,5000)
for (i in 1:length(counter))
{
 #wind.cap.total.mw<-counter[i]
  #define total number of turbines
     N.t <-counter[i]
     #develop MILP

     #Create a linear program model with number of rows = number of constraints, number of columns = number
of variables
lp.case2 <- make.lp(nrow=n.cons, ncol=n.variables)

#Setting our linear program as a minimization problem
lp.control(lp.case2, sense="min")

#Determining coefficients of objective function (0 for everything and 1 for other generation)
set.objfn(lp.case2, obj=c(rep(0,n.sites),rep(1,n.hrs)))

#We need to force the number of turbine at each site to be an integer
set.type(lp.case2,c(1:n.sites),type="integer")

#We will be adding constraints by row(rather than adding variable coefficients by column)
```

```
#Turn on "row entry mode"- remember to turn it off after adding all constraints
row.add.mode(lp.case2,"on")

##Sum of number of turbines at all sites equals total number of turbines
add.constraint(lp.case2,xt=rep(1,n.sites),type="=",rhs=N.t,indices=c(1:n.sites))

#name a column in data
j=1
sites<-c(1:66)

##Energy balance
for(j in 1:n.hrs)
  {
  add.constraint(lp.case2,xt=c(sites.wind.power.per.original[j,1+sites],1),type=">=",rhs=hrly.demand[j]-hrl
y.baseload,indices=c(sites,(n.sites+j)))
}

##We are done adding constraints, so need to turn off row entry mode
row.add.mode(lp.case2,"off")
#Use the data with original order
dataOriginal<-read.csv("wind.sites.csv")
#set upper bounds on decision variables
set.bounds(lp.case2,lower=rep(0,n.variables),upper=c(dataOriginal$n.t.max,rep(MaxCapOtherWhole,n.hrs)))

#solve optimization problem
solve(lp.case2)
turbine.distr.2<-get.variables(lp.case2)[1:n.sites]
turbine.distr.mat.2[,i]<-turbine.distr.2
```

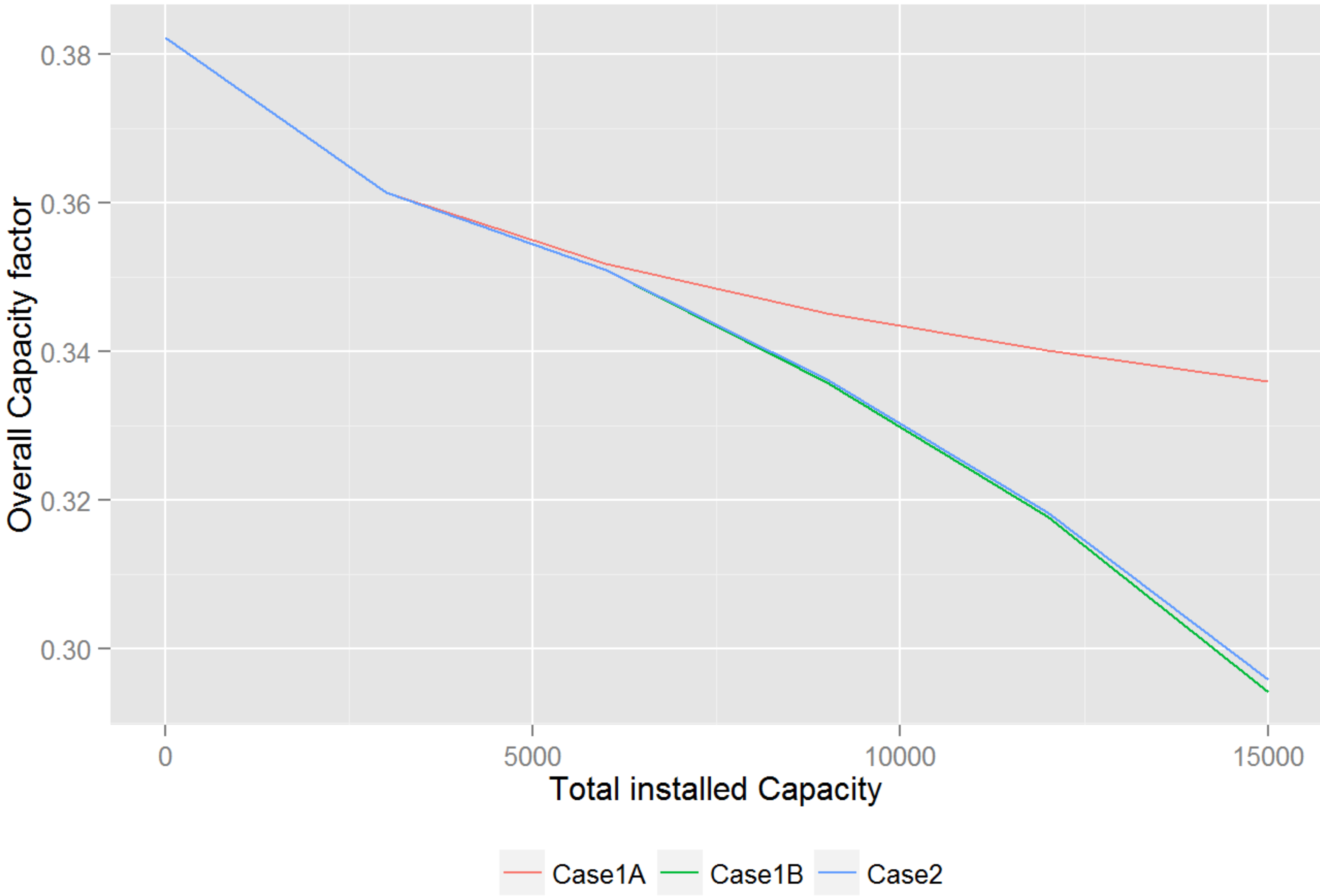
```
othergen.total.mwh<-get.objective(lp.case2)
cf.wind[i]<-(sum(demand.mw$whole)-othergen.total.mwh-hrly.baseload*n.hrs)/(N.t*8760*3)
#Actually hrly.gen*n.hrs=whole.base.gen

#Save the turbine distribution, othergen.total and transmission totals for each scenario
#assign(sprintf("turbine.distr.case2.%s",wind.cap.total.mw),turbine.distr)
#assign(sprintf("othergen.total.mwh.case2.%s",wind.cap.total.mw),othergen.total.mwh)
#assign(sprintf("cf.wind.case2.%s",wind.cap.total.mw),cf.wind)

#close for loop
}
cf.wind
```

```
## [1] 0.3823 0.3614 0.3510 0.3363 0.3184 0.2959
```

```
#set a data frame for Case2
Case2<- data.frame(x=N.t*3,y= cf.wind)

#View turbine distribution in Case2
colnames(turbine.distr.mat.2) <- c("3MW","3GW","6GW","9GW","12GW","15GW")
turbine.distr.mat.2[,4]
```

```
##  [1]   0  50  49 127  50  59 215  50 357  87 112  79 112  70 129  84  84
## [18]  47  53 112   0 188  58   0   0  29   0   0   0  75   0   0   0   0
## [35]   0   0   0 184   0   0   0   0   0   0   0   0   0   0   0   0   0
## [52]   0   0   0  49  65  78  49  50  60   0   0 108  81   0   0
```

```
#Plotting Case 1A and 1B and 2 together
#plot the overall capacity factor vs. total installed capacity for both case1A&B&2
ggplot() + geom_line(data=Case1A, aes(x=3*counter[1:6], y=CF, color="Case1A")) + geom_line(data=Case1B, aes
(x=3*counter[1:6], y=CFb, color="Case1B")) +geom_line(data=Case2, aes(x=3*counter[1:6],y=y,color="Case2"))+
xlab("Total installed Capacity") +ylab("Overall Capacity factor") +theme(legend.position="bottom",legend.ti
tle=element_blank(),axis.title.x=element_text(),axis.title.y=element_text())
```




```
#Case3
#read data
trans.limits<-read.csv("trans.limits.csv")
```

```
## Warning: incomplete final line found by readTableHeader on
## 'trans.limits.csv'
```

```
trans.west<-trans.limits$trans.limit.mw[which(trans.limits$from=="west")]
trans.east<-trans.limits$trans.limit.mw[which(trans.limits$from=="east")]
#sort sites by region, return the colume of sites' number
sites.west<-data$site[which(data$region=="west")]
sites.east<-data$site[which(data$region=="east")]
sites.downstate<-data$site[which(data$region=="downstate")]
#total number of sites in each region
n.sites.west<-length(sites.west)
n.sites.east<-length(sites.east)
n.sites.downstate<-length(sites.downstate)
#use these to determin the number of parameters in the optimization
```

```
##Identifying the number of the variables, constraints and bounded variables
##n of variable= number of turbines at each site+ hrly generation from other nonbase sources of west, east
and downstate region + transmission form west to east and from east to downstate
n.variables.3<-n.sites + 5*n.hrs
##number of constraints =1(total number turbines constraint)+energy balances for 3 regions
n.cons.3<-1+3*n.hrs
##numbers of bounded variables+number of turbines at each site+hrly generation from nonbase source
n.bounds.3<-n.variables.3

#Create a for loop to evaluate different levels of wind penetration:
cf.wind.3<-NULL
```

```
turbine.distr.mat.3<-matrix(0,66,6)
counter <- c(1,1000,2000,3000,4000,5000)
for (i in 1:length(counter))
 {
  #wind.cap.total.mw<-counter[i]
  #define total number of turbines
    N.t <-counter[i]
    #N.t <- 1
     #develop MILP

     #Create a linear program model with number of rows = number of constraints, number of columns = number
of variables
lp.case3 <- make.lp(nrow=n.cons.3, ncol=n.variables.3)

#Setting our linear program as a minimization problem
lp.control(lp.case3, sense="min")

#Determining coefficients of objective function (0 for everything and 1 for other generation)
set.objfn(lp.case3, obj=c(rep(0,n.sites),rep(1,3*n.hrs),rep(0,2*n.hrs)))

#We need to force the number of turbine at each site to be an integer
set.type(lp.case3,c(1:n.sites),type="integer")

#We will be adding constraints by row(rather than adding variable coefficients by column)
#Turn on "row entry mode"- remember to turn it off after adding all constraints
row.add.mode(lp.case3,"on")

##Sum of number of turbines at all sites equals total number of turbines
```

```
add.constraint(lp.case3,xt=rep(1,n.sites),type="=",rhs=N.t,indices=c(1:n.sites))

#name a column in data
j=1
sites<-c(1:66)

##Energy balance
for(j in 1:n.hrs)
  {
add.constraint(lp.case3,xt=c(sites.wind.power.per.original[j,1+sites.west],1,-1),type=">=",rhs=demand.mw[j,
"west"]-west.base.gen/8760,indices=c(sites.west,(n.sites+j),(n.sites+3*n.hrs+j)))
  }

for(j in 1:n.hrs)
  {
add.constraint(lp.case3,xt=c(sites.wind.power.per.original[j,1+sites.east],1,1,-1),type=">=",rhs=demand.mw[
j,"east"]-east.base.gen/8760,indices=c(sites.east,(n.sites+n.hrs+j),(n.sites+3*n.hrs+j),(n.sites+4*n.hrs+j)
))
}

for(j in 1:n.hrs)
  {
add.constraint(lp.case3,xt=c(sites.wind.power.per.original[j,1+sites.downstate],1,1),type=">=",rhs=demand.m
w[j,"downstate"]-donwstate.base.gen/8760,indices=c(sites.downstate,(n.sites+2*n.hrs+j),(n.sites+4*n.hrs+j))
)
}

#Turn off row entry mode:
```

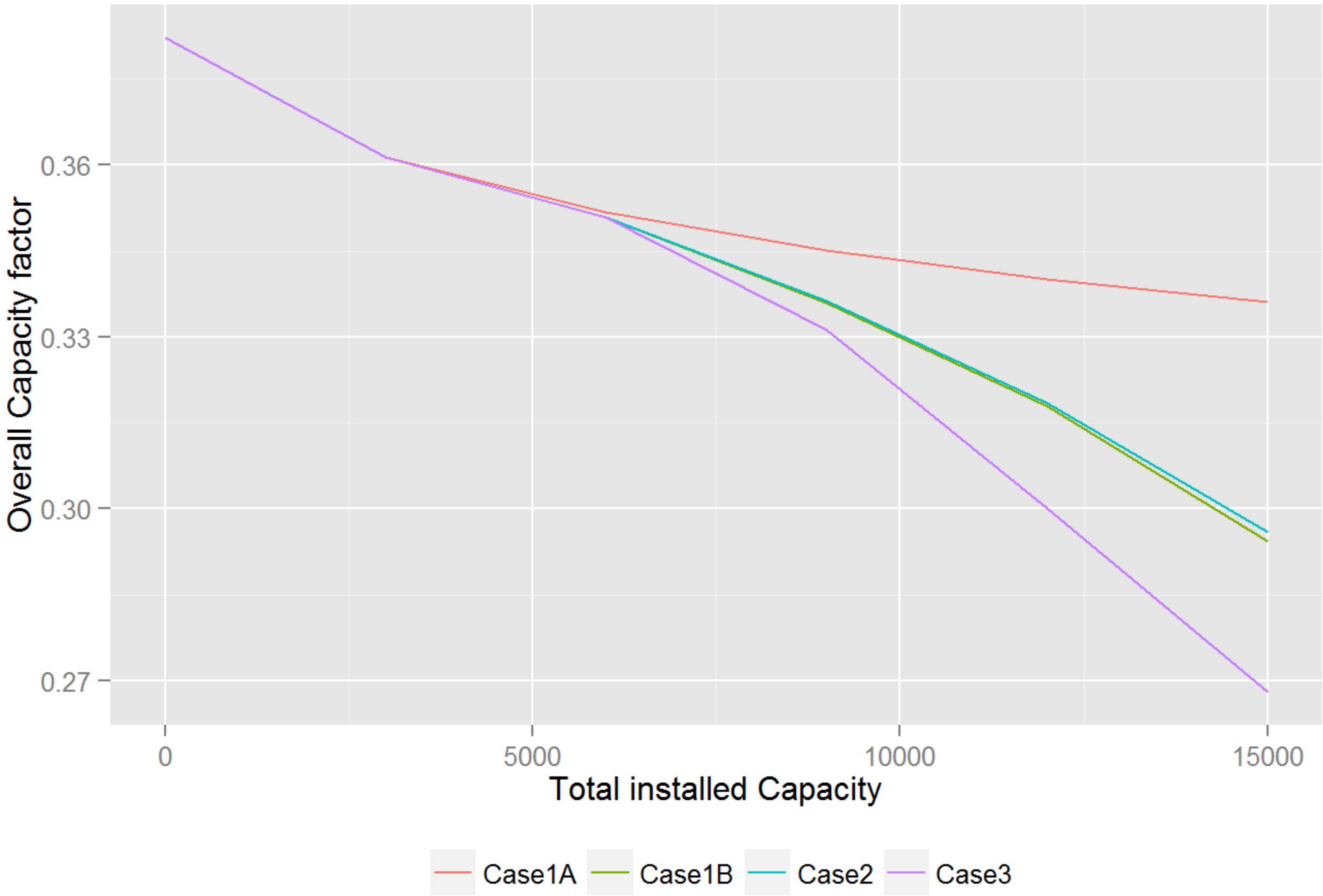
```
row.add.mode(lp.case3,"off")
#set bounds on decision variables:
set.bounds(lp.case3,lower=rep(0,n.variables.3),upper=c(dataOriginal$n.t.max[which(dataOriginal$region=="wes
t")],dataOriginal$n.t.max[which(dataOriginal$region=="east")],dataOriginal$n.t.max[which(dataOriginal$regio
n=="downstate")],rep(MaxCapOtherWest,n.hrs),rep(MaxCapOtherEast,n.hrs),rep(MaxCapOtherDownstate,n.hrs),rep(
trans.west,n.hrs),rep(trans.east,n.hrs)))

#solve optimization problem
solve(lp.case3)
turbine.distr<-get.variables(lp.case3)[1:n.sites]
turbine.distr.mat.3[,i]<-turbine.distr
othergen.total.mwh<-get.objective(lp.case3)
cf.wind.3[i]<-(sum(demand.mw$whole)-othergen.total.mwh-hrly.baseload*n.hrs)/(N.t*8760*3)

#close for loop
}
cf.wind.3
```

```
## [1] 0.3823 0.3614 0.3508 0.3313 0.3000 0.2679
```

```
##set a data frame for Case3
Case3<- data.frame(x=N.t*3,y= cf.wind.3)
##show the turbine distribution of Case3
colnames(turbine.distr.mat.3) <- c("3MW","3GW","6GW","9GW","12GW","15GW")
#show the turbine distribution of Case3,9GW
turbine.distr.mat.3[,4]
```

```
## [1]   0  50  49 127  50  59 215  50 357  87 112  79 112  70 129  84   0
## [18]   0   0 112   0 188  58   0   0 122   0   0   0  75   0   0   0   0
## [35]   0   0   0 164   0   0   0   0   0   0   0   0   0   0   0   0   0
## [52]   0   0   0  49  65  78  49  50  60   0   0 108  81  53  58
```

```
#Plotting Case 1A and 1B, 2 and 3 together
#plot the overall capacity factor vs. total installed capacity for both case1A&B&2
ggplot() + geom_line(data=Case1A, aes(x=3*counter[1:6], y=CF, color="Case1A")) + geom_line(data=Case1B, aes
(x=3*counter[1:6], y=CFb, color="Case1B")) +geom_line(data=Case2, aes(x=3*counter[1:6],y=y,color="Case2"))+
geom_line(data=Case3, aes(x=3*counter[1:6],y=y,color="Case3"))+ xlab("Total installed Capacity") +ylab("Ove
rall Capacity factor") +theme(legend.position="bottom",legend.title=element_blank(),axis.title.x=element_te
xt(),axis.title.y=element_text())
```




```
#Case4
#New variables
#n.lines.west_east
```

```
#n.lines.east_downstate

# recovery factor
rfactor <- 0.1
# Wind turbine costs 1,500,000$/MW = 1500$/KW
cost.wind.per.mw <- 1500000
# Additional transmission lines cost 1,000,000$/mile
cost.trans.per.mile <- 1000000
length.line.west_east <- 200
length.line.east_downstate <- 150
cap.per.line.mw <- 400

#Calculate the annual cost for wind and additional transmission
ann.cost.wind.per.mw <- rfactor*cost.wind.per.mw
ann.cost.west_east.per.line <- cost.trans.per.mile*length.line.west_east*rfactor
ann.cost.east_downstate.per.line <- cost.trans.per.mile*length.line.east_downstate*rfactor

##number of variables=number of turbines at each site (66)+ hrly generation from other nonbase sources of west, east and downstate region(8760*3) + transmission form west to east and from east to downstate+ number of lines(2)
n.variables.4<-n.sites+n.hrs*5+2
##number of constraints =1(total number turbines constraint)+energy balances for 3 regions+ 2*8760 constraints for transmission from west to east and form east to downstate + 1 constraint related to cost
n.cons.4<-1+3*n.hrs+2*n.hrs+1
##numbers of bounded variables+number of turbines at each site+hrly generation from nonbase source
n.bounds.4<-n.variables.4
```

```
# Create a for loop to evaluate different levels of wind penetration. set counter as the number of turbines
```

```
.
counter<-c(1,1000,2000,3000,4000,5000)
cf.wind.4<-NULL
turbine.distr.mat.4<-matrix(0,66,6)
for (i in 5:6)
  {
  # Define total number of turbines
  N.t <- counter[i]

  # Develop MILP

  # Create linear program model with number of rows=num of constraints and number of columns = number of va
riables
  lp.case4 <- make.lp(nrow=n.cons.4,ncol=n.variables.4)
  ##Setting our linear program as a minimization problem
  lp.control(lp.case4, sense="min")
  ## Define coefficients for the objective function (all "0", except a coefficient of "1" for gen_other for
each hour)
  set.objfn(lp.case4,obj=c(rep(0,n.sites),rep(1,3*n.hrs), rep(0,(2*n.hrs+2))))

  # We need to force the number of turbines at each site to be an integer
  set.type(lp.case4,c(1:n.sites,1+n.sites+5*n.hrs,2+n.sites+5*n.hrs),type="integer")

  #set.type(lp.case3,c(1:n.sites),type="integer")



  # We will be be adding constraints by row (rather than adding variable coefficients by column)
  ## Turn on "row entry mode" - remember to turn it off after adding all constraints
```

```
  row.add.mode(lp.case4,"on")

  #Add the first constraint. Sum of number of turbines at all sites equals total number of turbines
  add.constraint(lp.case4,xt=rep(1,n.sites),type="=",rhs=N.t,indices=c(1:n.sites))

  ##Energy balance same as case3
for(j in 1:n.hrs)
  {
add.constraint(lp.case4,xt=c(sites.wind.power.per.original[j,1+sites.west],1,-1),type=">=",rhs=demand.mw[j,
"west"]-west.base.gen/8760,indices=c(sites.west,(n.sites+j),(n.sites+3*n.hrs+j)))
  }

for(j in 1:n.hrs)
  {
add.constraint(lp.case4,xt=c(sites.wind.power.per.original[j,1+sites.east],1,1,-1),type=">=",rhs=demand.mw[
j,"east"]-east.base.gen/8760,indices=c(sites.east,(n.sites+n.hrs+j),(n.sites+3*n.hrs+j),(n.sites+4*n.hrs+j)
))
}

for(j in 1:n.hrs)
  {
add.constraint(lp.case4,xt=c(sites.wind.power.per.original[j,1+sites.downstate],1,1),type=">=",rhs=demand.m
w[j,"downstate"]-donwstate.base.gen/8760,indices=c(sites.downstate,(n.sites+2*n.hrs+j),(n.sites+4*n.hrs+j))
)
}

#Two new constraint related to additional transmission lines
 for (j in 1:n.hrs)
```

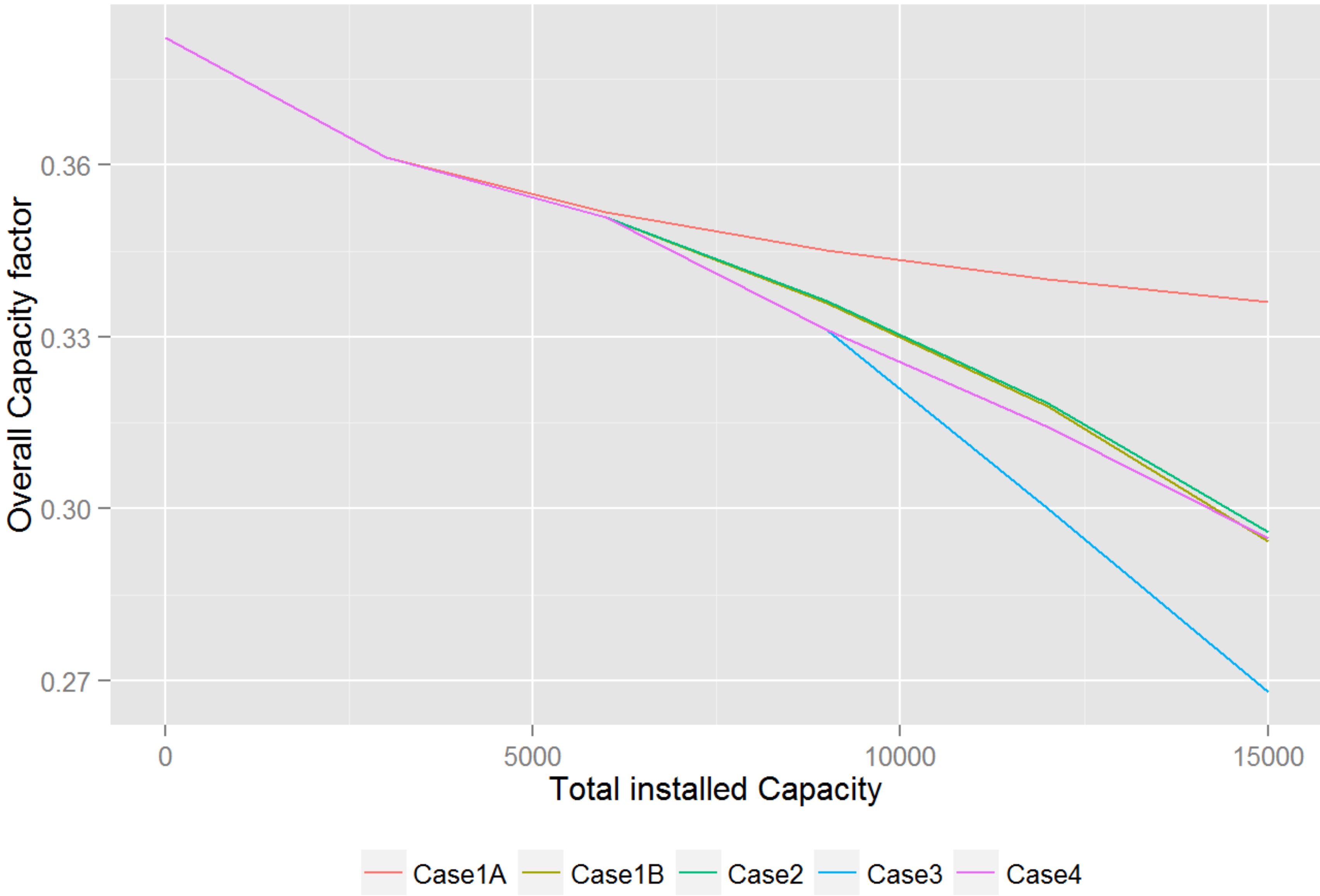
```
  {
    add.constraint(lp.case4,xt=c(1,-cap.per.line.mw),type="<=",rhs=trans.west,indices=c((n.sites+3*n.hrs+j)
,(n.sites+5*n.hrs+1)))
  }

  for (j in 1:n.hrs)
  {
    add.constraint(lp.case4,xt=c(1,-cap.per.line.mw),type="<=",rhs=trans.east,indices=c((n.sites+4*n.hrs+j)
,(n.sites+5*n.hrs+2)))
  }

add.constraint(lp.case4,
               xt=c(rep(1,3*n.hrs),
                    (8760*Case3$y[i]*ann.cost.west_east.per.line/ann.cost.wind.per.mw),
                    (8760*Case3$y[i]*ann.cost.east_downstate.per.line/ann.cost.wind.per.mw)),
               type="<=",
               rhs=sum(demand.mw$whole)-8760*hrly.baseload-8760*Case3$y[i]*3*counter[i] ,
               indices=c((n.sites+1):(n.sites+3*n.hrs),(n.sites+5*n.hrs+1),(n.sites+5*n.hrs+2)))

#Turn off row entry mode:
row.add.mode(lp.case4,"off")

## Set bounds on decision variables
  set.bounds(lp.case4,lower=rep(0,n.variables.4),upper=c(dataOriginal$n.t.max[which(dataOriginal$region=="w
est")],dataOriginal$n.t.max[which(dataOriginal$region=="east")],dataOriginal$n.t.max[which(dataOriginal$reg
ion=="downstate")],rep(MaxCapOtherWest,n.hrs),rep(MaxCapOtherEast,n.hrs),rep(MaxCapOtherDownstate,n.hrs),re
p(Inf,2*n.hrs),rep(Inf,2)))
```

```
  # Solve optimization problem
solve(lp.case4)
turbine.distr<-get.variables(lp.case4)[1:n.sites]
turbine.distr.mat.4[,i]<-turbine.distr
othergen.total.mwh <- get.objective(lp.case4)
cf.wind.4[i]<-(sum(demand.mw$whole)-othergen.total.mwh-hrly.baseload*n.hrs)/(N.t*8760*3)

#close for loop
}
cf.wind.4
```

```
## [1]     NA     NA     NA     NA 0.3144 0.2948
```

```
#set up a data frame for Case4.CF
Case4 <- as.data.frame(counter)
Case4$N.t <- counter[i]
 # We assume that CFs for cap 3MW, 3GW, 6GW and 9GW are the same for cases 3 and 4
Case4$CF4 <- c(Case3$y[1:4],cf.wind.4[5],cf.wind.4[6])
```

```
#Plotting Case 1A and 1B, 2, 3 and 4 together
#plot the overall capacity factor vs. total installed capacity for both case1A&B&2
ggplot() + geom_line(data=Case1A, aes(x=3*counter[1:6], y=CF, color="Case1A")) + geom_line(data=Case1B, aes
(x=3*counter[1:6], y=CFb, color="Case1B")) +geom_line(data=Case2, aes(x=3*counter[1:6],y=y,color="Case2"))+
geom_line(data=Case3, aes(x=3*counter[1:6],y=y,color="Case3"))+ geom_line(data=Case4, aes(x=3*counter[1:6],
y=Case4$CF4,color="Case4"))+xlab("Total installed Capacity") +ylab("Overall Capacity factor") +theme(legend
.position="bottom",legend.title=element_blank(),axis.title.x=element_text(),axis.title.y=element_text())
```




```
#Create a data frame that contains every turbine distribution of 9GW scinario
TurbineDistribution<-data.frame(c(1:66),TurbineDistCase1A$NumberofTurbine,TurbineDistCase1B$NumberofTurbine
,turbine.distr.mat.2[,4],turbine.distr.mat.3[,4],turbine.distr.mat.4[,i])
```

```
#Change the column names to "Case1A","Case1B","Case2","Case3","Case4"
colnames(TurbineDistribution)<-c("SitesNo","Case1A","Case1B","Case2","Case3","Case4")
TurbineDistribution
```

```
##    SitesNo Case1A Case1B Case2 Case3 Case4
## 1        1      0      0     0     0     0
## 2        2     50     50    50    50    50
## 3        3     49     49    49    49    49
## 4        4    127    127   127   127   127
## 5        5     50     50    50    50    50
## 6        6     59     59    59    59    59
## 7        7    215    215   215   215   215
## 8        8     50     50    50    50    50
## 9        9    357    357   357   357   357
## 10      10     87     87    87    87    87
## 11      11    112    112   112   112   112
## 12      12     79     79    79    79    79
## 13      13    112    112   112   112   112
## 14      14     70     70    70    70    70
## 15      15    129    129   129   129   129
## 16      16     84     84    84    84    84
## 17      17     84     84    84     0    84
## 18      18     83     83    47     0    83
## 19      19     63     63    53     0    63
## 20      20    112    112   112   112   112
## 21      21     19     19     0     0    94
## 22      22    188    188   188   188   188
## 23      23     58     58    58    58    58
```

```
## 24      24   56   56   0   0  56
## 25      25    0    0   0   0 331
## 26      26  122  122  29 122 122
## 27      27    0    0   0   0  50
## 28      28    0    0   0   0  77
## 29      29    0    0   0   0  84
## 30      30    0    0  75  75  75
## 31      31    0    0   0   0  82
## 32      32    0    0   0   0 195
## 33      33    0    0   0   0 128
## 34      34   50   50   0   0   0
## 35      35    0    0   0   0   0
## 36      36    0    0   0   0  50
## 37      37    0    0   0   0  50
## 38      38  184  184 184 164 184
## 39      39    0    0   0   0   0
## 40      40    0    0   0   0   0
## 41      41    0    0   0   0  59
## 42      42    0    0   0   0 130
## 43      43    0    0   0   0   0
## 44      44    0    0   0   0   0
## 45      45    0    0   0   0  70
## 46      46    0    0   0   0   0
## 47      47    0    0   0   0   0
## 48      48    0    0   0   0   0
## 49      49    0    0   0   0 109
## 50      50    0    0   0   0   0
## 51      51    0    0   0   0   0
```

```
## 52      52     0      0     0     0     0
## 53      53     0      0     0     0     0
## 54      54     0      0     0     0    50
## 55      55    49     49    49    49    49
## 56      56    65     65    65    65    65
## 57      57    78     78    78    78    78
## 58      58    49     49    49    49    49
## 59      59    50     50    50    50    50
## 60      60    60     60    60    60    60
## 61      61     0      0     0     0    49
## 62      62     0      0     0     0    86
## 63      63     0      0   108   108   108
## 64      64     0      0    81    81    81
## 65      65     0      0     0    53    53
## 66      66     0      0     0    58    58
```

```
#Create a data frame that contains all CFs
#Find the CFs for every case and make the name neat.
cf.1A<-CF
cf.1B<-CFb
cf.2<-cf.wind
cf.3<-cf.wind.3
cf.4<-Case4$CF4
CF.all.cases<-data.frame(nrow=6,ncol=6)
CF.all.cases<-data.frame(3*counter,cf.1A,cf.1B,cf.2,cf.3,cf.4)
#Change the column names to "Case1A","Case1B","Case2","Case3","Case4"
colnames(CF.all.cases)<-c("Capacity","Case1A","Case1B","Case2","Case3","Case4")
CF.all.cases
```

```
##   Capacity Case1A Case1B  Case2  Case3  Case4
## 1        3 0.3823 0.3823 0.3823 0.3823 0.3823
## 2     3000 0.3614 0.3614 0.3614 0.3614 0.3614
## 3     6000 0.3519 0.3510 0.3510 0.3508 0.3508
## 4     9000 0.3451 0.3359 0.3363 0.3313 0.3313
## 5    12000 0.3401 0.3179 0.3184 0.3000 0.3144
## 6    15000 0.3361 0.2942 0.2959 0.2679 0.2948
```