Validating nowcast routine

John M. Drake

3/15/2020

Introduction

This script uses our stochastic simulator to try and validate the nowcasting scheme.

Nowcasting is implemented here.

```
US.params <- list(admin = "US", # state or country
                    q = 1, # Detection probability
                    a = 0.346, # Proportion of cases that are asymptomatic
                    c = 1, # Transmissability of undetectable cases
                    effective.infectious.period = list(dist="exponential", mean=2.67),
                    infectious.period = list(dist="exponential", mean=7),
                    incubation.period = list(dist="gamma", mean=7.67, shape=4.1775942)
                    )

nowcast_from_case_reports(US.params)
```

This code implements the stochastic simulation model.

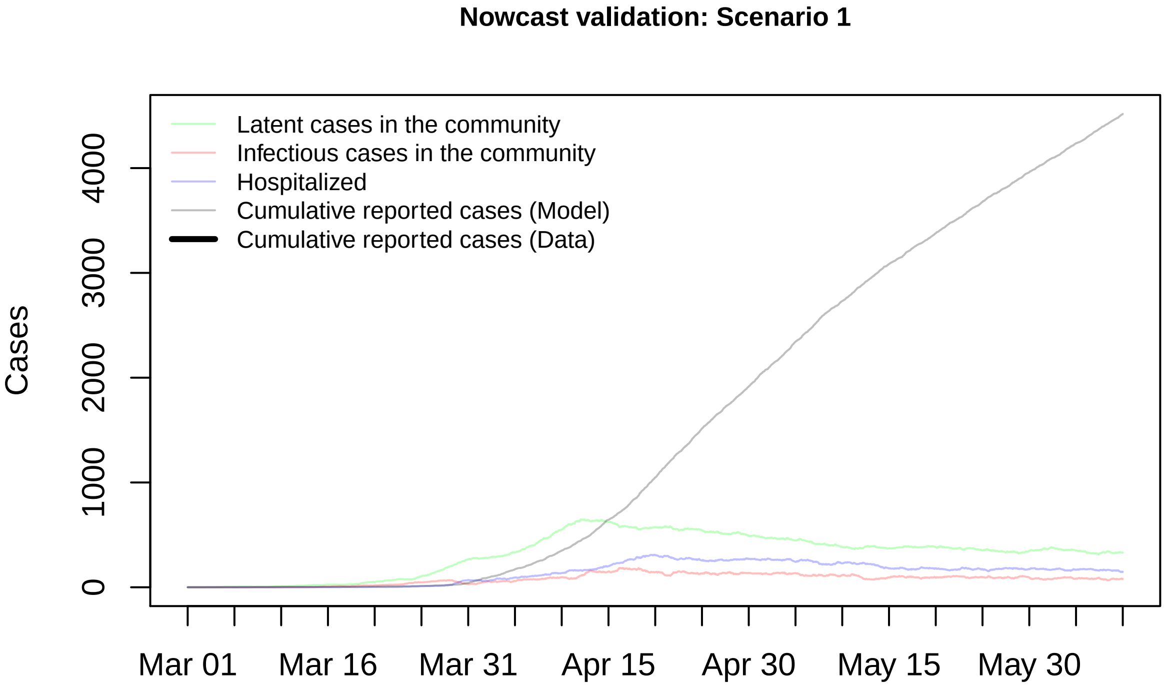
```
onestep <- function (x, params) {  #function to calculate one step of stochastic SIR

  S <- x[2]                            #local variable for susceptibles

  E1 <- x[3]                           #exposed classes
  E2 <- x[4]
  E3 <- x[5]
  E4 <- x[6]
  E5 <- x[7]
  E6 <- x[8]

  I1 <- x[9]                           #detected infectious classes
  I2 <- x[10]
  I3 <- x[11]
  I4 <- x[12]

  Iu1 <- x[13]                           #undetected infectious classes
  Iu2 <- x[14]
  Iu3 <- x[15]
  Iu4 <- x[16]

  I.detected <-I1+I2+I3+I4
  I.undetected <- Iu1+Iu2+Iu3+Iu4
  I <- I1+I2+I3+I4+Iu1+Iu2+Iu3+Iu4     #total infectious

  H <- x[17]                           #local variable for hospitalized
  Ru <- x[18]                          #local variable for undetected recovereds

  C <- x[19]    # local variable for notifications
```

```
N <- S+E1+E2+E3+E4+E5+E6+I1+I2+I3+I4+Iu1+Iu2+Iu3+Iu4+H+Ru          # total size of population

t <- x[20]                           #get current time

with(                                 #use with to simplify code
     as.list(params),
     {
       gammai <- 4*gamma(z=z, b=b, a0=a0, t=as.numeric(t))  # multiplier 4 for pseudo stages
       sigmai <- 6*sigma  # multiplier 6 for pseudo stages
       etat <- eta(t,w)     # case notification rate
       betat <- beta(t,w)   # time dependent transmissibility, presymptomatic=1 causes this transmiss

       rates <- as.numeric(c(betat*I.detected/N+betat*c*I.undetected/N+presymptomatic*betat*c*E6/N,
                  sigmai, sigmai, sigmai, sigmai, sigmai, sigmai,   # movements out of E
                  gammai, gammai, gammai, gammai,                   # movements out of I (detected)
                  b, b, b, b,                                       # movements out of I (undetected)
                  etat))

       states0 <- x[2:(length(x)-1)]

       # transition probabilities

       p <- matrix(0, nrow=length(rates),ncol=length(states0))                              #

       p[1,]  <- c(exp(-rates[1]*dt),1-exp(-rates[1]*dt), 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

       p[2,]  <- c(0, exp(-rates[2]*dt), 1-exp(-rates[2]*dt), 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       p[3,]  <- c(0, 0, exp(-rates[3]*dt), 1-exp(-rates[3]*dt), 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       p[4,]  <- c(0, 0, 0, exp(-rates[4]*dt), 1-exp(-rates[4]*dt), 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       p[5,]  <- c(0, 0, 0, 0, exp(-rates[5]*dt), 1-exp(-rates[5]*dt), 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       p[6,]  <- c(0, 0, 0, 0, 0, exp(-rates[6]*dt), 1-exp(-rates[6]*dt), 0, 0, 0, 0, 0, 0, 0, 0, 0,
       p[7,]  <- c(0, 0, 0, 0, 0, 0, exp(-rates[7]*dt), (1-exp(-rates[7]*dt))*q(w), 0, 0, 0, (1-exp(-

       p[8,]  <- c(0, 0, 0, 0, 0, 0, 0, exp(-rates[8]*dt), 1-exp(-rates[8]*dt), 0, 0, 0, 0, 0, 0, 0,
       p[9,]  <- c(0, 0, 0, 0, 0, 0, 0, 0, exp(-rates[9]*dt), 1-exp(-rates[9]*dt), 0, 0, 0, 0, 0, 0,
       p[10,] <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, exp(-rates[10]*dt), 1-exp(-rates[10]*dt), 0, 0, 0, 0, 0
       p[11,] <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, exp(-rates[11]*dt), 0, 0, 0, 0, 1-exp(-rates[11]*dt)

       p[12,]  <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  exp(-rates[12]*dt), 1-exp(-rates[12]*dt), 0, 0,
       p[13,]  <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  exp(-rates[13]*dt), 1-exp(-rates[13]*dt), 0,
       p[14,] <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  exp(-rates[14]*dt), 1-exp(-rates[14]*dt),
       p[15,] <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  exp(-rates[15]*dt), 0, 1-exp(-rates[15]


       p[16,] <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, exp(-rates[16]*dt), 0, 1-exp(-rates[1

       # update states

       states1 <- matrix(0, nrow=length(rates),ncol=length(states0))                          #

       for(i in 1:length(rates)){
         states1[i,] <- t(rmultinom(1, states0[i], p[i,]))
       }
```

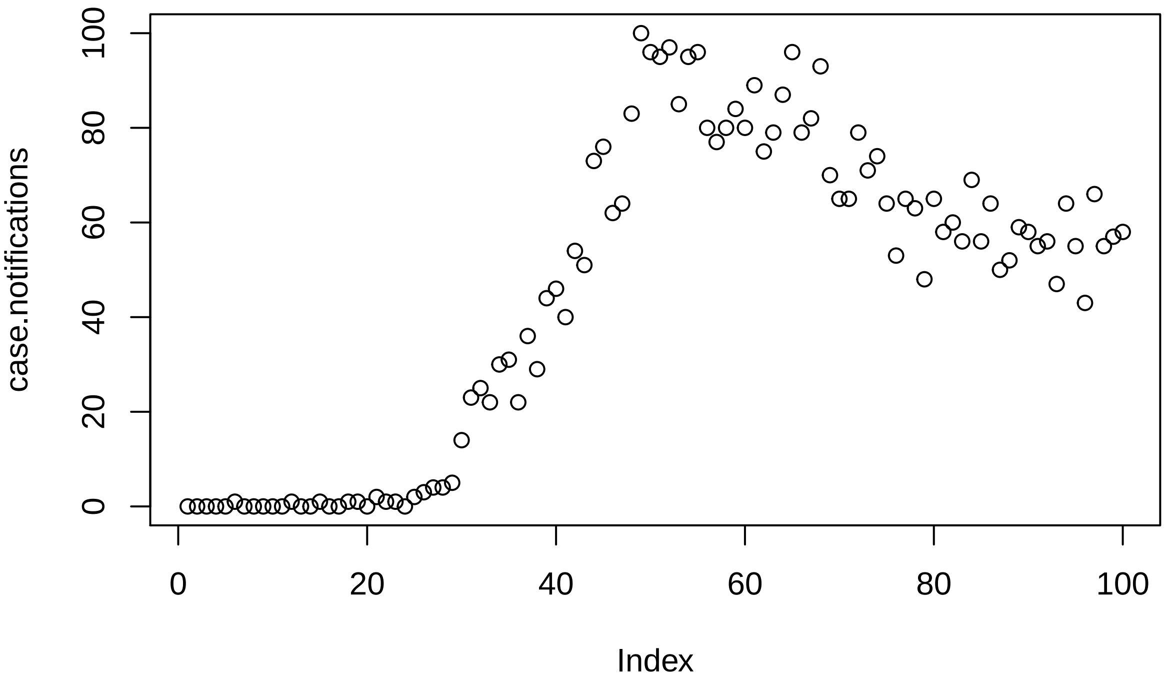
```
         states1 <- colSums(states1)
         states1[17] <- states1[17]+Ru  #add formerly Recovered undetected cases
         states1[18] <- states1[18]+C  #add formerly notified cases

         return(x <- c(dt, states1, tail(x,1)+dt))
       }
       )
}


model <- function (x, params, nstep) {  #function to simulate stochastic SIR
  output <- array(dim=c(nstep+1,length(x)))        #set up array to store results
  colnames(output) <- c("time","S",
                        "E1", "E2", "E3", "E4", "E5", "E6",
                        "I1", "I2", "I3", "I4", "Iu1", "Iu2", "Iu3", "Iu4",
                        "H", "Ru", "C", "cum.time") #name variables
  output[1,] <- x                           #first record of output is initial condition
  for (k in 1:nstep) {                      #iterate for nstep steps
    output[k+1,] <- x <- as.numeric(onestep(x,params))
  }
  output                                    #return output
}


gamma <- function(z = 12, b=0.143, a0=1/1.5, t){
  # piecewise function
  # default parameters z = 12, b=1/7, a0=1/1.5
  #    z: time at start of intervention (notionally March 12)
  #    b: intercept (positive)
  #    a0: post intervention isolation ratae
  #    t: time in the model

  gamma <- ifelse(t<=z, gamma <- b, gamma <- a0)
  return(gamma)
}

eta <- function(t, w=12) ifelse(t<=w,1/3,1/3)

q <- function(t, w=12, q0=1, q1=1) ifelse(t<=w,q0,q1)

beta <- function(t, w=12, beta0=0.6584, beta.factor=2) ifelse(t<=w,beta0,beta0/beta.factor)


evaluate.model <- function(params=list(beta0=0.6584, sigma=1/6.4, z=12, b=0.143, a0=1/1.5, w=12, c=1, p
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=6, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                           nsims=2, nstep=NULL, start=as.Date("2020-03-01"),today=Sys.Date()){

  if(is.null(nstep)) nstep <- (as.numeric(today-start)+1+28)/params$dt #run simulation from start to cu

  xstart <- c(time=0, unlist(init), cum.time = 0) #initial conditions
  data <- vector(mode='list',length=nsims) #initialize list to store the output

  for (k in 1:nsims) {              #simulate nsims times
    data[[k]] <- as.data.frame(model(xstart,params,nstep))
    data[[k]]$cum.time <- cumsum(data[[k]]$time)

  }

  return(data)
}

plot.model <- function(data, log='y', title=''){
  # The function `plot.model` provides automated visualization of model simulations

  # process data
  nsims <- length(data)

  for(i in 1:nsims) data[[i]]$I <- data[[i]]$I1 + data[[i]]$I2 + data[[i]]$I3 +
      data[[i]]$I4
  for(i in 1:nsims) data[[i]]$Iu <- data[[i]]$Iu1 + data[[i]]$Iu2 + data[[i]]$Iu3 +
      data[[i]]$Iu4
  for(i in 1:nsims) data[[i]]$E <- data[[i]]$E1 + data[[i]]$E2 + data[[i]]$E3 +
      data[[i]]$E4 + data[[i]]$E5 + data[[i]]$E6

  max.time<-data[[1]]$cum.time[max(which(data[[1]]$I>0))] #maximum time in first simulation
  max.y<-max(data[[1]]$C)       #find max total confirmed cases for plotting range

  # calculate means
  m1 <- m2 <- m3 <- m4 <- m5 <- matrix(nrow=length(data[[1]]$I), ncol=nsims)
  for(i in 1:nsims){
    m1[,i] <- data[[i]]$E
    m2[,i] <- data[[i]]$I+data[[i]]$Iu
   # m3[,i] <- data[[i]]$Iu
    m4[,i] <- data[[i]]$H
    m5[,i] <- data[[i]]$C
  }
  E.mean <- rowMeans(m1)
  I.mean <- rowMeans(m2)
 # Iu.mean <- rowMeans(m3)
  H.mean <- rowMeans(m4)
  C.mean <- rowMeans(m5)

  # colors
  E.col <- rgb(0,1,0,.25)
  I.col <- rgb(1,0,0,.25)
  Iu.col <- rgb(0.5, 0.5, 0, 0.25)
  H.col <- rgb(0,0,1,.25)
  C.col <- rgb(0,0,0,.25)
  E.mean.col <- rgb(0,1,0,1)
  I.mean.col <- rgb(1,0,0,1)
  Iu.mean.col <- rgb(0.5,0.5,0,1)
  H.mean.col <- rgb(0,0,1,1)
  C.mean.col <- rgb(0,0,0,1)
  #set up plot
  plot(I~cum.time,data=data[[1]],xlab='',ylab='Cases',col=1,
       xlim=c(0,max.time),ylim=c(1,max.y), type='n', lty=1, log=log,
       axes=FALSE, main=title, cex.main=0.8) # set up plot

  # add data to plot
  #day <- georgia$date - start
  #lines(day, cumsum(georgia$cases), type='h', col='black', lwd=3, lend='butt' )

  # plot spaghetti
  lines(E~cum.time,data=data[[1]], col=E.col, lty=1)
  lines(I+Iu~cum.time,data=data[[1]], col=I.col, lty=1)
 # lines(Iu~cum.time,data=data[[1]], col=Iu.col, lty=1)
  lines(H~cum.time,data=data[[1]], col=H.col, lty=1)
  lines(C~cum.time,data=data[[1]], col=C.col, lty=1, lwd=1)


  axis(1, at=seq(0,max.time,5), labels=format(start+seq(0,max.time,5), format= '%b %d'))
  axis(2)
  box()

  if(nsims > 1){
  for (k in 2:min(100,nsims)) {              #add multiple epidemics to plot
    lines(E~cum.time, data=data[[k]], col=E.col, type='l', lty=1)
    lines(I+Iu~cum.time, data=data[[k]], col=I.col, type='l', lty=1)
 #   lines(Iu~cum.time, data=data[[k]], col=Iu.col, type='l', lty=1)
    lines(H~cum.time, data=data[[k]], col=H.col, type='l', lty=1)
    lines(C~cum.time, data=data[[k]], col=C.col, type='l', lty=1, lwd=1)
  }

  # plot means
  lines(E.mean~cum.time, data=data[[k]], col=E.mean.col, lty=1)
  lines(I.mean~cum.time, data=data[[k]], col=I.mean.col, lty=1)
#  lines(Iu.mean~cum.time, data=data[[k]], col=Iu.mean.col, lty=1)
  lines(H.mean~cum.time, data=data[[k]], col=H.mean.col, lty=1)
  lines(C.mean~cum.time, data=data[[k]], col=C.mean.col, lty=1)
  }

   legend('topleft', lty=c(1,1,1,1,1,1), lwd=c(1,1,1,1,3,3), bty='n', cex=0.75,
         col=c(E.col, I.col, H.col, C.col, 'black'),
         legend=c('Latent cases in the community', 'Infectious cases in the community', 'Hospitalized',
                  'Cumulative reported cases (Model)', 'Cumulative reported cases (Data)'))
}
```

Now we simulate an epidemic. In this example, rapid case islation begins on day 28 and social distancing ("lockdown") begins in day 42. The epidemic is seeded with a single infectious individual. There is approximately 1 day of presymptomatic infection

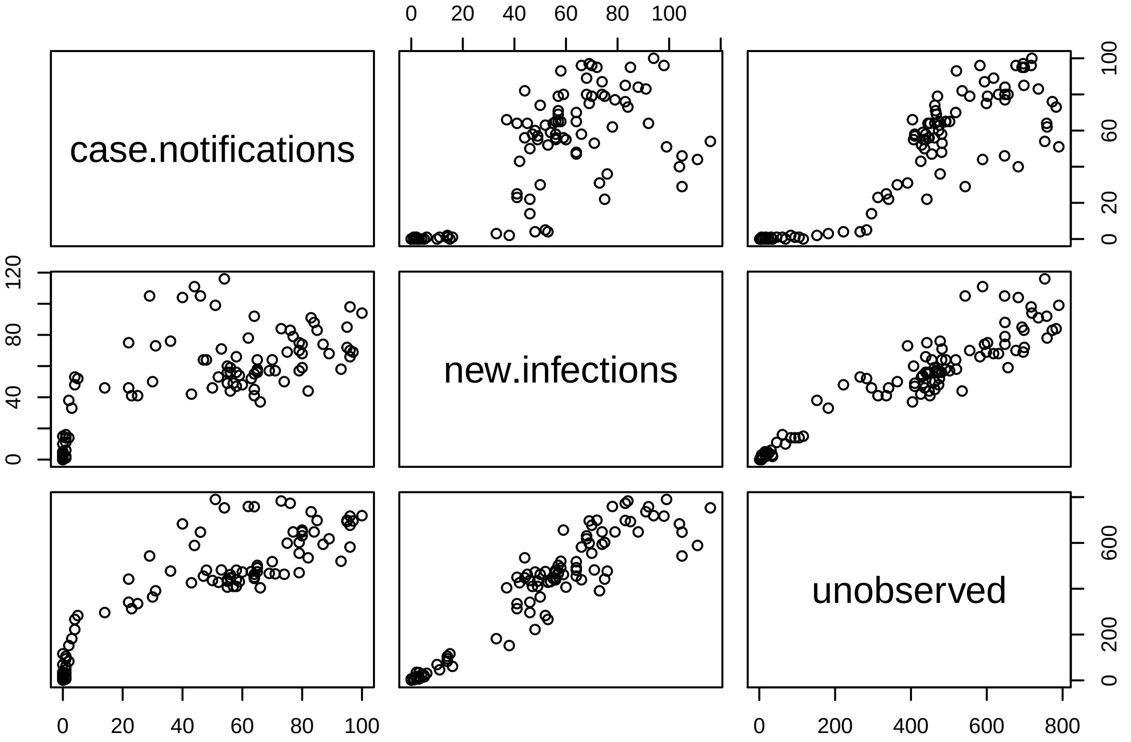
```
scenario <- 'Scenario 1'
start=as.Date("2020-03-01")
today <- Sys.Date()
out <- evaluate.model(params=list(beta0=0.6584, sigma=1/6.4, z=28, b=0.143, a0=1/1.5, w=42, presymptomat
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=0, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                                nstep = (as.numeric(today-start)+1+84)/0.05,   # solve for 12 weeks fro
                                nsims=1, start=as.Date("2020-03-01"))
plot.model(out, log='', title=paste('Nowcast validation:', scenario))
```



```
out <- out[[1]]
report.times <- seq(1,(length(out$cum.time)), by=20)
case.notifications <- diff(out$C[report.times])
plot(case.notifications)
```



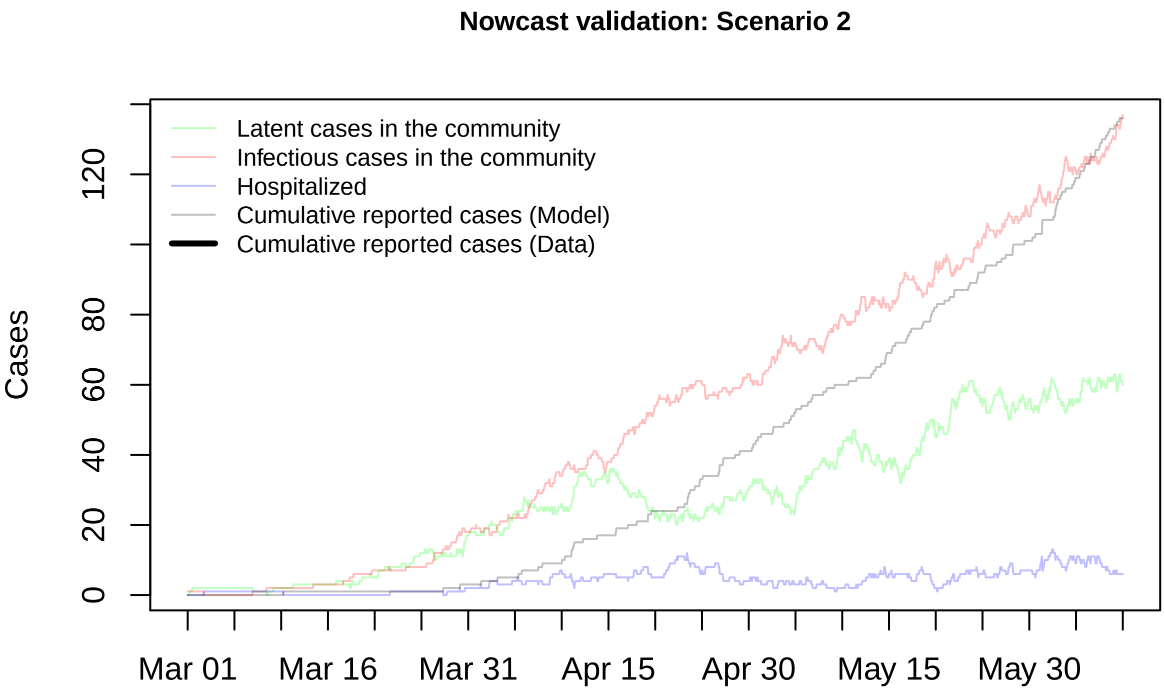
```
unobserved <- out$E1+out$E2+out$E3+out$E4+out$E5+out$E6+out$I1+out$I2+out$I3+out$I4+out$Iu1+out$Iu2+out
unobserved <- unobserved[tail(report.times,-1)]
new.infections <- head(-1*aggregate(diff(out$S), by=list(tail(floor(out$cum.time),-1)), FUN=sum)$x,-1)
nowcast.validation <- data.frame(case.notifications, new.infections, unobserved)
plot(nowcast.validation)
```



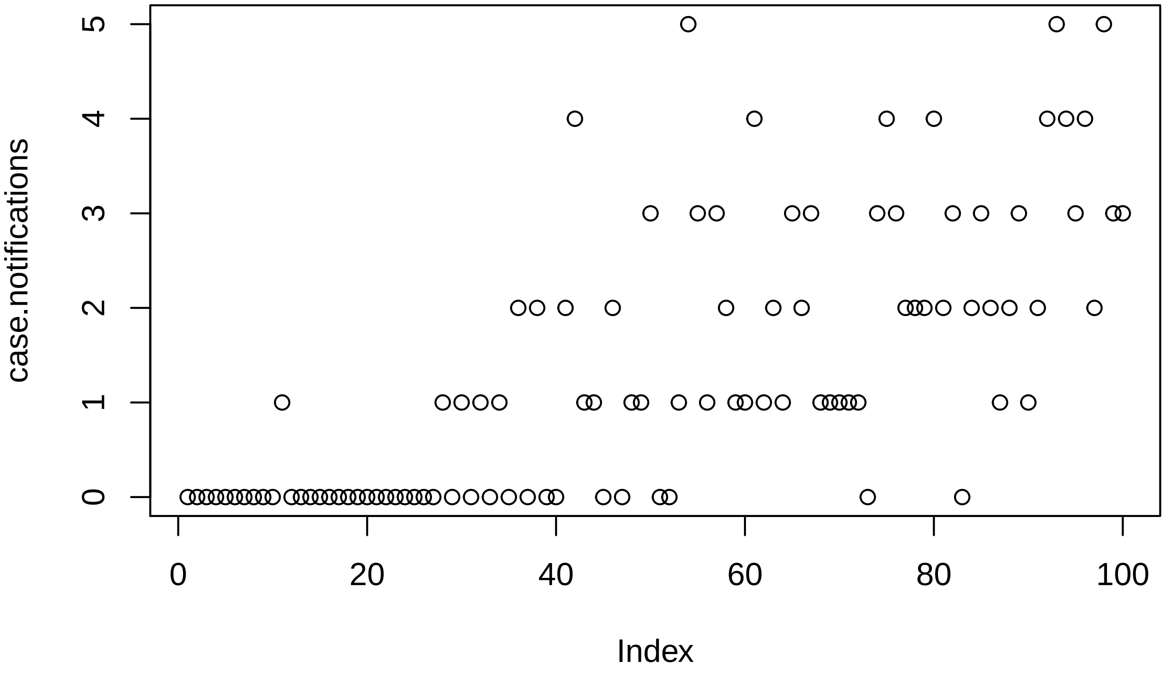
```
write.csv(nowcast.validation, file=paste('nowcast-validation-',scenario,'.csv', sep=''))
```

Here is a version with underreporting due to asymptomatic rate of 0.346. Note: this model assumes asymptomatic and presymptomatic persons transmit at the same rate as symptomatic persons.

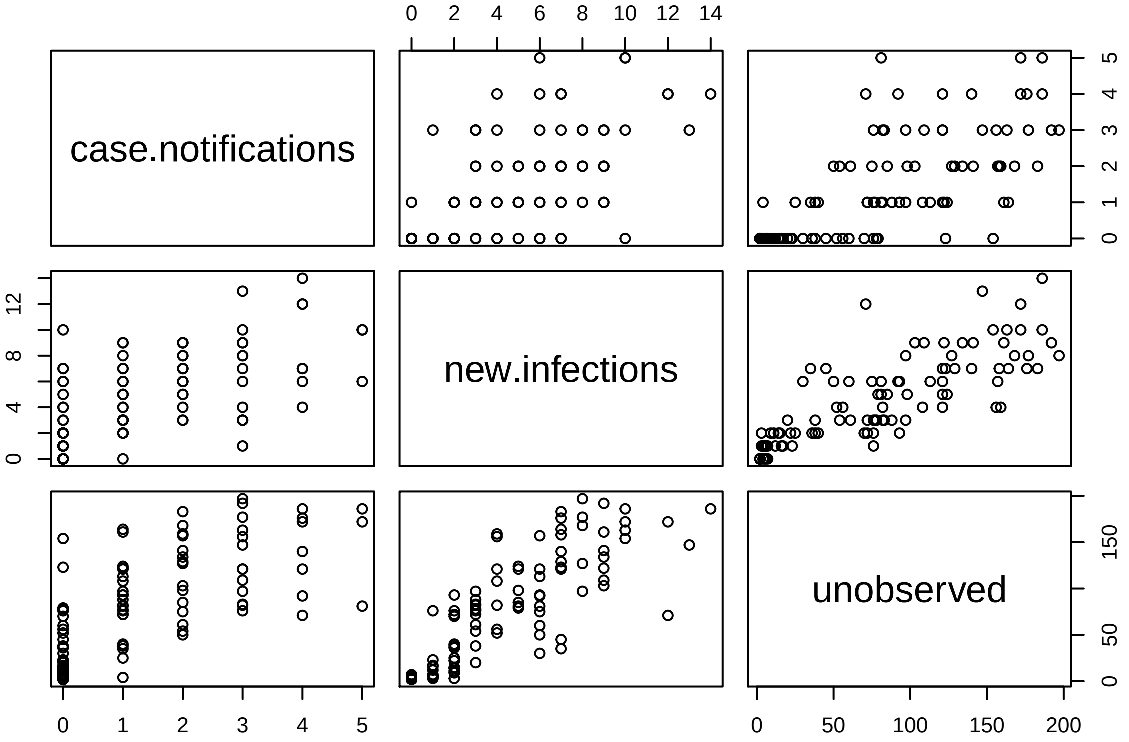
```
scenario <- 'Scenario 2'
start=as.Date("2020-03-01")
set.seed(03162020)

q <- function(t, w=12, q0=0.346, q1=0.346) ifelse(t<=w,q0,q1)

today <- Sys.Date()
out <- evaluate.model(params=list(beta0=0.6584, sigma=1/6.4, z=28, b=0.143, a0=1/1.5, w=42, presymptoma
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=0, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                                nstep = (as.numeric(today-start)+1+84)/0.05,   # solve for 12 weeks fro
                                nsims=1, start=as.Date("2020-03-01"))
plot.model(out, log='', title=paste('Nowcast validation:', scenario))
```



```r
out <- out[[1]]
report.times <- seq(1,(length(out$cum.time)), by=20)
case.notifications <- diff(out$C[report.times])
plot(case.notifications)
```



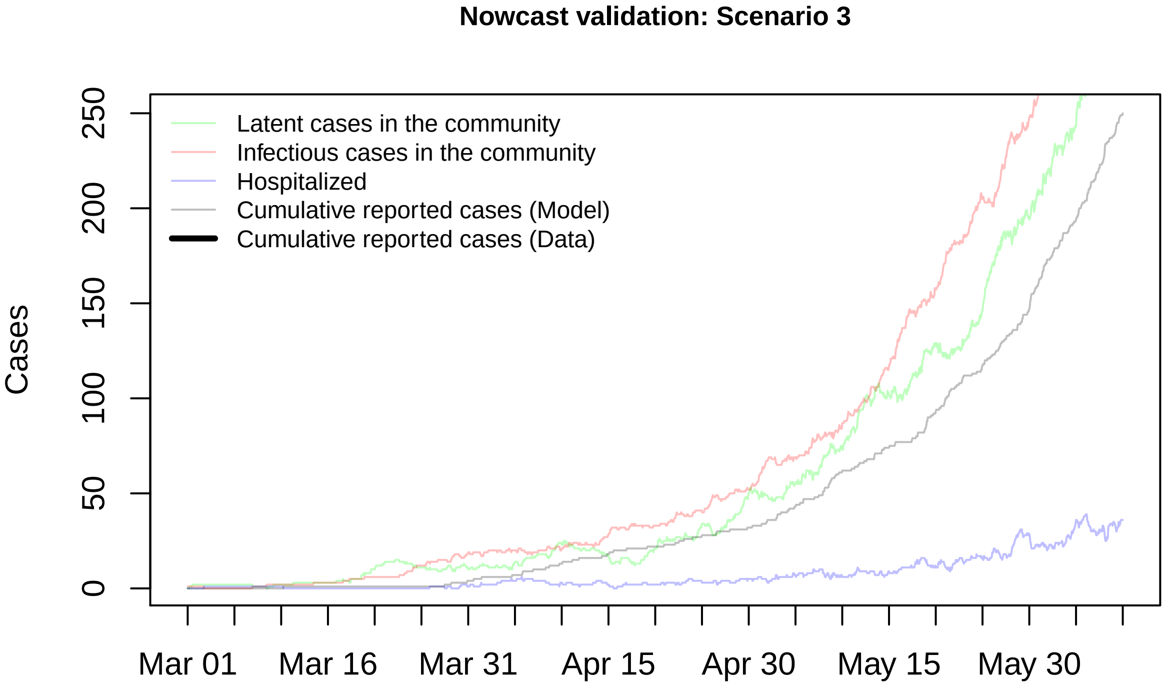
```r
unobserved <- out$E1+out$E2+out$E3+out$E4+out$E5+out$E6+out$I1+out$I2+out$I3+out$I4+out$Iu1+out$Iu2+out
unobserved <- unobserved[tail(report.times,-1)]
new.infections <- head(-1*aggregate(diff(out$S), by=list(tail(floor(out$cum.time),-1)), FUN=sum)$x,-1)
nowcast.validation <- data.frame(case.notifications, new.infections, unobserved)
plot(nowcast.validation)
```



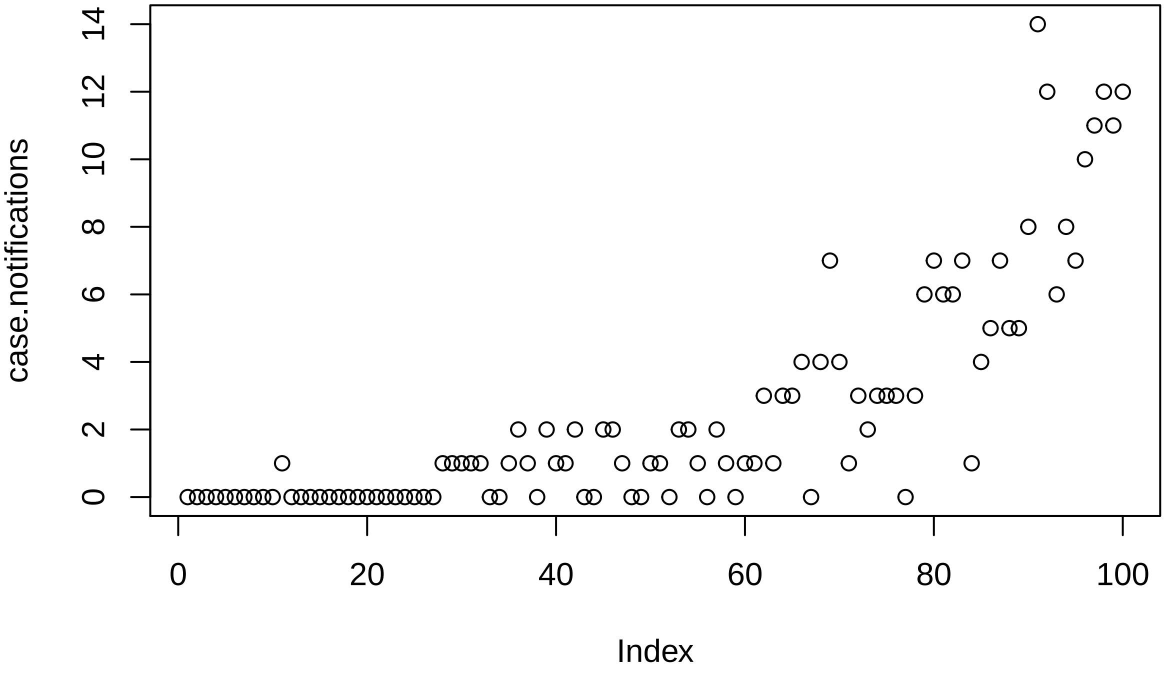
```
write.csv(nowcast.validation, file=paste('nowcast-validation-',scenario,'.csv', sep=''))
```

Here is another example with early intervention (day 21) and moderate asymptomatic transmission (35% of symptomatic).

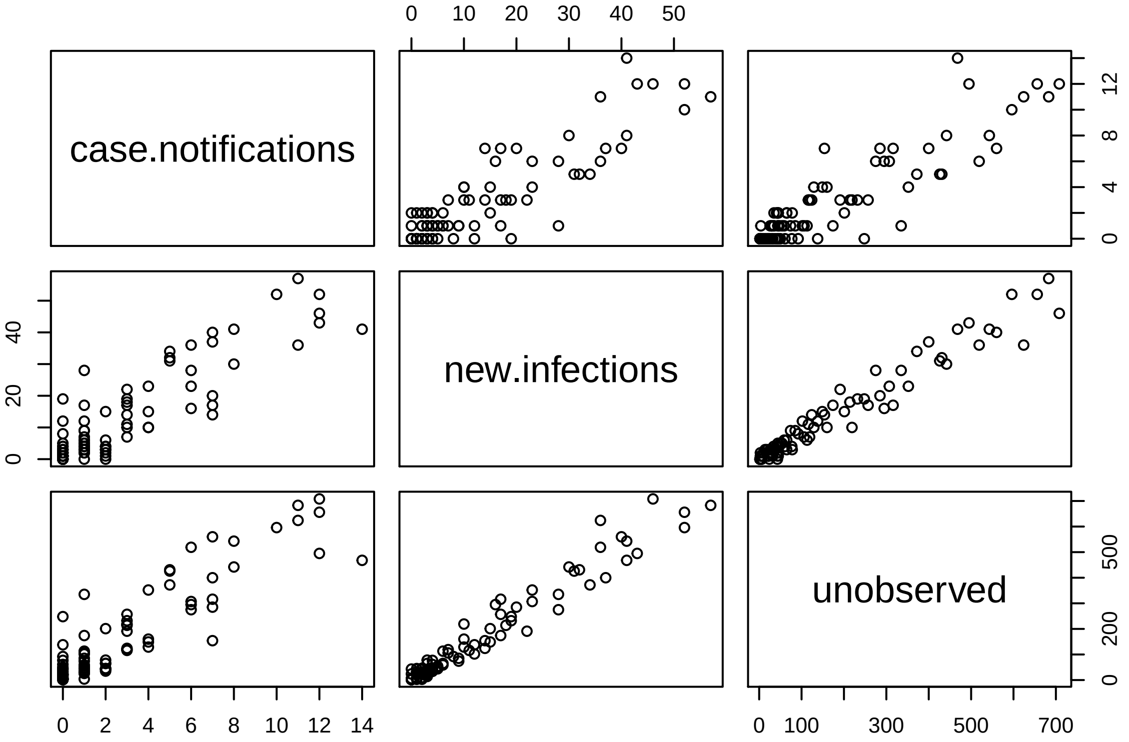
```
scenario <- 'Scenario 3'
start=as.Date("2020-03-01")
set.seed(03162020)

q <- function(t, w=12, q0=0.346, q1=0.346) ifelse(t<=w,q0,q1)

today <- Sys.Date()
out <- evaluate.model(params=list(beta0=0.6584, sigma=1/6.4, z=28, b=0.143, a0=1/1.5, w=21, presymptoma
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=0, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                                nstep = (as.numeric(today-start)+1+84)/0.05,    # solve for 12 weeks fro
                                nsims=1, start=as.Date("2020-03-01"))
plot.model(out, log='', title=paste('Nowcast validation:', scenario))
```



```
out <- out[[1]]
report.times <- seq(1,(length(out$cum.time)), by=20)
case.notifications <- diff(out$C[report.times])
plot(case.notifications)
```

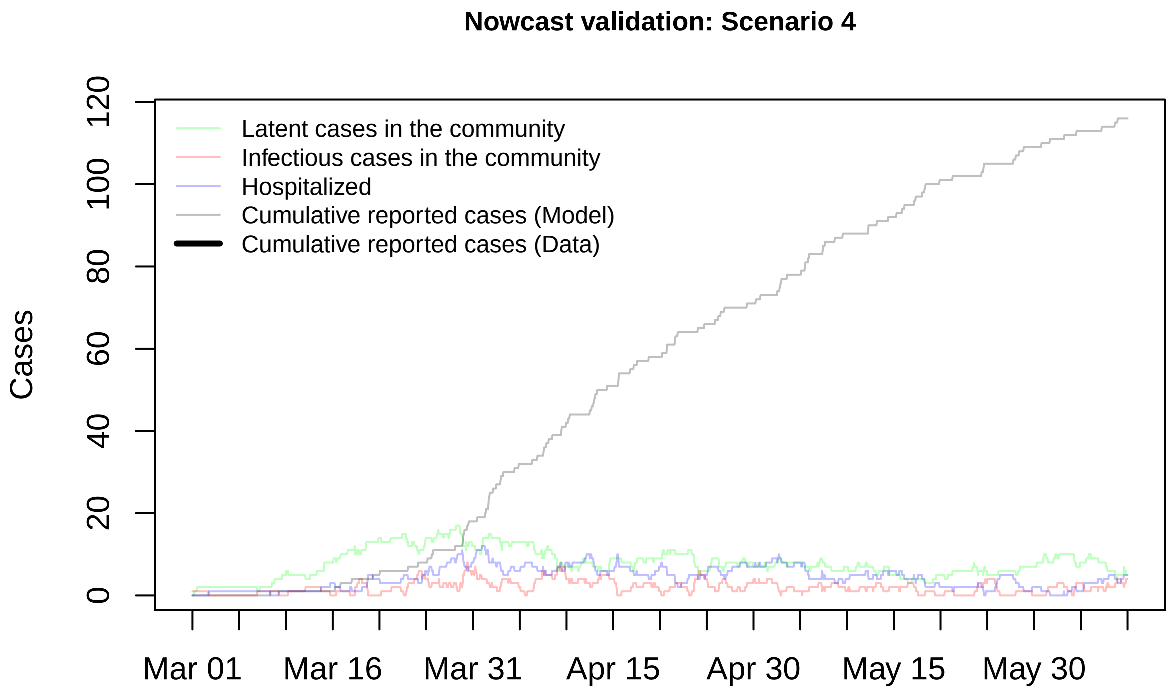


```
unobserved <- out$E1+out$E2+out$E3+out$E4+out$E5+out$E6+out$I1+out$I2+out$I3+out$I4+out$Iu1+out$Iu2+out
unobserved <- unobserved[tail(report.times,-1)]
new.infections <- head(-1*aggregate(diff(out$S), by=list(tail(floor(out$cum.time),-1)), FUN=sum)$x,-1)
nowcast.validation <- data.frame(case.notifications, new.infections, unobserved)
plot(nowcast.validation)
```

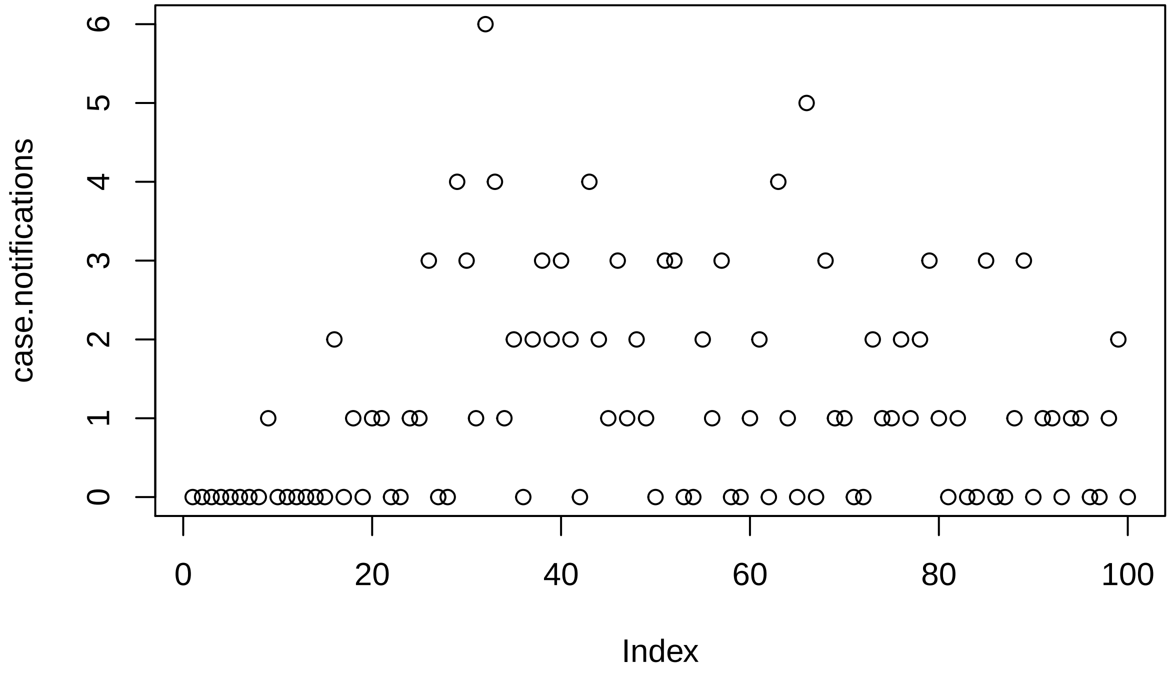


```
write.csv(nowcast.validation, file=paste('nowcast-validation-',scenario,'.csv', sep=''))
```

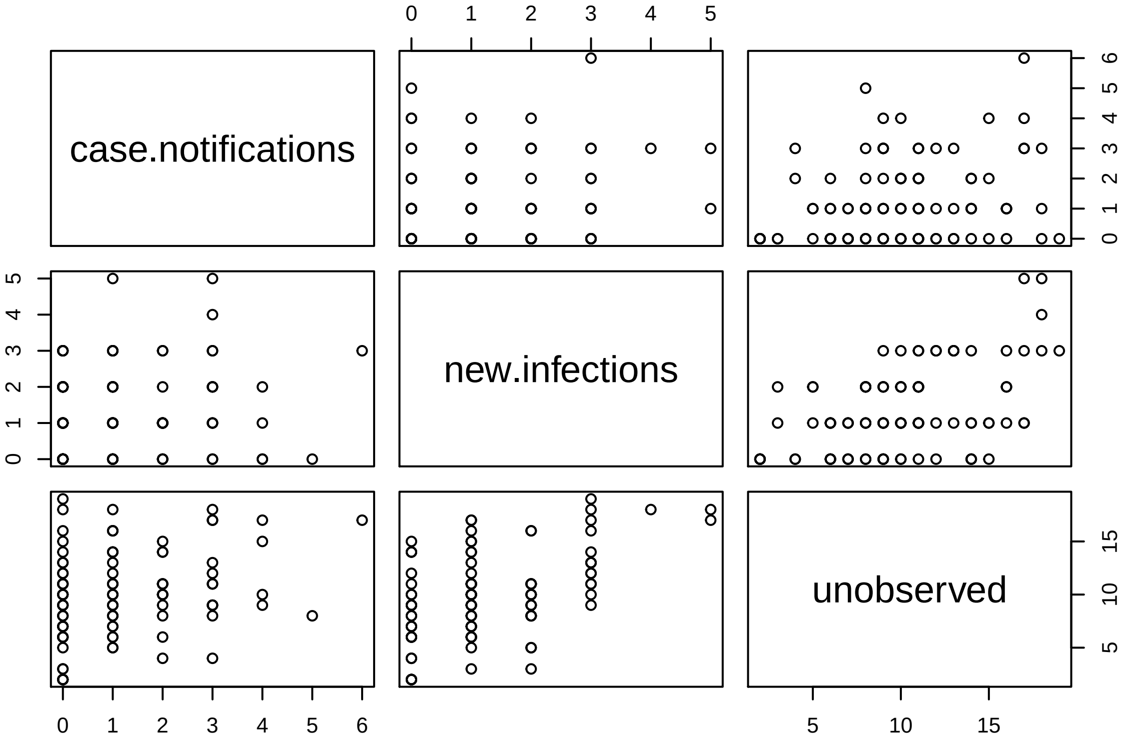
```
scenario <- 'Scenario 4'
start=as.Date("2020-03-01")
set.seed(03162020)

q <- function(t, w=12, q0=1, q1=1) ifelse(t<=w,q0,q1)

today <- Sys.Date()
out <- evaluate.model(params=list(beta0=0.6584, sigma=1/6.4, z=14, b=0.143, a0=1/1.5, w=21, presymptomat
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=0, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                                nstep = (as.numeric(today-start)+1+84)/0.05,    # solve for 12 weeks fro
                                nsims=1, start=as.Date("2020-03-01"))
plot.model(out, log='', title=paste('Nowcast validation:', scenario))
```



```
out <- out[[1]]
report.times <- seq(1,(length(out$cum.time)), by=20)
case.notifications <- diff(out$C[report.times])
plot(case.notifications)
```

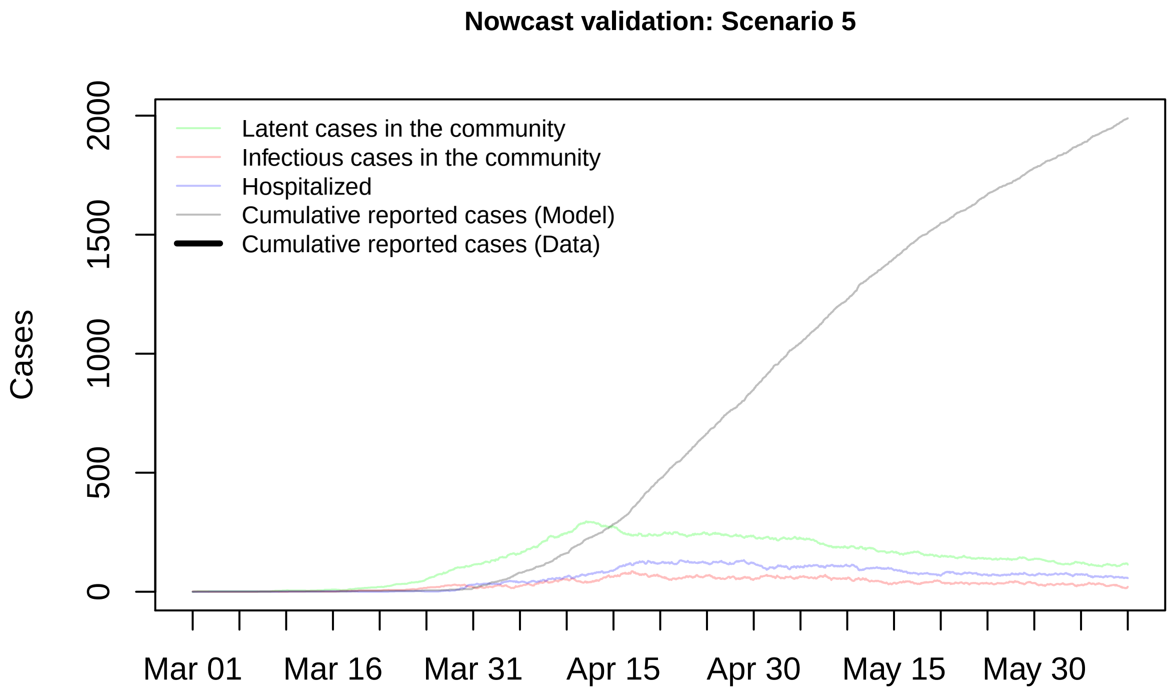


```
unobserved <- out$E1+out$E2+out$E3+out$E4+out$E5+out$E6+out$I1+out$I2+out$I3+out$I4+out$Iu1+out$Iu2+out
unobserved <- unobserved[tail(report.times,-1)]
new.infections <- head(-1*aggregate(diff(out$S), by=list(tail(floor(out$cum.time),-1)), FUN=sum)$x,-1)
nowcast.validation <- data.frame(case.notifications, new.infections, unobserved)
plot(nowcast.validation)
```

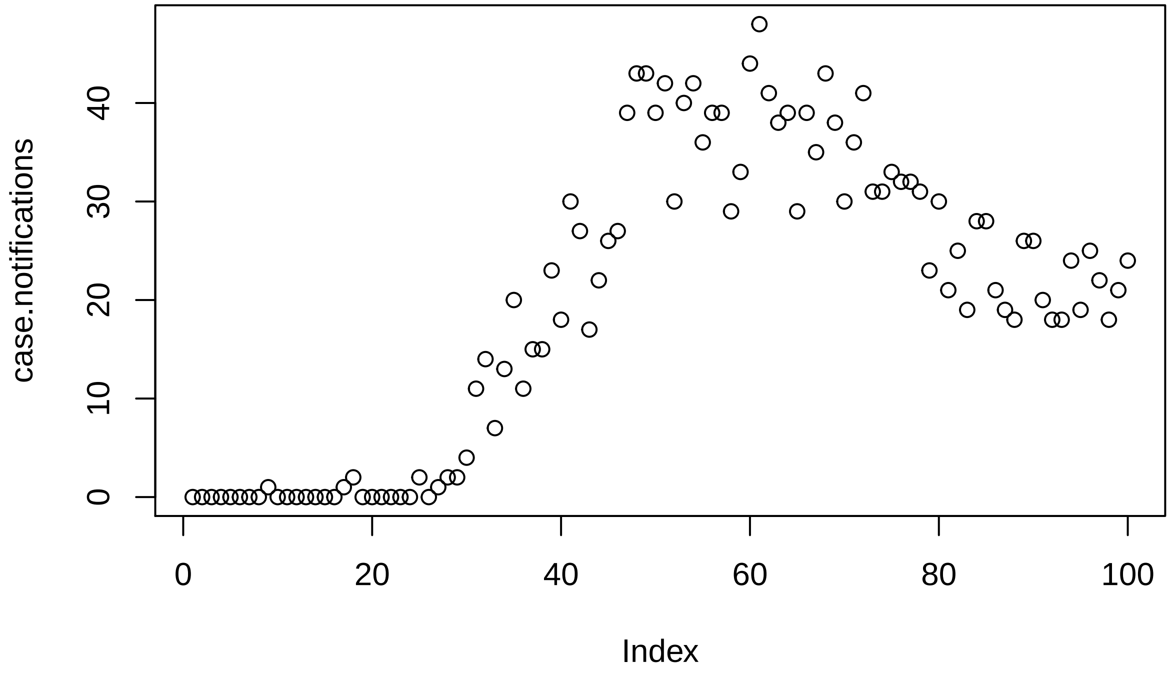


```
write.csv(nowcast.validation, file=paste('nowcast-validation-',scenario,'.csv', sep=''))
```

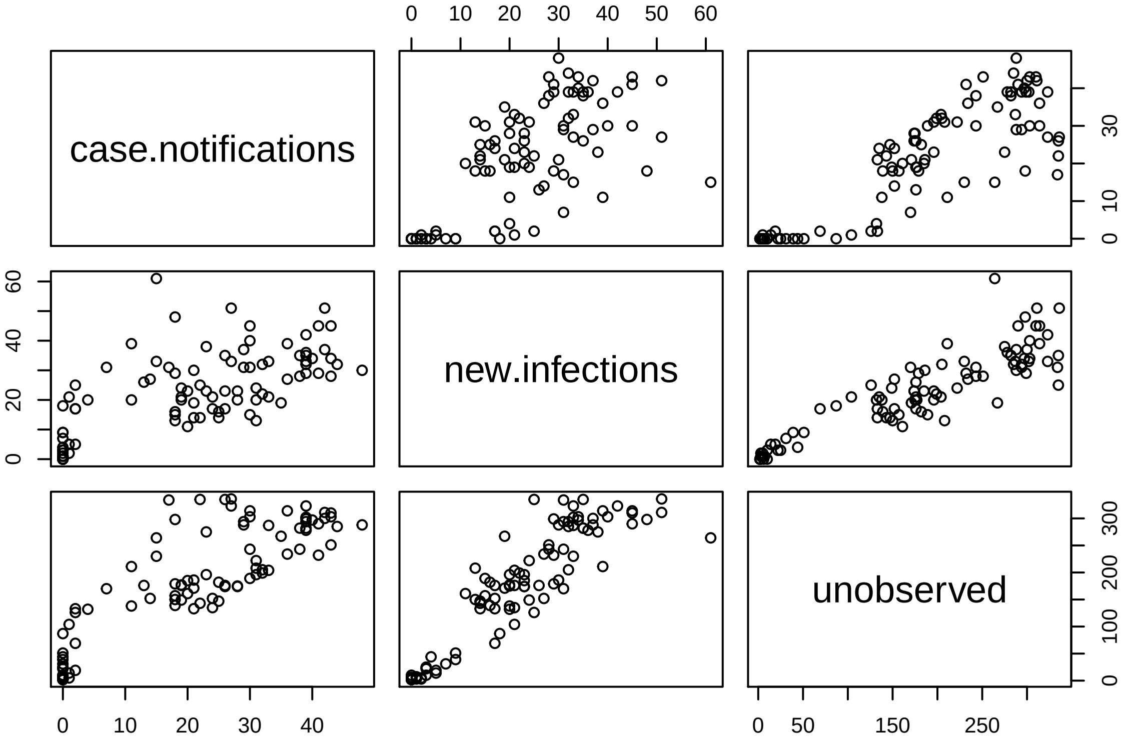
```
scenario <- 'Scenario 5'
start=as.Date("2020-03-01")
set.seed(03162020)

q <- function(t, w=12, q0=1, q1=1) ifelse(t<=w,q0,q1)

today <- Sys.Date()
out <- evaluate.model(params=list(beta0=0.6584, sigma=1/6.4, z=28, b=0.143, a0=1/1.5, w=42, presymptoma
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=0, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                                nstep = (as.numeric(today-start)+1+84)/0.05,   # solve for 12 weeks fro
                                nsims=1, start=as.Date("2020-03-01"))
plot.model(out, log='', title=paste('Nowcast validation:', scenario))
```



```
out <- out[[1]]
report.times <- seq(1,(length(out$cum.time)), by=20)
case.notifications <- diff(out$C[report.times])
plot(case.notifications)
```

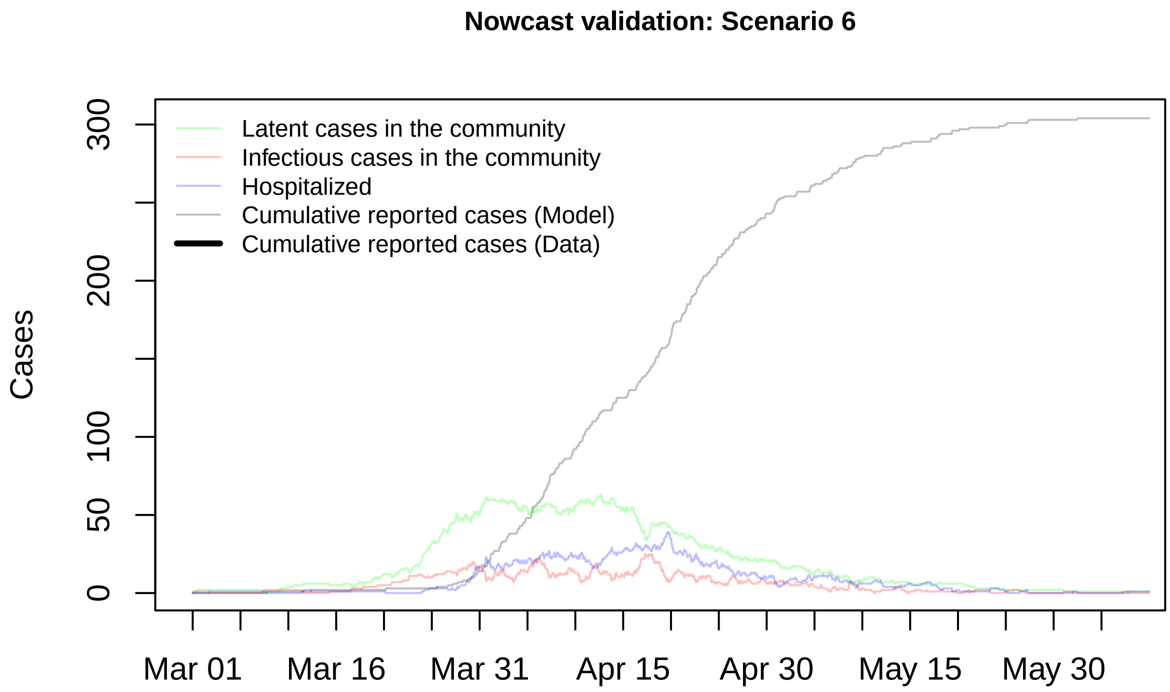


```
unobserved <- out$E1+out$E2+out$E3+out$E4+out$E5+out$E6+out$I1+out$I2+out$I3+out$I4+out$Iu1+out$Iu2+out
unobserved <- unobserved[tail(report.times,-1)]
new.infections <- head(-1*aggregate(diff(out$S), by=list(tail(floor(out$cum.time),-1)), FUN=sum)$x,-1)
nowcast.validation <- data.frame(case.notifications, new.infections, unobserved)
plot(nowcast.validation)
```

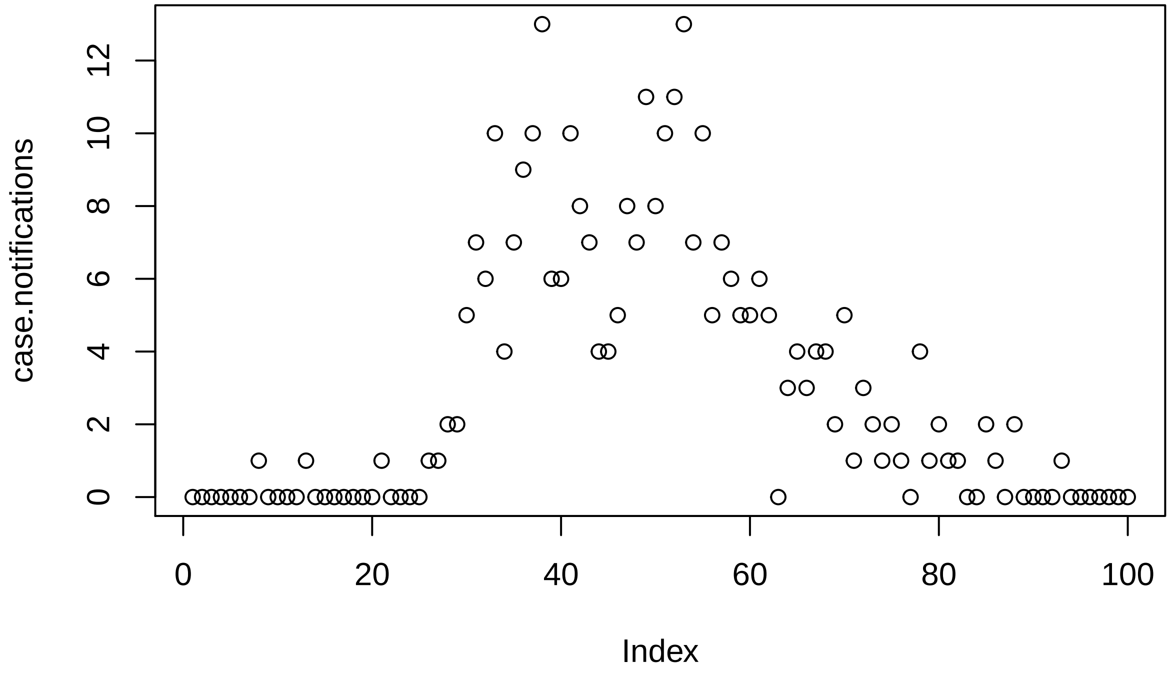


```
write.csv(nowcast.validation, file=paste('nowcast-validation-',scenario,'.csv', sep=''))
```

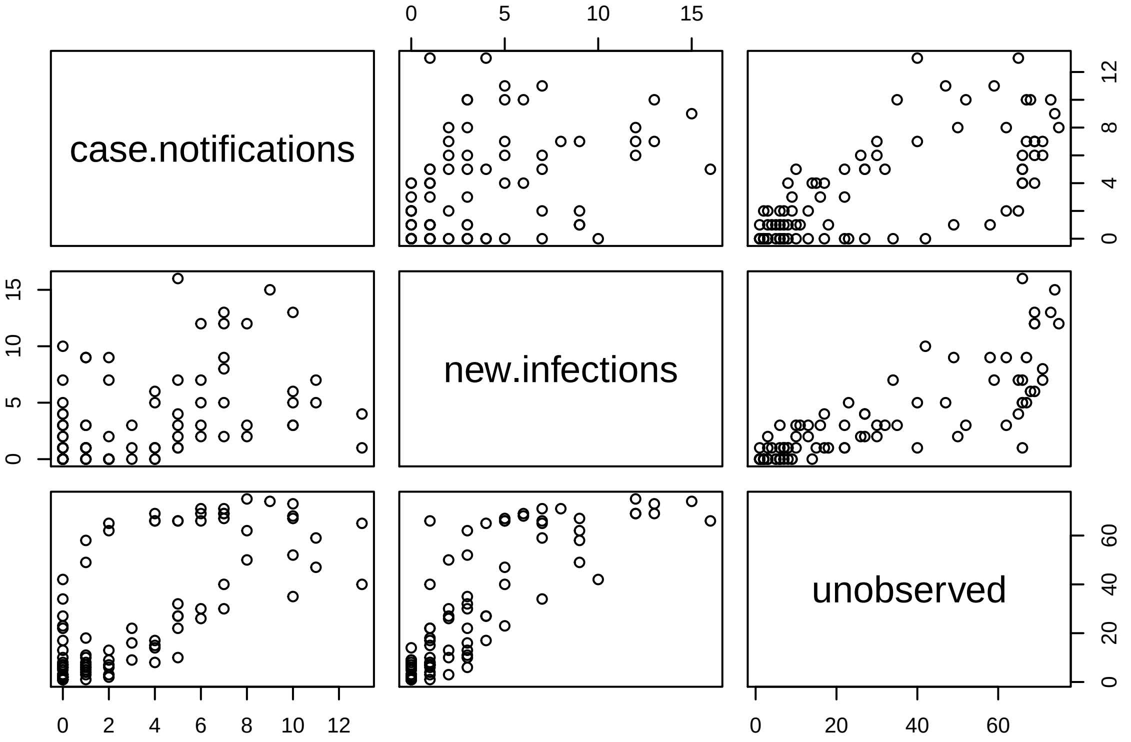
```
scenario <- 'Scenario 6'
start=as.Date("2020-03-01")
set.seed(03162020)

q <- function(t, w=12, q0=1, q1=1) ifelse(t<=w,q0,q1)

today <- Sys.Date()
out <- evaluate.model(params=list(beta0=0.6584, sigma=1/6.4, z=28, b=0.143, a0=1/1.5, w=42, presymptomat
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=0, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                                nstep = (as.numeric(today-start)+1+84)/0.05,   # solve for 12 weeks fro
                                nsims=1, start=as.Date("2020-03-01"))
plot.model(out, log='', title=paste('Nowcast validation:', scenario))
```



```
out <- out[[1]]
report.times <- seq(1,(length(out$cum.time)), by=20)
case.notifications <- diff(out$C[report.times])
plot(case.notifications)
```

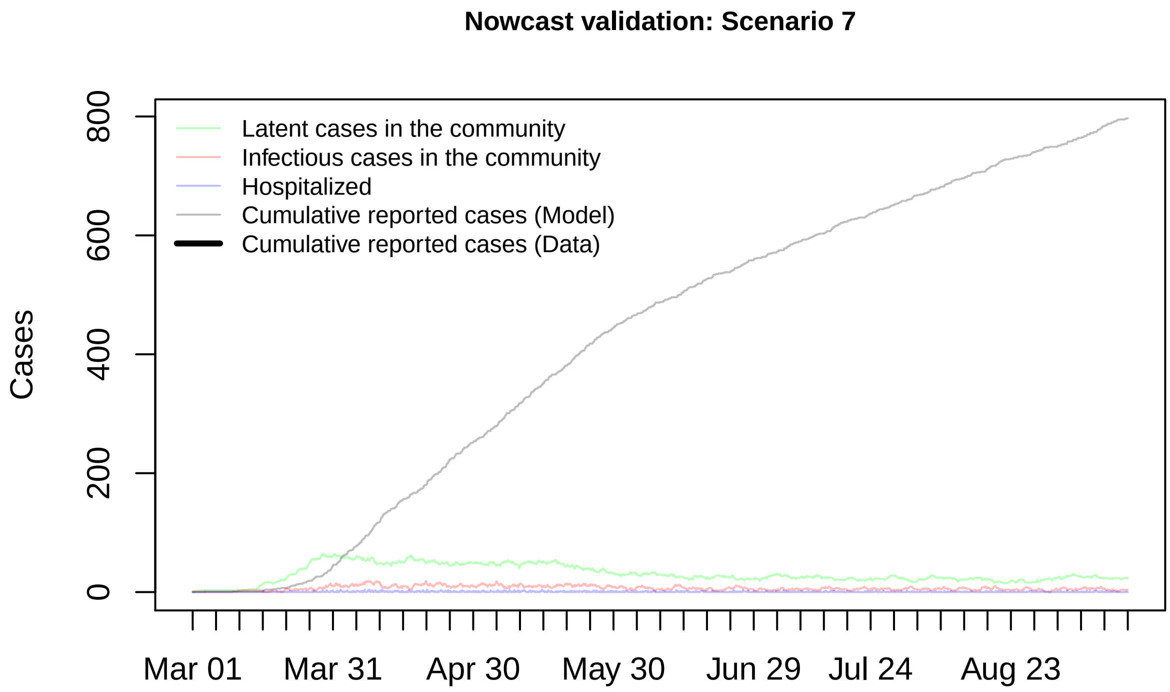


```
unobserved <- out$E1+out$E2+out$E3+out$E4+out$E5+out$E6+out$I1+out$I2+out$I3+out$I4+out$Iu1+out$Iu2+out
unobserved <- unobserved[tail(report.times,-1)]
new.infections <- head(-1*aggregate(diff(out$S), by=list(tail(floor(out$cum.time),-1)), FUN=sum)$x,-1)
nowcast.validation <- data.frame(case.notifications, new.infections, unobserved)
plot(nowcast.validation)
```

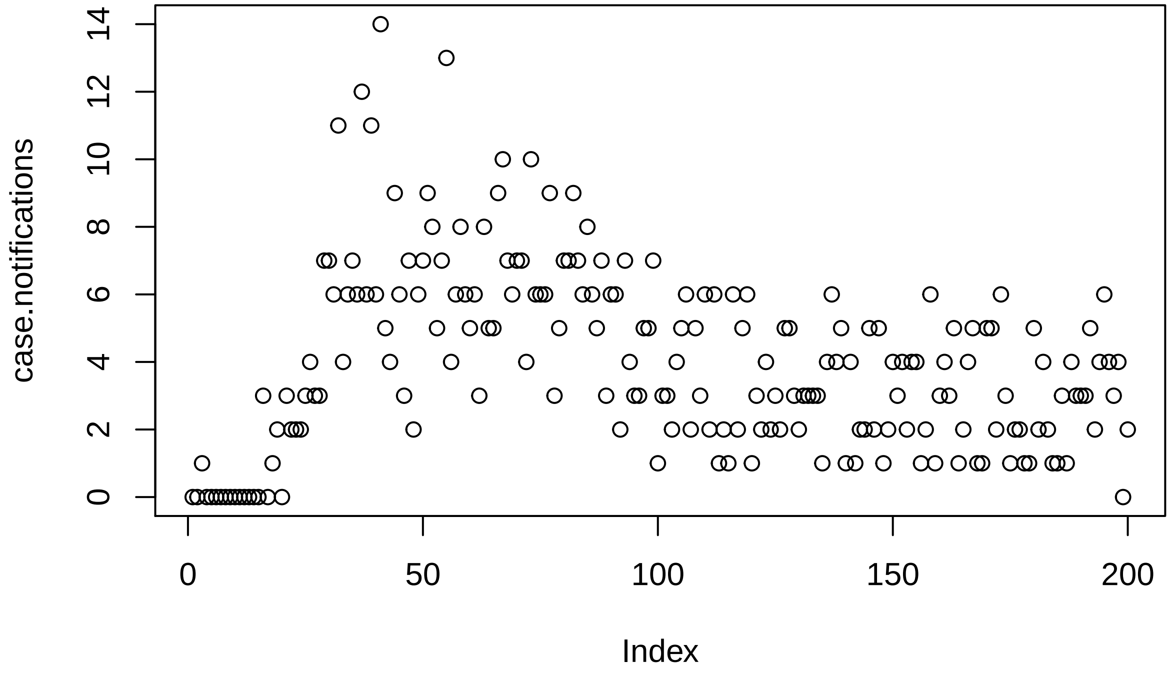


```
write.csv(nowcast.validation, file=paste('nowcast-validation-',scenario,'.csv', sep=''))
```

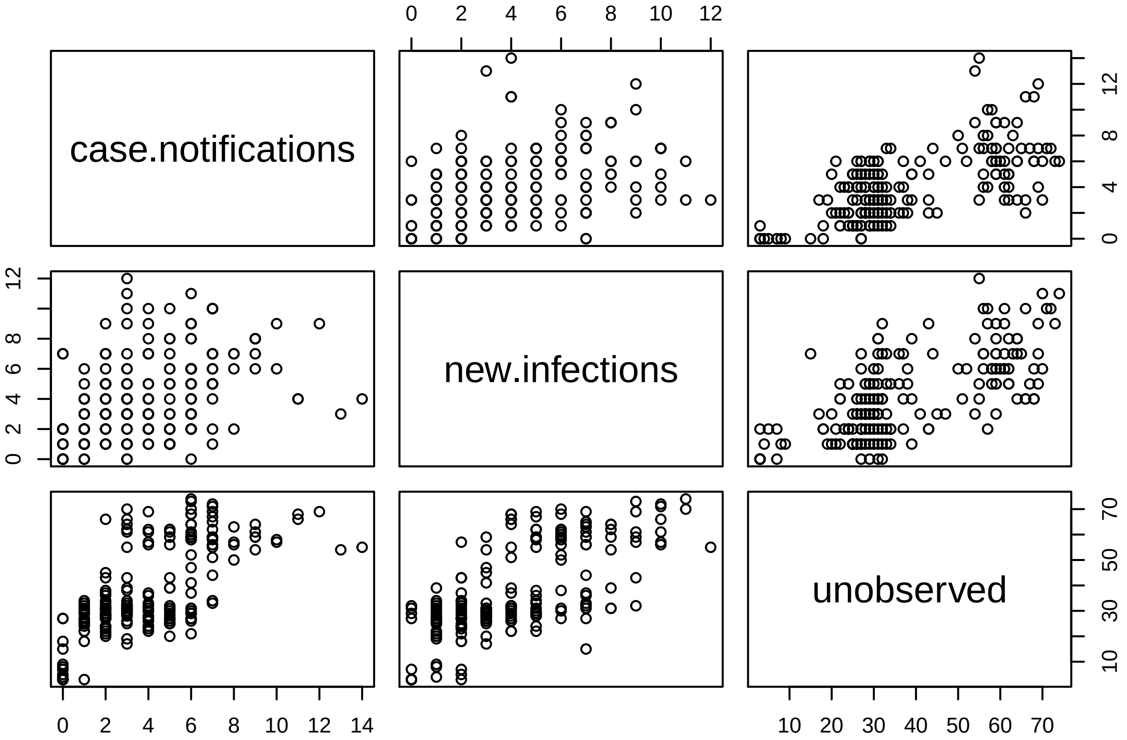
```
scenario <- 'Scenario 7'
start=as.Date("2020-03-01")
set.seed(03162020)

q <- function(t, w=12, q0=1, q1=1) ifelse(t<=w,q0,q1)
eta <- function(t, w=12) ifelse(t<=w,10,10)
beta <- function(t, w=12, beta0=0.6584, beta.factor=2) ifelse(t<=w,beta0,beta0/beta.factor)

today <- Sys.Date()
out <- evaluate.model(params=list(beta0=0.6584, sigma=1/7.67, z=14, b=0.143, a0=1/1.5, w=28, presymptoma
                                       init = list(S=10600000, E1=0, E2=0, E3=0, E4=0, E5=0, E6=0,
                                                   I1 = 1, I2= 0, I3=0, I4=0, Iu1=0, Iu2=0, Iu3=0, Iu4=
                                                   H=0, Ru=0, C=0),
                                nstep = (as.numeric(today-start)+1+184)/0.05,   # solve for 12 weeks fr
                                nsims=1, start=as.Date("2020-03-01"))
plot.model(out, log='', title=paste('Nowcast validation:', scenario))
```



```
out <- out[[1]]
report.times <- seq(1,(length(out$cum.time)), by=20)
case.notifications <- diff(out$C[report.times])
plot(case.notifications)
```



```
unobserved <- out$E1+out$E2+out$E3+out$E4+out$E5+out$E6+out$I1+out$I2+out$I3+out$I4+out$Iu1+out$Iu2+out
unobserved <- unobserved[tail(report.times,-1)]
new.infections <- head(-1*aggregate(diff(out$S), by=list(tail(floor(out$cum.time),-1)), FUN=sum)$x,-1)
nowcast.validation <- data.frame(case.notifications, new.infections, unobserved)
plot(nowcast.validation)
```



```r
write.csv(nowcast.validation, file=paste('nowcast-validation-',scenario,'.csv', sep=''))
```