The effective reproduction number of pandemic influenza: Prospective estimation

TECHNICAL APPENDIX

Outline of methods

This technical appendix provides further details of a method to estimate the effective reproduction number R_t during an epidemic based on the epidemic curve of illness onset dates of infected individuals (or a subset of cases such as notified cases, hospitalized cases) and an assumed or known serial interval distribution. For an epidemic in which the n_t cases with illness onset on day t subsequently infect X_t secondary cases, the reproduction number at time t, R_t, is defined as X_t / n_t. While n_t is observable from the epidemic curve, X_t is usually unobservable without extensive and detailed contact tracing. An estimator for R_t was proposed by Wallinga and Teunis by considering all possible transmission networks to infer the secondary cases X_t, using a likelihood-based approach.[1] The method was extended by Cauchemez et al. to allow prospective ('real-time') estimation during the course of an epidemic.[2] We describe the key steps to estimate R_t here. R syntax to reproduce the methods described here are available from the corresponding author on request.

Cauchemez et al. showed that the probability that a case with illness onset on day $k \le T$ has been infected by a case with illness onset on day $t \le T$ is given by

$$p_{tk} = \frac{(n_t - 1\{t = k\})w(k - t)}{\sum_{l \le T}(n_t - 1\{t = l\})w(l - t)}$$

where $1\{\ \}$ is the indicator function and $w(\ )$ is the Weibull density function. We assumed a Weibull distribution for the serial interval with mean 3.2 days and standard deviation 1.3 days based on data from a household-based study of 2009 pandemic (H1N1pdm) transmission in households in Hong Kong.[3]

In Hong Kong, H1N1pdm infection was a reportable disease throughout our study period. We estimated R_t based on all laboratory-confirmed H1N1pdm notifications, and also based on all hospitalizations associated with H1N1pdm. The two estimates of R_t should be very similar over time provided that the hospitalization rate among the cases is relatively stable over time (see Figure 1B in the main text).

Adjusting for imported cases

'Imported cases' are defined as cases that travelled to the local area after infection but before recovery from illness (as a proxy for infectiousness). Imported cases may generate local secondary cases of infection, but were not themselves infected locally. We modified the estimation algorithm proposed by Cauchemez et al. to incorporate imported cases by assigning a probability of 0 that the imported cases were infected by local cases, but allowing them to contribute to the onwards transmission chains as usual. Without this adjustment, there could be over-estimation of R_t when there is a substantial proportion of imported cases, which is possible particularly at the start of an epidemic.

Adjusting for reporting delays

We extended the methodology of Cauchemez et al. to adjust for delays in notification (reporting delays) and time from illness onset to hospitalization, which particularly affect estimates of R_t towards the end of the observation period (towards the current day in real-time analyses).

For estimation of R_t based on hospitalized cases, we assumed a bivariate Poisson distribution[4] for the reporting delay and time from illness onset to hospitalization, allowing for correlation between the two periods. The corresponding parameters were then estimated by maximum likelihood, allowing for the right-truncation at the end of the observation period. For a dataset with observation until day T', let O_i, C_i and H_i be the illness onset date, confirmation date and hospitalization date, respectively, for individual i.

Based on the conditional probability due to right-truncation of reporting delay at *T' – Oi*, the likelihood function is given by:

$$L(\theta \mid O,C,H) = \prod_{i=1}^{N} \frac{P(C_i - O_i, H_i - O_{i,} \mid \theta)}{F(T' - O_{i,} T' - O_i \mid \theta)},$$

where C_i - O_i, H_i - O_i, T' - O_i, $N = \sum_{t=1}^{T'} n_t$, are the reporting delay, time from illness onset to hospitalization, the maximum observable delays and the total number of hospitalized cases respectively; *P* and *F* are the probability mass function and the cumulative distribution function of a bivariate Poisson distribution with means $\theta = (\theta_1, \theta_2)$. The likelihood was maximized to obtain an estimate for the parameters θ. Delays of longer than 10 days were very rare so we restricted estimation to 10 days. The overall empirical and estimated bivariate distribution for the onset to hospitalization period and the reporting delay are shown in Figure A1 below. In practice we estimated empirical reporting delay distributions for time t prospectively based on data available for the week preceding t i.e. the period (t-7, t).

Given the estimated delay distribution, we estimated the actual number of hospitalized cases with illness onset on day *T'-k*, by $n / F(k,k \mid \hat{\theta})$. This assumed that the number of hospitalized cases reported at day *T'-k* follows a Binomial distribution with parameters N_k and $F(k,k \mid \hat{\theta})$, where N_k is the actual number of hospitalized cases with illness onset on day *T'-k*.

For estimation of R_t based on all notified cases, only adjustment of reporting delay is necessary. The probability mass function for the confirmation was estimated based on the likelihood given by

$$L(\delta \mid O,C) = \prod_{i=1}^{m} \frac{P'(C_i - O_i \mid \delta)}{F'(T' - O_i \mid \delta)}$$

where $m = \sum_{t=1}^{T'} m_t$ is the total number of cases; *P'* and *F'* are the probability mass function and the cumulative distribution function of a univariate Poisson distribution. The overall empirical and estimated distribution for the reporting delay is shown in Figure A1c. Similarly, we estimated the actual number of cases with illness onset on day *T'-k*, by $m / F'(k,k \mid \hat{\delta})$.

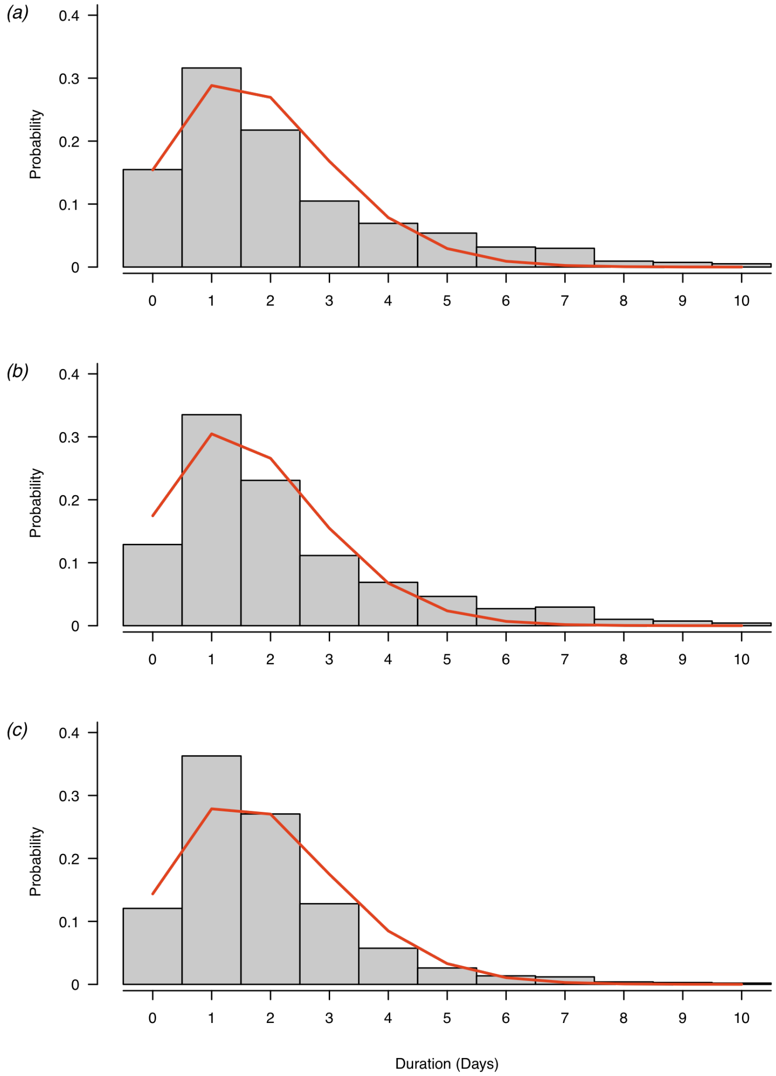

Figure A1. *Empirical distribution (bars) and fitted Poisson distribution (red lines) for (a) onset-to-hospitalization period of hospitalized cases (marginal distribution); (b) reporting delay of laboratory-confirmed cases of hospitalized cases (marginal distribution); (c) reporting delay of laboratory-confirmed cases of all cases.*

Before adjustment for reporting delays, we imputed the small fraction of missing illness onset dates in the dataset. On each day *t*, for each case with missing illness onset date *Oi*, we resampled with replacement the reporting delay Di^* based on the empirical distribution of onset-to-confirmation delays from cases with confirmation dates within the week preceding day *t*. Then the missing illness onset date was imputed by $Oi^* = Ci - Di^*$.

Based on the dataset adjusted for delays and missing illness onset dates, we estimated X_t and then R_t according to the relation $R_t = X_t / n_t$. X_{t} is decomposed into the sum of $X_t^-(T)$, the number of secondary cases with illness onset on or before day *T* who were infected by cases with illness onset on day *t*, and $X_t^+(T)$, the number of secondary cases with illness onset after day *T* who were infected by cases with illness onset on day *t*, so that $X_t^-(T)$ + $X_t^+(T)$ = X_{t}. $X_t^-(T)$ will then follow a binomial distribution:

$$X_t^-(T) \sim \sum_{k \le T} \mathrm{Bin}\left(n_k, p_{tk}\right)$$

Under the assumption that X_t follows a Poisson distribution with mean $n_t\lambda$, where we assumed λ has a vague Gamma prior with $\alpha = 10^{-5}$ and $\beta = 10^{-5}$, $X_t^+(T)$ will follow a Negative Binomial distribution with number $X_t^-(T) + \alpha$ and probability $(n_{\mathrm{t}}W(T\text{-}t) + \beta)$ / $(n_{\mathrm{t}} + \beta)$, where $W(T-t) = \sum_{u=0}^{x} w(u)$.

Sensitivity analysis

In a sensitivity analysis we prospectively estimated R_t under the alternative assumption that the serial interval followed a Weibull distribution with mean 2.6 days and standard deviation 1 day,[5-8] or mean 3.6 days and standard deviation 1.6 days.[9] The resulting real-time estimates of R_t (Figures A2 and A3) were very similar to our main results (compare with Figure 1 in the main text). Figure A4 plots the estimates of R_t for the three alternative serial interval distributions and shows no substantial differences in overall trends, with R_t slightly closer to 1 for the estimates based on shorter serial intervals.

Added value of adjusting for reporting delays

Figure A5 shows the estimates of R_t based on H1N1-associated hospitalizations, adjusting for reporting delays (as described above) versus the original method described by Cauchemez et al.[2] Without adjusting for reporting delays, the estimated real-time R_t tended to decline for the most recent few days with data available. The reporting delays in Hong Kong were fairly short and mostly less than 2-3 days (Figure A1) so the bias mainly affects estimates of R_t for approximately the most recent 1 week. If reporting delays were longer, e.g. 7 days, the bias associated with reporting delays could have a greater influence on the estimates of R_t.

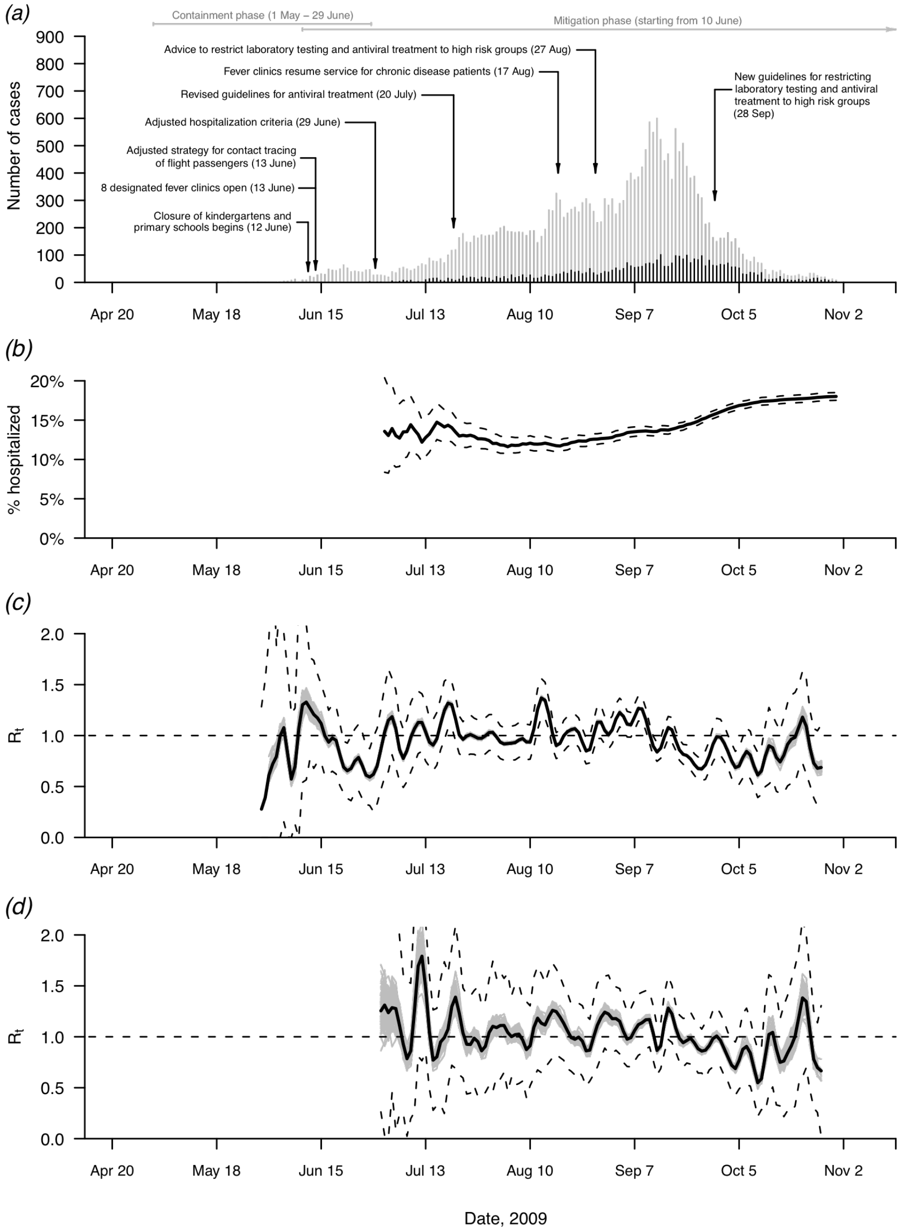

Figure A2. Sensitivity analysis based on a serial interval with mean 2.6 days. a) Number of cases of laboratory-confirmed cases of pandemic influenza A (H1N1) virus infection (gray) and hospitalizations (black) by date of illness onset; b) The cumulative proportion of hospitalized cases among all H1N1pdm notifications with 95% pointwise confidence intervals; c) Daily estimates of the effective reproduction number R_t based on H1N1pdm notifications with 95% confidence intervals; d) Daily estimates of the effective reproduction number R_t based on H1N1pdm-associated hospitalizations with 95% confidence intervals. In (c) and (d) the dashed line represents the threshold of R_t=1.

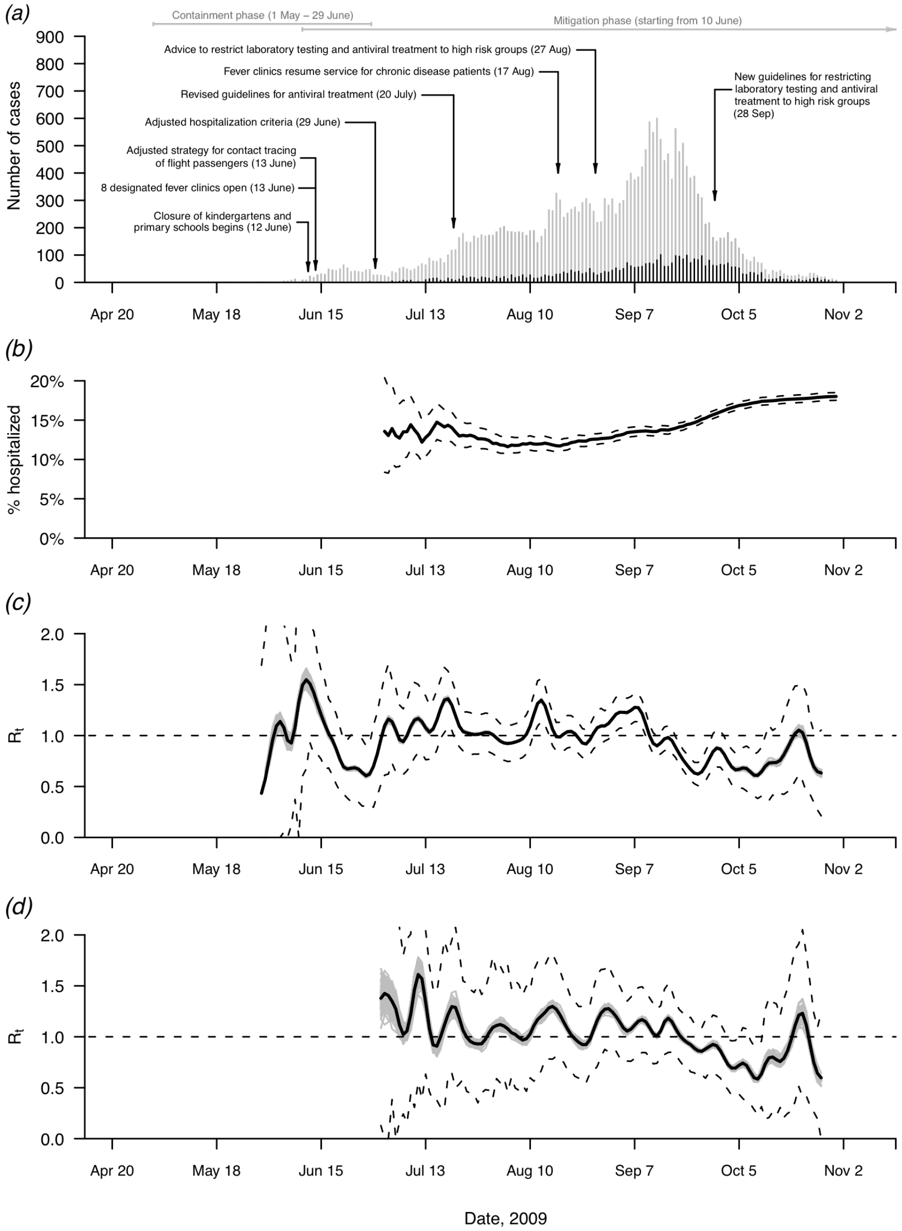

Figure A3. Sensitivity analysis based on a serial interval with mean 3.6 days. a) Number of cases of laboratory-confirmed cases of pandemic influenza A (H1N1) virus infection (gray) and hospitalizations (black) by date of illness onset; b) The cumulative proportion of hospitalized cases among all H1N1pdm notifications with 95% pointwise confidence intervals; c) Daily estimates of the effective reproduction number R_t based on H1N1pdm notifications with 95% confidence intervals; d) Daily estimates of the effective reproduction number R_t based on H1N1pdm-associated hospitalizations with 95% confidence intervals. In (c) and (d) the dashed line represents the threshold of R_t=1.

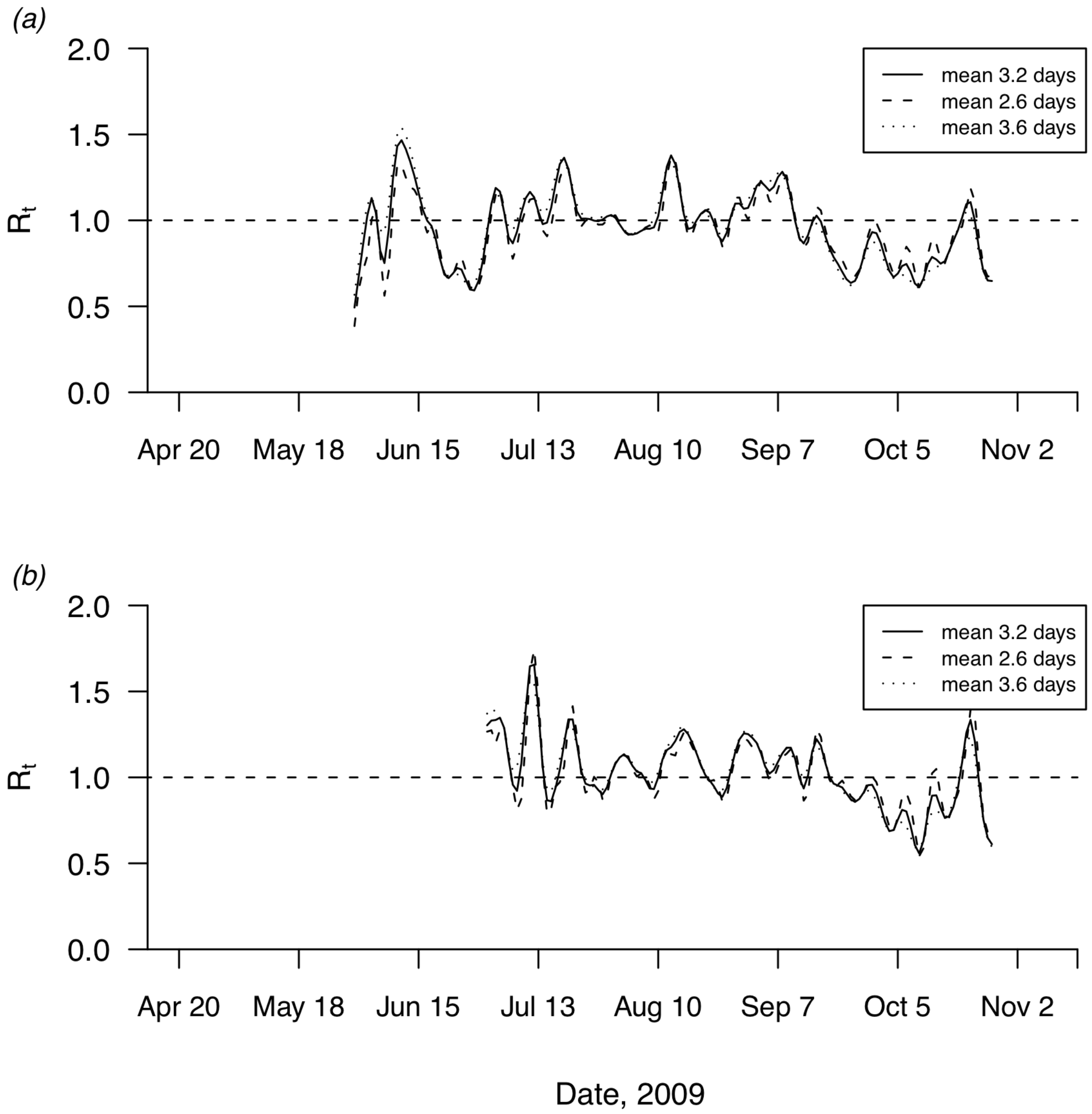

Figure A4. Estimates of R_t as presented in the main text (with serial interval with mean (sd) 3.2 (1.3) days, compared to sensitivity analyses based on a serial interval with mean (sd) 2.6 (1.0) days and 3.6 (1.6) days. Estimates of R_t are (a) based on H1N1pdm notifications; (b) based on H1N1pdm-associated hospitalizations.

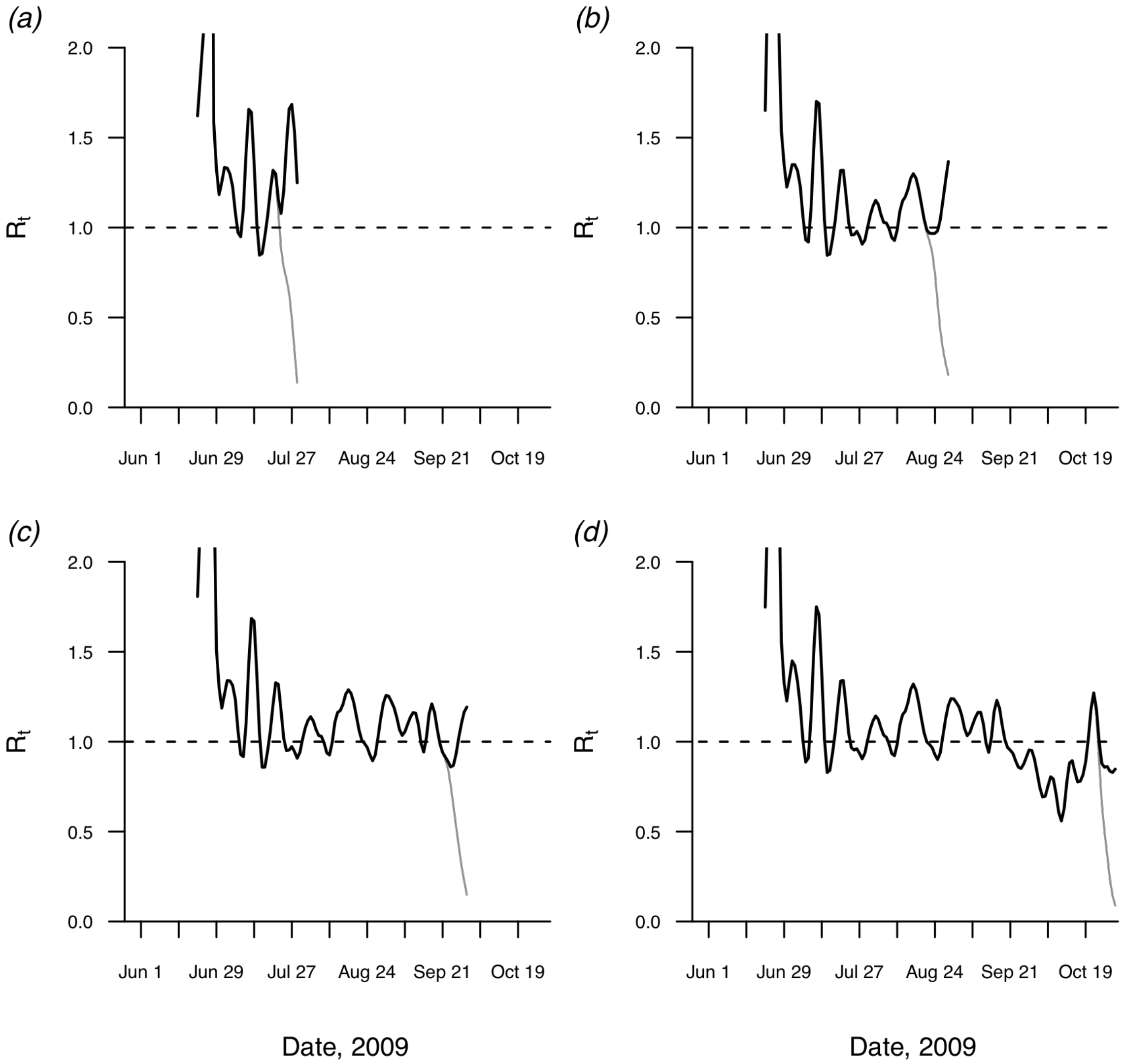

Figure A5. *The epidemic curves of H1N1pdm notifications up to different time points and corresponding real-time estimates of R_t adjusting for reporting delays (black lines) and not adjusting for reporting delays (gray lines) based on data for the periods: a) up to July 31, b) up to August 31, c) up to September 30, 2009, d) up to October 31, 2009.*

References

1. Wallinga J, Teunis P. Different epidemic curves for severe acute respiratory syndrome reveal similar impacts of control measures. *American Journal of Epidemiology.* 2004;160(6):509-16.
2. Cauchemez S, Boelle PY, Thomas G, Valleron AJ. Estimating in Real Time the Efficacy of Measures to Control Emerging Communicable Diseases. *American Journal of Epidemiology.* 2006;164(6):591-597.
3. Cowling BJ, Chan KH, Fang VJ, et al. Comparative epidemiology of pandemic and seasonal influenza A in households. *New England Journal of Medicine.* 2010;362:2175-84.
4. Kocherlakota S, Kocherlakota K. Bivariate Discrete Distributions. New York: Dekker, 1992.
5. Ferguson NM, Cummings DA, Cauchemez S, et al. Strategies for containing an emerging influenza pandemic in Southeast Asia. *Nature.* 2005;437(7056):209-14.
6. Boelle PY, Bernillon P, Desenclos JC. A preliminary estimation of the reproduction ratio for new influenza A(H1N1) from the outbreak in Mexico, March-April 2009. *Euro Surveillance.* 2009;14(19).
7. Suess T, Buchholz U, Dupke S, et al. Shedding and transmission of novel influenza virus A/H1N1 infection in households--Germany, 2009. *American Journal of Epidemiology.* 2010.
8. Cauchemez S, Donnelly CA, Reed C, et al. Transmission of novel influenza A (H1N1) virus in households in the USA. *New England Journal of Medicine.* 2009;361:2619-27.
9. Cowling BJ, Fang VJ, Riley S, Peiris JSM, Leung GM. Estimation of the serial interval of influenza. *Epidemiology.* 2009;20(3):344-7.

The following three scripts (h1n1_main.r, h1n1_functions.r and h1n1_Rt.r) include functions to reproduce the statistical methods described in our paper.

```
###
### h1n1_main.r
###

#
# The following codes calculate the Rt based on full dataset of
# laboratory confirmed H1N1 cases, and the Rt based on hospitalized
# cases only.
# Figure 1 in the paper will also be plotted.
#

# By setting a path in setwd() and putting h1n1_functions.r and
# h1n1_Rt.r in the same path, Figure 1 will be produced and saved in
the
# same path.

# setwd(...)

rm(list=ls())

require(date)
require(binom)
require(MASS)

source("h1n1_functions.r")
source("h1n1_Rt.r")

##### Data description:
# days: onset day
# cdays: h1n1 confirmation day
# hdays: hospitalization day
# hos: indicator for hospitalization
# diff: days from onset to confirmation
# diff.hos: days from onset to hospitalization
# import: indicator for imported case

#####  Read in data  #####
h1n1.total <- read.csv("h1n1_2009.csv", header=T)

# number of imputations for missing onset dates
n.impute <- 50

### Calculate Rt for full dataset
Rt.f <- Rt.f.function(h1n1.total, si.weibull.rd, n.impute)
attach(Rt.f)

### Calculate Rt for hospitalized cases only
Rt.h <- Rt.h.function(h1n1.total, si.weibull.rd, n.impute)
attach(Rt.h)


######## Panel Figures ##########
##################################
```

```
pdf("Figure1.pdf", width=6, height=8)
layout(matrix(c(1,2,3,4)),heights=c(1.5,1,1.2,1.2))

### Panel A
unique.t <- plot.t.all.rd
unique.t.h <- plot.t.all.rd.h
trunc.day <- 110

x.at <- c(96,110,138,166,194,222,250,278,306,320)
x.label <- c("Apr 6","Apr 20", "May 18","Jun 15","Jul 13","Aug 10",
  "Sep 7","Oct 5","Nov 2","")
y.at <- (0:5)*50

par(mar=c(4,4,0,0))
plot(0, xlim=c(trunc.day+1, 320),
ylim=c(0,250), axes=FALSE, type='n', xlab="", ylab="Number of cases",
cex.lab=1 )
axis(1, pos=0, at=x.at, labels=x.label, cex.axis = 0.9)
axis(2, las=1, at=y.at, cex.axis = 1)

lines(c(0,x.at[1]), c(0,0))
# black: hospitalized cases,
# white: non-hospitalized cases,
# gray(0.7): mean of the imputed onset
# gray(0.3): onset adjusted for reporting delay
for (i in ((trunc.day+1):304)){
  polygon(c(rep(i-0.2,2),rep(i+0.2,2)),
c(0,rep(all.days$cases2[i],2),0), col="black", border=F)
  polygon(c(rep(i-0.2,2),rep(i+0.2,2)),
c(all.days$cases2[i],rep(all.days$cases1[i]+all.days$cases2[i],2),
all.days$cases2[i]),
    col=gray(0.7), border=F)
}

mtext("A.", side=3, line=-0.5, outer=T, at=NA,
  adj=0, padj=1, cex=0.8, col=NA, font=2)


### Panel B
y.at <- (0:5)*0.05
y.label <- c("0%","5%","10%","15%","20%","25%")

plot(0, xlim=c(trunc.day+1,320), ylim=c(0,0.25),
xlab="", ylab="% hospitalized", axes=FALSE, type="n", cex.lab=1)
axis(1, pos=0, at=x.at, labels=x.label, cex.axis=0.9)
axis(2, las=1, at=y.at, labels=y.label, cex.axis=1)

hr <- subset(hr, (hr$t>182 & hr$t<=304))
lines(hr$t, hr$rate, lwd=2)
lines(hr$t, hr$lower, lty=2)
lines(hr$t, hr$upper, lty=2)
lines(c(0,x.at[1]), c(0,0))

mtext("B.", side=3, line=-17, outer=T, at=NA,
  adj=0, padj=1, cex=0.9, col=NA, font=2)
```

```
### Panel C
plot(0, xlim=c(trunc.day+1, 320), ylim=c(0,2),
  xlab="", ylab=expression(R[t]), axes=FALSE, type="n", cex.lab=1)
axis(1, at=x.at, labels=x.label,cex.axis=0.9, pos=0)
axis(2, las=1, cex.axis=1)

for (i in 1:n.impute){
  Rt.full.rd[[i]][1:16] <- NA
  Rt.full.rd[[i]][(length(Rt.full.rd[[i]])-3-
15):length(Rt.full.rd[[i]])] <- NA

lines(plot.t.rd[[i]][plot.t.rd[[i]]>=trunc.day+1],Rt.full.rd[[i]][plot.
t.rd[[i]]>=trunc.day+1],col=gray(0.7))
}

Rt.mean.rd[1:16] <- NA
Rt.mean.rd[(length(Rt.mean.rd)-3-15):length(Rt.mean.rd)] <- NA

Rt.mean.rd.up[1:16] <- NA
Rt.mean.rd.up[(length(Rt.mean.rd.up)-3-15):length(Rt.mean.rd.up)] <- NA

Rt.mean.rd.low[1:16] <- NA
Rt.mean.rd.low[(length(Rt.mean.rd.low)-3-15):length(Rt.mean.rd.low)] <-
NA

lines(plot.t.all.rd[plot.t.all.rd>=trunc.day+1],
Rt.mean.rd[plot.t.all.rd>=trunc.day+1],lwd=2)
lines(c(0,320),c(1,1),lty=2)

lines(plot.t.all.rd[plot.t.all.rd>=trunc.day+1],
Rt.mean.rd.up[plot.t.all.rd>=trunc.day+1],lty=2)
lines(plot.t.all.rd[plot.t.all.rd>=trunc.day+1],
Rt.mean.rd.low[plot.t.all.rd>=trunc.day+1],lty=2)

lines(c(0,x.at[1]), c(0,0))
mtext("C.", side=3, line=-29.5, outer=T, at=NA,
  adj=0, padj=1, cex=0.9, col=NA, font=2)

### Panel D
plot(0, xlim=c(trunc.day+1, 320), ylim=c(0,2),
  xlab="Date, 2009", ylab=expression(R[t]), axes=FALSE, type="n",
cex.lab=1)
axis(1, pos=0, at=x.at, labels=x.label, cex.axis=0.9)
axis(2, las=1, cex.axis = 1)

for (i in 1:n.impute){
  Rt.full.rd.h[[i]][1:5] <- NA
  Rt.full.rd.h[[i]][(length(Rt.full.rd.h[[i]])-3-
15):length(Rt.full.rd.h[[i]])] <- NA

lines(plot.t.rd.h[[i]][plot.t.rd.h[[i]]>=trunc.day+1],Rt.full.rd.h[[i]]
[plot.t.rd.h[[i]]>=trunc.day+1],col=gray(0.7))
}

Rt.mean.rd.h[1:5] <- NA
Rt.mean.rd.h[(length(Rt.mean.rd.h)-3-15):length(Rt.mean.rd.h)] <- NA

Rt.mean.rd.up.h[1:5] <- NA
Rt.mean.rd.up.h[(length(Rt.mean.rd.up.h)-3-15):length(Rt.mean.rd.up.h)]
<- NA

Rt.mean.rd.low.h[1:5] <- NA
Rt.mean.rd.low.h[(length(Rt.mean.rd.low.h)-3-
15):length(Rt.mean.rd.low.h)] <- NA
lines(c(0,320), c(1,1), lty=2)

lines(plot.t.all.rd.h[plot.t.all.rd.h>=trunc.day+1],
Rt.mean.rd.h[plot.t.all.rd.h>=trunc.day+1],lwd=2)
lines(c(0,x.at[1]), c(0,0))
lines(plot.t.all.rd.h[plot.t.all.rd.h>=trunc.day+1],
Rt.mean.rd.up.h[plot.t.all.rd.h>=trunc.day+1],lty=2)
lines(plot.t.all.rd.h[plot.t.all.rd.h>=trunc.day+1],
Rt.mean.rd.low.h[plot.t.all.rd.h>=trunc.day+1],lty=2)

mtext("D.", side=3, line=-44.5, outer=T, at=NA,
  adj=0, padj=1, cex=0.9, col=NA, font=2)

dev.off()
```

```
###
### h1n1_functions.r
###

#
# Functions to estimate Weibull incubation distribution and then Rt
#

#
# functions to give Wallinga's estimate of R
#

wallinga1_import <- function(times, import, weibull.par){

 #function to calculate the relative likelihoods that case k
 #has been infected by case l (output is a k x l matrix)
 #Wallinga 2004 AJE

  unique.t<- unique(times)
  n <- length(unique.t)
  import.n <- onset.n <- c(NA)
  Xneg <- c(NA)

  for (i in 1:length(unique.t)){
    onset.n[i]<-length(times[which(times==unique.t[i])])
  }

  for (i in 1:length(unique.t)){
    import.n[i]<-length(times[which(times==unique.t[i] & import==1)])
  }

  output.mat<- matrix(NA, ncol=n, nrow=n)
  for(k in 1:n){
    for(l in 1:n){
      output.mat[k,l] <- dweibull(unique.t[k]-unique.t[l],
      weibull.par[2], weibull.par[1])
    }
  }

  for (i in 1:length(unique.t)){
    Xneg[i]<-0
    for (j in 1:length(unique.t)){
      temp <- as.numeric(((onset.n[j]-
import.n[j])*output.mat[j,i]))/as.numeric((onset.n%*%(output.mat[j,])))
      Xneg[i] <- sum(Xneg[i], temp, na.rm=T)
    }
    Xneg[i]<- onset.n[i]* Xneg[i]
  }

  Xneg
}  # end of function

#
# Functions to calculate Xneg.var
#

wallinga1_import_var <- function(times, import, weibull.par){
 #function to calculate the relative likelihoods that case k
 #has been infected by case l (output is a k x l matrix)
 #Wallinga 2004 AJE

  unique.t<- unique(times)
  n <- length(unique.t)
  import.n <- onset.n <- c(NA)
  Xneg.var<-c(NA)

  for (i in 1:length(unique.t)){
    onset.n[i]<-length(times[which(times==unique.t[i])])
  }

  for (i in 1:length(unique.t)){
    import.n[i]<-length(times[which(times==unique.t[i] & import==1)])
  }

  output.mat <- matrix(NA, ncol=n, nrow=n)

  for(k in 1:n){
    for(l in 1:n){
      output.mat[k,l] <- dweibull(unique.t[k]-unique.t[l],
        weibull.par[2], weibull.par[1])
    }
  }

  for (i in 1:length(unique.t)){
    Xneg.var[i] <-0
    for (j in 1:length(unique.t)){
      temp <- as.numeric(((onset.n[j]-
import.n[j])*output.mat[j,i]))/as.numeric((onset.n%*%(output.mat[j,])))
      temp <-(1-temp) * temp
      Xneg.var[i] <- sum(Xneg.var[i], temp, na.rm=T)
    }

    Xneg.var[i]<-onset.n[i]* Xneg.var[i]
  }

  Xneg.var
}  # end of function


#######################################

sumweibull <- function(n.time, t, weibull.par){
#
# Summation of Weibull weights
#
  u <- n.time-t
  output.sum <-c(NA)

  for (i in 1:(u+1)){
    output.sum[i] <- dweibull(i-1, weibull.par[2], weibull.par[1])
  }
  sum <- sum(output.sum)
}

Xpos.func <- function(Xneg, unique.t, onset, weibull.par, n.time){
```

```
#
# Function to calculate the numbers, after the T,
# of infected cases by the onset cases at day t
#

  alpha.prior <- 1e-05
  beta.prior <- 1e-05

  r <- c(NA)
  p <- c(NA)
  Xpos <- c(rep(NA, length(Xneg)))

# for those days which have at least one onset case

  for (i in 1:length(unique.t)){
    sumwei <- sumweibull(n.time, unique.t[i], weibull.par)
    r[i] <- Xneg[i]+alpha.prior
    p[i] <-(onset[i]*sumwei+beta.prior)/(onset[i]+beta.prior)
    Xpos[i] <- (r[i]*(1-p[i]))/p[i]   #expectation
  }
  Xpos
}

# density function of bivariate poisson #
dbipo <- function(milk, y1, y2){
  milk1 <- milk[1]
  milk2 <- milk[2]
  milk3 <- milk[3]

  a <- exp(-1*(milk1+milk2+milk3))
  b <- c <- d <- c(NA)

  for (i in 1:(min(y1, y2)+1)){
    j <- i-1
    b[i] <- (milk1**(y1-j))*(milk2**(y2-j))*(milk3**(j))
    c[i] <- (factorial(y1-j)*factorial(y2-j)*factorial(j))
    d[i] <- b[i]/c[i]
  }

  dbipo <- a*sum(d)
  dbipo
}

# cdf of bivariate poisson #
pbipo <- function(milk, c1, c2){
  milk1 <- milk[1]
  milk2 <- milk[2]
  milk3 <- milk[3]

  if(c1>=20){
    t <-0
    for (i in 1:(20+1)){
      for (j in 1:(20+1)){
        t <- t + dbipo(milk, i-1, j-1)
      }
    }

    out <- NA
    if (t>=0.999) out <- t
  }

  if (c1<20){
    out <- 0
    for (i in 1:(c1+1)){
      for (j in 1:(c2+1)){
        out <- out + dbipo(milk, i-1, j-1)
      }
    }
  }
out
}

# Poisson log-likelihood of reporting delay distribution with right
truncation
LL.pois.rd <- function(lambda, diff, T){
  n <- length(diff)
  if(length(T)!=n) stop("T is not the same length as diff")
  loglik <- rep(NA, n)
  for(i in 1:n){
    loglik[i] <- log(dpois(diff[i], lambda)) - log(ppois(T[i], lambda))
  }
  sum(loglik)
}

# log-likelihood function of bivariate poisson distribution
# for time from onset to hospitalization delay and
# reporting delay with right truncation

LL.pois.rd.hos <- function (milk, x){

  combineab <- function(a, b){
     c.ab <- c(a, b)
     c.ab
  }

  y <- lapply(x, combineab, b=milk)

  m <- matrix(0, nrow=21, ncol=21)
  for (i in 1:21){
    for (j in 1:21){
      m[i,j] <- dbipo(milk, i-1, j-1)
    }
  }

  LL.single.delay <- function(v, mx){     # log-likelihood for the
delay
    c <- v[1]
    h <- v[2]
    t <- v[3]
    t2 <- min(t+1, 21)
    loglik <- log(mx[c,h])-log(sum(mx[1:t2,1:t2]))
    loglik
  }
  sum(as.numeric(sapply(y, LL.single.delay, mx=m)), na.rm=T)
}
```

```
###
### h1n1_Rt.r
###

###### Rt based on full dataset #######
Rt.f.function <- function(h1n1.total, si.weibull.rd, n.impute){
# Dataset without missing onset dates
h1n1.complete <- subset(h1n1.total, !(h1n1.total$days %in% c(NA)))

# Dataset with missing onset dates
h1n1.miss <- subset(h1n1.total, (h1n1.total$days %in% c(NA)))

# Parameters for the Weibull-distributed serial interval: c(scale,
shape)
# mean=3.2, sd=1.3
si.weibull.rd <- c(3.6, 2.65)

### Estimation of the reporting delay distribution
n.dist <- 10
cdist.delay <- c(rep(NA, n.dist+1))

cdays.last <- max(h1n1.complete$cdays, na.rm=T)
T.tmp <- c(rep(cdays.last, length(h1n1.complete$diff))) -
h1n1.complete$days  #truncation time
mean.diff <- mean(h1n1.complete$diff)

# MLE estimate of the mean reporting delay with Poisson distribution
mle <- (optim(par=mean.diff, LL.pois.rd, diff=h1n1.complete$diff,
  T=T.tmp, method="L-BFGS-B", control=list(fnscale=-1)))$par

for (i in 0:n.dist){
  cdist.delay[i+1] <- ppois(i, mle)
}

cdist.x <- 0:n.dist

##### imputation of missing onset dates #####
##### the empirical distribution of the delay distribution will be
updated sequentially #####

### Function to impute onset days
impute.day.a <- function(cday){
  h1n1.seq <- subset(h1n1.complete, (h1n1.complete$cdays<=min.cday))

  diff.seq <- h1n1.seq$cdays - h1n1.seq$days
  diff.seq <- pmin(diff.seq, n.dist)
  dist.seq <- hist(diff.seq, breaks=(0:(n.dist+1))-0.5, plot=F)$density

  cday - sample(cdist.x, n.cday[which(cday.unique==cday)], replace=T,
prob=dist.seq)
}

impute.day.b <- function(cday){
### Empirical distribution of reporting delay based on last 7 days
  h1n1.seq <- subset(h1n1.complete, (h1n1.complete$cdays>=cday-6 &
h1n1.complete$cdays<=cday))

  diff.seq <- h1n1.seq$cdays - h1n1.seq$days
  diff.seq <- pmin(diff.seq, n.dist)
  dist.seq <- hist(diff.seq, breaks=(0:(n.dist+1))-0.5, plot=F)$density

  cday - sample(cdist.x, n.cday[which(cday.unique==cday)], replace=T,
prob=dist.seq)
}

### Start imputation of onset days
imputed <- list(NA)
h1n1.miss <- h1n1.miss[order(h1n1.miss$cdays),]

for(i in 1:n.impute){
  imputed[[i]] <- h1n1.miss
  min.cday <- sort(unique(h1n1.complete$cdays, na.rm=T))[7]

  cday.unique <- unique(imputed[[i]]$cdays)
  n.cday <- as.numeric(table(imputed[[i]]$cdays))

  imputed.day.a <- lapply(cday.unique[1:5], impute.day.a)
  imputed.day.b <- lapply(cday.unique[-(1:5)], impute.day.b)
  imputed.day <- c(unlist(imputed.day.a), unlist(imputed.day.b))

  imputed[[i]]$days <- imputed.day
}


##### Construct imputed datasets
full.data <- list(NA)
for (i in 1:n.impute){
  full.data[[i]] <- rbind(h1n1.complete, imputed[[i]])
  full.data[[i]] <- full.data[[i]][order(full.data[[i]]$days),]
}


# Hospitalization proportion
all.days <- data.frame(day=1:365)
all.days$cases2 <- all.days$cases1 <- 0
for(i in 1:nrow(all.days)){
  all.days$cases1[i] <- sum(h1n1.complete$days==i &
h1n1.complete$hos==0)
  all.days$cases2[i] <- sum(h1n1.complete$days==i &
h1n1.complete$hos==1)
}


## Calculate Rt for the imputed datasets adjusted for reporting delay
Rt.full <- onset.full <- plot.t <- list(NA)
onset <- c(NA)

for (kk in 1:n.impute){   # imputation of missing onset dates
  times <- full.data[[kk]]$days
  unique.t <- unique(times)
  plot.t[[kk]] <- unique.t

  full.data[[kk]]$import[is.na(full.data[[kk]]$import)] <- 0

  for (kj in 1:length(unique.t)){
    onset[kj] <- length(which(times==unique.t[kj]))
    }
  onset.full[[kk]] <- onset
}  # end of n.impute loops


temp.plot.t <- unlist(plot.t)
plot.t.new <- c(min(temp.plot.t):max(temp.plot.t))
onset.loop.plot  <- data.frame(days=plot.t.new)

for (j in 1:length(plot.t.new)){
  for (i in 1:n.impute){
    onset.loop.plot[j,i+1] <- sum(plot.t[[i]]==onset.loop.plot$days[j])
    temp2 <-
onset.full[[i]][which(plot.t[[i]]==onset.loop.plot$days[j])]
    onset.loop.plot[j,i+1] <- onset.loop.plot[j,i+1]*sum(temp2)
  }
}

onset.full.sum <- onset.loop.plot[[2]]
for (i in 3:(n.impute+1)){
  onset.full.sum <- onset.full.sum + onset.loop.plot[[i]]
}

### Adjust onset for underreporting
onset.adj.rd <- onset.loop.plot
for (i in 1:(n.dist+1)){
  if(cdist.delay[n.dist+2-i]!=1) onset.adj.rd[dim(onset.adj.rd)[1]-
(n.dist+1)+i, 2:(n.impute+1)] <-
round(as.numeric(onset.adj.rd[dim(onset.adj.rd)[1]-(n.dist+1)+i,
2:(n.impute+1)]/(cdist.delay[n.dist+2-i])), 0)
}

Rt.full.rd <- var.full.rd <- onset.full.rd <- plot.t.rd <- list(NA)
full.data.rd <- ur.data <- list(NA)

arec <- matrix(NA, nrow=(n.dist+1), ncol=n.impute)
max.t <- max(plot.t.new)

for (i in 1:n.impute){ # computation of adjusted-Rt for each imputed
dataset
  nrec <- dim(full.data[[i]])[1]
  arec[,i] <- onset.adj.rd[(dim(onset.adj.rd)[1]-
n.dist):dim(onset.adj.rd)[1],i+1]-
onset.loop.plot[(dim(onset.adj.rd)[1]-n.dist):dim(onset.adj.rd)[1],i+1]
  ur.data[[i]] <- data.frame(full.data[[i]][1:sum(arec[,i]),])
  ur.data[[i]][,] <- NA
  ur.data[[i]]$days <- rep((max.t-n.dist):max.t, arec[,i])
  ur.data[[i]]$import <- 0
  full.data.rd[[i]] <- rbind(full.data[[i]], ur.data[[i]])
  full.data.rd[[i]] <- full.data.rd[[i]][sort(full.data.rd[[i]]$days,
index.return=T)$ix,]
}

for (kk in 1:n.impute){
  times.rd <- full.data.rd[[kk]]$days
  unique.t.rd <- unique(times.rd)

  plot.t.rd[[kk]] <- unique.t.rd
  full.data.rd[[kk]]$import[is.na(full.data.rd[[kk]]$import)] <- 0
  import.rd <- full.data.rd[[kk]]$import

  onset.rd <- rep(NA, length(unique.t.rd))
  for (kj in 1:length(unique.t.rd)){
    onset.rd[kj] <- length(which(times.rd==unique.t.rd[kj]))
  }

  onset.full.rd[[kk]] <- onset.rd

  Xneg.rd <- c(NA) # vector contains numbers of infected cases
  Xneg_var.rd <- c(NA)  #variance of Xt-
  Xpos.rd <- c(NA) # after T

  n.time.rd <- unique.t.rd[length(unique.t.rd)]

  Xneg.rd <- wallinga1_import(times.rd, import.rd,
weibull.par=si.weibull.rd)
  Xpos.rd <-  Xpos.func(Xneg.rd, unique.t.rd, onset.rd, si.weibull.rd,
n.time.rd)
  Xneg_var.rd <- wallinga1_import_var(times.rd, import.rd,
weibull.par=si.weibull.rd)

  X.rd <- Xneg.rd + Xpos.rd # Xneg and Xpos based on datasets adjusted
for reporting delay

# Calculate confidence interval for Rt
  Rt.rd <- var.rd <- c(NA)

  plot.t.all <- plot.t[[1]]

  if (n.impute>1){
    for (i in 2:n.impute){
      plot.t.all <- c(plot.t.all, plot.t[[i]])
    }
  }
  plot.t.all <- unique(plot.t.all)

  for (nn in 1:length(unique.t.rd)){
    Rt.rd[nn] <- X.rd[nn]/onset.rd[nn]
    sumwei.rd <- sumweibull(n.time.rd, unique.t.rd[nn], si.weibull.rd)
    var.rd[nn] <- (((1/sumwei.rd)**2+1)*Xneg_var.rd[nn]+(1-
sumwei.rd)/((sumwei.rd)**2)*Xneg.rd[nn])/(onset.rd[nn]**2)
  }

  Rt.full.rd[[kk]] <- Rt.rd
  var.full.rd[[kk]] <- var.rd
}  # end of n.impute loops for Cau

#############################################

# Calculate mean Rt (imputed missing onset dates, adjusted for
reporting delay)
plot.t.all.rd <- unlist(plot.t.rd)
plot.t.all.rd <- sort(unique(plot.t.all.rd))
```

```
Rt.mean.rd <- Rt.median.rd <- var.mean.rd <- var.mean.rd2 <- sd.mean.rd
<- rep(NA, length(plot.t.all.rd))
Rt.mean.rd.up <- Rt.mean.rd.low <- rep(NA, length(plot.t.all.rd))

for (i in 1:length(plot.t.all.rd)){
  temp <- rep(NA, n.impute)
  for (j in 1:n.impute){
    if (length(which(plot.t.rd[[j]]==plot.t.all.rd[i]))==1){
      temp[j] <-
Rt.full.rd[[j]][which(plot.t.rd[[j]]==plot.t.all.rd[i])]
    }
  }
  if (sum(is.na(temp))!=length(temp)){
    Rt.mean.rd[i] <- mean(temp, na.rm=T)
    Rt.median.rd[i] <- median(temp, na.rm=T)
  }
}

for (i in 1:length(plot.t.all.rd)){ # multiple imputation
  temp <- temp2 <- temp3 <- rep(NA, n.impute)
  for (j in 1:n.impute){
    if (length(which(plot.t.rd[[j]]==plot.t.all.rd[i]))==1){
      temp[j] <-
Rt.full.rd[[j]][which(plot.t.rd[[j]]==plot.t.all.rd[i])]
      temp2[j] <-
var.full.rd[[j]][which(plot.t.rd[[j]]==plot.t.all.rd[i])]
      temp3[j] <- (Rt.mean.rd[i] - temp[j])**2 / (n.impute-1)
    }
  }
  if (sum(is.na(temp2))!=length(temp2)) var.mean.rd[i] <- mean(temp2,
na.rm=T)
  if (sum(is.na(temp3))!=length(temp3)) var.mean.rd2[i] <- mean(temp3,
na.rm=T)/(n.impute-1)

  sd.mean.rd[i] <- (var.mean.rd[i] + ((n.impute+1)/n.impute) *
var.mean.rd2[i])**0.5
  Rt.mean.rd.up[i] <- Rt.mean.rd[i] + 2*sd.mean.rd[i]
  Rt.mean.rd.low[i] <- max(0, Rt.mean.rd[i] - 2*sd.mean.rd[i])
}

output=list(all.days=all.days, plot.t.all.rd=plot.t.all.rd,
  Rt.full.rd=Rt.full.rd, plot.t.rd=plot.t.rd,
  Rt.mean.rd=Rt.mean.rd, Rt.mean.rd.up=Rt.mean.rd.up,
  Rt.mean.rd.low=Rt.mean.rd.low)
}


##### Rt based on hospitalized cases only #####
Rt.h.function <- function(h1n1.total, si.weibull.rd, n.impute){
### Construct dataset with hospitalized cases only
h1n1.total.h <- subset(h1n1.total, (h1n1.total$hos==1))
h1n1.total.h <- subset(h1n1.total.h, ((h1n1.total.h$diff>=0 &
h1n1.total.h$diff.hos>=0)
  | is.na(h1n1.total.h$days)==T))

# Hospitalized cases without missing onset dates
h1n1.complete.h <- subset(h1n1.total.h, !(h1n1.total.h$days %in% c(NA)))
```

```
# Hospitalized cases with missing onset dates
h1n1.miss.h <- subset(h1n1.total.h, (h1n1.total.h$days %in% c(NA)))

# Parameters for the Weibull-distributed serial interval: c(scale,
shape)
# mean=3.2, sd=1.3
si.weibull.rd <- c(3.6, 2.65)

### MLE of poisson model assumed for reporting delay#######
n.dist <- 10
cdist.delay.h <- c(rep(NA, n.dist+1))

x <- y <- c(NA)
for (i in 1:nrow(h1n1.complete.h)){
  x[i] <- h1n1.complete.h$diff[i]
  y[i] <- h1n1.complete.h$diff.hos[i]
}

day1 <- max(h1n1.complete.h$cdays, na.rm=T)
day2 <- max(h1n1.complete.h$hdays, na.rm=T)
cdays.last.h <- max(day1, day2)

T.tmp <- c(rep(cdays.last.h, length(h1n1.complete.h$diff))) -
h1n1.complete.h$days #truncation time

x <- list(NA)
for (i in 1:length(h1n1.complete.h$diff)){
  x[[i]] <- c(h1n1.complete.h$diff[i], h1n1.complete.h$diff.hos[i],
T.tmp[i])
}

# initial values for optimization
mean.init <- c(0.7,0.8,1.5)

opt.len <- nrow(h1n1.complete.h)
mle.h <- (optim(par=mean.init, LL.pois.rd.hos, x=x, method="L-BFGS-B",
  control=list(fnscale=-opt.len, factr=1e10),
  lower=c(0.1,0.1,0.1), upper=c(2,2,3)))$par

for (i in 0:n.dist){
  cdist.delay.h[i+1] <- pbipo(mle.h, i, i)
}

cdist.x <- 0:n.dist


##### imputation of missing onset dates #####
##### the empirical distribution of the delay distribution will be
updated sequentially #####

### Function to impute onset days based on hospitalized cases
impute.day.h.a <- function(cday){
  h1n1.seq <- subset(h1n1.complete.h,
(h1n1.complete.h$cdays<=min.cday.h))

  diff.seq <- h1n1.seq$cdays - h1n1.seq$days
  diff.seq <- pmin(diff.seq, n.dist)
  dist.seq <- hist(diff.seq, breaks=(0:(n.dist+1))-0.5, plot=F)$density
  cday - sample(cdist.x, n.cday.h[which(cday.unique.h==cday)],
replace=T, prob=dist.seq)
}

impute.day.h.b <- function(cday){

### Empirical distribution of reporting delay based on last 7 days
  h1n1.seq <- subset(h1n1.complete.h, (h1n1.complete.h$cdays>=cday-6 &
h1n1.complete.h$cdays<=cday))

  diff.seq <- h1n1.seq$cdays - h1n1.seq$days
  diff.seq <- pmin(diff.seq, n.dist)
  dist.seq <- hist(diff.seq, breaks=(0:(n.dist+1))-0.5, plot=F)$density

  cday - sample(cdist.x, n.cday.h[which(cday.unique.h==cday)],
replace=T, prob=dist.seq)
}

### Start imputation of onset days
imputed.h <- list(NA)
h1n1.miss.h <- h1n1.miss.h[order(h1n1.miss.h$cdays),]

for(i in 1:n.impute){
  imputed.h[[i]] <- h1n1.miss.h
  min.cday.h <- sort(unique(h1n1.complete.h$cdays, na.rm=T))[7]

  cday.unique.h <- unique(imputed.h[[i]]$cdays)
  n.cday.h <- as.numeric(table(imputed.h[[i]]$cdays))

  imputed.day.h.a <- lapply(cday.unique.h[1:5], impute.day.h.a)
  imputed.day.h.b <- lapply(cday.unique.h[-(1:5)], impute.day.h.b)
  imputed.day.h <- c(unlist(imputed.day.h.a), unlist(imputed.day.h.b))

  imputed.h[[i]]$days <- imputed.day.h
}

##### Construct imputed datasets
full.data.h <- list(NA)
onset.h <- c(NA)

for (i in 1:n.impute){
  full.data.h[[i]] <- rbind(h1n1.complete.h, imputed.h[[i]])
  full.data.h[[i]] <- full.data.h[[i]][order(full.data.h[[i]]$days),]
}

# Rt for imputed dataset adjusted for reporting delay
onset.full.h <- plot.t.h <- list(NA)

for (kk in 1:n.impute){    #loop for n.impute Cauchemez estimates
  times.h <- full.data.h[[kk]]$days
  unique.t.h <- unique(times.h)
  plot.t.h[[kk]]<- unique.t.h

  full.data.h[[kk]]$import[is.na(full.data.h[[kk]]$import)] <- 0

  for (kj in 1:length(unique.t.h)){
    onset.h[kj] <- length(which(times.h==unique.t.h[kj]))
  }

  onset.full.h[[kk]] <- onset.h
}  # end of n.impute loops

temp.plot.t.h <- unlist(plot.t.h)
plot.t.new.h <- min(temp.plot.t.h):max(temp.plot.t.h)
onset.loop.plot.h  <- data.frame(days=plot.t.new.h)

for (j in 1:length(plot.t.new.h)){
  for (i in 1:n.impute){
    onset.loop.plot.h[j,i+1] <-
sum(plot.t.h[[i]]==onset.loop.plot.h$days[j])
    temp2 <-
onset.full.h[[i]][which(plot.t.h[[i]]==onset.loop.plot.h$days[j])]
    onset.loop.plot.h[j,i+1] <-  onset.loop.plot.h[j,i+1]*sum(temp2)
  }
}

### Adjust onset for underreporting
onset.adj.rd.h <- onset.loop.plot.h
for (i in 1:(n.dist+1)){
  if(cdist.delay.h[n.dist+2-i]!=1)
onset.adj.rd.h[dim(onset.adj.rd.h)[1]-(n.dist+1)+i,2:(n.impute+1)] <-
round(as.numeric(onset.adj.rd.h[dim(onset.adj.rd.h)[1]-
(n.dist+1)+i,2:(n.impute+1)]/(cdist.delay.h[n.dist+2-i])),0)
}

Rt.full.rd.h <- var.full.rd.h <- onset.full.rd.h <- plot.t.rd.h <-
list(NA)
full.data.rd.h <- ur.data.h <- list(NA)

arec.h <- matrix(NA, nrow=(n.dist+1), ncol=n.impute)
max.t.h <- max(plot.t.new.h)

for (i in 1:n.impute){
  nrec.h <- dim(full.data.h[[i]])[1]
  arec.h[,i] <- onset.adj.rd.h[(dim(onset.adj.rd.h)[1]-
n.dist):dim(onset.adj.rd.h)[1],i+1]-
onset.loop.plot.h[(dim(onset.adj.rd.h)[1]-
n.dist):dim(onset.adj.rd.h)[1],i+1]
  ur.data.h[[i]] <- data.frame(full.data.h[[i]][1:sum(arec.h[,i]),])
  ur.data.h[[i]][,] <- NA
  ur.data.h[[i]]$days <- rep((max.t.h-n.dist):max.t.h, arec.h[,i])
  ur.data.h[[i]]$import <- 0
  full.data.rd.h[[i]] <- rbind(full.data.h[[i]], ur.data.h[[i]])
  full.data.rd.h[[i]] <-
full.data.rd.h[[i]][sort(full.data.rd.h[[i]]$days, index.return=T)$ix,]
}

###############################################

for (kk in 1:n.impute){    #loop for n.impute Cauchemez estimates
  times.rd.h <- full.data.rd.h[[kk]]$days
  unique.t.rd.h <- unique(times.rd.h)

  plot.t.rd.h[[kk]] <- unique.t.rd.h
  full.data.rd.h[[kk]]$import[is.na(full.data.rd.h[[kk]]$import)] <- 0
  import.rd.h <- full.data.rd.h[[kk]]$import

  onset.rd.h <- rep(NA, length(unique.t.rd.h))
  for (kj in 1:length(unique.t.rd.h)){
    onset.rd.h[kj] <- length(which(times.rd.h==unique.t.rd.h[kj]))
  }

  onset.full.rd.h[[kk]] <- onset.rd.h
  Xneg.rd <- c(NA) # vector contains numbers of infected cases
  Xneg_var.rd <- c(NA)  #variance of Xt-
  Xpos.rd <- c(NA) # after T

  Xneg.rd <- wallinga1_import(times.rd.h, import.rd.h,
weibull.par=si.weibull.rd)

  n.time.rd.h <- unique.t.rd.h[length(unique.t.rd.h)]
  Xpos.rd <-  Xpos.func(Xneg.rd, unique.t.rd.h, onset.rd.h,
si.weibull.rd, n.time.rd.h)
  Xneg_var.rd <- wallinga1_import_var(times.rd.h, import.rd.h,
weibull.par=si.weibull.rd)

  X.rd <- Xneg.rd + Xpos.rd

### Calculate confidence interval for Rt
  Rt.rd.h <- var.rd.h <- c(NA)

  plot.t.all.h <- unlist(plot.t.h)
  plot.t.all.h <- unique(plot.t.all.h)

  for (nn in 1:length(unique.t.rd.h)){
    Rt.rd.h[nn] <- X.rd[nn]/onset.rd.h[nn]
    sumwei.rd.h <- sumweibull(n.time.rd.h, unique.t.rd.h[nn],
si.weibull.rd)
    var.rd.h[nn] <- (((1/sumwei.rd.h)**2+1)*Xneg_var.rd[nn]+(1-
sumwei.rd.h)/((sumwei.rd.h)**2)*Xneg.rd[nn])/(onset.rd.h[nn]**2)
  }

  Rt.full.rd.h[[kk]] <- Rt.rd.h
  var.full.rd.h[[kk]] <- var.rd.h
}  # end of n.impute loops for Cau

# Calculate mean Rt (imputed missing onset dates, adjusted for
reporting delay)
plot.t.all.rd.h <- unlist(plot.t.rd.h)
plot.t.all.rd.h <- sort(unique(plot.t.all.rd.h))

Rt.mean.rd.h <- Rt.median.rd.h <- var.mean.rd.h <- var.mean.rd2.h <-
sd.mean.rd.h <- rep(NA, length(plot.t.all.rd.h))
Rt.mean.rd.up.h <- Rt.mean.rd.low.h <- rep(NA, length(plot.t.all.rd.h))

for (i in 1:length(plot.t.all.rd.h)){
  temp <- rep(NA, n.impute)
  for (j in 1:n.impute){
    if (length(which(plot.t.rd.h[[j]]==plot.t.all.rd.h[i]))==1){
      temp[j] <-
Rt.full.rd.h[[j]][which(plot.t.rd.h[[j]]==plot.t.all.rd.h[i])]
    }
  }

  if (sum(is.na(temp))!=length(temp)){
    Rt.mean.rd.h[i] <- mean(temp, na.rm=T)
    Rt.median.rd.h[i] <- median(temp, na.rm=T)
  }
}

for (i in 1:length(plot.t.all.rd.h)){
  temp <- temp2 <- temp3 <- rep(NA, n.impute)
  for (j in 1:n.impute){
    if (length(which(plot.t.rd.h[[j]]==plot.t.all.rd.h[i]))==1){
      temp[j] <-
Rt.full.rd.h[[j]][which(plot.t.rd.h[[j]]==plot.t.all.rd.h[i])]
      temp2[j] <-
var.full.rd.h[[j]][which(plot.t.rd.h[[j]]==plot.t.all.rd.h[i])] # s(m)
square
      temp3[j] <- (Rt.mean.rd.h[i]- temp[j])**2/(n.impute-1) #
component of B head
    }
  }

  if (sum(is.na(temp2))!=length(temp2)) var.mean.rd.h[i] <- mean(temp2,
na.rm=T) # s square bar
  if (sum(is.na(temp3))!=length(temp3)) var.mean.rd2.h[i] <- mean(temp3,
na.rm=T)/(n.impute-1) # B head

  sd.mean.rd.h[i]<- (var.mean.rd.h[i] + ((n.impute+1)/n.impute)*
var.mean.rd2.h[i])**0.5
  Rt.mean.rd.up.h[i] <- Rt.mean.rd.h[i] + 2*sd.mean.rd.h[i]
  Rt.mean.rd.low.h[i] <- max(0, Rt.mean.rd.h[i] - 2*sd.mean.rd.h[i])
}

##### Calculate hospitalization rate #####
h1n1.rate <- subset(h1n1.total, (h1n1.total$cdays>180))

hr <- data.frame(t=1:365, days=NA, hdays=NA)

for(i in 1:nrow(hr)){
  hr$days[i] <- length(which(h1n1.rate$days<=i))
  hr$hdays[i] <- length(which(h1n1.rate$days<=i & h1n1.rate$hos==1))
}

hr$rate <-  hr$hdays/hr$days

cf <- binom.confint(hr$hdays, hr$days, conf.level = 0.95, methods =
"exact")

hr$lower <- cf$lower
hr$upper <- cf$upper

output <- list(plot.t.all.rd.h=plot.t.all.rd.h,
Rt.full.rd.h=Rt.full.rd.h,
  plot.t.rd.h=plot.t.rd.h, Rt.mean.rd.h=Rt.mean.rd.h,
  Rt.mean.rd.up.h=Rt.mean.rd.up.h, Rt.mean.rd.low.h=Rt.mean.rd.low.h,
hr=hr)
}
```