Basic ODE fitting

December 2, 2017

1 Preliminaries

Load packages:

```
library(fitode)
library(dplyr)
library(ggplot2); theme_set(theme_bw())
library(rbenchmark)
```

2 Simple fitting

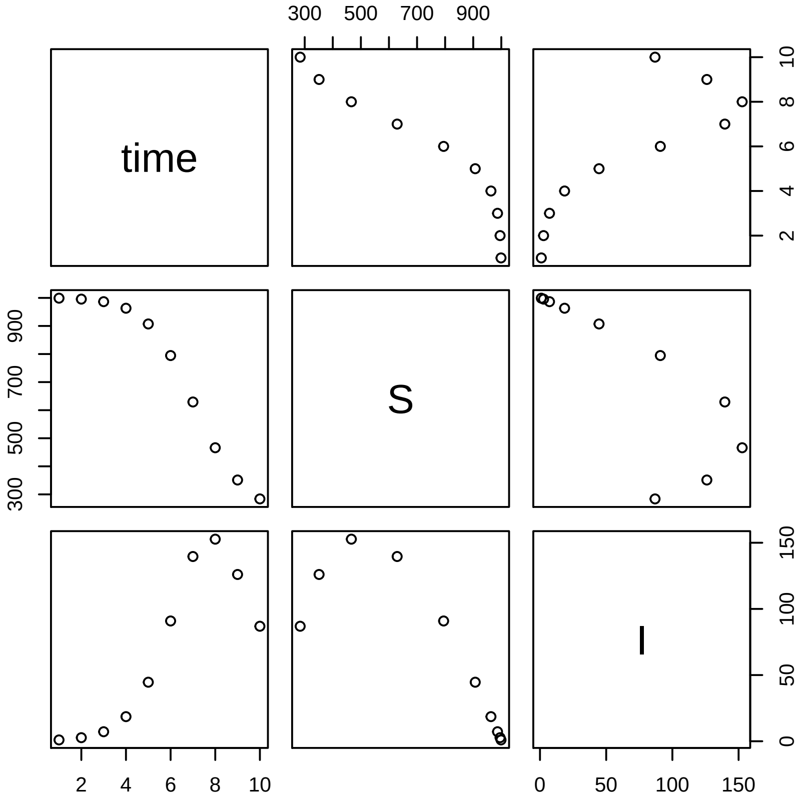
2.1 SIR model

Let's start with a simple SIR model. This is how you define a model in `fitode`:

```
sir <- new("model.ode",
    "sir model",
    model=list(
        S ~ -beta*S*I/N,
        I ~ beta*S*I/N - gamma*I
    ),
    initial=list(
        S ~ N*(1-i0),
        I ~ N*i0
    ),
    par=c("beta", "gamma", "N", "i0")

)
```

To solve this ode, one can use ‘ode.solve‘ function.

```
time <- c(1:10)
par <- c(beta=2, gamma=1, N=1000, i0=1/1000)
ss <- ode.solve(sir, time, par)
plot(ss@solution)
```



We can try to fit this model to Harbin Plague data (note that all log likelihood parameters start with prefix ‘ll.‘):

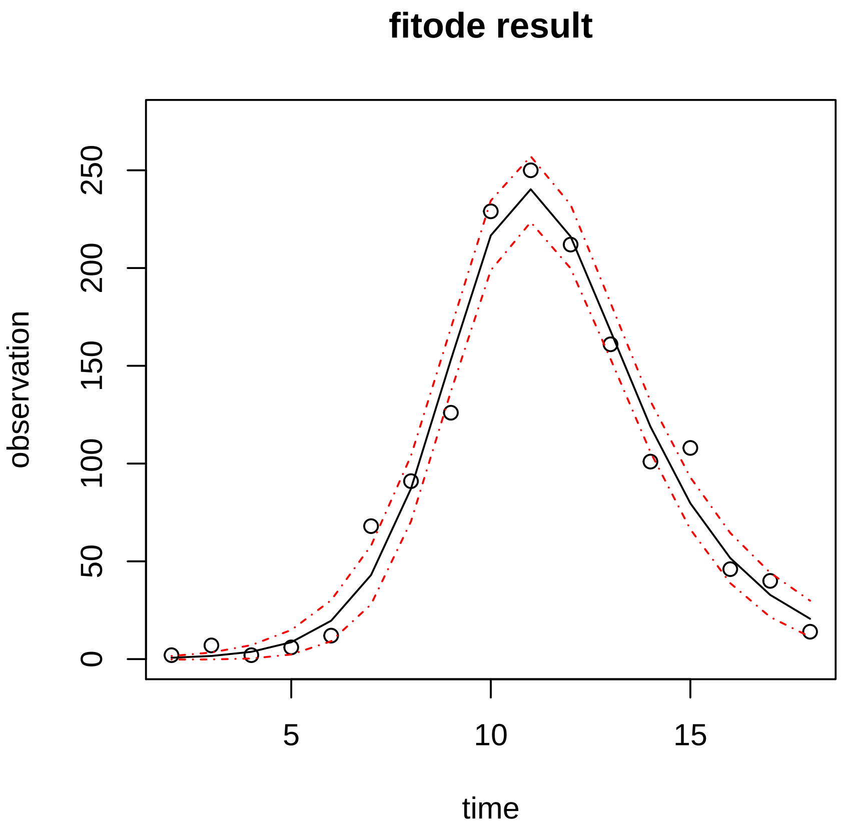
```
harbin <- fitsir::harbin

start <- c(beta=2, gamma=1, N=1e5, i0=1e-4, ll.sigma=5)

ff <- fitode(Deaths|week~gamma*I,
    start=start,
    model=sir,
    loglik=select_model("gaussian"),
    data=harbin
)


print(ff)

## Model: sir model
## Formula: Deaths | week ~ gamma * I
##
## Coefficients:
##          beta          gamma              N             i0      ll.sigma
## -1.018694e+00  1.158019e-02  1.000000e+05  4.841317e-03  1.190017e+02
##
## Log-Likelihood:-105.36
##
## link: log.ll.sigma
```

With this starting parameter, ‘fitode‘ fails to find mle. We can improve the fit by using link functions:

```
ff2 <- fitode(Deaths|week~gamma*I,
    start=start,
    model=sir,
    loglik=select_model("gaussian"),
    data=harbin,
    link=list(
        beta="log",
        gamma="log",
        N="log",
        i0="logit"
    )
)
print(ff2)

## Model: sir model
## Formula: Deaths | week ~ gamma * I
##
## Coefficients:
##         beta        gamma            N           i0     ll.sigma
## 1.623168e+00 7.704775e-01 1.815253e+03 4.892969e-04 1.334958e+01
##
## Log-Likelihood:-68.18
##
## link: log.beta log.gamma log.N logit.i0 log.ll.sigma
```

We can look at the predicted trajectory:

```
plot(ff2, level=0.95)
```

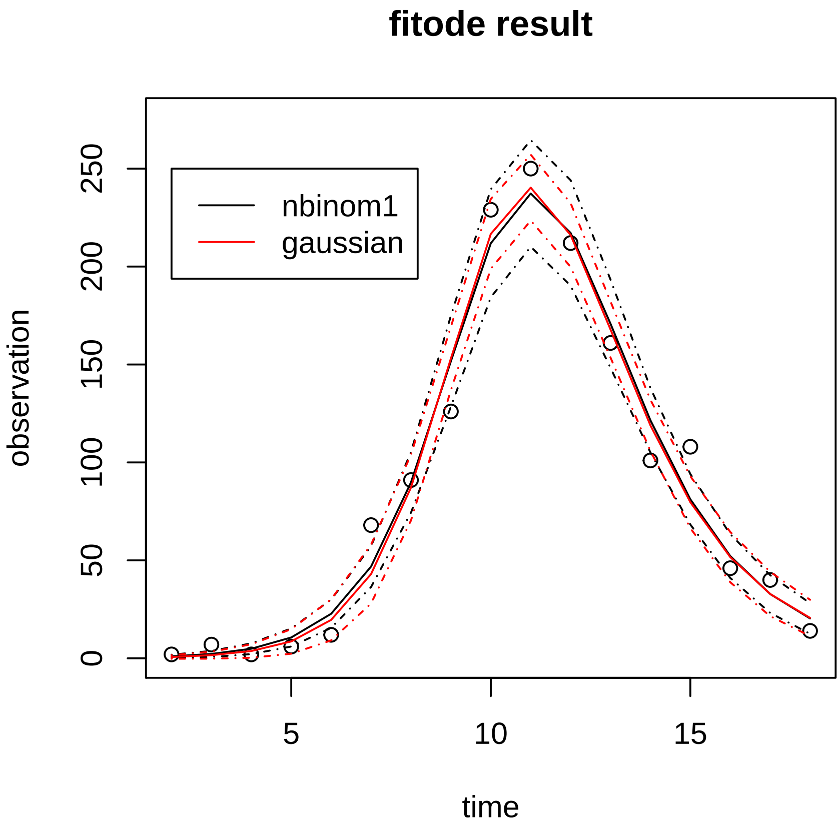


This looks pretty good but we are not dealing with overdispersion properly.

```
start2 <- c(coef(ff2)[1:4], ll.phi=2)

ff3 <- fitode(Deaths|week~gamma*I,
    start=start2,
    model=sir,
    loglik=select_model("nbinom1"),
    data=harbin,
    link=list(
        beta="log",
        gamma="log",
        N="log",
        i0="logit"
    )
)

print(ff3)

## Model: sir model
## Formula: Deaths | week ~ gamma * I
##
## Coefficients:
##         beta        gamma            N           i0       ll.phi
## 1.714493e+00 9.226696e-01 1.998729e+03 5.503243e-04 1.795928e+00
##
## Log-Likelihood:-64.59
##
## link: log.beta log.gamma log.N logit.i0 log.ll.phi
```

Compare the plots...

```
plot(ff3, level=0.95, col.traj=1, col.conf=1)
plot(ff2, level=0.95, col.traj=2, col.conf=2, add=TRUE)
legend(2, 250, legend=c("nbinom1", "gaussian"), col=c(1,2), lty=1)
```



2.2 Predator-Prey

3 Advanced fitting

3.1 Using other packages: EpiDynamics

Let's try using functions from other packages

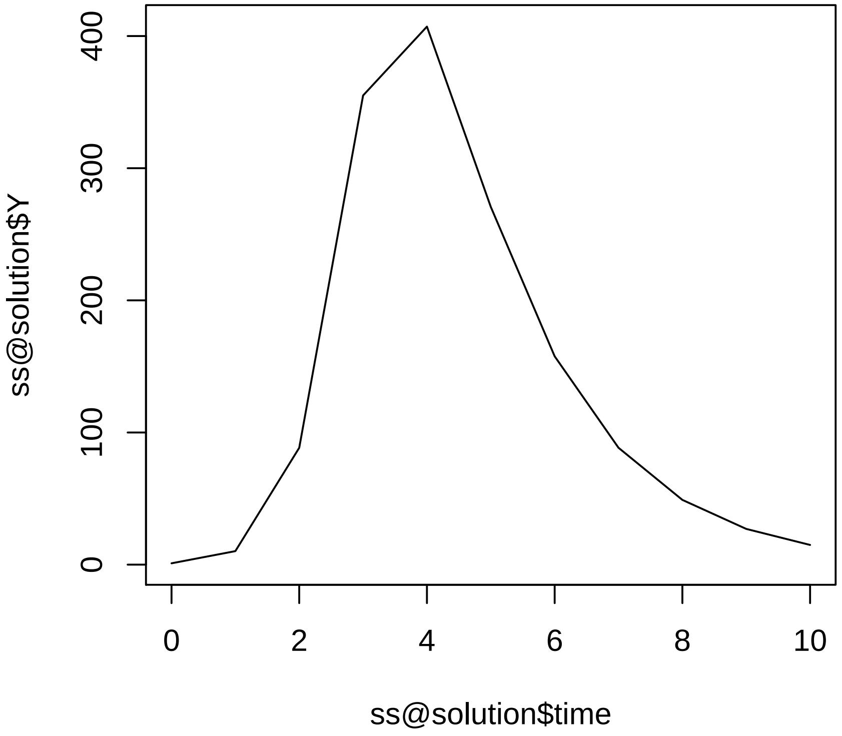
```
library(EpiDynamics)

SIRIM <- new("model.ode",
    name="SIR with induced mortality",
    model=SIRInducedMortality,
    initial=list(
        X~1-Y0,
        Y~Y0,
        Z~0
    ),
    par=c("rho", "mu", "nu", "beta", "gamma", "N", "Y0")
)
```

We have to define a custom solver because functions in `EpiDynamics` are ... ??

```
Epi.solver <- function(y, times, func, parms) {
    ss <- func(parms, y, times)$results
    ss[,-1] <- ss[,-1] * parms[["N"]]
    ss
}
```

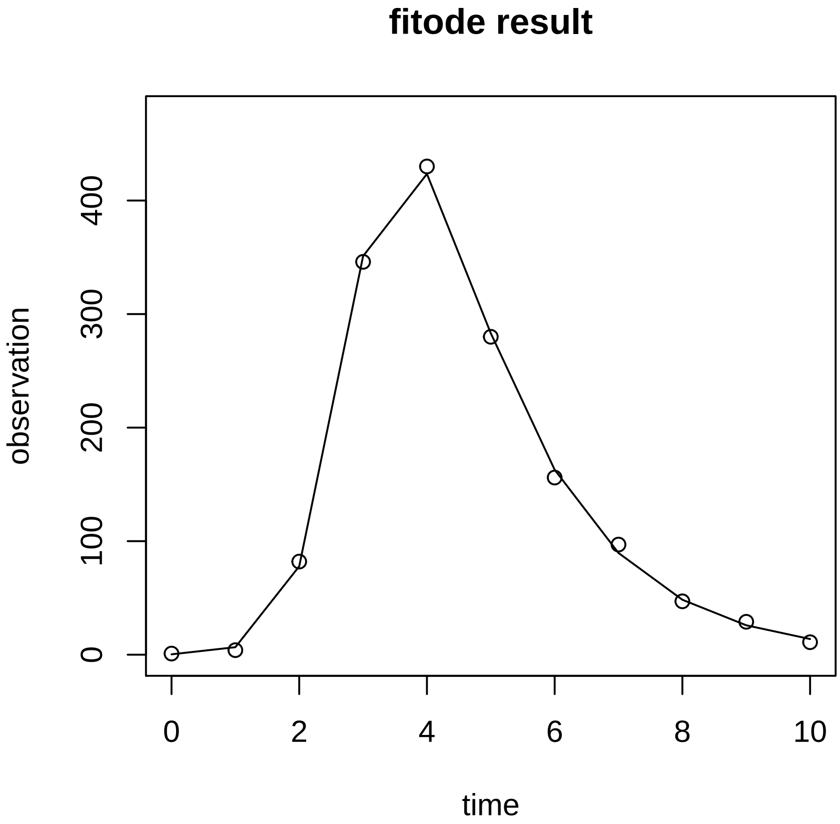
Trying out parameters:

```
start <- c(rho = 0.9, mu = 1/70, nu = 1/700,
                      beta = 3, gamma = 0.05, N=1000, Y0=1e-3)


ss <- ode.solve(SIRIM, 0:10, start, solver.opts=list(), solver=Epi.solver)
plot(ss@solution$time, ss@solution$Y, type="l")
```



We have to let `solver.opts=list()` because...

```
set.seed(101)
data <- data.frame(
    times=ss@times,
    prevalence=rpois(11, lambda=ss@solution$Y)
)

ff_IM <- fitode(prevalence|times~Y,
    start=start,
    model=SIRIM,
    loglik=select_model("poisson"),
    data=data,
    link=list(
        beta="log",
        gamma="log",
        N="log",
        mu="log",
        nu="log",
        rho="logit",
        Y0="logit"
    ),
    solver.opts=list(),
    solver=Epi.solver
)

## DLSODA-  At T (=R1) and step size H (=R2), the
##       corrector convergence failed repeatedly
##       or with ABS(H) = HMIN
## In above message, R1 = 0, R2 = 2.11863e-11
##

print(ff_IM)

## Model: SIR with induced mortality
## Formula: prevalence | times ~ Y
##
## Coefficients:
##          rho           mu           nu         beta        gamma
## 5.878386e-01 1.256057e-01 1.045955e-04 3.879401e+00 1.449617e-01
##            N           Y0
## 1.485091e+03 2.247902e-04
##
## Log-Likelihood:-33.71
##
## link: logit.rho log.mu log.nu log.beta log.gamma log.N logit.Y0

plot(ff_IM)
```



3.2 Nicholson experiment