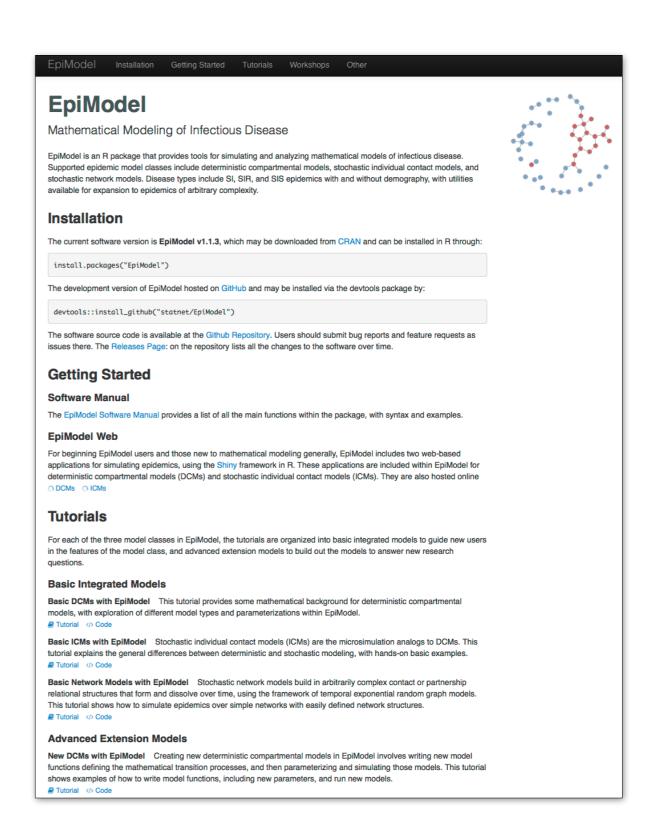


Introduction to EpiModel

Network Models for HIV/STI Transmission Dynamics with EpiModel June 27, 2017

EpiModel

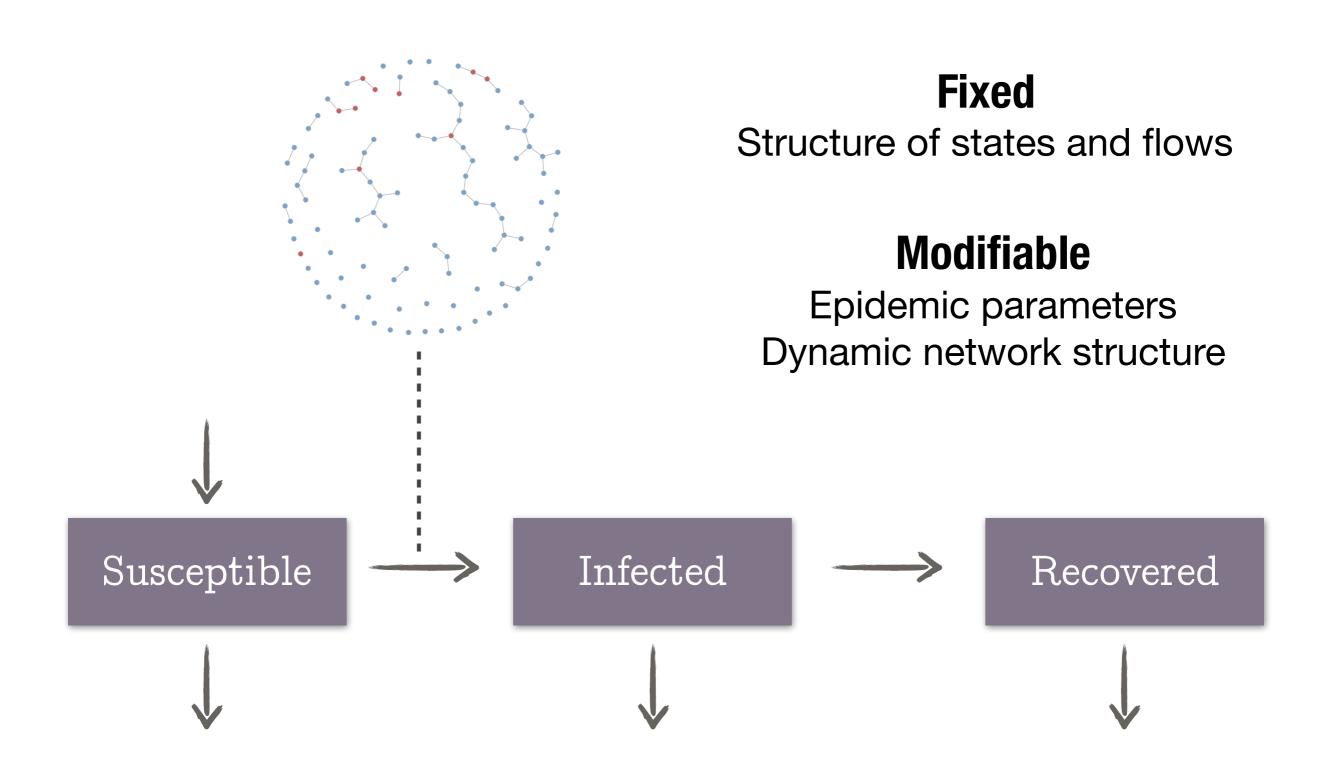
- Open-source software platform for epidemic modeling
- Tools for building, simulating, and analyzing models
- Three model classes
 - Deterministic compartmental models
 - Stochastic ABMs
 - Stochastic network models
- http://epimodel.org/



Learning vs Research Extensions

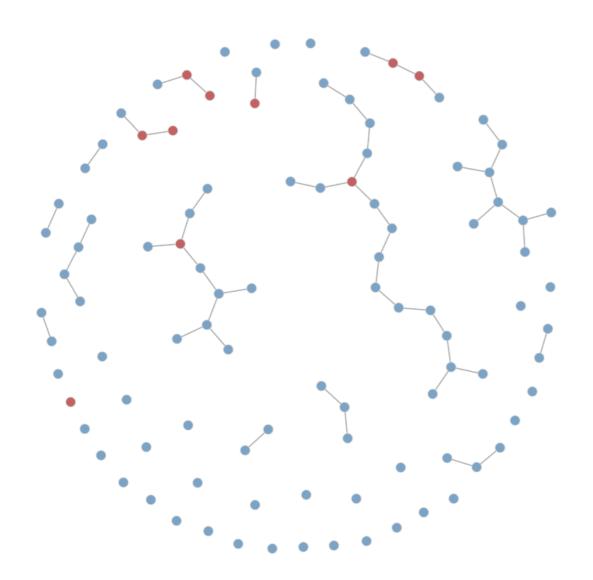
- EpiModel designed to be a tool for both teaching and research
- Software "integrated" epidemic models over basic disease types (SI, SIS, SIR) to learn the fundamentals of network modeling
- Not (necessarily) designed for novel research questions, but still allow for broad flexibility in parameterizing dynamic networks

Base Epidemic Models



```
sti recov <- function(dat, at) {</pre>
                                                                                 recovRGC <- c(recovRGC_asympt, recovRGC_tx, recovRGC_ntx)</pre>
                                                                                 recovUGC <- c(recovUGC asympt, recovUGC tx, recovUGC ntx)</pre>
# Parameters
rgc.dur.asympt <- dat$param$rgc.dur.asympt</pre>
                                                                                dat$attr$rGC[recovRGC] <- 0</pre>
ugc.dur.asympt <- dat$param$ugc.dur.asympt</pre>
                                                                                dat$attr$rGC.sympt[recovRGC] <- NA</pre>
gc.dur.tx <- dat$param$gc.dur.tx</pre>
                                                                                dat$attr$rGC.infTime[recovRGC] <- NA</pre>
gc.dur.ntx <- dat$param$gc.dur.ntx</pre>
                                                                                dat$attr$rGC.tx[recovRGC] <- NA</pre>
rct.dur.asympt <- dat$param$rct.dur.asympt</pre>
                                                                                dat$attr$uGC[recovUGC] <- 0</pre>
                                                                                dat$attr$uGC.sympt[recovUGC] <- NA</pre>
uct.dur.asympt <- dat$param$uct.dur.asympt</pre>
                                                                                dat$attr$uGC.infTime[recovUGC] <- NA</pre>
ct.dur.tx <- dat$param$ct.dur.tx</pre>
ct.dur.ntx <- dat$param$ct.dur.ntx</pre>
                                                                                dat$attr$uGC.tx[recovUGC] <- NA</pre>
                                                                                 dat$attr$GC.cease[c(recovRGC, recovUGC)] <- NA</pre>
# GC recovery
idsRGC asympt <- which(dat$attr$rGC == 1 & dat$attr$rGC.infTime < at &
                                                                                # CT recovery
                            dat$attr$rGC.sympt == 0)
                                                                                idsRCT asympt <- which(dat$attr$rCT == 1 & dat$attr$rCT.infTime < at &</pre>
idsUGC asympt <- which(dat$attr$uGC == 1 & dat$attr$uGC.infTime < at &</pre>
                                                                                                           dat$attr$rCT.sympt == 0)
                            dat$attr$uGC.sympt == 0)
                                                                                idsUCT asympt <- which(dat$attr$uCT == 1 & dat$attr$uCT.infTime < at &</pre>
idsRGC tx <- which(dat$attr$rGC == 1 & dat$attr$rGC.infTime < at &</pre>
                                                                                                           dat$attr$uCT.sympt == 0)
                       dat$attr$rGC.sympt == 1 & dat$attr$rGC.tx == 1)
                                                                                idsRCT tx <- which(dat$attr$rCT == 1 & dat$attr$rCT.infTime < at &</pre>
idsUGC tx <- which(dat$attr$uGC == 1 & dat$attr$uGC.infTime < at &</pre>
                                                                                                       dat$attr$rCT.sympt == 1 & dat$attr$rCT.tx == 1)
                       dat$attr$uGC.sympt == 1 & dat$attr$uGC.tx == 1)
                                                                                idsUCT tx <- which(dat$attr$uCT == 1 & dat$attr$uCT.infTime < at &</pre>
idsRGC ntx <- which(dat$attr$rGC == 1 & dat$attr$rGC.infTime < at &
                                                                                                       dat$attr$uCT.sympt == 1 & dat$attr$uCT.tx == 1)
                        dat$attr$rGC.sympt == 1 & dat$attr$rGC.tx == 0)
                                                                                idsRCT ntx <- which(dat$attr$rCT == 1 & dat$attr$rCT.infTime < at &</pre>
idsUGC ntx <- which(dat$attr$uGC == 1 & dat$attr$uGC.infTime < at &</pre>
                                                                                                        dat$attr$rCT.sympt == 1 & dat$attr$rCT.tx == 0)
                        dat$attr$uGC.sympt == 1 & dat$attr$uGC.tx == 0)
                                                                                 idsUCT ntx <- which(dat$attr$uCT == 1 & dat$attr$uCT.infTime < at &
                                                                                                        dat$attr$uCT.sympt == 1 & dat$attr$uCT.tx == 0)
recovRGC asympt <- idsRGC asympt[which(rbinom(length(idsRGC asympt), 1,</pre>
                                                  1/rgc.dur.asympt) == 1)]
                                                                                 recovRCT asympt <- idsRCT asympt[which(rbinom(length(idsRCT asympt),</pre>
                                                                                                                                  1, 1/rct.dur.asympt) == 1)]
recovUGC_asympt <- idsUGC_asympt[which(rbinom(length(idsUGC_asympt), 1,</pre>
                                                                                 recovUCT asympt <- idsUCT asympt[which(rbinom(length(idsUCT asympt),</pre>
                                                  1/ugc.dur.asympt) == 1)]
                                                                                                                                  1, 1/uct.dur.asympt) == 1)]
recovRGC tx <- idsRGC tx[which(rbinom(length(idsRGC tx), 1,</pre>
                                          1/gc.dur.tx) == 1)
                                                                                 recovRCT tx <- idsRCT tx[which(rbinom(length(idsRCT tx),</pre>
recovUGC tx <- idsUGC tx[which(rbinom(length(idsUGC tx), 1,</pre>
                                                                                                                          1, 1/ct.dur.tx) == 1)]
                                          1/gc.dur.tx) == 1)
                                                                                 recovUCT tx <- idsUCT tx[which(rbinom(length(idsUCT tx),</pre>
                                                                                                                          1, 1/ct.dur.tx) == 1)]
if (!is.null(gc.dur.ntx)) {
   recovRGC ntx <- idsRGC_ntx[which(rbinom(length(idsRGC_ntx), 1,</pre>
                                                                                if (!is.null(ct.dur.ntx)) {
                                              1/gc.dur.ntx) == 1)]
                                                                                   recovRCT ntx <- idsRCT ntx[which(rbinom(length(idsRCT ntx),</pre>
   recovUGC ntx <- idsUGC ntx[which(rbinom(length(idsUGC ntx), 1,</pre>
                                                                                                                              1, 1/ct.dur.ntx) == 1)
                                              1/gc.dur.ntx) == 1)
                                                                                  recovUCT ntx <- idsUCT ntx[which(rbinom(length(idsUCT ntx),</pre>
} else {
                                                                                                                              1, 1/ct.dur.ntx) == 1)]
   recovRGC ntx <- idsRGC ntx[which(rbinom(length(idsRGC ntx), 1,</pre>
                                                                                } else {
                                              1/rgc.dur.asympt) == 1)]
                                                                                   recovRCT ntx <- idsRCT ntx[which(rbinom(length(idsRCT ntx),</pre>
  recovUGC ntx <- idsUGC ntx[which(rbinom(length(idsUGC ntx), 1,</pre>
                                                                                                                              1, 1/rct.dur.asympt) == 1)]
                                              1/ugc.dur.asympt) == 1)
                                                                                   recovUCT ntx <- idsUCT ntx[which(rbinom(length(idsUCT ntx),</pre>
                                                                                                                              1, 1/uct.dur.asympt) == 1)]
```

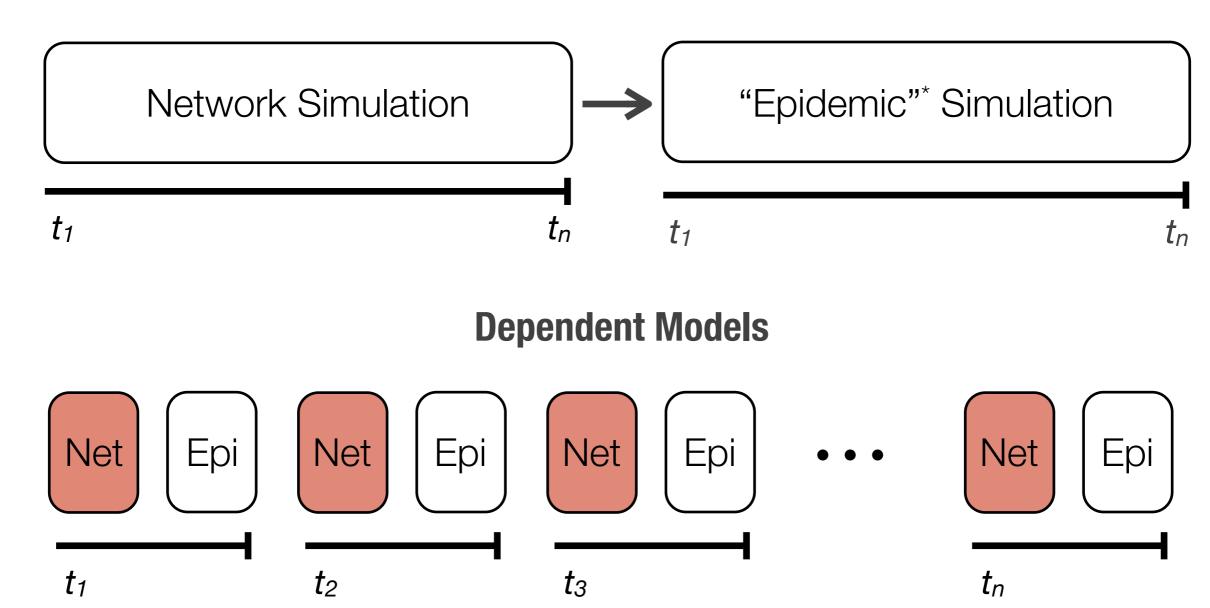
Closed Population



- Simplification conceptually and computationally
- Assumes no temporal feedback from exogenous dynamics on network structure
 - No demographic flows
 - No disease terms in the TERGM
 - No behavioral adaptation

Model Dependence

Independent Models



[&]quot;Epidemic" = biological, behavioral, demographic, etc., changes

EpiModel Workflow for Base Models

- 1. Construct the (empty) network data structure
- 2. Parameterize the TERGM (formation and dissolution formulas and target statistics)
- 3. Fit the TERGM, and diagnose the model fit
- 4. Parameterize the epidemic model
- 5. Simulate the epidemic
- 6. Analyze the simulation data

EpiModel Workflow for Base Models

- 1. Construct the (empty) network data structure: network.initialize, set.vertex.attribute
- 2. Parameterize the TERGM (formation and dissolution formulas and target statistics): dissolution_coefs
- 3. Fit the TERGM, and diagnose the model fit: netest, netdx
- 4. Parameterize the epidemic model: param.net, init.net, control.net
- 5. Simulate the epidemic: netsim
- 6. Analyze the model data: print, plot, summary, as.data.frame, ...