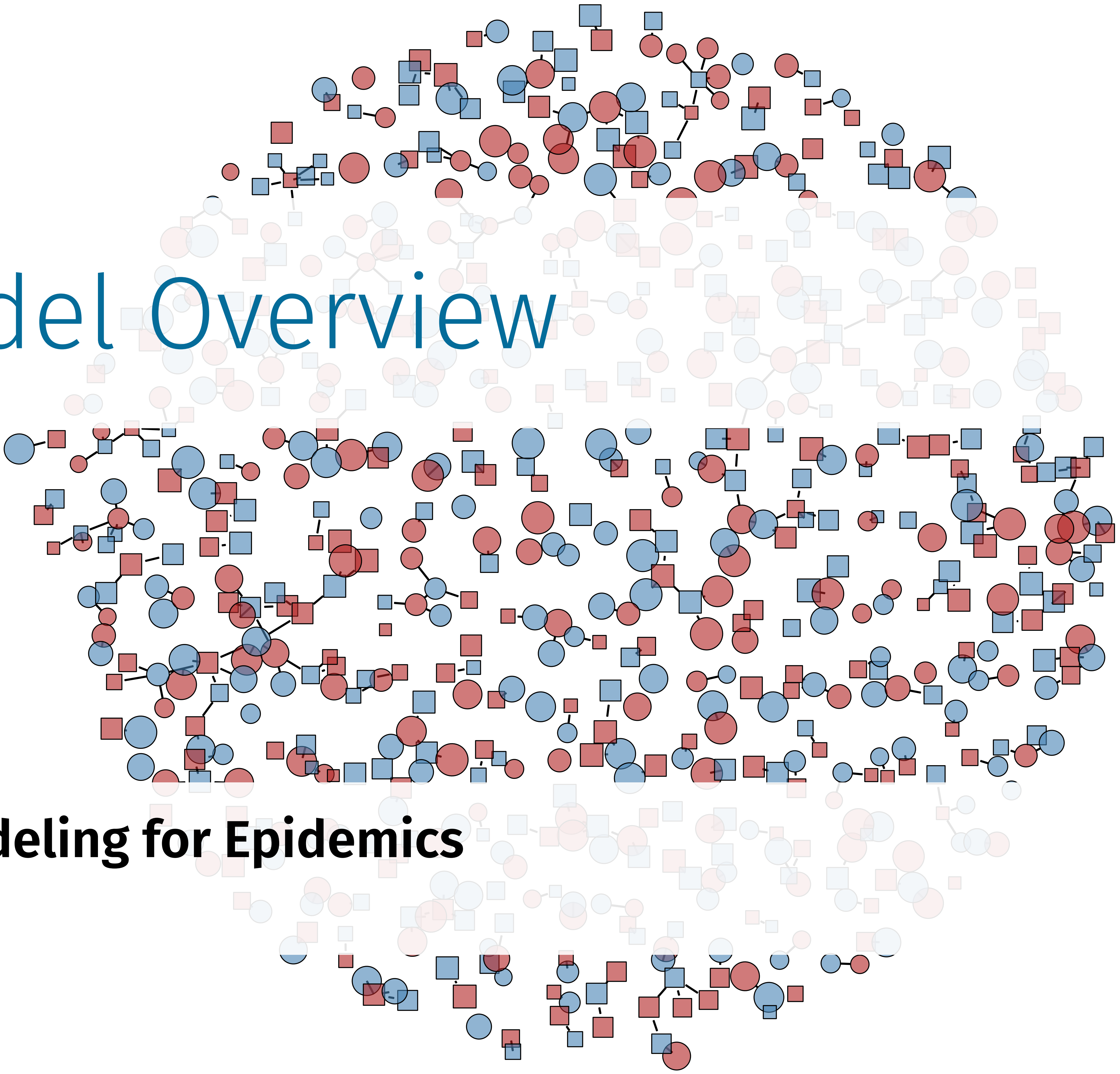


# EpiModel Overview

**Network Modeling for Epidemics**

Day 3

# Outline for Rest of Week

- Wednesday
  - Modeling epidemics + networks = modeling epidemics over networks
  - Core assumption: no feedback of epidemiology on networks
    - One important implication: closed populations
    - Still feedback: *network structure* $\Longrightarrow$ *epidemiology* and *incidence* $\Longrightarrow$ *prevalence*
  - Built-in **epidemiology** types (SI, SIR, SIS)
    - Working with nodal attributes, with heterogeneity in network structure and epidemiological parameters
- Thursday
  - Feedback: epidemiology $\Longrightarrow$ network structure
    - Vital dynamics, "sero-sorting" (edge formation based on changing nodal attributes)
  - Simple vaccine intervention
  - Built-in **epidemiology** types (SI, SIR, SIS), then getting started with extensions
- Friday
  - Getting comfortable with extensions
  - Building a network-based extension model for COVID, step-by-step…

# “Built-in Epidemiology”

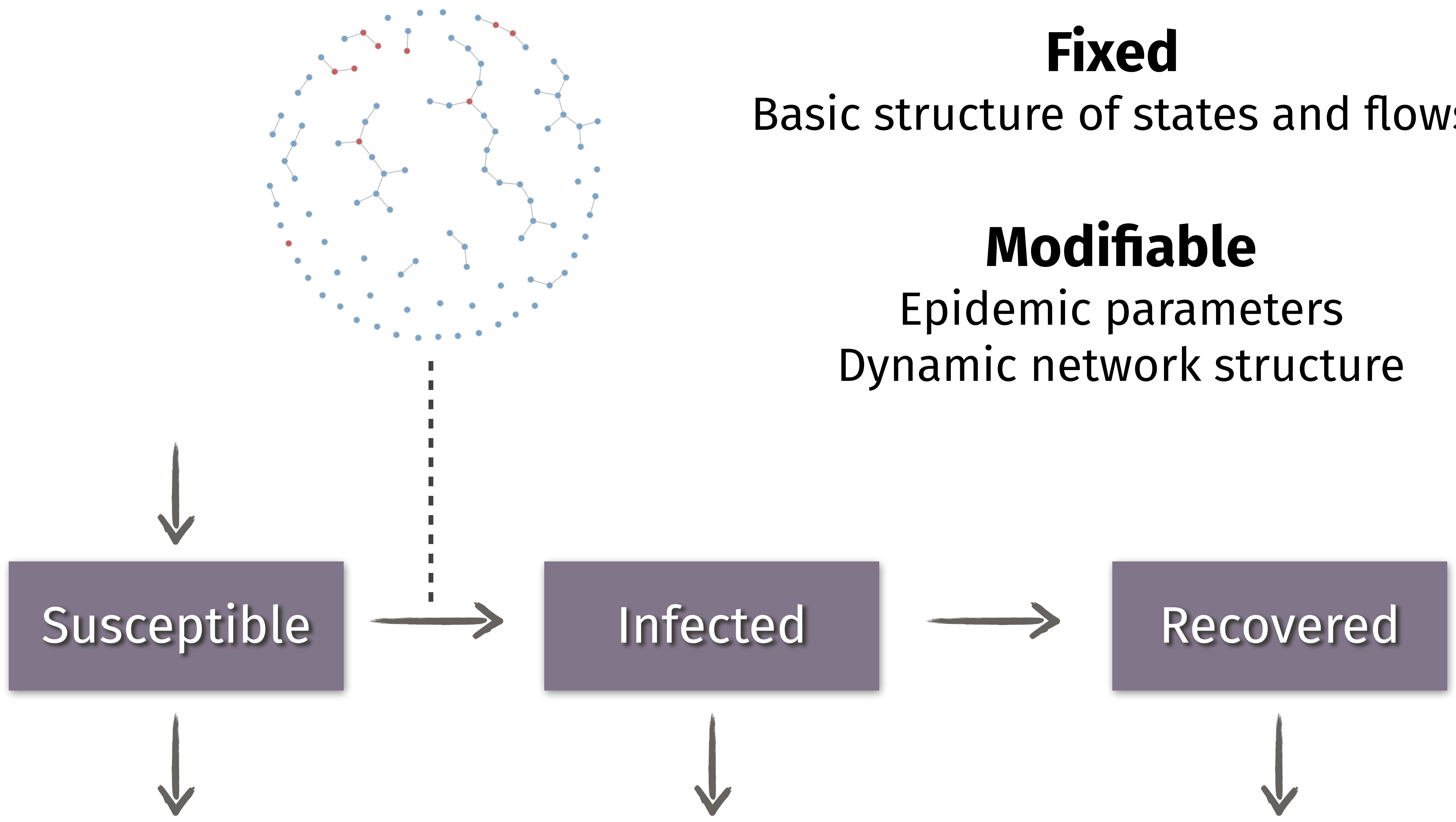
**Fixed**

Basic structure of states and flows

**Modifiable**

Epidemic parameters

Dynamic network structure

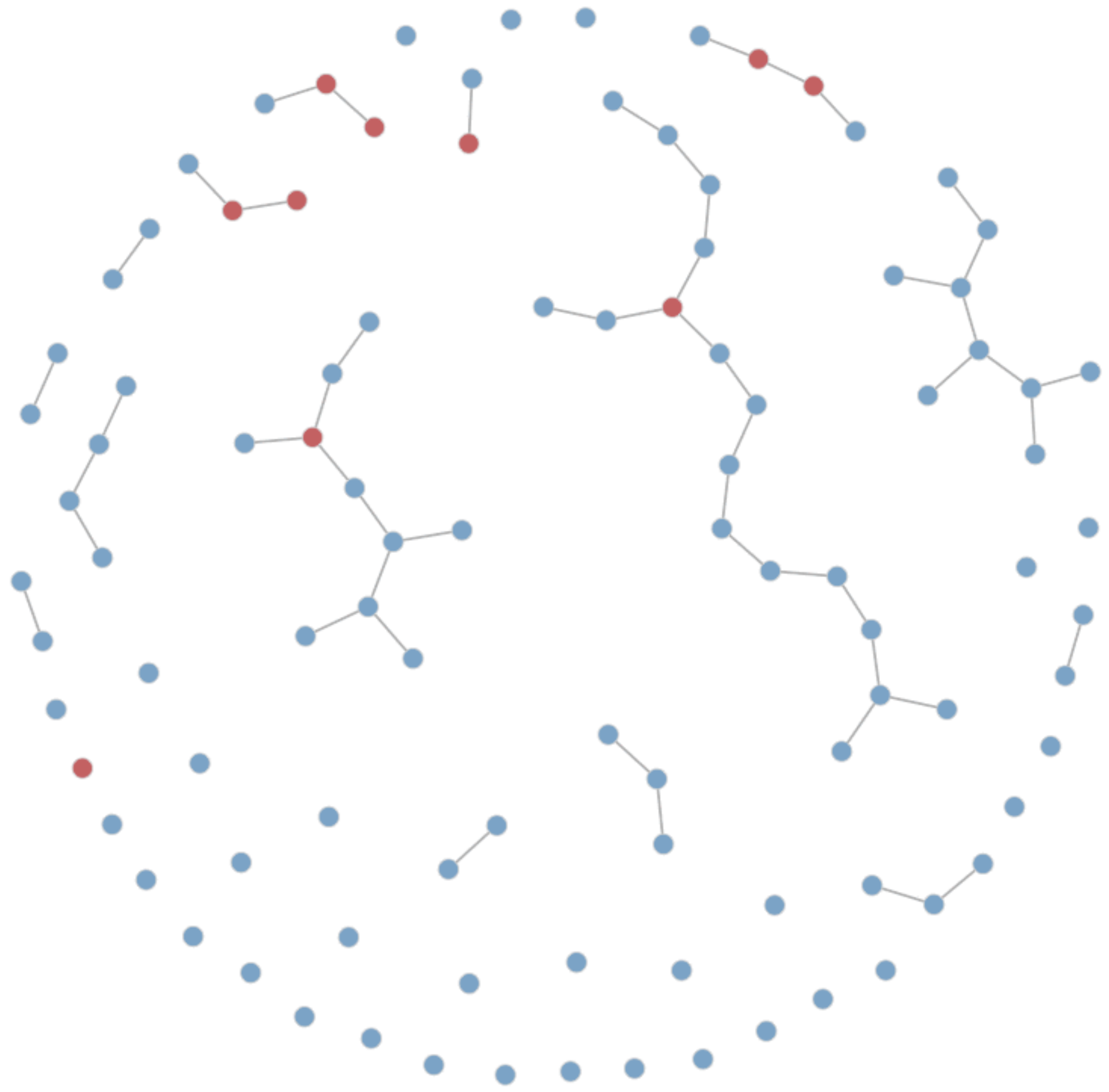


# Model Extensions Require Some More Advanced Coding

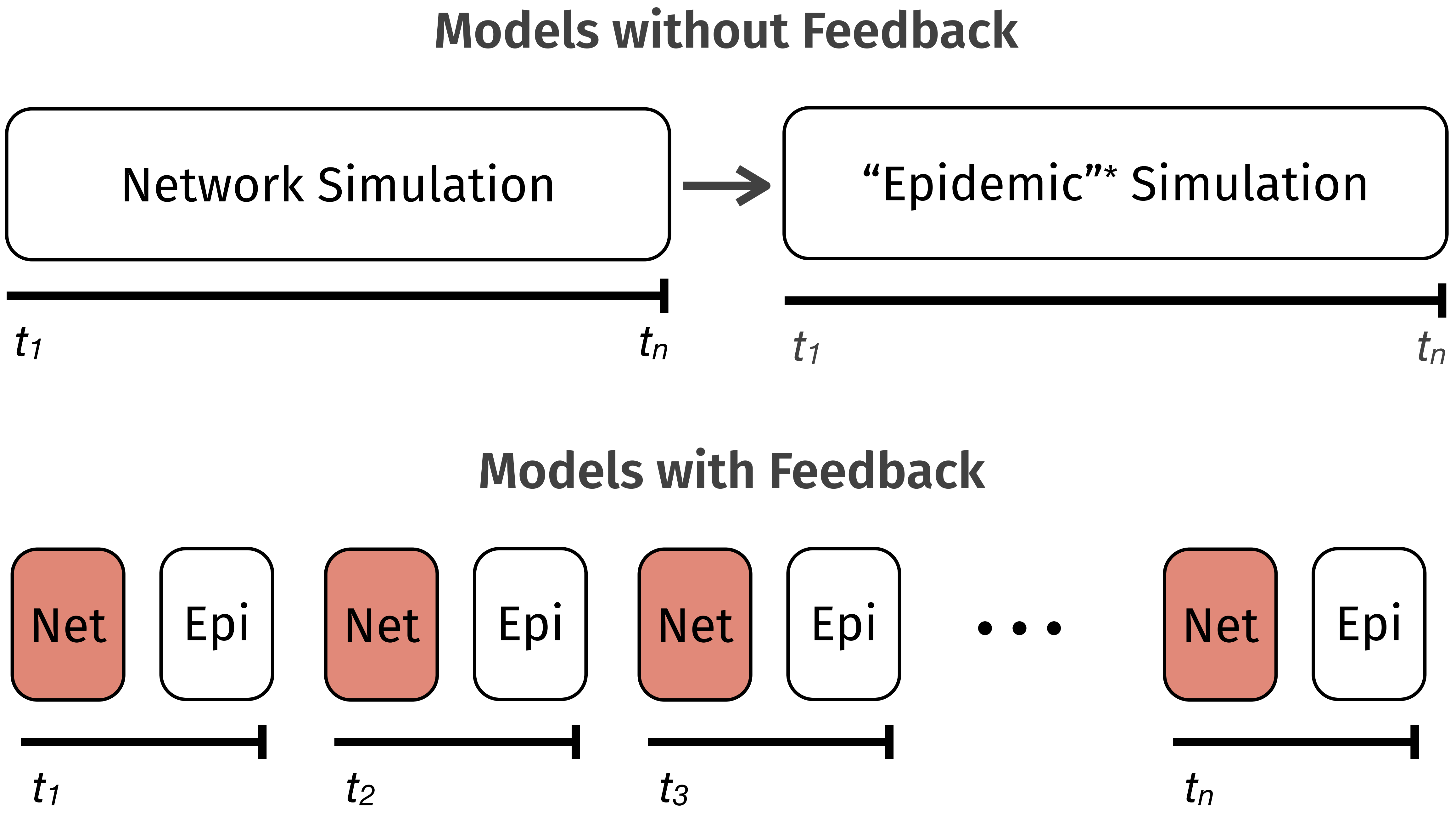
```r
sti_recov <- function(dat, at) {

  # Parameters
  rgc.dur.asympt <- dat$param$rgc.dur.asympt
  ugc.dur.asympt <- dat$param$ugc.dur.asympt
  gc.dur.tx <- dat$param$gc.dur.tx
  gc.dur.ntx <- dat$param$gc.dur.ntx

  rct.dur.asympt <- dat$param$rct.dur.asympt
  uct.dur.asympt <- dat$param$uct.dur.asympt
  ct.dur.tx <- dat$param$ct.dur.tx
  ct.dur.ntx <- dat$param$ct.dur.ntx


  # GC recovery
  idsRGC_asympt <- which(dat$attr$rGC == 1 & dat$attr$rGC.infTime < at &
                           dat$attr$rGC.sympt == 0)
  idsUGC_asympt <- which(dat$attr$uGC == 1 & dat$attr$uGC.infTime < at &
                           dat$attr$uGC.sympt == 0)
  idsRGC_tx <- which(dat$attr$rGC == 1 & dat$attr$rGC.infTime < at &
                       dat$attr$rGC.sympt == 1 & dat$attr$rGC.tx == 1)
  idsUGC_tx <- which(dat$attr$uGC == 1 & dat$attr$uGC.infTime < at &
                       dat$attr$uGC.sympt == 1 & dat$attr$uGC.tx == 1)
  idsRGC_ntx <- which(dat$attr$rGC == 1 & dat$attr$rGC.infTime < at &
                        dat$attr$rGC.sympt == 1 & dat$attr$rGC.tx == 0)
  idsUGC_ntx <- which(dat$attr$uGC == 1 & dat$attr$uGC.infTime < at &
                        dat$attr$uGC.sympt == 1 & dat$attr$uGC.tx == 0)

  recovRGC_asympt <- idsRGC_asympt[which(rbinom(length(idsRGC_asympt), 1,
                                                1/rgc.dur.asympt) == 1)]
  recovUGC_asympt <- idsUGC_asympt[which(rbinom(length(idsUGC_asympt), 1,
                                                1/ugc.dur.asympt) == 1)]

  recovRGC_tx <- idsRGC_tx[which(rbinom(length(idsRGC_tx), 1,
                                        1/gc.dur.tx) == 1)]
  recovUGC_tx <- idsUGC_tx[which(rbinom(length(idsUGC_tx), 1,
                                        1/gc.dur.tx) == 1)]

  if (!is.null(gc.dur.ntx)) {
    recovRGC_ntx <- idsRGC_ntx[which(rbinom(length(idsRGC_ntx), 1,
                                            1/gc.dur.ntx) == 1)]
    recovUGC_ntx <- idsUGC_ntx[which(rbinom(length(idsUGC_ntx), 1,
                                            1/gc.dur.ntx) == 1)]
  } else {
    recovRGC_ntx <- idsRGC_ntx[which(rbinom(length(idsRGC_ntx), 1,
                                            1/rgc.dur.asympt) == 1)]
    recovUGC_ntx <- idsUGC_ntx[which(rbinom(length(idsUGC_ntx), 1,
                                            1/ugc.dur.asympt) == 1)]
  }
```

```r
  recovRGC <- c(recovRGC_asympt, recovRGC_tx, recovRGC_ntx)
  recovUGC <- c(recovUGC_asympt, recovUGC_tx, recovUGC_ntx)

  dat$attr$rGC[recovRGC] <- 0
  dat$attr$rGC.sympt[recovRGC] <- NA
  dat$attr$rGC.infTime[recovRGC] <- NA
  dat$attr$rGC.tx[recovRGC] <- NA

  dat$attr$uGC[recovUGC] <- 0
  dat$attr$uGC.sympt[recovUGC] <- NA
  dat$attr$uGC.infTime[recovUGC] <- NA
  dat$attr$uGC.tx[recovUGC] <- NA

  dat$attr$GC.cease[c(recovRGC, recovUGC)] <- NA

  # CT recovery
  idsRCT_asympt <- which(dat$attr$rCT == 1 & dat$attr$rCT.infTime < at &
                           dat$attr$rCT.sympt == 0)
  idsUCT_asympt <- which(dat$attr$uCT == 1 & dat$attr$uCT.infTime < at &
                           dat$attr$uCT.sympt == 0)
  idsRCT_tx <- which(dat$attr$rCT == 1 & dat$attr$rCT.infTime < at &
                       dat$attr$rCT.sympt == 1 & dat$attr$rCT.tx == 1)
  idsUCT_tx <- which(dat$attr$uCT == 1 & dat$attr$uCT.infTime < at &
                       dat$attr$uCT.sympt == 1 & dat$attr$uCT.tx == 1)
  idsRCT_ntx <- which(dat$attr$rCT == 1 & dat$attr$rCT.infTime < at &
                        dat$attr$rCT.sympt == 1 & dat$attr$rCT.tx == 0)
  idsUCT_ntx <- which(dat$attr$uCT == 1 & dat$attr$uCT.infTime < at &
                        dat$attr$uCT.sympt == 1 & dat$attr$uCT.tx == 0)

  recovRCT_asympt <- idsRCT_asympt[which(rbinom(length(idsRCT_asympt),
                                                1, 1/rct.dur.asympt) == 1)]
  recovUCT_asympt <- idsUCT_asympt[which(rbinom(length(idsUCT_asympt),
                                                1, 1/uct.dur.asympt) == 1)]

  recovRCT_tx <- idsRCT_tx[which(rbinom(length(idsRCT_tx),
                                        1, 1/ct.dur.tx) == 1)]
  recovUCT_tx <- idsUCT_tx[which(rbinom(length(idsUCT_tx),
                                        1, 1/ct.dur.tx) == 1)]

  if (!is.null(ct.dur.ntx)) {
    recovRCT_ntx <- idsRCT_ntx[which(rbinom(length(idsRCT_ntx),
                                            1, 1/ct.dur.ntx) == 1)]
    recovUCT_ntx <- idsUCT_ntx[which(rbinom(length(idsUCT_ntx),
                                            1, 1/ct.dur.ntx) == 1)]
  } else {
    recovRCT_ntx <- idsRCT_ntx[which(rbinom(length(idsRCT_ntx),
                                            1, 1/rct.dur.asympt) == 1)]
    recovUCT_ntx <- idsUCT_ntx[which(rbinom(length(idsUCT_ntx),
                                            1, 1/uct.dur.asympt) == 1)]
  }
```

## Closed Population



# Model Feedback



*"Epidemic"* = biological, behavioral, demographic, etc., processes*

# EpiModel Workflow for Built-In 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 Built-In 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, …`

# Schedule for Today

(approximate)

| Session | Type | Title | Start (PST) | End (PST) |
|---|---|---|---|---|
| 1 | Lecture | EpiModel overview | 8:00 | 8:20 |
| 2 | Tutorial | SIS Dynamic Net | 8:20 | 9:00 |
| 3 | Lab | First Net Model | 9:00 | 9:50 |
|  | break |  | 9:50 | 10:00 |
| 4 | Lecture | Model Specification | 10:00 | 10:25 |
| 5 | Tutorial | Working with Attributes | 10:25 | 11:00 |
|  | Lunch |  | 11:00 | 12:00 |
| 6 | Lab | Pop Heterogeneity | 12:00 | 12:50 |
| 7 | Exercise | Ego Data Practice | 12:50 | 1:40 |
|  | Break |  | 1:40 | 1:50 |
| 8 | Tutorial/Lab | ndtv | 1:50 | 2:35 |
| 9 | Discussion | wrap-up | 2:35 | - |