Chris Lab Notebook

Chris Hoover

5/11/2020

5/11/2020

Goals:
* Familiarize with model
* Code model in STAN and try some initial fits to get idea of runtimes, outputs
* Start brainstorming potential improvements to model

Package/Model functionality

`R/InputsFromSpreadsheets.R`

Contains functions to read user inputs and translate them into model inputs, generate parameter sets to sample from. Also main wrapper function that users run to generate report: `CredibilityIntervalFromExcel`

`R/CredibilityInterval.R`

contains functions to run model, check which model runs "fit" input data (e.g. which parameter sets generate predictions that agree with input data)

`R/CombinedModels.R`

contains functions to run seir model

`R/CreateOutputs.R`

Takes model posteriors and generate plots of projections, posterior distributions and comparison to prior distributions

Overall process

1. User calls `CredibilityIntervalFromExcel` which processes data in excel spreadsheet and then calls `CredibilityInterval`

2. `CredibilityInterval` does some more preprocessing of model inputs and then calls `RunSim1`

3. `RunSim1` calls `RunSim` which uses `FitSEIR` to generate reasonable estimates of when the outbreak started based on comparison of model generated outputs to observed data done with `CalcError` function. Once FitSEIR has found a best start data for each parameter set, `Seir` is run.

4. The output of `RunSim1` comes from running `Seir` which produces model estiamtes from the input parameter estimate and best guess start date. `InBounds` is then called to determine if the output of `Seir` is within the user provided uncertainty bounds of the observed data.
   ** There seems to be some sort of iteration updating going on that I can't quite figure out

Test Run

```
file.copy(system.file("extdata", "SF-April13.xlsx", package = "LEMMA", mustWork = TRUE), "example.xlsx")

LEMMA::CredibilityIntervalFromExcel("example.xlsx")
```

Works for the most part except for a fairly obscure ggplot error

5/15/2020

Didn't make it to STAN on the 11th, so want to try and get an idea of some of the advantages and disadvantages STAN has compared to other approaches and maybe get around to coding up a version of the model in STAN

STAN Pros:

- Hamiltonian MCMC in STAN is quite fast and would generate more posterior samples
- Could incorporate priors to constrain some of the parameters like hospitalization rate
- Could place a prior on t_0 and fit epidemic start time more explicitly

STAN Cons:

- Not sure how to implement the same type of model fit in STAN that incorporates the min and max guess (i.e. confirmed hospitalizations and hospitalizations + 30% suspected cases). Could put a distribution on it, but could end up being a bit wonky

5/18/2020

Fixed some algebra slip-ups in the $\mathcal{R}_0$ calculation for the model in the `LEMMA_model` document

5/19/2020

Working on some initial network-based agent based models. read Maya and Mark's `Adaptive Surveillance` doc. From a coding perspective, making the model run quickly and having fairly flexible implementation of different interventions (e.g. testing, isolating) will be the most challenging. From a practical standpoint, having a realistic contact/network structure may be the most challenging. The `EpiModel` package seems like a good place to start, gonna toy around with that a bit now. Specifically, going to try constructing a stochastic network model with the package which uses the `tergm` and `networkDynamic` packages to implement separable-temporal exponential-family random graph models (STERGMs; whatever that means...) Basically following the tutorial on the EpiModel website

Working with `EpiModel` package

- Good background on networks and terminology here
  - Network characteristics to consider:
    * **degree distribution**: distribution of number of connections per node
    * **geodesic distribution**: distribution of the shortest path between nodes
    * **component size distribution**: distribution of the size of different components if multiple unconnected components in the network

Step 1) Develop network characteristics

Set network size and characteristics

```
require(EpiModel)

# Initialize network with n=1000
N <- 1000

nw <- network.initialize(n = N, directed = FALSE)

# Add characteristics to the vertexes of the network
  # add race characteristic; here 500 A and 500 W
  nw <- set.vertex.attribute(nw, "race", c(rep("A", 500), rep("W", 500)))
  # Add household characteristic (assumes mean HH size of 5, this could be more data-informed)
  nw <- set.vertex.attribute(nw, "house", sample(c(1:200), 1000, replace = T))
```

NOTE: could add these characteristics from a separate data frame in which household, race, occupation, dormitory, major, etc. could be correlated and drawn from an empirical distribution of some sort

Set network partnership formation formula

Here we declare characteristics of the network in order to generate edges. This consists of a `formation` formula that declares relationships between nodes based on their characteristics and a `target.stats` vector which gives the expected number of edges given the number of nodes, the network characteristics between nodes (e.g. their mean degree) and their relationships as declared in the `formation` formula.

```
formation <- ~edges + nodefactor("race") + nodematch("race") + concurrent

tot.mean.degree <- 2
raceA.mean.degree <- 1
prob.same.race.edge <- 0.75
prob.concurrent <- 0.7

target.stats <- c(N/2*tot.mean.degree, 500*raceA.mean.degree, (N/2*tot.mean.degree)*prob.same.race.edge
```

Get network dissolution formula

Rate at which network dissolves

```
coef.diss <- dissolution_coefs(dissolution = ~offset(edges), duration = 10)
coef.diss
```

```
## Dissolution Coefficients
## ======================
## Dissolution Model: ~offset(edges)
## Target Statistics: 10
## Crude Coefficient: 2.197225
## Mortality/Exit Rate: 0
## Adjusted Coefficient: 2.197225
```

Fit network

```
netfit1 <- netest(nw,
                  formation = formation,
                  target.stats = target.stats,
                  coef.diss = coef.diss,
                  edapprox = FALSE)
```

The above took forever to run, and I'm also beginning to question whether all this fuss is really worth it when the contact networks we want to simulate aren't necessarily all that complicated. `EpiModel` seems to be more suited for e.g. sexually transmitted diseases where contact networks relevant to transmission are clearly defined and are fairly dynamic and structured through time. For SARS-CoV2, relationships that lead to contacts that lead to transmission probably fall into one of two categories: highly persistent (e.g. households) or highly random (e.g. the person that I passed on the street that wasn't wearing a mask)

5/20/20

Bummed that I spent the entire day trying to get the `EpiModel` package to work only to give up on it, but I think it's best to just start with an ABM from scratch at this point. There are a couple good places to start, including in Maya and Mark's document, but also this code from epirecipe and this covid abm from oxford

Started coding the ABM (`COVID_ABM_V1.R`). Have code to generate a network as a matrix, but would be good to inform network characteristics with more data. Would be even better to code so that users could input network characteristics and resulting transmission probabilities and then simulate from there. Could be implemented as a spreadsheet with three columns: number of edges in this part of the network, transmission probability along these edges, and now I've forgotten the third one... Maybe specific transmission characteristics associated with each part of the network? Anyways, basically finished code to generate network structure, now working on simulating transmission on top of the network and then will incorporate interventions.

5/21/20

Added transmission parameters, created to do list for ABM below as well as some initial description of the ABM (see below). Also started working on implementing transmission across the network. Next step is to implement dynamic updating of the network based on interventions, testing, and infection status

5/22/20

Working on updating network based on interventions. Then will work on implementing testing and its effects. After that, can begin adding nuance like some workplaces remaining open, individual heterogeneities, etc. Basic model runs with transmission occurring across network, interventions affecting network. Now need to add effects of infection on network, e.g. severely infectious people stop interacting with most everyone and then add testing and contact tracing effects on network.

5/23-24/20

Forked (more like copy-pasted) UCSC SEIR stan model to start toying with it. Went through the actual code in more detail, but still not at a point where I understand it to the point where I feel like I could rebuild it from scratch. Which is probably a bit of a lofty goal anyways. First step is to be able to get it to run which require getting input data and putting it in the right format for the stan model. Got data for SF from their API and cleaned it. Code pulls directly from their website so should be nice and reproducible. Also got the stan model to run after some toying with it and then do some post-processing and plotting.

5/25/20

Edited the stan model to incorporate PUIs into the model. New stan model is `SEIR_PUI.stan`. Also evaluated model fits with this model compared to the original. Summary is in `stan_with_pui` Rmd/pdf. It doesn't make much of a difference, but it does bring down future projections a little bit. Not entirely sure how to make sense of this actually.

5/26/20

Working on getting to the bottom of some warning messages thrown by STAN when running the model:
* **Maximum treedepth exceeded** - seems to be a harmless warning more to do with efficiency than any model error; see here
* **Largest R-hat is NA**
* **Bulk and tail effective sampling size (ESS)** too low
* Occasionally getting a **divergent transitions** error which seems to be the only one that's truly problematic

Many of the suggestions on how to fix these involved running more chains/running more iterations. Tried a model run with 500 burnin/warmup, with 4500 subsequent samples with thinning every two samples (so effective sample size ~2250 per chain) and still got max treedepth, R-hat, Bulk ESS, and Tail ESS warnings as well as divergent transitions warning.

Tried increasing the `adapt_delta` parameter up to 0.9 (from default of 0.8) to fix the divergent transitions. Also increased the distance in time between spline knots as something tells me that allowing beta to be too flexible along with all the other parameters may cause identifiability issues. Running with above settings takes far too long, so just running with 1000 iterations, 100 burn in, no thinning.

Ok, so the R hat and sample size warnings are fine because some of the model outputs are included as "posteriors" and since e.g. initial hospitalizations are always 0, most of the posterior metrics can't be estimated or end up as NA.

The divergent transitions are probably still a major issue that needs to be resolved, though, and increasing `adapt_delta` to 0.9 didn't seem to fix the problem. Increased to 0.95 and that eliminated the divergent transitions and left me with only the R-hat, treedepth, and ESS errors. Tried increasing the max_treedepth control parameter as well, but it slows things down a lot.

5/27/20

Spent all day revisiting/revising my continuous time model, both stochastic and deterministic versions. Main goal was to try and fit a model forced with movement data of some sort and use to produce some projections based on movement data. Fitting is proving challenging. Tried just using optim with model fit based on least squared error of hospitalizations and deaths but that's pretty meh and not working for some reason. Also tried some old metropolis hastings algorithm code, but still working on that. Other end of the spectrum is to code it in STAN, but incorporating forcing functions in STAN appears to be less than simple. Long term goal would also be to divide the population in two and have the movement data allocated almost entirely to the second population following shelter in place

5/28/20

Kept working with continuous time model(s), but used a version closer to LEMMA to make things simpler. Still working on getting fitting to work to the point where I'm confident moving the model and fitting procedure to a two-population model. Will go back to working on the ABM tomorrow.

5/29/20

Back to work on the ABM, some main things to accomplish today are to:

- Introduce heterogeneity in network generation based on individual heterogeneity and also time of day dynamics (e.g. time when folks are at work/school, in the home, asleep, etc.)
- Introduce influence of illness on network
- Introduce testing, contact tracing

6/1/20

ABM

Introduced individual traits `q.prob` and `r.net.rate` corresponding to individual probabilities of quarantining given infection and being involved in random edges in the network, respectively. Right now these are generated randomly, but should be informed by other characteristics as well as testing/contact tracing in the future.

STAN

Did some debugging of the STAN model with PUI added by restricting the hospitalization/icu length of stay. Fit the STAN model to data from TN and StL for ultimate comparison to LEMMA fits with the spline on β that Josh is working on

6/2/2020

Working on editing stan models to fit to different data types since different areas have different combinations of testing, cases, hospital admissions, hospital census, icu admissions/census, PUI, and deaths. So have several versions of the STAN model now including:

- `UCSC_SEIR` - the original STAN model from UCSC
- `SEIR_PUI` - the original STAN model with PUI in hospitals added in the observed data and parameter `frac_puipos`
- `SEIR_Hosp_Only` - Fitting only to hospital census data, all other data and fitting removed
- `SEIR_Hosp_Census_admin` - Fitting to hospital census and admissions data (`obs_Hadm` and `hosp_adm` added, other observations removed)
- `SEIR_Hosp_Admissions` - Fitting to hospital admissions data only; *This one was causing some issues and after talking with Phil, seemed like the data wasn't strong enough to influence the fit given the data. Added some leading 0s in the hospitalizations data, reduced the initial beta estimate, reduced the variance parameter for the model fit*

6/4/2020

Investigating triggers based on hospitalizations/testing for returning to SiP/some other method to reduce $\mathcal{R}_e$. Using the continuous time model with a forcing function that causes relative changes in β at a few key time points (school closures, SiP start, mid April, mid May, and last time point). Fitting involves estmiating the baseline β, α, and μ parameters as well as the relative changes in β at each time point.

For ultimate goal of deriving triggers from testing data, think it helps to think about what exactly the question is here, because there's more to it than what it seems. First question is what kinds of people are being tested. SF is doing about 1600 tests per day with a test+ rate of around 2.5%, but we don't know the extent to which this sample is biased by 1) time, 2) infection status, and 3) risk. If positive cases are mostly those who are already in the hospital, that doesn't really tell us anything more than hopsitalizations tells us. If people are getting tested as soon as they have symptoms, but then aren't confirmed positive for another 3 days, that also introduces a lag that has big implications for our trigger.

Another set of questions revolves around what exactly we want the trigger to represent. E.g. should the trigger be tripped when we reach a point that threatens to overrun hospitals in the next week? In the next two weeks?

Meeting with Josh:

- Start with assumption that 1600 people per day are sampled randomly
- Don't worry about reporting time delay for now, or maybe next day?
- Answer question as to whether it IS possible to use as an early warning
- What are we trying to avoid? Total number of cases getting above X, or hospitalizations going above Y
- For a given trigger, what are the key Epi metrics at some date in the future?
- Two different levels of Re, 1.1 or 1.4 starting June 15th, different testing metrics for implementing trigger

ABM TODOs

Network TODOs

- Inform random network generation with individual characteristics, e.g. some individuals more "social": higher probability of generating random edges. Currently implemented with `r.net.prob`, but should introduce heterogeneity based on other characteristics rather than have it generated randomly
- Introduce heterogeneity in workplace network: e.g. some work networks dissolved under shelter in place, others persist (essential workers). Maybe have one type of workplace be healthcare workers whcih specifically interact with hospitalized individuals?
- Better inform `q.prob` vector in `quarantine_symptomatics` function such that worker status, past experience of infection (e.g. family members were infected/tested), etc. influence probability of staying home given symptoms
- Look at network characteristics like mean degree, degree distribution
- Implement testing and isolation on network

Transmission TODOs

- Transmission to households influenced by size/income of household (e.g. ability to isolate within home)
- Check $\mathcal{R}_0$ in relation to network and transmission probabilities. Pretty sure largest eigen value of the network matrix with entries equal to transmission probabilities can be interpreted as $\mathcal{R}_0$ in which case we have $\mathcal{R}_0 \approx 5.4$, which is maybe a bit high but in the right ballpark.

Other TODOs

- Introduce nighttime/daytime connections depending on time step, dt. E.g. if dt = 1/2 would have 1 active period, one inactive period per day, if dt = 1/3 two active periods, one inactive per day, and so on
- Ways to speed up? Transfer some/all functions to `C++`, do some profiling, etc.
- Move parameters into function argument and setup into function

ABM Description

Network structure

Individuals can be members of different networks such as work, family/household, school that are constant through time in the absence of interventions. They also participate in random networks that are generated at every time step based on parameter `r.net.prob`. School closures eliminate the school-based network (e.g. all transmission probabilities in school network to 0), increase familial contacts (e.g. within household), and reduce random edge generation from `r.net.prob` to `r.net.prob.sc`. Shelter in place reduces the generation of random edges to `r.net.prob.sip` and also increases household contacts **between household members who are not in essential workforce and therefore either work from home or are laid off (NOT IMPLEMENTED YET)**. The contact matrix is an $N \times N \times T$ array in which each t slice contains a symmetric matrix with entries corresponding to the transmission probability between i and j were one of them to be infectious and the other susceptible.

Transmission

The transmission model across the network is a modified SEIR that allows for variability in symptom severity by including presymptomatic (I_P), asymptomatic (I_A), mildly symptomatic (I_M), and severely symptomatic (I_{mh}, I_h) states. This allows infection status to influence behaviors such as test-seeking and self-isolation and variability in transmissibility. Parameters fall under two categories, `t.___` parameters draw from a distribution to generate a time until the next event occurs and `p.___` parameters return a probability of transitioning to one state or another at each transition (e.g. asymptomatic or not). Probability of being asymptomatic, of being severely symptomatic (eventually hospitalized), and of dying are age-dependent. As an example: `p.asymp` gets a probability that the infected individual will be asymptomatic, this probability is then used in a bernouli trial where success is defined as the individual transitioning to I_A where they then draw from `t.asymp` to determine how long they will remain asymptomatic before recovering.

Transition times are stored in the `t.til.nxt` matrix and infection status is stored in the `inf.mat` matrix, each with dimensions $N \times T$. Each matrix is updated at the beginning of each model iteration. If an entry in the `t.til.nxt` matrix becomes negative, the function `next.state` is implemented to determine the next state and the time that will be spent in that state. Following these updates to infection status, the network is updated. School closures and shelter in place interventions alter the network as described above. Infection status influences the network by quarantining individuals with symptomatic infection (e.g. I_m or I_{mh} states). These individuals' contacts outside of the household are dissolved, and their household contacts are changed from "H" to "Q" to indicate they are quarantining within the household. **This update facilitates changes in household transmission based on the individual's ability to self-isolate within their home, which is influenced by household size/density, income, and work/school status of other individuals in the home (NOT IMPLEMENTED YET)** Hospitalized (I_h) infections do not contribute to transmission (**But could if we make a healthcare worforce type that interacts with hospitalized patients (NOT IMPLEMENTED YET)**) then random edges are built with function `add.r.edges` with probabilities influenced by interventions **and individual heterogeneities (NOT IMPLEMENTED YET)**. Finally, transmission is simulated across the network using function `new.infection` which reduces the contact matrix to only those columns corresonding to infectious individuals, then performs a Bernoulli trial row by row to simulate the probability a contact results in transmission. New infections are then added to `inf.mat` if the Bernoulli trial is successful and the corresponding row is susceptible. For these new infections, `t.latent` is also sampled and added to the `t.til.nxt` matrix.

Interventions

Interventions such as school closures and shelter in place are currently implemented based on user input times (e.g. `t.sip` & `t.sip.end`) and affect the network as described above. Test-trace-isolate intervention occurs as follows: **NOT YET IMPLEMENTED, TO DISCUSS... user can input a matrix with columns corresponidng to time of testing, number of tests. How are tests allocated towards individuals at each time point?? Options include: random, based on their contact history, systematic (e.g. given N tests per month, N/4 individuals receive a test every fourth week)**

Testing

Potential improvements/ideas (running list)

Network models to examine sub-epidemics (like Mission study), implications for effectiveness of SiP, opening back up

Intro paper on network models here with corresponding R package and website with tutorials

Questions

- What are the characteristics of a network that gives rise to the findings of the Mission study?
- What are the implications of such a network for efficacy/limitations of SiP intervention? What are the implications for lifting SiP? i.e. how does the network change once SiP is lifted and how does transmission then propogate through this post-SiP network?
- How can test-trace-isolate be used to limit transmission through such a network?

Concepts

- **Degree** or **connectivity** (k) - number of neighbors an individual has; **degree distribution** (p_k) population level characteristic of connectivity
- **Size** of the network described by **distance** between two nodes - the length of the shortest path between them in the network; **diameter** of the network is the largest **distance** in the network
- **Mixing matrix** describes how individuals of type i within the network are connected to individuals of type j where "type" can refer to any characteristic that differs between individuals. **Assortative mixing** describes a network pattern in which individuals are more likely to interact with like individuals
- **local clustering** (ϕ) - how many pairs are shared between pairs of individuals in the network, e.g. given two pairs, how many of the networks between the two pairs will look like triangles in which A relates to B, B relates to C, AND A also relates to C.
- **Betweenness/Centrality** - Importance of particular individuals within the network; number of paths between i and j that pass through a particular node

Continuous time, stochastic model versions

I've got a continuous time stochastic model built with the `adaptivetau` package that's got a slightly different structure that I could work on building out a bit more/make more user friendly with similar input/output functionality. Would having three models all running projections help or make things more confusing? Maybe have this going on behind the scenes and convey where there's more or less agreement between the models to DPHs?

Quantifying impact: how many cases/hospitalizations/deaths avoided due to SiP?

Erlang distributed latent period, hospitalization length

Maybe a bit more of an academic exercise, but compartmental models make an implicit assumption that wait times between model states are exponentially distributed, which is rarely the case. Implementing "box cars", we can relax this assumption to make wait times Erlang (special case of a gamma distribution in which the shape parameter is a positive integer) distributed. Would be good for both E and H states given we have data on the distributions of latent period and hospitalization times we could fit to

Make some priors endogenous (e.g. non-adjustable), focus users on things most likely to vary by region/area/population:

- Population Size
- Start date of projection
- End date of projection
- Percent infected who are hospitalized
- Average hospital length of stay
- Timing and effectiveness ($\mathcal{R}_e$) of interventions