Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

Time invariant analysis of epidemics with EpiCompare

Shannon K. Gallagher
Biostatistics Research Branch
National Institute of Allergy
and Infectious Diseases

Benjamin LeRoy
Dept. of Statistics & Data Science
Carnegie Mellon University

Abstract

We present **EpiCompare**, an R package that suppliments and enhances current infectious disease analysis pipelines and encourages comparisons across models and epidemics. A major contribution of this work is the set of novel *time-invariate* tools for model and epidemic comparisons - including time-invariate prediction bands. **EpiCompare** embraces R's *tidy* coding style to make adoption of the package easier and analysis faster. This paper provides an overview of both the tools in and intuition behind **EpiCompare** and a thorough demonstrating of the tools through a detailed example of a full data analysis pipeline.

Keywords: keywords, not capitalized, Java.

1. Introduction

I something wory about including a full paragraph of introduction before we emphasis where and why we think this tool is useful. Especially the first "where" it could be useful, as I'm not sure how it connets to the more "global" view of the epidemic research area.

The recent (and on-going) COVID-19 global pandemic has galvanized public interest in understanding more about infectious disease modeling and has highlighted the usefulness of research in the area of infectious disease epidemiology. Infectious diseases inflict enormous burdens on the world: millions of lives lost and trillions of dollars spent yearly. Infectious disease models typically attempt to do one or more of the following: 1) predict the spread of current and future epidemics (e.g. flu prediction Biggerstaff *et al.* 2016), 2) analyze past and current epidemics to increase scientific knowledge (e.g. historical measle outbreaks Neal and Roberts 2004), and 3) forecast or project epidemic scenarios under pre-specified parameters (e.g. Ferguson *et al.* 2020). At the same time, descriptive statistics and visualizations

from universities, many branches and levels of government, and news organizations are an important first step of the process (Dong *et al.* 2020; CDC 2021; The Washington Post 2021) .

With the many visualization and exploratory tools, models and modeling paradigms, and reviews and comparisons in the literature and through the MIDAS (Models of Infectious Disease Agent Study) network (MIDAS Network 2021), this field has a lot of devices to aid an individual practitioner decide the correct approach. For example,R packages such as **surveillance**, **EpiModel**, and **pomp** have all made significant steps in standardizing the flow of the data analysis pipeline for epidemic modeling through digitizing data sets, making accessible statistical models, and providing a plethora of educational material for both coding novices and experts alike (Meyer *et al.* 2017; Jenness *et al.* 2018; King *et al.* 2016).

At the same time, analysis packages often only address a specific portion of the analysis pipeline, for instance focusing on certain types of models. Modeling tools, which usually require learning package-specific syntax, often don't provide easy ways to compare and assess their models on new data. Moreover, exploring and modeling epidemics require transforming and *tidying* data in different ways. To fill these gaps, we present our R package **EpiCompare**. Our package's primary focus is to aid and advance research in the area of comparison and assessment of epidemic and epidemiological models. In Figure 1, we illustrate the data analysis pipeline of infectious diseases as 1) data pre-processing, 2) exploratory data analysis (EDA), 3) modeling and simulating, 4) post-processing, and 5) comparison and assessment; where each previous part of the pipeline influences the next. **EpiCompare** provides tools to aids practitioners in all areas of this pipeline.

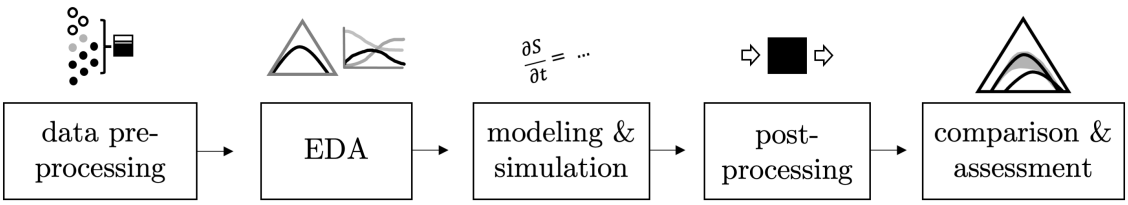

Figure 1: An idealized epidemiological data analysis pipeline.

EpiCompare also emphasizes the value of analyzing epidemics in a *time-invariant* way. Epidemics, despite by definition being a process that evolves over time, often need to be compared in a way not constrained to initial times or time scales to understand the processes at play. Moreover, many current comparison tools for state-space models (e.g. SIR models) highlight the proportion of individuals in each state (at a given time) in a piece-wise / marginal fashion. This approach may reduce the amount of connections that can be seen, similar to projections of a multidimensional distribution onto a single axis at a time. Tools in **EpiCompare** give the user the ability to extend their toolkit to evaluate epidemics within a time-invariant lens. The goal of **EpiCompare** is not to supplant existing infectious disease modeling tools and software but, rather, is a concerted effort to create standard and fair comparisons among models developed for disease outbreaks and outbreak data.

This paper is broken up into the following sections; section 2 motivates and showcases tools of time-invariant analysis, section 3 presents an outline of how **EpiCompare** aids a practitioner in every step of the pipeline and section 4 provides a thorough demonstrating of the tools through a detailed example of a full data analysis pipeline.

2. Motivation and tools for time-invariant analysis

the vast majority of this section has been changed / needs more updates.

EpiCompare delivers *time-invariance* centric analysis by (1) taking a global, not marginal view of how epidemics move through time and (2) by treating full epidemics as filaments, not functions (breaking away from a time-centric view). The following section aims to highlight the strengths of *time-invariance* and define the mathematical foundations that **EpiCompare**'s tools stand upon.

Epidemics are complex objects, and hard to assess and compare to one another due to the differences in the diseases, the location where the outbreak occurs, how the effected population reacts, and the time aspects (including start of the epidemic, speed of infection and more). Time-invariant analysis attempts to make different epidemics easier to compare by removing many time dependent aspects of an epidemic and instead focus on the overall "life" of an epidemic, emphasizing the number of lives affected.

2.1. Motivating time-invariance through R_0

Time-invariance analysis, as it appears in **EpiCompare**, can be seen as an attempt to escape many difficulties comparing different epidemics. With time-invariant analysis, comparing the decades-long outbreak of HIV in the US to a 10 day outbreak of norovirus on a cruise ship is still possible. Time-dependent problems can arise when estimation of certain epidemiological parameters, including the reproductive number R_0, which Gallagher *et al.* (2020) showed can be heavily impacted by choices of beginning and ending time points of an epidemic.

Story: (1) basic definition of R_0, (2) emphasis that R_0 captures a lot of information about an epidemic. Given 3 number example and discuss how one would usually interpret what the differences in R_0 meant (3) maybe connect to # of R_0 estimates for COVID-19 (Aronson et al 2020).

R_0 is probably the most famous time-invariant numerical summary of an epidemic and is associated with the Susceptible-Infectious-Recovered (SIR) model. For a demonstration of R_0 getting lost in a time-dependent analysis we introduce the Kermack and McKendrick (1927)'s common SIR model. This model captures the transitions from one state to the next as a system of ordinary differential equations, where N is the total number of individuals, β is the rate of infection, and γ is the rate of recovery,

$$\begin{aligned} S'(t) &= -\frac{\beta S(t) I(t)}{N} \\ I'(t) &= \frac{\beta S(t) I(t)}{N} - \gamma I(t) \\ R'(t) &= \gamma I(t). \end{aligned} \tag{1}$$

From this model, $R_0 = \beta/\gamma$, aka the ratio of the estimated infection rate compared to the estimated recovery rate.

Interestingly, with regards to traditional epidemic *state* vs. *time* plots, when comparing different epidemics their R_0s are often impossible to compare. This can be seen is a basic example, where we have 2 different epidemics that start in a population of 1000 with 10 individuals infected, but with different parameters related to equation **??**. Specifically suppose that $\beta_1, \gamma_1 = (a, b)$ and $\beta_2, \gamma_2 = (c, d)$ and that we observe both of these epidemics for 15 days. The epidemic trajectories are shown in the *state* vs. time plots in Figure 2. At a glance, we may assume that Model 1 has a larger R_0 than Model 2 because the peak of infection occurs more quickly than in Model 2. On the other hand, we may think Model 2 has a larger R_0 because we may think the number of infections in that model has not yet peaked at time 15. Yet both epidemics have the same R_0 of 2.

However, when we treat both epidemics a functional data through a space that tells us the proportion of the population each each class, as is seen in figure 3 we see a different story. The points seem to overlap and form the same trajectory. Now it seems to be that Model 2 is following the same trajectory as Model 1 but is not as far along in the infection process. We can see there is something fundamentally linking these two different epidemics, and this fundamental link turns out to be R_0.

We can see this mathematically if we let our two epidemics be presented as $\{(S_1(t), I_1(t), R_1(t))\}_{t\geq 0}$, $\{(S_2(s), I_2(s), R_2(s))\}_{s\geq 0}$ respectively. As with the example, assume both models have the same initial values $(S(0), I(0), R(0))$, and let $R_0 = \frac{\beta_1}{\gamma_1} = \frac{\beta_2}{\gamma_2}$ where β_i and γ_i are the average infection rate and recovery rate, respectively, for SIR model $i = 1, 2$. And define $a > 0$ to be the relative scalar such that $\beta_2 = a\beta_1$ if and only if $\gamma_2 = a\gamma_1$.

Theorem 1. *Let there be two SIR models as described above. Then for all $t > 0$ there exists an $s > 0$ such that $(S_1(t), I_1(t), R_1(t)) = (S_2(s), I_2(s), R_2(s))$. Moreover, $s = \frac{1}{a}t$.*

The proof of Theorem 1 relies on a fairly recent result from Harko *et al.* (2014) and is shown in detail in Proof 4.1. The consequence of Theorem 1 is that for two SIR models that have the same initial percent of individuals in each state and R_0 then for every point on the epidemic path of the first SIR model is also a point on the epidemic path of the second SIR model.

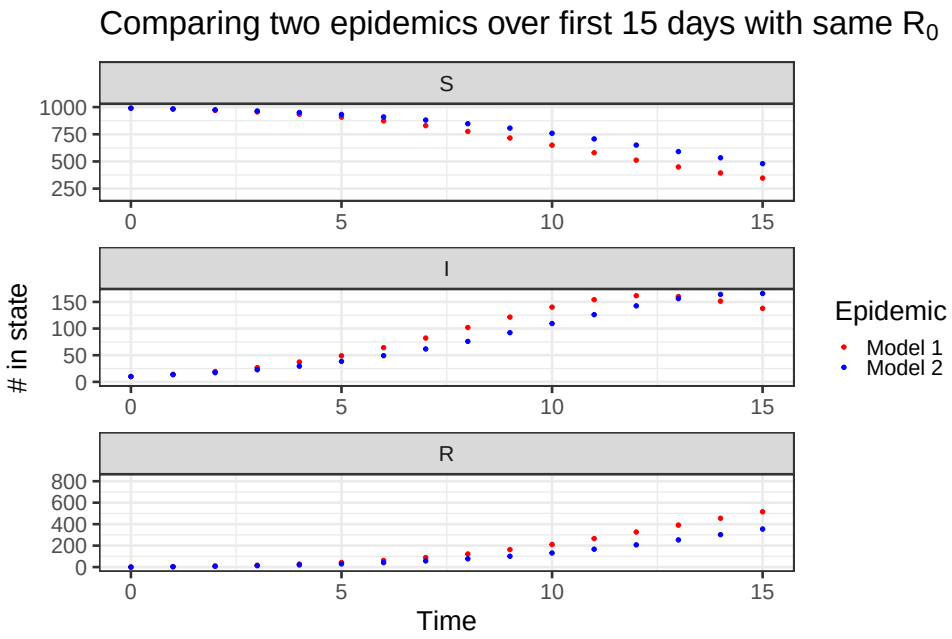

Figure 2: Example of two epidemics with different β and γ paremeters but the same initial reproduction number $R_0 = 2$. Both plots are generated from models with $N = 1000$ individuals with $S(0) = 990$ and $I(0) = 10$.

2.2. Beyond R_0 and Kerman's and McKendrick SIR Models

Through R_0 we saw that treating epidemics more like a function of the time points in some higher dimensional space (nicely visualizable with a ternary plot for the SIR models), then we could better compare the overall structure of the epidemic and how it impacted the population. Without knowing the generative structure underlying these two epidemics, we propose that time-invariant tools can help compare them in smart ways, and that these ideas can extend past the CITE kermack1927 model and to epidemics with more states.

to be continued - (1) highlight filamental view of the epidemics through their simplex space (2) describe distances that could compare different epidemics with this mindset. Also motivate uncertainty for epidemics now that we have a filamental view (3) emphasis that this idea should extend to higher dimensions (4) present the idea that one could also compare bands together.

3. Overview of EpiCompare

In this section, we present the tools implemented in **EpiCompare** and explain how they aid in the data analysis pipeline. In Fig. 3, we illustrate how our ~~package fits~~ package's functions fit into the data analysis pipeline introduced in Fig. 1. All front-facing functions are aimed to be as user-friendly as possible. We also focus on providing the user "tidyverse" style functions, that encourage piping and also follow clear verb naming schemes (Wickham *et al.* 2019). Although users can typically incorporate **EpiCompare** into any step in the data analysis process, there are two primary points of entry. The first point of entry is the very beginning

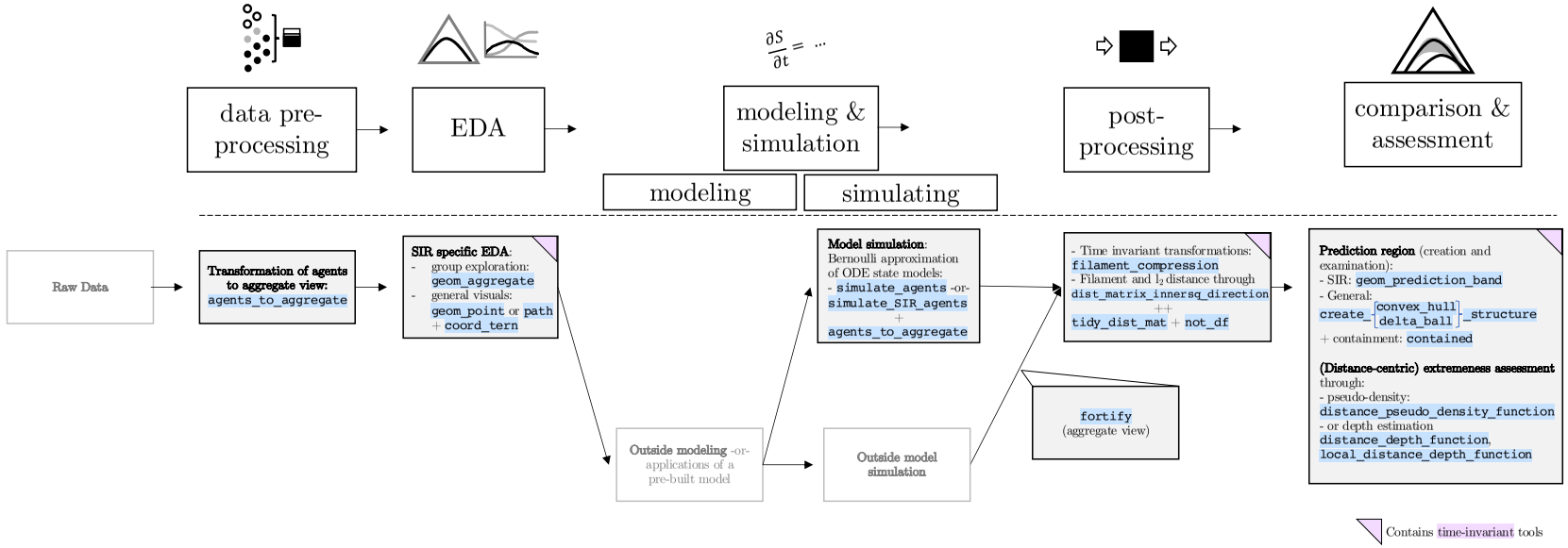

Figure 3: How **EpiCompare** supplements and aids in the epidemiological data analysis pipeline.

with pre-processing and visualizing raw data, and the second point of entry is after modeling and simulation. Figure 3 captures these different paths, and we will highlight both approaches and how to leverage **EpiCompare** in the subsections below.

Data Pre-processing

The first step of most data analysis is cleaning the data so it can ~~to~~ be explored. There are multiple ways to collect epidemiological data. Sometimes individual records are collected, with times of different states of the epidemic (infection, recovery, etc.) as well as individual information like network structure, location, and sub-population information.~~will be collected, whereas~~ Other data collections focus on aggregate counts of individuals in each epidemic state. In fact, usually only the number of new infections at each time step (e.g. weekly case counts) is observed. Compartment totals (amounts of individuals in each state) are then imputed from those case counts along with other information about the disease and the population of interest. In **EpiCompare**, we focus on understanding the overall impact of an outbreak at the aggregate/population level, which allows examination of overall trends of an epidemic more easily.

[Ben says: the following rewrite was done because the original text seemed to repeat the same thing 2x, once in text and the second time with the function `agents_to_aggregate`. The following two paragraphs try to merge the introduction of the function and it's capabilities into one story.]

[Ben says: this whole paragraph is new] In order to help the practitioner examine epidemics from an aggregate/ population lens, we provide a function called `agents_to_aggregate`. This function transforms data about individual/agents' initial entry into each state (e.g. start of infection, start of recovery, etc) to an aggregate view of how many individuals where in a state at a given time. There are often situations where grouping agents into subpopulations (e.g. subpopulations defined by age or sex) can highlight different aggregate level trends. For example, research by Rvachev and Longini (1985); Anderson and May (1992); Worby *et al.* (2015) develop state-based models that account for subpopulations in their models[1]. This function allows the user to easily group agents into new subpopulations by using the **tidyverse** paradigm to first `group_by()`[2] the subpopulation defining features and then aggregate (using `agents_to_aggregate`) conditional on these subpopulations. I think these last two sentences are repetitive

[Ben says: this whole paragraph is new] The `agents_to_aggregate` function is also flexible to deal with a wide range of information about each individual. It can, theoretically, account for infinitely many states, allowing the practitioner to aggregate information relative to the standard "Suspectible", "Infectious", and "Recovered" states (CITE) or also add states like "Exposed", "iMmune", "Hospitalized" and more. Additionally, `agents_to_aggregate` also permits indicators for death/exit and birth/entry dates (that change the overall population size). Don't think this is necessary ~~This function is the highest tested function in this package, and can deal with many nuances that might come-about in the individual data.~~ The function `agents_to_aggregate()` is a powerful tool in the step of pre-processing data, one that quickly and simply transforms individual level information into an aggregate view that can be processed and analyzed in a simpler manner.

~~In **EpiCompare**, we provide a function to transform information about each individual/agent's~~

[1] [Ben says: I don't really like this sentence, but I'm trying to include the references - are they needed?]
[2] The `group_by` function is from the **dplyr** package.

initial time of entry into each state (e.g. start of infection, start of recovery, etc). This transformation between *agent* information to *aggregate* information is useful for seeing the overall trend of the epidemic (and how it impacts different subpopulations). Our tools allow the user to easily group agents and define new subpopulations to explore. This is important as case studies highlight the usefulness of identifying differing subpopulations (e.g. age, sex) and many state based models provide for subpopulation-based states in their analysis (Rvachev and Longini 1985; Anderson and May 1992; Worby *et al.* 2015).

\ben{We provide a "tidyverse"-styled function, `agents_to_aggregate()` to transform agent information into aggregate state information. As a "tidy" function, our function `agents_to_aggregate()` allows the user to identify and analyze subpopulations by first executing a `group_by()`
\ben{ from **dplyr**. The function `agents_to_aggregate`is flexible in that there is no limit on the amount of permitted epidemiological states so users can incorporate information from compartments such as "Exposed", "iMmune", or "Hospitalized" groups, for instance, and also permits indicators for death/exit and birth/entry dates. Our function, `agents_to_aggregate()` does not require agents to pass through every state listed and agents are allowed to start in any state. This function is constrained to integer time steps (for example days), but transformations (linear or otherwise) of the time columns can account for finer grained time steps (e.g. hours or minutes). The function `agents_to_aggregate()` returns the total number of individuals in each state at each time point. We view `agents_to_aggregate()` as powerful tool in the step of pre-processing data, one that quickly and simply transforms individual level information into an aggregate view that can be processed and analyzed in a simpler manner.

EDA

[Ben says: I'm went back and thought that the version before this version was clearer. Wanna take a look and discuss? Side by side: [github](https://github.com/skgallagher/EpiCompare/commit/7dbb49

[Ben says: this is actually an older version:] With raw data, "getting to know" our data currently means figuring out good combinations of visualizations, numerical summaries and subsets. An expert coder can start from `agents_to_aggregate` to successfully do this in many ways, but **EpiCompare** also includes tools to rapidly explore your data if your has at least three epidemic states (although these tools work best with exactly three epidemic states). Our `geom_aggregate` provides a rapid way to explore different subpopulations' experience of the epidemic, and combines the ideas behind `agents_to_aggregate` for the SIR case, with `geom_path` and `coord_tern` to visualize any number of groups epidemic trajectory in 3d simplex space using **ggplot2** and **ggtern** (Wickham 2016; Hamilton and Ferry 2018). Visualization tools for SIR models were developed because (1) SIR models are some of the most common and basic epidemic state-based models and (2) our simplex representation of these epidemics emphasizes a "time-invarance" representation of the data (for a refresher see Section 2).

[Ben says: The 2 things I really like about the earlier (now current) version is (1) the introduction seems to make more sense in getting at what EDA really is, (2) the way it disucsses SIR models only 1 potential model instead of putting it as *the* model.]

With raw data, exploratory analysis frequently means figuring out good combinations of visualizations, numerical summaries, and groupings. An expert coder can start from `agents_to_aggregate()` to successfully accomplish EDA in many ways, but we have also developed tools to allow a novice coder to rapidly explore data, as long as there three unique epidemiological states (like the SIR model). Our `geom_aggregate()` provides a rapid way to explore different

~~subpopulations' experiences of the epidemic in the ternary lens. It combines the ideas behind~~ `agents_to_aggregate()` \ben{~~with **ggplot2**'s~~ `geom_path()` ~~and **ggtern**'s~~ `coord_tern()` ~~to visualize any number of groups epidemic trajectory in 3d simplex~~ (Wickham 2016; Hamilton and Ferry 2018).~~We developed these out-of-the-box visualization tools for SIR models because (1) SIR models are some of the most common and basic epidemic state-based models and (2) our simplex representation of these epidemics emphasizes a "time-invarance" representation of the data (see Section 2).~~

Model Fitting and Simulations [Ben says: see footnote 3 about the introduction to this paragraph.]

Although this package does not focus on fitting a model to data, we do provide some flexible functions for simulation of basic discrete-time epidemic-state models.~~ with Bernoulli or Multinomial transitions which can be used to (noisily) approximate ODEs which describe transitions from one state to the next~~[3]. These functions can be naturally combined with `agents_to_aggregate()` to proceed in the pipeline. The function `simulate_SIR_agents()` simulates an SIR epidemic with user inputs for the number of simulations, the initial number in each state, the infection and recovery parameters (β, γ), and the total number of discrete time steps. This function allows for easy access to SIR model analysis and comparison. Beyond SIR models, the function `simulate_agents()` takes as input a user-specified transition matrix and other epidemic parameters to allow the user to create simulations of an outbreak for *any* number of states and any number of transitions among them. This allows for users to explore the space of models in an intuitive way without getting bogged down by too much mathematical detail. For consistency, we have made output from `simulate_agents()` and `simulate_SIR_agents()` compatible with `agents_to_aggregate()` so aggregate information may easily be accessed.

Post-processing

Post-processing of modeling and simulation consists of making summary statistics, plots, tables, and other ways to disseminate information to the public. For example, comma separated value files (`.csv`) are a standard way to share information within tables. However, model output is often far more complicated than what a traditional `.csv` would allow. Do we need three sentences about csvs? As a result, a number of epidemic modeling packages return a special class, specific to their modeling. The special classes often contain a plethora of information from residuals, model diagnostics, input parameters, and more. While incredibly useful, these special classes can be difficult for novice coders to navigate.

To this end, we have adapted a series of `fortify`-style functions, called `fortify_aggregate()` which transform output from packages like **pomp** and **EpiModel** into tidy-styled data frames which contain information about the total number of individuals in each state at a given time, for a given simulation. These fortify functions have output that is consistent with that of `agents_to_aggregate()`.

Comparisons and Assessment

Comparison and assessment of model fit or comparisons of one model to another model can

[3]The original sentence seemed clearer to me (it may even be a run-on sentence now). I'm wondering if you didn't like the original sentence relative to the "truth-ful-ness" of the statement? Here's the original sentence slightly edited: "Although this package does not focus on estimating a model for the data, we do provide some power functions for simulation of basic state models with Bernoulli approximation of ODE state models encoded in simulate_agents and simulate_SIR_agents".

be performed in a variety of ways including mean square error, AIC, plots, and more. Perhaps the most useful tool **EpiCompare** has to offer to the expert, for comparison and assessment of models, is in its post-processing tools which create a standard output. It is then a matter of writing a script or function made for that standard output to assess the results from multiple models in the way the user desires.

However, for those who like more concrete tools, **EpiCompare** offers functions to compare prediction regions to one another including `geom_prediction_band()` (which plots the region), and `create_{convex_hull,delta_ball}_structure()`(which returns the `R` output for the given structure), and `contained()` (which allows the user to determine if one set of points is contained in a prediction band). Additionally, we offer ways to determine if model outputs are compatible with one another, that is how extreme one output is to another. Ben says something about distance

4. A tour of EpiCompare

In this section, we highlight a number of the functionalities available in **EpiCompare**. These functionalities include data cleaning, visualization, simulation, and comparison, in accordance with the data analysis pipeline 1. We show a full data analysis from beginning to end that can be accomplished in a streamlined and standardized manner.

4.1. Data and exploratory analysis

We analyze an outbreak of measles in the town of Hagelloch, Germany from 1861-1862, a data set organized by Pfeilsticker (1863). The data was later made visible by Oesterle (1992) and made available in an R by Meyer *et al.* (2017). The Hagelloch data includes a rich set of features including household members, school level, household locations, date of first symptoms (prodromes), date of measles rash, and even the alleged infector. A subset of the data is shown in Table 1. Because of these rich features, this data set has been an ideal testing ground methodology in infectious disease epidemiology and is used in work by Neal and Roberts (2004); Britton *et al.* (2011); Groendyke *et al.* (2012); Becker *et al.* (2016).

Table 1: Subset of Hagelloch infection data. Features include the person ID, household ID (HH ID), age, sex, class level (Pre-K/1st/2nd), date of first symptoms, date of the appearance of the measles rash, and the alleged infector ID of the individual.

ID	HH ID	Name	Age	Sex	Class	Symp. Start	Rash Date	Infector ID
1	61	Mueller	7	female	1st class	1861-11-21	1861-11-25	45
2	61	Mueller	6	female	1st class	1861-11-23	1861-11-27	45
3	61	Mueller	4	female	preschool	1861-11-28	1861-12-02	172
4	62	Seibold	13	male	2nd class	1861-11-27	1861-11-28	180
5	63	Motzer	8	female	1st class	1861-11-22	1861-11-27	45
45	51	Goehring	7	male	1st class	1861-11-11	1861-11-13	184

With **EpiCompare**, we can easily obtain the empirical cumulative incidence function with respect to the measles rash appearance (variable `ERU`) with the following tidy-style function,

`agents_to_aggregate`. The function `agents_to_aggregate` is a key component of **EpiCompare**, allowing the user to easily switch from an individual-level (i.e. an agent) view of a disease to an aggregate level. For example, the below code shows how we can convert the agent data to a cumulative incidence of the measles rash, in order to see how the disease spread through the population over time. We can then compare the cumulative incidence of the rash to the cumulative incidence of the prodromes, i.e. the initial symptoms. We do this with the below code, and a part of the cumulative incidence data output are shown in Table 2. The argument `integer_time_expansion` indicates whether we should include all time points in the recorded range of the data or only when there is a change in the incidence.

```
R> cif_rash  <- hagelloch_raw %>%
+   mutate(time_of_rash = as.numeric(ERU - min(PRO, na.rm = TRUE))) %>%
+   agents_to_aggregate(states = time_of_rash,
+                       integer_time_expansion = FALSE) %>%
+   mutate(type = "Rash")
```

Table 2: Turning the individual-level information from the Hagelloch data to an aggregate view of the cumulative incidence of the measles rash in the population over time.

Time	# Susceptible	# Total rash appearances
0	188	0
4	187	1
7	186	2
9	185	3
12	183	5

One question of interest is the duration between initial onset of prodromes or symptoms and the appearance of the measles rash. Since `agent_to_aggregate` outputs a tidy-style data frame, it is a simple task to plot the two sets of incidence curves on the same graph (Fig. 4).

```
R> cif_prodromes <- hagelloch_raw %>%
+   mutate(time_of_PRO = as.numeric(PRO - min(PRO, na.rm = TRUE))) %>%
+   agents_to_aggregate(states = time_of_PRO,
+                       integer_time_expansion = FALSE) %>%
+   mutate(type = "Pro")

R> plot_df <- bind_rows(cif_rash, cif_prodromes)
R>
R> ggplot(data = plot_df,
+        aes(x = t, y = X1, col = type)) +
+   geom_step() +
+   labs(title = "Cumulative incidence of measles appearance",
+        x = "Time (days relative to first prodrome appearance)",
+        y = "Cumulative incidence of event") +
+   coord_cartesian(xlim = c(0, 55)) +
+   scale_color_manual(values = c("blue", "red"))
```

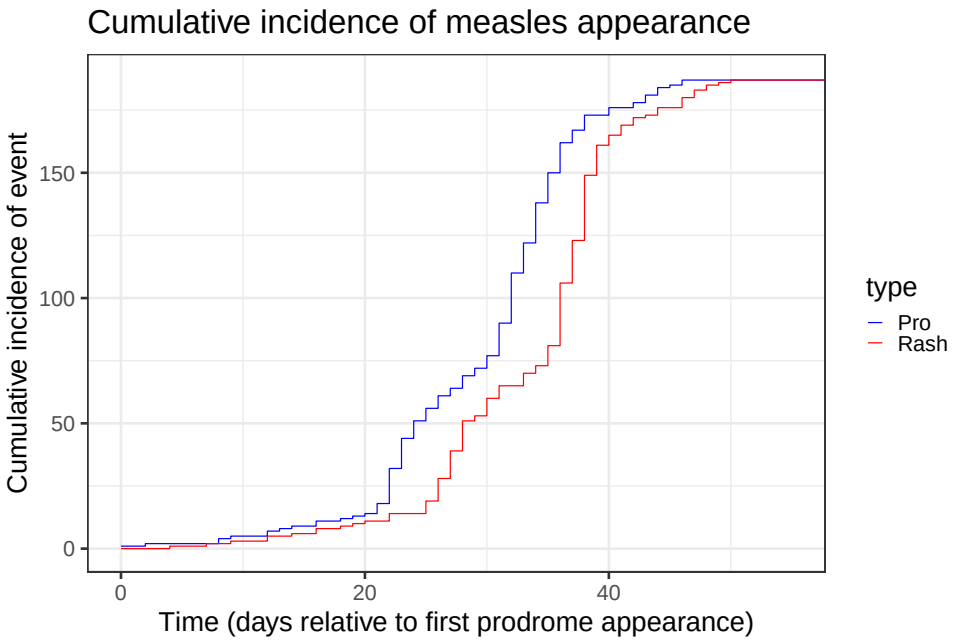

Figure 4: Empirical cumulative incidence functions of prodrome (symptom) onset and measles rash appearance. We see that there is approximately a a constant lag between the two curves.

The real power of `agents_to_aggregate()` lies in its ability to aggregate over any number of pre-specified states. For example, the Hagelloch data sets contains two columns, `tI` and `tR`, the time of infection and recovery, respectively of each individual. We can then plot the SIR values through a time-invariant lens using **ggplot2** and **ggtern** functions (as shown in Fig. 5) or with our custom `geom`, `geom_aggregate`, which takes the raw agent data as input.

```
R> hagelloch_sir <- hagelloch_raw %>%
+   agents_to_aggregate(states = c(tI, tR),
+                       min_max_time = c(0, 55)) %>%
+   rename(time = t, S = X0, I = X1, R = X2)
R>
R>
R> ggplot(hagelloch_sir, aes(x = S, y = I, z = R))+
+   coord_tern() +
+   geom_path() +
+   labs(x = "S", y = "I", z = "R",
+        title = "Time invariant view of Hagelloch measles outbreak") +
+   theme_sir(base_size = 24)
```

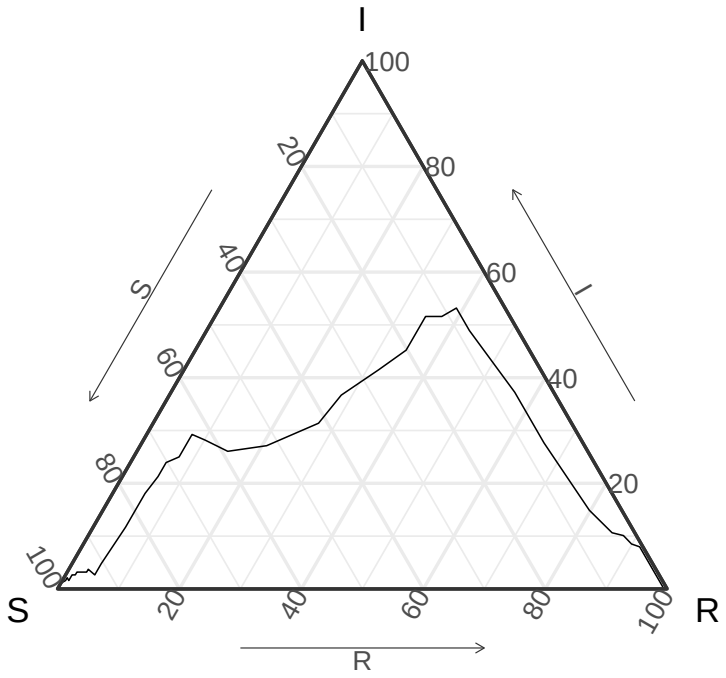

Figure 5: Time invariant view of the Hagelloch epidemic where we view the individuals in Susceptible, Infectious, or Recovered states. We see there are two peaks of infection (the vertical axis).

Moreover, we can look at the outbreaks of the disease by group within `agent_to_aggregate()` or `geom_aggregate()`. This allows us to examine differences among the different groups of individuals. For example, we show the time invariant outbreak by class level in Figure 6. Immediately, we see that time invariant infection curve is different for the pre-school class compared to the 1st class. In the 1st class, we see about 95% of the class become infected and less than 10% of them having recovered, which is indicative of a super-spreading event. This suspicion is further confirmed in that 26 of the 30 1st class students have been reportedly infected by the same individual.

```
R> hagelloch_raw %>%
+   ggplot(aes(y = tI, z = tR, color = CL)) +
+   geom_aggregate(size = 2) + coord_tern() +
+   labs(x = "S", y = "I", z = "R",
+        color = "Class") +
+   scale_color_brewer(palette = "Dark2") +
+   facet_wrap(~CL)
```

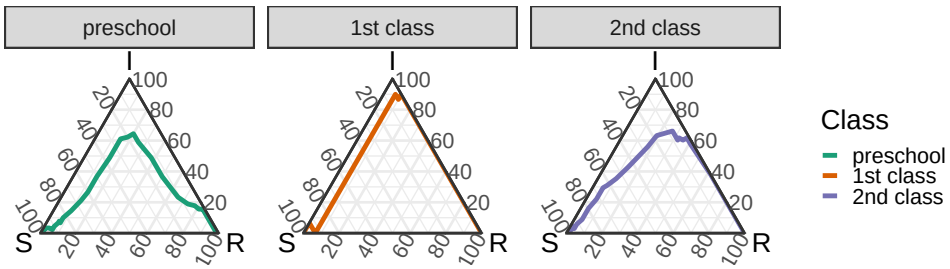

Figure 6: Time invariant outbreak curves for the three class groups. The pre-school class has a distinct peak of infection whereas the peak infection point for the other two classes are less well defined.

Along with multiple epidemic states, the function `agents_to_aggregate` can also be extended to populations with vital dynamics (e.g. birth and death) and examples of this are shown in the package vignette. In summary, `agents_to_aggregate()` is a multi-purpose workhorse that may be leveraged to convert individual level records into aggregate information that may be more useful for some forms of epidemic modeling such as compartment modeling.

Up to this point, we have used **EpiCompare** in the context of observed data. We also want to compare statistical models, and **EpiCompare** aids in that process via a simple but dynamic individual-level data generator, conversion tools for popular epidemic model packages, and model assessments. We demonstrate an example here.

We first try to model the Hagelloch data with a stochastic SIR model, which we refer to as the 'simple SIR.' In our vignette, we show how to fit this simple SIR model via maximum likelihood and simulate from the model with those best fit parameters. Our function `simulate_agents()` generates individual level data according to discrete time multinomial draws, which depend on the number of individuals in each state at the previous time step and a matrix of transition probabilities. For example, the below code generates 100 simulations of an outbreak of a disease with one initial infector in a population of $n = 188$ individuals.

```
R> trans_mat <- matrix(c("X0 * (1 - X1 * par1 / N)", "X0 * X1  * par1 / N", "0",
+                        "0", "X1 * (1 - par2)", "par2 * X1",
+                        "0", "0", "X2"), byrow = TRUE, nrow = 3)

R> set.seed(2020)
R>
R> best_params <- c("beta" = .36, "gamma" = .13)
R> ## This is the SIR representation
R>
R> rownames(trans_mat) <- c("S", "I", "R")
R> init_vals <- c(187, 1, 0)
R> par_vals <- c(par1 = best_params[1], par2 = best_params[2])
R> max_T <- 55
R> n_sims <- 100
R>
R> agents <- simulate_agents(trans_mat,
+                            init_vals,
```

```
+                         par_vals,
+                         max_T,
+                         n_sims,
+                         verbose = FALSE)

R> agg_model <- agents %>% group_by(sim) %>%
+   agents_to_aggregate(states = c(I, R)) %>%
+   mutate(Type = "Simple SIR")
```

The result of our simulation is the object `agents` which is a 18800 $\times$ 5 data frame, which details the time of entry into the S, I, and R states for a given simulation. Before we examine the results of this simple SIR model, we will also examine another, more sophisticated SIR model, this time from the package **EpiModel**. Briefly, this model first fits a contact network to the set of indivduals, where the class of the student is a covariate. The model then simulates a SIR-epidemic on that network.

```
R> library(EpiModel)
R> ## WARNING:  Will take a minute or two
R>
R> set.seed(42)
R> nw <- network.initialize(n = 188, directed = FALSE)
R> nw <- set.vertex.attribute(nw, "group", rep(0:2, each = 90, 30, 68))
R> formation <- ~edges + nodematch("group") + concurrent
R> target.stats <- c(200, 300, 200)
R> coef.diss <- dissolution_coefs(dissolution = ~offset(edges),  duration = 5)
R> est1 <- netest(nw, formation, target.stats, coef.diss, edapprox = TRUE)
R>
R> param <- param.net(inf.prob = 0.1, act.rate = 5, rec.rate = 0.1)
R> status.vector <- c(rep(0, 90), rep(0, 30), rep(0, 67), 1)
R> status.vector <- ifelse(status.vector == 1, "i", "s")
R> init <- init.net(status.vector = status.vector)
R> control <- control.net(type = "SIR", nsteps = 55,
+                         nsims = 100, epi.by = "group")
R> epimodel_sir <- netsim(est1, param, init, control)
```

The output of this model is `epimodel_sir`, an object of class `netsim`, which contains a plethora of modeling information. We provide the function `fortify_aggregate()`, which can take objects from specialized classes of modeling output and transform it into a tidy-style data frame.

```
R> fortified_net <- fortify_aggregate(epimodel_sir,
+                                     states = c("s.num", "i.num", "r.num")) %>%
+   mutate(Type = "EpiModel SIR",
+          sim = as.numeric(gsub("sim", "", sim)))
```

We can then analyze the results of the two models side by side as time-invariant epidemic curves. The results are shown in Figure 7, where a 90% prediction band is estimated from

the delta ball method for each of the two models. For the Simple SIR model, we see that the data generally covers the data fairly well but clearly misses the second peak of infection. We also see that the prediction band is very large, covering up a large area of the ternary plot. On the other hand, for the **EpiModel** model, we see that the prediction band covers the data quite well and takes up less area.

```
R> both_models <- bind_rows(agg_model, fortified_net)
R>
R>
R> g <- ggplot() + geom_prediction_band(data = both_models %>% filter(t != 0),
+          aes(x = X0, y = X1, z = X2,
+               sim_group = sim, fill = Type),
+          alpha = .5,
+          conf_level = .90)
```

```
R> g +   geom_path(data = both_models %>% filter(t !=0),
+             aes(x = X0, y = X1, z = X2, group = paste(Type, sim)),
+             alpha = .3, col = "gray40") +
+     coord_tern() + theme_sir(base_size = 24) +
+   geom_point(data = hagelloch_sir,
+              aes(x = S, y = I, z =R), col = "black") +
+   labs(title = "Simple SIR model",
+        subtitle = "90% Prediction band and original data",
+        x = "S", y = "I", z = "R") +
+   scale_fill_manual(values = c("#006677", "#AA6600")) +
+   facet_wrap(~Type) +
+   theme(legend.position = "bottom")
```

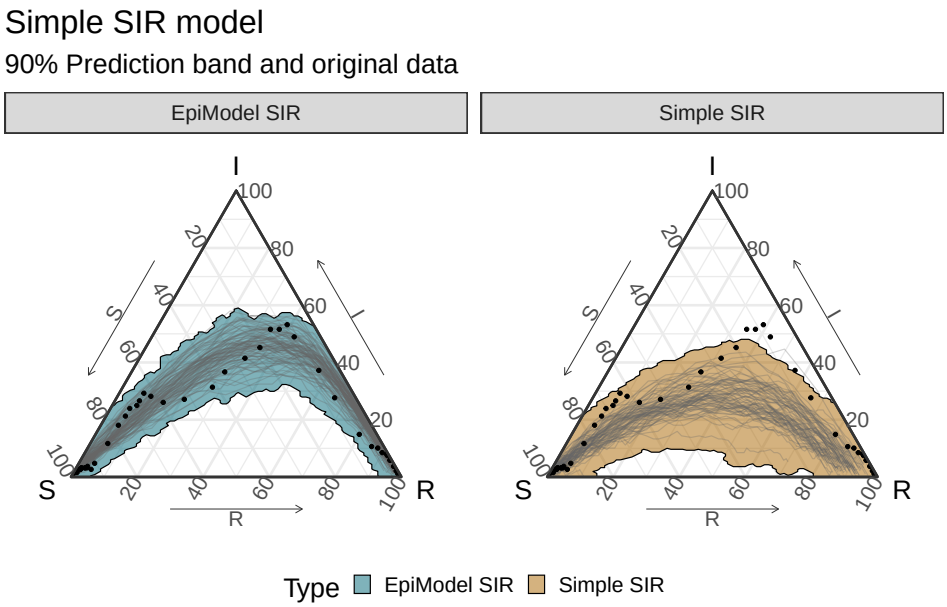

Figure 7: Original Hagelloch SIR data (black) along with 90% prediction band and actual simulation paths from the Simple SIR and the EpiModel SIR models.

However, both models are not a good fit to the filamental path as opposed to the individual points in (S, I, R)-space. This can be captures with the set of simulations both models predict, which all generally have a single defined peak of infection whereas the data certainly looks like it has two distinct peaks, likely caused by our assumed super-spreader event. This observation is backed up by the below analysis that demonstrates that the estimated psuedo-density of the observed epidemic (relative to the simulations from either model) is much less likely then **any** of the simulations (reported in Table 4. In conclusion, **EpiCompare** makes it clear that, at a glance, 1) the EpiModel network model is a better fit than the Simple SIR model, and 2) the fit is only good at the individual point level as opposed to the epidemic path level.

```
R> #-- after cleaning up and combining --
R> all_together_df <- rbind(simple_sir,
+                           hagelloch_sir2)
```

Table 3: Top and bottom 2 rows of `all_together_df, combining both simulated epidemics and the tr`

Type	sim	t	S	I	R
Simple SIR	1	0	188	0	0
Simple SIR	1	1	187	1	0
true observation	0	54	1	0	187
true observation	0	55	1	0	187

```
R> compression_df <- all_together_df %>% group_by(Type, sim) %>%
+   filament_compression(data_columns = c("S","I","R"),
+                        number_points = 20)
```

```
R> tdmat <- compression_df %>%
+   dist_matrix_innersq_direction(
+     position = c(1:length(compression_df))[
+       names(compression_df) %in% c("S","I", "R")],
+     tdm_out = T)
R>
R> simple_sir_true_obs_info <- tdmat %>%
+   compare_new_to_rest_via_distance(
+     new_name_id = data.frame(Type = "true observation", sim = 0),
+     distance_func = distance_psuedo_density_function,
+     sigma = "20%")
```

Table 4: The extremeness of the true simulations based on comparing psuedo-density estimates between true vs simulated curves

Type	simulations-based estimated psuedo-density	proportion of simulations with lower estimated psuedo-density
Simple SIR	0.0036733	0.00
EpiModel SIR	0.0149686	0.02

A. Appendix

A.1 Proof of Theorem 1

Proof. Harko *et al.* (2014) provide an analytical solution for the Kermack and McKendrick equations (Eq. (1)) by reparameterizing the ODEs so that $\mathcal{S}(u) = S(t)$, $\mathcal{I}(u) = S(t)$, and $\mathcal{R}(u) = R(t)$ for $0 < u_T < 1$ with

$$\begin{aligned} \mathcal{S}(u) &= S(0)u \\ \mathcal{I}(u) &= N - R(0) + NR_0^{-1}\log u - S(0)u \\ \mathcal{R}(u) &= R(0) - NR_0^{-1}\log u, \end{aligned} \tag{2}$$

and u and t are related by the following integral,

$$\begin{aligned} t &= \int_u^1 \frac{N}{\beta\tau(N - R(0) + R_0^{-1}\log\tau - S(0)\tau)}d\tau \\ &= \int_u^1 \frac{1}{\beta f(S(0), R(0), N, R_0, \tau)}d\tau \\ &= \int_u^1 \frac{1}{\beta f(\tau)}d\tau, \end{aligned}$$

where we have made the denominator of the integral a function of N, the initial values, R_0, and τ, which we further condense to $f(\tau)$ for brevity. Then for a given t we want to find s

such that $(S_1(t), I_1(t), R_1(t)) = (S_2(s), I_2(s), R_2(s))$. Or equivalently, for a fixed u want to find v such that $\mathcal{S}_1(u) = \mathcal{S}_2(v)$ and then the corresponding t and s are given by

$$t = \int_u^1 \frac{1}{\beta_1 f(\tau)} d\tau$$
$$s = \int_v^1 \frac{1}{\beta_2 f(\tau)} d\tau.$$

Note that since the equations in Eq. (2) are functions of the initial values and R_0, then $u = v$. We then can find a relation for s,

$$\begin{aligned} s &= \int_u^1 \frac{1}{\beta_2 f(\tau)} d\tau \\ &= \int_u^1 \frac{1}{a\beta_1 f(\tau)} d\tau \\ &= \frac{1}{a} \int_u^1 \frac{1}{\beta_1 f(\tau)} d\tau \\ &= \frac{1}{a} t. \end{aligned}$$

□

References

Anderson RM, May RM (1992). *Infectious diseases of humans: dynamics and control*. Oxford university press.

Becker AD, Birger RB, Teillant A, Gastanaduy PA, Wallace GS, Grenfell BT (2016). "Estimating enhanced prevaccination measles transmission hotspots in the context of cross-scale dynamics." *Proceedings of the National Academy of Sciences*, **113**(51), 14595–14600.

Biggerstaff M, Alper D, Dredze M, Fox S, Fung ICH, Hickmann KS, Lewis B, Rosenfeld R, Shaman J, Tsou MH, Velardi P, Vespignani A, Finelli L, Chandra P, Kaup H, Krishnan R, Madhavan S, Markar A, Pashley B, Paul M, Meyers LA, Eggo R, Henderson J, Ramakrishnan A, Scott J, Singh B, Srinivasan R, Bakach I, Hao Y, Schaible BJ, Sexton JK, Del Valle SY, Deshpande A, Fairchild G, Generous N, Priedhorsky R, Hickman KS, Hyman JM, Brooks L, Farrow D, Hyun S, Tibshirani RJ, Yang W, Allen C, Aslam A, Nagel A, Stilo G, Basagni S, Zhang Q, Perra N, Chakraborty P, Butler P, Khadivi P, Ramakrishnan N, Chen J, Barrett C, Bisset K, Eubank S, Anil Kumar VS, Laskowski K, Lum K, Marathe M, Aman S, Brownstein JS, Goldstein E, Lipsitch M, Mekaru SR, Nsoesie EO, Gesualdo F, Tozzi AE, Broniatowski D, Karspeck A, Tse ZTH, Ying Y, Gambhir M, Scarpino S (2016). "Results from the centers for disease control and prevention's predict the 2013-2014 Influenza Season Challenge." *BMC Infectious Diseases*, **16**(1), 1–10. ISSN 14712334. doi:10.1186/s12879-016-1669-x. URL http://dx.doi.org/10.1186/s12879-016-1669-x.

Britton T, Kypraios T, O'Neill PD (2011). "Inference for epidemics with three levels of mixing: methodology and application to a measles outbreak." *Scandinavian Journal of Statistics*, **38**(3), 578–599.

CDC (2021). "CDC COVID Data Tracker." URL https://covid.cdc.gov/covid-data-tracker/#cases_casesper100klast7days.

Dong E, Du H, Gardner L (2020). "An interactive web-based dashboard to track COVID-19 in real time." *The Lancet infectious diseases*, **20**(5), 533–534.

Ferguson N, Laydon D, Nedjati Gilani G, Imai N, Ainslie K, Baguelin M, Bhatia S, Boonyasiri A, Cucunuba Perez Z, Cuomo-Dannenburg G, *et al.* (2020). "Report 9: Impact of non-pharmaceutical interventions (NPIs) to reduce COVID19 mortality and healthcare demand."

Gallagher S, Chang A, Eddy WF (2020). "Exploring the nuances of R0: Eight estimates and application to 2009 pandemic influenza." *arXiv preprint arXiv:2003.10442*.

Groendyke C, Welch D, Hunter DR (2012). "A network-based analysis of the 1861 Hagelloch measles data." *Biometrics*, **68**(3), 755–765.

Hamilton NE, Ferry M (2018). "ggtern: Ternary Diagrams Using ggplot2." *Journal of Statistical Software, Code Snippets*, **87**(3), 1–17. doi:10.18637/jss.v087.c03.

Harko T, Lobo FS, Mak MK (2014). "Exact analytical solutions of the Susceptible-Infected-Recovered (SIR) epidemic model and of the SIR model with equal death and birth rates." *Applied Mathematics and Computation*, **236**, 184–194. ISSN 00963003. doi:10.1016/j.amc.2014.03.030. 1403.2160, URL http://dx.doi.org/10.1016/j.amc.2014.03.030.

Jenness SM, Goodreau SM, Morris M (2018). "EpiModel: An R Package for Mathematical Modeling of Infectious Disease over Networks." *Journal of Statistical Software*. doi:10.18637/jss.v084.i08.EpiModel.

Kermack WO, McKendrick AG (1927). "A contribution to the mathematical theory of epidemics." *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, **115**(772), 700–721.

King AA, Nguyen D, Ionides EL (2016). "Statistical inference for partially observed markov processes via the R package pomp." *Journal of Statistical Software*, **69**(12), 1–43. ISSN 15487660. doi:10.18637/jss.v069.i12. 1509.00503.

Meyer S, Held L, Höhle M (2017). "Spatio-Temporal Analysis of Epidemic Phenomena Using the R Package surveillance." *Journal of Statistical Software*, **77**(11), 1–55. doi:10.18637/jss.v077.i11.

MIDAS Network (2021). "Online Portal for COVID-19 Modeling and Research." URL https://midasnetwork.us/covid-19/.

Neal PJ, Roberts GO (2004). "Statistical inference and model selection for the 1861 Hagelloch measles epidemic." *Biostatistics*, **5**(2), 249–261. ISSN 14654644. doi:10.1093/biostatistics/5.2.249.

Oesterle H (1992). "Statistische Reanalyse einer Masernepidemie 1861 in Hagelloch."

Pfeilsticker A (1863). "Beiträge zur Pathologie der Masern mit besonderer Berücksichtigung der statistischen Verhältnisse." URL http://www.archive.org/details/beitrgezurpatho00pfeigoog.

Rvachev LA, Longini IM (1985). "A mathematical model for the global spread of influenza." *Mathematical Biosciences*, **75**(1), 3 – 22. ISSN 0025-5564. doi:http://dx.doi.org/10.1016/0025-5564(85)90064-1. URL http://www.sciencedirect.com/science/article/pii/0025556485900641.

The Washington Post (2021). "Coronavirus US Cases and." URL https://washingtonpost.com/graphics/2020/national/coronavirus-us-cases-deaths/.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.

Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, Grolemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V, Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). "Welcome to the tidyverse." *Journal of Open Source Software*, **4**(43), 1686. doi:10.21105/joss.01686.

Worby CJ, Chaves SS, Wallinga J, Lipsitch M, Finelli L, Goldstein E (2015). "On the relative role of different age groups in influenza epidemics." *Epidemics*, **13**, 10–16.

Affiliation:

Shannon K. Gallagher
Biostatistics Research Branch
National Institute of Allergy
and Infectious Diseases
5603 Fishers Lane
Rockville, MD 20852
E-mail: shannon.gallagher@nih.gov
URL: http://skgallagher.github.io

Benjamin LeRoy
Dept. of Statistics & Data Science
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
E-mail: bpleroy@andrew.cmu.edu
URL: https://benjaminleroy.github.io/

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd