SOFTWARE TOOL ARTICLE

epicontacts: Handling, visualisation and analysis of epidemiological contacts [version 1; referees: 1 approved with reservations]

VP Nagraj1, Nistara Randhawa2, Finlay Campbell3, Thomas Crellen4, Bertrand Sudre5, Thibaut Jombart3

1Research Computing, University of Virginia School of Medicine, Charlottesville, VA, 22903, USA
2One Health Institute, University of California, Davis, Davis, CA, 95616, USA
3MRC Centre for Outbreak Analysis and Modelling, Department of Infectious Disease Epidemiology, School of Public Health, Imperial College London, London, W2 1PG, UK
4Mahidol-Oxford Tropical Medicine Research Unit, Bangkok , 10400, Thailand
5European Centre for Disease Prevention and Control, Stockholm, Sweden

Abstract
Epidemiological outbreak data is often captured in line list and contact format to facilitate contact tracing for outbreak control. epicontacts is an R package that provides a unique data structure for combining these data into a single object in order to facilitate more efficient visualisation and analysis. The package incorporates interactive visualisation functionality as well as network analysis techniques. Originally developed as part of the Hackout3 event, it is now developed, maintained and featured as part of the R Epidemics Consortium (RECON). The package is available for download from the Comprehensive R Archive Network (CRAN) and GitHub.

Keywords
contact tracing, outbreaks, R

This article is included in the RPackage gateway.

This article is included in the R Epidemics Consortium (RECON) collection.
Corresponding author: VP Nagraj (vpnagraj@virginia.edu)

Author roles: Nagraj V: Conceptualization, Software, Writing – Original Draft Preparation; Randhawa N: Conceptualization, Software, Writing – Original Draft Preparation; Campbell F: Conceptualization, Software, Writing – Original Draft Preparation; Crellin T: Conceptualization, Software; Sudre B: Conceptualization; Jombart T: Conceptualization, Software, Writing – Original Draft Preparation

Competing interests: No competing interests were disclosed.

How to cite this article: Nagraj V, Randhawa N, Campbell F et al. epicontacts: Handling, visualisation and analysis of epidemiological contacts [version 1; referees: 1 approved with reservations] F1000Research 2018, 7:566 (doi: 10.12688/f1000research.14492.1)

Copyright: © 2018 Nagraj V et al. This is an open access article distributed under the terms of the Creative Commons Attribution Licence, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Grant information: The author(s) declared that no grants were involved in supporting this work.

Introduction
In order to study, prepare for, and intervene against disease outbreaks, infectious disease modellers and public health professionals need an extensive data analysis toolbox. Disease outbreak analytics involve a wide range of tasks that need to be linked together, from data collection and curation to exploratory analyses, and more advanced modelling techniques used for incidence forecasting\(^1\)\(^2\) or to predict the impact of specific interventions\(^3\)\(^4\). Recent outbreak responses suggest that for such analyses to be as informative as possible, they need to rely on a wealth of available data, including timing of symptoms, characterisation of key delay distributions (e.g. incubation period, serial interval), and data on contacts between patients\(^5\)\(^6\)\(^7\)\(^8\).

The latter type of data is particularly important for outbreak analysis, not only because contacts between patients are useful for unravelling the drivers of an epidemic\(^9\)\(^10\), but also because identifying new cases early can reduce ongoing transmission via contact tracing, i.e. follow-up of individuals who reported contacts with known cases\(^11\)\(^12\). However, curating contact data and linking them to existing line lists of cases is often challenging, and tools for storing, handling, and visualising contact data are often missing\(^3\)\(^4\).

Here, we introduce epicontacts, an R\(^13\) package providing a suite of tools aimed at merging line lists and contact data, and providing basic functionality for handling, visualising and analysing epidemiological contact data. Maintained as part of the R Epidemics Consortium (RECON), the package is integrated into an ecosystem of tools for outbreak response using the R language.

Methods

Operation
epicontacts is released as an open-source R package. A stable release is available for Windows, Mac and Linux operating systems via the CRAN repository. The latest development version of the package is available through the RECON Github organization. At minimum users must have R installed. No other system dependencies are required.

```
# install from CRAN
install.packages("epicontacts")

# install from Github
install.packages("devtools")
devtools::install_github("reconhub/epicontacts")

# load and attach the package
library(epicontacts)
```

Implementation

Data handling. epicontacts includes a novel data structure to accommodate line list and contact list datasets in a single object. This object is constructed with the make_epiconctacts() function and includes attributes from the original datasets. Once combined, these are mapped internally in a graph paradigm as nodes and edges. The epicontacts data structure also includes a logical attribute for whether or not this resulting network is directed.

The package takes advantage of R’s generic functions, which call specific methods depending on the class of an object. This is implemented in several places, including the summary.epicontacts() and print.epicontacts() methods, both of which are respectively called when the summary() or print() functions are used on an epicontacts object. The package does not include built-in data, as exemplary contact and line list datasets are available in the outbreaks package\(^8\).

```
# install the outbreaks package for data
install.packages("outbreaks")

# load the outbreaks package
library(outbreaks)

# construct an epicontacts object
x <- make_epiconctacts(linelist=mers_korea_2015[[1]],
                       contacts = mers_korea_2015[[2]],
                       directed=TRUE)

# print the object
x
```
### Epidemiological Contacts

// class: epicontacts
// 162 cases in linelist; 98 contacts; directed

// linelist

# A tibble: 162 x 15
## # id age age_class sex place_infect reporting_ctry loc_hosp
## # * <chr> <int> <chr>     <fct>  <fct>         <fct>          <fct>
## 1 SK_1 68 60-69     M      Middle East South Korea Pyeongtaek St-
## 2 SK_2 63 60-69     F      Outside Midd'- South Korea Pyeongtaek St-
## 3 SK_3 76 70-79     M      Outside Midd'- South Korea Pyeongtaek St-
## 4 SK_4 46 40-49     F      Outside Midd'- South Korea Pyeongtaek St-
## 5 SK_5 50 50-59     M      Outside Midd'- South Korea 365 Yeollin C-
## 6 SK_6 71 70-79     M      Outside Midd'- South Korea Pyeongtaek St-
## 7 SK_7 28 20-29     F      Outside Midd'- South Korea Pyeongtaek St-
## 8 SK_8 46 40-49     F      Outside Midd'- South Korea Seoul Clinic,-
## 9 SK_9 56 50-59     M      Outside Midd'- South Korea Pyeongtaek St-
## 10 SK_10 44 40-49   M      Outside Midd'- China Pyeongtaek St-
## ... with 152 more rows, and 8 more variables: dt_onset <date>,  dt_report
## # <date>, week_report <fct>, dt_start_exp <date>, dt_end_exp  <date>,
## # dt_diag <date>, outcome <fct>, dt_death <date>

// contacts

# A tibble: 98 x 4
## # from to exposure diff_dt_onset
## # <chr> <chr>  <fct>                 <int>
## 1 SK_14 SK_113 Emergency room           10
## 2 SK_14 SK_116 Emergency room           13
## 3 SK_14 SK_41  Emergency room           14
## 4 SK_14 SK_112 Emergency room           14
## 5 SK_14 SK_100 Emergency room           15
## 6 SK_14 SK_114 Emergency room           15
## 7 SK_14 SK_136 Emergency room           15
## 8 SK_14 SK_47  Emergency room           16
## 9 SK_14 SK_110 Emergency room           16
## 10 SK_14 SK_122 Emergency room           16
## ... with 88 more rows

# view a summary of the object
summary(x)

### Overview

// number of unique IDs in linelist: 162
// number of unique IDs in contacts: 97
// number of unique IDs in both: 97
// number of contacts: 98
// contacts with both cases in linelist: 100%

### Degrees of the network

// in-degree summary:
## Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
## 0.00 1.00 1.00  1.01 1.00 3.00

// out-degree summary:
## Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
## 0.00 0.00 0.00  1.01 0.00 38.00

// in and out degree summary:
## Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
## 1.000 1.000 1.000 2.021 1.000 39.000

### Attributes

// attributes in linelist:
## age age_class sex place_infect reporting_ctry loc_hosp dt_onset dt_report week_report
dt_start_exp dt_end_exp dt_diag outcome dt_death
## // attributes in contacts:
## exposure diff_dt_onset

Data visualisation. epicontacts implements two interactive network visualisation packages: visNetwork and threejs\textsuperscript{17,18}. These frameworks provide R interfaces to the vis.js and three.js JavaScript libraries respectively. Their functionality is incorporated in the generic plot() method (Figure 1) for an epicontacts object, which can be toggled between either with the “type” parameter. Alternatively, the visNetwork interactivity is accessible via vis_epicontacts() (Figure 2), and threejs through graph3D() (Figure 3). Each function has a series of arguments that can also be passed through plot(). Both share a color palette, and users can specify node, edge and background colors. However, vis_epicontacts() includes a specification for “node_shape” by a line list attribute as well as a customization of that shape with an icon from the Font Awesome icon library. The principal distinction between the two is that graph3D() is a three-dimensional visualisation, allowing users to rotate clusters of nodes to better inspect their relationships.

```r
plot(x)

vis_epicontacts(x,
    node_shape = "sex",
    shapes = c(F = "female", M = "male"),
    edge_label = "exposure")

graph3D(x, bg_col = "black")
```

Figure 1. The generic plot() method for an epicontacts object will use the visNetwork method by default.
Figure 2. The vis_epicontacts() function explicitly calls visNetwork to make an interactive plot of the contact network.

Figure 3. The graph3D() function generates a three-dimensional network plot.
**Data analysis.** Subsetting is a typical preliminary step in data analysis. epicontacts leverages a customized subset method to filter line list or contacts based on values of particular attributes from nodes, edges or both. If users are interested in returning only contacts that appear in the line list (or vice versa), the thin() function implements such logic.

```r
# subset for males
subset(x, node_attribute = list("sex" = "M"))

# subset for exposure in emergency room
subset(x, edge_attribute = list("exposure" = "Emergency room"))

# subset for males who survived and were exposed in emergency room
subset(x,
        node_attribute = list("sex" = "M", "outcome" = "Alive"),
        edge_attribute = list("exposure" = "Emergency room"))

thin(x, "contacts")
thin(x, "linelist")
```

For analysis of pairwise contact between individuals, the get_pairwise() feature searches the line list based on the specified attribute. If the given column is a numeric or date object, the function will return a vector containing the difference of the values of the corresponding “from” and “to” contacts. This can be particularly useful, for example, if the line list includes the date of onset of each case. The subtracted value of the contacts would approximate the serial interval for the outbreak\(^9\). For factors, character vectors and other non-numeric attributes, the default behavior is to print the associated line list attribute for each pair of contacts. The function includes a further parameter to pass an arbitrary function to process the specified attributes. In the case of a character vector, this can be helpful for tabulating information about different contact pairings with table().

```r
# find interval between date onset in cases
get_pairwise(x, "dt_onset")

# find pairs of age category contacts
get_pairwise(x, "age_class")

# tabulate the pairs of age category contacts
get_pairwise(x, "age_class", f = table)
```

**Use cases**

Those interested in using epicontacts should have a line list of cases as well as a record of contacts between individuals. Both datasets must be enumerated in tabular format with rows and columns. At minimum the line list requires one column with a unique identifier for every case. The contact list needs two columns for the source and destination of each pair of contacts. The datasets can include arbitrary features of case or contact beyond these columns. Once loaded into R and stored as data.frame objects, these datasets can be passed to the make_epicontacts() function (see ‘Methods’ section for more detail). For an example of data prepared in this format, users can refer to the outbreaks R package.

```r
# load the outbreaks package
library(outbreaks)

# example simulated ebola data
# line list
str(ebola_sim$linelist)
```

```
' data.frame': 5888 obs. of 9 variables:
$ case_id     : chr "d1f6d" "53371b" "f5c3d8" "6c286a" ...
$ generation  : int 0 1 1 2 2 0 3 3 2 3 ...
$ date_of_infection: Date, format: NA "2014-04-09" ...
$ date_of_onset : Date, format: "2014-04-07" "2014-04-15" ...
$ date_of_hospitalisation: Date, format: "2014-04-17" "2014-04-20" ...
$ date_of_outcome: Date, format: "2014-04-19" NA ...
$ gender      : Factor w/ 2 levels "f","m": 1 2 1 1 1 1 1 1 2 2 ...
$ hospital    : Factor w/ 11 levels "Connaught Hospital", ...
```

Page 7 of 11
Discussion

Benefits

While there are software packages available for epidemiological contact visualisation and analysis, none aim to accommodate line list and contact data as purposively as epicontacts\(^2\)\(^-\)\(^2\(^2\). Furthermore, this package strives to solve a problem of plotting dense graphs by implementing interactive network visualisation tools. A static plot of a network with many nodes and edges may be difficult to interpret. However, by rotating or hovering over an epicontacts visualisation, a user may better understand the data.

Future considerations

The maintainers of epicontacts anticipate new features and functionality. Future development could involve performance optimization for visualising large networks, as generating these interactive plots is resource intensive. Additionally, attention may be directed towards inclusion of alternative visualisation methods.

Conclusions

epicontacts provides a unified interface for processing, visualising and analyzing disease outbreak data in the R language. The package and its source are freely available on CRAN and GitHub. By developing functionality with line list and contact list data in mind, the authors aim to enable more efficient epidemiological outbreak analyses.

Software availability

Software available from: [https://CRAN.R-project.org/package=epicontacts](https://CRAN.R-project.org/package=epicontacts)

Source code available from: [https://github.com/reconhub/epicontacts](https://github.com/reconhub/epicontacts)

Archived source code as at time of publication: [https://zenodo.org/record/1210993](https://zenodo.org/record/1210993)

Software license: GPL 2
Competing interests
No competing interests were disclosed.

Grant information
The author(s) declared that no grants were involved in supporting this work.

Acknowledgements
The authors would like to thank all of the organizers and participants of the Hackout3 event held in Berkeley, California June 20–24, 2016. In particular, the authors acknowledge the support of the following organizations: MRC Centre for Outbreak Analysis, and Modelling at Imperial College London, the NIHR’s Modelling Methodology Health Protection Research Unit at Imperial College London, and the Berkeley Institute for Data Science.

References

Open Peer Review

Current Referee Status:  

Version 1

Referee Report 31 May 2018

doi:10.5256/f1000research.15777.r34084

Melissa A. Rolfes
Centers for Disease Control and Prevention (CDC) , Atlanta, GA, USA

The article describes an R-based software tool aimed to facilitate analysis of data from outbreaks that include line lists of cases and case-contact data. The R package, epicontacts, is part of a larger suite of tools housed at the R Epidemics Consortium (RECON). The epicontacts package has the ability to merge data about cases in a line list with case-contact details, which then allows the user to describe and visualize contact networks, incubation periods, and serial intervals within an outbreak.

The codes and methods for analysis are partly described in the article, and the authors should provide a link to the packages documentation, either at CRAN or RECON webpages, where readers could learn more about the package and its options.

The output of the package provided in the article was interesting and intriguing. I felt that it was only partly explained and the article could benefit from the authors annotating the output and its interpretation a bit further. I have explored the RECON website and found the RECON Learn modules to be quite helpful in providing annotation of the epicontacts output and some guidance on interpretation. I would recommend that the authors consider either expanding the annotation of the output in this article or explicitly direct readers to the RECON Learn website for further instruction.

Additional suggestions:

- Consider moving the section of the article called "Use cases" to before the "Data handling" subsection of the "Implementation" section. I felt that the description of the input datasets under "Use cases" was very informative and would have been organizational more helpful had it been placed earlier in the article.

- Consider describing the sample outbreak data in a bit further detail. It appears to be data describing the MERS outbreak that occurred in South Korea in 2015. I think the description should include whether the data are simulated or from a real outbreak (if from a real outbreak, then a reference to the outbreak description should be included), the scenario of the outbreak, how many cases, how many contacts, place of the outbreak, duration of the outbreak, and a brief description of the demographic details included in the dataset. This amount of detail would allow the reader to translate the details of the outbreak from your text to the output provided by epicontacts.

Is the rationale for developing the new software tool clearly explained?
Yes
Is the description of the software tool technically sound?
Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Partly

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Partly

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?
Yes

**Competing Interests:** No competing interests were disclosed.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

---

**The benefits of publishing with F1000Research:**

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com