Disclaimer: This vignette reflects package state at version 0.9-7 and is hence somewhat outdated. New functionality has been added to the package: this includes various endemic-epidemic modelling frameworks for surveillance data (`hhh4`, `twinSIR`, and `twinstim`), as well as more outbreak detection methods (`glrnb`, `boda`, and `farringtonFlexible`). These new features are described in detail in Meyer et al. (2017) and Salmon et al. (2016), respectively. Note in particular that use of the new `S4` class `sts` instead of `disProg` is encouraged to encapsulate time series data.

Getting started with outbreak detection

Michael Höhle*, Andrea Riebler and Michaela Paul
Department of Statistics
University of Munich
Germany

17 November 2007

Abstract

This document gives an introduction to the R package **surveillance** containing tools for outbreak detection in routinely collected surveillance data. The package contains an implementation of the procedures described by Stroup et al. (1989), Farrington et al. (1996) and the system used at the Robert Koch Institute, Germany. For evaluation purposes, the package contains example data sets and functionality to generate surveillance data by simulation. To compare the algorithms, benchmark numbers like sensitivity, specificity, and detection delay can be computed for a set of time series. Being an open-source package it should be easy to integrate new algorithms; as an example of this process, a simple Bayesian surveillance algorithm is described, implemented and evaluated.
Keywords: infectious disease, monitoring, aberrations, outbreak, time series of counts.

*Author of correspondence: Department of Statistics, University of Munich, Ludwigstr. 33, 80539 München, Germany, Email: `hoehle@stat.uni-muenchen.de`

1 Introduction

Public health authorities have in an attempt to meet the threats of infectious diseases to society created comprehensive mechanisms for the collection of disease data. As a consequence, the abundance of data has demanded the development of automated algorithms for the detection of abnormalities. Typically, such an algorithm monitors a univariate time series of counts using a combination of heuristic methods and statistical modelling. Prominent examples of surveillance algorithms are the work by Stroup et al. (1989) and Farrington et al. (1996). A comprehensive survey of outbreak detection methods can be found in (Farrington and Andrews, 2003).

The R-package `surveillance` was written with the aim of providing a test-bench for surveillance algorithms. From the Comprehensive R Archive Network (CRAN) the package can be downloaded together with its source code. It allows users to test new algorithms and compare their results with those of standard surveillance methods. A few real world outbreak datasets are included together with mechanisms for simulating surveillance data. With the package at hand, comparisons like the one described by Hutwagner et al. (2005) should be easy to conduct.

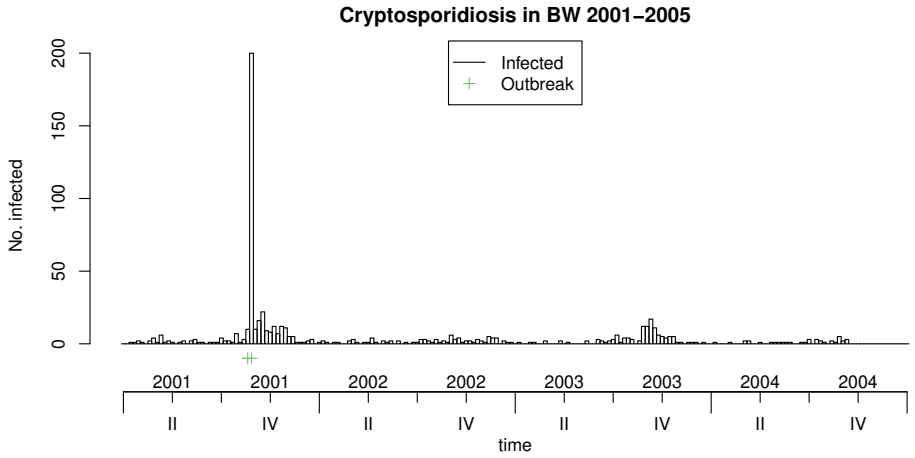
The purpose of this document is to illustrate the basic functionality of the package with R-code examples. Section 2 contains a description of the data format used to store surveillance data, mentions the built-in datasets and illustrates how to create new datasets by simulation. Section 3 contains a short description of how to use the surveillance algorithms and illustrate the results. Further information on the individual functions can be found on the corresponding help pages of the package.

2 Surveillance Data

Denote by $\{y_t; t = 1, \ldots, n\}$ the time series of counts representing the surveillance data. Because such data typically are collected on a weekly basis, we shall also use the alternative notation $\{y_{i:j}\}$ with $j = \{1, \ldots, 52\}$ being the week number in year $i = \{-b, \ldots, -1, 0\}$. That way the years are indexed such that most current year has index zero. For evaluation of the outbreak detection algorithms it is also possible for each week to store – if known – whether there was an outbreak that week. The resulting multivariate series $\{(y_t, x_t)\,; t = 1, \ldots, n\}$ is in `surveillance` given by an object of class `disProg` (disease progress), which is basically a `list` containing two vectors: the observed number of counts and a boolean vector `state` indicating whether there was an outbreak that week. A number of time series are contained in the package (see `data(package="surveillance")`), mainly originating from the SurvStat@RKI database at `https://survstat.rki.de/` maintained by the Robert Koch Institute, Germany (Robert Koch-Institut,

2004). For example the object `k1` describes cryptosporidiosis surveillance data for the German federal state Baden-Württemberg 2001-2005. The peak in 2001 is due to an outbreak of cryptosporidiosis among a group of army soldiers in a boot camp (Robert Koch-Institut, 2001).

```
> data(k1)
> plot(k1, main = "Cryptosporidiosis in BW 2001-2005")
```



For evaluation purposes it is also of interest to generate surveillance data using simulation. The package contains functionality to generate surveillance data containing point-source like outbreaks, for example with a Salmonella serovar. The model is a Hidden Markov Model (HMM) where a binary state $X_t, t = 1, \ldots, n$, denotes whether there was an outbreak and Y_t is the number of observed counts, see Figure 1.

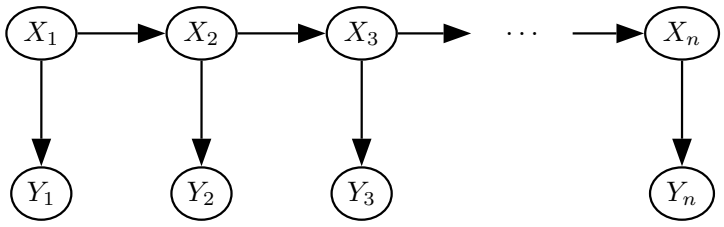


Figure 1: The Hidden Markov Model

The state X_t is a homogeneous Markov chain with transition matrix

$X_t \backslash X_{t+1}$	0	1
0	p	$1-p$
1	$1-r$	r

Hence $1-p$ is the probability to switch to an outbreak state and $1-r$ is the probability that $X_t = 1$ is followed by $X_{t+1} = 1$. Furthermore, the

observation Y_t is Poisson-distributed with log-link mean depending on a seasonal effect and time trend, i.e.

$$\log \mu_t = A \cdot \sin(\omega \cdot (t + \varphi)) + \alpha + \beta t.$$

In case of an outbreak ($X_t = 1$) the mean increases with a value of K, altogether

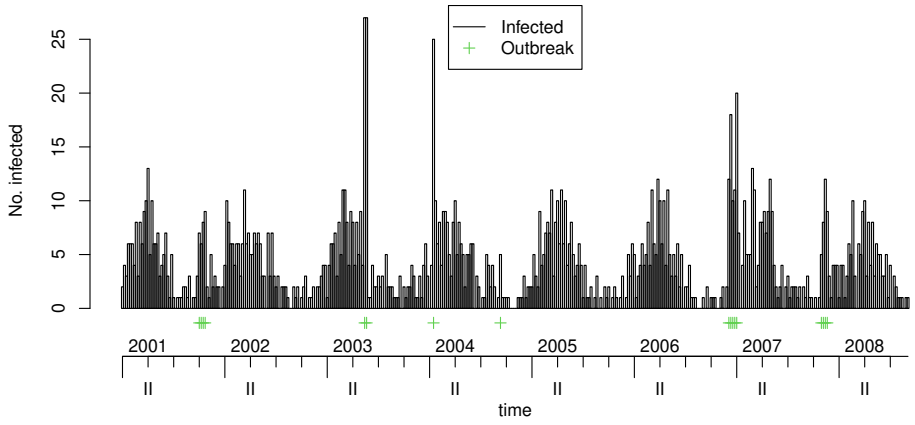
$$Y_t \sim \text{Po}(\mu_t + K \cdot X_t). \tag{1}$$

The model in (1) corresponds to a single-source, common-vehicle outbreak, where the length of an outbreak is controlled by the transition probability r. The daily numbers of outbreak-cases are simply independently Poisson distributed with mean K. A physiologically better motivated alternative could be to operate with a stochastic incubation time (e.g. log-normal or gamma distributed) for each individual exposed to the source, which results in a temporal diffusion of the peak. The advantage of (1) is that estimation can be done by a generalized linear model (GLM) using X_t as covariate and that it allows for an easy definition of a correctly identified outbreak: each $X_t = 1$ has to be identified. More advanced setups would require more involved definitions of an outbreak, e.g. as a connected series of time instances, where the number of outbreak cases is greater than zero. Care is then required in defining what a correctly identified outbreak for time-wise overlapping outbreaks means.

In **surveillance** the function `sim.pointSource` is used to simulate such a point-source epidemic; the result is an object of class `disProg`.

```
> set.seed(1234)
> sts <- sim.pointSource(p = 0.99, r = 0.5, length = 400,
+                        A = 1, alpha = 1, beta = 0, phi = 0,
+                        frequency = 1, state = NULL, K = 1.7)

> plot(sts)
```



3 Surveillance Algorithms

Surveillance data often exhibit strong seasonality, therefore most surveillance algorithms only use a set of so called *reference values* as basis for drawing conclusions. Let $y_{0:t}$ be the number of cases of the current week (denoted week t in year 0), b the number of years to go back in time and w the number of weeks around t to include from those previous years. For the year zero we use w_0 as the number of previous weeks to include – typically $w_0 = w$. Altogether the set of reference values is thus defined to be

$$R(w, w_0, b) = \left(\bigcup_{i=1}^{b} \bigcup_{j=-w}^{w} y_{-i:t+j} \right) \cup \left(\bigcup_{k=-w_0}^{-1} y_{0:t+k} \right)$$

Note that the number of cases of the current week is not part of $R(w, w_0, b)$.

A surveillance algorithm is a procedure using the reference values to create a prediction $\hat{y}_{0:t}$ for the current week. This prediction is then compared with the observed $y_{0:t}$: if the observed number of cases is much higher than the predicted number, the current week is flagged for further investigations. In order to do surveillance for time $0 : t$ an important concern is the choice of b and w. Values as far back as time $-b : t - w$ contribute to $R(w, w_0, b)$ and thus have to exist in the observed time series.

Currently, we have implemented four different type of algorithms in **surveillance**. The Centers for Disease Control and Prevention (CDC) method (Stroup et al., 1989), the Communicable Disease Surveillance Centre (CDSC) method (Farrington et al., 1996), the method used at the Robert Koch Institute (RKI), Germany (Altmann, 2003), and a Bayesian approach documented in Riebler (2004). A detailed description of each method is beyond the scope of this note, but to give an idea of the framework the Bayesian approach developed in Riebler (2004) is presented: Within a Bayesian framework, quantiles of the predictive posterior distribution are used as a measure for defining alarm thresholds.

The model assumes that the reference values are identically and independently Poisson distributed with parameter λ and a Gamma-distribution is used as Prior distribution for λ. The reference values are defined to be $R_{\text{Bayes}} = R(w, w_0, b) = \{y_1, \ldots, y_n\}$ and $y_{0:t}$ is the value we are trying to predict. Thus, $\lambda \sim \text{Ga}(\alpha, \beta)$ and $y_i|\lambda \sim \text{Po}(\lambda)$, $i = 1, \ldots, n$. Standard derivations show that the posterior distribution is

$$\lambda|y_1, \ldots, y_n \sim \text{Ga}(\alpha + \sum_{i=1}^{n} y_i, \beta + n).$$

Computing the predictive distribution

$$f(y_{0:t}|y_1, \ldots, y_n) = \int_0^\infty f(y_{0:t}|\lambda)\, f(\lambda|y_1, \ldots, y_n)\, d\lambda$$

we get the Poisson-Gamma-distribution

$$y_{0:t}|y_1,\ldots,y_n \sim \text{PoGa}(\alpha+\sum_{i=1}^{n} y_i, \beta+n),$$

which is a generalization of the negative Binomial distribution, i.e.

$$y_{0:t}|y_1,\ldots,y_n \sim \text{NegBin}(\alpha+\sum_{i=1}^{n} y_i, \tfrac{\beta+n}{\beta+n+1}).$$

Using the Jeffrey's Prior $\text{Ga}(\frac{1}{2},0)$ as non-informative Prior distribution for λ the parameters of the negative Binomial distribution are

$$\alpha+\sum_{i=1}^{n} y_i = \frac{1}{2}+\sum_{y_{i:j}\in R_{\text{Bayes}}} y_{i:j} \qquad \text{and} \qquad \frac{\beta+n}{\beta+n+1} = \frac{|R_{\text{Bayes}}|}{|R_{\text{Bayes}}|+1}.$$

Using a quantile-parameter α, the smallest value y_α is computed, so that

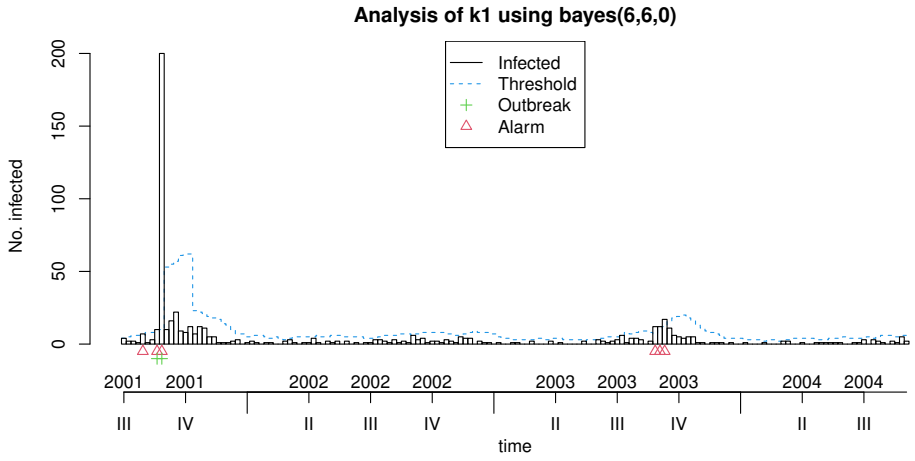
$$P(y \le y_\alpha) \ge 1-\alpha.$$

Now

$$A_{0:t} = I(y_{0:t} \ge y_\alpha),$$

i.e. if $y_{0:t} \ge y_\alpha$ the current week is flagged as an alarm. As an example, the `Bayes1` method uses the last six weeks as reference values, i.e. $R(w, w_0, b) = (6, 6, 0)$, and is applied to the `k1` dataset with $\alpha = 0.01$ as follows.

```
> k1.b660 <- algo.bayes(k1,
+   control = list(range = 27:192, b = 0, w = 6, alpha = 0.01))
> plot(k1.b660, disease = "k1")
```



Several extensions of this simple Bayesian approach are imaginable, for example the inane over-dispersion of the data could be modeled by using

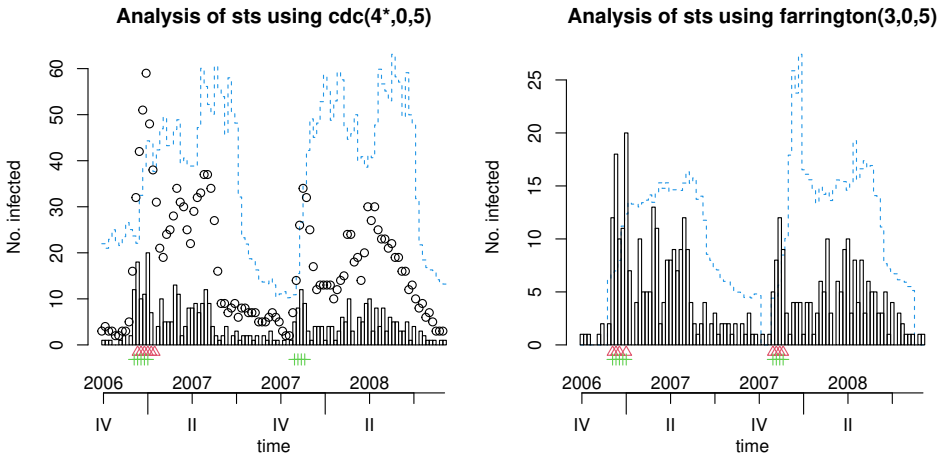
a negative-binomial distribution, time trends and mechanisms to correct for past outbreaks could be integrated, but all at the cost of non-standard inference for the predictive distribution. Here simulation based methods like Markov Chain Monte Carlo or heuristic approximations have to be used to obtain the required alarm thresholds.

In general, the `surveillance` package makes it easy to add additional algorithms – also those not based on reference values – by using the existing implementations as starting point.

The following call uses the CDC and Farrington procedure on the simulated time series `sts` from page 4. Note that the CDC procedure operates with four-week aggregated data – to better compare the upper bound value, the aggregated number of counts for each week are shown as circles in the plot.

```
> cntrl <- list(range=300:400,m=1,w=3,b=5,alpha=0.01)
> sts.cdc  <- algo.cdc(sts, control = cntrl)
> sts.farrington <- algo.farrington(sts, control = cntrl)

> par(mfcol=c(1,2))
> plot(sts.cdc, legend.opts=NULL)
> plot(sts.farrington, legend.opts=NULL)
```



Typically, one is interested in evaluating the performance of the various surveillance algorithms. An easy way is to look at the sensitivity and specificity of the procedure – a correct identification of an outbreak is defined as follows: if the algorithm raises an alarm for time t, i.e. $A_t = 1$ and $X_t = 1$ we have a correct classification, if $A_t = 1$ and $X_t = 0$ we have a false-positive, etc. In case of more involved outbreak models, where an outbreak lasts for more than one week, a correct identification could be if at least one of the outbreak weeks is correctly identified, see e.g. Hutwagner et al. (2005).

To compute various performance scores the function `algo.quality` can be used on a `survRes` object.

```
> print(algo.quality(k1.b660))

     TP FP TN  FN Sens Spec      dist       mlag
[1,] 2  4  160 0  1    0.9756098 0.02439024 0
```

This computes the number of false positives, true negatives, false negatives, the sensitivity and the specificity. Furthermore, `dist` is defined as

$$\sqrt{(Spec-1)^2+(Sens-1)^2},$$

that is the distance to the optimal point $(1,1)$, which serves as a heuristic way of combining sensitivity and specificity into a single score. Of course, weighted versions are also imaginable. Finally, `lag` is the average number of weeks between the first of a consecutive number of $X_t = 1$'s (i.e. an outbreak) and the first alarm raised by the algorithm.

To compare the results of several algorithms on a single time series we declare a list of control objects – each containing the name and settings of the algorithm we want to apply to the data.

```
> control <- list(
+   list(funcName = "rki1"), list(funcName = "rki2"),
+   list(funcName = "rki3"), list(funcName = "bayes1"),
+   list(funcName = "bayes2"), list(funcName = "bayes3"),
+   list(funcName = "cdc", alpha=0.05),
+   list(funcName = "farrington", alpha=0.05)
+ )
> control <- lapply(control, function(ctrl) {
+   ctrl$range <- 300:400; return(ctrl)
+ })
```

In the above, `rki1`, `rki2` and `rki3` are three methods with reference values $R_{\text{rki1}}(6,6,0)$, $R_{\text{rki2}}(6,6,1)$ and $R_{\text{rki3}}(4,0,2)$, all called with $\alpha = 0.05$. The `bayes*` methods use the Bayesian algorithm with the same setup of reference values. The CDC method is special since it operates on aggregated four-week blocks. To make everything comparable, a common $\alpha = 0.05$ level is used for all algorithms. All algorithms in `control` are applied to `sts` using:

```
> algo.compare(algo.call(sts, control = control))

                  TP FP TN FN sens  spec  dist   mlag
rki(6,6,0)        6  2  90 3  0.667 0.978 0.334  0
rki(6,6,1)        7  1  91 2  0.778 0.989 0.222  0.5
rki(4,0,2)        8  2  90 1  0.889 0.978 0.113  0.5
bayes(6,6,0)      6  2  90 3  0.667 0.978 0.334  0
bayes(6,6,1)      7  2  90 2  0.778 0.978 0.223  0.5
bayes(4,0,2)      9  2  90 0  1     0.978 0.0217 0
cdc(4*,0,5)       7  2  90 2  0.778 0.978 0.223  1
farrington(3,0,5) 9  1  91 0  1     0.989 0.0109 0
```

A test on a set of time series can be done as follows. Firstly, a list containing 10 simulated time series is created. Secondly, all the algorithms specified in the `control` object are applied to each series. Finally the results for the 10 series are combined in one result matrix.

```
> #Create 10 series
> ten <- lapply(1:10,function(x) {
+   sim.pointSource(p = 0.975, r = 0.5, length = 400,
+                   A = 1, alpha = 1, beta = 0, phi = 0,
+                   frequency = 1, state = NULL, K = 1.7)})

> #Do surveillance on all 10, get results as list
> ten.surv <- lapply(ten,function(ts) {
+   algo.compare(algo.call(ts,control=control))
+ })

> #Average results
> algo.summary(ten.surv)

                  TP FP  TN FN  sens  spec   dist mlag
rki(6,6,0)        31 22 945 12 0.721 0.977 0.2800 1.35
rki(6,6,1)        34  8 959  9 0.791 0.992 0.2095 1.35
rki(4,0,2)        37  6 961  6 0.860 0.994 0.1397 1.35
bayes(6,6,0)      31 43 924 12 0.721 0.956 0.2826 1.35
bayes(6,6,1)      36 19 948  7 0.837 0.980 0.1640 1.35
bayes(4,0,2)      39 20 947  4 0.907 0.979 0.0953 1.33
cdc(4*,0,5)       21 37 930 22 0.488 0.962 0.5131 8.80
farrington(3,0,5) 36 16 951  7 0.837 0.983 0.1636 1.73
```

A similar procedure can be applied when evaluating the 14 surveillance series drawn from SurvStat@RKI (Robert Koch-Institut, 2004). A problem is however, that the series after conversion to 52 weeks/year are of length 209 weeks. This is insufficient to apply e.g. the CDC algorithm. To conduct the comparison on as large a dataset as possible the following trick is used: The function `enlargeData` replicates the requested `range` and inserts it before the original data, after which the evaluation can be done on all 209 values.

```
> #Update range in each - cyclic continuation
> range = (2*4*52) +  1:length(k1$observed)
> control <- lapply(control,function(cntrl) {
+   cntrl$range=range;return(cntrl)})
> #Auxiliary function to enlarge data
> enlargeData <- function(disProgObj, range = 1:156, times = 1){
+   disProgObj$observed <- c(rep(disProgObj$observed[range], times),
+                            disProgObj$observed)
```

```
+   disProgObj$state <- c(rep(disProgObj$state[range], times),
+                         disProgObj$state)
+   return(disProgObj)
+ }
> #Outbreaks
> outbrks <- c("m1", "m2", "m3", "m4", "m5", "q1_nrwh", "q2",
+              "s1", "s2", "s3", "k1", "n1", "n2", "h1_nrwrp")
> #Load and enlarge data.
> outbrks <- lapply(outbrks,function(name) {
+   data(list=name)
+   enlargeData(get(name),range=1:(4*52),times=2)
+ })
> #Apply function to one
> one.survstat.surv <- function(outbrk) {
+   algo.compare(algo.call(outbrk,control=control))
+ }

> algo.summary(lapply(outbrks,one.survstat.surv))

                   TP  FP   TN  FN  sens  spec  dist mlag
rki(6,6,0)         38  62 2646 180 0.174 0.977 0.826 5.43
rki(6,6,1)         65  83 2625 153 0.298 0.969 0.703 5.57
rki(4,0,2)         80 106 2602 138 0.367 0.961 0.634 5.43
bayes(6,6,0)       46 101 2607 172 0.211 0.963 0.790 4.07
bayes(6,6,1)       84 130 2578 134 0.385 0.952 0.617 2.21
bayes(4,0,2)      117 200 2508 101 0.537 0.926 0.469 1.93
cdc(4*,0,5)        65  94 2614 153 0.298 0.965 0.703 7.14
farrington(3,0,5)  43  71 2637 175 0.197 0.974 0.803 4.79
```

In both this study and the earlier simulation study the Bayesian approach seems to do quite well. However, the extent of the comparisons do not make allowance for any more supported statements. Consult the work of Riebler (2004) for a more thorough comparison using simulation studies.

4 Discussion and Future Work

Many extensions and additions are imaginable to improve the package. For now, the package is intended as an academic tool providing a test-bench for integrating new surveillance algorithms. Because all algorithms are implemented in R, performance has not been an issue. Especially the current implementation of the Farrington Procedure is rather slow and would benefit from an optimization possible with fragments written in C.

One important improvement would be to provide more involved mechanisms for the simulation of epidemics. In particular it would be interesting to include multi-day outbreaks originating from single-source exposure,

but with delay due to varying incubation time (Hutwagner et al., 2005) or SEIR-like epidemics (Andersson and Britton, 2000). However, defining what is meant by a correct outbreak identification, especially in the case of overlapping outbreaks, creates new challenges which have to be met.

5 Acknowledgements

We are grateful to K. Stark and D. Altmann, RKI, Germany, for discussions and information on the surveillance methods used by the RKI. Our thanks to C. Lang, University of Munich, for his work on the R–implementation and M. Kobl, T. Schuster and M. Rossman, University of Munich, for their initial work on gathering the outbreak data from SurvStat@RKI. The research was conducted with financial support from the Collaborative Research Centre SFB 386 funded by the German research foundation (DFG).

References

Altmann, D. (2003). The surveillance system of the Robert Koch Institute, Germany. Personal communication.

Andersson, H. and Britton, T. (2000). *Stochastic Epidemic Models and their Statistical Analysis*, volume 151 of *Springer Lectures Notes in Statistics*. Springer-Verlag.

Farrington, C. P., Andrews, N. J., Beale, A. D., and Catchpole, M. A. (1996). A statistical algorithm for the early detection of outbreaks of infectious disease. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 159:547–563.

Farrington, P. and Andrews, N. (2003). Outbreak detection: Application to infectious disease surveillance. In Brookmeyer, R. and Stroup, D. F., editors, *Monitoring the Health of Populations*, chapter 8, pages 203–231. Oxford University Press.

Hutwagner, L., Browne, T., Seeman, G., and Fleischhauer, A. (2005). Comparing aberration detection methods with simulated data. *Emerging Infectious Diseases*, 11:314–316.

Meyer, S., Held, L., and Höhle, M. (2017). Spatio-temporal analysis of epidemic phenomena using the R package `surveillance`. *Journal of Statistical Software*, 77(11):1–55.

Riebler, A. (2004). Empirischer Vergleich von statistischen Methoden zur Ausbruchserkennung bei Surveillance Daten. Bachelor's thesis, Department of Statistics, University of Munich.

Robert Koch-Institut (2001). Gruppenerkrankung in Baden-Württemberg: Verdacht auf Kryptosporidiose. *Epidemiologisches Bulletin*, 39:298–299.

Robert Koch-Institut (2004). SurvStat@RKI. `https://survstat.rki.de/`. Date of query: September 2004.

Salmon, M., Schumacher, D., and Höhle, M. (2016). Monitoring count time series in R: Aberration detection in public health surveillance. *Journal of Statistical Software*, 70(10):1–35.

Stroup, D., Williamson, G., Herndon, J., and Karon, J. (1989). Detection of aberrations in the occurrence of notifiable diseases surveillance data. *Statistics in Medicine*, 8:323–329.