Vignette for **SANTA**

Alex Cornish and Florian Markovetz

April 10, 2013

Contents

1 Introduction

This document contains instructions on how to employ the functions contained within the **SANTA** package. **SANTA** builds upon Ripley's K-function [1], a well-established approach in spatial statistics that measures the clustering of points on a plane, by applying it to networks. Using this approach, the $\texttt{K}^{\texttt{net}}$ and $\texttt{K}^{\texttt{node}}$ functions are defined. $\texttt{K}^{\texttt{net}}$ detects the clustering of hits across a network in order to measure the strength of association between the network and a phenotype. $\texttt{K}^{\texttt{node}}$ ranks genes according their distance from the hits, providing a natural way of prioritising vertices for follow-up analyses.

The **SANTA** package contains functions that can be used to measure the distance between vertex pairs (`GraphDiffusion` and `GraphMFPT`), a function to create graphs according to set parameters (`CreateGraph`) and a function that can be used to visualise the $\texttt{K}^{\texttt{net}}$ function (`plot.Knet`).

SANTA uses the `igraph` package to handle the networks.

2 Overview of SANTA

2.1 Guilt-by-association

The guilt-by-association (GBA) principle states that genes with the same interaction partners are more likely to share biological function. For example, two genes seen to interact in a synthetic genetic array (SGA) are more likely to encode products involved in the same pathway or complex than two genes not seen to interact. $\texttt{K}^{\texttt{net}}$ uses this principle to measure the strength of association between a network and a phenotype. By measuring the clustering of the gene set across biological networks it becomes possible to identify the network that best explains the mechanisms and processes that produce the set.

This principle is also used by $\texttt{K}^{\texttt{node}}$ to prioritise genes for follow up studies. If a set of genes has been identified as being associated with a particular cellular function, then the GBA principle states that genes seen interacting with this set are also likely to be involved in the function.

SANTA addresses two complementary goals: it can (i) functionally annotate experimentally derived networks using curated gene sets, and (ii) annotate experimentally derived gene sets using curated networks. To exemplify the first goal, we show that our approach helps to elucidate the functional content of the *S. cerevisiae* genetic interaction network and its rewiring in DNA damage response (Section 3).

2.2 Ripley's K-statistic on networks

Ripley's K statistic is a tool used to analyse the spatial distribution of points on a plane. It is typically applied to two- or higher-dimensional spaces. In two dimensions, the function works by counting the number of points contained within circles of radius s, positioned around each point on the plane. As s is increased, a greater proportion of the points become contained within each circle. If the points on the plane are clustered, then the number of points contained within each circle will increase faster with s.

By taking the distances between vertex pairs in networks, it becomes possible to apply Ripley's K statistic and measure the clustering of high-weight vertices:

$$K^{net}(s) = \frac{2}{p^2} \sum_i p_i \sum_j (p_j - \bar{p})\ \mathbf{I}(d^g(i,j) \leq s) \tag{1}$$

where p_i is the weight of node i, $d^g(i,j)$ is the distance between node i and node j according to the distance method selected (see Section 2.5). $\mathbf{I}(d^g(i,j) \leq s)$ is an indicator function and equals 1 if $d^g(i,j) \leq s$ and 0 otherwise. $p = \sum_k p_k$ and $\bar{p} = \frac{p}{n_p}$, where n_p equals the number of vertices within the network.

Ripley's K-function has previously been applied to geographical networks (such as road networks) in order to identify the clustering of objects along these networks [2]. However, key differences between the previous implementation and the implementation of the K-function used in **SANTA** (such as the inclusion of continuous weights and the position of the weight on vertices rather than on edges) allows for the function to be applied to numerous biological networks.

2.3 Using $\mathtt{K}^{\mathtt{net}}$ to measure the clustering of high-weight vertices

SANTA applies spatial statistics and the GBA principle to networks in order to measure the strength of association between a network and a phenotype. It does this by quantifying the clustering of high-weight vertices within the network. High-weight vertices represent those genes that are part of a gene set or are most strongly linked with the phenotype. As previously explained, the GBA principle implies that the stronger the clustering, the better the network is at describing the mechanisms that produce and maintain the phenotype.

The AUK of a $\mathtt{K}^{\mathtt{net}}$ or $\mathtt{K}^{\mathtt{node}}$ curve can be computed by subtracting the area of the region between $K_i^{net}(s) = 0$ and negative regions of the curve from the region between $K_i^{net}(s) = 0$ and positive regions of the curve. If there is clustering of high-weight vertices on the network then the $\mathtt{K}^{\mathtt{net}}$ function will initially increase with s and the AUK will be high. Conversely, if there is no clustering of high-weight vertices then the function is unlikely to increase and AUK will equal a value closer to 0.

In some scenarios, weights may only be available for a subset of graph vertices. For example, in RNAi screens, some genes may be identified as being involved in maintaining a phenotype (and be given a high weight), some genes may be identified as not being involved (and be given a low weight or a weight of 0) and some genes may not be tested. Genes for which no data is available should be excluded from the $\mathtt{K}^{\mathtt{net}}$ function. However, the gene should remain in the graph to allow for the correct calculation of vertex pair distances. This is done by giving the vertex a weight of **NA**.

Because of the large number of possible graph structures, it is not possible to determine how significant an observed $\mathtt{K}^{\mathtt{net}}$ curve is by simply taking the AUK. In order to determine significance, the vertex weights on the graph are randomly permuted and the $\mathtt{K}^{\mathtt{net}}$ AUK calculated for each of these permutations. It is then possible to compare the observed AUK score to the distribution of permuted AUK scores and quantify the significance. The distribution of permuted AUK statistics is modelled as a normal distribution and a z-test performed to produce an empirical p-value for the observed AUK. The p-values produced represents the probability that the clustering

of vertex weights at least as extreme as what is observed occurs given that the vertex weights are distributed randomly across the network. The figures produced in Section 3 visualise this comparison of observed and permuted AUK scores.

2.4 Using $\texttt{K}^{\texttt{node}}$ to rank vertices by their strength of association with high-weight vertices

Taking the inner sum of the $\texttt{K}^{\texttt{net}}$ equation makes it possible to rank vertices by their distance from high-weight vertices. This inner sum is called the $\texttt{K}^{\texttt{node}}$ function:

$$K_i^{node}(s) = \frac{2}{p} \sum_j (p_j - \bar{p})\ \mathbf{I}(d^g(i,j) \leq s) \tag{2}$$

The $\texttt{K}^{\texttt{node}}$ function provides a natural way to prioritise genes for follow up studies. This function incorporates the network structure and each vertex weight in order to quantify the strength of association between an individual vertex and a distribution of vertex weights. For those vertices close to a large number of high-weight vertices, the $\texttt{K}^{\texttt{node}}$ function will initially increase with s, before returning to 0 as s reaches the diameter of the graph. Conversely, for those vertices positioned further away from the high-weight vertices, the $\texttt{K}^{\texttt{node}}$ function will remain closer to 0. By calculating the $\texttt{K}^{\texttt{node}}$ AUK of each vertex, it is possible to rank them according to their distance from the high-weight vertices. Vertices with high AUKs are closest to, and therefore most-strongly associated with, these high-weight vertices.

2.5 Measuring the distance between vertex pairs in a graph

2.5.1 Shortest path method

One of the simplest ways to measure the distance between vertex pairs in a graph is by taking the distance along the shortest path between the vertices. In a graph without weighted edges, the length of the shortest path is equal to the number of edges in the path. In a graph with weighted edges, the

length of the shortest path is equal to the sum of the edge weights in the path.

2.5.2 Diffusion kernel-based method

The diffusion kernel-based distance measure can be considered as the physical process of diffusion though the graph. Therefore, unlike the shortest paths distance measure, the diffusion kernel-based measure incorporates not only the distance of the shortest path connecting two vertices, but the distances across multiple paths.

The distance between each vertex pair is calculated using the negative graph Laplacian H, a square matrix of size $|V|$ with entries (in the unweighted and weighted case, respectively):

$$H_{ij}^{\text{unweighted}} = \begin{cases} 1 & \text{for } i \sim j \\ -k_i & \text{for } i = j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$H_{ij}^{\text{weighted}} = \begin{cases} w_{ij} & \text{for } i \sim j \\ -\sum_j w_{ij} & \text{for } i = j \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where k_i is the degree of node i (the number of edges associated with node i) and w_{ij} is the weight of the edge between nodes i and j. A diffusion kernel is then defined by the matrix exponential:

$$D := \exp(\beta H) = \lim_{n\to\infty} \left(1 + \frac{\beta H}{n}\right)^n = I + \beta H + \frac{1}{2!}\beta H = \frac{1}{3!}\beta H + \dots \tag{5}$$

To compute the matrix exponential we make use of the fact that H is diagonizable, i.e. $H = U\Delta U^{-1}$, where Δ is a diagonal matrix with entries $(\delta_i)_{i=1,\dots,n}$. It follows that $D = \exp(\beta H) = U\exp(\beta\Delta)U^{-1}$, where $\exp(\beta\Delta)$ simply is a diagonal matrix with entries $(\exp(\beta\delta_i))_{i=1,\dots,n}$.

2.5.3 Mean first-passage time-based method

The mean first-passage time distance measure computes the distance between vertex a and b ($m_{a,b}$) by calculating the expected amount of time that a ran-

dom walk emanating from node a takes to reach node b for the first time [3].

$$m_{a,b} = \sum_{n=1}^{\infty} n \; f_{a,b}^{(n)} \tag{6}$$

where n is the number of steps taken and $f_{a,b}^{(n)}$ is the probability that the random walk reaches node b for the first time after n steps. Because the mean first-passage time from vertex a to vertex b and the mean first-passage time from vertex b to vertex a may not be equal, a distance between the two nodes $d_{a,b}^{g}$ is derived by taking the mean.

3 Case study I - Using K$^{\texttt{net}}$ to investigate the re-wiring of a cell undergoing DNA damage

This section will demonstrate how the **SANTA** K$^{\texttt{net}}$ function can be used to measure the clustering of high-weight vertices on a network and thereby quantify the strength of association between a network and a phenotype.

Bandyopadhyay et al. mapped genetic interaction (GI) networks in yeast under normal laboratory conditions and in yeast exposed to the DNA-damaging agent methyl methanesulfonate (MMS) [4]. In order to investigate the differences between these functional networks, we will use K$^{\texttt{net}}$ to quantify the clustering of genes involved in responding to DNA damage. This will allow us to determine whether gene associated with this function cluster more strongly on one of the networks, thereby allowing us to better understand the changes that are occurring within the cell.

We will now load the **SANTA** package and two data frames containing the interactions within the networks. Lists of edges, along with associated scores, can be converted into **SANTA** -compatible graphs (`igraph` objects) using the `graph.data.frame` function contained within the `igraph` package.

```
> library(SANTA)
> data(treated.dataframe)
> data(untreated.dataframe)
```

```
> g.treated <- graph.data.frame(treated.dataframe, directed=FALSE )
> g.untreated <- graph.data.frame(untreated.dataframe, directed=FALSE)
```

SANTA requires any networks to be in the form of an `igraph` graph. Another commonly used graph structure is `graphNEL`. This `graphNEL` structure is used in other R-packages, including `KEGGgraph` and `Graphite`, which contain graph objects sourced from the NCI, KEGG, Biocarta and Reactome databases. Graphs can be converted between the `graphNEL` and `igraph` structures using the `igraph.from.graphNEL` and `igraph.to.graphNEL` functions contained within the `igraph` package.

In order to make a fair comparison between the two networks, it is advisable to ensure that the networks contain the same genes. Therefore, we will compare the genes contained within each network and add those that are missing.

```
> name.treated <- get.vertex.attribute(g.treated, "name")
> name.untreated <- get.vertex.attribute(g.untreated, "name")
> mis.treated <- name.untreated[which(! name.untreated %in% name.treated)]
> mis.untreated <- name.treated[which(! name.treated %in% name.untreated)]
> g.treated <- add.vertices(g.treated, length(mis.treated),
+         attr=list(name=mis.treated))
> g.untreated <- add.vertices(g.untreated, length(mis.untreated),
+         attr=list(name=mis.untreated))
```

We will use information from the Gene Ontology (GO) project [5], made available through the `org.Sc.sgd.db` package, to identify those genes involved in responding to DNA damage. The Gene Ontology project groups into genes according to shared function. The GO term with ID `GO:0006974` is *Response to DNA Damage Stimulus*. We will now use the `org.Sc.sgd.db` package to identify the network genes associated with `GO:0006974`. Similar code can be used to identify genes associated with other GO terms.

```
> library(org.Sc.sgd.db)
> xx <- as.list(org.Sc.sgdGO2ALLORFS)
> associated.genes <- xx[["GO:0006974"]]
> association.treated <- as.numeric(get.vertex.attribute(g.treated,
```

```
+         "name") %in% associated.genes)
> association.untreated <- as.numeric(get.vertex.attribute(g.untreated,
+         "name") %in% associated.genes)
```

If a gene is associated with the GO term it will given a score of 1 in `association.values`. If it is not associated, it will be given a score of 0. We will now store these scores in the 2 graphs under a vertex attribute called `rdds`.

```
> g.treated <- set.vertex.attribute(g.treated,
+         name="rdds", value=association.treated)
> g.untreated <- set.vertex.attribute(g.untreated,
+         name="rdds", value=association.untreated)
```

$\texttt{K}^{\texttt{net}}$ is also able to incorporate edge weights when quantifying the clustering of high-weight vertices. Edge weights can represent numerous different biological properties, including the strength of a physical interaction between 2 gene products, or in the case of our 2 networks, the strength of the genetic interaction. It is necessary to convert these weights into distances, so that strongly connected genes are linked by edges with small distances. We will convert the genetic interactions into distances by taking the absolute values of the interaction strengths and subtracting them from the largest absolute interaction strength value. This will also ensure that all edge distances are positive. The inclusion of negative edge distances will result in $\texttt{K}^{\texttt{net}}$ producing a error.

```
> s.treated <- get.edge.attribute(g.treated, name="gi-score")
> s.untreated <- get.edge.attribute(g.untreated, name="gi-score")
> g.treated <- set.edge.attribute(g.treated, name="distance",
+         value=max(abs(s.treated)) - abs(s.treated))
> g.untreated <- set.edge.attribute(g.untreated, name="distance",
+         value=max(abs(s.untreated)) - abs(s.untreated))
```

We will now apply the $\texttt{K}^{\texttt{net}}$ function to the GO term on each network in order to determine on which network the GO term clusters most strongly. The greater the number of permutations run, the greater the reproducibility of the p-value. We will use 100 permutations of the vertex weights in order

to produce the p-values.

By default, the function attempts to use a vertex attribute named `pheno` as vertex weights and an edge attribute named `distance` as edge distances. We will need to specify that the GO term associations should be used a vertex weights, but can leave the default edge attribute, as the modified GI scores were saved under this attribute name previously.

```
> res.treated <- Knet(g.treated, nperm=100, dist.method="shortest.paths",
+         vertex.attr="rdds")
> res.untreated <- Knet(g.untreated, nperm=100, dist.method="shortest.paths",
+         vertex.attr="rdds")
> res.treated$pval

[1] 5.52284e-09

> res.untreated$pval

[1] 7.052683e-05
```

As the permutations of the vertex weights are random, the p-values you calculate may not be the same as the ones shown above.

Since the GO term tested was the *Response to DNA Damage Stimulus* term, it is not surprising that the gene set clusters more significantly on the treated network (as indicated by the lower p-value). The yeast exposed to the DNA-damaging agent would have activated or upregulated pathways involved in responding to the agent, thereby increasing the strength of the genetic interaction between DNA-damage response-related genes. The same method can be used to GO terms for which the relative association strengths are less predictable.

We will now visualise the observed and permuted K^{net} curves and AUKs for the GO term on the DNA-damaged and undamaged networks.

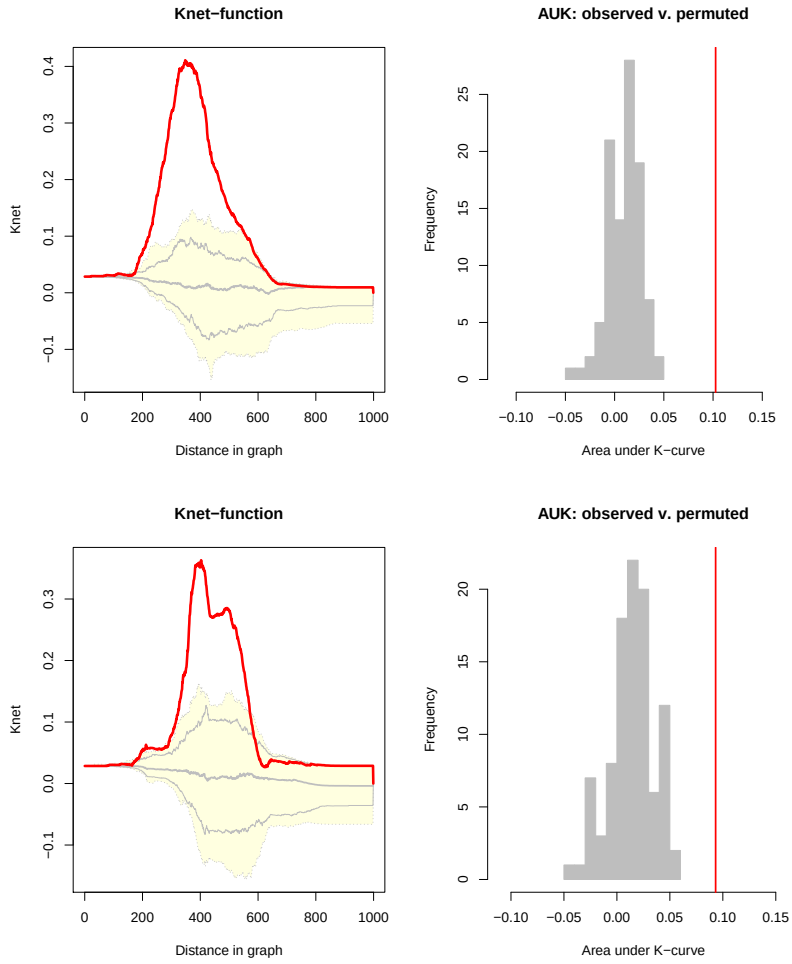

Figure 1: (**Left**) The observed $\mathtt{K}^{\mathtt{net}}$ -function curve (red line) and the permuted $\mathtt{K}^{\mathtt{net}}$ -function curve quantiles (yellow area) for the *Response to DNA Damage Stimulus* gene set on the DNA-damaged (**Top Left**) and undamaged (**Bottom Left**) GI networks. The position of the observe curves relative to the permuted quantiles indicates that the gene set clusters on both networks. (**Right**) The observed $\mathtt{K}^{\mathtt{net}}$ -function AUK (red line) and the permuted $\mathtt{K}^{\mathtt{net}}$ -function AUKs (grey histogram) on the DNA-damaged (**Top Right**) and undamaged (**Bottom Right**) GI networks. The fact that the observed AUKs are greater than the permuted AUKs again indicates that the gene set clusters on both networks. The greater distance between the observed AUK and the permuted AUKs in the DNA-damaged network indicates that clustering of the gene set is more significant in this network.

The left-hand plots display the the observed curve in red and the quantiles of the permuted curves in yellow. The quantile boundaries are displayed as grey lines. These boundaries are specified in the $\texttt{K}^{\texttt{net}}$ function. The right plot displays the observed AUK as a red line and the distribution of permuted AUKs in grey. If clustering of high-weight vertices is present on a graph, then the observed $\texttt{K}^{\texttt{net}}$ curve and AUK will be greater than the permuted $\texttt{K}^{\texttt{net}}$ curves and AUKs. The greater the degree of clustering, the greater the difference between the observed and permuted statistics. If the degree of clustering is low, then the observed and permuted curves and AUKs will overlap. A z-test is performed on the observed and permuted AUKs in order to produce a p-value for the clustering.

4 Case study II - Using $\texttt{K}^{\texttt{node}}$ to prioritise genes for follow-up functional studies

This section will demonstrate how the **SANTA** $\texttt{K}^{\texttt{node}}$ function can be used to rank vertices in a network by their respective strength of association with high-weight vertices present on the network.

The GO database contains the largest number of gene-function annotation currently available. However, a significant proportion of known genes remain unannotated or under-annotated. Experimentally testing the function of genes is an expensive and time-consuming process. Therefore, it is important to be able to computationally identify the unannotated genes that are most likely to share functionality with previously identified sets of genes. Under the GBA principle, unannotated genes that share a large number of functional interactions with an annotated gene set are likely to be involved in the gene sets function. Because of this, the $\texttt{K}^{\texttt{node}}$ function is a useful tool for ranking unannotated genes by their likely association with a function.

We will use a GI network used in the previous section [4] to rank unannotated genes by their likely involvement in the *Response to DNA Damage Stimulus* function [5]. We will now load the **SANTA** library and a data frame containing the edges from the MMS-treated (DNA-damaged) network and create a **SANTA** -compatible graph.

```
> library(SANTA)
> data(treated.dataframe)
> g.treated <- graph.data.frame(treated.dataframe, directed=FALSE)
```

As previously shown, the *Response to DNA Damage Stimulus* GO term associates more strongly with the MMS-treated GI network. This is due to the functional re-wiring that occurs within the cell in response to exposure to this DNA-damaging agent. We will therefore again identify the genes contained within this network that are association with this GO term and store them under a vertex attribute call `rdds`.

```
> library(org.Sc.sgd.db)
> xx <- as.list(org.Sc.sgdGO2ALLORFS)
> associated.genes <- xx[["GO:0006974"]]
> association.values <- as.numeric(get.vertex.attribute(g.treated,
+         "name") %in% associated.genes)
> g.treated <- set.vertex.attribute(g.treated, name="rdds",
+         value=association.values)
```

As mentioned in the previous section, it is important to convert edge weights to meaningful measures of distance between connected vertices. The GI scores are converted by subtracting the absolute GI score of each edge from the maximum absolute GI score. This ensures that those gene pairs seen to have the strongest interactions are connected by the shortest distances.

```
> s.treated <- get.edge.attribute(g.treated, name="gi-score")
> g.treated <- set.edge.attribute(g.treated, name="distance",
+         value=max(abs(s.treated)) - abs(s.treated))
```

We will use the vertex attribute containing GO term association information as the vertex weights in the $\texttt{K}^{\texttt{node}}$ function. By default, the edge attribute named `distance` is used as edge distances.

Alongside each vertices' AUK score, a number of other centrality scores can be returned by the $\texttt{K}^{\texttt{node}}$ function, including each vertices' weight, degree,

betweenness and Markov centrality. Specifying `only.Knode=FALSE` results in this information being returned alongside the AUK score as a data frame.

```
> res <- Knode(g.treated, vertex.attr="rdds", only.Knode=FALSE)
> res[1:10, 1:2]
```

```
          nodeAUK vertex.weights
YDL106C 0.1361992              0
YBL046W 0.1347715              1
YCR066W 0.1338610              1
YGL019W 0.1334129              1
YNL136W 0.1319775              1
YBR112C 0.1315502              1
YDR017C 0.1313863              0
YBR160W 0.1310819              1
YEL018W 0.1306317              1
YBR136W 0.1304297              1
```

The genes with the highest AUK scores are those with the strongest associations to the gene set. Since we only want to look at the ranking of the unannotated genes, we can remove annotated genes using the vertex weight information given in the second column of the data frame.

```
> res[res[,2]!=1, ][1:10, 1:2]
```

```
          nodeAUK vertex.weights
YDL106C 0.1361992              0
YDR017C 0.1313863              0
YGR135W 0.1271454              0
YDR216W 0.1258819              0
YBR245C 0.1256333              0
YDR173C 0.1252847              0
YGL059W 0.1233614              0
YJL187C 0.1208257              0
YGL096W 0.1199538              0
YNL204C 0.1193900              0
```

The genes ranked above are the unannotated genes most strongly associated with the functional set. These genes represent the unannotated genes most likely be involved in the response to DNA damage stimulus. Experimental testing of these genes would be able to verify whether this is true or not.

5 Session info

This document was produced using:

```
> toLatex(sessionInfo())
```

- R version 3.0.0 (2013-04-03), `x86_64-apple-darwin10.8.0`
- Locale: `C`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: AnnotationDbi 1.22.1, Biobase 2.20.0, BiocGenerics 0.6.0, DBI 0.2-5, RSQLite 0.11.2, SANTA 1.0.0, igraph 0.6.5-1, org.Sc.sgd.db 2.9.0, snow 0.3-12
- Loaded via a namespace (and not attached): IRanges 1.18.0, msm 1.1.4, mvtnorm 0.9-9994, splines 3.0.0, stats4 3.0.0, survival 2.37-4, tools 3.0.0

References

[1] Carlo Gaetan and Xavier Guyon. *Spatial Statistics and Modeling*. Springer, 2010.

[2] Atsuyuki Okabe and Ikuho Yamada. The k-function method on a network and its computational implementation. *Geographical Analysis*, 33(3):271–290, 2001.

[3] Scott White and Padhraic Smyth. Algorithms for estimating relative importance in networks. *Proceeding of the ninth ACM SIGKDD international conference*, pages 266–275, 2003.

[4] Sourav Bandyopadhyay, Monika Mehta, Dwight Kuo, et al. Rewiring of genetic networks in response to dna damage. *Science*, 330(6009):1385–1389, Dec 2010.

[5] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, May 2000.