Brazilian Journal of Cartography (2017), Nº 69/5, Special Issue GEOINFO 2017: 959-969
Brazilian Society of Cartography, Geodesy, Photgrammetry and Remote Sense
ISSN: 1808-0936

GEOINFO
Brazilian Symposium on GeoInformatics
November 27th to 30th, Campos do Jordão, SP, Brazil

SPATIOTEMPORAL DATA REPRESENTATION IN R

Representação de Dado Espaço-Temporal no R

Lorena A. Santos, Karine Reis Ferreira, Gilberto Ribeiro de Queiroz & Lubia Vinhas

Instituto Nacional de Pesquisas Espaciais - INPE
Divisão de Processamento de Imagens - DPI
Av. dos Astronautas, 1758, 12227-010 São José dos Campos, SP, Brasil
{lorena.santos, karine.ferreira, gilberto.queiroz, lubia.vinhas}@inpe.br

Received on March 14, 2017/ Accepted on April 28, 2017
Recebido em 14 de Março, 2017/ Aceito em 28 de Abril, 2017

ABSTRACT

Recent advances in devices that collect geospatial information have produced massive spatiotemporal data sets. Earth observation and GPS satellites, sensor networks and mobile gadgets are examples of technologies that have created large data sets with better spatial and temporal resolution than ever. This scenario brings a challenge for Geoinformatics: We need software tools to represent process and analyze these large data sets efficiently. R is an environment widely used for data analysis. In this work, we present a study of spatiotemporal data representation in R. We evaluate R packages to access and create three spatiotemporal data types as different views on the same observation set: time series, trajectories and coverage.

Keywords: Spatiotemporal Data, Geospatial, R Packages.

RESUMO

Os recentes avanços em dispositivos que coletam informações geoespaciais produziram grandes conjuntos de dados espaço-temporal. A observação da Terra e os satélites GPS, redes de sensores e dispositivos móveis são exemplos de tecnologias que criaram grandes conjuntos de dados com melhor resolução espacial e temporal. Este cenário traz um desafio para a Geoinformática: Faz-se necessário o uso de ferramentas de software para representar processos e analisar estes grandes conjuntos de dados de forma eficiente. R é um ambiente amplamente utilizado para análise de dados. Neste trabalho, apresenta-se um estudo da representação de dados espaço-temporal no R. Avaliamos os pacotes R para acessar e criar três tipos de dados espaço-temporal com diferentes visões sobre o mesmo conjunto de observações: séries temporais, trajetórias e *coverages*.

Palavras-chave: Dado Espaço-Temporal, Geospacial, Pacotes R.1. 1.

1. INTRODUCTION

Recently, the number of devices that collect geospatial information has greatly increased. Earth observation and GPS satellites, sensor networks and mobile gadgets are examples of technologies that have created large data sets with better spatial and temporal resolution than ever. This technological advance brings many challenges for Geoinformatics. We need novel software tools to represent process and analyze big spatiotemporal data sets efficiently.

In Geoinformatics, spatiotemporal data representation is an open issue. Spatial information is represented following well-established models and concepts. This includes the dichotomy between object-based and field-based models (GALTON, 2004). Examples of long-standing concepts are vector and raster data structures, topological operators, spatial indexing, and spatial joins (RIGAUX *et al.,* 2002). Most existing GIS and spatial database systems, such as *PostGIS* and Oracle Spatial, are grounded on these concepts. However, there is no consensus on how to represent spatiotemporal information in computational systems.

Many existing proposals of spatiotemporal data models focus on representing the evolution of objects and fields over time. Some proposals are specific for discrete changes in objects (WORBOYS, 1994) (HORNSBY & EGENHOFER, 2000), others for moving objects (GUTING & SCHNEIDER, 2005) (ISO, 2008) and still others for fields or coverage (LIU *et al.,* 2008) (OGC, 2006). To properly capture changes in the world, representing evolution of objects and fields over time is not enough. We also need to represent events and relationships between events and objects explicitly (WORBOYS, 2005). Events are occurrents (GALTON & MIZOGUCHI, 2009). They are individual happenings with definite beginnings and ends. The demand for models that describe events has encouraged recent research on spatiotemporal data modeling (GALTON & MIZOGUCHI, 2009).

R is a software tool widely used for data analysis (R Development Core Team, 2011). It provides a broad variety of statistical methods (time-series analysis, classification and clustering) and a high-level programming environment and language suitable for fast developing new algorithms. R is extended via packages. Although there are many packages for spatial data handling and analyzing, few of them can properly deal with the temporal dimension of spatial data.

This paper presents a study of spatiotemporal data handling in R. We evaluate R packages for spatiotemporal data access and representation. To guide this evaluation, we consider the spatiotemporal data types proposed by Ferreira *et al.* (2014). They propose a data model that represents objects and fields that change over time as well as events. Based on this model, we describe in this work how to load and create three spatiotemporal data types in R as different views on the same observation set: *time series*, *trajectory* and *coverage*.

This paper is based on Santos *et al.* (2016), previously presented in XVII Brazilian Symposium on Geoinformatics (GeoInfo 2016) (http://www.geoinfo.info/geoinfo2016/).

2. AN OBSERVATION-BASED MODEL FOR SPATIOTEMPORAL DATA

Ferreira *et al.* (2014) proposed a data model for spatiotemporal data and specified it using an algebraic formalism. Algebras describe data types and their operations in a formal way, independently of programming languages. The proposed algebra is extensible, defining data types as building blocks for other types. It takes observations as basic units for spatiotemporal data representation and allows users to create different views on the same observation set, meeting application needs.

Observations are our means to assess spatiotemporal phenomena in the real world. Recent research draws attention to the importance of using observations as a basis for designing geospatial applications (KUHN, 2009). The proposed model defines three spatiotemporal data types as abstractions built on observations: *time series, trajectory* and *coverage*. A *time series* represents the variation of a property over time in a fixed location. A *trajectory* represents how locations or boundaries of an object change over time. A *coverage* represents the variation of a property in a spatial extent at a time. We also define an auxiliary type called *coverage series* that represents a time-ordered set of coverages

that have the same boundary. Using these types, we can represent objects and fields that change over time as well as *events*.

2.1 Different views on the same observation set

Figure 1 shows an example of observations collected by five moving objects. Each observation is represented as a tuple in the form (*id, x, y, t, p*), where *id* is the object identification, *x* and *y* are spatial locations, *t* is time and *p* is a property value collected at the spatial local *x* and *y* and on time *t*. In this example, the collected property is air pollution.

On these observations, we can create different views depending on the kind of analysis we want to perform on them. Each view is materialized as a data type. Figure 2 illustrates trajectory and coverage instances built on the same observation set shown in Figure 1. For example, to analyze how the objects move over time and space, we create an instance of the *trajectory* data type for each object. Each trajectory instance contains observations of a specific object. To analyze how air pollution varies in a region, we create instances of the coverage data type. Each *coverage* instance contains observations in a specific period, mixing observations of different objects. To analyze how air pollution varies in a given spatial location over time, we can create an instance of *time series* data type.

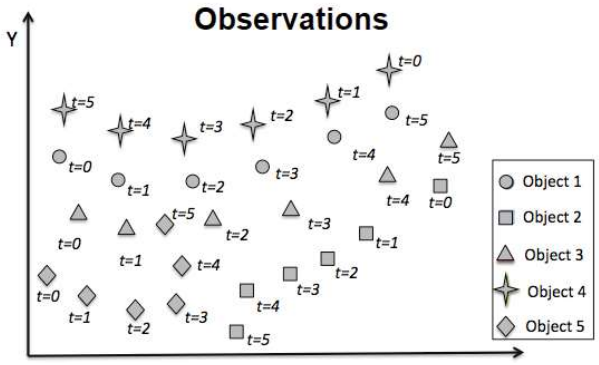

Fig. 1- Spatiotemporal observations.

In the proposed model, each spatiotemporal data type instance, *time series, trajectory* and *coverage*, has an interpolation function, called interpolator, suitable for it. A time series has an interpolator that estimates property values in non-measured times. A trajectory has an interpolator able to estimate locations in non-observed times. A coverage has an interpolator to estimate property values in non-observed locations.

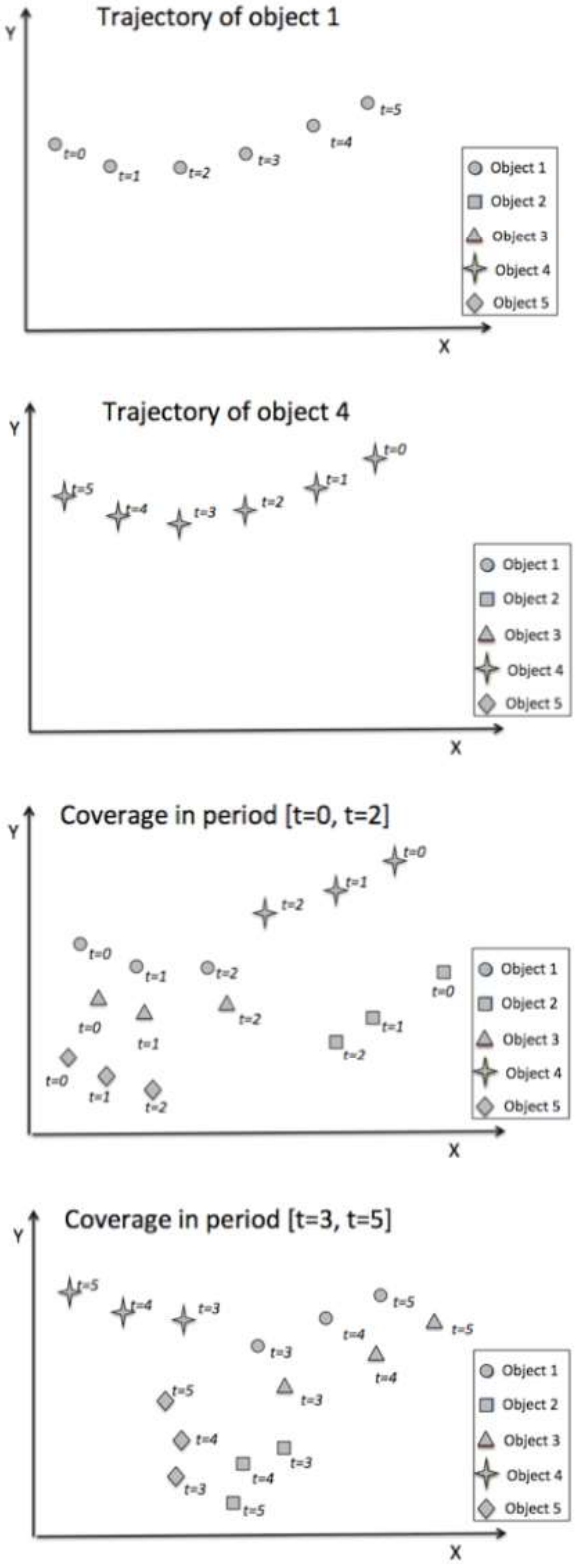

Fig. 2 - Different data types built on the same observation set

The capability to create different views or data types on the same observation set is an important requirement for spatiotemporal data representation. In this work, we evaluate how to

do this using R. We use equivalent R data types to represent observation sets and to create trajectory, time series and coverage from these sets.

3. SPATIOTEMPORAL DATA REPRESENTATION IN R

In this section, we describe a set of R packages for spatiotemporal data representation and access. Packages for data access are `Rgdal` (BIVAND *et al.,* 2013a), `Rpostgres` (CONWAY *et al.,* 2008) and `Rodbc` (RIPLEY and LAPSLEY, 2016). Packages with data types that can be used to represent spatiotemporal data are `spacetime` (PEBESMA, 2012), `xst` (RYAN and ULRICH, 2012), `trajectories` (PEBESMA and KLUS, 2015) and `raster` (HIJMANS, 2016). Furthermore, we evaluate some R packages that provide interpolation functions that are crucial to create spatiotemporal data type, such as `gstat` (PEBESMA and GRAELER, 2016).

Figure 3 shows a diagram with these packages and the relationships among them.

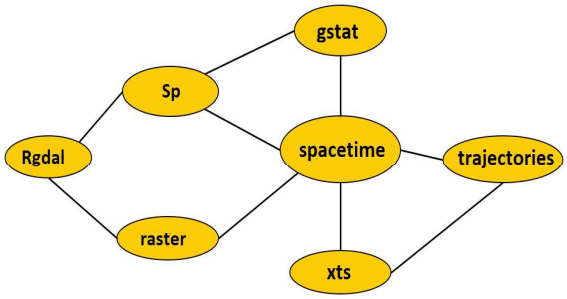

Fig.3 - Relationships between packages

Spatiotemporal data can be obtained from different data sources, such as, database (e.g. *Postgis*), data files (e.g. shapefiles and geotiff raster files) and web services (FERREIRA *et al.,* 2015). In R, there are packages that can access data from distinct types of source. However, these packages do not work directly with the concept of spatiotemporal data.

The `Rgdal` (BIVAND *et al.,* 2013a) package allows access to a broader range of spatial data sources (LOVELACE & CHESHIRE, 2014), such as shapefiles, raster data files and database systems. `RODBC` and `Rpostgres` packages allow to create SQL queries in R for accessing data in database systems, but they do not deal with spatial data.

The spacetime package contains a set of base classes for spatiotemporal data representation that are widely used for other R packages for spatiotemporal analyses. It is built upon the classes and methods for spatial data from package sp and for time series data from package xts (PEBESMA, 2012). The xts package was chosen due to its support to represent several types of date and time. Moreover, it extends functionality of zoo package that has good tools for aggregation over time (ZEILEIS & GROTHENDIECK, 2005).

Each spatial data type from `sp` package has two slots that contain bound box, a matrix of numerical coordinates and other slots that contain a `CRS` class object defining the coordinate reference system (BIVAND *et al.,* 2013b). The spatial data can be a particular spatial point, line, polygon or set of polygons, or a pixel (grid or raster cell) (PEBESMA, 2012).

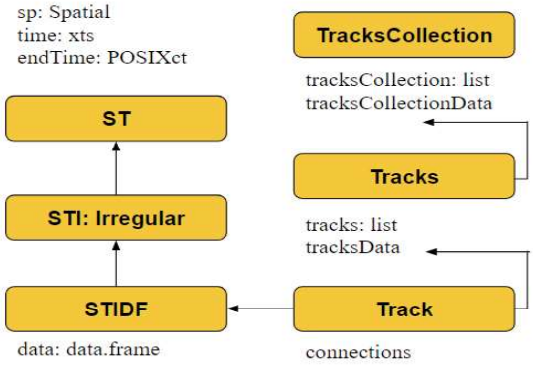

Fig.4 - Classes for spatiotemporal data in package spacetime. Source: (PEBESMA, 2016)

The `spacetime` package classes are shown in Figure 4. Spatial Full Grid (`STF`) and Sparse Grid (`STS`) have the same general layout, with observations on a space time grid. The main difference between both is that `STF` stores the full grid, all observations of all space time points, while `STS` only stores non-missing valued observations. Examples that can be represented by these classes are: time sequences of satellite imagery and measurements of air quality at every hour (PEBESMA, 2016).

Irregular Grid (`STI`) represent a layout where for each spatial data a time point is stored. An example that can be represented by `STI` is measurement from mobile sensors. All presented classes derive from a base class, `ST`. This class is virtual, which does not represent actual data. The ST class derives two order classes: a spatio-temporal geometries and an augment class with actual data, in the form of data frame.

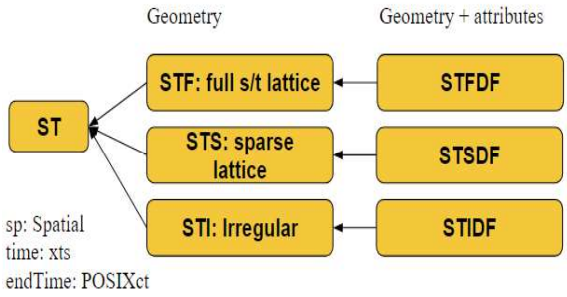

Fig.5 - Classes for spatiotemporal data as trajectories. (PEBESMA, 2016)

The objects from `xts` and `sp` packages are not used to store property values. For purely temporal information `xts` is used, and for purely spatial information `sp` objects is used. Then, it is necessary to use a `data.frame` to combine, space, time and property values. It represents the data as rectangle of rows containing observations on columns of property values (BIVAND *et al.,* 2013b).

The raster data set is composed by multiple layers, hence, the `raster` package has two classes for work with multi-layer data, `rasterStack` and `rasterBrick`. The principal difference between these classes is that a `rasterBrick` can only be linked to a single file, while `rasterStack` can be formed from separate files and/or from few layers from a single file (HIJMANS 2016). Each raster layer in the stack or brick needs to be in the same projection, spatial extent and resolution. In other cases, such as reading satellite images, when we do not use `Rgdal` package, the raster package can be used directly using the function `raster()`. To represent spatiotemporal raster data, from a raster collection, we add a time for each layer using the `setZ` function from `raster` package, then we can coerce these data to spatiotemporal type from `spacetime` package.

`Trajectories` package provides three data types to represent trajectories, `Track`, `Tracks` and `TracksCollection`, based on the `STIDF` type, as shown in Figure 5. The class `Track` represents a single trajectory followed by a person, animal or object. `Tracks` embodies a collection of trajectories followed by a single person, animal or object. The class `TracksCollection` represents a collection of trajectories followed by different persons, animals or objects. Besides that, this package provides a set of operations over trajectories, such as computing of trajectories STBox and calculating distances between two tracks.

The package `gstat` provides spatial and spatiotemporal interpolation functions. Types derived from `sp` and `spacetime` packages can be utilized in this package.

4. CASE STUDY

In this section, we present the use of the R packages listed in the previous section to approach the data types proposed by (FERREIRA *et al.,* 2014). We performed a case study using observations of vessels around the Brazilian coast. These observations are stored in a *PostGIS* database and contain trajectories of 993 vessels collected during 4 years, from 2008 to 2011. The Figure 6 presents the observations of all vessels.

Fig.6 – Observations of 993 marine vessels during 4 years

4.1 Observation Set

In this case study, we selected a small subset, containing trajectories of 166 vessels. We filtered data temporally obtaining trajectories for only one day and spatially for locations in Rio de Janeiro State. The filtered observations are shown in Figure 7.

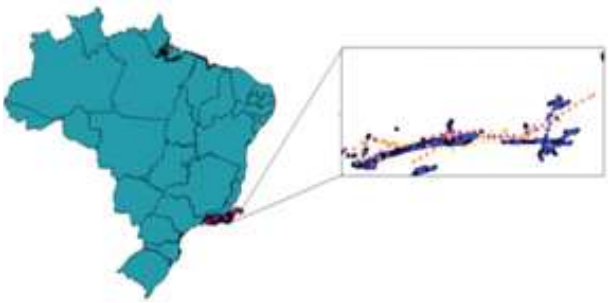

Fig.7 -Observations of 166 marine vessels during one day.

The filtered vessel observations are stored in table in a PostGIS database, where each row contains one observation of a vessel. Each observation contains the vessel id (integer type), time (timestamp type), spatial location (geometry type) and velocity of the vessel (numeric type). The table format is shown in Figure 8.

id integer	datahora timestamp	ponto Geometry	velocity numeric
583	2010-06-17 05:08:35	POINT(-43.1167 -23.0669)	0.408763
583	2010-06-17 06:08:05	POINT(-43.1622 -23.0769)	1.353742
583	2010-06-17 07:08:10	POINT(-43.1361 -23.0733)	0.751117

Fig. 8 - Observation set.

In this case study, we used RODBC to connect in the database and filter the observation data using SQL language inside R environment. This step is shown in the code bellow.

```
library (RODBC)
con    <- odbcConnect("PostgreSQL")
query <- "select id, datahora,
                velocity,
                st_x(ponto) x,
                st_y(ponto) y,
          from onedayVelocidadeAll
          order by datahora"

VesselsObs <- sqlQuery(con,query)
```

RODBC does not work with spatial data. Therefore, coordinates data is returned as numeric type and so must be converted to spatial type of R.

A matrix is created with coordinate values and then we transform this matrix in a SpatialPoint in R format with a coordinate reference system. For temporal data, we do not need to execute changes, because the data is returned in POSIXct type. This type is the standard way of representing time in R (WUERTZ *et al.*, 2015), and also in spacetime package. Finally we created a data.frame, velObsDF, with only one column, containing velocity data, VelocityDF. Combining the spatial, temporal and data frame objects we can apply in a spacetime class. In this case we apply STIDF, because our observations are irregular data, thus we create a spatiotemporal observation, called here as vesselObsST. Figure 9 shows how these three data type become a spatiotemporal observation in R.

```
Spatial
library(sp)
spVessel<-cbind(vesselsObs$x,vesselsObs$y)
spV <- SpatialPoints(spVessel)
spV@proj4string<-CRS("+init=epsg:4326")

Time
TimeObs<- vesselsObs$datahora

Data Frame
velObs<-vesselsObs$velocity
velObsDF=data.frame(VelocityDF=velObs)

STIDF
library(spacetime)
vesselObsST= STIDF(spV, TimeObs, velObsDF)
```

Fig. 9- Creating STIDF object

4.1 Trajectories and Time Series

Two vessels from the observation set were selected. We created two objects of type STIDF, one for each vessel, vesselST1 and vesselST2. Then, we created two instances of Track type from trajectories package, each one to represent a trajectory of an object. The R code for this step is:

```
library (spacetime)
VesselST1 <- STIDF (spv1, TimeObs1,          velObsDF1)
VesselST2 <- STIDF (spv2, TimeObs2, velObsDF2)

library (trajectories)
Tv1 <- Track (VesselST1)
Tv2 <- Track (VesselST2)
```

The generated trajectories from this code are shown in Figure 10.

Here, we can also analyze how two vessels are moving together. The trajectories has a function to calculate distances between two tracks using the method compare. This function returns an object of type difftrack, called here trajCmp. From this object, we can obtain the distance between trajectories over time and create a time

series using the xts type, as shown in Figure 11. This time series represents the distance variation between two trajectories over time. The code is shown below:

```
library (xts)
trajCmp <- compare (Tv1, Tv2)
timeCmp <- trajCmp@conns1$time
dist1   <- trajCmp@conns1$dists
distTS  <- xts(dist1, timeCmp)
```

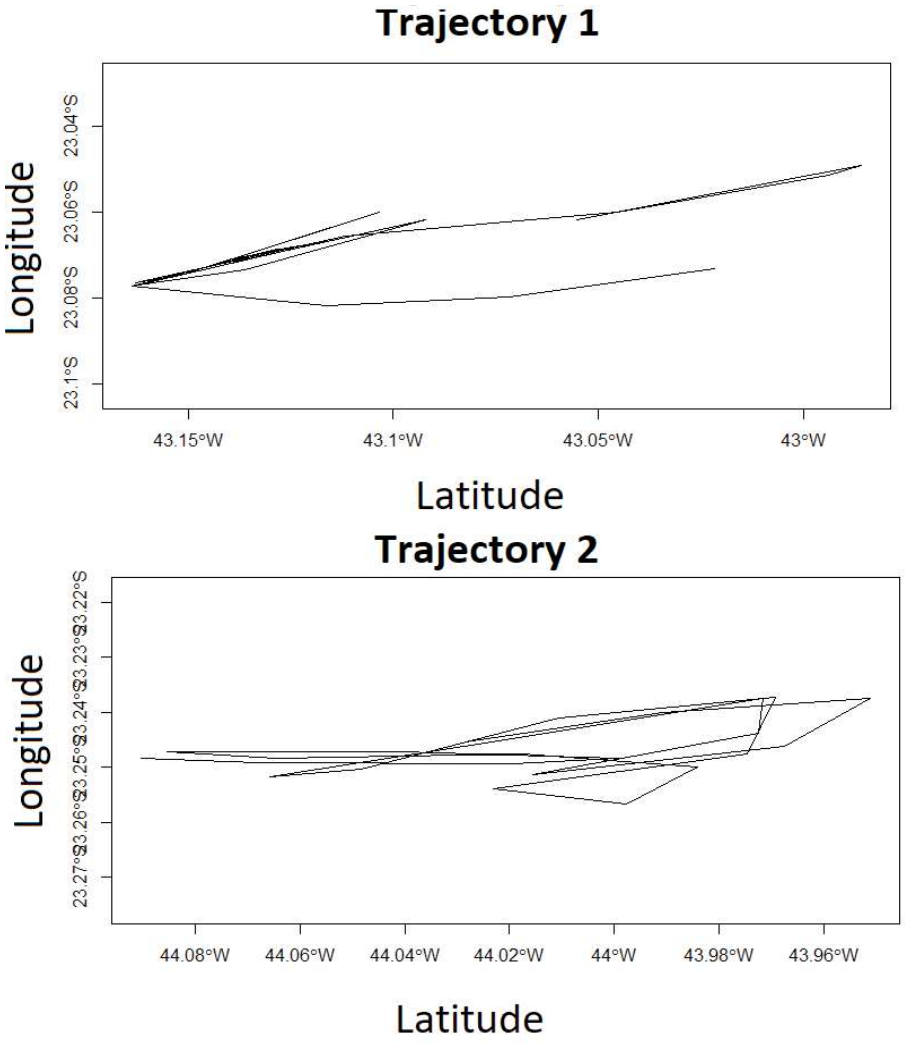

Fig. 10- Visualizing spatiotemporal data as Trajectories.

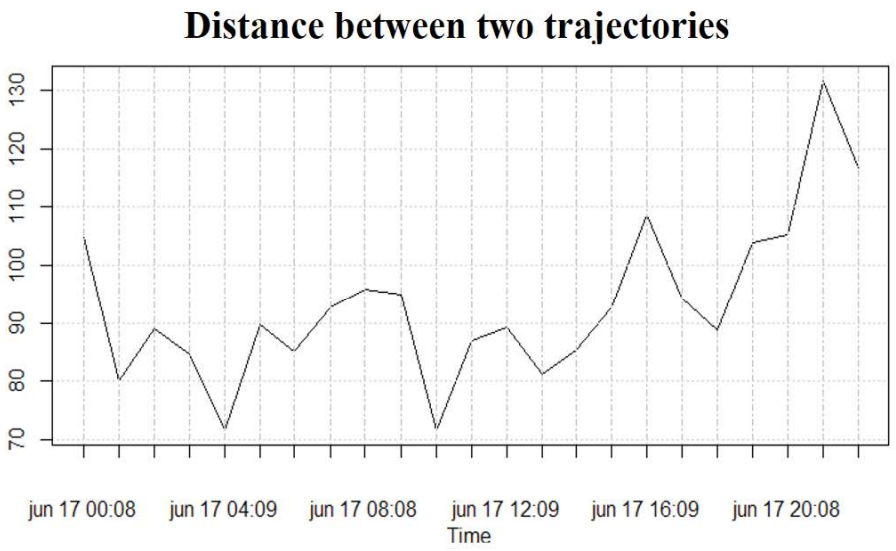

Fig. 11- Time Series: distance variation over time.

4.2 Coverage

To represent the velocity variation over time and space, we create coverages. We put together observations obtained by vessels during one day and produced a coverage that investigate how velocity varies within the boundary delimited by the spatiotemporal object bounding box previously created. This kind of view on the vessel observations allows users to identify possible regions where vessels are fishing. In fishing areas, vessels have low velocity.

Our observations are discrete. They need to be combined with interpolation functions to estimate values in non-observed spatial locations (FERREIRA *et al.*, 2014). We used a spatial interpolation and created a grid where every cell has a velocity value.

To create a grid, we constructed a GridTopoloy object using the bounding box of our subset region and a cell size of 0.01◦ in each direction. From the GridTopoloy object, we constructed the SpatialGrid object and associated a coordinate reference system (CRS) to it. The code bellow is used to create this grid.

```
cs <-c(0.01, 0.01)
cc <-vesselObsST@sp@bbox[,1]+(cs/2)
cd <-ceiling(
      diff(t(vesselObsST@sp@bbox))/
      cs)
gridVessel <-spatialGrid
            (GridTopology(
             cellcentre.offset= cc,
             cellsize= cs,
             cells.dim= cd))

proj4string(gridVessel) <-
```

We created one coverage for each hour of a day, using the Inverse Distance Weighted Interpolator (IDW) interpolation function. In this case, one coverage contains the observations of a specific hour, mixing observations of different vessels. The type returned by the IDW interpolator is SpatialGridDataframe. This type is typical for representation raster GIS (BIVAND *et al.*, 2013b). Using Rgdal, each grid can be saved as Tiff format. The code bellow shows how to obtain the time interval from STIDF object to be spatially interpolated over the grid created earlier. This step was executed 24 times, for each hour. The following code presents the interpolation for one hour.

```
Library (gstat)
t0 <- coredata(vesselObsST@time
      ['2010-06-17 00:00:00/
        2010-06-17 01:00:00'])
vessel0 <- idw(vesselObsST@data
           [t0[1]:t0[length(t0],],
           gridVessel, idp=2.5)
```

Once all grids were created, we read the 24 grids using stack() function from raster package and put these raster grids together using stack::raster function. Two stacks were generated, one containing all raster grids during 00:00 until 12:00 and another containing all raster grids during 12:00pm until 23:59pm. Furthermore, each raster pushed in the stack was associated to time interval using the setZ() function from the raster package. These raster stacks contain spatial and temporal data. They can be converted to the spatiotemporal data type STFDF of the spacetime package. The following code that describes this step for one time period.

```
library (raster)
s12 <-stack(vessel12)
period_12_24 <- seq(
                from=as.POSIXct(
                "2010-06-17 01:00",
                tz="UTC"),
                to=as.POSIXct(
                "2010-06-17 12:00",
                tz="UTC"),
                by="hour")

rasterCover12_24 <- raster::stack
                 (s12,s13,s14,s15,
                  s16,s17,s18,s19,
                  s20,s21,s22,s23)

rasterST12_24 <- setZ(
                 rasterCover12_24,
                 period_12_24)

names(rasterST12_24)<- format(
                  tempoTotal12_24,
                  '%a_%Y%m%d')
STRaster12_24 <- as(
                 rasterST12_24,
                 "STFDF")
```

The Figure 12 shows the spatiotemporal raster grids representing coverages in R. These coverages represent how the velocity varies over time within a specific region of the ocean. Looking these coverages, we can visually identify areas where the velocity is low. Such areas can be possible regions where vessels are fishing. Thus, using coverage types built on the vessel observations, we can extract information about the variation of velocity over time and space, using spatiotemporal data mining techniques.

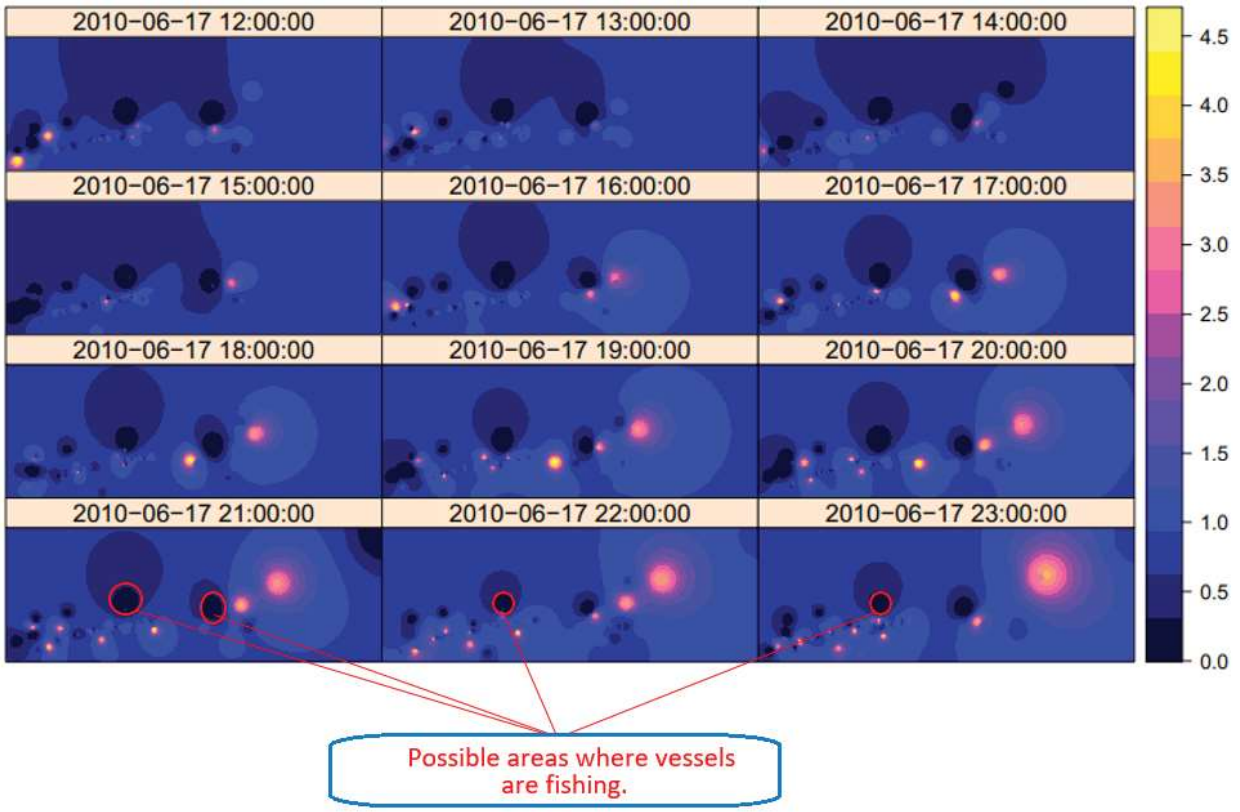

Fig. 12 - Velocity variation during 12:00 until 23:59

It is important to note that the time interval between 12:00 and 15:00 in Figure 12, contains large regions where the vessel velocity is low. We contrast with red circle regions where velocity is lower than others. Visually, we observed that these regions have low velocity in all time intervals from 12:00 until 23:59 pm.

5. EVALUATION AND FINAL REMARKS

Although there are conceptual differences between the classes provided by R and the data types proposed by Ferreira *et al.*, (2014), it is possible to use existing R classes to represent such data types. In summary, Table 1 lists the R packages and their classes that can be used to represent spatiotemporal data.

The data types proposed by Ferreira *et al.* (2014) have interpolation functions, called interpolator, associated to their instances. In R packages, the interpolation functions are not directly associated to their objects. Thus, we have to use interpolation functions provided by other packages to estimate values in non-observed locations and times. For example, to create the coverages we used the IDW interpolation function from gstat package.

We can conclude that the spacetime package plays a key role in spatiotemporal data representation in R. It provides the base data types, STIDF, STFDF and STSDF that are used by many other packages to represent and analyze spatiotemporal data. For example, the trajectories package defines their classes to represent trajectories based on the STIDF type. The gstat package provides spatiotemporal interpolation functions, such as spatiotemporal kriging based on the STFDF class.

Using existing R packages, such RODBC and Rgdal, we can access data from different data sources. Furthermore, we can load subsets of data using queries in the case of vector data or creating several stacks of raster data to be processed in batches to avoid overhead in memory. However, they do not deal with spatiotemporal data.

When we load the observation set using the RODBC package, spatial data sets are loaded as textual or numerical types. Then, it is necessary to convert these data types to spatial type. Using the Rgdal package, we can access the observation set as spatial data types. Neither of them can access spatiotemporal data directly. It is necessary to prepare spatial and temporal data separately and then construct spatiotemporal data types.

Table 1: R Packages and their classes to represent spatiotemporal data

Spatiotemporal Data Type	Package	Class
Observation	spacetime	STIDF
Time Series	xts	xts
Trajectory	Trajectories	Track
Coverage	Raster	setZ
	spacetime	STFDF

A disadvantage of not having packages that directly access spatiotemporal data sets, such as trajectory and coverage, is that we can not filter such data sets properly. For example, we can not load from data sources to R classes only the trajectories whose spatiotemporal bounding boxes intersect a given box. Or, we can not load from data sources to R classes only a part of coverages based on a spatiotemporal restriction. These filters are important because R has limitations of memory on handling large objects. According to KANE *et al.*, (KANE *et al.*, 2013), R is not well suited to work with data structures larger than about 10-20% of a computer RAM memory. Thus, it is necessary to handle data sets by parts in R, using filters to restrict the amount of data in memory.

As future work, we intend to use spatiotemporal data mining algorithms in R to analyze the coverages showed in Figure 12. The goal is to identify regions where vessel velocities are low, that is, regions where vessels are probably fishing.

REFERENCES

BIVAND, R.; KEITT, T.; ROWLINGSON, B. Rgdal: Bindings for the geospatial data abstraction library**. R package version 0.8-10.** 2013a. 53p.

BIVAND, R. S.; PEBESMA, E.; GOMEZ-Rubio, V. Applied spatial data analysis with R, Second edition**. Springer, NY**. 2013b. 405p.

CONWAY, J.; EDDELBUETTEL, D.; NISHIYAMA, T.; PRAYAGA, S. K.; TIFFIN, N. Rpostgresql: R interface to the postgresql database system. **R package version 0.4-1**. 2008. 37p.

FERREIRA, K. R.; CAMARA, G.; MONTEIRO, A. M. V. An algebra for spatiotemporal data from observations to events. **Transactions in GIS**, v.18(2), p:253–269, 2014.

FERREIRA, K. R.; DE OLIVEIRA, A. G.; MONTEIRO, A. M. V.; DE ALMEIDA, D. B.Temporal GIS and spatiotemporal data sources. **In XVI Brazilian Symposium on GeoInformatics (GEOINFO)**, Campos do Jordao, São Paulo, Brazil, 2015, p. 1–13.

GALTON, A. Fields and objects in space, time, and space-time. **Spatial cognition and computation**, 2004, v. 1, p. 39-68.

GALTON, A.; MIZOGUCHI, R. The water falls but the waterfall does not fall: New perspectives on objects, processes and events. **Applied Ontology**, 2009, v. 4(2), p. 71–107.

GUTING, R. H. and SCHNEIDER, M. (2005). **Moving Objects Databases**. . San Francisco, CA: Morgan Kaufmann, 2005. 389p.

HIJMANS, R. J. Introduction to the raster package. **R package version 0.4-1**, 2016. 27p.

HORNSBY, K.; EGENHOFER, M. Identitybased change: A foundation for spatiotemporal knowledge representation. **International Journal of Geographical Information Science**, 2000, v. 14, n. 3, p. 207–224.

ISO. Geographic information - schema for moving features. ISO 19141:2008, **International Organization for Standardization**, Geneva, Switzerland, 2008. 49p.

KANE, M. J.; EMERSON, J.; WESTON, S. Scalable strategies for computing with massive data. **Journal of Statistical Software**, v. 55(14), p. 1–19, 2013.

KUHN, W. A functional ontology of observation and measurement. *In:* **International Conference on GeoSpatial Semantics**, 2009. Berlin, Springer LNCS, v. 5892, 2009, p. 26-43.

LIU, Y.; GOODCHILD, M. F.; GUO, Q.; TIAN, Y.; WU, L. Towards a general field model and its order in GIS. **International Journal of Geographical Information Science**, 2008, v. 22, n. 6, p. 623–643.

LOVELACE, R.;CHESHIRE, J. Introduction to visualising spatial data in R**. National Centre**

for Research Methods Working Papers, 2014, 24p.

OPEN GEOSPATIAL CONSORTIUM (OGC). **OpenGIS abstract specification topic 6: Schema for coverage geometry and functions**. 2006.

PEBESMA, E. Spacetime: Spatio-temporal data in R**. Journal of Statistical Software**, 2012, v.51, p.1–30.

PEBESMA, E. **Handling and analyzing spatial, spatiotemporal and movement data**, 2016. Disponível em < https://edzer.github.io/UseR2016/>. Acesso: novembro 2016.

PEBESMA, E.; GRAELER, B. **Spatio-Temporal Interopolation using gstat**. 2016, 16p.

PEBESMA, E.; KLUS, B. **Analysing trajectory data in R.** 2015, 10p.

R DEVELOPMENT CORE TEAM R: A Language and Environment for Statistical Computing. **R Foundation for Statistical Computing**, Vienna, Austria. 2011.

RIGAUX, P.; SCHOLL, M.; VOISARD. Spatial Databases with Application to GIS. **Morgan Kaufmann Publishers Inc.**, San Francisco, 2002. 411p.

RIPLEY, B.; LAPSLEY, M. Rodbc: Odbc database. **R package version 1.3-13.** 2016, 28p.

RYAN, J.; ULRICH. xts: extensible time series. **R package version 0.8-6.7.** 2012, 22p.

SANTOS, A. L., FERREIRA, R. K., QUEIROZ, R.G., and VINHAS. L. Spatiotemporal Data Representation in R. **In XVII Brazilian Symposium on GeoInformatics(GEOINFO)**, Campos do Jordao, São Paulo, Brazil, November, 2016.

WORBOYS, M. F. A Unified Model for Spatial and Temporal Information. **The Computer Journal**, 1994, v. 37, p. 26-34.

WORBOYS, M. Event-oriented approaches to geographic phenomena. **International Journal of Geographical Information Science**, 2005, v. 19, p. 1-28.

WUERTZ, D.; SETZ, T.; CHALABI, Y.; BYERS, M. M. J. W. (2015). Rmetrics - Chronological and Calendar Objects. **R package.** 2012, 22p.

ZEILEIS, A; GROTHENDIECK, G. zoo: S3 infrastructure for regular and irregular time series. **Journal of Statistical Software**, 2005, v.14,p.1–27.