The “Tract” Class

J. H. Gove
USDA Forest Service, Northern Research Station, 271 Mast Road, Durham, NH 03824 USA
(603) 868-7667; e-mail: jgove@fs.fed.us

Thursday 5th December, 2013 1:56pm

Contents

1 Introduction 1
2 The “Tract” Class 1
  2.1 Class slots . . . . . . . . . . . . . . . . . . 1
  2.2 Object creation . . . . . . . . . . . . . . . 3
    2.2.1 GridTopology-like constructor . . . 3
    2.2.2 Maximum extents constructor . . . 6
3 The “bufferedTract” Class 7
  3.1 Class slots . . . . . . . . . . . . . . . . . . 8
  3.2 Object creation . . . . . . . . . . . . . . . 8
4 The “mirageTract” Class 10
  4.1 Class slots . . . . . . . . . . . . . . . . . . 10
  4.2 Object creation . . . . . . . . . . . . . . . 11
Bibliography 11

1 Introduction

This class forms the basis for the actual tract on which the areal sampling is conducted. It is
simple in concept, being a rectangular area with given cell resolution in which the cells themselves
are always square. This last limitation was imposed to make things simple (one could always define
another class that would allow rectangular cells.) Underneath, as we shall soon see, the class is
actually very rich in functionality because it is a subclass of “RasterLayer” in the raster package.

Subclasses at the moment include the “bufferedTract” and “mirageTract” classes as shown in the
diagram in Figure 1.

2 The “Tract” Class

The class itself is quite basic; there are only two slots that have been added to the “RasterLayer”
class—these are itemized below...

R> getClass('Tract')
The Tract Class

Figure 1: An overview of the “Tract” class, showing the existing subclasses: a “bufferedTract” subclass and a “mirageTract” subclass for handling the ‘mirage’ method of boundary overlap.

Class "Tract" [package "sampSurf"]

Slots:

Name: description units area file
Class: character character numeric .RasterFile

Name: data legend history title
Class: .SingleLayerData .RasterLegend list character

Name: extent rotated rotation ncols
Class: Extent logical .Rotation integer

Name: nrows crs z
Class: integer CRS list

Extends:
Class "RasterLayer", directly
Class "Raster", by class "RasterLayer", distance 2
Class "BasicRaster", by class "RasterLayer", distance 3
Class "RLNULL", by class "RasterLayer", distance 2

Known Subclasses: "bufferedTract", "mirageTract"
2.1 Class slots

- **description**: Some descriptive text about this class.
- **units**: A character string specifying the units of measure. Legal values are “English” and “metric.”
- **area**: The tract area in either square meters or square feet.

However, if we look at the actual class it was derived from, we see much more detail available in the listing of slots above. There are many methods associated with “RasterLayer” objects. These are covered in the vignettes and documentation for that class. All of them are available for use on an object of class “Tract” or any of its subclasses.

2.2 Object creation

The “Tract” class has more than one constructor. The constructors are differentiated in a call to `Tract` by their signatures. The single signature argument in each case is the first argument, called **obj**.

2.2.1 GridTopology-like constructor

The first constructor allows for tract dimensions in terms of cell resolution, origin and extents in both \(x\) and \(y\) directions the way one would specify a “GridTopology” object in the **sp** package. The signature argument is of class “missing” in this case; therefore, the first three arguments shown below are used to specify the underlying tract grid.\(^1\)

In the first case, for example, we might have...

```r
R> cd = c(x=200, y=200)
R> tr = Tract(cellSize = 0.5,
+     cellDims = cd,
+     cellCenter = c(-5,-5),
+     data = runif(prod(cd)),
+     description = 'a simple example tract')
R> validObject(tr)
```

\(^1\)Please note that because the first argument is missing, you must either specify all arguments by name as in the example, or alternatively explicitly list the missing signature **object** as: `Tract(,0.5, cd, c(-5,-5))` to invoke the constructor.
Here we have created a 200 by 200 cell raster grid with resolution of 0.5 meters (i.e., 1 hectare), origin at \((-5, -5)\) meters. The data values for all cells are initialized to zero by default. However,
here we have shown how you can initialize them with a vector of correct length, in this case simply random uniform values.

It is trivial to plot the object, and there is not much to show in this case, but it’s more interesting than a zero-valued surface at least...

\begin{verbatim}
R> plot(tr)
\end{verbatim}

Figure 2: Simple generic “Tract” object.

Note that the origin is expressed in terms of the center of the cell that holds it. You can see this in the above by noting the $x,y$ extents in comparison to the \texttt{cellCenter} argument specified. This particular arrangement for the constructor actually derives from the \texttt{sp} package and the construction of “SpatialGridDataFrame” objects\textsuperscript{2}.

\textsuperscript{2}This constructor was used in prototype code used in Gove and Van Deusen (2011), it has been kept here.
2.2.2 Maximum extents constructor

In the second constructor the signature argument should be a numeric vector of length two with names “x” and “y.” These specify the overall maximum limits in these coordinates, the minimum is always assumed to be (0,0). The cellSize argument is also required.

\[
R> \text{tr2} = \text{Tract}(c(x=20, y=20), \text{cellSize}=0.25)
\]
\[
R> \text{validObject(tr2)}
\]

[1] TRUE

2.2.3 Bounding box constructor

The third constructor uses a valid bounding box to set the tract limits. The signature argument is therefore of class “matrix” and must conform to a bbox with row names “x” and “y,” and column names “min” and “max.” It must also be a 2 \times 2 matrix with minima less than the maxima. One can set the minima to something other than (0,0) with this constructor.

\[
R> \text{bb} = \text{bbox(tr2)}
\]
\[
R> \text{bb[,1]} = c(2,3)
\]
\[
R> \text{bb}
\]

\[
\begin{array}{cc}
\text{min} & \text{max} \\
x & 2 & 20 \\
y & 3 & 20 \\
\end{array}
\]

\[
R> \text{tr3} = \text{Tract(bb, cellSize}=0.5)
\]

2.2.4 RasterLayer constructor

The final “Tract” constructor has signature argument of class “RasterLayer” from the raster package. This allows one to directly generate a tract from a given “RasterLayer” object. There are several constructors for such objects that are documented in the raster package help files. Suffice it to say that this should add all the flexibility required for making objects of class “Tract”.

\[
R> \text{ex} = \text{extent(tr3)}
\]
\[
R> \text{rl} = \text{raster(ex, nrows}=34, \text{nrows}=36, \text{crs}=\text{NA})
\]
3 The "bufferedTract" Class

The "bufferedTract" class simply adds a buffer to the tract. The buffer can be thought of most commonly as being internal; that is, it specifies an area within which the individual "Stem" object locations (tree or log centers) should fall in order to contain all of the inclusion zones. In other words, we are assuming that we will be placing stems randomly within this buffer so that there is no eventual slopover of the inclusion zones at the boundary. But beyond this, the buffer can be any width, it is up to the user.

The buffering method is essentially what was proposed by Masuyama (1953). If one wants to think in terms of an external buffer, then simply expand the entire tract by the buffer amount, making the internal portion the area of the internal tract desired.

Here's a look at the class structure, class "bufferedTract" is seen to be a subclass of "Tract"

R> getClass('bufferedTract')
The "Tract" Class... §3 The "bufferedTract" Class

Class "bufferedTract" [package "sampSurf"]

Slots:

Name: bufferRect spBuffer description units
Class: matrix SpatialPolygons character character

Name: area file data legend

Name: history title extent rotated
Class: list character Extent logical

Name: rotation ncols nrows crs
Class: .Rotation integer integer CRS

Name: z
Class: list

Extends:
Class "Tract", directly
Class "RasterLayer", by class "Tract", distance 2
Class "Raster", by class "Tract", distance 3
Class "RLNULL", by class "RasterLayer", distance 3
Class "BasicRaster", by class "Tract", distance 4

3.1 Class slots

The following slots are added to the “Tract” class definition...

- `bufferRect`: A matrix holding the buffer rectangle extents in the form of a bounding box that is used by the `sp` package.
- `spBuffer`: The buffer in polygon form as a “SpatialPolygons” object.

3.2 Object creation

A “bufferedTract” object can be created very simply from a “Tract” object by specifying a reasonable buffer width in the constructor. In the following we make a buffered tract and add several down logs whose center locations fall within the buffer rectangle...
The “Tract” Class...

§3 The “bufferedTract” Class

```r
R> tr = setValues(tr, rep(0, ncell(tr)))
R> btr = bufferedTract(10, tr)
R> dlogs = downLogs(25, btr)

R> plot(btr, bufferColor='navy', lty='dotted', gridColor='grey90')
R> plot(dlogs, add=TRUE)
```

Figure 3: Simple generic “bufferedTract” object with some random logs.

Note in the above that we can make a collection of “downLogs” by simply specifying the number of logs and a “bufferedTract” object, which supplies the bounding box of the buffer area from which to draw the logs.
4  The “mirageTract” Class

The mirage method, like buffering, is another method for correcting for “sloppover” bias, which can happen if inclusion zones extend beyond the tract boundary. It was first proposed by Schmid-Haas (1969), and its unbiasedness properties were subsequently shown by Gregoire (1982). More details on the method can also be found in Gregoire and Valentine (2008). A generalization of the mirage method that also works for Monte Carlo-based (variable surface) estimation methods was recently proposed by Lynch and Gove (2014). It is this latter procedure that is implemented in sampSurf, as it should work for all sampling methods in the package, and the original mirage is simply a special case.

The class structure follows, we see that “mirageTract” a subclass of “Tract”...

R> getClass('mirageTract')

Class "mirageTract" [package "sampSurf"]

Slots:

Name: description units area file
Class: character character numeric .RasterFile

Name: data legend history title
Class: .SingleLayerData .RasterLegend list character

Name: extent rotated rotation ncols
Class: Extent logical .Rotation integer

Name: nrows crs z
Class: integer CRS list

Extends:
Class "Tract", directly
Class "RasterLayer", by class "Tract", distance 2
Class "Raster", by class "Tract", distance 3
Class "RLNULL", by class "RasterLayer", distance 3
Class "BasicRaster", by class "Tract", distance 4

4.1 Class slots

Note that there are no new slots added to the “Tract” class definition.
4.2 Object creation

Objects of class “mirageTract” are created directly from an object of class “Tract”. So, one simply creates the “Tract” object and then...

\begin{verbatim}
R> mtr = mirageTract(tr)
R> class(mtr)
[1] "mirageTract"
attr("package")
[1] "sampSurf"
\end{verbatim}

Where applicable, the routines that develop the sampling surface will detect that a “mirageTract” object was passed in the signature of the call to the appropriate method and the mirage method will be applied automatically.

References


M. Masuyama. A rapid method for estimating basal area in a timber survey—an application of integral geometry to areal sampling problems. *Sankhya*, 12, 1953. 7