The “ArealSampling” Class

Jeffrey H. Gove[*]
Research Forester
USDA Forest Service
Northern Research Station
271 Mast Road
Durham, New Hampshire 03824 USA
e-mail: jgove@fs.fed.us or e-mail: jhgove@unh.edu

Wednesday 20th April, 2011
11:51am

Contents

1 Introduction

The “ArealSampling” class is a virtual class that is used as a basis for each of the possible different areal sampling methods we use in forestry, whether for down logs or standing trees. For each of the subclasses, relevant information defining the sampling method should be given that will allow the computation of its associated inclusion zone later in the “InclusionZone” class. Because most areal sampling methods also depend on the attributes of the “Stem” subclass that represents it (i.e., the inclusion zone for PPS methods especially are of this form), most subclasses will not have any “SpatialPolygons” slot available for rendering the object graphically. One obvious exception is with fixed-radius plots under, e.g., the ‘standup’ method (Gove and Van Deusen, 2011). In addition, since ‘standup,’ ‘chainsaw,’ and ‘sausage’ are simply protocol differences within the fixed-area circular

[*] Phone: (603) 868-7667; Fax: (603) 868-7604.

plot method of sampling, we do not differentiate them here, but wait until the "InclusionZone" class to make that distinction.

An overview of the "ArealSampling" class structure is presented in Figure 1. At this point it is uncertain whether there will be a division between standing tree and down log methods. Such a division is somewhat artificial as some of the methods, such as circular plot sampling, can be used on both, and it would be redundant to have them defined twice. But we can always have a "joint" subclass, encompassing these methods, or just let them hang by themselves as necessary. At this point in the design, none of these changes should impact what is already completed. Furthermore, it should be kept in mind that protocols within sampling methods, such as the "sausage" or "standup" protocols for down coarse woody debris (Gove and Van Deusen, 2011), are not "ArealSampling" methods per se. They could be defined as subclasses of the "circularPlot" class, but they really are characterized by their inclusion zones, and so we leave their definition for the "InclusionZone" class. In any case, if the divisions were to arise in further work, they would be defined as virtual classes, as they are in the "InclusionZone" class.

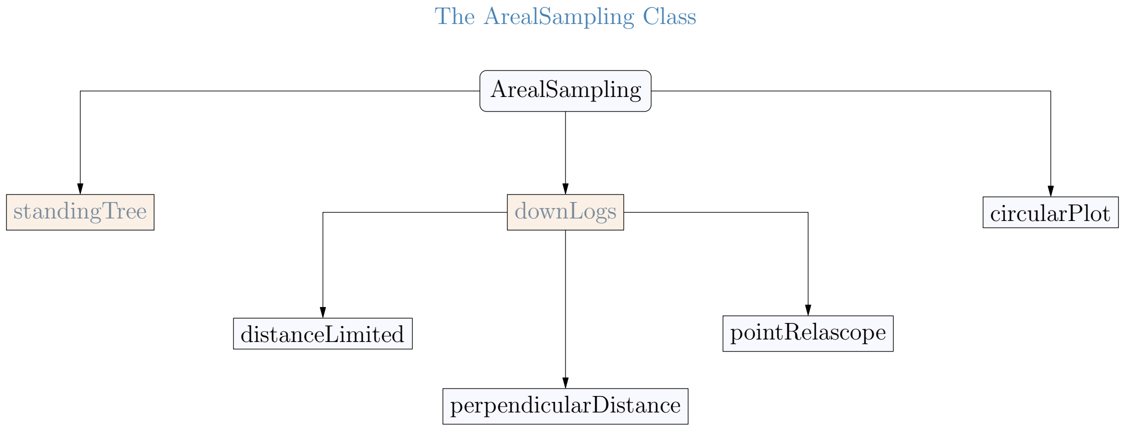

Figure 1: An overview of the "ArealSampling" class.

2 The "ArealSampling" Class

As mentioned above, this is the virtual base class, therefore, we really only care about its slots so we can see what will transfer to subclasses...

```
R> getClass('ArealSampling')
```

```
Virtual Class "ArealSampling" [package "sampSurf"]

Slots:

Name:  description       units
Class:   character   character

Known Subclasses: "circularPlot", "pointRelascope", "perpendicularDistance", "distanceLimited"
```

2.1 Class slots

- *description*: Some descriptive text about the object.
- *units*: A character string specifying the units of measure. Legal values are “English” and “metric.”

3 The “circularPlot” Class

This is a subclass of “ArealSampling”, for fixed-area circular plots. It shares all the slots of the virtual class; in addition, it defines the following other slots. . .

```
R> showClass('circularPlot')
```

```
Class "circularPlot" [package "sampSurf"]

Slots:

Name:           radius            area       perimeter        location
Class:         numeric         numeric SpatialPolygons   SpatialPoints

Name:             spID         spUnits     description           units
Class:       character             CRS       character       character

Extends: "ArealSampling"
```

3.1 “circularPlot” Class slots

The extra slots are defined as follows. . .

- *radius*: The fixed-plot radius in the correct units.
- *area*: The area of the plot in the correct units.
- *perimeter*: The "SpatialPolygons" object corresponding to the perimeter of the fixed-radius plot.
- *location*: This is a "SpatialPoints" representation of the location of the object. In the "circularPlot" class, this is the fixed-radius plot center, which will often correspond to the `location` slot in the "Stem" object under sampling surface simulations. But there are exceptions: for example, under the 'standup' method, it will be at the large-end of the log, while under the 'chainsaw' method, it will be some point within the "sausage" shaped inclusion zone for protocol 1 in (Gove and Van Deusen, 2011).
- *spID*: A unique identifier that will be used in the eventual "SpatialPolygons" representation of the object.
- *spUnits*: A valid string of class "CRS" denoting the spatial units coordinate system (`?CRS` for more information) as in package `sp`.

3.2 Object creation

One can use `new` to create a new object. However, as with other classes defined in `sampSurf`, the class is sufficiently tedious to create this way that a constructor function of the same name is provided. For example...

```
R> cp=circularPlot(37.237, units='English', center=c(x=10,y=3))
R> summary(cp)
```

```
Object of class: circularPlot
------------------------------------------------------------
fixed area circular plot
------------------------------------------------------------

ArealSampling...
  units of measurement:  English

circularPlot...
  radius = 37.237 feet
  area = 4356.1141 square feet (0.1 acres)
  spatial units:  NA
  spatial ID: cp:19h5ynp6
  location (plot center)...
```

```
    x coord:  10
    y coord:  3
  Number of perimeter points: 101 (closed polygon)
```

The arguments for the constructor are detailed in the help page (`?circularPlot`). However, as an example, we see from the above `summary` output that the number of points defining the perimeter of the plot in the "SpatialPolygons" object is given. It is in fact an argument to the constructor so the plot object can be created with as fine a perimeter of points as desired. The result will always be one more point than what is specified for the argument (default is 100 points), as it is necessary to close the polygon by repeating the starting point.

3.3 Plotting the object

The `plot` generic function has also been extended to be able to handle plotting of the objects of the "circularPlot" class. The arguments are again detailed in the help page, but here is a simple example...

```
R> plot(cp, axes=TRUE, showPlotCenter=TRUE, cex=2)
```

In Figure 2, the `cex` argument specifies the size of the symbol for the plot center; other `par` arguments can also be included.

4 The "pointRelascope" Class

This subclass of "ArealSampling" is used for point relascope sampling (Gove et al. 1999, Gove et al. 2001). As usual, it shares all the slots of the virtual class; in addition, it defines the following other slots...

```
R> showClass('pointRelascope')
```

```
Class "pointRelascope" [package "sampSurf"]

Slots:

Name:  angleDegrees angleRadians          phi     slFactor     rwFactor
Class:      numeric      numeric      numeric      numeric      numeric
```

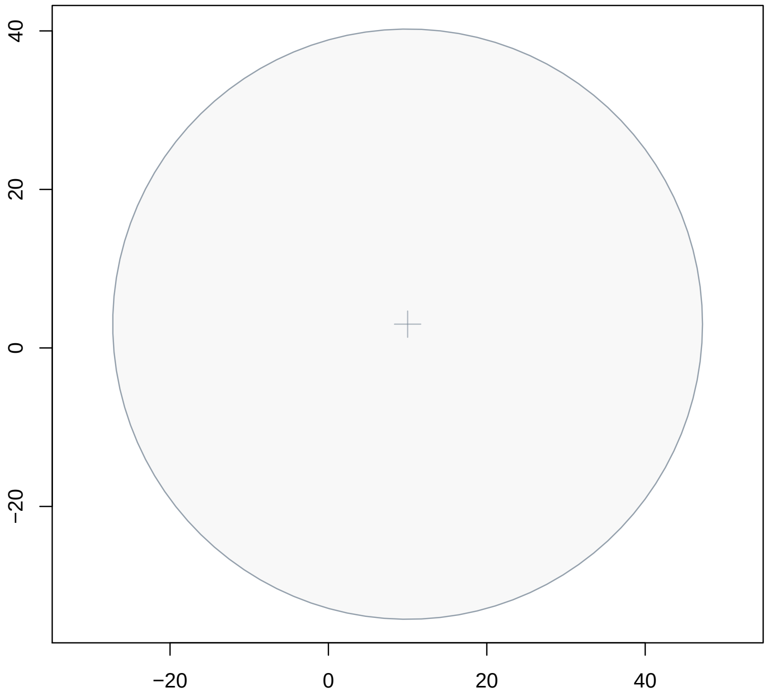

Figure 2: A "circularPlot" object.

```
Name:   description       units
Class:    character   character

Extends: "ArealSampling"
```

4.1 "pointRelascope" Class slots

The extra slots are defined as follows...

- *angleDegrees*: The relascope angle in degrees such that $0 < \nu \leq 90°$.
- *angleRadians*: The relascope angle in radians.
- *phi*: The area factor multiplier, φ, for angle ν, as described in the above references.

- *slFactor*: The squared length factor, $\mathcal{L}$, defining the constant amount of length-square per unit area (acre or hectare) as described in the above references.
- *rwFactor*: The reach:width ratio or factor that makes it simpler to keep track of some of the more useful relascope angles, especially when constructing a relascope.

4.2 Object creation

Once again, one can use `new` to create a new object. However, it is unnessessary and can cause problems if your conversions are not correct. Therefore, a constructor with the same name as the class has been provided; e.g....

```
R> (angle = .StemEnv$rad2Deg(2*atan(.5)))
```

```
[1] 53.130102
```

```
R> prs = pointRelascope(angle, units='English')
R> prs
```

```
Object of class: pointRelascope
------------------------------------------------------------
point relascope method
------------------------------------------------------------

ArealSampling...
  units of measurement:  English

pointRelascope...
  Angle (nu) in degrees = 53.130102
  Angle (nu) in radians = 0.92729522
  PRS area factor (phi) = 2.1049199
  PRS squared-length factor (L) = 20694.374 square feet per acre
  This angle has a 2:1 reach:width factor
```

The first line deduces the angle that exactly (to **R**'s precision) corresponds to the 2:1 reach:width relascope angle. This is subsequently used in the second line to generate an object of the class. Lastly, we see the newly created object's summary.

There is no spatial information in this class, so there is nothing graphical to plot. The graphical inclusion zones will be created when a "pointRelascope" object is coupled with a "downLog" object.

5 The “perpendicularDistance” Class

This subclass of “ArealSampling” is used for perpendicular distance sampling (Williams and Gove 2003, Williams et al. 2005, Ducey et al. 2008). As usual, it shares all the slots of the virtual class; in addition, it defines the following other slots...

```
R> showClass('perpendicularDistance')
```

```
Class "perpendicularDistance" [package "sampSurf"]

Slots:

Name:       factor        kpds description       units
Class:     numeric     numeric   character   character

Extends: "ArealSampling"
```

5.1 “perpendicularDistance” Class slots

The extra slots are defined as follows...

- *factor*: This is the volume, surface area or coverage area factor. At this point, it makes no difference which method we are going to use it for. The only thing this effects is its interpretation in terms of units.
- *kpds*: The perpendicular distance factor K as found in the references. Again, it makes no difference other than the interpretation in terms of units as to which method we are going to apply it at this point.

5.2 Object creation

Once again, one can use `new` to create a new object. However, it is unnessessary and can cause problems if your conversions are not correct. Therefore, a constructor with the same name as the class has been provided. The constructor wants the K PDS factor as the first argument; e.g.,...

```
R> lpds = lapply(c(435.6, 217.8), perpendicularDistance, units='English')
R> sapply(lpds, class)
```

```
[1] "perpendicularDistance" "perpendicularDistance"
```

```
R> t(sapply(lpds, function(x) c(x@factor, x@kpds)))
```

```
     [,1]  [,2]
[1,]   50 435.6
[2,]  100 217.8
```

```
R> (pdsEng = perpendicularDistance(10, units='English'))
```

```
Object of class: perpendicularDistance
------------------------------------------------------------
perpendicular distance method
------------------------------------------------------------

ArealSampling...
  units of measurement:  English

perpendicularDistance...
  kPDS factor = 10 per foot [dimensionless] for volume [surface/coverage area]
  volume [surface/coverage area] factor = 2178 cubic feet [square feet] per acre
```

The first three lines simply make two “perpendicularDistance” objects with different factors and then print a simple table of these. The last line shows how to create single “perpendicularDistance” object, and prints the summary showing the interpretation of the slots.

6 The “distanceLimited” Class

This subclass of “ArealSampling” is used for distance limited PDS (DLPDS) (Ducey et al., 2005) and distance limited Monte Carlo sampling (DLMCS) (in preparation). As usual, it shares all the slots of the virtual class; in addition, it defines the following slot. . .

```
R> showClass('distanceLimited')
```

```
Class "distanceLimited" [package "sampSurf"]
```

```
Slots:

Name:  distanceLimit   description         units
Class:       numeric     character     character

Extends: "ArealSampling", "dlsNumeric"
```

6.1 “distanceLimited” Class slots

The extra slots are defined as follows. . .

- *distanceLimit*: This is simply the design distance limit to be imposed on the sampling method, and hence on the “InclusionZone” object that is created from it.

6.2 Object creation

Once again, using `new` is unnessessary as a constructor with the same name as the class has been provided. The constructor wants the distance limit as the first argument; e.g.,. . .

```
R> distanceLimited(10, units='English')
```

```
Object of class: distanceLimited
------------------------------------------------------------
distance limited method
------------------------------------------------------------

ArealSampling...
  units of measurement:  English

distanceLimited...
  Distance limit = 10 feet
```

References

M. J. Ducey, M. S. Williams, S. Roberge, R. S. Kenning, and J. H. Gove. Distance limited perpendicular distance sampling for coarse woody material: Theory and field results. *Unpublished*, 2005. 9

M. J. Ducey, M. S. Williams, J. H. Gove, and H. T. Valentine. Simultaneous unbiased estimates of multiple downed wood attributes in perpendicular distance sampling. *Canadian Journal of Forest Research*, 38:2044–2051, 2008. 8

J. H. Gove and P. C. Van Deusen. On fixed-area plot sampling for downed coarse woody debris. *Forestry*, 2011. Submitted August 2010. 1, 2, 4

J. H. Gove, A. Ringvall, G. Ståhl, and M. J. Ducey. Point relascope sampling of downed coarse woody debris. *Canadian Journal of Forest Research*, 29(11):1718–1726, 1999. 5

J. H. Gove, M. J. Ducey, A. Ringvall, and G. Ståhl. Point relascope sampling: A new way to assess down coarse woody debris. *Journal of Forestry*, 4:4–11, 2001. 5

M. S. Williams and J. H. Gove. Perpendicular distance sampling: an alternative method for sampling downed coarse woody debris. *Canadian Journal of Forest Research*, 33:1564–1579, 2003. 8

M. S. Williams, M. J. Ducey, and J. H. Gove. Assessing surface area of coarse woody debris with line intersect and perpendicular distance sampling. *Canadian Journal of Forest Research*, 35: 949–960, 2005. 8