Radiation Oncology (RadOnc) Tools

Reid F. Thompson

October 21, 2013

Contents

1 Introduction

The *RadOnc* package provides a number of tools for the import and analysis of dose-volume histogram (DVH) data used routinely in Radiation Oncology clinical practice and research. Supported formats for data import currently include:

- Varian's Aria/Eclipse platform
- DICOM-RT files

The functionality contained herein also enables visualization of dosimetric and volumetric data, and statistical comparison among multiple DVHs and three-dimensional structures. In order to use these tools, you must first load the *RadOnc* package:

```
> library(RadOnc)
```

It is assumed that the reader is already familiar with DVH analysis. If this is not the case, consult the relevant literature for a thorough treatment of the subject (1).

Throughout this vignette, we will be exploring actual data for 2 patients, each possessing a set of 10 structures (including organs at risk and treatment planning volumes). We will also demonstrate rudimentary three-dimensional structural processing.

2 Changes for *RadOnc* in current release

- New function `read.DICOM.RT()` to import 3-dimensional structural information from one or more DICOM-RT files.
- New class `structure3D` to store 3-dimensional information encoding a structure.
- New functions to support interaction with `structure3D` class.
- New class `structure.list` to store a list of `structure3D` objects.
- New functions to support interaction with `structure.list` class.
- New function `compareStructures()` to assess similarities and differences among two or more `structure3D` objects within a `structure.list`.

3 DVH Analysis

3.1 DVH file import

The `read.DVH()` function is designed to take an input text file and output a list of DVH data objects containing all relevant data. Supported file types currently include Varian's Aria/Eclipse platform (v.10 and v.11). Other treatment planning systems are not currently supported however will be included in future releases.

For Varian-specific file types, data must be exported directly from the treatment planning system and should include all DVH structures of interest. In Eclipse, this is accomplished via the "Export DVH in Tabular Format..." option, accessed by right-clicking over DVHs in Plan Evaluation mode. Exported files will adhere to the following form (an example file, 50 lines of which are shown here, is contained within this release of the *RadOnc* package):

```
Patient Name         : Doe, Jane (1111111111)
Patient ID           : 1111111111
Comment              : DVHs for one plan
Date                 : 05.24.2013  00:00:00
Type                 : Cumulative Dose Volume Histogram
Description          : The cumulative DVH displays the percentage (relative)
                       or volume (absolute) of structures that receive a dose
                       equal to or greater than a given dose.

Plan: PLAN_NAME
Prescribed dose [cGy]: 5500.0
% for dose (%): 100.0

Structure: LIVER
Approval Status: Unapproved
Plan: PLAN_NAME
Course: COURSE_1
Volume [cc]: 1635.9
Dose Cover.[%]: 100.0
Sampling Cover.[%]: 100.0
Min Dose [cGy]: 42.7
Max Dose [cGy]: 5634.2
Mean Dose [cGy]: 707.0
Modal Dose [cGy]: 99.5
Median Dose [cGy]: 276.4
STD [cGy]: 917.2
Equiv. Sphere Diam. [cm]: N/A
Conformity Index: N/A
Gradient Measure [cm]: N/A

       Dose [cGy]   Relative dose [%] Ratio of Total Structure Volume [%]
```

```
                0                   0                       100
                5           0.0909091                       100
               10            0.181818                       100
               15            0.272727                       100
               20            0.363636                       100
               25            0.454545                       100
               30            0.545455                       100
               35            0.636364                       100
               40            0.727273                       100
               45            0.818182                   99.9247
               50            0.909091                   99.2638
               55                   1                   98.1752
               60             1.09091                   96.8538
               65             1.18182                   95.3989
               70             1.27273                   93.8907
               75             1.36364                   92.3371
               80             1.45455                   90.7697
               85             1.54545                   89.2061
               90             1.63636                   87.6496
...
...
```

This DVH data may be imported using the `read.DVH()` function, with an example shown here:

```
> read.DVH(file="Jane_Doe.dvh", type="aria10", verbose=TRUE)

Reading DVH file ('/Library/Frameworks/R.framework/Versions/3.0/Resources/library/RadOnc/
  Patient: Doe, Jane (1111111111)
  Plan: PLAN_NAME
  Dose: 5500cGy
  ..Importing structure: LIVER  [volume: 1635.9cc, dose: 42.7 - 5634.2cGy]
  ..Importing structure: LEFT_KIDNEY  [volume: 195.7cc, dose: 75.8 - 3846.8cGy]
  ..Importing structure: STOMACH  [volume: 695.2cc, dose: 59 - 5353.2cGy]
  ..Importing structure: DUODENUM  [volume: 34.2cc, dose: 2707.8 - 5620.1cGy]
  ..Importing structure: RIGHT_KIDNEY  [volume: 223.9cc, dose: 102.4 - 4201.9cGy]
  ..Importing structure: CTV  [volume: 146.7cc, dose: 5168.6 - 5646.9cGy]
  ..Importing structure: PTV  [volume: 239.4cc, dose: 4749.8 - 5664.7cGy]
  ..Importing structure: SMALL_BOWEL  [volume: 232.2cc, dose: 59.6 - 4934.1cGy]
  ..Importing structure: CORD  [volume: 64.9cc, dose: 0 - 3442.8cGy]
  ..Importing structure: BODY  [volume: 25507.5cc, dose: 0 - 5664.7cGy]
```

3.2 DVH list manipulation

The `read.DVH()` function returns a DVH list that can be manipulated in multiple ways. Subsets of DVH lists can be obtained using the `[]` modifier, and any number of DVH lists can be combined using the `c()` function. Additionally, single DVH objects can be directly accessed using the `[[]]` modifier, and individual elements of a DVH list may be directly replaced with other DVH objects using the `[[<-` function.

```
> janedoe[1:4]

[1] "List containing 4 DVH objects (LIVER, LEFT_KIDNEY, STOMACH, DUODENUM)"

> c(janedoe[c("PTV")], johndoe[c("CTV", "DUODENUM")])

[1] "List containing 3 DVH objects (PTV, CTV, DUODENUM)"

> johndoe[["CTV"]]

[1] "Structure: CTV (88.4095 cc), Dose: 96.8-102.6% (5500cGy prescribed), DVH: cumulativ

> janedoe[[1]] <- johndoe[["CTV"]]
> janedoe[1:4]

[1] "List containing 4 DVH objects (CTV, LEFT_KIDNEY, STOMACH, DUODENUM)"
```

Other list processing functions may be applied to DVH lists, enabling further data manipulation. The `rev()` function may be used to reverse the order of a DVH list, while the `names()` function may be used to extract (or set) the structure names for each DVH contained within the list. The `length()` function may be used to find the number of DVH objects contained within a DVH list, and the `lapply()` function can be used to perform a customizable set of operations on a DVH list and return a customizable set of values. Here are some examples employing each of these functions:

```
> names(janedoe)[1:4] <- c("A1", "B2", "C3", "D4")
> names(rev(janedoe[1:4]))

[1] "D4" "C3" "B2" "A1"

> length(johndoe)

[1] 10

> lapply(johndoe, function(DVH) { DVH[c("DMIN", "D50%", "DMAX", "V20%")] })
```

```
$LIVER
           %            %            %           cc 
  0.00000000   0.07174856  92.90000000 185.28500000 

$SMALL_BOWEL
          %           %           %          cc 
 0.00000000  0.06320805 99.80000000 13.94640000 

$DUODENUM
        %         %         %        cc 
  0.00000  81.34012 102.40000  83.10130 

$STOMACH
           %            %            %           cc 
  0.00000000   0.06726741 101.30000000  31.95860000 

$CTV
       %        %        %       cc 
 96.8000 100.0081 102.6000  88.4095 

$PTV
        %         %         %        cc 
 84.10000  99.80143 102.60000 155.73500 

$BODY
           %            %            %           cc 
0.000000e+00 6.202236e-02 1.026000e+02 1.893130e+03 

$LEFT_KIDNEY
          %           %           %          cc 
 0.00000000  0.07279133 44.40000000 18.60640000 

$RIGHT_KIDNEY
       %        %        %       cc 
 0.00000 24.25567 98.50000 85.93270 

$CORD
        %         %         %        cc 
 0.000000  4.078601 55.500000 17.496100 
```

3.3 DVH data

Each DVH structure contains a variety of data related to the structure itself as well as the distribution of radiation dose within the structure volume. Detailed slot list and parameters are described in the `DVH-class` documentation accompanying the *RadOnc* package. Specific parameters can be extracted using the `[]` modifier, which can take as its argument a character string representation of the desired dose/volume parameter. For instance, the volume of duodenum receiving 20Gy or the dose to the top 2.5% (2.3286cc) of the volume can be extracted from DVH data as follows:

```
> johndoe[["DUODENUM"]]["V20Gy"]

     cc
76.5153

> johndoe[["DUODENUM"]]["D2.5%"]

       %
100.5605

> johndoe[["DUODENUM"]]["volume"] * 0.025

      cc
2.328598

> johndoe[["DUODENUM"]]["D2.3286cc"]

       %
100.5605
```

These parameters are entirely flexible and multiple parameters can be requested for a given DVH object at the same time. This functionality can also be applied to a DVH list using the `$` modifier.

```
> johndoe[["DUODENUM"]][c("V5%", "V20Gy", "D2.5%", "D2.3286cc", "Dmax")]

      cc       cc        %        %        %
 90.1912  76.5153 100.5605 100.5605 102.4000

> johndoe[1:4]$"V20Gy,Dmax"

$LIVER
      cc        %
21.35734 92.90000

$SMALL_BOWEL
       cc         %
 9.370469 99.800000
```

```
$DUODENUM
      cc        %
 76.5153 102.4000

$STOMACH
       cc         %
 26.83795 101.30000
```

If an improper parameter is specified however, `NA` results will be returned for the affected parameter(s):

```
> johndoe[["DUODENUM"]][c("V5", "VGy", "volume", 2.5, "", "Dmax")]

    <NA>     <NA>       cc     <NA>     <NA>        %
      NA       NA  93.1439       NA       NA 102.4000
```

3.4 DVH plotting

Individual DVH plots can be generated by the `plot()` function, and may be altered to show dose and/or volume as relative or absolute values with DVH shown as cumulative or differential data.

```
> plot(janedoe[[3]], volume="relative", dose="absolute", type="cumulative")
```

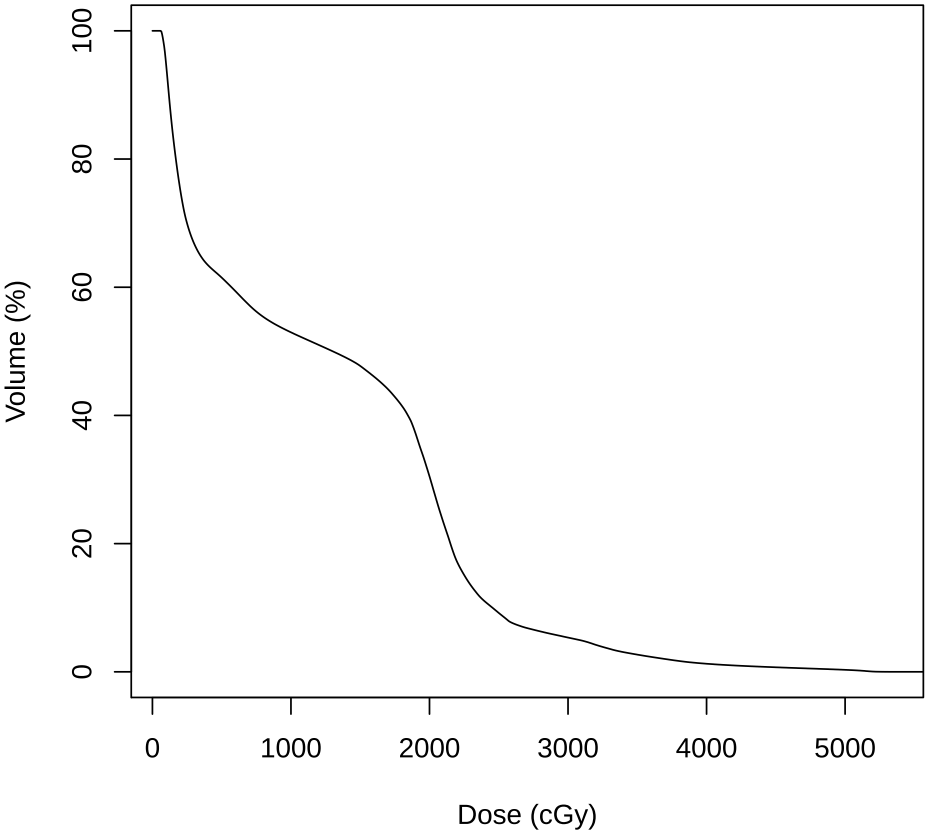

Figure 1: Standard dose-volume histogram for a single structure ("STOMACH") from patient Jane Doe. Data is shown as cumulative dose versus volume.

```
> plot(janedoe[1:3], plot.type="i", col=c("red", "green", "blue"),
+ legend="topright", legend.labels=names(janedoe[1:3]))
```

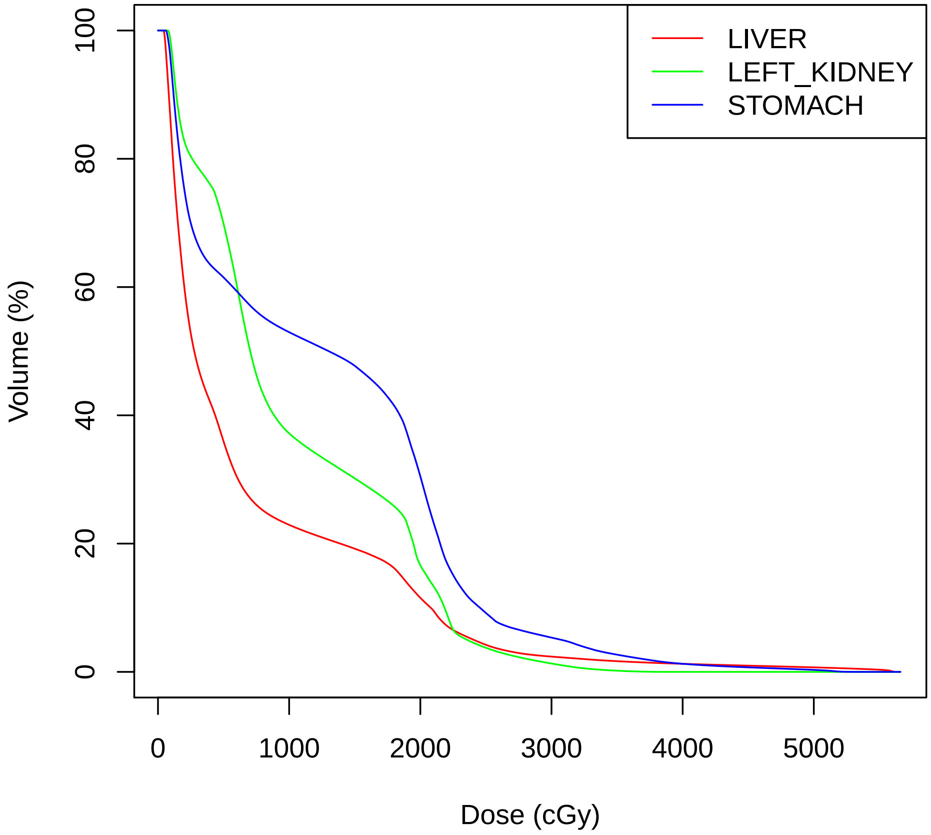

Figure 2: Standard dose-volume histogram for three structures from a single patient, Jane Doe. Data is shown as cumulative dose versus volume. Legend is displayed in the top right corner of the plot.

```
> plot(c(johndoe["STOMACH"],janedoe["STOMACH"]), #group 1
+ c(janedoe["LIVER"],johndoe["LIVER"]), #group 2
+ c(johndoe["DUODENUM"],janedoe["DUODENUM"]), #group 3
+ plot.type="g", dose="relative", col=c("blue", "red", "green"),
+ lwd=2, lty="dashed", fill.lty="solid", fill.transparency=0.3)
```

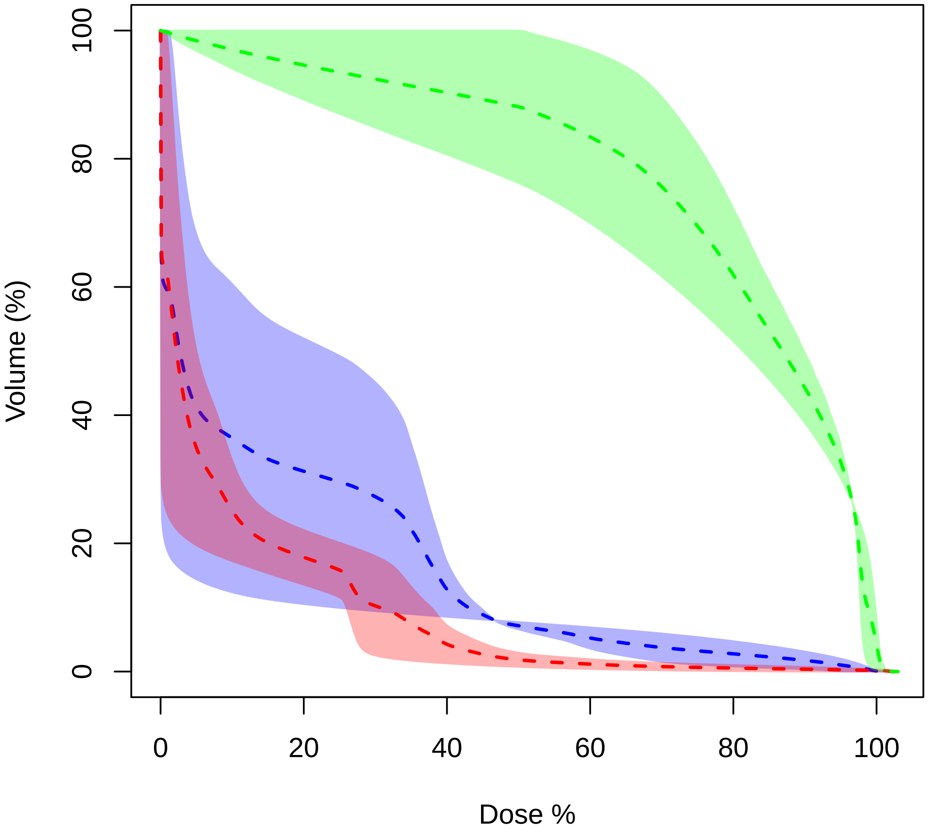

Figure 3: Mean dose-volume histograms are shown for three groups of DVHs, in this case corresponding to stomach, liver, and duodenum from two different patients (John Doe and Jane Doe). Data is shown as cumulative dose (relative) versus volume (relative). Shading represents the range of the data for each group (note that the width of the shading can be specified to represent other parameters instead of range – e.g. variance, standard deviation, interquartile range, median absolute deviation).

```
> group1 <- c("CTV", "PTV")
> group2 <- c("LIVER", "STOMACH", "SMALL_BOWEL")
> plot(c(johndoe[group1],janedoe[group1]),
+ c(janedoe[group2],johndoe[group2]),
+ plot.type="t", main="Target v. OAR t-Test", alpha=0.001,
+ col=c("red", "blue"), lty="dashed", fill.lty="solid")
```

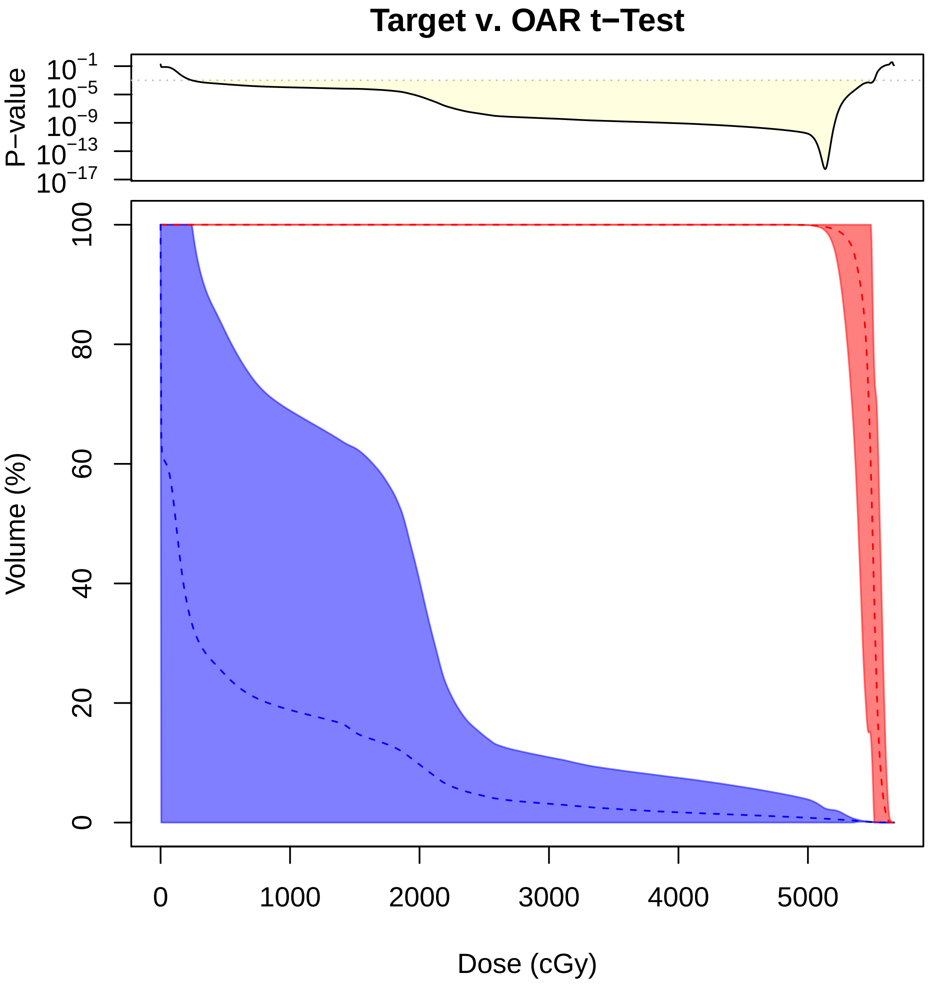

Figure 4: Mean dose-volume histograms are shown for two groups of DVHs, in this case corresponding to CTV/PTV and liver/stomach/small bowel from two different patients (John Doe and Jane Doe). Data is shown as cumulative dose (absolute) versus volume (relative). Shading represents the 99.9% confidence interval for each group (specified here by `alpha=0.001`). The corresponding p-values are shown in the upper panel, with corresponding significance threshold p<0.001.

```
> plot(janedoe[2:9], plot.type="b", volume="abs", dose="rel")
```

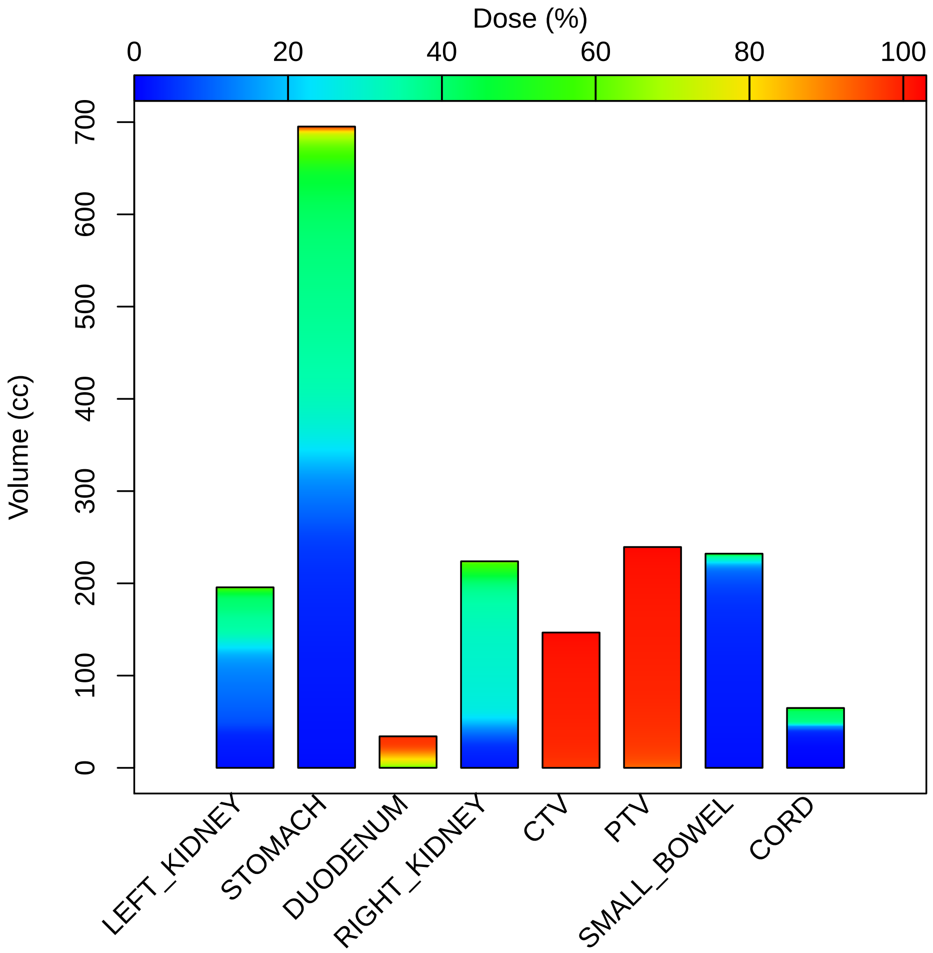

Figure 5: Bar representation of dose distributions for eight structures from a single patient (Jane Doe).

3.5 DVH statistics

Mean or median DVHs can be calculated using the `mean()` and `median()` functions, respectively. These functions take a DVH list as input and return a single object of class `DVH` representing the mean or median dose-volume histogram data calculated from the entire group.

```
> plot(janedoe)
> plot(median(janedoe), new=FALSE, col="red", lwd=2)
> plot(mean(janedoe), new=FALSE, col="blue", lwd=2, lty="dashed")
```

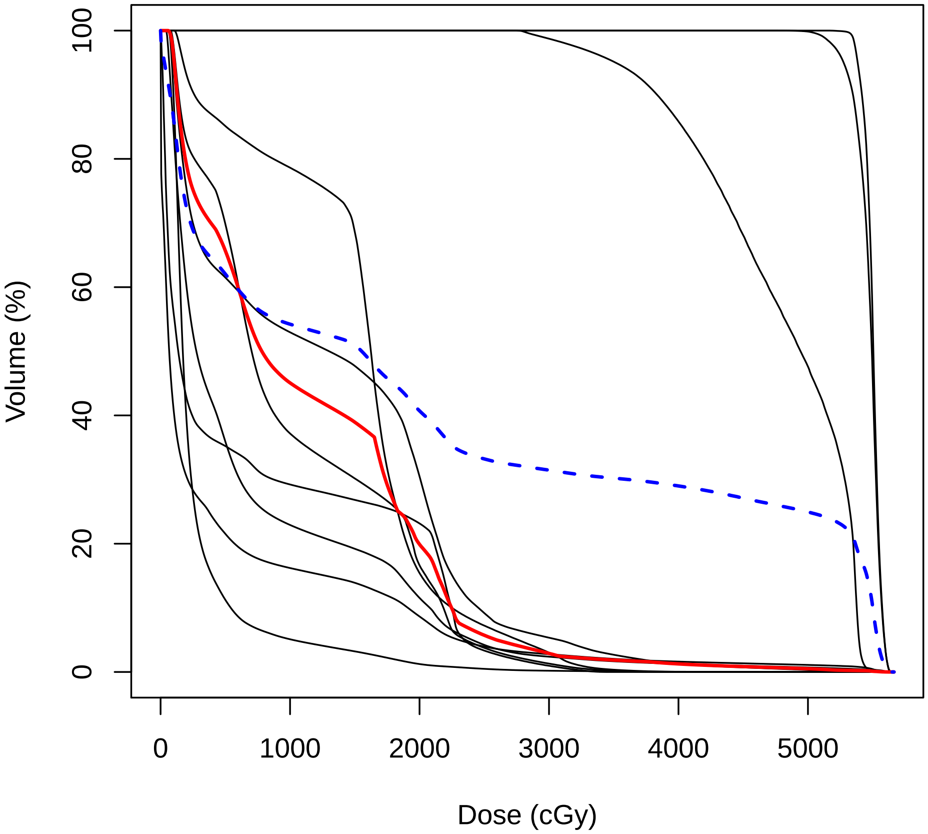

Figure 6: Mean and median DVHs are shown in blue dash and red, respectively.

In routine clinical practice and research, DVH comparisons are often performed at an individual parameter level (e.g. V20Gy from Group A compared to Group B). The *RadOnc* package enables automated comparison throughout the entire DVH. Functions such as `t.test()` and `wilcox.test()` are both enabled for DVH lists.

```
> groupA <- janedoe[c("LIVER","LEFT_KIDNEY","RIGHT_KIDNEY","CORD")]
> groupB <- janedoe[c("CTV", "PTV")]
> t.test(unlist(groupA$"V20Gy"), unlist(groupB$"V20Gy"))

        Welch Two Sample t-test

data:  unlist(groupA$V20Gy) and unlist(groupB$V20Gy)
t = -34.5156, df = 3, p-value = 5.347e-05
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -90.99543 -75.63187
sample estimates:
mean of x mean of y
 16.68635 100.00000
```

```
> AvB <- t.test(groupA, groupB)
> plot(AvB$dose, AvB$p, type="l", log="y", xlab="Dose (cGy)", ylab="p-value")
```

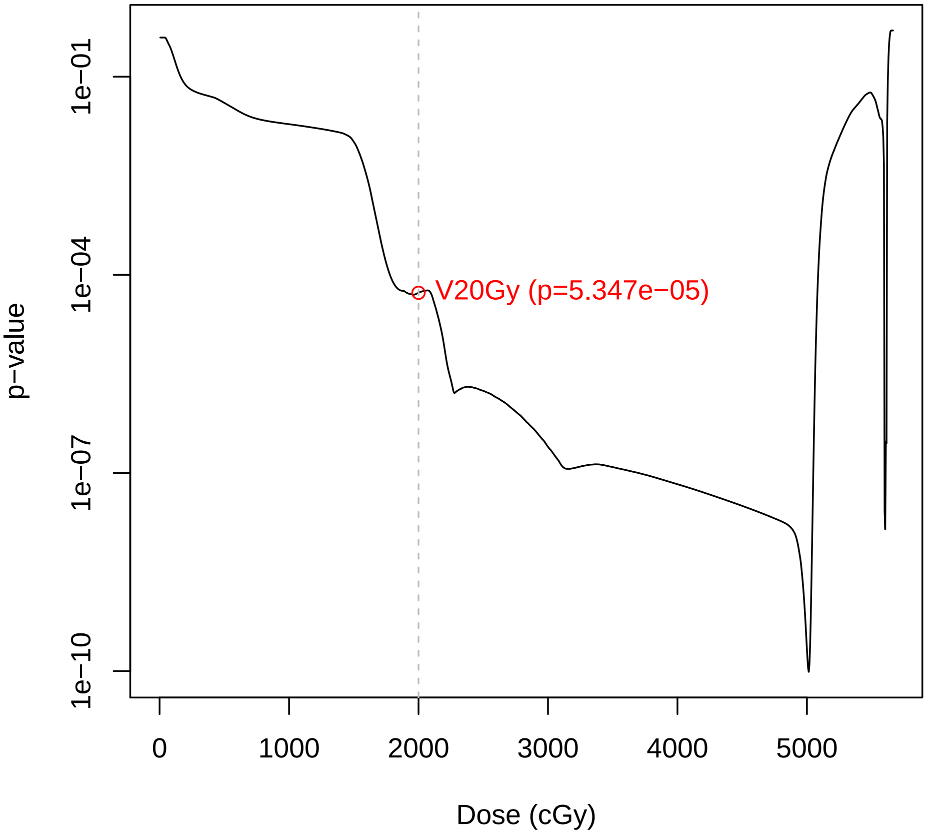

Figure 7: p-values from `t.test()` comparison as a function of dose. V20Gy is highlighted and its p-value corresponds closely to values generated from t-test of V20Gy directly.

4 Three-Dimensional Structure Analysis

4.1 DICOM-RT import

The `read.DICOM.RT()` function is designed to take input DICOM-RT file(s) and output a list of `structure3D` data objects containing all relevant data, in particular the axially-defined contours delineating each structure. Note that DICOM-RT file import was evaluated using Varian's Aria/Eclipse platform (v.10 and v.11). Other treatment planning systems may encode 3D structural information in a different format and this has not been evaluated in the current release of software.

For DICOM-RT data, the associated CT scan must be exported directly from the treatment planning system and should include all contoured structures of interest. In Eclipse, this can be accomplished via the "Export Wizard..." option in the "File" menu, accessed in either Countouring or Plan Evaluation modes. Note that the "Include structure set" option should be selected, and that Institution-specific filters will be required for proper data export. DICOM-RT data will consist of multiple files representing both the CT image as well as the relevant structure set(s).

DICOM-RT data may be imported using the `read.DICOM.RT()` function, with a mockup example shown here (note that "DICOM directory" should be replaced by the path to a specific directory containing the desired DICOM data):

```
> data <- read.DICOM.RT(path="<<DICOM directory>>", verbose=TRUE)
```

The DICOM-RT import process may take some time. We have included pre-loaded data for a single patient (included structures: spinal cord, mandible, teeth) which will be explored in this vignette.

4.2 3D structure manipulation

The `read.DICOM.RT()` function returns a `struct.list` object that can be manipulated in multiple ways. Subsets of structure lists can be obtained using the `[]` modifier, and any number of structure lists can be combined using the `c()` function. Additionally, single `structure3D` objects can be directly accessed using the `[[]]` modifier, and individual elements of a structure list may be directly replaced with other `structure3D` objects using the `[[<-` function.

```
> teeth[1:2]

[1] "List containing 2 structure3D objects (Tooth #1, Tooth #2)"

> c(cord, mandible)

[1] "List containing 2 structure3D objects (Spinal Cord, Mandible)"

> teeth[[1]]

[1] "Structure (Tooth #1) defined by 324 points in 23 axial slices"

> teeth[[1]] <- teeth[["Tooth #3"]]
> teeth

[1] "List containing 3 structure3D objects (Tooth #3, Tooth #2, Tooth #3)"
```

Other list processing functions may be applied to structure lists, enabling further data manipulation. The `rev()` function may be used to reverse the order of a structure list, while the `names()` function may be used to extract (or set) the structure names for each structure contained within the list. The `length()` function may be used to find the number of structures contained within a structure list, and the `lapply()` function can be used to perform a customizable set of operations on a structure list and return a customizable set of values. Here are some examples employing each of these functions:

```
> names(teeth) <- c("Larry", "Curly", "Moe")
> names(rev(teeth[1:3]))

[1] "Moe"   "Curly" "Larry"

> length(teeth)

[1] 3

> lapply(teeth, function(tooth) { range(tooth) })
```

```
$Larry
         x       y    z
min -31.71 -136.93 -111
max -16.82 -122.58  -90

$Curly
         x       y    z
min -31.43 -136.77 -111
max -16.27 -122.43  -90

$Moe
         x       y    z
min -31.28 -136.58 -111
max -16.49 -122.51  -90
```

4.3 Plotting 3D structures

Three dimensional surfaces renderings can be generated by the `plot()` function. The *RadOnc* package does not currently contain the functionality to generate surface triangulations for a given structure, however future releases of the package will implement surface triangulation. Thus, data imported using `read.DICOM.RT()` will not currently be processed for surface triangulation and will generate an empty plot if plotting is attempted. External applications such as MeshLab (2) can be used to generate triangulations which, for advanced users, can be imported into a given `structure3D` object.

```
> plot(mandible)
```

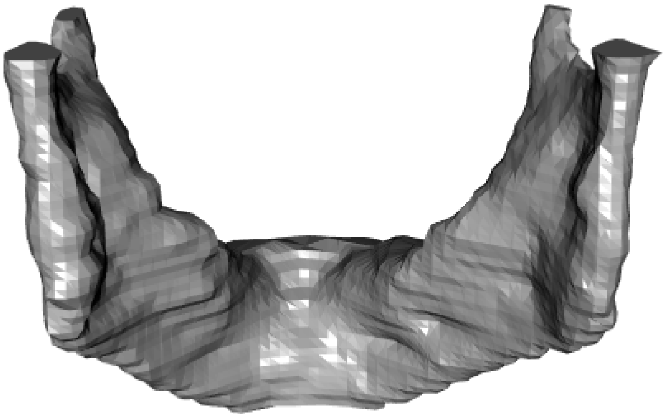

Figure 8: Three-dimensional surface reconstruction from triangulation of physician-contoured points for the mandible of a patient.

```
> plot(cord)
```

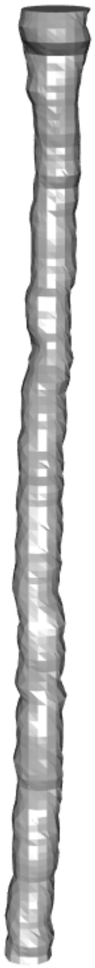

Figure 9: Three-dimensional surface reconstruction from triangulation of physician-contoured points for the spinal cord of a patient.

4.4 Structure comparison

Comparison of three-dimensional structures has numerous applications. In the case presented here, three physicians separately delineated a tooth on axial slices of a CT for a single patient. Variability among physician contours is demonstrated using the `compareStructures()` function:

```
> compareStructures(teeth, method="grid", plot=TRUE)
```

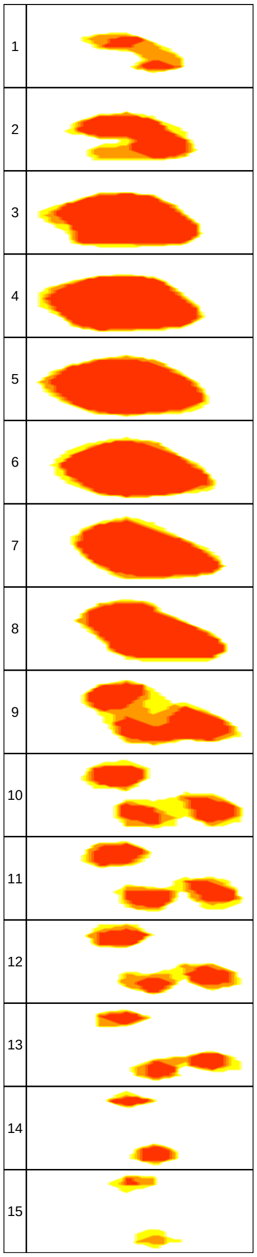

Figure 10: Axial comparison of overlap among three separate physician-defined contours for a single tooth. Red (and white) regions delineate consensus while decreasing degree of overlap is shown in decreasing shades of orange and yellow.

Structure comparison may also be performed by Hausdorff distance (3), which computes the distance between two points clouds, in this case structure surfaces. The absolute Hausdorff distance (`hausdorff.method="absolute"`) yields the maximum distance required to connect any point from one point cloud to its closest neighbor in the other. This metric is highly subject to outliers, thus an aggregate metric is implemented by selecting the average distance (`hausdorff.method="mean"` or `hausdorff.method="median"`) required to connect all points in one point cloud to their closest neighboring points in the other. Note that the Hausdorff distance between two completely superimposable point clouds is zero.

```
> compareStructures(teeth, method="hausdorff", hausdorff.method="mean")

Analyzing structure 1/2 (Tooth #1) ... FINISHED
Analyzing structure 2/2 (Tooth #3) ... FINISHED
          Tooth #1  Tooth #3
Tooth #1 0.0000000 0.4847732
Tooth #3 0.4847732 0.0000000
```

References

[1] R.E. Drzymala, R. Mohan, L. Brewster, J. Chu, M. Goitein, W. Harms, and M. Urie. Dose-volume histograms. *Int J Radiat Oncol Biol Phys*, 21(1):71–78, 1991.

[2] Open Source actively supported by the 3D-CoForm project. Meshlab, 2005. URL `http://meshlab.sourceforge.net`.

[3] Felix Hausdorff. *Grundzuge der Mengenlehre*. Veit and Company, Leipzig, 1914.

A Previous Release Notes

- Initial release (v1.0.0) on 2013-07-07