Package ‘DVHmetrics’

March 26, 2019

Type Package

Title Analyze Dose-Volume Histograms and Check Constraints

Version 0.3.8

Date 2019-03-26

Depends R (>= 3.2)

Imports ggplot2, grid, shiny, KernSmooth, stats, utils

Suggests knitr

VignetteBuilder knitr

LazyData yes

Description Functionality for analyzing dose-volume histograms (DVH) in radiation oncology: Read DVH text files, calculate DVH metrics as well as generalized equivalent uniform dose (gEUD), biologically effective dose (BED), equivalent dose in 2 Gy fractions (EQD2), normal tissue complication probability (NTCP), and tumor control probability (TCP). Show DVH diagrams, check and visualize quality assurance constraints for the DVH. Includes web-based graphical user interface.

License GPL (>= 2)

URL https://github.com/dwoll/DVHmetrics/

NeedsCompilation no

Author Daniel Wollschlaeger [aut, cre], Heiko Karle [aut], Heinz Schmidberger [ctb]

Maintainer Daniel Wollschlaeger <wollschlaeger@uni-mainz.de>

Repository CRAN

Date/Publication 2019-03-26 15:36:03 UTC
Description

Functionality for analyzing dose-volume histograms (DVH) in radiation oncology: Read DVH text files, calculate DVH metrics, gEUD, BED, EQD2, NTCP, TCP, show DVH diagrams, check and visualize quality assurance constraints for the DVH. Includes web-based graphical user interface.

Details

Package: DVHmetrics
Type: Package
Version: 0.3.8
Date: 2019-03-26
License: GPL (>= 2)
Author(s)
Daniel Wollschlaeger and Heiko Karle
Maintainer: Daniel Wollschlaeger <wollschlaeger@uni-mainz.de>

References
For a solution that also reads files in DICOM-RT format, see the RadOnC package: https://CRAN.R-project.org/package=RadOnC.

Examples
```r
dvh <- showDVH(dataMZ[[1]])
checkConstraint(dataMZ, "D1CC < 10Gy")
```

Description
Simultaneously checks one or more quality assurance constraints on one or more DVHs. Reports compliance with each constraint as well as observed difference between linearly interpolated DVH and the given constraints in terms of (relative) dose, (relative) volume, and (relative) minimal Euclidean distance.

Usage
```r
checkConstraint(x, constr, byPat=TRUE, semSign=FALSE,
               sortBy=c("none", "observed", "compliance", "structure",
                        "constraint", "patID", "deltaV", "deltaD",
                        "dstMin", "dstMinRel"),
               interp=c("linear", "spline", "smooth"), ...)
```

```r
## S3 method for class 'DVHs'
checkConstraint(x, constr, byPat=TRUE, semSign=FALSE,
               sortBy=c("none", "observed", "compliance", "structure",
                        "constraint", "patID", "deltaV", "deltaD",
                        "dstMin", "dstMinRel"),
               interp=c("linear", "spline", "smooth"), ...)

## S3 method for class 'DVHLst'
checkConstraint(x, constr, byPat=TRUE, semSign=FALSE,
               sortBy=c("none", "observed", "compliance", "structure",
                        "constraint", "patID", "deltaV", "deltaD",
                        "dstMin", "dstMinRel"),
```
checkConstraint

## S3 method for class 'DVHLstLst'

checkConstraint(x, constr, byPat=TRUE, semSign=FALSE, 
interp=c("linear", "spline", "smooth"), ...
)

### Arguments

x  A single DVH (object of class DVHS), multiple DVHs from one patient/structure (object of class DVHLst), or multiple DVHs from many patients/structures (object of class DVHLstLst). See 

constr  One or more constraints - given as a character vector or as a data.frame. See Details.

byPat  logical. Relevant if multiple DVHs are given. If x has class DVHLst: byPat=TRUE means that the DVHs are for one patient with multiple structures. byPat=FALSE means that the DVHs are for one structure from multiple patients. If x has class DVHLstLst: byPat=TRUE means that the DVHs are for multiple patients (list components of x) with multiple structures. byPat=FALSE means that the DVHs are for multiple structures (list components of x) from multiple patients.

semSign  logical. Meaning of the sign of the observed dose/volume differences between DVHs and constraints. semSign=TRUE means that negative differences indicate constraint compliance, positive differences indicate constraint violations. With semSign=FALSE, the algebraic differences are returned as is.

sortBy  character vector. Sorting criteria for the output data frame.

interp  character. Method of interpolation between DVH points: Linear interpolation using approx, monotone Hermite spline interpolation using spline, or local polynomial regression using locpoly with kernel bandwidth chosen by the direct plug-in method using dpill.

...  Additional parameters passed to getMetric. Use for constraints on EUD (see getEUD for parameter names), TCP (see getTCP), and NTCP (see getNTCP).

### Details

A DVH constraint is a character string that consists of three parts: The DVH metric, the comparison operator (<, >, <=, >=), and the reference value together with the measurement unit. See getMetric for defining a DVH metric, as well as for possible measurement units for dose and volume. For constraints involving the relative dose, the DVH must contain the prescription dose.

Some example constraints are "$V_{10\text{Gy}} > 80\%$" (more than 80% of the structure should have received 10Gy), "$V_{20\%} < 10\text{cc}$" (less than 10cm$^3$ of the structure should have received 20% of the prescription dose), or "$D_{10\text{cc}} > 500\text{cGy}$" (The "hottest" 10cm$^3$ of the structure should have received more than 500cGy).

For constraints on DEUD, DNTCP and DTCP (see getMetric), the reference measurement unit must be Gy, cGy, even though NTCP and TCP are probabilities. Example: "$D_{\text{TCP}} < 0.5\text{Gy}$".
A DVH constraint can apply to a specific patient or to all patients, and to a specific structure or to all structures.

- If constraints apply to all patients/structures in x, constr can be a character vector with elements like the examples above.
- If constraints apply only to some patients/structures, constr must be a data frame with variables constraint, patID and structure. Each row then defines one constraint and its scope: constraint must be a character string with one constraint definition as in the examples above. patID must be either a character string with a valid patient ID or "*" if the the constraint applies to all patients. structure must be either a character string with a valid structure or "*" if the the constraint applies to all structures. If variable patID is missing from the data frame, the constraints apply to all available patients. If variable structure is missing from the data frame, the constraints apply to all available structures. See readConstraint for reading appropriate constraint data.frames from external text files.

For calculating the minimal Euclidean distance between the constraint point and the DVH, the constraint point is orthogonally projected onto each DVH segment between (interpolated) DVH nodes. The relative Euclidean distance is the minimum of these distances divided by the distance of the constraint point to the closer one of both axes (dose and volume).

If volume or dose values outside the range of possible values for a structure are requested, metrics cannot be calculated, and the result will be NA with a warning.

Value

A data frame with details on constraint compliance/ violation.

<table>
<thead>
<tr>
<th>patID</th>
<th>Patient ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>structure</td>
<td>Structure</td>
</tr>
<tr>
<td>constraint</td>
<td>The checked constraint</td>
</tr>
<tr>
<td>observed</td>
<td>The observed value for the metric given in the constraint</td>
</tr>
<tr>
<td>compliance</td>
<td>Does the DVH satisfy the constraint?</td>
</tr>
<tr>
<td>deltaV</td>
<td>Volume difference between constraint and observed DVH (for the constraint dose) in measurement unit specified by constraint</td>
</tr>
<tr>
<td>deltaVpc</td>
<td>Percent volume difference between constraint and observed DVH (for the constraint dose) relative to constraint volume</td>
</tr>
<tr>
<td>deltaD</td>
<td>Dose difference between constraint and observed DVH (for the constraint volume) in measurement unit specified by constraint</td>
</tr>
<tr>
<td>deltaDpc</td>
<td>Percent dose difference between constraint and observed DVH (for the constraint volume) relative to constraint dose</td>
</tr>
<tr>
<td>dstMin</td>
<td>Minimal Euclidean distance between constraint and the cumulative DVH, using linear interpolation</td>
</tr>
<tr>
<td>ptMinD</td>
<td>Dose coordinate of closest point on cumulative DVH to constraint</td>
</tr>
<tr>
<td>ptMinV</td>
<td>Volume coordinate of closest point on cumulative DVH to constraint</td>
</tr>
</tbody>
</table>

See Also

gmetric, getEUD, getNTCP, getTCP, readConstraint, saveConstraint, showConstraint
Examples

```r
res <- checkConstraint(dataMZ, c("D10CC < 10Gy", "V20Gy < 20%"))
head(res)

# define constraints
constr <- data.frame(
  patID=c("P123", "P234"),
  structure=c("HEART", "*"),
  constraint=c("D1CC < 20Gy", "V10% > 8CC"),
  stringsAsFactors=FALSE) # this is important
checkConstraint(dataMZ, constr=constr)
```

---

**convertDVH**  
*Convert between differential and cumulative DVH*

Description

Convert between differential and cumulative DVH as well as between dose units.

Usage

```r
convertDVH(x, toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)
```

```r
## S3 method for class 'matrix'
convertDVH(x, toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)
```

```r
## S3 method for class 'DVHs'
convertDVH(x, toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)
```

```r
## S3 method for class 'DVHLst'
convertDVH(x, toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)
```

```r
## S3 method for class 'DVHLstLst'
convertDVH(x, toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
```

convertDVHsmooth

```r
interp=c("asis", "linear"),
nodes=NULL, rangeD=NULL, perDose=TRUE)
```

**Arguments**

- `x` One DVH (object of class `matrix` or `dvhs`, multiple cumulative DVHs from one patient with multiple structures (object of class `dvhlst`), or multiple cumulative DVHs from many patients, each with multiple structures (object of class `dvhlstlst`). See `readDVH`.
- `toType` character. Convert the DVH to this type. "asis" keeps the current DVH type.
- `toDoseUnit` character. Convert the DVH to this dose unit. "asis" keeps the current dose unit.
- `interp` character. Interpolation method for the cumulative DVH. "asis" for no interpolation and "linear" for linear interpolation.
- `nodes` numeric. Minimum number of nodes to use in linear interpolation. Number of available nodes is kept as is for NULL or if larger than nodes.
- `rangeD` numeric. Dose range for linear interpolation method. If NULL it is determined individually for each DVH.
- `perDose` logical. Are the differential DVH volume values per unit dose?

**Value**

Depending on the input, an object of class `matrix`, `dvhs`, `dvhlst`, or `dvhlstlst`.

**See Also**

`convertDVHsmooth`, `readDVH`, `showDVH`

**Examples**

```r
res <- convertDVH(dataMZ[[c(1, 1)]],
                  toType="cumulative",
                  toDoseUnit="CGY")
```

---

**Description**

Convert between differential and cumulative DVH as well as between dose units, using smoothing of the differential DVH.
Usage

convertDVHsmooth(x,
  toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear", "spline", "ksmooth", "smoothSpl"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)

## S3 method for class 'matrix'
convertDVHsmooth(x,
  toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear", "spline", "ksmooth", "smoothSpl"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)

## S3 method for class 'DVHs'
convertDVHsmooth(x,
  toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear", "spline", "ksmooth", "smoothSpl"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)

## S3 method for class 'DVHList'
convertDVHsmooth(x,
  toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear", "spline", "ksmooth", "smoothSpl"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)

## S3 method for class 'DVHListList'
convertDVHsmooth(x,
  toType=c("asis", "cumulative", "differential"),
  toDoseUnit=c("asis", "GY", "CGY"),
  interp=c("asis", "linear", "spline", "ksmooth", "smoothSpl"),
  nodes=NULL, rangeD=NULL, perDose=TRUE)

Arguments

x One DVH (object of class matrix or DVHs, multiple cumulative DVHs from one patient with multiple structures (object of class DVHList), or multiple cumulative DVHs from many patients, each with multiple structures (object of class DVHListList). See readDVH.

toType character. Convert the DVH to this type. "asis" keeps the current DVH type.

toDoseUnit character. Convert the DVH to this dose unit. "asis" keeps the current dose unit.

interp character. Interpolation method for the differential DVH. "asis" and "linear" for no interpolation. "spline" for spline interpolation using splinefun ("fmm" for differential, "monoh.FC" for cumulative DVHs), "ksmooth" for local polynomial regression using locpoly with kernel bandwidth chosen by the direct
plug-in method using dpill, "smoothSpl" for a smoothing spline using smooth.spline, with the smoothing parameter chosen by generalized crossvalidation.

**Value**

Depending on the input, an object of class matrix, DVHs, DVHLst, or DVHLstLst.

**See Also**

convertDVH, readDVH, showDVH

**Examples**

```r
res <- convertDVHsmooth(dataM[[c(1, 1)]],
                        toType="cumulative",
                        toDoseUnit="CGY")
```

---

**dataConstr**

*Constraint data frame*

**Description**

Data frame with quality assurance constraints to use with built-in DVH object dataM.

**Usage**

```r
data(dataConstr)
```

**Format**

A data frame with 6 entries for the following 3 variables.

- **constraint** The constraint character string.
- **patID** The patient ID character string or * wildcard.
- **structure** The structure character string or * wildcard.

**Details**

See checkConstraint for the definition of a constraint.

**See Also**

readConstraint, checkConstraint, showConstraint
Examples

checkConstraint(dataMZ, constr=dataConstr)

---

dataMZ | DVH data from 3 patients

Description

Data from 3 patients with radiotherapy. DVHs for 7 heart structures.

Usage

data(dataMZ)

Format

Object of class \texttt{DVHLstLst} with 3 components corresponding to 3 patients.

- \texttt{pQRS} Object of class \texttt{DVHLst}. 7 objects of class \texttt{DVHs} for structures AMYOCL (left anterior heart wall), AMYOCR (right anterior heart wall), AOVALVE (aortic valve), AVNODE (AV node), HEART (complete heart), PULMVALVE (pulmonary valve), MYOCARD (heart wall)
- \texttt{pRST} Object of class \texttt{DVHLst}. 7 objects of class \texttt{DVHs} for the same structures as patient \texttt{pQRS}.
- \texttt{pSTU} Object of class \texttt{DVHLst}. 7 objects of class \texttt{DVHs} for the same structures as patient \texttt{pQRS}.

Details

Data courtesy of Department of Radiation Oncology (Prof. Dr. Schmidberger), University Medical Center Mainz, Germany.

See \texttt{readDVH} for classes \texttt{DVHLstLst}, \texttt{DVHLst}, and \texttt{DVHs}.

See Also

\texttt{readDVH, print.DVHs}

Examples

print(dataMZ, verbose=TRUE)
getBED  

Calculate biologically effective dose (BED)

Description

Calculate biologically effective dose (BED) according to the linear-quadratic model.

Usage

getBED(D=NULL, fd=NULL, fn=NULL, ab=NULL)

## Default S3 method:
getBED(D=NULL, fd=NULL, fn=NULL, ab=NULL)

## S3 method for class 'DVHs'
getBED(D=NULL, fd=NULL, fn=NULL, ab=NULL)

## S3 method for class 'DVHLst'
getBED(D=NULL, fd=NULL, fn=NULL, ab=NULL)

## S3 method for class 'DVHLstLst'
getBED(D=NULL, fd=NULL, fn=NULL, ab=NULL)

Arguments

D Default: Total dose. If NULL, fn must be given. Alternative: One cumulative DVH (object of class DVHs, multiple cumulative DVHs from one patient with multiple structures (object of class DVHLst), or multiple cumulative DVHs from many patients, each with multiple structures (object of class DVHLstLst). See readDVH.

fd Fractional dose. If D is some kind of DVH object, only the first element will be used.

fn Number of fractions. If NULL, D must be the total dose. Ignored if D is some kind of DVH object.

ab alpha/beta ratio for the relevant tissue. If some kind of DVH object, only the first element will be used.

Value

Default method: A data frame with variables bed, fractDose, ab.

If D is some kind of DVH object, the same kind of object is returned with the individual dose values converted to BED.

References

See Also
getEQD2, getIsoEffD

Examples
getBED(D=50, fd=2.5, ab=c(2, 3, 4))
getBED(D=dataMZ[[c(1, 1)]], fd=1.8, ab=3)

getDMEAN

**DMEAN and other dose metrics**

Description
Calculate DMEAN and other dose metrics from the (interpolated) differential DVH without relying on the values exported by the TPS.

Usage

```r
getDMEAN(x, interp=c("linear", "spline", "ksmooth", "smoothSpl"),
    nodes=5001L)
```

### S3 method for class 'DVHs'

```r
getDMEAN(x, interp=c("linear", "spline", "ksmooth", "smoothSpl"),
    nodes=5001L)
```

### S3 method for class 'DVHLst'

```r
getDMEAN(x, interp=c("linear", "spline", "ksmooth", "smoothSpl"),
    nodes=5001L)
```

### S3 method for class 'DVHLstLst'

```r
getDMEAN(x, interp=c("linear", "spline", "ksmooth", "smoothSpl"),
    nodes=5001L)
```

Arguments

- **x**: One DVH (object of class DVHs, multiple DVHs from one patient with multiple structures (object of class DVHLst), or multiple DVHs from many patients, each with multiple structures (object of class DVHLstLst). See readDVH.
- **interp**: character. Method of interpolation between DVH points: Linear interpolation applies to the cumulative DVH (recommended). Spline interpolation with splinefun, local polynomial regression with locpoly, and smoothing splines with smooth.spline apply to the differential DVH (not recommended).
- **nodes**: numeric. Minimum number of nodes to use in interpolation. Number of available nodes is kept as is for NULL or if larger than nodes.
getEQD2

Value

A data frame with the following value(s).

- **patID**  Patient ID.
- **structure**  Structure name.
- **doseMin**  Minimum dose.
- **doseMax**  Maximum dose.
- **doseAvg**  Mean dose.
- **doseMed**  Median dose.
- **doseSD**  Dose standard deviation.
- **doseMode**  Dose mode.
- **doseAvgTPS**  Mean dose as exported from the TPS (if available).
- **doseMedTPS**  Median dose as exported from the TPS (if available).
- **doseMinTPS**  Minimum dose as exported from the TPS (if available).
- **doseMaxTPS**  Maximum dose as exported from the TPS (if available).

See Also

getmetric, convertDVHsmooth, approxfun, splinefun, smooth.spline, dpill, locpoly

Examples

```r
getEQD2(doseMedian, interp="linear")
```

### Description

Calculate dose in 2Gy fractions biologically equivalent dose according to the linear-quadratic model, assuming a homogeneous dose distribution within the volume.

#### Usage

```r
getEQD2(D=NULL, fd=NULL, fn=NULL, ab=NULL)
```

```
## Default S3 method:
getEQD2(D=NULL, fd=NULL, fn=NULL, ab=NULL)
```

```
## S3 method for class 'DVHs'
getEQD2(D=NULL, fd=NULL, fn=NULL, ab=NULL)
```

```
## S3 method for class 'DVHlst'
getEQD2(D=NULL, fd=NULL, fn=NULL, ab=NULL)
```

```
## S3 method for class 'DVHlstLst'
getEQD2(D=NULL, fd=NULL, fn=NULL, ab=NULL)
```
**getEqs2**

Arguments

- **d**: Default: Total dose. If NULL, fn must be given. Alternative: One cumulative DVH (object of class `dvhs`), multiple cumulative DVHs from one patient with multiple structures (object of class `dvhls`), or multiple cumulative DVHs from many patients, each with multiple structures (object of class `dvhlstlst`). See `readDvh`.
- **fd**: Fractional dose. If d is some kind of DVH object, only the first element will be used.
- **fn**: Number of fractions. If NULL, d must be given. Ignored if d is some kind of DVH object.
- **ab**: alpha/beta ratio for the relevant tissue. If d is some kind of DVH object, only the first element will be used.

Details

EQD2 is a special case of isoeffective dose calculation with fractional dose $dR/R$, see `getIsoEffD`. The calculation assumes a homogeneous dose distribution within the volume.

Value

Default method: A data frame with variables `EQD2`, `fractDose`, `ab`.

If d is some kind of DVH object, the same kind of object is returned with the individual dose values converted to EQD2.

References


See Also

`getBED`, `getIsoEffD`

Examples

```
getEqs2(d=50, fd=2.5, ab=c(2, 3, 4))
getEqs2(dataMZ[[c(1, 1)]], fd=1.8, ab=3)
```
getEUD

*Generalized equivalent uniform dose (gEUD)*

**Description**

Calculate generalized equivalent uniform dose (gEUD). May be based on EQD2.

**Usage**

```r
getEUD(x, EUDa, EUDfd=NULL, EUDab=NULL, ...)  
```

## S3 method for class 'DVHs'

```r
getEUD(x, EUDa, EUDfd=NULL, EUDab=NULL, ...)  
```

## S3 method for class 'DVHLst'

```r
getEUD(x, EUDa, EUDfd=NULL, EUDab=NULL, ...)  
```

## S3 method for class 'DVHLstLst'

```r
getEUD(x, EUDa, EUDfd=NULL, EUDab=NULL, ...)  
```

**Arguments**

**x**  
One cumulative DVH (object of class `DVHs`, multiple cumulative DVHs from one patient with multiple structures (object of class `DVHLst`), or multiple cumulative DVHs from many patients, each with multiple structures (object of class `DVHLstLst`). See `readDVH`.

**EUDa**  
Exponential parameter a.

**EUDfd**  
If gEUD should be based on EQD2: Fraction dose.

**EUDab**  
If gEUD should be based on EQD2: alpha/beta ratio for the relevant tissue.

**...**  
Ignored. Used to catch additional arguments passed from `getMetric`.

**Value**

A data frame with variables EUD, patID, and structure.

**References**


**See Also**

`getEQD2`, `getMetric`
**Examples**

```r
getEUD(dataM2[[1]], EUDa=2)

# based on EQD2
getEUD(dataM2[[1]], EUDa=2, EUDfd=1.8, EUDab=4)
```

---

**getIsoEffD**  
*Isoeffective dose calculation*

---

**Description**

Convert given (fractional) dose into a corresponding (fractional) dose for a different total dose / fractionation schedule according to the linear-quadratic model.

**Usage**

```r
getIsoEffD(D1=NULL, D2=NULL, fd1=NULL, fd2=NULL, ab=NULL)

## Default S3 method:
getIsoEffD(D1=NULL, D2=NULL, fd1=NULL, fd2=NULL, ab=NULL)

## S3 method for class 'DVHs'
getIsoEffD(D1=NULL, D2=NULL, fd1=NULL, fd2=NULL, ab=NULL)

## S3 method for class 'DVHlst'
getIsoEffD(D1=NULL, D2=NULL, fd1=NULL, fd2=NULL, ab=NULL)

## S3 method for class 'DVHlstlst'
getIsoEffD(D1=NULL, D2=NULL, fd1=NULL, fd2=NULL, ab=NULL)
```

**Arguments**

- **D1**
  - Default: numeric vector. Total dose 1. Alternative: One cumulative DVH (object of class DVHs, multiple cumulative DVHs from one patient with multiple structures (object of class DVHlst), or multiple cumulative DVHs from many patients, each with multiple structures (object of class DVHlstlst). See `readDVH`.

- **D2**
  - numeric vector. Total dose 2. Ignored if D is some kind of DVH object.

- **fd1**
  - numeric vector. Fractional dose 1. If D is some kind of DVH object, only the first element will be used.

- **fd2**
  - numeric vector. Fractional dose 2. If D is some kind of DVH object, only the first element will be used.

- **ab**
  - numeric vector. alpha/beta ratio for the relevant tissue in the linear-quadratic model. If D is some kind of DVH object, only the first element will be used.
getMeanDVH

Details

DVT methods: Calculate D2 based on D1, fd1, fd2, and ab. The default method can also calculate
fd2 based on D1, D2, fd1, and ab.

Value

The (vector of) corresponding (fractional) dose value(s). If D is some kind of DVH object, the same
kind of object is returned with the individual dose values converted to D2.

References

IAEA-TR 461). Vienna, Austria: IAEA (International Atomic Energy Agency) and ICRU (International
Commission on Radiation Units and Measurements).

See Also

getBED, getEQD2

Examples

# reference: 70Gy in 2Gy fractions
# new fractionation: 3Gy fractions
# calculate corresponding dose
(D2 <- getIsoEffD(D1=70, fd1=2, fd2=3, ab=c(3.5, 10)))

getIsoEffD(D1=dataMZ[[c(1, 1)]], fd1=1.8, fd2=2, ab=3.5)

getMeanDVH

Point-wise mean DVH with point-wise SDs

Description

Returns the point-wise mean and median DVH with the point-wise standard deviation for a given
list of input DVHs. Other point-wise measures may be calculated as well.

Usage

gemeanDVH(x, fun=list(mean=mean, median=median, sd=sd),
   cumul=TRUE, thin=1, byPat=TRUE, patID=NULL, structure=NULL,
   fixed=TRUE)

## S3 method for class 'DVHs'
gemeanDVH(x, fun=list(mean=mean, median=median, sd=sd),
   cumul=TRUE, thin=1, byPat=TRUE, patID=NULL, structure=NULL,
   fixed=TRUE)

## S3 method for class 'DVHLst'
getMeanDVH(x, fun=list(mean=mean, median=median, sd=sd),
cumul=TRUE, thin=1, byPat=TRUE, patID=NULL, structure=NULL,
fixed=TRUE)

## S3 method for class 'DVHLstLst'
getMeanDVH(x, fun=list(mean=mean, median=median, sd=sd),
cumul=TRUE, thin=1, byPat=TRUE, patID=NULL, structure=NULL,
fixed=TRUE)

**Arguments**

- **x**: A single DVH (object of class `dvhs`), multiple DVHs from one patient/structure (object of class `dvhlst`), or multiple DVHs from many patients/structures (object of class `dvhlstlst`). See `readDVH`.

- **fun**: Named list of functions that should be applied to yield 1 point-wise DVH measure. Functions must have exactly 1 return value.

- **cumul**: logical. Get point-wise mean and SD for cumulative or differential (per unit dose) DVH?

- **thin**: numeric. The number of DVH nodes (dose values) is reduced by 1/thin of the maximum number of nodes in x before interpolating and averaging.

- **byPat**: logical. Relevant if multiple DVHs are given. byPat=TRUE means that for each patient, DVHs for multiple structures are averaged point wise. byPat=FALSE means that for each structure, DVHs for multiple patients averaged point wise.

- **patID**: character vector. Include DVHs for these patients only when calculating mean/SD. If missing, all patients are used. Can be a regular expression with fixed=FALSE, see `regex`.

- **structure**: character vector. Include DVHs for these structures only when calculating mean/SD. If missing, all structures are used. Can be a regular expression with fixed=FALSE, see `regex`.

- **fixed**: logical. Use fixed=FALSE for regular expression matching of patID and structure.

**Details**

Before calculating the point-wise mean and SD, DVHs in x are first linearly interpolated with `convertDVH` using the same set of nodes.

**Value**

Returns a data frame with point-wise mean DVH averaged over structures (byPat=TRUE) or over patients (byPat=FALSE) including the point-wise standard deviation or other measures as controlled by fun.

**See Also**

`showDVH`, `convertDVH`
**Examples**

```r
res1 <- getMeanDVH(dataMZ, byPat=TRUE, structure=c("HEART", "AMYOCL"))
head(res1)

# average differential DVHs
# matches patients P123 and P234
res2 <- getMeanDVH(dataMZ, fun=list(min=min, max=max),
                   cumul=FALSE, byPat=FALSE,
                   patID="23", fixed=FALSE)
head(res2)
```

---

**getMetric**

*Calculate dose-volume-histogram metrics*

**Description**

Simultaneously calculates multiple metrics for multiple cumulative DVHs.

**Usage**

```r
getMetric(x, metric, patID, structure,
         sortBy=c("none", "observed", "patID", "structure", "metric"),
         splitBy=c("none", "patID", "structure", "metric"),
         interp=c("linear", "spline", "ksmooth"), fixed=TRUE, ...)"
```

```r
## S3 method for class 'DVHs'
getMetric(x, metric, patID, structure,
         sortBy=c("none", "observed", "patID", "structure", "metric"),
         splitBy=c("none", "patID", "structure", "metric"),
         interp=c("linear", "spline", "ksmooth"), fixed=TRUE, ...)"
```

```r
## S3 method for class 'DVHlst'
getMetric(x, metric, patID, structure,
         sortBy=c("none", "observed", "patID", "structure", "metric"),
         splitBy=c("none", "patID", "structure", "metric"),
         interp=c("linear", "spline", "ksmooth"), fixed=TRUE, ...)"
```

```r
## S3 method for class 'DVHlstLst'
getMetric(x, metric, patID, structure,
         sortBy=c("none", "observed", "patID", "structure", "metric"),
         splitBy=c("none", "patID", "structure", "metric"),
         interp=c("linear", "spline", "ksmooth"), fixed=TRUE, ...)"
```

**Arguments**

- **x**
  
  One cumulative DVH (object of class DVHs, multiple cumulative DVHs from one patient with multiple structures (object of class DVHlst), or multiple cumulative DVHs from many patients, each with multiple structures (object of class DVHlstLst). See `readDVH`. 
getMetric

**metric** character vector defining one or more DVH metrics. See Details for their definition. For metrics involving the relative dose, the DVH must contain the prescription dose.

**patID** character vector. Calculate given DVH metrics for these patients only. If missing, DVH metrics are calculated for all patients. Can be a regular expression if additional argument `fixed=FALSE` is supplied as well, see `regex`.

**structure** character vector. Calculate given DVH metrics for these structures only. If missing, DVH metrics are calculated for all structures. Can be a regular expression if additional argument `fixed=FALSE` is supplied as well, see `regex`.

**sortBy** character vector giving the sorting criteria for the output data frame.

**splitBy** character vector. Split results into a list of data frames where list components are defined by groups from combining these variables.

**interp** character. Method of interpolation between DVH points: Linear interpolation using `approx`, monotone Hermite spline interpolation using `splinefun`, or local polynomial regression using `locpoly` with kernel bandwidth chosen by the direct plug-in method using `dpill`.

**fixed** logical. Use `fixed=FALSE` for regular expression matching of `patID` and `structure`.

... Further arguments passed to `getEUD` (for metric="DEUD"), `getTCP` (for metric="DTCP"), or `getNTCP` (for metric="DNTCP").

### Details

A *pre-specified* DVH metric is one of the following character strings:

- "DMEAN": The volume-weighted mean dose of the structure.
- "DMEDIAN": Median dose, equal to D50%
- "DMIN": The minimum dose of the non-zero-dose voxels in the structure.
- "DMAX": The maximum dose of the non-zero-dose voxels in the structure.
- "DSD": The standard deviation of the dose in the structure.
- "DRX": The prescription dose.
- "DHI": The Homogeneity Index according to ICRU 83: (D2%-D98%)/D50%.
- "DEUD": The generalized equivalent uniform dose (gEUD). See `getEUD` for mandatory and optional parameters.
- "DNTCP": The normal tissue complication probability (NTCP). See `getNTCP` for mandatory and optional parameters.
- "DTCP": The tumor control probability (TCP). See `getNTCP` for mandatory and optional parameters.

A *free* DVH metric is a character string which has three mandatory elements and one optional element in the following order (AAPM TG263 2018, section 9.2, note that complementary / cold metrics are not yet implemented):

- 1st letter "D" or "V": "D" If the requested value is a dose, "V" if it is a volume.
• 2nd element <number>: If the first letter is "D", this gives the volume for which the dose value of the cumulative DVH should be reported. If the first letter is "V", this gives the dose for which the volume value of the cumulative DVH should be reported.
• 3rd element <measurement unit>: The measurement unit for the 2nd element of the metric. Absolute volumes are indicated by "cc" for cubic centimeter, relative volumes by "%". Absolute doses are indicated by "Gy" for Gray, "cGy" for Centigray, or "eV/g" for uncalibrated dose in DVHs exported by PRIMO. Relative doses are indicated by "%".
• Optional 4th element _<measurement unit>: The measurement unit of the output value. Possible units are as for the 3rd element. If missing, dose is reported as absolute dose in the measurement unit used in the DVH. Volume is reported as relative volume in %.

Examples:
• "D1%": Minimal absolute dose for the "hottest" 1% of the structure, i.e., the maximally irradiated 1% of the structure was exposed to at least this absolute dose.
• "D1cc_%": Minimal relative dose (% of prescription dose) for the maximally irradiated cm^3 of the structure.
• "V500cGy": Relative structure volume in % that was exposed to at least 500cGy.
• "V100_%cc": Absolute structure volume in cm^3 that was exposed to at least 10% of prescription dose.

If volume or dose values outside the range of possible values for a structure are requested, metrics cannot be calculated, and the result will be NA with a warning.
DMEAN, DMEDIAN, DMIN, DMAX, DSD are taken from the exported DVH if present. Otherwise, the differential DVH is generated and used for calculating these metrics.

Value
A data frame or a list with details on the calculated metrics.

<table>
<thead>
<tr>
<th>patID</th>
<th>Patient ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>structure</td>
<td>Structure</td>
</tr>
<tr>
<td>metric</td>
<td>The calculated DVH metric</td>
</tr>
<tr>
<td>observed</td>
<td>The observed value for the DVH metric</td>
</tr>
</tbody>
</table>

References
Rancati et al. (2004). Fitting late rectal bleeding data using different NTCP models: results from an Italian multi-centric study (AIROPROS0101). Radiotherapy Oncology, 73, 21-32.

See Also
saveMetric, getEUD, getNTCP, getTCP, getEQD2, approxfun, splinefun, dpill, locpoly
Examples

getMetric(dataMZ, c("D1CC", "V10%_CC"),
  sortBy=c("metric", "structure", "observed"))

# matching patients are P123 and P234
# matching structures are AMYOCL and AMYOCR
getMetric(dataMZ, c("D1CC", "V10%_CC"),
  patID="23",
  structure=c("AMYOC", "VALVE"),
  splitBy="patID",
  fixed=FALSE)

# gEUD with a=2
getMetric(dataMZ[[c(1, 1)]], "DEUD", EUDa=2)

# gEUD based on EQD2 with a=2, 20 fractions
getMetric(dataMZ[[c(1, 1)]], "DEUD", EUDa=2, EUDfd=1.8)

# NTCP Lyman probit model with TD50=20, m=4, n=0.5
getMetric(dataMZ[[c(1, 1)]], "DNTCP",
  NTCPtd50=20, NTCPm=4, NTCPn=0.5, NTCPtype="probit")

getNTCP

Normal tissue complication probability (NTCP)

Description

Calculate normal tissue complication probability (NTCP) from Lyman’s probit model, Niemierko’s logit model, or the Poisson model. May be based on EQD2.

Usage

getNTCP(x, NTCPtd50=NULL, NTCPm=NULL, NTCPn=NULL, NTCPgamma50=NULL,
  EUDa=NULL, EUDfn=NULL, EUDab=NULL,
  NTCPtype=c("probit", "logit", "poisson"), ...)

## S3 method for class 'DVHS'
getNTCP(x, NTCPtd50=NULL, NTCPm=NULL, NTCPn=NULL, NTCPgamma50=NULL,
  EUDa=NULL, EUDfn=NULL, EUDab=NULL,
  NTCPtype=c("probit", "logit", "poisson"), ...)

## S3 method for class 'DVHLst'
getNTCP(x, NTCPtd50=NULL, NTCPm=NULL, NTCPn=NULL, NTCPgamma50=NULL,
  EUDa=NULL, EUDfn=NULL, EUDab=NULL,
  NTCPtype=c("probit", "logit", "poisson"), ...)

## S3 method for class 'DVHLstLst'
getNTCP(x, NTCPtd50=NULL, NTCPm=NULL, NTCPn=NULL, NTCPgamma50=NULL,
Arguments

- **x**: One cumulative DVH (object of class `dvhs`), multiple cumulative DVHs from one patient with multiple structures (object of class `dvhlst`), or multiple cumulative DVHs from many patients, each with multiple structures (object of class `dvhlstlst`). See `readDvh`.

- **NTCPtd50**: Tolerance dose with 50% complication probability.

- **NTCPm**: Probit/logit Parameter m. Equal to 1 / (NTCPgamma50*sqrt(2*pi)).

- **NTCPn**: Parameter n. Equal to 1/a with exponential gEUD parameter a.

- **NTCPgamma50**: Poisson parameter gamma50. Equal to 1 / (NTCPm*sqrt(2*pi))

- **EUDA**: If gEUD should be based on EQD2: Exponential parameter a.

- **EUDf**: If gEUD should be based on EQD2: Number of fractions.

- **EUDab**: If gEUD should be based on EQD2: alpha/beta ratio for the relevant tissue.

- **NTCPtype**: "probit" - Lyman probit model, "logit" - Niemierko logit model, "poisson" - Poisson model.

- **...**: Ignored. Used to catch additional arguments passed from `getMetric`.

Details

For DVH reduction, gEUD is used. This is equivalent to the Kutcher-Burman DVH reduction scheme. The probit model is given in equation (1), the logit model in equation (2), and the Poisson model in equation (3) in Kaellman (1992), with gEUD plugged in for D.

Value

A data frame with variables `NTCP`, `patID`, and `structure`.

References


- Rancati et al. (2004). Fitting late rectal bleeding data using different NTCP models: results from an Italian multi-centric study (AIROPROS0101). Radiotherapy Oncology, 73, 21-32.
See Also

getTCP, getEUD, getMetric

Examples

getTCP(dataMZ[[1]],
      NTCPtd50=40, NTCPm=0.6, NTCPn=0.5, NTCPtype="probit")

Description

Calculate tumor control probability (TCP) from Lyman’s probit model, Niemierko’s logit model, or Kaellman’s Poisson model. May be based on EQD2.

Usage

getTCP(x, TCPtd50=NULL, TCPm=NULL, TCPn=NULL, TCPgamma50=NULL,
       EUDA=NULL, EUDfn=NULL, EUDA=NULL,
       TCPtype=c("probit", "logit", "poisson"), ...)

Arguments

- **x**: One cumulative DVH (object of class `dvhs`), multiple cumulative DVHs from one patient with multiple structures (object of class `dvhlst`), or multiple cumulative DVHs from many patients, each with multiple structures (object of class `dvhlstlst`). See `readdvh`.
- **TCPtd50**: Tolerance dose with 50% tumor control probability.
- **TCPm**: Probit/logit Parameter m. Equal to 1 / (NTCPgamma50*sqrt(2*pi)).
- **TCPn**: Parameter n. Equal to 1/a with exponential gEUD parameter a.
- **TCPgamma50**: Poisson parameter gamma50. Equal to 1 / (NTCPm*sqrt(2*pi))
- **EUDA**: If gEUD should be based on EQD2: Exponential parameter a.
- **EUDfn**: If gEUD should be based on EQD2: Number of fractions.
- **EUDab**: If gEUD should be based on EQD2: alpha/beta ratio for the relevant tissue.
- **TCPtype**: "probit" - Lyman probit model, "logit" - Niemierko logit model, "poisson" - Kaellman Poisson (relative seriality) model.
- **...**: Ignored. Used to catch additional arguments passed from `getMetric`.

Details

For DVH reduction, gEUD is used. This is equivalent to the Kutcher-Burman DVH reduction scheme.
A data frame with variables TCP, patID, and structure.

References


Rancati et al. (2004). Fitting late rectal bleeding data using different NTCP models: results from an Italian multi-centric study (AIROPROS0101). Radiotherapy Oncology, 73, 21-32.

See Also

getNTCP, getEUD, getMetric

Examples

getTCP(dataM[[1]],
   TCPtcd50=40, TCPm=0.6, TCPn=0.5, TCPTtype="probit")

mergeDVH

Merge existing DVH objects

Description

Combine several existing DVH objects into one object.

Usage

mergeDVH(...)

Arguments

... DVHLstLst objects.

Details

The first object determines whether the resulting object is organized by patient or by structure. Objects need not originally come from the same treatment planning system.
Value

Returns an object of class DVHLstLst.

Examples

## Not run:
# pick some DVH files interactively
a <- readDVH(type="Cadplan")

# pick other DVH files interactively
b <- readDVH(type="Eclipse")

# combine DVH data
res <- mergeDVH(a, b)
res

## End(Not run)

print.DVHs

Print basic information about one or more DVHs

Description

Print basic information (patients, structures, dose range) about one or more DVHs.

Usage

```r
## S3 method for class 'DVHs'
print(x, ...)

## S3 method for class 'DVHLst'
print(x, ...)

## S3 method for class 'DVHLstLst'
print(x, ...)
```

Arguments

- **x**
  - A single DVH (object of class DVHs), multiple DVHs from one patient/structure (object of class DVHLst), or multiple DVHs from many patients/structures (object of class DVHLstLst). See `readDVH`.

- **...**
  - Further arguments: `print.DVHLst(x, verbose=TRUE)` prints more information about each DVH.

Value

Prints summary information about the DVHs.
readConstraint

See Also

readDvh

Examples

print(dataMZ)
predict(dataMZ, verbose=TRUE)

Description

Reads the definition of quality assurance constraints from a text file.

Usage

readConstraint(x, ...)

Arguments

x
character string giving the path to a single text file with the constraint definition. May contain globbing symbols understood by Sys.glob. If missing and in interactive mode, readDvh opens a file selector widget. See Details.

...
Further arguments passed to read.table, e.g., sep="\t" to define the column separator as tab.

Details

This is a wrapper for read.table.

The text file should contain three columns with the column names patID, structure, constraint in the first line. Each further line then defines one constraint and the scope it applies to in terms of patients and structures. See checkConstraint for the definition of a constraint and for the definition of a scope. Example content:

"patID" "structure" "constraint"
"*" "HEART" "D1CC < 20Gy"
"234" "*" "V10% > 8CC"

Value

A data.frame with columns patID, structure, constraint that can be used in functions checkConstraint and showConstraint.

See Also

read.table, checkConstraint, saveConstraint, showConstraint
Examples

## Not run:
```r
readConstraint("constraint.txt")
readConstraint()
```
## End(Not run)

---

**readDVH**

*Read DVH text files*

**Description**

Reads single or multiple DVH text files as exported from Varian Eclipse(TM), CadPlan(TM), On-Centra MasterPlan(TM), Philips Pinnacle3 (TM), Elekta Monaco (TM), Tomo HiArt (TM), Ray-Search Labs RayStation (TM), or Medcom ProSoma (TM). Supports cumulative and differential DVHs.

**Usage**

```r
readDVH(x, 
  type=c("Eclipse", "Cadplan", "Masterplan", 
         "Pinnacle", "Monaco", "HiArt", 
         "RayStation", "ProSoma", "PRIMO"), 
  planInfo=FALSE, courseAsID=FALSE, add, ...)
```

**Arguments**

- **x** character vector giving paths to DVH text files. May contain globbing symbols understood by `Sys.glob`. If missing and in interactive mode, `readDVH` opens a file selector widget. Under Windows, this widget allows selecting multiple files simultaneously. For `type="Pinnacle"`, `x` should be one of the following: A directory with information for one patient, a directory with several sub-directories (one for each patient), or a zip file of such directories. Under Windows, if `x` is missing and `type="Pinnacle"`, `readDVH` opens a folder selector widget.
- **type** character. Indicates which program the DVH text files were exported from. Supported: "Cadplan" (tested with version 6.4.7), "Eclipse" (tested with Varian Eclipse version 10-15), "Masterplan" (tested with OnCentra MasterPlan version 4.3), "Pinnacle" (tested with Pinnacle3 version 9, see Details), "Monaco" (tested with Elekta Monaco version 5), "HiArt" (TomoTherapy HiArt), "RayStation" (RaySearch Labs RayStation), "ProSoma" (Medcom ProSoma), "PRIMO" (tested with version 0.3.1.1558).
- **planInfo** Experimental: Either FALSE or character string. In the latter case, `readDVH` tries to extract additional information from the Plan field in the DVH file, e.g., the prescription dose for a sum plan or the boost quadrant. Undocumented, see source.
readDVH

```r
courseAsID logical. If TRUE, the Course entry in the header section of a DVH file is appended to the regular patient ID. Currently supported only for type="Eclipse".
add DVHLstLst object. Existing object that should be merged with the new data from the files.
... Additional arguments passed on to file. Specify UTF-8 file encoding with encoding="UTF-8" or encoding="UTF-8-BOM" (when a byte-order-mark is used). Passing additional arguments is currently not supported when reading Pinnacle files. Additional arguments are also used for type="HiArt" where a list hiart may be supplied that specifies patient IDs, absolute structure volumes, and prescription dose (see Details).
```

**Details**

Absolute dose values need to be given in Gy, cGy, or eV/g for uncalibrated dose in DVHs exported by PRIMO. Absolute volume values need to be given in in cm^3.

Differential DVHs are automatically converted to cumulative DVHs, but the differential DVH information is kept.

Sum plans are supported.

For Eclipse starting with version 13, the date format is locale dependent as it uses words for day and month. Importing those dates as class Date requires that the correct locale is set (see Sys.setlocale), and that files containing accents are read using the correct encoding (see above). Otherwise, date is stored as a character string.

For RayStation, only cumulative DVHs with absolute volume are currently supported. Volume is assumed to be measured in cm^3.

Since files from HiArt, ProSoma and PRIMO do not contain info on patient ID, the current workaround is to generate a random ID.

To export data from Tomo HiArt, copy to clipboard and then save to file from a text editor. Support for Tomo HiArt files is currently limited to those with absolute dose and relative volume. Please send an anonymized sample file if you need this functionality. Alternatively, provide a list hiart with one component for each file you import. Each component itself has to be a named list with optional components

- **patName** - a character string for patient name
- **patID** - a character string for patient ID
- **doseRx** - a numeric value like 50.4 for prescription dose in the same dose unit as used in the DVHs
- **structVol** - a named list like list("PTV"=52Gy"=750, "LUNG"=1250) giving the absolute structure volumes with names equal to structure names and numeric components of length 1
- **volumeUnit** - a character string like "CC" for the structure volume unit)

Pinnacle3 files have to be exported using its own scripting facility such that information from one patient is contained in one directory. A suitable export script is available on request from the package authors. The directory layout for one patient has to be as follows (experimental, likely to change in future versions):

- **Files** (CSV format with column headers):

```
- DoseInfo.csv (variables "PrescriptionDose cGy", "NumberOfFractions", "Dosis cGy")
- PatInfo.csv (variables "LastName", "FirstName", "MedicalRecordNumber")
- PlanInfo.csv (variable "PlanName")

- Directory: Data:
  - Info.csv (variables "Filename", "RegionOfInterestName", "DoseMin cGy", "DoseMax cGy", "DoseMean cGy", "Volume ccm")
  - DVH1.csv, DVH2.csv, ... - the actual DVH data files with names defined in Info.csv variable "Filename". They should look like NumberOfDimensions = 2; NumberOfPoints = 431;
    Points[] ={
      0,0
      10,0
      ...
      4000,100
    };

Value

Returns an object of class DVHLstLst. This is a list (one component with class DVHLst for each original file from one patient) of lists (each component is an object of class DVHs). A DVHs object is a list with the following components:

dvh matrix - cumulative DVH values
dvhDiff matrix - differential DVH values, only created a) if original file contained a differential DVH or b) by convertDVH
patID character string - patient ID
date character string - date of DVH export
type character string - cumulative or differential DVH
plan character string - plan name
course character string - course - currently Eclipse only
structure character string - structure name
structVol numeric - structure volume
doseUnit character string - measurement unit dose
volumeUnit character string - measurement unit volume
doseRx numeric - prescription dose
isoDoseRx numeric - iso-dose percentage
doseMin numeric - minimum dose from DVH file
doseMax numeric - maximum dose from DVH file
doseAvg numeric - average dose from DVH file
doseMed numeric - median dose from DVH file
doseSD numeric - dose standard deviation from DVH file
runGUI

See Also

Sys.glob, readLines, print.DVHs, showDVH, getMetric, checkConstraint, convertDVH

Examples

```r
## Not run:
# pick DVH files interactively
res <- readDVH()
res

# read all txt files in subdirectory DVH
res <- readDVH("DVH/*.txt", type="Eclipse")
res

## End(Not run)
```

---

runGUI  Open web-based GUI in browser

Description

Opens the web-based GUI in an external browser.

Usage

```
runGUI(...)  
```

Arguments

```
...  Arguments passed to runApp. Supply port=80 if a web browser refuses to connect to the randomly chosen port for security reasons.
```

Details

This function calls runApp to run the included DVHshiny application. See vignette("DVHshiny") for documentation.

See Also

runApp

Examples

```
## Not run:
runGUI()

## End(Not run)
```
**saveConstraint**  
*Save constraint result to file*

---

**Description**

Saves results from `checkConstraint` to a text file.

**Usage**

```
saveConstraint(x, ...)  
```

**Arguments**

- `x`  
  data.frame - the result from `checkConstraint`.  
- `...`  
  Further arguments passed to `write.table` - e.g., `file="<filename>"` for the output filename, `dec="."` to define the decimal separator as point or `sep="\t"` to define the column separator as tab.

**Details**

This is a wrapper for `write.table`.

**See Also**

`write.table`, `checkConstraint`

**Examples**

```
res <- checkConstraint(dataMZ, c("D10CC < 10Gy", "V20Gy < 20%"))
## Not run:
saveConstraint(res, file="constrResults.txt", sep="\t")

## End(Not run)
```

---

**saveDVH**  
*Save DVH diagram to file*

---

**Description**

Saves one or multiple DVH diagrams to file.

**Usage**

```
saveDVH(x, file="", ...)  
```
Arguments

x  A single ggplot object or a list of multiple ggplot objects as returned by showDVH or showConstraint.

file  character. Path to file. The file-ending determines what kind of file is written, e.g., "filename.pdf" will write a pdf document, "filename.jpg" a JPEG image.

...  Further arguments passed to ggsave, e.g., width and height to determine the figure size.

Details

This is a wrapper for ggsave.

Value

If x is a list of ggplot objects, one file is written for each list component. If x is a single ggplot object, one file is written.

See Also

ggsave, showDVH, showConstraint

Examples

res <- showDVH(dataMZ, byPat=TRUE, structure=c("HEART", "AMYOCL"))
## Not run:
saveDVH(res, "out.pdf")
## End(Not run)

describeMetric  

Description

Saves results from getMetric to a text file.

Usage

saveMetric(x, file = "", ...)

## S3 method for class 'data.frame'
saveMetric(x, file = "", ...)  

## S3 method for class 'list'
saveMetric(x, file = "", ...)
showConstraint

Arguments

- **x**: data.frame or list - the result from `getMetric`.
- **file**: character. Path to file.
- **...**: Further arguments passed to `write.table` - e.g., `dec="."` to define the decimal separator as point or `sep="\t"` to define the column separator as tab.

Details

This is a wrapper for `write.table`.

Value

If x is a list, one text file is written for each list component. If x is a data.frame, one file is written.

See Also

`write.table`, `getMetric`

Examples

```r
res <- getMetric(dataMZ, c("D1CC", "V10% CC"),
         sortBy=c("metric", "structure"),
         splitBy="patID")

## Not run:
# not run
saveMetric(res, file="metricsResults.txt", sep="\t")

## End(Not run)
```

showConstraint  Display constraints for cumulative dose-volume histograms

Description

Displays quality assurance constraints for cumulative dose-volume histograms: Either one diagram per patient - including multiple structures. Or one diagram per structure - including multiple patients.

Usage

```r
showConstraint(x, constr, byPat=TRUE, rel=TRUE, guessX=TRUE, guessY=TRUE, thresh=1, show=TRUE, visible=FALSE)

## S3 method for class 'DVHs'
showConstraint(x, constr, byPat=TRUE, rel=TRUE, guessX=TRUE, guessY=TRUE, thresh=1, show=TRUE, visible=FALSE)
```
## S3 method for class 'DVHLst'

The `showConstraint` function can be used with the following arguments:

- `x`: A single DVH (object of class DVH), multiple DVHs from one patient/structure (object of class DVHLst), or multiple DVHs from many patients/structures (object of class DVHLstLst). See `readDVH`. See Details.
- `constr`: One or more constraints - given as a character vector or as a data.frame. See `checkConstraint` for their definition.
- `byPat`: logical. Relevant if multiple DVHs are given. If `x` has class DVHLstLst: `byPat=TRUE` means that one diagram shows DVHs from one patient with multiple structures. `byPat=FALSE` means that one diagram shows DVHs for one structure from multiple patients.
- `rel`: logical. Show relative volume?
- `guessX`: logical. Try to guess the best x-axis limits for better visibility of main DVH range? If FALSE, x-axis runs from 0 to maximum dose. If TRUE, x-axis runs from 0 to dose value where volume approaches 0. If a single number is given, it is interpreted as the maximum value. If a vector of two numbers is given, it is interpreted as the range of the axis.
- `guessY`: logical. Try to guess the best y-axis limits? If a single number is given, it is interpreted as the maximum value. If a vector of two numbers is given, it is interpreted as the range of the axis.
- `thresh`: numeric value. Relative volume threshold used with `guessX=TRUE`. Clip x-axis (+10%) such that the "highest" DVH is cut off at this relative volume.
- `show`: logical. If TRUE, diagrams are shown, if FALSE diagrams are not shown - only `ggplot` diagram objects are silently returned.
- `visible`: logical. Return `ggplot` diagram object visibly or invisibly. `show=FALSE` with `visible=TRUE` is useful for zooming in shiny apps.

### Details

Constraints are shown as points in the cumulative DVH with an additional arrow indicating where the cumulative DVH curve should lie relative to the constraint. On each DVH curve, the point with the minimal Euclidean distance to the constraint is indicated. Note that, visually, this point only has the minimal apparent distance if the aspect ratio of the diagram is 1.

If multiple diagrams are produced, they are shown in the same graphics device. If interactive inspection is required, make sure you use an R development environment that saves previous diagrams and allows navigating between them - e.g., RStudio or OpenAnalytics Architect.
Value

Silently returns a ggplot diagram object, or - when multiple diagrams are constructed - a list of ggplot diagram objects.

See Also

checkConstraint, saveDVH

Examples

data(dataMZ)

# define constraints
constr <- data.frame(
  patID=c("P123", "P234"),
  structure=c("HEART", "x"),
  constraint=c("D1CC < 20Gy", "V10% > 8CC"),
  stringsAsFactors=FALSE) # this is important
showConstraint(dataMZ, constr=constr, byPat=FALSE)

showDVH

Display dose volume histograms

Description

Displays dose volume histograms: Either one diagram per patient - including multiple structures. Or one diagram per structure - including multiple patients.

Usage

showDVH(x, cumul=TRUE, byPat=TRUE, patID=NULL, structure=NULL,
  rel=TRUE, guessX=TRUE, guessY=TRUE, thresh=1, addMSD=FALSE,
  show=TRUE, visible=FALSE, fixed=TRUE)

## S3 method for class 'DVHs'
showDVH(x, cumul=TRUE, byPat=TRUE, patID=NULL, structure=NULL,
  rel=TRUE, guessX=TRUE, guessY=TRUE, thresh=1, addMSD=FALSE,
  show=TRUE, visible=FALSE, fixed=TRUE)

## S3 method for class 'DVHLst'
showDVH(x, cumul=TRUE, byPat=TRUE, patID=NULL, structure=NULL,
  rel=TRUE, guessX=TRUE, guessY=TRUE, thresh=1, addMSD=FALSE,
  show=TRUE, visible=FALSE, fixed=TRUE)

## S3 method for class 'DVHLstLst'
showDVH(x, cumul=TRUE, byPat=TRUE, patID=NULL, structure=NULL,
  rel=TRUE, guessX=TRUE, guessY=TRUE, thresh=1, addMSD=FALSE,
  show=TRUE, visible=FALSE, fixed=TRUE)
showDVH

Arguments

- **x**: A single DVH (object of class dvhs), multiple DVHs from one patient/structure (object of class dvhlst), or multiple DVHs from many patients/structures (object of class dvhlstlst). See readDVH. See Details.
- **cumul**: logical. Show cumulative or differential (per unit dose) DVH?
- **byPat**: logical. Relevant if multiple DVHs are given. If x has class dvhlstlst: byPat=TRUE means that one diagram shows DVHs from one patient with multiple structures. byPat=FALSE means that one diagram shows DVHs for one structure from multiple patients.
- **patID**: character vector. Show diagram for these patients only. If missing, all patients are shown. Can be a regular expression with fixed=FALSE, see regex.
- **structure**: character vector. Show diagram for these structures only. If missing, all structures are shown. Can be a regular expression with fixed=FALSE, see regex.
- **rel**: logical. Show relative volume?
- **guessX**: logical. Try to guess the best x-axis limits for better visibility of main DVH range? If FALSE, x-axis runs from 0 to maximum dose. If TRUE, x-axis runs from 0 to dose value where volume approaches 0. If a single number is given, it is interpreted as the range of the axis.
- **guessY**: logical. Try to guess the best y-axis limits? If a single number is given, it is interpreted as the maximum value. If a vector of two numbers is given, it is interpreted as the range of the axis.
- **thresh**: numeric value. Relative volume threshold used with guessX=TRUE. Clip x-axis (+5%) such that the "highest" DVH is cut off at this relative volume.
- **addMSD**: logical. If TRUE, diagram shows the point-wise mean DVH as well as shaded areas for point-wise 1-standard deviation and 2-standard deviations around this mean. See details.
- **show**: logical. If TRUE, diagrams are shown. If FALSE diagrams are not shown - only ggplot diagram objects are silently returned.
- **visible**: logical. Return ggplot diagram object visibly or invisibly. show=FALSE with visible=TRUE is useful for zooming in shiny apps.
- **fixed**: logical. Use fixed=FALSE for regular expression matching of patID and structure.

Details

If multiple diagrams are produced, they are shown in the same graphics device. If interactive inspection is required, make sure you use an R development environment that saves previous diagrams and allows navigating between them - e.g., RStudio or OpenAnalytics Architect.

For addMSD=TRUE, the number of DVH nodes (dose values) is reduced by 1/3 of the maximum number of nodes in x. Before calculating the point-wise mean and SD, DVHs in x are first linearly interpolated using the same set of nodes.
showMeanDVH

Value

Silently returns a ggplot diagram object, or - when multiple diagrams are constructed - a list of ggplot diagram objects.

See Also

ggplot, readDVH, saveDVH, getMeanDVH

Examples

showDVH(dataMZ, byPat=TRUE, structure=c("HEART", "AMYOCL"))

# matches patients P123 and P234
showDVH(dataMZ, byPat=FALSE, patID="23", fixed=FALSE)

showMeanDVH (Show average dose volume histograms)

Description

Displays average dose volume histograms grouped by patients or structures.

Usage

showMeanDVH(x, byPat=TRUE, patID=NULL, structure=NULL, rel=TRUE, guessX=TRUE, thresh=1, show=TRUE, fixed=TRUE, showSD=TRUE, color=TRUE, facet=TRUE)

Arguments

x A data frame as returned by getMeanDVH or a list of such data frames.
byPat logical. Relevant if multiple DVHs are given. If x has class DVHList: byPat=TRUE means that one diagram shows DVHs from one patient with multiple structures. byPat=FALSE means that one diagram shows DVHs for one structure from multiple patients.
patID character vector. Show diagram for these patients only. If missing, all patients are shown. Can be a regular expression with fixed=FALSE, see regex.
structure character vector. Show diagram for these structures only. If missing, all structures are shown. Can be a regular expression with fixed=FALSE, see regex.
rel logical. Show relative volume?
guessX logical. Try to clip the x-axis for better visibility of main DVH range?
thresh numeric value. Relative volume threshold used with guessX=TRUE. Clip x-axis (+10%) such that the "highest" DVH is cut off at this relative volume.
show logical. If TRUE, diagrams are shown, if FALSE diagrams are not shown - only ggplot diagram objects are silently returned.
showMeanDVH

fixed  logical. Use fixed=FALSE for regular expression matching of patID and structure.
showSD logical. If TRUE, diagram shows shaded areas for point-wise 1-standard deviation and 2-standard deviations around this mean. See details.
color  logical. If TRUE, diagram uses color to distinguish groups. If FALSE, colors are greyscale, and line types are used to distinguish groups.
facet  logical. If TRUE, different structures (for byPat=FALSE or different patients (for byPat=TRUE go into separate panels using facet_grid. If FALSE, everything is shown in the same panel.

Details

TODO

Value

Silently returns a ggplot diagram object, or - when multiple diagrams are constructed - a list of ggplot diagram objects.

See Also

ggplot, showDVH, getMeanDVH

Examples

# mean DVH for HEART and AMYOCL averaged over patients
res <- getMeanDVH(dataMZ, byPat=FALSE, structure=c("HEART", "AMYOCL"))
showMeanDVH(res)