Package 'munsellinterpol'

September 20, 2026

Type Package

Title Interpolate Munsell Renotation Data from Hue Value/Chroma to CIE/RGB

Version 3.6-0

Encoding UTF-8

Date 2026-09-19

Description
Methods for interpolating data in the Munsell color system following the ASTM D-1535 standard. Hues and chromas with decimal values can be interpolated and converted to/from the Munsell color system and CIE xyY, CIE XYZ, CIE Lab, CIE Luv, or RGB. Includes ISCC-NBS color block lookup. Based on the work by Paul Centore, ``The Munsell and Kubelka-Munk Toolbox".

License GPL (>= 3)

LazyLoad yes

LazyData yes

Depends R (>= 4.0.0)

Imports rootSolve, spacesRGB, spacesXYZ (>= 1.6-0), logger (>= 0.4.1)

Suggests microbenchmark, mgcv, knitr, rmarkdown, colorSpec (>= 1.8-0), flextable, OxSR

Maintainer Glenn Davis <gdavis@gluonics.com>

Repository CRAN

NeedsCompilation no

VignetteBuilder knitr

BuildVignettes yes

Author Jose Gama [aut, trl],
Paul Centore [aut, cph],
Glenn Davis [aut, cre],
Hugo Rodrigues [ctb] (ORCID: <https://orcid.org/0000-0002-8070-8126>)

Date/Publication 2026-09-20 00:10:02 UTC

Contents

CentroidsISCCNBS . . . 2
ColorBlockFromMunsell . . . 3
ColorlabFormatToMunsellSpec . . . 5
HVCfromMunsellName . . . 7
IsWithinMacAdamLimits . . . 10
LabToMunsell . . . 12
LuvToMunsell . . . 14
Munsell2xy . . . 15
MunsellBooks . . . 17
MunsellNameFromHVC . . . 19
MunsellSpecToColorlabFormat . . . 21
MunsellToLab . . . 23
MunsellToLuv . . . 24
MunsellToRGB . . . 26
MunsellTosRGB . . . 28
MunsellToxyY . . . 29
MunsellToXYZ . . . 32
NickersonColorDifference . . . 35
plotLociHC . . . 36
plotPatchesH . . . 38
RGBtoMunsell . . . 40
roundHVC . . . 41
sRGBtoMunsell . . . 43
VandY . . . 45
xyYtoMunsell . . . 47
XYZtoMunsell . . . 50

Index **53**

`CentroidsISCCNBS` *Centroid Notations for the Revised ISCC-NBS Color-Name Blocks*

Description

`CentroidsISCCNBS` is a table with the centroids of the revised ISCC-NBS Color-Name Blocks, taken from *Kelly (1958)*

Format

This `data.frame` has 267 rows and these columns:

Number ISCC-NBS number (an integer from 1 to 267)

Name ISCC-NBS name

MunsellSpec Munsell specification of the centroid of the block a (character string)

Details

The earliest paper I am aware of is by *Nickerson, et. al.* in 1941. After the big Munsell renotation in 1943, the name blocks were revised in 1955.
When the central colors were recomputed in *Kelly (1958)*, they were called the "Central Colors", though the text makes it clear that most are truly centroids, which were computed from the centroid of an "elementary shape", which is a "sector of a right cylindrical annulus". For the "peripheral blocks" of high chroma, the centroids were "estimated graphically by plotting the MacAdam limits". In *Kelly (1965)* these were called "centroid colors", and that is the name we will use here.

Contributor

Glenn Davis

References

Nickerson, Dorothy and Sidney M. Newhall. **Central Notations for ISCC-NBS Color names**. J Opt. Soc. Am. Vol 31 Iss. 9. pp. 597-591. 1941.

Newhall, Sidney M., Dorothy Nickerson, Deane B. Judd. **Final Report of the O.S.A. Subcommitte on the Spacing of the Munsell Colors**. Journal of the Optical Society of America. Vol. 33. No. 7. pp. 385-418. July 1943.

Kelly, Kenneth L. and Deane B. Judd **The ISCC-NBS Method of Designating Colors and a Dictionary of Color Names**. National Bureau of Standards Circular 553. Washington DC: US Government Printing Office. November 1, 1955.

Kelly, Kenneth Low. **Central Notations for the Revised ISCC-NBS Color-Name Blocks**. Journal of Research of the National Bureau of Standards. Research Paper 2911. Vol. 61 No. 5. pp. 427-431. November 1958.

Kelly, Kenneth Low. **A Universal Color Language**. Color Engineering. Vol. 3 No. 2. pp. 2-7. March-April, 1965.

Examples

```
print( CentroidsISCCNBS[ 1:5, ] )

##    Number          Name   MunsellSpec
##  1      1    vivid pink     1.5R 7/13
##  2      2   strong pink  1.5R 7.5/9.1
##  3      3     deep pink 1.9R 6.0/11.1
##  4      4    light pink  2.5R 8.6/5.2
##  5      5 moderate pink  2.5R 7.2/5.2
```

`ColorBlockFromMunsell` *Get ISCC-NBS Number and ISCC-NBS Name from Munsell HVC or Munsell Notation*

Description

Get ISCC-NBS Number and ISCC-NBS Name from Munsell HVC or Munsell Notation.

Usage

```
ColorBlockFromMunsell( MunsellSpec )
```

Arguments

`MunsellSpec`	a numeric Nx3 matrix or a vector that can be converted to such a matrix. Each row has Munsell HVC, where H is Hue Number, and V and C are the standard Munsell Value and Chroma. The Hue is automatically wrapped to the interval (0,100]. `MunsellSpec` can also be a character N-vector with standard Munsell notation; it is converted to an Nx3 matrix.

Details

The ISCC-NBS System is a partition of Munsell Color Solid into 267 color blocks. Each block is a disjoint union of *elementary blocks*, where an elementary block is defined by its minimum and maximum limits in Hue, Value, and Chroma. Some blocks are non-convex. The *peripheral blocks*, of which there are 120, have arbitrary large chroma and are considered semi-infinite for this function; there is no consideration of the MacAdam limits. For each query vector HVC, the function searches a private `data.frame` with 932 elementary blocks, for the one elementary block that contains it.

Value

a `data.frame` with N rows and these columns:

`HVC`	the input Nx3 matrix, or such a matrix converted from Munsell notation
`Number`	the corresponding ISCC-NBS color number - an integer from 1 to 267
`Name`	the corresponding ISCC-NBS color name - a character string
`Centroid`	the centroid of the block in Munsell Notation - a character string; see `CentroidsISCCNBS`

The `rownames` are set to the input `MunsellSpec`.

History

The **Munsell Book of Color** was published in 1929. The first ISCC-NBS partition, in 1939, had 319 blocks and names (including 5 neutrals). There were no block numbers. The aimpoints of the Munsell samples were thoroughly revised in 1943. The ISCC-NBS partition was revised in 1955, and this is the version used here.

Future Work

It might be useful to compute the distance from the query point to the boundary of the containing color block.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, an error condition is signaled with the additional S3 class `"invalid_argument"`. The invalid argument is added to the condition object.

In the case that not all color blocks could be computed, a warning condition is signaled with the additional S3 class `"incomplete"` or `"table_error"` or `"munsell_clamp"`, as appropriate. The condition object also has the indexes of the invalid blocks as item `indexes`.

Author(s)

Glenn Davis

References

Munsell Color Company, A.H. Munsell, and A.E.O. Munsell. **Munsell book of color: defining, explaining, and illustrating the fundamental characteristics of color**. 1929.

Judd, Deane B. and Kenneth L. Kelly. **Method of Designating Colors**. Journal of Research of the National Bureau of Standards. Research Paper 1239. Volume 23 Issue 3. pp. 355-385. September 1939.

Newhall, Sidney M., Dorothy Nickerson, Deane B. Judd. **Final Report of the O.S.A. Subcommitte on the Spacing of the Munsell Colors**. Journal of the Optical Society of America. Vol. 33. No. 7. pp. 385-418. July 1943.

Kelly, Kenneth L. and Deane B. Judd **The ISCC-NBS Method of Designating Colors and a Dictionary of Color Names**. National Bureau of Standards Circular 553. Washington DC: US Government Printing Office. November 1, 1955.

See Also

`CentroidsISCCNBS`

Examples

```
ColorBlockFromMunsell( c( "3R 8/3", "7.4YR 3/4" ) )
```

```
##            HVC.H HVC.V HVC.C Number           Name      Centroid
##  3R 8/3      3.0   8.0   3.0      4     light pink  2.5R 8.6/5.2
##  7.4YR 3/4  17.4   3.0   4.0     58 moderate brown 5.5YR 3.5/3.9
```

ColorlabFormatToMunsellSpec

Convert Colorlab Munsell Format to Munsell HVC

Description

Convert Colorlab Munsell Format to Munsell HVC

Usage

```
ColorlabFormatToMunsellSpec( HVCH )
```

Arguments

HVCH	a numeric Nx4 matrix, or a vector that can be converted to such a matrix, by row. Each row of the matrix contains an HVCH vector.

Details

Colorlab Munsell format uses 4 numbers.

1. Hue Step, in the interval (0,10], or 0 for neutrals
2. Munsell Value, in the interval (0,10]
3. Munsell Chroma, non-negative
4. Hue Index, an integer from 1 to 10, or 0 for neutrals

Value

an Nx3 matrix, with each row an HVC vector. Value and Chroma are simply copied unchanged. The complex part is conversion of Colorlab Hue Step and Hue Index to Hue Number. For neutrals, both Hue Step and Hue Index are ignored. Invalid input values, such as a Hue Index that is not an integer from 0 to 10 (except for neutrals), are converted to NAs. The rownames of the input are copied to the output, but if these are NULL, the rownames are set to the Munsell notations.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR event, an error condition is signaled with the additional S3 class "invalid_argument". The invalid argument is added to the condition object.

Author(s)

Jose Gama and Glenn Davis

References

Color Processing Toolbox. Colorlab 1.0. https://www.uv.es/vista/vistavalencia/software/colorlab.html

See Also

MunsellSpecToColorlabFormat()

Examples

```
ColorlabFormatToMunsellSpec( c( 3.2,3,2,1,  2,5.1,0,0, 2,5.1,0.1,0 ) )
##                      H   V  C
##  3.20B 3.00/2.00 63.2 3.0  2
##  N 5.10/          0.0 5.1  0
##  <NA>              NA  NA NA
```

`HVCfromMunsellName` *Convert Munsell Notation to numerical Munsell HVC*

Description

Convert Munsell Notation to numerical Munsell HVC

Usage

```
HVCfromMunsellName( MunsellName )
HueNumberFromString( HueString )
MunsellHVC( MunsellName )
```

Arguments

`MunsellName`	a character vector of length N > 0, where each string should be a valid Munsell notation, e.g. `'2.3P 5/2.3'`, `'9.2YR 3/6'`, `'N 2.3/'`, and `'N 4/0'`. Whitespace is optional and ignored. It is OK for a neutral to end in either `'/'` or `'/0'`.
`HueString`	a character vector of length N > 0, where each string should be the initial hue part of a Munsell notation, e.g. '4.5GY', '2.5R', '10.3B', etc. Whitespace is optional and ignored. Neutrals, denoted by `'N'`, are invalid because the hue is undefined.

Value

`HVCfromMunsellName()` returns a numeric Nx3 matrix with HVC in the rows. For neutral colors, both H and C are set to 0. If a string cannot be parsed, the entire row is set to `NA`s. The rownames are set to `MunsellName`.

`HueNumberFromString()` returns the hue number H. If the string cannot be parsed, the output is set to `NA`.

For the 2 previous functions the Hue Number is wrapped to (0,100], except when Chroma==0 and then Hue=0.

`MunsellHVC()` returns a *character* Nx3 matrix with HVC in the rows. It is deprecated and and is here for backward compatibility with older versions of the package. For neutral colors, H is set to `'N'` and C is set to `'0'`.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, an error condition is signaled with the additional S3 class `"invalid_argument"`. The invalid argument is added to the condition object.

In the case that not all HVC rows, or Hue numbers, could be computed, because some of the strings are invalid, a warning condition is signaled with the additional S3 class `"incomplete"`. The condition object also has the subvector of invalid strings as item `invalid`, and their indexes as item `indexes`.

Note

Ever since the *Munsell Book of Color (1929)*, the Munsell hue circle has been divided into 5 *principal hues* and 5 *intermediate hues*, to define a total of 10 arcs. Each of these hues has been assigned a 10-point scale, with 5 at the midpoint of the arc. Moreover, the hue "origin" has been at '10RP'. So a 100-point scale (with no letters) for the entire hue circle is obvious and trivial to construct, but I have been unable to determine the first explicit mention of such a scale. The earliest I have have found is from *Nimeroff (1968)*, Figures 20 and 21 on page 27.

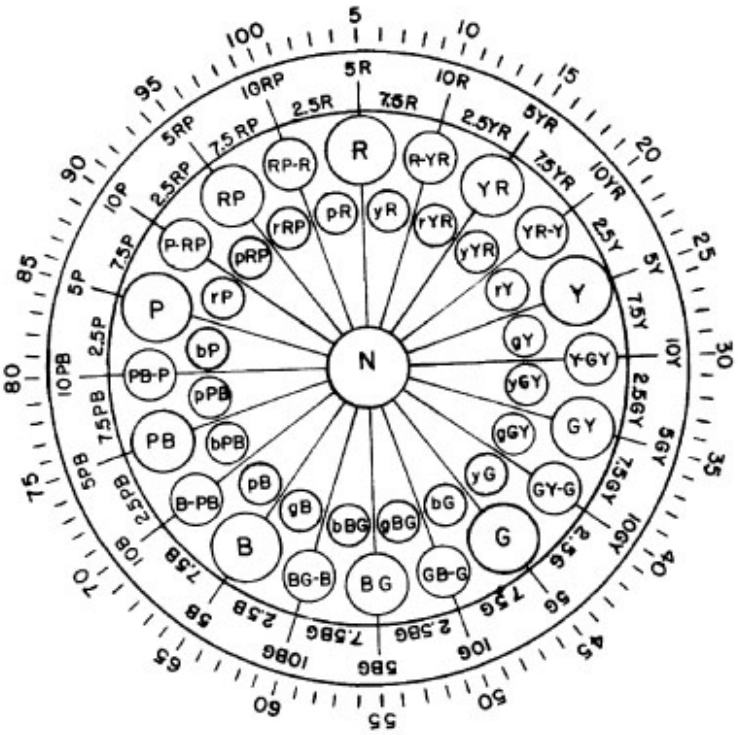

FIGURE 20. *Diagram of hue circle with Munsell hue notation.*

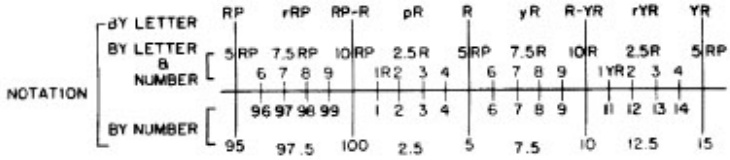

FIGURE 21. *Alternate ways of expressing Munsell hue notation.*

There is a reference to *ASTM D 1535* in the **References** of *Nimeroff*, but it is not dated, and the 2 figures are not attributed to it. There was an *ASTM D 1535* in 1968 but I have not been able to locate it; it is possible that the 100-point scale first appeared in *ASTM D-1535 (1968)*, or even earlier in *ASTM D 1525-58T (1958)*.

Interestingly, in the *Atlas of the Munsell Color System (1915)* there were only 5 principal hues (with no intermediate hues) and each of the 5 arcs was assigned a 10-point scale. If the entire hue circle of 1915 were assigned a scale, it would have been a 50-point scale.

Author(s)

Glenn Davis

References

Nimeroff, I. **Colorimetry**. National Bureau of Standards Monograph 104. January 1968. 35 cents.

ASTM D 1535-80. Standard Practice for Specifying Color by the Munsell System. 1980.

Munsell Book of Color: defining, explaining, and illustrating the fundamental characteristics of color. Munsell Color Co. 1929.

Atlas of the Munsell Color System. Malden, Mass., Wadsworth, Howland & Co., inc., Printers. 1915.

See Also

`MunsellNameFromHVC()`, `HueStringFromNumber()`

Examples

```
HVCfromMunsellName( c( "4.2P 2.9/3.8", "N 2.3/", "N 8.9/0" ) )
##                   H   V   C
##  4.2P 2.9/3.8 84.2 2.9 3.8
##  N 2.3/        0.0 2.3 0.0
##  N 8.9/0       0.0 8.9 0.0

HueNumberFromString( c('4B','4.6GY','10RP','N') )
##  [1]  64.0  34.6 100.0    NA
```

IsWithinMacAdamLimits *Test xyY Coordinates for being Inside the MacAdam Limits*

Description

Test xyY Coordinates for being Inside the MacAdam Limits for Illuminants C and D65

Usage

```
IsWithinMacAdamLimits( xyY, Illuminant='C' )
```

Arguments

`xyY`	a numeric Nx3 matrix with CIE xyY coordinates in the rows, or a vector that can be converted to such a matrix, by row. The reference white is asssumed to have Y=100.
`Illuminant`	either `'C'` or `'D65'`. Partial matching is enabled, and is case-insensitive.

Details

The *MacAdam Limit* is the boundary of the *optimal color solid* (also called the *RC6sch FarbkC6rper*), in XYZ coordinates. The optimal color solid is convex and depends on the illuminant. Points on the boundary of the solid are called *optimal colors*. It is symmetric about the midpoint of the segment joining black and white (the 50% gray point). It can be expressed as a *zonohedron* - a convex polyhedron with a special form; for details on zonohedra, see *Centore*.

For each of the 2 illuminants, a zonohedron Z is pre-computed (and stored in `sysdata.rda`). The wavelengths used are 380 to 780 nm with 5nm step (81 wavelengths). Each zonohedron has 81*80=6480 parallelogram faces, though some of them are coplanar. Z is expressed as the intersection of 6480 halfspaces. The plane equation of each parallelogram is pre-computed, but redundant ones are not removed (in this version).

For testing a query point xyY, a pseudo-distance metric δ is used. Let the zonohedron Z be the intersection of the halfspaces $\langle\, h_i, x \,\rangle \leq b_i \;\; i = 1, ..., n$, where each h_i is a unit vector. The point xyY is converted to XYZ, and δ(XYZ) is computed as: δ(XYZ) := max($\langle\, h_i$,XYZ $\rangle$ - b_i) where the maximum is taken over all $i = 1, ..., n$. This calculation can be optimized; because the zonohedron is centrally symmetric, only half of the planes actually have to be stored, and this cuts the memory and processing time in half. It is clear that XYZ is within the zonohedron iff δ(XYZ) ≤ 0, and that XYZ is on the boundary iff δ(XYZ)=0. This pseudo-distance is part of the returned `data.frame`.

An interesting fact is that if δ(XYZ)>0, then δ(XYZ) $\leq$ dist(XYZ,Z), with equality iff the segment from XYZ to the point z on the boundary of Z closest to XYZ is normal to one of the faces of Z that contains z. This is why we call δ a *pseudo-distance*. Another interesting fact is that if δ(XYZ) ≤ 0, then δ(XYZ) = -min($\Psi_Z(u)$ - $\langle\, u$,XYZ $\rangle$), where the minimum is taken over all unit vectors u and where Ψ_Z is the *support function* of Z.

Value

A `data.frame` with N rows and these columns:

`within`	a logical which is `TRUE` iff the corresponding row in `xyY` is inside the optimal color solid for the illuminant. If a point is exactly on the boundary (unlikely), `within=TRUE`. Explicitly, `within = (delta<=0)`.
`delta`	the pseudo-distance δ discussed in **Details**

The row names of the output value are set equal to the row names of `xyY`.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, a condition is signaled with one of these additional S3 classes: `"invalid_argument"` or `"package_unavailable"`, whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

Author(s)

Glenn Davis and Jose Gama

References

Centore, Paul. *A zonohedral approach to optimal colours*. **Color Research & Application**. Vol. 38. No. 2. pp. 110-119. April 2013.

RC6sch, S. **Darstellung der Farbenlehre fC<r die Zwecke des Mineralogen**. Fortschr. Mineral. Krist. Petrogr. Vol. 13 No. 143. 1929.

MacAdam, David L. **Maximum Visual Efficiency of Colored Materials**. Journal of the Optical Society of America. Vol 25, No. 11. pp. 361-367. November 1935.

Wikipedia. **Support Function**. https://en.wikipedia.org/wiki/Support_function

Examples

```
IsWithinMacAdamLimits( c(0.6,0.3,10, 0.6,0.3,20, 0.6,0.3,30, 0.6,0.3,40  ), 'C' )

##    within  delta
##  1   TRUE -1.941841
##  2   TRUE -1.332442
##  3  FALSE  3.513491
##  4  FALSE 12.826172
```

`LabToMunsell` *Convert CIE Lab coordinates to Munsell HVC*

Description

`LabToMunsell` Converts CIE Lab coordinates to Munsell HVC, by interpolating over the extrapolated Munsell renotation data

Usage

```
LabToMunsell( Lab, white='D65', adapt='Bradford', ... )
```

Arguments

`Lab`	CIE Lab coordinates. An Nx3 matrix, or a vector that can be converted to such a matrix. Each row of the matrix has Lab.
`white`	XYZ for the source white - a numeric 3-vector with scaling irrelevant. `white` can also be a numeric 2-vector; which is interpreted as xy chromaticity and converted to XYZ with Y=100. `white` can also be the name of any standard illuminant recognized by `spacesXYZ::standardXYZ()` which is scaled so that Y=100. The default `'D65'` is converted to `c(95.047,100,108.883)`.
`adapt`	method for chromatic adaptation, see `spacesXYZ::CAT()` for valid values. Also see **Details**.
`...`	other parameters passed to `XYZtoMunsell()`

Details

The conversion is done in these steps.

- Lab → XYZ using `spacesXYZ::XYZfromLab()` with the given `white`.
- XYZ → HVC using `XYZtoMunsell()`. Before the Munsell conversion, XYZ is internally adapted from the given `white` to Illuminant C using the given `adapt` method.

Value

An Nx3 matrix with the Munsell HVC coordinates in each row. The rownames are set to those of Lab.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, a condition is signaled with one of these additional S3 classes: `"invalid_argument"` or `"package_unavailable"`, whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

Note

The case of the letter 't' in the function name was recently changed from lower to upper. The function `LabtoMunsell()` is equivalent but deprecated and provided for a limited time.

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

`MunsellToLab()`, `XYZtoMunsell()`, `spacesXYZ::standardXYZ()`, `spacesXYZ::CAT()`, `spacesXYZ::XYZfromLab()`

Examples

```
LabToMunsell( c(74.613450, -20.4, 10.1,    80, 0, 0) )
##                      H        V       C
##  3.1G 7.4/3.6 43.13641 7.379685 3.62976
##  N 7.9/        0.00000 7.945314 0.00000
```

`LuvToMunsell` *Convert CIE Luv coordinates to Munsell HVC*

Description

`LuvToMunsell` Converts CIE Luv coordinates to Munsell HVC, by interpolating over the extrapolated Munsell renotation data

Usage

```
LuvToMunsell( Luv, white='D65', adapt='Bradford', ... )
```

Arguments

`Luv`	CIE Luv coordinates An Nx3 matrix, or a vector that can be converted to such a matrix. Each row of the matrix has Luv.
`white`	XYZ for the source white - a numeric 3-vector with scaling irrelevant. `white` can also be a numeric 2-vector; which is interpreted as xy chromaticity and converted to XYZ with Y=100. `white` can also be the name of any standard illuminant recognized by `spacesXYZ::standardXYZ()` which is scaled so that Y=100. The default `'D65'` is converted to `c(95.047,100,108.883)`.
`adapt`	method for chromatic adaptation, see **Details**
`...`	other parameters passed to `XYZtoMunsell()`

Details

The conversion is done in these steps:

- Luv → XYZ using `spacesXYZ::XYZfromLuv()` with the given `white`.
- XYZ → HVC using `XYZtoMunsell()`. Before the conversion to HVC, XYZ is internally adapted from the given `white` to Illuminant C using the given chromatic adaptation method, see `spacesXYZ::CAT()`.

Value

An Nx3 matrix with the Munsell HVC coordinates in each row. The rownames are set to those of Luv.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, a condition is signaled with one of these additional S3 classes: `"invalid_argument"` or `"package_unavailable"`, whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

Note

The case of the letter 't' in the function name was recently changed from lower to upper. The function `LuvtoMunsell()` is equivalent but deprecated and provided for a limited time.

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

`XYZtoMunsell()`, `spacesXYZ::XYZfromLuv()`, `spacesXYZ::standardXYZ()`, `spacesXYZ::CAT()`

Examples

```
LuvToMunsell( c( 74.613450, -5.3108, 10.6,  55, 0, 0 ) )
##                      H        V        C
##  8.7GY 7.4/1.2 38.6599 7.383948 1.229376
##  N 5.4/         0.0000 5.395003 0.000000
```

`Munsell2xy` *The Munsell HVC to xy 3D Lookup Table*

Description

This is the discrete data for the Munsell Renotation System, which is often considered to be the most perceptually uniform color atlas. It was created by the NBS and OSA from "3,000,000 color judgments" by 40 observers.

Format

`Munsell2xy` is a data.frame with 4995 rows and these 6 columns:

- `H` the Munsell Hue. Each `H` is a multiple of 2.5 and in the interval (0,100].
- `V` the Munsell Value. Each `V` is an integer from 1 to 10, or one of 0.2, 0.4, 0.6, 0.8
- `C` the Munsell Chroma. Each `C` is a positive even integer.
- `x` the x chromaticity coordinate, for Illuminant C.
- `y` the y chromaticity coordinate, for Illuminant C.
- `real` a logical value. If `TRUE` then x,y were published; otherwise they have been extrapolated from the published values.

Note that the luminance factor Y is *not* here, since Y is a simple function of V, see `YfromV()`.

Details

All the (x,y) data here comes from the file `all.dat` downloaded from Rochester Institute of Technology, see **Source**. The file `real.dat` is a subset, and contains the (x,y) published in *Newhall, et. al. (1943)*. These rows have `real=TRUE` and are only for Value $\geq$ 1. There are 2734 of these.
Similarly, for Value<1 (very dark colors), (x,y) data from the paper *Judd et. al. (1956)* also have `real=TRUE`. There are 355 of these.
So `all.dat` has 4995 colors, of which 2734+355=3089 are "real" colors, and the remaining 1906 are extrapolated. I am confident that the extrapolation was done by *Schleter et. al. (1958)* at the NBS, and put online by the Rochester Institute of Technology. For more details, and the abstract of the 1958 article, see the **munsellinterpol User Guide**.

Note

For the purpose of this package, I have found that the extrapolated (x,y) for V$\geq$1 work well. But for V<1 they did not work so well, and I was able to get better results with my own extrapolation. Moreover, to get reliable results in this package for high Chroma, it was necessary to extrapolate past the data in `all.dat`.

Author(s)

Glenn Davis

Source

Rochester Institute of Technology. Program of Color Science. Munsell Renotation Data. https://www.rit.edu/science/munsell-color-lab

References

Newhall, Sidney M., Dorothy Nickerson, Deane B. Judd. **Final Report of the O.S.A. Subcommitte on the Spacing of the Munsell Colors**. Journal of the Optical Society of America. Vol. 33. No. 7. pp. 385-418. July 1943.

Judd, Deane B. and Gunter Wyszecki. **Extension of the Munsell Renotation System to Very Dark Colors**. Journal of the Optical Society of America. Vol. 46. No. 4. pp. 281-284. April 1956.

Schleter, J. C, D. B. Judd, D. B., H. J. Keegan. **Extension of the Munsell Renotation System (Abstract)**. J. Opt. Soc. Am. Vol 48. Num. 11. pp. 863-864. presented at the Forty-Third Annual Meeting of the Optical Society of America. Statler Hilton Hotel, Detroit, Michigan. October 9, 10, and 11, 1958.

See Also

`YfromV()`

Examples

```
str(Munsell2xy)
```

```
##  'data.frame': 4995 obs. of  6 variables:
##   $ H   : num  32.5 35 37.5 37.5 40 40 42.5 42.5 45 45 ...
```

```
##   $ V   : num  0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 ...
##   $ C   : num  2 2 2 4 2 4 2 4 2 4 ...
##   $ x   : num  0.713 0.449 0.262 -0.078 0.185 -0.257 0.144 -0.235 0.117 -0.209 ...
##   $ y   : num  1.414 1.145 0.837 2.16 0.676 ...
##   $ real: logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
```

MunsellBooks *The Munsell Chips present in Specialized and Full Munsell Books*

Description

In addition to the standard glossy and matte books, Munsell publishes books for specialized fields. For 5 of these specialized books, we use a paper by **Ferguson**, which gives the chip contents of books for these fields: **soil**, **rocks**, **beads**, and **plants**. It also includes **newstudent** *Color Set*, see **References**.
For the full **glossy** and **matte** books, see **Source**.

Format

MunsellBooks is a data.frame with 1753 rows and these columns:

HVC	a matrix with 3 columns: Hue, Value, and Chroma. For the chromatic chips, Hue is a positive multiple of 2.5
FergusonName	the color name of the chip, as assigned by **Ferguson**. If a chip does not appear in one of the 5 specialized bo
bead	a logical value. If TRUE then the chip appears in the **bead** book
newstudent	a logical value. If TRUE then the chip appears in the **newstudent** book
plant	a logical value. If TRUE then the chip appears in the **plant** book
rock	a logical value. If TRUE then the chip appears in the **rock** book
soil	a logical value. If TRUE then the chip appears in the **soil** book
glossy	a logical value. If TRUE then the chip appears in the **glossy** book
matte	a logical value. If TRUE then the chip appears in the **matte** book

The rows are sorted first by Hue, then Value, and then Chroma, all in increasing order. Thus the achromatic neutral chips come first, followed by the chromatics.

Details

The chip counts in the 7 books are:

Book	Neutral Chips	Chromatic Chips	Total Chips

Soil	10	427	437
Rock	9	106	115
Bead	5	171	176
Plant	0	332	332
New Munsell Student	8	238	246
Glossy	37	1485	1522
Matte	32	1269	1301

The union of all 7 books has 1753 chips. There are only 3 inclusions among the books: Rock $\subset$ Glossy, Plant $\subset$ Glossy, and New_Munsell_Student $\subset$ Matte.

Acknowledgements

The conversion of the 5 specialized books data to `./inst/extdata/Supplement1_3.6.2024.csv` was done by Willie Ondricek, who provided all the motivation for this function. He also kindly provided photographs of the soil book, and much other material.

Source

The data on the first 5 books of chips in the following table was converted from the file `yjfa_a_11710256_sm0001.docx`, which is **Online Supplement #1** from **Ferguson**.

For the Matte book, the 1269 chromatic chips are taken from:
`https://sites.uef.fi/spectral/databases-software/munsell-colors-matt-spectrofotometer-measured/`.
For the Glossy book, the 1485 chromatic chips are taken from:
`https://www.munsellcolorscienceforpainters.com/MunsellResources/MunsellResources.html /SpectralReflectancesOf2007MunsellBookOfColorGlossy.txt`.

References

Ferguson, Jonathan. *Munsell notations and color names: Recommendations for archaeological practice*. **Journal of Field Archaeology**. 39:4. 327-335. (2014). Online Supplement #1.

Long, J. T. *The New Munsell Student Color Set*. New York: Fairchild Books. 2011.

Munsell Color. *Munsell Soil-Color Charts*. Grand Rapids, MI: 2009.

Munsell Color. *Geological Rock-Color Chart*. GrandRapids, MI: 2009.

Munsell Color. *Munsell Bead Color Book*. Grand Rapids, MI: 2012.

Munsell Color. *Munsell Color Charts for Plant Tissues*. New Windsor, NY: 1977.

See Also

`roundHVC()`

Examples

```
str(MunsellBooks)

## 'data.frame':   1753 obs. of  9 variables:
## $ HVC         : num [1:1753, 1:3] 0 0 0 0 0 0 0 0 0 0 ...
```

```
##  ..- attr(*, "dimnames")=List of 2
##  .. ..$ : NULL
##  .. ..$ : chr [1:3] "H" "V" "C"
## $ FergusonName: chr  NA NA "Black" NA ...
## $ bead        : logi  FALSE FALSE TRUE FALSE FALSE FALSE ...
## $ newstudent  : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
## $ plant       : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
## $ rock        : logi  FALSE FALSE TRUE FALSE FALSE FALSE ...
## $ soil        : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
## $ glossy      : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
## $ matte       : logi  FALSE FALSE FALSE FALSE FALSE TRUE ...
```

`MunsellNameFromHVC` *Convert Munsell Numeric to Munsell String Notation*

Description

Convert Munsell Numeric to Munsell String Notation

Usage

```
MunsellNameFromHVC( HVC, format='g', digits=2, ctol=0 )

HueStringFromNumber( Hue, format='g', digits=2 )
```

Arguments

`HVC`	a numeric Nx3 matrix or a vector that can be converted to such a matrix. Each row has an HVC vector, where H is Hue Number, and V and C are the standard Munsell Value and Chroma. The Hue is automatically wrapped to the interval (0,100].
`Hue`	a numeric vector of Hue Numbers, which are automatically wrapped to the interval (0,100].
`format`	determines the meaning of the argument `digits`, and usually equal to `'g'` or `'f'` though other options are available, see `formatC()` for details. When `format='f'` trailing 0s might be displayed. For `MunsellNameFromHVC()`, `format` may have length 2, and in that case the first one is applied to Hue, and the second one is applied to Value and Chroma. It may also have length 3, and in that case the 3 values are applied to Hue, Value, and Chroma in that order.
`digits`	when `format='g'` the number of significant digits, and when `format='f'` the number of digits displayed after the decimal point. Both `format` and `digits` are passed as arguments to `formatC()`. For `MunsellNameFromHVC()`, `digits` may have length 2, and in that case the first one is applied to Hue, and the second one is applied to Value and Chroma. It may also have length 3, and in that case the 3 values are applied to Hue, Value, and Chroma in that order.

`ctol` a small tolerance that is used to "snap" a very small Chroma to 0, which is then converted to an exact Munsell Neutral, see **Examples**. The default value, `ctol=0` has no effect on the Chroma.

Value

Both functions return a character vector of length N. `MunsellNameFromHVC()` returns the full notation. `HueStringFromNumber()` returns just initial the hue part; which is useful for labeling plots.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, an error condition is signaled with the additional S3 class `"invalid_argument"`. The invalid argument is added to the condition object.

In case of an `WARN` event, a warning condition is signaled with the additional S3 class `"incomplete"` or `"munsell_clamp"` as appropriate. The condition object also has the submatrix of the invalid rows of `HVC`, or the subvector of invalid elements of `Hue`, as item `invalid`, and their indexes as item `indexes`.

Note

If `format='f'` for Chroma, then Chroma is first *rounded* to the given `digits`. Chromas close to 0 may then become 0 and be converted to an exact Munsell Neutral, see **Examples**.
The `width` argument of `formatC()` is always set to 1, to suppress leading spaces.

Author(s)

Glenn Davis and Hugo Rodrigues

References

ASTM D 1535-97. Standard Practice for Specifying Color by the Munsell System. 1997

See Also

`formatC()`, `HVCfromMunsellName()`, `HueNumberFromString()`

Examples

```
white.D65 = c(95.047,100.000,108.883)

HVC = XYZtoMunsell( c( 0.01*white.D65, 0.1*white.D65), white=white.D65 )
HVC
##                              H         V            C
##  6.6GY 0.86/2.8e-15 36.62033 0.8634072 2.753467e-15
##  N 3.7/              0.00000 3.7206664 0.000000e+00

# note that the Chroma in the 2nd row is 0 as it should be,
# but the Chroma in the first row is a tiny bit off.
# This is annoying when converted to Munsell notation, using defaults.
```

```
MunsellNameFromHVC( HVC )
##  [1] "6.6GY 0.86/2.8e-15" "N 3.7/"

# Convert again with a positive ctol.

MunsellNameFromHVC( HVC, ctol=1.e-4 )
##  [1] "N 0.86/" "N 3.7/"      # now *both* are converted to Munsell Neutrals


# setting format='f' can achieve a similar goal

MunsellNameFromHVC( c(39,5.1,7.3,  14.1234,5.1234,0.0003 )  )
##  [1] "9GY 5.1/7.3"      "4.1YR 5.1/0.0003"

MunsellNameFromHVC( c(39,5.1,7.34,  14.1234,5.1234,0.003 ), format='f' )
##  [1] "9.00GY 5.10/7.34" "N 5.12/"


#  On the other hand, one may *want* to see more precision for V and C, but not H
MunsellNameFromHVC( c(39,5.1,7.3,  14.1234,5.1234,0.0003 ), digits=c(2,6)  )
##  [1] "9GY 5.1/7.3"         "4.1YR 5.1234/0.0003"


# make nice labels for a plot with all the Hues
HueStringFromNumber( seq( 2.5, 100, by=2.5 ) )
##   [1] "2.5R"  "5R"    "7.5R"  "10R"   "2.5YR" "5YR"   "7.5YR" "10YR"  "2.5Y"
##  [10] "5Y"    "7.5Y"  "10Y"   "2.5GY" "5GY"   "7.5GY" "10GY"  "2.5G"  "5G"
##  [19] "7.5G"  "10G"   "2.5BG" "5BG"   "7.5BG" "10BG"  "2.5B"  "5B"    "7.5B"
##  [28] "10B"   "2.5PB" "5PB"   "7.5PB" "10PB"  "2.5P"  "5P"    "7.5P"  "10P"
##  [37] "2.5RP" "5RP"   "7.5RP" "10RP"
```

`MunsellSpecToColorlabFormat` *Convert Munsell Specification to Colorlab Format*

Description

Convert Munsell Specification to Colorlab Format

Usage

```
MunsellSpecToColorlabFormat( MunsellSpec )
```

Arguments

`MunsellSpec`	a numeric Nx3 matrix, or a vector that can be converted to such a matrix, by row. Each row of the matrix contains an HVC vector. H is automatically wrapped to the interval (0,100]. `MunsellSpec` can also be a character vector with Munsell Notation; which is converted to an Nx3 matrix using `HVCfromMunsellName()`.

Details

Colorlab Munsell format uses 4 numbers.

1. Hue Step, in the interval (0,10], or 0 for neutrals. In Colorlab documentation it is called the *hue shade*. It is also the Hue Number H mod 10 (unless H is an exact multiple of 10).
2. Munsell Value, in the interval [0,10]
3. Munsell Chroma, non-negative
4. Hue Index, an integer from 1 to 10, or 0 for neutrals. This index defines the *principal hue*, see **Details**.

Value

an Nx4 matrix, with rows as described in **Details**. Value and Chroma are simply copied unchanged. The complex part is conversion of Hue Number to Colorlab Hue Step and Hue Index. If Chroma is 0, both the Hue Step and Hue Index are set to 0. Invalid input values are converted to `NA`s.
If the input is a character vector, the rownames of the returned matrix are set to that vector.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, an error condition is signaled with the additional S3 class `"invalid_argument"`. The invalid argument is added to the condition object.

In case of an `WARN` event, a warning condition is signaled with the additional S3 class `"munsell_clamp"`. The condition object also has the submatrix of the invalid rows of `HVC` as item `invalid`, and their indexes as item `indexes`.

Note

The Colorlab format is closer to the *Munsell Book of Color (1929)* than HVC. In the book the hue circle is divided into 10 *principal hues* - 5 simple and 5 compound. The 10 hue labels are: **R, YR, Y, GY G, BG, B, PB, P** (simple are 1 letter and compound are 2 letters). In Colorlab these labels are replaced by the Hue Index. WARNING: In the Munsell System, see *Cleland*, there is a different Hue Index - **R** is 1, **YR** is 2, ..., **P** is 10. The Colorlab index has a different origin, and goes around the circle in a different direction !
Each one of these major hues corresponds to an arc on the circle, with a 10-point hue scale. The midpoint of the arc has hue step = 5. Fortunately this 10-point hue scale is exactly the same as the Colorlab Hue Step.

Author(s)

Jose Gama and Glenn Davis

References

Color Processing Toolbox. Colorlab 1.0. https://www.uv.es/vista/vistavalencia/software/colorlab.html

Cleland, T. M. **A Practical description of the Munsell Color System with Suggestions for its Use**. (1921)

See Also

HVCfromMunsellName(), ColorlabFormatToMunsellSpec()

Examples

```
MunsellSpecToColorlabFormat( c(100,5,5, 10,3,4, 90,4,3, 77,1,2, 66,2,0, 0,1,2 ) )
##                     HN V C HI
##  10.00RP 5.00/5.00 10 5 5  8
##  10.00R 3.00/4.00  10 3 4  7
##  10.00P 4.00/3.00  10 4 3  9
##  7.00PB 1.00/2.00   7 1 2 10
##  N 2.00/            0 2 0  0
##  10.00RP 1.00/2.00 10 1 2  8
```

MunsellToLab *Convert a Munsell specification to CIE Lab coordinates*

Description

MunsellToLab Converts a Munsell specification to CIE Lab coordinates, by interpolating over the extrapolated Munsell renotation data

Usage

```
MunsellToLab( MunsellSpec, white='D65', adapt='Bradford', ... )
```

Arguments

MunsellSpec	a numeric Nx3 matrix with HVC values in the rows, or a vector that can be converted to such a matrix, by row. MunsellSpec can also be a character vector with Munsell Notations, which is converted to an Nx3 matrix HVC using HVCfromMunsellName()
white	XYZ for the source white - a numeric 3-vector with scaling irrelevant. white can also be a numeric 2-vector; which is interpreted as xy chromaticity and converted to XYZ with Y=100. white can also be the name of any standard illuminant recognized by spacesXYZ::standardXYZ() which is scaled so that Y=100. The default 'D65' is converted to c(95.047,100,108.883).
adapt	method for chromatic adaptation, see spacesXYZ::CAT() for valid values. Also see **Details**.
...	other parameters passed to MunsellToXYZ()

Details

The conversion is done in these steps:

- HVC → XYZ using MunsellToXYZ(). XYZ is adapted from Illuminant C to the given white using spacesXYZ::adaptXYZ(). and the given chromatic adaptation method
- XYZ → Lab using spacesXYZ::LabfromXYZ() with the given white

Value

An Nx3 matrix with the Lab coordinates in each row. The rownames of Lab are copied from the input HVC matrix, unless the rownames are NULL and then the output rownames are the Munsell notations for HVC.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR event, a condition is signaled with one of these additional S3 classes: "invalid_argument" or "package_unavailable", whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

If any of the input Values or Chromas are out of range, a WARN event is issued, and a warning condition is signaled with the additional S3 class "munsell_clamp". The condition object also has the submatrix of the invalid rows of HVC as item invalid, and their indexes as item indexes.

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

LabToMunsell(), MunsellToXYZ(), spacesXYZ::LabfromXYZ(), spacesXYZ::standardXYZ(), spacesXYZ::adaptXYZ(), spacesXYZ::CAT()

Examples

```
MunsellToLab( c('7.6P 8.9/2.2', 'N 5/' ) )
##                       L       a         b
##  7.6P 8.9/2.2 89.19097 5.09879 -3.250468
##  N 5/         51.00375 0.00000  0.000000
```

MunsellToLuv *Convert a Munsell specification to CIE Luv coordinates*

Description

MunsellToLuv Converts a Munsell specification to CIE Luv coordinates, by interpolating over the extrapolated Munsell renotation data

Usage

```
MunsellToLuv( MunsellSpec, white='D65', adapt='Bradford', ... )
```

Arguments

MunsellSpec	a numeric Nx3 matrix with HVC values in the rows, or a vector that can be converted to such a matrix, by row. MunsellSpec can also be a character vector with Munsell Notations, which is converted to an Nx3 matrix using HVCfromMunsellName()
white	XYZ for the source white - a numeric 3-vector with scaling irrelevant. white can also be a numeric 2-vector; which is interpreted as xy chromaticity and converted to XYZ with Y=100. white can also be the name of any standard illuminant recognized by spacesXYZ::standardXYZ() which is scaled so that Y=100. The default 'D65' is converted to c(95.047,100,108.883).
adapt	method for chromatic adaptation, see spacesXYZ::CAT() for valid values. Also see **Details**.
...	other parameters passed to MunsellToXYZ()

Details

The conversion is done in these steps:

- HVC → XYZ using MunsellToXYZ(). In this function, XYZ is internally adapted from Illuminant C to the given white using spacesXYZ::adaptXYZ() with the given chromatic adaptation method.
- XYZ → Luv using spacesXYZ::LuvfromXYZ() with the given white

Value

An Nx3 matrix with the Luv coordinates in each row. The rownames of Luv are copied from the input HVC matrix, unless the rownames are NULL and then the output rownames are the Munsell notations for HVC.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR event, a condition is signaled with one of these additional S3 classes: "invalid_argument" or "package_unavailable", whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

If any of the input Values or Chromas are out of range, a WARN event is issued, and a warning condition is signaled with the additional S3 class "munsell_clamp". The condition object also has the submatrix of the invalid rows of HVC as item invalid, and their indexes as item indexes.

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

LuvToMunsell(), MunsellToXYZ(), spacesXYZ::standardXYZ(), spacesXYZ::CAT(), spacesXYZ::adaptXYZ(), spacesXYZ::LuvfromXYZ()

Examples

```
MunsellToLuv( c('7.6P 8.9/2.2', 'N 5/' ) )
##                       L        u         v
##  7.6P 8.9/2.2 89.19097 5.247155 -5.903808
##  N 5/         51.00375 0.000000  0.000000
```

MunsellToRGB *Convert a Munsell specification to RGB coordinates*

Description

MunsellToRGB Converts a Munsell specification to RGB coordinates, by interpolating over the extrapolated Munsell renotation data

Usage

```
MunsellToRGB( MunsellSpec, space='sRGB', which='scene', maxSignal=255,
    adapt='Bradford', ... )
```

Arguments

MunsellSpec	a numeric Nx3 matrix with HVC values in the rows, or a vector that can be converted to such a matrix, by row. MunsellSpec can also be a character vector with Munsell Notations, which is converted to an Nx3 matrix using HVCfromMunsellName().
space	the name of an installed RGB space. Spaces 'sRGB' and 'AdobeRGB' are pre-installed, and others can be installed with spacesRGB::installRGB().
which	either 'scene' or 'display'; passed to spacesRGB::RGBfromXYZ()
maxSignal	maximum of the non-linear signal RGB; Other popular values are 1, 100, 1023, and 65535
adapt	method for chromatic adaptation, see spacesXYZ::CAT() for valid values. Also see **Details**.
...	other parameters passed to MunsellToxyY()

Details

The conversion is done with these steps:

- HVC → xyY using MunsellToxyY() with .... This xyY is for Illuminant C.
- xyY → XYZ using spacesXYZ::XYZfromxyY()
- XYZ is adapted from Illuminant C to the white-point (with which) of the RGB space, using spacesXYZ::adaptXYZ(), with the given chromatic adaptation method adapt
- XYZ → RGB using spacesRGB::RGBfromXYZ() with the given space, which, and maxSignal

Value

a data.frame with these columns:

SAMPLE_NAME	a character N-vector equal to the original MunsellSpec if that was a character vector. Or the Munsell notation string converted from the input matrix HVC.
xyY	an Nx3 matrix with xyY values in the rows, that are adapted to Illuminant C. This is an intermediate result that is sometimes useful, e.g. it can be passed to IsWithinMacAdamLimits(). The rownames of xyY are copied from those of HVC, unless the rownames of HVC are NULL when they are set to SAMPLE_NAME.
RGB	an Nx3 matrix with non-linear RGB signal values in the rows. All values are clamped to the appropriate cube, e.g. $[0, 255]^3$. The rownames of RGB are copied from those of xyY.
OutOfGamut	logical vector, TRUE means the result was out of gamut (the cube) before clamping it

In case of error, it returns NULL.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR event, a condition is signaled with the additional S3 classes "package_unavailable". The name of the unavailable package is added to the condition object.

If any of the input Values or Chromas are out of range, a WARN event is issued, and a warning condition is signaled with the additional S3 class "munsell_clamp". The condition object also has the submatrix of the invalid rows of HVC as item invalid, and their indexes as item indexes.

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

MunsellToXYZ(), IsWithinMacAdamLimits(), spacesRGB::installRGB(), spacesRGB::RGBfromXYZ(), spacesXYZ::XYZfromxyY(), spacesXYZ::CAT()

Examples

```
MunsellToRGB( c('7.6P 8.9/2.2', 'N 3/'), space='AdobeRGB' )
##     SAMPLE_NAME      xyY.x      xyY.y      xyY.Y    RGB.R    RGB.G    RGB.B OutOfGamut
## 1 7.6P 8.9/2.2  0.3109520  0.3068719 74.6134498 227.72419 220.18659 229.23297     FALSE
## 2        N 3/  0.3101000  0.3163000  6.3911778  73.01793  73.01793  73.01793     FALSE
```

`MunsellTosRGB` *Convert a Munsell specification to sRGB coordinates*

Description

`MunsellTosRGB` Converts a Munsell specification to non-linear sRGB coordinates, by interpolating over the extrapolated Munsell renotation data

Usage

```
MunsellTosRGB( MunsellSpec, maxSignal=255, ... )
```

Arguments

`MunsellSpec`	a numeric Nx3 matrix with HVC values in the rows, or a vector that can be converted to such a matrix, by row. `MunsellSpec` can also be a character N-vector with Munsell notations, which is converted to an Nx3 matrix using `HVCfromMunsellName()`.
`maxSignal`	maximum of signal sRGB; Other popular values are 1, 100, 1023, and 65535
`...`	other parameters passed to `MunsellToxyY()`

Details

The conversion is done in these steps.

- HVC → xyY using `MunsellToxyY()` and the given `...`. This xyY is for Illuminant C.
- xyY → XYZ using `spacesXYZ::XYZfromxyY()`
- XYZ is adapted from Illuminant C to Illuminant D65 (from the sRGB standard) using `spacesXYZ::adaptXYZ()` and the *Bradford* chromatic adaptation method
- XYZ → sRGB using `spacesRGB::RGBfromXYZ()` with the given `maxSignal`

Value

a `data.frame` with these columns:

`SAMPLE_NAME`	the original `MunsellSpec` if that was a character vector. Or the Munsell notation string converted from the input matrix HVC.
`xyY`	an Nx3 matrix with xyY values in the rows, that are adapted to Illuminant C. This is an intermediate result that is sometimes useful, e.g. it can be passed to `IsWithinMacAdamLimits()`. The rownames of xyY are copied from those of HVC, unless the rownames of HVC are `NULL` when they are set to `SAMPLE_NAME`.
`RGB`	an Nx3 matrix with non-linear signal sRGB values in the rows. All values are clamped to the appropriate cube, e.g. $[0, 255]^3$. The rownames of RGB are copied from those of xyY.
`OutOfGamut`	logical vector, TRUE means the RGB result was out of gamut (the cube) before clamping it

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR event, a condition is signaled with one of these additional S3 classes: "invalid_argument" or "package_unavailable", whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

If any of the input Values or Chromas are out of range, a WARN event is issued, and a warning condition is signaled with the additional S3 class "munsell_clamp". The condition object also has the submatrix of the invalid rows of HVC as item invalid, and their indexes as item indexes.

Note

The more general function MunsellToRGB() also performs this conversion. The main reason to use MunsellTosRGB() is that it takes a little less time, since the CAT (using the *Bradford* method) is precomputed during base::.onLoad().

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

MunsellToXYZ(), MunsellToRGB(), spacesXYZ::XYZfromxyY(), spacesXYZ::CAT(), spacesXYZ::adaptXYZ(), spacesRGB::RGBfromXYZ(), IsWithinMacAdamLimits()

Examples

```
MunsellTosRGB( c('7.6P 8.9/2.2', 'N 3/') )
##     SAMPLE_NAME      xyY.x      xyY.y      xyY.Y     RGB.R     RGB.G     RGB.B OutOfGamut
##  1 7.6P 8.9/2.2  0.3109520  0.3068719 74.6134498 231.35746 221.14207 230.35011     FALSE
##  2        N 3/  0.3101000  0.3163000  6.3911778  71.50491  71.50491  71.50491     FALSE
```

MunsellToxyY *Convert Munsell HVC to xyY coordinates*

Description

MunsellToxyY Converts Munsell HVC to xyY coordinates, by interpolating over the extrapolated Munsell renotation data.

Note that the returned xyY is relative to the white point of Illuminant C. Since Illuminant C is obsolete, it is almost certain that if xyY is used in subsequent calculations, it must be chromatically adapted to a different white point. This chromatic adaptation can be done using spacesXYZ::adaptxyY(). Alternatively, one can use MunsellToXYZ(), which does the chromatic adaptation internally.

Usage

```
MunsellToxyY( MunsellSpec, xyC='NBS', hcinterp='bicubic', vinterp='cubic',
                    YfromV='ASTM', warn=TRUE )
```

Arguments

MunsellSpec	a numeric Nx3 matrix or a vector that can be converted to such a matrix. Each row has Munsell HVC, where H is Hue Number, and V and C are the standard Munsell Value and Chroma. The Hue is automatically wrapped to the interval (0,100]. MunsellSpec can also be a character N-vector with standard Munsell notation; it is converted to an Nx3 matrix.
xyC	a numeric 2-vector with xy chromaticity of Illuminant C. It can also be one of the strings in the first column of this table; it is then replaced by the corresponding xy in the second column.

	xy white point	reference
'NBS'	c(0.3101,0.3163)	Kelly, et. al. [RP1549] (1943). Rheinboldt et al. (1960)
'JOSA'	c(0.31012,0.31631)	Judd, Deane B. (1933)
'NTSC'	c(0.310,0.316)	NTSC (1953)
'CIE'	c(0.31006,0.31616)	CIE:15 2004

The default 'NBS' is probably what is intended by *Newhall et. al.* although no xy for C appears in that paper. This is the C used in the first computer program for conversion: *Rheinboldt et al. (1960)*. The other options are provided so that a neutral Munsell chip has the xy that the user expects. Alternative values of xyC should not be *too far* from the above. If hcinterp is 'bicubic', this parameter only affects chips with Chroma < 4 (except Chroma=2). If hcinterp is 'bilinear', this parameter only affects chips with Chroma < 2.

hcinterp	the type of 2D interpolation for H and C; either 'bicubic' or 'bilinear' (partial matching enabled). In the bicubic case, for a general input point, the output value is interpolated using a 4x4 subgrid of the lookup table, and the interpolation function is class C^1 (except at the neutrals). In the bilinear case, the interpolation uses a 2x2 subgrid, and the function is class C^0.
vinterp	the type of interpolation for V; either 'cubic' or 'linear' (partial matching enabled). In the cubic case, for a general input point, the output value is interpolated using 4 planes of constant Value, and the interpolation function is class C^1. In the linear case, the interpolation uses 2 planes and the function is class C^0.
YfromV	passed as the parameter which to the function YfromV(). See YfromV() for details. Option 'MGO' is not allowed because then Y>100 when V=10.
warn	if a chip cannot be mapped (usually because the Chroma is too large), its x and y are set to NA in the returned data.frame. Just before returning, if any rows have NA, and this argument is TRUE, then a warning is logged.

Details

In case `hcinterp='bicubic'` or `vinterp='cubic'` a Catmull-Rom spline is used; see the article *Cubic Hermite spline*. This spline has the nice property that it is *local* and requires at most 4 points. And if the knot spacing is uniform: 1) the resulting spline is C^1, 2) if the knots are on a line, the interpolated points are on the line too.

Value

a `data.frame` with these columns:

`SAMPLE_NAME`	the original `MunsellSpec` if that was a character vector. Or the Munsell notation string converted from the input matrix HVC.
`HVC`	the input Nx3 matrix, or the HVC matrix converted from the input Munsell notation
`xyY`	the computed output matrix, with CIE xyY of MunsellSpec, relative to the white point of Illuminant C. In case of error, x and y are set to `NA`. The rownames of xyY are set to those of HVC, unless they are NULL when they are set to `SAMPLE_NAME`.

Warning

Even when `vinterp='cubic'` the function HVC $\rightarrow$ xyY is not C^1 on the plane V=1. This is because of a change in Value spacing: when V$\geq$1 the Value spacing is 1, but when V$\leq$1 the Value spacing is 0.2.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, an error condition is signaled with the additional S3 class `"invalid_argument"`. The invalid argument is added to the condition object.

In the case that `warn=TRUE`, and not all xyY rows could be computed, a `WARN` event is created, and a warning condition is signaled with the additional S3 class `"incomplete"` or `"munsell_clamp"`, as appropriate. The condition object also has the submatrix of invalid `HVC`'s as item `invalid`, and their row indexes as item `indexes`.

Note

When making plots in planes of constant Value, option `hcinterp='bicubic'` makes fairly smooth ovals, and `hcinterp='bilinear'` makes polygons. The ovals are smooth even when `vinterp='linear'`, but the function is not class C^1 at the planes of integer Value. To get a fully C^1 function (except at the neutrals and on the plane V=1), `hcinterp` and `vinterp` must be set to the defaults.

Author(s)

Jose Gama and Glenn Davis

Source

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK
https://www.rit.edu/science/munsell-color-lab
https://www.rit-mcsl.org/MunsellRenotation/all.dat
https://www.rit-mcsl.org/MunsellRenotation/real.dat

References

Judd, Deane B. **The 1931 I.C.I. Standard Observer and Coordinate System for Colorimetry**. Journal of the Optical Society of America. Vol. 23. pp. 359-374. October 1933.

Newhall, Sidney M., Dorothy Nickerson, Deane B. Judd. **Final Report of the O.S.A. Subcommitte on the Spacing of the Munsell Colors**. Journal of the Optical Society of America. Vol. 33. No. 7. pp. 385-418. July 1943.

Kelly, Kenneth L. Kasson S. Gibson. Dorothy Nickerson. **Tristimulus Specification of the Munsell Book of Color from Spectrophometric Measurements** National Bureau of Standards RP1549 Volume 31. August 1943.

Judd, Deane B. and Günther Wyszecki. **Extension of the Munsell Renotation System to Very Dark Colors**. Journal of the Optical Society of America. Vol. 46. No. 4. pp. 281-284. April 1956.

National Television System Committee. [Report and Reports of Panel No. 11, 11-A, 12-19, with Some supplementary references cited in the Reports, and the Petition for adoption of transmission standards for color television before the Federal Communications Commission] (1953)

Rheinboldt, Werner C. and John P. Menard. **Mechanized Conversion of Colorimetric Data to Munsell Renotations**. Journal of the Optical Society of America. Vol. 50, Issue 8, pp. 802-807. August 1960.

Wikipedia. Cubic Hermite spline. https://en.wikipedia.org/wiki/Cubic_Hermite_spline

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

xyYtoMunsell(), MunsellToXYZ(), spacesXYZ::adaptxyY()

Examples

```
MunsellToxyY( '7.6P 8.9/2.2' )
##    SAMPLE_NAME  HVC.H HVC.V HVC.C      xyY.x      xyY.y      xyY.Y
##  1 7.6P 8.9/2.2  87.6   8.9   2.2  0.3109520  0.3068719 74.6134498
```

MunsellToXYZ *Convert a Munsell specification to CIE XYZ coordinates*

Description

`MunsellToXYZ` Converts a Munsell specification to XYZ coordinates, by interpolating over the extrapolated Munsell renotation data.

Note that *by default* the returned `XYZ` is relative to the white point of Illuminant C. Since Illuminant C is obsolete, it is almost certain that if this `XYZ` is used in subsequent calculations, it must be chromatically adapted to a different white point.

But if the white point for subsequent calculations is supplied, then the chromatic adaptation is done internally by `spacesXYZ::adaptXYZ()`.

Usage

```
MunsellToXYZ( MunsellSpec, white=NULL, adapt='Bradford', ... )
```

Arguments

`MunsellSpec`	a numeric Nx3 matrix with HVC values in the rows, or a vector that can be converted to such a matrix, by row. `MunsellSpec` can also be a character vector with Munsell Notations, which is converted to an Nx3 matrix using `HVCfromMunsellName()`.
`white`	XYZ for the output white point. If `NULL` then the output XYZ are for viewing in an environment with Illuminant C, and the output `XYZ` is not chromatically adapted. If `white` is not `NULL`, then the interpolated `XYZ` is chromatically adapted from the white point of Illuminant C to the given white point. `white` can be a numeric 3-vector; by convention `Y=100` for Munsell work. `white` can also be a numeric 2-vector; which is interpreted as xy chromaticity and converted to XYZ with Y=100. `white` can also be the name of any standard illuminant recognized by `spacesXYZ::standardXYZ()` which is then scaled so that Y=100.
`adapt`	method for chromatic adaptation; for valid values see `spacesXYZ::CAT()`. Also see **Details**. If `white=NULL` then `adapt` is ignored.
`...`	other parameters passed to `MunsellToxyY()`

Details

The conversion is done in these steps:

- HVC → xyY using `MunsellToxyY()` and `...`. This xyY is for Illuminant C.
- xyY → XYZ using `spacesXYZ::XYZfromxyY()`
- If `white` is not `NULL`, XYZ is chromatically adapted from the white point of Illuminant C to the given `white` using the given `adapt` method.

Value

an Nx3 matrix with XYZ values in the rows. The rownames of XYZ are copied from the input HVC matrix, unless the rownames are `NULL` and then the output rownames are the Munsell notations for HVC.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR event, a condition is signaled with one of these additional S3 classes: `"invalid_argument"` or `"package_unavailable"`, whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

If any of the input Values or Chromas are out of range, a WARN event is issued, and a warning condition is signaled with the additional S3 class `"munsell_clamp"`. The condition object also has the submatrix of the invalid rows of HVC as item `invalid`, and their indexes as item `indexes`.

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

`MunsellToxyY()`, `spacesXYZ::XYZfromxyY()`, `spacesXYZ::adaptXYZ()`

Examples

```
## make expected XYZ for a Perfect Reflecting Diffuser, under Illuminant D65
white.D65 = 100 * as.numeric( spacesXYZ::standardXYZ( 'D65' ) )
white.D65
##  [1]  95.047 100.000 108.883

## convert Munsell 'N 10/' - the Perfect Reflecting Difuser - to XYZ
MunsellToXYZ( "N 10/" )
##               X   Y        Z
##  N 10/ 98.03984 100 118.1157
##  this is the white point of Illuminant C, and not equal to white.D65

## Redo and supply the true white point.
MunsellToXYZ( "N 10/", white=white.D65 )
##             X   Y       Z
##  N 10/ 95.047 100 108.883
##
##  now we have a match


MunsellToXYZ('7.6P 8.9/2.2')
##                     X        Y        Z
##  7.6P 8.9/2.2 75.6055 74.61345 92.92308
```

`NickersonColorDifference` *Calculate the Nickerson Color Difference between two Colors*

Description

Calculate the Nickerson Color Difference between two colors, given in Munsell HVC; see *Nickerson*.

Usage

```
NickersonColorDifference( HVC0, HVC1, symmetric=TRUE, coeffs=c(0.4,6,3) )
```

Arguments

`HVC0`	a numeric Nx3 matrix with HVC values in the rows, or a vector that can be converted to such a matrix, by row. `HVC0` can also be a numeric 3-vector with a single HVC, and it is then replicated to match the size of `HVC1`. `HVC0` can also be a character N-vector with Munsell Notations, which is converted to an Nx3 matrix using `HVCfromMunsellName()`.
`HVC1`	a numeric Nx3 matrix with HVC values in the rows, or a vector that can be converted to such a matrix, by row. `HVC1` can also be a numeric 3-vector with a single HVC, and it is then replicated to match the size of `HVC0`. `HVC1` can also be a character N-vector with Munsell Notations, which is converted to an Nx3 matrix using `HVCfromMunsellName()`.
`symmetric`	if `FALSE` then use the original Nickerson difference formula, and if `TRUE` then use a symmetrized version; see **Details**.
`coeffs`	the coefficients for Hue, Value, and Chroma in the difference equation, see **Details**. The defaults are the ones used by Nickerson, but tweaking them is possible in special circumstances. The argument must have length 3 and all coefficents positive.

Details

If `HVC0`=H_0, V_0, C_0 and If `HVC1`=H_1, V_1, C_1 then the original Nickerson formula is:

$$NCD(\texttt{HVC0}, \texttt{HVC1}) = 0.4C_0\Delta H \ + \ 6\Delta V \ + \ 3\Delta C$$

where $\Delta H = |H_0 - H_1|$ (on the circle), $\Delta V = |V_0 - V_1|$ and $\Delta C = |C_0 - C_1|$. Unfortunately, if `HVC0` and `HVC1` are swapped, the color difference is different. The first color is considered to be the *reference color* and the second one is the *test color*. The difference is not symmetric.
Another problem is that the difference is not continuous when the second color is a neutral gray, for rectangular coordinates on a plane of constant V.

Both of these problems are fixed with a slightly modified formula:

$$NCD(\texttt{HVC0}, \texttt{HVC1}) = 0.4\min(C_0, C_1)\Delta H \ + \ 6\Delta V \ + \ 3\Delta C$$

For the first formula set `symmetric=FALSE` and for the second formula set `symmetric=TRUE`.

Value

A numeric N-vector with the pairwise differences, i.e. between row i of `HVC0` and row i of `HVC1`.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, an error condition is signaled with the additional S3 class `"invalid_argument"`. The invalid argument is added to the condition object.

In the case that not all differences could be computed, a warning condition is signaled with the additional S3 class `"incomplete"`. The condition object also has the indexes of the invalid differences as item `indexes`.

If any of the input Values or Chromas are out of range, a `WARN` event is issued, and a warning condition is signaled with the additional S3 class `"munsell_clamp"`. The condition object also has the submatrix of the invalid rows of `HVC0` or `HVC1` as item `invalid`, and their indexes as item `indexes`.

Author(s)

Jose Gama and Glenn Davis

References

Nickerson, Dorothy. The Specification of Color Tolerances. **Textile Research**. Vol 6. pp. 505-514. 1936.

Examples

```
NickersonColorDifference( '7.6P 8.9/2.2', '8P 8.2/3'  )
##  [1] 6.952
```

---

plotLociHC *Plot Curves of Constant Munsell Hue and Chroma*

Description

Plot Curves of Constant Munsell Hue and Chroma

Usage

```
plotLociHC( value=5, hue=seq(2.5,100,by=2.5), chroma='auto', coords='xy',
                        main="Value %g/", est=FALSE, ... )
```

Arguments

value	a Munsell value for which the plot is created. It must be in the interval (0,10]. value can also be a numeric vector of such numbers, and then a separate plot is made for each element of the vector.
hue	a numeric vector for which curves of contant Hue are plotted. Each of these radial curves starts at Munsell Chroma = min(chroma,1) and extends to max(chroma). hue can also be a character vector, which is then converted to a numeric vector using HueNumberFromString().
chroma	a numeric vector for which ovoids of constant Chroma are plotted. Each of these ovoids is closed; i.e. goes full circle from Hue=0 to Hue=100. If chroma='auto' then a vector is chosen appropriate for the current value.
coords	either 'xy' or 'ab'. If coords='xy' then the plots are in the standard xy chromaticity plane. If coords='ab' then the plots are in the ab chrominance plane, from Lab. Even when it is 'ab', the interpolation of loci takes place in 'xy' *before* transformation to Lab.
main	a string used to set the main title of the plot. The optional placeholder '%g' is replaced by the current value.
est	if TRUE, initial estimates for the iteration used in xyYtoMunsell() are plotted
...	other arguments passed to the function MunsellToxyY(). This includes hcinterp, vinterp, and xyC. However warn=FALSE is forced.

Details

The plot limits (xlim and ylim) are set to include all points where the Hue radials intersect the Chroma ovoids, plus the white point.
If value is one of {0.2,0.4,0.6,0.8,1,2,3,4,5,6,7,8,9,10} then published points from real.dat are plotted with filled black points (real points), and extrapolated points from all.dat are drawn with open circles (unreal points).

Value

TRUE for success and FALSE for failure.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR or FATAL event, an error condition is signaled with one of these additional S3 classes: "invalid_argument" or "package_unavailable" or "internal_error", whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

In case of an WARN event, a warning condition is signaled with the additional S3 class "incomplete". The condition object also has the the subvector of Value's that could not be plotted as item invalid, and their indexes as item indexes.

Note

The option `hcinterp='bicubic'` makes fairly smooth ovoids, and `hcinterp='bilinear'` makes 40-sided polygons (when `coords='xy'`). Compare with the plots in *Newhall et. al. (1943)*, *Judd, et. al. (1956)*, and *Judd, et. al. (1975)* p. 263.

Author(s)

Glenn Davis

References

Newhall, Sidney M., Dorothy Nickerson, Deane B. Judd. **Final Report of the O.S.A. Subcommitte on the Spacing of the Munsell Colors**. Journal of the Optical Society of America. Vol. 33. No. 7. pp. 385-418. July 1943.

Judd, Deane B. and GC<nther Wyszecki. **Extension of the Munsell Renotation System to Very Dark Colors**. Journal of the Optical Society of America. Vol. 46. No. 4. pp. 281-284. April 1956.

Judd, Deane B. and GC<nther Wyszecki. **Color in Business, Science, and Industry**. 3rd edition. John Wiley & Sons. 1975.

See Also

`MunsellToxyY()`, `HueNumberFromString()`

---

plotPatchesH *Plot Colored Patches for a fixed Munsell Hue*

Description

This plot simulates a page from the Munsell Book of Color. The colors are best viewed on a display calibrated for the RGB space given as the second argument.

Usage

```
plotPatchesH( hue, space='sRGB', adapt='Bradford', background='gray50',
                        main="Hue %s  (H=%g)      [%s   adapt=%s]",
                        value=NULL, chroma=NULL, ... )
```

Arguments

hue	a Munsell hue for which the plot is created. It is automatically wrapped to the interval (0,100]. It does not have to be a multiple of 2.5. hue can also be a numeric vector of such numbers, and then a separate plot is made for each element of the vector. hue can also be a character vector of Hue Names, which is then converted to a numeric vector using `HueNumberFromString()`.
space	the name of an installed RGB space. Spaces `'sRGB'` and `'AdobeRGB'` are pre-installed, and others can be installed using `spacesRGB::installRGB()`

adapt	method used to adapt xyY for Illuminant C to xyY for Illuminant D65. It is passed to MunsellToRGB().
background	background color for the plot. It is passed to par() as argument bg.
main	a string used to set the main title of the plot. The optional placeholder '%s' is replaced by the Hue Name, '%g' is replaced by the Hue Number, the next '%s' is replaced by space, and the last '%s' is replaced by adapt.
value	a vector of Munsell Values to use for the plot. Values must be in the interval [0,10], and in strictly increasing order. Non-integers, such as 2.5 and 8.5, are valid. If value=NULL, then it is set to 0:10.
chroma	a vector of Munsell Chromas to use for the plot. Chromas must be non-negative and in strictly increasing order. Odd integers and non-integers are valid. If chroma=NULL, then a sequence of even numbers is chosen, starting at 0 and ending at the Chroma limit for the given Hue page, see **Details**.
...	other arguments passed to the function MunsellToRGB(). This includes hcinterp, vinterp, and xyC.

Details

If chroma=NULL, for the closest discrete Hue in real.dat, the patches in real.dat are transformed to xyY using simple lookup with no interpolation. These are then tested against the MacAdam Limits for Illuminant C using IsWithinMacAdamLimits(). The patches outside the limits are discarded, and the maximum Chroma of the remaining patches, which is always an even integer, determines chroma.
Patches inside the MacAdam Limits can still be outside the RGB cube; those patches are drawn in outline only, and with the clamped RGB coordinates printed inside.

Value

TRUE for success and FALSE for failure.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR or FATAL event, an error condition is signaled with one of these additional S3 classes: "invalid_argument" or "package_unavailable" or "internal_error", whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

In case of an WARN event, a warning condition is signaled with no additional S3 class.

Author(s)

Glenn Davis

See Also

MunsellToRGB(), HueNumberFromString(), IsWithinMacAdamLimits(), spacesRGB::installRGB()

RGBtoMunsell *Convert RGB coordinates to Munsell HVC*

Description

RGBtoMunsell Converts RGB coordinates to Munsell HVC, by interpolating over the extrapolated Munsell renotation data

Usage

```
RGBtoMunsell( RGB, space='sRGB', which='scene', maxSignal=255, adapt='Bradford', ... )
```

Arguments

RGB	a numeric Nx3 matrix with RGB coordinates in the rows, or a vector that can be converted to such a matrix, by row. These are non-linear display values, but they are not required to be integers.
space	the name of an installed RGB space. Spaces 'sRGB' and 'AdobeRGB' are pre-installed, and others can be installed using spacesRGB::installRGB().
which	either 'scene' or 'display'; passed to spacesRGB::XYZfromRGB()
maxSignal	maximum value of signal RGB. Other popular values are 1, 100, 1023, and 65535. Even when 1, they are still taken to be non-linear signal values.
adapt	method for chromatic adaptation, see spacesXYZ::CAT() for valid values. Also see **Details**.
...	other parameters passed to XYZtoMunsell()

Details

The conversion is done in these steps:

- RGB → XYZ using spacesRGB::XYZfromRGB() with the given space, which, and maxSignal
- XYZ is adapted from the white-point (with which) of space to Illuminant C, using spacesXYZ::adaptXYZ(), with the given chromatic adaptation method adapt
- XYZ → HVC using XYZtoMunsell() with ...

Value

a numeric Nx3 matrix with HVC coordinates in the rows. The rownames are copied from input RGB to output HVC, unless the rownames are NULL when they are set to the Munsell notations for HVC.
In case of error, it returns NULL.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR event, a condition is signaled with the additional S3 classes "package_unavailable". The name of the unavailable package is added to the condition object.

Author(s)

Jose Gama and Glenn Davis

References

Wikipedia. **sRGB**. https://en.wikipedia.org/wiki/SRGB.

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

XYZtoMunsell(), spacesXYZ::CAT(), spacesXYZ::adaptXYZ(), spacesRGB::XYZfromRGB()

Examples

```
RGBtoMunsell( c(255,45,67) )
##                      H        V        C
##  5.4R 5.5/18 5.401135 5.477315 18.01984

RGBtoMunsell( c(255,45,67), space='Adobe' )
##                      H        V        C
##  5.9R 6.2/22 5.924749 6.214155 21.83907
```

---

roundHVC *Round a Given Munsell HVC to the Closest Chip in One or More Books of Chips*

Description

The functions in this package compute Munsell HVC coordinates with high precision. But when reporting the Munsell notation it is often desirable, for historical consistency or other reasons, to report a notation that is actually appears in one or more published books of Munsell chips. This rounding operation requires exact knowledge of such books, which is taken from the data frame MunsellBooks.

Usage

```
roundHVC( HVC, books )
```

Arguments

HVC	a numeric Nx3 matrix with Munsell HVC values in the rows, or a vector that can be converted to such a matrix, by row. HVC can also be a character N-vector with Munsell Notations, which is converted to an Nx3 matrix using HVCfromMunsellName().
books	a comma-delimited string that specifies one or more books of chips. The words in the string can be 'soil', 'rock', 'bead', 'plant', 'newstudent', 'glossy', and 'matte'. Matching is partial (initial substring only) and case-insensitive. See **References** and **Examples**.

Details

First, the set of chips over all selected books is formed - a sub-frame of MunsellBooks. Then the closest chip in this set to the given HVC is found. For the color distance, the function NickersonColorDifference() is used. Unfortunately, the particular coefficients in this difference equation can lead to large regions of Munsell color space with exact ties for the two closest chips. These exact ties make the final result unpredictable and sometime unintuitive. As a tie-breaker, a very small multiple (1.e-6) of plain Eucliden distance is added to the Nickerson difference, which has proven effective.

Value

A data.frame with a N rows and these columns:

HVC	the given Nx3 input matrix HVC.
ISCC-NBS Name	the ISCC-NBS Name (the color name) of the given input chip, see ColorBlockFromMunsell().
MunsellRounded	the Munsell notation that is rounded from HVC. It is guaranteed to be in one of the books given by the argument books.
FergusonName	the color name of MunsellRounded, as assigned by **Ferguson**. If the chip does not appear in one of the 5 specialized books in MunsellBooks, it is set to NA.

The row names are set to the row names of HVC. But if these are NULL they are set to the Munsell notation of the input HVC. Finally, if these row names have duplicates, they are set to 1:N.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an ERROR or FATAL event, a condition is signaled with one of these additional S3 classes: "invalid_argument" or "internal_error", whichever is appropriate. The invalid argument is added to the condition object.

If any of the input Values or Chromas are out of range, a WARN event is issued, and a warning condition is signaled with the additional S3 class "munsell_clamp". The condition object also has the submatrix of the invalid rows of HVC as item invalid, and their indexes as item indexes.

Author(s)

Glenn Davis

References

Ferguson, Jonathan. *Munsell notations and color names: Recommendations for archaeological practice*. **Journal of Field Archaeology**. 39:4. 327-335. (2014). Online Supplement #1.

See Also

MunsellBooks, ColorBlockFromMunsell()

Examples

```
## search the soil book for the best match for 3 chips
roundHVC( c("7.7YR 3.4/6.1", "2.6PB 6.1/4.5", "N 6.6/" ), books='soil' )

##               HVC.H HVC.V HVC.C ISCC-NBS Name MunsellRounded Ferguson Name
## 7.7YR 3.4/6.1  17.7   3.4   6.1  strong brown      7.5YR 4/6  Strong brown
## 2.6PB 6.1/4.5  72.6   6.1   4.5     pale blue        5PB 6/1   Bluish gray
## N 6.6/          0.0   6.6   0.0    light gray           N 7/    Light gray

## The middle chip has a poor match, since "purplish-blue" with Chroma=4.5
## is not well-matched by soil color chips.
## Search again, but this time in 5 books.

roundHVC( c("7.7YR 3.4/6.1", "2.6PB 6.1/4.5", "N 6.6/" ), books='s,b,r,p,n' )

##               HVC.H HVC.V HVC.C ISCC-NBS Name MunsellRounded Ferguson Name
## 7.7YR 3.4/6.1  17.7   3.4   6.1  strong brown      7.5YR 4/6  Strong brown
## 2.6PB 6.1/4.5  72.6   6.1   4.5     pale blue        5PB 6/4     Pale blue
## N 6.6/          0.0   6.6   0.0    light gray           N 7/    Light gray

## This time, for the middle chip, a better match was found.
## The other 2 chip are unchanged.
```

`sRGBtoMunsell` *Convert sRGB coordinates to Munsell HVC*

Description

Converts non-linear sRGB coordinates to Munsell HVC, by interpolating over the extrapolated Munsell renotation data

Usage

```
sRGBtoMunsell( sRGB, maxSignal=255, ... )
```

Arguments

`sRGB`	a numeric Nx3 matrix with signal sRGB coordinates in the rows, or a vector that can be converted to such a matrix, by row. These are non-linear signal values, but they are not required to be integers.
`maxSignal`	maximum value of signal sRGB. Other popular values are 1, 100, 1023, and 65535. Even when 1, they are still taken to be non-linear signal values.
`...`	other parameters passed to `XYZtoMunsell()`

Details

The conversion is done in these steps:

- sRGB → XYZ using `spacesXYZ::XYZfromRGB()` with the given `maxSignal`
- XYZ is adapted from Illuminant D65 (from the sRGB standard) to Illuminant C, using `spacesXYZ::adaptXYZ()` with the *Bradford* chromatic adaptation method
- XYZ → HVC using `XYZtoMunsell()` and the given ...

Value

a numeric Nx3 matrix with HVC coordinates in the rows. The rownames are copied from input to output, unless the rownames are `NULL` when they are set to the Munsell notations for HVC.
In case of error, it returns `NULL`.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, a condition is signaled with one of these additional S3 classes: `"invalid_argument"` or `"package_unavailable"`, whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

Note

The more general function `RGBtoMunsell()` also performs this conversion. This function has the advantage that it takes a little less time, since the CAT (using the *Bradford* method) is precomputed during `base::.onLoad()`. But it has the disadvantage that the chromatic adaptation method cannot be changed.

Author(s)

Jose Gama and Glenn Davis

References

Wikipedia. **sRGB**. https://en.wikipedia.org/wiki/SRGB.

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

`RGBtoMunsell()`, `XYZtoMunsell()`, `spacesXYZ::CAT()`, `spacesXYZ::adaptXYZ()`, `spacesXYZ::XYZfromRGB()`

Examples

```
sRGBtoMunsell( c(255,45,67) )
##                      H        V        C
##  5.4R 5.5/18 5.401135 5.477315 18.01984

sRGBtoMunsell( c(1,0,1), maxSignal=1 )
##                  H        V        C
##  8P 6/26 87.98251 5.981297 25.64534
```

VandY *Convert Munsell Value V to Luminance Factor Y, and back again*

Description

Convert non-linear Munsell Value V to linear Luminance Factor Y, and back again

Usage

```
YfromV( V, which='ASTM' )
VfromY( Y, which='ASTM' )
```

Arguments

V	a numeric vector with elements in the interval [0,10]
Y	a numeric vector with elements in the interval [0,100]
which	one of the strings in the first column of this table

	reference
'ASTM'	ASTM D-1535 (2008)
'OSA'	Newhall, et. al. (1943)
'MgO'	Newhall, et. al. (1943)
'Munsell'	Munsell, et. al. (1933)
'Priest'	Priest, et. al. (1920)

Partial matching is enabled, and it is case insensitive.

Details

'Priest' is the earliest (1920) transfer function in this package. It is implemented as:

$$V = \mathrm{sqrt}(Y) \quad and \quad Y = V^2$$

One readily checks that when V=10, Y=100, and vice-versa. This transfer function has been implemented in colorimeters, using analog electric circuits. It is used in Hunter Lab - the precursor of CIE Lab.

'Munsell' is the next (1933) transfer function, and was proposed by Munsell's son (Alexander Ector Orr Munsell) and co-workers. It is implemented as:

$$V = \mathrm{sqrt}(1.474 * Y - 0.00474 * Y^2)$$

$$Y = 50 * ((1474 - \mathrm{sqrt}(1474^2 - 4 * 4740 * V^2))/474)$$

One readily checks that when V=10, Y=100, and vice-versa. The luminance factor Y is *absolute*, AKA *relative to the perfect reflecting diffuser*.

'Priest' and 'Munsell' are included in this package for historical interest only.

The remaining three define Y as a quintic polynomial in V.

The next one historically - `'MgO'` - is implemented as:

$$Y = (((((8404 * V - 210090) * V + 2395100) * V - 2311100) * V + 10000000) * V)/10000000$$

One readily checks that when V=10, Y=102.568. This Y is larger than 100, because the authors decided to make Y relative to a clean surface of MgO, instead of the perfect reflecting diffuser. In their words:

- It should be noted that the reflectances indicated are not absolute but relative to magnesium oxide; whereas the maximum at value 10/ was formerly 100 percent, it is now 102.57. Use of this relation facilitates results and also avoids the somewhat dubious conversion to absolute scale, by permitting Y determinations with a MgO standard to be converted directly to Munsell value.

Nowadays, the perfect reflecting diffuser is preferred over MgO. For users who would like to modify this quintic as little as possible, with the perfect reflecting diffuser in mind (going back to `'MUNSELL'`), I offer `'OSA'`, which is given by this quintic of my own design

$$Y = (((((8404 * V - 210090) * V + 2395100) * V - 2311100) * V + 10000000) * V)/10256800$$

ASTM had a similar modification in mind, but did it a little differently by scaling each coefficient. `'ASTM'` is given by this quintic:

$$Y = (((((81939 * V - 2048400) * V + 23352000) * V - 22533000) * V + 119140000) * V)/1.e8$$

One readily checks that when V=10, Y=100 exactly (for both `'OSA'` and `'ASTM'`).

The inverses - from Y to V - of all 3 quintics are implemented as 3 `splinefun()`s at a large number (about 300) of points. These inverses are both fast and accurate. The round-trip $Y \rightarrow V \rightarrow Y$ is accurate to 7 digits after the decimal. The round-trip $V \rightarrow Y \rightarrow V$ is accurate to 8 digits after the decimal.

Value

a numeric vector the same length as the input

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` or `FATAL` event, a condition is signaled with one of these additional S3 classes: `"invalid_argument"` or `"internal_error"`, whichever is appropriate.

In `VfromY()` if any computed `V`'s are negative, those `V`'s are clamped to 0 and a condition is signaled with additional S3 class `"munsell_clamp"`. The condition object list also has the vector of negative `V`'s as item `Value`, and their indexes as item `indexes`.

Similarly, in `YfromV()` if any computed `Y`'s are negative, those `Y`'s are clamped to 0 and a similar condition is signaled.

Note

The quintic functions `'ASTM'` and `'OSA'` are very close. They agree at the endpoints 0 and 10 exactly, and the largest difference is near V=6.767 where they differ by about 0.0007.

Author(s)

Glenn Davis and Hugo Rodrigues

References

Priest, I. G. Gibson, K. S. and McNicholas, H. J. **An Examination of the Munsell Color System. I. Spectral and and Total Reflection and the Munsell Scale of Value**. Technologic Papers of the Bureau of Standards, No. 167. pp. 1-33. Washington D.C. 1920.

Munsell, A. E. O., L. L. Sloan, and I. H. Godlove. **Neutral Value Scales. I. Munsell Neutral Value Scale**. Journal of the Optical Society of America. Vol. 23. Issue 11. pp. 394-411. November 1933.

Newhall, Sidney M., Dorothy Nickerson, Deane B. Judd. **Final Report of the O.S.A. Subcommitte on the Spacing of the Munsell Colors**. Journal of the Optical Society of America. Vol. 33. No. 7. pp. 385-418. July 1943.

ASTM D 1535-08. Standard Practice for Specifying Color by the Munsell System. 2008

See Also

`MunsellToxyY()`, `xyYtoMunsell()`

Examples

```
VfromY( c(0,50,100) )
##  [1]  0.00000  7.53772 10.00000
```

---

xyYtoMunsell *Convert xyY coordinates to Munsell HVC*

Description

`xyYtoMunsell` Convert xyY coordinates to Munsell HVC, by interpolating over the tabulated Munsell renotation data.

Note that the input `xyY` must be relative to the white point of Illuminant C. Since Illuminant C is obsolete, it is almost certain that `xyY` in modern calculations *must* be chromatically adapted to the white point of Illuminant C *before* passing to this function. This chromatic adaptation can be done using `spacesXYZ::adaptxyY()`. Alternatively, one can use `XYZtoMunsell()`, which does the chromatic adaptation internally.

HVC $\rightarrow$ xyY is the forward direction, and xyY $\rightarrow$ HVC is the inverse direction. This inversion requires requires root-finding, which is done using `rootSolve::multiroot()`.

Usage

```
xyYtoMunsell( xyY, xyC='NBS', hcinterp='bicubic', vinterp='cubic',
                  VfromY='ASTM', rtol=1.e-8, atol=1.e-6, warn=TRUE, perf=FALSE )
```

Arguments

xyY	a numeric Nx3 matrix with CIE xyY coordinates in the rows, or a vector that can be converted to such a matrix, by row. These are for viewing in an environment with the white point of Illuminant C. Y must be scaled so that Y=100 for the perfect reflecting diffuser.
xyC	a numeric 2-vector with xy chromaticity of Illuminant C. It can also be one of the strings given in MunsellToxyY().
hcinterp	the type of 2D interpolation for H and C; either 'bicubic' or 'bilinear' (partial matching enabled). See MunsellToxyY() for details.
vinterp	the type of interpolation for V; either 'cubic' or 'linear' (partial matching enabled). See MunsellToxyY() for details.
VfromY	passed as the parameter which to the function VfromY(). See VfromY() for details. Option 'MGO' is not allowed because then Y>100 when V=10.
rtol	relative tolerance passed to rootSolve::multiroot()
atol	absolute tolerance passed to rootSolve::multiroot()
warn	if an xyY cannot be mapped (usually because the root finder has wandered afar because Chroma is too large), its Hue and Chroma are set to NA in the returned data.frame. Just before returning, if any rows in the returned HVC have an NA, and warn == TRUE, then a warning is logged.
perf	if perf is TRUE, then extra performance related metrics are appended to the returned data.frame, see **Value**.

Details

See MunsellToxyY() and the **User Guide - Appendix C**.

Value

a data.frame with N rows and these columns:

xyY	The input xyY
HVC	the computed HVC. H is automatically wrapped to (0,100]. In case of failure, H and C are set to NA. The rownames of HVC are set to those of xyY, unless they are NULL when they are set to SAMPLE_NAME.
SAMPLE_NAME	the Munsell notation for HVC, a character vector

If perf is TRUE then there are these additional columns:

time.elapsed	elapsed time in seconds. If available, the function microbenchmark::get_nanotime() is used.
iterations	the number of iterations of rootSolve::multiroot()
evaluations	the number of forward (HVC → xyY) function evaluations
estim.precis	the estimated precision from rootSolve::multiroot(). This is in the HC plane for the Munsell Value computed from Y.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` event, an error condition is signaled with the additional S3 class `"invalid_argument"` or `"package_unavailable"`, as appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

In the case that `warn=TRUE`, and not all HVC rows could be computed, a `WARN` event is created, and a warning condition is signaled with the additional S3 class `"incomplete"`. The condition object also has the submatrix of invalid `xyY`'s as item `invalid`, and their row indexes as item `indexes`.

Warning

Even when `vinterp='cubic'` the function xyY $\rightarrow$ HVC is not C^1 on the plane V=1. This is because of a change in Value spacing: when V$\geq$1 the Value spacing is 1, but when V$\leq$1 the Value spacing is 0.2.

Author(s)

Jose Gama and Glenn Davis

Source

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK
https://www.rit.edu/science/munsell-color-lab
https://www.rit-mcsl.org/MunsellRenotation/all.dat
https://www.rit-mcsl.org/MunsellRenotation/real.dat

References

Judd, Deane B. **The 1931 I.C.I. Standard Observer and Coordinate System for Colorimetry**. Journal of the Optical Society of America. Vol. 23. pp. 359-374. October 1933.

Newhall, Sidney M., Dorothy Nickerson, Deane B. Judd. **Final Report of the O.S.A. Subcommitte on the Spacing of the Munsell Colors**. Journal of the Optical Society of America. Vol. 33. No. 7. pp. 385-418. July 1943.

Kelly, Kenneth L. Kasson S. Gibson. Dorothy Nickerson. **Tristimulus Specification of the Munsell Book of Color from Spectrophometric Measurements** National Bureau of Standards RP1549 Volume 31. August 1943.

Judd, Deane B. and Günther Wyszecki. **Extension of the Munsell Renotation System to Very Dark Colors**. Journal of the Optical Society of America. Vol. 46. No. 4. pp. 281-284. April 1956.

Paul Centore 2014 The Munsell and Kubelka-Munk Toolbox https://www.munsellcolorscienceforpainters.com/MunsellAndK

See Also

`MunsellToxyY()`, `rootSolve::multiroot()`, `spacesXYZ::adaptxyY()`, `microbenchmark::get_nanotime()`

Examples

```
xyYtoMunsell(c(0.310897, 0.306510, 74.613450))
##       xyY.1     xyY.2     xyY.3     HVC.H     HVC.V     HVC.C    SAMPLE_NAME
## 1  0.310897  0.306510 74.613450 87.541720  8.900000  2.247428   7.5P 8.9/2.2


##  The chromaticity of D65 is xy = (0.3127,0.3290) to 4 digits.
##  If this is passed without chromatic adaptation, the result is not neutral.
munsellinterpol::xyYtoMunsell( c(0.3127,0.3290,18) )

##      xyY.x   xyY.y   xyY.Y      HVC.H      HVC.V      HVC.C    SAMPLE_NAME
##  1  0.3127  0.3290 18.0000 37.6679246  4.8519274  0.6461961 7.7GY 4.9/0.65

##  The Chroma is 0.65 instead of 0.
```

XYZtoMunsell *Convert XYZ coordinates to Munsell HVC*

Description

`XYZtoMunsell()` converts XYZ coordinates to Munsell HVC, by interpolating over the Munsell renotation data.

Note that *by default* the input `XYZ` is relative to the white point of Illuminant C. Since Illuminant C is obsolete, it is almost certain that an `XYZ` appearing in modern calculations *must* be chromatically adapted to the white point of Illuminant C before interpolating. This chromatic adaptation is done internally if the white point of XYZ is supplied as argument `white`. The adaptation is performed by `spacesXYZ::adaptXYZ()`.

If the input XYZ is computed from a reflectance spectrum, then ideally the white point should be set to the XYZ computed when that reflectance spectrum has constant value 1 - the so-called *Perfect Reflecting Diffuser*. The white points in published standards, e.g. D65 white = (95.047, 100, 108.883), can differ from spectrally-derived white points because of different wavelength step sizes and ranges. For an example, see the vignette **Soil Colors**.

Usage

```
XYZtoMunsell( XYZ, white=NULL, adapt='Bradford', ... )
```

Arguments

`XYZ`	a numeric Nx3 matrix with CIE XYZ coordinates in the rows, or a vector that can be converted to such a matrix, by row. Ideally, XYZ should be scaled so that Y=100 for the perfect reflecting diffuser.
`white`	XYZ for the source white point. If `NULL` then the input XYZ are for viewing in an environment with Illuminant C, and the input `XYZ` is not chromatically adapted. If `white` is not `NULL`, then the input `XYZ` is chromatically adapted from the given `white` to the white point of Illuminant C before conversion to HVC. `white` can be a numeric 3-vector; by convention `Y=100` for Munsell work. `white`

can also be a numeric 2-vector; which is interpreted as xy chromaticity and converted to XYZ with Y=100. `white` can also be the name of any standard illuminant recognized by `spacesXYZ::standardXYZ()` which is then scaled so that Y=100.
If the input XYZ is computed from a reflectance spectrum, then ideally `white` should be set to the XYZ computed when that reflectance spectrum has constant value 1 (see **Description**).

`adapt` method for chromatic adaptation, see `spacesXYZ::CAT()` for valid values. Also see **Details**. If `white=NULL` then `adapt` is ignored.

`...` other parameters passed to `xyYtoMunsell()`

Details

The conversion is done in these steps:

- If `white` is not `NULL`, XYZ is chromatically adapted from the given `white` to the white point of Illuminant C using `spacesXYZ::adaptXYZ()` and the given `adapt` method.
- XYZ → xyY using `spacesXYZ::xyYfromXYZ()`
- xyY → HVC using `xyYtoMunsell()` and .... This function does the interpolation over the Munsell renotation data.

Value

an Nx3 matrix with Munsell HVC in the rows. The rownames are copied from input to output, unless the input rownames are `NULL` when the ouput rownames are set to the Munsell notations for HVC.
In case of error, it returns `NULL`.

Condition Signaling

For an overview see: **Appendix E - Logging and Condition Signaling** in the User Guide.

In case of an `ERROR` or `FATAL` event, a condition is signaled with one of these additional S3 classes: `"invalid_argument"` or `"package_unavailable"`, whichever is appropriate. The invalid argument, or the name of the unavailable package, is added to the condition object.

Author(s)

Jose Gama and Glenn Davis

References

Paul Centore 2014. The Munsell and Kubelka-Munk Toolbox. https://www.munsellcolorscienceforpainters.com/MunsellAnd

See Also

`MunsellToXYZ()`, `xyYtoMunsell()`, `spacesXYZ::xyYfromXYZ()`, `spacesXYZ::adaptXYZ()`

Examples

```
## make XYZ for an 18% gray card, under Illuminant D65
white.D65 = 100 * as.numeric( spacesXYZ::standardXYZ( 'D65' ) )
gray18 = 0.18 * white.D65
XYZtoMunsell( gray18 )
##                       H        V         C
##  7.6GY 4.9/0.65 37.6456 4.851927 0.6466169

## This is not neutral gray, as expected.  The hue is green-yellow.
## Chroma = 0.6466 is too large; it should be 0, up to numerical truncation.
## The problem is that the white points are not equal.
## Redo and supply the true white point.

XYZtoMunsell( gray18, white=white.D65 )
##         H        V C
##  N 4.9/ 0 4.851927 0        #  much better; an exact neutral

##  Although the hue H=0 here, it is actually undefined because C=0
##  (as in polar coordinates, theta is undefined when r=0).


## When XYZ=(0,0,0) then Value should be exactly 0.
## And when XYZ is equal to the white point, then Value should be exactly 10.
XYZtoMunsell( c( 0,0,0, white.D65 ), white=white.D65 )
##        H  V C
##  N 0/  0  0 0
##  N 10/ 0 10 0


##  By convention, Y=100 for the reference white; but this is not strictly required.
gray18
##              X  Y        Z
##  [1,] 17.10846 18 19.59894

XYZtoMunsell( gray18, white=gray18 )
##        H  V C
##  N 10/ 0 10 0
##  The Value is still exactly 10.
```

Index

∗ **Colorlab**
  ColorlabFormatToMunsellSpec, 5
  MunsellSpecToColorlabFormat, 21
∗ **ISCC-NBS**
  ColorBlockFromMunsell, 3
∗ **MacAdam**
  IsWithinMacAdamLimits, 10
∗ **MunsellName**
  HVCfromMunsellName, 7
  MunsellNameFromHVC, 19
∗ **Munsell**
  MunsellToxyY, 29
∗ **RGB**
  MunsellToRGB, 26
  RGBtoMunsell, 40
∗ **XYZ**
  MunsellToXYZ, 32
  XYZtoMunsell, 50
∗ **datasets**
  CentroidsISCCNBS, 2
  LabToMunsell, 12
  Munsell2xy, 15
  MunsellBooks, 17
  NickersonColorDifference, 35
  VandY, 45
  xyYtoMunsell, 47
∗ **plot**
  plotLociHC, 36
  plotPatchesH, 38
∗ **sRGB**
  sRGBtoMunsell, 43

base::.onLoad, *29*, *44*

CentroidsISCCNBS, 2, *4, 5*
ColorBlockFromMunsell, 3, *42*
ColorlabFormatToMunsellSpec, 5, *23*

formatC, *19, 20*

HueNumberFromString, *20*, *37–39*
HueNumberFromString *(*HVCfromMunsellName*)*, 7
HueStringFromNumber, *10*
HueStringFromNumber *(*MunsellNameFromHVC*)*, 19
HVCfromMunsellName, 7, *20, 21*, *23*, *25, 26*, *28*, *33*, *35*, *41*

IsWithinMacAdamLimits, 10, *27–29*, *39*

LabToMunsell, 12, *24*
LabtoMunsell *(*LabToMunsell*)*, 12
LuvToMunsell, 14, *26*
LuvtoMunsell *(*LuvToMunsell*)*, 14

microbenchmark::get_nanotime(), *49*
Munsell2xy, 15
MunsellBooks, 17, *41, 42*
MunsellHVC *(*HVCfromMunsellName*)*, 7
MunsellNameFromHVC, *10*, 19
MunsellSpecToColorlabFormat, *6*, 21
MunsellToLab, *13*, 23
MunsellToLuv, 24
MunsellToRGB, 26, *29*, *39*
MunsellTosRGB, 28
MunsellToxyY, *26*, *28*, 29, *33, 34*, *38*, *47–49*
MunsellToXYZ, *23–27*, *29*, *32*, 32, *51*

NickersonColorDifference, 35, *42*

plotLociHC, 36
plotPatchesH, 38

RGBtoMunsell, 40, *44*
rootSolve::multiroot(), *47–49*
roundHVC, *18*, 41

spacesRGB::installRGB, *26, 27*, *38–40*
spacesRGB::RGBfromXYZ, *26–29*
spacesRGB::XYZfromRGB, *40, 41*
spacesXYZ::adaptxyY(), *29*, *32*, *47*, *49*

spacesXYZ::adaptXYZ, *23–26*, *28*, *29*, *40*, *41*, *44*
spacesXYZ::adaptXYZ(), *33*, *34*, *50*, *51*
spacesXYZ::CAT, *12–15*, *23–27*, *29*, *33*, *40*, *41*, *44*, *51*
spacesXYZ::LabfromXYZ, *23*, *24*
spacesXYZ::LuvfromXYZ, *25*, *26*
spacesXYZ::standardXYZ, *12–15*, *23–26*, *33*, *51*
spacesXYZ::xyYfromXYZ, *51*
spacesXYZ::XYZfromLab, *13*
spacesXYZ::XYZfromLuv, *14*, *15*
spacesXYZ::XYZfromRGB, *44*
spacesXYZ::XYZfromxyY, *26–29*, *33*, *34*
sRGBtoMunsell, 43

VandY, 45
VfromY, *48*
VfromY *(*VandY*)*, 45

xyYtoMunsell, *32*, *37*, *47*, 47, *51*
XYZtoMunsell, *12–15*, *40*, *41*, *43*, *44*, *47*, 50

YfromV, *15*, *16*, *30*
YfromV *(*VandY*)*, 45