Visual Fractions

R.W. Oldford

```
require("grid")
```

```
## Loading required package: grid
```

```
xy2grid <- function(x, y) {
  n <- length(x)
  m <- length(y)
  # Return the coordinates of the m x n grid havng
  # locations (x,y) for all x and y
  cbind(rep(x, times=m), rep(y, each=n))
}
#
# For example,
#
xy2grid(1:4, 10:12)
```

```
##       [,1] [,2]
##  [1,]    1   10
##  [2,]    2   10
##  [3,]    3   10
##  [4,]    4   10
##  [5,]    1   11
##  [6,]    2   11
##  [7,]    3   11
##  [8,]    4   11
##  [9,]    1   12
## [10,]    2   12
## [11,]    3   12
## [12,]    4   12
```

You now have all of the tools to write a function `visualFraction` which will draw a display of a visual fraction:

```
## The function will look like:
##
visualFraction <- function(num, # the numerator
                           den, # the denominator
                           numCol="red",
                           # numerator colour
                           denCol="white",
                           # denominator colour
                           random=FALSE,
                           # a logical indicating
                           # whether the numerator values
                           # are to appear at random
                           # locations (if TRUE) or not.
```

```
                          ncols = NULL
                          # number of columns to be
                          # used in the array
) {
  # begin with some error checking
  #
  #  Check the logical
  if (!is.logical(random))
    stop(paste("random must be TRUE or FALSE, not:",
               random))
  #
  #  Check the numerator
  if (!is.numeric(num))
    stop(paste("num must be a number, not", num))
  if (length(num) != 1)
    stop(paste("num must be a single number, not of length",
               length(num)))
  if (floor(num) != num | num < 0 )
    stop(paste("num must be a non-negative integer, not",
               num))
  #
  #  Check the denominator
  if (!is.numeric(den))
    stop(paste("den must be a number, not", den))
  if (length(den) != 1)
    stop(paste("den must be a single number, not of length",
               length(den)))
  if (floor(den) != den | den < 0 )
    stop(paste("den must be a non-negative integer, not",
               den))
  #
  #  Check both
  if (num > den)
    stop(paste("num =", num, "> den =", den))
  #
  #  Check ncols
  #
  #  Default is NULL, so if user doesn't supply one let's
  #  try to make it close to square (default more cols than rows)
  if (is.null(ncols)) ncols <- ceiling(sqrt(den))

  #  Now check any user supplied value for ncols
  if (!is.numeric(ncols))
    stop(paste("ncols must be a number, not", ncols))
  if (length(ncols) != 1)
    stop(paste("ncols must be a single number, not of length",
               length(ncols)))
  if (floor(ncols) != ncols | ncols < 0 )
    stop(paste("ncols must be a non-negative integer, not",
               ncols))
  if (ncols > den )
    stop(paste("ncols =", ncols,"> den =", den))
```

```
  ## If we have ncols columns, we will need
  ## nrows rows where
  nrows <- ceiling(den/ncols)

  ## We'll also need a radius
  ## This is size provides spacing for most
  radius <- 1/(2*(max(nrows,ncols)+5))

  ##
  ## Now it's your turn
  ## The display should be an nrows x ncols array of den circles
  ##
  ## If random=FALSE, the first num circles (from the top left of the
  ## array and proceeding left to right, then top to bottom)
  ## should be coloured numCol, the remainder coloured denCol.
  ##
  ## If random=TRUE, num circles selected at random in the array
  ## should be coloured numCol, the remainder denCol.
  ##
  ## That is, if we index the array 1 to den from top left by row to bottom
  ## right, the indices we would need to colour numCol would be
  if (random) {indices <- sample(1:den, num)} else {indices <- 1:num}
  ##
  ## INSERT YOUR CODE BEL
  coords = xy2grid(1:ncols, 1:nrows)
  coords = coords/(max(nrows, ncols)+1)
  for(i in 1:den){
    if (i %in% indices){
      grid.circle(x=coords[i,1],y=1-coords[i,2],r=radius,gp=gpar(fill=numCol))
    }
    else{
      grid.circle(x=coords[i,1],y=1-coords[i,2],r=radius,gp=gpar(fill=denCol))
    }
  }
}
grid.newpage()
visualFraction(13,100,random = TRUE)
```