

# Thompson\_Hendley\_hw5

Bradley Thompson & John Hendley

11/18/2020

```
## -- Attaching packages ----- tidyverse 1.3.0 --  
## v ggplot2 3.3.2      v purrr  0.3.4  
## v tibble  3.0.3      v dplyr  1.0.2  
## v tidyr   1.1.2      v stringr 1.4.0  
## v readr   1.4.0      v forcats 0.5.0  
  
## -- Conflicts ----- tidyverse_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag()     masks stats::lag()
```

## Number 1

```
# Takes the basic color palette and spreads it into 50 colors  
expand_palette <- function(n, colors) {  
  spread <- colorRampPalette(colors)  
  spread(n)  
}  
  
colors <- brewer.pal(9, "Spectral")  
  
## Error in brewer.pal(9, "Spectral"): could not find function "brewer.pal"  
more_colors <- expand_palette(50, colors)  
  
## Error in as.character(col): cannot coerce type 'closure' to vector of type 'character'  
show_col(more_colors, labels = FALSE)  
  
## Error in show_col(more_colors, labels = FALSE): could not find function "show_col"
```

## Number 2

```
mgs <- function(A){  
  #make skipping vector  
  skip_vector <- vector(mode = "integer", length = ncol(A))  
  for(i in 1:ncol(A)){  
    #normalize  
    if(!is.zero(A[, i])){  
      A[, i] <- normalize(A[, i])  
    }  
    #subtract projections  
    if(i != ncol(A)){
```

```

    for(j in (i + 1):ncol(A)){
      #check if in skip vector
      if(j %in% skip_vector){
        break()
      } else {
        A[, j] <- A[, j] - project_onto(A[, j], A[, i])
        if(is.zero(A[, j])){
          skip_vector[j] <- j
        }
      }
    }
  }
}
}
discard_zero_cols(A)
}

#testing
A <- matrix(c(1, 6, 19, 2, 1, 2, 7, 3, 5, 6, 23, 2), nrow = 3, byrow = TRUE)
gs(A)

```

```

##           [,1]      [,2]      [,3]
## [1,] 0.1924501  0.9678053 -0.1622214
## [2,] 0.1924501  0.1248781  0.9733285
## [3,] 0.9622504 -0.2185367 -0.1622214

```

```
mgs(A)
```

```

##           [,1]      [,2]      [,3]
## [1,] 0.1924501  0.9678053 -0.1622214
## [2,] 0.1924501  0.1248781  0.9733285
## [3,] 0.9622504 -0.2185367 -0.1622214

```

## Number 3

```

#array to tibble
array_to_tibble <- function(a){
  df <- a %>%
    dim() %>%
    map(~ 1:.x) %>%
    expand.grid() %>%
    as_tibble()
  names(df) <- paste0("i", 1:length(dim(a)))
  df$value <- as.vector(a)
  df
}

#testing
mat <- matrix(1:6, nrow = 2)
array_to_tibble(mat)

```

```

## # A tibble: 6 x 3
##       i1    i2 value
##   <int> <int> <int>

```

```
## 1      1      1      1
## 2      2      1      2
## 3      1      2      3
## 4      2      2      4
## 5      1      3      5
## 6      2      3      6
```

```
a <- array(1:24, dim = c(2, 4, 3))
array_to_tibble(a)
```

```
## # A tibble: 24 x 4
##       i1     i2     i3 value
##   <int> <int> <int> <int>
## 1     1     1     1     1
## 2     2     1     1     2
## 3     1     2     1     3
## 4     2     2     1     4
## 5     1     3     1     5
## 6     2     3     1     6
## 7     1     4     1     7
## 8     2     4     1     8
## 9     1     1     2     9
## 10    2     1     2    10
## # ... with 14 more rows
```

## Number 4

```
library(tibble)
```

```
# spy <- function(mat) {
#   this is brad talking, but maybe its like df <- array_to_tibble(mat)
#   then just do ur basecase but with the df and then for x and y it'd
#   be aes(df$i1, df$i2) and you'd fill the squares with the
#   values from df$i3 ? also i'm gonna need you to talk me through the
#   this function at somepoint lol cause idek how that works just from
#   looking at it. oh also yeah put the library for brewer.pal up in the
#   first code chunk with tidyverse and purrr cause rn for me ur
#   number 1 code throws an error
# }
```

```
# n <- 50
# mat <- matrix(0L, nrow = n, ncol = n)
# set.seed(2L)
# mat[sample(n^2, n)] <- rpois(n, 5)
```

```
n <- 10
```

```
df <- tibble(
  x = 1:10,
  y = 10:1
)
```

```
ggplot(df, aes(x, y)) +
```

```
geom_rect(aes(xmin = 0.5, xmax = n + 0.5,
              ymin = 0.5, ymax = n + 0.5),
          fill = "white", color = "black", size = 0.2) +
geom_tile(fill = "black") +
scale_x_continuous(name = NULL, breaks = NULL, minor_breaks = NULL) +
scale_y_continuous(name = NULL, breaks = NULL, minor_breaks = NULL) +
coord_equal()
```

