PSET 02 - NBA Games

S&DS 361

Due 2024-02-06

1. GitHub workflow

First, if you created a branch on the `bmacGTPM/361-Spring-2024` repo,

- Make a backup copy of any work you've done on that branch.
- Then delete your branch on the `bmacGTPM/361-Spring-2024` repo.

If you have a pull request, please close it. This might automatically be deleted if you delete your branch.

Second, if you created a fork previously, please make sure it is private (Settings, "Change repository visibility" all the way at the bottom). If not, you may have to backup your work, delete the repo and create another fork, especially if your repo doesn't say "forked from `bmacGTPM/361-Spring-2024`" anymore.

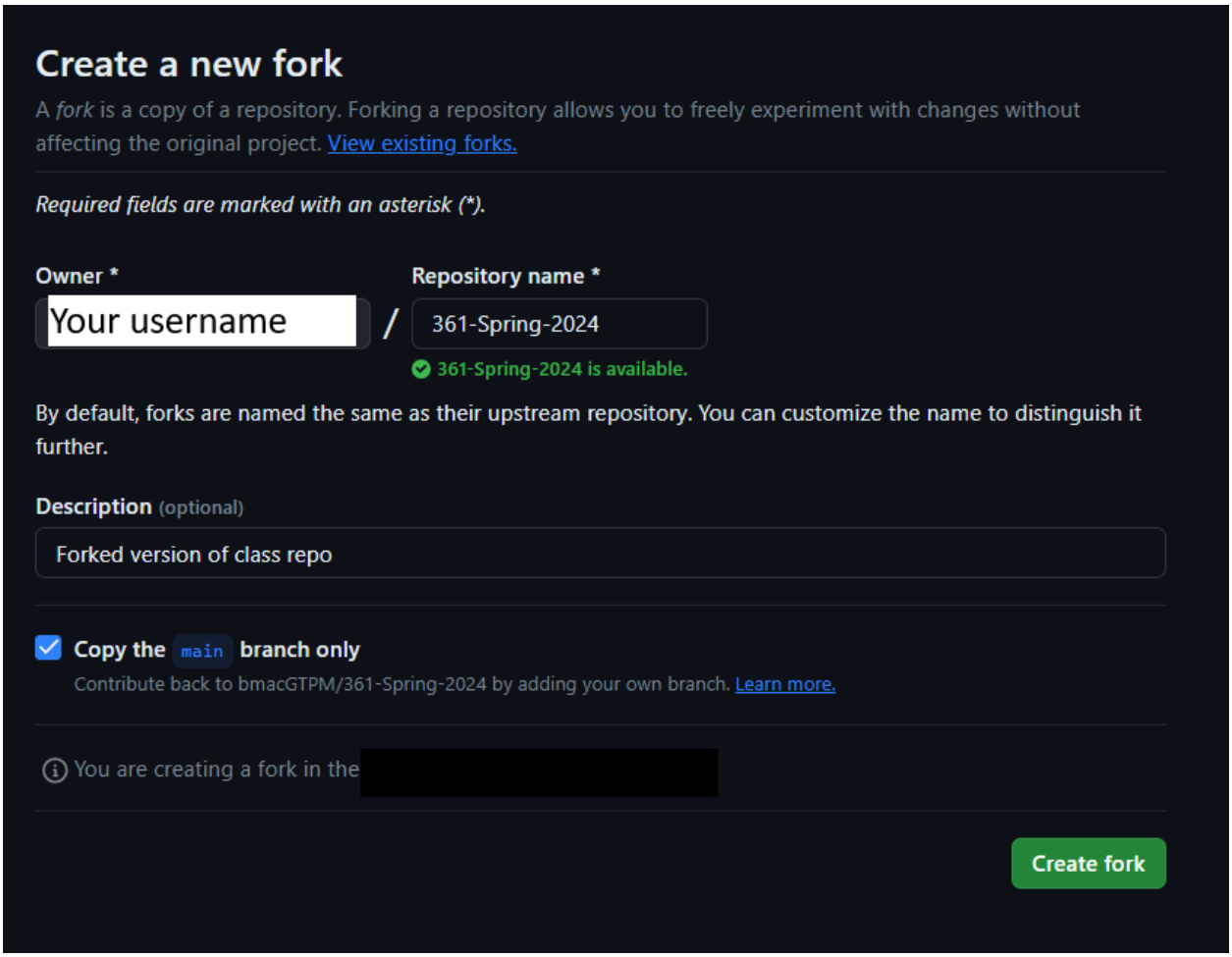
Going forward, there are a couple of options for using GitHub.

Option 1: If you don't want to fully use GitHub, you can manually download files as you would with Canvas.

Option 2: If you have cloned the repo directly (and didn't create a fork), you can continue to use that to fetch updates. Please do not create any branches, push changes, etc. there though. With this option, you can track changes to your work using Git, but you will not be able to back up your work by pushing to GitHub.

Option 3: If you want to use Git and GitHub to track changes as you work on your assignments, and back them up on GitHub, one option is to fork the repo so that you have a version of the repo on your own GitHub account.

- Go to https://github.com/bmacGTPM/361-Spring-2024,
- Click Fork (in the upper right), and
- Select Create New Fork.
- Choose your account as the destination for the fork
- Click Create Fork.



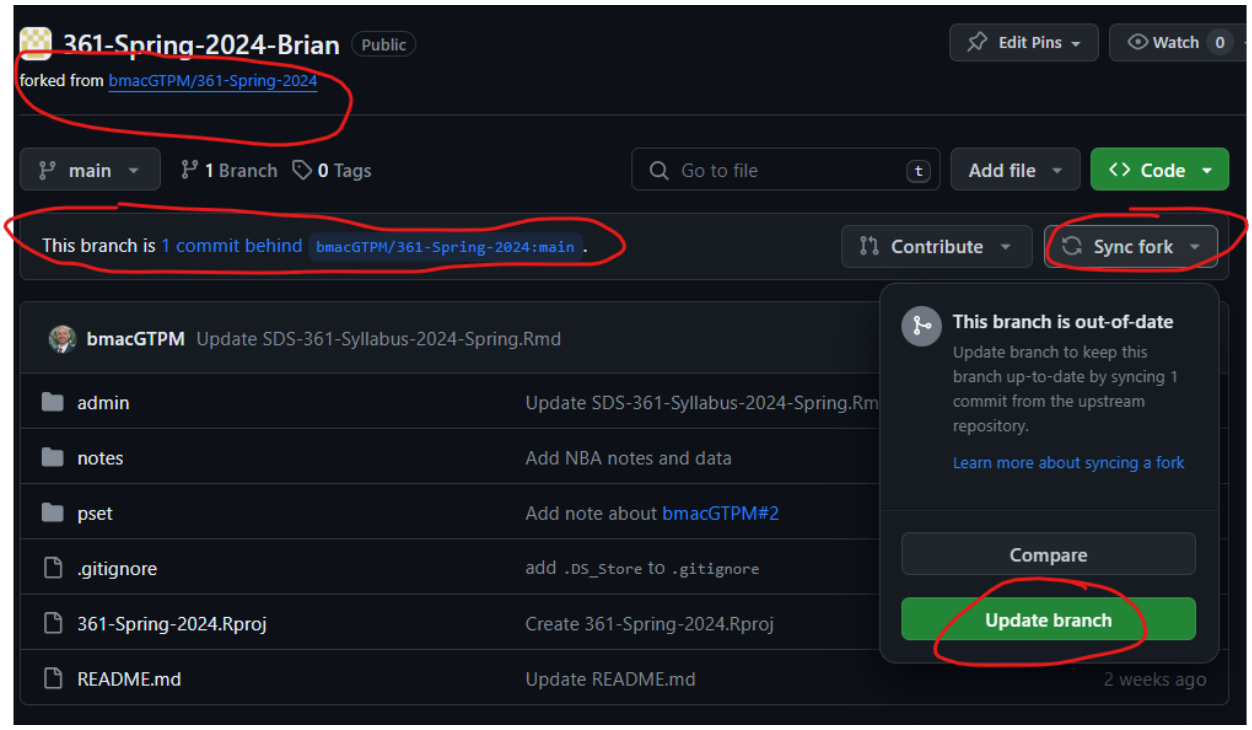
You should now have a copy of the repo on your GitHub account. This repo should automatically be private, since you are forking a private repo.

Now clone the repo you just created on your account to your computer.

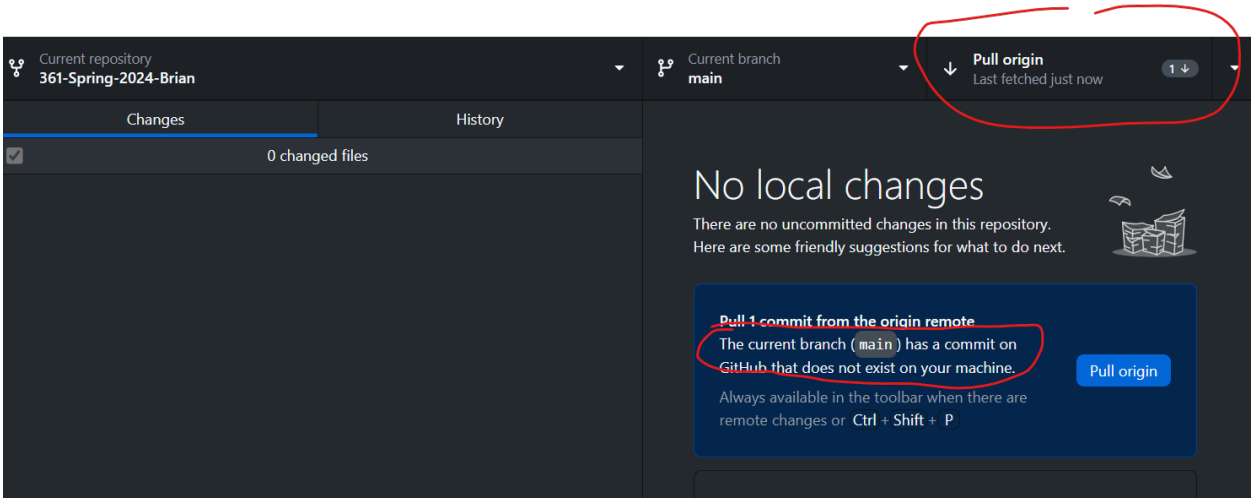
- Open GitHub Desktop
- Click File, New Repository
- Find the `yourGitHubUsername/361-Spring-2024` repo and select that
- Choose the local path where you want to save the repo on your computer
- Click Clone

You should now have a copy of the repo on your computer. You can open the repo in RStudio by clicking File, Open Project, and selecting the `.Rproj` file in the repo.

When changes are made on the course repo, you will see a message on your repo's main page, and you will have to update your fork.



That updates your fork on GitHub. To update your local copy of the repo, you will have to pull the changes from the remote repo.



Most times the changes should merge with your work automatically. If there are conflicts, you will get a message about them and you may have to resolve them manually.

Please state here which option you are using: 3.

Visualizing the NBA schedule

Let's visualize how often teams play each other in a season, to better understand the structure of the NBA schedule.

```
d = readRDS('data/games.rds')
d = d %>%
```

```
  filter(lg=='nba', season %in% 2022, season.type=='reg') %>%
  select(date, away, home, ascore, hscore, season, gid)
head(d)
```

```
##         date away home ascore hscore season      gid
## 1 2021-10-19  BKN  MIL    104    127   2022 22100001
## 2 2021-10-19  GSW  LAL    121    114   2022 22100002
## 3 2021-10-20  OKC  UTA     86    107   2022 22100011
## 4 2021-10-20  SAC  POR    124    121   2022 22100013
## 5 2021-10-20  DEN  PHX    110     98   2022 22100012
## 6 2021-10-20  ORL  SAS     97    123   2022 22100010
```

```
dg = d %>%
  group_by(away, home) %>%
  summarise(games = n()) %>%
  ungroup() %>%
  complete(away, home, fill=list(games=0)) ## new function!
```

```
## `summarise()` has grouped output by 'away'. You can override using the
## `.groups` argument.
```

```
head(dg)
```

```
## # A tibble: 6 x 3
##   away  home  games
##   <chr> <chr> <int>
## 1 ATL   ATL       0
## 2 ATL   BKN       1
## 3 ATL   BOS       2
## 4 ATL   CHA       2
## 5 ATL   CHI       2
## 6 ATL   CLE       2
```

2. Visualize the schedule with a grid plot Create a grid plot like the ones in these sections in the Appendix

- "Grid plot with `geom_tile`" (https://bmacgtpm.github.io/notes/grid-plot-with-geom_tile.html)
- "Customizing with `theme`" https://bmacgtpm.github.io/notes/customizing-with-theme.html,

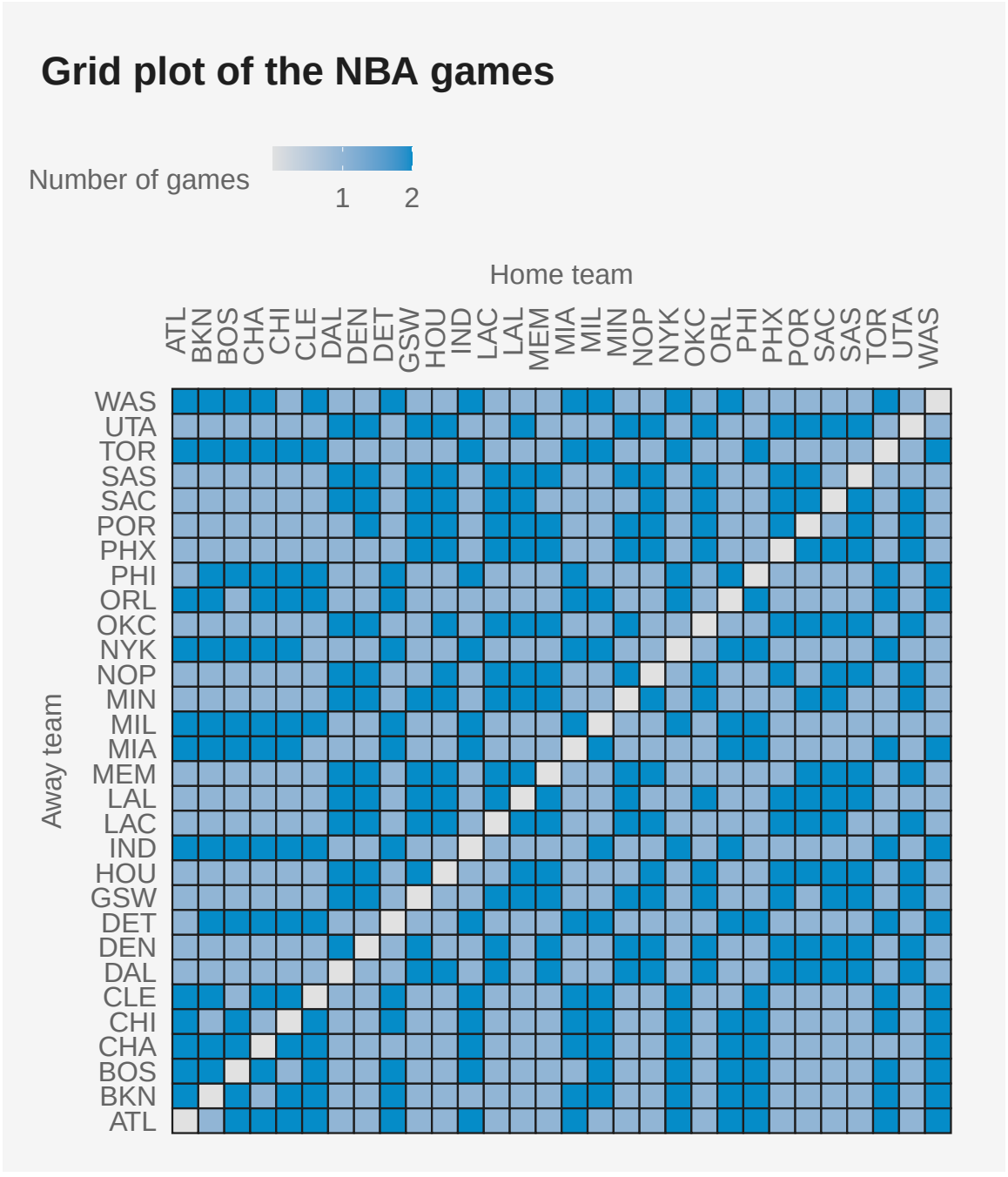
except use the `pubtheme` template from https://github.com/bmacGTPM/pubtheme to make it prettier. If you need to use `theme` to rotate the axis labels, you'll need to put that line of code *after* the line with `pub`. Also, since `pub` puts the x-axis labels at the top instead of the bottom, you'll need to use `axis.text.x.top` instead of `axis.text.x`. Since the grid is 30 by 30, you should change the dimensions of your plot if necessary to make sure the grid plot looks square.

```
library(tidyverse)
library(pubtheme)
## your code here
grid_plot = ggplot(dg,
           aes(x    = home,
               y    = away,
               fill = games)) +
  geom_tile(linewidth   = 0.4,
            show.legend = T,
            color       = pubdarkgray) +
  scale_fill_gradient(low      = pubgradgray,
                      high     = pubblue,
```

```
                      na.value = pubmediumgray, ## same color as below
                      oob      = squish,
                      breaks   = c(1, 2)) +

  labs(title    = 'Grid plot of the NBA games',
       fill = 'Number of games',
       x = 'Home team',
       y = 'Away team')


grid_plot %>% pub(type = 'grid') +    theme(axis.text.x = element_text(angle = 90, hjust=1),
        axis.ticks.x=element_blank(), axis.ticks.y=element_blank(),
        panel.background = element_blank())
```



3. Order the teams by division By default, R organizes the teams alphabetically. But there is often additional structure in the data that is lost if we let R sort our variables alphabetically.

There are two conferences, the Eastern Conference and Western Conference, with 15 teams each. As the names suggest, the Eastern Conference teams are in the eastern half of the United States (and one in Canada), and the Western Conference teams are in the western/midwest regions of the US.

Each conference has 3 divisions with 5 teams each. The teams within the same division tend to be geographically close to one another. You can read a little more about the divisions and conferences here if you'd like: https://www.lines.com/guides/what-are-the-6-nba-divisions/1572.

NBA teams play each other more often if they are in the same division or conference. That is important

structure to visualize, so let's organize teams by division. Use `factor` and specify the `levels` to be the order we want for both teams and divisions. We want, for example, the Eastern conference teams to be first, (Atlantic Division, then Central Division, and so on), followed by the Western Conference teams. Then make a grid plot with the new ordering.

Some data prep is done for you below.

```
tms = read.csv('data/nba.teams.csv')

## capitalize the first letter of each conf and div
## create a factor to specify the order we want for divisions
tms = tms %>%
  arrange(conf, div) %>%
  mutate(conf = paste0(toupper(substr(conf, 1, 1)),
                               substr(conf, 2, nchar(conf))),

         div  = paste0(toupper(substr(div , 1, 1)),
                               substr(div , 2, nchar( div))),

         div  = factor(div,
                       levels = unique(div)))
head(tms)
```

```
##   team      div conf
## 1  BKN Atlantic East
## 2  BOS Atlantic East
## 3  NYK Atlantic East
## 4  PHI Atlantic East
## 5  TOR Atlantic East
## 6  CHI  Central East
```

```
## your code here

df = dg %>%
    mutate(away = factor(away, levels = unique(tms$team)),
    home = factor(home, levels = unique(tms$team)),
    away_div = tms$div[match(away, tms$team)],
    home_div = tms$div[match(home, tms$team)])
head(df)
```

```
## # A tibble: 6 x 5
##   away  home  games away_div  home_div
##   <fct> <fct> <int> <fct>     <fct>
## 1 ATL   ATL       0 Southeast Southeast
## 2 ATL   BKN       1 Southeast Atlantic
## 3 ATL   BOS       2 Southeast Atlantic
## 4 ATL   CHA       2 Southeast Southeast
## 5 ATL   CHI       2 Southeast Central
## 6 ATL   CLE       2 Southeast Central
```

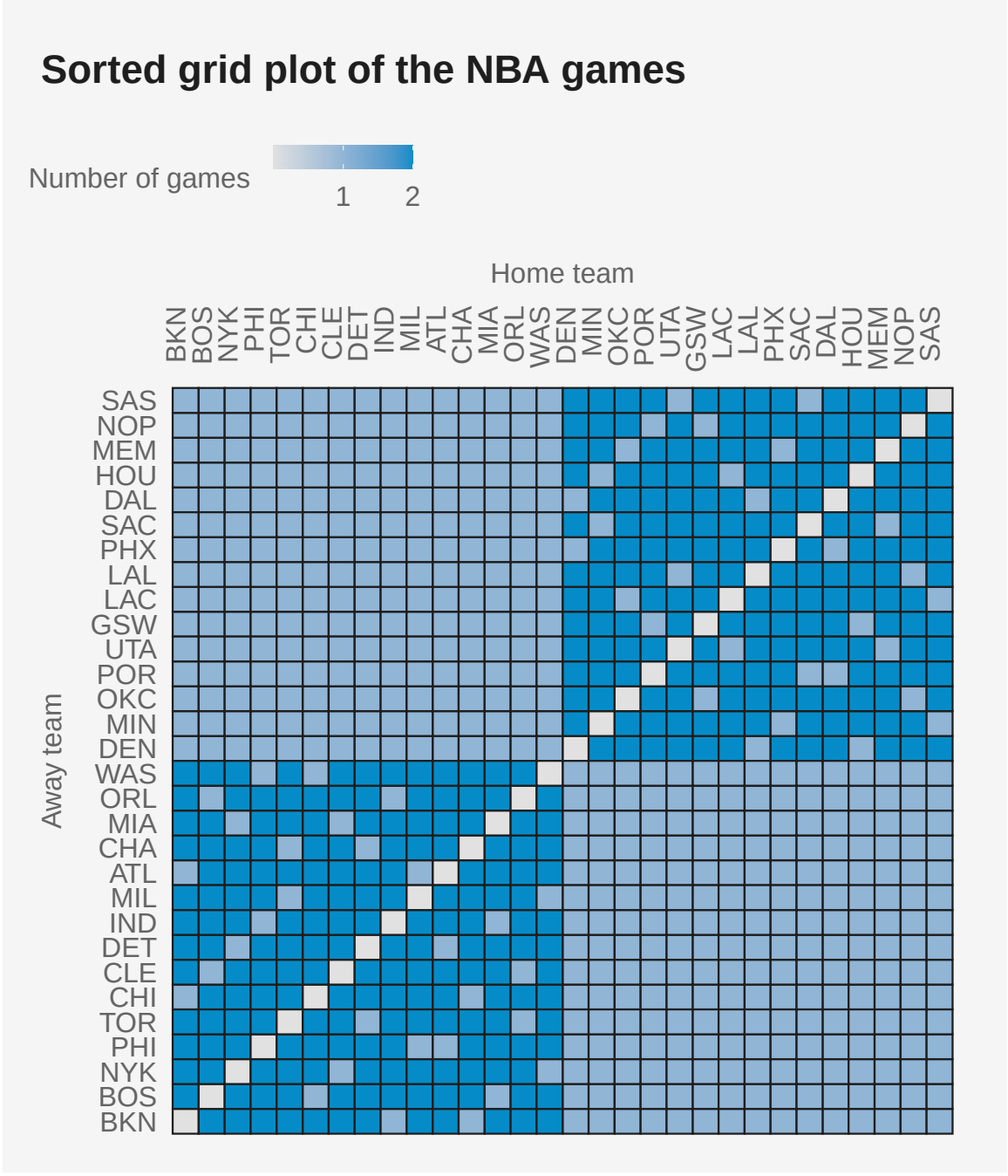
```
## your ggplot code here

## the same/similar ggplot code from previous question might work here too,
## depending what you did there
grid_plot2 = ggplot(df,
           aes(x    = home,
               y    = away,
```

```
              fill = games)) +
  geom_tile(linewidth   = 0.4,
            show.legend = T,
            color       = pubdarkgray) +
  scale_fill_gradient(low      = pubgradgray,
                      high     = pubblue,
                      na.value = pubmediumgray, ## same color as below
                      oob      = squish,
                      breaks   = c(1, 2)) +

  labs(title    = 'Sorted grid plot of the NBA games',
       fill = 'Number of games',
       x = 'Home team',
       y = 'Away team')


grid_plot2 %>% pub(type = 'grid') +    theme(axis.text.x = element_text(angle = 90, hjust=1),
        axis.ticks.x=element_blank(), axis.ticks.y=element_blank(),
        panel.background = element_blank())
```

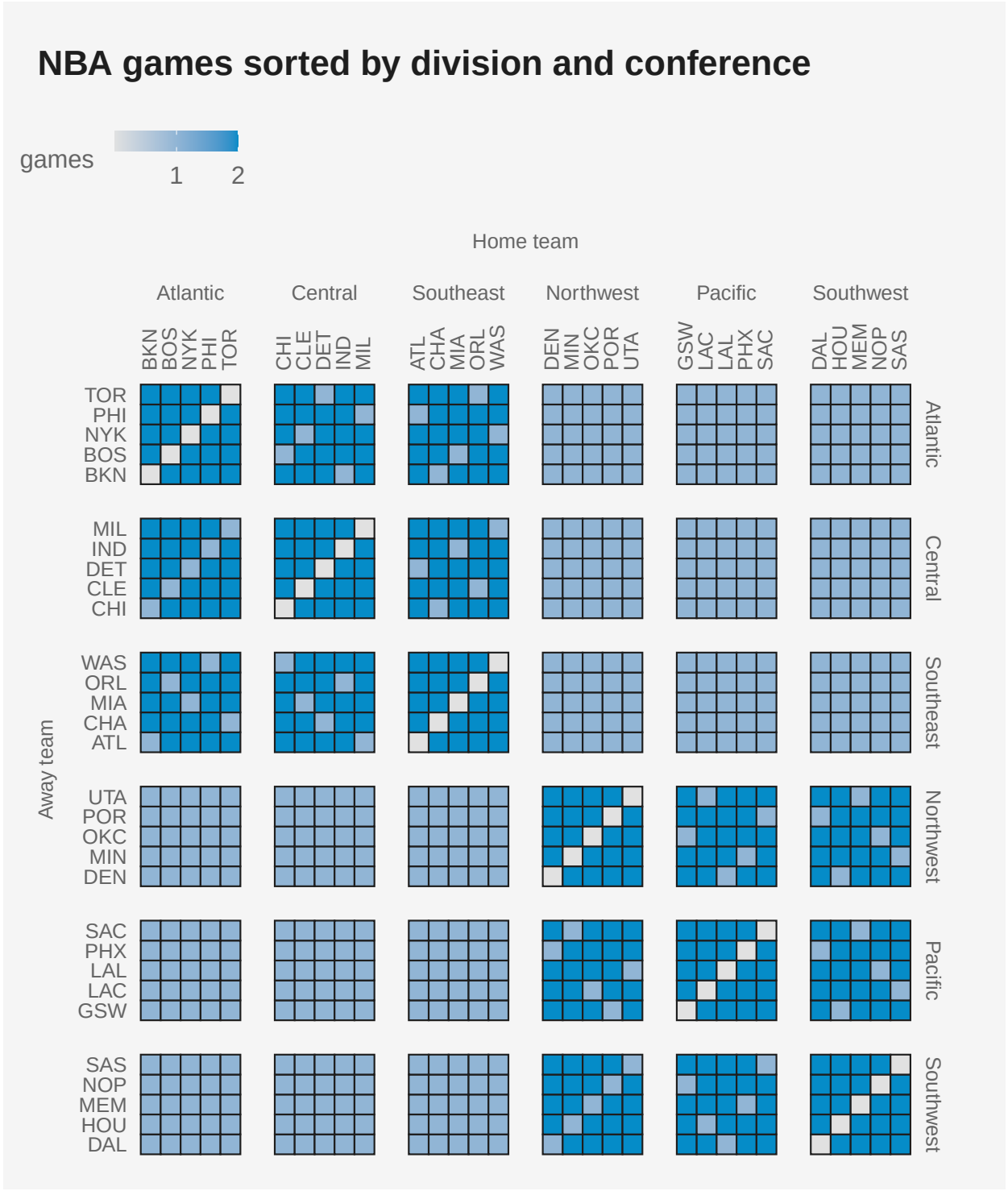


4. Creating separation between the divisions That looks a lot more orderly! We are starting to see the structure of the data. Now let's do one more thing and create some separation between the divisions. Use `facet_grid` to facet by division. Some data prep is done for you below. Note that `facet_grid` is similar to `facet_wrap` but requires you to specify two columns. See also the Arguments described in the help file. You'll need to use at least one of the arguments to make this look nice.

```
## your code here
grid_plot3 <- ggplot(df,
            aes(x = home, y = away, fill = games)) +
      geom_tile(linewidth   = 0.4,
            show.legend = T,
```

```
            color       = pubdarkgray) +
    scale_fill_gradient(low      = pubgradgray,
                        high     = pubblue,
                        na.value = pubmediumgray, ## same color as below
                        oob      = squish,
                        breaks   = c(1, 2)) +
    labs(title = 'NBA games sorted by division and conference',
    x = 'Home team', y = 'Away team',
    fill = 'games') + ## titles
    facet_grid(vars(away_div), vars(home_div), scales = "free")


grid_plot3 %>% pub(type = 'grid') +
    theme(axis.text.x.top = element_text(angle = 90, vjust = 0.4, hjust = 0),
          strip.text = element_text(size=10),
          text = element_text(size=10),
          aspect.ratio = 1)
```

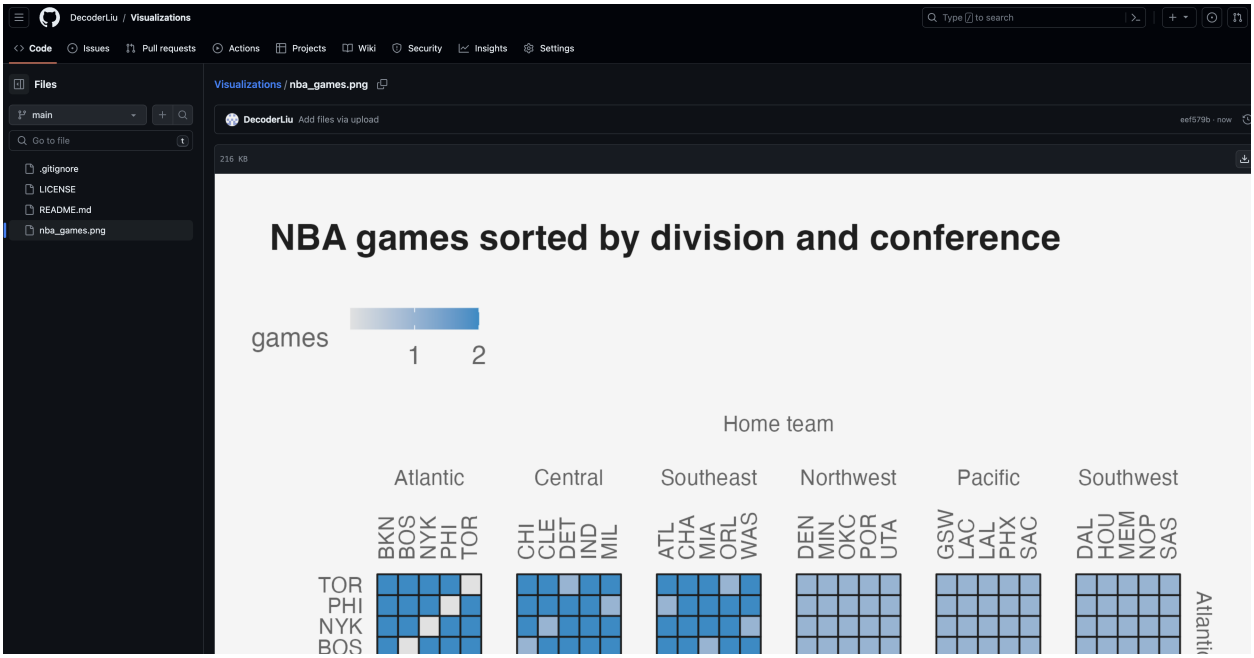



```
ggsave('img/nba_games.png', width = 7, height = 8)
```

5. Create a new repo

Create a new repo (on GitHub.com go to Repositories, New) on your GitHub account called `Visualizations`. Make it public. Initialize it with a README. Add a `.gitignore` file and use the `R` template. Choose any licence or none. Clone the repo to your computer. Save the visualization you created in #4 to a file in this

repo on your computer. Commit the change to the main branch. Push the commit to GitHub. Take a screenshot of the repo on GitHub showing that it has your visualization file in it. Paste the screenshot below.



GitHub is a good place to showcase your work to potential employers, graduate school programs, and peers. Similar to an artist who has a portfolio, you can have a data science portfolio. You can continue to add visualizations to this repo, or create another repo to publicly showcase data science projects and other work that you do.