Here we have it, the last meet of the State-Off semi-final round, where California (1) takes on the Cinderella story that is Pennsylvania (5) for a trip to the finals. Don’t forget to update your version of SwimmeR to 0.4.1, because we’ll be using some newly released functions. We’ll also do some more plotting/data-vis and use prop.test to do some statistical work with the results, this time looking at grade distributions across events.

library(SwimmeR) library(dplyr) library(purrr) library(tidyr) library(stringr) library(flextable) library(ggplot2)

My flextable styling function is still working great .Here it is again.

flextable\_style <- function(x) { x %>%

flextable() %>%

bold(part = "header") %>% # bolds header

bg(bg = "#D3D3D3", part = "header") %>% # puts gray background behind the header row

autofit()

}

# Getting Results

California\_Link <- "https://raw.githubusercontent.com/gpilgrim2670/Pilgrim\_Data/master/CA\_States\_2019.csv"

California\_Results <- read.csv(url(California\_Link)) %>% mutate(State = "CA")

Pennsylvania\_Link <- "https://raw.githubusercontent.com/gpilgrim2670/Pilgrim\_Data/master/PA\_States\_2020.csv"

Pennsylvania\_Results <- read.csv(url(Pennsylvania\_Link)) %>% mutate(State = "PA")

Results <- California\_Results %>% bind\_rows(Pennsylvania\_Results) %>% mutate(Gender = case\_when(

str\_detect(Event, "Girls") == TRUE ~ "Girls", str\_detect(Event, "Boys") == TRUE ~ "Boys"

)) %>%

filter(str\_detect(Event, "Swim-off") == FALSE)

# Scoring the Meet

Results\_Final <- results\_score( results = Results,

events = unique(Results$Event), meet\_type = "timed\_finals", lanes = 8,

scoring\_heats = 2,

point\_values = c(20, 17, 16, 15, 14, 13, 12, 11, 9, 7, 6, 5, 4, 3, 2, 1)

)

Scores <- Results\_Final %>% group\_by(State, Gender) %>% summarise(Score = sum(Points))

Scores %>%

arrange(Gender, desc(Score)) %>% ungroup() %>%

flextable\_style()

|  |  |  |
| --- | --- | --- |
| **State** | **Gender** | **Score** |
| CA | Boys | 1641 |
| PA | Boys | 684 |
| CA | Girls | 1629 |
| PA | Girls | 696 |

Scores %>% group\_by(State) %>%

summarise(Score = sum(Score)) %>% arrange(desc(Score)) %>% ungroup() %>%

flextable\_style()

**State Score**

CA 3270

PA 1380

Pennsylvania’s charmed run through (one round of) the State-Off Tournament has come to an end, with California dominating both the boys and girls meets, and winning the overall handily.

# Swimmers of the Meet

Swimmer of the Meet criteria is the same as it’s been for the entire State-Off. First we’ll look for athletes who have won two events, thereby scoring a the maximum possible forty points. In the event of a tie, where multiple athletes win two events, we’ll use All-American standards as a tiebreaker. Will anyone join Lillie Nordmann as a multiple Swimmer of the Meet winner?

Cuts\_Link <- "https://raw.githubusercontent.com/gpilgrim2670/Pilgrim\_Data/master/State\_Cuts.csv"

Cuts <- read.csv(url(Cuts\_Link))

Cuts <- Cuts %>% # clean up Cuts

filter(Stroke %!in% c("MR", "FR", "11 Dives")) %>% # %!in% is now included in SwimmeR

rename(Gender = Sex) %>% mutate(

Event = case\_when((Distance == 200 & #match events

Stroke == 'Free') ~ "200 Yard Freestyle", (Distance == 200 &

Stroke == 'IM') ~ "200 Yard IM", (Distance == 50 &

Stroke == 'Free') ~ "50 Yard Freestyle",

(Distance == 100 &

Stroke == 'Fly') ~ "100 Yard Butterfly", (Distance == 100 &

Stroke == 'Free') ~ "100 Yard Freestyle", (Distance == 500 &

Stroke == 'Free') ~ "500 Yard Freestyle", (Distance == 100 &

Stroke == 'Back') ~ "100 Yard Backstroke", (Distance == 100 &

Stroke == 'Breast') ~ "100 Yard Breaststroke", TRUE ~ paste(Distance, "Yard", Stroke, sep = " ")

),

Event = case\_when(

Gender == "M" ~ paste("Boys", Event, sep = " "), Gender == "F" ~ paste("Girls", Event, sep = " ")

)

)

Ind\_Swimming\_Results <- Results\_Final %>% filter(str\_detect(Event, "Diving|Relay") == FALSE) %>% # join

Ind\_Swimming\_Results and Cuts left\_join(Cuts %>% filter((Gender == "M" &

Year == 2020) | (Gender == "F" &

Year == 2019)) %>%

select(AAC\_Cut, AA\_Cut, Event), by = 'Event')

Swimmer\_Of\_Meet <- Ind\_Swimming\_Results %>% mutate(

AA\_Diff = (Finals\_Time\_sec - sec\_format(AA\_Cut)) / sec\_format(AA\_Cut), Name = str\_to\_title(Name)

) %>%

group\_by(Name) %>%

filter(n() == 2) %>% # get swimmers that competed in two events summarise(

Avg\_Place = sum(Place) / 2,

AA\_Diff\_Avg = round(mean(AA\_Diff, na.rm = TRUE), 3), Gender = unique(Gender),

State = unique(State)

) %>%

arrange(Avg\_Place, AA\_Diff\_Avg) %>%

group\_split(Gender) # split out a dataframe for boys (1) and girls (2)

## Boys

Swimmer\_Of\_Meet[[1]] %>% slice\_head(n = 5) %>% select(-Gender) %>% ungroup() %>% flextable\_style()

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Avg\_Place** | **AA\_Diff\_Avg** | **State** |
| Brownstead, Matt | 1.0 | -0.050 | PA |
| Mefford, Colby | 1.5 | -0.027 | CA |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Avg\_Place** | **AA\_Diff\_Avg** | **State** |
| Hu, Ethan | 2.0 | -0.052 | CA |
| Jensen, Matthew | 2.0 | -0.038 | PA |
| Faikish, Sean | 2.0 | -0.032 | PA |

Turns out yes, Matt Brownstead from Pennsylvania joins Lillie in the multiple winners club. Matt broke the national high school record in the 50 free, so that guaranteed him one win. He also won the 100 free – the only boy here to win two events. We don’t even need to go to the All-American tie breaker, but it is worth noting that Ethan Hu outperformed Matt by that metric. Pennsylvania also managed three of the top five finishers here – very nice!

Results\_Final %>%

filter(Name == "Brownstead, Matt") %>% select(Place, Name, School, Finals\_Time, Event) %>% arrange(desc(Event)) %>%

ungroup() %>% flextable\_style()

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Place** | **Name** | **School** | **Finals\_Time** | **Event** |  |
| 1 | Brownstead, Matt | State College-06 | 19.24 | Boys 50 Yard Freestyle |  |
| 1 | Brownstead, Matt | State College-06 | 43.29 | Boys 100 Yard Freestyle |  |
|  |  |  |  |  |  |

## Girls

Swimmer\_Of\_Meet[[2]] %>% slice\_head(n = 5) %>% select(-Gender) %>% ungroup() %>% flextable() %>% bold(part = "header") %>%

bg(bg = "#D3D3D3", part = "header") %>% autofit()

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Avg\_Place** | **AA\_Diff\_Avg** | **State** |
| Hartman, Zoie | 1.0 | -0.047 | CA |
| Ristic, Ella | 1.0 | -0.023 | CA |
| Tuggle, Claire | 1.5 | -0.031 | CA |
| Delgado, Anicka | 1.5 | -0.023 | CA |
| Kosturos, Sophi | 2.0 | -0.021 | CA |

Zoie Hartman heads a California sweep of the girls swimmer of the meet top 5 while also winning her second swimmer of the meet crown.

Results\_Final %>%

filter(Name == "Hartman, Zoie") %>%

select(Place, Name, School, Finals\_Time, Event) %>% arrange(desc(Event)) %>%

ungroup() %>% flextable\_style()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Place** | **Name** | **School** | **Finals\_Time** | **Event** |
| 1 | Hartman, Zoie | Monte Vista\_NCS | 1:55.29 | Girls 200 Yard IM |
| 1 | Hartman, Zoie | Monte Vista\_NCS | 59.92 | Girls 100 Yard Breaststroke |

## Performances By Grade

It might be interesting to see what fraction of athletes from each grade compete in various events. We might hypothesize that sprint events, like the 50 freestyle in particular, would have a higher percentage of older athletes, who benefit from extra years of growth, muscle development etc. SwimmeR can capture grade (or age) values, assuming they’re present in the original results. In this case we do have grade values, so let’s take a look.

Results\_Grade <- Results\_Final %>% filter(is.na(Grade) == FALSE,

Grade %!in% c("ST", "1")) %>% # remove nonsense values mutate(

Grade = case\_when(

# regularize encoding of grade values Grade == "9" ~ "FR",

Grade == "10" ~ "SO", Grade == "11" ~ "JR", Grade == "12" ~ "SR",

TRUE ~ Grade

)

) %>%

mutate(Grade = factor(Grade, levels = c("FR", "SO", "JR", "SR"))) %>% # factor to order grade levels

filter(str\_detect(Event, "Relay") == FALSE) %>% # remove relays since they don't have grades

mutate(Event = str\_remove(Event, "Girls |Boys "), Event = factor(Event, levels = rev(unique(

# order events by meet order, in this case with diving first str\_remove(unique(Results$Event), "Girls |Boys ")

))))

Results\_Grade\_Sum <- Results\_Grade %>% group\_by(Event, Gender, Grade) %>%

summarise(Numb = n()) %>% # number of athletes for each event/gender/grade combintation

ungroup() %>% group\_by(Event, Gender) %>%

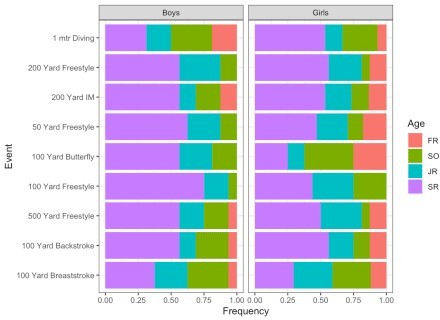
mutate(Percentage = Numb / sum(Numb)) # percentage of athletes in each grade for each event/gender combination

Results\_Grade\_Sum %>% ggplot() +

geom\_col(aes(x = Event, y = Percentage, fill = Grade)) + coord\_flip() +

facet\_wrap(. ~ Gender) + theme\_bw() +

labs(y = "Frequency")



That’s a lovely plot (if I do say so myself). Interestingly, the 50 freestyle doesn’t appear to be the most senior heavy event for boys or girls. It would be nice though to have the data in a table form to aid in taking a closer look, so let’s work on that.

## Tables

# Split results by gender, creating a list of two dataframes Results\_Grade\_Gender <- Results\_Grade\_Sum %>%

ungroup() %>% group\_by(Gender) %>% group\_split()

# function to apply to both dataframes that will produce columns with percentages for each grade and event

Grade\_Fill <- function(x) { x <- x %>%

mutate(Percentage = round(Percentage, 2)) %>% pivot\_wider(names\_from = Grade, values\_from = Percentage) %>% select(-Numb) %>%

group\_by(Event) %>%

fill(everything(), .direction = "updown") %>% unique() %>%

mutate(Event = factor(Event, levels = unique(str\_remove(unique( Results$Event), "Girls |Boys ")))) %>%

mutate\_if(is.numeric, ~replace(., is.na(.), 0))

return(x)

}

# map Grade\_Fill function over list of dataframes Results\_Grade\_Gender <- Results\_Grade\_Gender %>%

map(Grade\_Fill)

# print boys table Results\_Grade\_Gender[[1]] %>%

arrange(Event) %>% flextable\_style()



**Event**

**Gender**

**FR SO**

**JR SR**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event** | **Gender** | **FR** | **SO** | **JR** | **SR** |
| 1 mtr Diving | Boys | 0.19 | 0.31 | 0.19 | 0.31 |
| 200 Yard Freestyle | Boys | 0.00 | 0.12 | 0.31 | 0.56 |
| 200 Yard IM | Boys | 0.12 | 0.19 | 0.12 | 0.56 |
| 50 Yard Freestyle | Boys | 0.00 | 0.12 | 0.25 | 0.62 |
| 100 Yard Butterfly | Boys | 0.00 | 0.19 | 0.25 | 0.56 |
| 100 Yard Freestyle | Boys | 0.00 | 0.06 | 0.19 | 0.75 |
| 500 Yard Freestyle | Boys | 0.06 | 0.19 | 0.19 | 0.56 |
| 100 Yard Backstroke | Boys | 0.06 | 0.25 | 0.12 | 0.56 |
| 100 Yard Breaststroke | Boys | 0.06 | 0.31 | 0.25 | 0.38 |

# print girls table Results\_Grade\_Gender[[2]] %>%

arrange(Event) %>% flextable\_style()

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event** | **Gender** | **FR** | **SO** | **JR** | **SR** |
| 1 mtr Diving | Girls | 0.07 | 0.27 | 0.13 | 0.53 |
| 200 Yard Freestyle | Girls | 0.12 | 0.06 | 0.25 | 0.56 |
| 200 Yard IM | Girls | 0.13 | 0.13 | 0.20 | 0.53 |
| 50 Yard Freestyle | Girls | 0.18 | 0.12 | 0.24 | 0.47 |
| 100 Yard Butterfly | Girls | 0.25 | 0.38 | 0.12 | 0.25 |
| 100 Yard Freestyle | Girls | 0.00 | 0.25 | 0.31 | 0.44 |
| 500 Yard Freestyle | Girls | 0.12 | 0.06 | 0.31 | 0.50 |
| 100 Yard Backstroke | Girls | 0.12 | 0.12 | 0.19 | 0.56 |
| 100 Yard Breaststroke | Girls | 0.12 | 0.29 | 0.29 | 0.29 |

There’s been a rumor going around that “seniors rule” and I gotta say, these results are pushing me towards believing it. On the boys side seniors have the highest percentage representation in every event (although admittedly tied with sophomores (?) in diving). On the girls side seniors again dominated, although sophomores had the most representatives in the 100 butterfly and the 100 breaststroke was a three way tie between sophomores (again with the sophomores…), juniors and seniors.

Let’s check this out a bit further though. We can use prop.test to check whether or not a given population proportion matches what we expect. For starters let’s define “ruling” as being over represented in an event or in the meet. More formally, we can set up a null hypothesis, where the null value for population proportion (the percentage of athletes in a given event who are seniors) is 0.25. We can then accept or reject that null hypothesis based on a significance level. We’ll choose the standard 0.5 as a significance level. Running prop.test will give us (among other things) a p value, which we can compare to our significance level. If the p value is less than our significance level of 0.5 we can reject the null hypothesis and conclude that their are more seniors than we would expect. If there are more seniors than than an underlying 1/4 probability would suggest we can confirm an instance of seniors “ruling”.

Prop\_Test\_Results <- Results\_Grade\_Sum %>% group\_by(Event, Gender) %>% mutate(Total\_Athletes = sum(Numb)) %>% filter(Grade == "SR") %>% # only testing seniors rowwise() %>%

mutate(P\_Val = prop.test(Numb, Total\_Athletes, p = 0.25)$p.value[1]) # run prop test and extract p values

Prop\_Test\_Results\_Gender <- Prop\_Test\_Results %>%

ungroup() %>% group\_by(Gender) %>% group\_split()

We can look at each event for boys and girls, and highlight (in red) p values that are less than our significance value of 0.5, meaning that for those events. We’ll collect a list of rows meeting each criteria (greater or less than our significance value) and then use flextable::bg to provide the appropriate background fill color.

Please note, the number of athletes in an event can be more than 16 in the event of a tie, or less than 16 for the purposes of this analysis if an athlete didn’t have their grade specified.

## Boys Proportion Test

row\_id\_accept\_boys <- # values where P\_Val is greater or equal to than the significance value, should be green, fail to reject null hypothesis

with(Prop\_Test\_Results\_Gender[[1]], round(P\_Val, 2) >= 0.5) row\_id\_reject\_boys <- # values where P\_Val is less than the significance value, should be red, reject null hypothesis

with(Prop\_Test\_Results\_Gender[[1]], round(P\_Val, 2) < 0.5) col\_id <- c("P\_Val") # which column to change background color in

Prop\_Test\_Results\_Gender[[1]] %>% arrange(rev(Event)) %>% flextable\_style() %>%

bg(i = row\_id\_accept\_boys, j = col\_id,

bg = "green", part = "body") %>%

bg(i = row\_id\_reject\_boys, j = col\_id,

bg = "red",

part = "body") %>% colformat\_num(j = "P\_Val",

big.mark = ",", digits = 3) %>%

autofit()

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Event** | **Gender** | **Grade** | **Numb** | **Percentage** | **Total\_Athletes** | **P\_Val** |  |
| 1 mtr Diving | Boys | SR | 5 | 0.3125 | 16 | 0.773 |  |
| 200 Yard Freestyle | Boys | SR | 9 | 0.5625 | 16 | 0.009 |  |
| 200 Yard IM | Boys | SR | 9 | 0.5625 | 16 | 0.009 |  |
| 50 Yard Freestyle | Boys | SR | 10 | 0.6250 | 16 | 0.001 |  |
| 100 Yard Butterfly | Boys | SR | 9 | 0.5625 | 16 | 0.009 |  |
| 100 Yard Freestyle | Boys | SR | 12 | 0.7500 | 16 | 0.000 |  |
| 500 Yard Freestyle | Boys | SR | 9 | 0.5625 | 16 | 0.009 |  |
| 100 Yard Backstroke | Boys | SR | 9 | 0.5625 | 16 | 0.009 |  |
| 100 Yard Breaststroke | Boys | SR | 6 | 0.3750 | 16 | 0.386 |  |
|  |  |  |  |  |  |  |  |

## Girls Proportion Test

row\_id\_accept\_girls <- # values where P\_Val is greater or equal to than the significance value, should be green, fail to reject null

with(Prop\_Test\_Results\_Gender[[2]], round(P\_Val, 2) >= 0.5)

row\_id\_reject\_girls <- # values where P\_Val is less than the significance value, should be red, reject null hypothesis

with(Prop\_Test\_Results\_Gender[[2]], round(P\_Val, 2) < 0.5) col\_id <- c("P\_Val") # which column to change background color in

Prop\_Test\_Results\_Gender[[2]] %>% arrange(rev(Event)) %>% flextable\_style() %>%

bg(i = row\_id\_accept\_girls, j = col\_id,

bg = "green", part = "body") %>%

bg(i = row\_id\_reject\_girls, j = col\_id,

bg = "red",

part = "body") %>% colformat\_num(j = "P\_Val",

big.mark = ",", digits = 3) %>%

autofit()

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Event** | **Gender** | **Grade** | **Numb** | **Percentage** | **Total\_Athletes** | **P\_Val** |  |
| 1 mtr Diving | Girls | SR | 8 | 0.5333333 | 15 | 0.025 |  |
| 200 Yard Freestyle | Girls | SR | 9 | 0.5625000 | 16 | 0.009 |  |
| 200 Yard IM | Girls | SR | 8 | 0.5333333 | 15 | 0.025 |  |
| 50 Yard Freestyle | Girls | SR | 8 | 0.4705882 | 17 | 0.069 |  |
| 100 Yard Butterfly | Girls | SR | 4 | 0.2500000 | 16 | 1.000 |  |
| 100 Yard Freestyle | Girls | SR | 7 | 0.4375000 | 16 | 0.149 |  |
| 500 Yard Freestyle | Girls | SR | 8 | 0.5000000 | 16 | 0.043 |  |
| 100 Yard Backstroke | Girls | SR | 9 | 0.5625000 | 16 | 0.009 |  |
| 100 Yard Breaststroke | Girls | SR | 5 | 0.2941176 | 17 | 0.889 |  |
|  |  |  |  |  |  |  |  |

## Overall Proportion Test

In 15/18 (0.83%) of events the p-value is less than 0.5, meaning we can reject the null hypothesis for those events. There really are more seniors than a simple 1/4 probability would indicate, so in 83% of events seniors really do rule. We can also check for the whole meet:

Results\_Grade\_Sum %>% ungroup() %>%

mutate(Total\_Athletes = sum(Numb)) %>% filter(Grade == "SR") %>% mutate(Total\_Seniors = sum(Numb)) %>% select(Total\_Seniors, Total\_Athletes) %>% unique() %>%

mutate(P\_Val = prop.test(Total\_Seniors, Total\_Athletes, p = 0.25)$p.value[1])

%>%

flextable\_style()

|  |  |  |
| --- | --- | --- |
| **Total\_Seniors** | **Total\_Athletes** | **P\_Val** |
| 144 | 288 | 2.24775e-22 |

The p value (2.24e-22) is much less than 0.5 so we can reject the null hypothesis and conclude that at least

as far as this meet is concerned seniors really do rule.

# In Closing

Many thanks to all of you for joining us in another round of the State-Off here at Swimming + Data Science. Next week the final State-Off Champion will be crowned. Let’s update our bracket and prepare ourselves for a 1-2 matchup between California (1) and Texas (2)!

draw\_bracket( teams = c(

"California", "Texas", "Florida", "New York", "Pennsylvania", "Illinois", "Ohio", "Georgia"

),

round\_two = c("California", "Texas", "Florida", "Pennsylvania"), round\_three = c("California", "Texas"),

title = "Swimming + Data Science High School Swimming State-Off", text\_size = 0.9

)

