Get Trained & Get Strained: The Ingredients of a Big Data Bowl Metric

Big Data Bowl Workshop Sponsored by SumerSports

Carnegie Mellon Sports Analytics Conference 2023

Quang Nguyen

@qntkhvn qntkhvn qntkhvn.netlify.app

PLAN FOR TODAY

GOAL: an overview of a Big Data Bowl project from start to finish

What should you takeaway?

- NOT necessarily the code
- focus on the **process** and **concepts**
- how to present results
- generalizability to other sports

Case study: Developing a pass rush metric (inspired by 2023 BDB finalist entry)

Big Data Bowl Overview

NFL BIG DATA BOWL: BACKGROUND

- Premier sports analytics competition, focusing on player tracking data
- Data are collected at 10 Hz (10 frames per second) using RFID chips placed in player shoulder pads and ball
- Every year the NFL releases a sample a player-tracking data with a competition theme

NFL BIG DATA BOWL: THEMES

- Started in 2019, hosted on Kaggle since 2nd year
  - 2019: inaugural competition
  - 2020: yards gained after receiving a handoff (prediction competition)
  - 2021: defensive performance on pass plays
  - 2022: special teams performance
  - 2023: linemen on pass plays
  - ***2024: tackling***

Data

DATA OVERVIEW

- What are the provided datasets? (examples from BDB 2023)
  - Tracking data
  - Contextual data (game, play, player, PFF scouting data)
- Full data dictionary: https://www.kaggle.com/competitions/nfl-big-data-bowl-2023/data

(Note: Data are very similar for BDB 2024)

DATA: GAMES

`games.csv`: game info

DATA: PLAYS

`plays.csv`: play-level info

DATA: PLAYERS

`players.csv`: player info

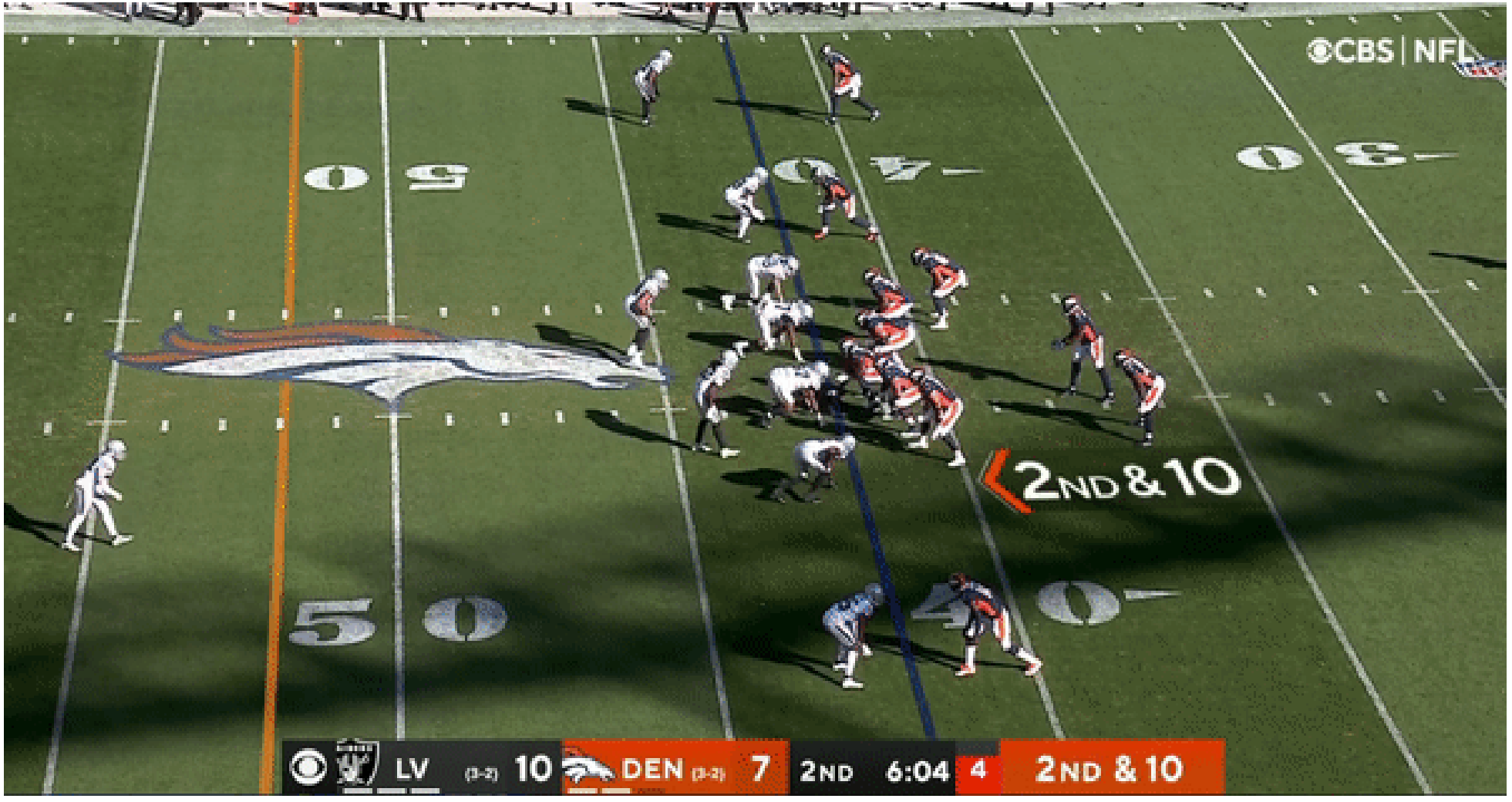
DATA: PFF SCOUTING

`pffScoutingData.csv`: PFF charted info about players in play (since BDB 2022)

DATA: TRACKING DATA

`week[i].csv`: tracking data for week `i` $\in [1:8]$ during 2021 NFL regular season

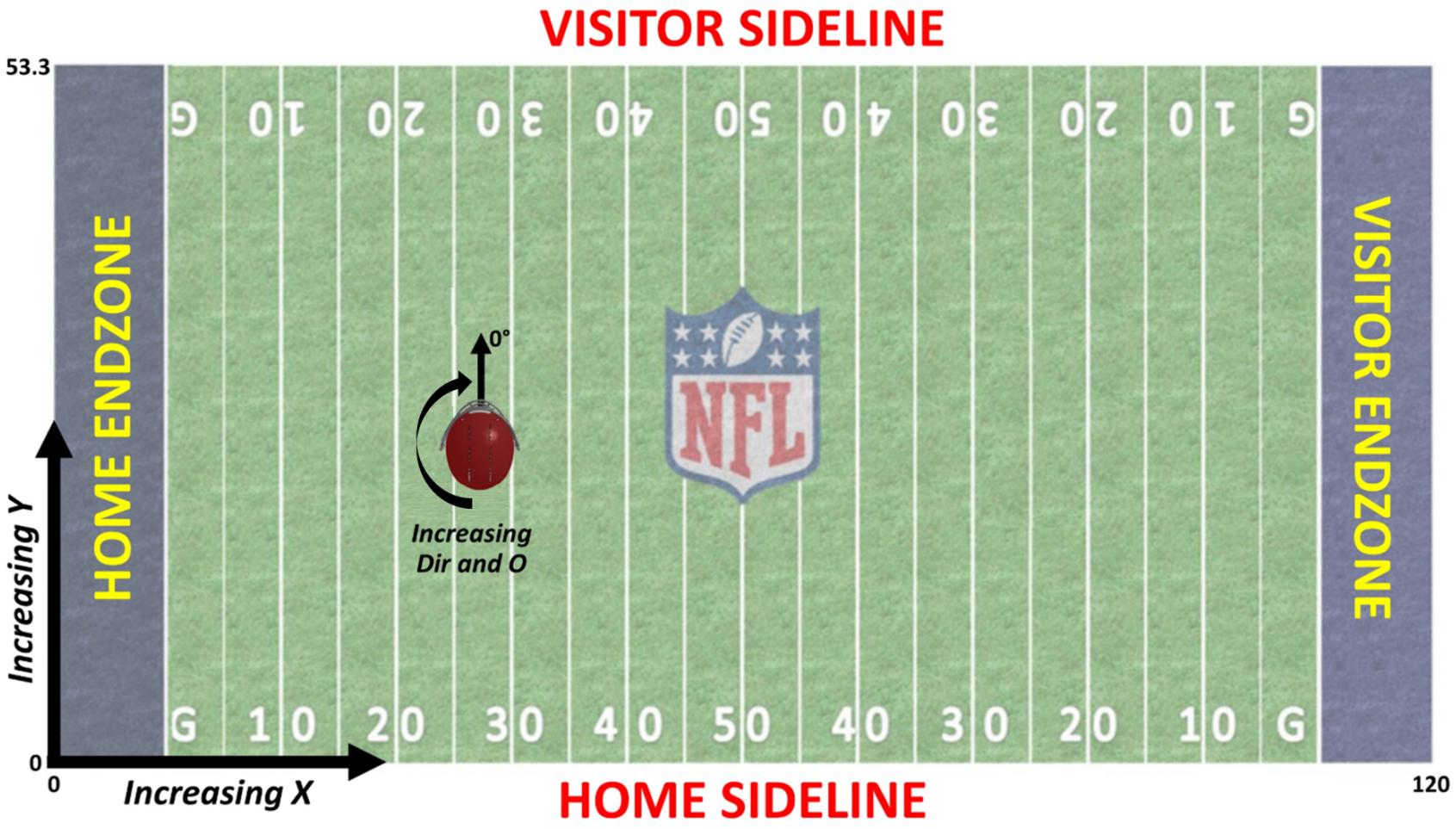
IN REAL LIFE



DATA PREP: FIELD STANDARDIZATION

- In the raw data, there are 2 play directions (with respect to the offense): left and right
- We need to standardize player tracking coordinates, so that the offense is always moving in the same direction
    - Helpful for feature engineering and modeling

DATA PREP: FIELD STANDARDIZATION



DATA PREP: FIELD STANDARDIZATION

- First, read in the data

(Data files are stored in the same directory as the slides' source file)

```
library(tidyverse)
theme_set(theme_bw())
games <- read_csv("games.csv")
plays <- read_csv("plays.csv")
players <- read_csv("players.csv")
pffScoutingData <- read_csv("pffScoutingData.csv")
tracking <- list.files(pattern = "week") |>
  map(read_csv) |>
  list_rbind()
```

- Make all plays go from left to right, then flip player direction and orientation

```
tracking <- tracking |>
  mutate(
    # make all plays go from left to right
    x = ifelse(playDirection == "left", 120 - x, x),
    y = ifelse(playDirection == "left", 160 / 3 - y, y),
    # flip player direction and orientation
    dir = ifelse(playDirection == "left", dir + 180, dir),
    dir = ifelse(dir > 360, dir - 360, dir),
    o = ifelse(playDirection == "left", o + 180, o),
    o = ifelse(o > 360, o - 360, o)
  )
```

VISUALIZING TRACKING DATA

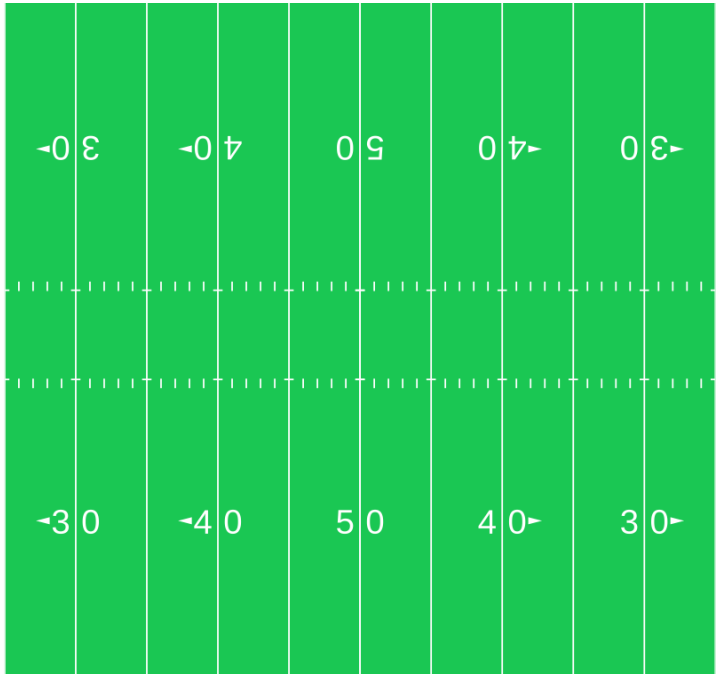
- Tip: pick out an example play as a case study throughout the project
- Example play: Raiders vs Broncos (week 6) – T. Bridgewater sacked by M. Crosby

frameId	x	y	s	a	dis	o	dir	event
7	67.68	29.89	0.34	1.57	0.04	124.86	88.21	ball_snap
8	67.76	29.89	0.69	2.13	0.08	124.07	89.59	None
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
50	73.67	25.06	4.19	2.62	0.42	134.21	125.26	qb_sack

VISUALIZING TRACKING DATA

- First, set up the field background with `sportyR`

```
library(sportyR)
field_params <- list(field_apron = "springgreen3",
                     field_border = "springgreen3",
                     offensive_endzone = "springgreen3",
                     defensive_endzone = "springgreen3",
                     offensive_half = "springgreen3",
                     defensive_half = "springgreen3")
nfl_field <- geom_football(league = "nfl",
                           display_range = "in_bounds_only",
                           x_trans = 60,
                           y_trans = 26.6667,
                           xlims = c(35, 85),
                           color_updates = field_params)
nfl_field
```



Metric Formulation

METRIC BACKGROUND

GOAL: Create a metric to evaluate pass rushers in the NFL

Performance metrics based on tracking data can be classified into two types:

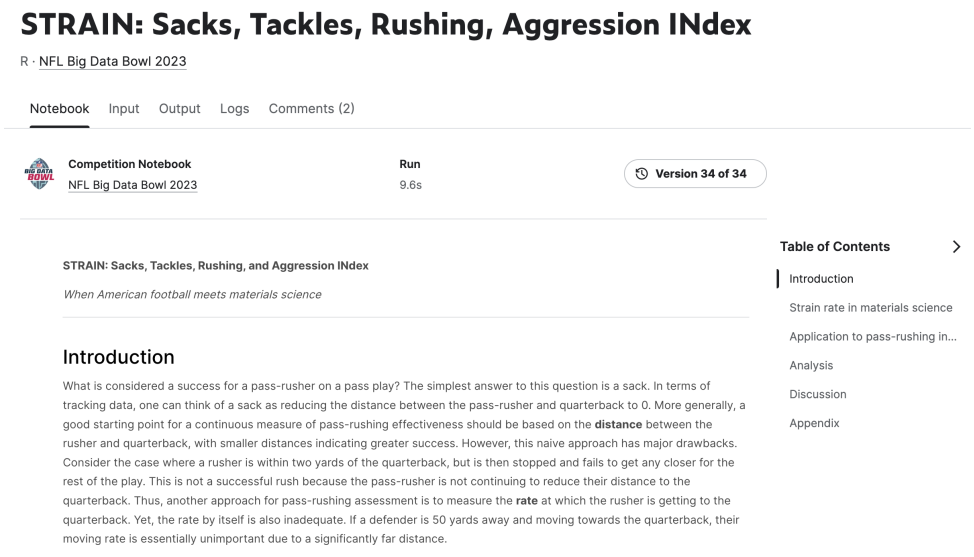
- directly-derived - directly calculated from data
- model-based - estimated through statistical model

Our proposed metric is directly-derived

Check out this review paper on Player Tracking Data in Sports by Kovalchik (2023)

RELATED LINKS

Link to Kaggle notebook



STRAIN: Sacks, Tackles, Rushing, Aggression INdex

R · NFL Big Data Bowl 2023

Notebook Input Output Logs Comments (2)

Competition Notebook
NFL Big Data Bowl 2023

Run
9.6s

Version 34 of 34

Table of Contents
- Introduction
- Strain rate in materials science
- Application to pass-rushing in...
- Analysis
- Discussion
- Appendix

STRAIN: Sacks, Tackles, Rushing, and Aggression INdex

When American football meets materials science

Introduction

What is considered a success for a pass-rusher on a pass play? The simplest answer to this question is a sack. In terms of tracking data, one can think of a sack as reducing the distance between the pass-rusher and quarterback to 0. More generally, a good starting point for a continuous measure of pass-rushing effectiveness should be based on the **distance** between the rusher and quarterback, with smaller distances indicating greater success. However, this naive approach has major drawbacks. Consider the case where a rusher is within two yards of the quarterback, but is then stopped and fails to get any closer for the rest of the play. This is not a successful rush because the pass-rusher is not continuing to reduce their distance to the quarterback. Thus, another approach for pass-rushing assessment is to measure the **rate** at which the rusher is getting to the quarterback. Yet, the rate by itself is also inadequate. If a defender is 50 yards away and moving towards the quarterback, their moving rate is essentially unimportant due to a significantly far distance.

Link to paper

Here Comes the STRAIN: Analyzing Defensive Pass Rush in American Football with Player Tracking Data

Quang Nguyen, Ronald Yurko, Gregory J. Matthews

In American football, a pass rush is an attempt by the defensive team to disrupt the offense and prevent the quarterback (QB) from completing a pass. Existing metrics for assessing pass rush performance are either discrete-time quantities or based on subjective judgment. Using player tracking data, we propose STRAIN, a novel metric for evaluating pass rushers in the National Football League (NFL) at the continuous-time within-play level. Inspired by the concept of strain rate in materials science, STRAIN is a simple and interpretable means for measuring defensive pressure in football. It is a directly-observed statistic as a function of two features: the distance between the pass rusher and QB, and the rate at which this distance is being reduced. Our metric possesses great predictability of pressure and stability over time. We also fit a multilevel model for STRAIN to understand the defensive pressure contribution of every pass rusher at the play-level. We apply our approach to NFL data and present results for the first eight weeks of the 2021 regular season. In particular, we provide comparisons of STRAIN for different defensive positions and play outcomes, and rankings of the NFL's best pass rushers according to our metric.

METRIC BRAINSTORMING

How do we measure pass rusher effectiveness?

- Distance to QB
- Speed toward QB

Think simple...

- Ideally, this measure should increase as speed increases and distance decreases
- Consider speed divided by distance

FOOTBALL MEETS MATERIALS SCIENCE

STRAIN RATE IN MATERIALS SCIENCE (EXAMPLE: STRETCHING A RUBBER BAND)	PASS RUSH IN FOOTBALL
Ratio of two quantities	Two factors for a good rush
(1) Stretching speed	(1) Fast velocity toward QB
(2) Original length	(2) Close distance to QB
Deformation of a material over time	Deformation of the pocket over time

MEASURING PRESSURE AT THE FRAME-LEVEL

Definition (STRAIN; informal)

For every pass rusher at each frame within a play, calculate

- d: distance between pass rusher and QB
- v: velocity at which pass rusher is moving toward QB
  - estimated by rate of change in distance between current & previous frames
  - $$\frac{\text{change in distance}}{\text{change in time}} = \frac{\text{current frame's distance} - \text{previous frame's distance}}{0.1}$$
- $$\text{STRAIN} = -\frac{v}{d}$$

ADVANTAGES OF STRAIN

- Simple, computation friendly: 2 features! #PutThatInYourXGBoost
- Interpretable: 1/STRAIN
  → time required for a pass rusher to get to QB with current velocity & distance
- Scalable: can be applied to every passing play (minus trick plays)
- Continuous-time within-play metric
  - Previous metrics are either discrete or based on subjective judgment

IMPLEMENTING STRAIN

What are the steps?

- Filter data to keep only frames between ball snap and a QB event (e.g., sack, pass forward, etc.)
- Identify locations of passer and pass rusher within each play
- Calculate distance and velocity, then STRAIN

IMPLEMENTING STRAIN

- Identify the starting and ending events, then collect the first observed start and end frames for each play

```
tracking_start_end <- tracking |>
  distinct(gameId, playId, frameId, event) |>
  mutate(is_start = as.numeric(event %in% c("autoevent_ballsnap", "ball_snap")),
         is_end = as.numeric(event %in% c("fumble", "handoff", "lateral", "autoevent_passforward",
                                          "pass_forward", "qb_sack", "qb_strip_sack", "run"))) |>
  group_by(gameId, playId) |>
  mutate(any_start = any(is_start == 1), any_end = any(is_end == 1)) |>
  filter(any_start, any_end) |>
  summarize(start_frame = frameId[which(is_start == 1)[1]],
            end_frame = frameId[which(is_end == 1 & frameId > start_frame)[1]]) |>
  ungroup()
```

- Join with tracking data and filter frames

```
tracking <- tracking |>
  left_join(tracking_start_end, by = c("gameId", "playId")) |>
  filter(!is.na(start_frame), !is.na(end_frame),
         frameId >= start_frame, frameId <= end_frame) |>
  mutate(frameId_corrected = frameId - start_frame)
```

IMPLEMENTING STRAIN

- Use PFF scouting data to identify player role for each play

```
tracking_roles <- tracking |>
  left_join(pffScoutingData, by = c("gameId", "playId", "nflId"))
```

- Get passer location for every frame of each play

```
tracking_qb <- tracking_roles |>
  filter(pff_role == "Pass", pff_positionLinedUp == "QB") |>
  select(gameId, playId, frameId, x_qb = x, y_qb = y)
```

Presenting Results

DATA VIZ PLAN

Goal: The visuals should tell stories and provide takeaway messages

- Viz 1: Updated version of animation for example play
  - Point size for each pass rusher represent their STRAIN at each frame
  - Highlight focused player (M. Crosby) with a different color
- Viz 2: How do the feature (distance, velocity, STRAIN) for M. Crosby interact within the play?
  - Line graph showing how each feature change as the play evolves

EXAMPLE PLAY ANIMATION

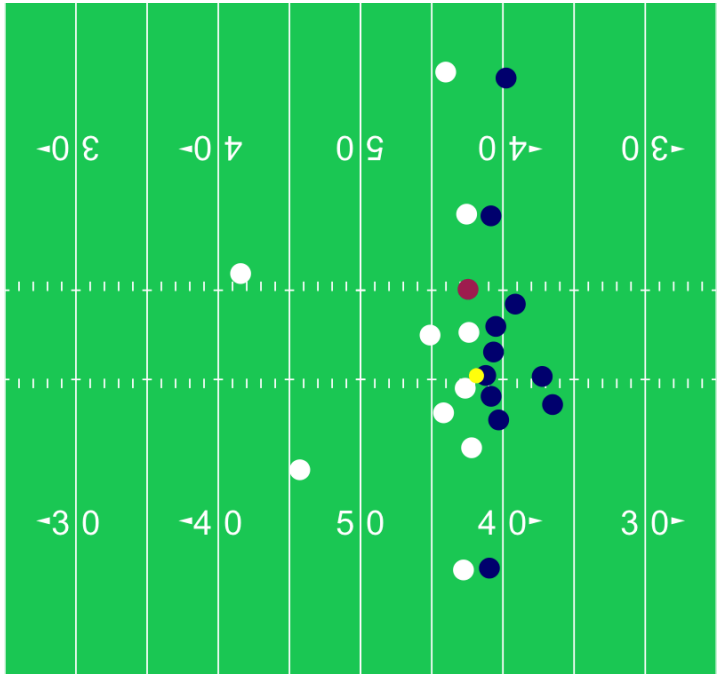
- Update original figure with STRAIN values for pass rushers
- Also adjust color and size of each player on the field

```
example_pass_rush <- tracking_pass_rush |>
  filter(gameId == 2021101709, playId == 1444, !is.na(strain))

example_play_strain <- example_play |>
  left_join(select(example_pass_rush, gameId:frameId, strain)) |>
  mutate(
    pt_color = ifelse(nflId != 47889, pt_color, "maroon"),
    pt_color = ifelse(team != "football", pt_color, "yellow"),
    pt_size = ifelse(is.na(strain), 3, scales::rescale(strain, to = c(3, 8), from = range(example_pass_rush$strain))),
    pt_size = ifelse(team == "football", 2, pt_size)
  )
```

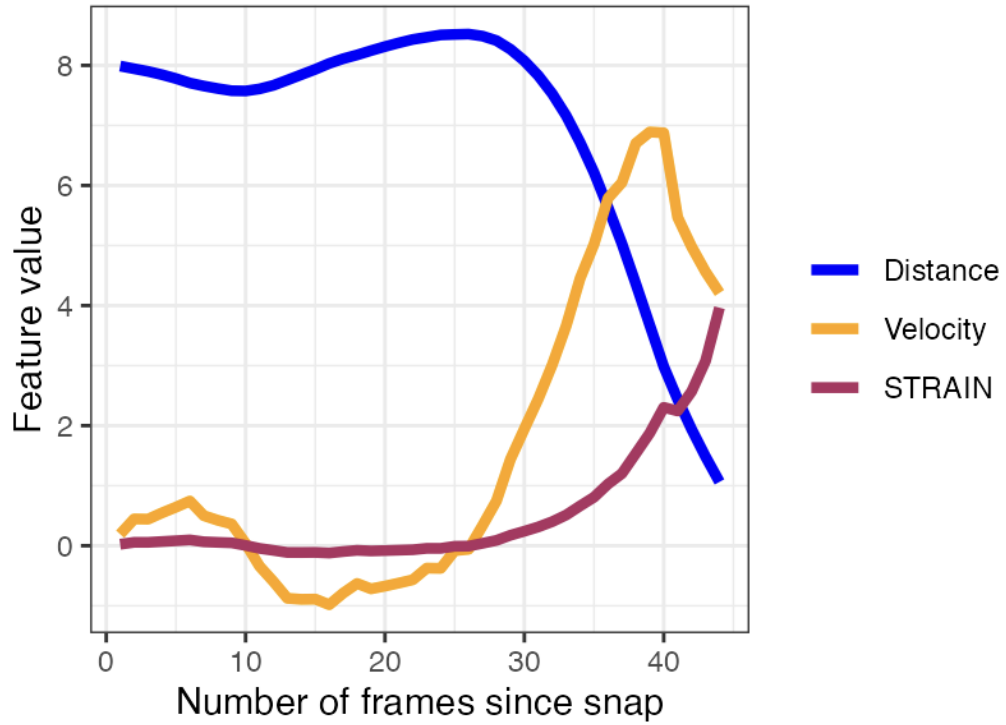
EXAMPLE PLAY ANIMATION

```
nfl_field +
  geom_point(data = example_play_strain,
             aes(120 - x, 160 / 3 - y),
             size = example_play_strain$pt_size,
             color = example_play_strain$pt_color) +
  transition_time(example_play_strain$frameId)
```



FEATURE CURVES

```
example_pass_rush |>
  filter(nflId == 47889) |>
  pivot_longer(cols = c(d_qb, v_qb, strain),
               names_to = "feature",
               values_to = "value") |>
  mutate(
    feature = fct_relevel(feature, "strain", after = Inf),
    feature = fct_recode(feature,
                         "Distance" = "d_qb",
                         "Velocity" = "v_qb",
                         "STRAIN" = "strain")
  ) |>
  ggplot(aes(frameId_corrected, value, color = feature)) +
  geom_line(linewidth = 1.5) +
  scale_color_manual(
    values = c("blue", "orange", "maroon")
  ) +
  labs(x = "Number of frames since snap",
       y = "Feature value",
       color = NULL)
```



BONUS: COMBINING ANIMATION

- We can also combine the play animation and the feature curves into one figure
- Check out this GitHub wiki on animation composition

(TLDR: the trick is to make sure both figures have the same duration (i.e., number of frames), then append them using `magick` package)

Metric Validation

DOES THE METRIC ACTUALLY MAKE SENSE?

- How does STRAIN differ across different positions? Play outcomes (sack, hit, hurry, none)?
- Who are the top players according to the metric? Does it pass the "eye test"?
- How does the metric relate to previous metrics?
- Is the metric stable over time?

Check out this paper on "meta-analytics" by Franks et al. (2017)

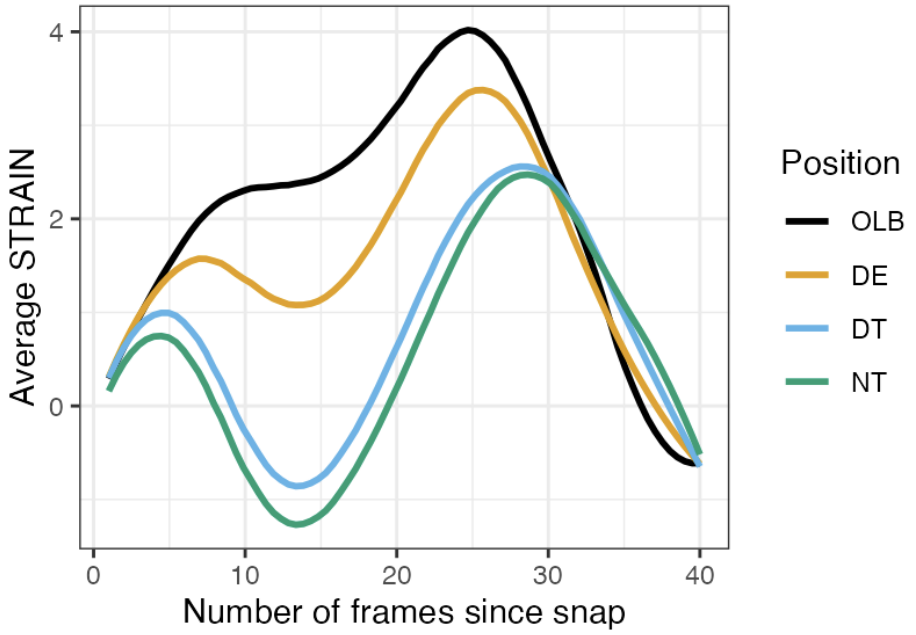
(and this sweet blog post by Tony ElHabr)

POSITIONAL STRAIN CURVES

- Join tracking info with `players` to get each player's position

```
tracking_pass_rush |>
  left_join(players) |>
  filter(
    frameId_corrected <= 40,
    officialPosition %in% c("OLB", "DE", "DT", "NT")
  ) |>
  mutate(
    officialPosition = fct_relevel(officialPosition,
                                   "OLB", "DE", "DT", "NT")
  ) |>
  group_by(frameId_corrected, officialPosition) |>
  summarize(avg_strain = 10 * mean(strain, na.rm = TRUE)) |>
  ggplot(aes(frameId_corrected, avg_strain,
             color = officialPosition)) +
  geom_smooth(se = FALSE, span = 0.3, linewidth = 1) +
  ggthemes::scale_color_colorblind() +
  labs(x = "Number of frames since snap",
       y = "Average STRAIN",
       color = "Position")
```

- STRAIN curve for each position (first 4s after snap)



PLAY OUTCOME STRAIN CURVES

- Determine whether a player recorded a sack, hit, hurry, or none for each play

```
pff_outcomes <- pffScoutingData |>
  group_by(gameId, playId, nflId) |>
  summarize(i_hit = sum(pff_hit, na.rm = TRUE) > 0,
            i_hurry = sum(pff_hurry, na.rm = TRUE) > 0 ,
            i_sack = sum(pff_sack, na.rm = TRUE) > 0,
            i_none = (i_hit + i_hurry + i_sack) == 0) |>
  ungroup() |>
  pivot_longer(i_hit:i_none,
               names_to = "outcome",
               names_prefix = "i_",
               values_to = "value") |>
  filter(value)
```

- STRAIN curves faceted by outcome

```
tracking_pass_rush |> left_join(pff_outcomes) |>
  left_join(players) |>
  filter(frameId_corrected <= 40,
         officialPosition %in% c("OLB", "DE", "DT", "NT")
  ) |>
  mutate(
    officialPosition = fct_relevel(officialPosition,
                                   "OLB", "DE", "DT", "NT"),
         outcome = fct_relevel(outcome,
                               "sack", "hit", "hurry", "none")
  ) |>
  group_by(frameId_corrected, outcome, officialPosition) |>
  summarize(avg_strain = 10 * mean(strain, na.rm = TRUE)) |>
  ggplot(aes(frameId_corrected, avg_strain,
             color = officialPosition)) +
  geom_smooth(se = FALSE, span = 0.3, linewidth = 1) +
  facet_wrap(~ outcome, nrow = 1) +
  ggthemes::scale_color_colorblind() +
  labs(x = "Number of frames since snap",
       y = "Average STRAIN",
       color = "Position") +
  theme(legend.position = "bottom")
```

PLAY OUTCOME STRAIN CURVES

RANKING THE BEST PASS RUSHERS

Goal: obtain a leaderboard of the top-performing pass rushers based on the average STRAIN across all frames played

Note: we will consider players with at least 100 snaps played as pass rushers

First, get players with at least 100 snaps played as pass rushers

```
pass_rushers_eval <- pffScoutingData |>
  filter(pff_role == "Pass Rush") |>
  group_by(nflId) |>
  summarize(snaps = n_distinct(gameId, playId)) |>
  filter(snaps >= 100)
```

RANKING THE BEST PASS RUSHERS

Calculate average STRAIN and output rankings with `gt`

- Columns: player, team, position, snaps played, average STRAIN
- Color gradient for metric column
- Check out `gt` and `gtExtras`

```
library(gt)
ranking_pass_rush <- tracking_pass_rush |>
  filter(nflId %in% pass_rushers_eval$nflId, !is.infinite(strain)) |>
  group_by(nflId) |>
  summarize(avg_strain = 10 * mean(strain, na.rm = TRUE),
            snaps = n_distinct(gameId, playId),
            team = first(team)) |>
  left_join(players) |>
  arrange(desc(avg_strain)) |>
  select(Player = displayName, Team = team, Position = officialPosition,
         Snaps = snaps, "Average STRAIN" = avg_strain)
```

TOP 10 EDGE RUSHERS (OLB & DE)

```
ranking_pass_rush |>
  filter(Position %in% c("OLB", "DE")) |>
  slice_head(n = 10) |>
  gt() |>
  fmt_number(
    columns = c("Average STRAIN"),
    decimals = 2
  ) |>
  data_color(
    columns = c("Average STRAIN"),
    fn = scales::col_numeric(
      palette = c("lightgreen", "darkgreen"),
      domain = NULL
    )
  ) |>
  tab_header(
    title = "Top 10 Edge Rushers",
    subtitle = "(Minimum 100 plays)"
  )
```

Top 10 Edge Rushers

(Minimum 100 plays)

Player	Team	Position	Snaps	Average STRAIN
Leonard Floyd	LA	OLB	200	2.77
Myles Garrett	CLE	DE	208	2.74
Justin Houston	BAL	OLB	138	2.68
T.J. Watt	PIT	OLB	155	2.67
Rashan Gary	GB	OLB	196	2.67
Alex Highsmith	PIT	OLB	136	2.65
Darrell Taylor	SEA	DE	112	2.63
Von Miller	DEN	OLB	159	2.60
Josh Sweat	PHI	DE	166	2.58
Preston Smith	GB	OLB	136	2.57

TOP 10 INTERIOR RUSHERS (DT & NT)

```
ranking_pass_rush |>
  filter(Position %in% c("DT", "NT")) |>
  slice_head(n = 10) |>
  gt() |>
  fmt_number(
    columns = c("Average STRAIN"),
    decimals = 2
  ) |>
  data_color(
    columns = c("Average STRAIN"),
    fn = scales::col_numeric(
      palette = c("lightblue", "darkblue"),
      domain = NULL
    )
  ) |>
  tab_header(
    title = "Top 10 Interior Rushers",
    subtitle = "(Minimum 100 plays)"
  )
```

Top 10 Interior Rushers

(Minimum 100 plays)

Player	Team	Position	Snaps	Average STRAIN
Aaron Donald	LA	DT	259	1.50
Chris Jones	KC	DT	150	1.44
Solomon Thomas	LV	DT	127	1.43
Quinton Jefferson	LV	DT	161	1.31
Cameron Heyward	PIT	DT	199	1.22
Javon Hargrave	PHI	DT	169	1.13
DeForest Buckner	IND	DT	217	1.08
Ed Oliver	BUF	DT	144	1.08
Greg Gaines	LA	NT	120	1.06
Jerry Tillery	LAC	DT	189	1.03

CORRELATION BETWEEN STRAIN AND PRESSURE RATE

Average STRAIN vs pressure rate $\left(\frac{\text{Hurries} + \text{Sacks} + \text{Hits}}{\text{Snaps}}\right)$

- Obtain hurries, sacks, hits from PFF scouting data

```
pff_pressure <- pffScoutingData |>
  group_by(nflId) |>
  summarize(hurries = sum(pff_hurry, na.rm = TRUE),
            sacks = sum(pff_sack, na.rm = TRUE),
            hits = sum(pff_hit, na.rm = TRUE))
```

CORRELATION BETWEEN STRAIN AND PRESSURE RATE

- Join with tracking data and calculate pressure rate

```
strain_pressure <- tracking_pass_rush |>
  filter(nflId %in% pass_rushers_eval$nflId,
         !is.infinite(strain)) |>
  group_by(nflId) |>
  summarize(avg_strain = 10 * mean(strain, na.rm = TRUE),
            snaps = n_distinct(gameId, playId)) |>
  left_join(pff_pressure) |>
  mutate(pressure_rate = (hurries + sacks + hits) / snaps)

strain_pressure |>
  slice_head(n = 5)
```

```
# A tibble: 5 × 7
  nflId avg_strain snaps hurries sacks  hits pressure_rate
  <dbl>      <dbl> <int>   <dbl> <dbl> <dbl>         <dbl>
1 33131      0.549   193      13     0     7        0.104
2 35441      0.696   210      12     2     5        0.0905
3 35454      1.36    183      11     2     1        0.0765
4 35470      1.75    133      24     1     3        0.211
5 35485      0.112   117      11     1     1        0.111
```

- Compute correlation

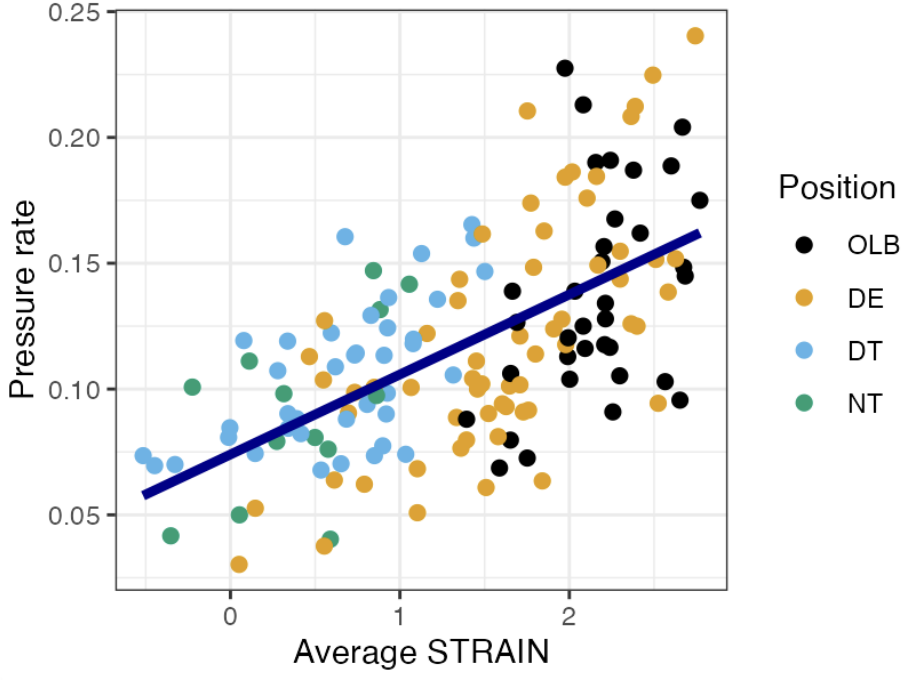
```
cor(strain_pressure$avg_strain,
    strain_pressure$pressure_rate)
```

```
[1] 0.6103378
```

STRAIN IS PREDICTIVE OF PRESSURE RATE

Scatterplot of pressure rate vs average STRAIN, color coded by position

```
strain_pressure |>
  left_join(players) |>
  mutate(
    officialPosition = fct_relevel(officialPosition,
                                   "OLB", "DE", "DT", "NT")
  ) |>
  ggplot(aes(avg_strain, pressure_rate)) +
  geom_point(aes(color = officialPosition), size = 2) +
  geom_smooth(method = "lm", se = FALSE,
              color = "darkblue", linewidth = 1.5) +
  ggthemes::scale_color_colorblind() +
  labs(x = "Average STRAIN",
       y = "Pressure rate",
       color = "Position")
```



METRIC STABILITY

Is previous performance predictive of future performance based on the metric? (How much does our metric vary over time?)

What's the correlation between a player's performance on our metrics in the first and second halves of the provided data (i.e. first 4 weeks vs last 4 weeks)?

METRIC STABILITY

- Obtain average STRAIN for the first and last 4 weeks of the 2021 season

```
strain_weeks <- tracking_pass_rush |>
  filter(nflId %in% pass_rushers_eval$nflId,
         !is.infinite(strain)) |>
  left_join(games) |>
  mutate(i_week = ifelse(week > 4, "last4", "first4")) |>
  group_by(nflId, i_week) |>
  summarize(avg_strain = 10 * mean(strain, na.rm = TRUE)) |>
  ungroup() |>
  pivot_wider(names_from = i_week, values_from = avg_strain)

strain_weeks |> slice_head(n = 5)
```

```
# A tibble: 5 × 3
  nflId first4 last4
  <dbl>  <dbl> <dbl>
1 33131  0.445 0.695
2 35441  0.813 0.574
3 35454  1.52  1.26
4 35470  1.92  1.62
5 35485  0.100 0.124
```

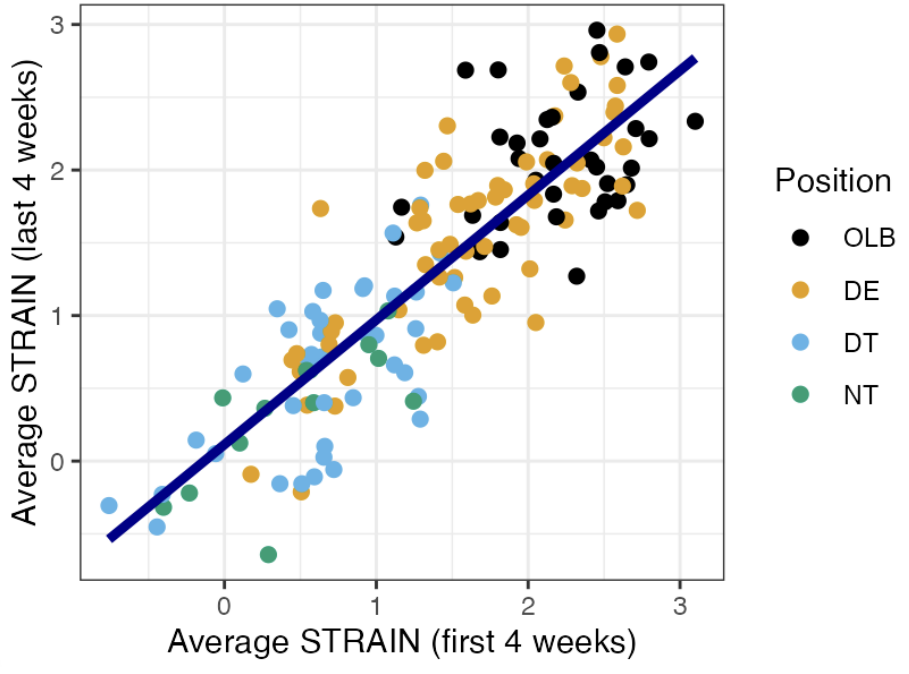
- Compute correlation

```
cor(strain_weeks$first4, strain_weeks$last4)
```

```
[1] 0.8602751
```

STRAIN IS HIGHLY STABLE OVER TIME

```
strain_weeks |>
  left_join(players) |>
  mutate(
    officialPosition = fct_relevel(officialPosition,
                                   "OLB", "DE", "DT", "NT")
  ) |>
  ggplot(aes(first4, last4)) +
  geom_point(aes(color = officialPosition), size = 2) +
  geom_smooth(method = "lm", se = FALSE,
              color = "darkblue", linewidth = 1.5) +
  ggthemes::scale_color_colorblind() +
  labs(x = "Average STRAIN (first 4 weeks)",
       y = "Average STRAIN (last 4 weeks)",
       color = "Position")
```



IS STRAIN PREDICTIVE OF PRESSURE THAN PRESSURE ITSELF?

Exercise. Make 2 scatterplots:

- Pressure rate (first 4 weeks) vs Pressure rate (last 4 weeks)
- Average STRAIN (first 4 weeks) vs Pressure rate (last 4 weeks)

Which plot depicts a stronger relationship?

Multilevel Model

MULTILEVEL MODEL: SPECIFICATION

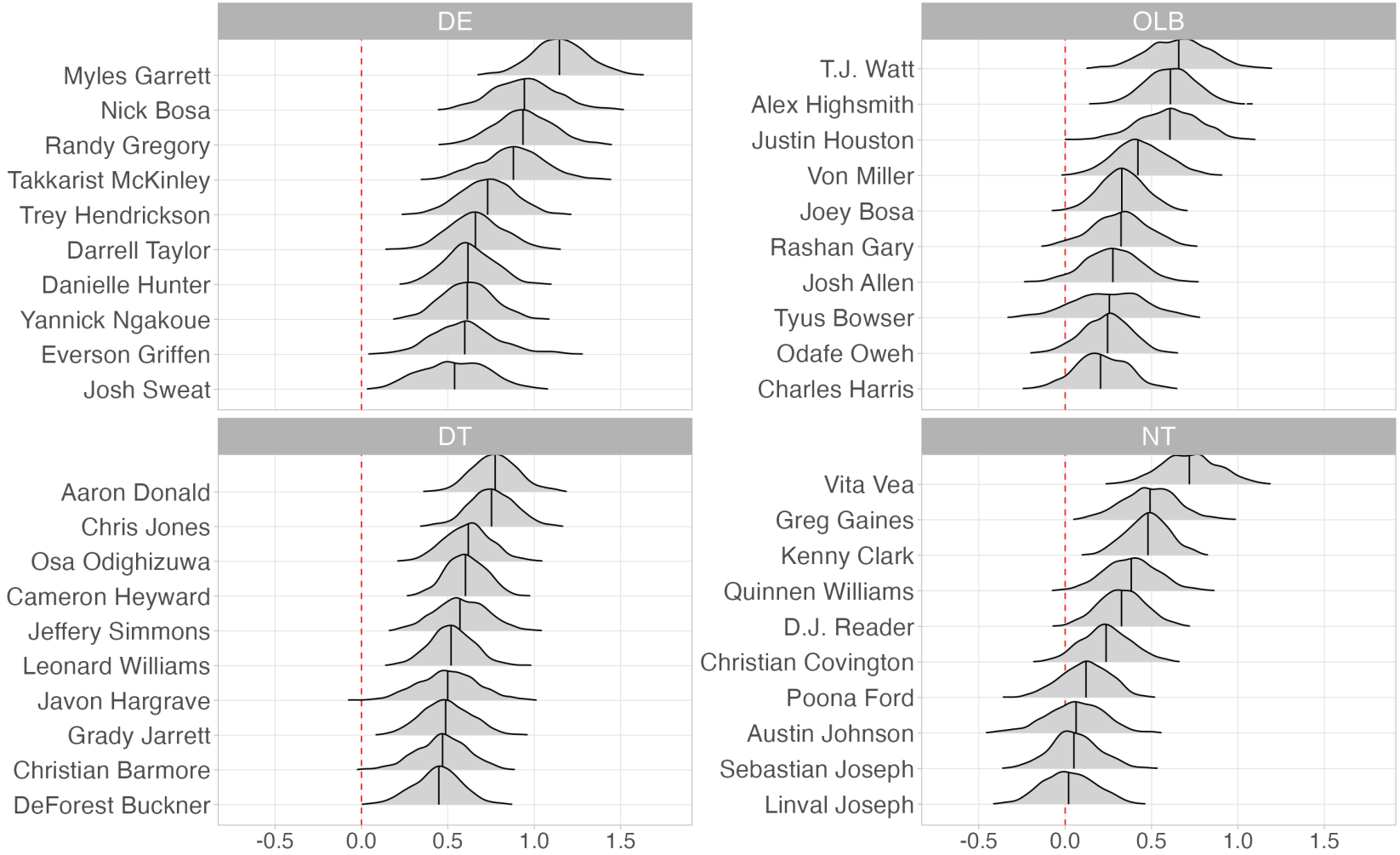
- Response: average STRAIN for pass rusher at the play-level
- Random effects: pass rusher, pass blocker, defensive team, offensive team
- Fixed effects
  - Number of blockers involved in a play
  - Blocker position (T, C, G, other)
  - Rusher position (DE, OLB, DT, NT, interior, secondary)
  - Play-context information: down, yards to go, current yardline

MULTILEVEL MODEL: RESAMPLING

- Capture uncertainty in the estimates for the player random effects
- It's unrealistic to bootstrap individual plays
- Instead, bootstrap team drives within games

(For code, see the paper's GitHub repo)

MULTILEVEL MODEL: PASS RUSHER RANKINGS



Pass rusher varying intercept

ADVICES

- Do it!
- Keep it simple. Don't try to do too much. Focus on a specific aspect.
- Think outside the box. Adapt concepts from other fields (physical sciences, etc.)
- Slick data viz is key
- Write (and prune words) as you work
- Enjoy learning (and the struggle)

MORE RESOURCES

- CMSAC22 workshop slides (Slide 29 provides more resources)
- Mike Lopez's tracking data tutorial
- Asmae Toumi's twitter thread on BDB tips
- Nick Wan's Kaggle tutorial notebooks + twitch stream during BDB season
- Past winner/finalist/honorable mention entries: 2019 and 2020, 2021, 2022, 2023

ACKNOWLEDGEMENTS

- Collaborators:

  Ron Yurko & Greg Matthews

- Sponsor: SumerSports

Thank you for your attention!