So, an exploration of how tennis tournament seeding and bracketing impacts on the end result has percolated to the top of my to-do list, inspired by the Melbourne Open currently in play.

This is the first of what will probably be two posts on this topic. Today I wanted to look at the impact of [seeding](https://en.wikipedia.org/wiki/Seed_(sports)) on the chance of the best players finishing first, or in the top 2, 4 or 8 players (ie the grand, semi or quarter finals). I compared the results of simulated single-elimination tournaments between these players in two different tournament types – completely random allocation of the draw, or seeding for the top 32 players as was standard in Grand Slam tournaments, [apparently until 2019](http://on-the-t.com/2018/02/02/slam-seeding-changes/).

There’s a material impact from the approach to the draw, which can be seen in these results:

This means, for example, that with 32 top players given special “seed” allocations in the draw, there is a 22% chance that the semi finals will include the four best players; but only a 2% chance if the draw is allocated completely at random. The chance of seeing the eight best players in the quarter finals under random allocation is effectively zero, whereas with a seeded draw this will happen 3% of the time (still not very often – which should be no surprise for tennis fans). On the other hand, the chance of seeing the top two seeds playing out the final is 42% in the seeded contest and only 20% when seeded.

Note that these results will depend upon the distribution of players’ strengths in a given tournament. At an extreme, if the top player completely dominates all others, seeding will make no difference to the winner. We can calculate some other theoretical values as limits. For example, if the top two players were effectively invincible when playing anyone except eachother, they would meet in the final 100% of the time in a seeded competition, and 50% of the time if the draw was unseeded. So anything below those levels (42% and 20% in our case) indicates the gap between the two players being “really good” compared to the rest of the field and “infinitely good”.

**Historical tennis results data**

To perform my analysis I used data on the actual relative strength of players from [{](https://github.com/skoval/deuce)deuce R Package} . I wanted a realistic range of strengths that could go head-to-head in a real life tournament, so I chose a single point of time rather than a hypothetical cross-history match up that (for example) would pitch Margaret Court versus Serena Williams. To avoid confusion with contemporary reality, I chose 1990 as my year. Here are the top 10 womens’ tennis players at the end of 1990, as judged by their Elo ratings:

# A tibble: 128 x 3

player\_id player\_name elo

1 200414 Steffi Graf 2608.

2 200293 Martina Navratilova 2474.

3 200652 Monica Seles 2388.

4 200572 Gabriela Sabatini 2312.

5 200597 Mary Joe Fernandez 2236.

6 200017 Arantxa Sanchez Vicario 2195.

7 200049 Conchita Martinez 2194.

8 200077 Jennifer Capriati 2169.

9 200401 Manuela Maleeva Fragniere 2151.

10 200404 Zina Garrison 2139.

This is the top 10 rows of a 128 row dataframe women128 created with the chunk of R code below:

library(foreach)

library(doParallel)

library(tidyverse)

library(scales)

library(kableExtra)

library(clipr)

# deuce package by Stephanie Kovalchik to get Elo ratings of real tennis players

# devtools::install\_github("skoval/deuce")

library(deuce)

data(wta\_elo)

# Identify the top 128 women playing in 1990 and their last Elo rating that year

women128 <- wta\_elo %>%

filter(year(tourney\_start\_date) == 1990) %>%

# pick just the latest Elo rating

group\_by(player\_name) %>%

arrange(desc(tourney\_start\_date)) %>%

slice(1) %>%

ungroup() %>%

arrange(desc(overall\_elo)) %>%

dplyr::select(player\_id, player\_name, elo = overall\_elo) %>%

slice(1:128)

They are a powerful method of calculating ratings based on actual performance, with the great advantage of being convertable into probabilities for any hypothetical match up. Incremental adjustments are made to the rating based on how actuality relates to the probabilities derived from Elo ratings going in to the match. This makes them a useful and self-correcting metric that is readily incorporated into a statistical model.

To be sure I am using the correct basis for converting Kovalchik’s Elo ratings into probabilities, I’m going to use her elo\_prediction() function for estimating the chance of either of a pairing winning. To illustrate, here is a chart showing the chance of a selection of the players ranked 2 to 128 by Elo rating of beating Steffi Graf, the highest rated player in our subset of the data (and in fact, the highest rated player by this method in the data available, beginning in 1968 with the start of the open era).

Here’s the code for that illustration:

# Chance of beating Steffi Graf

steffis\_opponents <- c(2:5, 10, 20, 50, 128)

y <- elo\_prediction(women128[steffis\_opponents, ]$elo, women128[1, ]$elo)

the\_caption <- "Source: <http://freerangestats.info> analysis based on 1990 WTA ratings in Stephanie Kovalchik's {deuce} R package."

cbind(women128[steffis\_opponents, ], y, rank = steffis\_opponents) %>%

mutate(lab = paste0(player\_name, " (", rank, ")")) %>%

ggplot(aes(x = elo, y = y, label = lab)) +

geom\_point() +

geom\_text\_repel(size = 3, colour = "steelblue") +

scale\_y\_continuous(label = percent) +

labs(title = "Probability of beating Steffi Graf at the end of 1990",

subtitle = "Various players from the top 128 WTA players, probability based only on Elo rating",

caption = the\_caption,

x = "Elo rating of Ms Graf's hypothetical opponent (ranking shown in brackets)",

y = "Ms Graf's opponent's probability of winning")

**Simulating tournament draws and results**

For the grunt work of simulating tournaments between these 128 players, I first write a function simulate\_tournament() which takes as its main argument a data frame of 128 rows in sequence to represent position in the draw. The input to this function is going to look like this:

> brackets

# A tibble: 896 x 5

player\_id round match player\_name elo

1 200494 128 1 Dianne Van Rensburg 1855.

2 200423 128 1 Carling Bassett Seguso 1822.

3 200481 128 2 Elna Reinach 1777.

4 200506 128 2 Wiltrud Probst 1768.

5 200699 128 3 Meredith Mcgrath 1857.

6 200086 128 3 Magdalena Maleeva 1825.

7 200419 128 4 Kathy Rinaldi Stunkel 1804.

8 200624 128 4 Tami Whitlinger Jones 1647.

9 200395 128 5 Catherine Tanvier 1732.

10 200360 128 5 Pam Shriver 2047.

...

This indicates (for example) that in the “round of 128” – the first round played – Van Rensburg will play Seguso in match one. If we filter the object to match one of round two (the “round of 64) we see:

> filter(brackets, round == 64 & match ==1)

# A tibble: 4 x 5

player\_id round match player\_name elo

1 200494 64 1 Dianne Van Rensburg 1855.

2 200423 64 1 Carling Bassett Seguso 1822.

3 200481 64 1 Elna Reinach 1777.

4 200506 64 1 Wiltrud Probst 1768.

There are now four players in match 1. However, one of Van Resnburg or Seguso will have lost in the round of 128; and so will one of Reinach and Probst. With a bit of care, this object brackets contains the full draw of the elimination tournament, and can be constructed so the top 32 seeds are allocated to [the brackets required in a 32 seed draw](https://en.wikipedia.org/wiki/Template:32TeamBracket). The remainder of the code in the chunk below does this for my two different methods of draws. It’s a bit clunky but it seems to work.

#' Function to simulate the result of a tournament of 128 players

#'

#' @param brackets a data frame or tibble with columns for player\_id (player id), round (referring to

#' the number of players left in that round of the tournament - 2,4,8,16,32,64 and 128),

#' match (match id within that round)

simulate\_tournament <- function(brackets){

one\_sim\_results <- tibble(winner = numeric(),

loser = numeric(),

round = numeric(),

match = numeric())

all\_rounds <- unique(brackets$round)

remaining\_players <- brackets

for(this\_round in all\_rounds){

all\_matchups <- remaining\_players %>%

filter(round == this\_round) %>%

pull(match) %>%

unique()

for(this\_match in all\_matchups){

these\_players <- remaining\_players %>%

filter(round == this\_round) %>%

filter(match == this\_match) %>%

select(player\_id, elo)

prob\_win <- deuce::elo\_prediction(these\_players[1, ]$elo, these\_players[2, ]$elo)

a\_wins <- rbinom(1, 1, prob = prob\_win) == 1

if(a\_wins){

update <- tibble(winner = these\_players[1, ]$player\_id,

loser = these\_players[2, ]$player\_id,

round = this\_round,

match = this\_match)

} else {

update <- tibble(winner = these\_players[2, ]$player\_id,

loser = these\_players[1, ]$player\_id,

round = this\_round,

match = this\_match)

}

one\_sim\_results <- rbind(one\_sim\_results, update)

}

remaining\_players <- brackets %>%

filter(!player\_id %in% one\_sim\_results$loser)

}

return(one\_sim\_results)

}

#' Assign bracket slots to 128 players

#'

#' @param d a data frame or tibble with 128 rows and one column player\_id

#' @return a long data frame with columns for player\_id, round and match indicating which

#' match each player is in in each round (if they survive to that round). "round" is indicated

#' by the convention of number of players left ie "Round of 128" is the first round, "Round of 2"

#' is the grand final.

#' @details The original order is retained so this can be used (with caution) to allocate

#' matches to a seeded tournament. For example, with 32 seeds if the original data frame

#' is in the order of position32 from the round\_of\_32\_seeding object, with each position

#' followed by 7 unseeded players.

assign\_rounds <- function(d){

d %>%

mutate(position128 = rep(1:64, each = 2),

position64 = rep(1:32, each = 4),

position32 = rep(1:16, each = 8),

position16 = rep(1:8, each = 16),

position8 = rep(1:4, each = 32),

position4 = rep(1:2, each = 64),

position2 = rep(1:1, each = 128)) %>%

gather(round, match, -player\_id) %>%

mutate(round = as.numeric(gsub("position", "", round)))

}

n\_sims <- 10000

#-----------set up parallel processing-----------

cluster <- makeCluster(7) # only any good if you have at least 7 processors :)

registerDoParallel(cluster)

clusterEvalQ(cluster, {

library(foreach)

library(tidyverse)

library(deuce)

})

clusterExport(cluster, c("women128", "assign\_rounds", "simulate\_tournament"))

#-------------------unseeded tournament-------------------

r1 <- foreach(this\_sim = 1:n\_sims, .combine = rbind) %dopar% {

# new random positionining of all 128 players

brackets <- women128 %>%

sample\_n(128) %>%

select(player\_id) %>%

assign\_rounds() %>%

left\_join(women128, by = "player\_id")

one\_sim\_results <- simulate\_tournament(brackets)

one\_sim\_results %>%

mutate(sim = this\_sim,

method = "No seeding")

}

#------------------------seeded tournament-----------

# <https://en.wikipedia.org/wiki/Template:32TeamBracket>

round\_of\_32\_seeding <- tibble(

position32 = rep(1:16, each = 2),

seed = c(1, 32, 17, 16, 9, 24, 25, 8, 5, 28, 21, 12, 13, 20, 29, 4,

3, 30, 19, 14, 11, 22, 27, 6, 7, 26, 23, 10, 15, 18, 31, 2)

)

stopifnot(all(1:32 %in% round\_of\_32\_seeding$seed))

# players and their position in the round of 32 bracket - doesn't change in each sim

players <- women128 %>%

arrange(desc(elo)) %>%

mutate(ranking = 1:128,

seed = ifelse(ranking <= 32, ranking, NA)) %>%

left\_join(round\_of\_32\_seeding, by = "seed")

clusterExport(cluster, "players")

r2 <- foreach(this\_sim = 1:n\_sims, .combine = rbind) %dopar% {

unseeded <- players[-(1:32), ] %>%

# in random order:

sample\_n(n()) %>%

mutate(position32 = 1:n() %% 16 + 1) %>%

arrange(position32)

# actual brackets change each sim for players who aren't one of the top 32 seeds

brackets <- rbind(filter(players[, c("player\_id", "position32")], ![is.na](http://is.na)(position32)),

unseeded[, c("player\_id", "position32")]) %>%

arrange(position32) %>%

assign\_rounds() %>%

left\_join(select(players, -position32), by = "player\_id") %>%

arrange(desc(round), match)

one\_sim\_results <- simulate\_tournament(brackets)

one\_sim\_results %>%

mutate(sim = this\_sim,

method = "Seeded (32 seeds)")

}

results <- rbind(r1, r2) %>%

left\_join(players, by = c("winner" = "player\_id")) %>%

rename(winner\_name = player\_name)

Which gets us to our results.

The distribution of the winner of the whole tournament is summarised in the table below. Unsurprisingly, Ms Graf wins more of our simulated tournaments than any other player; 5,454/10,000 when the draw is at random and 5,913/10,000 when we use the 32-seed draw method. But even in this period of Graf’s dominance in women’s tennis, other players (and more than just the top few) have a non-zero chance of winning. Which is why people watch, of course. Interestingly, once we get to Monica Seles (ranked 3 by Elo at this point in time) and lower, players are more likely to win the overall tournament with an unseeded rather than a seeded draw.

| **winner\_name** | **Actual ranking** | **No seeding** | **Seeded (32 seeds)** |
| --- | --- | --- | --- |
| Steffi Graf | 1 | 5454 | 5913 |
| Martina Navratilova | 2 | 2089 | 2219 |
| Monica Seles | 3 | 1064 | 1018 |
| Gabriela Sabatini | 4 | 510 | 418 |
| Mary Joe Fernandez | 5 | 213 | 132 |
| Conchita Martinez | 7 | 110 | 66 |
| Arantxa Sanchez Vicario | 6 | 113 | 63 |
| Jennifer Capriati | 8 | 89 | 42 |
| Manuela Maleeva Fragniere | 9 | 66 | 34 |
| Zina Garrison | 10 | 58 | 26 |

This is also the point at which we can interrogate our simulation results to get the image I started the blog with:

This analysis of the simulation results was made with this chunk of code

# Summarise the individual winners of the overall tournament:

results %>%

filter(round == 2) %>%

group\_by(winner\_name, method, ranking) %>%

summarise(wins = n()) %>%

ungroup() %>%

select(winner\_name, method, wins, `Actual ranking` = ranking) %>%

complete(winner\_name, method, fill = list(wins = 0)) %>%

pivot\_wider(names\_from = method, values\_from = wins) %>%

arrange(desc(`Seeded (32 seeds)`)) %>%

slice(1:10) %>%

kable() %>%

kable\_styling(full\_width = FALSE)

#' Present the success rate of different simulated draw methods

#'

#' @param results data frame of results as created earlier in this script

#' @param women128 data frame with extra details on the 128 players in the tourname

#' @param how many of the top players you want to compare the draw methods for. Must be a power of 2.

#' @details Very specific to this particular bit of analysis

success\_rate <- function(results, women128, top\_x){

if(log(top\_x, 2) != as.integer(log(top\_x, 2))){

stop("top\_x must be a power of 2")

}

top\_players <- women128[1:top\_x, ]$player\_name

# we're going to count this by counting the winners in the previous round. eg to find if the top 2

# players are the final 2 in the competition, we see if they are in the winners from the round of 4:

tmp <- results %>%

filter(round == (top\_x \* 2)) %>%

arrange(sim, method, match) %>%

group\_by(sim, method) %>%

summarise(best = all(winner\_name %in% top\_players)) %>%

group\_by(method) %>%

summarise(best = mean(best)) %>%

mutate(top\_x = top\_x)

return(tmp)

}

# Summary graphic

rbind(

success\_rate(results, women128, top\_x = 8),

success\_rate(results, women128, top\_x = 4),

success\_rate(results, women128, top\_x = 2),

success\_rate(results, women128, top\_x = 1)

) %>%

ggplot(aes(x = as.ordered(top\_x), y = best, colour = method)) +

geom\_point() +

geom\_line(aes(x = as.numeric(as.ordered(top\_x)))) +

scale\_y\_continuous(label = percent) +

expand\_limits(y = 0) +

labs(x = "Number of top players of interest",

y = "Chance of those players being the\nlast ones in the competition",

colour = "Method of allocating draw:",

title = "Impact of seeding on the better players' end results in a tennis tournament",

subtitle = "Seeding materially improves the chance of the best player winning or the best two players being in the

grand final,and impacts the semi- and quarter- finals even more.",

caption = the\_caption)

That’s all for today. Some time soon I hope to come back to this and compare both methods to the alternative proposed by Charles Dodgson *Lawn Tennis Tournaments. The True Method of Assigning Prizes with a Proof of the Fallacy of the Present Method*. Dodgson, who as well as being a mathematician and logician found time to invent one of the most recognised characters in English literature, was writing at a time before seeding the draw was common and proposed an alternative to the single elimination tournament that in his view was guaranteed to give the correct first three prizes to the three best players. However, he did depend on a non-probabilistic view of what “best” means which is worth probing.