An attempt at analyzing the TCEC Season 15 SuFi openings

Joseph V. Iris

2019-05-28

My aim here is to try to analyze and make sense of what happened in the TCEC Seasono 15 SuFi, based on the ECO group of openings. Based on Jeroen Noomen's blog, the following are the intended openings:

```
ECO code distribution
ECO A: 15 lines
ECO B: 14 lines
ECO C: 11 lines
ECO D: 3 lines
ECO E: 7 lines
```

However, the ECO code distribution may have changed because of some transpositions. I have stored the results of the TCEC Season 15 SuFi here.

```
library(tidyverse)
library(elo)
library(flextable)
library(officer)
data <- read_delim("./leelasf2.csv", delim = ";")
data %>% flextable() %>% autofit()
```

Opening	White	Black	points.White	points.Black	ECO1	plies	Leela.openeval	SF
1	SF	Leela	0.5	0.5	E73	13	0.79	
2	Leela	SF	0.5	0.5	E73	13	1.28	
3	SF	Leela	0.5	0.5	B84	30	0.42	
4	Leela	SF	0.5	0.5	B84	30	0.43	
5	SF	Leela	0.5	0.5	A80	3	0.81	
6	Leela	SF	0.5	0.5	A80	3	0.84	
7	SF	Leela	0.5	0.5	C37	8	-1.16	
8	Leela	SF	0.5	0.5	C37	8	-1.06	
9	SF	Leela	0.5	0.5	A60	6	1.20	
10	Leela	SF	1.0	0.0	A67	6	1.17	
11	SF	Leela	0.5	0.5	C05	17	0.35	
12	Leela	SF	0.0	1.0	C05	17	0.38	
13	SF	Leela	0.5	0.5	A30	22	0.58	
14	Leela	SF	0.5	0.5	A30	22	0.72	
15	SF	Leela	0.5	0.5	B06	8	1.30	
16	Leela	SF	1.0	0.0	B06	8	1.38	
17	SF	Leela	0.5	0.5	E97	22	1.10	
18	Leela	SF	1.0	0.0	E97	22	1.57	
19	SF	Leela	0.5	0.5	B69	13	0.66	
20	Leela	SF	0.5	0.5	B69	13	0.68	
21	SF	Leela	0.5	0.5	D31	14	0.66	
22	Leela	SF	0.5	0.5	D31	14	0.70	
23	SF	Leela	0.5	0.5	C92	24	0.91	
24	Leela	SF	1.0	0.0	C92	24	1.10	
25	SF	Leela	0.5	0.5	E15	21	0.88	
26	Leela	SF	1.0	0.0	E15	21	0.90	
27	SF	Leela	0.5	0.5	B01	4	1.10	
28	Leela	SF	0.5	0.5	B01	4	1.13	
29	SF	Leela	0.5	0.5	A50	4	0.95	
30	Leela	SF	0.5	0.5	A50	4	0.98	
31	SF	Leela	0.5	0.5	C52	17	-0.30	
32	Leela	SF	0.5	0.5	C52	17	-0.31	
33	SF	Leela	0.5	0.5	E83	12	0.73	
34	Leela	SF	0.5	0.5	E84	12	0.79	
35	SF	Leela	1.0	0.0	B90	22	0.99	
36	Leela	SF	1.0	0.0	B90	22	1.00	

First, let's estimate the ELO differences between Leela and Stockfish after every game. Initially, the estimated ELO's are 3589 for Leela and 3587 for Stockfish.

Let R_A be the ELO of engine A and R_B be the rating of engine B. Then the expected result for engine A against B is given by the logistic equation:

$$E_A = \frac{1}{1 + 10^{(R_A - R_B)/400}}. \tag{1}$$

Solving this equation for $R_A - R_B$, we have:

$$elodiff = R_A - R_B = 400 \log_{10}\left(\frac{1 - E_A}{E_A}\right) \tag{2}$$

Here we note that $(1 - E_A)/E_A$ can be expressed as win ratio / loss ratio without loss of generality. That is, we can put the win ratio of the leading engine in the numerator and we get the same result. The win ratio is the sum of the wins and draws.

In R, there is a package called `elo` which we will also use here. But we can write our own functions for this purpose.

```
elo <- function(win_ratio) {400 * log10(win_ratio / (1-win_ratio))}
```

We can also check for the standard errors of ELO differences using a normal approximation.

```
denom95 <- function(win_ratio, total) qnorm(0.975) * sqrt(win_ratio * (1-win_ratio)/(total-1))
```

We can also compute for the LOS as described in the chessprogramming wiki site. I used three estimators here. `LOS3` might become untenable with large data sets, but we only have 100 rows of data here so it will be fine.

```
LOS <- function(wins_losses, total) pnorm(total/2, sd = wins_losses)
LOS2 <- function(wins, losses) pnorm((wins-losses)/sqrt(wins+losses))
LOS3 <- function(wins, losses, draws) {
  total = wins + losses + draws
  exp = (wins/total)^wins * (losses/total)^losses * (draws/total)^draws
  factorials = factorial(total)/(factorial(wins)*factorial(losses)*factorial(draws))
  P = factorials * exp
  1-P
}
```

We will now extract the initials of the ECO codes, determine the points of Leela and SF after each game, the win rate (by the leading engine) after each game, the estimated ELO difference (`elodiff`) after each game, and the three LOS estimates after each game.

```
data <- data %>%
  mutate(ECO2 = substr(ECO1, start = 1, stop = 1)) %>%
  # calculate Leela's scores
  mutate(points.Leela = (White == "Leela") * points.White + (Black == "Leela") * points.Black) %>%
  # calculate SF's scores
  mutate(points.SF = (White == "SF") * points.White + (Black == "SF") * points.Black) %>%
  mutate(results.Leela = case_when(points.Leela == 1~"Win",
                                   points.Leela == 0.5~"Draw",
                                   points.Leela == 0~"Loss")) %>%
  mutate(results.SF = case_when(points.SF == 1~"Win",
                                   points.SF == 0.5~"Draw",
                                   points.SF == 0~"Loss")) %>%
```

```
  # calculate cumulative scores
  mutate(Score.Leela = cumsum(points.Leela)) %>%
  mutate(Score.SF = cumsum(points.SF)) %>%
  mutate(total = row_number()) %>%
  mutate(draw_ratio = cumsum(points.Leela == points.SF)/total) %>%
  mutate(wins.Leela = cumsum(results.Leela=="Win")) %>%
  mutate(losses.Leela = cumsum(results.Leela=="Loss")) %>%
  mutate(wins.SF = cumsum(results.SF=="Win")) %>%
  mutate(losses.SF = cumsum(results.SF=="Loss")) %>%
  mutate(Draws = cumsum(results.Leela=="Draw")) %>%
  # calculate win rate of Leela
  mutate(win_rate.Leela = Score.Leela/total) %>%
  mutate(elodiff = elo(win_rate.Leela)) %>%
  # calculate ELO's and LOS's
  mutate(SE = elo(win_rate.Leela + denom95(win_rate.Leela, total))-elodiff) %>%
  mutate(LOS = LOS(total*(1-draw_ratio), total)) %>%
  mutate(LOS2 = LOS2(wins.Leela, losses.Leela)) %>%
  mutate(LOS3 = LOS3(wins.Leela, losses.Leela, Draws))
data %>%
  select(Opening, ECO2, win_rate.Leela:LOS3) %>%
  flextable() %>% autofit()
```

Opening	ECO2	win_rate.Leela	elodiff	SE	LOS	LOS2	LOS3
1	E	0.50000	0.000	NaN	1.00000	NaN	0.00000
2	E	0.50000	0.000	NaN	1.00000	NaN	0.00000
3	B	0.50000	0.000	NaN	1.00000	NaN	0.00000
4	B	0.50000	0.000	NaN	1.00000	NaN	0.00000
5	A	0.50000	0.000	798.096	1.00000	NaN	0.00000
6	A	0.50000	0.000	472.706	1.00000	NaN	0.00000
7	C	0.50000	0.000	381.844	1.00000	NaN	0.00000
8	C	0.50000	0.000	330.843	1.00000	NaN	0.00000
9	A	0.50000	0.000	296.575	1.00000	NaN	0.00000
10	A	0.55000	34.860	303.216	1.00000	0.84134	0.61258
11	C	0.54545	31.672	275.265	1.00000	0.84134	0.61446
12	C	0.50000	0.000	235.953	0.99865	0.50000	0.85195
13	A	0.50000	0.000	222.815	0.99942	0.50000	0.85305
14	A	0.50000	0.000	211.674	0.99977	0.50000	0.85397
15	B	0.50000	0.000	202.066	0.99991	0.50000	0.85475
16	B	0.53125	21.743	201.982	0.99617	0.71815	0.88967
17	E	0.52941	20.461	193.375	0.99770	0.71815	0.89039
18	E	0.55556	38.764	193.252	0.98778	0.84134	0.90667
19	B	0.55263	36.708	185.531	0.99123	0.84134	0.90735
20	B	0.55000	34.860	178.692	0.99379	0.84134	0.90795
21	D	0.54762	33.190	172.577	0.99567	0.84134	0.90848
22	D	0.54545	31.672	167.066	0.99702	0.84134	0.90896
23	C	0.54348	30.288	162.064	0.99798	0.84134	0.90939
24	C	0.56250	43.658	161.609	0.99180	0.91014	0.91930
25	E	0.56000	41.894	157.009	0.99379	0.91014	0.91971
26	E	0.57692	53.879	156.601	0.98487	0.94876	0.92647
27	B	0.57407	51.854	152.357	0.98778	0.94876	0.92687
28	B	0.57143	49.975	148.453	0.99018	0.94876	0.92724
29	A	0.56897	48.230	144.845	0.99217	0.94876	0.92757
30	A	0.56667	46.602	141.496	0.99379	0.94876	0.92788
31	C	0.56452	45.082	138.377	0.99511	0.94876	0.92817
32	C	0.56250	43.658	135.463	0.99617	0.94876	0.92843
33	E	0.56061	42.321	132.731	0.99702	0.94876	0.92868
34	E	0.55882	41.065	130.163	0.99770	0.94876	0.92891
35	B	0.54286	29.853	125.947	0.99379	0.87158	0.94693
36	B	0.55556	38.764	125.458	0.98778	0.92135	0.95074

We see that by game 94, when Leela breached the 50.5 mark, the ELO difference is about 26, but with large error bar. The LOS's show though that there is very high likelihood that Leela is indeed stronger. At the end of SuFi, the estimated ELO difference is about 24.

The problem with ELO estimates based on results of chess engine tournaments is that each opening has to be played in reverse colors by each engine. Also, there are families of ECO code openings. As such, the ELO differences might actually be biased. Also, the sample size of 100 is actually small, leading to the large error bars.

Instead, we can calculate the ELO differences by ECO family of openings. The estimates will have larger error bars because we now have smaller samples.

```
data2 <- data %>%
  group_by(ECO2) %>%
  mutate(ECO2.Score.Leela = cumsum(points.Leela)) %>%
  mutate(ECO2.Score.SF = cumsum(points.SF)) %>%
  mutate(ECO2.total = row_number()) %>%
  mutate(ECO2.draw_ratio = cumsum(points.Leela == points.SF)/ECO2.total) %>%
  mutate(ECO2.wins.Leela = cumsum(results.Leela=="Win")) %>%
  mutate(ECO2.losses.Leela = cumsum(results.Leela=="Loss")) %>%
  mutate(ECO2.wins.SF = cumsum(results.SF=="Win")) %>%
  mutate(ECO2.losses.SF = cumsum(results.SF=="Loss")) %>%
  mutate(ECO2.Draws = cumsum(results.Leela=="Draw")) %>%
  mutate(ECO2.win_rate.Leela = ECO2.Score.Leela/ECO2.total) %>%
  mutate(ECO2.elodiff = elo(ECO2.win_rate.Leela)) %>%
  mutate(ECO2.SE = elo(ECO2.win_rate.Leela + denom95(ECO2.win_rate.Leela, ECO2.total))-ECO2.elodiff) %>%
  mutate(ECO2.LOS = LOS(ECO2.total*(1-ECO2.draw_ratio), ECO2.total)) %>%
  mutate(ECO2.LOS2 = LOS2(wins.Leela, losses.Leela)) %>%
  mutate(ECO2.LOS3 = LOS3(wins.Leela, losses.Leela, Draws))
```

We can now see the estimated ELO differences at the last of game of each ECO group of openings.

```
data2 %>%
  slice(n()) %>% select(starts_with("ECO2")) %>%
  select(1:8) %>%
  flextable() %>% autofit()
```

ECO2	ECO2.Score.Leela	ECO2.Score.SF	ECO2.total	ECO2.draw_ratio	ECO2.wins.Leela
A	14.5	11.5	26	0.73077	5
B	12.5	11.5	24	0.87500	2
C	12.0	13.0	25	0.80000	2
D	3.5	2.5	6	0.83333	1
E	11.0	8.0	19	0.73684	4

```
data2 %>%
  slice(n()) %>% select(starts_with("ECO2")) %>%
  select(9:16) %>%
  flextable() %>% autofit()
```

ECO2	ECO2.losses.SF	ECO2.Draws	ECO2.win_rate.Leela	ECO2.elodiff	ECO2.SE	ECO2
A	5	19	0.55769	40.268	152.79	0.9
B	2	21	0.52083	14.485	153.91	0.9
C	2	20	0.48000	-13.905	144.75	0.9
D	1	5	0.58333	58.451	NaN	0.9
E	4	14	0.57895	55.321	193.24	0.9

Here it is very interesting to note that Leela actually performed relatively better in A and E openings. This is interesting because of the nature of the A and E openings. In particular, Jeroen said that E openings are too easy for the current top programs and he considered them very drawish.

We can instead use the `elo` package instead to calculate the ELO estimates. This package doesn't have a function for estimating LOS though. The `elomod` object here is adjusted using a varying K after each round.

```
library(elo)
initial <- c(3589, 3587)
names(initial) <- c("Leela", "SF")
elomod <- elo.run(score(points.Leela, points.SF)~White+Black + regress(ECO2, initial, 0.2) + k(20*log(a
summary(elomod)
```

```
##
## An object of class 'elo.run.regressed', containing information on 2 teams and 100 matches, with 5 reg
##
## Mean Square Error: 0.0506
## AUC: 0.9082
## Favored Teams vs. Actual Wins:
##         Actual
## Favored  0 0.5  1
##   TRUE   1  36 13
##   (tie)  0   0  0
##   FALSE  6  43  1
```

```
elodf <- as.data.frame(elomod)
elodf$elodiff <- abs(elodf$elo.A - elodf$elo.B)
elodf$actual_score <- na.omit(data$Score.Leela)
elodf <- elodf %>%
  mutate(exp_score = cumsum(1 / (1+10^(elodiff/400))))
elodf %>%
  mutate_if(is.numeric, function(x) round(x, 3)) %>%
  flextable() %>% autofit()
```

team.A	team.B	p.A	wins.A	update.A	update.B	elo.A	elo.B	elodiff	a
SF	Leela	0.497	0.5	0.000	0.000	3587.0	3589.0	2.000	
Leela	SF	0.503	0.5	0.000	0.000	3589.0	3587.0	2.000	
SF	Leela	0.497	0.5	0.000	0.000	3587.0	3589.0	2.000	
Leela	SF	0.503	0.5	0.000	0.000	3589.0	3587.0	2.000	
SF	Leela	0.497	0.5	0.000	0.000	3587.0	3589.0	2.000	
Leela	SF	0.503	0.5	0.000	0.000	3589.0	3587.0	2.000	
SF	Leela	0.497	0.5	0.000	0.000	3587.0	3589.0	2.000	
Leela	SF	0.503	0.5	0.000	0.000	3589.0	3587.0	2.000	
SF	Leela	0.497	0.5	0.000	0.000	3587.0	3589.0	2.000	
Leela	SF	0.503	1.0	6.892	-6.892	3595.9	3580.1	15.783	
SF	Leela	0.477	0.5	0.000	0.000	3580.1	3595.9	15.783	
Leela	SF	0.523	0.0	-7.246	7.246	3588.6	3587.4	1.291	
SF	Leela	0.498	0.5	0.000	0.000	3587.4	3588.6	1.291	
Leela	SF	0.502	0.5	0.000	0.000	3588.6	3587.4	1.291	
SF	Leela	0.498	0.5	0.000	0.000	3587.4	3588.6	1.291	
Leela	SF	0.502	1.0	6.906	-6.906	3595.6	3580.4	15.102	
SF	Leela	0.478	0.5	0.000	0.000	3580.4	3595.6	15.102	
Leela	SF	0.522	1.0	6.630	-6.630	3602.2	3573.8	28.363	
SF	Leela	0.459	0.5	0.000	0.000	3573.8	3602.2	28.363	
Leela	SF	0.541	0.5	0.000	0.000	3602.2	3573.8	28.363	
SF	Leela	0.459	0.5	0.000	0.000	3573.8	3602.2	28.363	
Leela	SF	0.541	0.5	0.000	0.000	3602.2	3573.8	28.363	
SF	Leela	0.459	0.5	0.000	0.000	3573.8	3602.2	28.363	
Leela	SF	0.541	1.0	6.367	-6.367	3608.5	3567.5	41.097	
SF	Leela	0.441	0.5	0.000	0.000	3567.5	3608.5	41.097	
Leela	SF	0.559	1.0	6.115	-6.115	3614.7	3561.3	53.328	
SF	Leela	0.424	0.5	0.000	0.000	3561.3	3614.7	53.328	
Leela	SF	0.576	0.5	0.000	0.000	3614.7	3561.3	53.328	
SF	Leela	0.424	0.5	0.000	0.000	3561.3	3614.7	53.328	
Leela	SF	0.576	0.5	0.000	0.000	3614.7	3561.3	53.328	
SF	Leela	0.424	0.5	0.000	0.000	3561.3	3614.7	53.328	
Leela	SF	0.576	0.5	0.000	0.000	3614.7	3561.3	53.328	
SF	Leela	0.424	0.5	0.000	0.000	3561.3	3614.7	53.328	
Leela	SF	0.576	0.5	0.000	0.000	3614.7	3561.3	53.328	
SF	Leela	0.424	0.0	-5.876	5.876	3555.5	3620.5	65.079	
Leela	SF	0.593	1.0	5.648	-5.648	3626.2	3549.8	76.375	

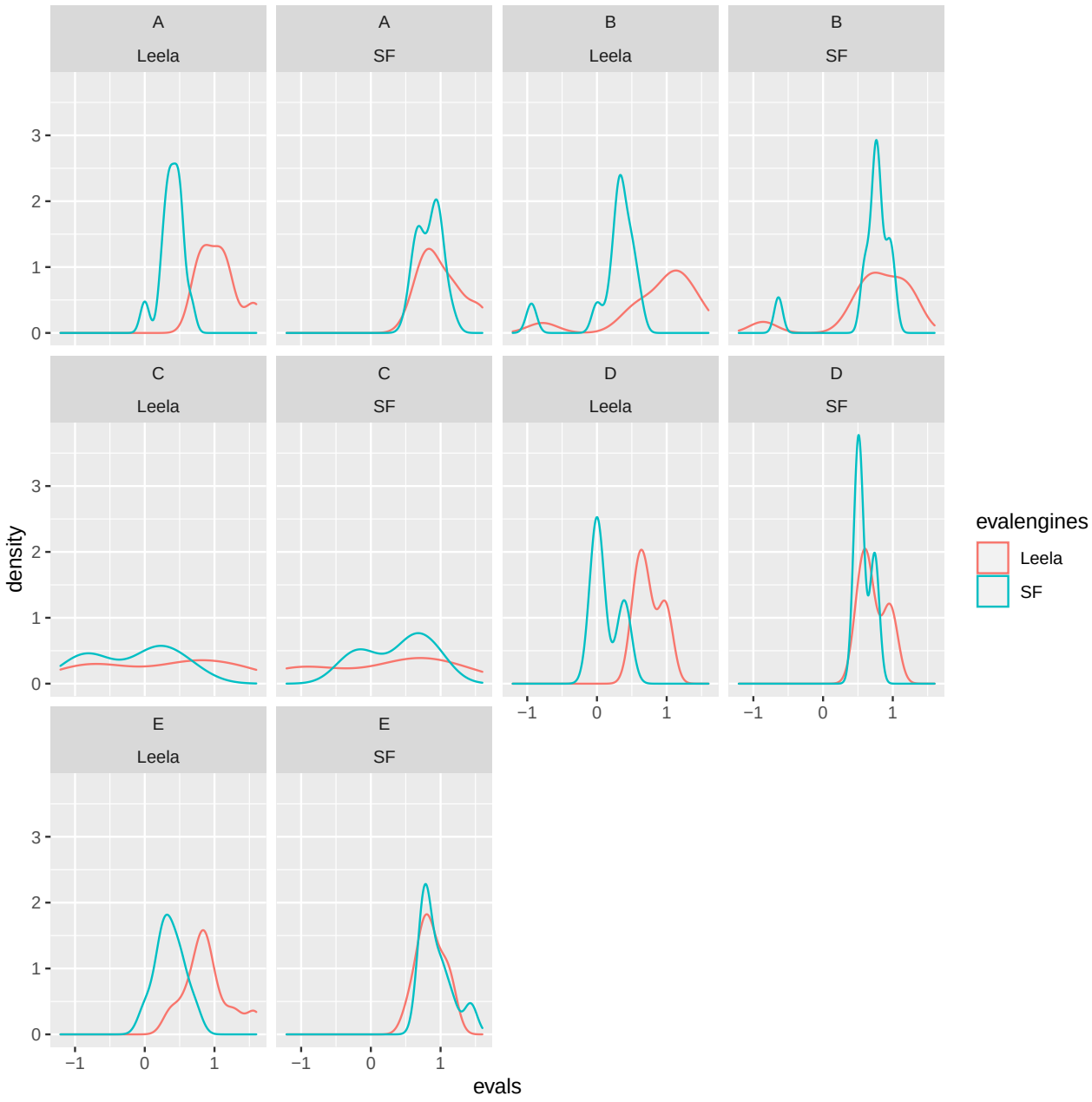
Let us now investigate the evals.

```
data_df <- data %>% gather(color, engine, White:Black)
```

```
data_dfwhite <- data_df %>%
  filter(color == "White") %>%
  group_by(engine) %>%
  gather(evalengines, evals, Leela.openeval:SF.openeval) %>%
  mutate(evalengines = str_remove(evalengines, ".openeval")) %>%
  group_by(ECO2, evalengines) %>%
  summarize(mean = round(mean(evals),3), sd = round(sd(evals),3))
data_dfwhite %>% flextable() %>% autofit()
```

ECO2	evalengines	mean	sd
A	Leela	1.033	0.288
A	SF	0.614	0.282
B	Leela	0.807	0.586
B	SF	0.456	0.464
C	Leela	0.192	0.917
C	SF	0.106	0.605
D	Leela	0.735	0.191
D	SF	0.358	0.300
E	Leela	0.878	0.274
E	SF	0.633	0.366

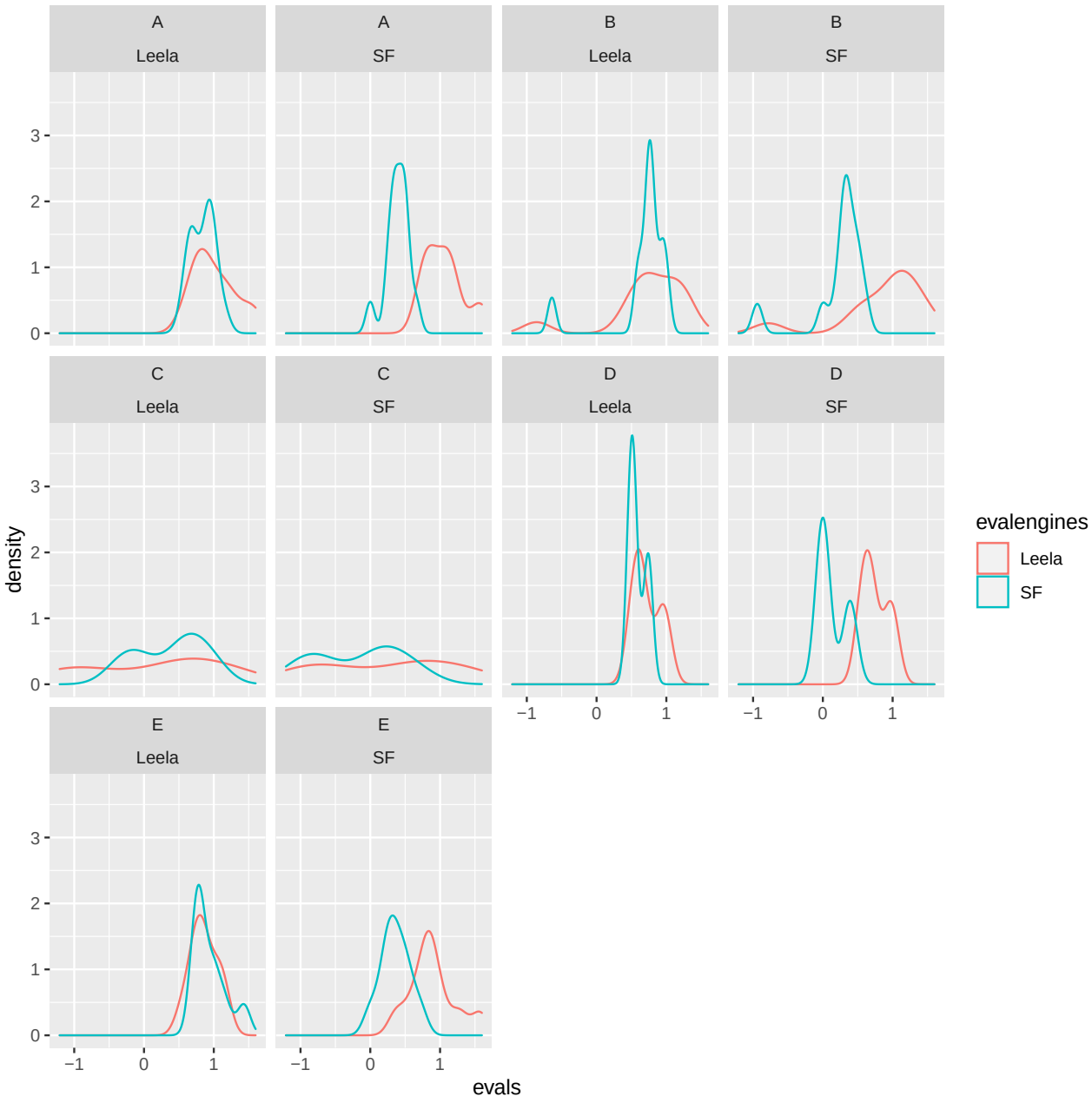
```
data_df %>%
  filter(color == "White") %>%
  group_by(engine) %>%
  gather(evalengines, evals, Leela.openeval:SF.openeval) %>%
  mutate(evalengines = str_remove(evalengines, ".openeval")) %>%
  ggplot(aes(evals, color = evalengines)) +
  geom_density() +
  facet_wrap(~ECO2+engine)
```




```
data_dfblack <- data_df %>%
  filter(color == "Black") %>%
  group_by(engine) %>%
  gather(evalengines, evals, Leela.openeval:SF.openeval) %>%
  mutate(evalengines = str_remove(evalengines, ".openeval")) %>%
  group_by(ECO2, evalengines) %>%
  summarize(mean = round(mean(evals),3), sd = round(sd(evals),3))
data_dfblack %>% flextable() %>% autofit()
```

ECO2	evalengines	mean	sd
A	Leela	1.033	0.288
A	SF	0.614	0.282
B	Leela	0.807	0.586
B	SF	0.456	0.464
C	Leela	0.192	0.917
C	SF	0.106	0.605
D	Leela	0.735	0.191
D	SF	0.358	0.300
E	Leela	0.878	0.274
E	SF	0.633	0.366

```
data_df %>%
  filter(color == "Black") %>%
  group_by(engine) %>%
  gather(evalengines, evals, Leela.openeval:SF.openeval) %>%
  mutate(evalengines = str_remove(evalengines, ".openeval")) %>%
  ggplot(aes(evals, color = evalengines)) +
  geom_density() +
  facet_wrap(~ECO2+engine)
```



We can see that Leela's opening evals are generally more optimistic than that of Stockfish, which can be attributed partly to SF's contempt. But a closer inspection of Leela's evals, we see that they are consistent even if playing as different colors. Leela also tends to win in openings where its opening evals are visibly more optimistic than that of Stockfish, signifying that Leela has better opening evaluation.

This has been a very exciting SuFi. I had a lot of fun engaging in many interesting and lively discussions in chat, although oftentimes the chat can quickly turn cancerous.

To end this post, I would like to congratulate the Leela devs and community for winning their first ever SuFi title. Kudos also to the SF team for continuing to improve a chess monster. I hope that Leela and SF continue to expose each other's weaknesses, and get better as a result. Exciting times for the chess engine fans!