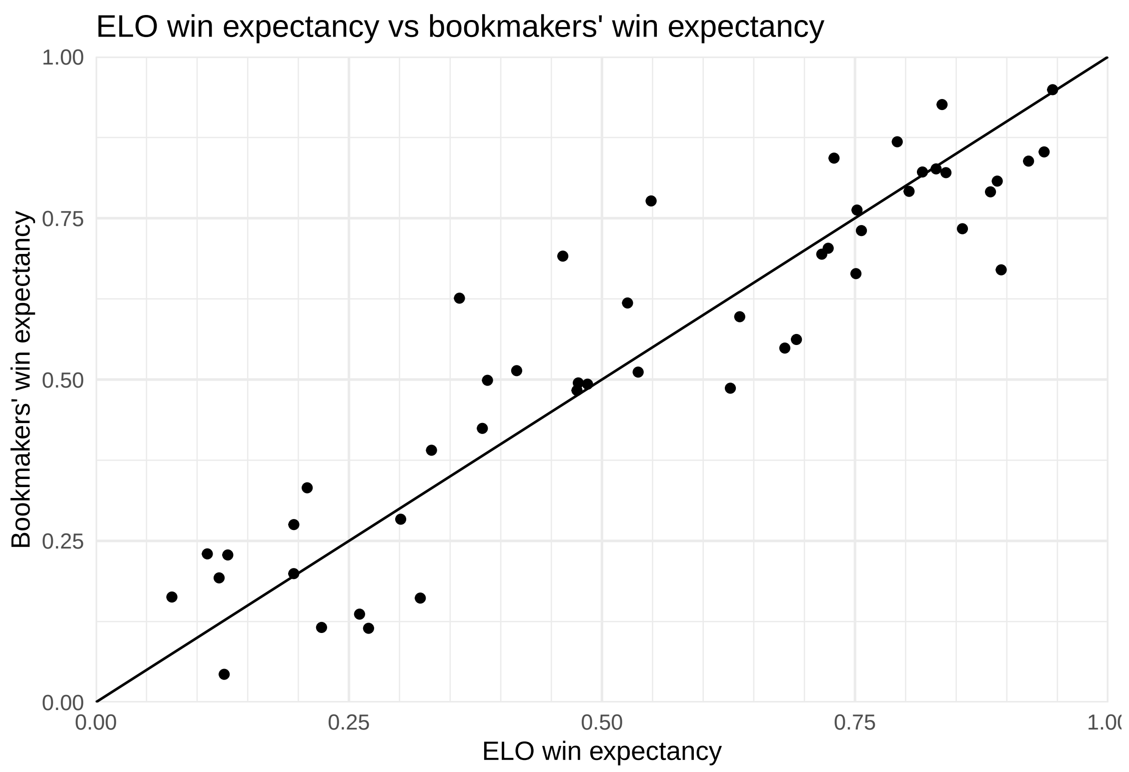
Quick report from modeling WC 2022 ELO simulation

@kseligga

Why ELO?

It's kinda the easiest, that's it.
In this simulation, we just calculate win expectancy [WE] based on teams' ELO rantings, and based on that we provide probability function for scoring n number of goals. Then all we need to do is to draw two pseudorandom numbers, that'll tell us how many times each team have scored in this particular simulated match (more explanation on this on page 2). When the result is produced, we update teams' ELO rating accordingly.

Just to be sure it's not that bad, let's compare it to some bookmaker's predictions. First of all, ELO WE vs bookmaker's WE, for 48 WC group stage matches:



yea, it could be better. Let's calculate correlation:

```r
cor(df$avg_elo_we, bookmaker_we$we, method='pearson')
```

```
## [1] 0.9193285
```

If you by chance create a better model for WE than ELO, let me know. Maybe I'll try to do it someday. For now, we just gonna assume it's good enough.

Scoring process

So now we have just win expectations, but how to draw results from them? We should be able to draw the draw, but all we have here is just *win* expectancy. So now, we start the scoring.

So basically, I've stolen the data for relation between win expectancy and scoring n goals probability from the bookmaker. Then I created polynomial trend lines in Excel for this data, for each $n \in \{0, 1, 2, 3, 4, 5\}$ (big flaw there: I assumed we can skip more than 5 goals, as its so uncommon. But its not, it'd be very cool to implement these in next tournament). The resulting functions are as follows:

$$g0(x) = 1.6163x^4 - 4.1663x^3 + 3.7334x^2 - 1.8312x + 0.713$$

$$g1(x) = -1.6687x^4 + 3.295x^3 - 2.6629x^2 + 0.9349x + 0.2587$$
$$g2(x) = -1.0356x^4 + 2.1526x^3 - 1.6883x^2 + 0.8078x + 0.0294$$
$$g3(x) = 0.1685x^4 - 0.0619x^3 - 0.0036x^2 + 0.1632x$$
$$g4(x) = 0.8176x^4 - 1.3061x^3 + 0.8288x^2 - 0.1631x + 0.0086$$
$$g5(x) = 0.4762x^4 - 0.7185x^3 + 0.3817x^2 - 0.0783x + 0.0048$$

where $gn(x)$ is the function that outputs probability for scoring n goals of team with WE$= x$.

The idea here is that we'd then select random value between 0 and 1, and then check if this value is less than g0(x). If so, the team with WE=x scored 0 goals in this match. If not, we compare the random value to g0(x)+g1(x), and go with 1 goal for interested team if it's less, and so on, increasing the sum we compare our random value to.

But given functions do not sum up to 1 for every $x \in (0, 1)$, so there is an error corrector, definied simply as:

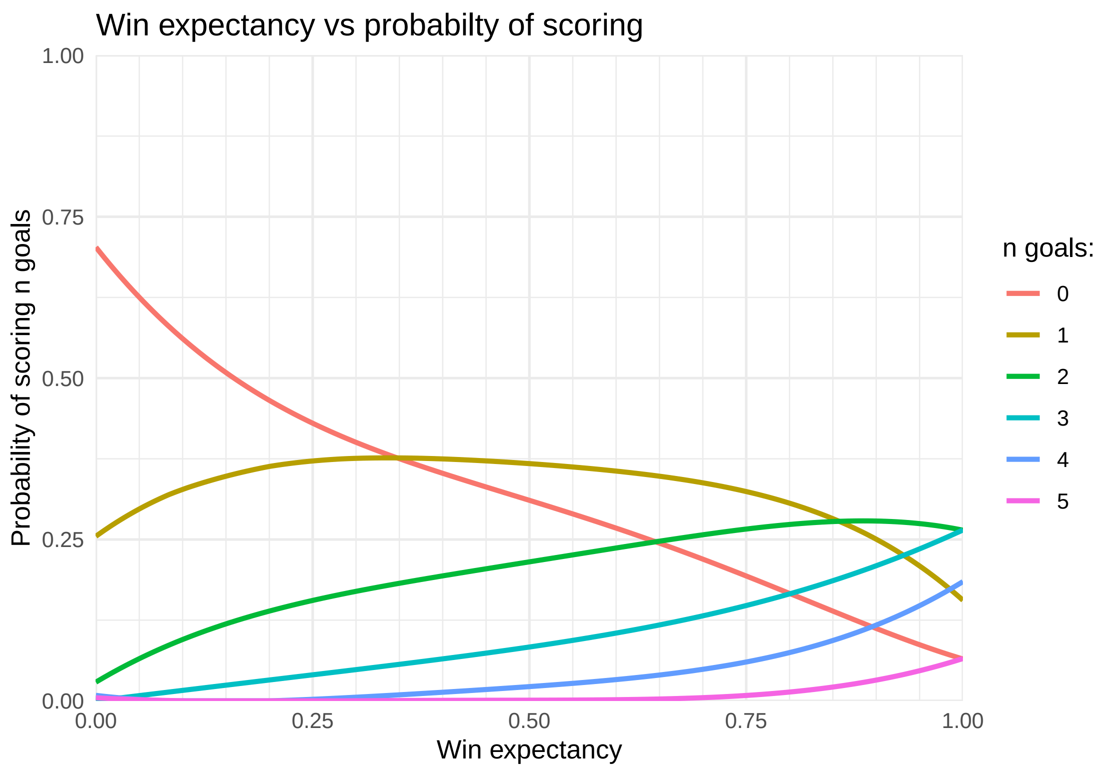
$$ErrorCorrector(x) = g0(x) + g1(x) + g2(x) + g3(x) + g4(x) + g5(x)$$

Dividing each gn(x) function by error corrector made sum of them all for given x very close to 1, but approximation error still exists. When the sum is a bit more than 1 we don't have a problem, as for each random value we get our number of goals. When the sum is less than 1 and we happen to draw value bigger than the sum of each gn(x), we just go with 1 goal for interested team, as it's most common for most matches.

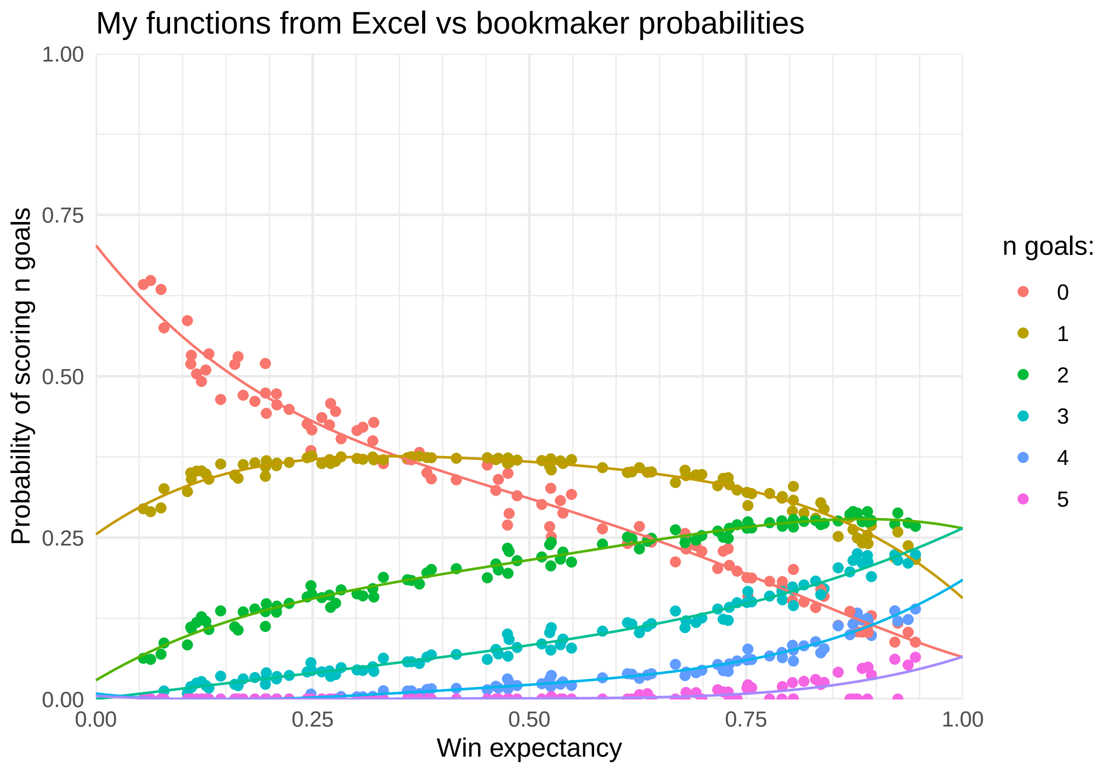
Now we just need to make sure the calculated probability isn't less than 0 (by default it is for small x in g4(x) and g5(x)), and we get these bad boys:

```r
error_corrector <- function(x) {
  0.3345 * x^4 - 0.7207 * x^3 + 0.5316 * x^2 - 0.1541 * x + 1.0144}
g0 <- function(x) {
  (1.6163*x^4 - 4.1663*x^3 + 3.7334*x^2 - 1.8312*x + 0.713)/error_corrector(x)}
g1 <- function(x) {
  (-1.6687 *x^4 + 3.295 * x^3 - 2.6629 * x^2 + 0.9349 * x + 0.2587)/error_corrector(x)}
g2 <- function(x) {
  (-1.0356 * x^4 + 2.1526 * x^3 - 1.6883 * x^2 + 0.8078 * x + 0.0294)/error_corrector(x)}
g3 <- function(x) {
  (0.1685 * x^4 - 0.0619 * x^3 - 0.0036 * x^2 + 0.1632 * x)/error_corrector(x)}
g4 <- function(x) {
  res<-((0.8176 * x^4 - 1.3061 * x^3 + 0.8288 * x^2 - 0.1631 * x + 0.0086)/error_corrector(x))
  ifelse(res<0, 0, res)}
g5 <- function(x) {
  res<-((0.4762 * x^4 - 0.7185 * x^3 + 0.3817 * x^2 - 0.0783 * x + 0.0048)/error_corrector(x))
  ifelse(res<0, 0, res)}
```

A lot of numbers, so lets just plot them:



But are they accurate? Let's compare them with bookmaker's probabilities on one plot:



Looks cool, but make it formal with mr. Pearson:

```
cor(bookmaker$n_goals_probability, bookmaker$my_n_goals_prob, method = "pearson")
```

```
## [1] 0.9964863
```

thats an overall correlation. we can also caltulate it for each of 6 functions:

```
for (n in 0:5) {
  res<-cor(bookmaker[bookmaker$n_goals==n,"n_goals_probability"],
      bookmaker[bookmaker$n_goals==n,"my_n_goals_prob"])
  print(res)
}
```

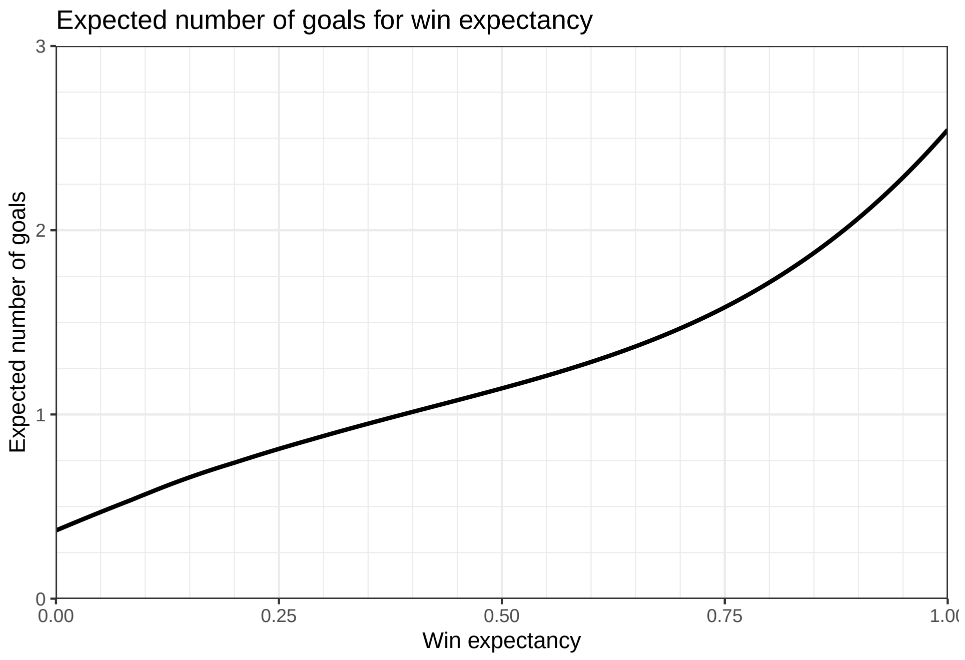
```
## [1] 0.9882171
## [1] 0.9714769
## [1] 0.9869432
## [1] 0.9886944
## [1] 0.9851243
## [1] 0.7561614
```

Sixth one (g5(x)) looks poor. But here is why: when I've scraped data from bookmaker, I've counted betting odds 250.0 (the biggest it could be there) as 0% probability. Of course, it occurred a lot in this case. So now without it?

```
tmp<-filter(bookmaker, n_goals==5, n_goals_probability>0) %>%
  select(n_goals_probability, my_n_goals_prob)
cor(tmp$n_goals_probability, tmp$my_n_goals_prob)
```

```
## [1] 0.9700043
```

I think we safe to say these functions work.
So we can show something with them, like trendy expected goals coefficient, so here it is:



As it should be, it's an increasing function.

What to do better

First of all, ELO model is not perfect. I think we all know bookmaker's predictions are far more accurate, considering playstyle of each team, their objectives and situation in tournament. And then ELO rating is just one little number, based just on previous results. As I mentioned at the end of page 1, Im open minded for trying to find a better fit, reflecting reality more accurately.

They're two easier things, that should've been done here, but aren't:
1. A probability of scoring more than 5 goals in a match. It's not common irl, but it should be considered here, where we simulate millions of matches.
2. Extra-time. Here, when we draw a draw in knockout stage, we go straight to penalties. There should be another function responsible for scoring in extra-time probability, but I thought it'd be too much work.