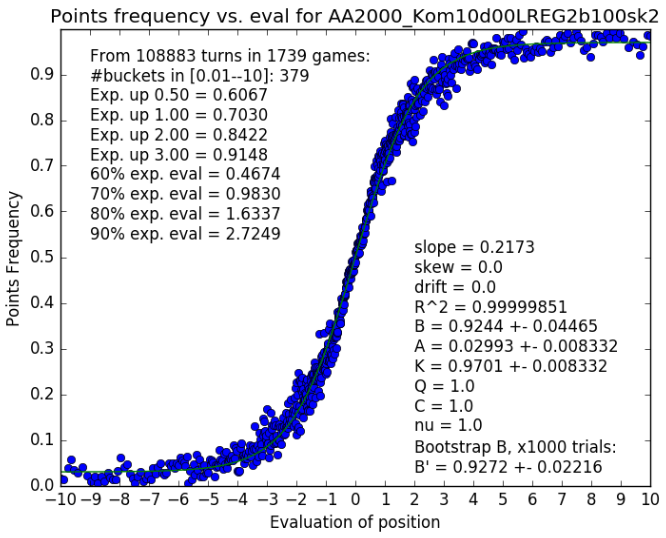
Bootstrap Interlude

[This was removed from the post in order to conserve length.] You are welcome to skip to the next section if you're already satisfied that the above procedures and reported results are sound.

The statistical Bootstrap technique provides a way to estimate error bars and do quantiling when theoretical error bars are unavailable. It also helps vet procedures when they are available. Originally I weighted buckets only by their sizes n, passing $1/\sqrt{n}$ as the `sigma` argument to the Python `scipy` package's `curve_fit` function. With the `absolute_sigma` option left false, the difference between this and $0.5/\sqrt{n}$ from the sigma of n coinflips, or any other constant factor applied to all weights, is immaterial. I got results typified by the following "sk2" figure:



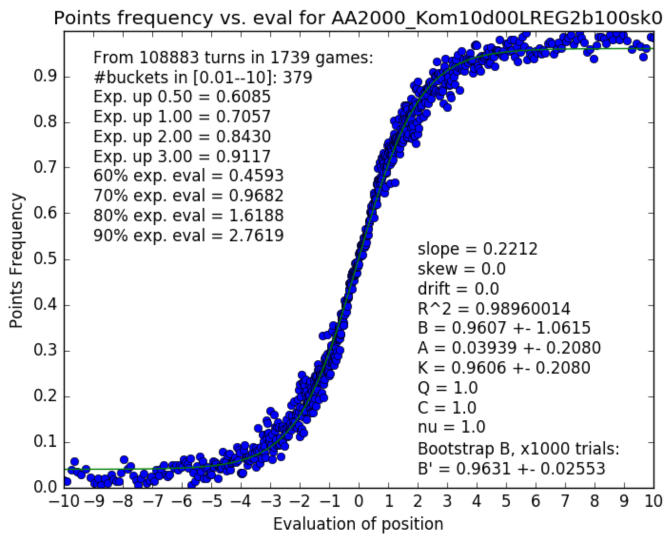
The fit is just as fine and the answers are close, but there is now over a $2x$ discrepancy between the theoretical and bootstrap error bars. This persisted across all rating levels and mystified me during my graduate seminar last spring—in which students verified cases of bootstrapping on other chess-related *linear* regressions.

I was puzzled until I realized that the `sigma` arguments should pertain to y not x values. Consider the bucket for 0.98 which gives $y = 0.7$. The sigma for a *biased* coinflip model is $\sqrt{y(1-y)/n}$. Including the y is *not* a constant factor across all buckets. Taking the actual frequency r of draws in the bucket into account, as well as the frequencies p of wins and q of losses, the numerator I use is

$$E[(R-y)^2] = py^2 + q(1-y)^2 + r(y-\frac{1}{2})^2 = pq + \frac{1}{4}(r-r^2),$$

where R is a game result drawn from (p,q,r). Intuititvely this means having higher confidence in the y values of buckets where one side is close to winning, thus upweighting the tails. In case a bucket has close to 100 wins, I bound the numerator away from zewro by using a fixed estimate for A. Doing so "magically" brought the reported error bars into line with the bootstrap values. This also largely carries over for setting `absolute_sigma=True` when a constant 0.5 is used to correct for redundant data under the symmetry option, and to weighting moves also according to the lengths of the games they came from.

The point is that the bootstrap process as applied here is completely *natural* and oblivious to such model choices. It merely re-samples the N positions with replacement and runs the same process in 1,000 (or more) trials. The bootstrap for size-only weighting did size-only weighting; that for (p, q, r)-weighting used (p, q, r) for the different buckets it obtained in any trial—in which some positions might be absent and others repeated several times (*without* any redundant-data correction). Yet it distinguishes between the weighting policies. The kicker is shown by the following figure under "sk0" meaning unit weight per bucket:



The reported error bars are larger than B and A themselves, whereas the bootstrap bars under unit weights stay in a similar range to above. The answer for B has shifted more than those error bars. All of this augments the conclusion of the part-I post that getting a fantastic fit does not absolve one from vigilance about procedure.

The upshot here, however, is that the "sk4" procedure and its answers and error bars are all confirmed. The fit is perfect to the 8th decimal place. So the rest should be a "turkey shoot" as we say. But.