

# Soft Optimal Stop For 99% Guarantee

By Guillaume Lathoud, August 2025 - [glat@glat.info](mailto:glat@glat.info) - Github website  
PDF

The present document and all accompanying files are covered by the Boost License, as described in `./LICENSE`

Starting point: [https://en.wikipedia.org/wiki/Secretary\\_problem](https://en.wikipedia.org/wiki/Secretary_problem)

The Optimal Stop Problem & its solution (37%...) sound nice, but 37% also means that one has a 37% chance to end up with the fallback solution - i.e. to pick the last candidate: say one already saw the best possible candidate before the 37% cutoff, then one mechanically ends up picking the last candidate, which gives a pretty random result. The output can be pretty bad, so the reliability is not guaranteed, at least not over a single pass.

And in life, there are quite a few single pass situations.

So instead, let us try to solve a slightly different problem: guarantee with 99% chance that we'll pick a "pretty good" candidate (not targetting the best one).

So we need a strategy to maximize the worst case, i.e. maximize the 1% lowest percentile across the results of many simulations.

Proposed strategy: very similar to the Optimal Stop one, but a bit softer:

- look at the first `N_STOP` candidates (e.g. cutoff 37%, or any other percentage of the whole number of candidates)
  - pick none of them
- determine `threshold_score := soft_factor * best_score_of_N_STOP`
  - example `soft_factor`: 80%
- now start looking at the rest candidates

- pick the first one with score  $>$  threshold\_score
- else pick the last one

So the differences with the Optimal Stop are:

- in the evaluation: for a given threshold, we repeat a simulation many times and look at the score of the lowest 1% percentile (instead of "percentage that picked the true best one").
- in the solution: introduced a soft\_factor

One possible implementation: uniform use case

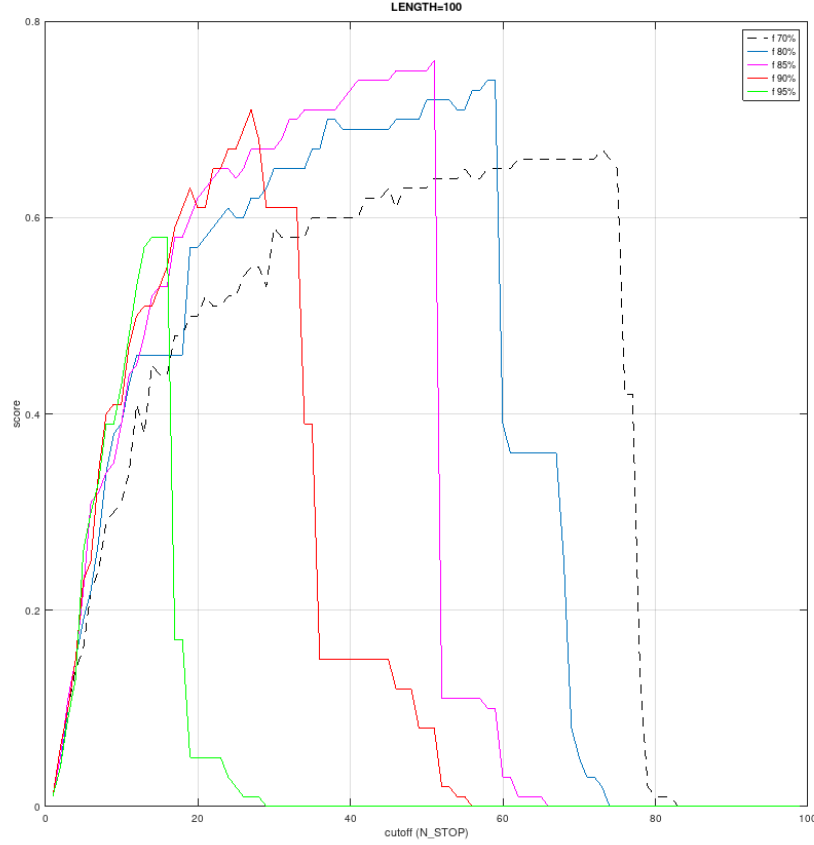
We don't know anything about the target market, so let's assume that the scores of the candidates are uniformly distributed, from worse (0.0) to best (1.0).

For a relatively total small number of candidates LENGTH=100 (for many scenarios, of a realistic order of magnitude), and various soft\_factor, here are the corresponding implementations:

- soft\_factor=70%
- soft\_factor=80% (my favorite)
- soft\_factor=85%
- soft\_factor=90%
- soft\_factor=95%

Figure: score of lowest 1% percentile for various soft\_factor values and various cutoff values (for/with Octave):

- octave code to produce the figure
- figure (click here to open a bigger version):



That figure shows the score of the lowest 1% percentile. Notice in all cases the break off when increasing too much the cutoff  $N\_STOP$ .

My favorite would be  $soft\_factor=80\%$  and cutoff threshold around  $40/100$ , which gives a lowest 1% percentile with a score of 0.68.

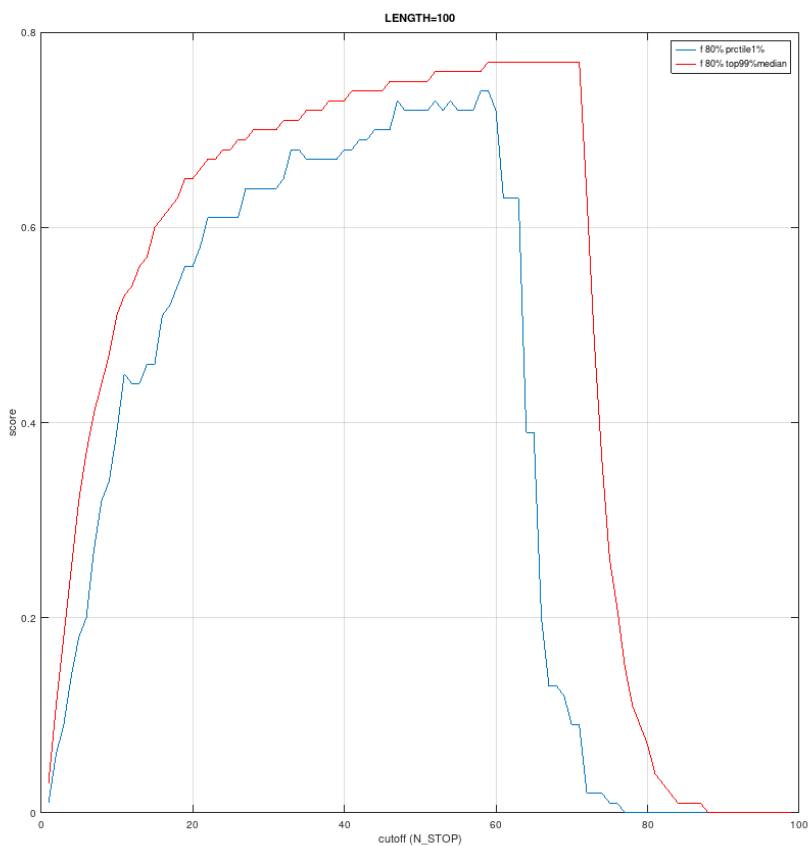
Tradeoff: When accepting the 1% risk, that result is a pretty good security, and most likely in practice we'll end up with a better pick. For  $soft\_factor=80\%$  and cutoff threshold around  $40/100$ :

- score of lowest 1% percentile: around 0.68
- median score of top 99% percentiles: around 0.72

Values around 0.72 can be judged as "pretty good", which was our objective.

Figure: score of the lowest 1% percentile, and median score of the top 99% percentiles, for soft\_factor=80% and various cutoff values (for/with Octave):

- Octave code to produce the figure
- figure (click here to open a bigger version):



Note: Generally, changing the order of magnitude of LENGTH can possibly change quite a bit the shape of the results.

### Conclusion

By **not** targetting the best candidate, but rather a "pretty good" candidate, we built a strategy that guarantees 99% success.