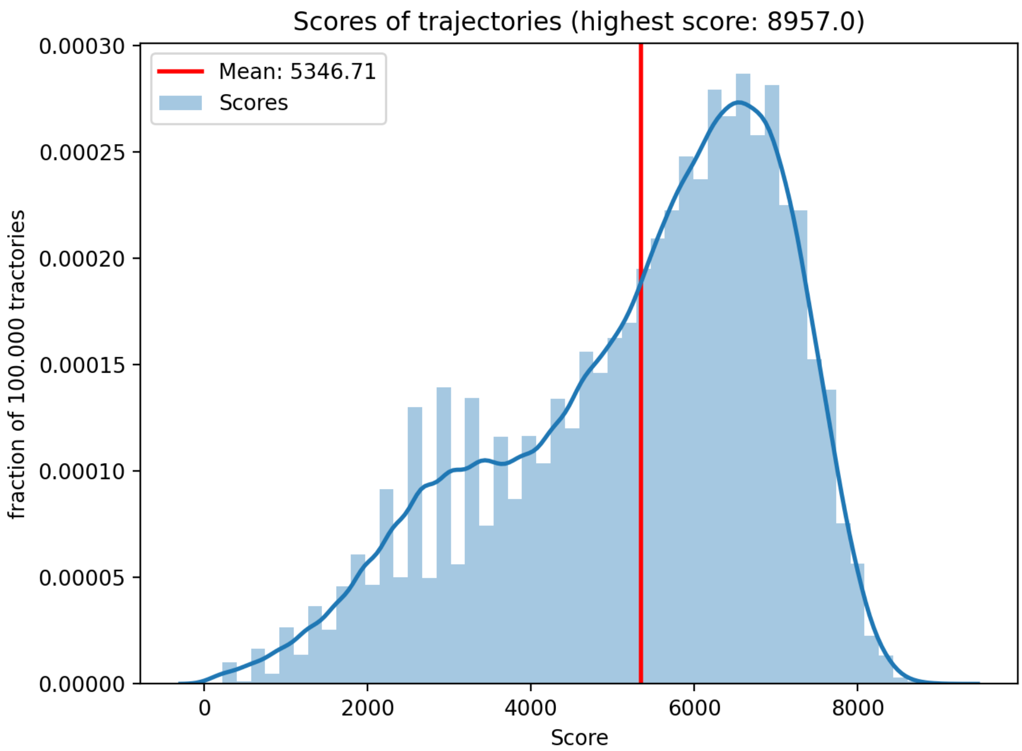
Baseline

When searching for a good solution to the RailNL case, one must ask itself a few things; what kind of solutions are feasible? What contributes to a good solution? Or how can one know that a solution is optimal? To investigate these questions, it is useful to generate some solutions randomly to see what sort solutions come up.

To start off, the case was analyzed by creating 100.000 random solutions. The assignment was: find a combination of at most seven trajectories that covers all connections in North- and South-Holland, where every trajectory takes up at most 2 hours. The second assignment was to choose the trajectories in such a way that the following function (which defines the quality of the solution) is maximized: k = p*10.000 – (100*T - Min), p stands for percentage of connections used, T is the number of trajectories and Min is the total time spend in all trajectories.

Graph A summarizes the results of the random track generator:



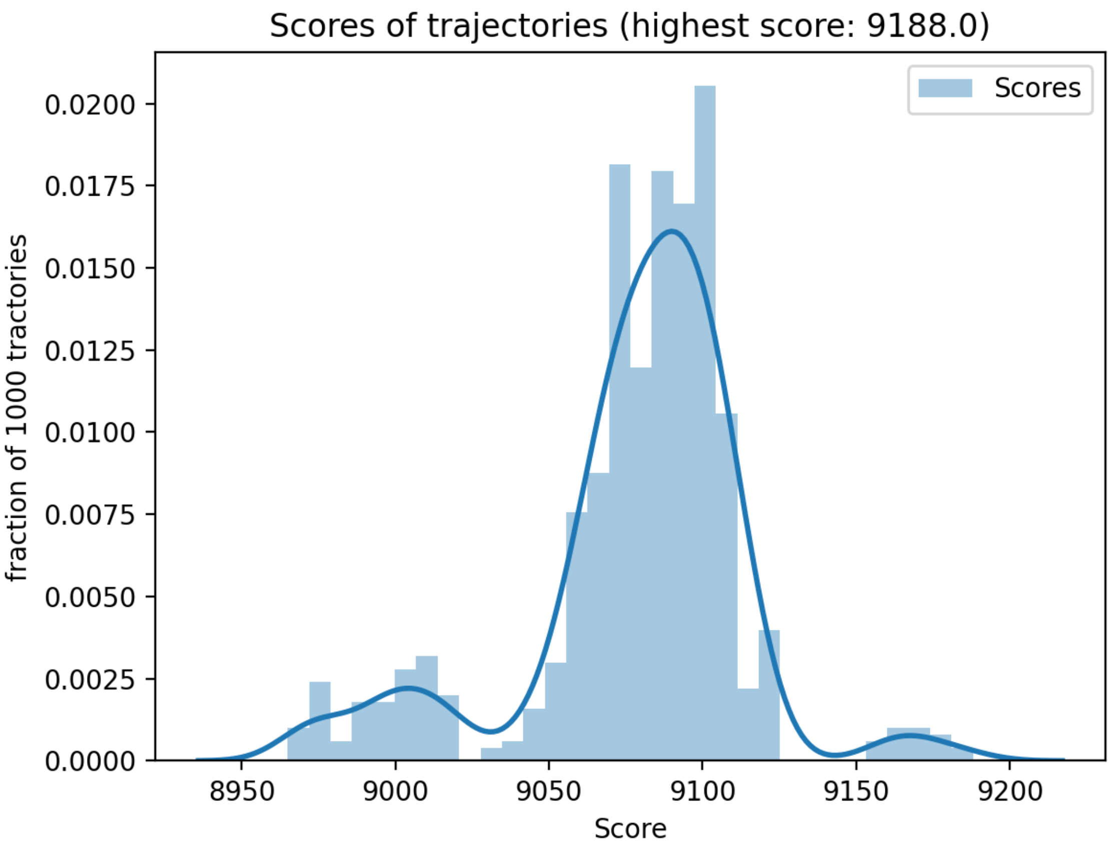
Before analyzing this graph, first some things to keep in mind. This is not a uniform sample from the state space, some constraints have been applied to the random trajectory generator. The first constraint is that a trajectory can at most take up 2 hours. The second constraint is that in the solution there are at most 7 trajectories. We constrained the random trajectory generator, because other solutions are not feasible for the problem stated in the case and are not worthwhile investigating.

First notice that the average of the simulation lies about 5300 points, which is about half the score one could possibly obtain. Secondly, all "low" scores (<5000) are mostly solutions with 1 or 2 trajectories. This is due to the fact that with 1 or 2 trajectories of at most 2 hours only very little connections are used, which is the largest contributor to amount of points scored. Thirdly, observed was that almost all high scores (>8000) seemed to have more than 4 trajectories. So what amount of trajectories is optimal? Looking at the objective function, it can be seen that the number of trajectories should be minimal (every extra trajectory yields -100 points) while using all connections (using a connection yields an increase in the score of (1/28)*10.000=357.14).

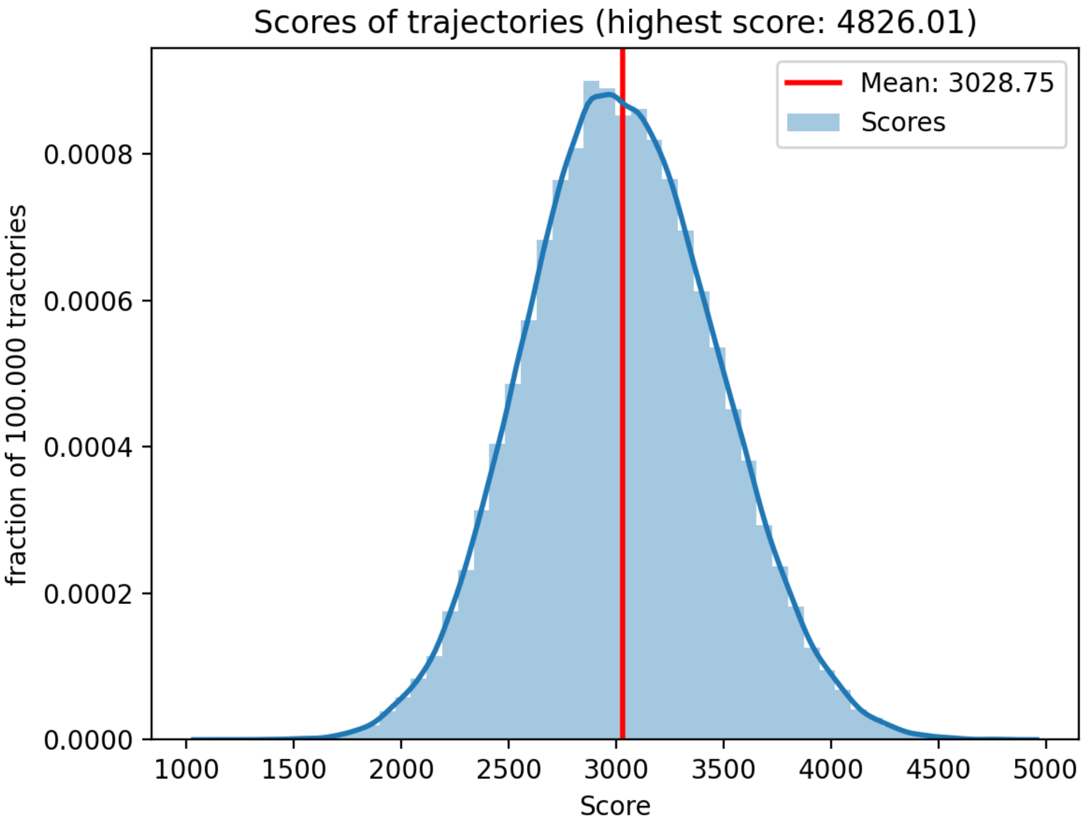
Thereby, the total time of all connections is 381 min, meaning that at least 4 trajectories of 2 hours are necessary if one intends to use all connections. All this combined yields a maximum score of 10.000-100*4-381=9219 points (all connections used, no connections used twice). The random simulation however, almost never surpasses the 9000 points, therefore other algorithms should and can do better (one of our algorithms already reached a score of 9180).

First algorithm

In the search for a better solution to the case, a new algorithm was put in action. The idea behind the algorithm was to start with a random solution containing one trajectory with a length of at most 2 or 3 hours depending on the region chosen (Holland/National). Subsequently, a new trajectory from the list of all possible trajectories (created in advance) is added to this solution, if and only if the objective function increases. This process is therefore probabilistic and not deterministic, because the starting solution differs every time the algorithm runs.
Running this “hill climber” algorithm 1000 times for Holland led to the following results:



The first thing to notice is that the maximum score of 9188 is close to the best possible solution, considering that this “best” solution is not even known to be existent. It also finds a better solution than the random trajectory generator. The second thing to notice is that the scores vary much less than in the random trajectory generator; scores vary from 8950 to 9200 approximately, whereas the random solution varies from 0 to 9000. Therefore, the standard deviation is much lower than for the random trajectory generator.
The national solution is computational much heavier so running it 1000 times is not possible like in the graph above for Holland. By reasoning the optimal solution for the national problem is around 7500 points, but due to the computational complexity the best solution found is much further off. The algorithm managed to score 7100 points, which is about 2000 points better than the best solution found by the random trajectory generator (see graph on next page).



In the coming days this score might still increase, since a better computer can run the algorithm more often. Also, some tweaks to the algorithm might help increase the score as well.