1. Generated data
2. Backtracking implement
3. [{'size': 8, 'time': 162629700, 'val': 0, 'length': 6, 'list': [66, -16, 16, -58, 33, -41]},

{'size': 10, 'time': 16159715000, 'val': 0, 'length': 3, 'list': [44, 8, -52]}]

Tb10/Tb8=99.36509137014949

I time grows exponentially with te data sise

1. [{"size": 8, "time": 31250000, "times": 0.03125, "val": 5, "length": 5, "list": [15, 56, -58, 33, -41]}, {"size": 10, "time": 15625000, "times": 0.015625, "val": 3, "length": 5, "list": [82, -67, 43, -63, 8]}, {"size": 50, "time": 1062500000, "times": 1.0625, "val": 0, "length": 36, "list": [75, 48, -54, -39, -41, -76, 49, -78, 24, 70, 55, 45, -98, -51, 40, 8, 62, 21, 70, 50, -34, -71, -52, 28, -64, -33, 26, 44, 44, 13, -63, 45, -1, -83, 14, 7]}, {"size": 100, "time": 7843750000, "times": 7.84375, "val": 0, "length": 69, "list": [-68, -44, -73, 67, 67, -56, 3, 86, -74, -9, 66, -8, -35, -12, 65, -17, -69, 61, -90, -5, 15, -50, -49, -37, 61, 16, -26, 25, 58, 77, -49, 10, 36, -87, 3, 83, -5, 80, 22, 60, 38, -26, -5, 47, -45, 63, -29, -44, 57, -14, -46, -69, 7, -28, 21, 57, 47, -1, -50, -72, 31, -92, -34, -17, 78, 100, -30, -42, 0]}]

Worse solutions for smaller data size

1. Faster time

TNS-10/TNS-8= 0.5‬ (strange)

TB-8/TNS-8 = 5.2041504

TB-10/TNS-10= 1,034.22176

It’s feasible to use neighborhood search, the time for backtracking grows exponentially higher but the one for neighborhood search.

1. Neighborhood search is better for bigger datasets/solution spaces at the expense of finding a global solution for finding a local minimum
2. Result\_annealing.txt
3. Implemented tabu search and generated the vectors
4. Tabu search is faster than simulated annealing, but gives worse results.

Both tabu search and simulated annealing are faster than neighborhood search and obviously than backtracking, but give worse results.

1. Tabu search and simulated annealing are significantly better than backtracking and exponentially better for higher datasizes. For bigger datasets backtracking could be impractical to use.
2. The execution time for tabu search and simulated annealing is significantly lower than for neighborhood search, but the solutions found are worse. Executing the algorithms multiple times could provide better solutions (is based on probability).
3. The initial temperature gives the running time of the algorithm and the alfa parameter gives the speed at which is decreases, a higher initial temperature gives a higher change of escaping a local solution.
4. The bigger the tenure, the longer an element is passed and the execution time is longer, but the solution is also better (smaller error => absolute value of solution)
5. Simulated annealing and tabu search are almost a must when solving a problem with a large solution space, the execution time is small, but the solutions have a high degree of difference to the global optimum, but the algorithms could be run more times, using neighborhood or backtracking being almost impossible from a practical standpoint. In some cases achieving the global optimum is not necessary.

Neighborhood search could be used for medium size solution spaces where execution time still is not impractically large.

Backtracking could be used for small solution spaces, always giving the global optimum.

1. Implemented