Worksheet 2: Introduction to Neural Networks and Genetic Algorithms

Prof. Andreas Geyer-Schulz, Sebastian Bell

May 29, 2025

Deadline: Tuesday, June 24, 2025, 23:59 CET.

Submission: Via Illias.

Grading: We will have four exercise sheets (Topics: 2 GA, 2 NN) which contribute 50 points each to the exercise grade. The Bonus points are computed by:

$$5 \cdot \sum_{i=1}^{4} \frac{\text{Points on Exercise } i}{200}$$

Organization of Work: Students can work on the exercises alone or in pairs. If you work together, please register as a team for the Ilias submission (Only one team member has to submit the results). Use the additional material provided!

Report of Results of Exercise: The documented R-code and a written report as a pdf-document.

Exercise 1. The Knapsack Problem

The knapsack problem is a problem in combinatorial optimization. The solution approach is to fill a specific object with items of varying values, volumes, weights, etc. The optimal solution is a combination of items that add up to the greatest value without exceeding the predefined constraints. You can code this problem as a binary genome and solve it with the method of Lagrange.

Consider the following knapsack problem. You have a suitcase that can hold 250 pounds and a volume capacity of 250 cubic inches. Each element has a certain value, and your goal is to pick the best combination of items that maximize your suitcase's value while remaining below your suitcase's weight and volume limits. How should your suitcase be packed with the given list of 19 items?

values = [48, 30, 42, 36, 22, 43, 18, 24, 36, 29, 30, 25, 19, 41, 34, 32, 27, 24, 18]
weights = [10, 30, 12, 22, 12, 20, 9, 9, 18, 20, 25, 18, 7, 16, 24, 21, 21, 32, 9]
volume = [15, 20, 18, 20, 5, 12, 7, 7, 24, 30, 25, 20, 5, 25, 19, 24, 19, 14, 30]

Write the knapsack problem environment according the given constraints (15 points) and return the indexes of the selected items with the maximum value of the suitcase using a genetic algorithm (10 points).

Exercise 2. Travelling Salesman Problem

Solve the asymmetric Travelling Salesman Problem from the TSPlib data set ftv47.atsp.gz (from [1]) by following the steps below.

Create an instance of the asymmetric Travelling Salesman Problem by using the *newTSP_bin* constructor from the attached file `newTSP_bin.R` (see Ilias). You do not have to implement the problem environment yourself! (5 points)
Hint: The *read_TSPLIB* function from the `TSP` package may help with the data import

Solve the asymmetric Travelling Salesman Problem using a simple binary-coded genetic algorithm. Choose an appropriate value for the *genemap* parameter! (5 points)

Create another instance of the asymmetric Travelling Salesman Problem by using the *newTSP* constructor from the `xegaSelectGene` package and taking the required inputs into account (see documentation). Solve the asymmetric Travelling Salesman Problem using a permutation-coded genetic algorithm. Choose an appropriate value for the *genemap* parameter! (5 points)

Based on the previous step, solve the asymmetric Travelling Salesman Problem using different heuristics by adjusting the mutation parameter in the *xegaRun* function. Consult the `xega` documentation to identify the availalable options. Use a greedy and a best greedy method as well as the Lin–Kernighan [2] algorithm. (5 points)

Compare the results of all algorithms and heuristics. (5 points)

References

[1] Reinelt, G. Tsplib, 2013. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/index.html

[2] Lin, S., Kernighan, B. W., 1973, An Effective Heuristic Algorithm for the Traveling-Salesman Problem, Operations Research, 21, 2, 498 – 516