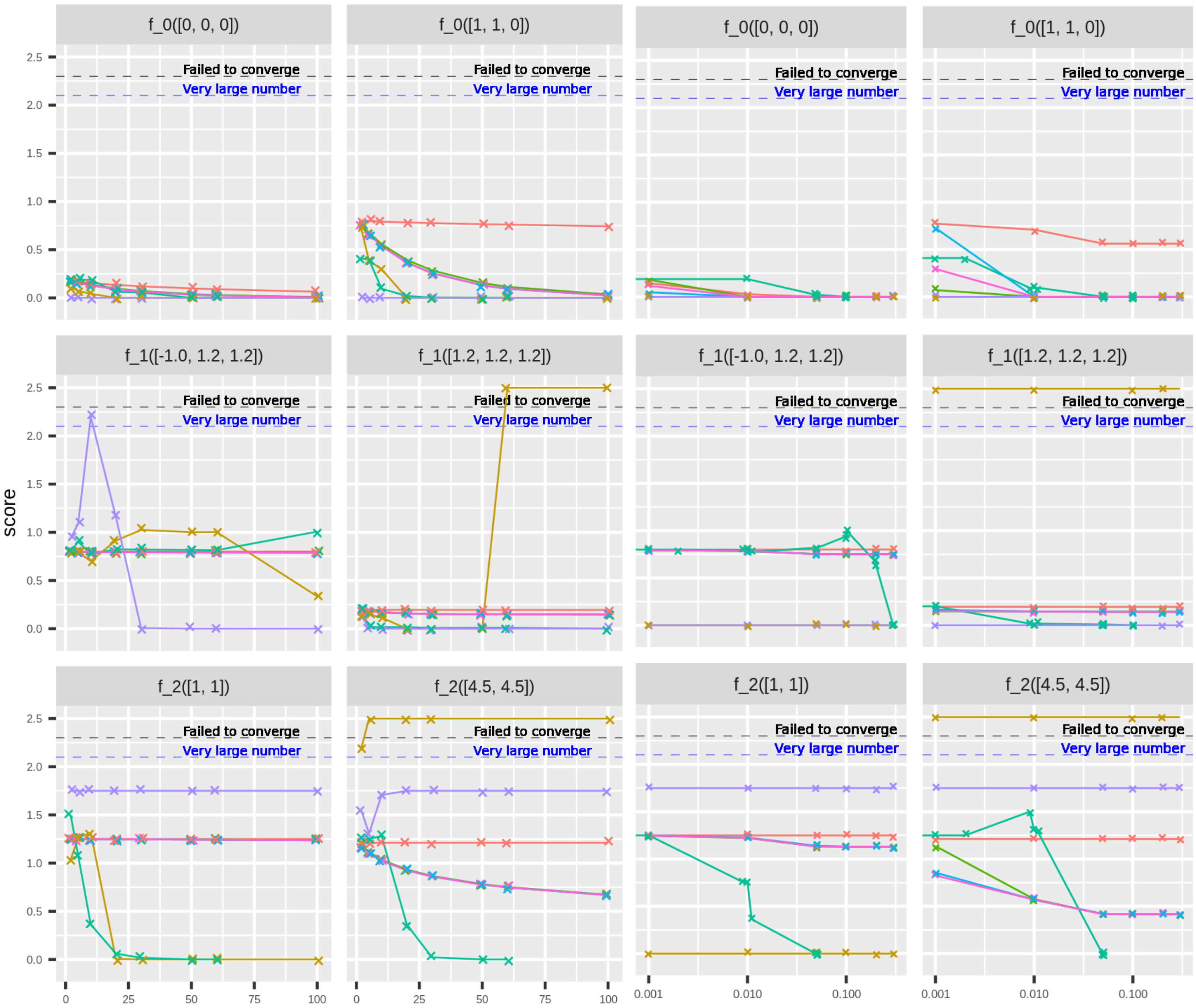
Mathematics 2: Nedler-Mead

David Ocepek

August 13, 2022

1 Nedler-mead method



(a) Convergence(iteration) (b) Convergence(time)

Figure 1: Comparison of convergence: Gradient methods vs Nelder-mead

We implemented standard n-dimensional Nelder-Mead method. Our method takes as input a single starting position and then constructs n-vertices, each vertex is cosntructed by shifting the starting position by the i-th basis vector.

Problem 2

For our comparison we used the same evaluation framework as in HW1. The function f_1, f_2, f_3 correspond to problem $5(a), 5(b), 5(c)$ respectively. From Fig. 1 we can see that Nelder-mead is the only method to always converge. While it is important to note that Nelder-mead is not theoretically guaranteed to converge, it appears that for most practical applications it has a higher likelihood of convergence than most gradient methods. From Fig. 1 we can also summarize that Nelder-mead on average has convergence speed similar to first order gradient methods. Of particular interest is example $f_1([-1, 1.2, 1.2])$ where Nelder-Mead fails to converge in 100 iterations but manages to converge in $100ms$. The spike on this particular subgraph most likely corresponds to a local minimum, which would indicate Nelder-Mead is less likely to get stuck in local minima than gradient methods.

2 Black box optimization

fun.	duration	num. iter.	min. fun. value	minimum
$f_{63160248,1}$	558 ± 35	94 ± 6	$0.84206 \pm 8e-11$	(0.842, 0.206, 0.6136)
$f_{63160248,2}$	1115 ± 541	194 ± 94	$0.84206 \pm 1e-10$	(0.6136, 0.206, 0.842)
$f_{63160248,3}$	568 ± 60	97 ± 13	$0.84206 \pm 2e-10$	(0.206, 0.6136, 0.842)

Table 1: Minima of black box function found by Nelder-Mead method.

Table 1 shows the results of five different runs of our Nelder-Mead method on all three black box functions. For each run we used an intial point generated from the uniform distribution $U(-5, 5)$. Given the low variance of our function minima we can conclude that our method converged to the minima.

2.1 Comparison with GD methods

In order to use gradient methods to find a minimum for a black box function we would need to numerically approximate the first and second order derivatives using the methods of finite differences. Eq. 1 is the equation for i-th first order partial derivative approximation at point x

$$\frac{\partial f}{\partial x_i} = \frac{f(x + hu_i) - f(x)}{h} \tag{1}$$

In Eq. 1 h represents a small number. It is important to note that each iteration of our gradient method requires at least n evaluation, where n is the number of dimensions, therefore we conclude that using gradient methods for black-box functions is inefficient. A perhaps better approach would use gaussian processes to fit some prior to our increasing number of points and determine our gradient based on our gaussian posterior. This would be feasible considering our problem space has a relatively low number of dimensions.

3 Local search study

Let $G = (V, E)$ be a graph. A matching M in G is a set of edges $M \subseteq E(G)$, so that no vertex $v \in V(G)$ is incident with more than one edge from M.

Problem 3

Maximal matching - is a matching M where we are unable to add a vertex $v \in V(G)$ to M so that the new $M^{'}$ is still a matching.

Perfect matching - maximal matching M such that all vertices V(G) are elements of M.

Near-Perfect matching - For graphs G with an odd number of vertices. Maximal matching M such that all except for exactly one vertex are elements of M.

Greedy algorithm - pick edge $e = (v_i, v_j)$; remove edged with vertices v_i or v_j from G; repeat until E(G) is empty. This algorithm will give maximal, not necessarily maximum matching.

Not all graph have near-perfect or perfect matching. - Counterexample: Beam of points.

Problem 4

We can express every fractional matching as a convex combination of matchings.

Proof (Fractional matching is a convex combination of matchings): By definition a single edge is a valid matching (in the remainding paragraph we refer to such a matching as an elementary matching). We can express any set as a weighed combination of elementary matchings or put another way as a linear inequality $e^T x = \mu \leq 1$, where e is a vector of all ones, x is the vector of coefficient and μ is the sum of weights for that particular matching. Since we know a linear function is both convex and concave we have therefore proven that all fractional matchings can be expressed as a convex combination of matchings. $\square$

Problem 5

Fig. 2 shows the solution returned by Nelder-mead. The highest score for the relaxMaximalWeightMatching problem is 338.48 with only 26% of all edges being non-zero.

Problem 6

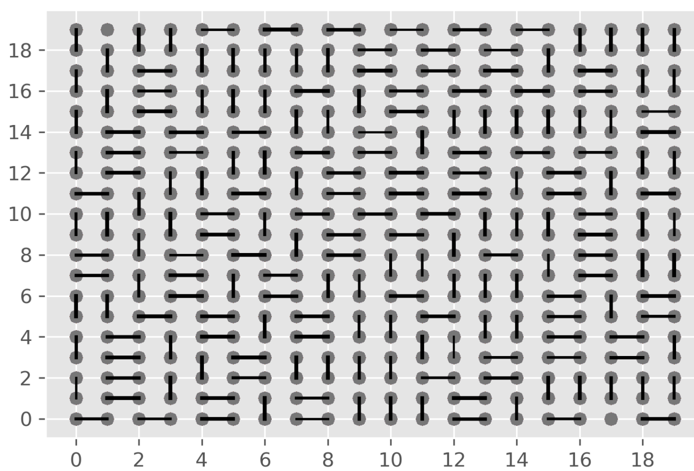


Figure 2: Interior-point maximal weighed fractional matching. Circles are vertices while lines are edges, with line-width representing total weight times fractions.

From figure 2 we can observe that the maximal weighed fractional matching is similar to the maximal weighed matching, since no vertices have more than one edge with a weight significantly above zero. Additionally, all but two vertices were joined by edges. We performed tests using random seed 15, 16, 17 and concluded that depending on weight initialization we ocasionally got perfect matchings.

Problem 7

For our local search we followed the described algorithm in our homework instructions. In order to decide whether a weight was small or large we used a normal distribution $N(\mu(W), 2\sigma(W))$, with $\mu(W)$ representing the average of our weights and $\sigma(W)$ representing the standard deviation of our weights. An edge was accepted if it had a higher weight than a sample from our normal distribution and rejected if it had a weight smaller than a sample from our normal distribution. We tested different standard deviations for our normal distribution and this one performed the best.

In table 2 we show the found maximum using five different adjacencies. The mean maximum appears to be monotonically increasing indicating additional improvements are likely possible. The highest score we achieved is 304, from this we conclude that we can get quite close to the actual optimal matching using a simple local search.

n-adjecency	maximum
1	291 ± 3.5
2	297 ± 4.6
3	301 ± 4.9
4	303 ± 3.6
5	304 ± 4.3

Table 2: maximum score using different adjacencies

Problem 8

We used a simple algorithm as a jump move for our local search. When our algorithm failed to improve a matching for a preset number we uniformly picked a preset fraction of edges for removal and then used a normal distribution for choosing whether to remove the selected edges. This meant we biased towards jump algorithm towards removing edges with smaller weights. This approach yielded slightly better results in testing.

Problem 9

Based on the slowly converging nature of our results and the sensitivity of our model to the choice of hyperparameters we assume we could using our current local search achieve at most a score of 310. Local search can therefore be a sufficient method for generating solutions for the problem of maximal fractional weighed matching depending on how close to the optimum we need to get for our problem.