HW2 STA243

Yu Zhu / Yanan Gao (both 50%)

2018/04/19

1 Problem 1

1.1 Result when p = 0.999, initial temperature = 400

We initiate the path from city A to city O then back to city A(a vector as [c(1:15),1]) as our θ, and for each iteration we uniformly and randomly exchange the order of two cities, making it as our θ^*, then calculate the $\delta = f(\theta^*) - f(\theta)$.

Our initial temperature is $\tau_1 = 400$. We repeat it for 10000 iterations, every iteration τ will be p = 0.999 times of before, and we set our stopping criterion as: stop when temperature $\tau \leq 0.0001$.

The minimum length of our path after iterations is 17, but the algorithm does not achieve this minimum for every time. Other results that can be obtained are 18, 19, 20 and so on.

So we run this function for 100 times to see the distribution of our minimum path length(only present distance from 17 to 21).

distance	number of occurrence
17	23
18	48
19	26
20	2
21	1

1.2 How the algorithm behaves with different p's and temperatures

We want to compare the results under different p and τ. We set $p_1 = 0.9$, $p_2 = 0.99$, $p_3 = 0.999$; $\tau_1 = 200$, $\tau_2 = 300$, $\tau_3 = 400$, and set them as 9 different combinations. To save time, we only run 50 times of the function this time, and the outcomes are shown below:

distance	p = 0.9,$\tau = 200$	p = 0.99,$\tau = 200$	p = 0.999,$\tau = 200$
17	0	2	4
18	3	18	23
19	7	11	20
20	16	13	2
21	12	6	1

distance	p = 0.9,τ = 300	p = 0.99,τ = 300	p = 0.999,τ = 300
17	0	5	8
18	5	18	21
19	9	17	17
20	11	7	3
21	19	3	1

distance	p = 0.9,τ = 400	p = 0.99,τ = 400	p = 0.999,τ = 400
17	2	3	8
18	4	20	22
19	4	14	16
20	17	10	4
21	11	1	0

Considering to the minimum achievement rate, theoretically the result is better if p is close to 1 and τ is relatively small. From the table, we can find out that:

• Under the same p, the increasing of the value of τ, seems not to lead to very obvious change of the result).

• Under the same τ, the more closer p is to 1, the better the result seems to be, but it consumes much more time to converge.

2 Problem 2

2.1 Introduction

The goal for this question is to develop a procedure to fit piecewise constant regression. Notice that, in order to fit this function, we need to get the estimated $\hat{B}$ (number of pieces) and estimated breakpoints $\hat{b}_i$ ($i = 1, ..., B-1$). To solve $\hat{B}$ and $\hat{b}_i$, two kinds of criteria are used: MDL and AIC. In this report, genetic algorithm is used to minimize these two criteria.

2.2 Result Using Original Parameter

The original values for each parameter are: S = 300, $P_{cross} = 0.9$, $P_c = 0.05$ and $N_{same} = 20$. The probability of a point being breakpoint is 0.5 (because we are assuming no prior distribution is known) for each chromosome in the first generation.
For both AIC and MDL, the number of pieces $\hat{B}$ is 13, their break points b_1 to b_{12} are shown below:

	b1	b2	b3	b4	b5	b6
MDL	0.10156	0.13086	0.15039	0.23049	0.25000	0.25195
AIC	0.10156	0.13086	0.15039	0.23049	0.25000	0.25195
True	0.10156	0.13086	0.15039	0.23047	0.25000	0.25195

	b7	b8	b9	b10	b11	b12
MDL	0.40039	0.44141	0.65039	0.76172	0.78125	0.81055
AIC	0.40039	0.44141	0.65039	0.76172	0.78125	0.81055
True	0.40039	0.44141	0.65039	0.76172	0.78125	0.81055

Apparently, both MDL and AIC give us the same break points.Moreover, they are very similar to the true breakpoints, indicating the algorithm in functioning very well. Further, $MDL(\hat{f}) = -495.9634$, $AIC(\hat{f}) = -1032.581$.

From the first three figures[Plots are in page 5], comparing with original fit plot, the MDL and AIC fit function produce similar results, which means the algorithm works pretty well.

According to two plots of function values through iterations, these two functions requires almost the same number of iterations to converge to the minimum value.

2.3 Using Different Values

Now we want to examine the effects of different values of parameters.
• Change S: Generally, increasing the population size leads to decreasing of MDL($\hat{f}$) and AIC($\hat{f}$), as well as significant decreasing of computation time. However, as the size increases to a certain number which is big enough, MDL($\hat{f}$) and AIC($\hat{f}$) no longer decrease.

• Change P_{cross} : If we adjust P_{cross} from 0.9 to 0.99, the algorithm will converge to the minimum with much faster speed; if we adjust P_{cross} from 0.9 to 0.8, the converging time will increase but still achieve to the minimum. We can predict that P_{cross} closed to 1 can help algorithm converge while with small P_{cross}, the algorithm may not be able to converge under this iteration number.

• Change P_c: The principle here is P_c needs to be small, for example we have tried 0.03, 0.1 and 0.15. The value of P_c has very little effect on neither the computation time nor the MDL($\hat{f}$) and AIC($\hat{f}$).

• Change N_{same} : relative small N_{same} will make iteration stop much earlier before convergence. $N_{same} = 20$ is a quite suitable stopping criteria. If we increase N_{same}, it only consumes more time but still achieves the same minimum.

2.4 Conclusion

In this problem, we have implemented the Genetic Algorithm to solve a real problem. Two model selection methods are adopted: MDL and AIC. Moreover,by tunning each parameter, we have gained a perceptual intuition on how each parameter can affect the MDL($\hat{f}$) and AIC($\hat{f}$).

[Plots in the next page]

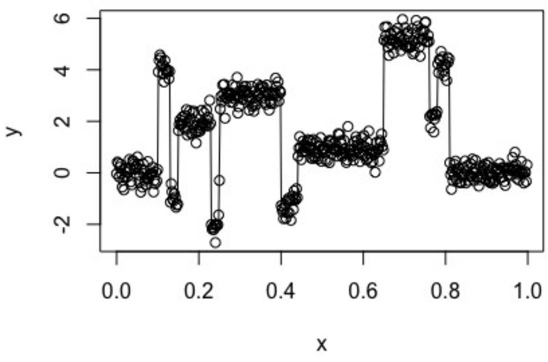

Figure 1: Original fit plot

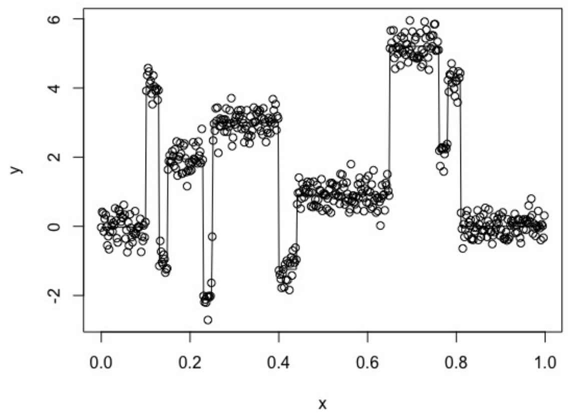

Figure 2: MDL fit plot

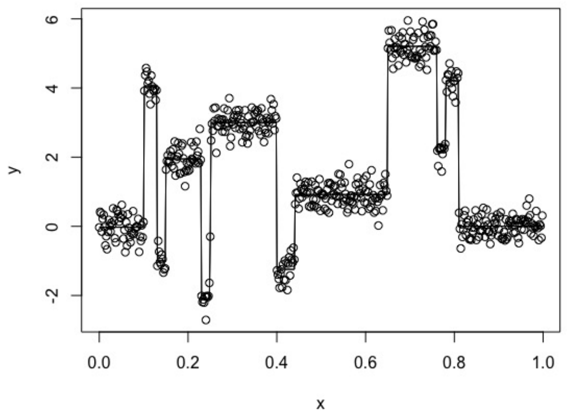

Figure 3: AIC fit plot

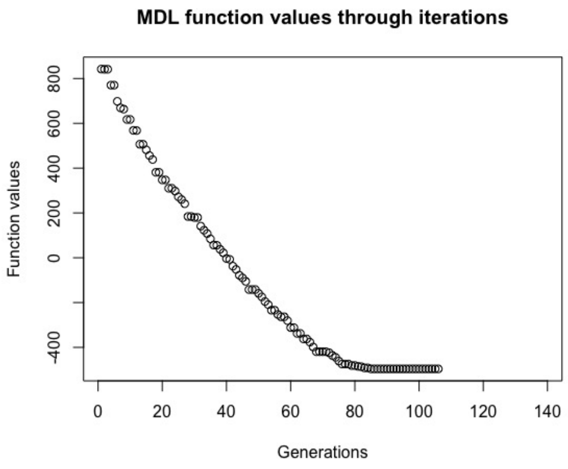

Figure 4: MDL function values through iterations

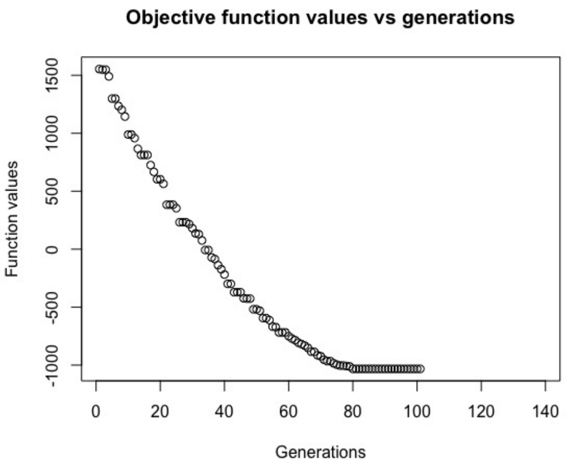

Figure 5: AIC function values through iterations