lab2question2

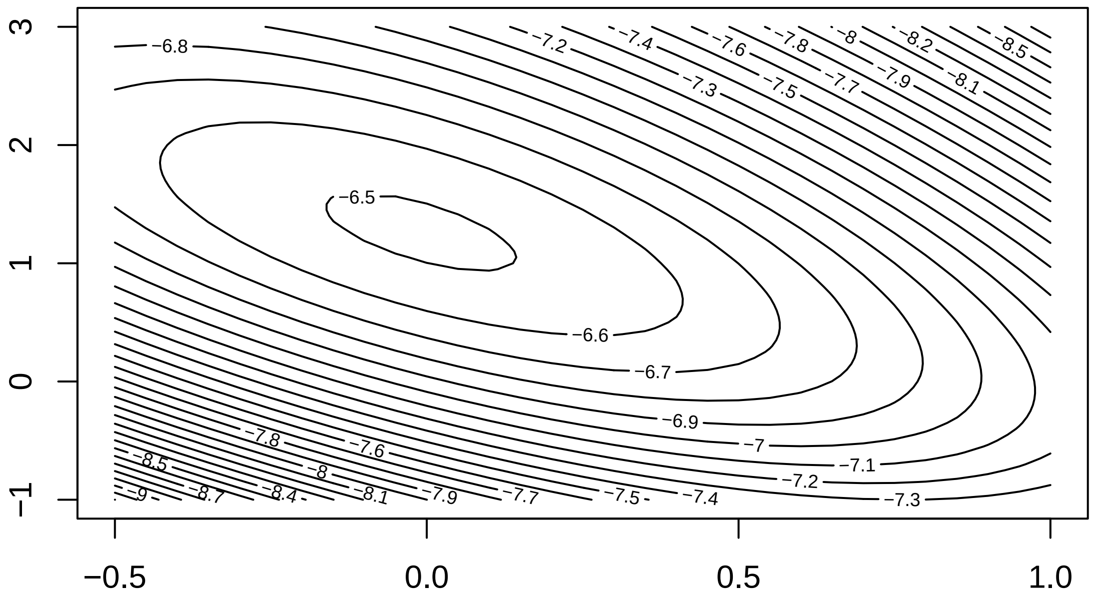
Helena Yi Yang

2025-02-02

QUESTION 2

a.

First,we implemented a maximum likelihood estimator using the steepest ascent method with a step-size-reducing line search.The function also counts the number of evaluations of the log-likelihood function and its gradient to monitor computational efficiency. Then we generated a contour plot, which illustrates the shape of the log-likelihood surface.From the plot we could roughly identify the location of global /local maximum.



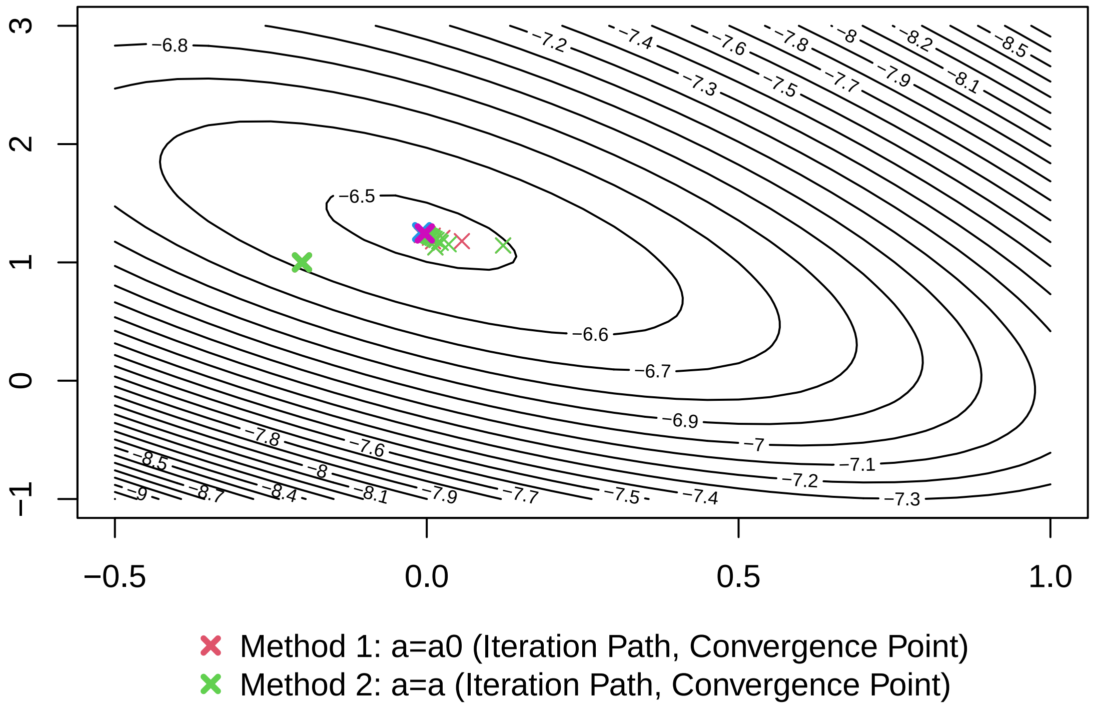
b.

Then we implemented the algorithm with two variants of the backtracking line search method:

The first variant keeps α as its initial value $\alpha_0 = 1$, while the second variant uses a reduced α.

Both strategies were initialized with $\beta_0 = -0.2$ and $\beta_1 = 1$, and the algorithm was stopped when the norm of the gradient was smaller than 10^{-5} to ensure convergence.

The number of function and gradient evaluations performed to convergence are shown as follows:



```
##        Method           B0       B1 Counts_function Counts_gradiet
## 1        a=ao -0.007299539 1.254972              35             14
## 2 a=reduced_a -0.003158611 1.244742              33             16
```

Comparison

Due to different choices of step sizes, the paths of parameter updates vary slightly, leading to some differences in the final parameter estimates.
The first method(fixed step strategy) has slightly more function evaluations but fewer gradient evaluations.

The second method(reduced step size strategy) has slightly fewer function evaluations but more gradient evaluations.

Reason for Differences

The first method: Since the step size is fixed, fewer function evaluations are required per iteration. However, because the step size is relatively large, the algorithm may “miss” the optimal value during updates, potentially requiring more iterations to converge.

The second method: More gradient evaluations are needed because the gradient must be recalculated after each step size adjustment.

c

We used function `optim` with both the `BFGS` and the `Nelder-Mead` algorithm, the number of function and gradient evaluations and the coefficients show as follows:

```
##        Method     Value           B0       B1 Counts_function Counts_gradiet
## 1        BFGS -6.484279 -0.009356126 1.262813              12              8
## 2 Nelder-Mead -6.484279 -0.009423433 1.262738              47             NA
```

Comparison

Results:

The results from `BFGS` and `Nelder-Mead` are similar but not identical from part b. However, the difference in β_0 and β_1 are relatively small due to differences in the optimization algorithms and their convergence criteria.

The precision of the result:
The precision of the results is high across all method. The small differences indicate that all method converge to a similar solution.

The number of function and gradient evaluations:
`BFGS` is the most efficient method, requiring only 12 function evaluations and 8 gradient evaluations.This efficiency is due to its use of gradient information and its ability to approximate the Hessian matrix, which accelerates convergence. `Nelder-Mead` doesn't require the computation of derivatives and relies sole on function evaluations.It requires 47 function evaluation.

d.

We used function `glm` and the results show as follow:

```
## B0: -0.009359853
```

```
## B1: 1.262823
```

Comparison

1.The results from `BFGS` and `Nelder-Mead` are almost identical to those from `glm`, indicating that these methods achieve high precision.

2.The results from the `steepest ascen`t method show some deviation from `glm` , suggesting slightly lower precision. This may be due to its slower convergence rate or step size selection. However, the results are still within a reasonable range.