HW1 第一次作业解答

更新：2025 年 10 月 13 日

项目复现：首先，配置环境。然后，进入项目根目录。最后，在终端输入命令，即可完整复现本项目所有结果。

- MacOS 系统 / Linux 系统：在 Terminal 输入

  ```
  bash main.sh
  ```

- Windows 系统：在 cmd 中输入

  ```
  .\main.bat
  ```

  或者在 PowerShell 中输入

  ```
  .\main.ps1
  ```

  或者更简单，直接双击 `main.bat` 脚本。

注 因为这里的目标函数的导数相对简单，所以有关导数的计算均使用解析形式。当然，为了普适性，本项目代码设计时，考虑了数值导数。可以在初始化类时，不传入导数函数，但需要补充方法 `_numerical_derivative` 的具体实现。

Exercise 1

使用二分法求 $f(x) = x^2 - 10$ 的正数根。

解答 本题的运行日志和示例结果保存在 `log/q1.log` 中。例如，选取区间 $[0.0, 10.0]$ 为起始区间，精度选取 10^{-6}，最大迭代次数为 50 次。最终结果为：迭代 22 次，求解的根为

$$x^* = 3.1622767448425293 \approx 3.162$$

Exercise 2

设密度函数 $f(x)=\frac{1}{2\pi}[1-\cos(x-\theta)]$, $x\in[0,2\pi]$，其中 $\theta\in[-\pi,\pi]$ 为参数。来自这个分布的 i.i.d. 的样本为

$$3.91, 4.85, 2.28, 4.06, 3.70, 4.04, 5.46, 3.53, 2.28, 1.96,$$
$$2.53, 3.88, 2.22, 3.47, 4.82, 2.46, 2.99, 2.54, 0.52, 2.50$$

我们希望估计 θ。

a 给出对数似然函数及其一阶二阶导函数的表达式并绘制 $[-\pi,\pi]$ 上的对数似然函数。
b 求 θ 的矩估计。
c 分别以 b 求得的估计值、-2.7、2.7 作为初始值，使用牛顿法求 θ 的 MLE。

注 注意到相关表达式的解析解相对比较容易，可以先求得解析解后，再进行数值优化。

解答 (a) 设样本为 $X_1, X_2, \cdots, X_n$ i.i.d. 其中 $n=20$，于是联合密度函数为

$$f_{X_1,X_2,\cdots,X_n}(x_1,x_2,\cdots,x_n)=\prod_{i=1}^{n}f_X(x_i)$$

则似然函数为

$$\mathcal{L}(x_1,x_2,\cdots,x_n;\theta)=\prod_{i=1}^{n}f_X(x_i;\theta)=\prod_{i=1}^{n}\frac{1}{2\pi}[1-\cos(x_i-\theta)]$$

那么对数似然函数为

$$L(x_1,x_2,\cdots,x_n;\theta)=\sum_{i=1}^{n}\log[1-\cos(x_i-\theta)]-n\log(2\pi)$$

简记为 L，则其一阶导为

$$L'=\frac{dL}{d\theta}=-\sum_{i=1}^{n}\frac{\sin(x_i-\theta)}{1-\cos(x_i-\theta)}$$

二阶导为

$$\begin{aligned}L''=\frac{dL'}{d\theta}&=-\sum_{i=1}^{n}\frac{\cos(x_i-\theta)(1-\cos(x_i-\theta))-\sin(x_i-\theta)\sin(x_i-\theta)}{[1-\cos(x_i-\theta)]^2}\cdot(-1)\\&=-\sum_{i=1}^{n}\frac{1}{1-\cos(x_i-\theta)}\end{aligned}$$

在 $[-\pi,\pi]$ 上绘制 L 的图像见问题 c 的收敛过程图像。

解答 (b) 首先计算一阶矩

$$\begin{aligned}\mathbb{E}(X)&=\int_0^{2\pi}x\cdot f_X(x)\,dx=\int_0^{2\pi}x\cdot\frac{1}{2\pi}[1-\cos(x-\theta)]\,dx\\&=\frac{1}{2\pi}\int_0^{2\pi}x\,dx-\frac{1}{2\pi}\int_0^{2\pi}x\cos(x-\theta)\,dx\end{aligned}\tag{1}$$

其中

$$\frac{1}{2\pi}\int_0^{2\pi} x\,dx = \frac{1}{2\pi}\cdot\frac{x^2}{2}\Big|_0^{2\pi} = \frac{1}{2\pi}\cdot\frac{(2\pi)^2}{2} = \pi$$

对于第二个式子，设 $y = x - \theta$ 则有

$$\int_0^{2\pi} x\cos(x-\theta)\,dx = \int_{-\theta}^{2\pi-\theta} (y+\theta)\cos y\,dy$$

而

$$\int_{-\theta}^{2\pi-\theta} \theta\cos y\,dy = \theta\sin y|_{-\theta}^{2\pi-\theta} = \theta\sin(2\pi-\theta) - \theta\sin(-\theta) = \theta\sin(-\theta) - \theta\sin(-\theta) = 0$$

故只需计算

$$\begin{aligned}
\int_{-\theta}^{2\pi-\theta} y\cos y\,dy &= \int_{-\theta}^{2\pi-\theta} y\,d\sin y \\
&= y\sin y|_{-\theta}^{2\pi-\theta} - \int_{-\theta}^{2\pi-\theta}\sin y\,dy \\
&= (2\pi-\theta)\sin(2\pi-\theta) - (-\theta)\sin(-\theta) - [-\cos y|_{-\theta}^{2\pi-\theta}] \\
&= 2\pi\sin(-\theta) - \theta\sin(-\theta) + \theta\sin(-\theta) + \cos(2\pi-\theta) - \cos(-\theta) \\
&= -2\pi\sin\theta
\end{aligned}$$

代回式 1 我们有

$$\mathbb{E}(X) = \pi - \frac{1}{2\pi}(0 - 2\pi\sin\theta) = \pi + \sin\theta$$

对一阶矩 $\mathbb{E}(X)$ 的估计为样本均值，即 $\hat{\mathbb{E}}(X) = \bar{X}$ 则参数 θ 的矩估计为

$$\hat{\theta} = \arcsin(\hat{\mathbb{E}}(X) - \pi) = \arcsin(\bar{X} - \pi)$$

代入数值计算有 $\hat{\theta}_M = 0.05844060614042408 \approx 0.058$。

解答 (c) 这里的一阶导数和二阶导数均有解析形式，可直接代入牛顿法求解

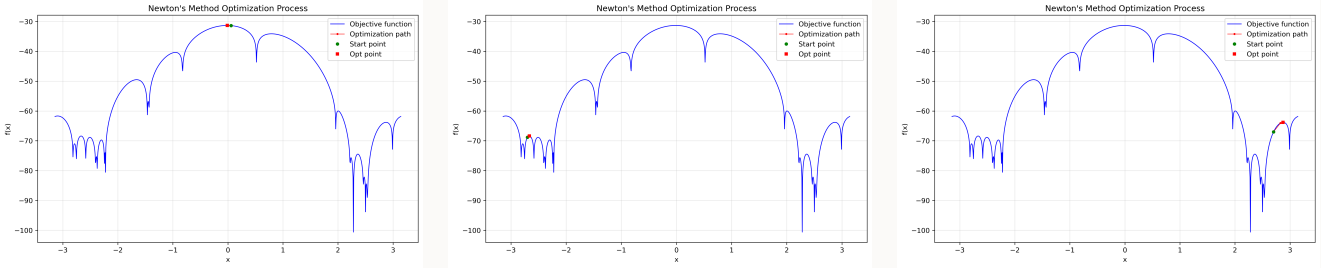
$$\theta_{t+1} = \theta_t - \frac{L'(\theta_t)}{L''(\theta_t)}$$

迭代过程保存在日志 `log/q2.log` 中，结果见表 1。

表 1: The estimate of θ and the likelihood under different $\hat{\theta}$.

Start Point	$\hat{\theta}$	$L(\hat{\theta})$	Num of Iteration
-	$\hat{\theta}_M = 0.058$	-31.399	-
0.058	-0.012	-31.343	4
-2.7	-2.667	-68.399	4
2.7	2.873	-63.806	5

迭代过程见图 1，原图存储在 `figure/` 目录下。



(a) Start from 0.058 **(b)** Start from -2.7 **(c)** Start from 2.7

图 1: Start from different point

Exercise 3

已知来自 Cauchy$(\theta, 1)$ 分布的 i.i.d. 样本为

$$1.77, -0.23, 2.76, 3.80, 3.47, 56.75, -1.34, 4.24, -2.44, 3.29,$$
$$3.71, -2.40, 4.53, -0.07, -1.05, -13.87, -2.53, -1.75, 0.27, 43.21$$

我们希望估计 θ。

a 给出对数似然函数及其一阶导函数的表达式并绘制 $[-5, 5]$ 上的对数似然函数。

b 分别以 -1、4 作为初始值，用下式给出的不动点法求 θ 的 MLE

$$x_{t+1} = x_t + \alpha g'(x_t)$$

其中 α 取 $1, 0.64, 0.25$。

c 分别以 $(-2, -1)$、$(2, 1)$ 作为初始值，使用割线法求 θ 的 MLE。

解答 (a) 设样本为 $X_1, X_2, \cdots, X_n$ i.i.d. 其中 $n = 20$，于是联合密度函数为

$$f_{X_1, X_2, \cdots, X_n}(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} f_X(x_i)$$

其中 Cauchy 分布的概率密度函数为

$$f_X(x; \theta, 1) = \frac{1}{\pi} \frac{1}{1 + (x - \theta)^2}$$

则似然函数为

$$\mathcal{L}(x_1, x_2, \cdots, x_n; \theta) = \prod_{i=1}^{n} f_X(x_i; \theta, 1) = \prod_{i=1}^{n} \frac{1}{\pi} \frac{1}{1 + (x_i - \theta)^2}$$

那么对数似然函数为

$$L(x_1, x_2, \cdots, x_n; \theta) = -\sum_{i=1}^{n} \log[1 + (x_i - \theta)^2] - n \log(\pi)$$

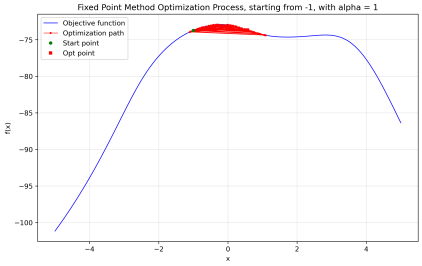
简记为 $g(\theta)$，其一阶导为

$$g'(\theta) = 2 \sum_{i=1}^{n} \frac{x_i - \theta}{1 + (x_i - \theta)^2}$$

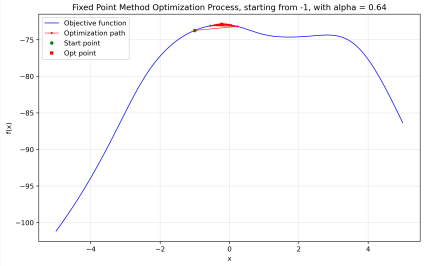
在 $[-5, 5]$ 上绘制对数似然函数的图像见问题 b 的收敛过程图像。

解答 (b) 使用不动点法进行迭代，设置最大迭代次数为 50 次，一阶导使用解析形式。注意：这里没有对一阶导进行停止条件的约束，即当 $|g'(\cdot)| \approx 0$，并不触发停止。增加该条件后，更容易在 `max_iter` 前停止，“收敛”更快。不同的参数，得到的结果见表 2，图像见图 2 和 3。

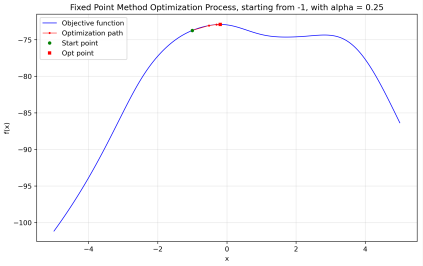
程序运行日志/结果存储在 `log/q3_a.log` 中，图像存储在 `figure/` 目录下。



(a) $\alpha = 1$

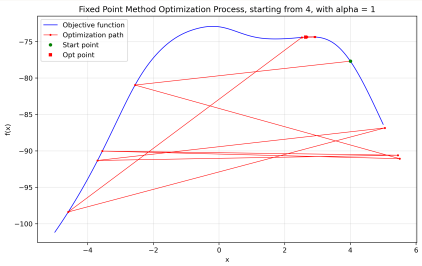


(b) $\alpha = 0.64$

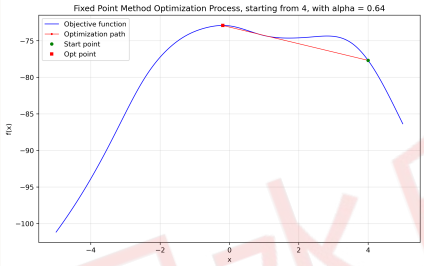


(c) $\alpha = 0.25$

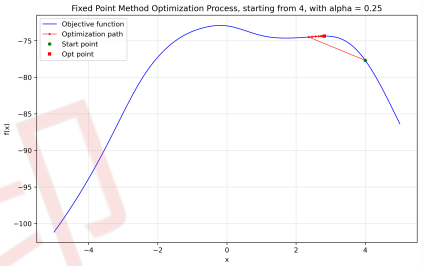
图 2: Start from -1



(a) $\alpha = 1$



(b) $\alpha = 0.64$



(c) $\alpha = 0.25$

图 3: Start from 4

表 2: The estimate of θ and the likelihood under different $\hat{\theta}$.

Start Point	α	$\hat{\theta}$	$L(\hat{\theta})$	Num of Iteration
-1	1	0.566	-73.679891	50
4	1	2.642	-74.388590	50
-1	0.64	-0.222	-72.917257	50
4	0.64	-0.191	-72.915820	50
-1	0.25	-0.192	-72.915819	12
4	0.25	2.817	-74.360461	19

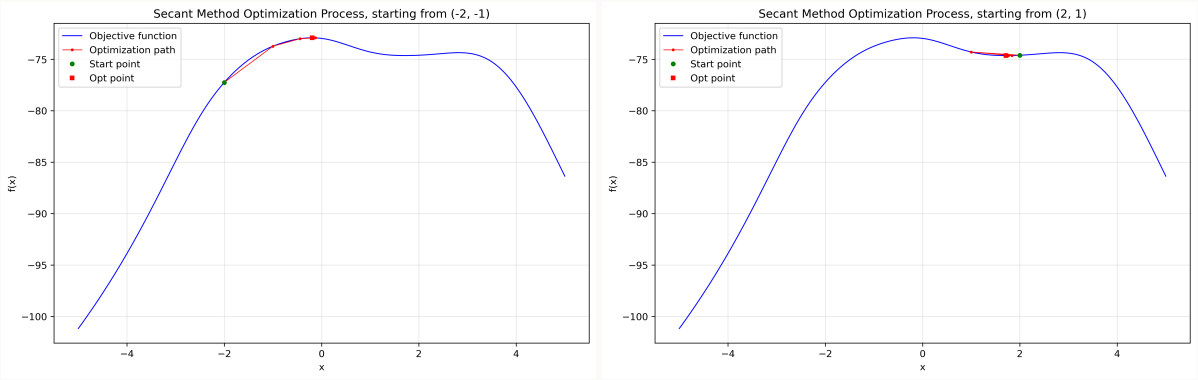
从迭代次数角度，只有 $\alpha = 0.25$ 时收敛，其中 $\alpha = 0.25, x_0 = -1$ 时收敛到全局最优。注意到，$\alpha = 1$ 时收敛过程异常，在最优点附近剧烈震荡。特别地，$\alpha = 0.64$ 虽然不收敛，但随着迭代次数增加，愈发靠近最优或局部最优点，这是因为 $g'(x_t) \approx 0$ 导致的，若增加对一阶导数不为 0 的限制，模型快速收敛。

这个例子说明，学习率 α 和初始点 x_0 的选取会直接影响模型收敛性和最优化结果。例如，当 α 过大，收敛过程可能会剧烈震荡，或陷入局部最优；起始值 x_0 的选取也会影响模型是否能收敛到全局最优处。

解答 (c) 使用割线法进行迭代，设置最大迭代次数为 50 次，一阶导使用解析形式。不同的起始点，得到的结果见表 3，图像见图 4。

表 3: The estimate of θ and the likelihood under different $\hat{\theta}$.

Start Point	$\hat{\theta}$	$L(\hat{\theta})$	Num of Iteration
$(-2,-1)$	-0.192	-72.915820	6
$(2,1)$	1.714	-74.642016	6



(a) Start from $(-2,-1)$ **(b)** Start from $(2,1)$

图 4: Start from different point

程序运行日志/结果存储在 `log/q3_b.log` 中，图像存储在 `figure/` 目录下。

Exercise 4

1974 至 1999 年间，在美国水域共有 46 起严重的原油泄露事件，每次从油轮泄露出的原油不少于 1000 桶。附件中包含以下数据：第 i 年的泄露数 N_i；第 i 年作为美国进出口一部分的在美国水域经油轮运输原油总量的估计值 b_{i1} (此值根据在国际或国外水域的泄露量进行了调整)；第 i 年在美国水域经国内油轮运输的原油总量 b_{i2}。原油运输总量以百万桶 (Bbbl) 计。

原油的油轮运输量是揭示溢出风险的一个度量。假设给定 b_{i1}, b_{i2} 下 N_i 的分布为 Poisson 分布，即 $N_i|b_{i1}, b_{i2} \sim P(\lambda_i)$, 其中 $\lambda_i = \alpha_1 b_{i1} + \alpha_2 b_{i2}$。此模型的参数为 α_1, α_2，它们分别表示在进出口和国内运输时每百万桶发生泄露的比率。我们希望估计 α_1, α_2。本题默认初始值取 $(0.1, 0.1)$。

a 给出对数似然函数及其所有一阶二阶偏导数的表达式。
b 使用牛顿法求 α_1, α_2 的 MLE。
c 估计 α_1, α_2 的 MLE 的标准误。（需要用到 Fisher 信息矩阵）
d 使用带有步长减半的梯度上升法求 α_1, α_2 的 MLE，并给出最终学习率。
e 以 $(0.1, 0.1), (0.2, 0.2)$ 作为初始值，使用拟牛顿法 (rank-one) 求 α_1, α_2 的 MLE，比较是否使用步长减半策略的结果。
f 使用 SGD 求 α_1, α_2 的 MLE。
g 使用 SGD-Momentum 并尝试不同 β 取值，求 α_1, α_2 的 MLE。

h 使用 RMSProp-Momentum 并尝试不同 β, ρ 取值，求 α_1, α_2 的 MLE。

解答 (a) 设样本为 $X_i = (N_i, b_{i1}, b_{i2}), i = 1, 2, \cdots, n$ 其中 $n = 26$，于是在 Poisson 分布的假设下，联合密度函数为

$$f_{X_1, X_2, \cdots, X_n}(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} f_{(N_i|b_{i1}, b_{i2})}(n_i|b_{i1}, b_{i2})$$

其中 Poisson 分布的概率质量函数为

$$f_{(N_i|b_{i1}, b_{i2})}(n_i|b_{i1}, b_{i2}) = \frac{\lambda_i^{n_i} e^{-\lambda_i}}{n_i!} = \frac{(\alpha_1 b_{i1} + \alpha_2 b_{i2})^{n_i} e^{-(\alpha_1 b_{i1} + \alpha_2 b_{i2})}}{n_i!}$$

则似然函数为

$$\mathcal{L}(\{(n_i, b_{i1}, b_{i2})\}_{i=1}^n; \alpha_1, \alpha_2) = \prod_{i=1}^{n} f_{(N_i|b_{i1}, b_{i2})}(n_i|b_{i1}, b_{i2}) = \prod_{i=1}^{n} \frac{(\alpha_1 b_{i1} + \alpha_2 b_{i2})^{n_i} e^{-(\alpha_1 b_{i1} + \alpha_2 b_{i2})}}{n_i!}$$

那么对数似然函数为。

$$L(\{(n_i, b_{i1}, b_{i2})\}_{i=1}^n; \alpha_1, \alpha_2) = \sum_{i=1}^{n} [n_i \log(\alpha_1 b_{i1} + \alpha_2 b_{i2}) - (\alpha_1 b_{i1} + \alpha_2 b_{i2}) - \log(n_i!)]$$

平凡地，我们只需考虑核函数，简记为 L，则其一阶偏导为

$$\frac{\partial L}{\partial \alpha_1} = \sum_{i=1}^{n} \left[\frac{n_i b_{i1}}{\alpha_1 b_{i1} + \alpha_2 b_{i2}} - b_{i1} \right]$$

$$\frac{\partial L}{\partial \alpha_2} = \sum_{i=1}^{n} \left[\frac{n_i b_{i2}}{\alpha_1 b_{i1} + \alpha_2 b_{i2}} - b_{i2} \right]$$

二阶偏导为

$$\frac{\partial^2 L}{\partial \alpha_1^2} = \sum_{i=1}^{n} \left[-\frac{n_i b_{i1}^2}{(\alpha_1 b_{i1} + \alpha_2 b_{i2})^2} \right]$$

$$\frac{\partial^2 L}{\partial \alpha_2^2} = \sum_{i=1}^{n} \left[-\frac{n_i b_{i2}^2}{(\alpha_1 b_{i1} + \alpha_2 b_{i2})^2} \right]$$

$$\frac{\partial^2 L}{\partial \alpha_1 \partial \alpha_2} = \frac{\partial^2 L}{\partial \alpha_2 \partial \alpha_1} = \sum_{i=1}^{n} \left[-\frac{n_i b_{i1} b_{i2}}{(\alpha_1 b_{i1} + \alpha_2 b_{i2})^2} \right]$$

故梯度为

$$\nabla L(\alpha_1, \alpha_2) = \begin{pmatrix} \dfrac{\partial L}{\partial \alpha_1} \\ \dfrac{\partial L}{\partial \alpha_2} \end{pmatrix} := \nabla L(\boldsymbol{\alpha})$$

Hessian 矩阵为

$$H(\alpha_1, \alpha_2) = \begin{pmatrix} \dfrac{\partial^2 L}{\partial \alpha_1^2} & \dfrac{\partial^2 L}{\partial \alpha_1 \partial \alpha_2} \\ \dfrac{\partial^2 L}{\partial \alpha_2 \partial \alpha_1} & \dfrac{\partial^2 L}{\partial \alpha_2^2} \end{pmatrix} := H(\boldsymbol{\alpha})$$

这里 $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)^T \in \mathbb{R}^2$。

解答 (b) 这里用最基础的牛顿法，即使用梯度和 Hessian 矩阵的真实表达式进行迭代。停止条件不考虑梯度过小的情况。

$$\boldsymbol{\alpha}_{t+1} = \boldsymbol{\alpha}_t - H(\boldsymbol{\alpha}_t)^{-1}\nabla L(\boldsymbol{\alpha}_t)$$

最终结果为：迭代 9 次，最优解为 $\boldsymbol{\alpha}^* = (1.09715253,\ 0.93755458)^T = (1.097,\ 0.938)^T$，对应的目标函数值（即对数似然函数的核函数）为 $L^* = -23.255$。

程序运行和结果日志保存在 `log/q4_bc.log` 中。

解答 (c) 由 Poisson 分布的性质知 $\mathbb{E}(N_i) = \lambda_i = \alpha_1 b_{i1} + \alpha_2 b_{i2}$。于是 Fisher 信息矩阵为

$$\mathcal{I}(\boldsymbol{\alpha}) = -\mathbb{E}[H(\boldsymbol{\alpha})] = \begin{pmatrix} \sum_{i=1}^n \frac{b_{i1}^2}{\alpha_1 b_{i1} + \alpha_2 b_{i2}} & \sum_{i=1}^n \frac{b_{i1}b_{i2}}{\alpha_1 b_{i1} + \alpha_2 b_{i2}} \\ \sum_{i=1}^n \frac{b_{i1}b_{i2}}{\alpha_1 b_{i1} + \alpha_2 b_{i2}} & \sum_{i=1}^n \frac{b_{i2}^2}{\alpha_1 b_{i1} + \alpha_2 b_{i2}} \end{pmatrix}$$

代入 $\boldsymbol{\alpha}^*$，计算得到 Fisher 信息矩阵为

$$\mathcal{I}(\boldsymbol{\alpha}^*) = \begin{pmatrix} 16.412 & 9.390 \\ 9.390 & 7.880 \end{pmatrix}$$

则 MLE 的标准误的估计，为 $[\mathcal{I}(\boldsymbol{\alpha}^*)]^{-1}$ 对角线元素的平方根

$$\hat{\mathrm{SE}}(\boldsymbol{\alpha}^*) = \sqrt{\mathrm{diag}([\mathcal{I}(\boldsymbol{\alpha}^*)]^{-1})}$$

代入计算得 $\hat{\mathrm{SE}}(\alpha_1^*) = 0.438, \hat{\mathrm{SE}}(\alpha_2^*) = 0.631$。

解答 (d) 步长减半的梯度上升法

$$\boldsymbol{\alpha}_{t+1} = \boldsymbol{\alpha}_t + \eta_t \nabla L(\boldsymbol{\alpha}_t)$$

其中初始学习率 $\eta_0 = 1$，当 $L(\boldsymbol{\alpha}_{t+1}) < L(\boldsymbol{\alpha}_t)$ 时，令 $\eta_{t+1} = \eta_t/2$，否则 $\eta_{t+1} = \eta_t$。

设置最大迭代次数为 300 次，一阶导使用解析形式。最终结果为：迭代 43 次，最优解为 $\boldsymbol{\alpha}^* = (1.097152,\ 0.937556)^T = (1.097,\ 0.938)^T$，对应的目标函数值（即对数似然函数的核函数）为 $L^* = -23.255$。最终学习率为 $\eta = 0.0625,\ 0.1250$。程序运行和结果日志保存在 `log/q4_d.log` 中。

解答 (e) 拟牛顿法 (rank-one)

$$\boldsymbol{\alpha}_{t+1} = \boldsymbol{\alpha}_t - \eta_t M_t^{-1}\nabla L(\boldsymbol{\alpha}_t)$$

其中 $M_t \in \mathbb{R}^{2\times 2}$ 的更新公式为

$$M_{t+1} = M_t + c_t v_t v_t^T$$

这里

$$z_t = \boldsymbol{\alpha}_{t+1} - \boldsymbol{\alpha}_t \in \mathbb{R}^2 \quad y_t = L'(\boldsymbol{\alpha}_{t+1}) - L'(\boldsymbol{\alpha}_t) \in \mathbb{R}^2$$
$$v_t = y_t - M_t z_t \in \mathbb{R}^2 \quad c_t = 1/[v_t^T z_t] \in \mathbb{R}$$

同样，设置学习率为 1，最大迭代次数为 300 次，一阶导使用解析形式，起始 $M_0 = -\mathcal{I}(\boldsymbol{\alpha}_0)$。最终结果为：

- 起始点 $(0.1, 0.1)$，使用步长减半，迭代 6 次，最优解为 $\boldsymbol{\alpha}^* = (1.09715253,\ 0.93755458)^T = (1.097,\ 0.938)^T$，对应的目标函数值（即对数似然函数的核函数）为 $L^* = -23.255$。
- 起始点 $(0.2, 0.2)$，使用步长减半，迭代 6 次，最优解为 $\boldsymbol{\alpha}^* = (1.09715253,\ 0.93755458)^T = (1.097,\ 0.938)^T$，对应的目标函数值（即对数似然函数的核函数）为 $L^* = -23.255$。

这里并没有触发步长减半，可能是学习率取的恰好。所以，步长减半策略次数无所谓。但假设使用 2 作为初始学习率，步长减半仍然能够收敛，不使用步长减半则无法得到最优解。

程序运行和结果日志保存在 `log/q4_e.log` 中。

注 学习率的选取十分重要，学习率过大，会导致无法收敛，即使迭代次数够多。故这里均采用了步长减半的策略。另外，拟牛顿法的初始 M_0 的选取也会影响收敛速度。这里选取了 Fisher 信息矩阵的负值作为初始 M_0，尝试了 $M_0 = -I_2$，收敛速度更慢。

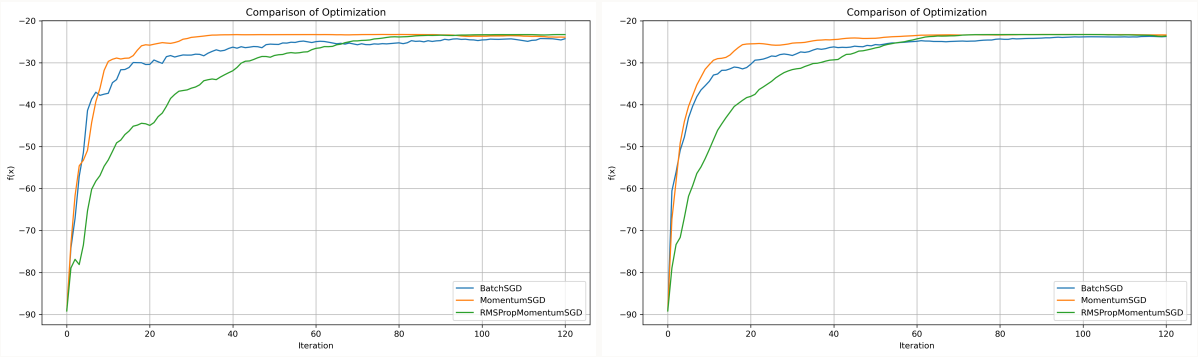
解答 (f, g, h) 基于 SGD 算法的程序运行和结果日志保存在 `log/q4_fgh.log` 中。尝试不同相关超参数后，这里选择的超参数和最终结果如表 4 所示。

总迭代次数均取 120 次，由于 SGD 的随机取点的特点，判断结束比较困难。这里设置迭代 120 次，绘制图像直观判断。批量 batch 大小设置为 1 （设置 > 1 我们会发现迭代过程曲线更平滑）。为方便比较，学习率、β 和 ρ 取相同。

表 4: Result of SGD, MomentumSGD, and RMSPropMomentumSGD about $\boldsymbol{\alpha}^*$

learning rate	batch size	β	ρ	$\boldsymbol{\alpha}^*$	$L(\boldsymbol{\alpha}^*)$
0.011	1	-	-	$(0.859,\ 0.791)$	-24.271
0.011	1	0.49	-	$(1.382,\ 0.959)$	-23.906
0.011	1	0.49	0.85	$(1.014,\ 1.075)$	-23.286

迭代图像见图 5。



(a) Batch size equals 1　　**(b)** Batch size equals 3

图 5: $L(\boldsymbol{\alpha})$ Curve with iteration under different batch size

Exercise 5

a 不同优化器在不同任务上有着不同的适配学习率。请尝试修改学习率、迭代轮次和优化器，找出在该场景下适配 SGD 和 Adam 的学习率分别是多少，并比较最终 SGD 和 Adam 的收敛曲线。

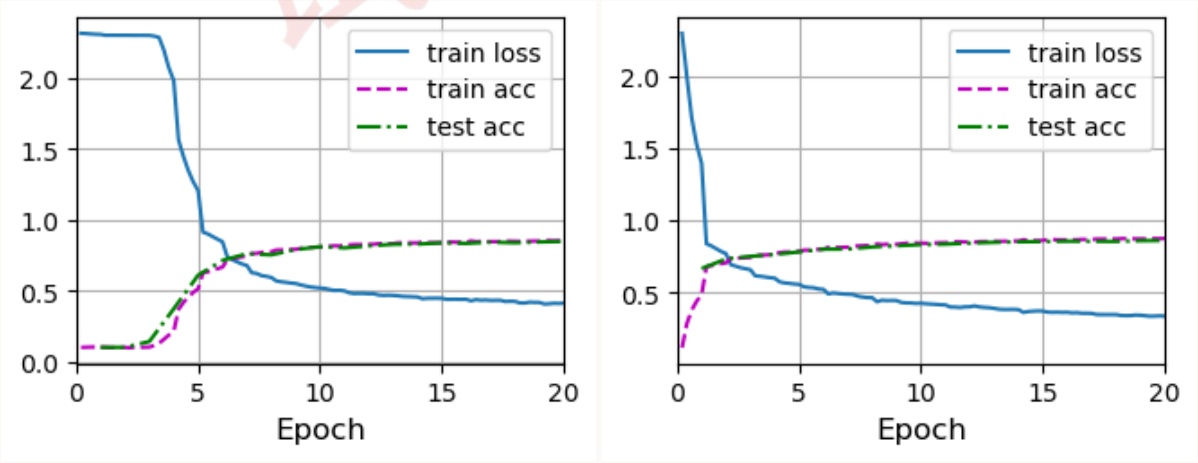
b 单变量优化问题中，牛顿法收敛速度较快，但在部分初始值或函数条件下可能无法收敛；二分法收敛速度较慢，却在函数满足“连续且区间端点函数值异号”的条件下十分稳健，能保证找到至少一个解。请设计一种混合算法，结合牛顿法的高效性与二分法的稳健性，实现对单变量优化问题高效且可靠的求解。

c 可以发现，无论是直接利用二阶梯度信息，还是通过“割线”思想近似二阶梯度的算法，其收敛速度通常快于仅使用一阶梯度信息的算法。试从几何直观层面分析：二阶梯度相较于一阶梯度，提供了哪些额外信息，从而帮助算法实现更快收敛?

d 从核心思想来看，拟牛顿法与割线法一致，均通过“割线”信息近似二阶梯度，以平衡计算复杂度与收敛速度，即拟牛顿法可视为割线法在多变量优化中的推广，但二者的公式形式却表现出显著的差异。请仔细阅读课件/课本，找出割线法与拟牛顿法形式差异巨大的根本原因。

e Adam/AdamW 是深度学习领域最常用的优化器之一，其中 Adam 已在课堂中学习。请自行探究 AdamW 的核心原理，从算法思想层面将其与 Adam 进行对比分析。

解答 (a) 简单地尝试后，可以使用学习率分别为 0.1 和 0.001，结果如图 6 所示。



(a) SGD: Learning rate = 0.1　　**(b)** Adam: Learning rate = 0.001

图 6: Different learning rate at SGD and Adam (Batch size = 128, Epoch = 20).

解答 (b) 牛顿法收敛快，但不稳健；二分法收敛慢，却比较好保障。混合算法：

- **Step 1** 当牛顿法有保障时，使用牛顿法快速迭代，若收敛则结束，否则进入 Step 2
- **Step 2** 当牛顿法失去保障时，换用二分法稳健迭代，若收敛则结束，否则
  - 牛顿法条件满足，则进入 Step 1
  - 若牛顿法条件仍然不满足，继续 Step 2

解答 (c) 对于 f 目标函数，梯度 ∇f 提供了 f 增大或减小的方向和程度，没有对 ∇f 自身的描述。但 Hessian 阵 H 提供了梯度 ∇f 的变化，即曲率，或者简单理解为该位置梯度变化的程度。

- 只使用梯度 ∇f 进行迭代，迭代步长难以估量。
- 但若知道 H 则额外知道了当前位置的曲率，即梯度自身变化的程度。对于曲率大的地方，弯曲程度大，步长过大，可能会跳过最优点，所以对应的取较小的步长。

在几何程度，直观地理解就是：当爬一座十分标准的山（即没有过于崎岖的轮廓）时，梯度只告知应该向最陡的地方爬，步长不确定。但 H 除了梯度的信息，还告知在山坡应该大跨步。但即将到达山顶时，山的弯曲程度大，应该小步走。

解答 (d) 想法上，二者相似，都希望能对二阶导进行近似，但

- **割线法**是单变量情形，对二阶导是有唯一的估计的，且“割线”含义明确。
- **拟牛顿法**主要针对多变量情形，且只是“割线”的近似想法，本质上还是在解一个方程组，实现对矩阵 H 的估计。而且这个方程组大约有 $p(p+1)/2$ 个未知数，却只有 p 个方程。即解是不唯一的，近似的方法也很多，而且对矩阵的正定负定也有要求。

解答 (e) 过拟合是模型训练总是会出现的问题，具体表现为在训练集表现较优，但在测试集上的表现却显著不佳，模型泛化能力弱。所以使用权重衰减 Weight Decay 抑制模型权重过大，防止过拟合。

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} f(x_i;\boldsymbol{\theta}) + \lambda||\boldsymbol{\theta}||_2^2 \tag{2}$$

Adam 和 AdamW 核心算法相似，主要区别在于 Weight Decay 的设计，简单来说：

- **Adam** 中的 Weight Decay 会受梯度的影响。注意式 2 对应的梯度 ∇L 会留存一项 $2\lambda\boldsymbol{\theta}$，如此传统 Adam 更新参数时的步长中 $2\lambda\boldsymbol{\theta}$ 会和旧函数的有关梯度 ∇f 的项相乘。

  $$\boldsymbol{\theta}_l^{t+1} = \boldsymbol{\theta}_l^t - \alpha[(\Delta_l^t) + c(\nabla f)\boldsymbol{\theta}_l^t] = (1 - \alpha c(\nabla f))\boldsymbol{\theta}_l^t - \alpha\Delta_l^t$$

  其中 $c(\nabla f)$ 是一个含有 ∇f 的函数。即 $\boldsymbol{\theta}$ 的衰减和梯度 ∇f 相关。
- **AdamW** 中的 Weight Decay 更符合最初设计的权重衰减的思想。AdamW 降衰减独立出来，即先进行参数 θ 迭代，然后进行衰减。

  $$\boldsymbol{\theta}_l^{t+1} = \boldsymbol{\theta}_l^t - \alpha(\Delta_l^t) - \lambda'\boldsymbol{\theta}_l^t = (1 - \lambda')\boldsymbol{\theta}_l^t - \alpha(\Delta_l^t)$$

  如此，$\boldsymbol{\theta}$ 的衰减和梯度 ∇f 无关，各分量衰减在每一步是公平的。