计算物理 A——Homework 13

何金铭 PB21020660

1 题目描述

用 Metropolis-Hasting 抽样方法计算积分：

$$I = \int_0^\infty (x-\alpha\beta)^2 f(x)\,dx = \alpha\beta^2 \tag{1}$$

$$f(x) = \frac{1}{\beta\Gamma(\alpha)}(\frac{x}{\beta})^{\alpha-1}\exp\left\{-\frac{x}{\beta}\right\}$$

设积分的权重函数为：$p(x) = f(x)$ 和 $p(x) = (x-\alpha\beta)^2 f(x)$
给定参数 α, β，并用不同的 γ 值，分别计算积分，讨论计算精度和效率
其中，设 $T_{ij} = T(x \to x') = T(x') = 0.5\exp\left\{-\frac{x'}{\gamma}\right\}$

2 理论分析

2.1 Metropolis-Hasting 抽样方法

采用不对称的分布矩阵 T 和接受矩阵 A
其中 T 为一个任意形状的方阵(最好与分布有着类似的形式)$T_{ij} = T(x \to x'), A_{ij} = \min\left\{1, \frac{p_j T_{ji}}{p_i T ij}\right\}$
根据细致平衡条件：$\frac{p_j}{p_i} = \frac{W_{ij}}{W_{ji}}$，有

$$W_{ij} = \begin{cases} T_{ij} & ,if\ p_j T_{ji} > p_i T_{ij} \\ \frac{p_j}{p_i} T_{ji} & ,if\ p_j T_{ji} < p_i T_{ij} \end{cases} \tag{2}$$

$$W_{ii} = 1 - \sum_{j\neq i} W_{ij} \tag{3}$$

2.2 权重函数 $p(x) = f(x)$

设前一个点为 x，后一个点为 x'
有关系：

$$\frac{p_j}{p_i} = \left(\frac{x'}{x}\right)^{\alpha-1}\exp\left\{-\frac{x'-x}{\beta}\right\} \tag{4}$$

$$\frac{T_{ji}}{T_{ij}} = \exp\left\{\frac{x'-x}{\gamma}\right\} \tag{5}$$

则，记：

$$R_1 = \frac{p_j T_{ji}}{p_i T_{ij}} = \left(\frac{x'}{x}\right)^{\alpha-1}\exp\left\{-\frac{x'-x}{\beta}\right\}\exp\left\{\frac{x'-x}{\gamma}\right\} \tag{6}$$

2.3 权重函数 $p(x) = (x - \alpha\beta)^2 f(x)$

设前一个点为 x，后一个点为 x'

有关系：

$$\frac{p_j}{p_i} = \left(\frac{x' - \alpha\beta}{x - \alpha\beta}\right)^2 \left(\frac{x'}{x}\right)^{\alpha-1} \exp\left\{-\frac{x' - x}{\beta}\right\} \tag{7}$$

$$\frac{T_{ji}}{T_{ij}} = \exp\left\{\frac{x' - x}{\gamma}\right\} \tag{8}$$

则，记：

$$R_2 = \frac{p_j T_{ji}}{p_i T_{ij}} = \left(\frac{x' - \alpha\beta}{x - \alpha\beta}\right)^2 \left(\frac{x'}{x}\right)^{\alpha-1} \exp\left\{-\frac{x' - x}{\beta}\right\} \exp\left\{\frac{x' - x}{\gamma}\right\} \tag{9}$$

Remark.

若选择 $p(x) = (x - \alpha\beta)^2 f(x)$，则和直接计算原积分没有任何区别，没有任何便利之处，故之后不再讨论这种情况。

2.4 重要抽样法计算积分

于之前的作业讨论过，这里不详细说明。

3 算法模拟

3.1 一些量的标定

在以下的讨论中，确定 $\alpha = 2, \beta = 1$，调整 γ，并且调增热化系数 $k = 0.025, 0.1, 0.4$，讨论热化系数对结果的精度与效率的影响；还进一步调整了初始位置 x_0 的值，验证其对 Markov Chain 没有任何影响。

3.2 $p(x) = f(x)$ 的算法过程

记 $r = \left(\frac{x'}{x}\right)^{\alpha-1} \exp\left\{-\frac{x' - x}{\beta}\right\} \exp\left\{\frac{x' - x}{\gamma}\right\}$，并代入 $x' = x_t, x = x_n$，并设初始值 $x_0 = 1, 100, 0.01$

1. 生成均匀分布的随机数 $\xi \in [0, 1]$
2. 定义 $x_t = -\ln \xi$，以便于生成一个遍历 $[0, \infty]$ 的 Markov Chain
3. 定义 r 如上式，若 $r > 1$，则 $x_{n+1} = x_t$；否则，生成均匀随机数 $\xi \in [0, 1]$，若 $r > \xi$，则 $x_{n+1} = x_t$，否则 $x_{n+1} = x_n$
4. 记录下所有的点 x, 并计数，其总数为 N，则其积分值 $I \cong \sum_{i=m}^{N} \frac{1}{N - m}(x_i - \alpha\beta)^2$。其中 m 代表热化所需要的步数。这里取 $m = k \times N$

3.3 $p(x) = (x - \alpha\beta)^2 f(x)$ 的算法过程

此处不进行讨论，于上面的理论分析部分已经做过说明。

4 程序说明

4.1 主要程序

MCMC.c Metropolis-Hasting 方法主程序。把所有功能写进了一个文件中的多个函数内，**编译时需要手动修改选用的函数！**

rn() 一个产生随机数的函数，每次调用一次即可获得一个随机数。

func(double x1,double x2,double gamma) 判别式 r 的表达式。

abs_d(double a) 一个 double 型绝对值函数

mc_accurate() 一个用于 Metropolis-Hasting 抽样的函数，γ 值在区间内改变，链长 $N = 10^6$，同时进行精度的计算。

mc_efficient() 一个用于 Metropolis-Hasting 抽样的函数，γ 值固定，N 的值取值为 $10, 100, ..., 10^8$，用于效率的计算。

mc_visual.py 对结果进行可视化操作，该文件为纯作图文件，助教可以不用检查，故此文件不加注释。

4.2 程序结果

MCMC.exe Metropolis-Hasting 方法主程序。把所有功能写进了一个文件中的多个函数内，**编译时需要手动修改！**

./data 文件夹路径，里面存放了各种导出的数据

accurate_1.csv $\gamma \in [0.0001, 1], N = 10^6, k = 0.1, x_0 = 1$ 时的情形，第一列代表 γ 值，第二列代表积分值 I，第三列代表相对误差 δI。以下几个同。

accurate_2.csv $\gamma \in [0.0001, 100], N = 10^6, k = 0.1, x_0 = 1$ 时

accurate_3.csv $\gamma \in [0.1, 10], N = 10^6, k = 0.1, x_0 = 1$ 时

accurate_4.csv $\gamma \in [1, 1.25], N = 10^6, k = 0.1, x_0 = 1$ 时

k_04.csv $\gamma \in [0.0001, 100], N = 10^6, k = 0.4, x_0 = 1$ 时（文件名后加 _s 的为其极值附近的点）

k_0025.csv $\gamma \in [0.0001, 100], N = 10^6, k = 0.025, x_0 = 1$ 时（文件名后加 _s 的为其极值附近的点）

x_0_100.csv $\gamma \in [0.0001, 100], N = 10^6, k = 0.1, x_0 = 100$ 时（文件名后加 _s 的为其极值附近的点）

x_0_1000.csv $\gamma \in [0.0001, 100], N = 10^6, k = 0.1, x_0 = 1000$ 时（文件名后加 _s 的为其极值附近的点）

efficient.csv

accurate_2.csv $\gamma = 1.0568, N = 10, 100, ...10^8, k = 0.1, x_0 = 1$ 时，第一列代表 N 的值，第二列代表积分值 I，第三列代表相对误差 δI。

./pic 文件夹路径，里面存放了各种由数据转来的图片，其具体含义请参见报告中的内容。

4.3 其他说明

1. 数据都写于 CSV 文件中
2. 其中 Python 程序用到的库有:

- matplotlib.pyplot：用于作图
- numpy：用于数据处理
- csv：用于读写 CSV 文件

5 结果分析

5.1 分析不同的 γ 值下的计算精度

记绝对误差为 $\Delta I = \left|I\right|_{calcuate} - I_0\left|\right.$，则有相对误差为 $\delta I = \dfrac{\Delta I}{I_0}$
于两个区间中取 γ 的值:

1. $\gamma \in [0.0001, 100]$，其中 γ 取 1000 个值，且步长均匀。
2. 考虑到上一种取值方法下，$\gamma \in [0.0001, 1]$ 中的取值太少，故再于 $\gamma \in [0.0001, 1]$ 中取 1000 个值，且步长均匀。
3. 并且于 $\gamma \in [0.1, 10]$ 中取 1000 个值，且步长均匀。

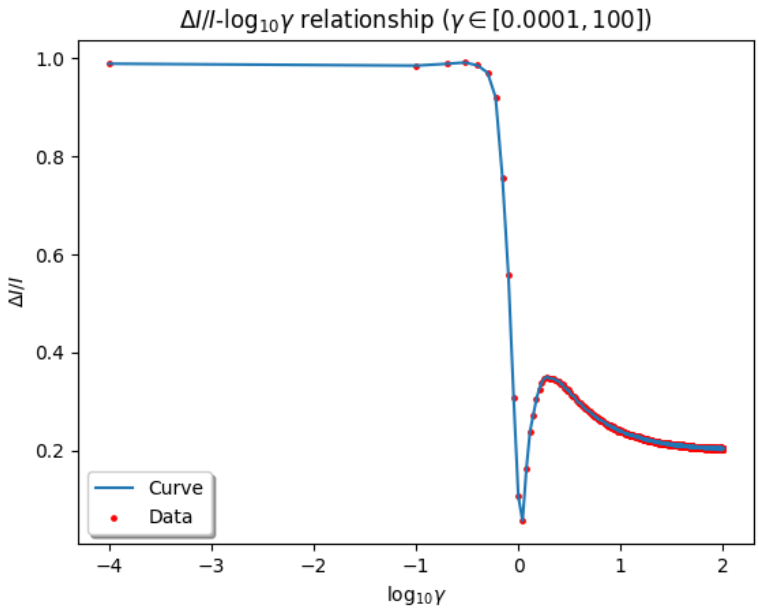

图 1: $\gamma \in [0.0001, 100]$

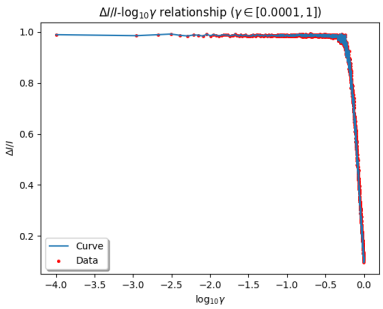

图 2: $\gamma \in [0.0001, 1]$

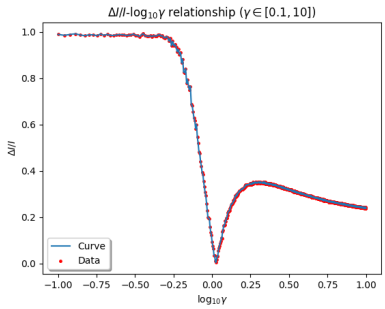

图 3: $\gamma \in [0.1, 10]$

发现误差的最小值于 $\log_{10}\gamma \in [0, 0.1]\,, (\gamma \in [1, 1.25])$ 之间取到，故继续做更精细的取值。

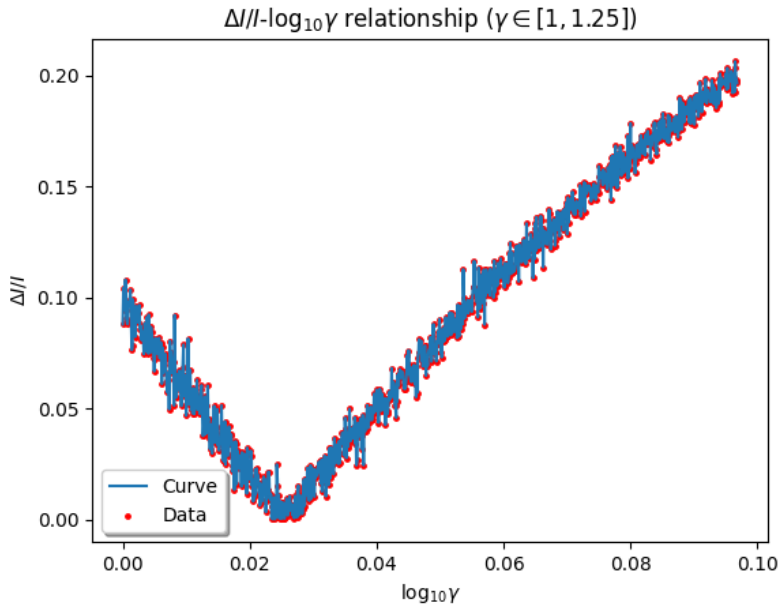

图 4: $\gamma \in [1, 1.25]$

发现相对误差随 $\gamma \in [1, 1.25]$ 的变化在抖动，并且对比数据点发现，于 $\gamma = 1.0568$ 附近，相对误差在最小值附近，其精度达到最大，误差仅为 0.3%。

分析

下面来分析为什么 δI-$\log_{10}\gamma$ 曲线会呈现这种形状。

1. $\log_{10}\gamma \in [-4, -0.25]$ 时，相对误差 δ 为 1，说明 $I\big|_{calcuate}$ 约为 0。
2. $\log_{10}\gamma \in [-0.25, 0.024]$ 时，相对误差 δ 快速降至接近于 0 的值，说明 $I\big|_{calcuate}$ 于此区间内迅速的接近于真实值。
3. $\log_{10}\gamma \in [0.024, 0.25]$ 时，相对误差 δ 开始上升，达到一个极大值，说明 $I\big|_{calcuate}$ 于此区间内又开始远离真实值。

4. $\log_{10}\gamma \in [0.25, 2]$ 时，相对误差 δ 又开始下降，达到一个渐进不变的值，说明 $I\big|_{calcuate}$ 于此区间内又开始逼近真实值，且最后保持不变。

对比 $f(x)$ 的大致图像，与式 $r = \left(\frac{x'}{x}\right)^{\alpha-1} \exp\left\{-\frac{x'-x}{\beta}\right\} \exp\left\{\frac{x'-x}{\gamma}\right\}$ 进行分析

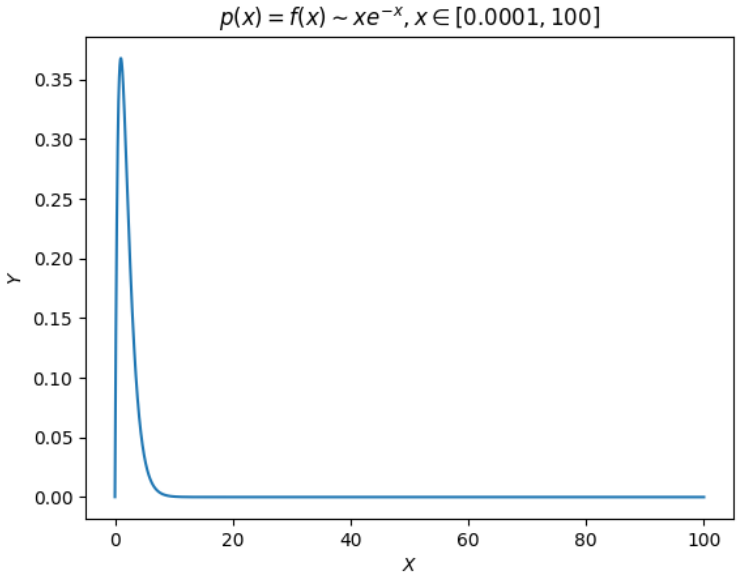

图 5: $f(x)$ 大致图像

发现：

1. $\log_{10}\gamma \in [-4, -0.25]$ 时，γ 值较小，以至于 r 中 $\exp\left\{\frac{x'-x}{\gamma}\right\}$ 占主导地位，Markov Chain 逐渐趋于 x 正半轴 $(x \to 100)$，积分值 $I \to 0$。

2. $\log_{10}\gamma \in [-0.25, 0.024]$ 时，γ 值开始变大，使得以上作用变小，且 $\gamma \to 1+\varepsilon$ 时，$r \cong \frac{x'}{x}\exp\{-\varepsilon(x'-x)\}$，此时当 $x' < x$ 时，$r > 1$，Markov Chain 向原点靠近，$I\big|_{calcuate} \to I_0$，积分趋于真实值。

3. $\log_{10}\gamma \in [0.024, 0.25]$ 时，当 γ 值，恰当的时候，Markov Chain 可能一直趋于峰值，导致误差又增加。

4. $\log_{10}\gamma \in [0.25, 2]$ 时，当 γ 值过大时，Markov Chain 可能过分趋于原点，导致误差稍下降，而且趋于一个定值。

总结

可见，当 $T(x \to x') = T(x') \sim f(x)$ 时（此处表现为 $\gamma \to \beta + \varepsilon = 1 + \varepsilon$），Markov Chain 越准确。

下面于 $\gamma = 1.0568$ 处进行计算效率的讨论。

5.2 分析 $\gamma = 1.0568$ 处的计算效率

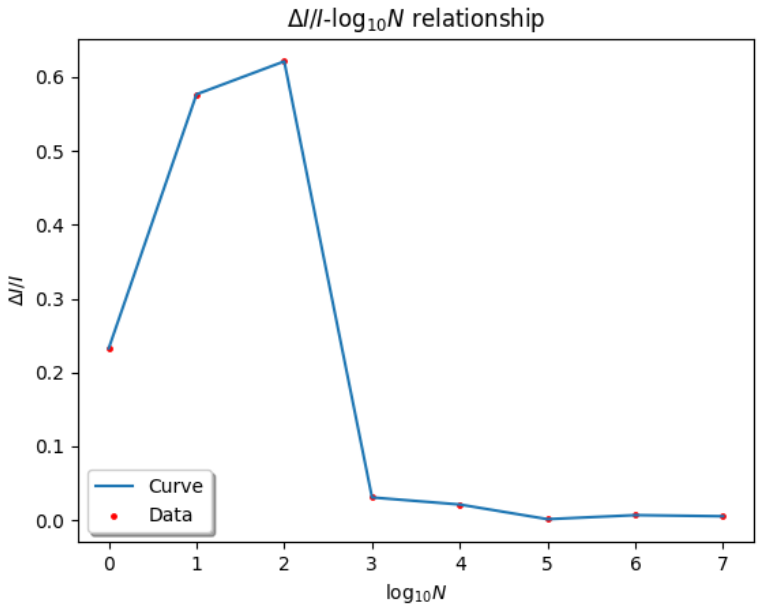

图 6: $\gamma = 1.0568$ 时的误差 δ-$\log_{10} N$ 关系图

$\log_{10} N$	I	δI
1	2.6062664	0.2326187
2	1.2685470	0.5766069
3	1.2336453	0.6212115
4	1.9402071	0.0308178
5	2.0433992	0.0212387
6	2.0027827	0.0013894
7	2.0138058	0.0068556
8	2.0107101	0.0053265

表 1: $\gamma = 1.0568$ 处 $\log_{10} N$ 与 δI 关系表

发现于 $\gamma = 1.0568$ 处，当 $\log_{10} N = 4$ 时，误差就已经很小了；当 $\log_{10} N = 6$ 时，误差已经最接近最小值了；而当 $\log_{10} N > 7$ 时，误差反而变大，这可能是计算机内部的浮点数计算误差所导致的。

为了进行比较，于其他 γ 处任意取了一点，这里取 $\gamma = 1.4786$。

$\log_{10} N$	I	δI
1	2.6062664	0.2326187
2	1.1859661	0.6863888
3	1.2618274	0.5850028
4	1.4888386	0.3433290
5	1.5359529	0.3021233
6	1.5302814	0.3069492
7	1.5343446	0.3034882
8	1.5355747	0.3024440

表 2: $\gamma = 1.4786$ 处 $\log_{10} N$ 与 δI 关系表

发现当 $\gamma = 1.4786$ 处时，当 $\log_{10} N = 4$ 时，误差已经几乎保持不变了。

结论

综合以上 2 个 γ 值推测：在初始点为 $x_0 = 1$ 处，链长 $N = 10^4$ 量级时，Markov Chain 就已经达到了稳定的状态，且与 γ 的取值可能无关。

推测

于不同的初始点 x_0 时，Markov Chain 达到稳定值时的链长是不一样的。

5.3 讨论不同热化系数 k 对结果的影响

下面进行不同热化系数 k 对结果的影响 ($m = N \times k$)

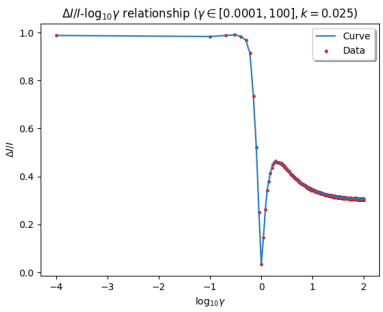

图 7: $\gamma \in [0.0001, 100]\,, N = 10^6, k = 0.025$

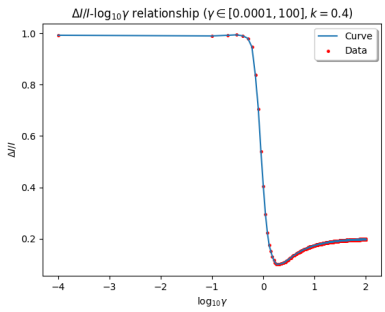

图 8: $\gamma \in [0.0001, 100]\,, N = 10^6, k = 0.4$

下面分别对它们的最小值部分进行放大观察：

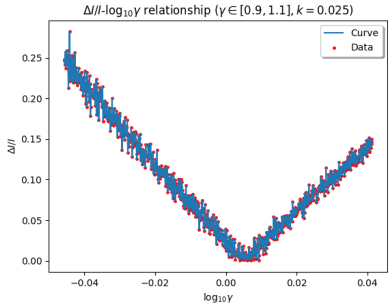

图 9: $\gamma \in [0.9, 1.1]\,, N = 10^6, k = 0.025$

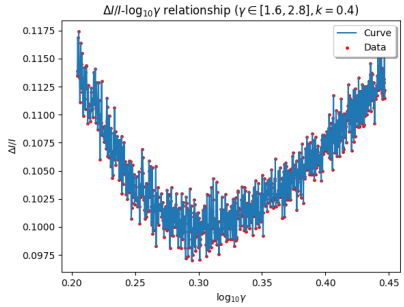

图 10: $\gamma \in [1.6, 2.8]\,, N = 10^6, k = 0.4$

发现，当 $k = 0.025$ 时，误差最小时 $\gamma = 1.0164$；当 $k = 0.4$ 时，误差最小时 $\gamma \in [1.67, 2.57]$ 可见，但热化系数的选择不同的时候，最佳的 γ 值也是在响应改变的。

5.4 讨论不同初始位置 x_0 对结果的影响

下面进行不同初始位置 x_0 对结果的影响。（此时取热化系数为 $k = 0.1$，链长 $N = 10^6$）

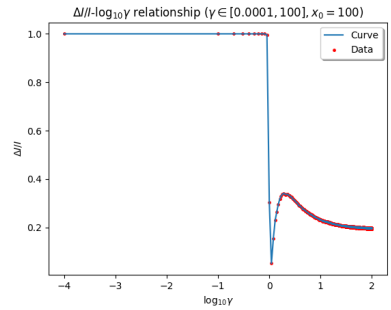

图 11: $\gamma \in [0.0001, 100]\,, N = 10^6, x_0 = 100$

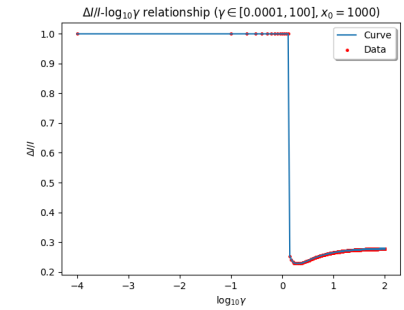

图 12: $\gamma \in [0.0001, 100]\,, N = 10^6, x_0 = 1000$

下面分别对它们的最小值部分进行放大观察：

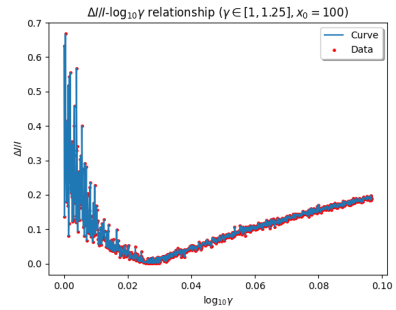

图 13: $\gamma \in [1, 1.25]\,, N = 10^6, x_0 = 100$

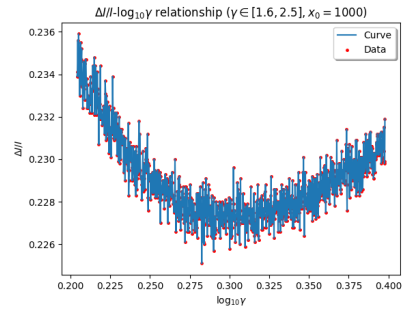

图 14: $\gamma \in [1.6, 2.5]\,, N = 10^6, x_0 = 1000$

可以发现：

1. 在 $x_0 = 100$ 时，步数相同的情况下，其 γ-$\log_{10} N$ 的图像与 $x_0 = 1$ 类似，且最小值位于 $\gamma = 1.0638$，也与 x_0 类似。

2. 在 $x_0 = 1000$ 时，在步数相同的情况下，其 γ-$\log_{10} N$ 的图像形状已经改变，且最小值位置也发生了改变，变为 $\gamma \in [1.7, 2.3]$。推测时由于在 $x_0 = 1000$ 的情况下，其步数为 $N = 10^6$ 不能让 Markov Chain 达到一个稳定值。

6 总结

1. 取值 $\alpha = 2, \beta = 1, k = 0.1, N = 10^6$ 时，最佳的 $\gamma = 1.0568$；且对于不同的 γ 值，Markov Chain 到达稳定值所需的步数 $\sim 10^4$ 量级。此处最佳的 $\gamma = 1 + \varepsilon \sim \beta + \varepsilon$，即要求 $T(x \to x') = T(x') \sim f(x)$，即 $T(x')$ 的形状需要与 $f(x)$ 的形状相似才能使效率更高。

2. 对于不同的热化系数或热化步长，最佳的 γ 会发生改变。

3. 不同的初始位置 x_0 也会对结果产生影响，当 x_0 远离峰值的时候，需要更长链长的 Markov Chain 才能使得其稳定于稳定值附近。

4. 通过本次学习对 $Metropolis$ 方法有了更深入的认识。