1

Назовем турниром ориентированный граф $G = (V, E)$ такой, что $(x, x) \in E$ для любой вершины $x \in V$, а для любых двух различных вершин $x \neq y$, $x, y \in V$ либо $(x, y) \in E$, либо $(y, x) \in E$. Множество вершин назовем игроками, каждая пара игроков ровно один раз встречаются на матче, если игрок x выигрывает у игрока y, то $(x, y) \in E$. Гамильтоновым путем графа назовем перестановку вершин $(x_1, x_2, \dots, x_n)$, что для всех i игрок x_i выигрывает у x_{i+1}. Покажите, что найдется такой турнир на n вершинах, для которого число гамильтоновых путей не меньше чем $n!/(2^{n-1})$.

Решение:
Пусть имеет место случайный ориентированный граф (турнир), то есть направление каждого ребра выбирается независимо от других с вероятностью 1/2. Это в свою очередь означает, что игрок x выигрывает у игрока y с этой же вероятностью 1/2.
Рассмотрим общее число гамильтоновых путей N в построенном турнире. Данная величина является суммой индикаторных событий $\mathbb{I}_\pi$ того, что турнир содержит гамильтонов путь с перестановкой π. Оценим среднее число таких путей N:

$$N = \sum_{\{\pi\}} \mathbb{I}_\pi, \quad \Rightarrow \mathbb{E}\{N\} = \sum_{\{\pi\}} \mathbb{E}\{\mathbb{I}_\pi\}.$$

Общее число перестановок π есть $n!$, а $\mathbb{E}\{\mathbb{I}_\pi\} = \prod_{j=1}^{n-1} \frac{1}{2} = 2^{-(n-1)}$, поскольку в пути есть $n-1$ ребро, а вероятность «наличия» каждого ребра в пути равна 1/2.
Окончательно получаем, что $\mathbb{E}\{N\} = \dfrac{n!}{2^{n-1}}$, что и означает существование такого турнира (случайного по построению), в котором число гамильтоновых путей не меньше чем $n!/(2^{n-1})$.

2

Показать, что распределение Бернулли с параметром $\theta \in (0, 1)$ относится к экспоненциальному классу распределений.

Решение:
Для распределения Бернулли имеем pmf: $f(\theta, k) = \theta^k (1-\theta)^{1-k},\ k \in \{0, 1\}$.

$$f(\theta, k) = \theta^k (1-\theta)^{1-k} = \exp\left[\log\left(\theta^k (1-\theta)^{1-k}\right)\right] = \exp\left[k \log\theta + (1-k)\log(1-\theta)\right] =$$
$$= \exp\left[k\left(\log\theta - \log(1-\theta)\right)\right] \cdot \exp\left[\log(1-\theta)\right] = (1-\theta)\exp\left[\log\left(\frac{\theta}{1-\theta}\right) k\right].$$

В обозначениях параметра $v = \log\left(\frac{\theta}{1-\theta}\right)$, $\theta = \frac{e^v}{1+e^v}$ получаем

$$f(v, k) = \frac{e^{vk}}{1+e^v} = \frac{1}{h(v)} e^{vk},$$

что доказывает принадлежность распределения Бернулли к экспоненциальному классу.

3

Плотность распределения Коши имеет вид

$$p(x|\theta) = \frac{1}{\pi(1 + (x-\theta)^2)}.$$

- Подсчитать информацию Фишера $\mathcal{I}(\theta)$ для такого распределения. Какая проблема у оценки $\frac{1}{n}\sum_{i=1}^{n} x_i$ медианы распределения θ?
- Для этого распределения нельзя явно вычислить оценку максимального правдоподобия. Поэтому нужно использовать численное моделирование. Необходимо построить процедуру для получения оценки максимального правдоподобия для медианы распределения Коши и выписать нижнюю оценку Рао-Крамера.
- Оценить численно дисперсию получаемой оценки максимального правдоподобия и сравнить с нижней оценкой Рао-Крамера.

Решение:

$$\mathcal{I}(\theta) = -\mathbb{E}_\theta \left\{ \frac{\partial^2 \ln f(\mathcal{X}, \theta)}{\partial \theta^2} \right\} = -\int \left(\frac{\partial^2 \ln f(x, \theta)}{\partial \theta^2} \right) f(x, \theta) dx = -\int \left(\frac{\partial^2 \ln p(x|\theta)}{\partial \theta^2} \right) p(x|\theta) dx =$$
$$= -\frac{2}{\pi} \int\limits_{-\infty}^{+\infty} \frac{(x-\theta)^2 - 1}{(1+(x-\theta)^2)^3} dx = \Big/ \xi = x - \theta \Big/ = -\frac{2}{\pi} \int\limits_{-\infty}^{+\infty} \frac{\xi^2 - 1}{(\xi^2+1)^3} d\xi = -\frac{4}{\pi} \int\limits_{0}^{+\infty} \frac{\xi^2 - 1}{(\xi^2+1)^3} d\xi =$$
$$= -\frac{4}{\pi} \int\limits_{0}^{+\infty} \frac{d\xi}{(\xi^2+1)^2} + \frac{8}{\pi} \int\limits_{0}^{+\infty} \frac{d\xi}{(\xi^2+1)^3} = \Big/ \xi = \tan v \Big/ = -\frac{4}{\pi} \int\limits_{0}^{\pi/2} \cos^2 v dv + \frac{8}{\pi} \int\limits_{0}^{\pi/2} \cos^4 v dv =$$
$$= -\frac{4}{\pi} \int\limits_{0}^{\pi/2} \frac{1}{2}(1+\cos 2v) dv + + \frac{8}{\pi} \int\limits_{0}^{\pi/2} \frac{1}{8}(3 + 4\cos 2v + \cos 4v) dv = \left(-\frac{4}{2\pi} + \frac{3}{\pi} \right) \frac{\pi}{2} = \frac{1}{2}.$$

Поскольку распределение Коши обладает бесконечной дисперсией, среднее значение выборки не будет уменьшаться с ростом n и будет становиться все более изменчивым по мере увеличения числа наблюдений, так как увеличивается вероятность встретить точки выборки с большим абсолютным значением. Распределение выборочного среднего будет равно распределению самих наблюдений; то есть выборочное среднее большой выборки оценивает медиану исходного распределения также как любое единичное наблюдение из выборки.

Для оценки максимума правдоподобия имеем следующие соотношения:

$$\hat{\ell} = \hat{\ell}(\mathbf{x}|\hat{\theta}_n) = -n \log \pi - \sum_{i=1}^{n} \log \left(1 + (x_i - \hat{\theta}_n)^2 \right),$$
$$\frac{d\hat{\ell}}{d\hat{\theta}_n} = 2 \sum_{i=1}^{n} \frac{x_i - \hat{\theta}_n}{1 + (x_i - \hat{\theta}_n)^2}, \quad \frac{d^2\hat{\ell}}{d\hat{\theta}_n^2} = -2 \sum_{i=1}^{n} \frac{1 - (x_i - \hat{\theta}_n)^2}{(1 + (x_i - \hat{\theta}_n)^2)^2}, \quad \frac{d^3\hat{\ell}}{d\hat{\theta}_n^3} = 4 \sum_{i=1}^{n} \frac{(x_i - \hat{\theta}_n)^3 - 3(x_i - \hat{\theta}_n)}{(1 + (x_i - \hat{\theta}_n)^2)^3}.$$

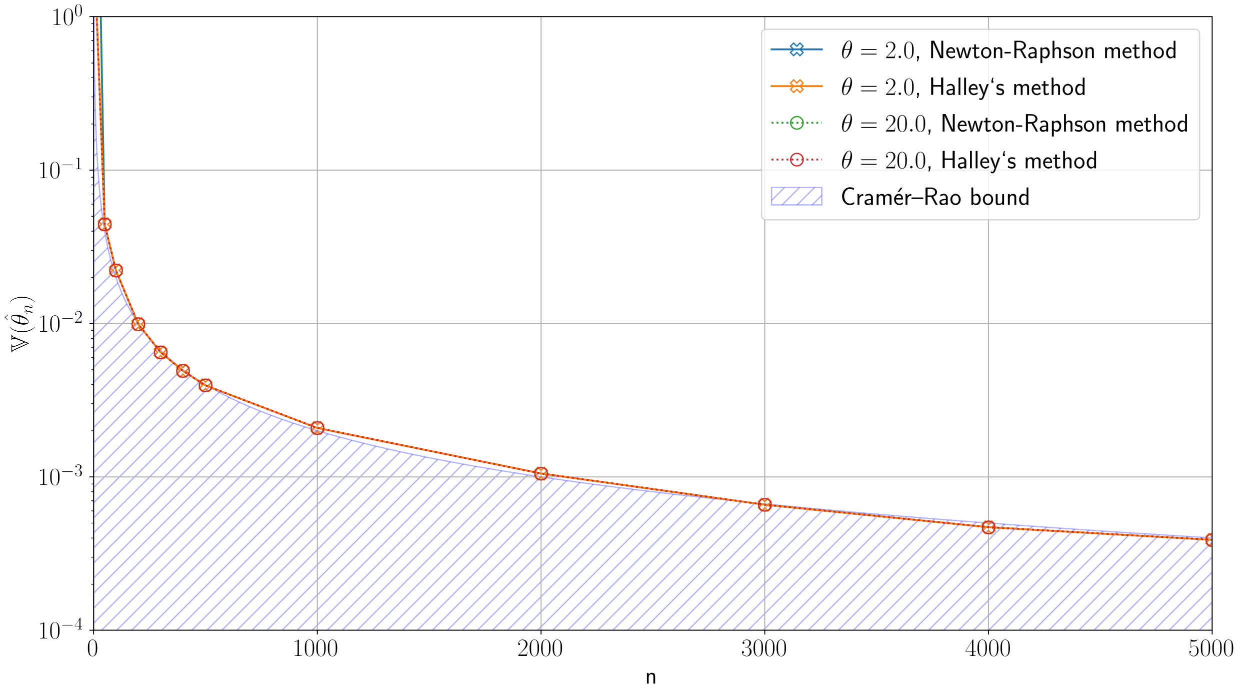
Для того чтобы максимизировать $\hat{\ell} = \hat{\ell}(\mathbf{x}|\hat{\theta}_n)$, можно найти стационарные точки целевой функции $g(\hat{\theta}_n) = \dfrac{d\hat{\ell}}{d\hat{\theta}_n}$, у которой известны аналитические выражения для производных. Это даёт возможность воспользоваться итеративными методами поиска нулей функций, например:

- Newton-Raphson method: $\theta_{k+1} = \theta_k - \dfrac{g(\theta_k)}{g'(\theta_k)}$;
- Halley's method: $\theta_{k+1} = \theta_k - \dfrac{g(\theta_k)}{g'(\theta_k)} \left[1 - \dfrac{g(\theta_k)}{g'(\theta_k)} \cdot \dfrac{g''(\theta_k)}{2g'(\theta_k)} \right]^{-1}$.

Предлагается использовать имплементации этих алгоритмов из *scipy.optimize*. В качестве начальных приближений можно использовать значения из выборки. В частности, в дальнейшем используется 1/4 доля центральных уникальных значений выборки, чтобы отбросить тяжёлые хвосты распределения, которые вызывают сильные проблемы со сходимостью алгоритмов. Финальной оценкой считается среднее корней (нулей), который выдал алгоритм в случае удачной сходимости.
Для несмещённой величины $\hat{\theta}_n$ нижняя оценка Рао-Крамера:

$$\mathbb{V}(\hat{\theta}_n) \geq \frac{1}{\mathcal{I}_n(\theta_*)} = \frac{1}{n\mathcal{I}(\theta)} = \frac{2}{n}.$$

Для смещённой величины нижняя оценка будет хуже, но построенные алгоритмы решения дают результаты близкие даже к границе для несмещённой величины. Дисперсия получаемой оценки максимального правдоподобия оценивается по 1000 запускам моделирования для каждого размера выборки n, каждого истинного значения θ и каждого алгоритма.



Представленные результаты демонстрируют, что дисперсия получаемой оценки лежит близко к границе Рао-Крамера. Стоит отметить, что существенных различий в результатах для двух алгоритмов не наблюдается, но второй алгоритм считается дольше из-за большего наличия промежуточных вычислений. Для ознакомления код приведён ниже.

[ ]:
```python
import numpy as np
import pandas as pd
from scipy.stats import cauchy
from scipy import optimize
```

[ ]:
```python
def target(theta, samples):
    return 2 * np.sum((samples - theta) / (1 + (samples - theta)**2))

def der1(theta, samples):
    return -2 * np.sum((1 - (samples - theta)**2) / (1 + (samples - theta)**2)**2)

def der2(theta, samples):
    return 4 * np.sum(((samples - theta)**3 - 3*(samples - theta))/ (1 + (samples -␣
↪theta)**2)**3)
```

[ ]:
```python
num_runs = 1000
theta = 2.

n_set = np.array([10, 20, 30, 40, 50, 100, 200, 300, 400, 500, 1000, 2000, 3000, 4000,␣
↪5000])
dfs = []
for n in n_set:
    for run in range(0, num_runs, 1):
        samples = cauchy(loc=theta, scale=1.).rvs(n, random_state=1+run)
        search_space = sorted(set(samples))[3*n//8: 5*n//8]
        solution = []
        for x0 in search_space:
            try:
                root, r = optimize.newton(
                    target,
                    x0,
                    args=(samples, ),
                    fprime=der1,
                    fprime2=der2, # comment this for pure Newton method
                    tol = 1./n,
                    full_output=True,
                )
                if (r.converged):
                    solution.append(root)
            except:
                continue
        if len(solution) > 0:
            dfs = dfs + [[theta, n, run, np.mean(solution)]]

df = pd.DataFrame(dfs, columns = ['true', 'n', 'run', 'est'])
#df.to_csv('')
```

[ ]:
```python
import matplotlib.pyplot as plt
import matplotlib as mpl

mpl.rcParams.update({'font.size': 20})
fig, ax = plt.subplots(figsize=(16,9))
plt.rcParams.update({
```

```
    'text.usetex': True,
    'text.latex.preamble': r'\usepackage{amsfonts}'
})

mstyles = [dict(linestyle='-', marker='X', markersize=10, fillstyle='none'), dict(linestyle=':', marker='o', markersize=10, fillstyle='none')]
algo_names = ['Newton-Raphson method', 'Halley`s method']
for j, theta in enumerate([2.0, 20.0]):
    for algo in [1, 2]:
        df = pd.read_csv('val_{:.1f}_algo{}.csv'.format(theta, algo))
        print(f'true: {theta}, estimation: {np.mean(df.est)}')
        var_set = []
        for n in n_set:
            row = df.query(f'(true == {theta}) and (n == {n})')
            var_set.append(np.var(row.est))
        plt.plot(n_set, var_set, **mstyles[j], label=f'$\\theta = {theta}$, {algo_names[algo-1]}')


plt.fill_between(np.arange(1, 5001), 2./np.arange(1, 5001), alpha = 0.3, label = 'Cramer-Rao bound', facecolor = 'white', edgecolor = 'blue', hatch = '//')
plt.grid()
plt.ylabel('$\mathrm{\mathbb{V}}(\hat{\\theta}_n)$')
plt.xlabel('n')
plt.ylim(1e-4, 1e0)
plt.xlim(0, 5000)
plt.yscale('log')
plt.title('')
plt.legend(ncol = 1)
plt.savefig('pic.png', dpi = 400, bbox_inches='tight')
plt.show()
```

```
true: 2.0, estimation: 2.016354023517402
true: 2.0, estimation: 1.99657529432372
true: 20.0, estimation: 20.016354023517486
true: 20.0, estimation: 19.99657529432372
```