Applied Statistic HW3

2020270026 王姿文、2020211316 周斯萤、2020211314 徐颢轩

2020/11/06

Question 1 矩阵积和式的近似计算。

1.1 利用GG算法、KKLLL算法计算0-1矩阵积和式；并且可以更一般地，将原始矩阵A的非零元素变换为其他期望为0方差为1的随机变量，计算0-1矩阵积和式。

将矩阵设为

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

根据作业附档Nperm(A)得出积和式精确值为2。
可由下表看出不同算法的近似积和式，每个算法都设定模拟1000次：

算法	Estimation
GG	1.968+0i
KKLLL	2.037+0i
期望为0方差为1的随机变量	2.003+0i

1.2 利用随机路径算法近似计算0-1矩阵积和式。

将矩阵设为

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

设定模拟1000次，由下得出随机路径算法近似值=2。近似值与精确值一模一样，故随机路径法比题1.1的三种算法还优。
即便模拟次数为1，随机路径算法计算而得的近似值也一样为2。

1.3 考察矩阵稀疏度、规模与计算效率的相关性。

1. 矩阵稀疏度与计算效率的相关性

   下图横轴为SP(稠密度)，纵轴为 $\frac{E[X_A^2]}{E[X_A]^2}$ （与模拟次数N正相关），而图内的n则为规模。其中模拟次数 $N = (\frac{E[X_A^2]}{E[X_A]^2})(\frac{1}{\epsilon^2}O(ln(\frac{1}{\delta})))$，且模拟次数N越大，代表计算效率越差。且SP(稠密度)的大小与稀疏度大小相反。

   综上所述，能够得出横轴SP(稠密度)越小，代表越稀疏；纵轴 $\frac{E[X_A^2]}{E[X_A]^2}$ 越大，计算效率越差。[1]

   KKLLL方法在矩阵稀疏时表现最好，而RP方法在矩阵相对稠密时表现最好。此外，RP对稀疏度最敏

感，而KKIII最不敏感。

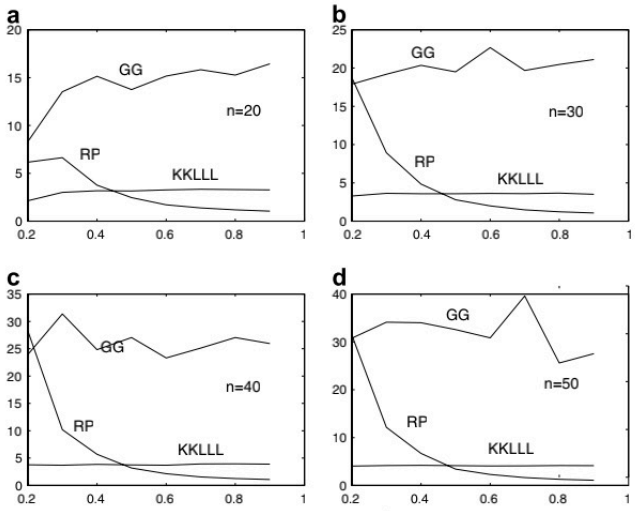


Fig. 5.1. The average of $\frac{E[\widehat{X_A^2}]}{E[X_A]^2}$ vs SP.

图片来源：Random path method with pivoting for computing permanents of matrices

2. 规模与计算效率的相关性

下图横轴为n(规模)，纵轴为 $\frac{E[\widehat{X_A^2}]}{E[X_A]^2}$ （与模拟次数N正相关），而图内的SP则为稠密度。其中模拟次数 $N = (\frac{E[X_A^2]}{E[X_A]^2})(\frac{1}{\epsilon^2}O(ln(\frac{1}{\delta})))$，且模拟次数N越大，代表计算效率越差。

明显可以看出n(规模)越大，纵轴 $\frac{E[\widehat{X_A^2}]}{E[X_A]^2}$ 越大，计算效率越差。

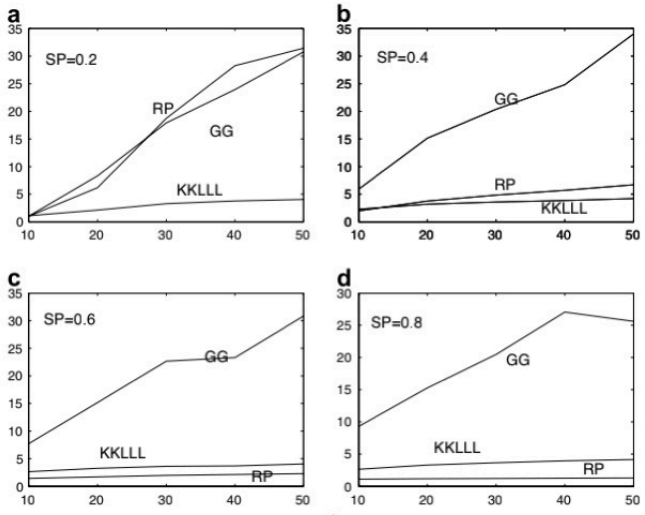


Fig. 5.2. The average of $\frac{E[\widehat{X_A^2}]}{E[X_A^2]}$ vs the size of matrices.

图片来源：Random path method with pivoting for computing permanents of matrices

1.4 比较各种算法的计算效率。

首先计算出随机路径算法的计算效率为$O(n^2)$、GG算法的计算效率为$3^{n/2}\frac{1}{\epsilon}log(\frac{1}{\delta}poly(n))$、KKLLL的计算效率为$2^{n/2}\frac{1}{\epsilon}log(\frac{1}{\delta})poly(n)$。

此计算效率公式为设定同样的误差条件下（误差 $< \epsilon$），达到同样的误差前所花费的计算时间。

因为$O(n^2) < 2^{n/2}\frac{1}{\epsilon}log(\frac{1}{\delta}poly(n)) < 3^{n/2}\frac{1}{\epsilon}log(\frac{1}{\delta})poly(n)$，所以**计算效率优劣为随机路径算法优于KKLLL算法优于GG算法。**

下图来自于文献，也可以证实计算效率优劣为随机路径算法优于KKLLL算法优于GG算法。

Table 5.1
Comparison of RP, KKLLL and GG in accuracy, $n = 15$, $N = 750$

SP	|RE| < 0.05			|RE| < 0.10			|RE| < 0.15		
	RP	KKLLL	GG	RP	KKLLL	GG	RP	KKLLL	GG
0.3	49	46	26	79	80	50	93	93	76
0.4	66	48	28	96	84	47	100	97	67
0.5	77	48	27	100	76	55	100	95	70
0.6	92	48	23	100	75	45	100	94	65
0.7	96	38	19	100	72	50	100	92	79

Table 5.2
Comparison of RP, KKLLL and GG in accuracy, $n = 20$, $N = 1000$

SP	|RE| < 0.05			|RE| < 0.10			|RE| < 0.15		
	RP	KKLLL	GG	RP	KKLLL	GG	RP	KKLLL	GG
0.3	52	46	23	79	78	48	92	93	70
0.4	59	45	27	92	81	46	98	96	70
0.5	83	43	25	99	83	47	100	95	65
0.6	94	42	18	100	76	38	100	94	64
0.7	100	49	25	100	80	40	100	94	67

图片来源：Random path method with pivoting for computing permanents of matrices

---

Question 2 实现第7讲中关于随机排列重要度抽样的实例，并且适当改变算法中参数设置，以及将目标中的290000换为其他值，比较计算效率。

若 $X = (X_1, X_2, \dots, X_{100})$ 是一个随机排列，也就是说X等可能地等于$100!$个排列中的任意一个。估计概率为 $\theta = P\sum_{j=1}^{100} j * X_j >$ 目标。

$i_1 < i_2, j_1 < j_2, i_1j_1 + i_2j_2 - (i_1j_2 + i_2j_1) = (i_2 - i_1)(j_2 - j_1) > 0 \Rightarrow i_1j_1 + i_2j_2 > i_1j_2 + i_2j_1$

则具体的模拟步骤为：

先以各自的参数 $\lambda_j = j^{参数}$生成独立的指数分布随机变量Y_j，$j = 1, 2, \dots, 100$；

再将Y_j排序，令X_k等于位次第k位的 Y_j的下标； 若$\sum_{j=1}^{100} j * X_j >$ 目标，则令$I = 1$，否则令其为0。

因此由$Y_{X_1} > Y_{X_2} > \dots > Y_{X_{100}}$得到排列$X = (X_1, X_2, \dots, X_{100})$的概率为

$$\frac{(X_{100})^{参数}}{\sum_{j=1}^{100}(X_j)^{参数}} * \frac{(X_{99})^{参数}}{\sum_{j=1}^{99}(X_j)^{参数}} * \dots * \frac{(X_2)^{参数}}{\sum_{j=1}^{2}(X_j)^{参数}} * \frac{(X_1)^{参数}}{\sum_{j=1}^{1}(X_j)^{参数}},$$

因此单个抽样的重要度估计为

$$\hat{\theta} = \frac{I}{100!} * \frac{\prod_{n=1}^{100}(\sum_{j=1}^{n}(X_j)^{参数})}{(\prod_{n=1}^{100} n)^{参数}}$$

下表为模拟次数为10000，目标分别290000;300000，参数=0.7分别为0.3的估计值和效率比较。

在参数=0.7时，由于$\sum_{j=1}^{100} j * X_j$的sample mean为286159(目标为290000)或288902(目标为300000)，因此目标数离sample mean越大，其$\hat{\theta}$越小。

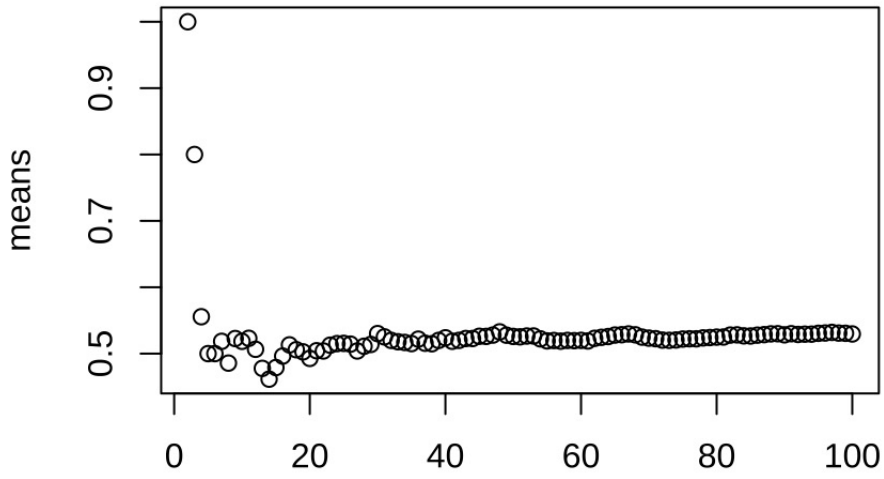
在参数=0.3时，由于$\sum_{j=1}^{100} j * X_j$的sample mean为278716(目标为290000)或275369(目标为300000)，因此目标数离sample mean越大，其$\hat{\theta}$越小。

而计算效率的部分，可以看出目标数及参数的改变均不太影响计算时间。

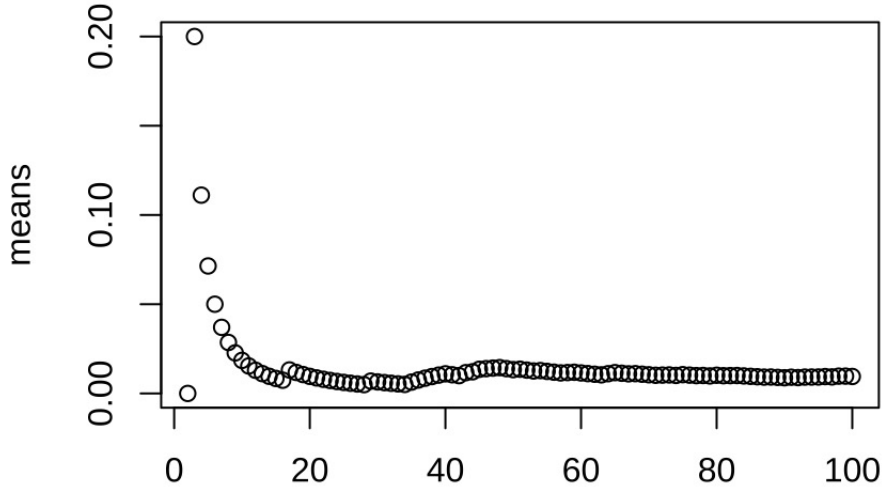
目标	参数	Estimation	Mean
290000	0.7	0.5283	286159
300000	0.7	0.0659	288902
290000	0.3	0.0104	278716
300000	0.3	0.0001	275369

在此题中，我们改变指数λ的取值，分别看λ取0.3和0.7时的情况。在不同λ下，我们通过绘制样本均值和模拟次数的关系图，发现λ取0.7时，模拟次数为40时样本均值就已经趋于稳定，说明此时样本均值已经逐渐收敛于真值；而当λ取0.3时，模拟次数达到60时，样本均值趋于稳定。从而我们分析得出，λ等于0.7时收敛速度更快。 同时，我们固定λ等于0.7不变，分别看阈值为290000和300000时的情况。我们发现，当阈值取290000时，模拟次数为40时样本均值趋于稳定；而阈值取300000时，模拟次数为20时样本均值就已经趋于稳定。因此我们分析得出，阈值取300000时收敛速度更快。

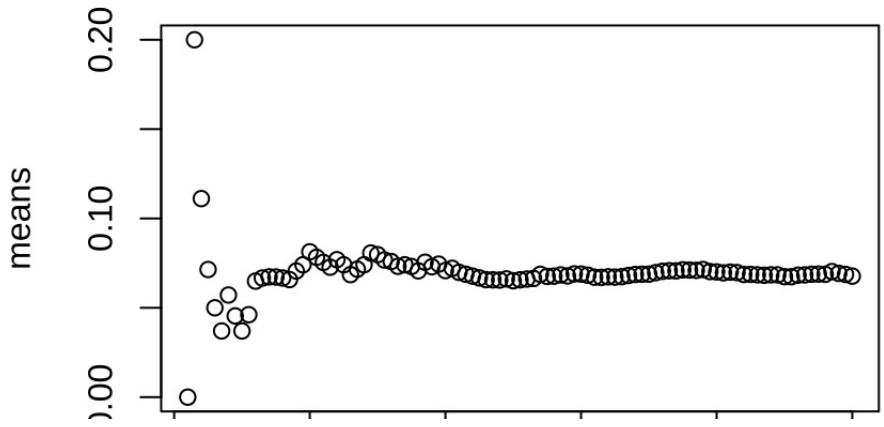
参数=0.7、目标=290000时均值变化图



参数=0.3、目标=290000时均值变化图



参数=0.7、目标=300000时均值变化图



參考文獻

[1] Heng Liang, Linsong Shi, Fengshan Bai, Xiaoyan Liu. Random path method with pivoting for computing permanents of matrices . Applied Mathematics and Computation 185 (2007) 59–71.