计算物理学作业 1 | 解答

喵

2018 年 9 月 28 日

1. **数据误差的避免**

   (a) 设舍入误差为 ϵ（相对意义下），

$$\begin{aligned} x_1 \oplus x_2 \oplus \cdots \oplus x_N &= (x_1+x_2)(1+\frac{\epsilon_M}{2}) \oplus x_3 \oplus \cdots \oplus x_N \\ &= (\cdots((x_1+x_2)(1+\frac{\epsilon_M}{2})+x_3)(1+\frac{\epsilon_M}{2})\cdots+x_N)(1+\frac{\epsilon_M}{2}) \\ &\approx (x_1+x_2+\cdots+x_N)+\frac{\epsilon_M}{2}\cdot[(N-1)x_1+(N-1)x_2+(N-2)x_3+\cdots+x_N] \end{aligned}$$

   最后一步略去了 $\frac{\epsilon_M}{2}$ 的高阶项.

$$\epsilon = \frac{\frac{\epsilon_M}{2}\cdot[(N-1)x_1+(N-1)x_2+(n-2)x_3+\cdots+x_N]}{x_1+x_2+\cdots+x_N}$$

   为了取到上限，取 $x_i = \delta_{1i}\cdot x_0$：

$$\max\epsilon = \frac{\epsilon_M}{2}\cdot(N-1)$$

   .

   (b) 结论：第二个公式更加稳定和准确.
   NOTE：这里两个公式处理的问题相同，病态性没有差别.

      i. 第一个公式需要计算 $\sum_{i=1}^{N} x_i^2$，这是一个相当大的数，可能导致数据上溢.

      ii. 第一个公式相加的加数大，舍入误差更大.

   (c) 首先证明公式（4），

$$I_0 = \int_0^1 \frac{dx}{x+5} = \ln|x+5|\Big|_0^1 = \ln(6/5).$$

$$I_k + I_{k+1} = \int_0^1 x^{k-1}dx = \frac{1}{k}.$$

   根据上面的递推公式计算 $n>>1$ 时的 I_n 不是稳定的. 每次迭代误差会扩大五倍，最终给 I_n 带来相当于 $\epsilon\cdot 5^n$ 的误差. 当 $n>>1$ 时，这个误差是不能容忍的.

2. **矩阵的模和条件数**

$$A = \begin{bmatrix} 1 & -1 & \cdots & -1 \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & 1 \end{bmatrix}.$$

   (a) 根据定义，$|A| = 1$. 显然，A 不是奇异矩阵.

(b) 对 $[A, E]$ 进行初等行变换，得到 $[E, A^{-1}]$. 过程技巧性不高，不做展示.（写起来太累了 orz）结果如下，

$$A^{-1} = \begin{bmatrix} 1 & 1 & 2 & \cdots & 2^{n-3} & 2^{n-2} \\ 0 & 1 & 1 & 2 & \cdots & 2^{n-3} \\ 0 & 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 & 1 & 2 \\ \vdots & \ddots & \ddots & 0 & 1 & 1 \\ 0 & \cdots & \cdots & 0 & 0 & 1 \end{bmatrix}.$$

(c) 令 $x = (x_1, x_2, \cdots, x_n)^T$. 并无妨假设 $\|x\|_\infty = 1$. 根据定义，$\max\{|x_i|\} = 1$. 即：$|x_i| \le 1, i = 1, 2, \cdots, n$.

$Ax = (a_{1k}x_k, a_{2k}x_k, \cdots, a_{nk}x_k)^T$**（这里我们使用 Einstein 求和规则）**

$$\sup_{x \neq 0} \frac{\|Ax\|_\infty}{\|x\|_\infty} = \sup_{\|x\|_\infty = 1} \max_i a_{ij}x_j$$

考虑到 $a_{ij}x_j \le \sum_{j=1}^n |a_{ij}|$, 等号取到当且仅当 $x_j = 1 \cdot sgn(a_{ij})$. 从而得到最后的结论：

$$\sup_{x \neq 0} \frac{\|Ax\|_\infty}{\|x\|_\infty} = \max_{i=1,\cdots,n} \sum_{j=1}^n |a_{ij}|$$

(d) 根据定义，$U^\dagger U = UU^\dagger = I_n$.

$$\|U\|_2 = \sup_{x \neq 0} \frac{\|Ux\|_2}{\|x\|_2} = \sup_{x \neq 0} \frac{(Ux)^\dagger Ux}{x^\dagger x} = \sup_{x \neq 0} \frac{x^\dagger (U^\dagger U) x}{x^\dagger x} = 1.$$

同理，$\left\|U^\dagger\right\|_2 = 1$.

$\forall A \in \mathbb{C}^{n \times n}$,

$$\|UA\|_2 = \sup_{x \neq 0} \frac{\|UAx\|_2}{\|x\|_2} = \sup_{x \neq 0} \frac{(UAx)^\dagger UAx}{x^\dagger x} = \sup_{x \neq 0} \frac{(Ax)^\dagger Ax}{x^\dagger x} = \|A\|_2.$$

据此，就像题目所述，利用欧式模定义条件数，$K_2(A) = K_2(UA)$.

(e) $K_\infty(A) = \|A\|_\infty \left\|A^{-1}\right\|_\infty = n \cdot 2^{n-1}$.

3. **Hilbert 矩阵**

(a) D 取到极小值，必要条件为：对于任意的 $j = 1, \cdots n$,

$$\frac{\partial D}{\partial c_i} = \int_0^1 2x^{i-1}(\sum_{j=1}^n c_j x^{j-1} - f(x))dx = 0.$$

这里调整了一下求和的赝指标.

$$0 = \int_0^1 \sum_{j=1}^n c_j x^{i+j-2} dx - \int_0^1 f(x)x^{i-1}dx = \sum_{j=1}^n \frac{c_j}{i+j-1} - \int_0^1 f(x)x^{i-1}dx.$$

$$\sum_{j=1}^n \frac{c_j}{i+j-1} = \int_0^1 f(x)x^{i-1}dx.$$

和 (8) 式进行对比：

$$(H_n)_{ij} = \frac{1}{i+j-1}, \quad b_i = \int_0^1 f(x)x^{i-1}dx.$$

(b)

$$P_n(x) = \sum_{i=1}^{n} c_i x^{i-1} = c^T \cdot \begin{bmatrix} 1 \\ x \\ \vdots \\ x^{n-1} \end{bmatrix}.$$

$$[P_n(x)]^2 = c^T \begin{bmatrix} 1 \\ x \\ \vdots \\ x^{n-1} \end{bmatrix} \begin{bmatrix} 1 & x & \cdots & x^{n-1} \end{bmatrix} c.$$

$$\int_0^1 [P_n(x)]^2 dx = c^T H_n c \geq 0.$$

等号取到当且仅当 $P_n(x) \equiv 0, \quad x \in [0,1]$. 即，$c = 0$.
H_n 明显是对称的，所以它是对称正定的矩阵. 正定矩阵的行列式大于 0. H_n 是非奇异的.

(c)

$$p = \log_{10} det(H_n) = \log_{10} \frac{{c_n}^4}{c_{2n}} = 4 \sum_{i=1}^{n-1} \log_{10} c_i - \sum_{i=1}^{2n-1} \log_{10} c_n$$

用程序计算 $n = 1, \cdots, 10$ 的 p 的数值解, 保留小数点后一位, 得到下表:

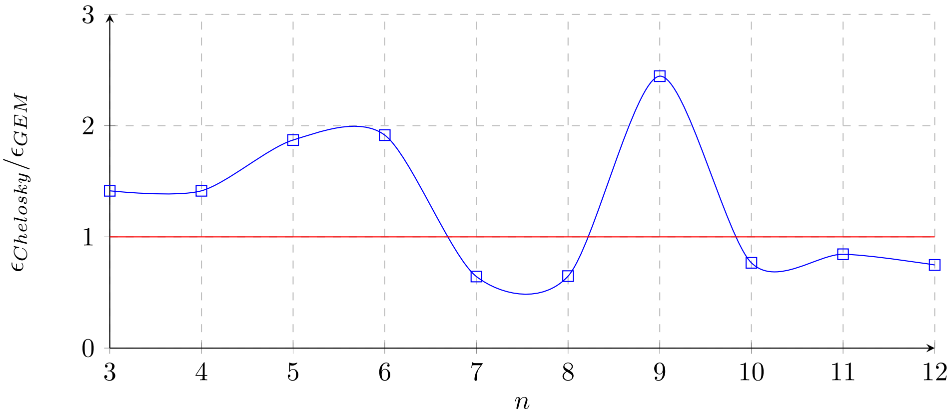
n	1	2	3	4	5	6	7	8	9	10
p	0	-1.1	-3.3	-6.8	-11.4	-17.3	-24.3	-32.6	-42.0	-52.7

相应的 $det(H_n) \approx 10^p$. 可以看到 $det(H_n)$ 随 n 增加而迅速减小. n 较小时增长速率快于指数，n 较大时稳定为指数增长.

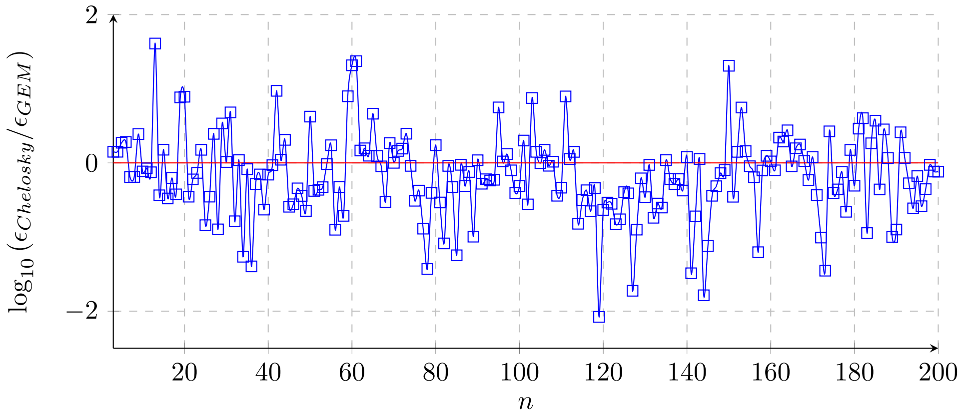
(d)
- 源代码说明
  - 主程序为 main.py. 运行则计算并打印出 $n = 1, \cdots, 20$ 时两种方法给出的解的向量表示 $x_\alpha, \quad \alpha = \text{GME, Chelosky}$，解之间的**相对偏差** δ, 两种方法解的**误差** ϵ_α 以及二者的比值. 加粗的物理量定义如下:

$$\delta = \frac{\|x_{GEM} - x_{Cholesky}\|_2}{\|x_{Cholesky}\|_2}, \quad \epsilon_i = \frac{\|H_n x_i - b\|_2}{\|b\|_2}.$$

  - 主程序导入两个模块：GEM 解方程的模块是 GEM.py, Cholesky 分解的模块是 Cholesky.py.
- 根据输出的结果，两种方法给出的解是有差别的. 差别由相对偏差 δ 体现（主程序打印的"Relative Deviation"）.
  计算表明，ϵ 随着 n 的增加约呈指数增长：n 每增加 1，ϵ 扩大约 1 个数量级. $\epsilon\big|_{n=10} \approx 5 \cdot 10^{-5}$，此时二者还可以认为近似相等. 但如果 n 继续变大，不久后二者的约等于关系就不复存在.
- 精确度是不确定的.
  - 理论上, 更少的计算量让 Cholesky 分解产生更小的误差. Cholesky 分解的计算量约为 $\frac{1}{3}n^3$；而 GEM 的计算量约为 $\frac{2}{3}n^3$，是前者的两倍.
  - 然而实际计算没有完全支持这个结论. 下图展示了 $n = 3, \cdots 12$ 时, $\frac{\epsilon_{Chelosky}}{\epsilon_{GEM}}$ 的数值. (n 从 3 开始是由于这时两个方法都展示出了可见的误差, n 到 12 结束是由于之后 Chelosky 分解的结果会出现很小的虚部.) 我们看到当 n 比较小的时候, 二者的误差并没有一致的大小关系.



我们扩大 n 的范围至 200. 出现虚部时, 在计算 $\epsilon_{Chelosky}$ 时取模. 计算得到下图.



从图中, 我们可以观察到这个范围下 Chelosky 算法相对 GEM 具有优势, 但并不绝对. 经过统计, $n = 3, \cdots 200$ 时, 有 72 个值 Chelosky 分解误差更大, 其余 126 个 Chelosky 分解误差均更小.