数理解析研究所講究録
986巻 1997年 185-194

一般逆行列のmodulo計算アルゴリズムとその効率

齊藤敏明 (工学院大学・情報) 牧野潔夫 (工学院大学・数学)

1. はじめに

我々は数式処理用の線形代数の各種計算 [1] のアルゴリズムの実現を目指している。工学の分野では、一般逆行列をはじめとする線形代数計算が不可欠である [5]。

また、工学で扱われる行列のサイズ・行列成分の桁数は莫大なものも多く、この点を考慮する必要がある。一般逆行列計算においてmoduloによる方法と割算による方法との効率を比較して、それぞれを使い分けたという報告があまりないように思われる。

そこで本報告では、線形代数計算のうち一般逆行列を扱い、moduloによる計算と割算によるものとの効率を比較する。

2. 定義

はじめに以下で用いるものを列記する。(m,n) 型行列 A に対して、${}^tA$ は A の転置行列、$\mathrm{rank}(A)$ は A のrankをあらわす。また、E は単位行列、O は零行列をあらわす。

定義 1 (Moore-Penrose逆行列,[2],p.68, 問31) A を任意の (m,n) 型行列としたとき、次の4条件をみたす (n,m) 型行列 A^+ が唯一存在する。

$$AA^+A = A \tag{1}$$

$$A^+AA^+ = A^+ \tag{2}$$

$${}^t(AA^+) = AA^+ \tag{3}$$

$${}^t(A^+A) = A^+A \tag{4}$$

この A^+ をMoore-Penrose逆行列という。また、n 次正方行列 B に対して B が正則ならば、$B^+ = B^{-1}$ となる [4]。

3. modulo計算における一般論

3.1. 中国剰余定理

定理 1 (中国剰余定理,[6],p.42) $m_1, m_2, \ldots, m_t$のどの 2 つも互いに素ならば、任意の $r_1, r_2, \ldots, r_t$に対して連立合同式

$$x \equiv \begin{cases} r_1 \bmod m_1 \\ r_2 \bmod m_2 \\ \quad \cdots \\ r_t \bmod m_t \end{cases} \tag{5}$$

に解は存在する。1 つの解を x_tとすると、一般解は

$$x \equiv x_t \bmod m_1 m_2 \cdots m_t \tag{6}$$

である。

実際にこの定理を利用して具体的に x を求める方法に、Gauss の方法と Garner の方法がある [6]。

3.2. Order-N Farey fractions の定理

定義 2 (order-N Farey fractions,[7],p.27) $N > 0$ なる整数について、整数 a, b が $\gcd(a,b) = 1$ かつ $0 \leq |a| \leq N, 0 < |b| \leq N$ である有理数$\frac{a}{b}$の集合 $\mathbf{F}_N$を order-N Farey fractions という。すなわち、次のように表せる。

$$\mathbf{F}_N = \left\{ \frac{a}{b} \in \mathbf{Q} | a, b \in \mathbf{Z}, \gcd(a,b) = 1, 0 \leq |a| \leq N, 0 < |b| \leq N \right\}$$

定理 2 (order-N Farey fractions の定理,[7],p.27,Theorem) N を整数とすると、

$$2N^2 + 1 \leq M \tag{7}$$

をみたす整数 M に対して、$(f \mod M)$ から $f \in \mathbf{F}_N$ がただ一つ定まる。

定理 3 $(f \bmod M)$ から、$f = \frac{a}{b} \in \mathbf{F}_N$の a, b は Extended Euclid Algorithm[7] で計算される。

定理 4 (m, n) 型行列 A に対して、$A = (a_{ij})$ とし $A^+ = (b_{ij})$ とおくと $b_{ij} \in \mathbf{F}_N$ である。ここで、N は次式をみたす最大の整数となることがわかっている。

$$N \quad \leq \prod_i^m \sqrt{a_{i1}^2 + a_{i2}^2 + \cdots + a_{in}^2} \tag{8}$$

3.3. Hensel 構成 (Newton-Schultz の方法)

定理 4 から A^+を決めるには、$2N^2 + 1 \leq M$ となる M に対して $A^+ \bmod M$ を求めればよい。

Hensel 構成の具体的な方法として、Newton-Schultz の方法がある [7]。任意の素数 p に対して、以下の漸化式より $B_1, B_2, \ldots, B_k, \ldots$ が計算できる (Hermite の方法 ([14] 定理 7) に適用した場合)。

$X=(A^tA)^2$とする。以下X_R^-とはXのreflexive g-inverse[7]を表す。つまり、(1),(2)の2式のみをみたすものである。

$$\begin{cases} B_1 \equiv X_R^- \bmod p \\ B_{2^k} \equiv B_{2^{k-1}}(2E - XB_{2^{k-1}}) \bmod p_{2^k} \ (k=1,2,\ldots) \end{cases} \tag{9}$$

また、

$$p^{2^k} \geq 2N^2+1 \tag{10}$$

をみたす最小のkをlとおく。これから、B_{2^l}がわかれば定理3によりX_R^-が決定できて、$A^+ = {}^tAX_R^-A^tA$よりA^+が求められる。また、Aが正則のときは(9)式は

$$\begin{cases} B_1 \equiv A^{-1} \bmod p \\ B_{2^k} \equiv B_{2^{k-1}}(2E - AB_{2^{k-1}}) \bmod p_{2^k} \ (k=1,2,\ldots) \end{cases} \tag{11}$$

となり、A^+の場合と同様にして求められる(具体例参照)。

4. 具体例

サンプル行列として、以下の行列を用いる($\mathrm{rank}(A)=3$, $\mathrm{rank}(C)=2$)。

$$A = \begin{bmatrix} 2 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 2 & 0 \end{bmatrix},\ C = \begin{bmatrix} -1 & 2 & 1 \\ -1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{12}$$

4.1. 逆行列の計算例

Faddeevの修正法[3][10]とNewton-Schultzの方法を用いた場合について解説する。

4.1.1. 割算を用いたFaddeevの修正法

以下のB_1, B_2と$q_1=1, q_2=4, q_3=-4$よりA^{-1}が求められる。

$$B_1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & -2 & 1 \\ 0 & 2 & -1 \end{bmatrix},\ B_2 = \begin{bmatrix} -2 & 2 & 1 \\ 0 & 0 & -2 \\ 0 & -4 & -2 \end{bmatrix},\ A^{-1} = \frac{B_2}{q_3} = \frac{1}{4}\begin{bmatrix} 2 & -2 & -1 \\ 0 & 0 & 2 \\ 0 & 4 & 2 \end{bmatrix} \tag{13}$$

4.1.2. 中国剰余定理を用いたFaddeevの修正法

まず(8)式から$N=6$となり、(7)式をみたすMとして$M=5\cdot 7\cdot 11$を用いる。定理1より

$$A^{-1} \equiv \begin{bmatrix} 3 & 2 & 1 \\ 0 & 0 & 3 \\ 0 & 1 & 3 \end{bmatrix} \bmod 5,\ A^{-1} \equiv \begin{bmatrix} 4 & 3 & 5 \\ 0 & 0 & 4 \\ 0 & 1 & 4 \end{bmatrix} \bmod 7,\ A^{-1} \equiv \begin{bmatrix} 6 & 5 & 8 \\ 0 & 0 & 6 \\ 0 & 1 & 6 \end{bmatrix} \bmod 11 \tag{14}$$

これから$A^{-1} \bmod 5\cdot 7\cdot 11$が以下のように求められ、Extended Euclid Algorithmより(13)が求められる。

$$A^{-1} \equiv \begin{bmatrix} 193 & 192 & 96 \\ 0 & 0 & 193 \\ 0 & 1 & 193 \end{bmatrix} \bmod 5\cdot 7\cdot 11 \tag{15}$$

4.1.3. Newton-Schultz の方法

まず (8) 式から $N=6$ となり、$p=3$ とした場合 (10) をみたす最小の k は $k=2$ となる。よって B_{2^2} まで計算する。

$$B_1=\begin{bmatrix}2&1&2\\0&0&2\\0&1&2\end{bmatrix},\ B_{2^1}=\begin{bmatrix}5&4&2\\0&0&5\\0&1&5\end{bmatrix},\ B_{2^2}=\begin{bmatrix}41&40&20\\0&0&41\\0&1&41\end{bmatrix} \tag{16}$$

これから Extended Euclid Algorithm より (13) が求められる。

4.2. 一般逆行列の計算例

Decell-Leverrier(Penrose) の方法 [10][14] を用いた場合について解説する。

4.2.1. 割算を用いた Penrose の方法

$B={}^tCC$ とおく。以下の $B_1, B_2, B_2B=O$ と $q_1=10, q_2=36$ より C^+が求められる。

$$B_1=\begin{bmatrix}7&3&0\\3&5&-2\\0&-2&8\end{bmatrix},\ B_2=\begin{bmatrix}6&6&-6\\6&6&-6\\-6&-6&6\end{bmatrix},\ C^+=\frac{B_1{}^tC}{q_2}=\frac{1}{18}\begin{bmatrix}-1&-4&7\\5&2&1\\4&-2&8\end{bmatrix} \tag{17}$$

4.2.2. 中国剰余定理を用いた Penrose の方法

まず (8) 式において $B={}^tCC$ に対して $N=79$ となり、(7) 式をみたす M として $M=19\cdot23\cdot29$ を用いる。定理 1 より

$$C^+\equiv\begin{bmatrix}1&4&12\\14&17&18\\15&2&11\end{bmatrix}\bmod 19,\ C^+\equiv\begin{bmatrix}14&10&17\\22&18&9\\13&5&3\end{bmatrix}\bmod 23,\ C^+\equiv\begin{bmatrix}8&3&2\\18&13&21\\26&16&23\end{bmatrix}\bmod 29 \tag{18}$$

これから $C^+ \bmod 19\cdot23\cdot29$ が以下のように求められ、Extended Euclid Algorithm より (17) が求められる。

$$C^+\equiv\begin{bmatrix}704&2816&7745\\9153&11265&11969\\9857&1408&7041\end{bmatrix}\bmod 19\cdot23\cdot29 \tag{19}$$

5. 実験方法

Newton-Schultz の方法と、Faddeev,Penrose の方法 (割算による方法と中国剰余定理による方法) について実際に計算し、実行時間を比較する。具体的に次の 4 つの項目についてデータを取った。なお、Newton-Schultz の方法において (9) 式の B_1 を求めるのに、階数分解の方法 ([14] 定理 1) を使用している。

1) Newton-Schultz の方法と Faddeev の方法 (割算による方法と中国剰余定理による方法) について、行列要素の桁数を 5,10,20 桁にそれぞれ固定して、正則な行列のサイズをそれぞれ 10×10 から 100×100 まで変化させる。

2) 1) の方法で、行列サイズを 20×20 に固定して、正則な行列要素の桁数を 10 桁から 250 桁まで変化させる。
3) Penrose の方法 (割算による方法と中国剰余定理による方法) について、行列サイズを 10×5、行列の rank を 1 から 5 それぞれに固定して、それぞれ行列要素の桁数を 10 桁から 1000 桁まで変化させる。
4) Penrose の方法 (中国剰余定理による方法) で、行列サイズを 10×5, 行列の rank を 4 に固定して、行列要素の桁数を 10 から 1000 桁まで変化させたときの $A^+ \bmod p_j$,Garner 処理,Extended Euclid Algorithm の各計算時間の合計を比較する。

また、素数 p の抽出方法は以下のようにした。

- 101 から 16381 までは組み込み関数 prime() により抽出する (prime(1899)=16381)。
- それ以降は組み込み関数 pari(nextprime,) により抽出する。

また、中国剰余定理における法には 4 桁の素数 1009 から順に使用するものとする。

5.1. Newton-Schultz の方法における p の決定法

ここで、Newton-Schultz の方法における p の決定には以下の方法を用いた。

p は 3 桁あるいは 4 桁の素数を用いるとする。p が 3 桁の場合、p^{2^1} は $10^4 < p^{2^1} < 10^6$ の間に、p^{2^2} は $10^8 < p^{2^2} < 10^{12}$ の間にある。また、p が 4 桁の場合、p^{2^1} は $10^6 < p^{2^1} < 10^8$ の間に、p^{2^2} は $10^{12} < p^{2^2} < 10^{16}$ の間にある。このように、p が 3 桁と 4 桁の場合において互いのカバーできない領域をうめあっていることがわかるため、この関係を用いる。

まず、(10) 式の右辺 $2N^2+1$ が上記で述べた領域のうちどこに入るのか調べて、p が 3 桁か 4 桁か、また p^2 のべき数 l を決定する。その後、その桁数における p の中で $2N^2+1$ より大きく、なおかつ最も近い値を p^{2^l} がもつような p を決定する。ここでは、二分探索法を用いた。

6. 実行結果および考察

6.1. 実行結果

使用環境は以下の通りである。

使用機種:HP9000/735, クロック周波数:99MHz, OS:HP-UX 9.01, メモリ:80Mbyte

また、使用ソフトは富士通情報社会科学研で開発中の数式処理システム Risa/Asir である。

以下、実行時間比とは (modulo による計算時間)/(割算による計算時間) の値を表すものとする。なお、実験 1),3),4) に関しては表 1,2,3 に結果の一部を示してある。

- 実験 1) について、行列成分の桁数を固定して行列の次数を変化させた場合、
  - ◇ Faddeev の修正法 (割算による方法) に対する Newton-Schultz の方法の実行時間の比は、行列成分を 5 桁に固定した場合行列サイズの増加に伴って 6.08 から 1.78 まで減少する。
  - ◇ この実行時間における CPU 時間/GC 時間の比は約 2 でほぼ一定である。つまり、割算による方法の方が計算時間全体から見た GC 時間の割合が高いことがわかる。

◇ Faddeevの修正法(中国剰余定理による方法)については、CPU時間・GC時間ともに急激に増加したため行列サイズが 40×40 の時点で計算をうち切った。メモリ配分などの点でまだプログラムに問題があると思われる。

- 実験2)について、行列のサイズを固定して行列成分の桁数を変化させた場合、
  ◇ Faddeevの修正法(割算による方法)に対するNewton-Schultzの方法の実行時間の比は、行列成分の桁数の増加に伴い4.31から4.07へと、多少減少してはいるもののほぼ一定であった。
  ◇ この実行時間におけるCPU時間/GC時間の比は約1でほぼ一定である。つまり、Newton-Schultzの方法においてCPU時間だけでなくGC時間もかなり費やしていることがわかる。
  ◇ Faddeevの修正法(中国剰余定理による方法)については、実験1)同様行列サイズが 60×60 の時点で計算をうち切った。
- 実験3)について、Penroseの方法(割算による方法と中国剰余定理による方法)で行列サイズとrankを固定して行列成分の桁数を変化させた場合、
  ◇ 割算による方法に対する中国剰余定理による方法の実行時間の比は、rankの値に関わらず桁数の増加に伴って減少している。
  ◇ rankが高い方が実行時間の比が小さくなるのが速い。
  ◇ この実行時間におけるCPU時間/GC時間の比は、rankが2のとき約1でほぼ一定であるが、rankが4のとき桁数の増加にともない増加している。
- 実験4)について、Penroseの方法(中国剰余定理による)で行列サイズと行列のrankを固定して行列成分の桁数を増加させた場合、
  ◇ 成分の桁数が100桁あたりでGarner処理の合計時間が $A^+ \bmod p_j$ 計算の合計時間を上まわる。
  ◇ 桁数の増加に伴い、$A^+ \bmod p_j$ 計算とGarner処理においてGC時間が急激に増加する。
  ◇ 桁数が2倍になると、$A^+ \bmod p_j$ 計算・Garner処理・Extended Euclid Algorithm処理のCPU時間はそれぞれ約2倍、4倍、4倍となっている。

6.2. 考察

- Newton-Schultzの方法とFaddeevの修正法(割算による)との実行時間の比較について
  ◇ 行列サイズを固定して成分の桁数を増加させても実行時間比がほぼ一定だった理由として、桁数の増加に伴ってとなりあう素数 p_i, p_j において $p_i^{2^l}$ と $p_j^{2^l}$ との間隔が広がりすぎて、よりよい p が選べなくなったためと考えられる。
  ◇ 行列成分を5,10,20桁に固定したものの実行時間を比較すると、固定桁数の増加に伴い実行時間の比の減少が穏やかになる。この点からも、桁数が増加した場合の p の選び方に問題があることがわかる。
- Penroseの方法(割算と中国剰余定理による)の比較について、
  ◇ rankが高い方が実行時間比が小さくなるのが速い理由として、rankが2と4のデータを比較した場合、rankが4の時の方が割算による方法において実際に割算を行う部分の計算時間が大幅にかかっており、全実行時間の90%以上をしめているためと

考えられる。

◇ rankが低い場合、$p_1p_2\cdots p_l$の積が$2N^2+1$よりも小さくてもExtended Euclid AlgorithmでA^+が正確に求められたケースが多数ある。行列のrankと$2N^2+1$との関係が数学的に説明できれば、中国剰余定理による方法はますます効果を発揮するものと思われるが、それは難しい。

- Penroseの方法(中国剰余定理による)で各プロセスにおける実行時間比について、
  ◇ $A^+ \bmod p_j$計算とGarner処理においては、桁数の増加に伴い$2N^2+1$の値が莫大なものになるので、GC時間がCPU時間の1/2をこえるまでに増加したと考えられる。
  ◇ 全計算時間においてその大部分を占めるのがExtended Euclid Algorithmによるプロセスであり、この部分の高速化が望まれる。
  ◇ 中国剰余定理を用いる方法においては、並列計算機を用いて$A^+ \bmod p_j$の計算プロセスを並列処理することにより高速化できる可能性がある。
- 全体的に、現段階で割算を用いた方法が速いのはFaddeevの修正法・Penroseの方法ともにアルゴリズムの最後に1回だけしか割算を行わないためと考えられる。

7. 結論

我々は本報告で一般逆行列におけるmodulo計算によるものと割算によるものとの効率を比較した。ここで次のようなことがわかった。

- Newton-Schultzの方法とFaddeevの修正法(割算による)の実行時間比は、
  ◇ 行列成分の桁数を固定して行列サイズを増加させるとき、1に近づく。
  ◇ 行列サイズを固定して行列成分の桁数を増加させるとき、ほぼ一定であった。
  後者について、法pの選び方の改善により実行時間の比が1に近づくようになることが考えられる。
- Penroseの方法(割算と中国剰余定理による)の実行時間比は、行列のrankが高い方が1に近づくのが速い。
- Penroseの方法(中国剰余定理による)の各プロセスにおいて、行列成分の桁数が2倍になると、$A^+ \bmod p_j$計算・Garner処理・Extended Euclid Algorithm計算の各実行時間におけるCPU時間は、それぞれ約2倍、4倍、4倍となっている。
- 全般的に、実行時間比を1より小さくする(modulo計算による方法が割算によるものよりも速くなる)ためには、さらにサイズ・成分の桁数がともに大きな行列での実験が必要であり、それにはさらに大容量・高速なマシンが不可欠である。

8. 今後の課題

- Newton-Schultzの方法で用いる法pの選択方法の改良
- 中国剰余定理を用いる場合での、$A^+ \bmod p_j$計算・Garner処理の各プロセスにおけるGC時間の増大の抑制
- Greville[11] その他の方法 [8][9][12][13][14] でmoduloを用いる場合の効率の考察
- 中国剰余定理とHensel構成を合成した方法の検討

- modp_jを同時に計算する並列処理の検討（ハードも必要である）

謝辞

本研究に有益な助言をいただいた愛媛大学工学部野田松太郎教授、富士通情報社会科学研の野呂正行氏に深く感謝いたします。

参 考 文 献

[1] 斉藤正彦:基礎数学 1 線形代数入門, 東京大学出版会 (1966)

[2] 斉藤正彦:基礎数学 4 線形代数演習, 東京大学出版会 (1985)

[3] 古屋茂・小国力:線形代数の計算法 (上・下), 産業図書 (1971)

[4] 柳井晴夫・竹内啓:射影行列・一般逆行列・特異値分解, 東京大学出版会 (1983)

[5] 半谷裕彦・川口健一:計算力学と CAE シリーズ 5 形態解析 一般逆行列とその応用, 培風館 (1991)

[6] 木田祐司・牧野潔夫: UBASIC によるコンピュータ整数論, 日本評論社 (1994)

[7] Gregory,R.T.and Krishnamurthy,E.V.:Methods and Applications of Error-Free Computation,Springer-Verlag(1984)

[8] Albert,A.:Regression and the Moore-Penrose Pseudoinverse, Academic Press(1972)

[9] Ben-Israel,A.and Wersan,S.J.:An Elimination Method for Computing the Generalized Inverse of an Arbitrary Complex Matrix,J.ACM,Vol.10, pp.532-537(1963)

[10] Decell,H.P.,Jr.:An Application of the Cayley-Hamilton Theorem to Generalized Matrix Inversion ,SIAM Review,Vol.7,No.4, pp.526-528(1965)

[11] Greville,T.N.E.:Some Applications of the Pseudoinverse of a Matrix,SIAM Review,Vol.2,No.1,pp.15-22(1960)

[12] Noda,M.,Izumida,M.,Ochi,M.(野田松太郎・泉田正則・越智正明): "一般逆行列の数式処理システムによる直接解法とその評価", 情報処理学会論文誌, Vol.30,No.11,pp.1376-1384(1989-11)

[13] Rao,T.M.,Subramanian,K.and Krishnamurthy,E.V.:Residue Arithmetic Algorithms for Exact Computations of g-inverses of Matrices,SIAM J.Numer. Anal.,Vol.13,No.2,pp.155-171(1976)

[14] Makino,I.,Saito,T.(牧野潔夫・齊藤敏明): "一般逆行列の計算アルゴリズムとその証明", 京都大学数理解析研究所講究録, Vol.941,pp.156-169(1996-3)

参考

実験データの一部を以下に示す。表中の数字は、特に断りのない限り実行時間を表すものとする。表示形式は"CPU time(sec)+GC time(sec)"である。また、表1の(p,l)は Newton-Schultz の方法で使用する法p^{2^l}のpとlの組である。

Table 1. 実験1) における成分が5桁の行列の A^{-1} 計算の計測結果

サイズ	$p^{2^l}:(p,l)$	Newton-Schultz	Faddeev(division)	Faddeev(Chinese)	実行時間比
10×10	(2111,5)	1.94+0.49	0.31+0.09	7.48+2.84	6.08
20×20	(2383,6)	19.34+2.19	4.69+0.61	141.87+61.9	4.06
30×30	(383,7)	79.56+5.79	24.68+3.25	920.39+419.45	3.04
40×40	(2999,7)	225.97+12.85	84.64+10.75	3689.81+1344.58	2.50
50×50	(157,8)	524.99+29.28	224.62+23.83		2.23
60×60	(457,8)	1068.46+77.09	480.11+54.99		2.14
70×70	(1289,8)	1936.66+95.09	951.2+126.48		1.89
80×80	(3727,8)	3317.98+159.65	1736.36+219.33		1.78
90×90	(113,9)	5618.36+381.84	2987.21+372.38		1.78
100×100	(179,9)	8311.85+490.49			

Table 2. 実験3) におけるサイズが 10×5,rank が4の行列の A^{+} 計算の計測結果 (p は4桁)

桁数	法の使用数	Penrose(Chinese)	Penrose(division)	実行時間比
10	55	2.79+0.84	0.2+0.06	13.9
20	105	6.34+1.74	0.51+0.14	12.4
40	197	15.51+2.93	1.65+0.41	8.95
60	297	29.73+3.86	3.49+0.6	8.21
80	391	47.78+5.86	6.16+0.9	7.60
100	476	66.75+6.76	9.16+1.61	6.83
200	914	229.15+34.1	34.43+5.36	6.62
300	1308	471.49+49.37	74.39+9.87	6.18
400	1727	825.65+84.38	134+14.13	6.14
500	2125	1256.39+180.28	215.03+40.72	5.62
600	2509	1809.94+226.54	305.65+51.85	5.69
700	2887	2425.35+245.12	422.71+69.61	5.42
800	3265	3136.45+373.24	551.78+94.76	5.43

Table 3. 実験4) における各プロセスごとの計算時間の計測結果 (p は4桁)

桁数	$A^+ \bmod p_j$計算の合計	Garber 処理	Extended Euclid
10	1.02+0.45	0.35+0.15	1.38+0.22
20	2.04+0.77	0.86+0.31	3.4+0.66
40	3.78+0.19	2.13+0.68	9.53+1.13
60	5.93+1.27	4.21+0.98	19.48+1.58
80	8.04+1.29	6.58+1.88	33+2.69
100	9.85+1.49	9.37+2.05	47.33+3.21
200	20.3+6.34	31.3+12.26	177.01+15.47
300	31.05+7.39	63.04+19.03	376.49+22.91
400	43.22+8.39	109.42+43.35	671.64+32.63
500	57.94+32.61	164.05+90.17	1031.67+57.02
600	72.46+38.11	231.22+121.26	1502.15+67.07
700	88.91+47.11	306.8+120.86	2023.92+76.97
800	106.08+58.16	397.14+210.93	2625.22+103.95