Scilab関数”determ”を高速化するためのツールボックス(Ver.4)

木村 欣司(京都大学大学院情報学研究科)

Scilab関数”determ”とは?

以下の形の行列式を計算する為の関数である

$$\begin{vmatrix} 2+2s^5 & 6+7s^2 & 9+3s^4 \\ 8+6.8s & 7.5+9.8s^2 & 11+2.5s \\ 7+4s^4 & 9+s^2 & 10+6s^3 \end{vmatrix}$$

2変数以上の多項式の行列式は，計算できない

2変数以上の多項式の行列式を計算したい人は，数式処理ソフトを使うべき

"determ" の使い方

res=determ(W [,k]) として行い,

W には, 実数正方多項式行列,

k には, 整数で, W の行列式の次数の上限を入れる

k は, 必ずしも入力する必要はない

k が入力されない場合,

$$k \ = \ n \times \max(\mathrm{degree}(W))$$

より大きな2の冪乗の最小値

"determ" のアルゴリズム(多項式補間法)

Input: Square polynomial matrix $P(s) = (P_{i,j}(s))$ of size n and max(degree($P_{i,j}(s)$)) $= N$

Outpu: Polynomial $p(s)$-the determinant of $P(s)$

Step 1: Compute the upper bound for the degree of the determinant: kが与えられていないならば, $N_0 = n \times N$より大きな2の冪乗の最小値, kが与えられていれば, $N_0 = k$

Step 2: Using FFT algorithm, perform direct DFT at N_0 points on the set $\{P_0, P_1, \cdots, P_{N_0-1}\}$ of coefficient matrices of $P(s)$ and obtain the set $\{Y_l | l = 0, 1, \ldots, N_0 - 1\}$.

Step 3: Compute the vector $z = [z_0, z_1, \ldots, z_{N_0-1}]$, where $z_l = \det(Y_l), l = 0, 1, \ldots, N_0 - 1$.

Step 4: Perform inverse DFT on z using the FFT algorithm to obtain the coefficient vector $p = [p_0, p_1, \cdots, p_{N_0-1}]$ of the determinant $p(s) = p_0 + p_1 s + \cdots + p_{N_0-1} s^{N_0-1}$

cf. D. Henrion, M. Hromcik, M. Sebek "Some Algorithms used in the Polynomial Toolbox for Matlab", Proceedings of the NICONET Workshop on Numerical Control Software in Control Engineering, pp. 71-76, Louvain-la-Neuve, Belgium, January 2001

基本アイディア

行列式のタイトな次数の上界を計算できると,

Step 3: Compute the vector

$z = [z_0, z_1, \ldots, z_{N_0-1}]$,

where $z_l = \mathbf{det}(Y_l), l = 0, 1, \ldots, N_0 - 1$.

少ない数の$\mathsf{det}(Y_l)$を計算することで, 解を構成できる

既存の結果

$$P(s) = \begin{pmatrix} 2 + \cdots + 3s^2 & \cdots & 4 + \cdots + 5s^4 \\ \vdots & \cdots & \vdots \\ 6 + \cdots + 7s^3 & \cdots & 8 + \cdots + 9s^5 \end{pmatrix}$$

行列$P(s)$は, 以下のように書くこともできる

$$P(s) = \begin{pmatrix} 2 & \cdots & 4 \\ \vdots & \cdots & \vdots \\ 6 & \cdots & 8 \end{pmatrix} + \cdots + \begin{pmatrix} 0 & \cdots & 0 \\ \vdots & \cdots & \vdots \\ 0 & \cdots & 9 \end{pmatrix} s^5$$

すなわち,

$$P(s) = P_0 + P_1 s + \cdots + P_N s^N$$

の形に書くことができる

次数そのものを与える公式(その1)

cf. Didier Henrion and Michael Šebek, "Improved Polynomial Matrix Determinant Computation"

$$T_i = \begin{pmatrix} P_N & \cdots & P_{-i+1} & P_{-i} \\ & \ddots & & \vdots \\ & & P_N & P_{N-1} \\ 0 & & & P_N \end{pmatrix}$$

は, Toeplitz matrixであり, P_iから生成されるものである. ここで, 行列$P(s)$がregular matrix polynomialであると仮定する. $i < 0$においては, $T_{i-N} = 0$であり, $P_i = 0$であるとする.

$$r_i = \text{rank}\, T_i - \text{rank}\, T_{i-1}$$

として, vを以下で定義する.

$$v = \mathsf{min}\{i : r_i = n, i \geq 0\}$$

このとき, δを, $P(s)$の行列式のsの最大次数とすると,

$$\delta = \mathsf{rank}\, T_v - n(v+1)$$

が成立する. 具体的には, $T_{-N}, T_{1-N}, \cdots$と計算する. T_vは, vが1増えた場合, 増える前のLU分解を結果を生かせるため, いろいろな計算の工夫が可能である. よって, このアルゴリズムの計算量を精密に評価することは困難である.

より効率的で, かつ数値的に安定なものを考える

Hungarian algorithm

与えられた行列式から

$$\begin{pmatrix} 5 & 2 & 4 \\ 1 & 2 & 1 \\ 4 & 2 & 3 \end{pmatrix}$$

という行列を考える, この行列に対して, Hungarian algorithm を適用するとこで, $O(n^3)$でタイトな bound を手に入れることができる

cf. http://en.wikipedia.org/wiki/Hungarian_algorithm

注意

多項式係数を無視した場合の多項式行列の行列式の最大次数の問題は,

Linear Assignment Problemに帰着する

Hungarian algorithmによる次数の上界の問題点

以下の入力に対して，Hungarian algorithmによる次数の上界計算アルゴリズムを適用してみる

$$\begin{vmatrix} 2+2s^5 & 6+7s^2 & 9+3s^4 \\ 8+6.8s & 7.5+9.8s^2 & 11+2.5s \\ 7+4s^4 & 9+s^2 & 10+6s^3 \end{vmatrix}$$

答えは，10次となる，しかし，正しい答えは，8次である，**Hungarian algorithmによる次数の上界計算アルゴリズムは，係数についての情報を考慮しないため計算量は小さいが，上界としては，緩い上界になる**

次数そのものを与える公式(その2)

一般化固有値問題を利用した次数の評価式

$$P(s) = P_0 + P_1 s + \cdots + P_N s^N,$$

より

$$K = \begin{pmatrix} 0 & I & & \\ & & \ddots & \\ & & & I \\ -P_0 & \cdots & \cdots & -P_{N-1} \end{pmatrix}, L = \begin{pmatrix} I & & & \\ & \ddots & & \\ & & I & \\ & & & P_N \end{pmatrix}$$

として,

$$\det(P(s)) = \det(sL - K)$$

を利用して，次数の評価を行う．すなわち，一般化固有値問題

$$Kv = sLv$$

の有限の値を持つ固有値の数が，**det**$(P(s))$の次数となる

最悪計算量は，$O(n^3N^3)$である

摂動に対する考察

$$\begin{vmatrix} 2+2s^5 & 6+7s^2 & 9+3s^4 \\ 8+6.8s & 7.5+9.8s^2 & 11+2.5s \\ 7+(4+\delta)s^4 & 9+s^2 & 10+6s^3 \end{vmatrix}$$

Hungarian algorithmによる次数の上界は，δの値に依らずに，10次

δ	10^{-11}	10^{-12}	10^{-13}	10^{-14}	10^{-15}
kを与えない時のdeterm	10	8	8	8	8
一般化固有値問題	10	10	10	8	8

解きにくい問題に対する考察(その1)

$$\begin{vmatrix} 2+2s^5 & 6+7s^2 & 9+3s^4 \\ 8+6s & 7.5+9s^2 & 11+2s \\ 7+(4\times\alpha)s^4 & 9+s^2 & 10+6s^3 \end{vmatrix}$$

Hungarian algorithmによる次数の上界は, αの値に依らずに,

10次

α	10^4	10^5	10^6	10^7	10^8
kを与えない時のdeterm	14	15	15	15	15
一般化固有値問題	10	10	10	10	10

解きにくい問題に対する考察(その2)

$$A = \begin{vmatrix} 2+2s^5 & 6+7s^2 & 9+3s^4 \\ 8+6s & 7+9s^2 & 11+2s \\ 7+(4\times 10^{10})s^4 & 9+s^2 & 10+6s^3 \end{vmatrix}$$

```
Scilab:determ(A,11)=

                                          2              3

ans=131.00049 + 174.00018s - 357.99976s - 475.99963s

             4              5              6              7

+ 1.200D+11s + 4.800D+11s - 1.600D+11s + 5.600D+11s

                          8              9             10

             - 8.400D+11s - 0.0003662s - 1.080D+12s
```

数式処理ソフトRisa/Asir:

```
-1079999999892*s^10-839999999920*s^8+560000000176*s^7
-160000000453*s^6+479999999876*s^5+119999999853*s^4
-476*s^3-358*s^2+174*s+131
```

なぜ，ここまでの精度が出るのか?

determ(A,”11”)において，“11”であることをScilabに教えているから

“11”であることをScilabに教えないならば

```
Scilab:roots(determ(A))=

ans  =

                                        2              3
    130.99994 + 173.99966s - 358.00009s - 475.99991s

              4              5              6              7
 + 1.200D+11s + 4.800D+11s - 1.600D+11s + 5.600D+11s

              8              9             10             12
 - 8.400D+11s - 0.0000610s - 1.080D+12s + 0.0002136s

              13             14             15
 - 0.0001221s + 0.0002136s - 0.0002136s
```