行列の減次法 (サマリー)

山下　紀幸

2007/12/08

はじめに

J の組み込み関数 +/ . *は使い勝手がよいが、行列の次数が高くなると計算にかなり時間がかかるので、1 次低減して det を計算する幾つかの方法を提案する。

```
   latin=:3 : 'rot ^:(i.y)a=.>:i.y'
```

1　det 時間測定関数

```
   latin 6 NB. 演算対象
1 2 3 4 5 6
2 3 4 5 6 1
3 4 5 6 1 2
4 5 6 1 2 3
5 6 1 2 3 4
6 1 2 3 4 5
```

```
   dett=:3 : 0
    t=.6!:1''
    dety=.det y
    wr (":(6!:1'')-t),'sec'
   dety
)

    dett latin 6
1.32sec NB. 0sec
_27216
```

2　ガウスの消去法を利用（極めて早い）

```
   detdeg
3 : 0
    n=.<:#y=.y
    i=.1 [s=.''[sig=.(_1)^n-1
    a0=.0{y [a00=.0{0{"1 y
    a0=.a0%a00
    while. n>:i do.
     data=.}.(i{y)-(i{0{"1 y)*a0
     s=.data,s
     i=.>:i
    end.
    ''
    ans=:a00*sig*det (n,n)$|.s
)

   detdeg latin 6   NB. can't find detdegt
_27216
```

1. 4 － 5 行 1 行目の関数の最初の文字を a00 に変換
2. 6 － 10 行目で 2 行目以降の関数から各冒頭数字倍した a0 を引いて冒頭数字を 0 とすると 5 次式の中身が出来る。
3. 5 次式を行列にして det 計算し、a0 をかける

3　1 行目の数列に関する余因子分解

5 次の余因子行列が 6 個出来るので、det 演算時間の短縮と余因子行列の増加との相殺で det と同程度

```
   detrowt
3 : 0
    t=.6!:1''
    n=.#y
    a=.0{y
    b=.(_1)^i.n
    mn=:minorl y
    mnd=:>det each mn
   wr (":(6!:1'')-t),'sec'
   ans=:+/a*b*mnd
)

    detrowt latin 6
1.37sec NB. 0sec
```

```
_27216
```

4　行列式の定義とおりの演算法

西川流の each を使用。以外と時間がかかる。

```
   detcoft
3 : 0
    t=.6!:1''
    a=.<"0 i.#y
    ij=.(,each)/~a
    ax=.<y
    b=.ij fni each ax
    c=.det each b
    k=.(<_1)^ each +/ each ij
    d=.+/>k*each c
    e=.0{y
    wr ans=:+/d*e
    (":(6!:1'')-t),'sec'
)
```

```
   detcoft latin 6
_27216
8.51sec NB. 0sec
```

5　ガウスの消去法による多段減次法と途中経過

```
   detky
3 : 0
    n1=.<:<:n=.#y0=.y
    i=.0 [s=.1 [sig=.(_1)^n
    while. n1>i do.
     data=.degrd y0
     if.(0{0{data)=0 do.data=.|."1 data end.
     y0=.data
     s=.coef*s
     i=.>:i
     end.
     ''
   ans=:sig*s*det2 y
```

)

0.11sec

2 次まで減次したところで det2 で答えを求めた。3 次でピボットが 0 になるのを 6 行目の if. 文で左右反転して防いでいる。

6　組み込み関数 det の計算法はどれか

ガウスの消去法が早い。