数理解析研究所講究録
91巻 1970年 212-223

ライブラリ作成上の問題点について

京大 大型計算機センター
星野 聰

科学計算用ライブラリとして

1. 代数方程式の根を求めるライブラリ
2. 行列の固有値・固有ベクトルを求めるライブラリ
3. 関数の極小点を求めるライブラリ

の開発を行ったので，この際の問題点について説明する。

1. 代数方程式の根

Iterative に根を求める方法として，Bairstow 近似や Newton 近似がある。これらは一般に速やかに根を求めることが出来る近似法であるが，その収束性の保証はないから危険がある。すなわち，Bairstow 近似のみ用いると，実軸に接近した複素根に対して，一つの近似実根が stick する恐れがある。そこで，一般に，両近似を組合せて用いる。

しかし，文献(1)のアルゴリズムのように，両近似を同時

に行うようにすると，複素根の近傍の実軸上で Newton 近似が非常に大きい変動を与えることがあるし，また無駄な繰返し計算を行うことにもなることがテストの結果からわかった。

したがって，まず Bairstow 近似を，ついで Newton 近似を一定回数づつ行い，なお収束しなければ収束判定基準をゆるめて再び上の操作をくりかえすことにした。

また，根が虚軸に関して対称であり，原点から出発すると虚軸上に search をくりかえすという不都合があるが，これは $|a_{n+1}/a_1|^{1/n}$ だけの shift を与えると効果的である。こゝで a_1 は最高次の係数，a_{n+1} は最低次の係数である。

たゞし，絶対値の小さい根から求めること(2) は出来なくなる恐れがある。

次に，収束の判定と，求められた根の精度の関係が問題である。プログラム上では，ひきつゞく繰返し計算における近似根の位置の変化（Bairstow 近似では，二つの近似根の和と積の変化）がある値以下になれば収束したと見なすのが普通であろう。しかし，実際に所要の根が求められたかどうかの確認のために，求められた近似根から，方程式の係数を再構成してみるということが行われている。(3)

等根又は接近した根の場合には，係数を再構成してみると与えられた元の方程式の係数とはよく合うが正解とは一般に

ずれている。これは根が多少変化しても係数は殆んど変化しないことを示している。係数は一般に、最後の桁より下の誤差は存在すると考えられるので、もしこのような場合にはさらに高精度の計算は意味がないといえる。

たとえば $n=5$ で、根 1，2，3（3重根）を持つ方程式として

$$0.10000\ 00000\ 000000\times10^{1}\ x^{5}$$
$$-0.12000\ 00000\ 000000\times10^{2}\ x^{4}$$
$$+0.56000\ 00000\ 000000\times10^{2}\ x^{3}$$
$$-0.12600\ 00000\ 00000\ 0\times10^{3}\ x^{2}$$
$$+0.13500\ 00000\ 00000\ 0\times10^{3}\ x$$
$$-0.54000\ 00000\ 00000\ 0\times10^{2}=0$$

を FACOM 230-60 により倍長計算（mantissa 61 ビット）によりとくと　根

$$0.10000\ \ 00000\ \ 000000\times10^{1}+j\ 0.0$$
$$0.29999\ \ 97425\ \ 13057\ 8\times10^{1}+j\ 0.44598\ \ 33778\ 39860\ 1\times10^{-5}$$
$$0.29999\ \ 97425\ \ 13057\ 8\times10^{1}-j\ 0.44598\ \ 33778\ \ 39860\ 1\times10^{-5}$$
$$0.30000\ 05149\ \ 73884\ 4\times10^{1}+j\ 0.0$$
$$0.20000\ 00000\ \ 00000\ 0\times10^{1}+j\ 0.0$$

をうるが、係数を再構成してみると、上で与えた係数が（上に示した桁数は）完全に復元されることがわかる。

また，等根又は接近した根をもつ場合に，収束の判定として関数値の減少の比率を使用すると，根の近傍では関数値が小さくなる範囲が割合広くて，係数を再構成してみると，係数は元の方程式のそれからはずれているにもかゝわらず，繰返し計算を打切ってしまうために精度が良くない解をうることがある。これも文献(1)のアルゴリズムのテストで経験したことである。

方程式の次数が大きくなると，overflow を生ずる危険が大きくなるのも問題である。このため，overflow が発生することは一つの情報と考えて，これによって初期値の変更（たとえば原点に近づける）をすればよい。overflow の発生を活用することは，他の数値計算法でも重要である。

根を求める方法としては，関数のノルムを減少させて行き，ノルムがゼロになる点として根を求める方法がある。(4) この方法は，やはり途中で（たとえば実軸上の鞍点）stick する危険があるのが問題である。また，極小化する関数は通常，根の近くではゼロに近く，変化も少ないので，極小点を正確に求めるのが困難である。これらの問題点に注意すると，3. で説明する関数の極小化のプログラムによって根を求めることができる。

また，根を companion 行列の固有値として求めることも

できる。計算量は n^3 に比例するので有利とはいえないが，2. で説明する固有値計算のプログラムによるテストでは $n=20$ で約2秒，$n=36$ で約6秒を要した。(FACOM 230-60による)

開発されたサブルーチンは，Bairstow 近似と Newton 近似を一定回数づゝ交互に繰返し，最高 200 次の代数方程式を解いたが，特に高次の場合には，求められた近似根の error bound について実用的な評価が出来ることが必要であると考えている。(5)

2. 行列の固有値・固有ベクトル

実行列について考える。行列が対称であっても非対称であっても QR 法によって三角化を行うのが普通である。行列はまず Hessenberg 形式に変形される。(対称行列では三項行列となる) QR 法についての問題点としては

1. origin shift の方法

対称行列の場合には，右下の二行二列

$$\begin{pmatrix} d_{n-1} & e_n \\ e_n & d_n \end{pmatrix}$$

の固有値のうちで，右下の要素 d_n に近いものを採用する。この方法によると，新しい e_n，e_{n-1} (e_{n-1} は e_n の左上の要素) を $\bar{e}_n$，$\bar{e}_{n-1}$ とするとき $|\bar{e}_n| \leq |e_{n-1}|$，

$|\bar{e}_n \bar{e}_{n-1}| \leq |e_n e_{n-1}|$ が成立し，これから収束が証明されている。[6] 単に d_n だけの shift をするのでは，非常に収束がおそい場合がある。上記の理論をもとにして，右下の三行三列の固有値から shift をきめるように工夫することも出来るが，実験してみると二行二列のときが最も良い

非対称行列のときは，行列が orthogonal の場合に QR 変換が invariant になる[7]。そこで Parlett は，与えられた行列 A に対して |A の determinant|$^{1/n}$ だけ shift することを提案している。[8] 我々のサブルーチンもこれをとり入れている。

2. 固有ベクトルの計算

固有ベクトルは inverse iteration によって求められ精度も良い。ただし，対称行列で等しい固有値があるとき，互いに直交する固有ベクトルを求めるためには，固有値を少しずらして inverse iteration を行う必要がある。このずらせる量についての理論的な説明が必要であると思われる。

さらに，非対称行列では，数値的に固有ベクトルが defective な行列が存在する。この場合には，行列の（多重）固有値は，要素の変化 ε に非常に敏感で $\varepsilon^{1/n}$ の order になりうる。

inverse iteration の方法は，固有値が real か complex かによって別々の処理をするので，procedure の大きさは大きくなる。[9] ところで，固有ベクトルは，途中の相似変換で用いた変換行列と，変換後の行列（三角行列又は block-triangular な行列）の固有ベクトルからなる行列の積として求めることが出来る。[10] この方法は，上にのべた固有値をずらせる必要はないので，我々のサブルーチンで採用している。

3. 行列の scaling

行列の scaling が ill-scaled matrix に対しては，結果に大きく影響することが知られている。これは固有値計算の誤差が，行列のノルムでおさえられるからである。したがって，固有値，固有ベクトルの計算を始めるに先立って，まず行列の scaling を行う必要がある。

固有値計算についても，誤差の計算が必要である。固有値の posteriori error bound を求める上での問題は，固有ベクトルが数値的に defective になる場合である。[11]

3. 関数の極小化

大別すると関数の微係数を用いるかどうかによって二つに分けられる。こゝでは，条件付きでない場合について考える。以下のような方法が作られたのは，たとえば底がゆるやかに低下し，切り立った側面をもつ谷にそって進み極小点に到達する場合のように従来の方法では，非常に能率が悪いときにも有効なものが必要とされたからである。

1. 微係数を用いない場合

Powellの方法[12]が有力な方法であり，linear minimizationを繰返して行って，互いにconjugateな方向を見出して進む。この方法は，searchの方向が一次従属になる危険があるので，多少の工夫が施されている。しかし，次元数 n が大きいと，良くない結果を生ずるといわれる。[10]（どのような場合かわからない）。

これに対して Swann[13] による方法は，直交する n 個の searchベクトルを作るので，searchの方向は n 次元空間を coverする。しかし，無駄な方向へのsearchをかなり行うことになるので，Powellの方法に比べて収束は遅い。

2. 微係数を用いる場合

最も有効な方法は Fletcher, Powell[14] の方法である。この方法では，conjugate な方向に search が行われ，しかも Newton 近似による方向にもなっている。また，曲面の Jacobian の逆行列 H を update して行くわけであるが，round-off error によって，行列 H が positive definite でなくなることがある。したがって，H を（たとえば単位行列に）reset する必要が生ずることがあるので注意を要する。[15]

この方法は，次元数 n が増しても有効である。

次に Fletcher, Reeves の conjugate gradient 法[16] は，search の新しい方向 p_{i+1} をその点の gradient g_{i+1} および前回の search の始点における gradient g_i およびその search ベクトル p_i だけから決めるものである。n が増すと，一次従属な search を生じやすいので，ときどき，gradient の方向への reset を必要とする。

以上のように，Fletcher, Powell の方法の成功からみると，特に n が大きい場合には，解法には解析的な裏付けが必要であるように考えられる。すなわち，n が小さい場合と大きい場合には，困難さには大きい差があると思われる。したがって n が小さい場合に有効であっても，n が大きくなると有効かどうかわからない。

参考文献

1. Ellenberger, K. W., ALGORITHM 30, Comm. ACM., pp. 643, (1960).

2. Wilkinson, J. H., Rounding Errors in Algebraic Processes, Notes on Applied Science NO. 32, Her Majesty's Stationary Office, (1963).

3. System/360 Scientific Subroutine Package (360A-CM-03X) Version III Programmer's Manual, IBM.

4. Busk and Svejgaad, Polynomial Equations, in Selected Numerical Methods (ed. C. Gram),

5. 理論的な取扱いについては
Householder, A. S., The Numerical Treatment of a single nonlinear equation, McGraw-Hill (1970).

6. Wilkinson, J. H., Global Convergence of Tridiagonal QR Algorithm with Origin Shifts, Linear Algebra and its Applications, 1, pp. 409-420, (1968).

7. Parlett, B., Singular and Invariant Matrices under the QR Transformation, Math. Comput.

pp. 611-615, (1966).

8. Parlett, B., The LU and QR Algorithms, in Mathematical Methods for Digital Computers, Vol. 2, (Ed. A. Ralston and H.S. Wilf), John Wiley & Sons, pp. 116-130, (1967).

9. たとえば
Grad, J. and Brebner, M. A., Eigenvalues and Eigenvectors of a Real General Matrix, ALGORITHM 343, Comm. ACM., vol. 11, pp. 820-826, (1963).

10. J. ウォルシュ編 高須達監訳, 数値解析概論, pp. 82-84, (1970), 日本評論社
(Numerical Analysis: An Introduction, (ed. J. Walsh), (1966), Academic Press)

11. Varah, J. M., The Computation of Bounds for the Invariant Subspaces of a General matrix Operator, Tech. Report NO. CS66, (1967), Computer Science Department, Stanford University.

12. Powell, M. J. D., An Efficient Method for Finding the Minimum of a Function of

Several Variable without Calculating Derivatives, Computer Journal, Vol. 7, pp. 155 – 162, (1964).

13. Swann, W. H., Report on the Development of a New Direct Search Method of Optimisation, C.I.I. Ltd., Central Instrument Laboratory Research Note 64/3.

14. Fletcher, R., and Powell, M.J.D., A Rapidly Convergent Descent Method for Minimization, Computer Journal, pp. 163 – 168, (1963).

15. McCormick, G.P. and Pearson, J.D., Variable metric methods and Unconstrained Optimization, in Optimization (ed. R. Fletcher), pp. 307 – 325, (1969), Academic Press.

16. Fletcher, R. and Reeves, C.M., Function minimization by Conjugate Gradients, Computer Journal, Vol. 7, pp. 149 , (1964).