多項式固有値問題に対するニュートン法を使った反復改良法

田中 和希

慶應義塾大学理工学部数理科学科
野寺研究室

目 次

1 序論 **1**
1.1 背景 1
1.2 問題 1

2 行列多項式の線形化 **3**
2.1 線形化 3

3 不変部分空間と Invariant Pair **4**
3.1 不変部分空間 4
3.2 Invariant Pair 4

4 高次元のニュートン法 **6**
4.1 1 変数関数に関するニュートン法 6
4.2 多次元関数に関するニュートン法 6

5 今回のケース **8**

6 数値実験 **9**

7 結論 **10**

謝辞 **10**

参考文献 **11**

A ISIAE 法 **13**

B BiCGStab 法 **15**

要 旨

abst

第 1 章

序論

1.1 背景

n 個の質点に関する多自由度系の振動と呼ばれる物理現象（空気抵抗がありそれぞれがバネで繋がっている）は以下のような運動方程式を生じさせる.

$$M\frac{d^2y}{dt^2} + C\frac{dy}{dt} + Ky = y(t)$$

M は質量行列と呼ばれ質点の質量を対角線上に並べた対角行列, C は減衰行列と呼ばれ空気抵抗などの速さと力の関係を表す行列, K は剛体行列と呼ばれ変位と力を関係を表す行列である. この運動方程式を解く際には以下の二次固有値問題を解く必要性が生じる.

$$(M\lambda^2 + C\lambda + K)x = 0$$

本レポートではこのような行列多項式（行列を係数とする多項式）一般に関する固有値問題について扱う.

1.2 問題

この節では本レポートが扱う問題について述べる. 固有値多項式問題とは一般に以下のように定義される.

$P(\lambda)$ を d 次元の $n \times n$ の行列を係数とする行列多項式とし

$$P(\lambda)x = 0$$

をみたすような $\lambda \in \mathbb{C}, x \in \mathbb{C}^n$ を見つける事である.

ただし, 本レポートでは行列多項式 $P(\lambda)$ を一般に degree-graded な行列多項式の形に変形する事が出来るという事実を使って [2]

$$P(\lambda) = A_0\Phi_0(\lambda) + A_1\Phi_1(\lambda) + \cdots + A_d\Phi_d(\lambda) \tag{1.1}$$

と表す事にする. ここで Φ_j は j 次の多項式で以下を満たす.

$$\lambda\Phi_j(\lambda) = \alpha\Phi_{j+1}(\lambda) + \beta\Phi_j(\lambda) + \gamma\Phi_{j-1}(\lambda)$$

ただし $\Phi_{-1} \equiv 0$, $\Phi_0 \equiv 1$ で $\alpha_j = \frac{c_j}{c_{j+1}}$ を満たし, c_j を $\Phi_j(\lambda)$ の最大次元の項の係数とする.

第 2 章

行列多項式の線形化

この章では 1 章で定義した問題設定の式 (??) を線形化という手法で $\mathbb{C}^{nd\times nd}$ の行列に関する固有値問題に還元をすることを考える.

2.1 線形化

$$L_0 = \begin{pmatrix} \beta_0 I & \alpha_0 I & & & & \\ \gamma_1 I & \beta_1 I & \alpha_1 I & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \gamma_{d-2} I & \beta_{d-2} I & \alpha_{d-2} I \\ \tilde{A}_0 & \tilde{A}_1 & \dots & \tilde{A}_{d-3} & \tilde{A}_{d-2} & \tilde{A}_{d-1} \end{pmatrix}$$

$$L_1 = \begin{pmatrix} I & & & \\ & \ddots & & \\ & & I & \\ & & & c_d A_d \end{pmatrix}$$

$$L(\lambda) = L_0 - L_1\lambda \tag{2.1}$$

第 3 章

不変部分空間とInvariant Pair

3.1 不変部分空間

V を $\mathbb{C}$ 上の線形空間，W を V の部分空間，$f : V \to V$ を線形写像とする．このとき，$f(W) \subset W$ ならば，W を f に関する不変部分空間という．固有空間は不変部分空間である事が一般的に知られている．

3.2 Invariant Pair

Definition 3.1 $(X, H) \in \mathbb{C}^{n\times k} \times \mathbb{C}^{k\times k}$ が $P(x)$ の invariant pair であるとは,

$$\mathbb{P}(X, H) \equiv A_0 X\Phi_0(H) + A_1 X\Phi_1(H) + \cdots + A_d X\Phi_d(H) = 0$$

を満たす事である.

χ を X の列ベクトルのなす線形空間とすると χ が P の不変部分空間であるとみなす事ができる.

Proposition 3.1 式 (2.1) の Invariant Pair を $(X, H) \in \mathbb{C}^{nd\times k} \times \mathbb{C}^{k\times k}$ とした時, X が列フルランクであれば H の固有値は式 (2.1) に対する固有多項式問題の固有値の解である.

Proof: $(X, H) \in \mathbb{C}^{nd\times k} \times \mathbb{C}^{k\times k}$ は式 (2.1) の Invariant Pair であるから

$$L_0 X - L_1 XH = 0$$

を満たす. この両辺に H の固有ベクトル x をかけると, H の固有値 λ, 固有ベクトル x は $Hx = \lambda x$ を満たすから,

$$L_0 Xx - L_1 \lambda Xx = 0$$

であり, X は列フルランク行列であるから,

$$(L_0 - L_1\lambda)y = 0$$

を満たすような $y = Xx \neq 0$ であるから, λ は式 (2.1) に対する固有多項式問題の固有値の解である.

Proposition 3.2 式 (2.1) の Invariant Pair を $(Z, H) \in \mathbb{C}^{nd\times k} \times \mathbb{C}^{k\times k}$ とすると, Z の上から d 分割した時の最も上のブロック行列を Z_0 とすると

$$Z = V_m(Z_0, H)$$

と表す事ができる. ただし V_m は

$$V_m(X, H) \equiv \begin{pmatrix} X \\ X\Phi_1(H) \\ \vdots \\ X\Phi_{m-1}(H) \end{pmatrix}$$

と定義する.

Proposition 3.3

(2) の Invariant Pair $(X, H) \in \mathbb{C}^{n\times k} \times \mathbb{C}^{k\times k}$ の H の固有値は (1) の固有値でもある.

Proposition **??**

第 4 章

高次元のニュートン法

ニュートン法は数値計算によって方程式を解くための反復法による求根アルゴリズムの1つである.

4.1　1変数関数に関するニュートン法

1変数関数に関するニュートン法とは関数 $f : \mathbb{R} \to \mathbb{R}$ が $f(x) = 0$ となるような x を求める問題を考えた時に真の解に近いと思われる x_n を通り傾き $f'(x_n)$ であるような x_n の接線と x 軸との交点を x_{n+1} とするような漸化式

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

を $n \to \infty$ とした時に x_n が真の解に収束する事を期待するアルゴリズムである.

4.2　多次元関数に関するニュートン法

多次元関数 $f : \mathbb{R}^m \to \mathbb{R}^m$ の $f(x) = 0$ となるような $x \in \mathbb{R}^m$ を求める問題において真の解に近いと思われる x_n 周りのテイラー展開

$$f(x) = f(x_n) + \partial f(x_n)(x - x_n) + ...$$

(但し $\partial f(x)$ はヤコビ行列)
右辺の二次以降の項を無視して得れる一次近似の零点を x_{n+1} とするような漸化式

$$x_{n+1} = x_n - \partial f(x_n)^{-1} f(x_n)$$

の数列が解に収束する事を期待するアルゴリズムである.
ただ数値計算において $\partial f(x_n)^{-1}$ を求める事は困難、もしくは誤差を生じさせるため

$$\partial f(x_n) r = f(x_n)$$

をガウスの消去法などで解き、残差 r を求め　$x_{n+1} = x_n - r$ として x_{n+1} を求める.

第 5 章

今回のケース

前章まで見てきたように, 今回の問題は

$$\mathbb{P}(X, H) = 0$$
$$\mathbb{V}(X, H) = 0$$

$\mathbb{L} : \mathbb{C}^{nd \times k} \times \mathbb{C}^{k \times k} \to a$

第 6 章

数値実験

本章では，数値実験により 3 章，4 章で述べた解法の有効性を示す．全ての数値実験は以下の環境で行った．

- OS : Ubuntu14.04LTS
- CPU : Intel(R) Xeon(R) E3-1270 V2 @ 3.50GHz
- メモリ : 16GB
- プログラム言語 : MATLAB 2015a

第 7 章

結論

1 階の時間微分項が含まれる線形発展方程式に対しては，空間方向のみ離散化を行い，ISIAE 法を用いて解を計算するのが効率的である．これにより，欲しい解の時間変数 t の値や離散化により生じた行列の次元に依存しない反復回数で解を計算できる．また，ISIAE 法は，各反復に現れる線形方程式を効率良く解くことで，大規模問題に対しても 1 回の反復を少ない計算時間で行うことができる．さらに，反復を進めれば残差や誤差が欲しい精度にまで減少した解を求めることができる．よって，特に，細かいメッシュでの離散化が必要な問題や大きな t の値に対する解が必要な場合はこの方法が有効である．今後は，A, B，c が t に依存する場合や，A が正則でない場合の拡張について検討したい．また，別な種類の偏微分方程式に対しても，Krylov 部分空間法を用いて解を計算する効率的なアルゴリズムを検討したい．

謝辞

本稿を作成するにあたり，野寺 隆先生には大変お世話になりました．また，野寺研究室の皆様には多くの助言をいただくとともに，大学生活においても様々なことで助けていただきました．この場を借りて感謝申し上げます.

参 考 文 献

[1] António, A., Adérito, A. and Ercília, S., *"Application of the Advection-dispersion Equation to Characterize the Hydrodynamic Regime in a Submerged Packed Bed Reactor,"* Advances in Computational & Experimental Engineering & Sciences (Atluri, S. N. and Tadeu, A. J. B., eds.), Maderia, Portugal, ICCES, Tech Science Press, pp. 548–553, 2004.

[2] 小平邦彦, "複素解析", 岩波書店, 1991.

付 録 A

ISIAE法

```
function y=isiae(gamma,t,A,B,v,c,tol,delta,mmax)

%%%initialize%%%
C=B+gamma*A;
[L,U]=ilu(C);
[L1,U1]=ilu(A);
d=mybicgstab(L1,U1,A,c,10^(-14));
init=v-d;
beta=norm(init);
V(:,1)=init/beta;
[L2,U2]=ilu(B);
u=mybicgstab(L2,U2,B,C*init,10^(-14));
tolsys=gamma*tol/(norm(u)*mmax);
tol1=tolsys;
j=0;
r=1;
flag=0;

%%%start the iteration%%%
while r>tol&&j<=mmax

    j=j+1;
    b=B*V(:,j);

    %%%solve Cu=b for u inexactly%%%
    u=mybicgstab(L,U,C,b,min(tol1,delta));

    %%%compute the Arnoldi process%%%
    for k=1:j
        H(k,j)=V(:,k)'*u;
```

```
        u=u-H(k,j)*V(:,k);
    end
    H(j+1,j)=norm(u,2);
    V(:,j+1)=u/H(j+1,j);

    %%%check the condition for convergence%%%
    J=(H(1:j,1:j)+H(1:j,1:j)')/2;
    se=eigs(J,1,'sa');
    if se<eps
        if flag==0
        sprintf('Warning:Please set delta smaller value, or set gamma
        smaller value.')
        flag=1;
        end
    end

    %%%compute the approximation and the residual%%%
    y=beta*expm(-(t/gamma)*(H(1:j,1:j)\eye(j)-eye(j)));
    y=y(:,1);
    f=H(1:j,1:j)\y;
    tol1=tolsys*abs(f(1))/abs(f(j));
    v=C*V(:,j+1);
    r=abs(H(j+1,j)/gamma*(f(j)))*norm(v,2);

end

%%%compute the solution%%%
y=V(:,1:j)*y+d;
```

付 録 B

BiCGStab法

```
function x=mybicgstab(L,U,A,b,tol)

n=length(b);
x=zeros(n,1);
rj=b-A*x0;
r0=rj;
p=rj;

while norm(rj)>tol
    Ap=A*(U\(L\p));
    alpha=(rj'*r0)/(Ap'*r0);
    s=rj-alpha*Ap;
    As=A*(U\(L\s));
    w=(As'*s)/(As'*As);
    x=x+alpha*p+w*s;
    rj1=s-w*As;
    beta=alpha/w*(rj1'*r0)/(rj'*r0);
    p=rj1+beta*(p-w*Ap);
    rj=rj1;
end

x=U\(L\x);
```