ALGORITMA DAN PROGRAM MENGHITUNG AKAR KE-*m* DARI MATRIKS KOMPLEKS NON SINGULAR

SKRIPSI

ABRAHAM WIASTADI

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS AIRLANGGA
SURABAYA

2005

ALGORITMA DAN PROGRAM MENGHITUNG AKAR KE-*m* DARI MATRIKS KOMPLEKS NON SINGULAR

SKRIPSI

Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Sains Bidang Matematika di Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Airlangga

Oleh :

ABRAHAM WIASTADI
NIM. 080012203

Tanggal Lulus : 15 Agustus 2005

Disetujui Oleh :

Pembimbing I,	**Pembimbing II,**
Dra. Inna Kuswandari, M.Si.	**Dra. Yayuk Wahyuni, M.Si.**
NIP. 131 933 022	**NIP. 131 933 017**

LEMBAR PENGESAHAN SKRIPSI

Judul	**: ALGORITMA DAN PROGRAM MENGHITUNG AKAR KE-*m* DARI MATRIKS KOMPLEKS NON SINGULAR**
Penyusun	**: ABRAHAM WIASTADI**
N.I.M.	**: 080012203**
Tanggal Ujian	**: 15 Agustus 2005**

Disetujui oleh :

Pembimbing I,

Dra. Inna Kuswandari, M.Si.
NIP. 131 933 022

Pembimbing II,

Dra. Yayuk Wahyuni, M.Si.
NIP. 131 933 017

Mengetahui
Ketua Jurusan Matematika
FMIPA Universitas Airlangga,

DEPARTEMEN PENDIDIKAN NASIONAL ★ UNIVERSITAS AIRLANGGA ★ JURUSAN MATEMATIKA ★ FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

Drs. Moh. Imam Utoyo, M.Si.
NIP. 131 801 397

PEDOMAN PENGGUNAAN SKRIPSI

Sesungguhnya
dalam penciptaan langit dan bumi
dan silih bergantinya siang dan malam terdapat
tanda-tanda bagi orang yang berakal.
Al-Imran 190

Hanya milik Allah asma-ul husna,
maka bermohonlah kepada-Nya dengan
menyebut asma-ul husna itu.
Al-A'raf 180

Skripsi ini kupersembahkan
untuk Bapak Ibuku dan kedua Adikku tersayang

KATA PENGANTAR

Syukur Alhamdulillah penulis panjatkan ke hadirat Allah SWT atas rahmat dan hidayat-Nya maka penyusunan skripsi dengan judul "Algoritma dan Program Menghitung Akar ke-*m* dari Matriks Kompleks Non Singular" dapat terselesaikan.

Dalam penyusunan skripsi ini tidak sedikit hambatan yang penulis alami, namun berkat petunjuk dan saran yang diberikan semua pihak maka penulisan skripsi ini dapat terselesaikan dengan baik. Dalam kesempatan ini pula penulis ingin mengucapkan terima kasih dan penghargaan setinggi-tingginya yang secara langsung maupun tidak langsung telah memberikan bantuan dan dukungan hingga penyusunan skripsi ini dapat terselesaikan, yaitu kepada :

1. Dra. Inna Kuswandari, M.Si. selaku pembimbing I.
2. Dra. Yayuk Wahyuni, M.Si. selaku pembimbing II.
3. Drs. Eto Wuryanto, DEA. selaku dosen wali.
4. Rekan-rekan dan segenap civitas akademik Jurusan Matematika FMIPA Universitas Airlangga.

Penulis menyadari sepenuhnya bahwa penulisan skripsi ini tentunya masih jauh dari kesempurnaan. Oleh karena itu kritik dan saran yang membangun sangat diharapkan agar skripsi ini dapat lebih bermanfaat.

Surabaya, Agustus 2005

Penulis

UCAPAN TERIMA KASIH

ALLAH SWT dan Rasul-Nya
yang telah memberikan petunjuk pada penulis untuk selalu berserah diri dan berusaha serta memberikan kesehatan

Yth. Bapak dan Ibuku tercinta
yang atas do'a dan restunya serta kasih sayangnya yang tak kenal batas serta adik-adikku yang memberikan keceriaan

Yts. Nastiti Putri Palupi "upix"
tiada henti kau memberi semangat dan dorongan serta kasih sayangmu, terima kasih telah memberiku hari^2 yang indah

Yth. Kel. Bp. Budi Surojo
yang selalu memberikan nasehat dan semangat untuk terus maju serta memberikan masukan^2 tentang kehidupan

Yth. Rektor UNAIR, Yth. Dekan FMIPA UNAIR
yang telah memberi kesempatan pada penulis untuk menuntut ilmu

Yth. Dra. Inna Kuswandari, M.Si. selaku dosen pembimbing 1 dan Yth. Dra. Yayuk Wahyuni, M.Si. selaku dosen pembimbing 2 yang meluangkan waktunya untuk memberikan nasehat serta masukan^2 sehingga penulis dapat menyelesaikan skripsi

Yth. Drs. Eto Wuryanto, DEA selaku dosen wali
yang membantu penulis memperkenalkan dunia kampus sejak pertama kali menjadi mahasiswa sampai lulus

Yth. Dra. Utami Dyah P. dan Drs. M. Kartono, M.Kom
yang telah sabar menguji penulis dan memberikan masukan^2 perbaikan pada naskah skripsi

Yth. Bp/Ibu dosen Matematika UNAIR
yang telah memberikan kesabaran dalam mendidik

Mbak Wuri, Mas Edy Wiroso, Sumilan, Udi'
yang telah membantu kelancaran penulis selama menjadi mahasiswa Matematika UNAIR

temen^2 '00

- gank high : k-mel(ketua), chriez(wakil), dita(sekretaris), bu dosen lina(bendahara), miauw, ibu mila, nito"ratu", andri, indi(anggota baru)
- gank medium : mbak mirza, kholif, ipeh, amey, indah, yanyan, tatam(kapan jadi ny. Komting?), nenek, winwin, indah, vidi, indri, nanik, yuli"solo", yuli"palu", laily(kabur), kumal
- gank low : mbak elok, innah, dani, dennik, henny
- gank penthung : cahjo, kirud, bang pea(tempat nginep, ngetik, dll), eco"sincan oemar bakrie", arief"blue boy", QQ"komting", emon(ayo semangat ndang lulus), abi"oldis"(wis ndang majuae), jambul(ga usah males^2)
- yang laen^2 : F. Ulricka(ky' apa sih dirimu itu?), ika"tante", erie"simbok", dadan, sigit"gundul", oki

TERIMA KASIH KEBERSAMAANNYA N SEMOGA SUKSES

temen^2 laen angkatan

- zoogank(ojo golek wong nganjuk terus), hamster, juned, didin, apet, haryoko, yanto"endut", kurniawan, haris, ardian, gendon ayo bal^{2}an terus
- ana, nopie, arie serta keluarga yang selalu memberi semangat n do'anya s'moga sukses buat kalian semua
- om pet(yg Bantu bikin program) n anak'00 ke atas yang pernah minjemi buku ato yang pernah bercanda bareng, anak'01 yang bareng lulus n kawan^{2}nya, anak'02 semua, anak'03 yang belum disebut, orang2 labkom, anak'04 yang nggak banyak kenal terima kasih kebersamaan n keceriaannya
- temen^2 jurusan laen yang kenal terima kasih sudah mau kenal

Yth. mbah Dien"kertajaya", Yth. keluarga Bp. H. Samsuri, Yth. keluarga mbah Zaenab yang telah memberikan tempat untuk berteduh bagi penulis selama di Surabaya

temen^2 yang pernah satu kosan
temen kosan MT 69"H 234 RI", arek kosan M 152 : lom"keceng", om acul (tempat ngeprint), wawan, dyo, nawi, bogel, cak hanif, irwan"emerson", irfan"superman", iyud"bongol", AB, tu^{2}k, om topo, oqim"sarkehe", yogi, aris thanx kebersamaannya

alex sekeluarga, kesty sekeluarga, mince, uyab, ade sekeluarga, wahyu"boim" n bapak ibu, semua anak SMUDA AE '00 n tak lupa Mr. Exo serta semua pihak yang tidak dapat penulis sebutkan satu per satu

Abraham Wiastadi, 2005. *Algoritma dan Program Menghitung Akar ke-m Dari Matriks Kompleks Non Singular.* Skripsi ini di bawah bimbingan Dra. Inna Kuswandari, M.Si. dan Dra. Yayuk Wahyuni, M.Si., Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Airlangga.

Abstrak

Skripsi ini bertujuan untuk menyusun algoritma dan membuat program untuk menghitung akar ke-*m* dari matriks kompleks non singular. *Software* MATLAB dapat digunakan untuk menentukan akar ke-*m* dari matriks kompleks non singular, tetapi yang dimunculkan hanya satu akar saja. Dengan menggunakan bahasa pemrograman M-File dalam *software* MATLAB akan dibuat program agar akar yang lain dapat dimunculkan.

Kata kunci : Matriks kompleks, Struktur Jordan, MATLAB.

Abraham Wiastadi, 2005. *Algorithm and Programme to Find the m-th Roots of A Non Singular Complex Matrix.* This script was under supervisor of Dra. Inna Kuswandari, M.Si. and Dra. Yayuk Wahyuni, M.Si., Department of Mathematics, Faculty of Mathematics and Natural Science Airlangga University.

Abstract

The script is purposed to construct algorithm and programme to find the m-th roots of a non singular complex matrix. MATLAB can be used to find this but there is only one root. Because there are many m-th roots of a non singular complex matrix, we want to explore MATLAB Programme to get the others.

Key word : Complex Matrix, Jordan Structure, MATLAB.

DAFTAR ISI

Halaman

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang.

Telah diketahui bahwa Matematika merupakan suatu ilmu yang dapat membantu menyelesaikan berbagai persoalan di bidang lain, baik bidang teknologi maupun bidang ilmu-ilmu murni. Aljabar merupakan suatu dasar dan salah satu cabang dari ilmu Matematika yang sangat berperan dalam beberapa penerapan pada bidang tersebut.

Salah satu bahasan dari aljabar adalah matriks. Teori matriks itu sendiri telah dikembangkan secara luas termasuk aplikasi-aplikasinya baik dalam Matematika sendiri maupun untuk ilmu-ilmu lainnya. Matriks ada yang singular dan ada yang non singular. Perkalian dua buah matriks atau lebih menarik untuk dipelajari karena sifat perkalian matriks secara umum tidak komutatif. Dalam hal ini suatu matriks jika dikalikan dengan matriks itu sendiri secara berulang-ulang dengan syarat matriksnya persegi, maka akan menghasilkan suatu matriks persegi pula. Matriks yang dikalikan terus menerus itulah yang disebut akar dari matriks hasil perkalian berulangnya. Dalam pembahasan kali ini penulis akan mengkaji matriks dengan elemen–elemennya adalah bilangan kompleks.

Misalkan M_n adalah aljabar dari semua matriks berukuran n x n dengan elemen bilangan kompleks, dan $A \in M_n$. Untuk bilangan bulat m > 1, matriks $B \in M_n$ disebut akar ke-m dari matriks A jika $B^m = A$. Jika A matriks non singular, maka A selalu mempunyai akar ke-m, dan akarnya tidak tunggal.

Skripsi ini didasarkan pada jurnal yang ditulis oleh P. J. Psarrakos dengan judul "On the *m*th Roots of A Complex Matrix". Dalam jurnal tersebut telah dibahas kriteria eksistensi dan ketunggalan akar ke-*m* dari matriks kompleks non singular. Dalam *software* MATLAB ada perintah yang dapat digunakan secara langsung untuk mencari akar ke-*m* dari suatu matriks, tetapi yang dimunculkan hanya satu akar. Oleh karena itu penulis tertarik untuk membuat program menggunakan bahasa pemrograman M-File dalam *software* MATLAB agar semua kemungkinan akarnya dapat dimunculkan. Pembahasan skripsi ini hanya dititikberatkan pada pembuatan program, bukan pada bukti analisisnya.

1.2 Rumusan Masalah

Bagaimana menyusun algoritma dan membuat program untuk menghitung akar ke-*m* dari matriks kompleks non singular dengan semua kemungkinan akarnya dimunculkan.

1.3 Tujuan

Menyusun algoritma dan membuat program untuk menghitung akar ke-*m* dari matriks kompleks non singular, sehingga semua kemungkinan akarnya muncul.

1.4 Manfaat

Dengan algoritma dan program yang sudah dibuat, dapat dengan mudah diketahui beberapa akar ke-*m* dari matriks kompleks non singular yang diberikan.

BAB II

LANDASAN TEORI

Pada bab ini akan dibahas beberapa definisi dan teorema untuk mendukung penyusunan algoritma dan pembuatan program untuk menghitung akar ke-*m* dari matriks kompleks non singular.

2.1. Ruang Vektor.

Definisi 2.1. Ruang vektor V atas lapangan F adalah himpunan tak kosong yang elemen-elemennya dinamakan vektor dan elemen lapangannya disebut skalar, dengan operasi penjumlahan dan pergandaan yang memenuhi sifat-sifat sebagai berikut :

1. Untuk setiap u, v, w $\in$ V, berlaku
    a. $u + v \in V$
    b. $(u + v) + w = u + (v + w)$
    c. terdapat $0 \in V$ sehingga $u + 0 = u$
    d. Untuk setiap $u \in V$ terdapat $-u \in V$ sehingga $u + (-u) = 0$
    e. $u + v = v + u$
2. Untuk setiap u, v $\in$ V dan α, $\beta \in$ F, berlaku
    a. $\alpha(u + v) = \alpha u + \alpha v$
    b. $(\alpha + \beta)u = \alpha u + \beta u$
    c. $(\alpha\beta)u = \alpha(\beta u)$
    d. $1.u = u$

(Jacob, 1990)

Contoh :

Himpunan semua matriks kompleks berukuran *n x n* merupakan ruang vektor atas lapangan bilangan kompleks.

Ruang vektor yang akan dibahas dalam skripsi ini terdefinisi atas lapangan bilangan kompleks dan disebut ruang vektor kompleks. Jika V adalah sebarang ruang vektor, maka himpunan bagian-himpunan bagian tertentu dari V itu sendiri membentuk ruang vektor dengan operasi penjumlahan vektor dan perkalian skalar yang didefinisikan pada V. Selanjutnya akan dipaparkan himpunan bagian seperti itu secara lebih terinci.

Definisi 2.2. Misalkan V adalah ruang vektor atas lapangan F dan S himpunan bagian dari V. S dikatakan ruang bagian *(subspace)* dari V jika S juga merupakan ruang vektor atas lapangan F.

(Jacob, 1990)

Di bawah ini diberikan teorema yang berguna untuk menyelidiki apakah suatu himpunan bagian dari ruang vektor merupakan ruang vektor.

Teorema 2.3. Misalkan V adalah ruang vektor atas lapangan F dan S himpunan bagian dari V. S merupakan ruang bagian dari V jika dan hanya jika untuk setiap u, v $\in$ S dan $\alpha \in$ F berlaku u + v $\in$ S dan αu $\in$ S.

(Jacob, 1990)

Tiap-tiap ruang vektor V mempunyai paling sedikit dua ruang bagian, yaitu V itu sendiri dan himpunan {0} yang terdiri dari vektor nol saja dalam V. Himpunan {0} ini dinamakan ruang bagian nol (*zero subspace*).

Definisi 2.4. Misalkan S dan T masing-masing ruang bagian dari V. Jumlahan dari ruang bagian S dan T, ditulis S + T, didefinisikan sebagai

$$S + T = \{s + t;\ s \in S \text{ dan } t \in T\}.$$

(Jacob, 1990)

Definisi 2.5. Misalkan S dan T masing-masing ruang bagian dari V. Himpunan V dikatakan hasil tambah langsung *(direct sum)* dari S dan T, ditulis $V = S \oplus T$, jika S + T = V dan S $\cap$ T = {0}.

(Cullen, 1966)

Definisi 2.6. Misalkan V ruang vektor atas lapangan F. Himpunan V dikatakan aljabar atas lapangan F jika terhadap perkalian bersifat asosiatif yang memenuhi :

$$u\,(\alpha v) = \alpha\,(uv), \quad \forall\, u, v \in V, \quad \forall\, \alpha \in F$$

(Cullen, 1966)

Contoh :

Himpunan semua matriks kompleks berukuran *n x n* merupakan aljabar atas lapangan bilangan kompleks.

Definisi 2.7. Misalkan V adalah ruang vektor atas lapangan F dan S himpunan bagian dari V. Kombinasi linear *(linear combination)* dari S adalah jumlahan berhingga yang berbentuk $\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n$, dengan $\alpha_i \in F$ dan $v_i \in S$, $i = 1, 2, \ldots, n$.

(Jacob, 1990)

Definisi 2.8. Himpunan vektor-vektor $\{v_1, v_2, \ldots, v_n\}$ dikatakan bebas linear *(linearly independent)* jika kombinasi linear $\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n = 0$ mengakibatkan $\alpha_1 = \alpha_2 = \ldots = \alpha_n = 0$.

(Jacob, 1990)

2.2. Bilangan Kompleks.

Definisi 2.9. Suatu bilangan kompleks merupakan bilangan yang berbentuk a+bi dengan a dan b bilangan real dan i satuan khayal (*imaginary unit*) yang bersifat $i^2 = -1$. Jika $z = a + bi$, maka a dinamakan bagian real dari z, dan b dinamakan bagian imajiner dari z, masing-masing dinotasikan dengan Re{z} dan Im{z}.

(Murray, 1994)

Definisi 2.10. Jika P(x, y) adalah suatu titik di bidang kompleks yang bersesuaian dengan bilangan kompleks $z = x + iy$, maka x dan y dapat ditulis sebagai

$$x = r\cos\theta \qquad y = r\sin\theta,$$

dengan $r = \sqrt{x^2 + y^2}$ merupakan modulus dari z. Argumen dari z, dinotasikan θ, didefinisikan sebagai :

$$\theta = arc\tan\left(y/x\right),\ 0 \le \theta \le 2\pi \text{ (radian)}.$$

Selanjutnya $r(\cos\theta + i\sin\theta) = re^{i\theta}$ dinamakan bentuk kutub dari z.

(Murray, 1994)

2.3. Matriks.

Definisi 2.11. Matriks persegi A dikatakan non singular jika determinannya tidak sama dengan nol, dinotasikan $\det(A) \neq 0$.

(Gantmacher, 1959)

Definisi 2.12. Himpunan semua nilai eigen dari matriks A dinamakan spektrum dari matriks A, dinotasikan dengan $\sigma(A)$, dan didefinisikan sebagai :

$$\sigma(A) := \{\lambda \in C : \det(I\lambda - A) = 0\}.$$

(Psarrakos, 2002)

Definisi 2.13. Misalkan A adalah matriks berukuran *n* x *n*. Vektor tak nol $x \in C^n$ dinamakan vektor eigen *(eigenvector)* dari A jika Ax adalah kelipatan skalar dari x, yakni :

$$Ax = \lambda x$$

untuk suatu skalar λ. Skalar λ dinamakan nilai eigen *(eigenvalue)* dari A, dan x dikatakan vektor eigen yang bersesuaian dengan λ.

(Anton, 1988)

Definisi 2.14. Misalkan A adalah matriks berukuran n x n. Polinomial karakteristik dari A dalam variabel x dinotasikan dengan $C_A(x)$, didefinisikan $C_A(x) := \det(xI - A)$, sedangkan persamaan $C_A(x) = 0$ disebut persamaan karakteristik dari A.

(Cullen, 1966)

Definisi 2.15. Misalkan A adalah matriks berukuran n x n dengan λ adalah nilai eigen dari A. Pangkat tertinggi dari $(x - \lambda)$ yang membagi $C_A(x)$ dinamakan multiplisitas aljabar, yang selanjutnya disebut multiplisitas.

(Jacob, 1990)

Definisi 2.16. Matriks persegi A disebut matriks segitiga atas jika semua elemen di bawah diagonal utama adalah nol, yaitu

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{bmatrix}$$

(Gantmacher, 1959)

Definisi 2.17. Matriks A disebut nilpotent indeks k jika $A^k = 0$ dan $A^{k-1} \neq 0$.

(Gantmacher, 1959)

Definisi 2.18. Misalkan A dan B masing-masing adalah matriks *n x n*. Jika ada matriks invertibel P sehingga $A = PBP^{-1}$ maka A dan B dikatakan similar.

(Jacob, 1990)

Definisi 2.19. Matriks persegi dengan bentuk :

$$J_i = \begin{bmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & \cdots & 0 \\ 0 & 0 & \lambda_i & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & \cdots & \lambda_i \end{bmatrix}, \lambda_i \in C, i = 1, 2, \ldots, \xi$$

disebut blok Jordan yang bersesuaian dengan nilai eigen λ_i. Matriks blok Jordan di atas dapat disajikan dalam bentuk $I_{k_j}\lambda_j + N_{k_j}$ dengan

$$I_{k_i}\lambda_i = \begin{bmatrix} \lambda_i & 0 & 0 & \cdots & 0 \\ 0 & \lambda_i & 0 & \cdots & 0 \\ 0 & 0 & \lambda_i & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_i \end{bmatrix} \text{ dan } N_{k_i} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ \cdots & \cdots & \cdots & \ddots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

Selanjutnya dari matriks blok Jordan dapat disusun matriks Jordan

$$J_A = \begin{bmatrix} J_1 & 0 & \cdots & 0 \\ 0 & J_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_\xi \end{bmatrix}$$

dengan J_i adalah blok Jordan yang bersesuaian dengan λ_i yang dapat disajikan pula sebagai $J_A = \oplus_{i=1}^{\xi}\left(I_{k_i}\lambda_i + N_{k_i}\right)$.

(Gohberg, 1982)

2.4. MATLAB.

MATLAB (*Matrix Laboratory*) adalah suatu bahasa pemrograman matematika yang dibentuk dengan dasar berpikir menggunakan sifat dan bentuk matriks. MATLAB merupakan *software* yang paling efisien untuk perhitungan numerik berbasis matriks. MATLAB juga merupakan suatu paket program yang memungkinkan membuat program sendiri walaupun di dalamnya sudah banyak tersedia program internal yang siap digunakan. Kelebihan dari paket program ini adalah baik program internal maupun program yang pernah dibuat dapat digunakan sebagai sub program dari program yang akan dibuat.

Beberapa perintah internal yang digunakan dalam skripsi ini antara lain :

a. A=[ ]

[ ] digunakan sebagai perintah untuk membuat matriks dengan spasi menandakan kolom dan tanda titik koma (;) menandakan pemisahan baris.

Bentuk :

```
» A=[1 -3 -2;-1 1 -1;2 4 5]
A =
     1    -3    -2
    -1     1    -1
     2     4     5
```

b. det( )

Perintah det( ) digunakan untuk mengetahui determinan suatu matriks.

Bentuk :

```
» det(A)
ans =
     12
```

c. $[X, J] = \text{jordan}(\ )$

Perintah di atas digunakan untuk mengetahui matriks Jordan dan rantai Jordan dari suatu matriks. Hasil dari perintah tersebut adalah X yang menunjukkan rantai Jordan dan J menunjukkan matriks Jordan.

Bentuk :

```
» [X,J]=jordan(A)
X =
    -1    -1     1
     0    -1     0
     1     2     0
J =
     3     0     0
     0     2     1
     0     0     2
```

d. $A\^\left(\frac{1}{m}\right)$

Perintah di atas adalah perintah untuk mencari akar ke-*m* dari matriks A.

Bentuk :

```
» A^(1/3)

ans =

   1.0499-0.0000i  -0.5746-0.0000i  -0.3923
  -0.2100-0.0000i   1.0499-0.0000i  -0.2100
   0.4200+0.0000i   0.7846+0.0000i   1.8622
```

Matriks yang terakhir merupakan akar ke-3 dari matriks A yang didapatkan dengan menggunakan *software* MATLAB.

BAB III

METODE PENELITIAN

Langkah – langkah penyelesaian yang berkaitan dengan tujuan penelitian adalah sebagai berikut :

1. Mengkaji definisi dan teorema yang berhubungan dengan eksistensi akar ke-m dari matriks kompleks non singular.
2. Menyusun algoritma untuk mencari akar ke-m dari matriks kompleks non singular.
3. Membuktikan kevalidan dari algoritma yang disusun berdasarkan **Lemma 4.3.** dan **Teorema 4.4.**.
4. Membuat program untuk mencari akar ke-m dari matriks kompleks non singular.
5. Menerapkan program untuk mencari akar ke-m dari matriks kompleks non singular yang diberikan.

BAB IV

HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas tentang struktur Jordan yang digunakan dalam penyusunan algoritma yang selanjutnya akan dibuat program untuk menghitung akar ke-*m* dari matriks kompleks non singular.

4.1. Struktur Jordan.

Matriks non singular A akan selalu mempunyai akar ke-*m*. Akar ini tidak tunggal dan struktur Jordannya berhubungan dengan struktur Jordan dari A. Matriks B disebut akar ke-*m* dari A jika dan hanya jika $B^m = A$. Untuk mengetahui struktur Jordan diperlukan definisi sebagai berikut.

Definisi 4.1. Polinomial matriks dalam variabel kompleks λ dinotasikan dengan $P(\lambda)$ didefinisikan sebagai $P(\lambda) := I_n\lambda^m - A$ dengan I_n adalah matriks identitas berordo-*n*.

(Psarrakos, 2002)

Definisi 4.2. Himpunan vektor $\{x_1, x_2, \ldots, x_k\}$ yang memenuhi persamaan-persamaan :

$$\begin{aligned} P(\omega_1)x_1 &= 0 \\ P(\omega_1)x_2 + \frac{1}{1!}P^{(1)}(\omega_1)x_1 &= 0 \\ &\vdots \\ P(\omega_1)x_k + \frac{1}{1!}P^{(1)}(\omega_1)x_{k-1} + \cdots + \frac{1}{k!}P^{(k)}(\omega_1)x_1 &= 0 \end{aligned}$$

dengan $P^{(i)} = \dfrac{d^i\, P(\lambda)}{d\,\lambda^i}$, i = 1, 2, ..., *k*, disebut rantai Jordan dari $P(\lambda)$ dengan panjang *k* yang bersesuaian dengan nilai eigen $\omega_1 \in C$ dan vektor eigen $x_1 \in C^n$.

(Psarrakos, 2002)

Dengan definisi ini, untuk setiap $\omega_j \in \sigma(A)$, $j = 1,2,\ldots,\xi$ terdapat rantai Jordan $X_j = (x_{j,1}, x_{j,2}, \ldots, x_{j,k_j})$. Jika A mempunyai nilai eigen $\omega_1, \omega_2, \ldots, \omega_\xi$ terdapat matriks $X_A = [x_{1,1}, \ldots, x_{1,k_1} \quad x_{2,1}, \ldots, x_{2,k_2} \quad \ldots \quad x_{\xi,1}, \ldots, x_{\xi,k_\xi}]$ dengan $(k_1 + k_2 + \cdots + k_\xi = n)$, sehingga berlaku $A = X_A J_A X_A^{-1}$ dengan J_A adalah matriks Jordan dari A. Dengan kata lain matriks A similar dengan matriks Jordannya.

Mengingat struktur Jordan akar ke-*m* dari matriks kompleks non singular A berhubungan dengan struktur Jordan dari A, maka untuk menentukan rantai Jordan dari $P(\lambda)$ dapat digunakan lemma berikut.

Lemma 4.3. Misalkan $\{x_1, x_2, \ldots, x_k\}$ adalah rantai Jordan dari $A \in M_n$ (dengan suku-sukunya bebas linier) berkaitan dengan nilai eigen yang tidak nol $\omega_1 = r_1 e^{i\varphi_1} \in \sigma(A)$ $(r_1 > 0, \quad \varphi_1 \in [0, 2\pi))$, dan $P(\lambda) = I_n \lambda^m - A$. Untuk setiap nilai eigen

$$r_1^{\frac{1}{m}} e^{i\frac{\varphi_1 + 2(t-1)\pi}{m}} \in \sigma(P) \quad ; \quad t = 1, 2, \ldots, m$$

polinomial matriks $P(\lambda)$ memiliki rantai Jordan dengan bentuk :

$$\begin{aligned} y_1 &= x_1 \\ y_2 &= a_{1,1}x_2 \\ y_3 &= a_{2,1}x_2 + a_{2,2}x_3 \\ &\vdots \\ y_k &= a_{k,1}x_2 + a_{k,2}x_3 + \cdots + a_{k,k}x_k \end{aligned} \tag{4.1}$$

dengan koefisien $a_{i,j}$ $(1 \le j \le i \le k)$ bergantung pada t dan untuk setiap i = 1, 2, ... , k, $a_{i,i} = \left(m r_1^{\frac{m-1}{m}} e^{i(m-1)\frac{\varphi_1 + 2(t-1)\pi}{m}} \right)^i \neq 0$. Dengan demikian vektor-vektor $y_1, y_2, \ldots, y_k$ bebas linier.

(Psarrakos, 2002)

Rantai Jordan $\{y_1, y_2, \ldots, y_k\}$ dari $P(\lambda)$ pada lemma di atas berhubungan dengan Rantai Jordan $\{x_1, x_2, \ldots, x_k\}$ dari matriks A yang bergantung pada pemilihan $t \in \{1, 2, \ldots, m\}$. Misalkan $\mu_j \in \sigma(P(\lambda))$ sehingga $\mu_j^m = \omega_j$ dengan $j = 1, 2, \ldots, \xi$. Himpunan vektor-vektor yang membentuk rantai Jordan dari $P(\lambda)$ yang bersesuaian dengan nilai eigen μ_j adalah

$$Y_A(s_1, s_2, \ldots, s_\xi) = [y_{1,1} \cdots y_{1,k_1} \quad y_{2,1} \cdots y_{k_2} \quad \cdots \quad y_{\xi,1} \cdots y_{\xi,k_\xi}]$$

yang bergantung pada pemilihan sebuah pasangan bilangan $(s_1, s_2, \ldots, s_\xi)$, $s_j \in \{1, 2, \ldots, m\}$ $(j = 1, 2, \ldots, \xi)$.

Teorema 4.4. Misalkan A $\in M_n$ sebuah matriks kompleks non singular dengan matriks Jordan $J_A = \oplus_{j=1}^{\xi} \left(I_{k_j} \omega_j + N_{k_j} \right)$ dan J-spektrum

$$\sigma_J(A) = \{\omega_1 = r_1 e^{i\varphi_1}, \omega_2 = r_2 e^{i\varphi_2}, \ldots, \omega_\xi = r_\xi e^{i\varphi_\xi}\}.$$

Untuk $m > 1$, $X_A \in M_n$ merupakan matriks non singular, sehingga $A = X_A J_A X_A^{-1}$. Jika pasangan bilangan $(s_1, s_2, \ldots, s_\xi)$ dengan $s_j \in \{1,2,\ldots,m\}$ $(j = 1,2,\ldots,\xi)$ dan matriks $Y_A(s_1, s_2, \ldots, s_\xi)$ adalah rantai Jordan dari $P(\lambda)$ maka matriks

$$B = Y_A(s_1, s_2, \ldots, s_\xi)\left(\oplus_{j=1}^{\xi}\left(I_{k_j} r_j^{\frac{1}{m}} e^{i\frac{\varphi_j + 2(s_j - 1)\pi}{m}} + N_{k_j}\right)\right) Y_A(s_1, s_2, \ldots, s_\xi)^{-1}$$

adalah akar ke-*m* dari A.

(Psarrakos, 2002)

4.2. Algoritma Program.

Berdasarkan **Lemma 4.3.** dan **Teorema 4.4.** selanjutnya disusun algoritma untuk menghitung akar ke-*m* dari matriks kompleks non singular. Sedangkan program untuk menghitung akar ke-*m* dari matriks kompleks non singular ada pada lampiran. Algoritma yang disusun adalah sebagai berikut :

1. Input matriks kompleks *n x n* (A).
2. Apakah matriks kompleks A non singular atau tidak?

   Jika determinan tidak sama dengan nol ($\det(A) \neq 0$) maka matriks A adalah non singular (Definisi 2.11.). Jika determinan sama dengan nol ($\det(A) = 0$) maka matriks singular sehingga program selesai.
3. Menghitung nilai eigen dan banyaknya nilai eigen yang berbeda (ξ) dari matriks kompleks A.

Dengan **Definisi 2.12.** dapat dihitung nilai eigen dari A menggunakan persamaan $\det(\lambda I_n - A) = 0$. Selanjutnya dihitung banyaknya nilai eigen yang berbeda (ξ) beserta multiplisitasnya sesuai dengan **Definisi 2.15.**.

4. Membuat matriks Jordan dan rantai Jordan dari A.

    Matriks Jordan dari A adalah matriks dengan elemen diagonal utamanya adalah nilai eigen-nilai eigen dari A. Matriks Jordan terdiri dari beberapa blok Jordan yang elemen diagonal utamanya adalah nilai eigen dari A. Dengan menggunakan **Definisi 4.2.** dapat dihitung rantai Jordan dari A dengan menggunakan polinomial matriks pada **Definisi 4.1.** sehingga $A = X_A J_A X_A^{-1}$ dengan J_A adalah matriks Jordan dari A dan X_A adalah rantai Jordan dari A. Apabila banyaknya nilai eigen yang berbeda (ξ) sama dengan *n* maka vektor eigen-vektor eigen dari A membentuk rantai Jordan terhadap nilai eigen yang bersesuaian pada matriks Jordan dari A.

5. Input bilangan bulat positif (*m*) dengan *m* menunjukkan akar ke-*m*.
6. Membuat matriks kemungkinan.

    Matriks kemungkinan adalah matriks dengan jumlah kolomnya adalah banyaknya nilai eigen yang berbeda (ξ) dan jumlah barisnya adalah m^{ξ} yang elemen-elemenya merupakan kemungkinan-kemungkinan untuk menghitung akar ke-*m* dari matriks kompleks non singular. Baris-

baris dari matriks ini adalah pasangan bilangan $(s_1, s_2, s_3, \ldots, s_\xi)$, dengan $s_j \in \{1,2,3,\ldots,m\}$ $(j = 1,2,\ldots,\xi)$.

7. Mengambil satu baris dari matriks yang dibuat pada langkah-6.

   Karena matriks non singular selalu mempunyai akar ke-*m* dan akarnya tidak tunggal, maka jumlah kemungkinan akarnya adalah jumlah baris pada matriks kemungkinan. Dengan mengambil satu baris pada matriks kemungkinan maka yang dimunculkan adalah salah satu kemungkinan dari akar ke-*m* matriks kompleks yang diberikan.

8. Menghitung nilai eigen baru.

   Nilai eigen baru adalah bentuk kutub dari nilai eigen A sesuai dengan persamaan $r_0^{\frac{1}{m}} e^{i\frac{\varphi_0 + 2(t-1)\pi}{m}}$ dengan r_0 adalah modulus dari nilai eigen, φ_0 adalah argumen dari nilai eigen dan t adalah elemen-elemen baris yang diambil pada langkah-7 sehingga didapatkan nilai eigen baru. Nilai eigen yang baru ini adalah nilai eigen dari $P(\lambda)$.

9. Menghitung matriks segitiga T dengan elemen-elemennya adalah koefisien pada persamaan rantai Jordan $P(\lambda)$.

   Persamaan (4.1) pada **Lemma 4.3.** dapat disajikan dalam bentuk

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & a_{1,1} & 0 & \cdots & 0 \\ 0 & a_{2,1} & a_{2,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & a_{k,1} & a_{k,2} & \cdots & a_{k,k} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_k \end{bmatrix}$$

ditulis $Y_j = T_j X_j$, $j = 1,2,\ldots,\xi$ dengan $\{x_1, x_2, \ldots, x_k\}$ adalah rantai Jordan dari A dengan panjang *k* dan $\{y_1, y_2, \ldots, y_k\}$ adalah rantai Jordan dari $P(\lambda)$.

10. Menghitung matriks Y_A dengan menggunakan matriks pada langkah-9 dan rantai Jordan dari A.

    Misalkan Y_j, $j = 1,2,\ldots,\xi$ adalah rantai Jordan dari $P(\lambda)$ yang bersesuaian dengan nilai eigen μ_j, $j = 1,2,\ldots,\xi$ yang elemen-elemennya adalah vektor-vektor yang membentuk rantai Jordan tersebut. Matriks Y_A disusun dari vektor kolom-vektor kolom yang membentuk rantai Jordan Y_j $j = 1, 2, \ldots, \xi$.

11. Membuat matriks Jordan dari $P(\lambda)$ dengan mengganti elemen pada diagonal utama matriks Jordan dari A dengan nilai eigen baru yang bersesuaian hasil perhitungan pada langkah-8 yang berbentuk $J_{P(\lambda)}$.

12. Menghitung akar ke-*m* dari matriks kompleks A dengan menggunakan persamaan pada **Teorema 4.4.**.

13. Akhir dari program.

    Pada akhir program akan dimunculkan pertanyaan "Apakah anda ingin mencari akar yang lain (Y/T)?" dengan jawaban yang harus diinputkan adalah "Y" untuk mencari akar yang lain dengan mengulang kembali ke langkah-7 dan seterusnya. Sedangkan "T" untuk keluar dari program.

Contoh : Menghitung akar ke-*3* dari matriks $A = \begin{bmatrix} 1 & -3 & -2 \\ -1 & 1 & -1 \\ 2 & 4 & 5 \end{bmatrix}$ secara manual.

Langkah-1 : Input matriks kompleks.

Dengan cara manual sudah jelas.

Langkah-2 : Apakah matriks A non singular?

$$\begin{aligned} \det(A) &= \begin{vmatrix} 1 & -3 & -2 \\ -1 & 1 & -1 \\ 2 & 4 & 5 \end{vmatrix} \\ &= (1)(1)(5) + (-3)(-1)(2) + (-2)(-1)(4) - (-2)(1)(2) - (-3)(-1)(5) - (1)(-1)(4) \\ &= 12 \end{aligned}$$

Jika determinannya tidak sama dengan nol ($\det(A) \neq 0$) maka dinamakan matriks non singular. Jika determinan dari matriks A sama dengan nol maka dinamakan matriks singular dan program tidak dapat dijalankan.

Langkah-3 : Menghitung nilai eigen dan menghitung banyaknya nilai eigen yang berbeda dari matriks A.

$$\det(I_n\lambda - A) = 0$$

$$\begin{vmatrix} \lambda - 1 & 3 & 2 \\ 1 & \lambda - 1 & 1 \\ -2 & -4 & \lambda - 5 \end{vmatrix} = 0$$

$$(\lambda - 1)^2(\lambda - 5) - 6 - 8 + 4(\lambda - 1) - 3(\lambda - 5) + 4(\lambda - 1) = 0$$
$$(\lambda - 3)(\lambda - 2)^2 = 0$$

Jadi nilai eigen dari A ada 2, yaitu nilai eigen 2 dengan multiplisitas 2 dan nilai eigen 3 dengan multiplisitas 1.

Langkah-4 : Matriks Jordan dan rantai Jordan dari A.

$$P(\lambda) = I_n\lambda - A$$

- Untuk nilai eigen 3

$P(\lambda)x_1 = 0$

$$\begin{pmatrix} 2 & 3 & 2 \\ 1 & 2 & 1 \\ -2 & -4 & -2 \end{pmatrix}\begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \text{ diperoleh } \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \text{ sehingga}$$

$$X_1 = \left\{\begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}\right\}.$$

- Untuk nilai eigen 2

$P(\lambda)x_1 = 0$

$$\begin{pmatrix} 1 & 3 & 2 \\ 1 & 1 & 1 \\ -2 & -4 & -3 \end{pmatrix}\begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \text{ diperoleh } \begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \end{pmatrix} = \begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix}.$$

$$P(\lambda)x_2 + \frac{1}{1!}P^{(1)}(\lambda)x_1 = 0$$

$$\begin{pmatrix} 1 & 3 & 2 \\ 1 & 1 & 1 \\ -2 & -4 & -3 \end{pmatrix}\begin{pmatrix} x_{21} \\ x_{22} \\ x_{23} \end{pmatrix} + \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \text{ diperoleh } \begin{pmatrix} x_{21} \\ x_{22} \\ x_{23} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

sehingga $X_2 = \left\{\begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\right\}$.

Dari X_1 dan X_2 diperoleh matriks $X_A = \begin{pmatrix} -1 & -1 & 1 \\ 0 & -1 & 0 \\ 1 & 2 & 0 \end{pmatrix}$ dan matriks

Jordan $J_A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}$ sehingga memenuhi $A = X_A J_A X_A^{-1}$.

Langkah-5 : Karena akan dihitung akar ke-*3* dari matriks A, maka *m*=3.

Langkah-6 : Membuat matriks kemungkinan.

Diketahui banyaknya nilai eigen yang berbeda 2 dan *m*=3, maka matriks kemungkinannya adalah

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

Baris-baris dari matriks kemungkinan ini adalah pasangan bilangan (s_1, s_2), dengan $s_1, s_2 \in \{1,2,3\}$.

Langkah-7 : Mengambil satu baris dari matriks kemungkinan yang didapatkan dari langkah-6.

Misalkan yang dipilih adalah baris ke-4 maka $t = (2 \quad 1)$.

Langkah-8 : Menghitung nilai eigen baru.

Untuk menghitung nilai eigen yang baru digunakan persamaan $\mu_j = r_j^{\frac{1}{m}} e^{i\frac{\varphi_j + 2(t_j - 1)\pi}{m}}$, $j = 1, 2$ dengan r_j adalah modulus, φ_j adalah argument dan t_j elemen-elemen baris yang diambil pada langkah-7.

- Untuk nilai eigen 3

$$r_1 = \sqrt{3^2 + 0^2} = 3$$

$\varphi_1 = \text{arc tg}\left(0/3\right) = 0$

$t_1 = 2$ (elemen dari t pada langkah-7)

Nilai eigen barunya $\mu_1 = -0.7211 + 1.2490i$

- Untuk nilai eigen 2

  $r_2 = \sqrt{2^2 + 0^2} = 2$

  $\varphi_2 = \text{arc tg}\left(0/2\right) = 0$

  $t_2 = 1$ (elemen dari t pada langkah-7)

  Nilai eigen barunya $\mu_2 = 1.2599$

Nilai eigen yang baru ini adalah bentuk kutub dari nilai eigen A yang bersesuaian dan berdasarkan pemilihan pasangan bilangan pada langkah-7.

Langkah-9 : Menghitung matriks T.

Untuk nilai eigen μ_1 dengan multiplisitas 1 maka matriks $T_1 = [1]$. Sedangkan untuk nilai eigen μ_2 dengan multiplisitas 2 matriks $T_2 = \begin{bmatrix} 1 & 0 \\ 0 & 3\mu_2^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 4.7622 \end{bmatrix}$ sehingga diperoleh matriks

$$T = T_1 \oplus T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 4.7622 \end{bmatrix}$$

Matriks T di atas adalah koefisien-koefisien dari persamaan rantai Jordan $P(\lambda)$.

Langkah-10 : Menghitung matriks Y_A yaitu rantai Jordan dari $P(\lambda)$.

Berdasarkan persamaan (4.1) dengan nilai eigen $\mu_1 = -0.7211 + 1.2490i$ dan $X_1 = \left\{ \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \right\}$ diperoleh $Y_1 = T_1 X_1 = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$. Untuk nilai eigen $\mu_2 = 1.2599$ dan $X_2 = \left\{ \begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right\}$ diperoleh $Y_2 = T_2 X_2 = \begin{pmatrix} -1 & 4.7622 \\ -1 & 0 \\ 2 & 0 \end{pmatrix}$

sehingga didapatkan $Y_A = \begin{bmatrix} Y_1 & Y_2 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 4.7622 \\ 0 & -1 & 0 \\ 1 & 2 & 0 \end{bmatrix}$.

Langkah-11 : Menghitung matriks Jordan dari $P(\lambda)$.

Dengan mengganti elemen diagonal utama matriks Jordan A dengan nilai eigen baru yang bersesuaian maka diperoleh

$$J_{P(\lambda)} = \begin{bmatrix} -0.7211 + 1.2490i & 0 & 0 \\ 0 & 1.2599 & 1 \\ 0 & 0 & 1.2599 \end{bmatrix}.$$

Langkah-12 : Menghitung akar ke-*3* dari A.

Berdasarkan **Teorema 4.4.** diperoleh akar ke-3 dari A dengan persamaan

$$\begin{aligned} B &= Y_A J_{P(\lambda)} Y_A^{-1} \\ &= \begin{bmatrix} -1 & -1 & 4.7622 \\ 0 & -1 & 0 \\ 1 & 2 & 0 \end{bmatrix} \begin{bmatrix} -0.7211 + 1.2490i & 0 & 0 \\ 0 & 1.2599 & 1 \\ 0 & 0 & 1.2599 \end{bmatrix} \begin{bmatrix} -1 & -1 & 4.7622 \\ 0 & -1 & 0 \\ 1 & 2 & 0 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} 1.0499 & 3.7521 - 2.4980i & 1.7711 - 1.2490i \\ -0.2100 & 1.0499 & -0.2100 \\ 0.4200 & -3.5421 + 2.4980i & -0.3012 + 1.2490i \end{bmatrix} \end{aligned}$$

Langkah-13 : Akhir program.

Dalam akhir program ini akan ditanyakan apakah akan dicari akar yang lain. Jika jawabannya "Ya" maka ulangi lagi langkah-7 dan seterusnya dengan pemilihan baris pada langkah-7 berbeda dengan pemilihan baris yang telah dilakukan. Jika jawabannya "Tidak" maka program selesai.

Program yang telah dibuat berdasarkan algoritma di atas dapat digunakan untuk menghitung akar ke-m dari matriks kompleks non singular. Tetapi bila ukuran matriks terlalu besar dan mempunyai nilai eigen bilangan kompleks semua dengan bagian imajiner dari nilai eigen adalah tidak sama dengan nol maka program ini membutuhkan waktu lama sehingga hasilnya dapat dimunculkan, mungkin juga sampai program *not responding*. Hal ini juga berlaku jika akan dihitung akar ke-m dari matriks tersebut dengan m yang sangat besar. Sebagai contoh jika akan dihitung akar ke-3 dari matriks $A = \begin{bmatrix} 1 & -3 & -1 & -2 \\ -2 & 1 & 4 & -5 \\ 3 & 4 & 1 & -2 \\ -5 & 1 & -4 & 1 \end{bmatrix}$ maka program ini membutuhkan waktu yang lama agar hasilnya dimunculkan, mungkin juga sampai program *not responding*.

Dalam kasus lain, apabila program ini diterapkan pada matriks A dari contoh, maka dapat dihitung akar ke-m dari A. Nilai m ini bisa sampai 100. Hal ini dikarenakan nilai eigen dari matriks A tersebut adalah bilangan real semua. Tetapi jika ukuran matriks terlalu besar, meskipun semua nilai eigennya bilangan real, kerja dari program sehingga hasilnya dapat dimunculkan juga membutuhkan waktu yang lama. Faktor lain yang juga menghambat kerja dari program adalah keterbatasan dari memori internal komputer dan memori dari *software* yang dipakai.

BAB V

KESIMPULAN DAN SARAN

5.1. KESIMPULAN.

Berdasarkan algoritma yang disusun dapat dibuat program untuk menghitung akar ke-*m* dari matriks kompleks non singular yang diberikan. Program ini dibuat menggunakan bahasa pemrograman M-File dalam *software* MATLAB. Hasil dari program ini adalah semua kemungkinan akar ke-*m* dari matriks kompleks yang diberikan dapat dimunculkan.

Jika matriks yang akan dihitung akar ke-*m*-nya mempunyai nilai eigen bilangan kompleks semua dengan bagian imajiner dari nilai eigen adalah tidak nol, maka membutuhkan waktu yang lama untuk dapat memunculkan akar ke-*m* hasil dari program ini. Hal ini juga berlaku untuk nilai *m* yang sangat besar untuk matriks tersebut.

5.2. SARAN.

Pada skripsi ini dibuat program untuk menghitung akar ke-*m* dari matriks kompleks non singular. Dalam program ini masih terdapat kelemahan. Selanjutnya dapat dibuat program untuk menghitung akar ke-*m* dari matriks kompleks non singular yang sempurna sehingga kelemahan tersebut dapat teratasi dan juga dapat dibuat program untuk menghitung akar ke-*m* dari matriks kompleks singular.

DAFTAR PUSTAKA

Anton, 1998, *Aljabar Linier Elementer*, Erlangga, Jakarta.

Cullen, C. G., 1966, *Matrices and Linear Transformation*, Addison-Wesley Publishing Company, Pensylvania.

Gantmacher, F. R., 1959, *The Theory of Matrices*, Vol. 1, Chelsea Publishing Company, New York.

Gohberg, et al, 1982, *Matrix Polynomial*, Academic Press, New York.

Jacob, 1990, *Linear Algebra*, W. H. Freeman and Company, New York.

Psarrakos, P. J., 2002, *On the m-th Roots of a Complex Matrix*, The Electronic Journal of Linear Algebra A Publication of the International Linear Algebra Society, Volume 9, pp.32-41.

Lampiran 1 : Program menghitung nilai eigen dan multiplisitas dari matriks kompleks yang diberikan.

```
function[ne,jml]=multiplisitas(matriks);

n=length(matriks);
[Xa,J]=jordan(matriks);
q=0;
for i=1:n
   l=1;
   p=0;
   for j=1:i
      if i~=j
         if J(i,i)==J(j,j)
            l=l+1;
         end;
      end;
   end;
   if l==1
      a(i,1)=J(i,i);
      for k=1:n
         if a(i,1)==J(k,k)
            p=p+1;
         end;
      end;
      a(i,2)=p;
      q=q+1;
   end;
end;
ne=a;
jml=q;
```

Lampiran 2 : Program untuk menghitung matriks kemungkinan.

```
function[a]=kemungkinan(m,juml);

brs=m^juml;
a=ones(brs,juml);
for i=2:brs
   a(i,juml)=a(i-1,juml)+1;
   if a(i-1,juml)==m
      a(i,juml)=1;
   end;
end;
if juml>1
   k=1;
   while k<=(juml-1)
      j=juml-k;
      b=m^k;
      for i=2:brs
         if (bagi(i-1,b)==bagi(i-2,b))
            a(i,j)=a(i-1,j);
         elseif a(i-1,j)==m
            a(i,j)=1;
         else a(i,j)=a(i-1,j)+1;
         end;
      end;
      k=k+1;
   end;
end;
```

Lampiran 3 : Program untuk menghitung nilai eigen baru.

```
function[neweig]=neweig(matriks,m,x);

[Xa,Ja]=jordan(matriks);
[ne,juml]=multiplisitas(matriks);
for i=1:juml

e(i)=abs(ne(i,1))^(1/m)*exp(complex(0,(((angle(ne(i,1))/pi
     +2*(x(i)-1))/m)*180)*pi/180));

end;
neweig=e;
```

Lampiran 4 : Program untuk menghitung matriks T.

- Program turunan.

```
function[D]=turunan(matriks,m,x);

[Xa,Ja]=jordan(matriks);
[ne,juml]=multiplisitas(matriks);
eig_baru=neweig(matriks,m,x);
for i=1:juml
    for j=1:length(matriks)
        if j<=m
            D(i,j)=((factorial(m)/factorial(m-j))*
                    eig_baru(i)^(m-j))/factorial(j);
         else D(i,j)=0;
         end;
    end;
end;
```

- Program matriks T.

```
function[T,D]=matriksT(matriks,m,x);

n=length(matriks);
T=zeros(n);
[Xa,Ja]=jordan(matriks);
[ne,juml]=multiplisitas(matriks);
eig_baru=neweig(matriks,m,x);
D=turunan(matriks,m,x);
p=0;
for i=1:juml
   p=p+ne(i,2);
   v=0;
   for j=1:ne(i,2)
      y=0;
      if j==1
         t(j,1)=1;
      else
       for k=2:j
             for l=1:(j-1)
                y=y+D(i,l)*t(j-l,k-1)/factorial(l);
             end;
             t(j,k)=y;
       end;
      end;
  end;
  for j=1:ne(i,2)
      if i==1
          q=1;
          for a1=q:p
            for b1=q:p
               T(a1,b1)=t(a1,b1);
            end;
          end;
```

```
            q=p+1;
        else
            r=1;
            for a1=q:p
                s=1;
                for b1=q:p
                    T(a1,b1)=t(r,s);
                    s=s+1;
                end;
                r=r+1;
            end;
            q=p+1;
        end;
    end;
end;
```

Lampiran 5 : Program untuk menghitung akar ke-*m* dari matriks kompleks non singular.

- Akar yang dimunculkan hanya satu akar.

```
function[B]=akarbaru(matriks,m);

[nrow,ncol]=size(matriks);
if nrow==ncol
   if det(matriks)~=0
      [Xa,Ja]=jordan(matriks);
      [ne,juml]=jumlah(matriks);
      matriks_kemungkinan=kemungkinan(m,juml);
      display(matriks_kemungkinan);
      i=input('Pilih satu baris dari matriks di atas !!!
               Baris ke- ');
      if and((i>=1),(i<=m^juml))
         x=matriks_kemungkinan(i,:);
         e=neweig(matriks,m,x);
         T=matriksT(matriks,m,x);
         Jb=Ja;
         for i=1:juml
             for j=1:length(matriks)
                 if ne(i,1)==Ja(j,j)
                     Jb(j,j)=e(i);
                 end;
             end;
         end;
         for i=1:length(matriks)
             y=0;
             for j=1:length(matriks)
                 y=y+T(i,j)*Xa(:,j);
             end;
             Yb(:,i)=y;
         end;
         B=Yb*Jb*inv(Yb);
      else msg1=('MAAF PILIHAN ANDA TIDAK BENAR ???');
         disp(msg1);
         msg2=('TOLONG ULANGI PROGRAM INI DAN PERHATIKAN
                PERINTAHNYA!!!');
         disp(msg2);
      end;
   else msg=('MATRIKS YANG ANDA INPUT BUKAN MATRIKS NON
             SINGULAR !!!');
      disp(msg);
   end;
else msg=('MATRIKS YANG ANDA INPUT HARUS MATRIKS PERSEGI !!!
          OK!');
   disp(msg);
end;
```

- Akar yang dimunculkan lebih dari satu akar.

```
function akar(matriks,m);

[nrow,ncol]=size(matriks);
if nrow==ncol
   if det(matriks)~=0
      [Xa,Ja]=jordan(matriks);
      [ne,juml]=multiplisitas(matriks);
      matriks_kemungkinan=kemungkinan(m,juml);
      r='Y';
      while or(r=='Y',r=='y')
         display(matriks_kemungkinan);
         i=input('Pilih satu baris dari matriks di atas !!!
                  Baris ke- ');
         if and((i>=1),(i<=m^juml))
            x=matriks_kemungkinan(i,:);
            e=neweig(matriks,m,x);
            T=matriksT(matriks,m,x);
            Jb=Ja;
            for i=1:juml
                for j=1:length(matriks)
                    if ne(i,1)==Ja(j,j)
                       Jb(j,j)=e(i);
                    end;
                end;
            end;
            for i=1:length(matriks)
                y=0;
                for j=1:length(matriks)
                    y=y+T(i,j)*Xa(:,j);
                end;
                Yb(:,i)=y;
            end;
            B=Yb*Jb*inv(Yb);
            display(B);
         else msg1=('MAAF PILIHAN ANDA TIDAK BENAR ???');
            disp(msg1);
            msg2=('TOLONG ULANGI PROGRAM INI DAN PERHATIKAN
                   PERINTAHNYA');
            disp(msg2);
         end;
         r=input('Apa anda ingin coba cari akar yang lain
                 (Y/T)? ','s');
         if and(r~='Y',r~='y')
            msg=('Terima kasih telah menggunakan paket
                  program ini !!!');
            disp(msg);
         end;
      end;
   else msg=('MATRIKS YANG ANDA INPUT BUKAN MATRIKS NON
             SINGULAR !!!'); disp(msg);
   end;
else msg=('MATRIKS YANG ANDA INPUT HARUS MATRIKS PERSEGI !!!
         OK!'); disp(msg);
end;
```

Lampiran 6 : Penerapan program untuk menghitung akar ke-m dari matriks kompleks non singular.

Menghitung akar ke-3 dari matriks $A = \begin{bmatrix} 1 & -3 & -2 \\ -1 & 1 & -1 \\ 2 & 4 & 5 \end{bmatrix}$ dengan bantuan *software* MATLAB dan program yang telah dibuat berdasarkan algoritma yang disusun.

Langkah 1 : » `A=[1 -3 -2;-1 1 -1;2 4 5]`

```
A =

     1    -3    -2
    -1     1    -1
     2     4     5
```

Jika matriks A tidak berukuran n x n maka akan muncul peringatan "`MATRIKS YANG ANDA INPUT HARUS MATRIKS PERSEGI !!! OK!`" kemudian program selesai.

Langkah 2 : » `det(A)`

```
ans =

    12
```

Jika determinan tidak sama dengan nol maka matriks A non singular dan jika determinan sama dengan nol maka matriks A singular sehingga akan muncul peringatan "`MATRIKS YANG ANDA INPUT BUKAN MATRIKS NON SINGULAR !!!`" kemudian program selesai.

Langkah 3 : » `[P,Q]=multiplisitas(A)`

```
P =

     3     1

     2     2

Q =

     2
```

Dari hasil di atas diketahui bahwa kolom-1 dari matriks P adalah nilai eigen dari A dan kolom-2 adalah multiplisitas nilai eigen. Sedangkan matriks Q adalah banyaknya nilai eigen yang berbeda.

Langkah 4 : » `[Xa,Ja]=jordan(A)`

```
Xa =

    -1    -1     1

     0    -1     0

     1     2     0

Ja =

     3     0     0

     0     2     1

     0     0     2
```

Dari hasil di atas diketahui bahwa matriks Xa adalah rantai Jordan dari A dan Ja adalah matriks Jordan dari A sehingga memenuhi $A = XaJaXa^{-1}$.

Langkah 5 : Karena akan dihitung akar ke-3 maka m=3.

Langkah 6 : » `matriks_kemungkinan=kemungkinan(3,2)`

```
matriks_kemungkinan =

    1     1
    1     2
    1     3
    2     1
    2     2
    2     3
    3     1
    3     2
    3     3
```

Dari hasil di atas untuk membuat matriks kemungkinan yang diinputkan adalah m=3 dan banyaknya nilai eigen yang berbeda Q=2.

Langkah 7 : » `x=matriks_kemungkinan(4,:)`

```
x =

    2     1
```

Jika dipilih baris-4 maka hasilnya seperti di atas.

Langkah 8 : » `new=neweig(A,3,x)`

```
new =

  -0.7211 + 1.2490i   1.2599
```

Dari hasil di atas diketahui bahwa elemen-1 dari `new` adalah nilai eigen baru yang bersesuaian dengan nilai eigen-1 matriks A, demikian juga dengan elemen-2.

Langkah 9 : » `T=matriksT(A,3,x)`

```
T =

    1.0000         0         0

         0    1.0000         0

         0         0    4.7622
```

Langkah 10 : » `Ya=rjord_B(A,3,x)`

```
Ya =
   -1.0000   -1.0000    4.7622
         0   -1.0000         0
    1.0000    2.0000         0
```

Langkah 11 : » `Jb=mjord_B(A,3,x)`

```
Jb =
    -0.7211 + 1.2490i         0         0
                    0    1.2599    1.0000
                    0         0    1.2599
```

Langkah 12 : Menghitung akar ke-3 dari A.

```
» akar(a,3)

matriks_kemungkinan =

     1     1

     1     2

     1     3

     2     1

     2     2

     2     3

     3     1
```

```
      3       2

      3       3

Pilih satu baris dari matriks di atas !!! Baris ke- 4

B =

   1.0499         3.7521-2.4980i   1.7711-1.2490i

  -0.2100                 1.0499          -0.2100

   0.4200        -3.5421+2.4980i  -0.3012+1.2490i
```

Jika baris yang dipilih tidak ada pada matriks kemungkinan, maka akan keluar peringatan "`MAAF PILIHAN ANDA TIDAK BENAR ??? TOLONG ULANGI PROGRAM INI DAN PERHATIKAN PERINTAHNYA`".

Langkah 13 : Akhir program.

Akhir dari program ini akan muncul :

`Apa anda ingin coba cari akar yang lain (Y/T)? Y` **(input)**

Jika ingin mencari akar ke-3 yang lain maka tekan Y sehingga akan muncul :

```
matriks_kemungkinan =
      1       1
      1       2
      1       3
      2       1
      2       2
      2       3
      3       1
      3       2
      3       3
```

```
Pilih satu baris dari matriks di atas !!! Baris ke- 2
```
(input)
```
B =
  -0.5250+0.9093i -4.0394+2.0004i  -1.9672+0.9093i
   0.1050-0.1819i -0.5250+0.9093i   0.1050-0.1819i
  -0.2100+0.3637i  3.9344-1.8185i   1.2323+0.3637i
Apa anda ingin coba cari akar yang lain (Y/T)? T
```
(input)

Jika tidak ingin mencari akar ke-3 yang lain maka tekan T dan akan muncul :

```
Terima kasih telah menggunakan paket program ini !!!
```

Dari penerapan program di atas diketahui bahwa dengan pemilihan baris dari matriks kemungkinan didapatkan akar ke-3 dari matriks A yang berbeda.

Semua langkah pada penerapan program ini tidak perlu dijalankan dalam program. Ada cara yang lebih cepat dalam menghitung akar ke-3 dari matriks A yaitu dengan mengerjakan langkah 1, langkah 2 dan langkah 12.