MATEMATIKA KOMPUTASI
BERBASIS PEMROGRAMAN MATLAB

6 alasan mengapa buku ini penting untuk Anda baca...

1. Dalam buku ini menggunakan bahasa yang mudah dipahami dan setiap pembahasan mencoba untuk menyajikan hubungan antara materi antar sub Bab dan materi antar BAB sehingga penyajiannnya berkesinambungan.
2. Struktur isi mengacu pada kurikulum berbasis KKNI yang diterjemahkan pada capaian pembelajaran mata kuliah dan kemampuan dasar yang dicapai pada setiap tahapan.
3. Setiap script program disertai simulasi dan penjelasan.
4. Buku ini juga menyertakan contoh-contoh soal yang representatif setiap sub-sub Babnya.
5. Listing program yang digunakan dapat diakses melalui perangkat penyimpanan CD yang menjadi bawaan dari buku ini.
6. Memberikan soal latihan yang memacu penguna dalam hal ini mahasiswa untuk menumbuhkan kreatifitas, sikap kritis, dan kolaboratif.

Kaaffah Learning Center

CV. KAAFFAH LEARNING CENTER
Kompleks Griya Bumi Harapan Permai B44
Jalan. Syamsul Alam Bulu, Parepare, Sulawesi Selatan
Telp/Fax. 0421-2914373
E-mail. kaaffahlearningcenter@gmail.com

ISBN 978-623-7426-44-8

9 786237 426448

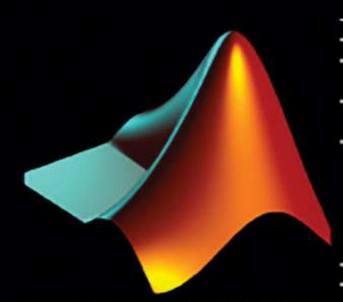
MATEMATIKA KOMPUTASI
BERBASIS PEMROGRAMAN MATLAB

Zulfiqar Busrah, M.Si

Kaaffah Learning Center

BUKU AJAR

MATEMATIKA KOMPUTASI
Berbasis Pemrograman MATLAB

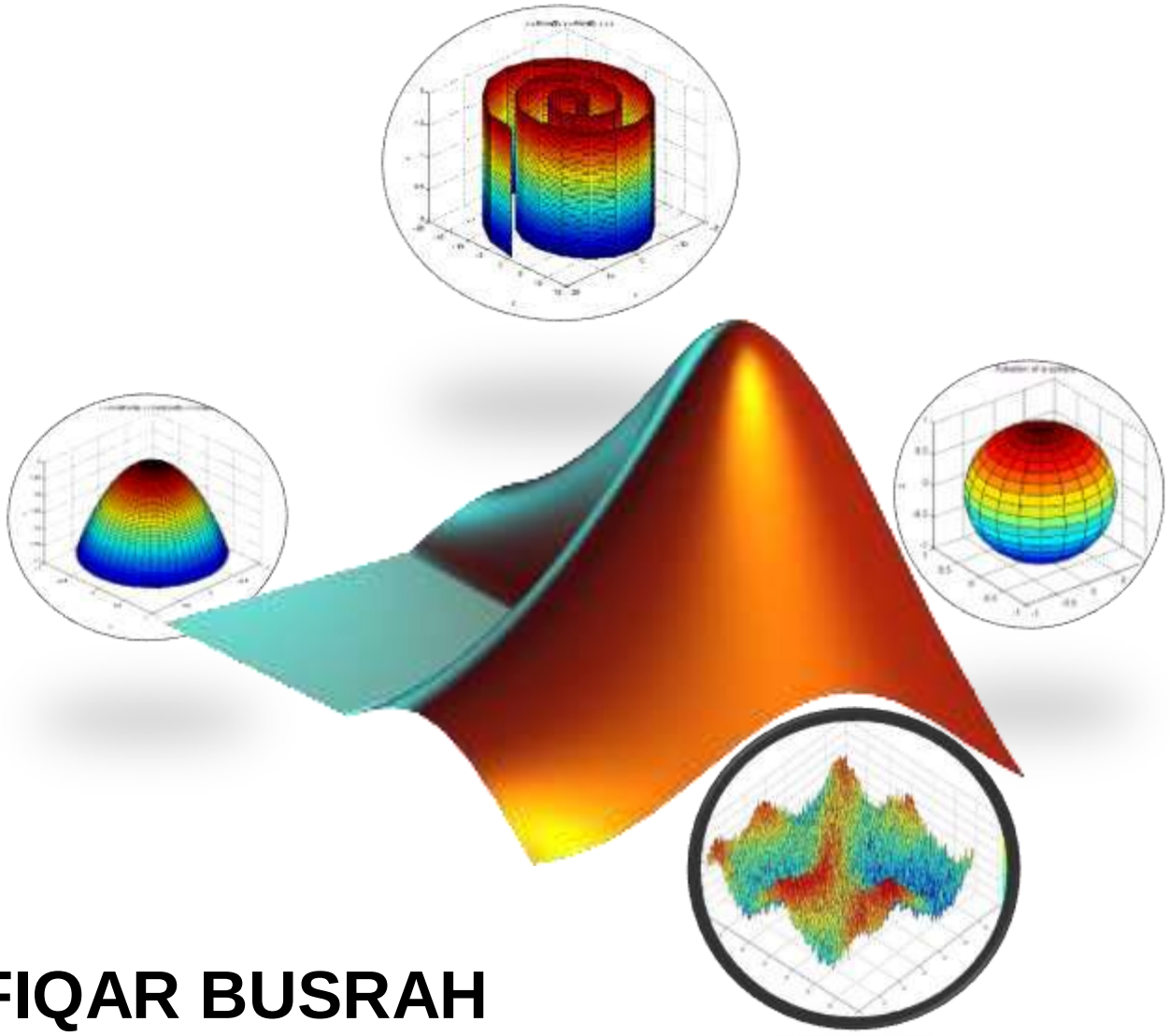


Zulfiqar Busrah, M.Si

Buku Ajar Matematika Komputasi

Berbasis Pemrograman MATLAB

2019



ZULFIQAR BUSRAH

MathWorks®

BUKU AJAR MATEMATIKA KOMPUTASI
Berbasis Pemrograman MATLAB

PENULIS: ZULFIQAR BUSRAH, M.SI.

Editor: Gusniwati

Desain Cover : Gusniwati

Layout : Andi Aras

Cetakan: Pertama, November 2019

Diterbitkan Oleh
Percetakan KAAFFAH

Cet. 1— Parepare**, 2019**
ix, 230 hlm; 21cm x 29,7cm
ISBN 978-623-7426-44-8

BUKU AJAR MATEMATIKA KOMPUTASI
BERBASIS PEMROGRAMAN MATLAB

PROGRAM STUDI TADRIS MATEMATIKA
FAKULTAS TARBIYAH
INSTITUT AGAMA ISLAM NEGERI PAREPARE

Kelebihan dari buku ini :

1. Dalam buku ini menggunakan bahasa yang mudah dipahami dan setiap pembahasan mencoba untuk menyajikan hubungan antara materi antar sub Bab dan materi antar BAB sehingga penyajiannnya berkesinambungan.
2. Struktur isi mengacu pada kurikulum berbasis KKNI yang diterjemahkan pada capaian pembelajaran mata kuliah dan kemampuan dasar yang dicapai pada setiap tahapan.
3. Setiap script Program disertai simulasi dan penjelasan.
4. Buku ini juga menyertakan contoh-contoh soal yang representatif setiap sub-sub Babnya.
5. Listing Program yang digunakan dapat diakses melalui perangkat penyimpanan CD yang menjadi bawaan dari buku ini.
6. Memberikan soal latihan yang memacu penguna dalam hal ini mahasiswa untuk menumbuhkan kreatifitas, sikap kritis, dan kolaboratif.

KATA PENGANTAR

Assalamualaikum Warahmatullahi Wabarakatu

Sesungguhnya segala pujian hanyalah milik Allah SWT., Tuhan Semesta Alam. Tidak ada *ilah* selain Allah semata, dan tidak ada sekutu bagi-Nya. Atas berkat, rahmat serta Hidayah-Nya telah memberi petunjuk bagi penulis dalam menyelesaikan penyusunan **Buku Ajar Matematika Komputasi Berbasis Pemrograman MATLAB** dengan lancar dan dalam keadaan sehat walafiat. Siapa yang Dia beri petunjuk, tidak ada yang dapat menyesatkannnya, dan siapa yang Dia sesatkan tidak ada yang dapat memberinya petunjuk. Dan aku bersaksi bahwa Muhammad adalah hamba dan Rasul-Nya, dengan ini Sholawat dan Salam semoga senantiasa tercurahkan kepada Nabi Muhammad SAW., para Sahabat, serta para pengikutnya.

Buku Ajar Matematika Komputasi Berbasis Pemrograman MATLAB ini dirancang sebagai buku panduan dalam kegiatan perkuliahan Matematika Komputasi dan metode numerik. Dalam penyusunannya melibatkan mahasiswa yang berkontribusi memberikan materi tambahan dalam hasil praktikum. Dalam hal ini, diikuti oleh Aminah, Nurhayati dan Rifka Usman. Kehadiran buku ini diharapkan mampu memberikan stimulus bagi mahasiswa di bidang ilmu matematika dalam meningkatkan keterampilan teknisnya dalam komputasi Matematka. Lebih khusus lagi diharapkan bahwa dengan dimilikinya buku ini dapat membuka jalan mahasiswa untuk mengenal kemampuan coding MATLAB dan membangun daya tariknya dalam berbagai Bahasa Pemrograman dalam rangka peningkatan daya saing di era industri 4.0.

Penyusunan buku ajar ini dapat terselesaikan atas bantuan dari berbagai pihak, dengan ini penulis mengucapkan Terima Kasih yang setinggi-tingginya saya sampaikan kepada seluruh pihak yang telah telibat. Terkhusus kepada pihak para pejabat dalam hal ini Rektor IAIN Parepare, Dekan Fakultas Tarbiyah, Ketua Program Studi, dan LP2M IAIN Parepare yang telah mendukung secara moril dan materil. Tak lupa pula segenap

rekan-rekan dosen Tadris Matematika dan seluruh rekan-rekan mahasiswa yang terlibat atas kerjasama dan kontribusinya. Semoga Allah SWT. membalas kebaikan saudara semua. Amin

Penulis sangat menyadari bahwa penyusunan Buku Ajar ini masih sangat jauh dari kata sempurna, seBab kesempurnaan itu sejatinya hanya milik Allah SWT., namun dengan tangan terbuka dan hati yang ikhlas, penulis sangat mengharapkan masukan bahkan kritikan yang produktif untuk tujuan dengan segala kebaikan bersama secara umum, dan untuk pembenahan isi dari Buku Ajar ini secara khusus.

Sebagai Penutup, penulis mengucapkan salam persaudaraan sesama hamba-Nya, semoga kita semua selalu dalam petunjuk-Nya menjadi sebaik baiknya pencinta ilmu dan kebijaksanaan semata mata untuk membuka jalan ibadah dan kebaikan bagi kita semua.

Wassalamu Alaikum Warahmatullahi Wabarakatu.

Parepare, November 2019

Penulis

DAFTAR ISI

ABSTRAK

Buku ini merupakan buku ajar yang dikembangkan untuk mengakomodir Mata Kuliah Matematika Komputasi dan Metode Numerik. Kumpulan Isi dari buku ini diarahkan dalam satu kesatuan yang mengkombinasikan beberapa mata kuliah seperti Kalkulus, Aljabar Linear, Persamaan Diferensial dan Metode Numerik yang dikaji dengan menggunakan pendekatan komputasi Matematika. Pemilihan Pemrograman MATLAB sendiri didasarkan pada cakupan pemecahan masalah yang dapat merepresentasikan berbagai topik bahasan seperti topik dasar dalam matematika seperti kajian Vektor, Matriks, Fungsi dan Penyajian fungsi melalui grafik. Untuk memudahkan penggunaan buku ini, maka dalam pembahasan di setiap BAB nya diawali penyajian kemampuan dasar yang menjadi target capaian. Dalam penyajian isi, lebih banyak memberikan contoh-contoh script program yang dilengkapi dengan penjelasan dan interpretasi hasil. Pada setiap BAB nya juga dilengkapi dengan soal-soal latihan yang memungkinkan pengguna untuk mengeksplorasi kemampuannya. Diharapkan dengan adanya buku ini, dapat menjadikan mata kuliah yang terkait dapat lebih terarah, sistematis, efisien dan komprehenif, baik untuk peningkatan wawasan dan keterampilan komputasi matematika secara individu bagi mahasiswa maupun untuk peningkatan kualitas lembaga dalam penyediaan media pembelajaran.

BAB 1. PENDAHULUAN

Kemampuan akhir yang diharapkan

- Mahasiswa dapat menguraikan hubungan Matematika Komputasi terhadap Mata kuliah lainnya.
- Mahasiswa dapat melakukan Instalasi Software pada umumnya dan MATLAB pada khususnya sebagai Fasilitas Komputasi di bidang Matematika.
- Mahasiswa dapat mengenal bagian-bagian interface MATLAB dan mengetahui tujuannya masing-masing.
- Mahasiswa mengenal karakteristik variabel dan mampu menggunakannya dalam menyelesaikan masalah matematika.
- Mahasiswa dapat membedakan jenis-jenis operator dan penggunaannya.
- Mahasiswa mampu mengeksplorasi fasilitas help yang terdapat pada MATLAB

A. Matematika Komputasi

Kompleksitas dalam pembelajaran matematika tidak terlepas dari adanya beragam cabang ilmu di dalamnya yang selalu dipandang secara terpisah. Dengan kompleksitas ini, cabang-cabang dari ilmu matematika perlu untuk dipetakan guna untuk melihat hubungan dan ruang lngkup masing-masing cabang ilmu. Tidak hanya melihat hubungannya sebagai struktur keilmuan, namun perlu juga dibutuhkan pemetaannya sebagai hasil dari pengalaman belajar. Klasifikasi dan pemetaan ini diperlukan agar proses belajar dapat berlangsung secara berkesinambungan. Dengan demikian arah pembelajaran menjadi jelas, terstruktur, sistematis dan komprehensif. Demikian halnya dengan Matematika Komputasi, maka penting untuk dilakukan pemetaan secara jelas hubungan antara struktur atau isi dari topik matematika lainnya serta pengalaman belajar oleh mahasiswa itu sendiri.

Matematika dalam perkembangannya diklasifikasikan ke dalam dua kelompok besar yaitu matematika murni dan matematika sebagai ilmu terapan. Pada kelompok matematika murni, mengandung berapa topik atau kajian khusus seperti sistem bilangan, strukutur, kajian ruang, dan kajian perubahan. Sedangkan pada kelompok matematika terapan dipandang sebagai kombinasi matematika dengan keilmuan diluar matematika seperti, matematika fisika, matematika kimia, matematika keuangan, matematika biologi, ilmu komputer, kriptografi, ilmu rekayasa (*enginering*), ilmu peluang, dan *machine learning*.

Matematika komputasi sendiri merupakan mata kuliah lanjutan sebagai respon terhadap perkembangan teknologi, informasi dan komunikasi baik untuk pemecahan masalah dalam kehidupan sehari-mari maupun sebagai pengembangan dari ilmu matematika. Ruang lingkup pembahasan dalam matematika komputasi merupakan pendekatan untuk memahami konsep-konsep dasar dalam matematika. Selain itu, cakupan dari matematika komptasi yakni mengintegrasikan ilmu-ilmu dasar seperti kalkulus, aljabar linear, dan geometri secara komputasional. Oleh karena itu, pembahasan dalam buku ini diawali dengan menguraikan hubungan posisi Matematika Komputasi terhdadap mata kuliah lainnya seperti Kalkulus, Aljabar dan Geometri baik sebagai ilmu dasar matematika mapun sebagai hasil dari pengalaman belajar.

Adanya kecendrungan bagi mahasiswa melihat mata kuliah yang ada sebagai subjek subjek yang saling terpisah. Pemahaman mengenai keterpaduan dan kesinambungan materi amatlah penting untuk dijadikan sebagai dasar dalam memulai satu kajian baru.

❖ Matematika Komputasi dan Aljabar

Aljabar merupakan salah satu kajian khusus dan mendasar dalam matematika murni yang tergabung dalam kelompok kajian struktural. Kaitan antara aljabar dengan matematika komputasi dapat dilihat bahwa fasilitas

komputasi matematika dikembangkan atas teori-teori dasar dalam aljabar. Seperti halnya terhadap mata kuliah aljabar linear sebagai mata kuliah dasar yang membekali mahasiswa tentang konsep dasar dan hukum-hukum pada matematika mengenai persamaan, pertidaksamaan, vektor dan matriks.

Sebagai cabang ilmu dari matematika, aljabar menjadi dasar dalam kajian penggunaan simbol-simbol matematis dan seperangkat aturan yang dimanipulasi untuk melihat hubungan antar simbol. Seperti halnya, simbol yang mewakili operator dasar aritmatika $(+,-,\times,/,\div)$, dan simbol operator relasional dasar $(=,\neq,\leq,\geq)$ dalam menghubungkan sifat antar simbol lain misal penggunaan huruf untuk mewakili angka, memungkinkan adanya keumuman sifat tanpa memperhatikan angka-angka yang digunakan. Hal ini, umum dikenal sebagai persamaan atau pertidaksamaan. Hal-hal tersebut dieksplorasi lebih mendalam pada Matematika Komputasi.

Kajian aljabar dalam matematika komputasi dapat ditemukan dalam memandang persamaan sebagai bentuk lain dari grafik. Diantara beberapa grafik yang disajikan merupakan perpaduan dari titik-titik baik yang kontinu maupun diskrit. Dengan ini kita dapat melihat kajian matematika komputasi lebih komprehensif yaitu dengan mengintegrasikan konsep konsep aljabar. Pada perangkat lunak yang dipilih pada buku ini yaitu *MATLAB* yang tidak lain merupakan singkatan dari Matriks Laboratory, hal ini menunjukkan bagaimana peran aljabar terhadap kajian matematika komputasi. Topik aljabar dalam buku ini yang dikaji adalah Matriks, Vektor dan juga pada BAB 7 terdapat pembahsan Sistem Persamaan Linear.

❖ Matematika Komputasi dan Kalkulus

Selain Aljabar, kalkulus juga dikenal sebagai salah satu kajian dari matematika murni yang tergabung dalam kelompok kajian perubahan. Dalam konteks komputasi matematika, kalkulus telah membekali pemahaman konsep **sistem bilangan** dan **sistem koordinat**, penyelesaian **persamaan** dan **pertidaksamaan** serta pemahaman tentang

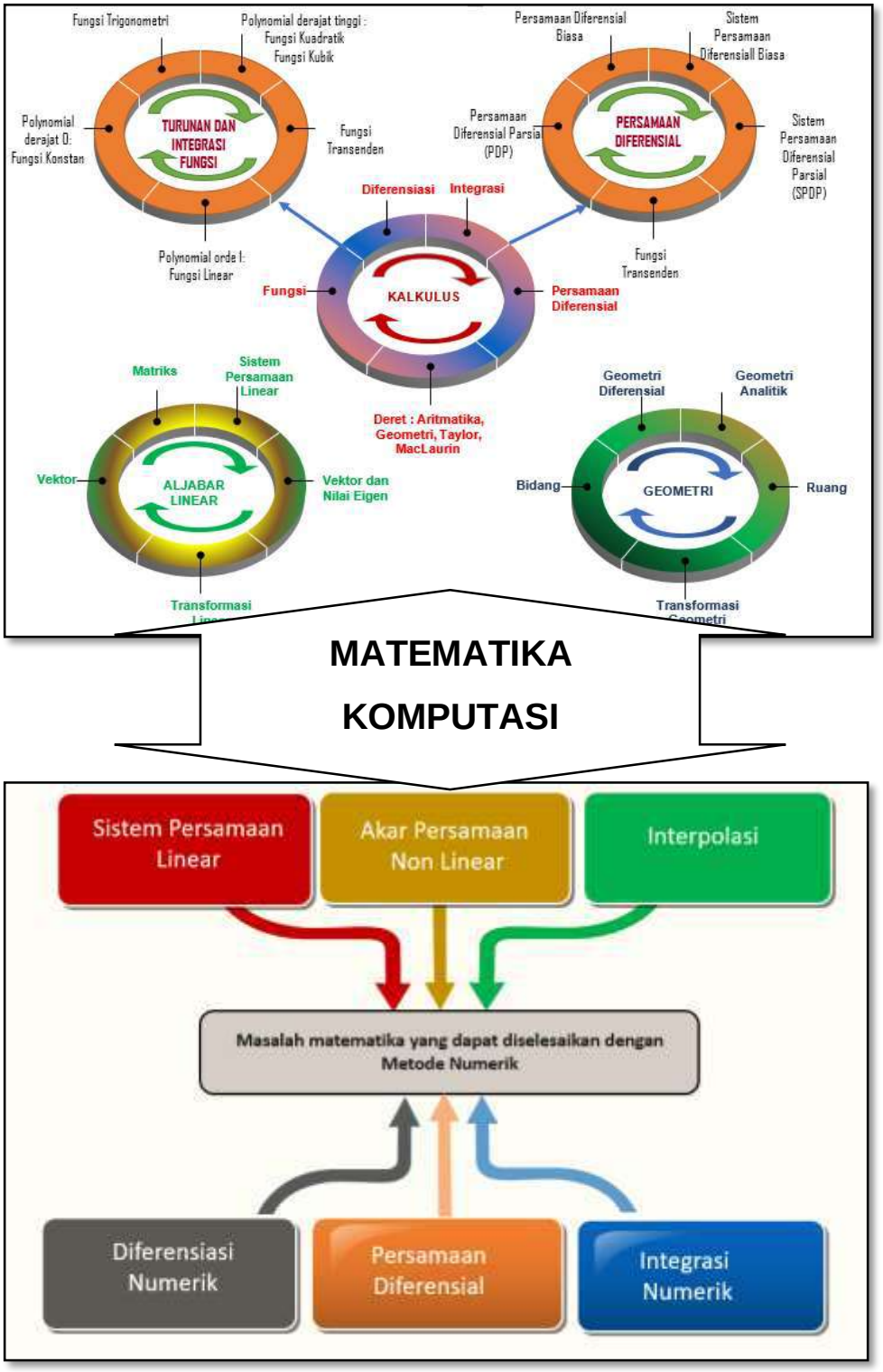
konsep **relasi** dan **fungsi**. Berikut penguraian hubungan kalkulus dalam pembelajaran matematika komputasi.

- **Keterpaduan sistem bilangan real dan sistem koordinat baik kartesius maupun polar** menjadi wadah dalam mengilustrasikan dan menvisualisasikan konsep konsep geometris seperti konsep titik, garis dan bidang, khususnya dalam ruang berdimensi dua. Melalui buku ajar matematika komputasi kita dapat melihat bagaimana keterpaduan sistem-sistem di atas cecara sistematis, efisien dan efektif tentunya dengan menggunakan perangkat digital.
- K**alkulus juga telah membekali kita tentang konsep relasi dan fungsi**.

  Relasi yang dinyatakan sebagai suatu aturan yang memetakan anggota-anggota dari himpunan tertentu yang dikenal dengan himpunan daerah asal ke dalam anggota-anggota himpunan lain yang dikenal dengan daerah hasil. Sedangkan fungsi, secara spesifik sebagai bagian relasi yang memetakan setiap anggota daerah asal tepat satu ke anggota himpunan daerah hasil. Relasi dan fungsi menjadi dasar penyajian suatu persamaan ke dalam grafik pada sistem koordinat, ataupun sebaliknya dengan parameter parameter tertentu.

  Melalui matematika komputasi berbasis pemrograman MATLAB, kita kemudian akan mengkaji fungsi lebih komprehensif dan mendalam.
- **Permasalahan gradien, diferensial dan integral** adalah topik-topik khusus dalam kalkulus yang mengkaji hukum baik berupa definisi, aksioma, teorema dan sebagainya. Melalui hukum-hukum inilah dikembangkan berbagai fasilitas komputasi, yang dapat menyelesaiakan masalah matematika secara praktis, efektif dan efisien. Namun demikian, masalah diferensiasi dan integrasi tidak selamanya dapat diselesaikan secara analitik melalui penerapan hukum-hukum yang ada, sehingga butuh pendekatan khusus dalam menyelesaikannya. Dalam matematika komputasi, dikenal satu topik

khusus yaitu Komputasi Numerik yang mencoba membangun pendekatan terhadaap masalah yang dimaksudkan.



Gambar 1. 1. Peta konsep hubungan matematika komputasi dengan mata kuliah lainnya

Pada BAB 7 kita akan melihat bagaimana penerapan algoritma-algoritma secara khusus memberi solusi pendekatan terhadap masalah diferensial dan integral.

B. Pengantar MATLAB

MATLAB (Matrix Laboratory) adalah sebuah lingkungan komputasi numerikal dan bahasa pemrograman komputer generasi keempat yang banyak digunakan untuk menyelesaikan berbagai masalah dengan melibatkan proses komputasi dalam berbagai bidang. Sebagai bahasa pemrograman lanjutan, MATLAB menggunakan Matriks sebagai dasar pemikiran dalam analisis dan komputasi. MATLAB memungkinkan manipulasi matriks, pem-plot-an fungsi dan data, implementasi algoritma, pembuatan antarmuka pengguna, dan peng-antarmuka-an dengan program dalam bahasa lainnya. Meskipun hanya bernuansa numerik, sebuah kotak kakas (toolbox) yang menggunakan mesin simbolik MuPAD. Perangkat lunak ini menawarkan kemudahan, kesederhanaan dan kejelasan dalam menyelesaakn permasalahan yang dapat direpresentasikan dengan vektor dan matriks.

Dalam sejarah awal dan perkembangannya, MATLAB berupa suatu *interface* untuk koleksi rutin-rutin numerik dari proyek LINPACK dan EISPACK, dan dikembangkan menggunkan bahasa FORTRAN. Secara komersial, MATLAB merupakan produk perusahaan Mathworks, Inc.yang dalam perkembangan selanjutnya dikembangkan menggunakan bahasa C++ dan assembler (utamanya untuk fungsi-fungsi dasar MATLAB).

MATLAB telah berkembang menjadi sebuah *environment* pemrograman yang canggih yang berisi fungsi-fungsi *built-in* untuk melakukan tugas pengolahan sinyal, aljabar linier, dan kalkulasi matematis lainnya. MATLAB juga berisi *toolbox* yang berisi fungsi-fungsi tambahan untuk aplikasi khusus . MATLAB bersifat *extensible*, dalam arti bahwa seorang pengguna dapat menulis fungsi baru untuk ditambahkan pada *library* ketika fungsi-fungsi *built-in* yang tersedia tidak dapat melakukan

tugas tertentu. Kemampuan pemrograman yang dibutuhkan tidak terlalu sulit bila Anda telah memiliki pengalaman dalam pemrograman bahasa lain seperti C, PASCAL, atau FORTRAN.

Sebagai merk software yang dikembangkan oleh Mathworks.Inc.(lihat http://www.mathworks.com), MATLAB dikenal paling efisien untuk perhitungan numerik berbasis matriks. Dengan demikian jika di dalam perhitungan kita dapat menformulasikan masalah ke dalam format matriks maka MATLAB merupakan software terbaik untuk penyelesaian numeriknya.

MATLAB yang merupakan bahasa pemrograman tingkat tinggi berbasis pada matriks sering digunakan untuk teknik komputasi numerik, yang digunakan untuk menyelesaikan masalah-masalah yang melibatkan operasi matematika elemen, matrik, optimasi, aproksimasi dll. Sehingga MATLAB banyak digunakan pada :

- ❖ Matematika dan Komputansi
- ❖ Pengembangan dan Algoritma
- ❖ Pemrograman modeling, simulasi, dan pembuatan prototipe
- ❖ Analisa Data , eksplorasi dan visualisasi
- ❖ Analisis numerik dan statistik
- ❖ Pengembangan aplikasi teknik

C. Instalasi MATLAB

Dalam melakukan instalasi MATLAB, tidak jauh berbeda dengan instalasi aplikasi-aplikasi pemrograman lainnya. Dalam hal ini dibutuhkan driver atau master MATLAB yang akan dinstal. Berikut tahapan tahapan umumnya :

Langkah 1: Pada penginstalan MATLAB kali ini menggunakan master MATLAB versi R2017aa yang berada pada directory dengan kapasitas 9.8 GB. Pastikan master MATLAB yang akan diinstal tersedia di directory.

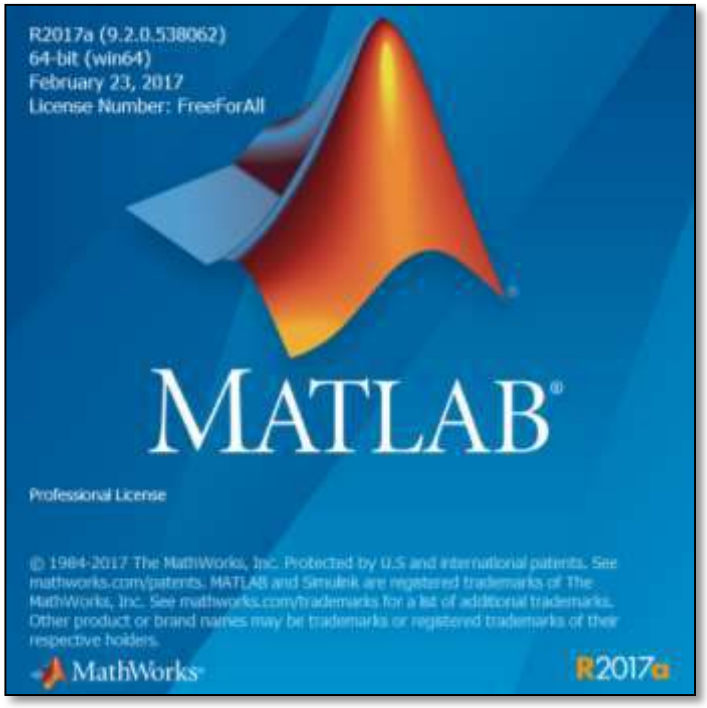
Langkah 2 : Buka Master MATLAB R2017aa , kemudian pilih file setup bertipe aplication, maka akan muncul jendela MathWorks Installer. Langkah 3 : Pada jendela MathhWorks Installer :
❖ Pilih metode instalasi dengan mode use a file instalation key yang berarti mode penginstalan tidak menggunakan internet namun menggunakan kode kunci yang tersedia. ❖ Selanjutnya klik next, dan akan muncul Jendelea *License Agreement*
Langkah 4 : Pada kotak dialog License Agreement memberikan pilihan atau persetujuan untuk menggunakan Nomor License. Klik next untuk masuk ke tahap berikutnya yaitu Jendela Folde Selections Langkah 6: Pada jendela file instalation Key, masukkan Key for My License yang disediakan :
❖ Buka Folder Crack Master MATLAB yang disediakan. ❖ Buka file notepad ❖ Selanjutnya copy serial number sebagai Key for My License yang tersedia. ❖ Klik Next untuk masuk pada jendela Product Selection.
Langkah 7 : Pada jendela Product Selections :
❖ Pastikan semua products dalam keadaan tercentang. ❖ Selanjutnya klik next untuk masuk padajendela confirmations. ❖ Pada Jendela confirmations Klik Install
Selanjutnya proses Installing MATLAB berjalan hingga 100 %, , tunggu beberapa menit sampai complete Langkah 8:
❖ Selanjutnya pada jendela Product Congfiguration Notes, Klik Next. Hingga muncul jendela Installation is Complete. ❖ Klik Finish
Langkah 9: Setelah instalasi pada tahap complete, maka langkah selanjutnya adalah masuk pada tahap activasi MATLAB.

Masuk pada
❖ Folder Program Files>> MATLAB>> RR2017aa>> Bin>> Win64 >> ❖ Klik 2x file activate MATLAB ❖ Selanjutnya akan muncul jendela aktivasi,
Langkah 10: Pada jendela ofline Activation menunjukkan mode aktivasi yang tidak membutuhkan koneksi internet, pada bagian ini :
❖ Pilih Enter the full path to your license file, including the file name. Pada bagian ini, user diminta untuk memasukkan licenseMATLAB yang tersedia dalam master MATLAB. ❖ Klik Browse untuk mencari full path. ❖ Klik next, maka MATLAB telah berhasil di activasi.
Langkah 11: Setelah MATLAB diaktivasi, maka langkah selanjutnya adalah copy file Imgripl pada crack Master MATLAB ke dalam: programFiles \ MATLAB\ RR2017aa\ bin\win64\ MATLAB_startup_pluggins \ Imgrimpl Langkah 12: Setelah file Imgriplnya dicopy, maka selanjutnya file license_RR2017aa yang juga berada dalam folder crack dicpy pada ProgramFiles\ MATLAB\ RR2017aa\ licenses.

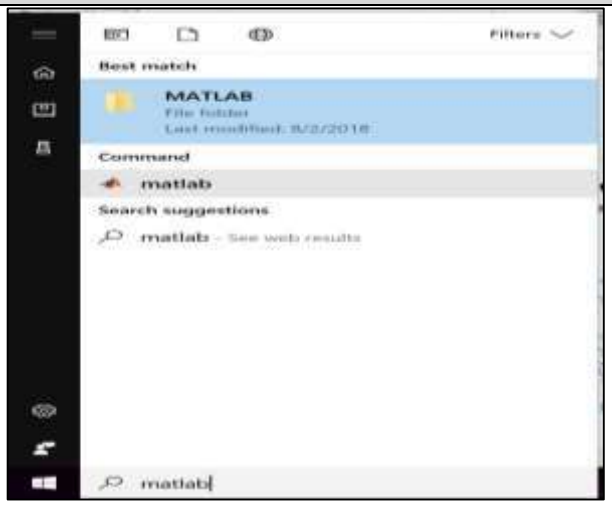
Tabel 1. Langkah-langkah intalasi MATLAB pada Windows

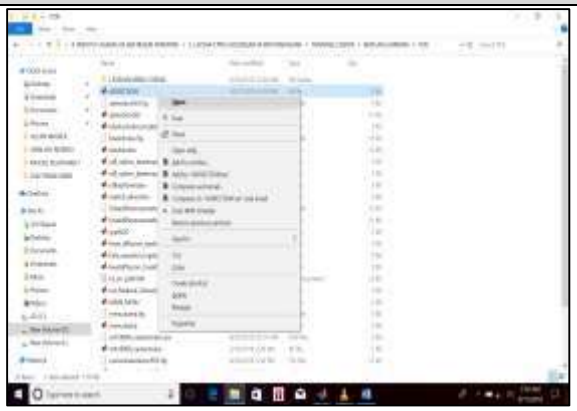
❖ **Starting MATLAB**

Setelah melakukan instalasi MATLAB pada windows, maka hal berikutnya yang perlu untuk dipahami adalah bagaimana memulai membuka MATLAB. Dalam memulai menjalankan MATLAB, dapat dilakukan dengan berbagai cara diantaranya :



Gambar 1. 2. Tampilan Starting Up MATLAB

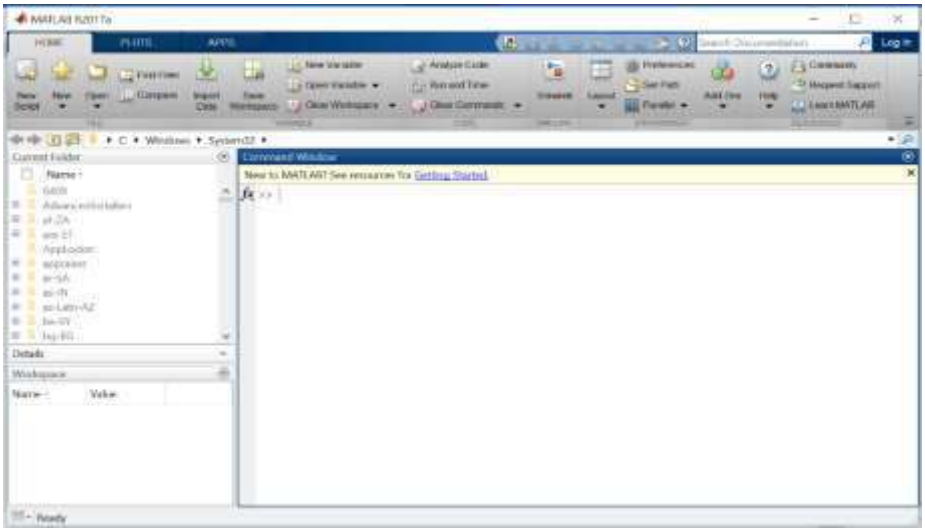
1. Menjalankan MATLAB melalui tombol Start	
a. Klik Start b. Ketikkan MATLAB pada kolom pencarian c. Muncul icon MATLAB, klik 2x icon MATLAB yang muncul d. Jendela MATLAB akan terbuka	 Gambar 1. 3. Tampilan Pencarian Aplikasi Pada Type Here to Search
2. Melalui Desktop	

a. Pastikan icon MATLABnya berada pada desktop b. Klik 2x icon MATLAB c. Jendela MATLAB segera terbuka	 Gambar 1. 4. Icon MATLAB pada Desktop
3. Melalui file MATLAB yang erada dalam directory	
a. Cari m.file yang berada dalam directory b. Klik 2x file MATLAB yang akan dibuka c. Jendela MATLAB akan segera terbuka dengan segenap perintah yag telah dibuat pada file yang dibuka	 Gambar 1. 5. M-File pada Directori

Kegiatan 1. Jalankan MATLAB pada komputernya masing-masing !

D. Pengenalan Interface MATLAB

Ketika memulai menjalankan MATLAB melalui icon pada desktop ataupun melalui tombol start, tampilan MATLAB R2017a yang muncul tampak seperti berikut :



Gambar 1. 6. Interface utama MATLAB

❖ **Penjelasan Interface :**

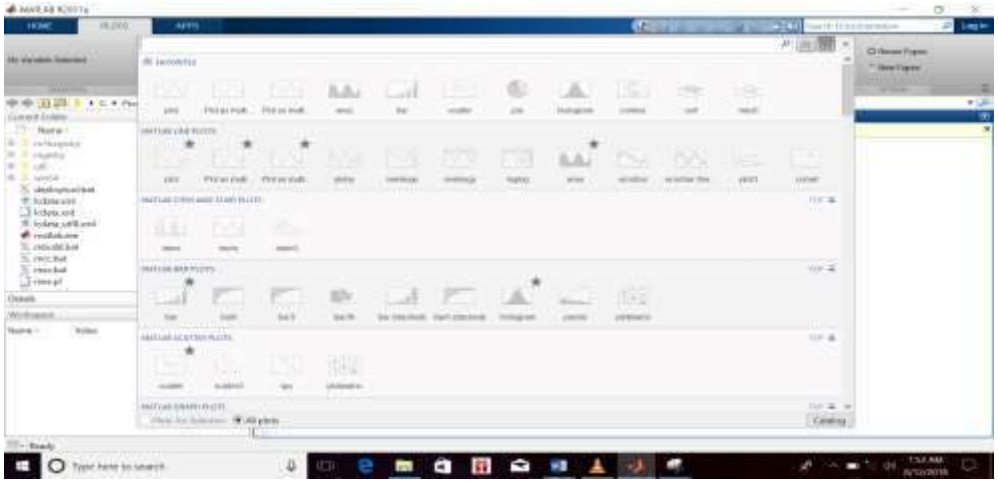
Tampilan antarmuka pada MATLAB, dari versi ke versi berbeda-beda. Pada bagian interface MATLAB R2017a menampilkan 3 bagian utama yaitu Menu utama, Sub Menu dan Ruang Kerja Utama.

Pada bagian atas menyediakan menu-menu utama nampak seperti pada gambar 1.6 yaitu terdapat 3 menu utama yaitu Home, Plots dan Apps. Pada masing-masing menu menyediakan tombol-tombol sebgai sub menu yaitu :

- Pada menu Home menyediakan tombol-tombol khusus dengan fungsinya masing-masing.

Gambar 1. 7. Tampilan Sub Menu pada Menu Home

Pada menu Plots menyediakan pilihan-pilihan tipe grafik seperti pada gambar berikut:



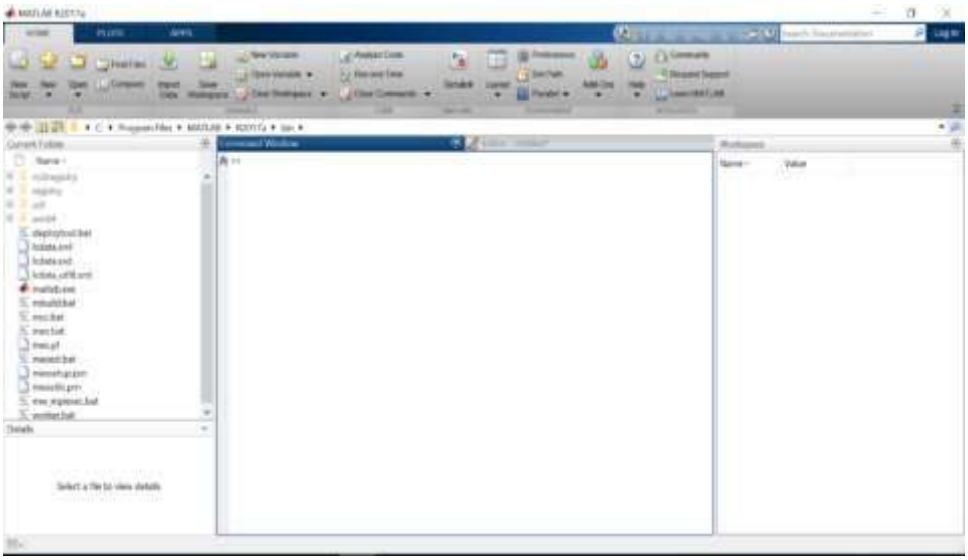
Gambar 1. 8. Tampilan Sub Menu pada Menu Plots

Pada menu Apps : Menyediakan aplikasi-aplikasi khusus

Gambar 1. 9. Tampilan Utama Pada Sub Menu Apps

Kegiatan 2. Kenali menu-menu utama dan masing-masing submenunya

Selanjutnya, penting pula untuk dikenali yaitu pada sisi ruang kerja utama terdiri dari 5 sub window , diantaranya :



Gambar 1. 10. Sub Window sebagai Ruang Kerja Utama

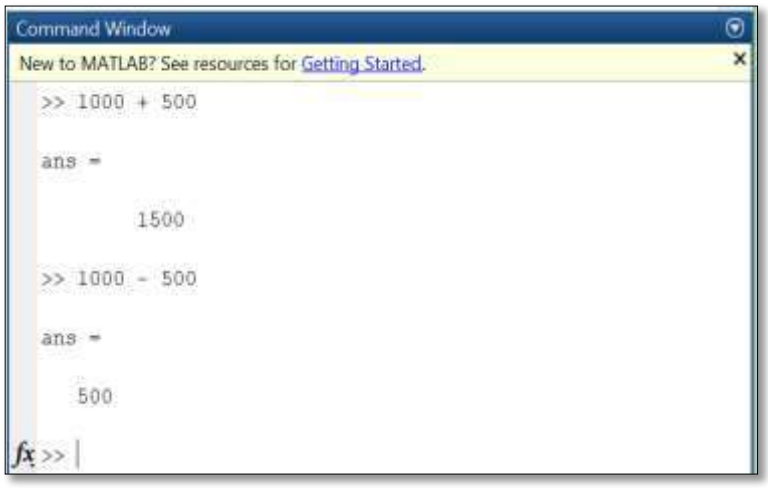
- **Command Window** (Jendela Perintah) adalah jendela utama pada ruang kerja utama yang digunakan untuk menuliskan perintah, menjalankan perintah ataupun melihat hasil-hasil tertentu dalam menjalankan suatu perintah.
- **Current Folder** (Directory Terkini) merupakan sub jendela yang digunakan untuk memanggil, ataupun mengaktifkan serta menampilkan file-file yang sedang bekerja, ataupun yang akan diaktifkan.
- **Command History** (Riwayat Perintah) merupakan sub jendela yang digunakan untuk menampilkan riwayat perintah-perintah yang telah dijalankan oleh pengguna sebelum diberikan perintah clear (hapus).
- **Editor** adalah jendela utama pada ruang kerja utama yang digunakan untuk menuliskan perintah yang panjang dan disimpan sebagai M-file pada directory, kemudian menjalankan perintahberbasis file, bukan berbasis baris .
- **Workspace** (ruang kerja) merupakan sub jendela pada ruang kerja utama yang dapat diatur tampilannya dan digunakan dalam memonitoring variable, data, ataupun tipe data yang sedang aktif ataupun yang dijalankan.

❖ Mengenal Command Window

Pada gambar interface MATLAB di atas, posisi Command Window berada ada ruang kerja utama dan terletak di tengah. Meskipun demikian command window secara tampilan dapat digeser diminimize ataupun maximize. Sebelumnya telah dijelaskan bahwa penggunaan Command Window diperuntukkan menulis perintah, menjalankan perintah secara interktif atapun memanggil file MATLAB untuk dijalankan.

Dengan kemampuan penyelesaian masalah aritmatika, command window biasa digunakan sebagai lembaran Buram, atau cakaran untuk menyelesaikan operasi penjumlahan, pengurangan pembagian, perpangkatan, dan berbagai operasi aritmatika lainnya. Berikut beberapa hal yang dapat dijelaskan mengenai command window dalam penggunaanya.

1. Pada Command Window, terdapat tanda >> (prompt) yang menyatakan bahwa MATLAB siap menerima perintah yang dituliskan oleh user, yang akan dituliskan di depan tanda >>.
2. Jika tanda >> belum muncul berarti MATLAB belum siap untuk menerima perintah.



Gambar 1. 11. Prompt (<<) pada Command Window

Pada gambar 1.11, dengan menuliskan perintah 1000 + 500 di depan tanda >> dan kemudian menekan tombol enter maka akan menghasilkan ans = 1500. Selanjutnya tanda >> akan kembali muncul dan kembali siap

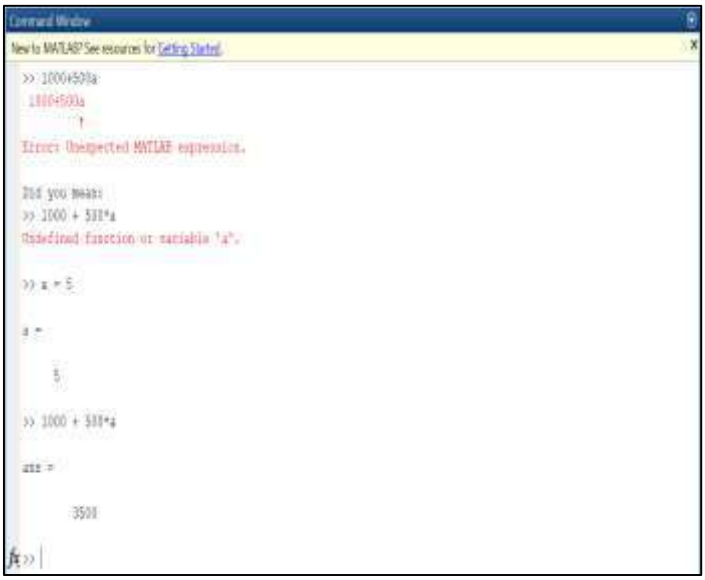
menerima perintah. Misal dilanjutkan perintah baru dengan 1000 – 500 yang dituliskan di depan tanda >> , jika kembali ditekan enter maka akan muncul ans = 500. Tulisan ans berasal dari kata answer yang artinya sebagai jawaban atas perintah yang diberikan

Setiap perintah yang dapat dieksekusi pada command window disebut sebagai statement. Umumnya pernyataan berbentuk : `Variabel = ekspresi atau Ekspresi`

Kegiatan 3. Lakukan latihan pendefinisian beberapa variabel beserta data-datanya.

❖ Kesalahan Perintah pada Commad Window

Pada Command Window, user memungkinkan menemukan masalah kesalahan penulisan perintah. Apabila user memberikan perintah yang tidak dikenal, MATLAB akan menampilkan keterangan kesalahan. Sebagai contoh Pada gambar di bawah ini menunjukkan adanya perintah yang dituliskan yang tidak dikenal oleh MATLAB dengan keterangan Error : Unexpected MATLAB Expression. Kesalahan ini berupa ekpresi dari operator penjumlahan tidak mengenali operand 500a (bukan numerik dan juga bukan variabel), dan ini merupakan kesalahan umum dari penulisan perintah.



Gambar 1. 12. Pembacaan Kesalahan pada Penggunaan Perintah di Commad window

Dan pada MATLAB RR2017a dapat mengidentifikasi kesalahan penulisan perintah dengan meberikan keterangan perbaikan :

```
Did you mean :
>> 1000+ 500*a
```

Namun setelah dienter kembali sebagai perintah baru, muncul kesalahan baru yaitu undefined function or variable ‘a’, kesalahan ini menyatakan bahwa ada variabel ‘a’ yang nilainya belum terdefinisi pada perintah sebelumnya. Berikut beberapa tombol yang digunakan untuk mengedit perintah yang ada pada command window :

	Untuk memperbaiki ekspresi yang salah, gunakan tombol untuk menampilkan perintah sebelumnya. Kemudian lakukan pengeditan dan akhiri dengan menekan tombol enter.
	Untuk menghapus karakter di depan kursor gunakan tombol
	Untuk menghapus karakter di sebelah kiri kursor
	Untuk menggeser kursor ke kanan
	Untuk menggeser kursor ke kanan
	Untuk memperoleh perintah setelah perintah sekarang
	Untuk mengosongkan perintah pada prompt

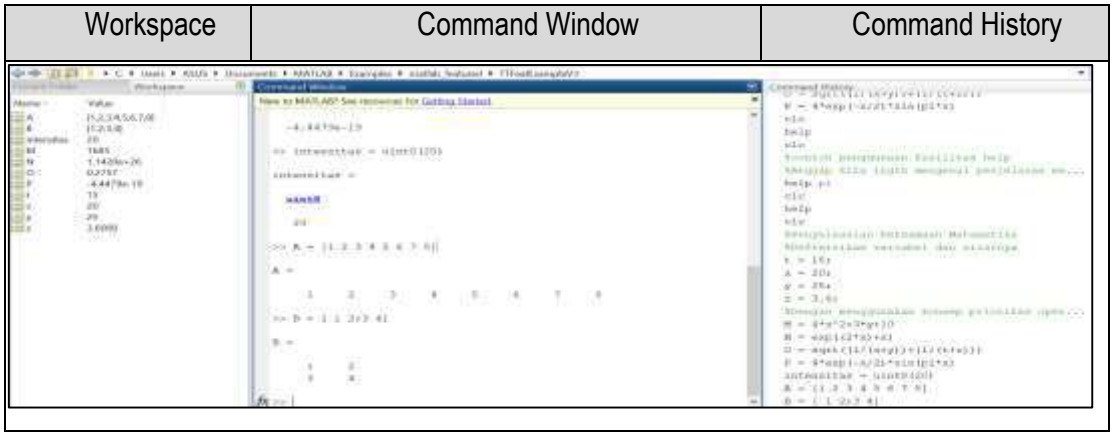
Tabel 2. Penggunaan Tombol Direction

Kegiatan 4. Lakukan latihan editing-editing perintah pada command window dengan menggunakan tombol-tombol di atas

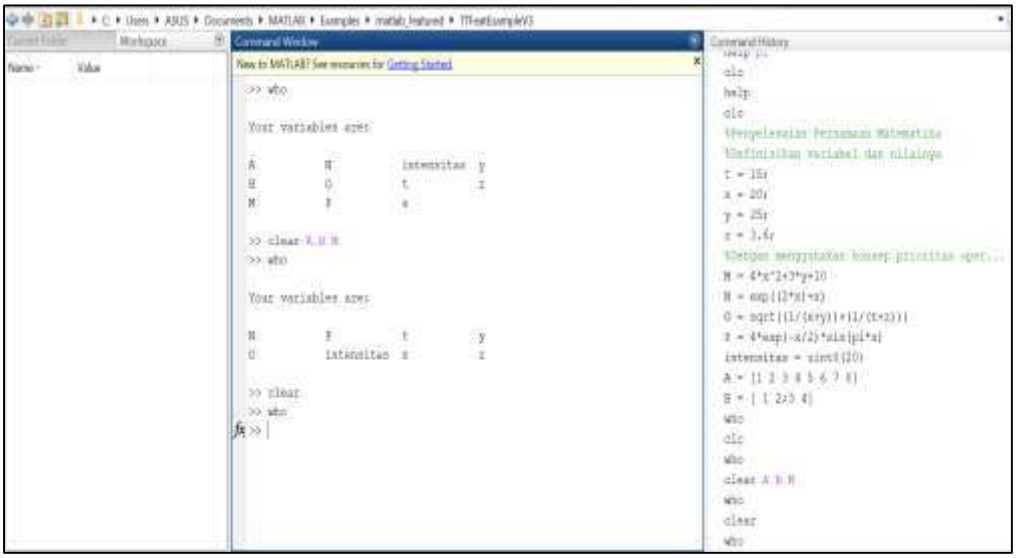
❖ Mengenal Workspace dan Command History

Sebelumnya telah diperkenalkan *Workspace* sebagai sub window pada ruang kerja utama MATLAB. Ketika kita bekerja pada MATLAB terkadang kita membutuhkan monitoring variable, data, ataupun tipe data yang sedang aktif ataupun yang dijalankan. Dari sekian banyak perintah yang dituliskan pada penggunaan Command Window di contoh-contoh sebelumnya , workspace merekam variabel-variabel dan nilai-nilainya yang telah didefinisikan.

Selanjutnya perintah-perintah yang telah dituliskan sebelumnya pda command window dapat terhapus jika diberikan perintah `clc` , namun semua perintah yang kita hapus pada command window dapat kita temukan kembali pada Sub Window Command Histori. Berikut gambar Workspace, Command Window dan Command History yang ditampilkan dengan mode layout sejajar (Three Column).



Gambar 1. 13. Tampilan Perekaman Variabel dan perintah pada Workspace dan Command History



Gambar 1. 14. Perintah clear untuk menghapus rekaman data pada workspace

E. Pengenalan Operator

Catatan :

- ❖ Untuk mengetahui variabel apa saja yang telah didefinisikan pada command window dan terekam oleh workspace tanpa melihat jendela workspace, maka kita bisa menggunakan perintah who.
- ❖ Setiap variabel dan nilainya akan membutuhkan memori komputer.
- ❖ Untuk menghemat penggunaan memori komputer, maka variabel dan nilai-nilai yang terdapat di dalam workspace dapat dihapus. Untuk menghapus semua variabel, kita dapat menggunakan perintah clear.
- ❖ Namum jika variabel workspace yang akan dihapus hanya beberapa variabel saja maka kita dapat menggunakan perintah clear namavariabel1 namavariabel2. Untuk menghapusnya, kita dapat menggunakan perintah.

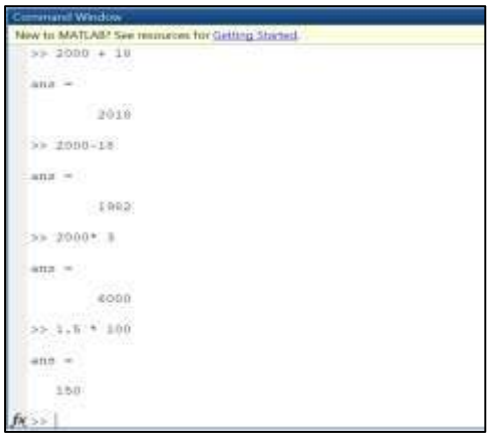
Pada program aplikasi MATLAB, penggunaannya tidak akan terlepas dari penggunaaan operator-operator dasar. Adapun operator yang difokuskan pada buku ini adalah pengenalan operator dasar diantaranya operator aritmatika, operator relasional dan operator logika. Ketiga jenis operator ini digunakan sesuai dengan kebutuhan dan tujuan dari masalah yang akan dipecahkan. Berikut diuraikan jenis-jenis operator dasar pada MATLAB.

❖ Operator Aritmatika

Dalam ilmu matematika, penggunaan operator aritmatika menjadi bagian dari dasar perhitungan. Pada masalah matematika komputasi, cara kerja operator aritmatika sama seperti dengan konsep operator secara teori. Berikut beberapa operator aritmatika

Operator	Keterangan
+	Penjumlahan atau tanda positif
-	Pengurangan atau tanda negatif
*	Perkalian
/ atau \	Pembagian ke depan dan pembagian ke belakang
^	Perpangkatan

Tabel 3. Tabel operator matematika



Gambar 1. 15. Penggunaan Operator Matematika pada Command Window

Adapun dalam penggunaan beberapa operator dalam satu baris perintah atau dalam suatu ekspresi, adakalanya kita perlu mengatur sendiri urutan suatu pengerjaan operasi aritmatika, karena setiap operator mengikuti aturan prioritas yang biasa (hirarki). Aturan tersebut dapat diringkas sebagai berikut :

Ekspresi dikerjakan dari kiri ke kanan dengan pemangkatan mempunyai prioritas tertinggi, diikuti dengan perkalian atau pembagian yang mempunyai prioritas yang sama, diikuti dengan penambahan dan pengurangan yang juga memiliki prioritas yang sama. Tanda kurung dapat digunakan untuk merubah urutan pengerjaan yang biasa dimana bagian yang dikerjakan terlebih dahulu adalah bagian yang yang ada di bagian kurung paling dalam kemudian keluar.

Berikut tabel prioritas keseluruhan operator dalam MATLAB.

Prioritas	Operator
1 (Tertinggi)	Kurung
2	Perpangkatan
3	Logika NOT (~)
4	Perkalian, pembagian
5	Penjumlahan, Pengurangan
6	Operator Relasional
7	Logika AND (&)
8 (Terendah)	Logika OR (|)

Tabel 4. Prioritas Operator Aritmatika

Contoh Soal 1: Misalnya anda mengambil kuliah sebanyak 12 SKS , yang terdiri dari Aljabar Elementer 4 SKS, Kalkulus 3 SKS , Statistika Dasar 3 SKS dan Wawasan Filsafat 2 SKS. Lalu pada akhir semester anda mendapat nilai sebagai berikut Aljabar Elementer A , Kalkulus B , Statistika Dasar C dan Wawasan Filsafat A . Dengan point nilai A=4 , B=3 , C=2 Berapa nilai IP anda ?

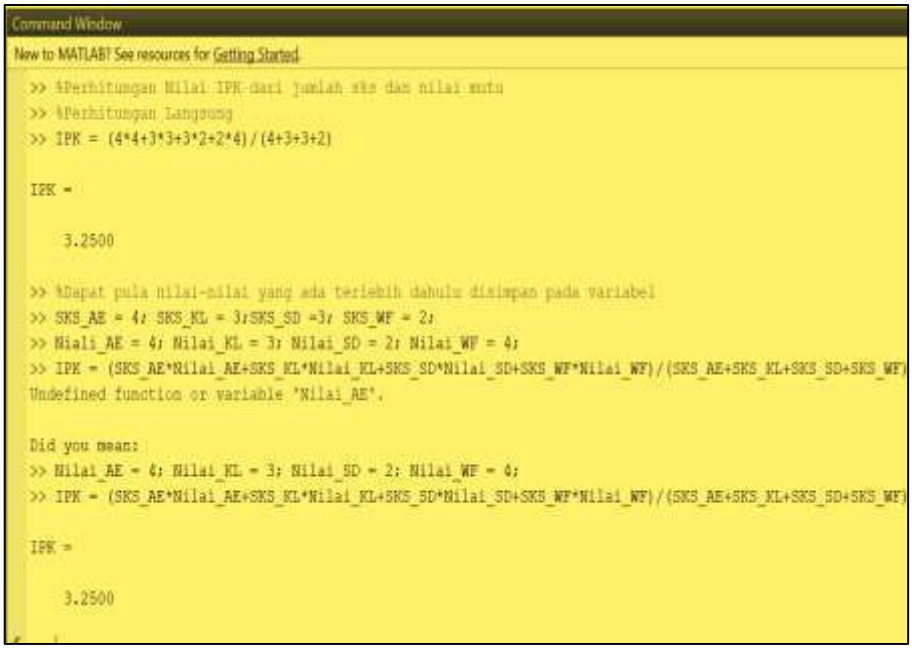
No.	Mata Kuliah	Kode	Jumlah SKS	Nilai Mutu
1	Aljabar Elementer	AE	4	A
2	Kalkulus	KL	3	B
3	Statistika Dasar	SD	3	C
4	Wawasan Filsafat	WF	2	A

Tabel 5. Perolehan Nilai

$$IPK = \frac{\sum SKS * Point\ Nilai\ Mutu}{\sum SKS}$$

Untuk memyelesaikan ini kita menggunakan pendekatan seperti perhitungan di kalkulator :

Melalui tabel di atas, perhitungan IPK dapat dilakukan pada command window dengan menerapkan penggunaan operator-operator aritmatika yang majemuk dalam satu perintah.



Gambar 1. 16. Perhitungan IPK pada Command Window dengan Menggunakan Operator Matematika

Kegiatan 5. Lakukan latihan penggunaan operator aritmatika secara majemuk pada ekspresi-ekspresi perhitungan dengan melibatkan variabel

❖ Operator Relasional

Selain Operator Aritmatika, MATLAB juga menyediakan operator logika dan relasional, hal ini diperlukan untuk menjawab pertanyaan benar atau salah dan salah satu manfaat yang penting dari kemampuan ini adalah untuk mengontrol urutan eksekusi sederetan perintah MATLAB (biasanya dalam M-File) berdasarkan pada hasil pertanyaan benar/salah.

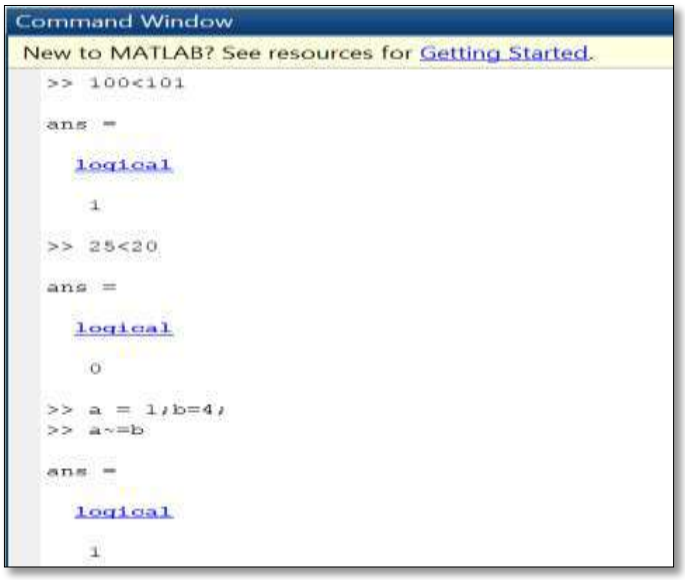
Sebagai masukan pada semua ekpresi relasi dan logika , MATLAB menganggap semua angka tidak nol sebagai benar, nol sebagi salah. Hasil dari semua ekspresi logika relasi dan logika adalah satu untuk benar dan nol untuk salah dengan tipe array logika yaitu hasilnya memuat bilangan 1 dan 0 yang tidak saja dapat digunakan untuk statemen matematika akan tetapi dapat juga untuk pengalamatan. Berikut operator relasional MATLAB terdiri dari semua pembandingan :

Operator Relasi	Deskripsi
<	Kurang dari
>	Lebih dari
<=	Kurang dari atau sama dengan
>=	Lebih dari atau sama dengan
=	Sama dengan
~=	Tidak sama dengan

Tabel 6. Operator Relasional

Operator relasi MATLAB dapat digunakan untuk membandingkan dua array berukuran sama atau untuk membandingkan array dengan skalar. Setiap ekspresi operator relasional mempunyai nilai kebenaran tersendiri. Angka 1 menandakan bahwa ekspresi bernilai benar, sedangkan 0 menandakan ekspresi bernilai salah.

Contoh di bawah ini menunjukkan berbagai jenis penggunaan operator relasional. Jika kita melihat hasil masing-masing perintah, nilai 0 menyatakan salah dan 1 menyatakan benar.



Gambar 1. 17. Penggunaan Operator Relasional

Kegiatan 6. Lakukan latihan penggunaan operator relasional untuk memaham cara kerja masing-masing operator.

❖ Operator Logika

Operator logika berguna untuk menggabungkan dua buah ekspresi relasional, selain itu operator logika juga digunakan untuk menegasikan atau membalik suatu ekspresi relasional. Di dalam operator logika, dikenal dua nilai kebenaran yaitu benar dan salah pada suatu ekspresi tunggal. Benar dinotasikan dengan nilai 1, sedangkan salah dinotasikan nilai 0. Dua atau lebih ekspresi tunggal yang digabungkan membentuk satu ekspresi baru yang juga memiliki nilai kebenaran. Selanjutnya Operator-operator logika dalam MATLAB disajikan dalam tabel berikut :

Operator Logika	Deskripsi	Kegunaan
&	AND	Membentuk ekspresi yang menghasilkan nilai kebenaran apabila seluruh operand bernilai benar
|	OR	Menghasilkan nilai benar kalau dalam penggabungan dua operand, salah satunya bernilai benar.
~	NOT	Menghasilkan nilai kebenaran yangberlawanan dari operand tunggal.

Tabel 7. Operator Logika

Berikut daftar kombinasi nilai kebenaran dari penggabungan dua ekspresi yang memiliki nilai kebenaran.

Ekspresi 1 (x)	Ekspresi 2 (y)	AND (&)	OR (|)
Benar (1)	Benar (1)	Benar (1)	Benar (1)
Benar (1)	Salah (0)	Salah (0)	Benar (1)
Salah (0)	Benar (1)	Salah (0)	Benar (1)
Salah (0)	Salah (0)	Salah (0)	Salah (0)

Tabel 8. Tabel Nilai Kebenaran

Dengan adanya operator logika, menjadikan penggunaaan MATLAB dalam pemecahan masalah matematika menjadi lebih variatif. Dengan operator logika, memungkin berbagai bentuk penyeleksian kondisi yang mempunyai manfaat yang besar.

F. Pengenalan Variabel

Istilah variabel di dalam ruang lingkup dunia matematika, telah sering kita dengar bahkan sudah sering kita gunakan, di mana variabel dapat dipandang sebagai peubah yang mengandung nilai tertentu baik diketahui maupun belum diketahui. Dalam dunia komputasi variabel, khususnya dalam MATLAB dikenal sebagai suatu nama yang dapat digunakan untuk menyimpan suatu nilai dan nilai yang ada didalamnya dapat diubah sewaktu-waktu. Selanjutnya variabel tersebut juga dapat dioperasikan dengan variabel lain yang memuat nilai tertentu.

Penggunaan variabel pada dunia matematika biasanya direpresentasikan dengan huruf-huruf tertentu yang memuat nilai tertentu, seperti *a, b, c, x , y, z* dan sebagainya. Namun pemilihan nama variabel pada MATLAB biasanya tidak hanya sekedar 1 karakter huruf saja. Berikut beberapa hal yang perlu diperhatikan dalam pemberian nama variabel :

Berikut beberapa contoh penggunaan variabel pada MATLAB yang dapat diterima oleh MATLAB dan yang ditolak oleh MATLAB

NAMA VARIABEL YANG DAPAT DITERIMA	NAMA VARIABEL YANG TIDAK DAPAT DITERIMA
A1	1A (Awalan Variabel Harus huruf)
Kelas_1	Kelas-1(Operator pengurangan tidak boleh digunakan)

- MATLAB bersifat case sensitif, seperti halnya dengan bahasa pemrograman lain MATLAB membedakan huruf kecil dan huruf kapital pada penamaan variabel. Sebagai contoh variabel1 ban variAbeL2 adalah dua variabel yang berbeda.
- Dalam pemberian nama variabel haus diawali dengan huruf, sedangkan karakter selanjutnya dapat berupa huruf, angka, atau tanda (_)
- Panjang nama variabel dapat mencapai 31 Karakter, jika nama variabel lebih dari 31 karakter maka karakter ke-32 dan seterusnya diabaikan.

`Tarbiyah456`	`Tarbiyah 456 (Tidak boleh ada spasi)`
`Subuh`	`Subuh? (Tidak Boleh menyisipkan tanda baca apapun) kecuali (_)`

Gambar 1. 18. Ketentuan Penggunaan Variabel

Catatan Teknis :

- Pemilihan nama variabel untuk merepresentasikan skalar menggunakan huruf kecil biasanya berupa **k, s** sedangkan nama larik (vektor atau matriks) biasanya menggunakan huruf kapital. Namun, tidak ada aturan baku untuk pemberian nama variabel pada skalar atau vektor.
- Kata tercadang pada MATLAB tidak dapat digunakan sebagai bariabel seperti **if, for, else, end** .

❖ Pengenalan Beberapa Variabel Khusus

Di dalam MATLAB tersedia beberapa variabel khusus, yaitu variabel yang dipakai oleh MATLAB dan memiliki makna secara khusus. Berikut beberapa variabel khusus yang tersedia di dalam MATLAB.

Variabel	Keterangan	Penjelasan
`Ans`	Menampung hasil suatu ekspresi yang dijadikan sebagai sebuah pernyataan	Perlu diketahui, MATLAB mempunyai batasan bilanagan real yang berkisar dari realmin sampai dengan realmax. Bila ada operasi yang menghasilkan nilai di luar
`Eps`	Bilangan terkecil dalam komputer yang apabila	

	ditambah dengan satu maka akan bernilai lebih besar dari satu	jangkauan tersebut, akan terjadi keadaan yang disebut *overflow* atau *underflow*. Jika nilai melebihi batas realmax dan disebut *underflow* jika nilai lebih kecil daripada realmin.  Gambar 1. 19. Variabel-variabel Khusus
`Flops`	Jumlah operasi bilangan pecahan	
`Inf`	Berasal dari kata "infinitive" yang artinya adalah tak terhingga (hasil 1/10)	
`NaN`	Berasal dari "Not-a-number" atau "bukan sebuah bilangan. Misalnya hasil dari 0/0	
`Pi`	Menyatakan bilangan π	
`realmin`	Bilanagn Real Positif terkecil	
`realmax`	Bilangan Real Positif terbesar	
Hasil di bawah ini menunjukkan bahwa : Keadaan overflow akan membuat nilai diubah menjadi tak berhingga Keadaan underflow akan membat nilai diubah menjadi nol		

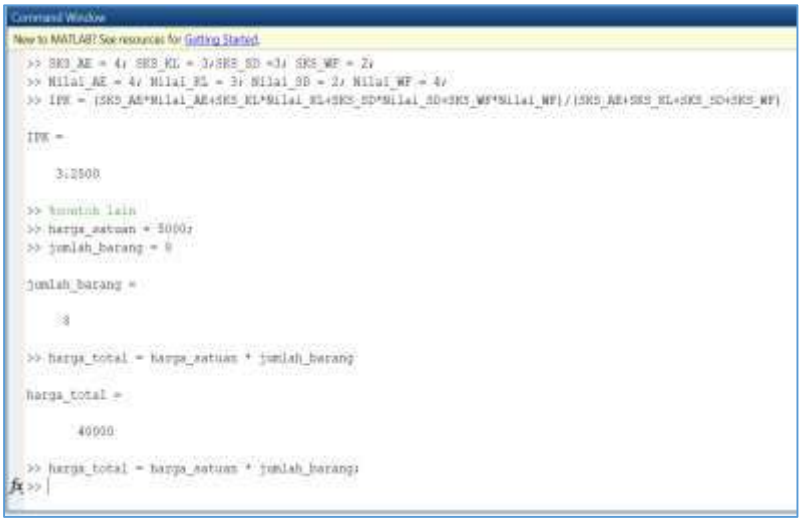
Tabel 9. Beberapa contoh variabel yang tesedia pada MATLAB

Kegiatan 7. Gunakan perintah realmin, realmax untuk melihat nilai-nilainya dalam MATLAB

❖ Menahan Tampilan Hasil Penugasan Variabel

Setelah memperkenalkan bagaimana penguunaan variabel dan beberapa variabel khusus dalam MATLAB. Kita melihat bahwa variabel yang didefinisikan di dalam MATLAB mengandung nilai atau isis di dalam variabel tersebut. MATLAB menyediakan mekanisme yang membuat penugasan tehadap suatu variabel tidak secara otomatis isi dari variabel

tersebut. Hal ini bisa dilakukan dengan menambahkan titiik-koma (;) di belakang pernyataan penugasan. Berikut dapat kita melihat contoh pendefinisian variabel yang tidak menampilkan isi.



Gambar 1. 20. Pendefinisian dan Penggunaan Variabel pada Command Window

Tampak bahwa :

Pada baris pertama dan kedua beberapa variabel didefinisikan di dalam satu baris dan diakhiri dengan tanda (;), setelah ditekan enter isi dari variabel tersebut tidak ditampilkan. Pada baris ketiga didefinisikan satu variabel baru yang tidak diakhiri dengan tanda (;), setelah ditekan enter maka isi dari variabel tersebut dimunculkan.

Kegiatan 8. Lakukan latihan pendefinisian variabel dengan menahan tampilan saat eksekusi

G. Tipe Data dalam Variabel

Sebelumnya kita telah mengenal karaktristik variabel, yang paling utama untuk dipahami pada variabel adalah adanya data yang dimilikinya atau dikandungnya. Data yang terdapat di dalam variabel atau dalam suatu ekspresi memiliki tipe yang dinyatakan dalam kelas. Sebuah kelas pada

dasarnya adalah gabungan antara tipe dan operasi yang dapat dikenakan terhadap nilai pada tipe tersebut. Berikut beberapa nama tipe data atau kelas yang tersedia di dalam MATLAB :

Bilangan pecahan atau titik mengambang :	
Single	Bilangan pecahan berpresisi tunggal
Double	Bilanag pecahan berpresisi ganda
Bilangan Bulat	
int8	Bilanag bulat berukuran 8-bit
int16	Bilangan bulat berukuran 16-bit
int32	Bilangan bulat berukuran 32-bit
int64	Bilangan bulat berukuran 64-bit
Karakter	
Char	Sebuah Karakter atau deretan karakter (string)
Logika	
Logical	Nilai logika true (benar) atau false (salah)

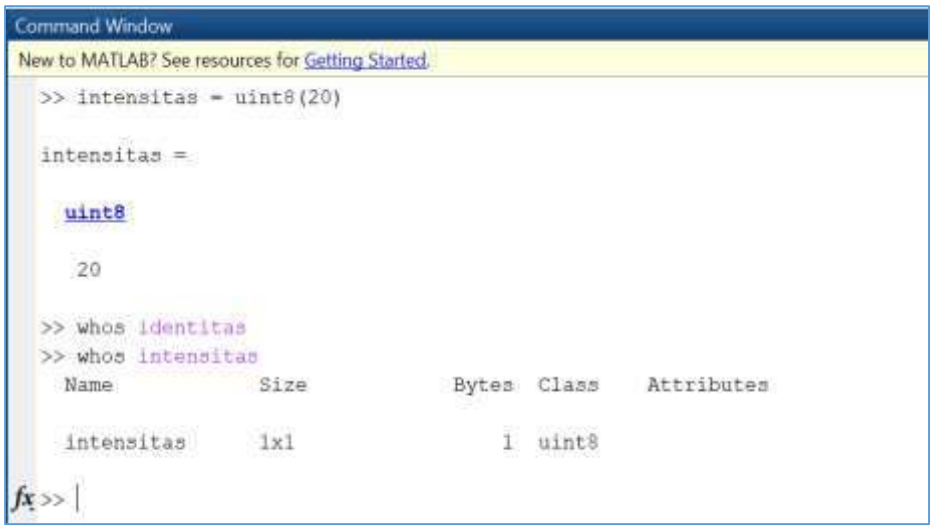
Tabel 10. Tipe Data dalam MATLAB

Untuk membuat suatu variabel dengan kelas tertentu secara eksplisit, kita bisa menggunakan format sebagai berikut : `X = kelas(nilai)`

```
intensitas = uint8(20)
```

dengan cara seperti itu, intensitas berkelas **uint8.** Perlu diketahui, uint8 berarti kelas bilangan bulat tak bertanda (hanya mencakup nilai positif).

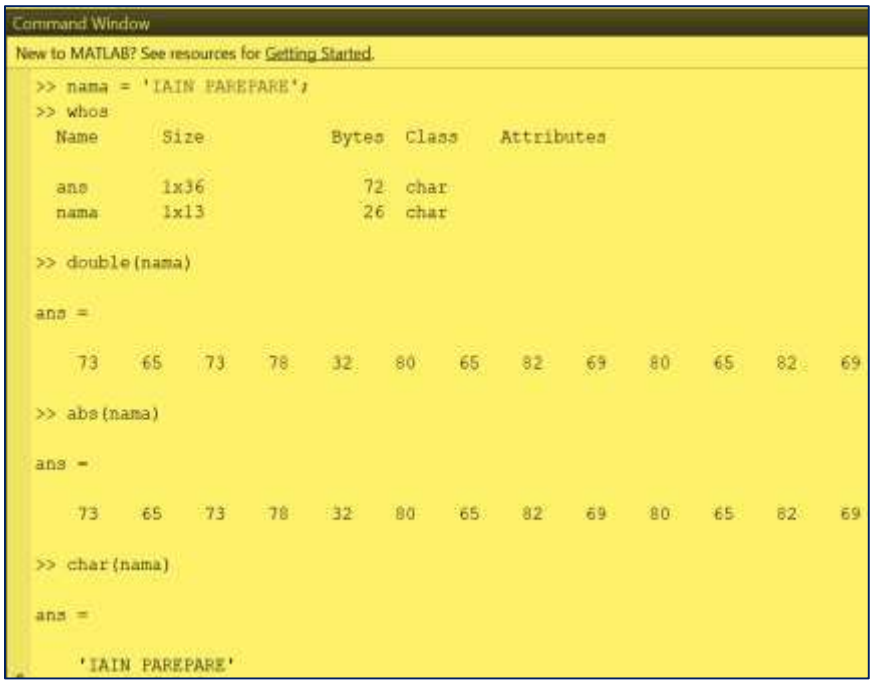
Kita bisa melihat informasi mengenai variabel intensitas dengan memberikan perintah whos :



Gambar 1. 21. Penggunaan Perintah whos dalam mengidentifikasi tipe data

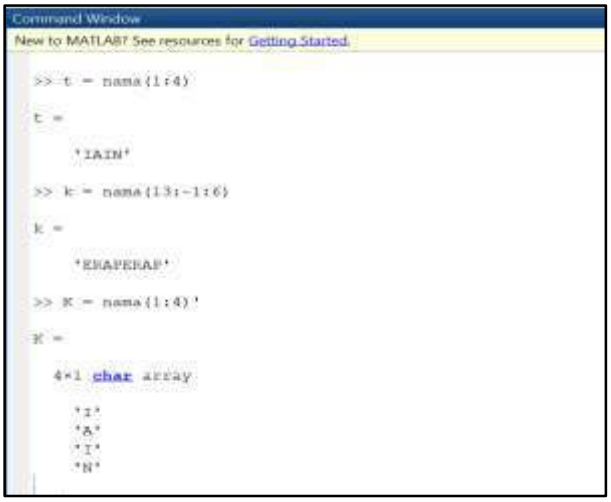
Pada Gambar 1.21 bahwa kelas untuk variabel `intensitas` berupa uint8 String dalam MATLAB adalah type data yang terdiri atas huruf-huruf dan atau nilai-nilai ASCII yang ditampilkan representasinya. String adalah teks yang diawali dan diakhiri dengan apostrof ‘ ‘.

Setiap karakter dalam suatu string adalah satu elemen dalam array, dengan setiap elemennya sebesar 2 byte.Untuk melihat representasi ASCII karakter string dapat dilakukan dengan melakukan operasi aritmetik terhadap string atau mengkonversikannya menggunakan fungsi double. Fungsi char menyediakan transformasi balikan.



Gambar 1. 22. Representasi ASCII suatu karakter String

Karena string merupakan array numeric dengan atribut khusus, string dapat dimanipulasi dengan menggunakan semua metode manipulasi array yang tersedia dalam MATLAB. Dari contoh di atas elemen ke 1 sampai ke 4 memuat kata *IAIN.*Jika karakter ke 13 sampai ke enam diperoleh : ERAPERAP dan jika penggunaan operator transpose maka kata IAIN akan dibaca dalam format kolom yaitu :



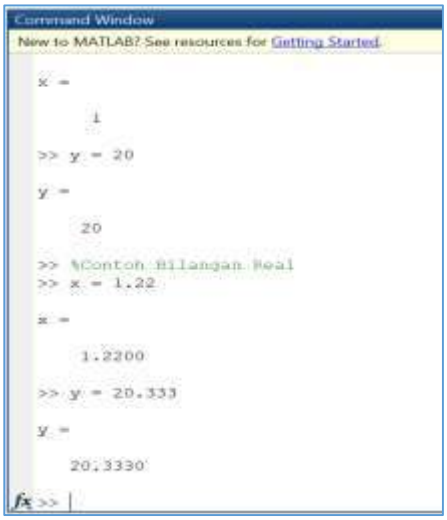
Gambar 1. 23. Penggabungan String

Penggabungan string mengikuti aturan penggabungan array, berikut contoh pengolahan String seabagi Array

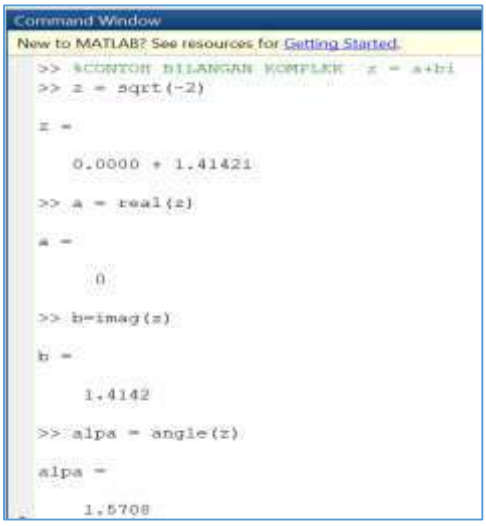
```
» a='jika anda belajar rajin,'
a =
jika anda belajar rajin,
» b='maka anda akan dapat nilai yang bagus'
b =
maka anda akan dapat nilai yang bagus
» c=[a b]
c =
jika anda belajar rajin,maka anda akan dapat nilai yang
bagus
```

di dalam MATLAB kita perlu memahami bahwa ada tiga tipe bilangan yang mempunyai kondisi dan karakter tersendiri.

- ❖ Bilangan bulat (integer)
- ❖ Bilangan real
- ❖ Bilangan kompleks



Gambar 1. 24. Tipe Bilangan Real



Gambar 1. 25. Tipe Bilangan Imaginer

❖ **Menyimpan dan Memanggil Data**

Metode penyimapan data juga sangatlah penting untuk dipahami. Dalam menyimpan dan memanggil data dari file pilih **File Save Workspace As …** . Untuk memanggil data digunakan pilihan **Load WorkSpace As** atau **Open** pada menu file. Sedangkan untuk mengimport data , untuk **MATLAB**

versi 6 keatas pilih **file Import Data .. . MATLAB** juga menyediakan dua perintah **---- save dan load -------** yang jauh lebih fleksibel. Perintah **save** untuk menyimpan satu atau lebih variabel dalam file format Yang sesuai dengan pilihan anda.

contoh :

1. Menyimpan semua variabel **MATLAB** dalam format biner di file **MATLAB**.mat

```
clear all
x=1:10;y=10:10:10:100; % membuat array baru
save
Saving to: MATLAB.mat
save data
```

2. Menyimpan semua variabel **MATLAB** dalam format biner di file data.mat

```
save data_x x
```

3. Menyimpan variabel x dalam format biner di file data_x.mat

```
save data_xy x y /ascii
```

4. Menyimpan variabel x dan y dalam format biner di file data_xy dalam format ascii untuk membuka data digunakan perintah load, contoh;

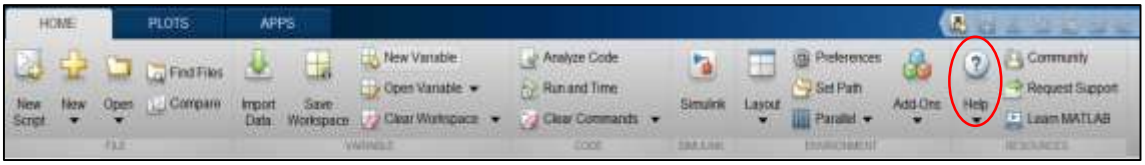
```
load data_x.mat
```

H. Pengenalan Fasilitas Help

Dalam mengoperasikan MATLAB, kadang kita sebagai user membutuhkan informasi atau penjelasan mengenai fungsi, contoh penggunaan fungsi, atau ingin mempelajari topik-topik tertentu. Masalah tersebut dapat diatasi dengan menggunakan Fasilitas Help. MATLAB menyediakan fasilitas bantuan yang sangat berguna dalam membuat suatu program khusus pada MATLAB. Untuk mendapatkan bantuan tentang suatu materi, gunakan help diikuti dengan materi yang penjelasannnya ingin diketahui. Contoh :

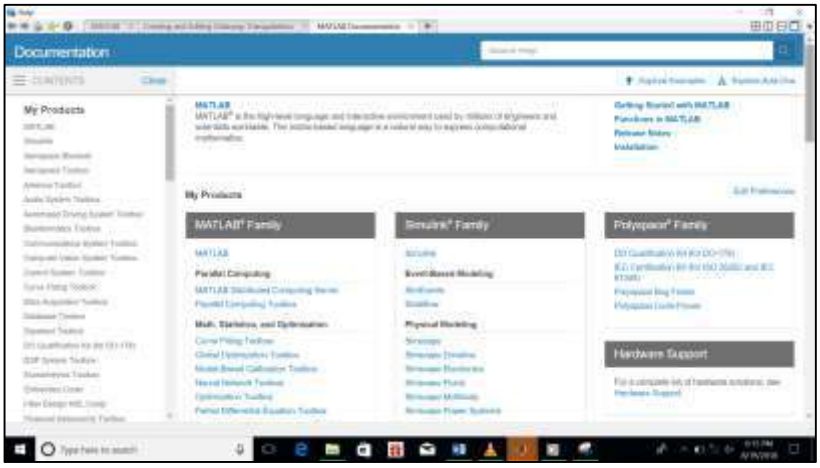
Gambar 1. 26. Penggunaan perintah help pada command window untuk mwncari informasi mengenai nilai pi

Dalam menampilkan informasi yang lebih banyak mengenai help kita dapat masuk pada jendela help, kita dapat mengklik langsung tombol help pada sub menu utama.

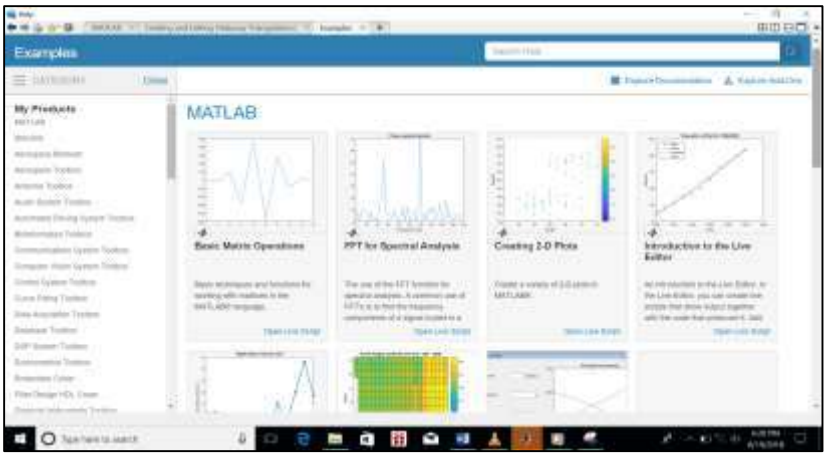


Gambar 1. 27. Tombol Help pada Menu Home

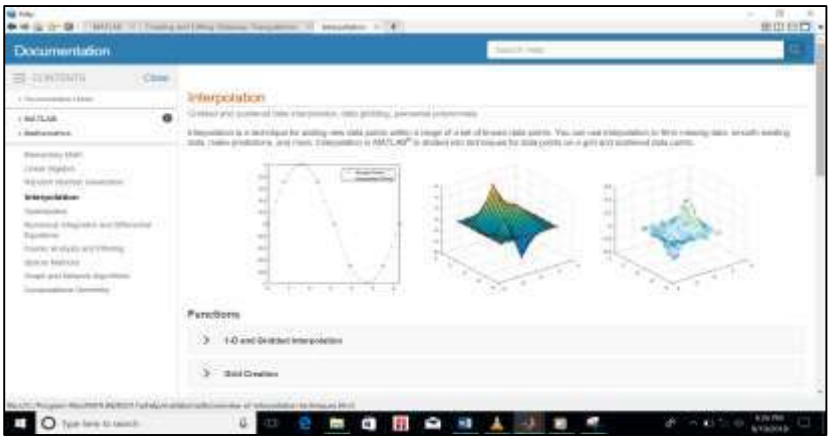
Dengan mengklik tombol help maka akan terbuka jendela help yang didalamnya lebih interaktif dalam mencari materi-materi dan contoh penggunaan fungsi-fungsi



Gambar 1. 28. Jendela Help



Gambar 1. 29. Topik-topik pilihan yang akan dicari informasinya



Gambar 1. 30. Informasi dan penggunaan Interpolasi pada jendela help

I. Sumber-sumber MATLAB yang ada di Internet

Jika anda ingin lebih mendalami lagi tentang MATLAB dan mendownloadnya maka di bawah ini beberapa informasi yang berkenaan dengan MATLAB, dan informasinya bisa didapatkan dibeberapa situs berikut ini :

J. Rangkuman

- Web site MathWorks : http://www.mathworks.com/ disitus ini bisa di dapatkan informasi tentang produk baru MATLAB seperti buku dan lain-lainnya.
- Newsgroup MATLAB: news://saluki news.siu.edu/comp.soft-sys.MATLAB/
- http://dir.yahoo.com/science/mathematics/software/MATLAB/ penggunaan source informasi tentang MATLAB dan merupakan langkah awal untuk memperoleh web site MATLAB lainnya.
- http://www.cse.uiuc.edu/cse301/MATLAB.html, web site ini merupakan websitenya University of Illinois di Champaign-Urbana, yang menyediakan beberapa link untuk MATLAB di internet
- Mastering MATLAB Web site: http://www.eece.maine.edu/mm

- Matematika komputasi hadir sebagai wujud dari respon atas kemajuan teknologi di bidang ilmu matematika untuk mengintegrasikan keilmuan-keilmuan dasar dalam matematika seperti aljabar, kalkulus, geometri, dan teori bilangan ke dalam kajian terapan seperti matematika fisika, matematika biologi dan sebagainya.
- MATLAB sebagai bahasa pemrograman yang dikembangkan pada generasi komputer keempat, memiliki peranan yang vital pada berbagai jenis pemecahan masalah komputasional yang dapat diterapkan dalam berbagai kelompok keilmuan lainnya.
- MATLAB dikembangkan oleh mathwork.inc paling efisien untuk perhitungan numerik berbasis matriks. Dengan demikian jika di dalam perhitungan kita dapat menformulasikan masalah ke dalam format matriks maka MATLAB merupakan software terbaik untuk penyelesaian numeriknya.

- Hal-hal mendasar untuk dipahami dalam memulai penggunaan MATLAB adalah instalas MATLAB ke perangkat komputer, susunan dan fungsi dari interface MATLAB meliputi, menu, sub menu dan ruang kerja pada MATLAB.

- Pada bagian Interface utama MALAB terdapat empat bagian utama dalam layoutnya, yaitu Command Window sebagai jendela untuk menjalankan perintah, Command History untuk melihat riwayat perintah yang telah digunakan, current directory sebagai ruang untuk memanajemen file, memanggil dan mengaktifkan file pada MATLAB, dan bagian terakhir adalah workspace adalah rekaman variabel dan klasifikasi data yang termuat dalam variabel.
- Dengan mengenal bagian-bagian Interface MATLAB, penting kemudian untuk mengenal variabel beserta kriteria-kriteria variabel yang legal bagi MATLAB, penggunaan operator aritmatika, operator logika dan operator relasional, serta tipe-tipe data yang ada di dalam MATLAB.
- Ketika pengguna mengalami kesulitan dalam menggunakan fungsi-fungsi yang ada di dalam MATLAB, maka dapat menggunakan fasilitas help yang di dalamnya menjadi virtual bagi pengguna untuk mengeksplorasi penjelasan dari fungsi-fungsi yang diiginkan.

K. Latihan

Latihan 1. 1: Misalkan diberikan $u = 4$ dan $v = 7$. Tentukanlah hasil dari ekspresi berikut dengan menggunakan MATLAB

- $\frac{14u}{4v}$
- $\frac{5uv^2}{(4u+3v)}$
- $\frac{5uv^2}{(v^3+3u^2)}$
- $\frac{4}{3}\pi v^2$

Latihan 1. 2: Berikut diberikan data pengukuran tes penalaran verbal (X) dan tes skor bahasa Inggris (Y)

Anak	A	B	C	D	E	F	G	H
X	112	113	110	113	112	114	109	113
Y	69	65	75	70	70	75	65	76

Dari data diatas ini gunakan operator matematika untuk menentukan .

- Nilai Nilai maksimum data X
- Nilai Maksimu data Y
- Nilai ratarata data X
- Nilai rata data Y

Latihan 1. 3: Gunakan fasilitas help untuk mengeksplorasi

- Penjumlahan Matriks
- Perkalian Matriks
- Fungsi-fungsi matematika
- Fungsi-fungsi statistika
- Jenis-jenis grafik

Latihan 1. 4: Lakukanlah pengolahan data string untuk

- Menuliskan nama lengkap anda dan NIM anda dengan menggunakan anama variabel “nama saya”
- Akses karakter dari nama anda huruf pertama dan huruf terakhir
- Akses karakter dari NIM anda pada digit ke dua sampai keempat.
- Gabungkan nama dan NIM anda

Latihan 1. 5: Tentukan status dari variabel berikut, legal atau tidak legal, jelaskan !

- ALPHA_1
- 1alpha
- Beta 1
- Beta alpha
- Institutagamaislamnegeripareparefakultasterbiyah
- Whats’up

Latihan 1. 6: Misalkan $\boldsymbol{u} = (-3, 2, 1, 0)$, $\boldsymbol{v} = (4, 7, -3, 2)$, $\boldsymbol{w} = (5, -2, 8, 1)$

- Tentukan $(6\boldsymbol{v} - \boldsymbol{w}) - (4\boldsymbol{u} + \boldsymbol{v})$!
- Tentukan $\|3\boldsymbol{u} - 5\boldsymbol{v} + \boldsymbol{w}\|$

BAB 2. PENGGUNAAN M-FILE

Kemampuan Akhir yang Diharapkan

- ❖ Mahasiswa dapat menjalankan berkas skrip M-File dalam menyelesaiakan masalah Matematika.
- ❖ Mahasiswa dapat secara teknis membedakan kelebihan penggunaan berkas scrip M-File dengan Command Window
- ❖ Mahasiswa dapat menangani pemasukan data yang interaktif dari keyboard
- ❖ Mahasiswa dapat menampilkan dengan disp dan memformat keluaran.
- ❖ Mahasswa dapat menggunakan M-File dalam membuat program perhitungan voluume dan luas permukaan bangun ruang.

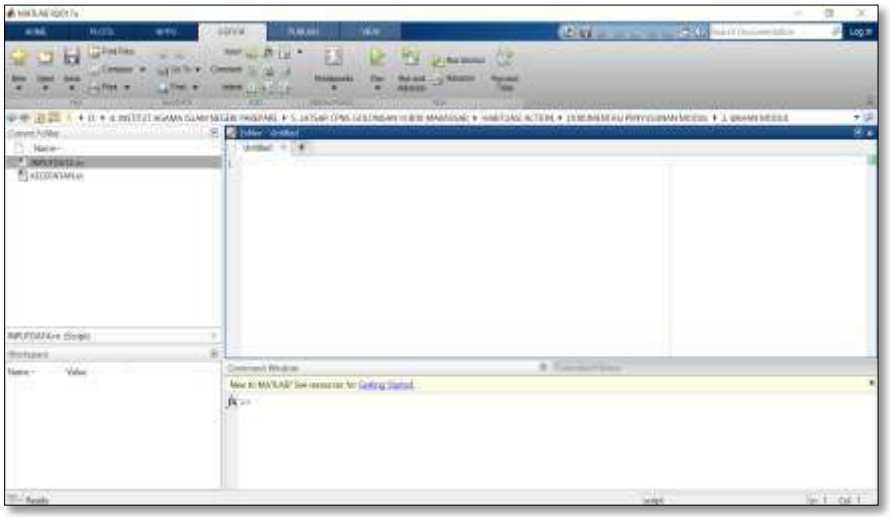
A. PENGENALAN M-FILE

Pada materi Bab 1, kita telah mengenal dan mampu menggunakan command window dalam menuliskan perintah-perintah baik dalam bentuk pendefinisian variabel maupun dalam bentuk ekspresi yang melibatkan operator-operator khusus. Penulisan rangkaian perintah pada command window sebelumnya memiliki sejumlah kekurangan berupa kurang praktis untuk menjalankan sederetan perintah yang sama. Selain itu perintah yang dijalankan berbasis baris. Sehingga ketika program yang dibuat melibatkan banyak rangkaian perintah, maka hal ini menjadi kurang efektif dan efisien. Untuk mengatasi hal tersebut, MATLAB menyediakan solusi berupa adanya berkas skrip atau yang dikenal dengan M-file. Dinamakan M-file karena nama dari setiap file yang dibuat berekstensi *.m.

Pada MATLAB, berkas skrip atau berkas teks ini dibuat dalam editor. Satu jendela editor, didalamnya kita dapat menuliskan deretan perintah MATLAB. Jika halaman editor disimpan maka akan menjadi satu M-file yang tersimpan pada directori yang dipilih. Dengan M-file, kita mampu menghimpun perintah-perintah yang biasanya diketikkan pada prompt di command Window. Pendefinisian variabel ataupun ekspresi, serta aturan-aturan yang berlaku pada command window juga berlaku pada editor M-file. Setiap menjalankan perintah ataupun sekumpulan perintah pada

Editor, maka juga akan mempengaruhi isi dari workspace dan command history. Untuk mengaktifkan satu jendela editor baru dapat dilakukan dengan mengklik tombol New Script pada menu Home atau dengan menekan tombol Ctrl+N yang berarti membuka satu editor baru.

Sebelum kita menggunakan berkas skrip maka penting untuk mengenal kembali sub window current directory. Telah dijelaskan pada pertemuan sebelumnya bahwa current directory menyatakan folder atau directory yang sedang aktif. Current Directory (Directory Terkini) merupakan sub jendela yang digunakan untuk memanggil, ataupun mengaktifkan serta menampilkan file-file yang sedang bekerja, ataupun yang akan diaktifkan. Sehingga file yang akan dijalankan perlu dipastikan bahwa folder atau directory dimana file berada sedang terbuka (exist) pada current directory. Jika M-file yang yang dibuat belum tersimpan maka kembali perlu dipastikan bahwa folder atau directory yang akan ditempati untuk menyimpan M-file sedang aktif pada current directory pada MATLAB. Berikut beberapa hal yang perlu diketahui dalam menggunakan berkas script :



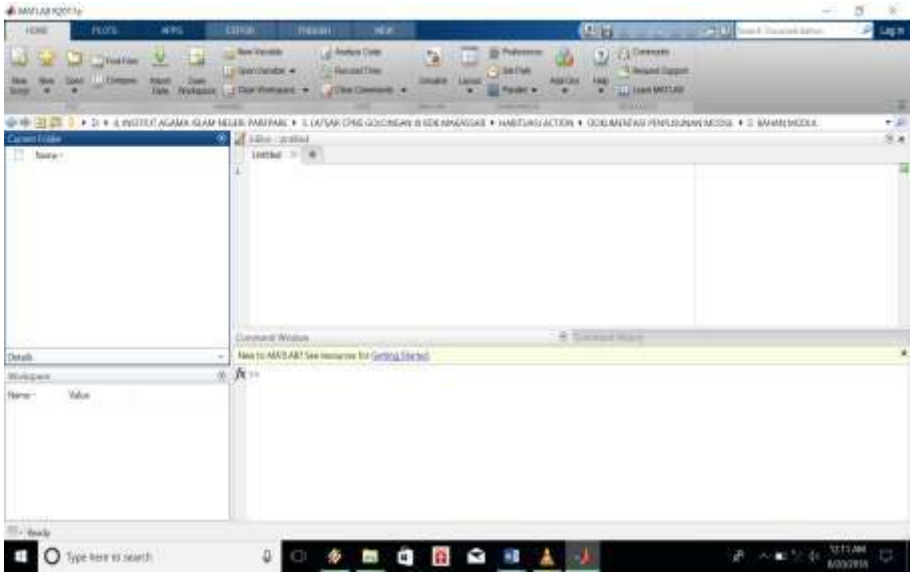
Gambar 2. 1. Layout Jendela Editor

❖ **Mengaktifkan Jendela Editor**

Untuk mengaktifkan editor dapat dilakukan dengan :

- ❖ Klik tombol New Script pada Menu Home
- ❖ Kemudian akan muncul jendela editor yang dengan sendirinya mengatur tempat pada ruang kerja utama.
- ❖ Layout dari MATLAB di bawah ini dapat diatur kembali posisinya.

Berikut tampilan Jendela editor yang baru dengan layout kolom baris.



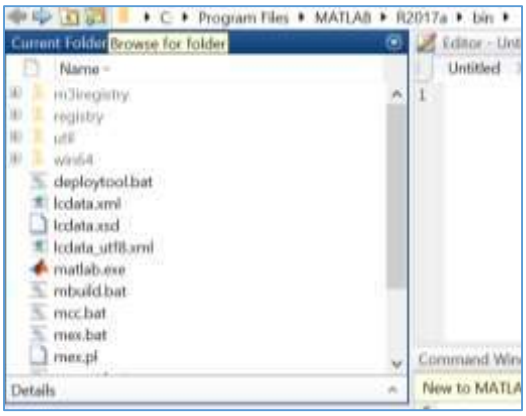
Gambar 2. 2. Mengaktifkan Jendela Editor

Cara yang lain dapat dilakukan dengan menekan tombol Ctrl+N

❖ Menyimpan File

Current Folder secara standar berada pada C:Program Files/ MATLAB/RR2017aa/bin. Untuk mengubah current folder yang ada maka :

- ❖ Klik browse for folder
- ❖ Pilih pada directory mana M-file akan disimpan.



Gambar 2. 3. Browse For Folder

Dari sini diperoleh bahwa current folder pada MATLAB berubah. Pada kasus ini dipilih Current Foldernya pada Directori D:/…../BAHAN MODUL.

❖ **Bekerja pada jendela Editor.**

Ketika kita baru mengguunakan jendela editor, Misal pada jendela editor kita menuliskan beberapa baris komentar dan beberapa baris ekspresi, maka program tersebut tidak dapat dijalankan sebelum tersimpan pada current folder yang aktif .

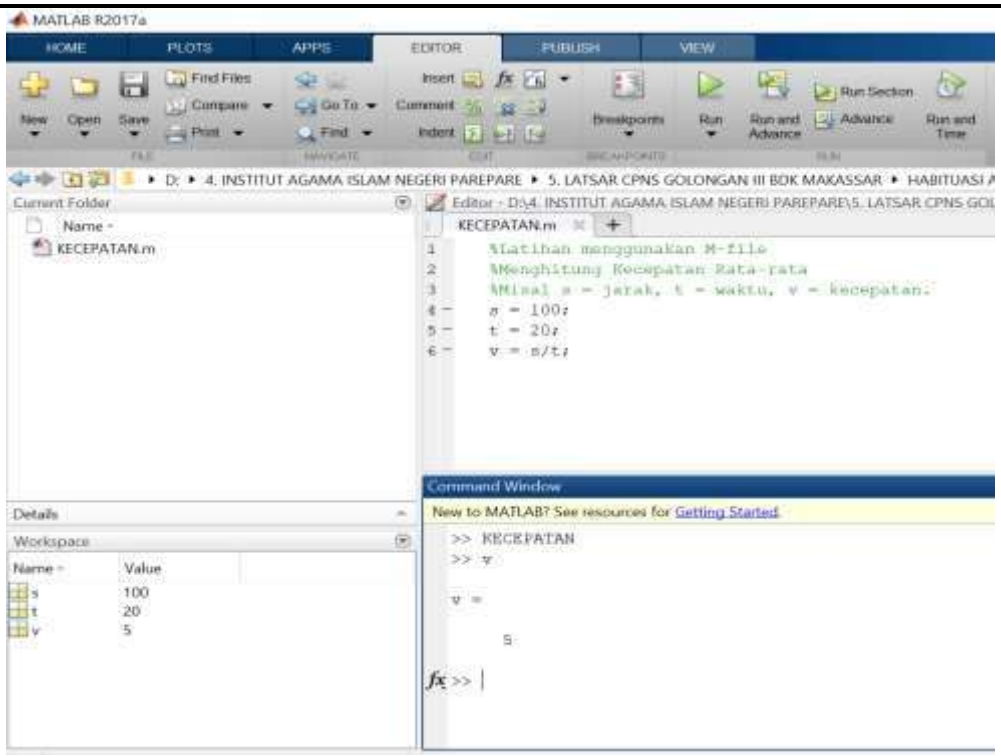
- ❖ Ketikkan sejumlah perintah pada editor
- ❖ Ctrl+S
- ❖ Secara otomatis tempat penyimpanan akan terarah pada current folder yang sedang aktif.
- ❖ Beri nama, kemudian klik tombol save.

Pada kasus ini M-file ditempatkan pada folder BAHAN MODUL dengan nama kecepatan. Terlihat bahwa sebelum disimpan atau dijalankan current folder dan workspace masih dalam keadaan kosong.

❖ **Menjalankan Jendela Editor M-file**

Setelah membuat sekumpulan perintah pada editor baru dan menyimpannya, maka current folder secara otomatis berisi M-file yang siap dijalankan. Dalam menjalankan M-file yang sudah aktif dapat dilakukan dengan 2 cara.

- ❖ Cara pertama : Klik run pada menu editor.
- ❖ Cara kedua : dengan menuliskan nama M-file pada command window dalam hal ini (Kecepatan)

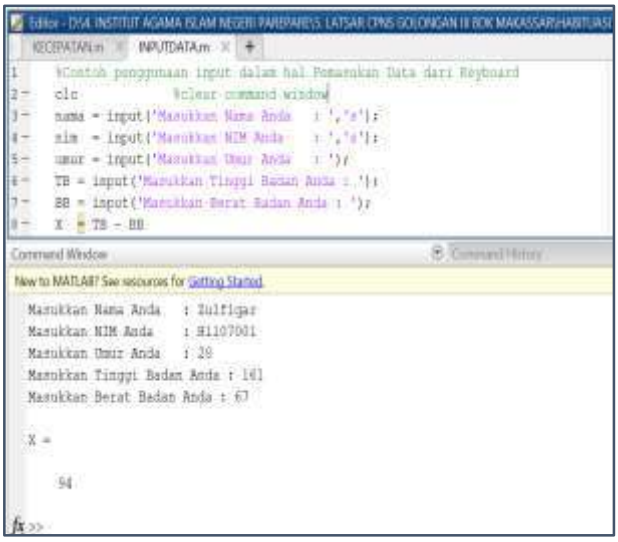


Gambar 2. 4. Menjalankan jendela editor

Setelah menjalankan M-file ini, seluruh variabel yang didefinisikan terekam di dalam workspace.

❖ **Menangani Pemasukan Data dari Keyboard**

Sebelumnya kita telah mempelajari bagaimana mendefinisikan data (numerik, teks) dari suatu variabel baik melalui command window, maupun pada jendela editor. Pada kasus ini, terkadang kita membutuhkan adanya nilai atau data yang dimasukkan secara interaktif. Pada penanganan seperti itu, dimungkinkan untuk memproses data yang berasal dari keyboard ketika M-file dieksekusi. Kebutuhan tersebut, MATLAB menyediakan pernyataan bernama **input.** Berikut contoh penggunaannya :



Gambar 2. 5. Contoh Penggunaan Input

Contoh pada gambar 2.5 di atas menunjukkan penggunaan perintah input dalam mendefinisikan data dari suatu variabel yang dibuat pada M-file. Ketika kumpulan perintah pada jendela editor dijalankan, diperoleh tampilan

```
Masukkan Nama Anda    :
```

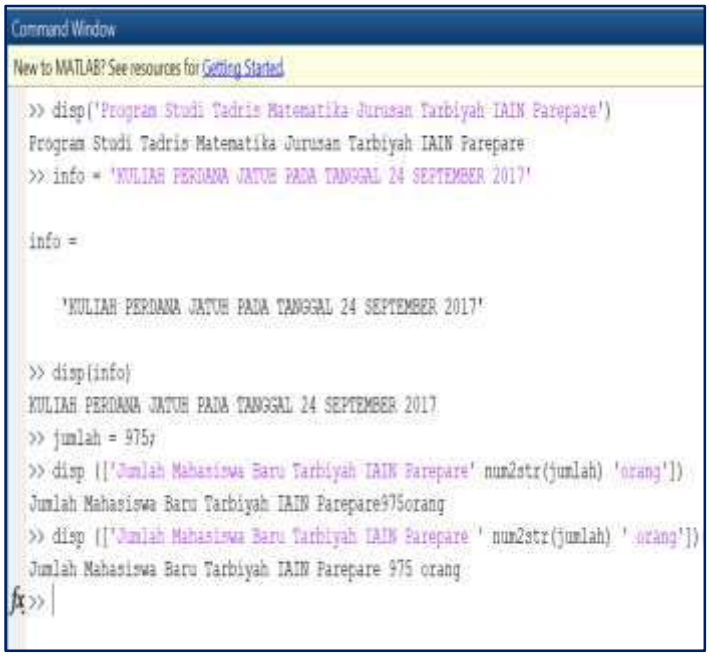
Selanjutnya menunggu pemakai memasukkan namanya serta menunggu tombol Enter ditekan oleh pemakai. Argumen ‘s’ menyatakan bahwa yang dimasukkan dari keyboard adalah suatu string. Jika argumen kedua tidak dimasukkan, MATLAB akan mendefinisikan string yang dimasukkan dari keyboard dan hasil evaluasi menjadi nilai seperti variabel nilai, TB – BB. Nilai-nilai yang dimasukkan kemudian dapat dioperasikan satu sama lain.

B. Mengatur Format Tampilan

❖ Menampilkan dengan Disp

Sejauh ini, dalam menampilkan isi suatu variabel dapat dilakukan dengan mengetikkan nama variabelnya pada command window. Namun penting untuk dietahui bahwa, kita dapat menampilkan suatu data baik berupa teks maupun dalam foemat numerik dengan menggunakan perintah

disp. Berikut contoh yang menunjukkan contoh penggunaan perintah disp untuk menampilkan, mengatur tampilan teks dengan disp maupun penampilan isi variabel yang berisi teks.



Gambar 2. 6. Contoh penggunaan disp dalam menggabungkan informasi string dan isi sebuah variabel.

Dapat dilihat bahwa pada gambar 2.6., peggabungan antara string dilakukan dengan meletakkan ketiga item tersebut dalam tanda [ ]. Perlu juga diketahui, notasi [ ] adalah notasi larik (array) dan elemen dalam larik harus bertioe yang sama.

- **Memformat Keluaran dengan perintah fprintf**

Format keluaran merupakan salah satu bagian yang penting untuk dikenali dalam menggunakan MATLAB yang berorientasi pada ilmu matematika. Format keluaran berkaitan dengan penggabungan sejumlah item yang akan menjadi keluaran ke layar Command Window.

Sebelumnya telah dikenal perintah display yang dapat menampilkan isi suatu variabel dengan cara menyebutkan nama variabel. Namun kegiatan memformat keluaran, kita dapat menggunakan perintah **fprintf** yang lebih praktis dibanding perintah **disp** . Perintah ini memiliki format standar sebagai berikut :

```
fprintf(string_pemformat, nilai1, nilai2, …)
```

fprintf merupakan fungsi yang digunakan untuk memformat keluaran dengan argument masukan :

- ❖ Argumen masukan pertama berupa string yang menentukan format keluaran.
- ❖ Argumen masukan kedua dan seterusnya berupa nilai-nilai yang akan diformat.

Contoh : `fprintf(string_pemformat, nilai1, nilai2, …)`

Berikut contoh pemformatan keluaran bilangan Real (B berarti satu karakter dengan spasi kosong)

PERINTAH	HASIL	KETERANGAN
`fprintf('%f',pi)`	`3.141593`	Format Bilangan Real dengan Notasi Biasa dengan 6 angka dibelakang koma dari angka pi
`fprintf('%3.0f',pi)`	`Bb3` `B= satu spasi kosong`	Format Bilangan Real dengan Notasi Biasa 3 karakter dengan 0 angka dibelakang koma dari angka pi
`fprintf('%2.0f',pi)`	`B3` `B = satu karakter dengan satu spasi kosong`	Format Bilangan Real dengan Notasi Biasa 3 karakter dengan 0 angka dibelakang koma dari angka pi
`fprintf('%1.0f',pi)`	`3`	Format Bilangan Real dengan Notasi Biasa 1 karakter dengan 0 angka dibelakang koma dari angka pi
`fprintf('%0.0f',pi)`	`3`	Format Bilangan Real dengan Notasi Biasa 1 karakter dengan 0 angka dibelakang koma dari angka pi
`fprintf('%0f',pi)`	`3`	Format Bilangan Real dengan Notasi Biasa 1

		karakter dengan 0 angka dibelakang koma dari angka pi
`fprintf('%.1f',pi)`	`3.1`	Format Bilangan Real dengan Notasi Biasa 1 karakter dengan 0 angka dibelakang koma dari angka pi
`fprintf('%.2f',pi)`	`3.14`	Format Bilangan Real dengan Notasi biasa dengan 2 karakter dibelakang koma dari angka pi
`fprintf('%.3f',pi)`	`3.142`	Format Bilangan Real dengan Notasi biasa dengan 3 karakter dibelakang koma dari angka pi
`fprintf('%5.2f',pi)`	`B3.14`	Format Bilangan Real dengan Notasi Biasa 5 karakter dengan 2 angka dibelakang koma dari angka pi
`fprintf('%10f',pi)`	`3.1415926536`	Format Bilangan Real dengan Notasi Biasa dengan 10 angka dibelakang koma dari angka pi
`fprintf('%e',pi)`	`3.141593e+000`	Format Bilangan Real dengan Notasi Sains 6 karakter di belakang koma diikuti dengan e+000 sebagai notasi ilmiah.
`fprintf('%.2e',pi)`	`3.14e+000`	Format Bilangan Real dengan Notasi Sains 2 karakter di belakang koma e+000 sebagai notasi ilmiah.
`fprintf('%.3e',pi)`	`3.142e+000`	Format Bilangan Real dengan Notasi Sains 3 karakter di belakang koma e+000 sebagai notasi ilmiah.

`fprintf('%.4e',pi)`	`3.1426e+000`	Format Bilangan Real dengan Notasi Sains 4 karakter di belakang koma e+000 sebagai notasi ilmiah.
`fprintf('%.10e',pi)`	`3.1415925636e+000`	Format Bilangan Real dengan Notasi Sains 10 karakter di belakang koma e+000 sebagai notasi ilmiah.
`fprintf('%E',pi)`	`3.141593E+000`	Format Bilangan Real dengan Notasi Sains 6 karakter di belakang koma diikuti dengan E+000 sebagai notasi ilmiah.
`fprintf('%.2E',pi)`	`3.14E+000`	Format Bilangan Real dengan Notasi Sains 2 karakter di belakang koma E+000 sebagai notasi ilmiah.
`fprintf('%.3E',pi)`	`3.142E+000`	Format Bilangan Real dengan Notasi Sains 3 karakter di belakang koma E+000 sebagai notasi ilmiah.
`fprintf('%.4eE',pi)`	`3.1426E+000`	Format Bilangan Real dengan Notasi Sains 4 karakter di belakang koma E+000 sebagai notasi ilmiah.
`fprintf('%.10eE',pi)`	`3.1415925636E+000`	Format Bilangan Real dengan Notasi Sains 10 karakter di belakang koma E+000 sebagai notasi ilmiah.
`fpritf('%g',pi)`	`3.14159`	Format Bilangan real dengan notasi biasa (serupa dengan f)
`fpritf('%.2g',pi)`	`3.1`	Format Bilangan real dengan notasi biasa (serupa dengan f) 2 karakter(.1) setelah bilangan bulat
`fpritf('%.3g',pi)`	`3.14`	Format Bilangan real dengan notasi biasa (serupa dengan f) 3

		karakter(.14) setelah bilangan bulat
`fpritf('%.4g',pi)`	`3.142`	Format Bilangan real dengan notasi biasa (serupa dengan f) 4 karakter(.142) setelah bilangan bulat
`fpritf('%.10g',pi)`	`3.141592654`	Format Bilangan real dengan notasi biasa (serupa dengan f) 10 karakter(.141592654) setelah bilangan bulat
Catatan : *Teks atau string di MATLAB dapat di tampilkan dengan cara menggunakan ' di awal dan ' diakhir perintah ini mirip dengan disp, namun lebih memiliki fleksibilitas dalam format output yang diinginkan.*		

Tabel 11. Perintah -perintah untuk Mengatur format luaran bilangan real

C. Pengenalan Fungsi Matematika

MATLAB mempunyai banyak fungsi yang berguna untuk melakukan operasi tertentu. Fungsi merupakan suatu nama yang mewakili sekumpulan operasi tertentu yang secara umum dapat dibedakan atas **built function** dan **function by user defined** dengan format `namafungsi(input1,input2,…input n)`. Sebagian besar fungsi tersebut hampir sama dengan bila anda menuliskannya secara matematis. Pada bagian ini akan diperkenalkan beberapa fungsi matematika yang berkategori **built in function** dalam hal ini fungsi yang sudah tersedia di dalam MATLAB. Berikut beberapa fungsi matematika yang tersedia di dalam MATLAB

FUNGSI	KETERANGAN

`abs(x)`
`contoh :`
`abs(-25)= 25`
`abs(3+4i) = 5`
	Fungsi ini meghasilkan nilai absolut suatu bilanagn. Jika dikenakan pada bilangan kompleks, hasilnya berupa magnitudenya.

`angle(x)` `contoh :`	Fungsi ini menghasilkan sudut dari suatu bilanagn kompleks (x). Satuan sudut adalah radian.

angle(4-4i) = -0.7854	
ceil(x) contoh : ceil(4.5) = 5 ceil(-4.5)= 4	Fungsi ini melakukan pembulatan ke bilangan bulat terdekat yang nilainya lebih besar dari x.
conj(x) contoh : conj(4+8i) = 4-8i conj(4-8i) = 4+8i	Fungsi ini menghasilkan conjugate dari suatu bilanagn kompleks. Conj(x) = real(x)-i*imag(x)
exp(x)	Fungsi ini menghasilkan nilai eksponen dari bilangan x
fix(x)	Fungsi ini menhasilkan bagian bulat dari suatu bilangan.
floor(x) contoh : floor(4.5) = 4 floor(-3.2)=-4	Fungsi ini menghasilkan bilangan bulat terdekat dari x yang nilainya kurang dari x.
imag(x) contoh : imag(3+4i) = 4 imag(4-7i) = -7	Fungsi ini menghasilkan bagian imajiner dari suatu bilanagn kompleks
log(x)	Fungsi ini menghasilkan logaritma alami suatu bilanagn (ln x)
log10(x) contoh : log10(100) = 2 log10(1000)= 3	Fungsi ini menghasilkan bilanagan dari suatu bilangan
rem(x,y)	Fungsi ini menghasilkan sisa dari x/y
round(x) contoh round(4.3) = 4 round(4.7) = 5	Fungsi ini menghasilkan bilangan bulat yang merupakan pembulatan terhadap suatu bilanagn
sign(x) contoh : sign(-5)=-1 sign(0) = 0 sign(7) = 1	Fungsi ini menghasilkan bilanagn berupa : 1 jika x bernilai lebih dari nol 0 jika bernilaia nol -1 jika x bernilai kurang dari 0
sqrt(x) sqrt(144)=12	Fungsi ini menghasilkan akar kuadrat dari Bilangan x

`gcd(x,y)`
`contoh :`
`gcd(60,12)=6`
	Fungsi ini menghasilkan bilanagn yang merupakan faktor persekutuan terbesar dari bilanagn x dan y

Tabel 12. Beberapa Fungsi Matematika yang tersedia dalam MATLAB

Fungsi Trigonometri dan Hiperbolik	Keterangan
`sin (x)`	Fungsi ini menghasilkan nilai sinus dari x. Dalam hal ini x merupakan argumen inputan berupa sudut dalam satuan radian.
`cos (x)`	Fungsi ini menghasilkan nilai cosinus dari x. Dalam hal ini x merupakan argumen inputan berupa sudut dalam satuan radian
`tan (x)`	Fungsi ini menghasilkan nilai tangent dari x. Dalam hal ini x merupakan argumen inputan berupa sudut dalam satuan radian
`asin(x)`	Fungsi ini menghasilkan inversi sinus dari x, dalam hal ini argument inputan x berupa nilai yang berkisar antara -1 dan 1. Sedangkan hasilnya berkisar antara $-\frac{\pi}{2}$ dan $\frac{\pi}{2}$ dalam satuan radian.
`acos(x)`	Fungsi ini menghasilkan inversi cossinus dari x, dalam hal ini argument inputan x berupa nilai yang berkisar antara -1 dan 1. Sedangkan hasilnya berkisar antara $-\frac{\pi}{2}$ dan $\frac{\pi}{2}$ dalam satuan radian.
`atan(x)`	Fungsi ini menghasilkan inversi tangent dari x, dalam hal ini x berupa nilai yang berkisar antara -1 dan 1. Sedangkan hasilnya berkisar antara $\mathbf{0}$ dan $\boldsymbol{\pi}$ dalam satuan radian.
`atan2(x,y)`	Fungsi ini menghasilkan inversi sinus dari $\frac{y}{x}$, dalam hal ini x berupa nilai yang berkisar antara -1 dan 1. Sedangkan hasilnya berkisar antara $-\frac{\pi}{2}$ dan $\frac{\pi}{2}$ dalam satuan radian.

`sinh(x)`	Fungsi ini menghasilkan sinus hiperbolik dari x yaitu $\frac{e^x - e^{-x}}{2}$
`cosh(h)`	Fungsi ini menghasilkan cosinus hiperbolik dari x yaitu $\frac{e^x - e^{-x}}{2}$
`tanh(x)`	Fungsi ini menghasilkan tangent hiperbolik x.
`asinh(x)`	Fungsi ini menghasilkan inversi sinus hiperbolik dari x
`acosh(x)`	Fungsi ini menghasilkan inversi cosinus hiperbolik dari x
`atanh(x)`	Fungsi ini menghasilkan inversi tangent hiperbolik dari x
Perlu diketahui, hubungan antara sudut dalam satua derajat dan radian adalah $\boldsymbol{180^\circ = \pi\, radian}$	

Tabel 13. Fungsi Trigonometri yang tersedia di dalam MATLAB

Contoh Soal 2: Persamaan fungsi :Tentukan nilai fungsi berikut

$$M = 4x^2 + 3y + 10$$

$$N = e^{(2x)+x}$$

$$O = \sqrt{\frac{1}{x+y} + \frac{1}{t+z}}$$

$$P = 4e^{-\frac{x}{2}}\sin(\pi x)$$

dengan nilai-nilai $t = 15; x = 20; y = 25; z = 3.6$

```
>> %Penyelesaian Persamaan Matematika
>> %Definisikan variabel dan nilainya
>> t = 15;
>> x = 20;
>> y = 25;
>> z = 3.6;
>> %Dengan menggunakan konsep prioritas operator
>> M = 4*x^2+3*y+10
M =       1685
>> N = exp((2*x)+x)
N =   1.1420e+26
>> O = sqrt((1/(x+y))+(1/(t+z)))
O =    0.2757
>> P = 4*exp(-x/2)*sin(pi*x)
P =
  -4.4479e-19
```

Catatan : Teks-teks perintah yang dituliskan pada command window dapat di copy paste pada Ms.office Word.

Contoh Soal 3: Penentuan volume silinder berlubang Sebuah silinder mempunyai diameter luar 6.3125 dan diameter dalam 5.762 , tentukan volumenya bila diketahui rumus untuk mencari volumenya : $V = \frac{4}{3}\pi\left(RE^3 - RI^3\right)$, Dimana RE adalah diameter luar dan RI adalah diameter dalam

Penyelesaian :

```
re=6.3125;ri=5.762;
v=4/3*pi*(re^3-ri^3);
disp(['Volume = ',num2str(v)])
Volume = 252.3169
```

D. Mengenal Metodologi Penyelesaian Masalah

Dalam menyelesaikan suatu masalah diperlukan suatu metodologi. Pada dasarnya, metodologi untuk menyelesaikan suatu masalah mencakup 5 langkah seperti berikut :

- Mengidentifikasi masalah dengan jelas
- Menjabarkan output/input
- Penentuan Algoritma untuk menyelesaikan masalah.
- Menuliskan kode MATLAB untuk menyelesaikan masalah.
- Memuji dan memperbaiki kode MATLAB sehingga diperoleh hasil yang benar untuk berbagai keadaan masukan.

Pada BAB 7 kita akan melihat pengimplementasian teori-teori praktis dengan menggunakan MATLAB yang berkaitan dengan komputasi numerik. Penerapan metodologi dalam penyelesaian masalah, seperti masalah penyelesaian Sistem Persamaan Linear, penentuan akar persamaan tak linear, masalah interpolasi, masalah diferensiasi dan maslah integrasi numerik.

E. Rangkuman

- M-file merupakan salah satu fasilitas MATLAB yang dibuat pada jendela editor untuk menutupi kekurangan pada Command Window. Command Window sebagai jendela perintah adalah ruang kerja pada MATLAB untuk mengeksekusi perintah berbasis baris, sehingga terbatas untuk perintah perintah yang majemuk atau bertingkat. Dengan adanya M-File, kita dapat menuliskan beragam pendefinisian variabel, penggunaan built function, penggunaan fungsi-fungsi yang dapat dieksekusi secara simultan, sistematis dan fleksibel. M-file yang telah dibuat juga dapat diedit kembali tanpa harus menuliskan perintah-perintah yang lainnya.
- Melalui fasilitas input pada jendela editor, pengguna dapat merekayasa masukan-masukan dan memberi keterangan-keterangan penjelas untuk data-data yang akan dijadikan sebagai data variabel. Dengan ini, pengguna dapat lebih mudah menjalankan perintah dan mengeksekusi secara simultan sesuai dengan kumpulan perintah yang telah disusun pada jendela editor.
- Didalam M-File , luaran-luaran dapat diatur sedemikian rupa dengan menggunakan format luaran melalui perintah **fprintf .**
- Dalam menyimpan M-file yang perlu untuk diperhatikan adalah, kriteria pemberian nama file ketika disimpan. Seperti halnya pada pemberian nama variabel, pemberian nama M-File juga memiliki kaidah, sehingga nama M-Filenya legal dan dapat dieksekusi.
- Dalam menjalankan M-File, pengguna harus memastikan bahwa M-file yang digunakan sedang aktif pada Current Dirrectory.
- Dalam menyusun script program dalam M-File, kita dapat mengidentifikasi kesalahan-kesalahan dalam program. Perlu untuk dipahami kembali bahwa dalam menyusun satu berkas program dimungkinkan oleh kesalahan dalam penulisan program, dan juga kesalahan yang diakibatkan oleh logika dalam algoritma yang dikembangkan.

F. Latihan

Latihan 2. 1: Gunakan Jendela Editor untuk membuat satu M-file yang digunakan untuk menginput Biodata yang terdiri atas

- Nama
- NIM
- Alamat
- Tanggal lahir
- Nomor HP

Latihan 2. 2: Gunakan jendela editor untuk membuat satu M-File yang dapat digunakan untuk menghitung kecepatan rata-rata dengan :

- Memberikan Inputan data jarak dan waktu
- Mengeluarkan hasil kecepatan.

Latihan 2. 3: Jelaskan kegunaan dari penambahan komentar pada pembuatan program di jendela editor

Latihan 2. 4: Gunakan Jendela editor untuk menghitung :

- Volume dan Luas Permukaan Balok dengan memasukkan p, l, t
- Volume Bola, Luas Permukaan Bola dengan inputan jari-jari bola

BAB 3. PENGOLAHAN VEKTOR-MATRIKS

Kemampuan akhir yang diharapkan

- Mahasiswa mampu memahami esensi dari matriks dalam Matematika Komputasi
- Mengeksplorasi pembuatan vektor dan matriks
- Mahasiswa mampu menggunakan fungsi-fungsi untuk membangkitkan data-data vektor dan matriks
- Mahasiswa mampu menggunakan fungsi-fungsi MATLAB dalam mengoperasikan vektor dan Matriks
- Mahasiswa mampu melakukan pengolahan data statistika dengan mengolah data vektor ataupun data matriks.
- Mengurai Hubungan antara dua garis
- Mahasiswa mampu menginterpretasi suatu matriks sebagai representasi fungsi yang dapat disajikan dalam bentuk grafik dalam sistem Koordinat

A. Matriks Dalam Matematika Komputasi

Dalam ilmu komputer, larik dikenal sebagai suatu tipe data terstruktur yang dapat menyimpan banyak data dengan suatu nama atau variabel yang sama serta bertipe data sama pula. Untuk membedakan satu dengan yang lainnya, elemen-elemen larik mempunyai indeks.Setiap elemen diakses langsung melalui indeksnya. Salah satu tipe data yang termuat dalam larik yaitu jenis data numerik. Oleh karena itu, larik atau array biasanya direpresentasikan oleh vektor dan matriks.

Selanjutya vektor dan matriks merupakan topik dalam ilmu matematika yang mempunyai peranan penting dalam kehidupan sehari-hari. Kajian mengenai vektor dan matriks dapat kita temui hampir pada seluruh topik pembelajaran matematika. Secara mendasar pengolahan vektor dan matriks dapat dilakukan secara manual. Dengan menggunakan formula-formula tertentu dalam pengolahannnya. Namun dalam ukuran

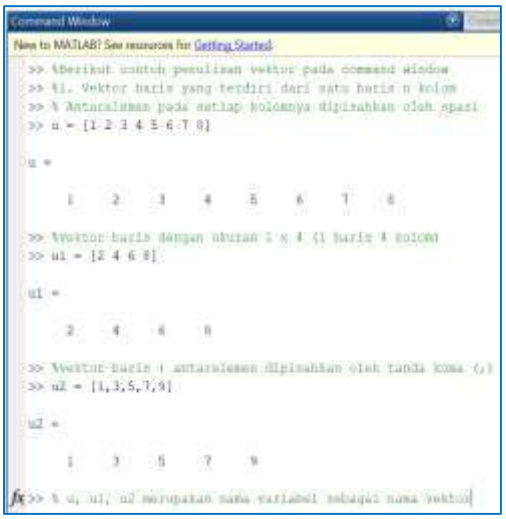
yang lebih besar dibutuhkan perangkat komputasi yang dapat melakukan pengolahan vektor dan Matriks secara tcepat dan tepat.

Salah satu perangkat komputasi yang dapat digunakan dalam pengolahan vektor dan Matriks yaitu MATLAB. Melalui MATLAB, vektor dan matriks tidak hanya dapat diolah dengan cepat tetapi juga MATLAB sangat cocok dalam menvisualisasikannnya dalam berbagai jenis grafik. Pada pertemuan ini akan diperkenalkan beberapa fungsi pada MATLAB dalam membangkitkan dan mengolah array dalam hal ini vektor dan Matriks.

B. Vektor

Dalam kajian aljabar geometri, besaran fisik adalah segala sesutu yang dapat diukur. Besaran fisik dibedakan atas skalar dan vektor, Skalar adalah besraan fisik yang hanya memiliki besar (magnitude), misalnya : panjang, massa, waktu, suhu, dan energi. Dari penjelasan ini, skalar dapat dipadankan dengan bilanagna, dan penghitungan skalar terkait operasi bilanagn. Sedangkan vektor adalah besaran fisik yang mengandung besar sekaligus arah, misalnya :gaya, kecepatan, percepatan, medan listrik, medan magnet. Dari pengertian ini, jelas bahwa setiap vektor pasti mengandung skalar. Secara geometris, vektor dapat direpresentasika.n sebagai ruas garis berarah, dengan panjang garis merepresentasikan skalarnya.

Seperti pada perintah-perintah sebelumnya, bahwa perintah-perintah dapat dituliskan pada command window, ataupun jendela editor. Berikut pendefinisian vektor yang dibuat pada command window. Dalam mendefinisikan suatu vektor pada MATLAB dapat dilakukan dengan cara membuat nama variabel dari vektor sebagai nama vektor dan menuliskan elemen-elemen vektor. Gambar di bawah ini menunjukkan pendefinisian nama vektor dan elemen-elemen vektor dalam bentuk vektor baris. Tanda [ ] digunakan untuk menyetakan larik, tanda spasi digunakan untuk memisahkan elemen antar kolom.



Gambar 3. 1. Cara membuat vektor pada Command Window

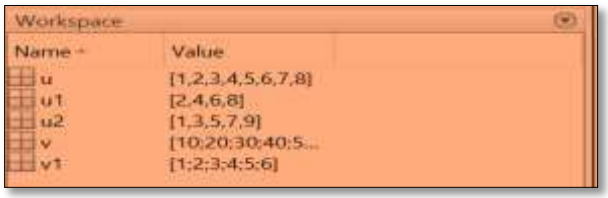
Nama dan nilai-nilai elemen vektor dapat ditinjau pada workspace

Gambar 3. 2. Rekaman Data Pada Workspace

Demikian halnya pada pendefinisian vektor kolom. Pada command window vektor kolom dapat dibuat dengan menentapkan nama vektor dan menuliskan elemen elemen vektor. Tanda [ ] digunakan untuk menyatakan larik dan tanda (;) digunakan untuk memisahkan elemen antar baris.

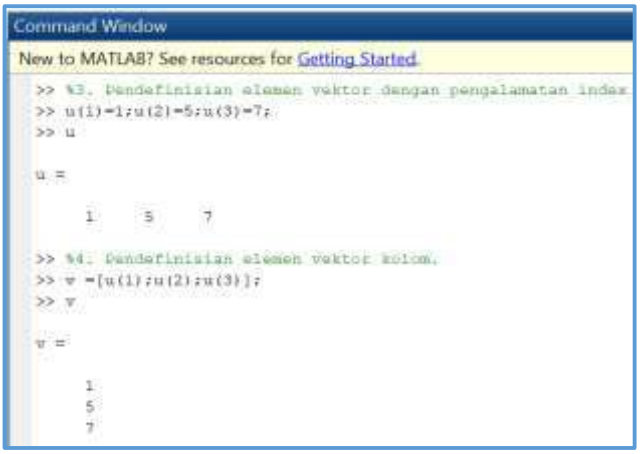
Gambar 3. 3. Pembuatan Vektor Baris

Nama-nama vektor baris dan vektor kolom serta elemen-elemen antar vektor dapat dipantau pada worrkspace.



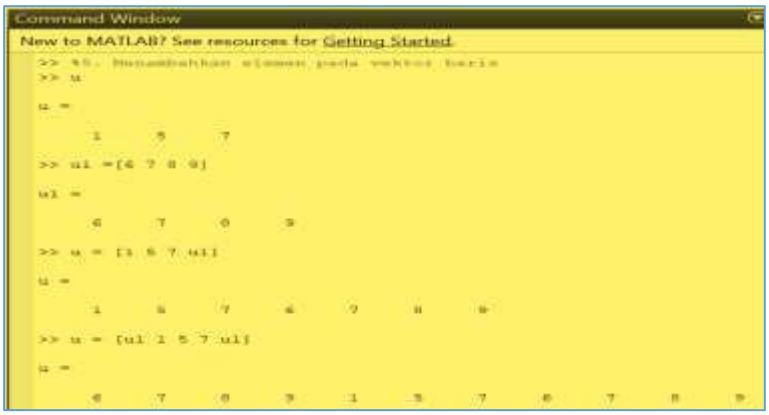
Gambar 3. 4. Perekaman data melalui workspace

Selanjutnya pembuatan vektor juga dapat dilakukan melalui pengalamatan indeks. Pada contoh di bawah ini menunjukkan pendefinisian elemen-elemen vektor, dilakukan pada masing-masing indeks. Selain itu gambar dibawah ini juga menunjukkan pendefinisian vektor kolom dengan mengambil nilai-nilai pada vektor baris yang telah didefinisikan sebelumnya. Caranya sama seperti pada cara sebelumnya dengan menggunakan tanda [ ] dan antar elemen dipisahkan dengan tanda (;).



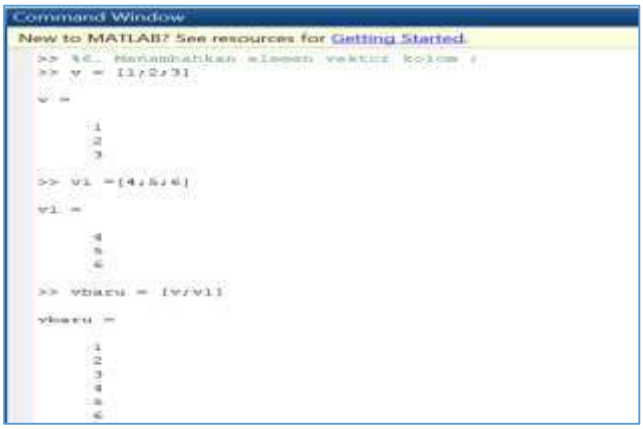
Gambar 3. 5. Pembuatan vektor dengan memberi indeks pada elemen-elemennya

Dalam membuat vektor, baik vektor baris maupun vektor kolom juga dapat dilakukan penambahan elemen baru. Gambar di bawah ini menunjukkan adanya vektor yang didefinisikan, kemudian elemennya ditambahkan dengan elemen-elemen baru pada posisi yang dikehendaki. (awal, tengah atau akhir).



Gambar 3. 6. Penambahan elemn dengan menggunakan indeks

Penambahan elemen vektor tidak hanya dapat dilakukan pada vektor baris, tapi juga dapat dilakukan pada vektor kolom. Gambar di bawah ini menunjukkan contoh penambahan elemen pada vektor kolom. Dalam menambahkan elemen vektor pada vektor kolom dilakukan dengan cara yang sama yaitu menambahkan tanda (;) pada kolom yang dikehendaki.



Gambar 3. 7. Penambahan elemen vektor kolom

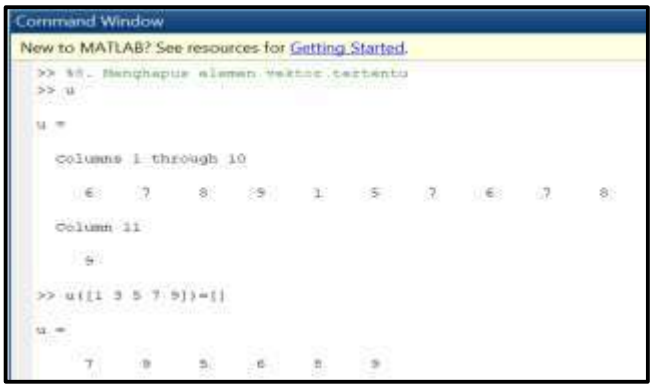
Dengan adanya vektor yang telah didefinisikan bersama dengan elemen-elemennya, maka kita dapat mengakses berdasarkan indeks-indeksnya.Indeks disini menunjukkan letak elemen vektor berada. Gambar di bawah ini menunjukan, bagaimana mengakses nilai elemen vektor berdasarkan indeks-indeksnya.

Gambar 3. 8. Mengakses elemen vektor

Selain mengakses elemen-elemen vektor pada setiap indeksnya, kita juga dapat mengakses nilai-nilai vektor berdasarkan indeks yang dikehendaki. Gambar di bawah ini menunjukkan cara dalam mengakses elemen vektor pada indeks yang dikehendaki. Didefinisikan vektor u dengan 11 elemen. Selanjutnya didefinisikan vektor baris baru dengan nama x dengan mengakses elemen vektor u pada indeks ke 2, 4 , 6, 8, 10.

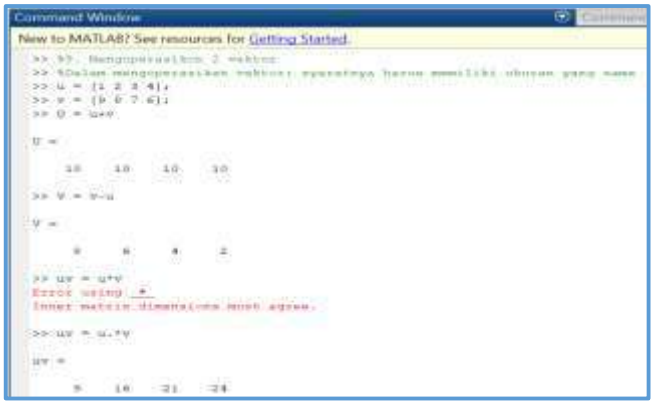
Gambar 3. 9. Mengakses Elemen Vektor Setiap Indeksnya

Selain menambah, mengakses elemen vektor, kita juga dapat melakukan penghapusan elemen vektor. Menghapus elemen vektor dapat dilakukan dengan mengganti elemen vektor pada indeks-indeks tertentu dengan elemen kosong. Gambar di bawah ini menunjukkan adanya vektor u yang didefinsikan dengan 11 indeks, selanjutnya pada indeks ke-1, 3, 5, 7 dan 9 diganti dengan elemen kosong [ ].



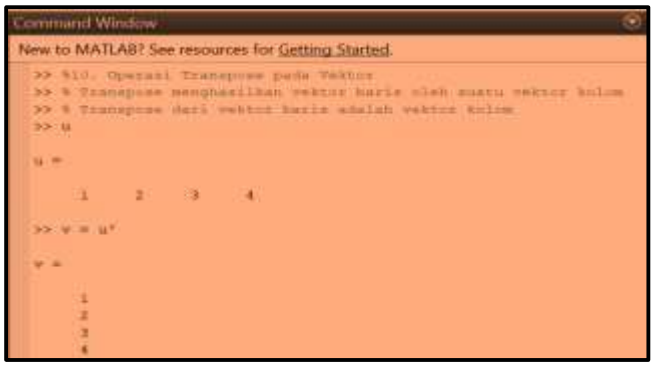
Gambar 3. 10. Mengahpus Elemen Vektor

Hal umum yang paling penting dalam kajian vektor adalah melakukan pengoperasian antar vektor. Operasi antar vektor yang dapat diterapkan adalah operasi penjumlahan (+), pengurangan (x), pembagian (/) ataupun perkalian antar elemen (.*). Tanda perkalian (*) tidak dapat digunakan pada perkalian dua vektor baris atau dua vektor kolom karena tanda (*) merupakan perkalian yang mensyaratkan pemenuhan kesesuaian ukuran.



Gambar 3. 11. Mengoperasikan Vektor

Operasi yang lain yang dapat dilakukan pada vektor baris atau kolom adalah menentukan Transpose. Transpose dari suatu vektor menyatakan posisi balikan (baris jadi kolom, kolom jadi baris). Gambar di bawah ini menunjukkan operasi transpose dari suatu vektor baris yang menghasilkan vektor kolom. Operasi Transpose dinyatakan dengan tanda (‘).



Gambar 3. 12. Operasi Transpose Vektor

C. Matriks

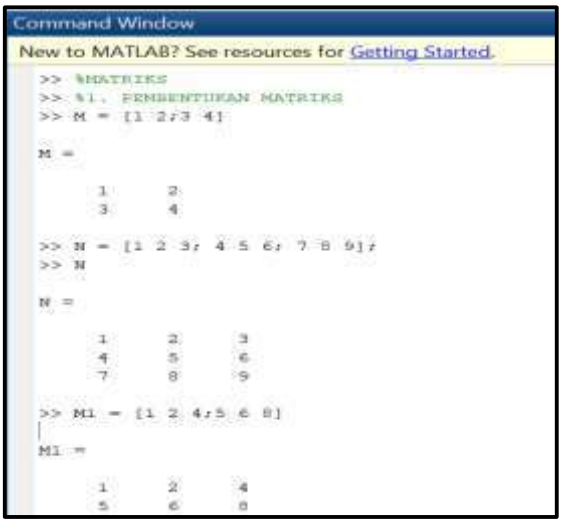
Matriks merupakan basis utama yang menjadi objek pengolahan dari MATLAB, seperti halnya array matriks juga didefenisikan elemen demi elemen. Pembentukan vektor pada sub bahasan sebelumnya menagantarkan kita untuk memahami bagaimana pembentukan matriks pada MATLAB. Matriks dalam kajian matematis dipandang sebagai himpunan skalar yang disusun menurut baris dan kolom. Notasi Matriks A_{ij} dimana a_{ij} adalah elemen pada baris ke i kolom ke j

$$A_{ij} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2} \\ . & . & \dots & . \\ a_{m1} & a_{n2} & \dots & a_{mn} \end{pmatrix}$$

Matriks dalam kajian matematis dipandang sebagai himpunan skalar yang disusun menurut baris dan kolom. Konsep pembentukan Matriks pada command window dilakukan dengan mendefinisikan vektor vektor barisnya setiap baris. Dalam hal ini vektor baris setiap barisnya dipisahkan dengan tanda (;).

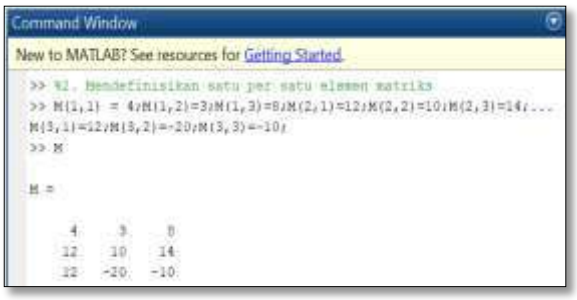
Gambar di bawah ini menunjukkan pembentukan matriks M dengan ukuran 2 x 2. Cara pembentukannya tetap di buka dan di tutup oleh tanda [ ], selanjutnya dibuat vektor barisnya denagn ukuran yang seragam pada

baris-baris sebelumnya. Setiap elemen pada setiap barisnya dipisahkan oleh spasi, selanjutnya setiap baris yang dibuat dipisahkan oleh tanda (;).



Gambar 3. 13. Pembentukan matriks standar melalui input elemen

Cara yang lain yang dapat dilakukan dalam pembentukan matriks yaitu dengan mendefinisikan elemen-elemen matriks pada setiap indeksnya. Setiap eelemen matriks menempati indeks **M(baris, kolom)**. Gambar di bawah ini menunjukkan pembentukan matriks M pada setiap indeksnya.



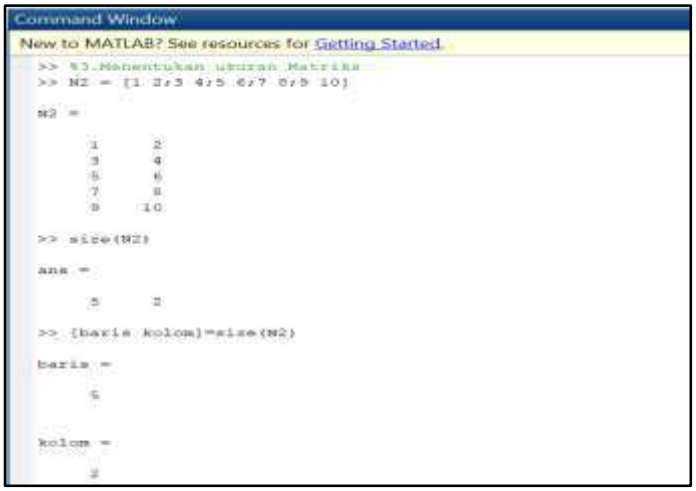
Gambar 3. 14. Membuat matriks dengan mendefinisikan nilai setiap elemennya

Hal utama yang menjadi catatan dalam pembentukan matriks yaitu, banyak elemen pada setiap barisnya harus seragam. Atau banyaknya elemen pada stiap kolomnya harus seragam.

❖ Ukuran Matriks

Setiap matriks, ataupun vektor dapat diidentifikasi ukuran atau dimensinya. Perintah yang digunakan dalam menentukan ukuran matriks yaitu dengan menggunakan perintah size. Secara standar pada MATLAB, akan mengeluarkan ukuran baris dan kolomnya.

Gambar di bawah ini dengan mendefinisikan matriks N2 yang selanjutnya diberi perintah size(N2) menghasilkan 5 2, hal ini menunjukkan ukuran baris pada matriks N2 yaitu 5 bari dikali 2 kolom.



Gambar 3. 15. Fungsi size dalam mengidentifikasi ukuran matriks

❖ Operasi Pada Matriks

Penjumlahan pada Matriks (berlaku untuk matriks –matriks yang berukuran sama). Jika A = (a_{ij}) dan B = (b_{ij}) , matriks yang berukuran sama , maka A + B adalah suatu matriks C = (c_{ij}) , di mana $c_{ij} = a_{ij} + b_{ij}$ untuk setiap i dan j.

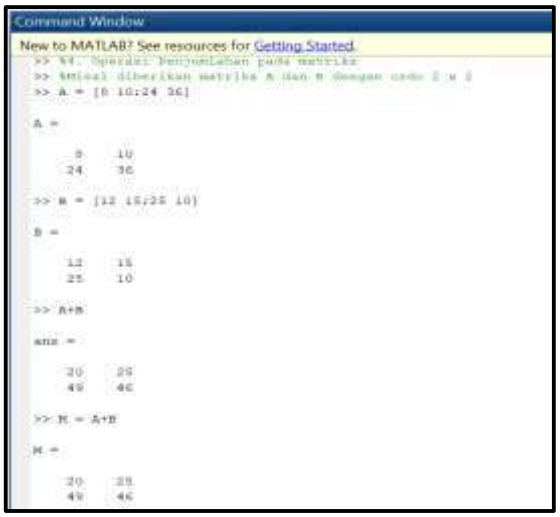
Contoh Soal 4: Misal diberikan dua Matriks $A = \begin{bmatrix} 8 & 10 \\ 24 & 36 \end{bmatrix}$ dan $B = \begin{bmatrix} 12 & 15 \\ 25 & 10 \end{bmatrix}$

Tentukan A+B !

Jawab : Penyelesaian Secara Manual

$$A + B = \begin{pmatrix} 8 & 10 \\ 24 & 36 \end{pmatrix} + \begin{pmatrix} 12 & 15 \\ 25 & 10 \end{pmatrix} = \begin{pmatrix} 8+12 & 10+15 \\ 24+25 & 36+10 \end{pmatrix} = \begin{pmatrix} 20 & 25 \\ 49 & 46 \end{pmatrix}$$

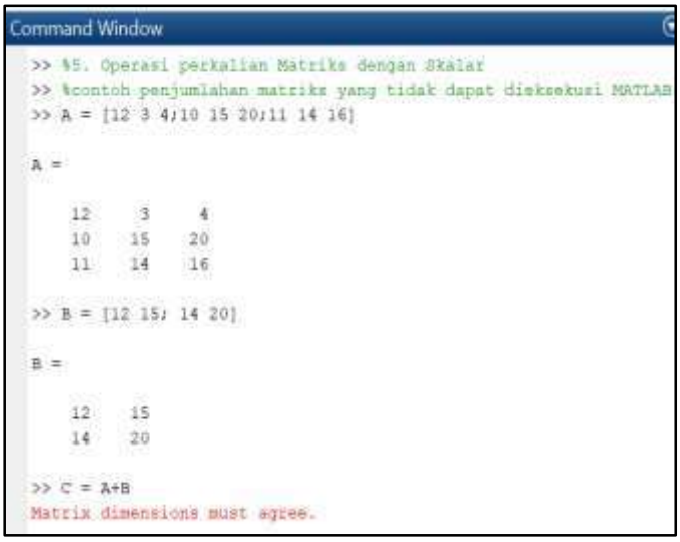
Namun dengan Fasilitas MATLAB pengoperasian MATLAB menjadi terasa lebih mudah dalam menentukan hasil dari jumlah dua matriks, seperti pada ambar di bawah ini.



Gambar 3. 16. Penjumlahan Matriks

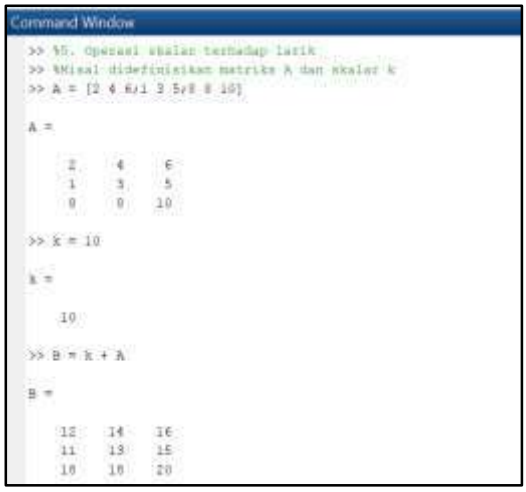
Konsep penjumlahan Matriks pada MATLAB menerapkan konsep aturan penjumlahan matriks, yang mensyaratkan ukuran matriks yang dijumlahkan harus sama, karena dalam prosesnya penjumlahan matriks mengoperasikan penjumlahan elemen-elemen matriks pada indeks yang bersesuaian.

Gambar command window di bawah ini menunjukkan kesalahan penjumlahan matriks yang diseBabkan oleh perbedaan dimensi atau ukuran matriks.



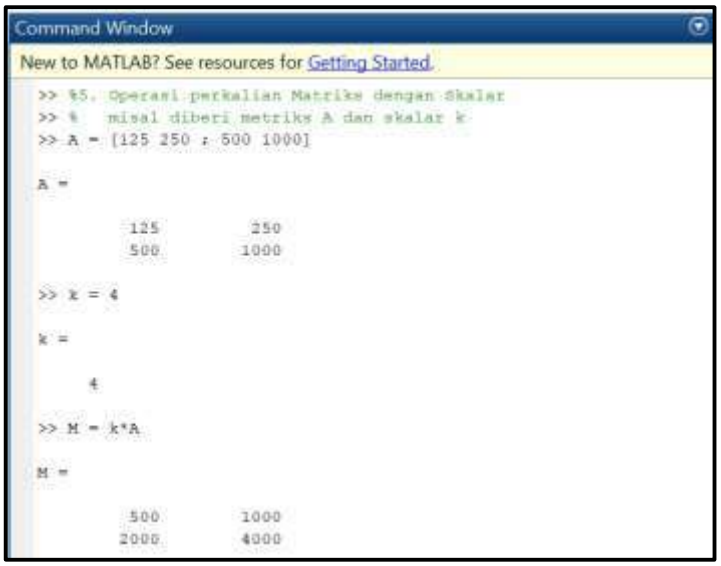
Gambar 3. 17. Kesalahan dalam penjumlahan matriks karena perbedaan dimensi

MATLAB menyediakan cara yang sangat mudah dalam mengoperasikan nilai skalar terhadap suatu array baik vektor maupun matriks. Sebagai contoh, untuk menjumlahkan setiap elemen matriks dengan skalar tertentu dapat dilakukan seperti gambar di bawah ini.



Gambar 3. 18. Penjumlahan elemen matriks dengan suatu skalar

Selain operasi penjumlahan skalar terhadap matriks , juga dapat dilakukan operasi perkalian skalar terhadap matriks. Jika λ suatu scalar (bilangan) dan A = $\left(a_{ij}\right)$ maka matriks λA = (λa_{ij}) .Gambar di bawah ini menunjukkan perkalian antara skalar dengan matriks. Gambar di bawah ini menunjukkan penjumlahan skalar dengan matriks.



Gambar 3. 19. Perkalian matriks dengan skalar

❖ Perkalian Antar Matriks

Pada umumnya perkalian Matriks tidak komutatif terhadap operasi perkalian : AB ≠BA.

Syarat Perkalian Matriks :

Banyaknya kolom matriks pertama = banyaknya baris matriks kedua.

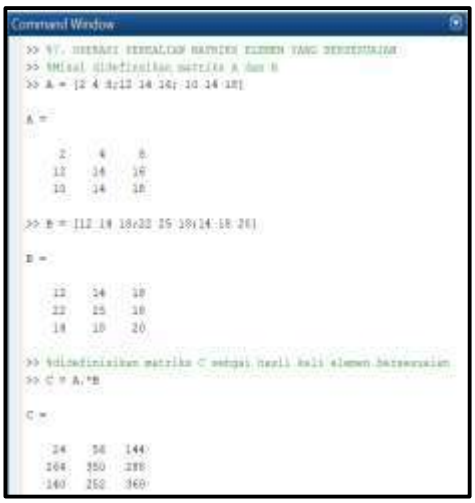
Definisi :

Misal $A = (a_{ij})$ *berukuran* ($m \times n$) *dan* $B = (b_{ij})$ *berukuran* ($n \times p$) . *Maka perkalian AB adalah suatu matriks* $C = (c_{ij})$ *berukuran* ($m \times p$) *di mana* $c_{ij} = a_{i1} b_{1j} + a_{i2} b_{2j} + \ldots + a_{in} b_{nj}$ *untuk setiap* $I = 1,2,\ldots,m$ *dan* $j = 1,2,\ldots,p$.

Gambar di bawah ini menunjukkan perkalian antara matriks A dengan ukuran (3 x 3) dengan matriks B dengan ukuran (3 x 2) menggunakan operator perkalian (*) yang selanjutnya didefinisikan sebagai matriks C (3 x 2) sebagai hasil kali dengan aturan kolom baris

Gambar 3. 20. Operasi Perkalian dua Matriks

Selain operasi perkalian yang mengalikan dua matriks dengan aturan perkalian kolom-baris. MATLAB juga menyediakan perkalian matriks yang mengalikan dua matriks dengan elemen-elemen yang mempunyai indeks yang bersesuaian. Gambar di bawah ini menunjukkan hasil kali dua matriks yang tidak mengikuti aturan perkalian antara matriks. Aturan yang digunakan yaitu perkalian antara elemen yang bersesuaian.



Gambar 3. 21. Perkalian Matriks anatar Elemen yang Bersesuaian

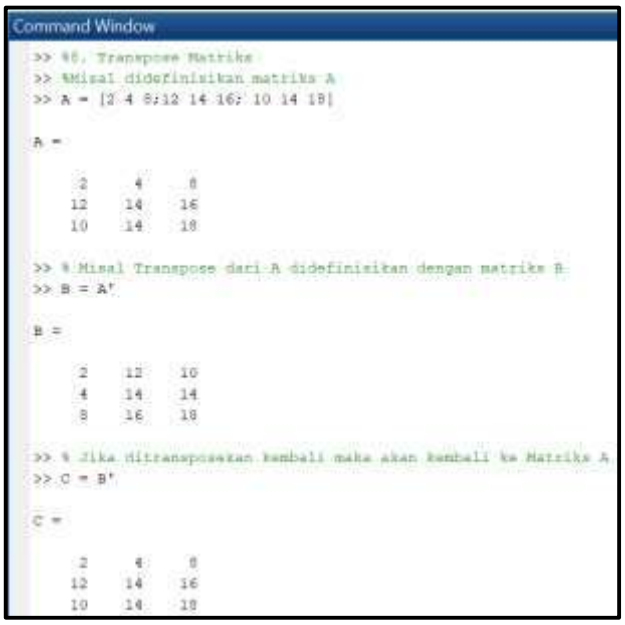
Namun untuk perkalian elemen yang bersesuaian mensyaratkan ukuran matriks pertama dengan ukuran matriks kedua harus sama. Perkalain matriks seperti ini, menggunakan operator (.*).

❖ Transpose dari suatu Matriks

Misal A = (a_{ij}) berukuran (m x n) maka transpose dari A adalah matriks A^T berukuran (nxm) maka $A^T = (a_{ji})$. Beberapa Sifat matriks transpose :

- ❖ $(A + B)^T = A^T + B^T$
- ❖ $(A^T)^T = A$
- ❖ $\lambda(A^T) = (\lambda A)^T$
- ❖ $(AB)^T = B^T A^T$

MATLAB menyediakan operator transpose yang disebut sebagai transpos dan dinotasikan dengan tanda (‘).



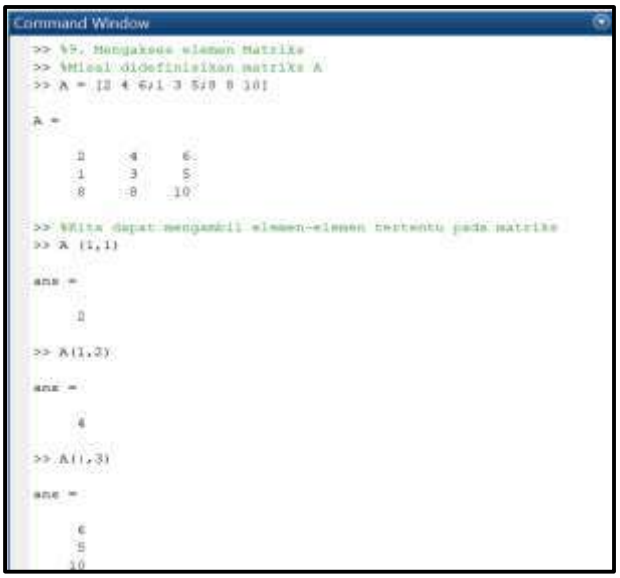
Gambar 3. 22. Transpose Matriks

Gambar (3.20) di atas menunjukkan transpose dari matriks A dengan ukuran 3 x 3. Transpose dari matriks A didefinisikan sebagai matriks B, yang jika diperhatikan baris pertama pada matriks B merupakan kolom pertama pada matriks A, baris kedua pada matriks B merupakan kolom kedua dari matriks A dst. Selanjutnya matriks B jika ditransposekan kembali makan akan kembali menjadi matriks A dlaam hal ini didefinisikan sebagai matriks Didalam MATLAB, transpose matriks menggunakan tanda (‘).

Seperti halnya pada transpose vektor, prinsip pada transpose dari suatu matriks juga membalik posisi baris menjadi posisi kolom.

❖ **Mengakses elemen-elemen matriks**

Seperti halnya pada vektor, pada Matriks kita juga dapat mengakses elemen-elemen tertentu pada suatu matriks. Gambar di bawah ini menunjukkan cara mengakses elemen-elemen tertentu pada indeks-indeks yang dikehendaki.

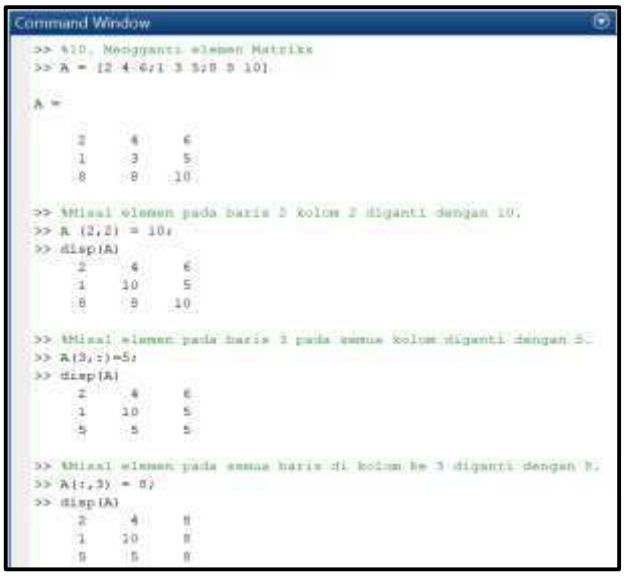


Gambar 3. 23. Mengakses elemen matriks berdasarkan indeks yang dikehendaki

Misal indeks (1,1) dalam hal ini baris pertama dan kolom pertama dari matriks A, baris pertama kolom kedua pada matriks A. Selanjutnya A(:,3), mengakses semua baris pada kolom ketiga dari matriks A.

❖ **Mengganti elemen matriks**

Selain mengambil kita juga dapat mengubah nilai dari elemen matriks pada indeks tertentu. Gambar di bawah ini menunjukkan bagaimana cara mengganti elemen matriks pada indeks tertentu dengan elemen baru.



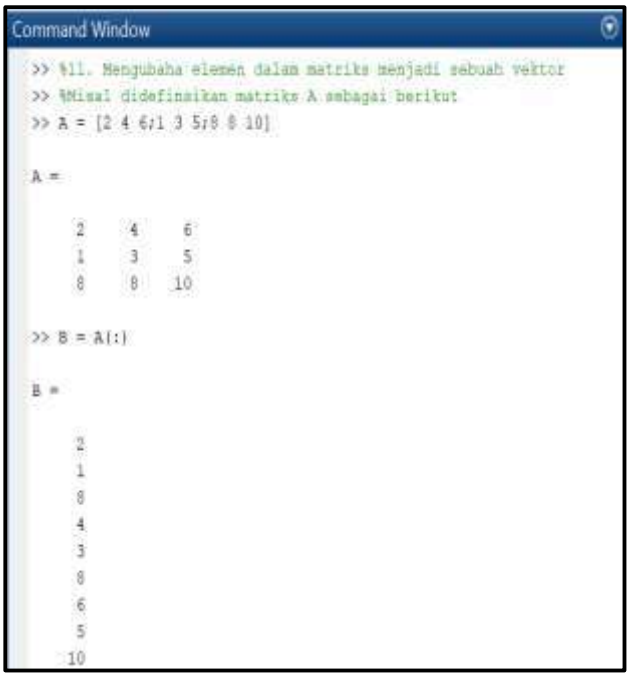
Gambar 3. 24. Mengganti elemen matriks

Tampak bahwa dengan matriks A yang didefinisikan, elemen pada baris ke-2 kolom ke-2 dapat diganti dengan nilai yang baru (10). Selain itu kita juga dapat memperbaharui elemen pada beberapa indeks tertentu dengan nilai yang seragam. Seperti pada baris ketiga untuk semua kolom, atau kolom ketiga untuk semua baris.

❖ **Mengakses Elemen Vektor**

Sebelumnya telah dijelaskan bahwa baik vektor maupun matriks, elemen-elemen nya menempati indeksnya masing-masing. Indeks pada vektor dinotasikan dengan A(i), indeks matriks dinotasikan dengan A(i,j) . Indeks pada vektor baris maupun vektor kolom, elemen-elemennnya terurut sesuai dengan indeksnya. Indeks pada vektor baris dibaca dari kiri ke kanan. Sedangkan pada vektor, indeksnya berjalan mulai dari atas ke bawah. Selanjutnya indeks vektor mengikuti posisi baris, kolom. Namun indeksnya secara berurutan dibaca dari atas sebelah kiri ke bawah pada kolom pertama dilanjutkan dengan baris pertama pada kolom kedua.

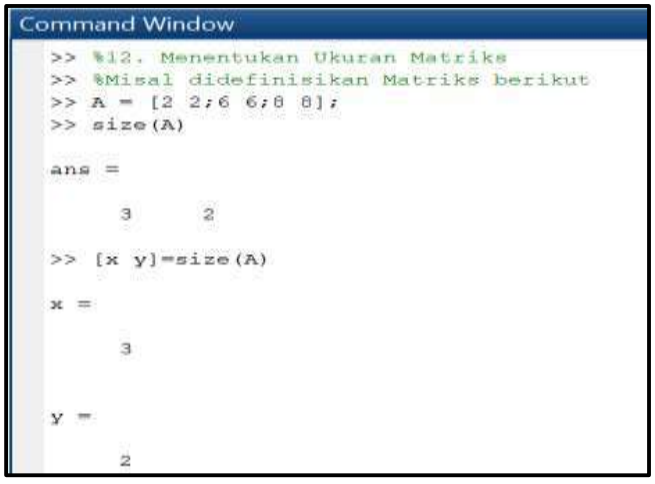
Dengan ini kita dapat mengakses elemen matriks tanpa menuliskan indeks baris, kolomnya. Gambar di bawah ini menunjukkan bagaimana suatu matriks selanjutnya dinyatakan dalam bentuk vektor kolom.



Gambar 3. 25. Mengubah elemen matriks menjadi vektor

❖ Mengidentifikasi Ukuran Matiks

Seperti pada vektor, pada matriks kita juga dapat memperoleh ukuran dari matiks dengan menggunakan perintah size.

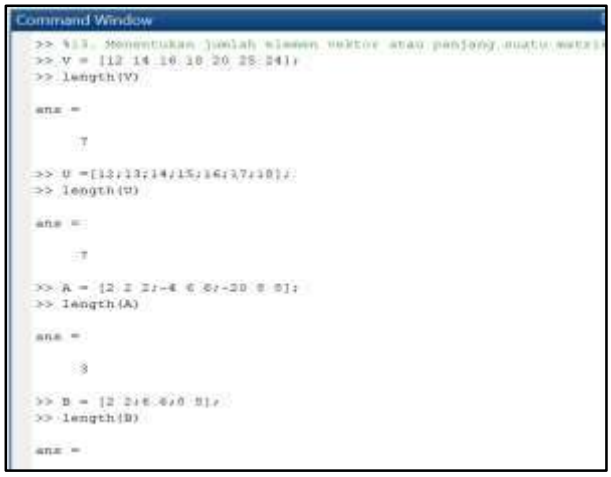


Gambar 3. 26. Mengidentifikasi ukuran matriks

Pembacaan indeks pada matriks dengan menggunakan fungsi size menghasilkan vektor baris yang elemen-elemennya adalah jumlah baris dan jumlah kolom.

❖ Mengidentifikasi Panjang Vektor

Selain fungsi size, kita juga dapat menggunakan fungsi **length** yang dapat digunakan untuk mengetahui jumlah elemen dalam suatu vektor. Namun jika fungsi **length** digunakan pada matriks yang berukuran m x n. Hasilnya berupa nilai terbesar diantara m dan n.



Gambar 3. 27. Menentukan panjang suatu vektor

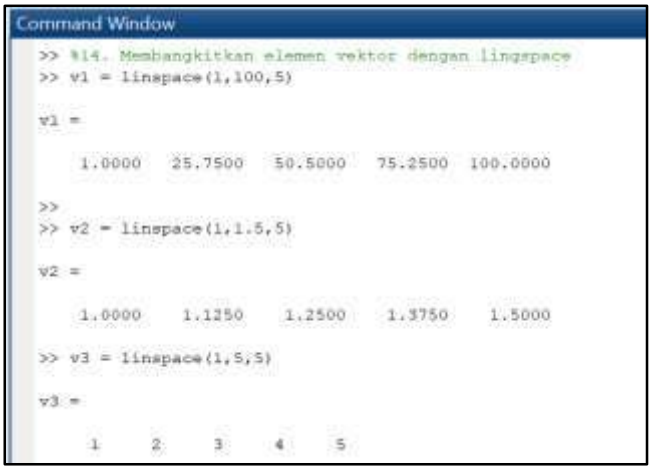
❖ Membangkitkan element Matriks

Berikut ini diperkenalkan beberapa fungsi pada MATLAB yang digunakan untuk membagkitkan elemen-elemen vektor diantaranya :

linspace(x,y,n)

Salah satu fungsi larik yang digunakan untuk memabangkitkan elemen-elemen vektor yang diawali dengan elemen x dan diakhiri dengan elemen y sebanyak n buah dengan selang antara elemen satu dengan berikutnya bernilai sama secara logaritmik.

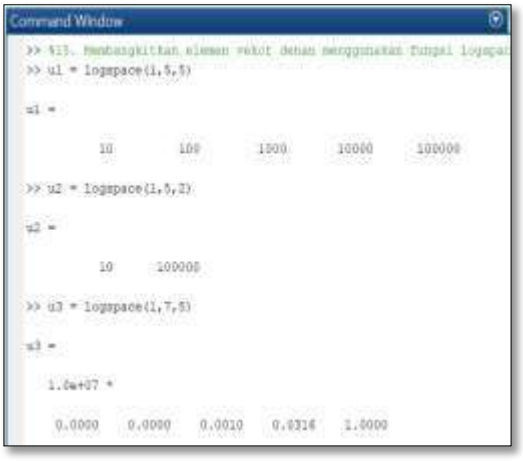
Gambar di bawah ini menunjukkan bahwa argumen pertama menemukan elemen pertama, argumen kedua menunjukkan elemen terakhir, sedangkan argumen ketiga menunjukkan banyaknya elemen vektor yang dihasilkan



Gambar 3. 28. Membangkitkan data vektor dengan menggunakan linspace

```
logspace(x,y,n)
```

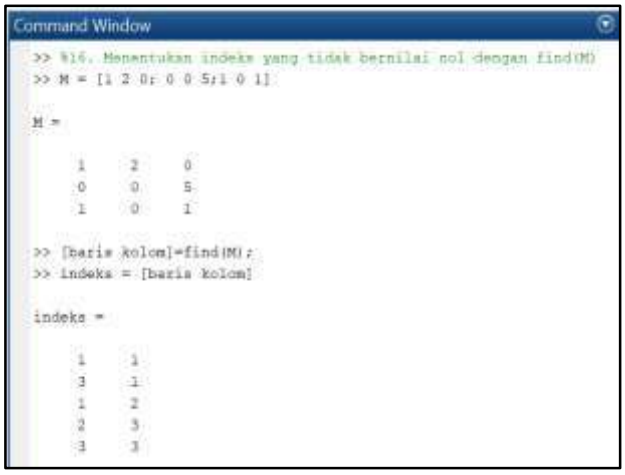
Salah satu fungsi larik yang digunakan untuk membangkitkan elemen-elemen vektor yang dimulai dari elemen pertama 10^x dan diakhiri dengan elemen 10^y dengan jumlah elemen satu dengan berikutnya bernilai sama secara logaritmik



Gambar 3. 29. Membangkitkan data vektor melalui penggunaan fungsi logspace

```
find(M)
```

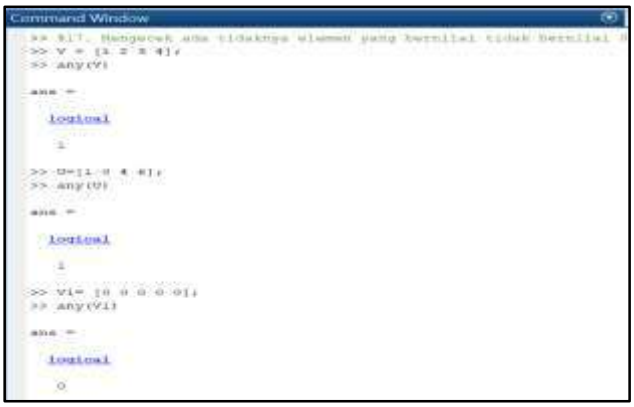
Salah satu fungsi yang tersedia dalam MATLAB digunakan untuk menghasilkan indeks untuk semua elemen yang tidak bernilai nol.



Gambar 3. 30. Penggunaan fungsi find (M) dalam menemukan indeks yang tidak bernilai nol

```
any(M)
```

Menghasilkan nilai benar (1) kalau ada elemen dalam larik (M) yang tidak bernilai 0 atau nilai salah (0) kalau semua elemen dalam M bernilai nol



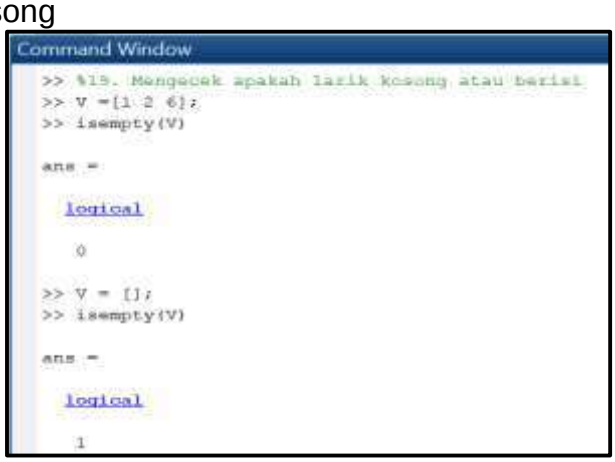
Gambar 3. 31. Penggunaan any(M) untuk mengecek ada atau tidak elemen nol

```
all(M)
```

Menghasilkan nilai benar (1) kalau semua elemen dalam larik M bernilai tidak nol.

`isempty(M)`

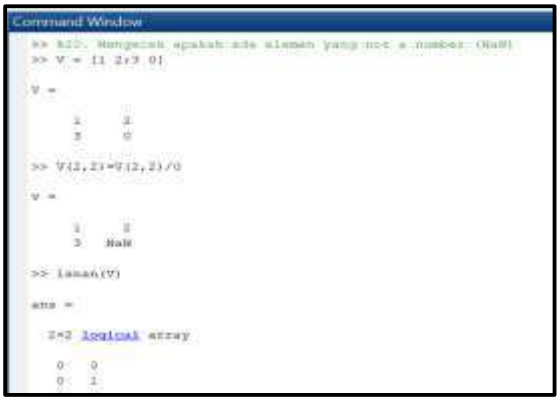
Menghasilkan nilai benar (1) kalau semua elemen dalam larik M adalah larik kosong



Gambar 3. 32. Mengecek apakah larik kosong atau tidak

`isnan(M)`

Suatu fungsi larik yang menghasilkan nilai benar (1) untuk setiap elemen yang bernilai NaN (Not a Number)



Gambar 3. 33. Mengecek adanya elemen Not a Number

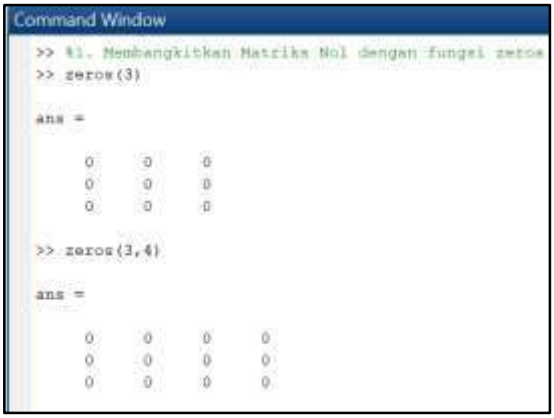
❖ **Pengenalan Matriks Khusus**

Pada tabel berikut diperkenalkan beberapa perintah dalam MATLAB ubtuk membangkitkan matriks –matriks khusus.

Matriks Nol

Matriks nol merupakan matriks yang setiap elemennnya bernilai nol. Pada MATLAB kita dapat membangkitkan matriks nol dengan menggunakan fungsi zeros(n) atau zeros(m,n).

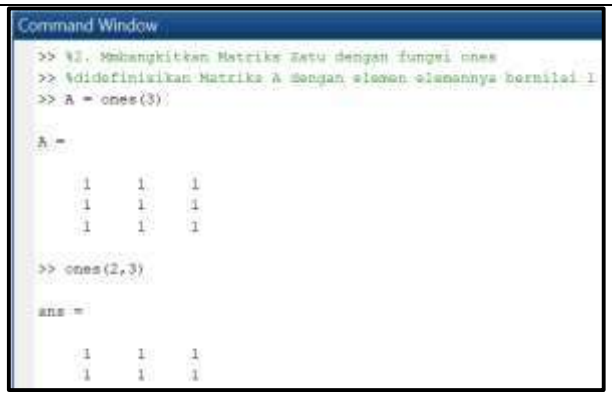
Gambar di bawah ini menunjukkan pembangkitan elemen-elemen matriks nol. Pada bagian pertama zeros (3), dimaksudkan membentuk matriks nol yang berukuran 3 x 3. Sedangkan, pada bagian kedua zeros(3,4) membentuk matriks nol berukuan 3 x4.



Gambar 3. 34. Penggunaan fungsdi zeros dalam membuat matriks nol

Matriks satu

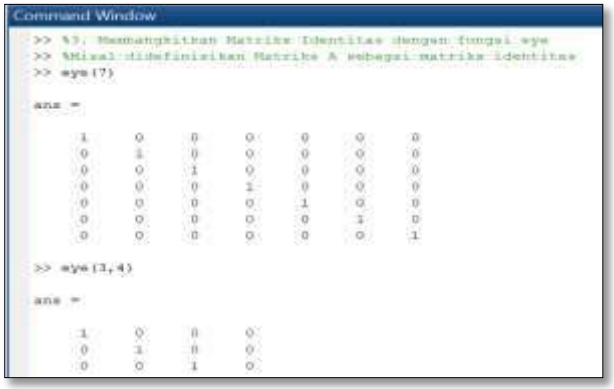
Matriks nsatu merupakan matriks yang setiap elemennnya bernilai satu. Pada MATLAB kita dapat membangkitkan matriks nol dengan menggunakan fungsi ones(n) atau ones(m,n). Gambar di bawah ini menunjukkan pembangkitan elemen-elemen matriks 1. Pada bagian pertama ones(3) membentuk matriks satu yang berukuran 3 x 3. Pada bagian kedua ones(2,3) membentuk matriks nol berukuan 2 x3.



Gambar 3. 35. Penggunaan fungsi ones untuk membangkitkan Matriks satu

Matriks Identitas

Matriks identitas merupakan matriks bujursangkar yang elemen-elemen pada diagonal utamanya bernilai 1, sedangkan elemen lainnya bernilai nol. Pada MATLAB kita dapat membangkitkan matriks identitas bujur sangkar dan Dengan menggunakan fungsi eye(n). Gambar di samping ini menunjukkan pembangkitan matriks identitas ukuran 7x7 dan 3x4.



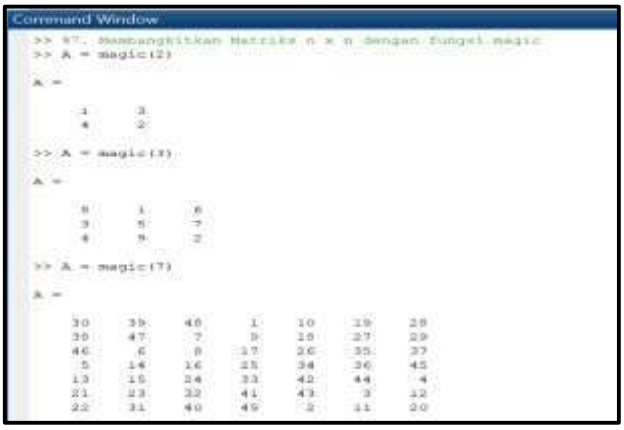
Gambar 3. 36. Fungsi eye dalam membangkitkan matriks identitas

Matiks Magic

Matriks magic merupakan matriks bujur sangkar (nxn) yang elelemen-elemennya berupa bilanagn bulat yang berkisar dari 1 hingga n^2.

Pada MATLAB kita dapat membangkitkan matriks magic dengan menggunakan fungsi magic(n) diamana n merupakan ukuran matriks dan n^2 merupakan nilai tertinggi dari elemen matriks yang dibangkitkan.

Pada gambar di bawah ini menunjukkan bagaimana membangkitkan matriks magic dengan nilai n=2, n= 3 dan n = 7.



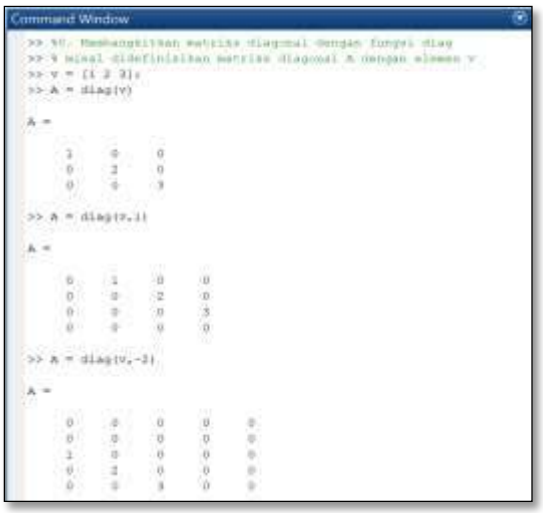
Gambar 3. 37. Membangkitkan elemen-elemen matriks yang kurang dari n^2

Matriks Diagonal

Matriks diagonal merupakan matriks bujursangkar yang unsur-unsurnya berada di garis diagonal utama dari matriks bukan nol dan unsur lainnya adalah nol. Pada MATLAB, matriks diagonal dapat dibangkitkan dengan menggunakan fungsi diag(v), diaman v merupakan vektor yang akan menjadi unsur-unsur pada diagonal utama dengan ukuran yang menyesuaikan dari vektor yang menjadi masukan.

Selanjutnya dapat pula menggunakan fungsi diag(v,r) dimana v merupakan argumen masukan dengan yang menjadi diagonal, sedangkan r merupakan argumen pegeseran diagonal utama.

Pada gambar di bawah ini menunjukkan penggunaanfungsi diag(v) dan diag(v,r).



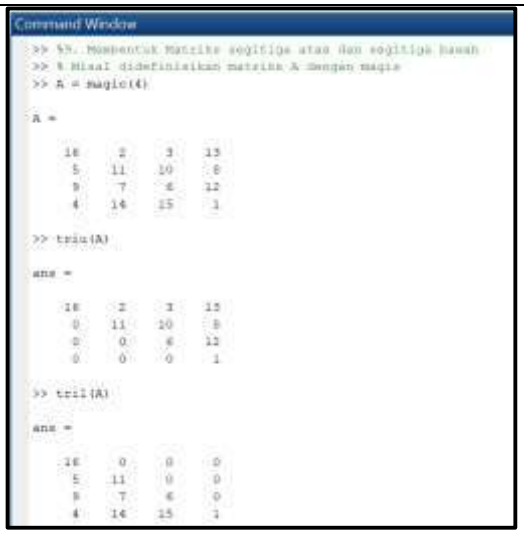
Gambar 3. 38. Penggunaan diag(v) Membangkitkan matriks diagonal

Matriks Segitiga

Matriks bujur sangkar dianamakan sebagai matriks segitiga atas jika semua elemen-elemen di bawah diagonal utama adalah nol, sedangkan matriks bujur sangkar dinamakan matriks segitiga bawah jika semua elemen-elemen di atas diagonal utama adalah nol. Matriks segitiga atas atau matriks segitiga bawah dinamakan sebagai matriks segitiga (triangular).

Pada MATLAB, suatu matriks bujursangkar dapat diubah kedalam matriks segitiga atas dengan menggunakan fungsi triu(A) , diamana A merupakan argumen masukan berupa matriks bujursangkar. Sedangkan untuk membentuk matriks segitiga bawah dapat dibentuk dengan menggunakan tril(A).

Pada gambar di bawah ini, suatu matriks bujur sangkar dibentuk menjadi matriks segitiga atas dan matriks segitiga bawah yang berukuran 4 x 4.



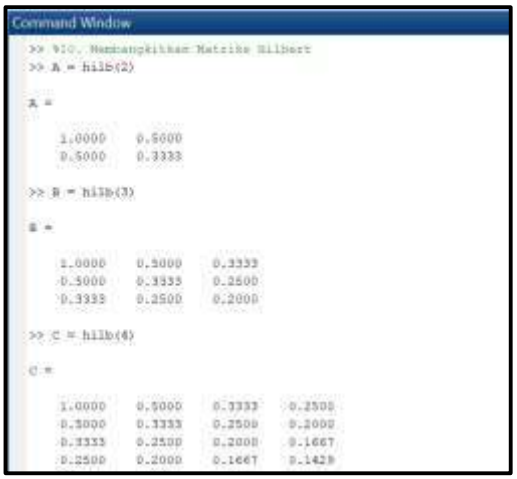
Gambar 3. 39. Membentuk matriks segitiga

Matriks Hilbert

Matriks Hilbert diperkenalkan oleh Hilbert pada tahun 1984. Matriks tersebut berupa matriks bujursangkar berupa fraksi unit. Dengan algoritma :$\boldsymbol{A(i,j) = \frac{1}{(i+j-1)}}$

Pada MATLAB untuk membangkitkan matriks Hilbert maka kita dapat menggunakan fungsi hilb(n), diamana n merupakan ordo matriks Hilber yang akan dibuat.

Pada gambar di bawah ini menunjukkan, bagaimana membangkitkan matriks hilbert berordo 2x2, 3x3 dan 4x4.



Gambar 3. 40. Membangkitkan matriks Hilbert

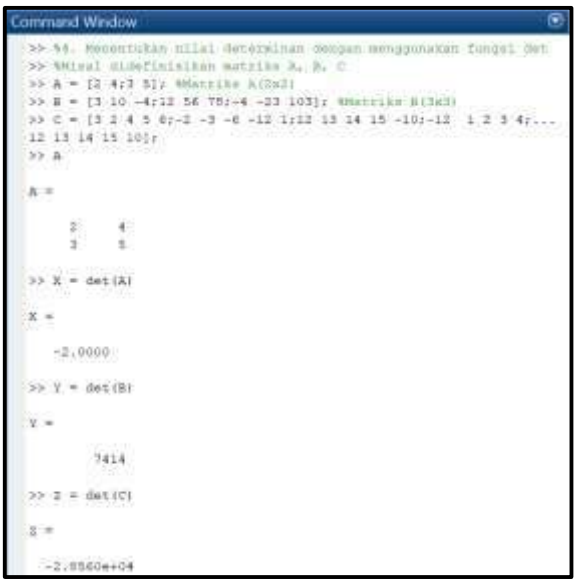
Tabel 14. Matriks khsusus dalam MATLAB

❖ Determinan, Inverse dan Rank

Determinan Matriks

Dalam tinjauan aljabar linear, determinan adalah nilai yang dapat dihitung dari unsur suatu matriks persegi atau bujur sangkar. Determinan suatu matriks berupa sebuah bilangan. Pada MATLAB, kita dpat menentukan nilai determinan secara cepat dan akurat pada matriks bujursangkar dalam ukuran yang besar dengan menggunakan det(M), diamana M sebagai argumen masukan berupa matriks bujursangkar yang akan ditentukan nilai determinannya.

Pada gambar di bawah ini menunjukkan penentuan Determinan Matriks yang berukura 2 x 2, 3 x 3 dan 5x5.



Gambar 3. 41. Penggunaan fungsi det(M) untuk menentukan determinan matriks

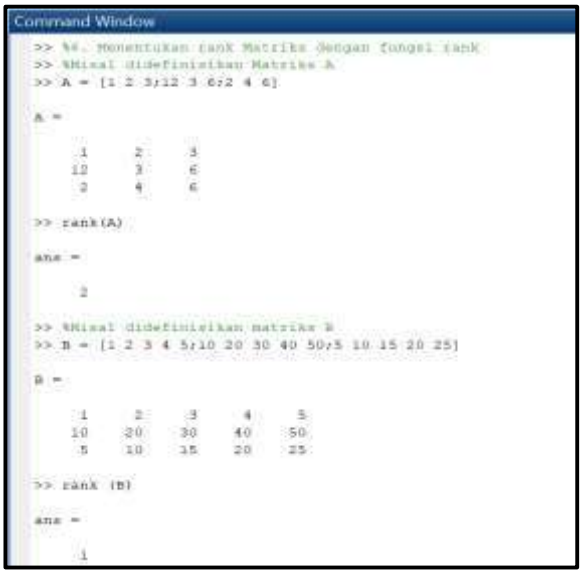
❖ Rank Matriks

Rank baris dari matriks A adalah dimensi dari ruang baris matriks A. Rank kolom dari matriks A adalah dimensi dari ruang kolom matriks A. Dan Ternyata Rank Baris = Rank Kolom ditulis r(A)

Catatan :

- ❖ Rank dari matriks menyatakan jumlah maksimum vektor-vektor baris/kolom yang bebas linier
- ❖ Untuk mencari rank dari suatu matriks dapat digunakan transformasi elementer. Dengan mengubah sebanyak mungkin baris/kolom menjadi vektor nol (karena vektor nol adalah bergantung linier).

Pada MATLAB, untuk menentukan rank Matriks, kita dapat menggunakan fungsi rank(M).

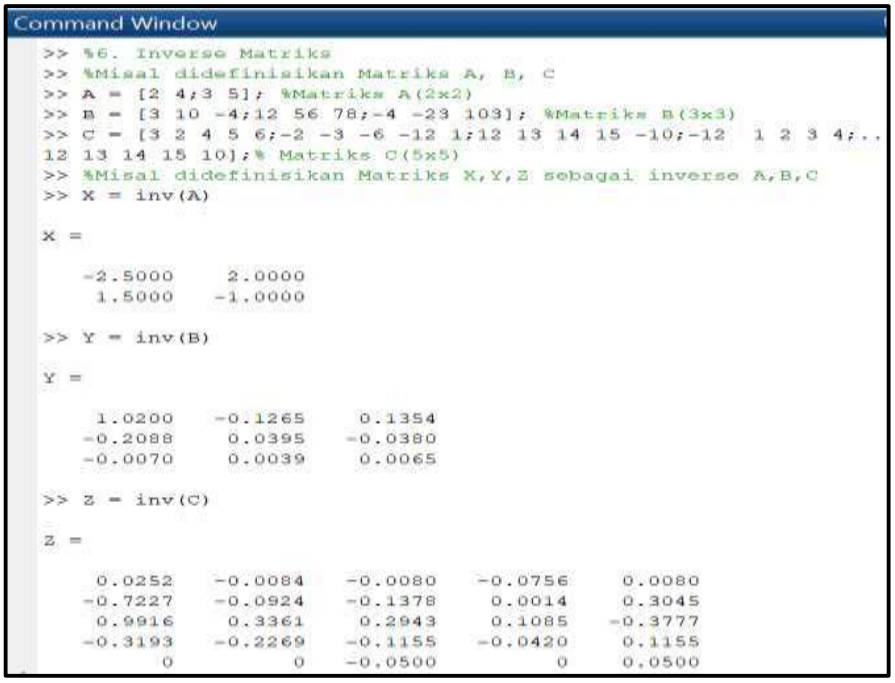


Gambar 3. 42. Penggunaan fungsi rank(A) untuk menentukan nilai rank dari suatu matriks

❖ Inverse Matriks

Inverse dari suatu matriks biasa dikenal sebagai matriks balikan. Inverse Matriks dari A dinotasikan dengan A^{-1}. Hasil kali dari Matriks A dengan inverse dari Matriks A berupa matriks identitas. Matriks yang mempunyai inverse matriks dikenal dengan matriks non singular(matriks invertible) sedangkan matriks yang tidak mempunyai inverse dikenal dengan matriks singular. Pada MATLAB, untuk menentukan inverse matriks dapat dilakukan dengan memanfaatkan fungsi inv(A) dimana A merupakan argumen masukan.

Pada gambar di bawah ini Menunjukkan penggunaan fungsi inv(A) pada beberapa matriks bujur sangkar : 2 x2, 3 x 3dan 5 x 5 yang apabila ditentukan secara manual membutuhkan metode khusus dan waktu yang cukup lama



Gambar 3. 43. Penggunaan fungsi inv(M) dalam mencari inverse matriks

❖ Rotasi dan Pencerminan Matriks

```
%MISAL DIDEFINISIKAN MATRIKS M (3X3)
M = [12 16 18;22 28 26;15 17 18]
%Memutar Matriks
rot90 : Memutar Matriks sebesar 90° Berlawanan arah jarum jam
PUTAR1 = rot90(M)
%Memutar Matriks
rot90(M,k) : Memutar Matriks sebesar 90° Berlawanan arah jarum jam sebanyak k kali
```

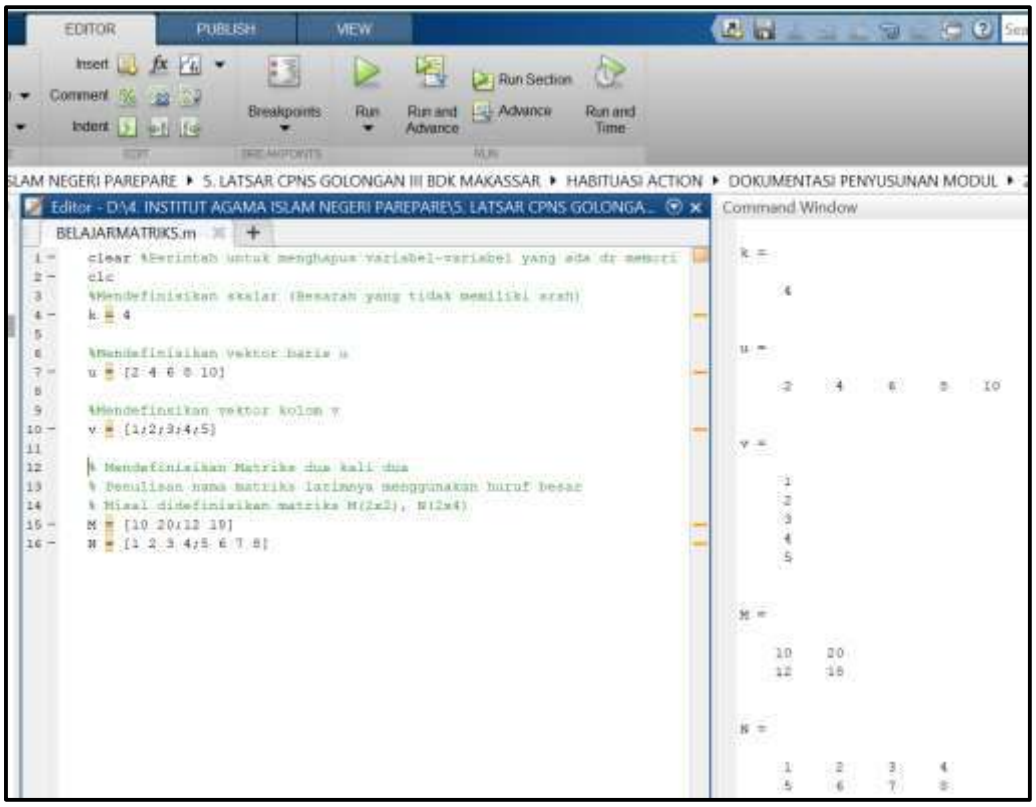
```
>> MANIPULASIMATRIKS
M =
    12    16    18
    22    28    26
    15    17    18
PUTAR1 =
    18    26    18
    16    28    17
    12    22    15
PUTAR2 =
    18    17    15
    26    28    22
    18    16    12
CERMIN1 =
```

PUTAR2 = rot90(M,2) % Pencerminan Matriks : flipup : Mencerminkan matriks secara vertikal (atas – bawah) CERMIN1 = fliplr(M) % Pencerminan Matriks : fliplr : Mencerminkan matriks secara horizontal (kiri-kanan) CERMIN2 = flipud(M) %MISAL DIDEFINISIKAN MATRIKS N (3X4) N = [1 2 3 4;5 6 7 8;9 10 11 12] %Mengubah bentuk : reshape(N,size baru) ubahbentuk1 =reshape(N,2,6) ubahbentuk2 =reshape(N,4,3)	18 16 12 26 28 22 18 17 15 CERMIN2 = 15 17 18 22 28 26 12 16 18 N = 1 2 3 4 5 6 7 8 9 10 11 12 ubahbentuk1 = 1 9 6 3 11 8 5 2 10 7 4 12 ubahbentuk2 = 1 6 11 5 10 4 9 3 8 2 7 12

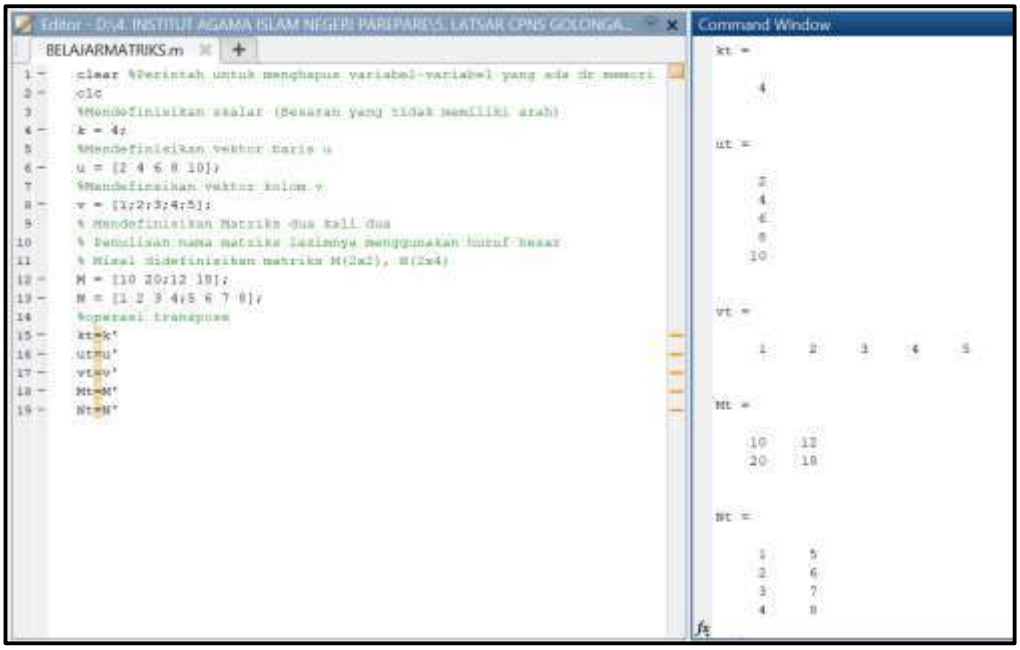
Tabel 15. Pencerminan dan pengbahan bentuk Matriks

❖ Pendefinisian Matriks pada M-File

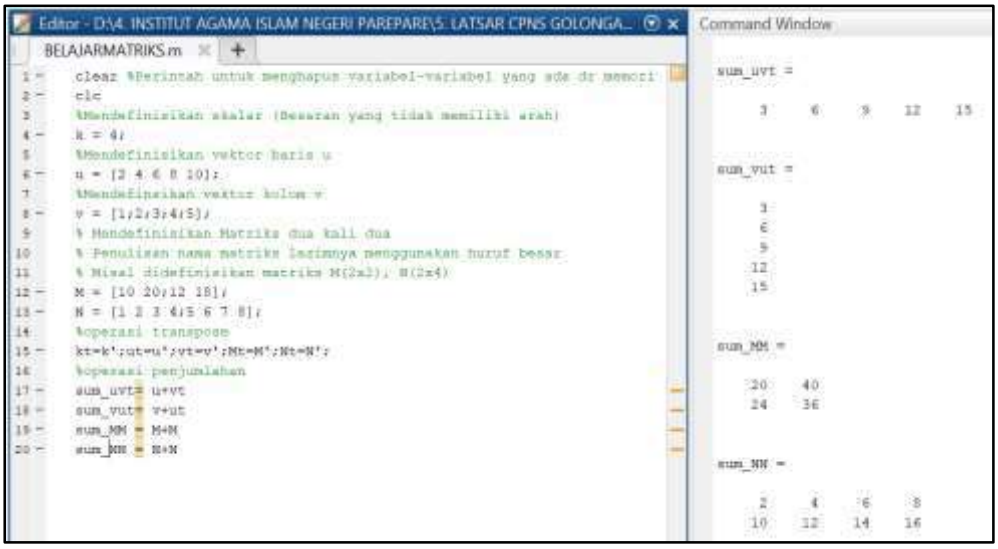
Pada contoh diatas diberikan beberapa contoh pengolahan vektor dan matriks yang dibuat pada jendela command window. Umumnya pengolahan dari suatu matriks menggunakan operasi dan pembangkitan yang berkesinambungan, masalah tersebut kurang efektif untuk dibuat di buat pada command window. Dalam mengatasi hal tersebut, maka mestilah kita menggunakan jendela editor untuk menjalankan perintah secara berkseinambungan. Pada gambar-gambar diberikut diberikan contoh pendefinisian dan pengolahan matriks pada jendela editor.



Gambar 3. 44. Penggunaan jendela editor dalam mendefiniskan dan mengolah matriks



Gambar 3. 45. Penggunaan transpose pada jendela editor



Gambar 3. 46. Pengoperasian Matriks pada jendela editor

3. A. Fungsi-fungsi Statistika

Pada bagia sebelumnya kita telah mempelajari bahwa penyajian data melalui vektor ataupun matriks baik yang elemennnya diinput secara manual, maupun yang dibangkitkan melalui penggunaan fungsi-fungsi pembangkit data. Melalui nilai-nilai vektor ataupun Matriks yang ada dapat dipandang sebagai statistik (data hasisl pengukuran) yang dapat dianalisis lebih lanjut. Analisis sederhana pada statistik yaitu analisis deskriptif dalam mengamati ukuran-ukuran pemusatan dan penyebaran data.

Pada sub Bab ini akan diperkenalkan beberapa fungsi statistik yang sering digunakan dalam satistika deskriptif. Selain pengenalan fungsi, juga akan diberikan contoh-contoh penggunaan fungsi fungsi tersebut. Dengan ini, diharapkan mampu memberikan gambaran mengenai cara kerja dari masing-masing fungsi. Berikut fungsi fungsi statistik yang disajikan dalam tabel berikut

FUNGSI DAN KETERANGANNNYA	CONTOH PENGGUNANNYA PADA COMMAND WINDOW
max(X) Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi max	>> %Misal didefinisikan vektor v dan Matriks M >> v = [12 14 10 16 17 19 45 90 98]; >> max(v)

maka akan menghasilkan nilai balik yaitu nilai terbesar dalam vektor tersebut. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing nilai terbesar pada setiap kolomnya	ans = 98 >> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25] M = 12 14 17 18 13 12 15 18 20 12 8 6 10 27 19 25 >> max(M) ans = 20 27 19 25 >>max(max(M)) Ans = 27
min (X) Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi min maka akan menghasilkan nilai balik yaitu nilai terkecil dari elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing nilai terkecil pada setiap kolomnya.	**>> %Misal didefinisikan vektor v dan Matriks M** **>> v = [12 14 10 16 17 19 45 90 98];** **>> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25];** **>> nilaiminimum_v = min(v)** **nilaiminimum_v =** **10** **>> nilaiminimum_M = min(M)** **nilaiminimum_M =** **10 12 8 6** **>> nilaiminimum_M = min(min(M))** **nilaiminimum_M = 6**
mean (X) Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi mean maka akan menghasilkan nilai balik yaitu nilai rata-rata dari	**>> %Misal didefinisikan vektor v dan Matriks M** **>> v = [12 14 10 16 17 19 45 90 98];** **>> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25];** **>> ratarata_v = mean(v)**

elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing nilai rata-rata pada setiap vektor kolomnya.	ratarata_v = 35.6667 >> ratatata_M = mean(M) ratatata_M = 13.7500 16.2500 14.7500 16.7500
median (X) Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi median maka akan menghasilkan nilai balik yaitu nilai tengah terurut (median) dari elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing nilai tengah terurut (median) pada setiap vektor kolomnya.	>> %Misal didefinisikan vektor v dan Matriks M >> v = [12 14 10 16 17 19 45 90 98]; >> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25]; >> nilaitengah_v = median(v) nilaitengah_v = 17 >> nilaitengah_M = median(M) nilaitengah_M = 12.5000 13.0000 16.0000 18.0000
sum(X) Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi sum maka akan menghasilkan nilai balik yaitu penjumlahan dari elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing	>> %Misal didefinisikan vektor v dan Matriks M >> v = [12 14 10 16 17 19 45 90 98]; >> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25]; >> jumlahan_v = sum(v) jumlahan_v = 321 >> jumlahan_M = sum(M) jumlahan_M = 55 65 59 67

penjumlahan pada setiap vektor kolomnya.	

`prod(X)`
Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi prod maka akan menghasilkan nilai balik yaitu perkalia dari elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing perkalian pada setiap vektor kolomnya.
	`>> %Misal didefinisikan vektor v dan Matriks M`
	`>> v = [12 14 10 16 17 19 45 90 98];`
	`>> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25];`
	`>> perkalian_v = prod(v)`
	`perkalian_v =`
	`3.4460e+12`
	`>> perkalian_M = prod(M)`
	`perkalian_M =`
	`31200 54432 38760 48600`

`std(X)`
Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi std maka akan menghasilkan nilai balik yaitu standar deviasi dari elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing standar deviasi pada setiap vektor kolomnya.
	`>> %Misal didefinisikan vektor v dan Matriks M`
	`>> v = [12 14 10 16 17 19 45 90 98];`
	`>> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25];`
	`>> standardeviasi_v = std(v)`
	`standardeviasi_v =`
	`34.6879`
	`>> standardeviasi_M = std(M)`
	`standardeviasi_M =`
	`4.3493 7.2284 4.7871 7.8899`

`skweness(X)` Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi	`>> %Misal didefinisikan vektor v dan Matriks M` `>> v = [12 14 10 16 17 19 45 90 98];`

skewness maka akan menghasilkan nilai kecondongan dari elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing nilai kecondongan pada setiap vektor kolomnya.	`>> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25];`
	`>> nilaikecondongan_v = skewness(v)`
	`nilaikecondongan_v =`
	`1.0938`
	`>> nilaikecondongan_M = skewness(M)`
	`nilaikecondongan_M =`
	`0.8684 1.0980 -0.7697 -0.5305`

kurtosis (X)
Jika X didefinisikan sebagai inputan vektor (baris atau kolom), dengan fungsi kurtosis maka akan menghasilkan nilai kurtosis dari elemen-elemen vektor. Sedangkan jika X didefinisikan sebagai Matriks, nilai baliknya berupa vektor baris yang diambil dari masing-masing niali kurtosis pada setiap vektor kolomnya.
	`>> %Misal didefinisikan vektor v dan Matriks M`
	`>> v = [12 14 10 16 17 19 45 90 98];`
	`>> M = [12 14 17 18;13 12 15 18;20 12 8 6;10 27 19 25];`
	`>> kurtosis_v = kurtosis(v)`
	`kurtosis_v =`
	`2.4414`
	`>> kurtosis_M = kurtosis(M)`
	`kurtosis_M =`
	`2.1528 2.2845 2.0546 2.0636`

Tabel 16. Fungsi-fungsi dalam statistika

D. Rangkuman

- Program MATLAB merupakan program dengan bahasa tingkat tinggi sangat efisien dalam mendefinisikan, membangkitkan, mengoperasikan dan memanipulasi array atau larik baik dalam bentuk vektor maupun matriks.

- Dalam membangkitkan elemen-elemen Matriks dapat dilakukan dalam berbagai cara yaitu dengan mendefinisikan elemen-elemen matriks berdasarkan indeksnya, atau dengan menuliskan elemen-elemennya menyerupai penulisan elemen vekor. Penulisan elemen elemen matriks dimulai dengan menuliskan elemen-elemen setiap barisnya, dan untuk membuat baris baru dipisahkan dengan menggunakan tanda (;).
- Di dalam MATLAB tersedia perintah-perintah untuk membangkitkan matriks-matriks khusus, seperti matriks nol, matriks satu, matriks identitas, matriks diagonal, matriks Hilbert

E. Latihan

Latihan 4. 1: Misal didefinisikan A, B, u, k

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{2} & \mathbf{1} \\ \mathbf{-1} & \mathbf{2} \end{bmatrix}; \boldsymbol{B} = \begin{bmatrix} \mathbf{0} & \mathbf{-1} \\ \mathbf{3} & \mathbf{1} \end{bmatrix}; \boldsymbol{u} = \begin{bmatrix} \mathbf{1} \\ \mathbf{2} \end{bmatrix}; \boldsymbol{k} = \mathbf{5}$$

Tentukan hasil dari secara manual dan bandingkan hasil dengan menggunakan MATLAB

- $\boldsymbol{C = A + B}$ **F = A/B**
- $\boldsymbol{D = A.* B}$ **G = A\B**
- $\boldsymbol{E = A * B}$ **F = A./B**
- $\boldsymbol{A + u}$ **H= A* k**
- $\boldsymbol{A * u}$ **M= u*b**

Latihan 4. 2: Misal didefinisikan Matriks A pada kotak bilangan berikut :

0.0	0.5	2	1.5	-3
4	2	-4	3	6
-2.4	4	-5	2.8	1
4	8	7.2	3	0

Tentukan hasil dari :

- Ukuran dari Matriks A
- Tentukan nilai dari indeks (3,2)
- Akses nilai yang berada pada Baris ketiga untuk semua kolom.
- Tentukan ukuran dari A([1,3], end)

Latihan 4. 3: Bangkitkan vektor dan matriks dengan ketentuan sebagai berikut

Latihan 4. 4: Gunakan fasilitas help untuk mengeksplorasi pembangkitan, dan pengolahan array skalar, vektor dan matriks.

- Tuliskan dan jalankan kembali perintah-perintah yang belum tersedia pada pembahasan dalam Bab ini.
- Berikan narasi tentang hubungan matriks dengan dunia terapan dan pengolahannnya dalam Matematika Komputasi.

- Vektor berukuran 1 x 10 yang elemen-elemnnya dimulai dari 2.8 dengan kenaikan 1.8
- Vektor berukuran 8 x 1 yang elemen-elemennya adalah kelipatan dari 7 dimulai dari 7077.
- Matriks berukuran 4 x 4 dengan baris pertama mempunyai kenaikan 8, dimulai dari 12, sedangkan baris kedua kenaikan -2.5 dimulai dari 10. Baris ketiga adalah bilangan kelipatan 3 mulai dari 12 dan baris keempat adalah kebalikan dari baris pertama.

BAB 4. CONTROL FLOW, PERULANGAN, PENYELEKSIAN KONDISI DAN PENGAMBILAN KEPUTUSAN

Kemampuan akhir yang diharapkan

A. Pentingnya Control Flow

Dalam meyelesaikan masalah-masalah matematika komputasi, efektivitas dan efisiensi dari penulisan perintah menjadi hal yang utama untuk mengoptimalkan penggunaan perintah maupun kompleksitas memori dalam mengeksekusi perintah. Sebelumnya kita telah mengenal dan menggunakan M-File yang diperuntukkan dalam mengkonstruksi satu pemecahan masalah yang melibatkan sejumlah perintah secara simultan.

- ❖ Mahasiwa dapat menggunakan Control Flow dalam Pengambilan Keputusan dan Penyeleksian Kondisi.
- ❖ Mahasiswa dapat dengan mudah memahami dan menggunakan pernyataan for dalam perulangan.
- ❖ Mahasiswa dapat memahami dan menggunakan pernyataan while dalam penyeleksian kondisi yang berulang.
- ❖ Mahasiswa dapat dengan mudah memahami dan menggunakan pernyataan if
- ❖ Mahasiswa dapat menggunakan pernyataan if bersarang
- ❖ Mahasiswa dapat dengan mudah memahami dan menggunakan pernyataan switch dalam penyeleksian kondisi.
- ❖ Mahasiswa dapat dengan mudah memahami dan menggunakan perintah break dan continue.

Untuk mengatur Efektivitas dan efisiensi pada Pemrograman MATLAB di dalam script/m-file maka diperlukan adanya kontrol program.

Masalah penyeleksian kondisi dan pengambilan keputusan serta perulangan yang melibatkan hukum-hukum logika menjadi satu bentuk kontrol program yang dapat dilakukan dalam pemrograman. Dalam membuat suatu program dibutuhkan sistematika yang bersifat dinamis, efisien dan efektif. Kontrol program ini sangat berguna karena memungkinkan komputasi- komputasi yang lalu mempengaruhi komputansi yang akan datang. MATLAB menyediakan empat struktur kontrol program, yaitu **loop for, loop while, kontruksi switch-case dan kontruksi if-else-end.**

Kontruksi-kontruksi tersebut seringkali melibatkan banyak perintah MATLAB, yang oleh karenanya konstruksi ini lebih banyak terdapat dalam M-file. Pada Bab ini diperkenalkan beberapa statement sebagai control flow yaitu statement for..end, if … else…end, while, switch case, break dan continue. Implementasi control flow pada MATLAB agar mencapai efisiensi ruang dan waktu komputasinya, perlu di atur perangkat dan langkah-langkahnya secara sistematis, terbatas dan logis, hal tersebut dikenal sebagai Algoritma. Penyajian algoritma dapat berupa langkah-langkah

instruksional, diagram alir (*flow chart*) atau dalam bentuk kode-kode semu (*pseudo code*). Melalui algoritma inilah disusun seperangkat program yang melibatkan *control flow* sesuai dengan kebutuhan dan masalah yang dipecahkan.

Dalam prosesnya, nilai kebenaran dari suatu ekspresi dioperasikan dengan menggunakan operator logika (BAB Pertemuan 2). Ekspresi-ekspresi dikenal dengan ekspresi Boolean. Ada beberapa daftar ekspresi boolean yang ada dalam MATLAB seperti tabel berikut ini :

`a==b`	Benar jika a sama dengan b
`a>b`	Benar jika a lebih besar b
`a<b`	Benar jika a lebih kecil b
`a>=b`	Benar jika a lebih besar ata sama dengan b
`a<=b`	Benar jika a lebih kecil atau sama dengan b
`a~=b`	Benar jika a tidak sama dengan b
`a & b`	Benar jika kedua ekspresi boolean a dan b benar
`a | b`	Benar jika paling sedikit satu diantara ekpresi boolean a dan b benar
`a xor b`	Benar jika hanya satu diantara ekpresi boolean a dan b benar
`~a`	Negasi a, benar jika ekspresi boolean a bernilai salah

Tabel 17. Nilai kebenaran logika Boolean

Pada contoh diBab-Bab sebelumnya, script pada editor dan perintah-perintah pada command window, ditunjukkan melalui potongan gambar. Pada Bab Kali ini, script pada jendela editor dalam hal ini M-File disajikan dalam bentuk text yang disalin langsung dari jendela editor. Demikian pula dengan hasil running ataupun perintah pada command window, teks-teks dari perintahnya disalin langsung ke dokumen modul ini.

B. Statement For

Dalam mengangani perulangan perintah, MATLAB menyediakan statement **for** memungkinkan sekelompok perintah diulang (Looping)

sebanyak suatu jumlah yang tetap yang dikenal sebagai loop for. Bentuk umum loop for adalah:

```
for variabel= ekspresi
statement1;
statement 2;
…
End
```

Perintah antara statement for dan statement end dikerjakan sekali untuk setiap kolom dalam array. Untuk tiap iterasi, x diisi dengan kolom array berikutnya, yaitu dalam iterasi ke-n dalam loop. Berikut beberapa contoh penggunaan statement **for** pada M-File.

SCRIPT PADA M-FILE	HASIL EKSEKUSI PADA COMMAND WINDOW

`%1. CONTOH PENGGUNAAN STATEMENT for`
`for i = 1:10`
`  fprintf('%d\n',i)`
`end`
	`>> PERULANGAN`
	`1` (iterasi pertama)
	`2` (iterasi kedua)
	`3`
	`4`
	`5`
	`6`
	`7`
	`8`
	`9`
	`10` (iterasi kesepuluh)
	`>>`

Ekspresi `for i = 1:10` menyatakan bahwa pada setiap iterasi, i akan bernilai dari 1 hingga berulang pada i=10. Selanjutnya pernyataan `fprintf('%d\n',i)` menformat keluaran setiap iterasi.	

`%2. CONTOH PENGGUNAAN STATEMENT for`
`clc`
`clear`
`for i = 1:10`
`  x(i)=sin(i*pi/10);`
`end`
`x`
	`x =`
	`Columns 1 through 5`
	`0.3090 0.5878 0.8090 0.9511 1.0000`
	`Columns 6 through 10`
	`0.9511 0.8090 0.5878 0.3090 0.0000`

Ekspresi 1:10 pada `for i = 1:10` menyatakan bahwa pada setiap iterasi, i akan bernilai dari 1 hingga berulang pada i=10. Namun pada I kali ini direkam sebagai indeks dari vektor x, dan digunakan untuk menentukan elemen-elemen vektor x yang dirumuskan dengan `x(i)=sin(i*pi/10);` yang	

dieksekusi setiap iterasi. Setelah iterasi kesepuluh, selanjutnya nilai x ditampilkan.

```
%3.    CONTOH    PENGGUNAAN
STATEMENT for
for i = 20:-2:0
    fprintf('%d\n',i)
end
```

```
>> PERULANGAN
20
18
16
14
12
10
8
6
4
2
0
```

Ekspresi 1:10 pada `for i = 20:-2:10` menyatakan bahwa pada setiap iterasi, Nilai i akan dibangkitkan mulai dari 20 dengan decrement 2 (penurunan) pada setiap iterasinya dan berhenti pada saat i = 10. Namun pada nilai i yang dibangkitkan akan digunakan sebagai argumen masukan pada perintah `fprintf('%d\n',i)`.

```
%4.    CONTOH    PENGGUNAAN
STATEMENT for
data=[1 2 3 4;8 7 6 5];
for n=data
x=n(1)-n(2)
end
```

```
>> PERULANGAN
x =    -7
x =    -5
x =    -3
x =    -1
```

```
%5.    CONTOH    PENGGUNAAN
STATEMENT for
for i=1:10
   for j=1:5
U(i,j)=i^2+j^2;
   end
end
U
```

```
>> PERULANGAN
U =
     2     5    10    17    26
     5     8    13    20    29
    10    13    18    25    34
    17    20    25    32    41
    26    29    34    41    50
    37    40    45    52    61
    50    53    58    65    74
    65    68    73    80    89
    82    85    90    97   106
   101   104   109   116   125
```

Dengan cara diatas kita membuat suatu matriks dimulai dengan membuat elemen matriks baris 1 elemen kolom 1 sampai 5, dilanjutkan dengan membuat elemen matriks baris 2 kolom 1 sampai 5 dan seterusnya sampai baris ke 10.

```
%6. CONTOH PENGGUNAAN STATEMENT for dalam membentuk Matriks Hillbert
H = zeros(5);
   for k=1:5
     for l=1:5
       H(k,l) = 1/(k+l-1);
   end
end
H
```

```
>> PERULANGAN
H =
1.0000    0.5000    0.3333    0.2500    0.2000
0.5000    0.3333    0.2500    0.2000    0.1667
0.3333    0.2500    0.2000    0.1667    0.1429
0.2500    0.2000    0.1667    0.1429    0.1250
0.2000    0.1667    0.1429    0.1250    0.1111
```

Matrix H yang dibuat tersebut dinamakan Hilbert matrix. Perintah pertama menandakan ruang dalam memori komputer untuk matriks yang dibangkitkan.

```
%7. CONTOH PENGGUNAAN STATEMENT for dalam membentuk Tabel Kali
%membuat Judul Kolom
fprintf('    ');
for i=1:8
    fprintf('%3d',i)
end
fprintf('\n');% Pindah Baris
%Menampilkan Tabel perkalian
for i = 1:8
    fprintf ('%3d',i)
% Judul baris
    for     j     =1:8
fprintf('%3d',i*j)
    end
    fprintf('\n');
end
```

```
>> PERULANGAN
     1   2   3   4   5   6   7   8
  1  1   2   3   4   5   6   7   8
  2  2   4   6   8  10  12  14  16
  3  3   6   9  12  15  18  21  24
  4  4   8  12  16  20  24  28  32
  5  5  10  15  20  25  30  35  40
  6  6  12  18  24  30  36  42  48
  7  7  14  21  28  35  42  49  56
  8  8  16  24  32  40  48  56  64
```

Tabel 18. Contoh-contoh penggunaan Statemen For

Pada contoh diatas, menunjukkan penggunaan for di dalam for. Bentuk pernyatan yang dieksekusi berupa perintah format keluaran dengan melibatkan indeks I yang berulang.

C. Statement While

Selain pernyataan for sebagai looping dalam mengeksekusi perintah yang berulang. Terdapat pula statement **while** dalam mengeksekusi

sekelompok perintah pengulangan. Perbedaan mendasar antara statement **for** dan **while,** statement for perulangan dilakukan dengan mendefinisikan syarat perulangan secara langsung dan tertutup sedangkan **while** melibatkan pengujian kondisi dan mengeksekusi pernyataan apabila ekspresi yang dibuat bernilai benar. Bagian yang berulang adalah bagian ekpresi yang secara terus menerus diperbaharui pada bagian pernyataan. Berikut bentuk umum dari pernyataan **while**

```
operand dari operator logika pada ekspresi
While ekspresi
----Pernyataan yang dieksekusi
----ekspresi yang diperbaharui
end
```

Perintah yang terdapat diantara statemen **while** dan **end** dieksekusi berulang kali selama semua elemen dalam ekspresi adalah benar. Bentuk perintah yang dieksekusi dapat berupa perintah yang tidak berkaitan dengan ekspresi maupun yang berkaitan dengan ekspresi.

Biasanya evaluasi dari ekspresi menghasilkan nilai skalar, tetapi hasil yang berupa array juga dapat diterima . Jika hasilnya adalah array, semua elemen array harus bernilai benar.

SCRIPT PROGRAM PADA M-FILE	HASIL RUNNING PROGRAM PADA COMMAND WINDOW

`%1. CONTOH PERULANGAN while`
`pencacah = 1;`
`while (pencacah < 11)`
`  disp('MATLAB')`
`  pencacah = pencacah +1;`
`end`
	`>> PERULANGANWHILE1`
	`MATLAB`
	`MATLAB`
	`MATLAB`
	`MATLAB`
	`MATLAB`
	`MATLAB`
	`MATLAB`
	`MATLAB`
	`MATLAB`
	`MATLAB`

Pada Contoh di atas, dapat dilihat bahwa diberikan ekspresi awal, kemudian diseleksi menggunakan perintah while selanjutnya diberi perintah ketika memenuhi (bernilai benar) disp('MATLAB') selanjutnya ekspresi berikutnya diperbaharui. Sehingga hasilnya dapat kita lihat bahwa	

kata MATLAB berulang sebanyak 10 kali. Hal ini diseBabkan karena penacacah terakhir setelah 9 +1 yaitu 10 +1 =11, yang bernilai salah sehingga tidak lagi mengeksekusi perintah disp ('MATLAB') yang ke 11.

```
%2. CONTOH PENGULANGAN DENGAN MENGGUNAKAN while
pencacah = 1;
while(pencacah<11)

fprintf('%d\n',pencacah)
    pencacah = pencacah +1;
end
```

```
>> PERULANGANWHILE2
1
2
3
4
5
6
7
8
9
10
```

Pada contoh di atas hampir serupa dengan contoh sebelumnya, yang membedakan pada contoh ini yaitu perintah bergantung pada operand yang didefinisikan. Pada bagian ini, ketika kondisi terpenuhi maka mengambil nilai pencacah yang terbaru dengan perintah `fprintf('%d\n',pencacah)`. Proses ini berjalan secara terus menerus sampai kondisi tidak lagi terpenuhi yaitu pada saat pencacah = 10+1=11.

```
%3. CONTOH PENGULANAGN DENGAN MENGGUNAKAN STATEMENT while
disp('Memperoleh Faktor Persekutuan Terbesar bilangan m dan n')
m = input('m = ');
n = input('n = ');
r = rem(m,n); %Memperoleh sisa pembagian m dengan n
while r~= 0
    m = n;
    n = r;
    r = rem(m,n);
end
fprintf('Hasilnya yaitu:%d' , n)
```

```
>> PERULANGANWHILE3
Memperoleh Faktor Persekutuan Terbesar bilangan m dan n
MASUKKAN BILANGAN PERTAMA = 48
MASUKKAN BILANGAN KEDUA = 60
Hasilnya yaitu:12
>>
```

Pada contoh terakhir, berupa program untuk memperoleh faktor persekutuan terbesar dari 2 bilangan yang dimasukkan secara interaktif. Selanjutnya operand yang diuji yaitu berupa nilai r.

Tabel 19. Contoh-contoh Penggunaan perulanan While

D. Penyeleksian Kondisi dengan Statement if – else - end

Penyeleksian kondisi merupakan masalah yang sering dihadapi dalam dunia matematika komputasi terutama untuk hal-hal yang berkaitan dengan pengambilan keputusan. Kondisi-kondisi yang diseleksi berupa suatu ekspresi yang mempunyai nilai kebenaran (Benar-Salah). Benar atau salahnya suatu ekspresi didasarkan dengan menggunakan logika komputasi baik yang bersifat numerical maupun berkarakter string.

Penyeleksian kondisi biasa kita jumpai pada, formula Ms. Office Excel, Bahasa Pemrograman Visual Basic, Delphi, java, Maple dengan gaya yang berbeda-beda namun memiliki konsep control flow yang sama yakni ada ekspresi yang diuji kebenarannya , jika bernilai Benar maka akan diberikan perintah tertentu, atau jika salah atau tidak memenuhi maka akan dikenakan perintah tertentu pula.

Didalam MATLAB, biasanya ekspresi yang diuji berupa ekspresi yang bersifat matematis sepeti yang disajikan pada contoh-contoh. Seringkali sederetan perintah harus dikerjakan dengan didasarkan pada hasil tes rasional . Dalam bahasa pemograman, logical ini dikerjakan dengan variasi kontruksi **if – else - end**. Bentuk paling sederhana kontruksi if-else-end pada MATLAB, biasanya tersusun atas :

```
if ekspresi (Benar/Salah)
   perintah (Dijalankan jika ekspresi bernilai Benar)
end
```

Selanjutnya jika kondisi yang diseleksi sebanyak dua kemungkinan mempunyai strukur seperti berikut :

```
if ekspresi (Benar/Salah)
  perintah1 (Dijalankan jika ekspresi bernilai Benar)
else
  perintah2  (Dijalankan  jika  ekspresi  tidak  memenuhi
ekspresi sebelumnya)
end
```

Berikut beberapa contoh penggunaan **if else** yang sederhana yang dibuat pada jendela editor dan dijalankan di command window

SCRIPT PADA M-FILE	HASIL EKSEKUSI PROGRAM PADA COMMAND WINDOW
%1. Contoh Penggunaan Perintah if end x = 12; if x>=0 %x>=0 merupakan kondisi yang diseleksi fprintf('%d merupakan bilangan positif \n',x) end	>> PENYELEKSIANKONDISIIFEND 12 merupakan bilangan positif
Pada contoh di atas menunjukkan penggunaan if ... end dalam menentukan apakh bilangan yang didefinisikan merupakan bilangan positif atau bilangan negatif. Dengan menggunakan penyeleksian kondisi ditetapkan suatu ekspresi logic dalm hal ini if x>=0 yang ketika bernilai benar maka akan mengeluarkan perintah fprintf('%d merupakan bilangan positif \n',x)	
%2. Contoh Penggunaan Perintah if end x = -1; if x>=0 %x>=0 merupakan kondisi yang diseleksi fprintf('%d merupakan bilangan positif \n',x) end	>> PENYELEKSIANKONDISIIFEND

%3. Contoh Penggunaan Perintah if end
x = input('MASUKKAN NAMA ANDA : ','s');
y = input('Apakah anda Mahasiswa IAIN Parepare (YA atau TIDAK)?','s');
if y=="YA" %x>=0 merupakan kondisi yang diseleksi
fprintf('%s adalah mahasiswa IAIN Parepare\n',x)
end
	>> PENYELEKSIANKONDISIIFEND
	MASUKKAN NAMA ANDA : AYU ANDIRA
	Apakah anda Mahasiswa IAIN Parepare (YA atau TIDAK)?YA
	AYU ANDIRA adalah mahasiswa IAIN Parepare

%4. Contoh Penggunaan Perintah if end x = input('Masukkan Bilangan : ');	

`if x>=0 %x>=0 merupakan kondisi yang diseleksi`
`fprintf('%d merupakan bilangan positif \n',x)`
`else`
`fprintf('%d merupakan bilangan negatif \n',x)`
`end`	

Tabel 20. Contoh-contoh penyeleksian kondisi if end

E. Konstruksi if Bersarang dan if.. elseif

Pada penggunaan if … end , kita terbatas pada satu pilihan kondisi yang harus terpenuhi, sedangkan penggunaan if … else… end, terbatas pada dua pilihan yaitu pilihan ekspresi 1 terpenuhi atau benar maka pernyataan 1 dieksekusi, kalau salah pernyataan dua dijalankan. Sementara jika terdapat 3 atau lebih pilihan, kontruksi if-else-end dapat dibuat dalam bentuk pernyataan if di dalam if, selain itu dapat menggunakan statement if … elseif … elseif… else…end dengan bentuk umum seperti berikut:

```
if ekspresi 1
perintah dikerjakan jika ekpresi 1 benar
elseif ekspresi 2
perintah dikerjakan jika ekpresi 2 benar
elseif ekspresi 3
perintah dikerjakan jika ekpresi 3 benar
….
else
perintah dikerjakan jika tidak ada ekspresi yang benar
end
```

Berikut beberapa contoh penggunaan dan cara kerja pernyataan **if… elseif….else…end.**

SCRIPT PADA M-FILE	HASIL EKSEKUSI PADA COMMAND WINDOW
`%1. CONTOH PENGGUNAAN IF BERSARANG IF DALAM IF` `%PROGRAM BERIKUT BERTUJUAN UNTUK`	`>> PENYELEKSIANKONDISIIFBERSARANG` `MENGHITUNG AKAR PERSAMAAN ax^2 + bx + c = 0`

```
MENENTUKAN AKAR PERSAMAAN KUADRAT
disp('MENGHITUNG AKAR PERSAMAAN ax^2 + bx + c = 0')
a = input('a  = ');
b = input('b  = ');
c = input('c  = ');

diskriminan = b^2-4*a*c;
if diskriminan > 0
    disp('Akara Real dan Berbeda');
    x1 = (-b+sqrt(diskriminan))/(2*a)
    x2 = (-b -sqrt(diskriminan))/(2*a)
fprintf('x1 = %f\n', x1)
fprintf('x2 = %f\n', x2)
else
    if diskriminan == 0
        disp('Akar Kembar')

        x = -b/(2*a)
        fprintf ('x1 = x2 = %f\n',x);
    else
        disp('Akar Kompleks')

        x1 = (-b + sqrt(diskriminan))/(2*a);
        x2 = (-b - sqrt(diskriminan))/(2*a);

        fprintf('x1 = ')
        disp(x1)

        fprintf('x2 = ')
        disp(x2)
    end
end
```

```
a  = 1
b  = 6
c  = 9
Akar Kembar

x =

    -3

x1 = x2 = -3.000000
>>
PENYELEKSIANKONDISIIFBERSARANG
MENGHITUNG AKAR PERSAMAAN ax^2 + bx + c = 0
a  = 1
b  = -5
c  = 6
Akara Real dan Berbeda

x1 =

     3


x2 =

     2

x1 = 3.000000
x2 = 2.000000
```

Program di atas merupakan contoh penggunaan if dalam if yang dikenal dengan istilah if bersarang (*nested if*). Dapat dilihat bahwa if tersebut berada dalam if sebelumnya sebelum diakhiri dengan end. Penulisan if

yang berada dalm if utama mempunyai struktur yang sama dengan struktur dasar if.

```
%2. CONTOH PENGGUNAAN IF ELSE PADA KASUS YANG LAIN
%PROGRAM PEMBAYARAN
x    =    input('MASUKKAN JUMLAH BUKU = ');
y    =    input('MASUKKAN HARGA SATUAN = ');

%JIKA PEMBELIAN DIATAS 5 BUAH DISKON 10 %
%JIKA  PEMBELIAN  DIATAS 10 BUAH DISKON 20 %
%JIKA  PEMBELIAN  DIATAS 15 BUAH DISKON 25 %
%JIKA  PEMBELIAN  DIATAS 20 BUAH DISKON 30 %
if x >=20
    diskon = y*(30/100);
    harga      =    (y    -
diskon)*x;
elseif x>= 15
    diskon = y*(25/100);
    harga      =    (y    -
diskon)*x;
elseif x>= 10
    diskon = y*(20/100);
    harga      =    (y    -
diskon)*x;
elseif x>= 5
    diskon = y*(10/100);
    harga      =    (y    -
diskon)*x;
else
    harga = y*x;
end

fprintf('TOTAL      HARGA YANG      DIBAYAR      = %d\n',harga)
```

```
>>
PENYELEKSIANKONDISIIFENDELSE2
MASUKKAN JUMLAH BUKU = 10
MASUKKAN HARGA SATUAN = 1000
TOTAL  HARGA  YANG  DIBAYAR  = 9000
>>
PENYELEKSIANKONDISIIFENDELSE2
MASUKKAN JUMLAH BUKU = 8
MASUKKAN HARGA SATUAN = 900
TOTAL  HARGA  YANG  DIBAYAR  = 7200
>>
PENYELEKSIANKONDISIIFENDELSE2
MASUKKAN JUMLAH BUKU = 4
MASUKKAN HARGA SATUAN = 8000
TOTAL  HARGA  YANG  DIBAYAR  = 32000
>>
PENYELEKSIANKONDISIIFENDELSE2
MASUKKAN JUMLAH BUKU = 5
MASUKKAN HARGA SATUAN = 8000
TOTAL  HARGA  YANG  DIBAYAR  = 36000
```

Program diatas menunjukkan penggunaan perintah if … elseif …elseif ... else … end. Dapat dilihat bahwa pilihan kondisi terdiri atas 5 pilihan

diantaranya 4 pilihan yang didefinisikan secara spesifik dan 1 pilihan yang merupakan komplementer dari 4 kondisi pilihan yang didefinisikan. Selanjutnya dapat pula dijelaskan bahwa setiap pilihan jika terpenuhi didefinisikan perintah yang berbeda.

```
%3. CONTOH PENGGUNAAN IF ELSEIF END
%PROGRAM PENENTUAN NILAI MUTU
nilai = input('Masukkan Nilai Akhir : ');
if nilai >=90
    Mutu ='A';
elseif nilai >=70
    Mutu = 'B';
elseif nilai>=60
    Mutu = 'C';
elseif nilai >= 50
    Mutu ='D';
else
    Mutu ='E';
end
fprintf ('Nilai Mutu : %c\n', Mutu)
```

```
>>
PENYELEKSIANKONDISIIFENDELSE
Masukkan Nilai Akhir : 95
Nilai Mutu : A
>>
PENYELEKSIANKONDISIIFENDELSE
Masukkan Nilai Akhir : 70
Nilai Mutu : B
>>
PENYELEKSIANKONDISIIFENDELSE
Masukkan Nilai Akhir : 65
Nilai Mutu : C
>>
PENYELEKSIANKONDISIIFENDELSE
Masukkan Nilai Akhir : 15
Nilai Mutu : E
```

Program diatas menunjukkan penggunaan perintah if … elseif …elseif ... else … end. Dapat dilihat bahwa pilihan kondisi terdiri atas 5 pilihan diantaranya 4 pilihan yang didefinisikan secara spesifik dan 1 pilihan yang merupakan komplementer dari 4 kondisi pilihan yang didefinisikan. Selanjutnya dapat pula dijelaskan bahwa setiap pilihan jika terpenuhi didefinisikan perintah yang berbeda.

Tabel 21. Contoh penyeleksian kondisi dengan perintah if elseif else end

F. Konstruksi Switch Case

Selain menggunakan statement if … else… end, bentuk penyeleksian kondisi dan pengambilan keputusanyang lain dapat menggunakan statement **switch** sebenarnya yang berguna untuk melakukan pengambilan keputusan yang melibatkan banyak alternatif. Bila sederetan perintah harus dikerjakan dengan didasarkan pada **penggunaan**

berulang-ulang suatu tes dengan argumen yang sama, kontruksi **switch** – **else** akan lebih tepat digunakan.

Bentuk umum pernyataan ini tersusun atas :

```
switch ekspresi
case test ekspresi 1
deret perintah 1 (statement,…,statement)
case {test ekspresi 2, test ekspresi 3, test ekspresi 4}
deret perintah 2 (statement,…,statement)
   ….
otherwise
deret perintah 3 (statement,…,statement)
end
```

Expresi harus berupa skalar atau karakter string. Jika ekspresinya adalah skalar, ekspresi= =test_ekspresi di test oleh statemen case. Jika ekspresi berupa karakter string maka **strcm(ekspresi, test_ekspresi)** ditest. Pada contoh diatas ekspresi dibandingkan dengan dengan test ekspresi 1 pada statemen case pertama, jika keduanya sama maka deret perintah akan dikerjakan, dan deret statemen berikutnya yang berada sebelum statemen end diabaikan. Jika pembandingan pertama tidak memberikan nilai benar maka akan dijalankan deret perintah pada statemen case yang kedua. Jika semua pembanding dengan case gagal akan dikerjakan deret perintah 3 yang mengikuti statemen otherwise.

Berikut beberapa contoh sederhana dari kontruksi switch-case adalah :

SCRIPT PROGRAM PADA M-FILE	HASIL RUNNING PROGRAM PADA COMMAD WINDOW
%CONTOH PENGGUNAAN STATEMENT SWITCH --- ELSE %PENENTUAN BILANGAN GANJIL ATAU GENAP N=input('Masukkan Bilangan Asli : '); x=rem(N,2); switch x case 1 disp(['bilangan',num2str(bilangan),'adalah bilangan ganjil']) case 0 disp(['bilangan ' ,num2str(bilangan),'adalah bilangan genap']) otherwise disp('Bilangan ini tidak mungkin ada') end	>> PENYELEKSIANKONDISISWITCHELSE2 Masukkan Bilangan Asli : 1 bilangan5adalah bilangan ganjil >> PENYELEKSIANKONDISISWITCHELSE2 Masukkan Bilangan Asli : 12 bilangan 5adalah bilangan genap >> PENYELEKSIANKONDISISWITCHELSE2 Masukkan Bilangan Asli : 75 bilangan5adalah bilangan ganjil

%CONTOH PENGGUNAAN SWITCH CASE ST = input('MASUKKAN ARAH MATA ANGIN (INGGRIS/INDONESIA) : ','S'); switch ST case{'UTARA','NORTH'} disp('UTARA / NORTH') case{'SELATAN','SOUTH'} disp('SELATAN/SOUTH') case{'BARAT','WEST'} disp('BARAT / WEST') case{'TIMUR','EAST'} disp('TIMUR/EAST') case'KULON' disp('JANGAN PAKAI BAHASA JAWA') otherwise disp('MASUKKAN SATU KATA AJA YA!') end	>> PENYELEKSIANKONDISISWITCHELSE3 MASUKKAN ARAH MATA ANGIN (INGGRIS/INDONESIA) : UTARA UTARA / NORTH >> PENYELEKSIANKONDISISWITCHELSE3 MASUKKAN ARAH MATA ANGIN (INGGRIS/INDONESIA) : SELATAN SELATAN/SOUTH >> PENYELEKSIANKONDISISWITCHELSE3 MASUKKAN ARAH MATA ANGIN (INGGRIS/INDONESIA) : BARAT BARAT / WEST >> PENYELEKSIANKONDISISWITCHELSE3 MASUKKAN ARAH MATA ANGIN (INGGRIS/INDONESIA) : TIMUR TIMUR/EAST >> PENYELEKSIANKONDISISWITCHELSE3 MASUKKAN ARAH MATA ANGIN (INGGRIS/INDONESIA) : KULON JANGAN PAKAI BAHASA JAWA

% Script file fswitch.
x = ceil(10*rand); % Generate a random integer in {1, 2, ... , 10}
switch x
case {1,2}
disp('ProBability = 20%');
case {3,4,5}
disp('ProBability = 30%');
otherwise
disp('ProBability = 50%');
end
	>>
	PENYELEKSIANKONDISISWITCHELSE
	ProBability = 50%
	>>
	PENYELEKSIANKONDISISWITCHELSE
	ProBability = 50%
	>>
	PENYELEKSIANKONDISISWITCHELSE
	ProBability = 20%
	>>
	PENYELEKSIANKONDISISWITCHELSE
	ProBability = 30%
	>>
	PENYELEKSIANKONDISISWITCHELSE
	ProBability = 50%
	>>
	PENYELEKSIANKONDISISWITCHELSE
	ProBability = 50%

Tabel 22. Contoh penggunaan perintah switch else

G. Mengakhiri Perintah

❖ Break

Di dalam control flow, adakalanya perintah perulangan **for** dan **while** ingin diakhiri pada looping tertentu. Pada kasus ini, MATLAB menyediakan perintah **break** yang berguna untuk mengakhiri eksekusi pada proses perulangan **for** dan **while** . Berikut bentuk umum peletakan perintah **break** di dalam perulangan **for.**

```
for variabel = ekspresi
if ekspresi Boolean
break
end
. . .
end
```

Berikut beberapa contoh sederhana dari kontruksi perintah break didalam for adalah :

SCRIPT PROGRAM PADA M-FILE	HASIL RUNNING PROGRAM PADA COMMAD WINDOW
%1. CONTOH PENGGUNAAN PERINTAH BREAK DI DALAM PERULANAGN FOR for i = 1:7 if i == 6 break end fprintf('%d\n',i) end	>> PERINTAHBREAKDALAMFOR 1 2 3 4 5
Pada program di atas menunjukkan contoh penggunaaan perintah break pada perulanagn for. Perintah break di dalam for menghentikan proses looping dengan menambahkan perintah if sebagai perintah untuk mendefinisikan pada looping ke berapa perintah break akan dieksekusi.	

Tabel 23. Contoh penggunaan perintah break

Pada program di atas didefinisikan perulangan dengan perintah for dengan mendefinisikan nilai i dari 1 sampai 7. Pada bagian tengah diletakkan perintah break. Dalam prosesnya, looping berhenti pada saat i = 6, sehingga dalam hasil runningnya hanya mengeluarkan nilai 1 – 5.

❖ **Pernyataan Continue**

Selain perintah **break** yang diletakkan dalam perintah **for,** terdapat pula perintah **continue** yang digunakan bersama **for** atau **while.** Perintah **continue** digunakan untuk mengatur eksekusi ke iterasi berikutnya. Perbedaan mendasar dengan perintah **break ,** yaitu pada perintah **break** menghentikan langkah-langkah selanjutnya, sedangkan perintah **continue** hanya pada loping tertentu yang dihentikan. Pada pernyataan **for, continue** membuat semua pernyataan di bawahnya akan diabaikan dan variabel pencacah **for** dinaikkan (atau diturunkan) ke nilai berikutnya. Selanjutna,

eksekusi dilanjutkan ke bagian awal pernyataan dalam tubuh **for** sepanjang batas akhir pada variabel **for** belum tercapai**.** Berikut bentuk umum peletakan perintah **break** di dalam perulangan **for**

```
for variabel = ekspresi
if ekspresi Boolean
break
end
. . .
end
```

Berikut contoh sederhana dari kontruksi perintah break didalam for adalah

SCRIPT PROGRAM PADA M-FILE	HASIL RUNNING PROGRAM PADA COMMAD WINDOW
%1. CONTOH PENGGUNAAN PERINTAH CONTINUE DI DALAM PERULANAGN FOR for i = 1:7 if i == 6 continue end fprintf('%d\n',i) end	>> PERINTAHCONTINUE 1 2 3 4 5 7 >>

Tabel 24. Contoh penggunaan perintah continue

Pada program di atas menunjukkan contoh penggunaaan perintah **continue** pada perulanagn for. Perintah continue di dalam for menghentikan proses looping pada satu titik looping yang ditentukan dengan menambahkan perintah if sebagai perintah untuk mendefinisikan pada looping ke berapa perintah **continue** akan dieksekusi.

Pada program di atas didefinisikan perulangan dengan perintah for dengan mendefinisikan nilai i dari 1 sampai 7. Pada bagian tengah diletakkan perintah if i == 6 yang berarti menyeleksi kondisi pada I = 6 apabila bernilai benar maka akan menjalankan perintah continue. Dalam prosesnya, looping berhenti pada saat i = 6, sehingga dalam hasil runningnya hanya mengeluarkan nilai tidak menjalankan perintah i= 6.

H. Rangkuman

- Perbedaan mendasar antara perulangan loop for dengan loop while yaitu, loop for mempunyai kondisi-kondisi dan jumlah perulangan yang

- Dalam pengembangan program pada Matematika komputasi dikehendaki dapat efektif dan efisien secara komputasional, baik efektivitas ruang memori maupun efisiensi waktu komputasi. Dalam mencapai tujuan ini, dibutuhkan kontrol program yang dikenal dengan *Control Flow*.
- Beberapa masalah yang ditemukan dalam dunia komputasi khususnya dalam matematika komputasi diantaranya masalah penyeleksian kondisi, masalah pengambilan keputusan, serta perulangan yang melibatkan hukum-hukum logika. Dalam membuat suatu program yang dinamis, efisien dan efektif serta dapat memecahkan masalah maka dibutuhkan suatu perumusan langkah-langkah kerja yang sistematis, logis, terukur, terbatas dan berdasarkan memecahkan masalah yang akan dipecahkan.
- Dalam mengatasi masalah perulangan, MATLAB menyediakan kontrol program, yaitu **loop for** dan **loop while**. Konsep perulangan dalam matematika komputasi dikembangkan untuk pendefinisian indeks pada Matriks, proses rekursif dan iteratif.

sudah ditetapkan, sedangkan loop while kondisinya diulang dengan penyeleksian kondisi. Sehingga selama kondisi memenuhi maka proses perulangan tetap berjalan atau terjadi proses looping.

- Selanjutnya dalam mengatasi masalah penyeleksian kondisi dan pengambilan keputusan, MATLAB menyediakan kontrol program, yaitu **if.. else.. end** dan **if..elseif.... elseif... else...end** serta **switch .. case**. Konsep penyeleksian kondisi dalam pengembangan Program dalam Matematika Komputasi dapat ditemukan dalam berbagai eksplorasi masalah. Penyeleksian dan pengambilan keputusan dapat dipahami sebagai pengujian kondisi dari suatu ekspresi yang mempunyai nilai kebenaran (benar atau Salah).
- Dalam mengeksekusi kondisi dari suatu ekspresi, **if.. else.. end** dan **if..elseif.... elseif... else...end** serta **switch .. case** mempunyai struktur penulisan dan cara kerja yang berbeda-beda.

- Statemen **if.. else ..end** , digunakan untuk menyleksi kondisi tunggal, atau satu eksprsesi tunggal yang memiliki nilai kebenaran dengan dua pilihan keputusan, apabila ekspresi bernilai benar maka akan dieksekusi sesuai dengan perintah setelah penulisan **if ekspresi.** Sedangkan jika bernilai salah akan mengeksekusi perinah yang berada di bawaj pernyataan **else**, selanjutnya ditutup dengan perintah **end.** Sedangkan Statement **if.. elseif.. elseif .... else ..end,** mempunyai multiekspresi untuk dieksekusi, setiap ekspresi dituliskan setelah perintah **if,** atau **elseif..** Selanjutnya perintah setelah **else** menyatakan kondisi yang tidak diberikan pada ekspresi ekspresi yang ada pada perinah sebelumnya. Setiap keputusan untuk masig-masing kondisi dituliskan pada baris baru setelah penulisan ekspresi yang diseleksi.

- Selanjutnya statement **switch case** berguna untuk melakukan pengambilan keputusan yang melibatkan banyak alternatif. Bila sederetan perintah harus dikerjakan dengan didasarkan pada **penggunaan** berulang-ulang suatu tes dengan argumen yang sama, kontruksi **switch** – **else** akan lebih tepat digunakan.
- Hal mendasar dalam case flow atau kontrol program yang dilakukan pada pengembangan Program Matematika Komputasi adalah bagaimana kemampuan pengguna untuk mengeksplorasi ekspresi-ekspresi matematis yang diseleksi dan dibangun atas logika matematika yang bersifat umum dengan melibatkan operator logika dan operator Relasional yang mensyaratkan mempunyai nilai kebenaran. Atas dasar ini, berbagai masalah matematika dapat dipecahkan denagn pengembanagn konsep perulangan, Penyeleksian kondisi dan Pengambilan keputusan.

I. Latihan

Latihan 4. 1: Tahun kabisat didefinisikan sebagai tahun yang habis dibagi 4 namun tidak habis dibagi 100, atau tahun yang merupakan kelipatan 400.

- Buatlah fungsi **iskabisat(t)** yang dapat menentukan apakah tahun t merupakan tahun kabisat atau tidak (menghasilkan 1 bila tahun kabisat, 0 bila bukan tahun kabisat.
- Tentukan apakah tahun 2000, 2001, 2002, 2003, 2004, 2012, 2100, 2200, 2300, dan 2400 merupakan tahun kabisat atau bukan.

Latihan 4. 2: Buatlah suatu M-File yang dapat mengkombinasikan penggunaan perintah masukan data input dan penyeleksian kondisi yang dapat menampilkan hasil :

- Bilangan yang ada masukkan adalah bilangan positif yang bernilai lebih besar dari nol
- Bilangan yang ada masukkan adalah bilangan negatif yang bernilai lebih kecil dari nol
- Bilangan yang ada masukkan bukan bilangan positif dan negatif

Buatlah suatu M-File yang dapat digunakan untuk mengalikan seperti aturan berikut , Misal dimasukkan n maka kan menghasilkan $\mathbf{1} \times \mathbf{2} \times \mathbf{3} \times \mathbf{4} \times \cdots \times \boldsymbol{n} - \mathbf{1} \times \boldsymbol{n}$

- **Perintah masukan : “Masukkan angka :”**
- **Hasill dari n Faktoial adalah :**

Latihan 4. 3: Buatlah suatu M-File yang dapat menampilkan hasil jumlahan

$\mathbf{1} = \mathbf{1}$	$\mathbf{1} + \mathbf{0} = \mathbf{1}\ (\mathbf{1} \times \mathbf{1})$
$\mathbf{1} + \mathbf{2} = \mathbf{3}$	$\mathbf{1} + \mathbf{3} = \mathbf{4}\ (\mathbf{2} \times \mathbf{2})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} = \mathbf{6}$	$\mathbf{3} + \mathbf{6} = \mathbf{9}\ (\mathbf{3} \times \mathbf{3})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} + \mathbf{4} = \mathbf{10}$	$\mathbf{6} + \mathbf{10} = \mathbf{16}\ (\mathbf{4} \times \mathbf{4})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} + \mathbf{4} + \mathbf{5} = \mathbf{15}$	$\mathbf{10} + \mathbf{15} = \mathbf{25}\ (\mathbf{5} \times \mathbf{5})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} + \mathbf{4} + \mathbf{5} + \mathbf{6} = \mathbf{21}$	$\mathbf{15} + \mathbf{21} = \mathbf{36}\ (\mathbf{6} \times \mathbf{6})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} + \mathbf{4} + \mathbf{5} + \mathbf{6} + \mathbf{7} = \mathbf{28}$	$\mathbf{21} + \mathbf{28} = \mathbf{49}\ (\mathbf{7} \times \mathbf{7})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} + \mathbf{4} + \mathbf{5} + \mathbf{6} + \mathbf{7} + \mathbf{8} = \mathbf{36}$	$\mathbf{28} + \mathbf{36} = \mathbf{64}\ (\mathbf{8} \times \mathbf{8})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} + \mathbf{4} + \mathbf{5} + \mathbf{6} + \mathbf{7} + \mathbf{8} + \mathbf{9} = \mathbf{45}$	$\mathbf{36} + \mathbf{45} = \mathbf{81}\ (\mathbf{9} \times \mathbf{9})$
$\mathbf{1} + \mathbf{2} + \mathbf{3} + \mathbf{4} + \mathbf{5} + \mathbf{6} + \mathbf{7} + \mathbf{8} + \mathbf{9} + \mathbf{10} = \mathbf{55}$	$\mathbf{45} + \mathbf{55} = \mathbf{100}\ (\mathbf{10} \times \mathbf{10})$

BAB 5. GRAFIK

Kemampuan akhir yang diharapkan

- Mahasiswa dapat memahami esensi dari grafik melalui telaah matematika komputasi
- Mahasiswa dapat membuat grafik dari berbagai representasi persamaan dan data.
- Mahasiswa dapat secara teknis menggunakan fungsi-fungsi MATLAB yang berkaitan dengan grafik
- Mahasiswa dapat menginterpretasikan berbagai jenis grafik.
- Mahasiswa dapat membuat project media pembelajaran grafik melalui Aplikasi MATLAB

A. Grafik Dalam Matmatika Komputasi

Sebagai pembelajar yang aktif dalam disiplin ilmu matematika, suatu keharusan untuk memahami esensi dari suatu grafik. Ketika suatu penyajian himpunan data dalam bentuk tabulasi, kita masih sangat sulit untuk dapat membaca karakteristik atau pola hubungan antar data yang

berakibat pada kesulitan kita dalam merumuskan perlakukan atau control dari sajian data tersebut. Meskipun, sebelumnya kita telah melihat, bahwa MATLAB juga menyediakan beberapa fungsi-fungsi statistika yang dapat memformulasikan karakteristik data secara deskriptif.

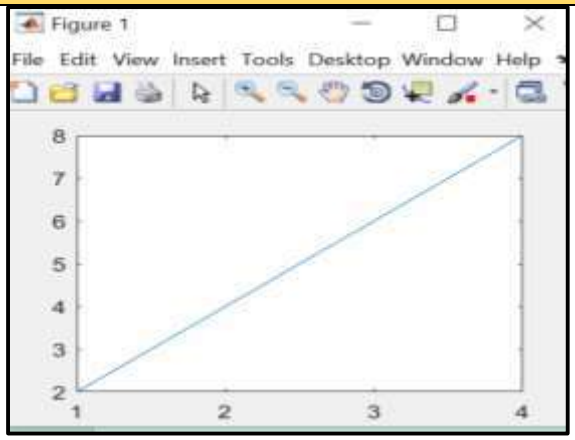
Dalam memahami grafik, tidak cukup hanya dengan memahami jenis-jenisnya. Namun, kita dituntut untuk dapat membuat grafik, menempatkan property-properti grafik, menginterpretasi grafik dan dapat memahami kelakuan grafik yang tidak hanya bersifat statis namun juga yang bersifat dinamis. Secara spesifik, grafik dapat dimaknai sebagai media yang digunakan dalam menvisualisasikan karakteristik, pola, kelakuan dari himpunan data yang berasosiasi dengan data numerik. Secara sederhana, grafik dapat dikatakan sebagai himpunan pasangan titik-titik yang merepresentasikan entitas tertentu sehingga membentuk satu kesatuan pola, warna, bentuk yang didefinisikan oleh pembuat grafik itu sendiri, dengan ini bagi para pengguna atau pembaca grafik dapat menemukan informasi-informasi mengenai karakteristik dari entitas entitas yang digambarkan dalam grafik.

Secara matematis, pembuatan ataupun pengolahan grafik dari MATLAB tidak terlepas dari prinsip kerja MATLAB yang berbasis pada Array (Larik) baik berupa vector, matriks maupun himpunan pasangan antar vector, antar matriks, ataupun antar vector dan matriks. Pada pertemuan ini, kegiatan pelatihan ini akan fokus pada pembuatan grafik sederhana, penambahan properti-properti grafik, dan pengenalan jenis-jenis grafik 3D. Pembuatan grafik juga tidak terlepas dari penggunaan fungsi-fungsi yang disediakan oleh MATLAB, dengan berbagai karakteristik data inputan. Grafik yang diperkenalkan pada pertemuan ini dapat dibedakan atas grafik 2 D dan Grafik 3D.

B. Plotting Grafik

Seperti yang telah dijelaskan pada pengantar di sub Bab sebelumnya, bahwa grafik merupakan representasi dari hubungan titik-titik tertentu dari himpunan data. Pada bagian ini akan diperkenalkan fungsi MATLAB dalam membuat suatu grafik dari kumpulan data yang secara struktur sangat sederhana. Fungsi pertama yang akan diperkenalkan yaitu : **plot(x,y).** Plot dalam istilah sehari-hari dapat dipandang sebagai suatu kegiatan menempatkan titik secara fisik yang diletakkan pada sebaran posisi posisi tertentu. Pada MATLAB, **plot** merupakan salah satu fungsi dengan argumen masukan x dan y untuk setiap titiknya yang selanjutnya dihubungkan secara terurut dengan menggunakan properti, properti yang dikehendaki. Sehingga dalam menggunakan perintah plot, kita harus memastikan bahwa kita telah menyiapkan data x dan data y dengan jumlah elemen yang sama. Ketika perintah tersebut dijalankan pada MATLAB, maka dengan sendirinya MATLAB akan mengeluarkan satu jendela figure yang baru. Berikut beberapa contoh sederhana penggunaan fungsi **plot** dalam membuat grafik

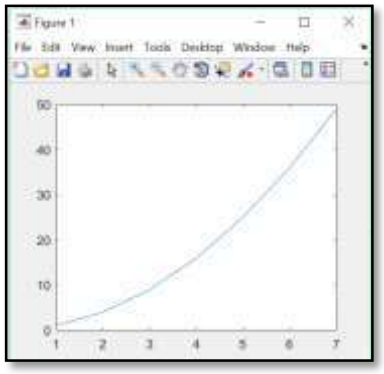
SCRIPT PADA COMMAND WINDOW DAN JENDELA EDITOR	HASIL RUNNING PERINTAH PLOT

`>> X = [1 2 3 4];`
`>> Y = [2 4 6 8];`
`>> plot(X,Y)`
	
	Gambar 5. 1. Grafik dari hasil perintah plot dari data 2 pasangan vektor

Program di atas dibuat secara langsung pada command window dengan mendefinisikan vektor X dan vektor Y. Melalui perintah plot(X,Y), maka jendela figure 1 keluar dan menampilkan satu garis lurus yang menunjukkan pasangan titik-titik dari X dan Y. Secara standar, grafik yang

muncul berupa garis yang kontinu berwarna yang melewati pasangan-pasangan titik X, Y yang terbentuk berdasarkan indeksnya masing-masing.

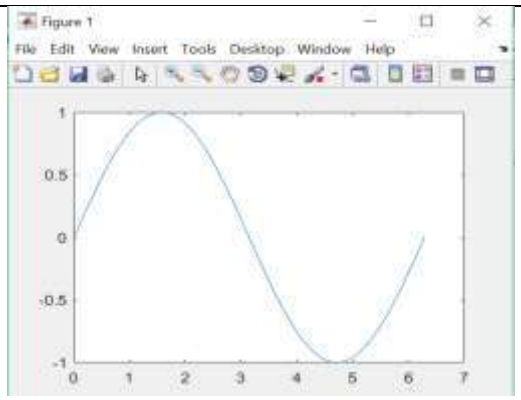
```
>> X = [1 2 3 4 5 6 7]
X =
1     2     3     4     5     6     7
>> Y= X.^2
Y =
1     4     9    16    25    36    49
>> plot(X,Y)
```



Gambar 5. 2. Grafik dari penggunaan perintah Plot dari input data dan persamaan

Seperti halnya pada contoh pertama, bahwa data-data X dan Y didefinisikan dan dijalankan langsung pada Command Window. Pada contoh, elemen-elemne dari vektor Y didefinisikan sebagai fungsi kuadrat dari elemen-elemen X. Sehingga dengan menyambungkannya dengan perintah plot, maka figure 1 pada contoh sebelumnya berganati dengan figure yang baru, dengan membentuk grafik persamaan kuadrat. Secara standar, grafik yang keluar juga berbentuk lengkungan garis yang berwarna biru secara kontinu yang melewati titik-titik x = [1,7].

```
>> X = linspace(0, 2*pi);
>> Y = sin(X);
>> plot(X,Y)
```

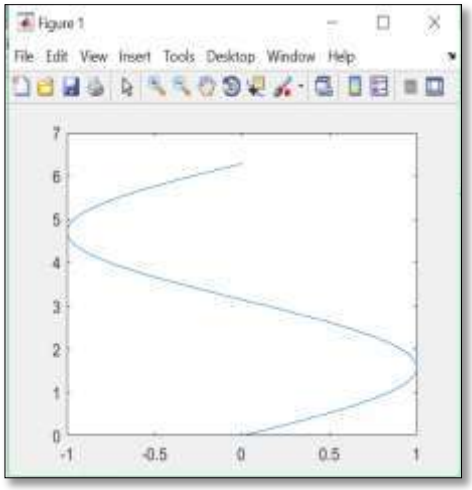


Gambar 5. 3. Penggunaan fungsi plot dari pembangktan data linspace

Pada contoh ini, dapat dilihat bahwa nilai X yang dibangkitkan dengan menggunakan fungsi linspace(0, 2*pi) dalam hal ini berarti bahwa elemen-

elemen vektor X berupa interval nilai dari 0 sampai 2pi (360^0) sebanyak 100 titik dengan panjang yang sama. Selanjutnya nilai Y didefinisikan sebagai fungsi sin dari nilai-nilai X yang dibangkitkan. Melalui perintah plot, secara otomatis dengan running time beberapa detik membentuk satu grafik fungsi trigonometri yang kontinu dalam interval 0 sampai 2Pi. Grafik tersebut merupakan pasangan berurutan dari nilai-nilai X dan Y pada masing-masing titik dengan indeks yang bersesuaian. Perlu diketahui bahwa X dan Y hanyalah penamaan variabel vektor yang sewaktu-waktu dapat diganti dengan nama vektor lain. Bahkan dapat ditukar, X menjadi Y dan Y menjadi X. dan ketika diberikan perintah plot, kita juga dapat mengubah dari (X,Y0 menjadi (Y, X) berikut contohnya.

```
>> X = linspace(0, 2*pi);
>> Y = sin(X);
>> plot(Y,X)
```



Gambar 5. 4. Penggunaan perintah plot dari fungsi trigonometri sinus dengan menampilkan secara terbalik daerah hasil da daerah asal

Dari grafik diatas terlihat bahwa sumbu horizontal berada pada range -1 sampai 1. yang sebelumnya merupakan sumbu vertikal. Hal ini dapat disimpulkan beberapa ketentuan yaitu :

- ❖ Jumlah elemen dari titik titik yang berpasangan atau vektor yang berpasangan harus sama.
- ❖ Perintah plot(X,Y) merupakan fungsi yang menempatkan pasangan-pasangan titik pada himpunan X yang ditempatkan pada arah sumbu horizontal dan pada arah sumbu vertikal).

C. Penambahan Beberapa Properti Pada Grafik

Grafik sebagai representasi dari keberadaan dari suatu kumpulan data, menghendaki penyajian grafik tersaji secara jelas. Dengan demikian penyajian grafik membutuhkan penambahan beberapa properti grafik. Pada bagian ini akan diperkenalkan bagaimana menambahkan Judul, Legenda, dan Label pada sumbu-sumbu grafik. Selain itu juga akan diperkenalkan bagaimana memilih jenis garis, jenis penanda dan warna. Serta diperkenalkan pula bagaimana penanganan sumbu dan penggunaan subplot. Untuk lebih jelasnya berikut secara detail diberikan contoh mengenai penanganan beberapa properti grafik.

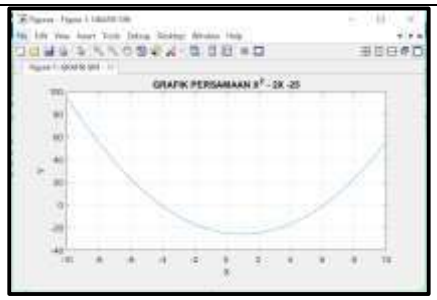
❖ Judul, Grid Line dan Label pada Sumbu

Judul pada grafik merupakan infomasi yang sangat penting dalam suatu penyajian data, melalui judul kita dapat mengetahui dan mengenal informasi utama dalam grafik. Penambahan judul dalam suatu Grafik pada MATLAB dapat dilakukan dengan menambahkan perintah **title('Judul Grafik', 'Font size','12')** yang dituliskan di bawah perintah plot. Sedangkan label dari sumbu-sumbu grafik dapat dibuat dengan menggunakan perintah **xlabel('Judul pada sumbu horizontal x')** dan **yalbel('Judul pada sumbu Vertikal y').**

Penambahan gridline juga kadang dibutuhkan yang membantu kita menentukan posisi posisi titik yang dilewati oleh sumbu. Dalam menambahkan grid line, cukup dengan menuliskan perintah grid on di bawah perintah plot. Berikut contoh penambahan judul, penembahan keterangan sumbu keterangan sumbu pada suatu grafik .

SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PROGRAM PADA JENDELA FIGURE

```
>> X = -10:10;
>> Y = X.^2-2*X-25;
>> plot(X,Y)
>> title('GRAFIK PERSAMAAN X^2 -
2X -25');
>> xlabel('X');
>> ylabel('Y');
>> grid on
```



Gambar 5. 5. Penambahan title pada grafik

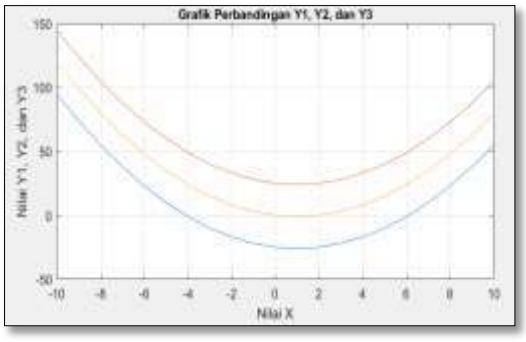
Dari gambar diatas dapat dilihat bahwa terdapat judul grafik, keterangan sumbu horizontal dan vertikal, serta grid pada grafik setelah menambahkan perintah title, xlabel, ylabel dan grid on setelah perintah plot.

❖ Menampilkan Lebih dari Satu Grafik pada sumbu yang sama

Dalam menvisualisasikan hubungan antar himpunan data, kadang dibutuhkan grafik yang menampilkan beberapa himpunan data dalam satu pasangan sumbu sekaligus. Dalam arti yang sederhana, nilai dari X yang sama dipetakan pada persamaan Y yang berbeda. Sehingga dalam penulisan perintahnya dapat dilakukan dengan : plot(x,y1,x,y2,x,y3) , dalam hal ini argumen masukan pada perintah plot tidak hanya satu pasang titik, namun dibuat menjadi 3 pasang titik. Cara lain yang dapat digunakan yaitu dengan menggunakan perintah **hold on – hold off.**

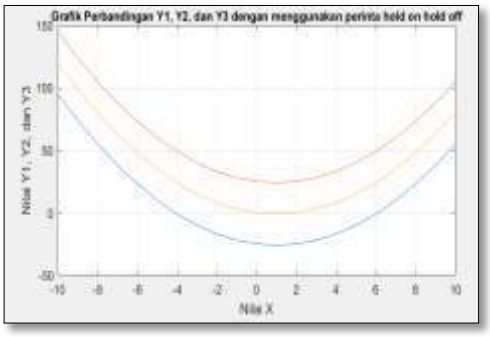
Berikut cara penulisan perintah dalam menampilkan himpunan data yang berbeda dalam satu grafik melalui dua cara yang telah dijelaskan.

```
>> X = -10:10;
>> Y1 =X.^2-2*X-25;
>> Y2 =X.^2-2*X+25;
>> Y3 =X.^2-2*X;
>> plot(X,Y1,X,Y2,X,Y3);
>>            title('Grafik
Perbandingan  Y1,  Y2,  dan
Y3', 'FontSize',10);
>> xlabel('Nilai X');
>>  ylabel('Nilai  Y1,  Y2,
dan Y3');
>> grid on
```



Gambar 5. 6. Penyajian himpunana data yang berbeda dalam satu grafik

```
>> X = -10:10;
>> Y1 =X.^2-2*X-25;
>> Y2 =X.^2-2*X+25;
>> Y3 =X.^2-2*X;
>> plot(X,Y1);
>> hold on
>> plot(X,Y2);
>> plot(X,Y3);
>> hold off
>> title('Grafik Perbandingan Y1, Y2, dan Y3 dengan menggunakan perinta hold on hold off', 'FontSize',10);
>> xlabel('Nilai X');
>> ylabel('Nilai Y1, Y2, dan Y3');
>> grid on
```



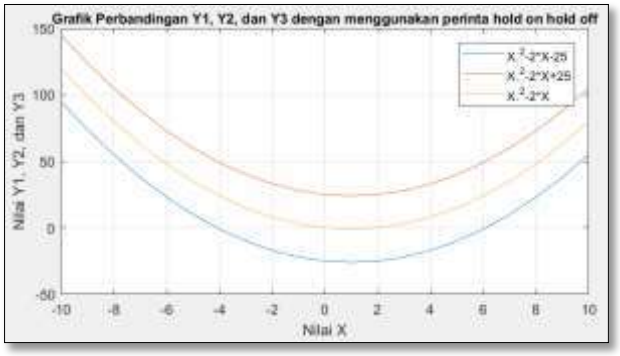
Gambar 5. 7. Penggunaan perintah hold on hold off dalam menyajikan garfik

Pada penyajian beberapa himpunan data dalam satu grafik di atas dapat dilihat bahwa kita dapat melakukan dua cara yaitu dengan membuatnya dalam satu perintah plot dengan argumen masukan sebanyak pasangan himpunan data yang ada. Selanjutnya cara yang lain dapat pula dilakukan dengan menggunakan perintah **hold on** dan **hold off.**

❖ Menambahkan Legenda

Penyajian beberapa himpunan data dalam satu grafik, juga dibutuhkan penambahan legenda sebagai penjelas dari beberapa grafik yang ditambilkan dalam satu pasang sumbu (X dan Y). Berikut contoh penambahan Legenda dengan menggunakan contoh yang ada sebelumnya.

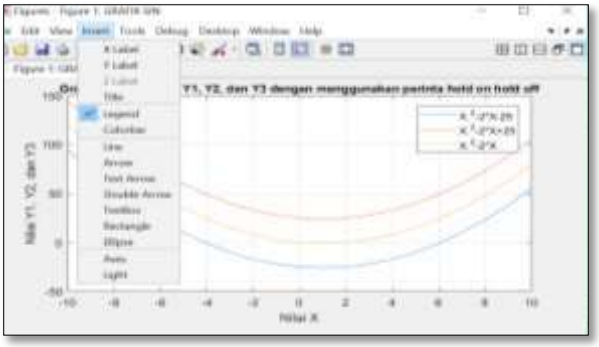
```
>> X = -10:10;
>> Y1 =X.^2-2*X-25;
>> Y2 =X.^2-2*X+25;
>> Y3 =X.^2-2*X;
>> plot(X,Y1);
>> hold on
>> plot(X,Y2);
>> plot(X,Y3);
>> hold off
>>             title('Grafik Perbandingan Y1, Y2, dan Y3 dengan menggunakan perinta hold on hold off', 'FontSize',10);
>> xlabel('Nilai X');
>> ylabel('Nilai Y1, Y2, dan Y3');
>> grid on
```



Gambar 5. 8. Penambahan Legenda pada Grafik

```
>> legend('X.^2-2*X-
25','X.^2-2*X+25','X.^2-
2*X')
```

Cara lain yang dapat digunakan dalam menambahkan properti properti pada grafik seperti judul, label sumbu, grid, legenda juga dapat dilakukan dengan menklik menu **insert** pada jendela figure kemudian pilih title, xlabel, ylabel kemudian Ketik Nama Judul , keterangan sumbu horizontal dan vertikal pada text box masing-masing dan properti-properti lainnya.



Gambar 5. 9. Penggunaan Properties Figure pada Matlab dalam menambhkan berbagai Keterangan dalam grafik

❖ Pengaturan Jenis Garis, Jenis Penanda dan Warna

Pada bagian-bagian sebelumnya, telah diketahui bahwa dalam membuat suatu grafik perlu dilakukan pengaturan atau penambahan properti untuk menghasilkan keutuhan informasi dari karakteristik data yang direpresentasikan. Telah kita lihat bahwa secara standar, grafik yang ditampilkan berupa garis kontinu yang berwarna biru. Selanjutnya dengan perintah hold on pada penambahan himpunan data, masing-masing himpunan data pada grafik dapat dibedakan melalui warna. Dalam MATLAB untuk memplot data, tidak hanya sebatas garis kontinu yang berwarna., namun adakalanya kita membutuhkan style yang lebih beragam. Pada bagian ini akan diperkenalkan beberapa kode perintah yang digunakan untuk mengganti jenis garis, jenis penanda dan warna garis.

Kode Jenis Garis	Keterangan
'-'	Garis kontinu dan utuh seperti pada jenis garis standar
'--'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda - - - - - -
':'	Garis kontinu namun dinampakkan secara putus-putus dengan menggunakan .....

'-.'	Garis kontinu namun dinampakkan secara putus-putus dengan menggunakan kombinasi tanda – dan .
'.'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda .
'x'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda titik cross huruf x
'*'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda bintang(*)
'd'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda belah ketupat (diamond)
'^'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda segitiga atas
'>'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda segitiga ke kanan
'<'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda segitiga ke kiri
'h'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda Heksgram
'o'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda lingkaran kecil
'+'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda tambah
's'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda kotak (square)
'p'	Garis kontinu namun dinampakkan secara putus-putus menggunakan tanda pentagram

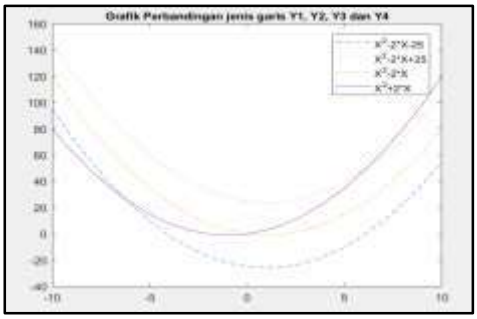
Tabel 25. Tabel kode style garis grafik pada MATLAB

Secara umum, struktur penambahan kode warna disisipkan pada fungsi plot seperti pada contoh berikut :

```
>> plot(X,Y1,'kode jenis garis pasangan X Y1', X,Y2, 'kode jenis garis X Y2', dan seterusnya);
```

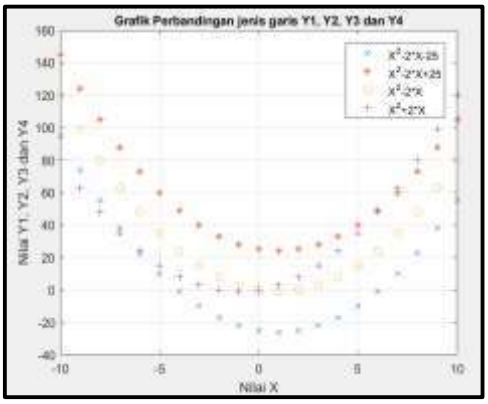
Berikut beberapa contoh penggunaan kode-kode jenis garis yang ditempatkan pada penyajian grafik yang memiliki beberapa himpunan data dalam satu grafik.

SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE

`>> X = -10:10;`
`>> Y1 =X.^2-2*X-25;`
`>> Y2 =X.^2-2*X+25;`
`>> Y3 =X.^2-2*X;`
`>> Y4 =X.^2+2*X;`
`>> plot(X,Y1,'--',X,Y2,':',X,Y3,'-.',X,Y4,'-');`
`>> title('Grafik Perbandingan jenis garis Y1, Y2, Y3 dan Y4', 'FontSize',10);`
`>> legend('X.^2-2*X-25','X.^2-2*X+25','X.^2-2*X','X.^2+2*X')`
	

Gambar 5. 10. Pengaturan style garis pada grafik

```
>> X = -10:10;
>> Y1 =X.^2-2*X-25;
>> Y2 =X.^2-2*X+25;
>> Y3 =X.^2-2*X;
>> Y4 =X.^2+2*X;
>> plot(X,Y1,'x', X,Y2,'*', X,Y3,
'o',X,Y4,'+');
>> title('Grafik Perbandingan
jenis garis Y1, Y2, Y3 dan Y4',
'FontSize',10);
>> xlabel('Nilai X');
>> ylabel('Nilai Y1, Y2, Y3 dan
Y4');
>> grid on
>> legend('X^2-2*X-25','X^2-
2*X+25','X^2-2*X','X^2+2*X')
```



Gambar 5. 11. Penggunaan style ploting titik

Tabel 26. Contoh penggunaan tipe-tipe garis dalam membuat grafik

Pada contoh-contoh sebelumnya terlihat bahwa dengan adanya beberapa garis dalam satu grafik, secara standar sudah dapat dibedakan berdasarkan warna. Namun adakalanya dikehendaki warna-warna tertentu pada himpunan data tertentu. Pengaturan warna tersebut dapat dilakukan dengan menambahkan kode-kode pewarnaan garis dalam berbagai jenis tanda. Secara umum, struktur penambahan kode warna disisipkan pada fungsi plot seperti pada contoh berikut :

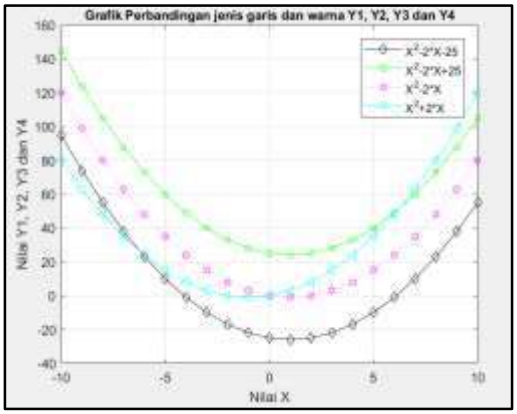
```
>> plot(X,Y1,'kode jenis garis kode dan jenis warna pasangan X Y1', X,Y2, 'kode jenis garis kode jenis warna pasangan X Y2', dan seterusnya);
```

Berikut beberapa kode pewarnaaan dalam membuat grafik.

Kode Jenis Warna	Keterangan
'y'	Pewarnaan grafik dengan menggunakan warna kuning (yellow)
'g'	Pewarnaan grafik dengan menggunakan warna hijau (green)
'c'	Pewarnaan grafik dengan menggunakan warna hijau biru (cyan)
'w'	Pewarnaan grafik dengan menggunakan warna putih (white)
'm'	Pewarnaan grafik dengan menggunakan warna merah cokelat (magenta)
'b'	Pewarnaan grafik dengan menggunakan warna biru (blue)
'k'	Pewarnaan grafik dengan menggunakan warna hitam (black)
'r'	Pewarnaan grafik dengan menggunakan warna merah (red)

Tabel 27. Tabel kumpulan kode pewarnaan grafik

Berikut beberapa contoh penggunaan kode-kode jenis garis yang ditempatkan pada penyajian grafik yang memiliki beberapa himpunan data dalam satu grafik.

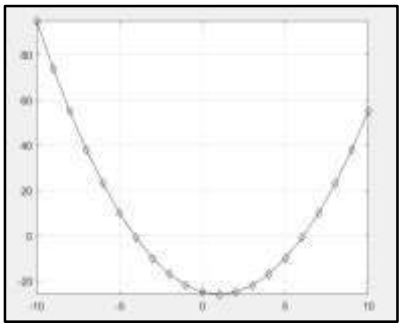
SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE
>> X = -10:10; >> Y1 =X.^2-2*X-25; >> Y2 =X.^2-2*X+25; >> Y3 =X.^2-2*X; >> Y4 =X.^2+2*X; >> plot(X,Y1,'d-k',X,Y2,'s-g',X,Y3,'h:m',X,Y4,'<-c'); >> title('Grafik Perbandingan jenis garis dan warna Y1, Y2, Y3 dan Y4', 'FontSize',10); >> xlabel('Nilai X'); >> ylabel('Nilai Y1, Y2, Y3 dan Y4'); >> grid on >> legend('X^2-2*X-25','X^2-2*X+25','X^2-2*X','X^2+2*X')	

Gambar 5. 12. Penggabungan Grafik dalam satu sistem koordinat

❖ Penanganan Nilai pada Sumbu

Pada contoh-contoh di atas, kita telah mengenal dan memahami penggunaan kode-kode dalam penanganan plotting data dalam grafik. Hal yang lain yang kemudian perlu untuk diperhatikan dalam penyajian grafik adalah, penyajian nilai dari sumbu. Sebelumnya kita hanya melakukan penanganan sumbu dalam hal pemberian label. Namun pada bagian ini yang menjadi fokus perhatian adalah penanganan nilai pada sumbu horizontal dan vertikal.

Dengan menggunakan perintah **axis** yang berguna untuk mengatur sumbu pada grafik. Dalam pengaturan sumbu, dapat dilakukan dengan menyatakan nilai minimum pada sumbu horizontal dan vertikal(xmin) , nilai maksimum pada sumbu horizontal (xmax dan ymax).

Perintah yang dituliskan dalam Command Window	Running Program yang Menyajikan Grafik pada Jendela Figure
>> X = -10:10; >> Y =X.^2-2*X-25; >> xmin =min(X); >> xmax =max(X); >> ymin =min(Y); >> ymax =max(Y); >> plot(X,Y,'d-k'); >> axis([xmin xmax (ymin) (ymax)]); >> grid on	 Gambar 5. 13. Pendefinisian Batas sumbu –x dan sumbu-y

Pada contoh di atas dapat dilihat bahwa dengan menggunakan perintah axis, kita dapat menetapkan batas-batas sumbu sesuai dengan yang dikehendaki. Pada perintah yang dituliskan di command window, nilai-nilai batas maximum dan minimum pada sumbu horizontal dan vertikal harus didefinisikan.

>> X = -10:10; >> Y =X.^2-2*X-25; >> xmin =min(X); >> xmax =max(X); >> ymin =min(Y); >> ymax =max(Y); >> plot(X,Y,'d-k'); >> axis([xmin xmax (ymin) (ymax)]); >> grid on >> axis off	 Gambar 5. 14. Menonaktifkan sumbu-x dan sumbu-y

Berikut beberapa perintah lain yang dapat digunakan dalam mengatur grafik dapat dilihat pada tabel berikut :

BEBERAPA PERINTAH AUTO	KETERANGAN
axis on	Menampilkan sumbu X dan Y
axis off	Menyembunyikan sumbu X dan sumbu Y
axis auto	Mengembalikan ke bentuk standard
axis square	Mengatur panjang sumbu X dan sumbu Y yang dibuat sama
axis normal	Mengembalikan axis square ke bentuk normalnya.

❖ Penggunaan Subplot

Pada contoh-contoh di atas, kita telah melihat penyajian-penyajian grafik dengan menambahkan berbagai properti untuk melengkapi informasi. Namun pada penyajian grafik dengan perintah plot, hanya menampilkan satu pasang sumbu dalam satu jendela figure. Adakalanya kita membutuhkan penyajian beberapa grafik langsung ke dalam satu figure. Pada kasus ini, MATLAB menyediakan perintah subplot untuk menyajikan lebih dari satu sumbu yang ditampilkan dalam satu figure.

Konsep atau cara kerja fungsi subplot dalam menyajikan beberapa grafik dalam satu figure yaitu dengan memandang bahwa satu pasang sumbu itu merupakan satu elemen vektor atau matriks yang menempati indeks tetentu. Dengan konsep ini bentuk dasar dari fungsi subplot yaitu : `subplot(jumlah baris, jumlah kolom, posisi gambar akan diletakkan)`. Setelah itu dilanjutkan dengan fungsi `plot` yang akan menempati sesuai perintah yang diberikan sebelumnya.

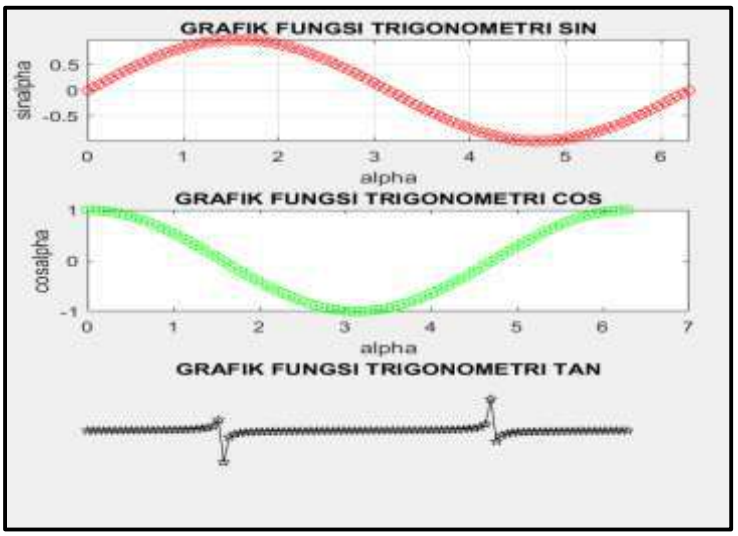
Cara pembacaan posisi pada matriks diurutkan berdasarkan baris. Penempatan posisi grafik sebagai elemen matriks dapat dilihat pada ilustrasi matriks berikut.

Indeks (1,1) sebagai posisi 1	Indeks (1,2) sebagai posisi 2	Indeks (1,3) sebagai posisi 3
Indeks (2,1) sebagai posisi 4	Indeks (2,2) sebagai posisi 5	Indeks (2,3) sebagai posisi 6

Misal perintahnya berupa `subplot(2,3,4)` maka akan menempati posisi yang berwarna abu-abu di atas. Berikut diberikan beberapa contoh penggunaan subplot.

```
PROGRAM BERIKUT MENMBERIKAN CONTOH PENGGUNAAN SUBPLOT
%MISAL PENYAJIAN GRAFIK TRIGONOMETRI SIN COS TAN
alpha    = linspace(0,2*pi);
sinalpha = sin(alpha);
cosalpha = cos(alpha);
tanalpha = tan(alpha);
subplot(3,1,1)
plot(alpha,sinalpha,'r-d');
title('GRAFIK FUNGSI TRIGONOMETRI SIN');
xlabel('alpha');
ylabel('sinalpha');
axis ([min(alpha) max(alpha) min(sinalpha) max(sinalpha)])
```

```
grid on
subplot(3,1,2)
plot(alpha,cosalpha,'g-o');
title('GRAFIK FUNGSI TRIGONOMETRI COS');
xlabel('alpha');
ylabel('cosalpha');
grid off

subplot(3,2, 3)
plot(alpha,tanalpha,'k-p');
title('GRAFIK FUNGSI TRIGONOMETRI TAN');
xlabel('alpha');
ylabel('sinalpha');
grid on
axis off
```

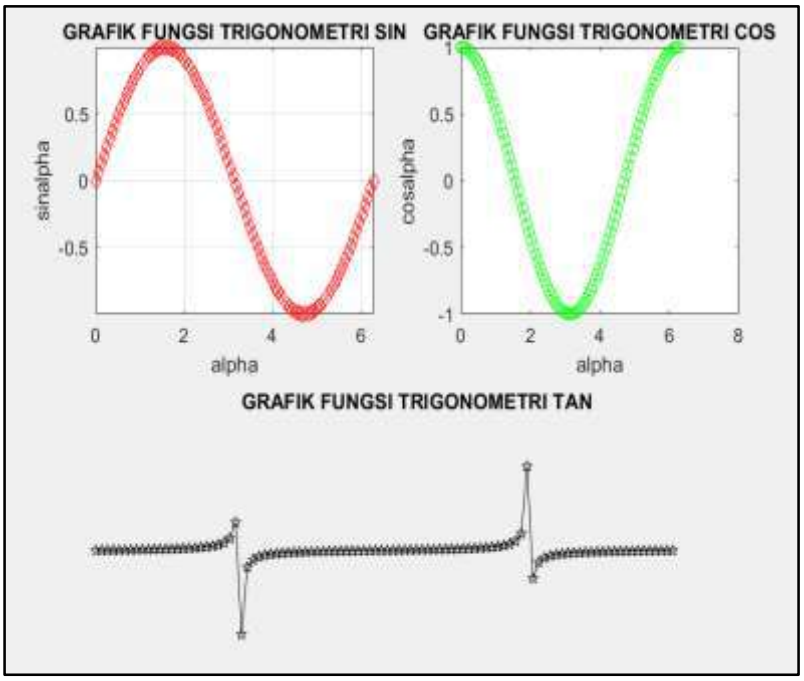


Gambar 5. 15. Hasil penggunaan perintah subplot

Pada program diatas terlihat jelas bagaimana cara penulisan perintah subplot dan bagaimana cara kerja subplot dalam menyajikan beberapa grafik dalam satu jendela figure. Pada contoh tersebut diberikan perintah `subplot(3,2, 1)`, `subplot(3,2, 2)`, dan `subplot(3,2, 3)` pada masing-masing pasangan fungsi secara berurutan, dengan menambahkan beberapa perintah untuk melengkapi properti secara berbeda. Dapat pula dilihat bahwa hasil running programnya, menampilkan susunan grafik yang menempati posisi matriks 3 baris 1 kolom yang berurutan.

```
%PROGRAM BERIKUT MENMBERIKAN CONTOH PENGGUNAAN SUBPLOT
%DALAM MENYAJIKAN BEBERAPA GRAFIK DALAM 1 FIGURE
```

```
%MISAL PENYAJIAN GRAFIK TRIGONOMETRI SIN COS TAN
alpha    = linspace(0,2*pi);
sinalpha = sin(alpha);
cosalpha = cos(alpha);
tanalpha = tan(alpha);

subplot(2,2,1)
plot(alpha,sinalpha,'r-d');
title('GRAFIK FUNGSI TRIGONOMETRI SIN');
xlabel('alpha');
ylabel('sinalpha');
axis ([min(alpha) max(alpha) min(sinalpha) max(sinalpha)])
grid on

subplot(2,2,2)
plot(alpha,cosalpha,'g-o');
title('GRAFIK FUNGSI TRIGONOMETRI COS');
xlabel('alpha');
ylabel('cosalpha');
grid off
subplot(2,2,[3,4])
plot(alpha,tanalpha,'k-p');
title('GRAFIK FUNGSI TRIGONOMETRI TAN');
xlabel('alpha');ylabel('sinalpha');grid on;axis off
```

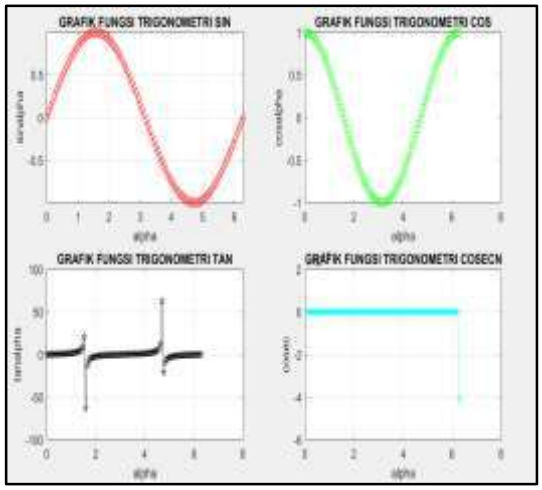


Gambar 5. 16. Penggunaan subplot dengan jumlah susunan yang tak beraturan

Pada contoh ini menunjukkan cara kerja subplot yang berbeda, pada bagian ini plot grafik yang terakhir ditempatkan pada posisi [3 4]. Hal ini berarti

dalam matriks gambar, perintah plot yang ada dibawahnya menempati dua posisi sekaligus. Hal ini dapat kita lihat dari hasil runningnya pada jendela figure.

```
%PROGRAM BERIKUT MENMBERIKAN CONTOH PENGGUNAAN SUBPLOT
%DALAM MENYAJIKAN BEBERAPA GRAFIK DALAM 1 FIGURE
%MISAL PENYAJIAN GRAFIK TRIGONOMETRI SIN COS TAN
alpha    = linspace(0,2*pi);
sinalpha = sin(alpha);
cosalpha = cos(alpha);
tanalpha = tan(alpha);
cosecalpha = csc(alpha);

subplot(2,2,1)
plot(alpha,sinalpha,'r-d');
title('GRAFIK FUNGSI TRIGONOMETRI SIN');
xlabel('alpha');
ylabel('sinalpha');
axis ([min(alpha) max(alpha) min(sinalpha) max(sinalpha)])
grid on

subplot(2,2,2)
plot(alpha,cosalpha,'g-o');
title('GRAFIK FUNGSI TRIGONOMETRI COS');
xlabel('alpha');
ylabel('cosalpha');
grid off

subplot(2,2,3)
plot(alpha,tanalpha,'k-p');
title('GRAFIK FUNGSI TRIGONOMETRI TAN');
xlabel('alpha');
ylabel('tanalpha');
grid on
subplot(2,2,4)
plot(alpha,cosecalpha,'c-p');
title('GRAFIK FUNGSI TRIGONOMETRI COSECN');
xlabel('alpha');
ylabel('cosec');
grid on
```



Gambar 5. 17. Penggunaan perintah subplot secara standar

Pada contoh ini dengan ukuran 2 baris 2 kolom terdiri dari 4 jenis grafik yang ditempatkan pada posisi yang berbeda menunjukkan cara kerja perintah **subplot** secara standar. Setiap perintah **subplot** diikuti perintah perintah yang mengatur properti masing-masing grafik.

D. Diagram Batang

Diagram Batang secara visual merupakan salah satu bentuk penyajian data yang ditampilkan menyerupai jejeran batang yang dapat berdimensi dua atau tiga. Pada bagian sebelumnya fungsi utama yang digunakan adalah fungsi **plot**. Namun pada penyajian diagram batang menggunakan fungsi bar untuk diagram batang dimensi dua, dan bar3 untuk diagram batang tiga dimensi.

Hal utama yang membedakan antara fungsi **plot** dan **bar,** yaitu pada output figure, hasil dari plot menampilkan grafik secara kontinu, sedangkan pada fungsi **bar** , menyajikan pasangan titik secara diskrit. Berikut beberapa fungsi utama dalam MATLAB yang digunakan dalam menyajikan diagram batang.

	Keterangan
	`bar(X,Y)`

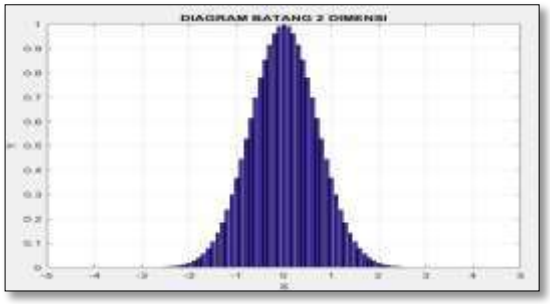
Perintah untuk menggambar diagram batang 2D dan menampilkan datanya pada posisi tegak dengan argumen masukan daerah asal X yang telah didefinisikan sebagai vektor data, dan daerah kawan Y yang merupakan pasangan dari masing-masing anggota X.
`barh(x,y)`
Perintah untuk menggambar diagram batang 2D dan memampilkannya secara horizontal dengan argumen masukan daerah asal X yang telah didefinisikan sebagai vektor data, dan daerah kawan Y yang merupakan pasangan dari masing-masing anggota X
`bar3(x,y)`
Perintah untuk menggambar diagram batang 3D dan menampilkannya secar tegak dengan argumen masukan daerah asal X yang telah didefinisikan sebagai vektor data, dan daerah kawan Y yang merupakan pasangan dari masing-masing anggota X
`bar3(data,'stacked')`
Perintah untuk menggambar diagram batang 3D dan menampilkannya secara tegak dengan argumen masukan berupa Matriks Data dengan bentuk yang menumpuk.
`bar3h(data,'stacked')`
Perintah untuk menggambar diagram batang 3D dan menampilkannya secara horizontal dengan argumen masukan berupa Matriks Data dengan bentuk yang menumpuk.
`bar3h(data,'grouped')`
Perintah untuk menggambar diagram batang 3D dan menampilkannya secara horizontal dengan argumen masukan berupa Matriks Data dengan bentuk yang tergrup pada masing-masing objek.
`colormap(matriks warna)`
Perintah yang digunakan untuk mengatur pewarnaan batang pada diagram batang yang terbentuk.

Tabel 28. Tabel Perintah Penggunaan Diagram Batang

Berikut contoh-contoh penyajian data melalui diagram batang.

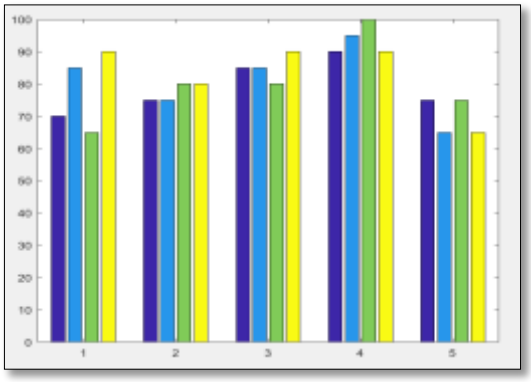
SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PROGRAM PADA JENDELA FIGURE

`>> X = -4:0.1:4;`
`>> Y = exp(-X.*X);`
`>> bar(X,Y);`
`>> title('DIAGRAM BATANG 2 DIMENSI');`
`>> xlabel('X');`
`>> ylabel('Y');`
`>> grid on`	



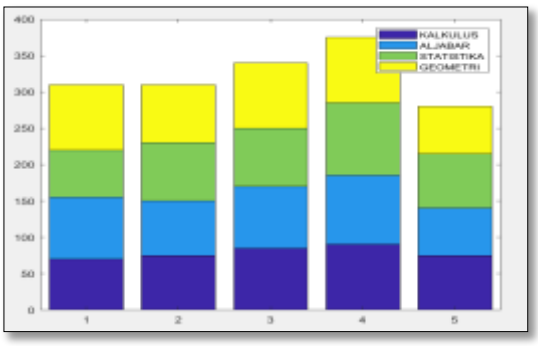
Gambar 5. 18. Hasil dari penggunaan bar

```
>> data = [70 85 65 90;75 75 80
80;85 85 80 90;90 95 100 90;75 65
75 65]

   data =

      70     85     65     90
      75     75     80     80
      85     85     80     90
      90     95    100     90
      75     65     75     65

>> bar(data)
```



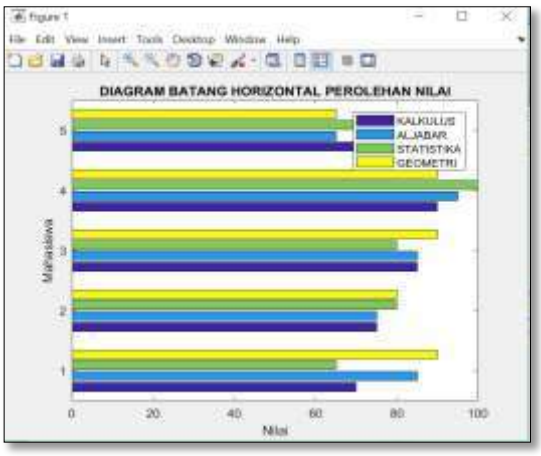
Gambar 5. 19. Hasil dari penggunaan bar dari data yang mengelompok

```
>> data = [70 85 65 90;75 75 80
80;85 85 80 90;90 95 100 90;75 65
75 65]

   data =

      70     85     65     90
      75     75     80     80
      85     85     80     90
      90     95    100     90
      75     65     75     65

>> bar(data,'stacked')
>>legend('KALKULUS',     'ALJABAR',
'STATISTIKA', 'GEOMETRI')
```



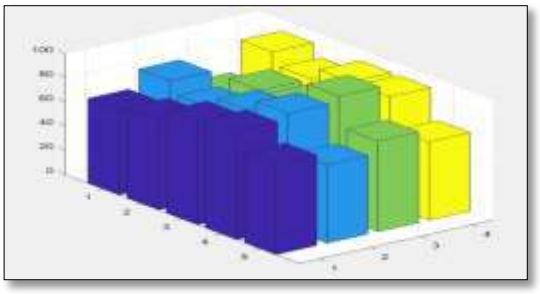
Gambar 5. 20. Penggunaan perintah bar dengan stile stacked

```
>> data = [70 85 65 90;75 75 80
80;85 85 80 90;90 95 100 90;75 65
75 65]

   data =

       70     85     65     90
       75     75     80     80
       85     85     80     90
       90     95    100     90
       75     65     75     65

>> barh(data)
>> title('DIAGRAM BATANG HORIZONTAL
PEROLEHAN NILAI');
>> xlabel ('Nilai');
>> ylabel('Mahasiswa')
>>  legend('KALKULUS',  'ALJABAR',
'STATISTIKA', 'GEOMETRI')
```



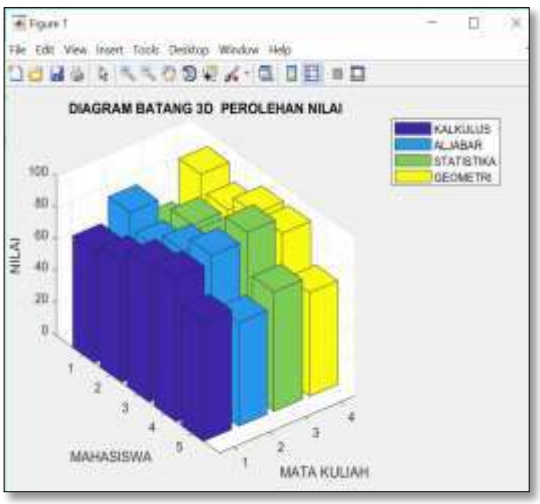
Gambar 5. 21. Penggunaan perintah barh(data) untuk membuat diagram batang dari data berkelompok secara horizontal

```
>> data = [70 85 65 90;75 75 80
80;85 85 80 90;90 95 100 90;75 65
75 65]
   data =

       70     85     65     90
       75     75     80     80
       85     85     80     90
       90     95    100     90
       75     65     75     65
>> bar3(data)
```

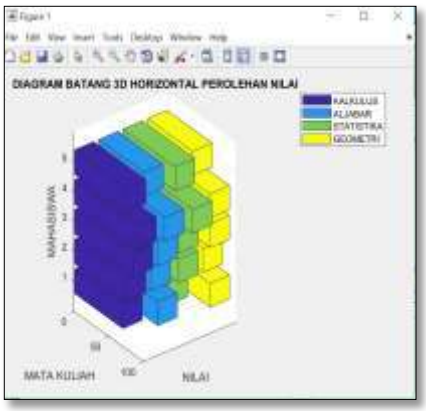


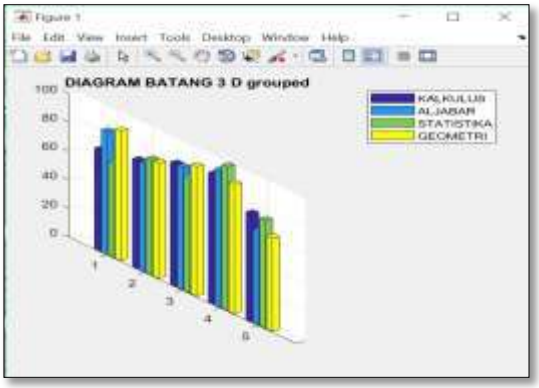
Gambar 5. 22. Penggunaan bar3(data) untuk membuat diagram batang 3 Dimensi secara berkelompok

```
>> data = [70 85 65 90;75 75 80
80;85 85 80 90;90 95 100 90;75 65
75 65]

data =

    70     85     65     90
    75     75     80     80
    85     85     80     90
    90     95    100     90
    75     65     75     65

>> bar3(data)
>> xlabel('MATA KULIAH');
>> ylabel('MAHASISWA');
>> zlabel('NILAI');
>>    title('DIAGRAM    BATANG    3D
PEROLEHAN NILAI');
>>                legend('KALKULUS',
'ALJABAR','STATISTIKA','GEOMETRI')
```

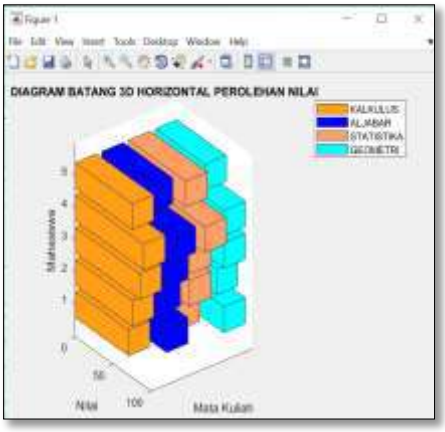


	Gambar 5. 23. Penggunaan bar3(data) untuk membuat diagram batang 3 Dimensi secara berkelompok

`>> data = [70 85 65 90;75 75 80 80;85 85 80 90;90 95 100 90;75 65 75 65]`
 	
`data =`
 	
`70 85 65 90`
`75 75 80 80`
`85 85 80 90`
`90 95 100 90`
`75 65 75 65`
 	
`>> bar3h(data)`
`>> xlabel('MATA KULIAH');`
`>> ylabel('MAHASISWA');`
`>> zlabel('NILAI');`
`>> title('DIAGRAM BATANG 3D PEROLEHAN NILAI');`
`>> legend('KALKULUS','ALJABAR','STATISTIKA','GEOMETRI')`
`>>`
	
	Gambar 5. 24. Penggunaan bar3(data) untuk membuat diagram batang 3 Dimensi secara berkelompok dan ditampilkan horizontal

`>> data = [70 85 65 90;75 75 80 80;85 85 80 90;90 95 100 90;75 65 75 65]`
 	
`data =`
 	
`70 85 65 90`
`75 75 80 80`
`85 85 80 90`
`90 95 100 90`
`75 65 75 65`
 	
`>> bar3(data,'grouped')`
`>> title('DIAGRAM BATANG 3 D grouped');`
`>> legend('KALKULUS','ALJABAR','STATISTIKA','GEOMETRI')`
	
	Gambar 5. 25. Hasil penggunaan perintah bar 3D dengan style grouped

```
>> data = [70 85 65 90;75 75 80
80;85 85 80 90;90 95 100 90;75 65
75 65]

data =

    70    85    65    90
    75    75    80    80
    85    85    80    90
    90    95   100    90
    75    65    75    65

>> bar3h(data)
>>   title('DIAGRAM BATANG 3D
HORIZONTAL PEROLEHAN NILAI');
>>              legend('KALKULUS',
'ALJABAR','STATISTIKA','GEOMETRI')
>> colormap([1 0.62 0.1;0 0 1;1
0.62 0.4;0 1 1])
>> zlabel('Mahasiswa');
>> ylabel('Nilai');
>> xlabel('Mata Kuliah');
```



Gambar 5. 26. Hasil penggunaan pernah color map untuk mengatur warna diagram batang

Dari contoh-contoh diagram batang di atas dapat kita melihat bahwa Perintah dasar yang digunakan adalah Perintah **bar, bar3, barh, bar3h** dan beberapa perintah lainnya. Seperti pada contoh-contoh pada bagian plotting, kita dapat melakukan pengaturan properti sumbu, judul, legenda, pelabelan, dan pengaturan warna. Hal lain yang dapat dijelaskan yaitu pada penginputan data yang dilakukan tidak hanya berupa vektor saja, namun data yang akan disajikan melalui diagram batang dapat berupa Matriks.

Berikut diberikan daftar, pengaturan komposisi nilai dalam menentukan warna yang dapat dipasangkan pada batang-batang yang terbentuk dari diagram batang yang ada. Secara mendasar, range warna yang dapat digunakan bergantung pada nilai R, G dan B yang didefinisikan yang masing-masing berkisar dari 0 sampai 1.

R	G	B	WARNA
0	0	0	Hitam
1	1	1	Putih
1	0	0	Merah
0	1	0	Hijau
0	0	1	Biru
1	1	0	Magenta
0	1	1	Cyan
0.5	0.5	0.5	Abu-abu
0.5	0	0	Merah Tua
1	0.62	0.4	Oranye Tua

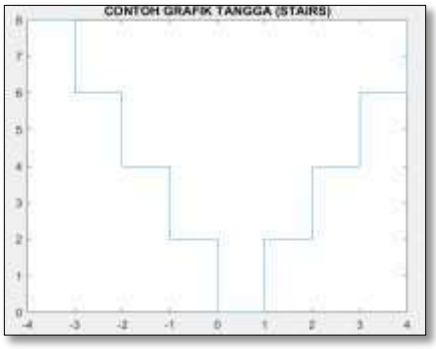
Tabel 29. Tabel Skor warna pada batang-batang diagram

E. Perintah Plot Dua Dimensi

❖ Perintah Stair

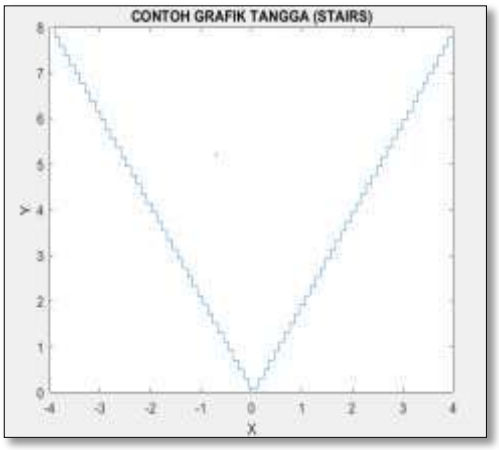
Dalam menyajikan suatu grafik yang menghendaki grafik menyerupai anak tangga, MATLAB menyediakan Perintah **stair** untuk menangani masalah tersebut. Berikut contoh penggunaan Perintah **stair.** Berikut contoh pengguanaan Perintah **stair** dalam menyajikan Perintah tangga.

```
>>      %CONTOH      PENGGUNAAN
PERINTAH STAIR
>> %DEFINISIKAN NILAI X DAN Y
>> X = [-4 -3 -2 -1 0 1 2 3
4];
>> Y = [8 6 4 2 0 2 4 6 8];
>> stairs(X,Y)
>>    title('CONTOH    GRAFIK
TANGGA (STAIRS)')
```



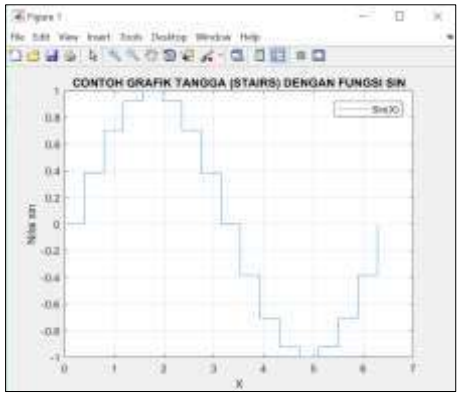
Gambar 5. 27. Hasil penggunaan perintah stairs dari satu pasangan data

```
>> %CONTOH PENGGUNAAN PERINTAH
STAIR
>> %DEFINISIKAN NILAI X DAN Y
>>   %DENGAN   NILAI   YA   YANG
BERGANTUNG PADA NILAI X
>> X = linspace(-4,4,80);
>> Y = abs(X*2);
>> stairs(X,Y)
>> title('CONTOH GRAFIK TANGGA
(STAIRS)')
>> xlabel('X');
>> ylabel('Y');
```



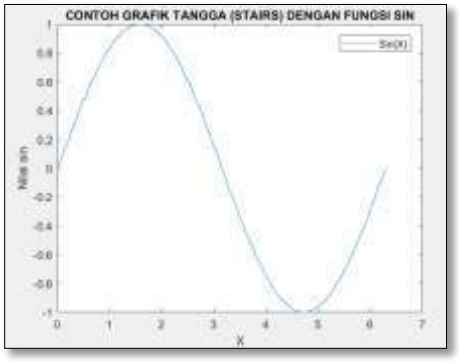
Gambar 5. 28. Hasil penggunaan perintah stairs dari data yang dibangkitkan dengan linspace

```
>> %CONTOH PENGGUNAAN PERINTAH
STAIR YANG LAIN
>> %DEFINISIKAN NILAI X DAN Y
>>  %DENGAN  NILAI  YA  YANG
BERGANTUNG PADA NILAI X
>> X = 0:pi/8:2*pi;
>> Y = sin(X);
>> stairs(X,Y)
>> title('CONTOH GRAFIK TANGGA
(STAIRS)  DENGAN  PERINTAH
SIN');
>> xlabel('X');
>> ylabel('Nilai sin');
>> grid on
>> legend('Sin(X)')
```



Gambar 5. 29. Hasil penggunaan stairs pada fungsi sin

```
>> %CONTOH PENGGUNAAN PERINTAH
STAIR YANG LAIN
>> %DEFINISIKAN NILAI X DAN Y
>>  %DENGAN  NILAI  YA  YANG
BERGANTUNG PADA NILAI X
>> X = 0:pi/100:2*pi;
>> Y = sin(X);
>> stairs(X,Y)
>> title('CONTOH GRAFIK TANGGA
(STAIRS)  DENGAN  PERINTAH
SIN');
>> xlabel('X');
>> ylabel('Nilai sin');
>> legend('Sin(X)')
>>
```



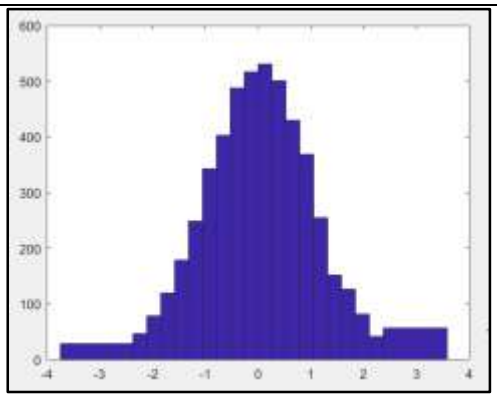
Gambar 5. 30. Penggunaan fungsi stairs dengan grid yang sangat kecil sehingga menyerupai garis biasa

❖ Perintah Hist

Dalam beberapa kasus, adakalanya dibutuhkan penyajian data dalam bentuk histogram. Perintah yang disediakan oleh MATLAB dalam menangani masalah tersebut yaitu Perintah **hist.**

SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE

```
>>          %CONTOH
PENYAJIAN      DATA
DENGAN MENGGUNAKAN
HISTOGRAM
>> %Perintah utama
yang      digunakan
adalah :hist(X,Y)
>> X = linspace(-
2.5,2.5,20);
>>        Y        =
randn(5000,1);
>> hist(X,Y)
Error using histc
Edge vector must
be   monotonically
non-decreasing.
Error    in    hist
(line 121)
    nn           =
histc(y,edgesc,1);

>> hist(Y,X)
>>
```



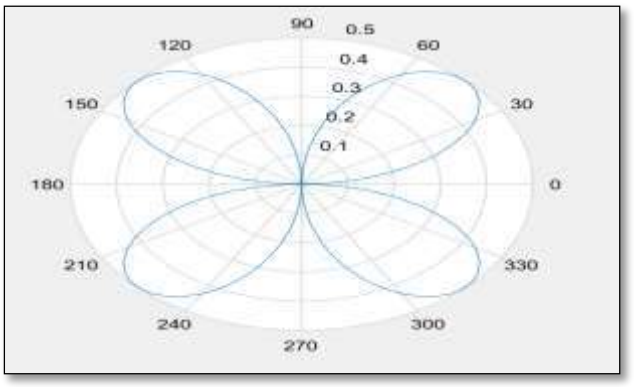
Gambar 5. 31. Penggunaan perintah hist untuk menampilkan histoogram dari data yang dibangkitkan

Pada bagian ini, bentuk grafik yang dihasilkan dari penggunaan Perintah **hist(Y,X)** menyerupai diagram batang yang dihasilkan dari penggunaan bar. Data yang digunakan pada contoh ini, yaitu data yang dibangkitkan dari Perintah. Dapat kita lihat pada baris kelima, perintah yang dimasukkan mengalami kesalahan.

❖ Perintah Polar

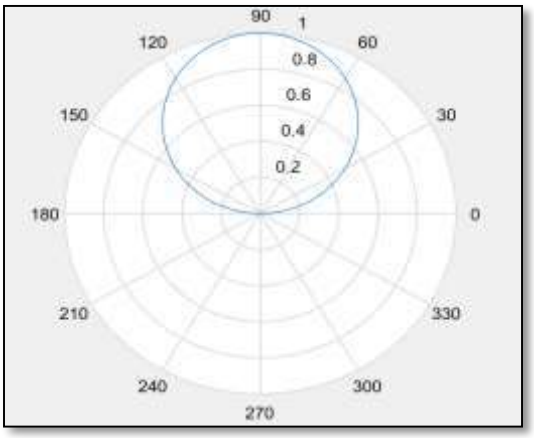
Perintah polar digunakan untuk menyajikan grafik dalam bentuk koordinat polar. Perintah utama yang diguanakan berupa **polar(Sudut, Perintah yang bergantung pada sudut)**.

```
>> %CONTOH PENGGUNAAN PERINTAH POLAR DALAM MENYAJIKAN DATA
DALAM KOORDINAT POLAR
>> Theta = linspace(0,2*pi);
>> Rho = sin(Theta).*cos(Theta);
>> polar(Theta, Rho)
```



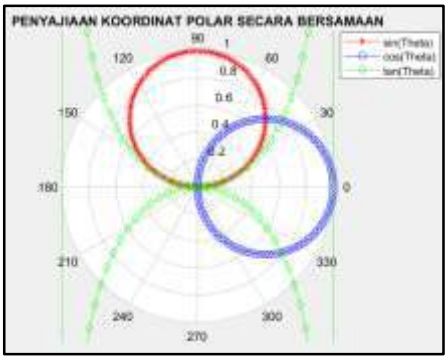
Gambar 5. 32. Hasil penggunaan perintah polar dalam menyajikan grafik pada koordinat polar

```
>> %CONTOH PENGGUNAAN PERINTAH POLAR DALAM MENYAJIKAN DATA
DALAM KOORDINAT POLAR
>> Theta = linspace(0,2*pi);
>> Rho = sin(Theta);
>> polar(Theta, Rho)
```



Gambar 5. 33. Penyajian Koordinat Polar

```
>> %CONTOH PENGGUNAAN PERINTAH POLAR DALAM MENYAJIKAN DATA
DALAM KOORDINAT POLAR
>> Theta = linspace(0,2*pi);
>> Trigonsin = sin(Theta);
>> Trigoncos = cos(Theta);
>> Trigontan = tan(Theta);
>> polar(Theta,Trigonsin,'r-*')
>> hold on
>> polar(Theta,Trigoncos,'b-o')
>> polar(Theta,Trigontan,'g-d')
>> hold off
>> title('PENYAJIAN KOORDINAT POLAR SECARA BERSAMAAN');
>> legend('sin(Theta)', 'cos(Theta)','tan(Theta)')
```



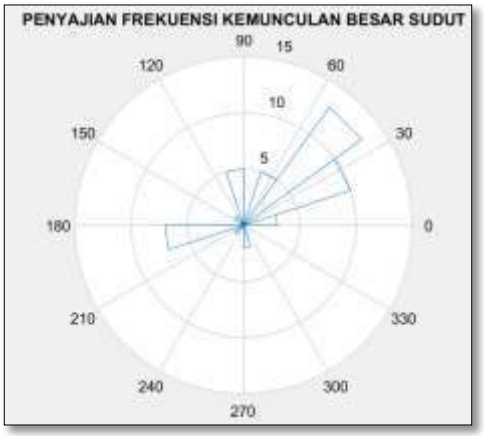
Gambar 5. 34. Penyajian koordinat polar dalam menyajikan fungsi trigonometri secara hold on

Penggunaan Perintah polar tidak lain adalah memplot titik Perintah dengan argumen masukan berupa besar sudut dan nilai yang dihasilkan berupa nilai Perintah trigonometri yang disajikan dalam bentuk likngkarang.

❖ **Perintah Rose**

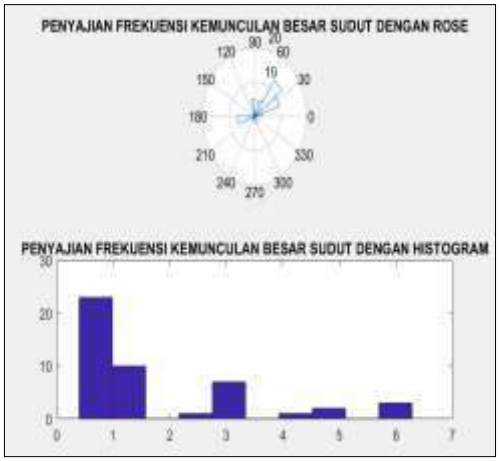
Sebelumnya telah dikenal Perintah **hist** yang menyajikan data dalam bentuk histogram. Bentuk histogram lain yang dapat digunakan adalah histogram sudut. Untuk masalah penyajian histogram sudut dapat digunakan Perintah **rose.**

```
>> %CONTOH PENYAJIAN DATA DENGAN MENGUNAKAN PERINTAH ROSE
>> %MENYATAKAN FREKUENSI KEMUNCULAN SUDUT
>> T = [pi pi pi 2*pi pi/2 pi/2 pi/6 pi/6 pi/5 pi/4 pi/2 pi/4
pi/4 pi/6 pi/6 pi/6 ...
pi/4 pi/4 pi/4 pi/4 pi/4 pi/8 pi/8 pi/8 pi/8 pi/8 pi/2 pi pi pi
pi/2 pi/4 pi/3 pi/3 pi/3 pi/3 pi/3 pi/4 pi/4 pi/4 3*pi/4 pi
3*pi/2 3*pi/2 5*pi/4 2*pi 2*pi];
>> rose(T)
>> title('PENYAJIAN FREKUENSI KEMUNCULAN BESAR SUDUT')
>>
```



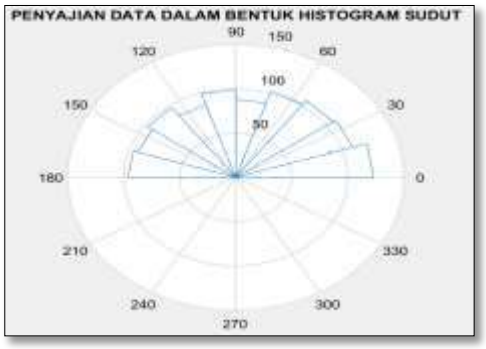
Gambar 5. 35. Hasil penggunaan perintah Rose dalam menampilkan grafik fungsi

```
>> %CONTOH PENYAJIAN DATA DENGAN MENGUNAKAN PERINTAH ROSE
>> %MENYATAKAN FREKUENSI KEMUNCULAN SUDUT
>> T = [pi pi pi 2*pi pi/2 pi/2 pi/6 pi/6 pi/5 pi/4 pi/2 pi/4
pi/4 pi/6 pi/6 pi/6 ...
pi/4 pi/4 pi/4 pi/4 pi/4 pi/8 pi/8 pi/8 pi/8 pi/8 pi/2 pi pi pi
pi/2 pi/4 pi/3 pi/3 pi/3 pi/3 pi/3 pi/4 pi/4 pi/4 3*pi/4 pi
3*pi/2 3*pi/2 5*pi/4 2*pi 2*pi];
>> rose(T)
>> title('PENYAJIAN FREKUENSI KEMUNCULAN BESAR SUDUT')
>> subplot (2,1,1)
>> rose(T)
>>  title('PENYAJIAN FREKUENSI KEMUNCULAN BESAR SUDUT DENGAN
ROSE')
>> subplot(2,1,2)
>> hist(T)
>>  title('PENYAJIAN FREKUENSI KEMUNCULAN BESAR SUDUT DENGAN
HISTOGRAM')
```



Gambar 5. 36. Penggunaan subplot dalam membandingkan frekuensi kemunculan sudut menggunakan rose dan histogram

```
>> %CONTOH PENYAJIAN DATA DENGAN MENGUNAKAN PERINTAH ROSE
>> T = rand(1000,1)*pi;
>> rose(T)
>> title('PENYAJIAN DATA DALAM BENTUK HISTOGRAM SUDUT')
```



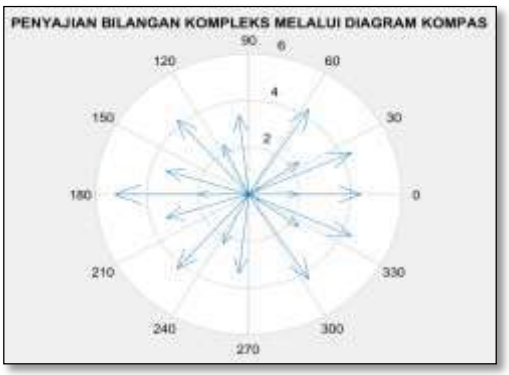
Gambar 5. 37. Penggunaan perintah rose dalam penyajian data dalam bentuk sudut

Pada bagian ini, Perintah rose menunjukkan penyajian frekunsi sudut dalam bentuk diagram lingkaran yang dapat pula disajikan dalam bentuk histogram.

❖ Perintah Compass

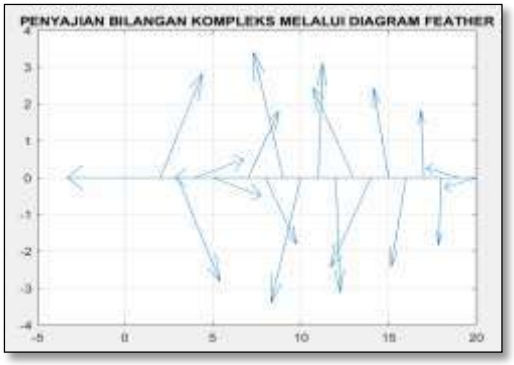
Penyajian data yang lain dalam format 2D yaitu dengan menggunakan bentuk kompas/ Perintah yang digunakan yaitu **compass.** Sedangkan penggunaan **feather** dikhususkan pada penyajian data yang menggambarkan magnitude dan sudut blangan-bilangan kompleks terhadap titik pusat dan dilengkapi dengan panah di ujung nya. Berikut diberikan contoh peggunaaan **compass** dan **feather** dengan hasil gambarnya.

```
>> %CONTOH PENYAJIAN DATA MELALUI PERINTAH COMPASS
>> %MEMBANGKITKAN MATRIKS 20 x 20 KEMUDIAN MENCARI NILAI EIGENNYA
>> Z = randn(20,20);
>> eigZ = eig(Z);
>> compass(eigZ)
>>       title('PENYAJIAN BILANGAN KOMPLEKS MELALUI DIAGRAM KOMPAS');
```



Gambar 5. 38. Penggunaan perintah kompas dalam menyajikan nilai eigen dari suatu vektor

```
>> %CONTOH PENYAJIAN DATA
MELALUI PERINTAH FEATHER
>> %MEMBANGKITKAN MATRIKS
20 x 20 KEMUDIAN MENCARI
NILAI EIGENNYA
>> Z = randn(20,20);
>> eigZ = eig(Z);
>> feather(eigZ)
>>       title('PENYAJIAN
BILANGAN KOMPLEKS MELALUI
DIAGRAM FEATHER');
>> grid on
```



Gambar 5. 39. Penggunaan perintah feather dalam menyajikan bilangan kompleks

❖ Perintah Stem

Perintah **stem** merupakan Perintah yang digunakan untuk menyajikan data dalam bentuk garis-garis tegak diskret untuk setiap data.

Berikut contoh penggunaan Perintah **stem**

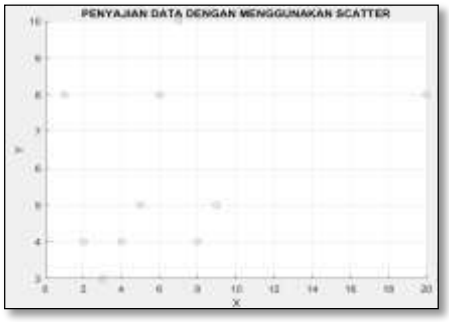
```
>>    %CONTOH    PENGUNAAN
STEM
>> Y = randn(25,1);
>> stem(Y,':')
>> title('PENYAJIAN DATA
DENGAN        MENGGUNAKAN
STEM')
```

Gambar 5. 40. Penyajian data dengan menggunakan perintah stem

❖ Perintah scatter

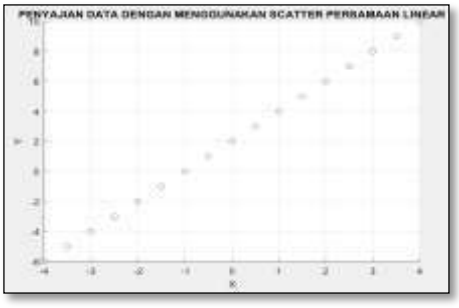
Penyajian data pada koordinat cartesius 2D yang lain adalah **scatter(X,Y).** Perintah **scatter** cara kerjanya hampir sama dengan Perintah 2 Dimensi yang lain, yang memasangkan nilai X dan Y dan ditandai dengan lingkaran kecil disetiap pasangan titiknya seperti . Berikut diberikan contoh penggunaan scatter dari inputan data X dan Y

```
>> X = [1 2 3 4 5 6 7 8 9
20];
>> Y =[8 4 3 4 5 8 10 4 5
8];
>> scatter(X,Y);
>>  title('PENYAJIAN DATA
DENGAN MENGGUNAKAN
SCATTER');
>> xlabel('X');
>> ylabel('Y');
>> grid on
```



Gambar 5. 41. Penyajian data dengan menggunakan perintah scatter

```
>> X = -4:0.5:4;
>> Y = 2*X + 2;
>> scatter(X,Y);
>>  title('PENYAJIAN DATA
DENGAN MENGGUNAKAN SCATTER
PERSAMAAN LINEAR');
>> xlabel('X');
>> ylabel('Y');
>> grid on
```



Gambar 5. 42. Penyajian data melalui pendefinisian persamaan dengan menggunakan perintah scatter

```
>> X = -4:0.5:4;
>> Y = X.^2 + 2;
>> Y = X.^2 + 2*X+1;
>> scatter(X,Y);
>> xlabel('X');
>>  title('PENYAJIAN DATA
DENGAN MENGGUNAKAN SCATTER
PERSAMAAN KUADRAT');
>> ylabel('Y');>> grid on
```

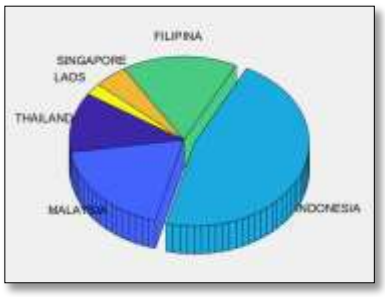
Gambar 5. 43. Penggunaa scatter dalam memplot fungsi kuadrat

F. Diagram Lingkaran

Pie atau kue, merpakan salah satu bentuk penyajian data yang menyerupai bentuk potongan-potongan kue. Biasanya **pie** paling cocok untuk merepresentasikan persentase suatu objek tertentu. Besar

persentase tersebut menentukan bentuk besaran potongan kue tersebut. Berikut contoh penggunaan Perintah **pie.**

SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE
>> %CONTOH PENGGUNAAN PERINTAH PIE >> X = [25 25 25 25]; >> pie(X) >> title('PENYAJIAN DIAGRAM LINGKARAN') >> legend('TARBIYAH', 'SYARIAH', 'DAKWAH DAN KOMUNIKASI','EKONOMI')	 Gambar 5. 44. Penyajian data dalam diagram lingkaran dengan perintah Pie tipe 1
>> %CONTOH PENGGUNAAN PERINTAH PIE >> X = [25 40 100 35 10 5]; >> pie(X) >> title('PENYAJIAN DIAGRAM LINGKARAN') >> legend('THAILAND', 'MALAYSIA', 'INDONESIA','FILIPINA','SINGAPORE','LAOS')	 Gambar 5. 45. Penyajian diagram lingkaran dengan menggunakan perintah pie dilengkapi dengan legenda tipe 2

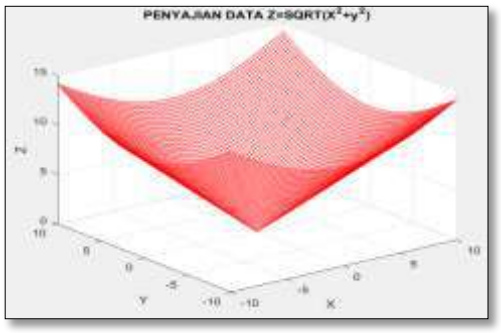
> %CONTOH PENGGUNAAN PERINTAH PIE >> X = [25 40 100 35 10 5]; >> pie(X,[1 0 1 0 1 0],{'THAILAND', 'MALAYSIA', 'INDONESIA', 'FILIPINA', 'SINGAPORE','LAOS'}); >> title('PENYAJIAN DIAGRAM LINGKARAN')	 Gambar 5. 46. Penyajian diagram lingkaran dengan menggunakan perintah pie tipe 3
>> %CONTOH PENGGUNAAN PERINTAH PIE >> X = [25 40 100 35 10 5]; >> pie3(X) >> title('PENYAJIAN DIAGRAM LINGKARAN 3 DIMENSI') >> legend('THAILAND', 'MALAYSIA', 'INDONESIA','FILIPINA','SINGAPORE','LAOS') >>	 Gambar 5. 47. Penyajian data dalam diagram lingkaran dengan menggunakan pie 3D dan legenda tipe 1
>> %CONTOH PENGGUNAAN PERINTAH PIE >> X = [25 40 100 35 10 5]; >> pie3(X,[0 0 1 0 0 0]) >> pie3(X,[0 0 1 0 0 0],{'THAILAND', 'MALAYSIA', 'INDONESIA', 'FILIPINA', 'SINGAPORE', 'LAOS'});	 Gambar 5. 48. Penyajian data dalam diagram lingkaran dengan menggunakan pie 3D dan legenda tipe 2

Tabel 30. Penyajian data melalui lingkaran dengan menggunakan perintah pie dan pie3D

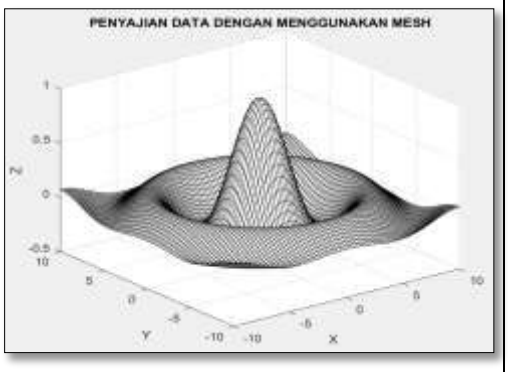
G. Penyajian Grafik 3 Dimensi

❖ Mesh

Mesh merupakan salah satu Perintah yang digunakan untuk menvisualisasikan himpunan data 3 dimensi. Jadi dalam setiap satu titik dibentuk oleh tiga titik yang berpasanagan. Secara visual, gambar yang dihasilkan menyerupai jaring yang terbuka. Secara umum bentuk penulisan Perintah tersebut membutuhkan pendefinisian grid awal sumbu dengan perintah **meshgrid** . Sedangkan Perintah **mesh(X,Y,Z)** membutuhkan tiga argumen input yaitu X dan Y merupakan Sumbu Horizontal yang saling berpotongan, serta Z berupa sumbu vertikal. Berikut akan diberikan beberapa contoh penggunaan Perintah **meshgrid** dan **mesh** dalam memvisualisasikan gambar 3 Dimensi yang dibentuk dari beberapa jenis persamaan**, dapat dilihat pada tabel berikut :**

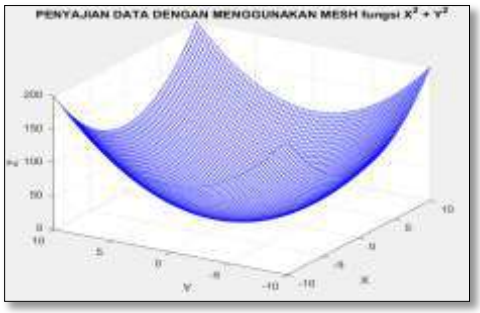
SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE
>> %CONTOH PENGGUNAAN PERINTAH MESH >> [X,Y] = meshgrid(-10:0.25:10); >> Z = sqrt(X.^2+Y.^2); >> mesh(X,Y,Z,'EdgeColor','red'); >> title('PENYAJIAN DATA Z=SQRT(X^2+y^2') >> xlabel(X); >> ylabel(Y); >> zlabel(Z);	 Gambar 5. 49. Penyajian grafik dengan menggunakan perintah mesh

```
>>     [X,Y]     =     meshgrid(-
10:0.25:10);
>> R = sqrt(X.^2+Y.^2);
>> Z = sin(R)./R;
>>
mesh(X,Y,Z,'EdgeColor','black')
;
>> xlabel(X);
>> ylabel(Y);
>> zlabel(Z);
>> title('PENYAJIAN DATA DENGAN
MENGGUNAKAN MESH')
```



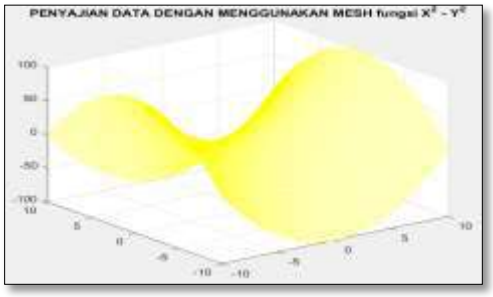
Gambar 5. 50. Penyajian grafik dengan menggunakan mesh berwarna dasar hitam pada fungsi $Z = \sin(R).\frac{1}{R}$

```
>> %CONTOH PENGGUNAAN PERINTAH
MESH
>>     [X,Y]     =     meshgrid(-
10:0.25:10);
>> Z = (X.^2+Y.^2);
>>
mesh(X,Y,Z,'EdgeColor','blue');
>> title('PENYAJIAN DATA DENGAN
MENGGUNAKAN MESH Perintah X^2 +
Y^2')
>> xlabel(X);
>> ylabel(Y);
>> zlabel(Z);
```



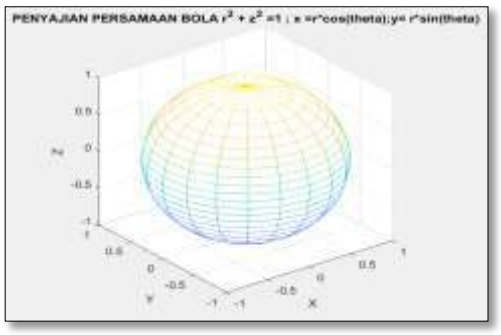
Gambar 5. 51. Penyajian grafik dengan menggunakan perintah mesh$\boldsymbol{Z = x^2 + y^2}$

```
>> %CONTOH PENGGUNAAN PERINTAH
MESH
>>     [X,Y]     =     meshgrid(-
10:0.25:10);
>> Z = (X.^2-Y.^2);
>>
mesh(X,Y,Z,'EdgeColor','yellow'
);
>> title('PENYAJIAN DATA DENGAN
MENGGUNAKAN MESH Perintah X^2 -
Y^2')
>>
```



Gambar 5. 52. Penyajian grafik dengan menggunakan perintah mesh
$\boldsymbol{Z = x^2 - y^2}$

```
>> [theta , Z] = meshgrid((0:0.1:2)*pi,(-1:0.1:1));
>> X = sqrt(1-Z.^2).*cos(theta);
>> Y = sqrt(1-Z.^2).*sin(theta);
>> mesh(X,Y,Z);
>> axis square;
>> title('PENYAJIAN PERSAMAAN BOLA r^2 + z^2 =1 ; x =r*cos(theta);y= r*sin(theta)')
>> xlabel(X);
>> ylabel(Y);
>> zlabel(Z);
```



Gambar 5. 53. Penyajian grafik dari persamaan bola dengan menggunakan perintah mesh

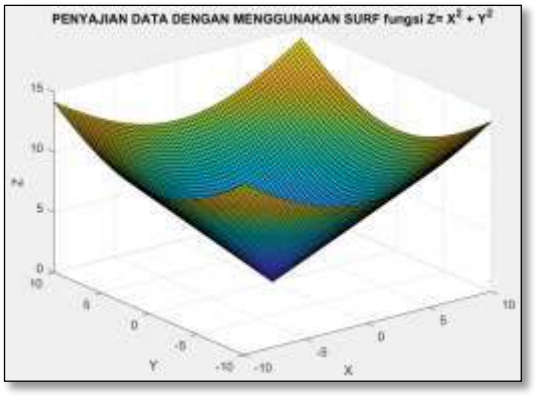
Tabel 31. Contoh Penggunaan mesh dalam menyajikan Grafik 3D

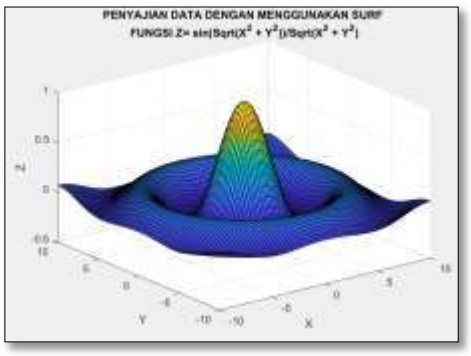
Pada contoh-contoh diatas menunjukkan penggunaan Perintah `meshgrid` dan Perintah `mesh`. Dimana `meshgrid` digunakan untuk membuat grid awal pada sumbu X dan Y sekaligus mendefinisikan nilai X dan Y sebagai inputan untuk Perintah X. Dengan Nilai X,Y,Z tersebut , kemudian digunakan untuk menggambar setiap titik pertemuan setiap elemen X,Y, dan Z dengan perintah `mesh(X,Y,Z)`. Dimana X dan Y menempati sumbu horizontal yang saling memotong sedangkan Y merupakan daerah hasil yang menempati posisi vertikal. Adapun properti warna yang berbeda pada beberapa hasil running di atas diperoleh dari pendefinisian langsung pada Perintah `mesh`.

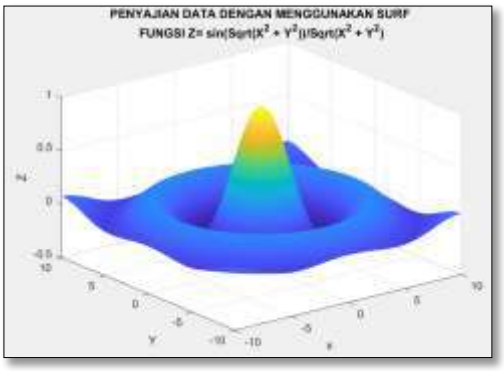
❖ Surf

Jenis visualisasi data 3 dimensi yang telah diperkenalkan sebelumnya yaitu dengan Perintah mesh memberikan gambar berupa jaring pengawatan. Pada bagian ini, akan diperkenalkan Perintah **surf** yang cara kerjanya hampir sama dengan Perintah **mesh.** Hanya saja pada Perintah **surf** ini membuat gambar tiga dimensi yang utuh, dalam hal ini sisi sisi jaring yang terbentuk juga diwarnai. Berikut beberapa contoh hasil visualisasi data dengan menggunakan Perintah **surf.**

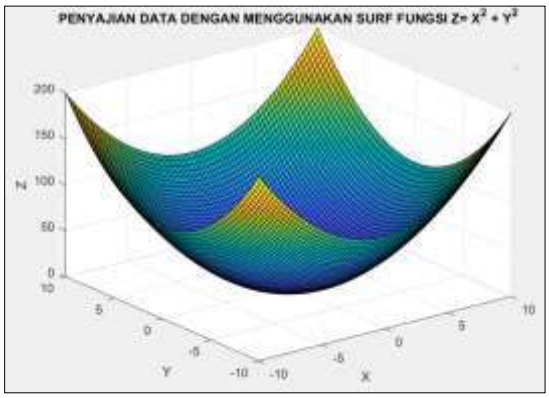
SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE

`>> [X,Y] = meshgrid(-10:0.25:10);`
`>> Z = sqrt(X.^2+Y.^2);`
`>> surf(X,Y,Z);`
`>> title('PENYAJIAN DATA DENGAN MENGGUNAKAN SURF Perintah Z= X^2 + Y^2')`
`>> xlabel(X);`
`>> ylabel(Y);`
`>> zlabel(Z);`
	
	Gambar 5. 54. Penyajian grafik 3D mwnggunakan perintah Surf fungsi `Z = sqrt(X.^2+Y.^2);`

`>> [X,Y] = meshgrid(-10:0.25:10);`
`>> R = sqrt(X.^2+Y.^2);`
`>> Z = sin(R)./R;`
`>> surf(X,Y,Z);`
`>> title('PENYAJIAN DATA DENGAN MENGGUNAKAN SURF PERINTAH Z= sin(Sqrt(X^2 + Y^2))/Sqrt(X^2 + Y^2)')`
`>> xlabel(X);>> ylabel(Y);`
`>> zlabel(Z);`
	
	Gambar 5. 55. Penyajian grafik 3D dengan menggunkan perintah surf

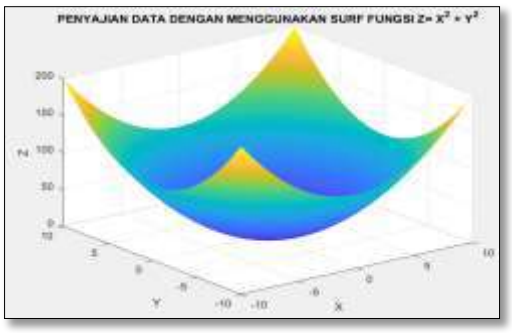
`>> [X,Y] = meshgrid(-10:0.25:10);`
`>> R = sqrt(X.^2+Y.^2);`
`>> Z = sin(R)./R;`
`>> surf(X,Y,Z);`
`>> title('PENYAJIAN DATA DENGAN MENGGUNAKAN SURF PERINTAH Z= sin(Sqrt(X^2 + Y^2))/Sqrt(X^2 + Y^2)')`
`>> xlabel(X);>> ylabel(Y);`
`>> zlabel(Z);`
`>> shading interp;`
	
	Gambar 5. 56. Penggunaan perintah surf dengan menahan tampilan jaring melalui perintah shading interp

```
>> [X,Y] = meshgrid(-
10:0.25:10);
>> Z = sqrt(X.^2+Y.^2);
>> Z = (X.^2+Y.^2);
>> surf(X,Y,Z);
>> title('PENYAJIAN DATA
DENGAN MENGGUNAKAN SURF
PERINTAH Z= X^2 + Y^2')
>> xlabel(X);
>> ylabel(Y);
>> zlabel(Z);
```



Gambar 5. 57. Penggunaan perintah surf tanpa tambahan shading interp pada $\boldsymbol{Z = x^2 + y^2}$

```
>> [X,Y] = meshgrid(-
10:0.25:10);
>> Z = sqrt(X.^2+Y.^2);
>> Z = (X.^2+Y.^2);
>> surf(X,Y,Z);
>> title('PENYAJIAN DATA
DENGAN MENGGUNAKAN SURF
PERINTAH Z= X^2 + Y^2')
>> xlabel(X);
>> ylabel(Y);
>> zlabel(Z);
>> shading interp;
```



Gambar 5. 58. Penggunaan perintah surf tanpa shadng interp pada fungsi $\boldsymbol{Z = x^2 + y^2}$

```
>> [X,Y] = meshgrid(-
10:0.25:10);
>> Z = (X.^2-Y.^2);
>> surf(X,Y,Z);
>> title('PENYAJIAN DATA
DENGAN MENGGUNAKAN SURF
PERINTAH Z= X^2 - Y^2')
>> xlabel(X);
>> ylabel(Y);
>> zlabel(Z);
```

Gambar 5. 59. Penggunaan perintah surf tanpa shadng interp pada fungsi $\boldsymbol{Z = x^2 - y^2}$ dengan nilai pasangan $(\boldsymbol{x, y})$ didefinisikan melalui meshgrid

`>> [theta , Z] = meshgrid((0:0.1:2)*pi,(-1:0.1:1));`
`>> X = sqrt(1-Z.^2).*cos(theta);`
`>> Y = sqrt(1-Z.^2).*sin(theta);`
`>> surf(X,Y,Z);`
`>> title('PENYAJIAN DATA DENGAN MENGGUNAKAN SURF PERSAMAAN BOLA r^2 + z^2 =1 ; x =r*cos(theta);y= r*sin(theta)')`
`>> xlabel(X);`
`>> ylabel(Y);`
`>> zlabel(Z);`
	
	Gambar 5. 60. Penggunaan surf dalam menyajikan grafik persamaan bola tanpa shading interp

`>> [theta , Z] = meshgrid((0:0.1:2)*pi,(-1:0.1:1));`
`>> X = sqrt(1-Z.^2).*cos(theta);`
`>> Y = sqrt(1-Z.^2).*sin(theta);`
`>> surf(X,Y,Z);`
`>> title('PENYAJIAN DATA DENGAN MENGGUNAKAN SURF PERSAMAAN BOLA r^2 + z^2 =1 ; x =r*cos(theta);y= r*sin(theta)')`
`>> xlabel(X);`
`>> ylabel(Y);`
`>> zlabel(Z);`
`>> shading interp;`
	
	Gambar 5. 61. Penggunaan surf dalam menyajikan grafik persamaan bola tanpa shading interp

Tabel 32. Script dan tampilan hasil dari penggunaan perintah surf, dan shading hading

Dari beberapa contoh penggunaan Perintah surf, terlihat bahwa cara kerjanya dalam menyajikan data hampir sama dengan menggunakan Perintah **mesh.** Perbedaan yang mendasar antara Perintah **mesh** dengan

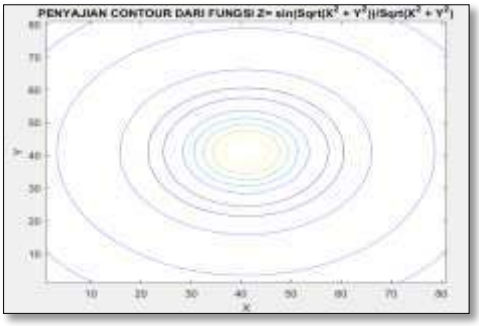
surf yaitu pada aspek warna. Pewarnaan pada **mesh** hanya pada ruas-ruas jaring, sedangkan sisi-si yang terbentuk dari pertemuan antara ruas jaring tidak diberi warna. Selanjutnya pewarnaan dengan menggunakan **surf,** bagian sisi yang terbentuk oleh pertemuan antar ruas-ruas jaring juga diwarnai dengan gradasi warna sesuai dengan nilai sumbu vertikal yang diperoleh. Pada beberapa contoh di atas juga diberi perintah **shading interp** yang digunakan untuk menginterpolasi warna pada permukaan objek. Dengan penambgahan perintah ini, terlihat bahwa ruas-ruas jaring tidak ditampilkan namun terinterpolasi oleh warna dari permukaan objek.

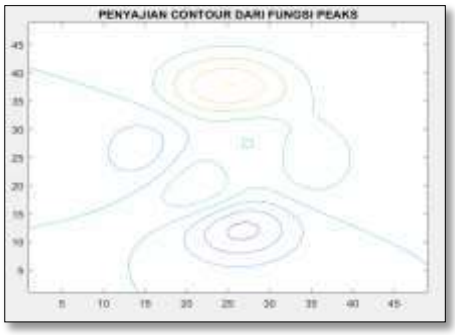
❖ Contour

Penyajian data selanjutnya yaitu **contour,** Perintah **contour** digunakan untuk mensketsakan garis-garis yang menghubungkan nilai yang sama. Pada gambar-gambar yang disajikan dengan menggunakan Perintah **mesh** dan **surf** , terlihat bahwa terdapat pasangan X, Y yang berbeda berada pada nilai Z yang sama secara vertikal, dengan perintah **contur** nilai-nilai yang sama pada ketinggian Z akan dihubungkan sehingga tersketsa beberapa garis kontur. Berikut diberikan contoh penyajian kontur dengan menggunakan persamaan-persamaan yang serupa.

SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE

`>> [X,Y] = meshgrid(-10:0.5:10);`
`>> Z = sqrt(X.^2+Y.^2);`
`>> contour(Z)`
`>> title('PENYAJIAN CONTOUR DENGAN MENGGUNAKAN PERINTAH Z=sqrt(X.^2+Y.^2)');`
`>> grid on`
`>> xlabel('X');`
`>> ylabel('Y');`
`>>`
	
	Gambar 5. 62. Penggunaan perintah contour

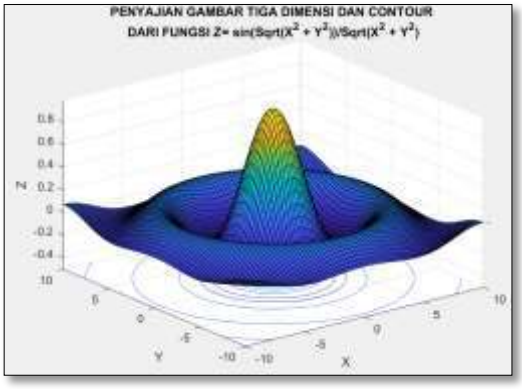
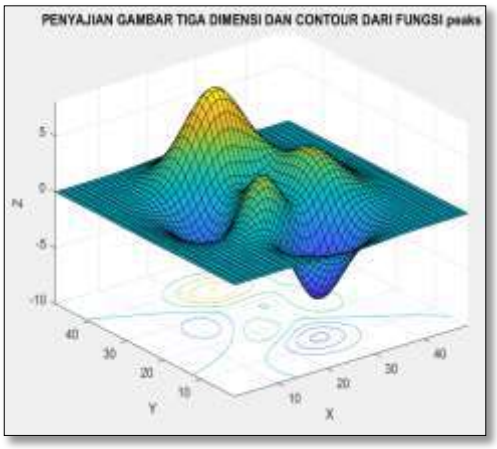
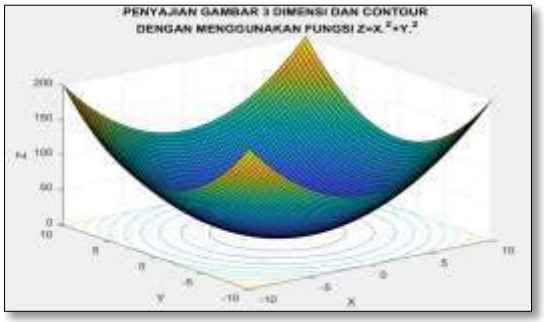
`>> [X,Y] = meshgrid(-10:0.25:10);`
`>> R = sqrt(X.^2+Y.^2);`
`>> Z = sin(R)./R;`
`>> contour(Z)`
`>> title('PENYAJIAN CONTOUR DARI PERINTAH Z= sin(Sqrt(X^2 + Y^2))/Sqrt(X^2 + Y^2)')`
`>> xlabel('X');`
`>> ylabel('Y');`
	
	Gambar 5. 63. Penggunaaan Perintah Contour 2

`>> peaks`
`z = 3*(1-x).^2.*exp(-(x.^2) - (y+1).^2) - 10*(x/5 - x.^3 - y.^5). *exp(-x.^2-y.^2) - 1/3*exp(-(x+1).^2 - y.^2)`
`>> contour(peaks)`
`>> title('PENYAJIAN CONTOUR DARI PERINTAH PEAKS')`
	
	Gambar 5. 64. Penggunaan perintah contour

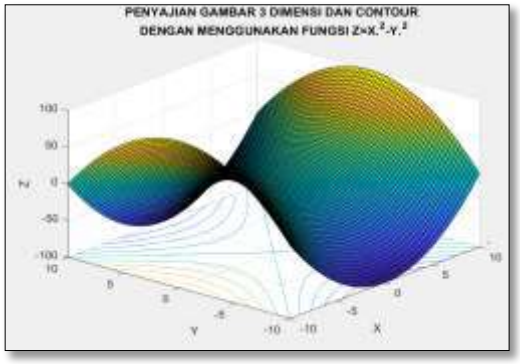
Tabel 33. Contoh-contoh penggunaan perintah contour

❖ Surfc

Dengan mengetahui Perintah surf dan contour, maka kita dapat menyimpulkan bahwa pembentukan contour merupakan representasi dua dimensi dari Perintah surf pada nilai-nilai yang sama. Pada bagian ini akan digabungkan hasil dari surf dan contour dengan menggunakan Perintah surfc. Berikut beberapa contoh penggunaan **surfc** pada persamaan yang sama sebelumnya :

SCRIPT PROGRAM PADA COMMAND WINDOW	HASIL RUNNING PADA JENDELA FIGURE
>> [X,Y] = meshgrid(-10:0.25:10); >> R = sqrt(X.^2+Y.^2); >> Z = sin(R)./R; >> surfc(X,Y,Z) >> title('PENYAJIAN GAMBAR TIGA DIMENSI DAN CONTOUR DARI PERINTAH Z= sin(Sqrt(X^2 + Y^2))/Sqrt(X^2 + Y^2)') >> xlabel('X'); >> ylabel('Y'); >> zlabel('Z');	 Gambar 5. 65. Penggunaan perintah surfc
>> peaks z = 3*(1-x).^2.*exp(-(x.^2) - (y+1).^2) ... - 10*(x/5 - x.^3 - y.^5).*exp(-x.^2-y.^2) ... - 1/3* exp(-(x+1).^2 - y.^2) >> surfc(peaks) >> title('PENYAJIAN GAMBAR TIGA DIMENSI DAN CONTOUR DARI PERINTAH peaks') >> xlabel('X'); >> ylabel('Y'); >> zlabel('Z');	 Gambar 5. 66. Penggunaan perintah surfc yang menggabungkan grafik 3D dan contour
>> [X,Y] = meshgrid(-10:0.25:10); >> Z = (X.^2+Y.^2); >> surfc(X,Y,Z) >> title('PENYAJIAN GAMBAR 3 DIMENSI DAN CONTOUR DENGAN MENGGUNAKAN PERINTAH (X.^2+Y.^2)'); >> xlabel('X'); >> ylabel('Y'); >> zlabel('Z');	 Gambar 5. 67. Penggunaan perintah surfc pada fungsi $\boldsymbol{Z = X^2 + Y^2}$

```
>> [X,Y] = meshgrid(-10:0.25:10);
>> Z = (X.^2-Y.^2);
>> surfc(X,Y,Z)
>> title('PENYAJIAN GAMBAR 3 DIMENSI DAN CONTOUR DENGAN MENGGUNAKAN PERINTAH Z=X.^2-Y.^2');
>> xlabel('X');
>> ylabel('Y');
>> zlabel('Z');
```



Gambar 5. 68. Penggunaan perintah surfc yang menggabungkan grafik 3D dan contour pada fungsi $Z = X^2 - Y^2$

Tabel 34. Contoh penggunaan perintah surfc dalam menggabungkan permukaan 3D dan contour

Dari contoh-contoh tersebut, dapat dijelaskan bahwa gambar 3 Dimensi dan contour dapat digabungkan kedalam satu grafik. Melalui perpaduan ini, kita dapat melihat pola kemiripan data dengan inputan yang berbeda.

H. Penulisan Simbol-Simbol Khusus

Pada beberapa contoh sebelumnya kita menyatakan besar sudut dengan **theta** dalam memberikan judul, atau pada pelabelan sumbu. Namun pada kenyataannnya **theta** merupakan salah satu simbol khusus yang mempunyai tanda tersendiri, sebagai huruf latin. Masalah tersebut dapat diatasi oleh MATLAB dengan menyediakan beberapa perintah dalam menuliskan simbol simbol khusus. Berikut daftar-daftar simbol yang dituliskan dalam memunculkan huruf-huruf Latin yunani.

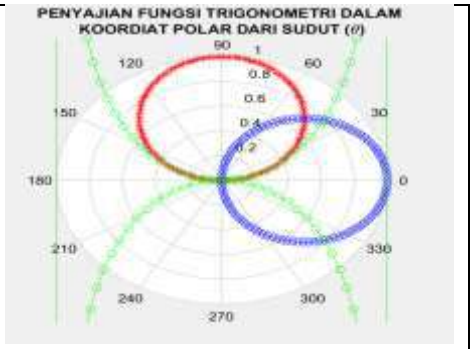
Simbol		Simbol		Simbol		Simbol	
\Delta	Δ	\circ	$\circ$	\kappa	κ	\rho	ρ
\Gamma	Γ	\clubsuit	$\clubsuit$	\lambda	λ	\righttarrow	$\rightarrow$
\Im	$\Im$	\cong	$\cong$	\leftarrow	$\leftarrow$	\sigma	σ
\Lambda	Λ	\cup	$\cup$	\leftrightarrow	$\leftrightarrow$	\sim	$\sim$
\Omega	Ω	\delta	δ	\leq	$\leq$	\spadesuit	$\spadesuit$
\Phi	Φ	\diamondsult	$\diamond$	\mu	μ	\subset	$\subset$
\Pi	Π	\div	$\div$	\neq	$\neq$	\subseteq	$\subseteq$
\Psi	Ψ	\downarrow	$\downarrow$	\ni	$\ni$	\supset	$\supset$

\Re	$\Re$	\epsilon	ε	\nu	ν	\supseteq	$\supseteq$
\Sigma	Σ	\equiv	$\equiv$	\o	o	\tau	τ
\Theta	Θ	\eta	η	\omega	ω	\theta	θ
\Upsilon	Υ	\exist	$\exists$	\oplus	$\oplus$	\uparrow	$\uparrow$
\Xi	Ξ	\forall	$\forall$	\oslash	$\oslash$	\upsilon	υ
\aleph	$\aleph$	\gamma	γ	\otimes	$\otimes$	\varsigma	ς
\alpha	α	\geq	$\geq$	\partial	∂	\vartheta	ϑ
\approx	$\approx$	\heartsult	$\heartsuit$	\phi	ϕ	\wp	$\wp$
\beta	β	\in	$\in$	\pi	π	\xi	ξ
\bullet	$\cdot$	\infinity	∞	\pm	$\pm$	\zeta	ζ
\cap	$\cap$	\int	$\int$	\propto	$\propto$		
\chi	χ	\iota	ι	\psi	ψ		

Tabel 35. Kode-kode penggunaan symbol pada MATLAB

Berikut beberapa contoh penggunaannya :

```
>>  %Contoh  Penggunaan  Perintah
Polar dalam Penyajian Sudut
>> theta = linspace(0,2*pi);
>> Trigonsin = sin(theta);
>> Trigoncos = cos(theta);
>> Trigontan = tan(theta);
>> polar(theta,Trigonsin,'r-*');
>> hold on
>> polar(theta,Trigoncos,'b-d');
>> polar(theta,Trigontan,'g-o');
>> hold off
>>    title('PENYAJIAN    PERINTAH
TRIGONOMETRI  DALAM  KOORDIAT  POLAR
DARI SUDUT (\theta)');
```



Gambar 5. 69. Penyajian fungsi trigonometri pada sistem koordinat polar serta penggunaan simbol theta pada

Dengan adanya pendefinisian simbol `(\theta)` pada bagian judul, maka simbol huruf Latin muncul pada judul

Tabel 36. Penggunaan perintah polar dalam penyajian sudut

I. Rangkuman

- Penyajian grafik dalam Matematika Komputasi merupakan satu kajian sangat penting dalam melihat satu sisi kebesaran dari matematika. Grafik sendiri dalam berbagai jenis tipe mempunyai unsur intrinsik antara lain nilai atau kuantitas-kuantitas yang berada dibaliknya dan yang terpenting adalah hubungan antar nilainya yang kemudian direpresentasikan dalam bentuk dan pewarnaan tertentu.

- Grafik sebagai penyajian fungsi dapat dipandang sebagai representasi geometris dari pasangan niali nilai tertentu yang memebri bentuk tertentu maupun warna tertentu. Melalui bentuk ini, gambaran data secara numerik dapat diidentifikasi secara lebih mudah dan praktis melalui representasi geometris. Seperti halnya dalam meninjau titik maksimum atau minimum suatu kondisi tertentu dapat dibaca dengan mudah melalui data.
- Selain pembacaan nilai maksimum dan minimum, kita dapat melihat kelakukan data melalui gambar mengenai trend, dominasi dan unsur-unsur perubahan pada grafik. Grafik-grafik ini juga dengan sendirinya mengandung estetika khusus.
- Pada dasarnya grafik adalah bentukan dari pasangan nilai-nilai tertentu yang menempati sistem koordinat. Nlai-nilai tersebut ketika disajikan dalam bentuk data numerik maka sejatinya kumpulan nilai tersebut meruapakan vektor atau matriks matriks tertentu di dalam MATLAB.
- Penyajian grafik dua dimensi dalam MATLAB mempunyai perintah-perintah khusus yang membutuhkan nilai-nilai inputan. Perintah-perintah tersebut antara lain : **plot(x,y)** untuk jenis grafik sesuai dengan jenis data dan keutuhan penyajian grafik dua dimensi mempunyai jenis yang lebih khusus seperti

Penyajian data Tiga Dimensi

- **mesh** sebagai perintah untuk membuat grafik 3 Dimensi memasangkan tiga komponen yaitu *x, y,* dan *z* . Penggunaan mesh dapat digunakan dalam menggambarkan beragam fungsi 3 dimensi seperti bola. Tampila dari mesh sendiri berupa jaring-jaring terbuka.
- **surf** merupakan perintah penyajian garfik 3 Dimensi yang menyerupai perintah mesh. Komponen masukannya jua terdiri 3 unsur yaitu

Penyajian data dua Dimensi

- **bar** untuk diagram batang,
- **pie** untuk diagram lingkaran,
- **stairs** untuk fungsi tangga,
- **hist** untuk penyajian frekuensi histogram,
- **rose** untuk penyajian data menyerupai bunga mawar yang fungsinya sama dengan histogram dalam penyejian frekuensi data,
- **polar** untuk penyajian koordinat polar,
- **compas** penyajian data a yang menggambarkan magnitude dan sudut blangan-bilangan kompleks terhadap titik pusat dan dilengkapi dengan panah di ujung nya,
- **stem** untuk penyajian titik-titik dan garis tegak,
- **scatter** berupa penyajian data berupa bulatan-bulatan kecil menyerupai stem namun tanpa garis tegak.

pasangan-pasangan *x, y,* dan *z.* Perbedaan mendasar dengan mesh yaitu terdapat pada tampilan grafik yang menginterpolasi warna pada permukaannnya.

- **Contour** merupakan perintah penyajian data tiga dimensi berupa garis yang menghubungkan nilai yang sama dengan ini dalam penyajiannya tampak disajikan dalam sistem koordinat dua dimensi.
- **Surfc** merupakan perintah untuk menggabungkan antara perintah surf dan contour yang memberikan penyajian data 3 Dimensi denga permukaan yang berwarna serta contour ada sisi pasanyan (x,y).

J. Latihan

Latihan 5. 1: Sketsakanlah garfik $f(x) = x^{1/3}$ dan $g(x) = x^{2/3}$ pada daerah $-5 \leq x \leq 5$, dengan ketentuan sebagai berikut :

Latihan 5. 2: Sketsakanlah fungsi polynomial berikut :

$$f(x) = x^4 + 4x^3 + 2x^2 + x + 2$$
$$g(x) = 4x^3 + 12x^2 + 4x + 1$$
$$h(x) = 12x^2 + 24x + 4$$
$$p(x) = 24x + 24$$
$$q(x) = 24$$

- Grafik $f(x)$ menggunakan tanda * berwarna biru pada satu sistem koordinat.
- Grafik $g(x)$ menggunakan tanda diamond berwarna merah pada sistem koordinat yang berbeda tanpa grid.
- Gabungkan grafik $f(x)$dan $g(x)$ dalam satu koordinat yang sama.
- Gabungkan ketiga gambar di atas pada satu figure dengan tata letak 1 baris 3 kolom, dan lengkapi properti grafiknya, seperti legenda, grid dan judul,serta keteangan sumbu.

pada daerah $-5 \leq x \leq 5$, dengan ketentuan sebagai berikut :

- Grafik $f(x)$, $g(x)$, $h(x)$, $p(x)$ dan $q(x)$digabungkan dalam satu sistem koordinat dengan warna dan style garis yang berbeda, lengkapi dengan legenda dan judul.
- Grafik $f(x)$, $g(x)$, $h(x)$, $p(x)$ dan $q(x)$dipisahkan pada sistem koordinat yang berbeda, namun pada satu figure dengan tata susunan, Grafik $f(x)$, $g(x)$, $h(x)$ pada kolom pertama $p(x)$ dan $q(x)$ dan gabungan grafik berada pada kolom kedua.
- Berikan narasi atau deskripsi pada bagian a dan b di atas

Latihan 5. 3: Diberikan data jumlah kehadiran dan nilai mutu mahasiswa pada mata kuliah Kalkulus 1 tahun akademi 2018/2019 smester genap.

NO.	NIM	NAMA MAHASISWA	JML	NILAI MUTU
1	**2**	**3**	**4**	**25**
1	171.600.011	ABDUL HARIS	1	**E**
2	181.600.001	UMMI KHAERI	16	**A**
3	181.600.002	MILDAYANTI	15	**B**
4	181.600.003	MUSDALIFAH IBRAHIM	16	**A**
5	181.600.004	ALMAIDA AYU	15	**B**
6	181.600.005	ABDUL WAHAB A	16	**A**

7	181.600.006	BESSE NUR ISLAMIYAH SYAM	15	**B**
8	181.600.007	ELMA MEI FERONIKA	16	**B**
9	181.600.008	IRAWATI	16	**A**
10	181.600.009	AHMAD	3	**E**
11	181.600.010	ANDI NIA DAENG PUJI	15	**B**
12	181.600.011	NELLY JULIA	16	**B**
13	181.600.012	NARDA	16	**B**
14	181.600.013	MELLY	16	**B**
15	181.600.014	SITTI RAHMA	16	**A**
16	181.600.034	SEIMA ISWANA TAUFIK	14	**C**
17	181.600.016	ARWINDA WULANDRI	16	**B**
18	181.600.017	NURFAIKA	16	**A**
19	181.600.018	MUFLIH MAHMUD	16	**B**
20	181.600.019	NURFADILLA	16	**A**
21	181.600.020	CATTERINE FRESSA MIADY	16	**A**
22	181.600.021	SUHARTINI ALIMUDDIN	16	**A**
23	181.600.022	MUHAMMMAD RESKY	16	**B**
24	181.600.023	SRI ATIRA YUNUS	16	**B**
25	181.600.035	NOVITA SARI	16	**C**
26	181.600.025	NURUL SELVIANI	15	**A**
27	181.600.027	FITRAH AZIZAH	16	**B**
28	181.600.028	MULYANTI RAHMA	16	**B**
29	181.600.029	SULFA	16	**A**
30	181.600.033	HAFIS	16	**A**

Gunakan data-data di atas untuk :

- Membuat diagram batang kehadiran.
- Membuat diagram lingkaran dengan klasifikasi jumlah kehadiran : kurang dari 8, antara 9 sampai 12, antara 13 sampai 15 dan jumlah kehadiran 16. lengkapi legenda , judul diagram dan bertipe diagram lingkaran 3 Dimensi.
- Membuat diagram lingkaran tentang jumlah mahasiswa (bobot %) yang mendapat A, B, C D dan E , lengkapi legenda , judul diagram dan bertipe diagram lingkaran 3 Dimensi.
- Buatlah narasi dari ketiga bagian di atas

Latihan 5. 4: Diberikan fungsi $Z_1 = \frac{X^2Y^2}{X^2+Y^2}; Z_2 = \frac{X^2Y^2}{X^2-Y^2}$ dari fungsi tersebut lakukan kegiatan berikut :

- Gunakan **mesh** untuk menggambar dua fungsi di atas
- Gunakan **surf** untuk menggambar dua fungsi di atas
- Gunakan **contour** untuk menggambar fungsi berikut.
- Gunakan **surfc** untuk menggambar fungsi berikut.

Latihan 5. 5: Gambarkan fungsi trigonometri berikut $\sin(\theta), \sin(2\theta), \cos(\theta), \cos(2\theta)$ pada daerah $-2\pi \leq \theta \leq 2\pi$ dengan ketentuan sebagai berikut :

- Gunakan 4 fungsi di atas dalam satu sistem koordinat dengan style yang berbeda, tambahkan legenda yang mempertahankan simbol θ dan π
- Pisahkan fungsi $\sin(\theta), \cos(\theta)$ dan $\sin(2\theta), \cos(2\theta)$, dalam dua satu sistem koordinat dengan style yang berbeda kemudian gabungkan dalam satu figure.

BAB 6. FUNGSI

Kemampuan akhir yang diharapkan

A. Pengantar Fungsi

Pada materi-materi sebelumya telah diperkenalkan penggunaan Command Window dan M-File dalam membuat suatu program tertentu. Diperkenalkan pula berbagai jenis penggunaan fungsi dengan struktur-struktur yang hampir sama yaitu terdapat nama fungsi dan terdapat pula argumen-argumen inputan. Fungsi-fungsi yang digunakan sebelumnya merupakan fungsi-fungsi standar yang telah tersedia di dalam aplikasi MATLAB yan dikenal dengan **built in function**. Seperti pada fungsi dalam

- ❖ Mahasiswa dapat membuat peta konsep dari fungsi (*function*) berdasarkan penyajian, bentuk persamaan dan sifat-sifat berdasarkan telaah matematika komputasi.
- ❖ Mahasiswa dapat membuat dan menyimpan fungsi sederhana berdasarkan masalah yang diberikan sesuai dengan kaidah yang berlaku.
- ❖ Mahasiswa dapat menggunakan fungsi sesuai aturan penggunaan fungsi.
- ❖ Mahasiswa dapat membuat fungsi baru dan menggunakannya secara langsung.
- ❖ Mahasiswa dapat mengkombinasikan fungsi-fungsi yang telah dibuat sendiri pada fungsi yang lain.

menyelesaikan persoalan aritmatika, fungsi penanganan matriks atau fungsi fungsi yang digunakan dalam menvisualisasikan himpunan data ke dalam suatu grafik.

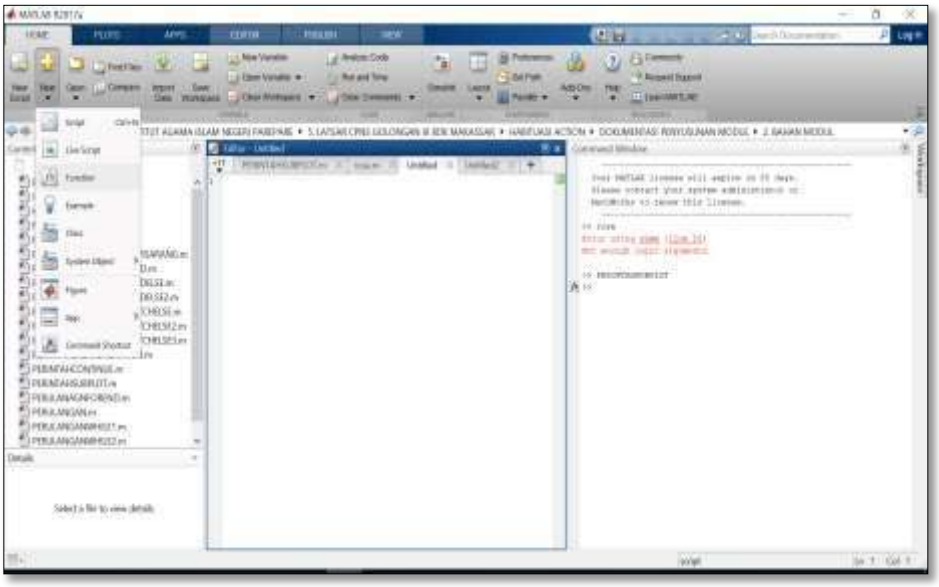
Masalah yang kemudian dihadapi adalah terkadang dalam mengkonstruksi suatu program ada fungsi yang tidak tersedia di dalam MATLAB. Namun masalah tersebut dapat di atasi dengan cara mengkonstruksi fungsi baru dengan algoritma-algoritma yang dibutuhkan sesuai dengan peruntukan masalah yang akan diselesaikan. Fungsi yang dibangun sendiri oleh pengguna dikenal dengan **function by user defined.** Oleh karena itu, pada BAB ini akan diperkenalkan kembali mengenai struktur dasar fungsi, dan ketentuan-ketentuan dasar fungsi. Selain itu, akan dipelajari pula mengenai bagaimana mengkontruksi fungsi baru serta bagaimana menggunakan fungsi-fungsi yang telah dibuat.

B. Pembuatan Fungsi Pada MATLAB

Secara mendasar dalam membangun suatu kerangka dasar fungsi dapat dilakukan dalam beberapa cara :

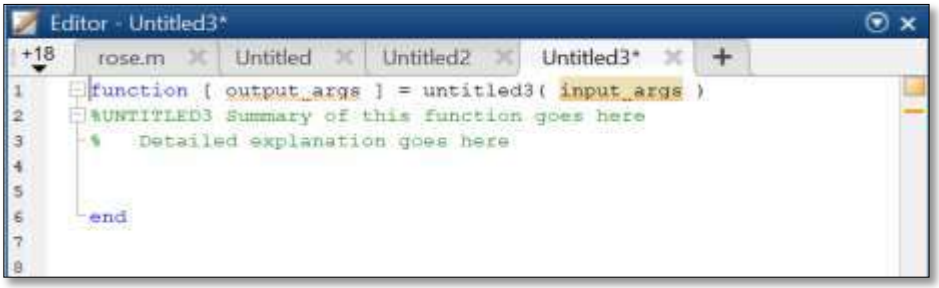
- ❖ **Melalui Menu New**

Ketika mengklik tombol New pada Interface utama MATLAB, maka pada bagian ini terdapat pilihan **function** yang berarti kita siap membuat fungsi dengan memunculkan template dasar fungsi.



Gambar 6. 1. Submenu untuk pembuatan fungsi baru

Ketika tombol **function** tersebut diklik, maka akan muncul template dasar dari function yang dibangun ole user.



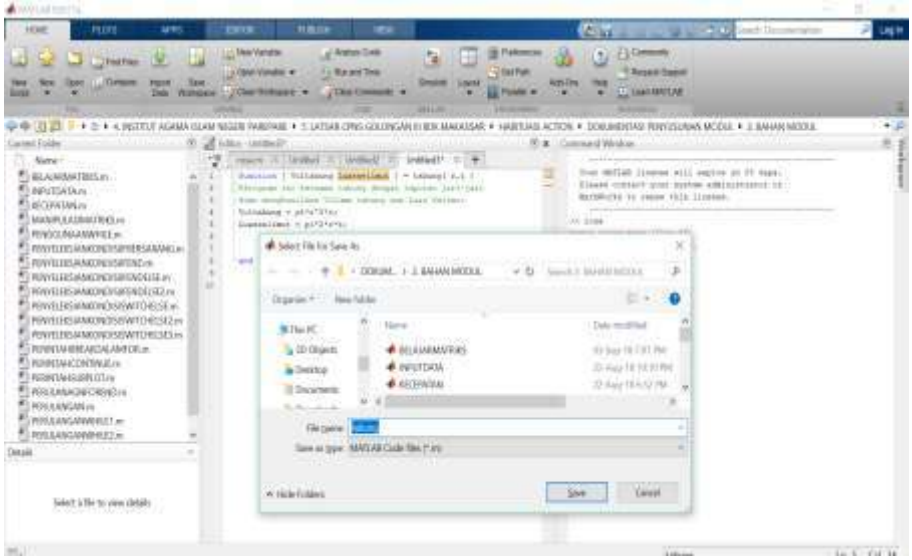
Gambar 6. 2. Template standar fungsi pada jendela editor

Dengan tampilan ini, kita siap memodifikasi template **function** tersebut pada jendela editor. (Pada sub Bab selanjutnya akan dibbahas bagaimana memodifikasi).

❖ **Membangun Strukur Function Secara Manual**

Sebelumnya, struktur dasar Fungsi dilakukan dengan memanfaatkan tools Klik Menu New, kemudian pilih functin. Pada bagian

ini struktur function dibuat oleh user dari awal dengan menekan **CTRL+N** atau klik **New script** pada Home Bar MATLAB. **Pada penjelasan berikut digunakan contoh: membuat function untuk menghitung Volumetabung dan luas selimut sebagai input dengan diketahui jarijari dan tingginya sebagai input.**



Gambar 6. 3. Cara membuat function pada matlab

❖ **Function Definition Line**

Function Definition Line adalah syntax yang mengindikasi bahwa program script file yang anda buat merupakan function. Berikut sistematika function definition line. function **[output arguments]** = **function_name** (input arguments) Syntax "**function**" dideklarasikan untuk mengindikasi bahwa program merupakan function.

Input dan Output Argumen

Input dan Output argumen digunakan untuk mengambil data dari function dibawa ke luar function, umumnya berupa luaran pada command window.

❖ **H1 Line dan Help Text**

H1 Line merupakan comment pada baris pertama dari function file. Comment pada MATLAB diawalai dengan tanda %. Help Text merupakan comment baris selanjutnya setelah H1 line. Help Text dapat ditampilkan

- **[output argumen]** adalah nama variabel output function yang dituliskan dalam tanda kurung siku. Jika output argumen terdiri lebih dari satu variabel, maka setiap variabel dipisahkan dengan tanda koma.
- **function_name** adalah nama function yang dibuat dapat memuat karakter, angka, dan underscore. Nama function tidak boleh sama dengan nama built-in function yang sudah ada dan tidak boleh memuat spasi. Nama function digunakan untuk memanggil function. Nama M-file function nantinya disamakan dengan **function_name**.
- **(input argumen)** adalah variabel input yang dituliskan dalam tanda kurung. Jika input argumen terdiri lebih dari satu varibel, maka setiap variabel dipisahkan dengan tanda koma. Dapat kita analisa function yang akan kita buat mempunyai 2 input dan 2 output sehingga dapat dibuat function definition line sebagai berikut.

- Input pada contoh diatas adalah (panjang, lebar) yang akan dihitung oleh function. Dalam penggunaan lebih lanjut input argumen dapat didefinisikan secara interaktif menggunakan perintah **input**.
- Output pada contoh datas adalah [keliling, luas] yang nilai variabelnya umumnya ditampilkan pada command window. Dalam penggunaan lebih lanjut output argumen dapat ditampilkan secara interaktif menggunakan perintah disp, fprintf maupun plot.

dengan menggunakan perintah "**help function_name**", dengan syntax **help** yang mengindikasi untuk memanggil Help Text suatu function.

Gambar 6. 4. Menjalankan fungsi yangtelah dibuat pada command window

C. Bodi Fungsi

Function Body atau yang lebih dikenal dengan bodi fungsi adalah isi dari function yang kita buat. Function Body dapat memuat semua syntax MATLAB seperti operasi matematika, built-in function, operasi logika, user defined function, perintah input output serta flow control (conditional statement dan perulangan) yang telah kita pelajari pada BAB sebelumnya. Berdasarka contoh dapat dibuat function body sebagai berikut :

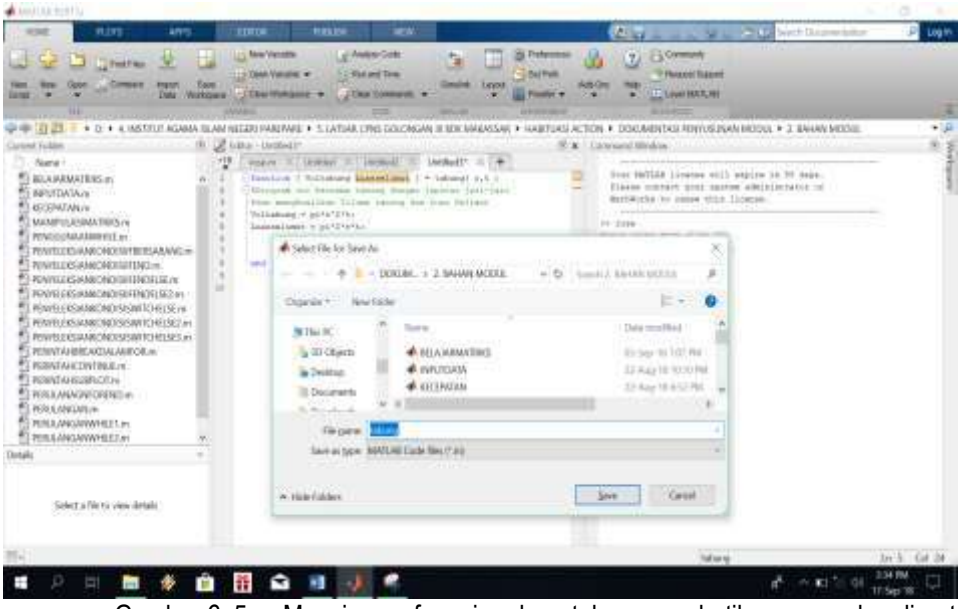
```
function [ Voltabung Luasselimut ] = tabung( r,t )
%Program ini bernama tabung dengan inputan jari-jari(r) dan
tinggi(t)
%Menghasilkan Volume tabung dan Luas Selimut sebgai output
Voltabung = pi*r^2*t;
Luasselimut = pi*2*r*t;
End
```

D. Menyimpan Fungsi

Menyimpan fungsi yang dibuat oleh user juga terdapat ketentuan ketentuan yang harus diperhatikan. Simpan **User Defined function** yang telah dibuat pada **current folder** yang anda gunakan. Setelah tersimpan maka terlihat file .m yang dibuat berisi logo fx seperti gambar diatas pada current folder yang aktif. Ini berarti MATLAB telah mengetahui User Defined Function yang telah kita buat.

- Pada fungsi sederhana diatas dapat dijelaskan bahwa :
- `function` : untuk mendeklarasikan bahwa dalam jendela editor ini, program yang dibuat adalah fungsi. Selanjutnya, `[ Voltabung, Luasselimut ]` merupakan argumen output yang terdiri atas 2 output yaitu Voltabung dan Luas Selimut. Selanjutnya `tabung` merupakan nama fungsi yang dibuat, ketika akan dijalankan pada command window, maka nama fungsi `tabung` yang dipanggil, sekaligus akan menjadi nama file ketika disimpan pada directory yang digunakan. Kemudian pada baris kedua dan ketiga `%Program ini bernama tabung dengan inputan jari-jari(r) dan tinggi(t) %Menghasilkan Volume tabung dan Luas Selimut sebgai output`
- Merupakan komentar sekaligus menjadi informasi mengenai program yang dibuat pada saat dijalankan di command window.
- Selanjutnya pada baris keempat dan kelima `Voltabung = pi*r^2*t; Luasselimut = pi*2*r*t;` pada baris keempat dan kelima merupakan body function yang digunakan untuk mendeklarasikan perintah argumen output yang mengolah argumen argumen input. Dengan perintah ini dapat dikatakan bahwa argumen output Voltabung dan Luasselimut merupakan fungsi yang bergantung pada argumen input jarijari dan tinggi (t). Selanjutnya untuk perintah end, merupakan perintah yang wajib dalam membuat fungsi sebagai tanda penutup pendeklarasian fungsi pada baris pertama.

Dalam menyimpannya, pastikan bahwa nama file yang disimpan sama dengan nama fungsi yang dibuat, olehnya itu tidak perlu untuk merubah nama file. Hal yang harus diperhatikan adalah pada directori mana akan disimpan. Penyimpanan file fungsi sebaiknya berada pada folder dimana fungsi yang didefinisikan itu digunakan untuk menjalankan program lain.

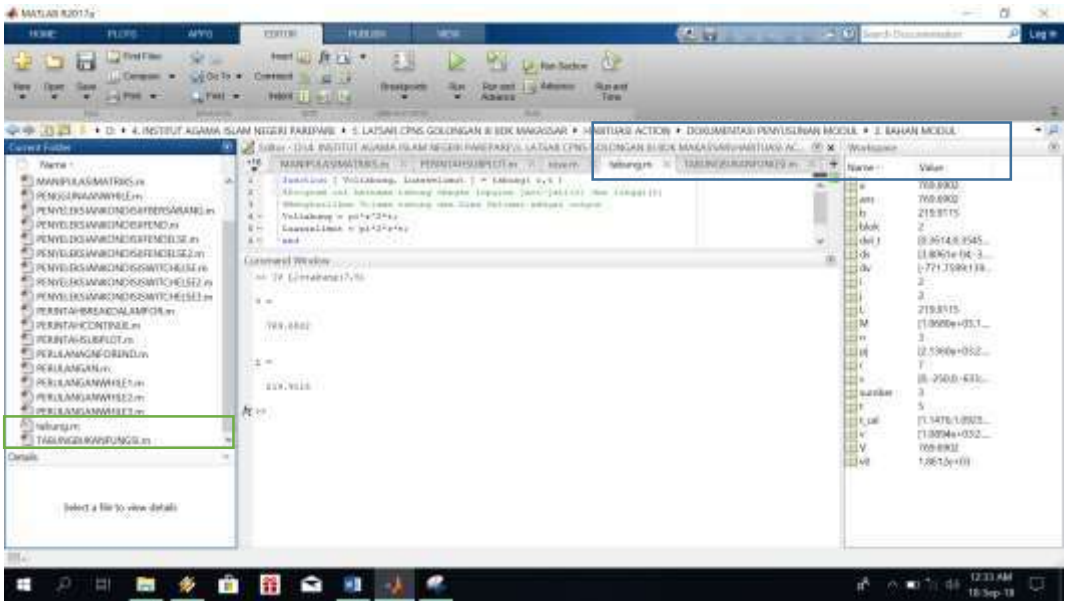


Gambar 6. 5. Menyimpan fungsi perlu untuk memperhatikan nama dan directori

Dari gambar diatas terlihat bahwa dalam menyimpan file fungsi, secara standar nama file yang siap untuk disimpan sama dengan nama fungsi yang dibuat.

❖ **Cara Menggunakan User Defined Function**

Untuk mencoba menggunakan function yang dibuat tadi, anda dapat menutup software MATLAB dan membukanya kembali. **Lalu mengaktifkan current folder tempat anda menyimpan function files.**



Fungsi yang telah dibuat dapat digunakan pada program lain Corrent folder menyimpan function

❖ **Menggunakan Perintah Help**

Fungsi yang telah dibuat sebelumnya dengan nama tabung dapat kita temukan informasinya dengan menuliskan perintah `help` **`tabung,`** maka akan mengeluarkan informasi mengenai fungsi tabung. Informasi tersebut diambil dari komentar yang telah dibuat pada baris setelah pendeklarasian fungsi.

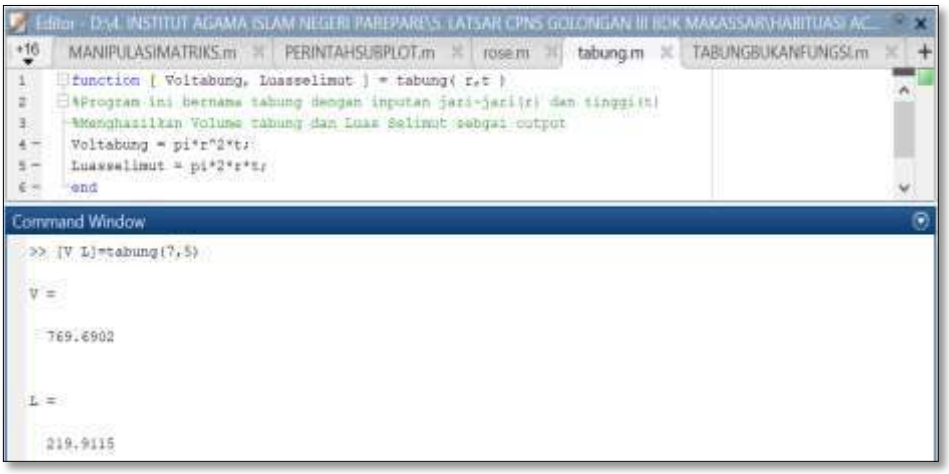
Gambar 6. 6. Mengeluarkan informasi fungsi dengan perintah help

❖ **Menggunakan Function Pada Command Window**

Seperti yang telah dijelaskan pada bagian sebelumnya bahwa fungsi yang telah disimpan dapat langsung dijalankan pada command window. Dengan format menggunakan :

```
[output variabel] = function_name (input variabel)
```

Berikut contoh pemanggilannya pada Command Window :

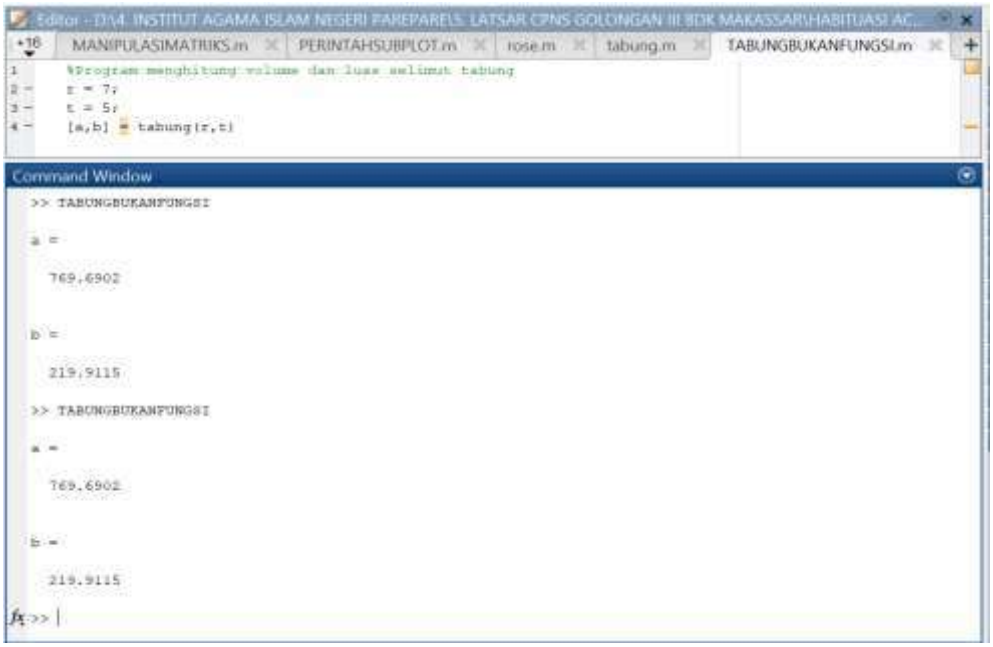


Gambar 6. 7. Menjalankan Fungsi pada Command Window

Terlihat bahwa meskipun dalam pendeklarasian dibaris pertama adalah [Voltabung, Luasselimut] sebagai argumen output, tapi pada command window tetap dijalankan dengan membuat argumen output yang berbeda yaitu V dan L. Selanjutnya pada penulisan nama fungsi harus tepat sesuai dengan nama fungsi yang dibuat, sedangkan argumen inputnya harus menempati posisi yang bersesuaian dengan yang diinginkan seperti 7 sebagai jari jari dan 5 sebagai tinggi tabung.

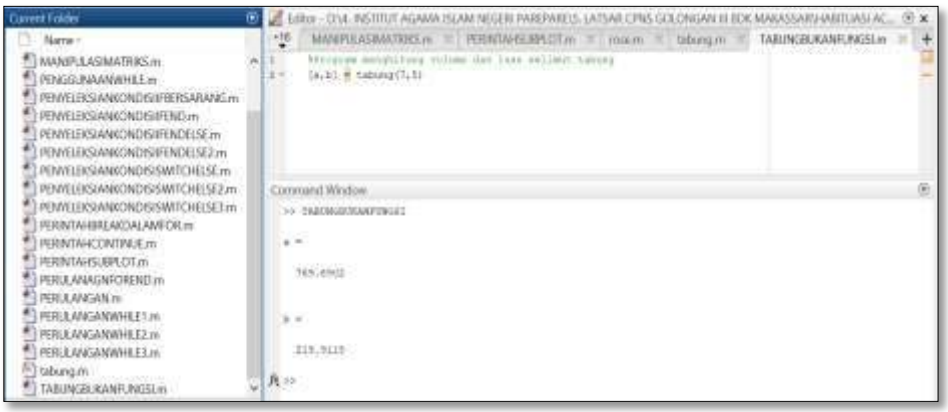
E. Penggunaan Function pada M-File yang berbeda

Cara menggunakan function pada MATLAB yang didefinisikan oleh user pada M-File yang berbeda adalah cukup dengan mendefinisikan variabel-variabel inputan baik dalam fungsi maupun dideklarasikan secara terpisah. Berikut contoh penggunaannya pada M-File berbeda dengan cara mendefinisikan argumen inputan secara terpisah dari fungsi



Gambar 6. 8. Penggunaan Fungsi pada M-File yang berbeda

Dari gambar di atas terlihat bahwa pada M-File berbeda, fungsi tabung digunakan dengan mendefinisikan nilai r dan t yang terpisah dari nama fungsi. Selanjutnya argumen output yang dibuat adalah variabel a dan variabel b.



Gambar 6. 9. Simulasi Fungsi yang telah dibuat

Pada contoh ini, terlihat bahwa nilai r dan t langsung dituliskan ke dalam fungsi tabung dan dijalankan langsung pada command window. Sebagai catatan kesimpulan untuk pembuatan-pembuatan fungsi adalah sebagai berikut :

- Dalam membuat fungsi, kita harus memperhatikan aturan-aturan penulisannya, seperti pada nama fungsi (harus berbeda dari nama fungsi yang tersedia di dalam MATLAB dan aturan penulisannya mengikuti aturan penulisan variabel).
- Selanjutnya untuk argumen output dalam hal ini variabel output harus terdeklarasikan pada bodi fungsi. Demikian pula dengan argumen input, harus digunakan dalam menentukan output. Penulisan-penulisannya juga, harus mengikuti aturan-aturan penulisan variabel.
- Informasi dari fungsi dapat dibuat pada baris komentar.
- Dalam hal penyimpanan file fungsi, harus diperhatikan bahwa nama file harus seuai dengan dengan nama fungsi dan ditempatkan sesuai dengan directori dimana fungsi akan digunakan.
- Menjalankan fungsi berbeda dengan menjalankan M-file yang bukan fungsi (dengan mengklik tombol running), menjalankan fungsi dapat dilakukan dengan menuliskan secara terurut dan sesuai `argumen output = namafungsi(argument input)` pada command window dan dipastikan file fungsi berada pada current dirrectori. Demikian pula penggunaan fungsi pada M-File yang berbeda, penulisan fungsi harus sesuai dengan nama fungsi dan argumen input.

- Fungsi dalam dunia matematika dipandang sebagai aturan yang memetakan setiap anggota himpunan A sebagai daerah asal ke tepat satu anggota himpunan B sebagai daerah hasil.
- Secara khusus dalam dunia komputasi , fungsi dikenal sebagai suatu tools yang digunakan untuk mengolah inputan menjadi suatu output sesuai dengan algoritma atau aturan yang telah dibangun pada fungsi yang digunakan.

F. Rangkuman

- Pada Matematika Komputasi fungsi dipandang sebagai perintah-perintah dalam berbagai perangkat lunak atau *software* matematika yang digunakan untuk mengolah inputan secara matematis baik berupa data numerik, data string maupun data double.
- Pada MATLAB secara khusus, fungsi dibedakan atas dua jenis yaitu fungsi yang telah tersedia di dalam MATLAB yang dikenal dengan *Built function* dan fungsi yang didefinisikan oleh pengguna yang dikenal dengan *function user by defined.*
- Eksplorasi penggunaan fungsi yang bersifat bawaan MATLAB dapat dilakukan dengan bantuan help dan bantuan tombol fx. Didalamnya terdapat informasi mengenai bagaimana penggunaan fungsi-fngsi yang dibutuhkan dan deskripsi secara teoritis mengenai fungsi tersebut.
- Fungsi yang didefinisikan oleh pengguna merupakan fugsi yang dibuat berdasarkan algoritma yang dibangun oleh penggun. Dalam pembuatannya dapat dilakukan dengan menggunakan template yang terseda ataupun dibuat secara langsung oleh pengguna.
- Hal utama yang perlu diperhatikan dalam pembuatan bahwa fungsi mempunyai strutur penulisan tersendiri. Pada baris pertama sebagai digunakan untuk menetapkan nama fungsi dan mendefinsikan vaiabel-variabel yang terdiri atas argumen input dan argumen output.
- Argumen output dapat berupa variabel yang ditentukan atau bergantung pada variabel input yang didefinisikan.
- Penamaan fungsi jga memliki ketentuan seperti halnya dengan vaiabel. Dalam hal ini nama fungsi yang dipilih harus berbeda dengan fungsi yang telah tersedia di dalam MATLAB. Perlu pula diperhatikan nama fungsi tidak melibatkan kaakter spasi. Nama fungsi juga harus sesuai dengan nama file ketika dilakukan penyimpanan.
- Penjelasan mengani fungsi yang didefinisikan dapat dibuat dengan menambahkan baris-baris komentar untuk medeskripsikan fungsi tanpa mengganggu perintah-perintah utama.

G. Latihan

Latihan 6. 1: Buatlah fungsi yang digunakan untuk menyelesaikan perhitungan Volume kubus dan luas permukaan dengan ketentuan sebagai berikut :

- Nama fungsi kubus
- Menggunakan argumen inputan panjang sisi.

Latihan 6. 2: Buatlah fungsi untuk menyelesaikan menggambar parabola dengan menggunakan parameter p sebagai jarak titik fokus ke titik puncak.

Latihan 6. 3: Identifikasilah fungsi-fungsi yang berkaitan dengan :

- Integral
- Diferensial
- Ekonomi
- Biologi
- Pengolahan Citra Digital
- Pengolahan sinyal

BAB 7. KOMPUTASI NUMERIK

Kemampuan Akhir Yang Diharapkan

- ❖ Mahasiswa dapat mengimplementasikan penggunaan konsep pengolahan vektor, matriks, penyajian grafik, perulangan, penyeleksian kondisi, dan pembuatan fungsi dengan merancang Listing program MATLAB dan simulasi dari masing-masing program yang dibuat untuk menyelesaikan :

1) Masalah Sistem persamaan Linear melalui algoritma metode Iterasi Jacobi dan metode iterasi Gauss Seidel.
2) Masalah pencarian akar dari persamaan Non Linear melalui algoritma dari metode bagi dua (*bisection*) dan metode Newton Raphson.
3) Masalah Interpolasi melalui metode selisih terbagi dan polynomial Newton.
4) Masalah Integrasi Numerik melalui Aturan Romberg
5) Masalah Persamaan diferensial melalui metode Runge Kutta Orde 4

A. Penyelesaian Sistem Persamaan Linear

Salah satu wujud model matematika yang banyak dijumpai di dalam berbagai disiplin ilmu seperti ekonomi, fisika, statistika, keteknikan, dan riset operasional adalah Sistem Persamaan Linear (SPL). Sahid (2005) menyatakan bahwa kemunculan sistem persamaan linear beraifat langsung dari masalah-masalah nyata dan penyelesaiannya merupakan bagian dari proses penyelesaian masalah yang lain.

Pembelajaran mengenai dasar SPL pada bagian sebelumnya dijumpai pada mata kuliah Aljabar Linear Elementer. Terdapat beberapa konsep mendasar yang ditekankan pada pembelajaran SPL diantaranya :

1. Sistem Persamaan Linear secara matematis dibentuk oleh beberapa persamaan linear yang dapat merepresentasikan berbagai fenomena. Masalah yang akan diselesaikan dalam Sistem

Persamaan Linear yaitu bagaimana menemukan solusi secara simultan pada suatu masalah SPL.

2. Secara teoritis, penyelesaian SPL dapat dilakukan dengan beberapa metode seperti metode grafik dan metode eliminasi-subtitusi. Namun metode tersebut terbatas pada SPL yang hanya terdiri dari dua variabel. Pada SPL yang memilik tiga variabel atau lebih dapat diselesaikan melalui penyelesaian matriks. SPL yang telah direpresentasikan oleh Matriks Koefisien, Vektor Solusi dan Vektor
3. Konstanta dapat diselesaikan dengan metode Eliminasi Gauss, Eliminasi Gauss Jordan atau metode perkalian Inverse Matriks. Metode-metode tersebut dikenal sebagai metode langsung dengan melibatkan perhitungan Operasi Baris Dasar dalam matriksnya.
4. Pada kenyataanya Operasi Baris Dasar suatu matriks membutuhkan waktu dan ketelitian tingkat tinggi dalam penyelesaiannya, terlebih ketika matriks koefisien unsur-unsurnya berupa pecahan dan mempunyai ukuran yang besar. Oleh karena itu, dibutuhkan suatu metode penyelesaian alternatif yang dapat diterapkan dalam menyelesaikan masalah SPL.
5. Konsep dasar metode iteratif dalam menyelesaikan suatu SPL yaitu dengan memanipulasi setiap bentuk SPL dengan menyatakan suatu barisan solusi. Solusi pada langkah pertama disebut sebagai solusi awal, dengan masing-masing algoritma akan konvergen ke solusi analitik.

❖ **Bentuk umum SPL**

Suatu Sistem Persamaan Linear secara bentuk dapat dinyatakan sebagai susunan persamaan-persamaan linear, dimana setiap persamaan linear terdiri dari unsur variabel, koefisien dan konstanta. Berikut bentuk Umum Persamaan Linear :

SISTEM PERSAMAAN LINEAR	$a_{ij} : Koefisien$	$x_j : Variabel\ ke-j$	$c_i : Konstanta$
$i = 1 : Persamaan\ ke-1$	$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = c_1$		
$i = 2 : Persamaan\ ke-2$	$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = c_2$		
$i = 3 : Persamaan\ ke-3$	$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \cdots + a_{3n}x_n = c_3$		
dan seterusnya hingga ...	$\vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots$		
$i = m : Persamaan\ ke$	$a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \cdots + a_{mn}x_n = c_m$		

A sebagai Matriks koefisien, b vektor konstanta, x vektor hampiran solusi.

Pada buku matematika komputasi, penentuan solusi persamaan linear akan diselesaikan melalui metode iterasi yang berbeda yaitu metode iterasi Jacobi dan Gauss Seidel. Metode yang bersifat iteratif digunakan nuk menyeesaikan SPL yang berukuran besar. Dari kedua metode ini akan diuraikan prinsip kerjanya dalam bentuk algoritma yang selanjutnya akan dibandingkan laju konvergennya.

❖ Metode Iterasi Jacobi

Metode Iterasi Jacobi merupakan teknik penyelesaian SPL berukuran $\boldsymbol{n \times n}$ secara iteratif. Prosesnya dimulai dari hampiran awal terhdap penyelesaian $\boldsymbol{X_0}$ kemudian membentuk serangkain vektor-vektor $\boldsymbol{X_1, X_2,}$ ... yang konvergen ke $\boldsymbol{X}$.

Dalam mencari solusi SPL pada matriks yang berukuran kecil, metode langsung seperti metode Gauss lebih praktis dan efisien dibandingkan dengan metode iteratif. Namun untuk ukuran matriks yang besar, maka akan sangat sulit untuk menerapkan metode langsung. Dibutuhkan iterasi hingga konvergen ke solusi yang diinginkan.

Konsep dasar dari iterasi jacobi menggunakan rumusan :

$$x_i^{(k+1)} = \frac{b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(k)}}{a_{ii}},\ k = 0,1,2,\ldots$$

Algoritma Metode Iterasi Jacobi

1. Misal diberikan masalah SPL yang direpresentasikan oleh $\boldsymbol{Ax} = \boldsymbol{b}$ dimana $\boldsymbol{A}$ merupakan matriks konstanta dan $\boldsymbol{x}$ adalah vektor solusi serta $\boldsymbol{b}$ adalah vektor konstanta.
2. Dengan demikian dibutuhkan input antara lain $n, A, b, Imaks$ dan hampiran awal $\boldsymbol{Y} = (\boldsymbol{y_1^0}, \boldsymbol{y_2^0}, \boldsymbol{y_3^0}, \cdots, \boldsymbol{y_n^0})$ serta $\boldsymbol{Tol}$ sebagai toleransi kesalahan.
3. Output : $\boldsymbol{X} = (\boldsymbol{x_1^{kmax}}, \boldsymbol{x_2^{kmax}}, \boldsymbol{x_3^{kmax}}, \cdots, \boldsymbol{x_n^{kmax}})$
4. Proses :

- Identifikasi solusi awal $k = 1$
- Jika $k \leq Imaks$, Definisikan iterasi pertama dengan $k = 1$
- Untuk iterasi $i = 1,2,3, \ldots, kmax$, gunakan formula

$$x_i^{(k+1)} = \frac{b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(k)}}{a_{ii}},\ k = 0,1,2, \ldots$$

- Atur kembali $\boldsymbol{X} = (\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \boldsymbol{x_3^k}, \cdots, \boldsymbol{x_n^k})$
- Jika $\|X - Y\|$ lebih kecil dari toleransi, maka solusi telah didapatkan. Jika belum, lanjutkan!
- Lanjutkan ke iterasi $k = k + 1$, misal dari $k = 1$ menjadi $k = 2$.
- Definisikan kembali $\boldsymbol{Y} = (\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \boldsymbol{x_3^k}, \cdots, \boldsymbol{x_n^k})$ sebagai hampiran awal yang baru.

5. Jika telah mencapai $Imaks$ maka “Solusi tidak ditemukan”
6. Stop

Algoritma di atas akan diimplementasikan dalam suatu listing program dan disimulasikan untuk melihat bagaimana algoritma dari metode ini bekerja.

Implementasi program MATLAB

Implementasi Metode Iterasi Jacobi dalam Program MATLAB untuk Penyelesaian SPL metode Jacobi (*Editor*)

```
function [y,k,Xt,k1,s]= jacobi(A,b,x)
%Metode Iteratif Jacobi, input matriks koefisien A, vektor
konstanta b
kmax=100;delta = 1e-10;epsilon = 0.5e-4;
Xt =[];          %Untuk Tabel
k1 = [];         %Iterasi
s = [];          %Norm(y-x)
n = size(A,1);
for k = 1: kmax,
    y =x;
    for i = 1:n,
    jumlah = b(i);
    diagonal=A(i,i);
    if abs(diagonal)< delta;
        disp('Diagonal element to small');
        return;
    end
    for j = 1:n,
        if j~= i,
            jumlah = jumlah - A(i,j)*y(j);
        end
        end
    x(i)= jumlah/diagonal;
    end
    r = norm(y-x);
    Xt=[Xt; k x' norm(y-x)];
    k1 =[k1; k'];
    s  =[s;r'];

fprintf('==========================================================
====\n');
disp('  Iterasi(k)     x1    x2    x3     x4       norm(y-x)');
fprintf('==========================================================
====\n');
Xt
fprintf('==========================================================
====\n');
plot(k1,s,'r-*')
title('GRAFIK PERUBAHAN SELISIH ANTAR VEKTOR HAMPIRAN SOLUSI (k
- k+1)');
xlabel('Iterasi (k)');
ylabel('Selisih Hampiran Solusi (Norm(y-x))');
grid on
if r < epsilon,
disp('norm < epsilon');
return;
end
end
disp('maximum iteration reachad');
return;
```

Simulasi

1. Simulasi 1
   Misal suatu SPL memiliki Matriks Koefisien, vektor solusi dan vektor konstanta didefinisikan dalam Comand Window berikut :

```
>> %Matriks Koefisien A, vektor konstanta b dan vektor hampiran
solusi x
>> A = [7 1 -1 2;1 8 0 -2;-1 0 4 -1;2 -2 -1 6]
A =
     7     1    -1     2
     1     8     0    -2
    -1     0     4    -1
     2    -2    -1     6
>> b = [3;-5;4;-3]
b =  3
    -5
     4
    -3
>> x =[0;0;0;0]
x =  0
     0
     0
     0
>>
```

❖ **Metode Iterasi Jacobi**

List Program Penyelesaian SPL metode Jacobi (*Editor*), program tersebut digunakan untuk menyelesaikan Sistem Persamaan Linear yang terbentuk oleh persamaan beda masalah nilai batas. Pada penyelesian SPL tersebut digunakan iterasi maksimum sampai 1000000 iterasi. Secara tekhnis, program tersebut harus dipastikan berada pada folder yang sama dengan program diskritisasi dan pembentukan matriks koefisien dari SPL.

❖ **List Program Penyelesaian SPL metode Jacobi**

```
function [y,k,Xt,k1,s]= jacobi(A,b,x)
%Metode Iteratif Jacobi, input matriks koefisien A, matriks
konstanta b
kmax=1000000;delta = 1e-10;epsilon = 0.5e-4;
Xt =[];          %Untuk Tabel
k1 = [];         %Iterasi
s = [];          %Norm(y-x)
n = size(A,1);
for k = 1: kmax,
    y =x;
    for i = 1:n,
    jumlah = b(i);
    diagonal=A(i,i);
    if abs(diagonal)< delta;
```

```
        disp('Diagonal element to small');
        return;
    end
    for j = 1:n,
        if j~= i,
            jumlah = jumlah - A(i,j)*y(j);
        end
        end
    x(i)= jumlah/diagonal;
    end
    r = norm(y-x);
    Xt=[Xt; k x' norm(y-x)];
    k1 =[k1; k'];
    s  =[s;r'];

fprintf('=============================================================\n');
disp('  Iterasi(k)     x1    x2    x3     x4       norm(y-x)');
fprintf('=============================================================\n');
Xt
fprintf('=============================================================\n');
plot(k1,s,'r-*')
title('GRAFIK PERUBAHAN SELISIH ANTAR VEKTOR HAMPIRAN SOLUSI (k - k+1)');
xlabel('Iterasi (k)');
ylabel('Selisih Hampiran Solusi (Norm(y-x))');
grid on
      if r < epsilon,
            disp('norm < epsilon');
      return;
      end
end
disp('maximum iteration reachad');
return;
```

Running Program Jacobi

```
>> [Y,K,XT,K1,S]=JACOBI(A,B,X)
Y = 1.0000
   -1.0000
    1.0000
   -1.0000
K = 16
XT =
    XT =
    1.0000    0.4286   -0.6250    1.0000   -0.5000    1.3507
    2.0000    0.8036   -0.8036    0.9821   -0.6845    0.4548
    3.0000    0.8793   -0.8966    1.0298   -0.8720    0.2276
    4.0000    0.9529   -0.9529    1.0018   -0.9203    0.1082
    5.0000    0.9708   -0.9742    1.0081   -0.9683    0.0558
    6.0000    0.9884   -0.9884    1.0006   -0.9803    0.0267
    7.0000    0.9928   -0.9936    1.0020   -0.9922    0.0138
    8.0000    0.9971   -0.9971    1.0002   -0.9951    0.0066
    9.0000    0.9982   -0.9984    1.0005   -0.9981    0.0034
```

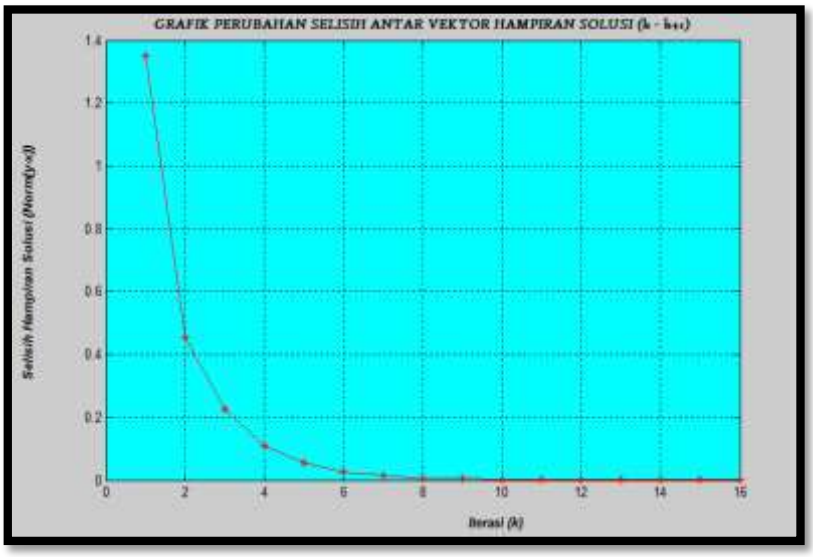
```
   10.0000    0.9993   -0.9993    1.0000   -0.9988    0.0016
   11.0000    0.9996   -0.9996    1.0001   -0.9995    0.0008
   12.0000    0.9998   -0.9998    1.0000   -0.9997    0.0004
   13.0000    0.9999   -0.9999    1.0000   -0.9999    0.0002
   14.0000    1.0000   -1.0000    1.0000   -0.9999    0.0001
   15.0000    1.0000   -1.0000    1.0000   -1.0000    0.0001
   16.0000    1.0000   -1.0000    1.0000   -1.0000    0.0000
```

k1 = 1	s = 1.3507
2	0.4548
3	0.2276
4	0.1082
5	0.0558
6	0.0267
7	0.0138
8	0.0066
9	0.0034
10	0.0016
11	0.0008
12	0.0004
13	0.0002
14	0.0001
15	0.0001
16	0.0000

Hasil iterasi Jacobi, dengan delta = 1e-10 dan epsilon = 0.5e-4 membutuhkan 16 iterasi :

```
============================================================
  Iterasi(k)     x1          x2          x3          x4        norm(y-x)
============================================================
Xt =

    1.0000    0.4286   -0.6250    1.0000   -0.5000    1.3507
    2.0000    0.8036   -0.8036    0.9821   -0.6845    0.4548
    3.0000    0.8793   -0.8966    1.0298   -0.8720    0.2276
    4.0000    0.9529   -0.9529    1.0018   -0.9203    0.1082
    5.0000    0.9708   -0.9742    1.0081   -0.9683    0.0558
    6.0000    0.9884   -0.9884    1.0006   -0.9803    0.0267
    7.0000    0.9928   -0.9936    1.0020   -0.9922    0.0138
    8.0000    0.9971   -0.9971    1.0002   -0.9951    0.0066
    9.0000    0.9982   -0.9984    1.0005   -0.9981    0.0034
   10.0000    0.9993   -0.9993    1.0000   -0.9988    0.0016
   11.0000    0.9996   -0.9996    1.0001   -0.9995    0.0008
   12.0000    0.9998   -0.9998    1.0000   -0.9997    0.0004
   13.0000    0.9999   -0.9999    1.0000   -0.9999    0.0002
   14.0000    1.0000   -1.0000    1.0000   -0.9999    0.0001
   15.0000    1.0000   -1.0000    1.0000   -1.0000    0.0001
   16.0000    1.0000   -1.0000    1.0000   -1.0000    0.0000
norm < epsilon
```

Grafik perubahan selisih norm vektor-vektor solusi hampiran dengan menggunakan iterasi jacobi :



Gambar 7. 1. Grafik perubahan selisih norm vektor-vektor solusi hampiran

❖ **Metode Iterasi Gauss Seidel :**

Pada metode iterasi Jacobi, pencarian solusi didasarkan pada penggunaan hampiran awal pada langkah 0 secara bersamaan untuk digunakan pada langkah $k = 1$. Selanjutnya hasil yang diperoleh pada langkah $k = 1$, secara bersamaan digunakan pada langkah $k = 2$. Dalam hal ini metode Iterasi Jacobi, nilai dari langkah $\boldsymbol{X^k} = \left(\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \boldsymbol{x_3^k}, \cdots, \boldsymbol{x_n^k}\right)$ digunakan secara bersamaan untuk menentukan $\boldsymbol{X^{k+1}} = \left(\boldsymbol{x_1^{k+1}}, \boldsymbol{x_2^{k+1}}, \boldsymbol{x_3^{k+1}}, \cdots, \boldsymbol{x_n^{k+1}}\right) = \boldsymbol{f}\left(\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \boldsymbol{x_3^k}, \cdots, \boldsymbol{x_n^k}\right)$. Sedangkan prinsip kerja dari metode iterasi Gauss Seidel adalah penggunaan nilai elemen vektor dari $\left(\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \boldsymbol{x_3^k}, \cdots, \boldsymbol{x_n^k}\right)$ apabila telah menghasilkan $\boldsymbol{x_1^{k+1}}$maka dapat langsung digunakan untuk menentukan $\boldsymbol{x_2^{k+1}}$, demikian halnya $\boldsymbol{x_2^{k+1}}$dapat digunakan untuk menentukan $\boldsymbol{x_3^{k+1}}$ dan seterusnya., jadi dapat dituliskan menjadi :

$$\boldsymbol{X^{k+1}} = \boldsymbol{f}\left(\boldsymbol{x_1^{k+1}}, \boldsymbol{x_2^{k+1}}, \boldsymbol{x_3^{k+1}}, \cdots, \boldsymbol{x_n^{k+1}}\right)$$

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)}\right)$$

Dari Formula ini, algoritma metode Gauss Seidel dapat dituliskan menjadi

- Metode ini disebut juga metode penggantian berurutan, oleh karena penyelesaian SPL diperoleh dengan cara mengganti setiap variabel secara berurutan.
- Metode Gauss Seidel dilakukan dengan cara menggunakan hasil setiap iterasi pada langkah iterasi berikutnya
- Rumus iterasi ke-k:

Algoritma Metode Iterasi Jacobi umum

1. Misal diberikan masalah SPL yang direpresentasikan oleh $\boldsymbol{Ax} = \boldsymbol{b}$ dimana $\boldsymbol{A}$ berukuran $n \times n$ merupakan matriks konstanta dan $\boldsymbol{x}$ adalah vektor solusi serta $\boldsymbol{b}$ adalah vektor konstanta.
2. Dengan demikian dibutuhkan input antara lain $n, A, b, Imaks$ dan hampiran awal $\boldsymbol{Y} = (\boldsymbol{y_1^0}, \boldsymbol{y_2^0}, \boldsymbol{y_3^0}, \cdots, \boldsymbol{y_n^0})$ serta $\boldsymbol{Tol}$ sebagai toleransi kesalahan.
3. Output : $\boldsymbol{X} = (\boldsymbol{x_1^{kmax}}, \boldsymbol{x_2^{kmax}}, \boldsymbol{x_3^{kmax}}, \cdots, \boldsymbol{x_n^{kmax}})$
4. Proses :

- Identifikasi solusi awal $k = 1$
- Jika $k \leq Imaks$, Definisikan iterasi pertama dengan $k = 1$
- Untuk iterasi $i = 1,2,3, \ldots, kmax$, gunakan formula

$$x_i^{(k)} = \frac{1}{a_{ii}}(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)})$$

- Atur kembali $\boldsymbol{X} = (\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \boldsymbol{x_3^k}, \cdots, \boldsymbol{x_n^k})$
- Jika $\|X - Y\|$ lebih kecil dari toleransi, maka solusi telah didapatkan. Jika belum, lanjutkan!
- Lanjutkan ke iterasi $k = k + 1$, misal dari $k = 1$ menjadi $k = 2$.
- Definisikan kembali $\boldsymbol{Y} = (\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \boldsymbol{x_3^k}, \cdots, \boldsymbol{x_n^k})$ sebagai hampiran awal yang baru.

5. Jika telah mencapai $Imaks$ maka “Solusi tidak ditemukan”; Metode Gagal
6. Stop

Implementasi Program Penyelesaian SPL metode Gauss Seidel pada MATLAB

```
function [y,k,Xt,k2,s2]= GaussSeidel(A,b,x)
%Metode Iteratif Gauss Seidel
%input matriks koefisien A, matriks konstanta b
kmax=100;
delta = 1e-10;
epsilon = 0.5e-4;
Xt = [];     % Hasil Iterasi
k2 = [];     % Vektor kolom iterasi 1:kmax
s2 = [];     % Vektor kolom norm(y-x)
n = size(A,1);
for k = 1: kmax,
    y =x;
    for i = 1:n,
    jumlah = b(i);
    diagonal=A(i,i);
    if abs(diagonal)< delta;
        disp('Diagonal element to small');
        return;
    end
    for j = 1:i-1,
          jumlah = jumlah - A(i,j)*x(j);
    end
    for j = i+1:n,
          jumlah = jumlah - A(i,j)*y(j);
    end
    x(i)= jumlah/diagonal;
    end
    r = norm(y-x);
    Xt=[Xt; k x' r];
    k2 =[k2; k'];
    s2  =[s2;r'];
fprintf('=============================================================\n');
    disp('  Iterasi(k)     x1       x2     x3        x4      norm(y-x)');
fprintf('=============================================================\n');
    Xt
fprintf('=============================================================\n');
plot(k2,s2,'b--o')
title('GRAFIK PERUBAHAN SELISIH ANTAR VEKTOR HAMPIRAN SOLUSI (k - k+1)
METODE GAUSS SEIDEL');
xlabel('Iterasi (k)');
ylabel('Selisih Hampiran Solusi (Norm(y-x))');
grid on
    if norm(y-x)< epsilon,
        disp('norm < epsilon');
        return;
    end
end
disp('maximum iteration reachad');
return;
```

Running Program Gauss Seidel :

```
>> [y,k,Xt,k2,s2]= GaussSeidel(A,b,x)
y = 1.0000
```

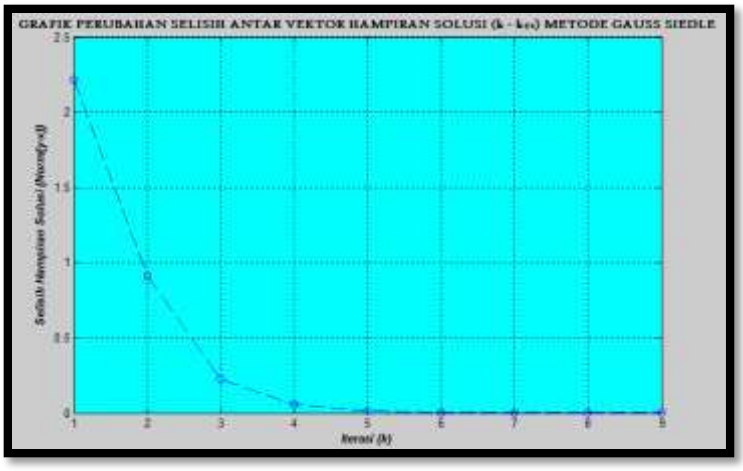
```
   -1.0000
    1.0000
   -1.0000
k =     9
```

k2 =	s2 =
1	1.5292
2	0.5602
3	0.1350
4	0.0333
5	0.0082
6	0.0020
7	0.0005
8	0.0001
9	0.0000

```
==========================================================================
  Iterasi(k)     x1          x2          x3          x4        norm(y-x)
==========================================================================
Xt =
    1.0000    0.4286   -0.6786    1.1071   -0.6845    1.5292
    2.0000    0.8793   -0.9060    1.0487   -0.9203    0.5602
    3.0000    0.9708   -0.9764    1.0126   -0.9803    0.1350
    4.0000    0.9928   -0.9942    1.0031   -0.9951    0.0333
    5.0000    0.9982   -0.9986    1.0008   -0.9988    0.0082
    6.0000    0.9996   -0.9996    1.0002   -0.9997    0.0020
    7.0000    0.9999   -0.9999    1.0000   -0.9999    0.0005
    8.0000    1.0000   -1.0000    1.0000   -1.0000    0.0001
    9.0000    1.0000   -1.0000    1.0000   -1.0000    0.0000
==========================================================================
norm < epsilon
```

Tabel 37. Iterasi dan Perubahan selisih Norm Vektor

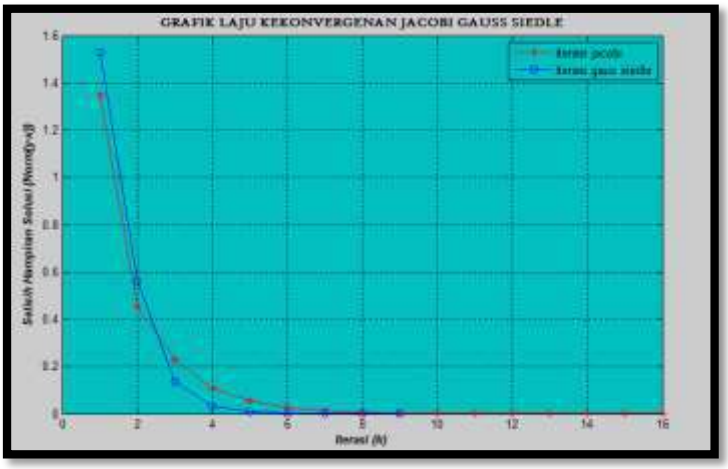
Grafik perubahan selisih norm vektor-vektor solusi hampiran dengan menggunakan iterasi gauss Seidel :



Gambar 7. 2. Grafik perubahan selisih norm vektor solusi hampiran melalui iterasi Gauss

Grafik perbandingan laju kekonvergenan menggunakan iterasi jacobi dengan gauss Seidel:

```
%Perbandingan Kekonvergenan Metode Iterasi Jacobi dan Gauss
Seidel
function konvergensi(k1,s1,k2,s2)
    plot(k1,s1,'r-*',k2,s2,'b-o')
    title('GRAFIK PERBEDAAN LAJU KEKONVERGENAN JACOBI VS
GAUSS SEIDEL');
    xlabel('Iterasi (k)');
    ylabel('Selisih Hampiran Solusi (Norm(y-x))');
    grid on
end
```



Gambar 7. 3. Perbedaan laju kekonvergenan metode Jacobi dan Gauss Seidel

Dari kedua metode di atas memberi hasil bahwa metode Gauss Seidel lebih cepat konvergen ke solusi yang sebenarnya dibandingkan dengan metode iterasi Jacobi.

B. Akar-akar Persamaan Non Linier

❖ Background Masalah

Permasalahan menentukan akar persamaan non linear merupakan salah satu topik masalah dalam metode numerik. Pada buku ini, masalah penentuan akar persamaan non linear dipilih sebagai sub bahasan untuk mengeksplorasi dan melihat bagaimana prinsip dari pada Matematika

Komputasi itu sendiri, dan bagaimana secara teknis pengetahuan mendasar mengenai keterampilan penggunaan MATLAB dapat diimplementasikan. Pada dasarnya mencari akar-akar persamaan non linier dapat dibedakan atas metode kurungan dan metode terbuka

❖ **Metode kurungan (bracketing method)**

Metode ini biasa juga disebut metode jebakan karena langkah pertama yang dilakukan adalah menaksir dimana letak akar dengan cara mengurung. Selanjutnya kurungan tersebut diperkecil. Berbagai prosedur dapat dilakukan untuk memperkecil kurungan tersebut, misalnya :

- ❖ Metode Tabel atau Grafik
- ❖ Bisection method (metode bagi dua)
- ❖ Regula falsi / False position method (metode posisi palsu)

❖ **Metode Terbuka**

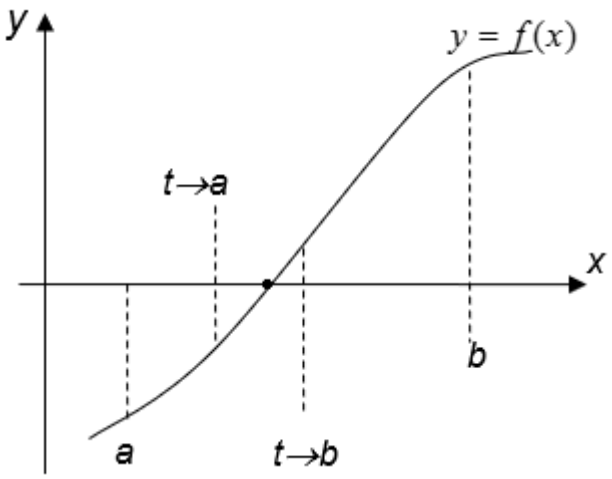
Perbedaan dasar metode ini dengan metode kurungan adalah pada metode ini, pemilihan sebuah (atau beberapa) nilai awal tidak perlu mengurung akar, akibatnya dengan metode ini dapat saja diperoleh iterasi yang konvergen (semakin mendekati akar) atau divergen (semakin menjauhi akar). Beberapa pendekatan (formulasi) dapat digunakan antara lain :

- ❖ Fixed point method (metode titik tetap)
- ❖ Newton-Raphson Method
- ❖ Secant Method (metode tali busur)

Pada buku ini dipilih satu metode kurungan dan satu metode terbuka, dengan melihat bagaimana prinsip kerja atau algoritma dari masing-masing matode. Olehnya itu akan di tinjau dari pemecahan maslahnya secara manual dan menggunakan Fasilitas MATLAB. Metode yang akan dikaji adalah metode bagi dua (*bisection*) dan Metode newton Raphson.

Metode Bagi Dua

Metode bagi dua adalah salah satu jeni smetode kurungan. Prosedur metode ini didahului dengan menaksir dimana letak akar $f(x)$ dengan cara mengurung akar pada interval tertutup [*a*,*b*] dengan syarat $f(a)$ dan $f(b)$ haruslah berbeda tanda. Kenyataan ini didasarkan pada konsep teorema harga menengah. Prosedur yang digunakan untuk mempersempit jebakan adalah dengan cara kurungan sebelumnya (interval tertutup [*a*,*b*]) atas dua bagian yang sama besar. Yakni interval $a < x < t$ dan interval $t < x < b$, dengan demikian *t* merupakan titik tengah interval [a,b]. Ilustrasi geometris dari metode bagi dua dapat dilihat dari gambar berikut ini



Gambar 7. 4. Ilustrasi Iterasi Metode Bagi Dua

Algoritma Metode Bagi Dua

Secara simbolik, algoritma metode bagi dua dapat dilihat pada kotak berikut :

1. Input : *f*(*x*), *a*, *b*, *Ftol* (Nilai *a* dan *b* haruslah memenuhi syarat $f(a) * f(b) < 0$)

2. Output: *Akar*
3. Proses : $t = \frac{1}{2}(a+b)$ dan hitung $f(t)$
4. Jika $|f(t)| < Ftol$ maka $\text{Akar} := t$. **Selesai**

5. Jika $f(a) * f(t) < 0$ maka $b := t$ jika tidak $a := t$
6. Kembali ke 1.

Dapat dilihat bahwa, pada Algoritma metode bagi dua kriteria konvergensi yang digunakan adalah toleransi nilai nol fungsi, yaitu $|f(t)| < Ftol$. Kriteria ini dapat saja diganti dengan kriteria yang lain, misalnya jika digunakan kriteria tolerasi nilai kesalahan (*Etol*), maka L2 menjadi :

7. **2a.** Jika $|t - tlama| < Etol$ maka $\text{Akar} := t$ **Selesai**
8. **2b.** $tlama := t$

jika digunakan toleransi nilai kesalahan relatif (*Rtol*) maka L2 menjadi :

9. **2a.** Jika $\left|\frac{t - tlama}{t}\right| < Rtol$ maka $\text{Akar} := t$ **Selesai**
10. **2b.** $tlama := t$

Untuk memahami cara kerja dari algoritma di atas berikut diberikan beberapa contoh soal yang diselesaikan secara manual dan menggunakan Fasilitas Komputasi MATLAB.

Contoh Soal 5: Misal diberikan suatu fungsi $f(x) = x^3 - 3$, tentukan akar dari persamaan non linear tersebut dengan menggunakan nilai $Ftol = 10^{-4}$.

Jawab :

Berdasarkan algoritma metode bisection dapat dipilih nilai awal $a = 1$ dan $b = 2$, diperoleh nilai $f(1) = 1^3 - 3 = -2$ dan $f(2) = 2^3 - 3 = 5$, berarti syarat berlawanan tanda dipenuhi. Untuk menjalankan prinsip pengerjaan metode bagi dua dapat di sajikan dalam bentuk tabel. Berikut disusun tabel hasil iterasi yang perhitungannnya dapat dibuat menggunakan fasilitas Microsoft office Excel.

Perhatikan bahwa dipilih nilai awal dengan *f*(*a*) yang **bernilai negatif**.

Iterasi	a	b	t	$f(t)$	$f(a) * f(t) < 0$

0	1	2	1.5	0.375	*Ya*
1	1	1.5	1.25	-1.0468	*Tidak*
2	1.25	1.5	1.375	-0.4004	*Tidak*
3	1.375	1.5	1.4375	-0.0295	*Tidak*
4	1.4375	1.5	1.4688	0.1684	*Ya*
5	1.4375	1.4688	1.4531	0.0684	*Ya*
6	1.4375	1.4531	1.4453	0.0192	*Ya*
7	1.4375	1.4453	1.4414	-0.0053	*Tidak*
8	1.4414	1.4453	1.4434	0.0069	*Ya*
9	1.4414	1.4434	1.4424	0.0008	*Ya*
10	1.4414	1.4424	1.4419	-0.0022	*Tidak*
11	1.4419	1.4424	1.4421	-0.0007	*Tidak*
12	1.4421	1.4424	1.44225	**0.000003**	

Tabel 38. Perhitungan dengan Metode Bagi Dua untuk $f(x) = x^3 - 3$

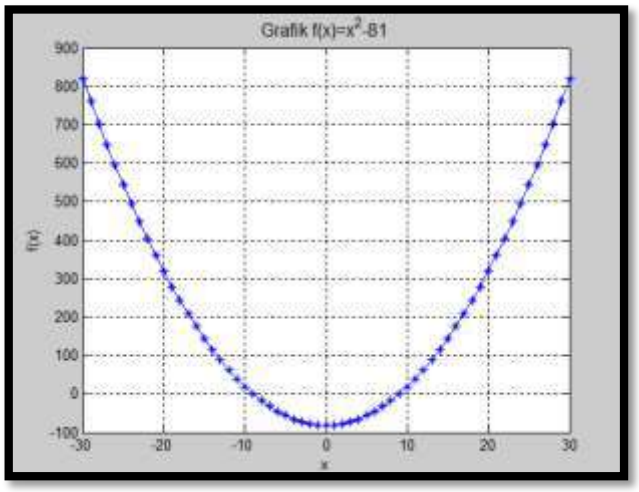
Pada tabel 23, dapat dilihat bahwa setelah iterasi ke-12, diperoleh *t*=1.44225 dengan $f(t) = 0.000003 = 3 \times 10^{-6} < 10^{-4}$, maka disimpulkan bahwa salah satu akar dari $f(x) = x^3 - 3$ adalah *x*=1.44225.

Contoh Soal 6: Misal diberikan suatu fungsi $f(x) = x^2 - 81$. Buatlah dan simulasikan satu Fungsi dengan mengimplementasikan Metode Bagi Dua (Bisection),

- Plotlah Grafik dari Fungsi tersebut
- Buatlah satu **function** baru yang dapat menentukan solusi numerik dari persamaan tersebt dengan menerapkan Algoritma metode bagi dua. Pada function yang dibuat, Pemilihan nilai awal : a dan b, nilai Toleransi dan Jumlah Iterasi Maksimal dipilih sebagai argumen input pada function yang dibuat.

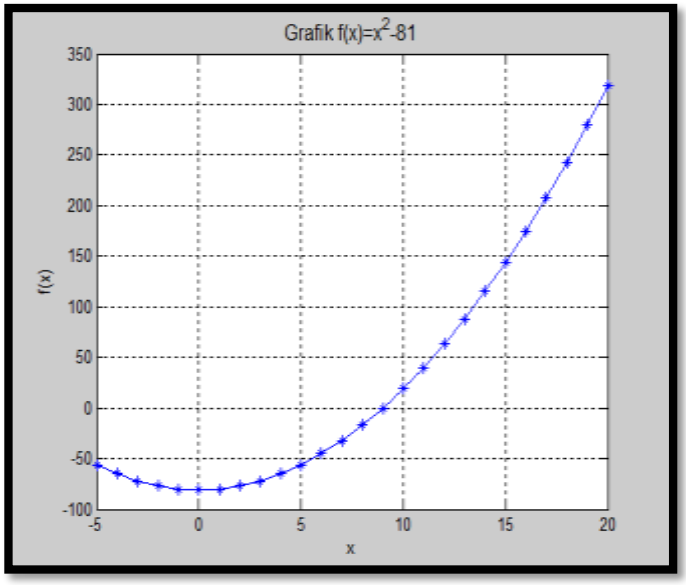
Jawab :

Dari $f(x) = x^2 - 81$, grafiknya dapat dibuat pada MATLAb dengan hasil seperti ini :

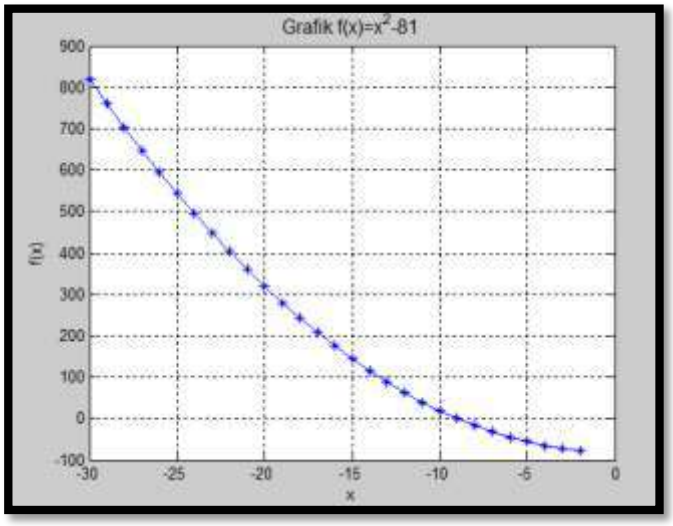


Gambar 7. 5. Grafik persamaan $f(x) = x^2 - 81$; $-30 \leq x \leq 30$

Seperti penjelasan sebelumnya bahwa secara analitik akar persamaan $f(x) = x^2 - 81$ dapat secara langsung diperoleh bahwa nilai x yang memenuhi $f(x) = 0$ adalah $x_1 = 9\ atau\ x_1 = -9$ sebagai akarnya. Sebagai latihan untuk menentukan akar persamaan di atas dapat dilakukan dengan menerapkan metode numeric. Metode penyelesaian yang digunakan pada noomor ini yaitu dengan menggunakan algoritma metode bagi dua atau *bisection*.



Gambar 7. 6. Tampilan bagian funsi naik dari persamaan $f(x) = x^2 - 81$



Gambar 7. 7. Tampilan bagian fungsi turun $f(x) = x^2 - 81$

Berdasarkan algoritma metode bagi dua, maka listing program yang dapat menyelesaiakan akar persamaan non linear seperti pada kotak berikut :

```
%editor kelebihannya, jika ada kesalahn maka dapat dilakukan
perbaikan
function bisection(f,a,b,tol,n)
%Metode kekonvergenan global bisection (Metode Bagi Dua)
%a = kiri; b = kanan

syms x
N = [];

for i=1:n
        c = (a+b)/2;                        %pendugaan akar antara a
dan b
        fa= subs(f,x,a);                    %nilai fungsi di x=a
        fb= subs(f,x,b);                    %nilai fungsi di x=b
        fc= subs(f,x,c);                    %nilai fungsi di x=c
        N = [N;i a c b fa fb fc ];
 if fa*fc<0
     b=c;                                   %mempersempit interval a
dan b
 else
     a=c;
 end

 if abs(fa-fc) < tol                        %Kriteria Tolerani nilai
fungsi F.tol
     break;
     %keluar
 else
 end
end

%Pengaturan Tabel Iterasi
```

```
fprintf('=================================================================
================\n');
disp('  Iterasi(n)      a         c          b        f(a)       f(b)
f(c)');
fprintf('=================================================================
================\n');
disp(N)
fprintf('=================================================================
================\n');
Kesalahan = abs(fa-fc)
```

Simulasi Metode bagi Dua

1. Simulasi 1 digunakan dua nilai titik awal, fungsi, F.toleransi, dan jumlah iterasi. Input titik pendugaan akar di Command Window :

```
>> syms x
>> f = x^2-81;
>> a=4;b=30;
>> tol = 1e-5;
>> n=100;
>> bisection(f,a,b,tol,n)


================================================================================
  Iterasi(n)      a         c          b        f(a)       f(b)        f(c)
================================================================================
    1.0000    4.0000   17.0000   30.0000  -65.0000  819.0000  208.0000
    2.0000    4.0000   10.5000   17.0000  -65.0000  208.0000   29.2500
    3.0000    4.0000    7.2500   10.5000  -65.0000   29.2500  -28.4375
    4.0000    7.2500    8.8750   10.5000  -28.4375   29.2500   -2.2344
    5.0000    8.8750    9.6875   10.5000   -2.2344   29.2500   12.8477
    6.0000    8.8750    9.2813    9.6875   -2.2344   12.8477    5.1416
    7.0000    8.8750    9.0781    9.2813   -2.2344    5.1416    1.4124
    8.0000    8.8750    8.9766    9.0781   -2.2344    1.4124   -0.4213
    9.0000    8.9766    9.0273    9.0781   -0.4213    1.4124    0.4929
   10.0000    8.9766    9.0020    9.0273   -0.4213    0.4929    0.0352
   11.0000    8.9766    8.9893    9.0020   -0.4213    0.0352   -0.1932
   12.0000    8.9893    8.9956    9.0020   -0.1932    0.0352   -0.0791
   13.0000    8.9956    8.9988    9.0020   -0.0791    0.0352   -0.0220
   14.0000    8.9988    9.0004    9.0020   -0.0220    0.0352    0.0066
   15.0000    8.9988    8.9996    9.0004   -0.0220    0.0066   -0.0077
   16.0000    8.9996    9.0000    9.0004   -0.0077    0.0066   -0.0005
   17.0000    9.0000    9.0002    9.0004   -0.0005    0.0066    0.0030
   18.0000    9.0000    9.0001    9.0002   -0.0005    0.0030    0.0012
   19.0000    9.0000    9.0000    9.0001   -0.0005    0.0012    0.0003
   20.0000    9.0000    9.0000    9.0000   -0.0005    0.0003   -0.0001
   21.0000    9.0000    9.0000    9.0000   -0.0001    0.0003    0.0001
   22.0000    9.0000    9.0000    9.0000   -0.0001    0.0001    0.0000
   23.0000    9.0000    9.0000    9.0000   -0.0001    0.0000   -0.0000
   24.0000    9.0000    9.0000    9.0000   -0.0000    0.0000   -0.0000
   25.0000    9.0000    9.0000    9.0000   -0.0000    0.0000   -0.0000
   26.0000    9.0000    9.0000    9.0000   -0.0000    0.0000    0.0000
================================================================================
Kesalahan =  6.9737e-006
```

Terlihat dengan mengambil dua nilai awal yaitu a=4 dan b=30 membutuhkan 26 iterasi untuk mencapai F-toleransi yang ditentukan yang merupakan bagian fungsi naik dari fungsi ini. Nilai akarnya konvergen ke c=9.0000.

2. Selanjutnya simulas 2, untuk bagian fungsi turunnya dipilih nilai awal a=-30 dan b =-4, membutuhkan 26 iterasi hingga mencapai syrat F.toleransinya. Nilai akarnya konvergen ke c=-9.0000

```
>> syms x
>> f = x^2-81;
>> a=-30;b=-4;
>> n=100;
>> tol = 1e-5;
>> bisection(f,a,b,tol,n)
================================================================================
  Iterasi(n)     a         c         b       f(a)       f(b)        f(c)
================================================================================
    1.0000  -30.0000  -17.0000   -4.0000  819.0000  -65.0000  208.0000
    2.0000  -17.0000  -10.5000   -4.0000  208.0000  -65.0000   29.2500
    3.0000  -10.5000   -7.2500   -4.0000   29.2500  -65.0000  -28.4375
    4.0000  -10.5000   -8.8750   -7.2500   29.2500  -28.4375   -2.2344
    5.0000  -10.5000   -9.6875   -8.8750   29.2500   -2.2344   12.8477
    6.0000   -9.6875   -9.2813   -8.8750   12.8477   -2.2344    5.1416
    7.0000   -9.2813   -9.0781   -8.8750    5.1416   -2.2344    1.4124
    8.0000   -9.0781   -8.9766   -8.8750    1.4124   -2.2344   -0.4213
    9.0000   -9.0781   -9.0273   -8.9766    1.4124   -0.4213    0.4929
   10.0000   -9.0273   -9.0020   -8.9766    0.4929   -0.4213    0.0352
   11.0000   -9.0020   -8.9893   -8.9766    0.0352   -0.4213   -0.1932
   12.0000   -9.0020   -8.9956   -8.9893    0.0352   -0.1932   -0.0791
   13.0000   -9.0020   -8.9988   -8.9956    0.0352   -0.0791   -0.0220
   14.0000   -9.0020   -9.0004   -8.9988    0.0352   -0.0220    0.0066
   15.0000   -9.0004   -8.9996   -8.9988    0.0066   -0.0220   -0.0077
   16.0000   -9.0004   -9.0000   -8.9996    0.0066   -0.0077   -0.0005
   17.0000   -9.0004   -9.0002   -9.0000    0.0066   -0.0005    0.0030
   18.0000   -9.0002   -9.0001   -9.0000    0.0030   -0.0005    0.0012
   19.0000   -9.0001   -9.0000   -9.0000    0.0012   -0.0005    0.0003
   20.0000   -9.0000   -9.0000   -9.0000    0.0003   -0.0005   -0.0001
   21.0000   -9.0000   -9.0000   -9.0000    0.0003   -0.0001    0.0001
   22.0000   -9.0000   -9.0000   -9.0000    0.0001   -0.0001    0.0000
   23.0000   -9.0000   -9.0000   -9.0000    0.0000   -0.0001   -0.0000
   24.0000   -9.0000   -9.0000   -9.0000    0.0000   -0.0000   -0.0000
   25.0000   -9.0000   -9.0000   -9.0000    0.0000   -0.0000   -0.0000
   26.0000   -9.0000   -9.0000   -9.0000    0.0000   -0.0000    0.0000
================================================================================
Kesalahan =  6.9737e-006
```

3. Simulasi 3, diuji pula untuk nilai awal yang berbeda yaitu a=1 dan b =81.

```
>> syms x
>> f =x^2-81;
>> a=1;b=81;
>> tol = 1e-5;
```

```
>> n=100;
>> bisection(f,a,b,tol,n)
================================================================================
  Iterasi(n)      a         c         b        f(a)       f(b)        f(c)
================================================================================
  1.0e+003 *

    0.0010    0.0010    0.0410    0.0810   -0.0800    6.4800    1.6000
    0.0020    0.0010    0.0210    0.0410   -0.0800    1.6000    0.3600
    0.0030    0.0010    0.0110    0.0210   -0.0800    0.3600    0.0400
    0.0040    0.0010    0.0060    0.0110   -0.0800    0.0400   -0.0450
    0.0050    0.0060    0.0085    0.0110   -0.0450    0.0400   -0.0088
    0.0060    0.0085    0.0098    0.0110   -0.0088    0.0400    0.0141
    0.0070    0.0085    0.0091    0.0098   -0.0088    0.0141    0.0023
    0.0080    0.0085    0.0088    0.0091   -0.0088    0.0023   -0.0033
    0.0090    0.0088    0.0090    0.0091   -0.0033    0.0023   -0.0006
    0.0100    0.0090    0.0090    0.0091   -0.0006    0.0023    0.0008
    0.0110    0.0090    0.0090    0.0090   -0.0006    0.0008    0.0001
    0.0120    0.0090    0.0090    0.0090   -0.0006    0.0001   -0.0002
    0.0130    0.0090    0.0090    0.0090   -0.0002    0.0001   -0.0000
    0.0140    0.0090    0.0090    0.0090   -0.0000    0.0001    0.0001
    0.0150    0.0090    0.0090    0.0090   -0.0000    0.0001    0.0000
    0.0160    0.0090    0.0090    0.0090   -0.0000    0.0000   -0.0000
    0.0170    0.0090    0.0090    0.0090   -0.0000    0.0000   -0.0000
    0.0180    0.0090    0.0090    0.0090   -0.0000    0.0000    0.0000
    0.0190    0.0090    0.0090    0.0090   -0.0000    0.0000    0.0000
    0.0200    0.0090    0.0090    0.0090   -0.0000    0.0000   -0.0000
    0.0210    0.0090    0.0090    0.0090   -0.0000    0.0000   -0.0000
    0.0220    0.0090    0.0090    0.0090   -0.0000    0.0000    0.0000
    0.0230    0.0090    0.0090    0.0090   -0.0000    0.0000    0.0000
    0.0240    0.0090    0.0090    0.0090   -0.0000    0.0000   -0.0000
    0.0250    0.0090    0.0090    0.0090   -0.0000    0.0000   -0.0000
    0.0260    0.0090    0.0090    0.0090   -0.0000    0.0000    0.0000
    0.0270    0.0090    0.0090    0.0090   -0.0000    0.0000    0.0000
    0.0280    0.0090    0.0090    0.0090   -0.0000    0.0000   -0.0000

================================================================================
Kesalahan =
  5.3644e-006
```

Dari simulasi ini diperoleh bahwa akar persaman non linear membutuhkan 26 iterasi hingga mencapai syrat F.toleransinya. Nilai akarnya konvergen ke c=-9.0000, dengan memilih nilai awal a=1 dan b=81 diperoleh c=41 dan diperoleh nilai f(c)= 1.0e+003 *1.6000, dengan memilih dua titik awal tersebut membutuhkan 28 iterasihingga mencapai batas pada nilai F.toleransi. Akar dari peramaan nonlinear tersebut yaitu c=1.0e+003 *0.0090=9.000.

❖ **Metode Newton Raphson**

Metode Newton Raphson merupakan salah satu jenis metode tanpa kurungan. Pada metode sebelumnya, metode jebakan dapat dengan mudah dikerjakan, mengingat adanya jaminan konvergensi dari teorema harga menengah.

Kesulitan yang mungkin muncul adalah mungkinkah kita dapat dengan mudah menebak dua nilai awal yang memenuhi syarat agar nilai fungsinya berbeda tanda?

Jika diketahui : $f(x) = x^4 + x^3 + 0.1$, Bagaimana memilih dua nilai awal yang memenuhi persyaratan ? Berapa kali anda melakukan trial and error (mencoba, jika gagal coba lagi) sebelum mendapatkan nilai awal yang memenuhi syarat ? Adakah referensi yang dapat digunakan untuk menyimpulkan bahwa $f(x)$ definit positif. Untuk mengatasi hal demikian berikut diperkenalkan metode terbuka, karena dengan metode ini jebakan akar tidak diperlukan.

Metode Newton Rapshon

Metode ini memanfaatkan garis singgung kurva, kemudian nilai *x* dimana garis singgung memotong sumbu-*x* dipilih sebagai titik iterasi berikutnya. Persamaan garis singgung di titik (x_0,$f(x_0)$) adalah :

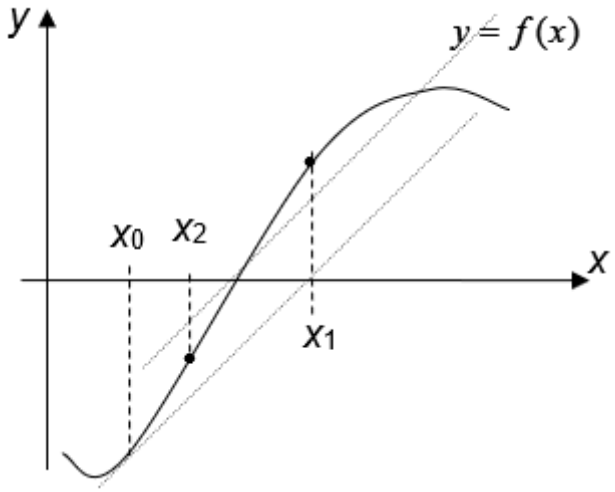
$$y = f(x_0) + (x - x_0) f'(x_0)$$

Misalkan titik potong garis singgung tersebut dengan sumbu-*x*, adalah titik $(x_1,0)$, maka diperoleh :

$$0 = f(x_0) + (x_1 - x_0) f'(x_0)$$

atau

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$



Gambar 7. 8. Ilustrasi Metode Newton Raphson

Dari keterangan di atas, dapat dibentuk iterasi : $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

Persamaan ini dikenal dengan nama *Iterasi Newton Rapshon*.

Algoritma Metode Newton Raphson

Dari cara ini, kemudian Algoritma dari Metode Newton Raphson dapat disusun seperti berikut ini :

1. Input : x_0, $f(x)$, *Rtol* , *Maxi*
2. Output: *Akar* 3. Proses : 4. n:=0 5. Hitung $f(x_n)$ dan $f'(x_n)$ 6. Jika $f'(x_n)$=0, proses gagal. **Selesai.**
$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$
7. Jika $\left\|\frac{x_{n+1} - x_n}{x_{n+1}}\right\| < Rtol$ maka $Akar := x_{n+1}$. **Selesai.** 8. Jika $n \le Maxi$ maka $n = n+1$: Kembali ke L2.
"Iterasi divergen sampai dengan iterasi yang ke-*Maxi*". **Selesai**.

Catatan : *Rtol*:= Toleransi kesalahan relatif yang dimungkinkan. *Maxi* := Maksimum banyaknya iterasi yang dilakukan.

Untuk memahami prinsip kerja dari metode Newton Raphson, maka berikut diberikan beberapa contoh soal yang menerapkan metode Newton Raphson secara manual dan menggunakan fasilitas komputasi MATLAB.

Contoh Soal 7: Misal diberikan suatu persamaan $f(x) = x^4 + x^3 + 0.1$, Tentukan akar diatas dengana menggunakan Metode Newton Raphson.

Jawab :

$$f(x) = x^4 + x^3 + 0.1 \qquad \Rightarrow \qquad f'(x) = 4x^3 + 3x^2$$

Dengan nilai awal $x_0 = -1.0$ dan *Rtol*:=10^{-6}, diperoleh barisan iterasi sebagai berikut :

i	$x(i)$	$f(x(i))$	$f'(x(i))$	$R(x(i))$
0	-1.0	0.1	-1.0	-
1	-0.9	0.0271	-0.486	0.1111111
2	-0.8442386831	0.0062750207	-0.2686703218	0.0660492
3	-0.8208828446	9.21204E-04	-0.1910572349	0.0284521
4	-0.8160612340	3.64866E-05	-0.1759754840	5.90839E-03
5	-0.8158538948	6.65083E-08	-0.1753340397	2.54138E-04
6	-0.8158535155	2.22516E-13	-0.1753328668	4.64941E-07

Tabel 1. Hasil perhitungan metode Newton

Karena nilai $R(x(i)) = 4.64941 \times 10^{-7} < 10^{-6} = Rtol$, maka disimpulkan bahwa akar dari $f(x) = x^4 + x^3 + 0.1$ adalah x = -0.8158535155.

Pada contoh di atas telah diperlihatkan secara manual bahwa dengan menggunakan prinsip atau algorima metode Newton Rapson dapat dilakukan penentuan akar persamaan non linear.

Selanjutnya algoritma dari newton Raphson akan dikodifikasi menjadi satu listing program dengan prosedur yang sama dan fungsi yang sama dari metode bagi dua, akan diperlihatkan bagaimana Metode Newton Raphson

dari segi Program MATLABnya. Melengkapi program Metode Newton Raphson untuk memunculkan iterasi .

Contoh Soal 8: Misal diberikan suatu fungsi $f(x) = x^2 - 81$. Buatlah dan simulasikan satu Fungsi dengan mengimplementasikan Newton raphson !

- Plotlah Grafik dari Fungsi tersebut
- Buatlah satu **function** baru yang dapat menentukan solusi numerik dari persamaan tersebt dengan menerapkan Algoritma metode bagi dua. Pada function yang dibuat, Pemilihan nilai awal : a dan b, nilai Toleransi dan Jumlah Iterasi Maksimal dipilih sebagai argumen input pada function yang dibuat.

Secara analitik akar persamaan $f(x) = x^2 - 81$ dapat secara langsung diperoleh bahwa nilai x yang memenuhi $f(x) = 0$ adalah $x_1 = 9\, atau\, x_1 = -9$ sebagai akarnya. Sebagai latihan untuk menentukan akar persamaan di atas dapat dilakukan dengan menerapkan metode numerik untuk menyelesaikan persamaan non linear yaitu metode *Newton Raphson.*

Implementasi Algoritma Metode Newton Raphson

Berdasarkan instruksi soal maka berikut Listing program dan Simulasi Newton Rapson dalam menentukan Akar Persamaan Non Linear:

```
function newton(f,a,tol,n)
%Metode Newton Raphson
%Mencari akar Persamaan Non-Linear
%a = nilai awal
syms x
N=[];
x_lama = a;
tur_f=diff(f,x);
for i =1:n
    fx = subs(f,x,x_lama);
    tur_fx = subs(tur_f,x,x_lama);
    x_baru = x_lama - (fx/tur_fx);
    N=[N;i x_lama fx  tur_fx x_baru];
    if abs(fx) < tol,
    break;
    end
    x_lama = x_baru;
```

```
end
%Pengaturan Tabel Iterasi
fprintf('=================================================\n');
disp('  Iterasi(n)      x        f(x)     tur_f(x)    x_baru ');
fprintf('=================================================\n');
disp(N)
fprintf('=================================================\n');
```

Simulasi Program Metode Newton Raphson

Input titik pendugaan akar di Command Window

```
>> a=-6;
>> syms x
>> f = x^2-81;
>> tol = 1e-5;
>> n=100;
>> newton(f,a,tol,n)
=================================================
  Iterasi(n)      x        f(x)     tur_f(x)    x_baru
=================================================
    1.0000   -6.0000  -45.0000  -12.0000   -9.7500
    2.0000   -9.7500   14.0625  -19.5000   -9.0288
    3.0000   -9.0288    0.5201  -18.0577   -9.0000
    4.0000   -9.0000    0.0008  -18.0001   -9.0000
    5.0000   -9.0000    0.0000  -18.0000   -9.0000

=================================================
```

Dengan memilih x=-6 sebagai titik awal hanya dibutuhkan 5 iterasi hingga mencapai F-toleransi yang ditetapkan. Dan diperoleh x=-9.0000 sebagai akar dari fungsi di atas. Slanjutnya pendugaan solusi dipilih a=20, dengan maksud mendekati fungsi dari f(a) yang positif dan dapat konvergen ke solusi x=9. Dimana nilai F.tolernsinya sama dan itearsi n=100.

```
>> a=20;
>> newton(f,a,tol,n)
=================================================
  Iterasi(n)      x        f(x)     tur_f(x)    x_baru
=================================================
    1.0000   20.0000  319.0000   40.0000   12.0250
    2.0000   12.0250   63.6006   24.0500    9.3805
    3.0000    9.3805    6.9935   18.7610    9.0077
    4.0000    9.0077    0.1390   18.0154    9.0000
    5.0000    9.0000    0.0001   18.0000    9.0000
    6.0000    9.0000    0.0000   18.0000    9.0000
=================================================
```

Ternyata dengan menetapkan titik awal a=20,dibutuhkan 6 iterasi untuk memenuhi batas F.toleransinya.dan diperoleh dan konvergen ke x=9.0000 sebgai akar persamaan fungsi f(x)=x^2-81. >> syms x

```
>> a=81;
>> f =x^2-81;
>> newton(f,a,tol,n)
================================================
  Iterasi(n)      x         f(x)     tur_f(x)    x_baru
================================================
  1.0e+003 *
    0.0010    0.0810    6.4800    0.1620    0.0410
    0.0020    0.0410    1.6000    0.0820    0.0215
    0.0030    0.0215    0.3807    0.0430    0.0126
    0.0040    0.0126    0.0785    0.0253    0.0095
    0.0050    0.0095    0.0097    0.0190    0.0090
    0.0060    0.0090    0.0003    0.0180    0.0090
    0.0070    0.0090    0.0000    0.0180    0.0090
    0.0080    0.0090    0.0000    0.0180    0.0090
================================================
```

Untuk pemilihan titik awal a=81 diperoleh solusi x=1.0e+003 *0.0090, karena untuk f(81)= 1.0e+003 *6.4800. Dengan pemilihan a=81 membutuhkan 8 iterasi untuk mencapai batas F.toleransi.

C. Interpolasi Numerik

❖ Background Masalah

Dalam pengambilan titik-titik data biasanya hanyalah berupa sampel baik yang dihasilkan dari suatu percobaan maupun dihasilkan dari suatu formulasi khusus. Permasalahan yang muncul adalah bagaimana mengidentifikasi sifat keumuman dari titik-titik data yang diperoleh. Sifat keumuman dari suatu data biasanya direpresentasikan oleh fungsi dalam bentuk persamaan. Hampiran fungsi yang biasa digunakan adalah fungsi polinomial atau suku banyak.Secara lebih khusus konsep interpolasi digunakan dalam beberapa pemecahan masalah di antaranya.

- ❖ Memplot kurva mulus pada data diskret
- ❖ Menurunkan atau mengintegralkan data dalam tabel
- ❖ Membaca informasi antar baris dalam tabel
- ❖ Mengevaluasi fungsi secara cepat dan mudah
- ❖ Merepresentasikan fungsi yang kompleks dalam bentuk yang lebih sederhana

Dalam metode numerik diperkenalkan satu topik khusus yang digunakan untuk memfit data-data yanga ada dan mengidentifikasi fungsinya, hal ini dikenal dengan Interpolasi. Selain mengestimasi fungsi atau persamaannya, kadang dibutuhkan pencarian dan perhitungan nilai dari suatu fungsi yang melewati sekumpulan titik data yang diberikan. Ada beberapa metode dalam penyelesaian masalah ini yaitu :

- ❖ Metode Selisih Terbagi Newton
- ❖ Interpolasi Lagange dan
- ❖ Interpolasi dengan Spline.

Pada buku ini akan diperkenalkan salah satu metode dalam Interpolasi yang melibatkan algoritma, dan implementasi program MATLAB. Salah satu metode yang diperkenalkan adalah Metode Selisih Terbagi Newton

Contoh Soal 9: Misal diberikan pasangan data berikut.

Prediksi nilai fungsi dari suatu pasangan data di bawah ini.

X	0	1	-2	2
Y	-1	3	2	5

Implementasi Metode Interpolasi Newton dengan MATLAB

Dengan menggunakan aloritma Interoplasi Newton, berikuut List Program untuk memperoleh selisih terbaginya:

```
function A=divdiff(x,y)
%Fungsi Divided Difference
%Input = x,y vektor dimensi n
%Output = A koefisien

n =length(x);  %menghitung ukuran vektor
A = zeros(n);   %matriks nol ukuran n x n

for i =1:n;
    A(i,1)  =  y(i);    %mengisi  kolom  pertama
dengan vektor y
end
```

```
for j = 2:n,                %setiap kolom diisi
baris per baris
    for  i=1:n-j+1,      %menghitung matriks A,
mengisi kolom ke-2
        A(i,j)=   (A(i+1,j-1)-A(i,j-1))/(x(i+j-
1)-x(i));
    end
end
```

List Program Interpolasi Newton :

Selanjutnya dari selisih terbagi yang diperoleh berikut listing program untuk membentuk fungsi polinomialnya

```
function f=newdiff(x,y)
%memanggil  program  program  yang  lain  dari
koefosoen A
A= divdiff(x,y);
syms z  %Subtitusi simbolic
n = length(x);
temp  =  A(1,n);              %variabel  pengganti
temporary
for i=n-1:-1:1,     %n-1 jalan mundur
    temp = temp*(z-x(i)) + A(1,i);
end
f = temp;
```

Dapat dijelaskan bahwa pada kedua fungsi di atas, merupakan satu kesatuan yang dibuat secara terpisah dalam hal ini, terdapat fungsi dengan nama **divdiff** sebagai bagian untuk mencari selisih terbagi, sedangkan untuk fungsi kedua dinamakan dengan **newdiff** yang digunakan untuk membuat polynomial Newton.

Simulasi Program Selisih Terbagi dan Integrasi Newton

Menjalankan perintah di Command Window :

```
>> x = [0 1 -2 2];
>> y = [-1 3 2 5];
>> A = divdiff(x,y)
A =
   -1.0000    4.0000    1.8333   -0.7083
    3.0000    0.3333    0.4167         0
    2.0000    0.7500         0         0
    5.0000         0         0         0
```

```
>> f = newdiff(x,y)
f =
- z*(((17*z)/24 - 5/12)*(z - 1) - 4) - 1
```

Membandingkan turunan asli fungsi dengan turunan diferensial numeriknya dengan menggunakan metode beda tengah (Central Differences) :

$$f(z) = -z * \left(\left(\frac{17*z}{24-\frac{5}{12}} \right) * (z-1) - 4 \right) - 1$$

Turunan asli dapat dicari dengan menggunakan built function dengan mengguakan diff(f) sekaligus menentukan nilai turunannya di z = 0.

```
f =
 - z*(((17*z)/24 - 5/12)*(z - 1) - 4) - 1

>> f_tur = diff(f)
 f_tur =
 4 - ((17*z)/24 - 5/12)*(z - 1) - z*((17*z)/12 -
9/8)
 >> syms z
>> subs (f_tur,z,0)

ans =
    3.5833
```

Diferensial numerik dengan menggunakan metode beda tengah dengan rumus :

$$\boldsymbol{f}'(z) = \left(\frac{f(z+h) - f(z-h)}{2h} \right)$$

asli dapat dicari dengan menggunakan built function dengan mengguakan diff(f) sekaligus menentukan nilai turunannya di z = 0.

```
>> %Beda Tengah (Center Diferences) CD
>> CD = (subs(f,z,0+h)-subs(f,z,0-h))/(2*h)
Undefined function or variable 'h'.
>> h=0.1
h =
 0.1000
>> CD = (subs(f,z,0+h)-subs(f,z,0-h))/(2*h)
```

```
CD =
    3.5763
>> %Selisih antara Turunan Asli dengan Diferensial
numerik Beda Tengah
>> subs (f_tur,z,0)-CD
ans =
    0.0071
```

```
function CD=centraldifferences(f,z,a,h)
%Menyelesaikan turunan fungsi dengan metode beda
maju di z=a
syms z
CD = (subs(f,z,a+h)-subs(f,z,a-h))/(2*h);
Selisih = subs(diff(f),z,a)-CD
Command Window :
>> centraldifferences(f,z,0,0.1)
CD =
   3.5763
Selisih =
    0.0071
```

D. Integrasi Numerik

Pada mata kuliah kalkulus, salah satu topik utamanya adalah Integral. Penyelesaian integral pada kalkulus telah diperkenalkan sebagai Teorema Dasar Kalkulus (TDK), di dalam TDK telah diulas dalam buku Kalukulus oleh Varberg dkk, tentang bagaimana pembentukan rumusan secara analitik dari suatu masalah integral didekati dengan anti turunan seperti pada rumusan berikut :

$$\int_a^b f(x)\,dx = F(b) - F(a)$$

Permasalahan yang kemudian muncul bahwa tidak semua penyelesaian Integral dari $f(x)$ dapat ditentukan melalui anti turunan dari $f(x)$. Dari permasalahan ini dibutuhkan suatu pendekatan khusus untuk menyelesaiakan masalah Integral yaitu Integrasi Numerik.

Dalam buku Kalkulus didefinsikan Integral tentu dengan :

Misalkan :

P *adalah banyaknya partisi dengan panjang* $\|P\|$

$\Delta\bar{x}_i$ *merupakan panjang setiap interval*

$\bar{x}_i$ *adalah titik tengah antara* $\bar{x}_i$ *dengan* $\bar{x}_{i+1}$

f *adalah suatu fungsi yang terdefinisi pada interval tertutup* $[a,b]$*, jika* $\lim_{\|P\|\to 0} \sum_{i=1}^{n} f(\bar{x}_i)\Delta\bar{x}_i$ *ada, maka dapat dikatakan* f *adalah terintegrasikan pada selang* $[a,b]$ *.*

Sehingga $\int_a^b f(x)\,dx$*, disebut Integral Tentu* f *dari* a *ke* b*, dinyatakan sebagai :*

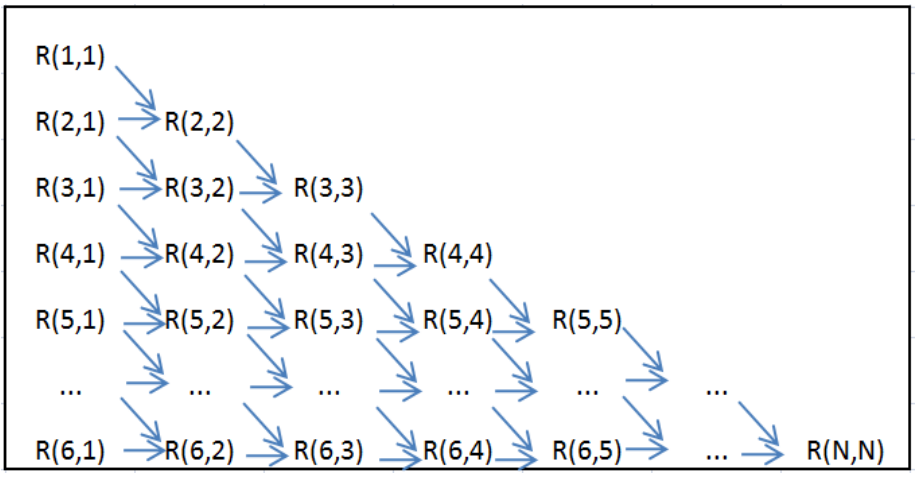
$$\int_a^b f(x)\,dx = \lim_{\|P\|\to 0} \sum_{i=1}^{n} f(\bar{x}_i)\Delta\bar{x}_i$$

Selain itu diperkenalkan pula pendekatan khsusus dalam menyelesaiakn suatu maalah Integral, yakni Jumlah Riemann, metode Trapezoid dan Parabolik.

Konsep dari metode-metode tersebut dengan memandang bahwa menentukan nilai atau integral tentu dari suatu fungsi merupakan menghitung luas daerah di bawah kurva $f(x)$. Dengan konsep ini, jumlah Riemann mendekati kurva dengan kumpulan luaan bangun Persegi Panjang yang berhingga, metode trapezoid dengan jumlahan luas dari bangun Trapesium (Aturan Trapesium) dan aturan parabolik mengaproksimasi kurva dengan menggunakan ruas-ruas parabola, metode ini dikenal dengan Aturan Simpson.

Pada buku ini akan diperkenalkan Metode yang lain yakni Metode Romberg yang digunakan untuk menyelesaiakan Integral tentu. Metode Romberg dalam algoritmanya menggabungkan aturan Simpson rekusrsif dan aturan Boole.

Dalam merumuskan Algoritmanya, Integrasi Romberg menngkatkan keakuratan hampiran dengan ekstrapolasi Richardson yang menghasilkan Jajaran Segitiga seperti pada gambar di bawah ini :



Gambar 7. 9. Jajaran Bilangan Segitiga dai Algoritma Romberg

Dimana :

$$R(j,k) = \frac{4^{k-1}R(j,k-1) - R(j-1,k-1)}{4^{k-1}-1}$$

Untuk $2 \le k \le j$, *dengan nilai awal kuadratur Trapesium adalah*

$$R(1,1) = T_0 = \frac{b-a}{2}\big(f(a)+f(b)\big)$$

Rumusan dan skema di atas kemudian dikembangkan dalam listing program MATLAB.

List Program Integrasi Numerik Metode Romberg (*Editor*)

```
function A= romberg(f,a,b,n)
%Integral fungsi dengan menggunakan Metode Romberg
%a,b batas atas dan bawah pengintegralan
%n banyaknya  baris romberg
syms z;
A = zeros(n+1);
h = b-a; %inisialisasi lebar interval
A(1,1)= (subs(f,z,a)+subs(f,z,b))*(b-a)/2;
for i =2:n+1,
    h = h/2; %iterasi 1
    jumlah = 0 ;   % Inisialisasi
    for k =1:2^(i-2),  %masalah indeks k pada i =2
        jumlah = jumlah + subs(f,z,a+(2*k-1)*h);
    end
    A (i,1) = 1/2 * A(i-1,1) +jumlah*h;
    for j=2:i,
```

```
            A(i,j) = A(i,j-1) + 1/(4^j - 1)*(A(i,j-1)-A(i-
1,j-1));
        end
end
```

Running Program Integrasi Numerik Metode Romberg dengan integral asli dengan built function , dari listing program diatas, dibandingkan hasil integral suatu fungsi dengan integral asli melalui built function dengan integrasi numerik melalui metode Romberg:

```
>> x = [0 1 -2 2];
>> y = [-1 3 2 5];
>> syms z
>> f = newdiff(x,y)
f = - z*(((17*z)/24 - 5/12)*(z - 1) - 4) - 1
>> %Integral Asli dengan menggunakan built in function :
>> Integral_asli = int(f,z,-1,3)
Integral_asli = 20/3
 >> romberg(f,-1,3,5)
ans =

   -4.0000         0         0         0         0         0
    4.0000    4.5333         0         0         0         0
    6.0000    6.1333    6.1587         0         0         0
    6.5000    6.5333    6.5397    6.5412         0         0
    6.6250    6.6333    6.6349    6.6353    6.6354         0
    6.6563    6.6583    6.6587    6.6588    6.6588    6.6589
>> Selisih =abs(( 20/3)-6.6589)
Selisih =    0.0078
```

Dari hasil di atas nampak bahwa dengan Integrasi Romberg membentuk jajaran bilangan segitiga dengan bagian ujung merupakan solusi numerik dari masalah Integrasi numerik yang ada.

E. Persamaan Diferensial Biasa

Masalah Persamaan Diferensial dapat dikatakan sebagai masalah yang paling Populer dalam Pemodelan Matematika. Melalui prinsip laju perubahan terhadap waktu, persamaan diferensial diterapkan dalam berbagai fenomena, sebut saja peluruhan zat dalam kimia, distribusi panas dalam fisika, dinamika populasi dalam biologi, model pertumbuhan ekonomi, bahkan dalam model Traffic Flow dan masih banyak fenomena

lainnya. Dengan beragam fenomena inilah kemudian membentuk kompleksitas antar fenomena yang biasanya dibentuk dalam satu sistem. Tentunya kompleksitas matematis sebagai representasi dari fenomena yang ada, tidak selamanya dapat diselesaikan secara eksak. Penyelesaian Persamaan Diferensial secara eksak dapat kita jumpai pada mata kuliah Persamaan Diferensial.

Bentuk Umum Persamaan Diferensial

Persamaan Diferensial adalah persamaan yang mengandung beberapa turunan dari suatu fungsi yang memiliki variabel bebas. Bentuk umum dai Persamaan Diferensial dapat dituliskan dalam bentuk :

$$y'(t) = f\big(t, y(t)\big), t \geq t_0$$

Dalam hal ini $y(t)$ merupakan fungsi yang tidak diketahui, dan menjadi sasaran sebagai penyelesaian bagi PD. Sedangkan t merupakan variabel bebas.`Kerumitan dari suatu Persamaan Diferensial bergantung pada bentuk $f\big(t, y(t)\big)$. Beberapa Persamaan diferensial dapat diselesaikan secara eksak seperti dengan menerapkan metode pemisahan variabel, atau melalui penyelesaian PD Eksak. Penyelesaian yang dihasilkan dapat dalam bentuk fungsi secara eksplisit. Seperti dalam contoh berikut :

Contoh Soal 10: Tentukan solusi umum dari $y'(x) = 6x^2 - 4x + 2$

Jawaban : $dy = (6x^2 - 4x + 2)dx$

$$\int 1\, dy = \int (6x^2 - 4x + 2)\, dx$$

$y + C_1 = (2x^3 - 2x^2 + 2x + C_2)$; $y = 2x^3 - 2x^2 + 2x + C_3$

Dari contoh soal di atas dapat dilihat bahwa masalah persamaan diferensial, penyelesaiannya berupa fungsi secara eksplisit. Namun dalam banyak kasus masalah Peramaan Diferensial tidak dapat diselesaikan secara eksak. Olehnya itu, diperkenalkanlah beberapa metode numerik yang dapat digunakan dalam menyelesaikan PD.

Pada buku ini mencoba untuk menguraikan kajian komputasi numerik dalam menyelesaikan maslah Persamaaan Diferensial. Seperti

pada sub BAB sebelumnya, akan dipilih satu metode yang dapat memberi solusi hampiran terhadap masalah persamaan diferensial.

❖ Metode Euler

Metode Euler dikenal sebagai metode hampiran yang paling sederhana untuk menyelesaiakn maslah nilai awal. Prinsip pengerjaan Metode Euler ialah dengan membagi interval yang diberikan menjadi n subinterval, dengan panjang masing-masing subinterval adalah h. Misal fungsi yang ditinjau terdefinisi pada interval $[a, b]$. Maka partisi yang dibentuk dapat dinyatakan dengan :

$$a = t_0 < t_1 < t_2 < \cdots < t_{n-1} < t_n = b$$

Dengan menggunakan pendekatan garsis singgung, nilai ietarai awal dapat digunakan pada iterasi berikutnya. Dalam hal ini, dengan mengetahui syarat awal $y_0 = y(a)$, sehingga dengan nilai ini dapat ditentukan nilai $f(a, y_0)$. Melalui penggambaran garis lurus , diperoleh suatu persamaan garis yaitu

$$y = y_0 + (t - t_0) * f(t_0, y_0)$$

Secara iteratif, akhirnya nilai-nilai pada titik nterval berikutnya dapat dituliskan dalam bentuk $y_k = y_{k-1} + h * f(t_{k-1}, y_{k-1})$ dimana y_k merupakan hampiran penyelesaian $y(t_1)$. Proses diatas secara sistematis, dituliskan dalam Algoritma Metode Euler berikut

Algoritma Metode Euler

Masalah : Misal diberikan masalah PD dalam bentuk masalah nilai awal $y'(t) = f(t, y(t))$ dengan syarat awal $y(0) = y_0$ pada interval $[t_0, b]$
1. Dengan demikian dibutuhkan input antara lain n, t_0, b, y_0 $Imaks$ dan 2. Output : $(t_k, y_k) = k = 1,2, \ldots, n$

3. Proses :

- Hitung panjang interval $h = \frac{b-t_0}{n}$
- Untuk iterasi $k = 1,2,3,\dots,n$, gunakan formula

$$t_k = t_{k-1} + h$$

- Hitung $y_k = y_{k-1} + h * f(t_{k-1}, y_{k-1})$

4. Selesai

Implementasi Metode Euler pada Program MATLAB

```
function output = Euler4ode(f,y0,a,b,n)
% Fungsi Euler digunakan untuk mencari fungsi hampiran dari sebuah
PDB.
% Input:
% f fungsi
% (y0, a) titik awal
% (a, b) interval
% n banyaknya subinterval
% Output:
% Nilai -nilai fungsi yang dievalusi pada beberapa titik di [a,b]
 tic;                % start hitung running time
syms x y t;
h = (b - a)/n;
t=[a];
y=[y0];
for i = 2:n+1,
    t=[t;a+(i-1)*h
    y=[y;y(k-1)*h*f(t(k-1),y(k-1)]
end
      toc
```

- **Metode Runge Kutta**

Pada bagian sebelumnya, kita telah melihat bagaimana metode Euler memberikan formula melalui prinsip kerjanya yaitu dengan:

$$y_k = y_{k-1} + h * f(t_{k-1}, y_{k-1})$$

Dengan menggunakan input dari langkah sebelumnya dalam menentukan hampiran sekarang, metode Euler dikenal sebagai metode satu langkah. Pada metode Runge-Kutta 4 memberikan konsep atau prinsip kerja dengan menghitung nilai fungsi $f(t,y)$ pada interval $(t_k, t_k + h)$ dengan ini memberikan hampiran yang lebih baik terhadap y_{k+1}

- Menentukan Gradien yang berada pada titik (t_k, y_k) yaitu $S_1 = f(t_{k-1}, y_{k-1})$
- Menentukan Gradien yang berada pada titik $(t_k + h, y_k + h)$ yaitu $S_2 = f(t_{k-1}, y_{k-1} + h * S_1)$

Selanjutnya untuk Metode Runge Kutta ini dikenal sebagai Runge Kutta orde dua, atau metode Heun sebagai pendekatan dari aturan trapesium. Berikut ini algoritma Runge Kutta Orde Dua.

Algoritma Metode Runge Kutta Orde-Dua

Masalah :

Misal diberikan masalah PD dalam bentuk masalah nilai awal $y'(t) = f(t, y(t))$ dengan syarat awal $y(0) = y_0$ pada interval $[t_0, b]$

1. Dengan demikian dibutuhkan input antara lain n, t_0, b, y_0, f dan
2. Output : $(t_k, y_k) = k = 1,2, \dots, n$
3. Proses :

- Hitung panjang interval $n = \frac{b-t_0}{h}$
- Untuk iterasi $k = 1,2,3, \dots, n$, gunakan formula

  Tentukan $t_k = t_{k-1} + h$

  Tentukan $S_1 = f(t_{k-1}, y_{k-1})$

  Tentukan $S_2 = f(t_{k-1}, y_{k-1} + h * S_1)$

Dari nilai S_1 dan S_2gunakan ada langkah berikutnya

- Hitung $y_k = y_{k-1} + \frac{h}{2}(S_1 + S_2)$

4. Selesai

Selanjutnya metode Rung Kutta juga dapat didekati dengan menggunakan aturan Simpson, yang dikenal sebagai Metode Runge Kutta orde 4.

Algoritma Metode Runge Kutta Orde Empat

Masalah :

Misal diberikan masalah PD dalam bentuk masalah nilai awal $y'(t) = f(t, y(t))$ dengan syarat awal $y(0) = y_0$ pada interval $[t_0, b]$

1. Dengan demikian dibutuhkan input antara lain n, t_0, b, y_0, f dan
2. Output : $(t_k, y_k) = k = 1,2, \dots, n$

3. Proses :

- Hitung panjang interval $n = \frac{b-t_0}{h}$
- Untuk iterasi $k = 1,2,3, \dots, n$, gunakan formula

Tentukan $t_k = t_{k-1} + h$

Tentukan $S_1 = f(t_{k-1}, y_{k-1})$

Tentukan $S_2 = f(t_{k-1} + \frac{h}{2}, y_{k-1} + h * \frac{S_1}{2})$

Tentukan $S_3 = f(t_{k-1} + \frac{h}{2}, y_{k-1} + h * \frac{S_2}{2})$

Tentukan $S_4 = f(t_{k-1}, y_{k-1} + h * S_3)$

Dari nilai S_1, S_2, S_3 dan S_4gunakan pada langkah berikutnya

- Hitung $y_k = y_{k-1} + \frac{h}{6}(S_1 + 2(S_1 + 2(S_2 + S_3) + S_4))$

4. Selesai

Berikut list program, metode Runge Kutta Orde-4 dalam suatu fungsi **rk4sode :**

Listing Program Metode Runge Kutta Orde Empat

```
function output = rk4sode(f1,f2,x0,y0,t0,a,b,n)
% Fungsi Runge-Kutte 4 (rk4) digunakan untuk mencari fungsi
hampiran dari sebuah PDB.
% Input:
% f fungsi
% (x0, y0, t0) titik awal
% (a, b) interval
% n banyaknya subinterval
% Output:
% Nilai -nilai fungsi yang dievalusi pada beberapa titik di [a,b]

tic;                % start hitung running time
syms x y t;
h = (b - a)/n;
output = [x0 y0 t0];
for i = 1:n,
    K1x = h*subs(f1, {x, y, t}, {x0, y0, t0});
    K1y = h*subs(f2, {x, y, t}, {x0, y0, t0});

    K2x = h*subs(f1, {x, y, t}, {x0 + K1x/2, y0 + K1y/2, t0+h/2});
    K2y = h*subs(f2, {x, y, t}, {x0 + K1x/2, y0 + K1y/2, t0+h/2});

    K3x = h*subs(f1, {x, y, t}, {x0 + K2x/2, y0 + K2y/2, t0+h/2});
    K3y = h*subs(f2, {x, y, t}, {x0 + K2x/2, y0 + K2y/2, t0+h/2});
```

```
    K4x = h*subs(f1, {x, y, t}, {x0 + K3x, y0 + K3y, t0 + h});
    K4y = h*subs(f2, {x, y, t}, {x0 + K3x, y0 + K3y, t0 + h});

    x1 = x0 + (K1x + 2*K2x + 2*K3x + K4x)/6;
    y1 = y0 + (K1y + 2*K2y + 2*K3y + K4y)/6;
    t1 = t0 + h;
    % Masukkan nilai (x1, y1, t1) ke dalam keluaran.
    output = [output; x1 y1 t1];
    x0 = x1; y0 = y1; t0 = t1;
end
toc
```

Untuk memahami prinsip kerja metode Rang-Kutta orde 4berikut diberikan contoh masalah sistem Persamaan Diferensial Biasa, kemudian diimplementasikan Metode Runge Kutta Orde Empat.

Contoh Soal 11: Diberikan Persamaan diferensial ordo-2 :

$$x''(t) = -x(t) + t^2 + 2$$

- Dengan Masalah Nilai Awal : $x(0) = 0; x'(0) = 1$
- Dengan Solusi eksak : $x(t) = sin(t) + t^2$
- Diberikan $a = 0; b = 5; n = 100$

Transformasi Persamaan ordo-2 menjadi sistem persamaan diferensial biasa

Pemisalan

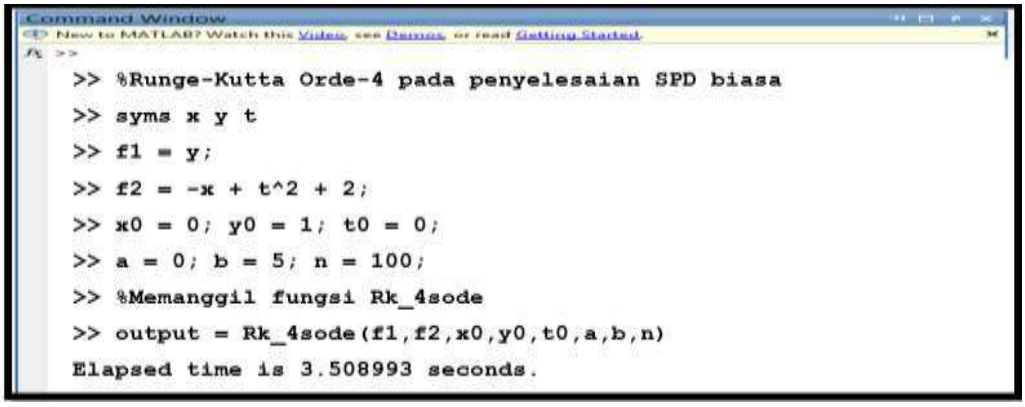
```
x' = y
y' = -x + t^2 + 2

Diketahui solusi eksak:  x(t) = sin (t) + t^2
sehingga                 x'(t)= y(t) = cos (t) + 2t
```

Jadi telah diperoleh suatu sistem persamaan diferensial biasa yang baru yang terdiri dari $\boldsymbol{x'(t)}$ $\boldsymbol{dan}$ $\boldsymbol{y'(t)}$ dengan MNA sebagai berikut:

$$x(0) = 0; x'(0) = y(0) = 1$$

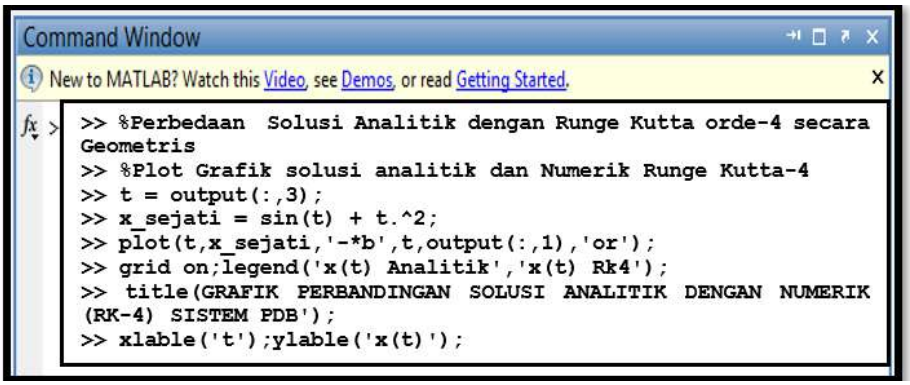
Simulasi Program dengan Menggunakan Metode Runge Kutta 4

Hampiran solusi dari masalah persamaan diferensial biasa tersebut, dapat dieksekusi pada *Command Window* dengan memanggil function Rk_4sode, seperti pada gambar berikut.



Gambar 7. 10. Pemanggilan fungsi Rk_4sode untk diesekusi pada Command Window

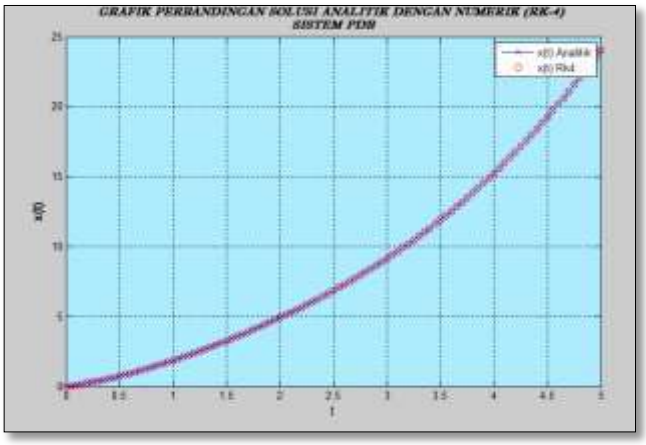
Dengan menggunakan fasilitas tic toc , running program membutuhkan waktu 3.508993 seconds yang menghasilkan nilai-nilai pada beberapa titik asal yaitu pada selang [0,5] dengan 100 jumlah sub interval, dan hasilnya disajikan dalam bentuk table, sebagai solusi hampiran Runge Kutta orde 4, di mana pasangan titik-titiknya akan diplot dan dibandingkan dengan solusi eksak. Berikut perintah plotnya yang dilakukan pada ***Command Window :***



Gambar 7. 11. Perintah untuk melakukan ploting perbandingan solusi analitik dan solusi numerik Runge Kutta Orde 4

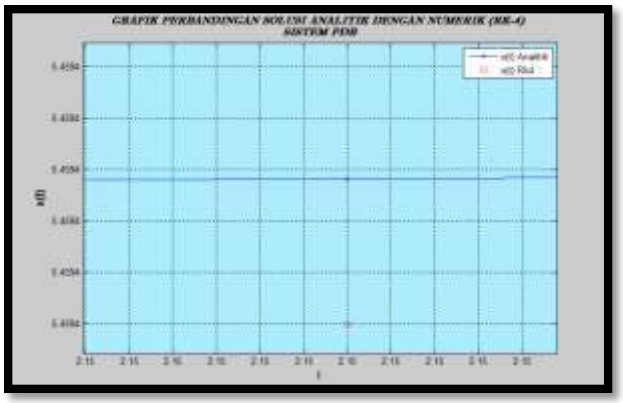
Plotting Grafik Metode Runge Kutta Orde-4.

Berikut adalah grafik perbandingan solusi x(t) eksak dengan solusi x(t) numerik dengan menggunakan metode Runge Kutta RK-4:



Gambar 7. 12. Grafik perbandingan solusi Analitik dengan numerik Runge Kutta Orde 4

Dari grafik yang dihasilkan, terdapat dua jenis series yaitu nilai x(t) yang diperoleh secara analitik (biru) dan nilai x(t) yang diperoleh melalui metode Runge Kutta dengan symbol o. Melalui grafik, tidak terlihat perbedaan yang cukup signifikan, selain karena selisihnya sangatlah kecil, juga karena skala yang digunakan masih besar. Padahal, pada prinsipnya nilai hampiran berbeda dengan nilai sebenarnya, oleh karena itu untuk melihat posisi titiknya perlu dilakukan pembesaran gambar dengan demikian skalanya menjadi kecil. Berikut hasil pembesaran disalah satu titik hampiran :

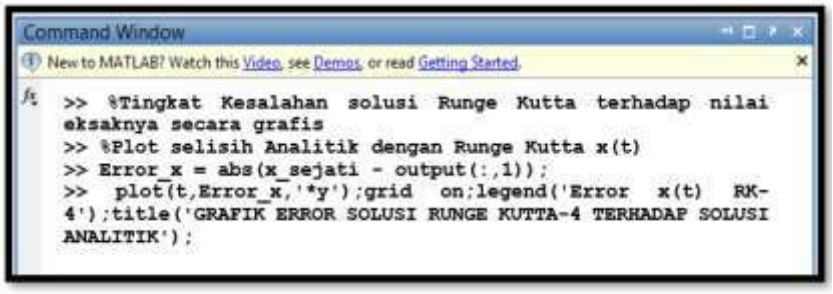


Gambar 7. 13. Pembesaran hasil plotting solusi x(t) secara analitik dengan Runge Kutta orde 4

Mengevaluai Selisih Runge Kutta Orde-4 dengan solusi sejati .

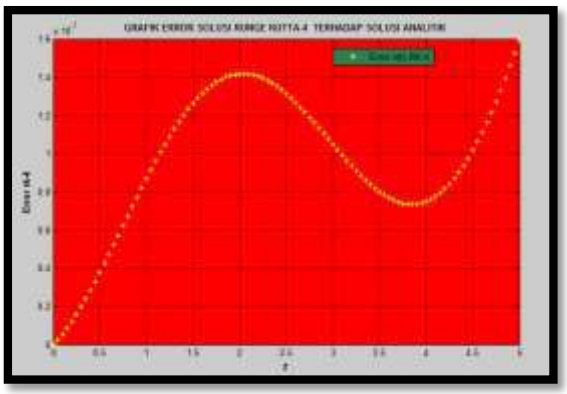
Melanjutkan perintah sebelumnya di Command Window, pada bagian ini akan ditinjau tingkat kesalahan (*error*) dari solusi x(t) yang dihasilkan

metode Runge Kutta orde-4 terhadap solusi eksak yang dievaluasi pada nilai-nilai t yang berada di selang [0,5]. Berikut perintahnya di ***Command Window :***



Gambar 7. 14. Running Program unuk melakukan plotting error Solusi Numerik Runge Kutta Orde 4

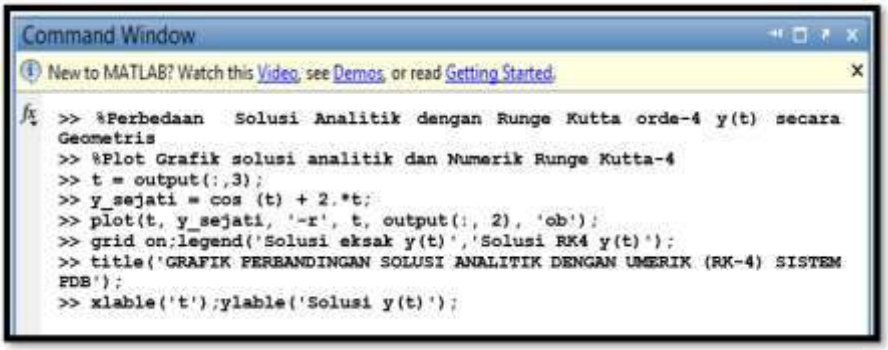
Grafik nilai Error RK-4.



Gambar 7. 15. Perubahan Nilai Error dari Metode Runge Kutta terhadap Solusi Numerik

Dari nilai error yang dievaluasi pada setiap titik-titik t pada selang [0,5] diperoleh dengan menghitung nilai mutlak dari selisih solusi eksak dengan solusi hampiran dimasing-masing titik yang dievaluasi, ternyata tidak linear, dan nilai-nilai eror tersebut sangatlah kecil eror maximal yang dihasilkan 1.6×10^{-7} . Hal ini mengindikasikan, bahwa metode Runge Kutta orde-4 memiliki tingkat keakuratan yang baik untuk menyelesaikan persamaan difernsial biasa.

Mengevaluai Solusi y(t) eksak dengan solusi y(t) hampiran dengan menggunakan metode Rung Kutta orde-4 .

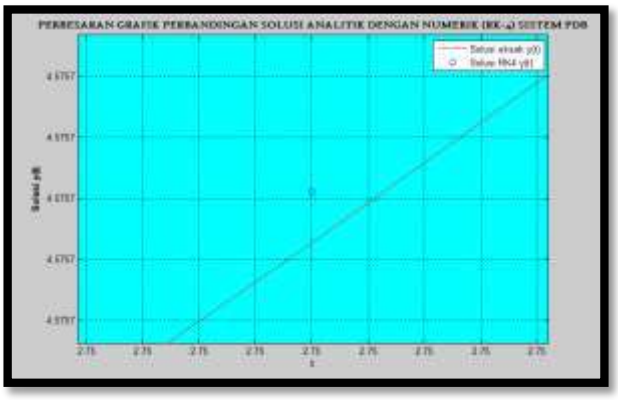


Gambar 7. 16. Running Program pada Command Window untuk mengevaluasi solusi y(t) eksak dengan solusi y(t) Runge Ktta Orde 4

Berikut adalah grafik perbandingan solusi y(t) eksak dengan solusi y(t) numerik dengan menggunakan metode Runge Kutta RK-4:

Gambar 7. 17. Grafik perbandingan solusi y(t) eksak dengan solusi y(t)

Seperti halnya pada bagian perbandingan solusi x(t) eksak dengan x(t) RK-4. Pada bagian ini, juga terdapat dua jenis series yaitu nilai y(t) yang diperoleh secara analitik (merah) dan nilai y(t) yang diperoleh melalui metode Runge Kutta dengan symbol **o**. Terlihat pula bahwa grafik yang dihasilkan tidak memberikan perbedaan yang signifikan. Oleh karena itu untuk melihat perbedaannya melalui visualisasi grafik perlalu dilakukan pembesaran

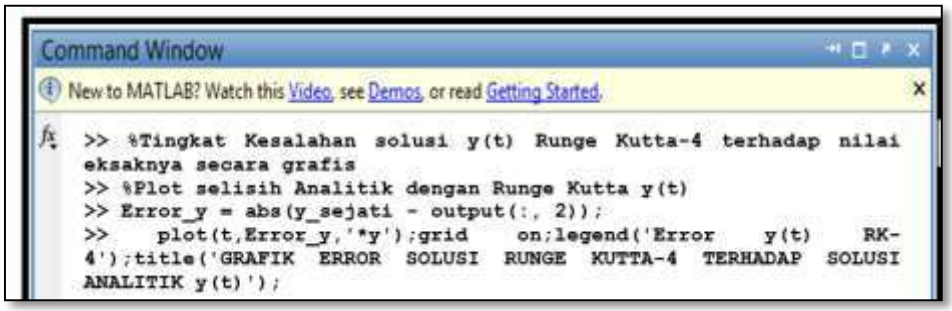


Gambar 7. 18. Pembesaran Grafik Perbandingan solusi y(t) eksak dengan solusi y(t)

Dari gambar 7.18 adalah gambar yang telah diperbesar, terlihat jelas bahwa tetap ada perbedaan antara solusi hampiran dengan solusi sejati. Adapun nilai-nilai pada sumbu vertikalnya sama, hal ini diseBabkan karena keterbatasan penggunaan angka penting dalam menunjukkan skala sumbu.

Mengevaluai Selisih Runge Kutta Orde-4 dengan solusi sejati .

Seperti pada bagian sebelumnya, pada bagian ini juga akan ditinjau tingkat kesalahan (*error*) dari solusi y(t) yang dihasilkan metode Runge Kutta orde-4 terhadap solusi eksak yang dievaluasi pada nilai-nilai t yang berada di selang [0,5]. Berikut perintahnya pada ***Command Window*** **:**



Gambar 7. 19. Menjalankan program untuk mengukur selisih antara solusi analitik dengan solusi numerik y(t)

Grafik Error solusi y(t) Runge Kutta-4 terhadap solusi y(t) Numerik.

Gambar 7. 20. Grafik Error solusi y(t) Runge Kutta-4 terhadap solusi y(t) Numerik.

Dari grafik yang dihasilkan, nilai error yang dievaluasi pada setiap titik-titik t pada selang [0,5] diperoleh dengan menghitung nilai mutlak dari selisih solusi eksak y(t) dengan solusi hampiran y(t) dimasing-masing titik yang dievaluasi, dan juga memberi hasil bahwa error tidak konstan dan tidak linear, dan nilai-nilai eror tersebut sangatlah kecil eror maximal yang dihasilkan 1.4×10^{-7} . Hal ini mengindikasikan, bahwa metode Runge Kutta orde-4 memiliki tingkat ketelitian yang baik untuk menyelesaikan persamaan difernsial biasa.

Sebagai kesimpulan solusi dari suatu persamaan differensial secara eksak adalah fungsi yang memenuhi persamaan diferensial yang diberikan dan juga memenuhi syarat awal fungsi tersebut. Sedangkan pada metode numerik menghasilkan tabel nilai-nilai fungsi pada beberapa nilai variabel bebasnya, yang tidak dinyatakan secara eksplisit dalam bentuk rumus fungsi.

Sementara metode Runge Kutta Orde-4 merupakan salah satu metode numerik yang dapat digunakan untuk menyelesaikan persamaan diferensial, baik persamaan tunggal maupun dalam bentuk sistem. Prinsip penyelesaian persamaan diferensial dengan menggunakan metode Runge Kutta berusaha mendapatkan ketelitian yang lebih tinggi, dan sekaligus menghindarkan keperluan mencari turunan yang lebih tinggi dengan jalan mengevaluasi fungsi f1(t,x) dan f2(t,y) pada titik terpilih dalam setiap selang langkah.

F. Rangkuman

Masalah Sistem Persamaan Linear

- Sistem Persamaan Linear adalah salah satu topik khusus dalam aljabar yang banyak digunakan pada kajian keilmuan lain seperti optimasi, sistem arus dari keilmuan eketronika, dan masih banyak lainnya. Dalam menentukan Sistem Persamaan Linear dapat dilakukan melalui metode langsung dan metode iteratif.
- Secara spesifik metode terbuka terdiri atas metode Eliminasi Gauss, Metode eleminasi Gauss Jordan dan Metode Inverse. Sedangkan Metode Iteratif, terdapat metode Iterasi Jacobi, Iterasi Gauss Seidel dan Iterasi.
- Dalam komputasi numerik, metode Iterasi Jacobi dan Gauss Seidel membangun gagasan berupa penentuan solusi awal dan penentuan langkah iterasi. Perbedaan mendasar antara metode Jacobi dan Metode Gauss berada pada langkah iterasi. Pada meode Jacobi, pada setiap iterasinya mendahulukan penemuan solusi secara simultan pada setiap iterasinya dengan formula $x_i^{(k+1)} = \frac{b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(k)}}{a_{ii}}$, $k = 0,1,2, \ldots.$
- Sedangkan metode iterasi Gauss Seidel, penemuan solusi hampiran pada setiap langkah langsung dapat digunakan pada iterasi yang sama dengan rumusan $x_i^{(k)} = \frac{1}{a_{ii}} (b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)})$.
- Aspek matematika komputasi yang ditekankan pada pemecahan masalah penentuan akar persamaan non linear adalah bagaimana menerapkan konsep perulangan dan penyeleksian kondisi pada metode yang ada dalam

Masalah Akar Persamaaan Non Linear

- Akar persamaan non linear dapat dipahami sebagai penentuan nilai x yang mengakibatkan $f(x) = 0$. Dalam penentuan akar persamaan non linear yang paling sederhana adalah dengan metode pemfaktoran pada penentuan akar persamaan kuadrat. Didalam menyelesaikan masalah pencarian akar persamaan linear secara numerik dapat diterapkan dua metode yakni metode kurungan dan metode terbuka.
- Metode bagi dua adalah salah satu jeni smetode kurungan. Prosedur metode ini didahului dengan menaksir dimana letak akar $f(x)$ dengan cara mengurung akar pada interval tertutup [*a*,*b*] dengan syarat $f(a)$ dan $f(b)$ haruslah berbeda tanda. Kenyataan ini didasarkan pada konsep teorema harga menengah. Prosedur yang digunakan untuk mempersempit jebakan adalah dengan cara kurungan sebelumnya (interval tertutup [*a*,*b*]) atas dua bagian yang sama besar. Yakni interval $a < x < t$ dan interval $t < x < b$, dengan demikian *t* merupakan titik tengah interval [a,b].
- Metode newton Raphson merupakan salah satu jenis metoterbuka memanfaatkan garis singgung kurva, kemudian nilai *x* dimana garis singgung memotong sumbu-*x* dipilih sebagai titik iterasi berikutnya. Persamaan garis singgung di titik (x_0,*f*(x_0)) adalah :$y = f(x_0) + (x - x_0) \cdot f'(x_0)$. Misalkan titik potong garis singgung tersebut dengan sumbu-*x*, adalah titik $(x_1,0)$, maka diperoleh :$x_1 = x_0 - \frac{f(x_o)}{f'(x_0)}$.

Interpolasi

- Interpolasi merupakan salah satu pendekatan yang dapat digunakan dalam menentukan suatu fungsi aproksimasi dari sekumpulan titik yang diberikan. Selain mengestimasi bentuk fungsi, interpolasi juga dapat dipandang sebagai proses penentuan titik atau nilai yang diapit oleh titik-titik yang berkaitan.

- Dalam mengestimasi fungsi dari sekumpulan titik yang diberikan dapat berupa fungsi polinomial yang dipandang sebagai hampiran suatu fungsi yang tidak diketahui. Salah satu metode numerik yang dapat diimplementasikan dalam proses interpolasi adalah metode selisih terbagi yang menghasilkan polinomial Newton. Metode yang lain yang dapat diterapkan adalah polinomial lagrange dan Spline.

Integrasi Numerik

- Integrasi Numerik merupakan bagian dari metode numerik untuk menentukan solusi hampiran dari suatu masalah integral tentu baik yang dapat diselesaikan secara analitik maupun tidak. Pada bagian pembahasan telah diperkenalkan bahwa konsep penentuan integral tentu secara numerik adalah memberikan hitungan luasan dari suatu area pendekatan untuk menghitung luas daerah di bawah kurva.
- Metode-metode yang dapat diimplementasikan adalah metode Jumlah Riemann dengan hampiran bangun segi empat beraturan, metode Trapesoid dengan hampiran ruas-ruas bangun trapesium, dan metode Simpson dengan hampiran ruas-ruas parabola.
- Secara lebih khusus metode yang mengintegrasikan rumus rekursif trapesium, Simpson dan Boole dikenal sebagai Integrasi Romberg. Selain itu terdapat pula metode Romberg dengan menggunakan ekstrapolasi Richardon sebagai metode yang dapat meningkatkan keakuratan hampiran solusi dari masalh Integral yang dipecahkan. Adapun aturan dari metode ini dikenal dengan Teorema Integrasi Romberg- Ekstrapolasi Richardson.

G. Latihan

- **Sistem Persamaan Linear**

Latihan 7. 1: Tentukan solusi dari Sistem Persamaan Linear berikut :

$$4x_1 - 6x_2 + 4x_3 - x_4 = 0.43$$

$$-6x_1 - 14x_2 - 11x_3 + 3x_4 = -1$$
$$4x_1 - 11x_2 + 10x_3 - 3x_4 = 0.82$$
$$-x_1 + 3x_2 - 3x_3 + x_4 = -0.23$$

dengan menggunakan :

- ❖ Metode Inverse
- ❖ Metode Langsung Gauss Jordan
- ❖ Metode Iterasi Jacobi
- ❖ Metode Gauss Seidle

Latihan 7. 2: Tentukan solusi dari Sistem Persamaan Linear Berikut

$$2x_1 - x_2 + 10x_3 = -11$$
$$3x_2 - x_3 + 8x_4 = -11$$
$$10x_1 - x_2 + 2x_3 = 6$$
$$-x_1 + 11x_2 - x_3 + 3x_4 = 25$$

Dengan ketentuan sebagai berikut :

- ❖ Tentukan matriks Koefisien, vektor konstanta dan vektor awalnya
- ❖ Gunakan iterasi Jacobi secara manua sampai iterasi ke lima
- ❖ Gunakan Perintah MATLAB untuk memberikan solusi numeriknya
- ❖ Narasikan metode iterasi Jacobi sebagai metode numerik dalam menyelesaikan masalah SPL

Latihan 7. 3: Terapkan algoritma Gauss Seidel untuk dalam menyelesaikan SPL berikut :

$$10x_1 - x_2 + 2x_3 = 6$$
$$-x_1 + 11x_2 - x_3 + 3x_4 = 25$$
$$2x_1 - x_2 + 10x_3 - x_4 = -11$$
$$3x_2 - x_3 + 8x_4 = 15$$

Dengan ketentuan sebagai berikut :

- ❖ Tentukan matriks Koefisien, vektor konstanta dan vektor awalnya
- ❖ Gunakan iterasi Gauss Seidel secara manua sampai iterasi ke lima
- ❖ Gunakan Perintah MATLAB untuk memberikan solusi numeriknya
- ❖ Narasikan metode iterasi Gauss Siedel sebagai metode numerik dalam menyelesaikan masalah SPL

Sumber : Sahid (2005)

❖ **Akar Persamaan Non Linear**

Latihan 7. 4: Diketahui $f(x) = x^2 - 2x + 3$.

- ❖ Dengan *"rumus abc"* tentukan akar-akar dari $f(x)$
- ❖ Nyatakan $f(x) = 0$ dalam tiga bentuk $x = g(x)$ yang berbeda
- ❖ Hitung lima iterasi pertama untuk masing-masing bentuk yang saudara pilih pada pertanyaan (b).
- ❖ Dengan menggunakan kriteria konvergen $TOLN = 10^{-3}$, apakah hasil yang saudara dapatkan pada pertanyaan (c) konvergen ?
- ❖ Jika konvergen, ke salah satu akar [jawab (a)], gantilah nilai awal yang saudara gunakan agar konvergen ke nilai akar yang lainnya.

Latihan 7. 5: Tentukan akar dari : $f(x) = x - e^{-x}$

- ❖ Nyatakan $f(x) = 0$ dalam tiga bentuk $x = g(x)$ yang berbeda
- ❖ Hitung lima iterasi pertama untuk masing-masing bentuk yang saudara pilih pada pertanyaan (b).
- ❖ Dengan menggunakan kriteria konvergen $Ftol = 10^{-3}$, apakah hasil yang saudara dapatkan pada pertanyaan (c) konvergen ?

Latihan 7. 6: Tentukan akar positif dari $f(x) = 10 - x^2$ dengan menggunakan, $Maxi := 5$; $Ftol = 10^{-3}$, dan nilai awal pilihan anda.

- ❖ Metode Grafik (tabel) dalam 2DP, gambarkan grafik $f(x) = 10 - x^2$
- ❖ Metode Bagi Dua
- ❖ Metode Newton Rapshon

Latihan 7. 7: Tentukan akar dari $f(x) = x^3 - 6x^2 + 11x - 6$, lakukan dengan $Maxi = 3$.

- ❖ Metode Grafik
- ❖ Metode Bagi Dua, gunakan nilai awal $x_0 = 2.5$ dan $x_1 = 3.6$
- ❖ Metode Newton Rapshon dengan $x_0 = 3.6$

❖ Interpolasi Numerik

Latihan 7. 8: Gunakan metode Selisih terbagi newton untuk menentukan Polynomial dar pasangan titik-titik berikut : (0,1),(1,1), (2,2), dan (4,5) secara manual , dengan mengikuti formula selisih terbag Newton.

Latihan 7. 9: Gunakan metode Selisih terbagi Newton untuk menentukan polynomial Newton dengan menggunakan Program MATLAB :

- (0,-2),(1,2), (2,4), (3,4),(4,2),(5,-2)
- (1,8),(2,17),(3,24),(4,29),(5,32),(6,33)
- (0,5),(1,5),(2,3),(3,5),(4,27),(5,45),(6,95)
- (0,1),(1,-1),(4,1),(6,-1)

❖ Integrasi Numerik

Latihan 7. 10: Implementasi metode Integrasi dengan aturan Romberg untuk menentukan solusi numerik dari integral tentu berikut :

- $\int_0^3 \frac{sin\,(2x)}{2+x^3}\,dx$
- $\int_0^3 e^{-2x}\sin(4x)\,dx$
- $\int_0^2 \frac{1}{1+x^2}\,dx$

Latihan 7. 11: Hitunglah hampiran-hampiran dengan metode Romberg $R(N,N)$ untuk menaksir nilai $\int_0^1 \frac{4}{1+x^2}\,dx = \pi$ untuk $N = 1,2,3,..,10$.

DAFTAR PUSTAKA

Addison, Wesley.1999. Introduction to Numerical Analysis Alastair Wood School of Computing and Mathematics. University of Bradford.

Chapman SJ. 2008., MATLAB Programming of Enginering Fourth Edition. Thomson : Canada.

Cheney W., Kincaid D., 2004. Numerical Mathematics and Computing Sixth Edition. Thomson Brooks/Cole. United States of America.

Gazali, W. 2007. Kalkulus Lanjut Edisi 2.Graha Ilmu : Yogyakarta.

Irawan,FA. 2012. Buku Pintar Pemrograman MATLAB. Buku Seru: Cara Cepat dan Mudah Mempelajari Bahasa Pemrograman Penyelesaian Masalah Komputasi. Media Kom : Jakarta.

Iswanto RJ., 2012. Pemodelan Matematika. Aplikasi dan Terapannya. Graha Ilmu : Yogyakarta.

Kiusalaas J.,Numerical Methids in engineerring wiyh MATLAB. Second Edition. Cambridge University Press : New York.

Robinson, J.C., 2004. An Introduction Ordinary Differential Equations. Cambridge University Press: New York.

Sahid, 2005., Pengantar Komputasi Numerik dengan MATLAB. Andi : Yogyakarta.

Suarga. 2007., Fisika Komputasi: solusi Problema Fisika dengan MATLAB. Andi : Yogyakarta.

S.D. Conte, Carl De Boor, 1980. Elementary Numerical Analysis An Algorithmic Approach, Third Edition. Mc-Graw-Hill. Book Company. New York.

Varberg, Purcell, Rigdon. 2007. Kalkulus Edisi Kesembilan Jilid 1. Erlangga :Jakarta.

RIWAYAT PENULIS

Zulfiqar Busrah, S.Si., M.Si adalah seorang dosen Tadris Matematika Fakultas Tarbiyah IAIN Parepare terhitung sejak Maret 2018, lahir di Tanuntung Bulukumba 01 Oktober 1989. Dalam Riwayat Pendidikannya, tahun 2001 tamat di SD 253 Tanuntung, kemudian melanjutkan pendidikannya di SMP Negeri 1 Herlang dan lulus pada tahun 2004. Pada tahun yang sama, dia melanjutkan pendidikan pada tingkat menengah atas di SMA Negeri 1 Herlang dan lulus pada tahun 2007. Kemudian masuk pada Perguruan Tinggi Universitas Hasanuddin (Unhas) tepatnya pada Program Studi Matematika Fakultas MIPA pada tahun 2007 dan menyelesaikannya pada tahun 2011. Satu tahun berikutnya, dia melanjutkan Pendidikannnya pada Sekolah Pascasarjana Program Studi Matematika Terapan Institut Pertanian Bogor (IPB) dan menyelesaikannnya pada bulan oktober 2014. Dalam menjalankan studinya di IPB tepat pada semester III (2013) dia diamanahkan sebagai asisten dalam Praktikum Matematika Komputasi.

Dalam penyelesaian studinya, topik skripsi yang dipilihnya adalah Analisis Numerik dengan judul Penyelesaian Persamaan Panas pada Koordinat Silinder dengan Menggunakan Metode Eksplisit FTCS dan Metode Implisit BTCS. Di mana dalam skripsi tersebut menggunakan pendekatan Matematika Komputasi dan memilih MATLAB sebagai bahasa pemrograman dalam menyelesaikan Model Matematika Persamaan Panas Koordinat Silinder. Selanjunya dalam menyelesaikan tesisnya yang berjudul Perdandingan Metode Analisis Homotopi dengan Metode Iterasi Variasional dalam Menyelesaikan Masalah Gelombang Internal pada Lapisan Atmmosfer, dia kembali memilih MAPLE dan MATLAB sebagai media komputasi dalam penelitiannya.

Pada tahun 2015, penulis memulai profesinya sebagai dosen tetap pada Perguruan Tinggi Swasta Universitas Cokroaminoto Palopo pada Program Studi Teknik Informatika Fakultas Teknik Komputer. Dalam perjalanan karirnya sebagai tenaga pengajar, dia diamanahkan mengampu Mata kuliah Sistem Informasi Geografis (SIG), Pengolahan Citra Digital (PCD), Kriptografi, dan Fisika Komputasi. Dari Mata Kuliah tersebut, beberapa diantaranya juga tidak terlepas dari penggunaan MATLAB sebagai Bahasa Pemrograman yang relevan yang menunjang mata kuliah tersebut.

Pada Kemetrian Agama satuan kerja IAIN Parepare Fakultas Tarbiyah Program Studi Tadris Matematika. Dalam kesempatan ini, penulis menggeluti kembali Ilmu Matematika dengan mengampuh Mata Kuliah, Kalkulus, Geometri Analitik, Persamaan Diferensial, Metode Numerik dan Matematika Komputasi. Dari integrasi mata kuliah ini, penulis berinisiatif membuat buku ajar yang dapat digunakan sebagai pegangan Mahasiswa agar pembelajaran berjalan secara terarah, sistematis dan Komprehensif.

Buku ini merupakan buku ajar yang dikembangkan untuk mengakomodir Mata Kuliah Matematika Komputasi dan Metode Numerik. Kumpulan Isi dari buku ini diarahkan dalam satu kesatuan yang mengkombinasikan beberapa mata kuliah seperti Kalkulus, Aljabar Linear, Persamaan Diferensial dan Metode Numerik yang dikaji dengan menggunakan pendekatan komputasi Matematika.

Pemilihan Pemrograman MATLAB sendiri didasarkan pada cakupan pemecahan masalah yang dapat merepresentasikan berbagai topik bahasan seperti topik dasar dalam matematika seperti kajian Vektor, Matriks, Fungsi dan Penyajian fungsi melalui grafik. Untuk memudahkan penggunaan buku ini, maka dalam pembahasan di setiap BAB nya diawali penyajian kemampuan dasar yang menjadi target capaian.

Dalam penyajian isi, lebih banyak memberikan contoh-contoh script program yang dilengkapi dengan penjelasan dan interpretasi hasil. Pada setiap BAB nya juga dilengkapi dengan soal-soal latihan yang memungkinkan pengguna untuk mengeksplorasi kemampuannya. Diharapkan dengan adanya buku ini, dapat menjadikan mata kuliah yang terkait dapat lebih terarah, sistematis, efisien dan komprehenif, baik untuk peningkatan wawasan dan keterampilan komputasi matematika secara individu bagi mahasiswa maupun untuk peningkatan kualitas lembaga dalam penyediaan media pembelajaran Komprehensif.

BUKU AJAR MATEMATIKA KOMPUTASI BERBASIS PEMROGRAMAN MATLAB

Kaaffah Learning Center
CV. KAAFFAH LEARNING CENTER
Kompleks Griya Bumi Harapan Permai B44
Jalan. Syamsul Alam Bulu, Parepare, Sulawesi Selatan
Telp/Fax. 0421-2914373
E-mail. kaaffahlearningcenter@gmail.com

MathWorks