

خودآموز سریع MATLAB

نویسنده: دکتر علی مس فروش

عضو هیات علمی دانشگاه صنعتی شاهرود

به یاد مادرم که همه مهر بود و صفا

پیشگفتار

MATLAB یک نرم‌افزار قدرتمند برای انجام محاسبات ریاضی است که به کمک آن می‌توان مسایل محاسباتی را به سادگی و با سرعت زیاد حل کرد. این نرم‌افزار در حالت پایه‌ای خود و بدون استفاده از بسته‌های جانبی، دارای دستوراتی است که با استفاده از آنها می‌توان به حل دسته گسترده‌ای از مسایل، مانند حساب دیفرانسیل و انتگرال، محاسبات ماتریسی، رسم نمودارها در حالت دوبعدی و سه‌بعدی، معادلات دیفرانسیل معمولی و ... پرداخت. با استفاده از برخی از بسته‌ها که برای MATLAB وجود دارند، می‌توان برخی مسایل پیچیده‌تر را حل کرد. همچنین برای رشته‌های مختلف مانند مهندسی برق، مهندسی مکانیک، فیزیک و برخی رشته‌های دیگر، بسته‌های ویژه‌ای وجود دارند که با استفاده از آنها می‌توان مسایل مربوط به این رشته‌ها را حل کرد. در این کتاب تلاش شده است اصول اولیه کار با MATLAB و برخی از بسته‌های پرکاربرد آن آموزش داده شود. بدیهی است که کاربران پس از آشنایی با MATLAB می‌توانند بسته‌های مورد نیاز خود را شناسایی کنند و با مطالعه راهنمای مربوط به آنها به حل مسایل خود بپردازند. در این کتاب هیچ مطلب اضافه‌ای گفته نشده است، لذا از خوانندگان گرامی خواهشمندم اگر با MATLAB آشنایی ندارند، از پراکنده‌خوانی کتاب خودداری کنند و با صرف اندکی زمان و با صبر و پرهیز از شتاب در یادگیری به مطالعه کتاب بپردازند. بدیهی است که در هنگام مطالعه کتاب باید همواره به نرم‌افزارهای مورد نیاز دسترسی داشته باشید تا تمام دستوراتی که با آن برخورد می‌کنید را در محیط نرم‌افزار آزمایش کنید.

برای دسترسی به ویدیوهای آموزشی به صورت کاملاً رایگان می‌توانید به کانال یوتیوب من به آدرس <https://www.youtube.com/@DrMesforushAcademy> مراجعه فرمایید. در پایان از خوانندگان گرامی تقاضا دارم تا هرگونه اشکال تایپی، ایراد نوشتاری، یا هر پیشنهادی در جهت بهتر شدن کتاب را از طریق ali.mesforush@shahroodut.ac.ir به اینجانب منتقل فرمایند. این کتاب به مناسبت درگذشت مادرم به صورت رایگان در اختیار عموم قرار داده شده است.

علی مس‌فروش

پنجم اسفند ۱۴۰۱

مشهد

نسخه
دایگان

فهرست مطالب

۱	این فصل را حتما بخوانید!	۱
۱	۱.۱ معرفی Octave	۱
۲	۲.۱ تهیه و نصب Octave	۲
۳	۳.۱ اجرای Octave و آشنایی با محیط	۳
۴	۴.۱ نصب بسته در Octave	۴
۷	۲ آغاز کار	۷
۷	۱.۲ عملگرهای محاسباتی	۷
۸	۲.۲ تعریف و استفاده از متغیرها	۸
۱۴	۳.۲ توابع کتابخانه‌ای ریاضی در MATLAB	۱۴
۱۴	۱.۳.۲ اعداد مختلط	۱۴
۱۵	۲.۳.۲ توابع مقدماتی ریاضی	۱۵
۱۷	۴.۲ m-فایل	۱۷
۲۲	۵.۲ تمرین	۲۲
۲۷	۳ بردارها و ماتریس‌ها	۲۷
۲۷	۱.۳ آرایه‌های یک‌بعدی	۲۷
۳۲	۲.۳ آرایه‌های دو بعدی	۳۲
۳۳	۱.۲.۳ تعریف ماتریس	۳۳
۳۸	۲.۲.۳ استخراج بخشی از ماتریس	۳۸

۴۵	۳.۳ تمرین
----	---------------------

۴ عملیات ریاضی روی آرایه‌ها

۵۱	۱.۴ عملیات محاسباتی روی آرایه‌ها
۵۲	۱.۱.۴ جمع و تفریق
۵۳	۲.۱.۴ ضرب آرایه‌ها
۵۵	۳.۱.۴ تقسیم آرایه‌ها
۵۹	۴.۱.۴ به توان رساندن آرایه‌ها
۶۱	۲.۴ اعمال توابع پیش‌ساخته بر آرایه‌ها
۶۳	۳.۴ توابع پیش‌ساخته مخصوص آرایه‌ها
۶۸	۴.۴ تمرین

۵ اصول برنامه‌نویسی در MATLAB

۷۵	۱.۵ ایجاد، اجرا و ذخیره‌سازی یک m-فایل
۷۷	۲.۵ عبارات محاسباتی و دستورات ورودی و خروجی
۸۳	۳.۵ عملگرهای مقایسه‌ای و دستورات شرطی
۸۵	۴.۵ دستورات شرطی
۸۶	۱.۴.۵ ساختار if-end
۸۸	۲.۴.۵ ساختار if-else-end
۹۱	۳.۴.۵ ساختار switch-case
۹۲	۵.۵ حلقه‌های تکرار
۹۳	۱.۵.۵ حلقه for-end
۹۶	۲.۵.۵ حلقه while-end
۱۰۱	۶.۵ توابع غیر کتابخانه‌ای
۱۰۲	۱.۶.۵ ایجاد و استفاده از تابع
۱۰۶	۲.۶.۵ توابع بدون نام
۱۰۹	۷.۵ تمرین

۶ رسم نمودارهای دوبعدی

۱۱۳	۱.۶ دستورات رسم نمودار
-----	----------------------------------

۱۱۳ دستور plot	۱.۱.۶
۱۲۴ دستور fplot	۲.۱.۶
۱۲۵ دستور line	۳.۱.۶
۱۲۷ رسم نمودارهای قطبی	۴.۱.۶
۱۲۹ رسم چند نمودار در کنار هم	۲.۶
۱۳۱ قالب‌بندی نمودارها	۳.۶
۱۳۲ قالب‌بندی نمودارها به کمک دستورات MATLAB	۱.۳.۶
۱۳۷ قالب‌بندی متن درون شکل	۲.۳.۶
۱۴۳ ایجاد تغییرات در محورهای مختصات	۳.۳.۶
۱۴۹ برخی توابع خاص در رسم نمودار	۴.۶
۱۵۹ تمرین	۵.۶

۷ چندجمله‌ای‌ها، برازش منحنی و درونیابی

۱۶۳ چندجمله‌ای‌ها	۱.۷
۱۶۴ تعریف چندجمله‌ای‌ها	۱.۱.۷
۱۶۴ محاسبه مقدار چندجمله‌ای	۲.۱.۷
۱۶۶ ریشه چندجمله‌ای‌ها	۳.۱.۷
۱۶۷ عملیات جبری روی چندجمله‌ای‌ها	۴.۱.۷
۱۶۹ مشتق‌گیری از چندجمله‌ای‌ها	۵.۱.۷
۱۷۰ برازش منحنی	۲.۷
۱۷۰ برازش منحنی با چندجمله‌ای‌ها	۱.۲.۷
۱۷۳ برازش منحنی با توابع دیگر	۲.۲.۷
۱۷۵ درونیابی	۳.۷
۱۸۰ تمرین	۴.۷

۸ کاربرد MATLAB در ریاضیات محاسباتی

۱۸۳ کاربرد در آنالیز عددی	۱.۸
۱۸۳ یافتن ریشه‌های معادلات یک متغیره	۱.۱.۸
۱۸۹ تعیین نقاط ماکزیمم و مینیمم توابع	۲.۱.۸

۳.۱.۸	انتگرال گیری عددی	۱۹۱
۲.۸	کاربرد در معادلات دیفرانسیل با مقدار اولیه	۱۹۵
۳.۸	تمرین	۲۰۰

۹ کاربرد MATLAB در جبر خطی ۲۰۵

۱.۹	تجزیه ماتریس ها	۲۰۵
۱.۱.۹	تجزیه LU	۲۰۵
۲.۱.۹	تجزیه QR	۲۰۸
۳.۱.۹	تجزیه چولسکی	۲۰۹
۴.۱.۹	مقادیر ویژه و بردارهای ویژه	۲۰۹
۵.۱.۹	مقادیر تکین و بردارهای تکین	۲۱۱
۲.۹	توابع مورد استفاده در ماتریس ها	۲۱۲

۱۰ رسم نمودارهای سه بعدی ۲۱۹

۱.۱۰	رسم منحنی های خط	۲۱۹
۲.۱۰	رسم رویه ها	۲۲۱
۳.۱۰	رسم اشکال خاص در فضا	۲۳۰
۴.۱۰	رسم نمودارهای قطبی در فضا	۲۳۳
۵.۱۰	تمرین	۲۳۴

۱۱ جعبه ابزار Symbolic ۲۳۷

۱.۱۱	معرفی جعبه ابزار Symbolic	۲۳۷
۱.۱.۱۱	معرفی متغیرهای نمادین	۲۳۸
۲.۱.۱۱	تعریف عبارات به شکل نمادین	۲۳۹
۳.۱.۱۱	ساده سازی عبارات ریاضی	۲۴۱
۲.۱۱	کاربرد Symbolic در حساب دیفرانسیل و انتگرال	۲۴۶
۱.۲.۱۱	حل معادلات جبری	۲۴۶
۲.۲.۱۱	حدگیری	۲۵۱
۳.۲.۱۱	مشتق گیری	۲۵۲
۴.۲.۱۱	انتگرال گیری	۲۵۳

۲۵۴	۵.۲.۱۱ محاسبه سری
۲۵۷	۳.۱۱ کاربرد Symbolic در حل معادلات دیفرانسیل
۲۵۸	۱.۳.۱۱ حل معادلات دیفرانسیل مرتبه اول
۲۶۰	۲.۳.۱۱ معادلات دیفرانسیل مرتبه اول با مقدار اولیه
۲۶۱	۳.۳.۱۱ حل معادلات دیفرانسیل مرتبه دوم
۲۶۲	۴.۳.۱۱ حل دستگاه معادلات دیفرانسیل
۲۶۴	۵.۳.۱۱ محاسبه تبدیل لاپلاس و تبدیل معکوس لاپلاس
۲۶۶	۴.۱۱ کاربرد جعبه‌ابزار Symbolic در رسم نمودار
۲۶۶	۱.۴.۱۱ رسم در فضای دوبعدی
۲۷۱	۲.۴.۱۱ رسم نمودار در فضای سه‌بعدی
۲۷۲	۳.۴.۱۱ رسم رویه‌های فضایی
۲۷۶	۵.۱۱ کاربرد جعبه‌ابزار Symbolic در جبرخطی
۲۷۸	۶.۱۱ تمرین

نسخه
دایگان

فهرست تصاویر

۳	محیط MATLAB	۱.۱
۳	محیط Octave	۲.۱
۵	حاصل اجرای دستور pkg list	۳.۱
۱۹	روش ایجاد m-فایل در MATLAB	۱.۲
۱۹	گزینه Add to Path را کلیک کنید.	۲.۲
۷۶	روش ایجاد m-فایل در MATLAB	۱.۵
۷۶	گزینه Add to Path را کلیک کنید.	۲.۵
۲۲۱	نمونه‌ای از یک شبکه در صفحه xy	۱.۱۰
۲۳۸	نصب جعبه‌ابزار Symbolic در هنگام نصب MATLAB	۱.۱۱

نسخه
رایگان

فهرست جداول

۸	عملگرهای محاسباتی در MATLAB	۱.۲
۱۲	متغیرهای از پیش تعریف شده در MATLAB	۲.۲
۱۶	توابع مثلثاتی در MATLAB	۳.۲
۱۸	توابع مقدماتی ریاضی در MATLAB	۴.۲
۱۹	توابع گرد کردن اعداد در MATLAB	۵.۲
۶۳	توابع پیش ساخته برای آرایه ها	۲.۴
۶۴	توابع پیش ساخته برای آرایه ها	۱.۴
۸۶	توابع منطقی در MATLAB	۱.۵
۱۱۶	کاراکترهای تعیین نوع خط	۱.۶
۱۱۶	کاراکترهای تعیین رنگ خط	۲.۶
۱۱۷	کاراکترهای تعیین شکل نقاط	۳.۶
۱۱۹	گزینه های کنترل مشخصات نمودار	۴.۶
۱۲۱	گزینه های تعیین نوع خط	۵.۶
۱۳۹	گزینه های کنترل مشخصات متن	۶.۶
۱۴۱	حروف یونانی قابل استفاده در MATLAB	۷.۶
۱۷۴	برازش با توابع غیر چندجمله ای در MATLAB	۱.۷
۱۹۶	روش های حل معادلات دیفرانسیل در MATLAB	۱.۸

۲۱۴	MATLAB	دستورات مرتبط با ماتریس‌ها در	۱.۹
۲۱۵	MATLAB	دستورات مرتبط با ماتریس‌ها در	۲.۹

۱ این فصل را حتما بخوانید!

پیش از آغاز به کار و یادگیری MATLAB، مانند هر نرم افزار دیگری، نیاز به خود نرم افزار داریم. برخلاف تصور، برای استفاده از دستورات و کدهای MATLAB، نیازی به خود نرم افزار نیست و می توان از گزینه های دیگری استفاده کرد که در ادامه به یکی از آنها خواهیم پرداخت. لذا پیش از اقدام به تهیه و نصب نرم افزار MATLAB پیشنهاد می کنم این بخش را تا انتها بخوانید، سپس نسبت به انتخاب و تهیه نرم افزار اقدام کنید.

یک پیشنهاد خوب

چون نرم افزار MATLAB رایگان نیست و برای استفاده از آن یا باید هزینه زیادی پرداخت، حدود ۱۰۰۰ دلار، یا به صورت کرک شده آن را مورد استفاده قرار داد، که نقض قانون کپی رایت است، لذا پیشنهاد می شود از یک نرم افزار جایگزین برای یادگیری MATLAB استفاده کنید. نرم افزارهای زیادی وجود دارند که مشابه MATLAB هستند و برای یادگیری کاملاً مناسب هستند، یکی از بهترین نرم افزارهای جایگزین، Octave می باشد که در ادامه به معرفی چگونگی نصب و استفاده از آن خواهیم پرداخت.

۱.۱ معرفی Octave

Octave نرم افزاری است که شبیه MATLAB بوده و بنا به ادعای شرکت سازنده آن تا بیش از ۹۵ درصد با MATLAB شباهت دارد و دستورات MATLAB در محیط این نرم افزار نیز قابل استفاده می باشند. برای استفاده از این نرم افزار بجای MATLAB دلایل زیادی وجود دارد:

- تقریباً تمام دستورات MATLAB در محیط Octave قابل اجرا هستند و هیچ نگرانی بابت سازگاری کدها و دستورات نوشته در Octave با MATLAB وجود ندارد.
- Octave به صورت قانونی رایگان است در حالیکه برای استفاده از MATLAB باید چیزی حدود ۱۰۰۰ دلار پرداخت کرد، یا از نسخه کرک شده آن استفاده نمود.
- حجم نرم افزار Octave حدود ۳۰۰ مگابایت است ولی MATLAB حدود ۱۵ گیگا بایت حجم دارد.
- نصب و سرعت اجرای Octave در مقایسه با MATLAB بسیار بیشتر است.
- در Octave تمام بسته‌ها در ابتدا نصب نمی‌شوند و کاربر بسته به نیاز خود می‌تواند بسته‌های مختلف را نصب کرده و از آنها استفاده نماید. در آغاز یادگیری MATLAB به هیچ وجه نیازی به بسیاری از بسته نیست و تنها دستورات پایه کافی می‌باشند.

اگر به نصب و استفاده از Octave تمایل پیدا کرده‌اید، ادامه این فصل را مطالعه کنید و در غیر این صورت از مطالعه ادامه این فصل صرف نظر کنید و نرم افزار MATLAB را تهیه کرده و آن را طبق راهنمای نصب که معمولاً در فایل `readme.txt` وجود دارد نصب کنید و به فصل دوم برای آغاز کار مراجعه کنید.

۲.۱ تهیه و نصب Octave

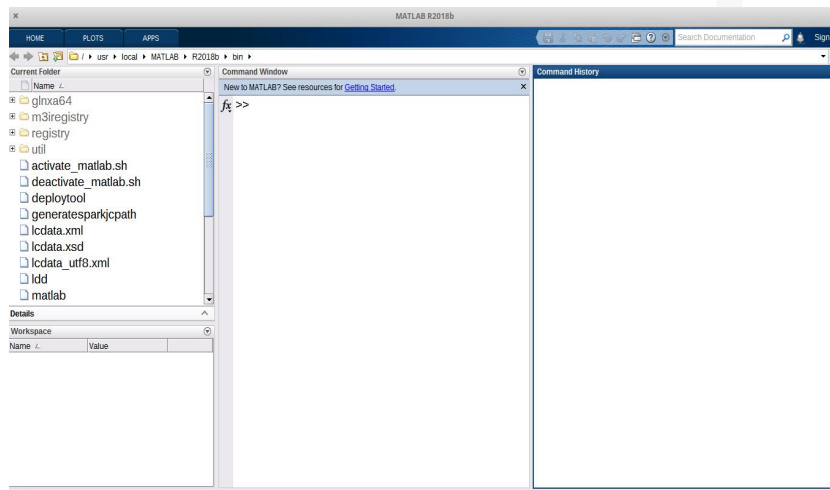
برای تهیه Octave ابتدا به وبسایت <https://www.gnu.org/software/octave/> مراجعه کنید و در صورتی که کاربر Windows هستید، آخرین نسخه موجود را دانلود کرده و نصب کنید. اگر کاربر لینوکس هستید (مثلاً سیستم عامل Ubuntu) با استفاده از دستور زیر که باید در محیط Terminal وارد شود، می‌توانید به صورت آنلاین Octave را دانلود و نصب نمایید.

```
sudo apt-get install octave
```

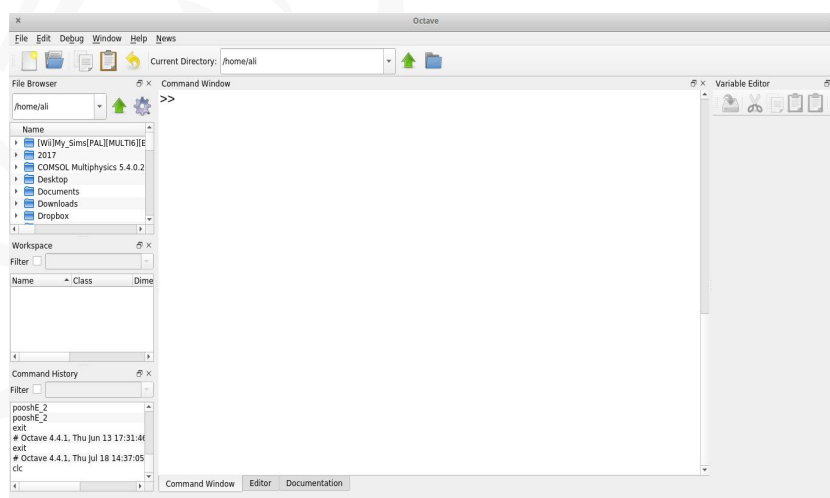
📌 توجه کنید برای نصب Octave در محیط لینوکس باید اتصال به اینترنت برقرار باشد.

۳.۱ اجرای Octave و آشنایی با محیط

پس از نصب و اجرای نرم‌افزارهای MATLAB و Octave با محیط‌های کاری به شکل ۱.۱ و ۲.۱ مواجه خواهیم شد.



شکل ۱.۱: محیط MATLAB



شکل ۲.۱: محیط Octave

با مقایسه دو شکل ۱.۱ و ۲.۱ دیده می‌شود که پنجره‌های دو محیط یکسان هستند و کارایی یکسانی نیز دارند، پس در Octave با محیط جدید روبه‌رو نخواهیم شد و از این جهت نیز نگرانی وجود ندارد.

۴.۱ نصب بسته در Octave

در هنگام نصب Octave تنها هسته اصلی نرم‌افزار نصب می‌شود و تنها بسته‌ای که نصب می‌شود بسته io می‌باشد. برای آشنایی با بسته‌های^۱ موجود برای نرم‌افزار Octave به وبسایت آن مراجعه کنید. در وبسایت می‌توانید اسامی بسته‌ها را به همراه کاربرد هر بسته و راهنمای بسته مشاهده کنید.

نکته عملی

برای آگاهی از بسته‌های نصب شده روی Octave ابتدا نرم‌افزار را اجرا کنید و در سطر فرمان دستور زیر را وارد کنید:

```
>> pkg list
```

اگر هنوز بسته‌ای نصب نکرده‌اید با اجرای این دستور تنها بسته io نمایش داده خواهد شد. برای نصب بسته جدید باید ابتدا به اینترنت متصل شوید، سپس دستور زیر را وارد کنید:

```
>> pkg install -forge PackageName
```

بدیهی است که بجای PackageName باید نام بسته مورد نظر خود را بنویسید. برای استفاده از یک بسته جدید باید آن را فراخوانی کرد، فراخوانی یک بسته با دستور زیر انجام می‌شود:

```
>> pkg load PackageName
```

که بجای PackageName باید نام بسته نصب شده را نوشت.

❗ اگر بسته‌ای را فراخوانی نکنید، قادر به استفاده از دستورات آن بسته نخواهید بود و در صورت استفاده با خطا مواجه خواهید شد.

❗ در هر بار اجرای نرم‌افزار Octave باید بسته‌های مورد نیاز فراخوانی شوند ولی پس از یک بار فراخوانی تا زمانی که از محیط Octave خارج نشده باشید، دیگر نیاز به فراخوانی بسته نیست.

^۱package

پس از فراخوانی یک بسته اگر دستور `pkg list` را اجرا کنید، در کنار بسته‌های فراخوانی شده علامت * قرار می‌گیرد ولی بسته‌های نصب شده و فراخوانی نشده علامت ستاره را نخواهند داشت. شکل ۳.۱ را ببینید.

```
>> pkg list
Package Name | Version | Installation directory
-----+-----+-----
      cgi    | 0.1.2   | /home/ali/octave/cgi-0.1.2
    control  | 3.1.0   | /home/ali/octave/control-3.1.0
    geometry | 3.0.0   | /home/ali/octave/geometry-3.0.0
       io *  | 2.4.11  | /home/ali/octave/io-2.4.11
    symbolic *| 2.7.0   | /home/ali/octave/symbolic-2.7.0
>> |
```

شکل ۳.۱: حاصل اجرای دستور `pkg list`

نکته عملی

برای خارج کردن بسته از حالت فراخوانی از دستور زیر استفاده کنید:

```
>> pkg unload PackageName
```

و برای حذف کامل یک بسته نصب شده از دستور

```
>> pkg uninstall PackageName
```

استفاده کنید. بدیهی است که در هر دو دستور بجای `PackageName` باید نام بسته مورد نظر نوشته شود.

امیدوارم با مطالعه این فصل علاقه‌مند به نصب Octave شده باشید.

نسخه
دیپگان

۲ آغاز کار

در این فصل به بیان برخی مفاهیم پایه‌ای MATLAB می‌پردازیم. مطالبی که در این فصل بیان خواهند شد از اهمیت زیادی برخوردارند و باید به خوبی فرا گرفته شوند، زیرا تا انتهای کتاب با این مفاهیم سروکار خواهیم داشت و همواره آنها را بکار خواهیم برد. پس بهتر است که برای یادگیری مطالب این فصل وقت کافی صرف کنید و دستوراتی که بیان خواهد شد را در محیط MATLAB امتحان کنید.

۱.۲ عملگرهای محاسباتی

ساده‌ترین کاربرد MATLAB، استفاده از آن به عنوان یک ماشین حساب است که برای این کار، نیاز به استفاده از عملگرهای محاسباتی می‌باشد. در MATLAB انواع گوناگونی از عملگرها وجود دارد، ولی تمام آنها در این بخش معرفی نخواهند شد و این کار به مرور انجام خواهد گرفت.

عملگرهای محاسباتی

عملگرهای محاسباتی، ساده‌ترین و پرکاربردترین نوع عملگرها در MATLAB می‌باشند. این دسته از عملگرها روی اعداد، صحیح و اعشاری، عمل می‌کنند و حاصل آنها نیز یک عدد می‌باشد. عملگرهای محاسباتی بر روی ماتریس‌ها نیز قابل استفاده هستند، ولی روش استفاده از آنها اندکی تفاوت دارد که در بخش مربوط به ماتریس‌ها بیان خواهد شد.

در MATLAB شش عملگر محاسباتی وجود دارد که در جدول ۱.۲ آورده شده‌اند.

جدول ۱.۲: عملگرهای محاسباتی در MATLAB

عملگر	عمل	مثال
+	جمع	>> 5 + 2 ans = 7
-	تفریق	>> 5 - 2 ans = 3
*	ضرب	>> 5 * 2 ans = 10
/	تقسیم	>> 5 / 2 ans = 2.5000
\	تقسیم چپ	>> 5 \ 2 ans = 0.40000
^	توان	>> 5^2 ans = 25

📌 در MATLAB همواره نتیجه آخرین محاسبات انجام شده در متغیر `ans` ذخیره می‌شود.

📌 تقدم عملیاتی عملگرهای محاسباتی به صورت: ۱. توان ۲. ضرب، تقسیم و تقسیم چپ ۳. جمع و تفریق می‌باشد. بدیهی است که در صورت وجود پرانتز در عبارت محاسباتی، حق تقدم با داخلی‌ترین پرانتز است.

📌 در تقسیم چپ، عدد سمت راست عملگر بر عدد سمت چپ تقسیم می‌شود.

📌 در هر شش عملگر محاسباتی، اگر یکی از اعداد اعشاری باشد، حاصل نیز اعشاری خواهد بود.

۲.۲ تعریف و استفاده از متغیرها

در MATLAB می‌توان متغیرهایی تعریف کرد و در آنها مقادیری ذخیره نمود. نامگذاری متغیرها کاملاً اختیاری است ولی در نامگذاری باید قواعدی را رعایت کرد.

قواعد نامگذاری متغیرها

در هنگام نامگذاری متغیرها موارد زیر را حتما رعایت کنید:

۱. نام متغیر حتما باید یکی از حروف الفبای انگلیسی آغاز شود، پس یک عدد یا یکی از کاراکترهای خاص، نمی‌توانند آغازگر نام یک متغیر باشند.
۲. در نامگذاری متغیرها، فقط استفاده از حروف الفبای انگلیسی، ارقام و خط زیر مجاز می‌باشد. پس استفاده از فضای خالی، و کاراکترهایی مانند + - ! @ # \$ % ^ & در نامگذاری متغیرها مجاز نیستند.
۳. در MATLAB تعدادی واژه کلیدی وجود دارد، این واژه‌ها عبارتند از
break, case, catch, classdef, continue, else, elseif, end,
for, function, global, if, otherwise, parfor, persistent,
return, spmd, switch, try, while
استفاده از این واژه‌ها به عنوان نام متغیر مجاز نیست.
۴. طول متغیرها می‌تواند بین یک تا ۶۳ کاراکتر باشد.
۵. MATLAB نسبت به حروف کوچک و بزرگ حساس است، برای مثال n و N دو متغیر متفاوت می‌باشند.

👉 با استفاده از تابع `iskeyword(txt)` می‌توان به کلیدی بودن واژه `txt` پی برد. بدیهی است که بجای `txt` واژه مورد نظر نوشته می‌شود و این واژه باید در میان یک جفت کوتیشن قرار بگیرد. حاصل این دستور 0، در صورت کلیدی نبودن واژه، یا 1، در صورت کلیدی بودن واژه می‌باشد. برای مثال دو دستور زیر را ببینید که برای واژه کلیدی `if` و واژه غیرکلیدی `Name` مورد استفاده قرار گرفته است.

```
>> iskeyword('if')
ans = 1
>> iskeyword('Name')
ans = 0
```

👉 تابع `isvarname(txt)` مشابه دستور `iskeyword(txt)` است با این تفاوت که مقدار 1 زمانی برگشت داده می‌شود که `txt` بتواند نام یک متغیر باشد.

```
>> isvarname('str')
ans = 1
>> isvarname('elseif')
ans = 0
```

مثال ۱.۲. اسامی زیر می‌توانند به‌عنوان نام یک متغیر به کار برده شوند،

Name, Student_Number, stu_avg, i, I, Field_Of_Study, If

ولی اسامی زیر به‌عنوان نام متغیر غیر مجاز هستند،

2Ali, if, Student Number, A!21, B^A

👉 پیشنهاد می‌شود در نامگذاری متغیرها از اسامی با معنی استفاده کنید تا در هنگام برنامه‌نویسی دچار سردرگمی نشوید.

👉 در مثال ۱.۲ `If` می‌تواند نام متغیر باشد، زیرا کاراکتر اول با حروف بزرگ نوشته شده است.

دستورات گمارشی

برای استفاده از یک متغیر می‌توان از دستورات گمارشی به‌شکل کلی زیر استفاده کرد،

```
Var_Name = Value
```

که در آن `Var_Name` نام متغیر و `Value` مقداری است که می‌خواهیم در متغیر ذخیره شود. این مقدار می‌تواند انواع گوناگونی مانند مقادیر عددی، رشته‌ای، ماتریسی و منطقی باشد که در ادامه به آنها خواهیم پرداخت.

مثال ۲.۲. به دستورات زیر و چگونگی مقداردهی به متغیرها توجه کنید.

```
>> Name = 'Ali', Surname = 'Mesforush', Age = 48
Name = Ali
```

```
Surname = Mesforush
Age = 48
>> Field_Of_Study = 'Math';
>>
```

👉 به قرار گرفتن اسامی در میان یک جفت کوتشین توجه کنید.

👉 در مثال ۲.۲ به دلیل استفاده از کاما (,) امکان قرار دادن چند دستور گمارشی در یک سطر وجود دارد. اگر از کاما استفاده نکنید با خطا مواجه خواهید شد.

👉 در حالت کلی بعد از اجرای هر دستور نتیجه در سطر بعد نمایش داده می شود. اگر بخواهید پس از اجرا، نتیجه‌ای نمایش داده نشود، کافیت در انتهای دستور یک سمی کالن (;) قرار دهید.

👉 برخی از عبارات محاسباتی طولانی هستند و در یک سطر نمی توان آنها را نوشت، برای شکستن یک عبارت محاسباتی به دو سطر یا بیشتر، کافیت در انتهای سطر اول سه نقطه (...) قرار دهیم، سپس در سطر بعد به نوشتن ادامه فرمول پردازیم.

در مثال ۳.۲ به چگونگی استفاده از کاما، سمی کالن و سه نقطه اشاره شده است.

مثال ۳.۲. دستورات زیر محیط و مساحت یک مستطیل را محاسبه می کنند.

```
>> length = 10; width = 5;
>> Area = length*width
Area = 50
>> perimeter = 2*...
(length+width)
perimeter = 30
```

نکته عملی

با دستور `clear` می توان همه، یا برخی از متغیرهای در حال استفاده را از حافظه پاک کرد.

`clear all` باعث پاک شدن تمام متغیرهای تعریف شده می گردد.

`clear var` فقط متغیر مشخص شده، که با `var` مشخص شده است پاک خواهد شد.

❏ اگر متغیری با دستور `clear` پاک شود، دیگر در دسترس نیست و امکان استفاده از آن وجود ندارد.

در MATLAB تعدادی متغیر از پیش تعریف شده وجود دارد که مقادیر خاصی را در خود نگهداری می‌کنند، برخی از این متغیرها در جدول ۲.۲ آورده شده‌اند.

❏ این متغیرها کلیدی نیستند و می‌توان مقدار آنها را تغییر داد، ولی این کار خیلی توصیه نمی‌شود، چراکه در بسیار از مواقع به این مقادیر نیاز داریم.

جدول ۲.۲: متغیرهای از پیش تعریف شده در MATLAB

متغیر	مقدار
eps	کوچکترین مقدار تفاضل میان دو عدد
inf	بی‌نهایت
pi	مقدار عدد π
nan	NaN به معنای مقدار عددی نیست
i	$\sqrt{-1}$
j	$\sqrt{-1}$
realmax	بزرگترین مقدار عددی حقیقی که $1.7977e+308$ می‌باشد.
realmin	کوچکترین مقدار عددی حقیقی که $2.2251e-308$ می‌باشد.
intmax	بزرگترین مقدار عددی صحیح که 2147483647 می‌باشد.
intmin	کوچکترین مقدار عددی صحیح که -2147483648 می‌باشد.

سه دستور مهم

در MATLAB دستوراتی وجود دارد که با آنها می‌توان اطلاعاتی از متغیرها بدست آورد.

who اسامی تمام متغیرهای تعریف شده و مورد استفاده در محیط را نمایش می‌دهد.

whos باعث نمایش اطلاعاتی پیرامون متغیر می‌شود.

clc تمام محتویات پنجره فرمان را پاک می‌کند.

📌 دستور whos به صورت whos var_name بکار برده می‌شود.

مثال ۴.۲. به نمونه اجرای دستورات who و whos در زیر توجه کنید.

```
>> Name = 'Ali';
>> Age = 48;
>> height = 1.75;
>> who
Your variables are:
Age      Name      height
>> whos Age
Name      Size      Bytes  Class      Attributes
Age        1x1          8  double
>> whos Name
Name      Size      Bytes  Class      Attributes
Name        1x3          6   char
```

نکته عملی

در MATLAB نمایش اعداد در حالت پیش فرض با پنج رقم انجام می‌شود. با استفاده از دستورات زیر می‌توان تعداد ارقام و شکل نمایش اعداد را در MATLAB تغییر داد.

format short باعث نمایش اعداد به شکل پیش فرض، یعنی با پنج رقم می‌شود.

format long باعث نمایش اعداد با ۱۵ رقم می‌شود.

format hex باعث نمایش اعداد در مبنای ۱۶ می‌شود.

📌 علاوه بر سه دستور بیان شده، دستورات دیگری نیز برای فرمت وجود دارد که با گذاشتن e و g و eng پس از short حاصل می‌شوند. برای اطلاعات بیشتر دستور زیر را اجرا کنید.

```
>> help format
```

۳.۲ توابع کتابخانه‌ای ریاضی در MATLAB

در MATLAB توابع کتابخانه‌ای بسیاری وجود دارد که از پیش نوشته هستند و می‌توان از آنها برای انجام محاسبات ریاضی استفاده کرد. در این بخش به معرفی برخی از توابع ریاضی پرکاربرد می‌پردازیم. استفاده از این توابع کار دشواری نیست و معمولاً به همان شکلی که در ریاضی معرفی می‌شوند، در MATLAB نیز مورد استفاده قرار می‌گیرند.

۱.۳.۲ اعداد مختلط

در جدول ۲.۲ دیدید که از i و j به عنوان $\sqrt{-1}$ استفاده می‌شود. پس انتظار بر این است که با استفاده از آنها بتوان اعداد مختلط را تولید نمود.

📌 برای معرفی یک عدد مختلط می‌توان از یکی از دو شکل $a+ib$ یا $a+jb$ استفاده کرد.

مثال ۵.۲. دستورات زیر را در محیط MATLAB خود اجرا کنید:

```
>> z1 = 2 + 3i , z2 = 3 - 2j
```

```
z1 = 2 + 3i
```

```
z2 = 3 - 2i
```

📌 اگرچه در تعریف متغیر $z2$ از j استفاده شده است ولی در خروجی، i دیده می‌شود.

📌 عملگرهای محاسباتی را به همان صورتی که در بخش ۱.۱ معرفی شدند، می‌توان برای اعداد مختلط نیز بکار برد.

مثال ۶.۲. به عملیات محاسباتی زیر و چگونگی استفاده از عملگرهای محاسباتی توجه کنید،

```
>> z1 = 2 + 3i; z2 = 3 - i;
```

```
>> z3 = z1 + z2, z4 = z1 - z2, z5 = z1 * z2
```

```
z3 = 5 + 2i
```

```
z4 = -1 + 4i
```

```
z5 = 9 + 7i
```

```
>> z6 = z1/z2, z7 = z1\z2, z8 = z1^z2
```

```
z6 = 0.30000 + 1.10000i
```

```
z7 = 0.23077 - 0.84615i
```

```
z8 = -11.893 + 124.672i
```

شش دستور مرتبط با اعداد مختلط

در هنگام کار با اعداد مختلط می‌توان از شش دستور زیر استفاده کرد. فرض کنید $z = a + ib$

real که به صورت `real(z)` بکار می‌رود و بخش حقیقی عدد مختلط z را برمی‌گرداند.

imag که به صورت `imag(z)` بکار می‌رود و بخش موهومی عدد مختلط z را برمی‌گرداند.

abs که به صورت `abs(z)` بکار می‌رود و اندازه عدد مختلط z را برمی‌گرداند.

angle که به صورت `angle(z)` بکار می‌رود و $\tan^{-1}(\frac{b}{a})$ را برمی‌گرداند.

complex که به صورت `complex(a, b)` بکار می‌رود و عدد مختلط $a + ib$ را برمی‌گرداند.

conj که به صورت `conj(z)` بکار می‌رود و مزدوج عدد مختلط z را برمی‌گرداند.

مثال ۷.۲. به چگونگی استفاده از شش دستور بیان شده در عبارات زیر دقت کنید.

```
>> rp=real(z), ip=imag(z), l=abs(z), d=angle(z), w=complex(2,4)
rp = 3
ip = 4
l = 5
d = 0.92730
w = 2 + 4i
```

۲.۳.۲ توابع مقدماتی ریاضی

در MATLAB توابع ریاضی به صورت کتابخانه‌ای تعریف شده‌اند. توابع مثلثاتی، معکوس مثلثاتی، توابع هایپر بولیک و توابع معکوس هایپر بولیک از این دسته توابع هستند. لیست کامل توابع مثلثاتی در جدول ۳.۲ آورده شده است.

❏ ورودی توابع مثلثاتی معمولی در حالت معمولی برحسب رادیان می‌باشد. اگر بخواهیم ورودی

جدول ۳.۲: توابع مثلثاتی در MATLAB

ریاضی	MATLAB	ریاضی	MATLAB	ریاضی	MATLAB
$\tan x$	<code>tan(x)</code>	$\cos x$	<code>cos(x)</code>	$\sin x$	<code>sin(x)</code>
$\csc x$	<code>csc(x)</code>	$\sec x$	<code>sec(x)</code>	$\cot x$	<code>cot(x)</code>
$\tan^{-1} x$	<code>atan(x)</code>	$\cos^{-1} x$	<code>acos(x)</code>	$\sin^{-1} x$	<code>asin(x)</code>
$\csc^{-1} x$	<code>acsc(x)</code>	$\sec^{-1} x$	<code>asec(x)</code>	$\cot^{-1} x$	<code>acot(x)</code>
$\tan x$	<code>tand(x)</code>	$\cos x$	<code>cosd(x)</code>	$\sin x$	<code>sind(x)</code>
$\csc x$	<code>cscd(x)</code>	$\sec x$	<code>secd(x)</code>	$\cot x$	<code>cotd(x)</code>
$\tan^{-1} x$	<code>atand(x)</code>	$\cos^{-1} x$	<code>acosd(x)</code>	$\sin^{-1} x$	<code>asind(x)</code>
$\csc^{-1} x$	<code>acscd(x)</code>	$\sec^{-1} x$	<code>asecd(x)</code>	$\cot^{-1} x$	<code>acotd(x)</code>
$\tanh x$	<code>tanh(x)</code>	$\cosh x$	<code>cosh(x)</code>	$\sinh x$	<code>sinh(x)</code>
$\operatorname{csch} x$	<code>csch(x)</code>	$\operatorname{sech} x$	<code>sech(x)</code>	$\coth x$	<code>coth(x)</code>
$\tanh^{-1} x$	<code>atanh(x)</code>	$\cosh^{-1} x$	<code>acosh(x)</code>	$\sinh^{-1} x$	<code>asinh(x)</code>
$\operatorname{csch}^{-1} x$	<code>acsch(x)</code>	$\operatorname{sech}^{-1} x$	<code>asech(x)</code>	$\coth^{-1} x$	<code>acoth(x)</code>

برحسب درجه باشد می‌توانیم از کاراکتر `d` در انتهای دستورات استفاده کنیم. (ستون‌های پنجم تا هشتم جدول را ببینید.)

مثال ۸.۲. به چگونگی استفاده از توابع مثلثاتی در دستورات زیر توجه کنید.

```
>> sin(pi/6),sind(30),asin(0.5),asind(0.5),sinh(1),asinh(10)
ans = 0.50000
ans = 0.50000
ans = 0.52360
ans = 30.000
ans = 1.1752
ans = 2.9982
>> E = (2*sind(30) + tand(60)^2 - 1)/(secd(30) + cscd(60))
E = 1.2990
```


دسته دیگری از توابع ریاضی کتابخانه‌ای ۴.۲ آورده شده‌اند. این توابع را می‌توان در عبارات محاسباتی و در برنامه‌هایی که در آینده خواهیم نوشت بکار برد.

مثال ۹.۲. عبارت ریاضی زیر را در محیط MATLAB برای $x = 10$ بنویسید.

$$f = \sqrt[3]{\log(x^2 + e^x)} + e^{\sin \ln x}$$

```
>> x = 10;
>> f = nthroot(log10(x^2+exp(x)),3) + exp(sin(log(x)))
f = 3.7361
```

گرد کردن اعداد در MATLAB به تمام روش‌های متداول در ریاضی انجام می‌شود. لیست کامل توابع مرتبط با گرد کردن اعداد در جدول ۵.۲ آورده شده است.

اگر هر یک از دستورات گرد کردن، مثلاً دستور fix، روی عدد مختلط $z = a + ib$ اجرا شود مقداری که برگشت داده می‌شود به صورت $\text{fix}(\text{real}(z)) + i \text{fix}(\text{imag}(z))$ می‌باشد.

مثال ۱۰.۲. به چگونگی استفاده از دستورا گرد کردن اعداد برای اعداد مختلط در زیر توجه کنید.

```
>> z = 2.265 + 3.75i
z = 2.2650 + 3.7500i
>> round(z)
ans = 2 + 4i
```

۴.۲ -m فایل

در MATLAB (و همچنین در Octave) می‌توان تعدادی دستورات را پشت سر هم نوشت و در یک فایل با نام دلخواه که تابع قواعد نامگذاری متغیرها می‌باشد، ذخیره کرد. به این گونه فایل‌ها -m فایل گفته می‌شود. با این روش می‌توان مجموعه دستورات را به صورت یکجا اجرا کرد. برای کار با -m فایل‌ها باید ابتدا یک -m فایل ایجاد کرد و سپس دستورات مورد نظر را در آن نوشت. برای تولید یک -m فایل به روشی که در شکل ۱.۲ نشان داده شده است عمل کنید.

جدول ۴.۲: توابع مقدماتی ریاضی در MATLAB

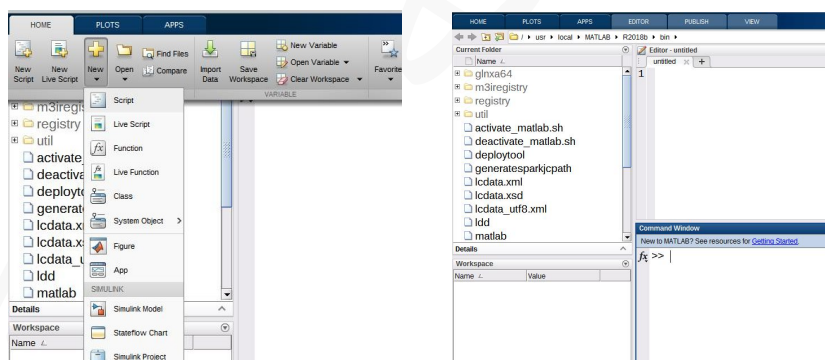
تابع	عمل	مثال
<code>exp(x)</code>	تابع نمایی e^x	<pre>>> exp(1) ans = 2.7183</pre>
<code>sqrt(x)</code>	\sqrt{x}	<pre>>> sqrt(8) ans = 2.8284</pre>
<code>nthroot(x,n)</code>	$\sqrt[n]{x}$	<pre>>> nthroot(8,3) ans = 2</pre>
<code>factorial(n)</code>	$n!$	<pre>>> factorial(5) ans = 120</pre>
<code>sign(x)</code>	تابع علامت	<pre>>> sign(5) ans = 1</pre>
<code>abs(n)</code>	تابع قدرمطلق	<pre>>> abs(-5) ans = 5</pre>
<code>log(x)</code>	$\ln x$	<pre>>> log(100) ans = 4.6052</pre>
<code>log10(x)</code>	$\log x$	<pre>>> log10(100) ans = 2</pre>
<code>gcd(x,y)</code>	بزرگترین مقسوم علیه مشترک	<pre>>> gcd(15,6) ans = 3</pre>
<code>lcm(x,y)</code>	کوچکترین مضرب مشترک	<pre>>> lcm(15,6) ans = 30</pre>
<code>rem(x,y)</code>	محاسبه باقیمانده	<pre>>> rem(15,6) ans = 3</pre>

👉 پس از نوشتن دستورات مورد در نظر در فایل، `m`-فایل ایجاد شده را باید ذخیره کنید. پسوند پیشنهادی `m` می باشد که باید پذیرفته شود و فایل با نام دلخواه که تابع قواعد نامگذاری متغیرها می باشد در محلی ذخیره شود.

👉 با زدن کلید `F5` می توان `m`-فایل را اجرا کرد، در این صورت و در اولین اجرا با شکل ۲.۲ برخورد خواهید کرد که باید روی `Add to Path` کلیک کنید.

جدول ۵.۲: توابع گرد کردن اعداد در MATLAB

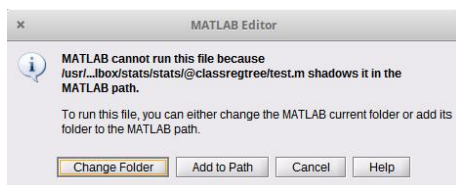
مثال	عمل	تابع
<pre>>> round(2.785) ans = 3</pre>	گرد کردن به نزدیکترین عدد صحیح	<code>round(x)</code>
<pre>>> fix(2.785) ans = 2</pre>	گرد کردن با برش	<code>fix(x)</code>
<pre>>> ceil(2.385) ans = 3</pre>	گرد کردن به نزدیکترین عدد صحیح بیشتر از x	<code>ceil(x)</code>
<pre>>> floor(2.85) ans = 2</pre>	گرد کردن به نزدیکترین عدد صحیح کوچکتر از x	<code>floor(x)</code>



(ب) script یا New Script را انتخاب کنید

(آ) m-فایل ایجاد شده

شکل ۱.۲: روش ایجاد m-فایل در MATLAB



شکل ۲.۲: گزینه Add to Path را کلیک کنید.

توجه

با استفاده از `m`-فایل‌ها می‌توان در MATLAB برنامه‌نویسی کرد، ولی در این فصل این موضوع آموزش داده نخواهد شد و در فصل ۵ به آموزش برنامه‌نویسی به کمک MATLAB خواهیم پرداخت.

در پایان این فصل به بیان چند مثال در رابطه استفاده از `m`-فایل‌ها برای اجرای محاسبات در MATLAB می‌پردازیم. پیشنهاد می‌شود تمامی کدهای نوشته شده در مثال‌ها را در محیط نرم‌افزار خود وارد کرده و اجرا کنید.

مثال ۱۱.۲. معادله درجه دوم $ax^2 + bx + c = 0$ را با نوشتن دستورات زیر در یک `m`-فایل حل می‌کنیم.

```
a = 2; b = 3; c = 5;
delta = b^2 - 4*a*c;
x_1 = (-b + sqrt(delta))/(2*a)
x_2 = (-b - sqrt(delta))/(2*a)
```

با اجرای `m`-فایل حاوی دستورات بالا، خروجی به صورت زیر نمایش داده خواهد شد.

```
x_1 = -0.7500 + 1.3919i
x_2 = -0.7500 - 1.3919i
```

مثال ۱۲.۲. درستی اتحاد مثلثاتی $\cos^2 \frac{x}{2} = \frac{\tan x + \sin x}{2 \tan x}$ را برای $x = \frac{\pi}{5}$ بررسی کنید. کد زیر را در یک `m`-فایل بنویسید:

```
x = pi/5;
LHS = cos(x/2)^2
RHS = (tan(x) + sin(x))/(2*tan(x))
```

نتیجه اجرا به شکل زیر خواهد بود.

LHS = 0.9045

RHS = 0.9045

مثال ۱۳.۲. با استفاده از فرمول انتگرال گیری دو نقطه‌ای گاوس که به شکل

$$\int_{-1}^1 f(x) dx = f\left(-\frac{\sqrt{3}}{3}\right) + f\left(\frac{\sqrt{3}}{3}\right),$$

است، مطلوبست محاسبه

$$\int_0^2 \frac{\sqrt[3]{x^2 + \tan x}}{\ln(x^2 + e^x)} dx$$

برای حل باید تغییر متغیری به شکل $x = \frac{1}{3}(b-a)t + \frac{1}{3}(a+b)$ بدهیم که در این مساله به صورت $x = t + 1$ می باشد. پس عملاً باید انتگرالی به شکل زیر را محاسبه کنیم:

$$\int_{-1}^1 f(t) dt, \quad f(t) = \frac{\sqrt[3]{(t+1)^2 + \tan(t+1)}}{\ln((t+1)^2 + e^{t+1})}.$$

کد زیر این محاسبات را انجام می دهد.

```
x = sqrt(3)/3;
I = nthroot((x + 1)^2 + tan(x+1),3)/log((x+1)^2 + exp(x+1))...
+ nthroot((-x + 1)^2 + tan(-x+1),3)/log((-x+1)^2 + exp(-x+1))
```

با اجرای این کد، $I = -1.0619$ حاصل می شود.

📌 به کاربرد سه نقطه در این مثال برای نوشتن عبارت محاسباتی در دو سطر توجه کنید.

نکته عملی

در MATLAB هرگاه نیاز به اطلاعاتی پیرامون دستوری پیدا کردید کفایت در سطر فرمان دستور

`help Command-Name`

را اجرا کنید که در آن `Command-Name` دستوری است که نیاز به اطلاعاتی پیرامون آن دارید، در

این صورت تمام مطالب مرتبط با دستور نمایش داده می شود.

۵.۲ تمرین

تمام تمرین‌هایی که در ادامه آورده شده است را می‌توانید به‌طور مستقیم در پنجره فرمان حل کنید، یا با استفاده از یک m -فایل نسبت به حل آنها اقدام کنید ولی پیشنهاد می‌شود از m -فایل به این منظور استفاده نمایید.

تمرین ۱.۲. عبارات زیر را با استفاده از MATLAB حساب کنید.

$$\begin{array}{ll}
 ۱. \quad \frac{(۱۴/۸^۲ + ۶/۵^۲)}{۳/۸^۲} + \frac{۵۵}{\sqrt{۲} + ۱۴} & ۵. \quad \frac{۲/۵^۳ (۱۶ - \frac{۲۱۶}{۲۲})}{۱/۷^۴ + ۱۴} + \sqrt[۴]{۲ \cdot ۵^\circ} \\
 ۲. \quad (-۳/۵)^۳ + \frac{e^۶}{\ln ۵۲۴} + ۲ \cdot ۶^{۱/۳} & ۶. \quad \frac{۲/۳^۳ \cdot ۱/۷}{\sqrt{(1 - \circ/۸^۲)^۲ + (۲ - \sqrt{\circ/۸۷})^۲}} \\
 ۳. \quad \frac{۵/۲^۳ - ۶/۴^۲ + ۳}{۱/۶^۸ - ۲} + \left(\frac{۱۳/۳}{۵}\right)^{۱/۵} & ۷. \quad \frac{\sin\left(\frac{\sqrt{9}\pi}{9}\right)}{\cos^۲\left(\frac{۵}{\sqrt{9}}\pi\right)} + \frac{۱}{۷} \tan\left(\frac{۵}{۱۲}\pi\right) \\
 ۴. \quad ۱۵ \left(\frac{\sqrt{۱^\circ} + ۳/۷^۲}{\log_{۱^\circ}(۱۳۶۵) + ۱/۹}\right) & ۸. \quad \frac{\tan ۶۴^\circ}{\cos^۲ ۱۴^\circ} - \frac{۳ \sin ۸^\circ}{\sqrt{\circ/۹}} + \frac{\cos ۵۵^\circ}{\sin ۱۱^\circ}
 \end{array}$$

تمرین ۲.۲. با فرض $x = ۲/۳۴$ عبارات زیر را حساب کنید.

$$۱. \quad ۲x^۴ - ۶x^۳ + ۱۴/۸x^۲ + ۹/۱ \quad ۲. \quad \frac{e^{۲x}}{\sqrt{۱۴ + x^۲} - x}$$

تمرین ۳.۲. با فرض $t = ۶/۸$ عبارات زیر را حساب کنید.

$$۱. \quad \ln(|t^۲ - t^۳|) \quad ۲. \quad \frac{۷۵}{۲t} \cos(\circ/۸t - ۳)$$

تمرین ۴.۲. با فرض $x = ۸/۳$ و $y = ۲/۴$ عبارات زیر را حساب کنید.

$$۱. \quad x^۲ + y^۲ - \frac{x^۲}{y^۲} \quad ۲. \quad \sqrt{xy} - \sqrt{x+y} + \left(\frac{x-y}{x-۲y}\right)^۲$$

تمرین ۵.۲. مقادیر a, b, c و d را به‌شکل زیر تعریف کنید،

$$a = ۱۳, \quad b = ۴/۲, \quad c = \frac{۴b}{a}, \quad d = \frac{abc}{a+b+c},$$

سپس عبارات زیر را حساب کنید.

$$1. \quad a \frac{b}{c+d} + \frac{d}{c} \frac{a}{b} - (a - b^2)(c + d) \quad 2. \quad \frac{\sqrt{a^2 + b^2}}{(d - c)} + \ln(|b - a + c - d|)$$

تمرین ۶.۲. محیط یک بیضی با قطر بزرگ $2b$ و قطر کوچک $2a$ به صورت $P = 2\pi \sqrt{\frac{1}{4}(a^2 + b^2)}$ محاسبه می شود.

۱. محیط یک بیضی با $a = 9$ و $b = 3$ را محاسبه کنید.

۲. اگر محیط یک بیضی $P = 20$ باشد و $b = 2a$ ، مقادیر a و b را محاسبه کنید.

تمرین ۷.۲. دو اتحاد مثلثاتی زیر را در نظر بگیرید:

$$\cos 2x = \frac{1 - \tan^2 x}{1 + \tan^2 x}, \quad \sin 4x = 4 \sin x \cos x - 8 \sin^3 x \cos x$$

درستی این دو اتحاد را برای $x = \frac{\pi}{12}$ و $x = 38^\circ$ بررسی کنید.

تمرین ۸.۲. با تعریف دو متغیر $\alpha = \frac{\pi}{8}$ و $\beta = \frac{\pi}{11}$ درستی اتحاد مثلثاتی زیر را بررسی کنید.

$$\sin \alpha \cos \beta = \frac{1}{4} [\sin(\alpha - \beta) + \sin(\alpha + \beta)]$$

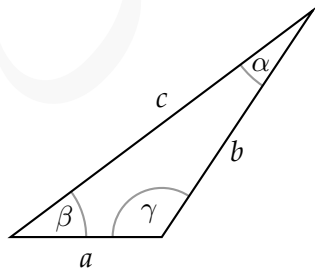
تمرین ۹.۲. می دانیم، $\int \cos^2(ax) dx = \frac{1}{4}x - \frac{\sin 2ax}{4a}$. با استفاده از این رابطه مطلوب است محاسبه انتگرال، $\int_{\frac{\pi}{9}}^{\frac{2\pi}{9}} \cos^2(x/5) dx$

تمرین ۱۰.۲.

در مثلث مقابل $a = 9$ ، $b = 18$ و $c = 25$ می باشد. مقادیر a ، b ، c را در محیط MATLAB تعریف کنید و

۱. با استفاده از قانون کسینوس ها مقدار α را محاسبه کنید. قانون کسینوس ها به شکل زیر می باشد:

$$c^2 = a^2 + b^2 - 2ab \cos \gamma.$$

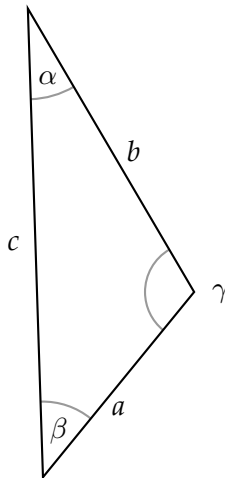


۲. با استفاده از قانون سینوس‌ها مقادیر α و β را بر حسب درجه بدست آورید. قانون سینوس‌ها به صورت زیر می‌باشد.

$$\frac{a}{\sin A} = \frac{b}{\sin B} = \frac{c}{\sin C}.$$

تمرین ۱۱.۲.

در مثلث مقابل $a = 5$, $b = 7$ و $\gamma = 25^\circ$ می‌باشد. مقادیر a , b و γ را در محیط MATLAB تعریف کنید، سپس با کمک MATLAB به سوالات زیر پاسخ دهید.



۱. با استفاده از قانون کسینوس‌ها مقدار c را محاسبه کنید.

۲. زوایای α و β را بر حسب درجه و با استفاده از قانون سینوس‌ها بدست آورید.

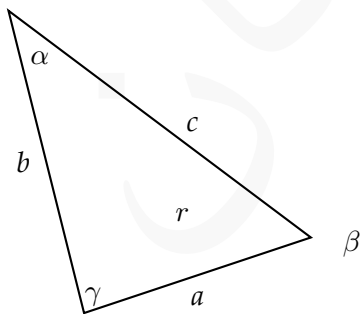
۳. با استفاده از نتایج حاصل از قسمت ۲، درستی قانون

$$\frac{a-b}{a+b} = \frac{\tan\left[\frac{1}{2}(\alpha-\beta)\right]}{\tan\left[\frac{1}{2}(\alpha+\beta)\right]}$$

می‌باشد بررسی کنید.

تمرین ۱۲.۲.

در مثلث مقابل $a = 200$, $b = 250$ و $c = 300$ میلی‌متر می‌باشد. مقادیر a , b و c را در محیط MATLAB تعریف کنید، سپس با کمک MATLAB به سوالات زیر پاسخ دهید.



۱. با استفاده از قانون کسینوس‌ها زاویه γ را محاسبه کنید.

۲. شعاع دایره محاطی را با استفاده از رابطه

$$r = \frac{1}{2}(a + b - c) \tan\left(\frac{1}{2}\gamma\right),$$

بدست آورید.

۳. شعاع دایره محاطی را با استفاده از فرمول‌های زیر بدست آورید.

$$r = \frac{\sqrt{s(s-a)(s-b)(s-c)}}{s}, \quad s = \frac{1}{2}(a+b+c)$$

تمرین ۱۳.۲. فاصله نقطه (x_0, y_0, z_0) تا صفحه $Ax + BY + CZ + D = 0$ از فرمول زیر بدست می‌آید،

$$d = \frac{|Ax_0 + By_0 + Cz_0 + D|}{\sqrt{A^2 + B^2 + C^2}}.$$

با استفاده از این فرمول، فاصله نقطه $(8, -3, 10)$ را از صفحه $2x + 23y + 13z - 24 = 0$ بدست آورید. ابتدا متغیرهای لازم را تعریف کنید، سپس آنها را در فرمول جایگذاری نمایید.

نسخه
دیباگان

۳ بردارها و ماتریس‌ها

همان‌گونه که از نام MATLAB پیداست^۱، تمام کارهایی که با MATLAB می‌توان انجام داد، مبتنی بر ماتریس‌ها می‌باشد، پس در این نرم‌افزار ماتریس و بردار از جایگاه ویژه‌ای برخوردار هستند. در حالت کلی، ماتریس‌ها و بردارها را با دید آرایه می‌نگریم و بردارها را آرایه‌های یک‌بعدی و ماتریس‌ها را آرایه‌های دوبعدی در نظر می‌گیریم. پیش از آنکه به کاربردهای آرایه‌ها در MATLAB بپردازیم، ابتدا به چگونگی تولید آنها خواهیم پرداخت.

۱.۳ آرایه‌های یک‌بعدی

از نظر هندسی، یک بردار بیانگر موقعیت مکانی یک نقطه در صفحه یا فضا می‌باشد. در حالت کلی یک بردار می‌تواند n عنصر داشته باشد. در ریاضی بردار به صورت

$$\mathbf{x} = \vec{x} = (x_1, x_2, \dots, x_n),$$

نوشته می‌شود. در MATLAB بردار را به عنوان یک آرایه یک‌بعدی می‌شناسیم. بردارها در MATLAB کاربردها بسیار زیادی دارند و تقریباً هرکاری که در MATLAB خواهیم انجام دهیم وابسته به آرایه‌ها، یک‌بعدی یا چند بعدی، می‌باشد. پیش از پرداختن به استفاده از آرایه‌ها باید با چگونگی تولید انواع مختلف بردار آشنا شویم. در این بخش به این موضوع خواهیم پرداخت و برخی از دستورات برای تولید برخی بردارهای خاص را معرفی می‌کنیم.

¹MATrix LABoratory

تولید و دسترسی به بردار

در MATLAB بردار عبارتست از تعدادی عنصر که به طریق خاصی به یک متغیر نسبت داده می‌شوند.

- برای تولید یک بردار می‌توان به یکی از دو روش زیر عمل کرد:

```
Arr_Name = [e_1,e_2,...,e_n]
```

```
Arr_Name = [e_1;e_2;...;e_n]
```

در دستورات بالا Arr_Name نام متغیر است که اختیاری است و تابع قواعد نامگذاری متغیرها می‌باشد. دستور اول منجر به تولید یک بردار سطری می‌شود و دستور دوم یک بردار ستونی تولید می‌کند.

- برای دسترسی به عناصر یک بردار کافایت نام متغیر را بنویسیم و اندیس مورد نظر را در داخل یک جفت پرانتز و در کنار نام متغیر قرار دهیم. برای مثال $x(2)$ به عنصر دوم بردار x دسترسی پیدا می‌کند.

- تابع length به صورت length(Arr_Name) تعداد عناصر آرایه را برگشت می‌دهد.

👉 برای تولید بردار سطری می‌توان از دستور اول بدون کاما نیز استفاده کرد.

مثال ۱.۳. به دستورات زیر برای تولید بردارهای سطری و ستونی و دسترسی به عناصر آنها توجه کنید.

ورودی

```
>> x = [2,-2+2i,-1,7]
>> x(2)
>> y = [3-i;2]
>> y(1)
```

خروجی

```
x =
2 + 0i -2 + 2i -1 + 0i 7 + 0i
ans = -2 + 2i
y =
3 - 1i
2 + 0i
ans = 3 - 1i
```

❏ اگر عناصر بردار به شکل اعداد موهومی باشند، برای مقادیر حقیقی موجود در بردار مقدار 0i به عنوان بخش موهومی در نظر گرفته می شود.

❏ اولین اندیس در بردار عدد 1 می باشد و بزرگترین اندیس برابر با تعداد عناصر بردار خواهد بود. استفاده از هر عدد دیگری در خارج این محدوده منجر بروز خطا می شود.

نکته عملی

برای تولید بردارهایی که تعداد عناصر آنها زیاد باشد ولی مقدار عناصر با یکدیگر فاصله یکسانی داشته باشند، می توان از دستوری به یکی از دو شکل زیر استفاده کرد،

```
Var_Name = [m:step:n]
```

```
Var_Name = m:step:n
```

که در آن

m اولین عنصر آرایه است،

n آخرین عنصر آرایه می باشد،

step فاصله میان هر دو عنصر متوالی است که به آن طول گام گفته می شود و می تواند یک عدد حقیقی مثبت یا منفی باشد.

در هر دو دستور بالا اگر step نوشته نشود، طول گام یک در نظر گرفته خواهد شد.

مثال ۲.۳. برای تولید آرایه ای شامل مضرب های ۷ بین ۲۱ تا ۸۰ و مضارب ۱۳ از ۱۶۹ تا ۶۵ می توان از دستورات زیر استفاده کرد:

```
>> M7 = [21:7:80]
```

```
M7 =
```

```
21    28    35    42    49    56    63    70    77
```

```
>> M13 = 169:-13:65
```

```
M13 =
```

```
169    156    143    130    117    104    91    78    65
```

👉 در مثال دوم از گروه‌های ابتدا و انتها استفاده نشده است و طول گام منفی است.

👉 بجای دو دستور بیان شده در مثال ۲.۳ می‌توان از دستور colon به یکی از دو شکل $\text{colon}(m,n)$ و $\text{colon}(m,\text{step},n)$ استفاده کرد. دستور اول معادل با $m:n$ و دستور دوم معادل با $m:\text{step}:n$ می‌باشد.

اگر بخواهیم آرایه‌هایی تولید کنیم که دارای تعداد مشخصی عنصر باشند می‌توان از دستورات خاصی استفاده کرد.

نکته عملی

با استفاده از دو دستور زیر می‌توان آرایه‌هایی با تعداد مشخصی عنصر تولید کرد،

linspace(m,n) آرایه‌ای تولید می‌کند که عنصر اول آن m و عنصر پایانی آن n می‌باشد و تعداد کل عناصر آرایه ۱۰۰ عدد می‌باشد.

linspace(m,n,N) آرایه تولید می‌کند که عنصر اول آن m و عنصر پایانی آن n می‌باشد و تعداد کل عناصر آرایه N تا می‌باشد.

نتیجه حاصل از دو دستور بالا به صورت بردارهای سطری می‌باشند، به گونه‌ای که فاصله هر دو عنصر متوالی آرایه برابر است.

مثال ۳.۳. به چگونگی استفاده از دستور linspace در دستورات زیر دقت کنید.

```
>> x = linspace(1,30,6)
x =
1.0000    6.8000   12.6000   18.4000   24.2000   30.0000
>> y = linspace(10,1,4)
y =
10     7     4     1
```

👉 مشابه دستور linspace می‌توان با دستور logspace و دقیقاً با ساختاری مشابه با دستور linspace می‌توان آرایه‌های یک بعدی تولید کرد.

نکته عملی

با استفاده از دو دستور زیر می‌توان آرایه‌هایی با تعداد مشخصی عنصر تولید کرد،

logspace(M,N) آرایه‌ای تولید می‌کند که عنصر اول آن 10^M و عنصر پایانی آن 10^N بوده و تعداد کل عناصر آرایه ۵۰ می‌باشد. اگر بجای N مقدار π وارد شود، عنصر پایانی عدد π خواهد بود.

logspace(M,N,P) آرایه‌ای تولید می‌کند که عنصر اول آن 10^M و عنصر پایانی آن 10^N می‌باشد و تعداد کل عناصر آرایه P تا می‌باشد. اگر بجای N مقدار π وارد شود، عنصر پایانی عدد π خواهد بود.

پیشتر دیدید که برای دسترسی به عناصر بردار باید نام آرایه را به همراه شماره اندیس مورد نظر بنویسیم. اما این امکان در آرایه‌ها وجود دارد که بخشی از آرایه را استخراج کرده و در آرایه جدید ذخیره کنیم.

استخراج بخشی از آرایه

برای استخراج بخشی از آرایه یک بعدی می‌توان از کالن، : به شکل زیر استفاده کرد،

`Var_Name = Arr_Name(start:step:stop)`

که در آن:

start اندیسی است که می‌خواهیم استخراج از آن اندیس شروع شود،

stop اندیسی است که می‌خواهیم استخراج تا آن اندیس ادامه یابد،

step طول گام اندیس‌ها می‌باشد.

بدیهی است که `Arr_Name` نام آرایه‌ای است که می‌خواهیم استخراج از آن انجام شود و `Var_Name` نام آرایه‌ای است که نتیجه استخراج در آن ذخیره می‌شود.

مثال ۴.۳. به چگونگی استخراج بخشی از آرایه `x` در دستورات زیر توجه کنید.

```

>> x = 10:10:100
x =
10    20    30    40    50    60    70    80    90   100
>> x(2:7)
ans =
20    30    40    50    60    70
>> x(2:2:end)
ans =
20    40    60    80   100
>> x(10:-3:1)
ans =
100    70    40    10
>> x([3,8,2,9,1,4])
ans =
30    80    20    90    10    40

```

👉 در دستور سوم، با قرار دادن end بجای اندیس پایانی، استخراج تا انتهای آرایه انجام شد.

👉 در دستور چهارم از طول گام اندیس منفی برای استخراج از آخر به اول استفاده شده است.

👉 در دستور آخر، با قرار دادن اندیس‌های مورد نظر در داخل جفت کروشه، تنها عناصر متناظر با اندیس‌های مشخص شده از آرایه استخراج شدند.

۲.۳ آرایه‌های دوبعدی

در MATLAB آرایه‌های دوبعدی همان ماتریس‌ها می‌باشند که از اهمیت بسیار زیادی برخوردار هستند. در این بخش به چگونگی تولید ماتریس‌ها می‌پردازیم و برخی از عملیاتی که می‌توان روی آنها انجام داد را معرفی خواهیم کرد. چون آرایه‌ها در MATLAB پایه و اساس تمام کارهایی است که می‌توان با این نرم‌افزار انجام داد، لذا داشتن مهارت کافی در این زمینه می‌تواند در استفاده بهتر از MATLAB کمک زیادی به کاربر بکند.

۱.۲.۳ تعریف ماتریس

تعریف ماتریس در MATLAB تا اندازه‌ای شبیه تعریف بردار است.

تعریف آرایه دوبعدی

برای تعریف آرایه‌های دوبعدی، می‌توان به یکی از دو روش زیر عمل کرد:

```
Mat_Name = [row1;row2;...;rown]
Mat_Name = [row1
row2
.
rown]
```

که در هر دو دستور row1 و بقیه سطرها کاملاً مشابه آرایه‌های یک‌بعدی تعریف می‌شوند. همچنین Mat_Name نام متغیر دلخواهی است که برای ماتریس انتخاب می‌شود. تابع $\text{size}(A)$ یک بردار دو عنصری به شکل $[m,n]$ برگشت می‌دهد که در m تعداد سطرها و در n تعداد ستون‌های ماتریس ذخیره می‌شوند.

👉 به قرار گرفتن سمی کالن بین هر دو سطر دقت کنید.

مثال ۵.۳. به چگونگی استفاده از دو روش بیان شده در تعریف ماتریس‌های زیر توجه کنید.

ورودی

```
>> A = [2,-1,3 ; 1,4,-3 ; 8,2,-4]
>> B = [2,-1
3,1
4,3]
```

خروجی

```
A =
2    -1     3
1     4    -3
8     2    -4

B =
2    -1
3     1
4     3
```

👉 در تولید ماتریس‌ها می‌توان از توابع ریاضی که پیشتر معرفی کردیم، به‌عنوان درآیه استفاده کرد، برای مثال کد زیر را به همراه خروجی ببینید.

ورودی	خروجی
<pre>>> A = [sind(30),exp(1),cos(pi/3) sqrt(8),log10(50),log(30) ceil(pi),floor(2.718),5] >> [m,n] = size(A)</pre>	<pre>A = 0.5000 2.7183 0.5000 2.8284 1.6990 3.4012 4.0000 2.0000 5.0000 m = 2 n = 3</pre>

👉 در تعریف بردار دیدید که از سمی‌کالن به‌عنوان جداساز استفاده کردیم. در واقع بردار یک ماتریس $n \times 1$ می‌باشد. برای مثال به کد زیر و خروجی حاصل از آن دقت کنید.

ورودی	خروجی
<pre>>> v = [1;2;3;4;5]</pre>	<pre>v = 1 2 3 4 5</pre>

یکی از موارد بسیار پرکاربرد در ماتریس‌ها و بردارها بحث ترانزاده کردن ماتریس و بردار می‌باشد. برای این کار دستور خاصی وجود ندارد و به‌سادگی می‌توان ترانزاده یک ماتریس حقیقی یا مختلط را بدست آورد. ولی اگر ماتریس مختلط باشد به دو روش می‌توان ترانزاده را یافت که حاصل دو روش با هم تفاوت دارند.

ترانهاده کردن ماتریس

برای ترانهاده کردن ماتریس کافیست علامت کوتیشن (پرایم) را در کنار نام ماتریس یا بردار قرار دهیم. به دو روش می‌توان ماتریس A را ترانهاده کرد: A' که ترانهاده مزدوج ماتریس A ، و $A.'$ که ترانهاده ماتریس A (بدون مزدوج کردن عناصر) را محاسبه می‌کند.

مثال ۶.۳. به تفاوت میان خروجی حاصل از اجرای $'$ و $.$ در دستورات زیر توجه کنید.

ورودی

```
>> B = [1+i,2,3-2i
1-i,2,-1]
>> B1 = B'
>> B2 = B.'
```

خروجی

```
>> B = [1+i,2,3-2i
1-i,2,-1]
B1 =
1 - 1i    1 + 1i
2 - 0i    2 - 0i
3 + 2i   -1 - 0i
B2 =
1 + 1i    1 - 1i
2 + 0i    2 + 0i
3 - 2i   -1 + 0i
```

❗ اگر عناصر ماتریس همگی اعداد حقیقی باشند، تفاوتی میان $'$ و $.$ نیست. در جبرخطی، سه ماتریس خاص وجود دارد که از اهمیت ویژه‌ای برخوردارند: ماتریس همانی، ماتریس یک و ماتریس صفر، این سه ماتریس را می‌توان با دستورات خاصی تعریف کرد.

ماتریس‌های همانی، یک و صفر

با دستورات زیر می‌توان این سه ماتریس را به سادگی تولید کرد.

eye(n) ماتریس همانی $I_{n \times n}$ را تولید می‌کند،

ones(m,n) یک ماتریس $m \times n$ تولید می‌کند که تمام عناصر آن یک می‌باشد،

zeros(m,n) یک ماتریس $m \times n$ تولید می‌کند که تمام عناصر آن صفر می‌باشد.

👉 در دو دستور آخر اگر یک عدد به عنوان آرگومان داده شود، ماتریس حاصل مربعی خواهد بود.

مثال ۷.۳. به دستورات زیر و خروجی آنها توجه کنید.

ورودی	خروجی
<code>>> I = eye(3)</code>	$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
<code>>> O = ones(3,4)</code>	$O = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$
<code>>> Z = zeros(3,2)</code>	$Z = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$

دستور diag

در MATLAB دستوری وجود دارد که می‌تواند از روی یک بردار، یک ماتریس قطری بسازد. شکل کلی این دستور به صورت `diag(v)` است که در آن v یک بردار سطری یا ستونی است. حاصل این دستور یک ماتریس مربعی است به گونه‌ای که عناصر بردار روی قطر قرار می‌گیرند و بقیه عناصر ماتریس صفر می‌باشند.

اگر ورودی این دستور یک ماتریس مربعی باشد، خروجی، یک بردار ستونی است که عناصر آن قطر اصلی ماتریس می‌باشد.

ورودی

```
>> v = [-1,1,2,4];
>> A = diag(v)
>> B = [1,2,3;4,5,6;7,8,9];
>> d = diag(B)
```

خروجی

```
A =
-1    0    0    0
 0    1    0    0
 0    0    2    0
 0    0    0    4

d =
1
5
9
```

📌 در دستور آخر توجه کنید که خروجی به صورت یک بردار ستونی است و اگر بخواهیم خروجی به صورت بردار سطری در متغیر d ذخیره شود، باید دستور diag را به شکل 'diag(B)' بکار ببریم. ماتریس‌های خاص دیگری نیز وجود دارند که در فصل ۴ به معرفی و استفاده از آنها خواهیم پرداخت.

در ادامه این بخش به بیان چگونگی دستیابی و استفاده از عناصر یک ماتریس خواهیم پرداخت و استفاده از دونه‌قطه را در استخراج بخش‌هایی از یک ماتریس بیان خواهیم کرد.

دسترسی به عناصر ماتریس

دسترسی به عناصر ماتریس شبیه دسترسی به عناصر بردار می‌باشد، با این تفاوت که باید بجای یک اندیس، دو اندیس را مشخص نمود. برای مثال اگر $A_{3 \times 5}$ یک ماتریس باشد و در MATLAB تعریف شده باشد، با دستور $A(m,n)$ می‌توان به عناصر آن دسترسی پیدا کرد که در آن m عددی بین ۱ تا ۳ و n عددی بین ۱ تا ۵ می‌باشد.

مثال ۸.۳. به چگونگی دسترسی به عناصر ماتریس تعریف شده در کد زیر دقت کنید.

ورودی

```
>> A = [-1,4,3,7
2,-1,4,2
3,0,4,-5]

>> A(2,3)
```

خروجی

```
A =
-1     4     3     7
2    -1     4     2
3     0     4    -5

ans = 4
```

۲.۲.۳ استخراج بخشی از ماتریس

در MATLAB مانند کاری که در بردارها انجام دادیم، می‌توان بخش‌هایی از ماتریس را استخراج کرد. با توجه به اینکه ماتریس یک آرایه دوبعدی است استخراج بخش‌هایی از ماتریس در مقایسه بردارها گسترده‌تر است.

استخراج بخش‌هایی از ماتریس

برای استخراج بخش‌هایی از ماتریس می‌توان از حالات زیر استفاده کرد. در تمام حالات فرض بر این است که ماتریس مورد استفاده A می‌باشد که ابتدا تعریف شده است.

$A(:)$ تمام عناصر ماتریس را به شکل ستون به ستون و به صورت یک بردار برمی‌گرداند.

$A(:, :)$ تمام عناصر ماتریس را برمی‌گرداند.

$A(:, n)$ تمام عناصر ستون n ام ماتریس را برمی‌گرداند.

$A(n, :)$ تمام عناصر سطر n ام ماتریس را برمی‌گرداند.

$A(:, m:n)$ ستون‌های ماتریس از ستون m ام تا ستون n ام را برمی‌گرداند.

$A(m:n, :)$ سطرهاى ماتریس از سطر m ام تا سطر n ام را برمی‌گرداند.

$A(m:n, p:q)$ عناصر واقع بر سطرهاى m تا n و ستون‌های p تا q ماتریس را برمی‌گرداند.

مثال ۹.۳. به خروجی حاصل از دستورات زیر دقت کنید. پیشنهاد می‌شود دستورات ورودی را در یک m-فایل بنویسید و اجرا کنید.

ورودی

```
A = [1,2,3,4,5
6,7,8,9,10
11,12,13,14,15
16,17,18,19,20];
B = A(:, :)
C = A(:, 3)
D = A(2, :)
E = A(:, 2:4)
F = A(2:3, :)
G = A(2:3, 2:4)
```

خروجی

```
B =
1      2      3      4      5
6      7      8      9     10
11     12     13     14     15
16     17     18     19     20
C =
3
8
13
18
D =
6      7      8      9     10
E =
2      3      4
7      8      9
12     13     14
17     18     19
F =
6      7      8      9     10
11     12     13     14     15
G =
7      8      9
12     13     14
```

تولید آرایه خالی

در MATLAB می‌توان با دستور

```
A = []
```

یک آرایه خالی تولید کرد، که در آن A نام دلخواهی می‌باشد. در این صورت می‌توان با دستورات

```
A(i,j) = value
```

```
A(i) = value
```

به عنصر A_{ij} یا A_i مقداردهی کرد. با اجرای این دستور فقط عنصر مشخص شده مقدار می‌گیرد و بقیه عناصر صفر خواهند بود. بدیهی است اگر در دستورات بالا مقادیر i و j یک باشند، آرایه حاصل یک عنصری خواهد بود.

مثال ۱۰.۳. به چگونگی تولید آرایه‌های خالی در دستورات زیر توجه کنید.

ورودی

```
A = []
A(2,3) = 5
v = []
v(3) = 2
```

خروجی

```
A = []
A =
0      0      0
0      0      5
v = []
v =
0      0      2
```

اگر یک آرایه موجود باشد می‌توان در آن تغییراتی ایجاد کرد. فرض کنید A یک ماتریس $m \times n$ و v یک بردار n تایی باشد. در این صورت به روشی که در بالا بیان کردیم، می‌توان برخی عناصر موجود در آرایه را تغییر داد، یا عناصر جدیدی به آرایه در خارج از محدوده اندیس‌های موجود اضافه کرد، در این صورت ابعاد آرایه تغییر خواهد کرد و به عناصری که داری مقدار نیستند، مقدار صفر داده خواهد شد. برای مثال کد زیر را به همراه خروجی مشاهده کنید.

ورودی

```
v = [1,2]
v(4) = 7
u = [1:3]
w = [4,5,6]
u(4:6) = w
```

خروجی

```
v =
1      2
v =
1      2      0      7
u =
1      2      3
w =
4      5      6
u =
1      2      3      4      5      6
```

👉 به وجود عنصر صفر در محل سوم آرایه توجه کنید.

👉 با سه دستور آخر بردار w را به انتهای بردار u اضافه کردیم.

حال اگر بخواهیم مثلاً سطر یا ستون جدیدی به ماتریس اضافه کنیم، یا سطر یا ستونی را از ماتریس حذف کنیم، راه حل چیست؟ این کار را می‌توان با استفاده از روشی که برای استخراج بخشی از آرایه یا ماتریس بیان کردیم انجام داد. در قالب چند مثال به بیان چگونگی انجام این گونه عملیات می‌پردازیم.

مثال ۱۱.۳. در این مثال یک سطر جدید به ماتریس $\text{eye}(2)$ افزوده شده است.

ورودی

```
A = eye(2)
r = [5,3];
A(3,:) = r
```

خروجی

```
A =
1      0
0      1
A =
1      0
0      1
5      3
```

مثال ۱۲.۳. در این مثال یک ستون جدید به ماتریس $\text{eye}(2)$ افزوده شده است.

ورودی

```
A = eye(2)
r = [5;3];
A(:,3) = r
```

خروجی

```
A =
1     0
0     1
A =
1     0     5
0     1     3
```

مثال ۱۳.۳. در این مثال بخشی از ماتریس $\text{eye}(4)$ با یک ماتریس 2×2 جایگزین شده است.

ورودی

```
A = eye(4);
r = [5,3
7,4];
A(2:3,3:4) = r
```

خروجی

```
A =
1     0     0     0
0     1     5     3
0     0     7     4
0     0     0     1
```

اگر بخواهیم تمام عناصر یک سطر، یک ستون یا بخشی از یک ماتریس را به یک عدد تغییر دهیم می‌توان به شکل ساده‌تری این کار را انجام داد.

مثال ۱۴.۳. در این مثال به تمام عناصر ستون سوم ماتریس $\text{eye}(3)$ مقدار -1 داده شده است.

ورودی

```
A = eye(3);
A(:,3) = -1
```

خروجی

```
A =
1     0    -1
0     1    -1
0     0    -1
```

مثال ۱۵.۳. در این مثال به تمام عناصر بخشی از ماتریس $\text{eye}(4)$ مقدار -1 داده شده است.


```
A = zeros(4,10);
A(4,:) = 1;
A(2,:) = [100:-10:10];
A(1,:) = [5:5:50]
```

خروجی دستورات بالا به شکل زیر می باشد.

```
A =
5    10    15    20    25    30    35    40    45    50
100   90   80   70   60   50   40   30   20   10
0      0      0      0      0      0      0      0      0      0
1      1      1      1      1      1      1      1      1      1
```

مثال ۱۸.۳. یک ماتریس $A_{n \times n}$ با n زوج تولید کنید، به گونه ای که عناصر دو سطر و دو ستون وسط ماتریس یک و بقیه عناصر صفر باشند. کد زیر را برای مقادیر مختلف n در یک m -فایل آزمایش کنید.

```
n = 6
A = zeros(n);
A(floor(n/2):floor(n/2)+1,:) = ones(2,n);
A(:,floor(n/2):floor(n/2)+1) = ones(n,2);
A
```

خروجی کد بالا به صورت زیر است.

```
A =
0      0      1      1      0      0
0      0      1      1      0      0
1      1      1      1      1      1
1      1      1      1      1      1
0      0      1      1      0      0
0      0      1      1      0      0
```

۳.۳ تمرین

تمرین ۱.۳. یک بردار سطری با عناصر زیر تولید کنید،

$$3, 4, 2/55, \frac{68}{16}, 45, \sqrt{11}, \cos 25^\circ, 0/5.$$

تمرین ۲.۳. یک بردار سطری با عناصر زیر تولید کنید،

$$\frac{54}{3 + 4/32}, 32, 6/32 - 7/32, 54, e^{3/7}, \sin 66^\circ + \cos \frac{3\pi}{8}.$$

تمرین ۳.۳. یک بردار ستونی با عناصر زیر تولید کنید،

$$25/5, \frac{(14 \tan 58^\circ)}{(2/12 + 11)}, 6!, 2/74, 0/375, \frac{\pi}{5}.$$

تمرین ۴.۳. یک بردار ستونی با عناصر زیر تولید کنید،

$$\frac{32}{3/32}, \sin^2 35^\circ, 6/1, \ln 29^2, 0/0552, \ln^2 29, 133.$$

تمرین ۵.۳. فرض کنید $x = 0/125$ و $y = 2/5$. یک بردار ستونی با عناصری به شکل زیر تولید کنید،

$$y, y^x, \ln(y/x), y \cdot x, x + y$$

تمرین ۶.۳. فرض کنید $a = 3/5$ و $y = -6/4$. یک بردار سطری با عناصری به شکل زیر تولید کنید،

$$a, a^2, \frac{a}{b}, a \cdot b, \sqrt{a}$$

تمرین ۷.۳. برداری سطری تولید کنید که عنصر اول آن ۲ و عنصر پایانی آن ۳۷ و طول گام ۵ باشد.

تمرین ۸.۳. برداری سطری با ۹ عنصر که با فاصله مساوی از هم توزیع شده‌اند تولید کنید به گونه‌ای که عنصر اول آن ۸۱ و عنصر آخر آن ۱۲ باشد.

تمرین ۹.۳. برداری ستونی با ۱۵ عنصر که با فاصله مساوی از هم توزیع شده‌اند تولید کنید به گونه‌ای که عنصر اول آن ۲۱- و عنصر آخر آن ۱۲ باشد.

تمرین ۱۰.۳. یک بردار سطری تولید کنید به گونه‌ای که عنصر اول آن ۲۵، عنصر آخر آن صفر و طول گام $2/5$ باشد.

تمرین ۱۱.۳. یا استفاده از دونقطه، یک بردار ۷ تایی به نام seven تولید کنید که همه عناصر آن ۷ باشد.

تمرین ۱۲.۳. با استفاده از یک دستور و بدون آنکه عناصر را یک‌به‌یک بنویسید برداری سطری با ۹ عنصر تولید کنید به گونه‌ای که عنصر آخر آن ۵ و بقیه عناصر آن صفر باشد.

تمرین ۱۳.۳. بردار زیر را با یک دستور و بدون نوشتن تک تک عناصر تولید کنید،

$$v = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1).$$

تمرین ۱۴.۳. برداری با ۱۴ عنصر به نام a به گونه‌ای تولید کنید که عنصر اول آن ۱۴ و عنصر پایانی، ۴۹ و طول گام ۳ باشد. سپس برداری به نام b تولید کنید به گونه‌ای که ۴ عنصر اول آن، چهار عنصر آخر بردار a و چهار عنصر آخر آن، چهار عنصر اول بردار a باشد.

تمرین ۱۵.۳. بردار c را با ۱۶ عضو به گونه‌ای بسازید که عنصر اول آن ۱۳، عنصر پایانی ۷۳ و طول گام ۴ باشد. سپس به دو پرسش زیر پاسخ دهید.

- بدون نوشتن تک تک عناصر تمام مقادیر واقع در اندیس‌های فرد را از بردار c استخراج کنید و در برداری سطری به نام codd ذخیره کنید.
- بدون نوشتن تک تک عناصر تمام مقادیر واقع در اندیس‌های زوج را از بردار c استخراج کنید و در برداری سطری به نام ceven ذخیره کنید.

تمرین ۱۶.۳. بدون نوشتن تک تک عناصر ماتریس زیر را تولید کنید،

$$A = \begin{bmatrix} 0 & 5 & 10 & 15 & 20 & 25 & 30 \\ 600 & 500 & 400 & 300 & 200 & 100 & 0 \\ 0 & 0.8333 & 1/6667 & 2/5 & 3/3333 & 4/1667 & 5 \end{bmatrix}$$

تمرین ۱۷.۳. ماتریس $A_{n \times n}$ را که n عددی فرد است بدون نوشتن تک تک عناصر به گونه‌ای بسازید که سطر میانی و ستون میانی به شکل بردار $(1, 2, 3, \dots, n)$ باشد و بقیه عناصر صفر

باشند. برای مثال اگر $n = 5$ ماتریس زیر تولید شود.

$$B = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 1 & 2 & 3 & 4 & 5 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 \end{bmatrix}$$

تمرین ۱۸.۳. ماتریس‌های زیر را با یک دستور و بدون نوشتن تک‌تک عناصر ماتریس تولید کنید.

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & 6 \\ 0 & 0 & 0 & 6 & 6 \end{bmatrix}, \quad E = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 4 & 5 & 6 \\ 0 & 0 & 7 & 8 & 9 \end{bmatrix}$$

تمرین ۱۹.۳. ماتریس‌های زیر را با یک دستور و بدون نوشتن تک‌تک عناصر ماتریس تولید کنید.

$$F = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 10 & 20 \\ 0 & 0 & 2 & 8 & 26 \\ 0 & 0 & 3 & 6 & 32 \end{bmatrix}$$

تمرین ۲۰.۳. سه بردار زیر را تعریف کنید،

$$\mathbf{a} = (7, 2, -3, 1, 0), \quad \mathbf{b} = (-3, 7, 10, 0, -2), \quad \mathbf{c} = (1, 0, 4, -6, 5)$$

سپس به پرسش‌های زیر بدون نوشتن تک‌تک عناصر پاسخ دهید.

- یک ماتریس ۳ در ۵ تولید کنید که سطرهاى آن a , b و c باشند.
- یک ماتریس ۵ در ۳ تولید کنید که ستون‌های آن a , b و c باشند.

تمرین ۲۱.۳. سه بردار زیر را تعریف کنید،

$$\mathbf{a} = \begin{bmatrix} 7 & 2 & -3 & 1 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} -3 & 7 & 10 & 0 & -2 \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} 1 & 0 & 4 & -6 & 5 \end{bmatrix},$$

سپس به پرسش‌های زیر بدون نوشتن تک‌تک عناصر پاسخ دهید.

• یک ماتریس ۳ در ۳ تولید کنید که سطرهای اول، دوم و سوم آن شامل سه عنصر اول بردارهای a , b و c باشند.

• یک ماتریس ۳ در ۳ تولید کنید که ستون‌های اول، دوم و سوم آن شامل سه عنصر اول بردارهای a , b و c باشند.

تمرین ۲۲.۳. دو بردار زیر را تعریف کنید،

$$a = [-4 \quad 10 \quad 0.5 \quad 1/8 \quad -2/3 \quad 7], \quad b = [-1 \quad 3/14 \quad 2/781 \quad -2/6 \quad 5 \quad -0.25],$$

سپس به پرسش‌های زیر بدون نوشتن تک تک عناصر پاسخ دهید.

• یک ماتریس ۲ در ۴ تولید کنید که سطرهای اول آن شامل عناصر دوم تا پنجم بردار a و سطر دوم آن شامل عناصر سوم تا ششم بردار b باشد.

• یک ماتریس ۳ در ۴ تولید کنید که ستون اول آن شامل عناصر دوم تا چهارم بردار a ، ستون دوم آن شامل عناصر چهارم تا ششم بردار a ، ستون سوم آن شامل عناصر اول تا سوم بردار b و ستون چهارم آن شامل عناصر سوم تا پنجم بردار b باشد.

تمرین ۲۳.۳. خروجی دستورات زیر را حدس بزنید و بر روی کاغذ یادداشت کنید، سپس دستورات را در محیط MATLAB اجرا کنید و درستی حدس‌های خود را بررسی کنید.

• دستورات را در سطرهای جداگانه اجرا کنید.

$$a = 9:-3:0, \quad b = [a \ a], \quad c = [a;a], \quad d = [a',a'], \\ e = [[a;a;a;a] \ a']$$

• دستورات را در سطرهای جداگانه اجرا کنید.

$$v = [1 \ -2.3 \ 3.14 \ 5 \ -2.25 \ 11 \ 2.781 \ -4.5 \ 8.75 \ 3, \ -7] \\ a = v(2:5), \quad b = v([1,3:7,11]), \quad c = v([10,2,9,4]) \\ d = [v([2 \ 7:10]); v([3,5:7,2])] \\ e = [v([3:5,8]))' \ v([10 \ 6 \ 4 \ 1])' \ v([7:-1:4]))']$$

تمرین ۲۴.۳. ماتریس زیر را در نظر بگیرید،

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 8 & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 & 18 \end{bmatrix}$$

با استفاده از این ماتریس و بدون نوشتن عناصر به صورت تک تک به پرسش‌های زیر پاسخ دهید.

- یک بردار سطری با شش عنصر تولید کنید که شامل عناصر سطر اول ماتریس A باشد.
- یک بردار ستونی با سه عنصر تولید کنید که شامل عناصر ستون ششم ماتریس A باشد.
- یک بردار سطری با شش عنصر تولید کنید که شامل سه عنصر اول از عناصر سطر اول ماتریس A و سه عنصر آخر از سطر سوم ماتریس A باشد.

تمرین ۲۵.۳. ماتریس زیر را در نظر بگیرید،

$$B = \begin{bmatrix} 18 & 17 & 16 & 15 & 14 & 13 \\ 12 & 11 & 10 & 9 & 8 & 7 \\ 6 & 5 & 4 & 3 & 2 & 1 \end{bmatrix}$$

با استفاده از این ماتریس و بدون نوشتن عناصر به صورت تک تک به پرسش‌های زیر پاسخ دهید.

- یک بردار ستونی با شش عنصر تولید کنید که شامل عناصر ستون دوم و پنجم ماتریس B باشد.
- یک بردار ستونی با هفت عنصر تولید کنید که شامل عناصر سوم تا ششم از سطر سوم و ستون دوم ماتریس B باشد.
- یک بردار ستونی با نه عنصر تولید کنید که شامل عناصر ستون‌های دوم، چهارم و ششم ماتریس B باشد.

تمرین ۲۶.۳. ماتریس‌های زیر را تولید کنید،

$$M = \begin{bmatrix} 3 & 5 & 7 & 9 & 11 & 13 \\ 15 & 14 & 13 & 12 & 11 & 10 \\ 1 & 2 & 3 & 1 & 2 & 3 \end{bmatrix}, \quad N = \begin{bmatrix} 33 & 21 & 9 & 14 & 30 \\ 30 & 18 & 6 & 18 & 34 \\ 27 & 15 & 6 & 2238 & \\ 24 & 12 & 1026 & 42 & \end{bmatrix},$$

سپس دستورات زیر را حدس بزنید و روی کاغذ یادداشت کنید سپس دستورات را در محیط MATLAB-LAB اجرا کنید و با نتایجی که یادداشت کرده‌اید مقایسه کنید.

$$\begin{aligned} A &= M([1,2], [2,4,5]), & B &= M(:, [1:3,6]) \\ C &= M([1,3], :) & D &= M([2,3], 5) \end{aligned}$$

$$E = [N(1,1:4)]', \quad N(2,2:5)']$$

$$F = [N(:,3)]' \quad N(3,:)]$$

$$G(3:4,5:6) = N(2:3,4:5)$$

تمرین ۲۷.۳. با دستورات zeros، one و eye ماتریس‌های زیر را تولید کنید.

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

9

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}, \quad \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

۴ عملیات ریاضی روی آرایه‌ها

در MATLAB پس از تعریف متغیرها و مقادیردهی به آنها می‌توان در عبارات محاسباتی دیگر از آنها استفاده کرد، یا آنها را به عنوان آرگومان‌های ورودی به توابع پیش‌ساخته و توابعی که توسط کاربر نوشته می‌شود ارسال کرد. در فصل ۱ با چگونگی اعمال عملگرهای محاسباتی بر مقادیر اسکالری، که می‌توان آنها را به عنوان یک بردار تک عنصری یا یک ماتریس تک عنصری در نظر گرفت، آشنا شدیم، در این فصل به معرفی چگونگی اعمال عملگرهای محاسباتی بر آرایه‌ها خواهیم پرداخت و همچنین با برخی از توابع پیش‌ساخته که آرایه‌ها را به عنوان آرگومانهای ورودی می‌پذیرند آشنا خواهیم شد. استفاده از برخی عملگرهای محاسباتی بر روی آرایه‌ها مشابه حالت اسکالری می‌باشد ولی استفاده از برخی عملگرها بر روی آرایه‌ها کاملاً متفاوت با اعمال آنها بر اسکالرها می‌باشد، لذا نیاز به تمرین بیشتری دارد به همین منظور در بخش ۱.۴ به طور مفصل به بیان چگونگی استفاده از عملگرهای محاسباتی خواهیم پرداخت و در بخش‌های بعدی به معرفی توابع پیش‌ساخته اقدام خواهیم کرد.

۱.۴ عملیات محاسباتی روی آرایه‌ها

برخی از عملگرهای معرفی شده در ۱.۲، را می‌توان به دو روش مختلف بر آرایه‌ها اعمال کرد و نتایج کامل متفاوتی را بدست آورد. استفاده از برخی عملگرها، مانند جمع و تفریق، تا حدی به استفاده از آنها در حالت اسکالری شباهت دارد، ولی عملگرهایی مانند ضرب، تقسیم، توان، تقسیم و تقسیم چپ را می‌توان به دو روش بکار برد که نتیجه حاصل متفاوت بوده و هریک مفهوم ویژه‌ای در جبرخطی دارند. به همین دلیل به طور جداگانه و با دقت زیاد به معرفی چگونگی استفاده از عملگرهای محاسباتی روی آرایه‌ها خواهیم پرداخت.

۱.۱.۴ جمع و تفریق

دو عملگر جمع، +، و تفریق، -، را می‌توان مشابه حالت اسکالری بکار برد و نتایجی که حاصل می‌شود نیز تقریباً مشابه حالت اسکالری می‌باشد.

جمع و تفریق

با توجه به عملوندهایی که دو عملگر جمع و تفریق بر آنها اعمال می‌شود، به دو شکل می‌توان از این دو عملگر استفاده کرد. فرض کنید A و B دو آرایه باشند و m یک کمیت اسکالری، حقیقی یا موهومی، باشد، آنگاه

- نتیجه حاصل از $A+B$ (یا $A-B$) یک آرایه هم‌اندازه با دو آرایه A و B می‌باشد که عناصر آن از جمع (یا تفریق) نظیر به نظیر عناصر آرایه‌های A و B بدست آمده است.
- نتیجه حاصل از $A + c$ ، $A - c$ (یا $c + A$ ، $c - A$) یک ماتریس هم‌اندازه با آرایه A است که عناصر آن از جمع (یا تفریق) عناصر A با (از) مقدار c حاصل شده است.

مثال ۱.۴. به چگونگی استفاده از عملگرهای جمع و تفریق در دستورات زیر توجه کنید.

ورودی	خروجی
$A = [1, 2; 3, 4];$	$D =$
$B = [-1, 3; 5, 7];$	0 5
$c = 2;$	8 11
$v = [10, 20];$	$E =$
$D = A + B$	9 23
$E = v + B$	15 27
$F = A - c$	$F =$
	-1 0
	1 2

👉 در $A + B$ اگر A و B هر دو ماتریس باشند ولی دارای اندازه یکسان نباشند با خطای Matrix dimensions must agree برخورد خواهید کرد.

❏ در دستور $v + B$ یک بردار را با یک ماتریس جمع کرده‌ایم، در این صورت تعداد عناصر بردار باید با تعداد ستون‌های ماتریس برابر باشند.

۲.۱.۴ ضرب آرایه‌ها

عمل ضرب در آرایه با استفاده از کاراکتر $*$ انجام می‌شود و به دو دسته تقسیم می‌شود،

- ضرب اسکالر در آرایه که منجر به ضرب تمام عناصر آرایه در کمیت اسکالری خواهد شد.

ورودی	خروجی
$A = [1, 2, 3,$ $4, 5, 6];$ $c = 2;$ $B = c * A$	$B =$ <div> <div>2</div> <div>4</div> <div>6</div> </div> <div> <div>8</div> <div>10</div> <div>12</div> </div>

- ضرب آرایه در آرایه که در این باره در ادامه به‌طور کامل خواهیم گفت.

ضرب آرایه‌ها

اگر $A_{m \times p}$ و $B_{p \times n}$ دو ماتریس با ابعاد مشخص شده باشند، آنگاه عملگر $*$ به صورت $A * B$ قابل استفاده است. حاصل این ضرب دقیقاً منطبق بر تعریف ضرب دو ماتریس می‌باشد که در جبر خطی تعریف شده است.

مثال ۲.۴. در دستورات زیر ماتریس‌های $A_{3 \times 2}$ و $B_{2 \times 2}$ به صورت زیر تعریف شده‌اند،

$$A = \begin{bmatrix} 1 & 2 \\ 4 & 6 \\ -2 & 5 \end{bmatrix}_{3 \times 2}, \quad B = \begin{bmatrix} 1 & 2 \\ -1 & 4 \end{bmatrix}_{2 \times 2},$$

تعریف شده‌اند. بنا به تعریف ضرب دو ماتریس، می‌دانیم ضرب AB امکان‌پذیر است. دستورات زیر ماتریس حاصل ضرب AB را محاسبه می‌کند. اگر برای این دو ماتریس اقدام به محاسبه BA کنیم با خطا مواجه خواهیم شد.

ورودی

$$A = [1, 2; 4, 6; -2, 5];$$

$$B = [1, 2; -1, 4];$$

$$C = A * B$$

خروجی

$$C =$$

-1	10
-2	32
-7	16

با عملگر ضرب، می‌توان دو بردار را در هم ضرب داخلی کرد، تنها نکته‌ای که باید به آن توجه داشت، این است که ضرب باید به شکل $A * B'$ انجام شود تا انجام ضرب امکان‌پذیر شود، در غیر این صورت با پیغام خطای مبنی بر عدم همخوانی اندازه‌های دو ماتریس مواجه خواهید شد. برای مثال کد زیر را ببینید.

ورودی

$$A = [1, 2 \ 3];$$

$$B = [4 \ 5 \ 6];$$

$$C = A * B'$$

خروجی

$$C =$$

32

ضرب عنصر به عنصر آرایه‌ها

نوع دیگری از ضرب آرایه‌ها وجود دارد که با عملگر $*$ و به صورت $A * B$ بین دو آرایه A و B انجام می‌شود. در ارتباط با این نوع ضرب توجه به نکات زیر توجه کنید:

- این ضرب تنها زمانی امکان‌پذیر است که دو آرایه کاملاً هم‌اندازه باشند.
- در این ضرب قوانین ضرب ماتریس‌ها در جبر خطی حاکم نیستند.
- نتیجه این ضرب، از ضرب هر عنصر آرایه A در عنصر نظیر در آرایه B حاصل می‌شود.
- این ضرب خاصیت جابجایی دارد یعنی همواره رابطه $A * B = B * A$ برقرار است.

برای درک بهتر ضرب عنصر به عنصر در حالت خاص دو بردار را در نظر بگیرید:

$$a = (a_1, a_2, a_3), \quad b = (b_1, b_2, b_3), \quad a * b = (a_1 b_1, a_2 b_2, a_3 b_3).$$

این نکته برای هر دو آرایه هم‌اندازه برقرار می‌باشد.

مثال ۳.۴. ضرب عنصر به عنصر را برای دو ماتریس A و B حساب کنید.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 5 & 8 \\ 1 & 4 & 7 \end{bmatrix}, \quad A.*B = \begin{bmatrix} 2 & 10 & 24 \\ 4 & 20 & 42 \end{bmatrix}$$

کد مربوط به ضرب دو ماتریس بالا در زیر آورده شده است.

ورودی	خروجی
$A = [1, 2 \ 3; 4 \ 5 \ 6];$	$C =$
$B = [2 \ 5 \ 8; 1 \ 4 \ 7];$	2 10 24
$C = A.*B$	4 20 42
$D = B.*A$	$D =$
$v = [1 \ 2 \ 3];$	2 10 24
$u = [4 \ 5 \ 6];$	4 20 42
$w = u.*v$	$w =$
$z = v.*u$	4 10 18
	$z =$
	4 10 18

۳.۱.۴ تقسیم آرایه‌ها

در MATLAB چند تقسیم مختلف را می‌توان برای تقسیم آرایه‌ها بر یکدیگر بکار برد. در این بخش به بیان انواع تقسیم قابل استفاده در آرایه‌ها می‌پردازیم.

تقسیم آرایه بر اسکالر

اگر A یک آرایه و c یک کمیت اسکالری باشد، آنگاه، حاصل تقسیم A/c برداری است که عناصر آن از تقسیم عناصر بردار A بر c حاصل می‌شوند.

برای مثال خروجی کد و خروجی زیر را ببینید.

ورودی	خروجی						
$A = [2, 4 \ 6; 8 \ 10 \ 12];$ $c = 2;$ $B = A/c$	$B =$ <table><tr><td>1</td><td>2</td><td>3</td></tr><tr><td>4</td><td>5</td><td>6</td></tr></table>	1	2	3	4	5	6
1	2	3					
4	5	6					

❗ اگر از دستور c/A استفاده کنید با خطای Matrix dimensions must agree مواجه خواهید شد.

اما اصل ماجرای تقسیم زمانی آغاز می‌شود که در دو طرف عملگر تقسیم، آرایه قرار داشته باشد. در این صورت با دو تعبیر برای تقسیم مواجه هستیم.

تقسیم دو ماتریس بر هم

فرض کنید A ، B و X سه ماتریس باشد آنگاه

• حاصل دستور $X = B/A$ ماتریسی است که در تساوی $XA = B$ صدق می‌کند.

• حاصل دستور $X = A \setminus B$ ماتریسی است که در رابطه $AX = B$ صدق می‌کند.

توجه کنید که X می‌تواند به صورت بردار نیز تولید شود.

مثال ۴.۴. اگر

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 3 & -1 \\ 4 & 1 & 3 \\ 4 & 3 & -3 \end{bmatrix},$$

به چگونگی عمل تقسیم در دستورات زیر توجه کنید. دستورات زیر را در یک m -فایل بنویسید و با برداشتن سمی‌کالن از انتهای دستورات خروجی را مشاهده کنید. توجه کنید که باید $X*A$ و $A*Y$ مساوی B شوند.

ورودی

```
A = [1,2,3;4,5,6;7,5,9];
B = [2,3,-1;4,1,3;4,3,-3];
X = B / A
X*A;
Y = A \ B
A*Y;
```

خروجی

```
X =
-3.5000    2.8333   -0.8333
-1.1667   -0.1667    0.8333
-6.8333    4.1667   -0.8333
Y =
-0.3333   -1.8333    0.8333
0.6667   -1.3333    3.3333
0.3333    2.5000   -2.8333
```

مهمترین کاربرد تقسیم چپ در حالتی است که A یک ماتریس $n \times n$ ، و X ، b ، دو بردار $n \times 1$ باشند. در این حالت دستور $X = A \setminus B$ منجر به جواب دستگاه معادلات خطی زیر می‌گردد،

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

مثال ۵.۴. دستگاه معادلات خطی زیر را حل کنید.

$$\begin{cases} 4x + 2y - z + 2w = 13 \\ -x + 2y + 3z - 4w = -4 \\ x + y - 2z + w = 1 \\ -2x + y + 3z - 2w = 1 \end{cases} \Rightarrow \begin{bmatrix} 4 & 2 & -1 & 2 \\ 1 & 2 & 3 & -4 \\ 1 & 1 & -2 & 1 \\ -2 & 1 & 3 & -2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} 13 \\ -4 \\ 1 \\ 1 \end{bmatrix}.$$

با کد زیر و با استفاده از تقسیم چپ می‌توان جواب این دستگاه را بدست آورد.

ورودی	خروجی
$A = [4, 2, -1, 2$ $-1, 2, 3, -4$ $1, 1, -2, 1$ $-2, 1, 3, -2];$ $b = [13; -4; 1; 1];$ $x = A \setminus b$	$x =$ 1.0000 2.0000 3.0000 4.0000

تقسیم عنصر به عنصر آرایه‌ها

نوع دیگری از تقسیم آرایه‌ها وجود دارد که با عملگر $./$ و به صورت $A ./ B$ بین دو آرایه A و B انجام می‌شود. در ارتباط با این نوع تقسیم به نکات زیر توجه کنید:

- این تقسیم تنها زمانی امکان‌پذیر است که دو آرایه کاملاً هم‌اندازه باشند.
- نتیجه این تقسیم، از تقسیم هر عنصر آرایه A بر عنصر نظیر در آرایه B حاصل می‌شود.
- این تقسیم خاصیت جابجایی ندارد.

📌 برای درک بهتر تقسیم عنصر به عنصر، در حالت خاص دو بردار زیر را در نظر بگیرید:

$$\mathbf{a} = (a_1, a_2, a_3), \quad \mathbf{b} = (b_1, b_2, b_3), \quad \mathbf{a} ./ \mathbf{b} = \left(\frac{a_1}{b_1}, \frac{a_2}{b_2}, \frac{a_3}{b_3} \right).$$

برای ماتریس‌ها نیز همین رول برقرار است و عمل تقسیم دقیقاً به همین صورت انجام می‌شود.

📌 اگر اندازه دو آرایه یکسان نباشند، با خطای `Matrix dimensions must agree` برخورد خواهید کرد.

مثال ۶.۴. به دستورات زیر و چگونگی استفاده از عملگر $./$ در دستورات زیر توجه کنید. پیشنهاد می‌شود دستورات زیر را در یک m -فایل بنویسید و با ایجاد تغییرات در مقادیر ورودی، نتایج حاصل را بررسی کنید.

ورودی	خروجی
<pre>A = [10 20 30 40 50 60 70 80]; B = [2 5 10 8 25 20 14 8]; C = A./B D = B./A</pre>	<pre>C = 5 4 3 5 2 3 5 10 D = 0.2000 0.2500 0.3333 0.2000 0.5000 0.3333 0.2000 0.1000</pre>

۴.۱.۴ به توان رساندن آرایه‌ها

در MATLAB امکان به توان رساندن آرایه نیز وجود دارد. مانند ضرب و تقسیم، عمل توان نیز به دو شکل قابل انجام است که در ادامه هر دو شکل استفاده را بیان خواهیم کرد.

به توان رساندن آرایه‌ها

فرض کنید $A_{n \times n}$ یک ماتریس مربعی و n یک عدد صحیح باشد، آنگاه، برای مقادیر مثبت n ، دستور A^n توان n ام ماتریس A را محاسبه می‌کند که از n بار ضرب ماتریس A در خودش حاصل می‌شود و برای مقادیر منفی n ، این دستور توان n ام ماتریس A^{-1} را محاسبه می‌کند.

👉 اگر $n = -1$ باشد، آنگاه حاصل دستور A^{-1} وارون ماتریس A می‌باشد.

👉 اگر ماتریس مربعی نباشد و از عملگر توان برای آن استفاده کنید، با خطا مواجه خواهید شد.

مثال ۷.۴. فرض کنید، $A = \begin{bmatrix} 4 & 3 \\ -1 & 5 \end{bmatrix}$ ماتریس مفروضی باشد. به خروجی حاصل از دستورات زیر توجه کنید. در دستورات زیر حاصل دستور A^{-1} وارون ماتریس A است. درستی این موضوع با دستور آخر، یعنی دستور $A * C$ بررسی شده است.

ورودی	خروجی
$A = \begin{bmatrix} 4 & 3 \\ -1 & 5 \end{bmatrix};$ $B = A^2$ $C = A^{-1}$ $A * C$	$B =$ $\begin{bmatrix} 13 & 27 \\ -9 & 22 \end{bmatrix}$ $C =$ $\begin{bmatrix} 0.2174 & -0.1304 \\ 0.0435 & 0.1739 \end{bmatrix}$ $ans =$ $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

مفهوم به توان رساندن عنصر به عنصر را نیز می‌توان مشابه ضرب و تقسیم بیان کرد.

به توان رساندن عنصر به عنصر

فرض کنید A و B دو آرایه هم‌اندازه باشند، در این صورت با دستور $A.^B$ می‌توان آرایه جدیدی ساخت که هم‌اندازه با دو آرایه دیگر بوده و عناصر به صورت $a_{ij}^{b_{ij}}$ می‌باشند.

👉 برای درک بهتر ضرب عنصر به عنصر در حالت خاص دو بردار زیر را در نظر بگیرید:

$$\mathbf{a} = (a_1, a_2, a_3), \quad \mathbf{b} = (b_1, b_2, b_3), \quad \mathbf{a}.\mathbf{b} = (a_1^{b_1}, a_2^{b_2}, a_3^{b_3}).$$

برای ماتریس‌ها نیز همین روال برقرار است و عمل تقسیم دقیقاً به همین صورت انجام می‌شود. همچنین اگر اندازه‌های دو آرایه یکسان نباشند با خطای Matrix dimensions must agree مواجه خواهید شد.

مثال ۸.۴. به چگونگی اجرای دستورات زیر برای دو ماتریس ۲ در ۲ توجه کنید.

ورودی	خروجی
$A = \begin{bmatrix} 4 & 3; -1 & 5 \end{bmatrix};$ $B = \begin{bmatrix} 2 & 3; 4 & -1 \end{bmatrix};$ $C = A.^B$	$C =$ $\begin{bmatrix} 16.0000 & 27.0000 \\ 1.0000 & 0.2000 \end{bmatrix}$

۲.۴ اعمال توابع پیش‌ساخته بر آرایه‌ها

در MATLAB تقریباً هر کاری که بخواهیم انجام دهیم، با آرایه‌ها درگیر خواهیم شد. اما همیشه آرایه‌ها به شکل خام مورد استفاده قرار نمی‌گیرند و شاید نیاز به اعمال برخی توابع پیش‌ساخته ریاضی که در بخش ۳.۲ معرفی شدند بر آنها باشد. در برخی مسایل حتی اثری از بردار در صورت سوال دیده نمی‌شود، ولی برای حل آن با MATLAB نیازمند به بردارها و ماتریس‌ها خواهیم بود. پیش از پرداختن به توضیحات بیشتر در این زمینه، مثالی بیان می‌کنیم که نیاز به این موضوع را بتوان بهتر احساس کرد.

مثال ۹.۴. فرض کنید $x \in [0, 100]$ ، مطلوبست محاسبه مقدار تابع $y = x^2 - \sqrt{\ln(x+1)}$ در نقاط $x_k = 1 + 0.5k$ برای $k = 0, \dots, 200$

برای حل این مساله با MATLAB بهترین روش، استفاده از آرایه‌ها و اعمال توابع ریاضی بر آنهاست، چراکه در غیر این صورت باید ۲۰۰ مقدار را در تابع جایگزین کنیم و حاصل را محاسبه نماییم، ولی با استفاده از آرایه‌ها و توابع پیش‌ساخته ریاضی به سادگی و با دو دستور می‌توان مساله را حل کرد.

اعمال توابع ریاضی بر آرایه‌ها

تمام توابع ریاضی پیش‌ساخته در MATLAB را می‌توان به سادگی بر هر آرایه‌ای، بردار یا ماتریس، اعمال کرد. به این منظور کافیهست به شکل زیر عمل کنید:

`Function_Name(Arg)`

که در آن `Function_Name` نام تابع و `Arg` نام آرایه می‌باشد. در این صورت تابع مورد نظر بر تک تک عناصر آرایه اثر کرده و یک آرایه جدید و هم اندازه با آرایه ورودی تولید می‌کند.

👉 اگر در عبارت محاسباتی، به جز توابع ریاضی، عملیات ضرب، تقسیم و توان وجود داشت، مانند مثال ۹.۴، باید از عملگرهای ضرب، تقسیم و توان به شکل عنصر به عنصر استفاده کرد، یعنی عملگرهای `*`، `/` و `^`. توجه کنید که اگر از ضرب، تقسیم و توان معمولی استفاده کنید با خطا مواجه خواهید شد.

حال به سادگی و با کد زیر می‌توان مثال ۹.۴ را حل کرد. برای کوتاه شدن خروجی مساله را

بجای 200° برای 1° و بر بازه $[0, 5]$ حل می‌کنیم. به استفاده از دستور \wedge در تولید بردار y توجه کنید.

```
x = 0:0.5:5
x =
Columns 1 through 6
0    0.5000    1.0000    1.5000    2.0000    2.5000
Columns 7 through 11
3.0000    3.5000    4.0000    4.5000    5.0000
>> y = x.^2 + log(x+1)
y =
Columns 1 through 6
0    0.6555    1.6931    3.1663    5.0986    7.5028
Columns 7 through 11
10.3863    13.7541    17.6094    21.9547    26.7918
```

مثال ۱۰.۴. مقدار تابع

$$y = \frac{x^2 - \cos(x)}{\sqrt[3]{x \sin(x) + 1}}, \quad x \in [-\pi, \pi],$$

را در بازه مشخص شده و در ۷ نقطه با فاصله یکسان حساب کنید.

```
>> x = linspace(-pi,pi,7)
x =
-3.1416   -2.0944   -1.0472         0    1.0472    2.0944    3.1416

>> y = (x.^2 - cos(x))./(nthroot(x.*sin(x) + 1,3))
y =
10.8696    3.4612    0.4811   -1.0000    0.4811    3.4612   10.8696
```

۳.۴ توابع پیش‌ساخته مخصوص آرایه‌ها

علاوه بر توابع ریاضی که در بخش ۲.۴ مورد استفاده قرار گرفتند، برخی توابع پیش‌ساخته در MATLAB وجود دارند که مخصوص آرایه‌ها هستند. در این بخش به معرفی این توابع خواهیم پرداخت. در جدول‌های ۱.۴ و ۲.۴ برخی از توابع که بر یک آرایه، بردار یا ماتریس، اثر می‌کنند آورده شده است. تمام توابع این دو جدول بر یک کمیت اسکالری قابل اعمال هستند، چراکه کمیت اسکالری یک ماتریس 1×1 یا یک بردار تک عضوی است.

جدول ۲.۴: توابع پیش‌ساخته برای آرایه‌ها

تابع	عمل	مثال
<code>cross(a,b)</code>	ضرب خارجی دو بردار یعنی $a \times b$ را برمی‌گرداند.	<pre>>> a = [1,-2,3]; >> b = [4,-5,6]; >> c = cross(a,b) c = 3 6 3</pre>
<code>cross(a,b)</code>	ضرب داخلی دو بردار یعنی $a \cdot b$ را برمی‌گرداند.	<pre>>> a = [1,-2,3]; >> b = [4,-5,6]; >> c = cross(a,b) ans = 32</pre>
<code>prod(x)</code>	حاصل ضرب عناصر بردار x را برمی‌گرداند.	<pre>>> x = [1,2,3,4]; >> prod(x) ans = 24</pre>

📌 دستور `cross` فقط برای بردارهای سه عنصری قابل اجراست و در صورت استفاده از آن برای بردارهایی با عناصر بیشتر یا کمتر از سه عنصر، با خطا مواجه خواهید شد.

در MATLAB یک تابع پیش‌ساخته وجود دارد که به کمک آن می‌توان اعداد و آرایه‌های تصادفی ایجاد کرد در ادامه به معرفی این تابع و برخی از استفاده‌های آن می‌پردازیم.

جدول ۱۰۴: توابع پیش‌ساخته برای آرایه‌ها

مثال	عمل	تابع
<pre>>> A = [2,4;-3,1]; >> inv(A) ans = 0.0714 -0.2857 0.2143 0.1429</pre>	وارون ماتریس A را برمی‌گرداند.	<code>inv(A)</code>
<pre>>> A = [2,4;-3,1]; >> det(A) ans = 14</pre>	دترمینان ماتریس A را برمی‌گرداند.	<code>det(A)</code>
<pre>>> x = [2,7,9,12,18]; >> median(x) ans = 9 >> y = [1,-4,3,9,6,2]; >> median(y) ans = 5000.2</pre>	مقدار میانه را برای بردار x بر می‌گرداند.	<code>median(x)</code>
<pre>>> x = [1,3,9,6,2]; >> sort(x) ans = 1 2 3 6 9</pre>	عناصر بردار x را به‌شکل صعودی مرتب می‌کند.	<code>sort(x)</code>
<pre>>> x = [1,3,9,6,2]; >> sum(x) ans = 21</pre>	مجموع عناصر بردار x را بر می‌گرداند.	<code>sum(x)</code>
<pre>>> x = [1,3,9,6,2]; >> [v,p] = min(x) v = 1 p = 1</pre>	کوچکترین عنصر بردار x را برمی‌گرداند. v مقدار p اندیس است.	<code>min(x)</code>
<pre>>> x = [1,3,9,6,2]; >> [v,p] = min(x) v = 1 p = 1</pre>	بزرگترین عنصر بردار x را برمی‌گرداند. v مقدار p اندیس است.	<code>max(x)</code>
<pre>>> x = [1,3,9,6,2]; >> mean(x) ans = 4.2000</pre>	مقدار میانگین عناصر بردار x را برمی‌گرداند.	<code>mean(x)</code>

تولید اعداد و آرایه‌های تصادفی

با استفاده از تابع `rand(m,n)` می‌توان یک آرایه $m \times n$ از اعداد تصادفی بین صفر و یک ساخت. اگر این دستور بدون آرگومان ورودی استفاده شود یک عدد تصادفی بین صفر و یک تولید می‌شود.

👉 برای تولید یک عدد تصادفی می‌توان تابع را به صورت `rand` نیز بکار برد.

مثال ۱۱.۴. به چگونگی استفاده از دستورات در کد زیر دقت کنید.

ورودی

```
a = rand
A = rand(1,3)
B = rand(2,3)
x = randperm(6)
```

خروجی

```
a =
0.1270
A =
0.9134    0.6324    0.0975
B =
0.2785    0.9575    0.1576
0.5469    0.9649    0.9706
x =
4     2     5     6     3     1
```

👉 در دستور آخر با استفاده از `randperm(n)` جایگشتی برای اعداد طبیعی بین ۱ تا n تولید شده است.

اما اگر بخواهیم اعداد یا آرایه‌های تصادفی با عناصر طبیعی تولید کنیم می‌توان از دستور `randi` استفاده کرد.

تولید اعداد و آرایه‌های طبیعی

دستور `randi` را می‌توان به سه شکل برای تولید اعداد یا آرایه‌هایی با اعداد طبیعی بکار برد.

`randi(max)` یک عدد تصادفی طبیعی بین ۱ تا `max` تولید می‌کند.

`randi(max,n)` یک ماتریس $n \times n$ با عناصر تصادفی طبیعی بین ۱ تا `max` تولید می‌کند.

`randi(max,m,n)` یک ماتریس $m \times n$ با عناصر تصادفی طبیعی بین ۱ تا `max` تولید می‌کند.

دستور `randn(m,n)` یک ماتریس $m \times n$ تولید می‌کند که میانگین عناصر آن صفر و انحراف معیار عناصر آن یک می‌باشد.

مثال ۱۲.۴. به چگونگی استفاده از دستور `randi` به شکل‌های مختلفی که در کدهای زیر آورده شده است توجه کنید.

ورودی

```
a = randi(6)
A = randi(5,3)
B = randi(7,2,3)
D = randn(2,3)
```

خروجی

```
a = 3
A =
3     5     2
2     3     3
3     3     4
B =
5     3     1
3     7     7
D =
-0.2539    -0.0209     2.1778
-1.4286    -0.5607     1.1385
```

در انتهای این بخش اشاره‌ای کوتاه به ذخیره‌سازی رشته‌ها در MATLAB داریم. کاربردهای رشته‌ها را در فصل‌های بعد به تدریج بیان خواهیم کرد.

ذخیره‌سازی رشته‌ها


اگر یک رشته را به صورت

```
Str_Name = 'Some String Here'
```

در متغیر دلخواهی ذخیره کنیم، با آرایه‌ای یک‌بعدی مواجه هستیم که عناصر آن به صورت کاراکتر می‌باشند و اگر از تابع پیش‌ساخته char به صورت

```
Str_Name = char('str 1','str 2',...,'str n')
```

استفاده کنیم، آرایه‌ای دوبعدی تشکیل خواهد شد که تعداد سطرهای آن n و تعداد ستون‌های آن، طول بزرگ‌ترین رشته می‌باشد. دسترسی به عناصر آرایه‌های حاصل مشابه آنچه برای آرایه‌های یک‌بعدی و دوبعدی گفتیم، است.

در هنگام ذخیره‌سازی رشته‌ها به عنوان آرایه‌های یک‌بعدی و دوبعدی، برای دستیابی به آنها باید از روش‌هایی که برای استخراج بخش‌هایی از بردار و ماتریس بیان کردیم استفاده کنید.  برخی از توابع کتابخانه‌ای که برای آرایه‌ها معرفی شدند را می‌توان برای رشته‌ها نیز بکار برد. برای نمونه کد زیر را به همراه خروجی حاصل ببینید.

ورودی

```
str = 'I am Ali Mesforush';
L = length(str)
s1 = sort(str)
```

خروجی

```
L = 18
s1 =
    '    AIMaefhilmorssu'
```

همچنین امکان دسترسی به عناصر آرایه و استخراج بخش‌هایی از عناصر آرایه مشابه سایر آرایه‌ها وجود دارد. در این بخش از بیان مطالب بیشتر در زمینه آرایه‌ها خودداری می‌کنیم و مطالب بیشتر در این مورد را در فصلی جداگانه و به شکل کامل‌تر بیان خواهیم کرد.

مثال ۱۳.۴. به چگونگی تعریف رشته‌های زیر و دسترسی به آنها در دستورات زیر توجه کنید. پیشنهاد می‌شود دستورات را در یک m-فایل بنویسید و با ایجاد تغییرات، آنها را چند بار اجرا کنید و نتایج حاصل را با هم مقایسه کنید.

ورودی

```
S1 = 'My Name is Ali Mesforush'
S1(5)
S1(12:14)
S2 = char('My','Name','is','Ali')
S2(4,1)
S2(4,:)
```

خروجی

```
S1 =
'My Name is Ali'
ans =
'a'
ans =
'Ali'
S2 =
4×4 char array
'My      '
'Name    '
'is      '
'Ali     '
ans =
'A'
ans =
'Ali     '
```

📌 به چگونگی استفاده از دونقطه برای استخراج واژه Ali توجه کنید. مشابه این کار را در بخش استخراج یک سطر کامل در مثال ۹.۳ انجام دادیم.

۴.۴ تمرین

تمرین ۱.۴. مقدار y را با استفاده از حالت عنصر به عنصر عملگرها برای $x \in [-10, 10]$ و برای طول گام‌های ۱، ۲، ۵، ۱۰ حساب کنید.

$$y = \frac{\ln(x^2 + 1) - 3 \sin(x) + 1}{\sqrt[3]{x + \cos x + x^2 + 1}}, \quad y = \sqrt[3]{\tanh(x^2 + 4 \log(x^2 + 1) + 1)}$$

تمرین ۲.۴. مقدار y را با استفاده از حالت عنصر به عنصر عملگرها برای $t = 1, 2, 3, 4, 5, 6, 7, 8$

در تابع زیر حساب کنید.

$$y = \frac{2 \cdot t^{2/3}}{t+1} - \frac{(t+1)^2}{e^{(3t+5)}} + \frac{2}{t+1}, \quad y = \frac{(x-3)(x^2+3)}{x^2+1}$$

تمرین ۳.۴. طول بردار $\mathbf{u} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ از رابطه $|\mathbf{u}| = \sqrt{x^2 + y^2 + z^2}$ محاسبه می‌شود. طول بردار $\mathbf{u} = 12\mathbf{i} + 3/5\mathbf{j} - 7\mathbf{k}$ را به دو روش زیر حساب کنید،

۱. بردار را در MATLAB تعریف کنید و با استفاده از یک عبارت که از عناصر بردار استفاده می‌کند طول بردار را محاسبه کنید.

۲. بردار را در MATLAB تعریف کنید، سپس با استفاده ضرب عنصر به عنصر بردار جدیدی بسازید که عناصر آن مربع عناصر بردار اولیه باشد. سپس با کمک توابع sum و sqrt طول بردار را حساب کنید.

تمرین ۴.۴. بردار یکه متناظر با بردار $\mathbf{u} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ به شکل $\mathbf{u}_n = \frac{x\mathbf{i} + y\mathbf{j} + z\mathbf{k}}{\sqrt{x^2 + y^2 + z^2}}$ می‌باشد. بردار یکهمتناظر با بردار $\mathbf{u} = 12\mathbf{i} + 3/5\mathbf{j} - 7\mathbf{k}$ حساب کنید.

تمرین ۵.۴. بردارهای

$$\mathbf{u} = (5, 2, 7), \quad \mathbf{v} = (21, 5),$$

را در MATLAB تعریف کنید و خروجی دستورات زیر را حدس بزنید و یادداشت کنید. سپس با استفاده از MATLAB دستورات را اجرا کنید و درستی حدس خود را بررسی کنید.

$$\mathbf{v}' * \mathbf{u} \quad \mathbf{v} * \mathbf{u}' \quad \mathbf{v} . * \mathbf{u}$$

تمرین ۶.۴. دو بردار $\mathbf{u} = -3\mathbf{i} + 8\mathbf{j} - 2\mathbf{k}$ و $\mathbf{u} = 6/5\mathbf{i} - 5\mathbf{j} - 4\mathbf{k}$ را در نظر بگیرید. حاصل ضرب داخلی $\mathbf{u} \cdot \mathbf{v}$ را به سه روش زیر بدست آورید.

۱. از تابع کتابخانه‌ای sum و یک عبارت محاسباتی استفاده کنید.

۲. بردار \mathbf{u} را به صورت یک بردار سطری و بردار \mathbf{v} را به صورت یک بردار ستونی تعریف کنید و از ضرب ماتریس‌ها استفاده کنید.

۳. از تابع کتابخانه‌ای dot استفاده کنید.

تمرین ۷.۴. بردار $\mathbf{v} = (2, 4, 6, 8, 10)$ را در نظر بگیرید، سپس بردارهای زیر را بدون نوشتن مستقیم عناصر تولید کنید.

$$۱. \mathbf{a} = \left(\frac{1}{4}, \frac{1}{6}, \frac{1}{8}, \frac{1}{10}\right) \quad ۳. \mathbf{c} = (1, 2, 3, 4, 5)$$

$$۲. \mathbf{b} = \mathbf{b} = \left(\frac{1}{4^2}, \frac{1}{6^2}, \frac{1}{8^2}, \frac{1}{10^2}\right) \quad ۴. \mathbf{d} = (1, 1, 1, 1, 1)$$

تمرین ۸.۴. بردار $\mathbf{v} = (5, 4, 3, 2, 1)$ را در نظر بگیرید، سپس بردارهای زیر را بدون نوشتن مستقیم عناصر تولید کنید.

$$۱. \mathbf{a} = (5^2, 4^2, 3^2, 2^2, 1^2) \quad ۳. \mathbf{c} = (25, 20, 15, 10, 5)$$

$$۲. \mathbf{b} = (5^5, 4^4, 3^3, 2^2, 1^1) \quad ۴. \mathbf{d} = (4, 3, 2, 1, 0)$$

تمرین ۹.۴. بردارهای \mathbf{x} و \mathbf{y} را به شکل

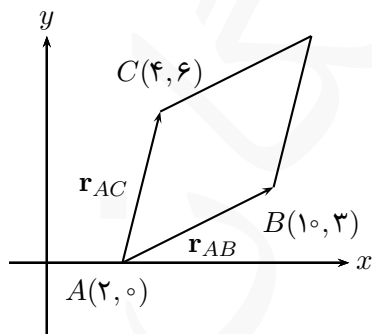
$$\mathbf{x} = (1, 3, 5, 7, 9) \quad \mathbf{y} = (2, 5, 8, 11, 14)$$

تعریف کنید، سپس عبارات محاسباتی زیر را بدست آورید.

$$۱. z = \frac{xy^2}{x+y} \quad ۲. z = x(x^2 - y) - (x - y)^2$$

تمرین ۱۰.۴.

مساحت متوازی‌الاضلاع مقابل را می‌توان به شکل $|\mathbf{r}_{AB} \times \mathbf{r}_{AC}|$ محاسبه کرد. کارهایی که در ادامه خواهد آمد را به صورت دستورات MATLAB در یک فایل -m بنویسید و در آخر با اجرای آن مساحت متوازی‌الاضلاع را محاسبه کنید.



۱. نقاط A ، B و C را به شکل بردارهای زیر تعریف کنید،

$$A = (2, 0), \quad B = (10, 3), \quad C = (4, 6).$$

- با استفاده از نقاط تعریف شده، بردارهای \mathbf{r}_{AB} و \mathbf{r}_{AC} را بدست آورید.
- با استفاده از توابع کتابخانه‌ای `sum`، `cross` و `sqrt` مساحت را بدست آورید.

تمرین ۱۱.۴. بردارهای زیر را تعریف کنید،

$$\mathbf{u} = -2\mathbf{i} + 6\mathbf{j} + 5\mathbf{k}, \quad \mathbf{v} = 5\mathbf{i} - \mathbf{j} + \mathbf{k}, \quad \mathbf{w} = 4\mathbf{i} + 7\mathbf{j} - 2\mathbf{k}.$$

با استفاده از توابع کتابخانه‌ای cross و abs درستی اتحاد زیر را بررسی کنید،

$$\mathbf{u} \times (\mathbf{v} \times \mathbf{w}) = \mathbf{v}(\mathbf{u} \cdot \mathbf{w}) - \mathbf{w}(\mathbf{u} \cdot \mathbf{v})$$

تمرین ۱۲.۴. از ضرب داخلی می‌توان برای تعیین زاویه بین دو بردار استفاده کرد. این زاویه با فرمول $\theta = \cos^{-1} \left(\frac{\mathbf{r}_1 \cdot \mathbf{r}_2}{|\mathbf{r}_1| |\mathbf{r}_2|} \right)$ محاسبه می‌شود. با استفاده از توابع کتابخانه‌ای sqrt، cosd و dot زاویه بین دو بردار

$$\mathbf{r}_1 = 3\mathbf{i} - 2\mathbf{j} + \mathbf{k}, \quad \mathbf{r}_2 = \mathbf{i} + \mathbf{j} - 4\mathbf{k},$$

را محاسبه کنید. توجه کنید که $|\mathbf{r}| = \sqrt{\mathbf{r} \cdot \mathbf{r}}$.

تمرین ۱۳.۴. ثابت کنید، $\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$.

برای این کار، ابتدا بردار x را به صورت $x = (1/5, 1, 0.5, 0.1, 0.01, 0.001, 0.0001, 0.00001)$ تعریف کنید سپس بردار y را به صورت $y = \frac{\sin x}{x}$ بدست آورید و مقادیر y را با یک مقایسه کنید. در محاسباتی که انجام می‌دهید از format long استفاده کنید.

تمرین ۱۴.۴. ثابت کنید، $\lim_{x \rightarrow 1} \frac{x^2 - 1}{x - 1} = 2$.

برای این کار، ابتدا بردار x را به صورت $x = (5, 3, 2, 1.5, 1.1, 1.01, 1.0001, 1.00001)$ تعریف کنید سپس بردار y را به صورت $y = \frac{x^2 - 1}{x - 1}$ بدست آورید و مقادیر y را با ۲ مقایسه کنید. در محاسباتی که انجام می‌دهید از format long استفاده کنید.

تمرین ۱۵.۴. با استفاده از MATLAB نشان دهید که سری نامتناهی $\sum_{n=1}^{\infty} \frac{1}{n^2}$ به عدد یک همگرا می‌باشد. به این منظور سری را برای مقادیر مختلف n محاسبه کنید.

تمرین ۱۶.۴. با استفاده از MATLAB نشان دهید که سری نامتناهی $\sum_{n=1}^{\infty} \frac{(-3)^{-n}}{2n+1}$ به π همگرا می‌باشد. به این منظور سری را برای مقادیر مختلف n محاسبه کنید.

تمرین ۱۷.۴. ماتریس‌های زیر را تولید کنید،

$$A = \begin{bmatrix} 2 & 4 & 0 \\ 3 & 1 & -5 \\ 0 & 1 & 4 \end{bmatrix}, \quad B = \begin{bmatrix} -2 & 5 & 0 \\ -3 & 2 & 7 \\ -1 & 6 & 9 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 3 & 5 \\ 2 & 1 & 0 \\ 4 & 6 & -3 \end{bmatrix},$$

سپس به پرسش‌های زیر پاسخ دهید.

۱. با محاسبه $A + B$ و $B + A$ نشان دهید که جمع ماتریس‌ها خاصیت جابجایی دارد.

۲. با محاسبه $A + (B + C)$ و $(A + B) + C$ نشان دهید که جمع ماتریس‌ها خاصیت شرکت‌پذیری دارد.

۳. با محاسبه $5(A + C)$ و $5A + 5C$ تساوی عبارات را بررسی کنید.

۴. با محاسبه $A(B + C)$ و $AB + AC$ نشان دهید که ضرب ماتریس‌ها خاصیت پخشی دارد.

تمرین ۱۸.۴. با استفاده از ماتریس‌های تمرین قبل به پرسش‌های زیر پاسخ دهید.

۱. آیا رابطه $AB = BA$ برقرار است؟ ۳. آیا رابطه $(AB)^t = B^t A^t$ برقرار است؟

۲. آیا رابطه $A(BC) = (AB)C$ برقرار است؟ ۴. آیا رابطه $(A + B)^t = A^t + B^t$ برقرار است؟

منظور از A^t ترانهاد ماتریس A می‌باشد.

تمرین ۱۹.۴. یک ماتریس 4×4 با عناصر تصادفی طبیعی بین ۱ تا ۱۰ تولید کنید، سپس دستورات زیر را در MATLAB اجرا کنید.

- | | | |
|--------------|----------------------|--------------------|
| ۱. $A * A$ | ۳. $A \setminus A$ | ۵. $\det(A)$ |
| ۲. $A . * A$ | ۴. $A . \setminus A$ | ۶. $\text{inv}(A)$ |

تمرین ۲۰.۴. دستگاه معادلات خطی زیر را حل کنید.

$$\begin{cases} 3x - 2y + 5z = 7/5 \\ -4/5x + 2y + 3z = 5/5 \\ 5x + y - 2/5z = 4/5 \end{cases}$$

نسخه
رایگان

۵ اصول برنامه‌نویسی در MATLAB

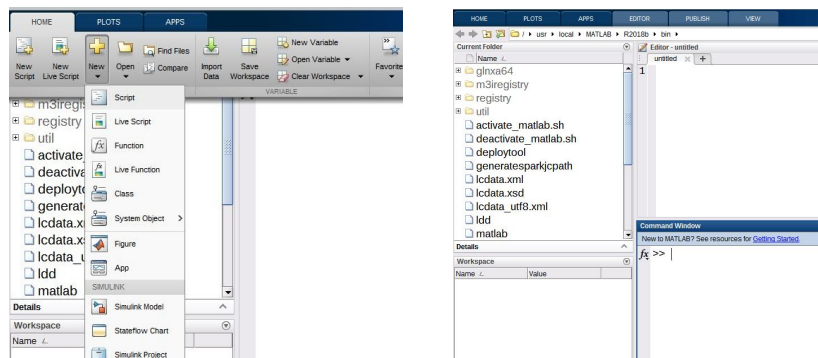
در MATLAB علاوه بر انجام محاسبات با توابع کتابخانه‌ای، این امکان وجود دارد که با استفاده از دستوراتی مشابه زبان‌های برنامه‌نویسی دیگر، مانند زبان C، برنامه‌های تولید کرد. نکته جالب توجه این است که در برنامه‌های نوشته شده در MATLAB می‌توان از تمام توابع کتابخانه‌ای موجود در MATLAB نیز استفاده کرد، به همین دلیل نوشتن برنامه‌هایی که جنبه محاسباتی دارند با MATLAB، نسبت به زبان‌های دیگر برنامه‌نویسی توجیه بیشتری دارد. به‌عنوان مثال اگر در فرآیند حل یک مسأله نیاز به حل دستگاه معادلات خطی باشد، نیازی به نوشتن زیربرنامه جداگانه‌ای برای این کار نیست و می‌توان جواب دستگاه را با استفاده از عملگر تقسیم چپ بدست آورد.

در این فصل به معرفی اصول و قواعد برنامه‌نویسی در MATLAB می‌پردازیم تا خوانندگان کتاب بتوانند در فصل‌های بعدی و در صورت نیاز از این توانمندی MATLAB در حل مسایل خود بهره بگیرند، لذا پیشنهاد می‌شود تا از مطالعه این فصل صرف‌نظر نکنید و با صرف زمان کافی بر مطالبی که بیان خواهد شد تسلط کافی پیدا کنید. بدیهی که خوانندگان گرامی اگر پیش‌زمینه‌ای در یک زبان برنامه‌نویسی داشته باشند کار ساده‌تری پیش‌رو خواهند داشت ولی در این فصل تلاش شده است تا تمام مطالب مورد نیاز در برنامه‌نویسی با MATLAB از پایه بیان شود تا هر کاربری با هر سطح اطلاعات از برنامه‌نویسی بتواند به رفع نیازهای خود از این طریق اقدام کند.

۱.۵ ایجاد، اجرا و ذخیره‌سازی یک m-فایل

برنامه‌نویسی در MATLAB در پنجره فرمان انجام نمی‌گیرد و باید کدهای برنامه در فایل جدیدی نوشته شوند که به m-فایل موسوم است، پس پیش از آغاز برنامه‌نویسی باید یک m-فایل ایجاد

کرد. m- فایل یک فایل متنی است که دارای نامی دلخواه و پسوند m. است. برای تولید یک m- فایل به روشی که در شکل ۱.۵ نشان داده شده است عمل کنید.



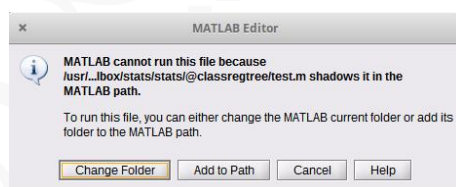
(ب) script یا New Script را انتخاب کنید

(آ) m- فایل ایجاد شده

شکل ۱.۵: روش ایجاد m- فایل در MATLAB

پس از شروع به نوشتن دستورات مورد در نظر در فایل ایجاد شده، m- فایل را باید ذخیره کنید. پسوند پیشنهادی m. می‌باشد که باید پذیرفته شود در این صورت فایل با نامی دلخواه که تابع قواعد نامگذاری متغیرها می‌باشد در محلی ذخیره شود. محل ذخیره‌سازی را نیز کاربر می‌تواند تعیین کند.

با زدن کلید F5 می‌توان m- فایل را اجرا کرد، در این صورت و در اولین اجرا با شکل ۲.۵ برخورد خواهید کرد که باید روی Add to Path کلیک کنید. پس از انجام مراحل که بیان کردیم



شکل ۲.۵: گزینه Add to Path را کلیک کنید.

m- فایل تولید شده آماده استفاده است و می‌توانید برنامه خود را در آن بنویسید و اجرا کنید. یک برنامه نوشته شده در MATLAB در هنگام اجرا کامپایل نمی‌شود، بلکه صورت سطر به سطر اجرا می‌شود، به عبارت بهتر MATLAB مفسر^۱ دارد. لذا توجه داشته باشید که در هنگام

^۱interpreter

اجرای یک m -فایل اگر در یکی از دستورات خطایی وجود داشته باشد، تمامی دستورات قبل از آن دستور اجرا می‌شوند و با رسیدن به دستوری که خطا دارد، اجرای برنامه متوقف شده، خطای رخ داده گزارش خواهد شد.

۲.۵ عبارات محاسباتی و دستورات ورودی و خروجی

تمام مطالبی که درباره عبارات محاسباتی و عملگرهای محاسباتی در بخش ۱.۱ بیان کردیم در هنگام برنامه‌نویسی نیز برقرار است، لذا با هیچ مفهوم جدیدی در این زمینه مواجه نیستیم. همچنین اگر در هنگام برنامه‌نویسی نیاز به آرایه‌ها باشد، می‌توان از تمام مواردی که در بخش ۱.۴ گفته شد نیز استفاده کرد. پس نخستین موضوعی که با آن روبه‌رو خواهیم شد بحث دستورات ورودی و خروجی به برنامه می‌باشد. در MATLAB مانند زبان‌های برنامه‌نویسی دیگر دستوراتی وجود دارد که با استفاده از آنها می‌توان ورودی و خروجی داده‌ها را در یک برنامه کنترل کرد.

دستور ورودی

در MATLAB با استفاده از دستور `input` می‌توان مقادیری را از کاربر دریافت کرد و در متغیرهایی ذخیره نمود. شکل کلی این دستور به‌صورت زیر است،

```
var_name = input('message','s')
```

که در آن

- بجای `message` می‌توان هر متنی قرار داد، این متن در خروجی ظاهر خواهد شد.
- اگر نوع ورودی عددی باشد نیازی به استفاده از 's' نیست.
- اگر داده ورودی از نوع رشته‌ای باشد باید از 's' استفاده کرد.

مثال ۱.۵. در برنامه زیر ضرایب معادله درجه دوم $ax^2 + bx + c = 0$ را از کاربر دریافت کرده، ریشه‌های معادله را محاسبه می‌کند.

ورودی

```
a = input("a: ");
b = input("b: ");
c = input("c: ");
Delta = b^2 - 4*a*c;
x_1 = (-b + sqrt(Delta))/(2*a)
x_2 = (-b - sqrt(Delta))/(2*a)
```

خروجی

```
a: 2
b: 3
c: -5
x_1 = 1
x_2 = -2.5000
```

در این برنامه برای نمایش خروجی برنامه در انتهای عبارات محاسباتی سمی‌کالن قرار ندادیم، لذا محتوای متغیرها در خروجی نمایش داده شده‌اند، ولی دستوراتی وجود دارند که به کمک آنها می‌توان بر خروجی کنترل بیشتری داشت و خروجی را با قالب دلخواه تولید کرد.

دستورات خروجی

در MATLAB برای نمایش اطلاعات در خروجی می‌توان از دو دستور استفاده کرد،

disp این دستور به صورت `disp(var_name)` و `disp('message')` استفاده می‌شود که در این صورت با دستور اول محتوای متغیر و با دستور دوم رشته نوشته شده در خروجی نمایش داده می‌شود.

fprintf این دستور به شکل

```
fprintf('FORMAT',var_names)
```

بکار می‌رود که در آن

- **FORMAT** رشته‌ای است که به کمک آن قالب‌بندی خروجی انجام می‌شود و در این زمینه در ادامه بیشتر خواهیم گفت،
- **var_names** اسامی متغیرهایی است که می‌خواهیم مقادیر آنها در خروجی نمایش داده شود.

مثال ۲.۵. برنامه مثال ۱.۵ را با دستور disp می‌توان به شکل زیر نوشت.

```
a = input("a: "); b = input("b: "); c = input("c: ");
Delta = b^2 - 4*a*c;
x_1 = (-b + sqrt(Delta))/(2*a); disp(x_1)
x_2 = (-b - sqrt(Delta))/(2*a); disp(x_2)
```

نتیجه اجرای این کد به صورت زیر می‌باشد.

```
a: 2
b: 4
c: -5

0.8708
-2.8708
```

- ❏ دستور disp انعطاف زیادی ندارد و نمی‌توان با استفاده از این دستور کنترل مناسبی بر خروجی داشت، ولی اگر از دستور fprintf استفاده کنید می‌توان به طور کامل خروجی را کنترل کرد، ولی برای استفاده از این دستور نیاز به کمی اطلاعات پیرامون انواع داده‌ها در MATLAB داریم که در ادامه به آن خواهیم پرداخت.

نکته عملی

در دستور `fprintf` باید بجای `FORMAT` یک رشته نوشت که شامل برخی کاراکترهای کنترلی است که به کمک آنها می‌توان خروجی را کنترل کرد. در حالت کلی رشته `FORMAT` می‌تواند به شکل زیر باشد،

```
'Some text %ch Some text \ch Some text'
```

که در آن

- بجای `Some text` هر متنی که قرار داده شود در خروجی نمایش داده می‌شود.
- بجای `ch` در `%ch` یکی از مقادیر زیر قرار می‌گیرد. کاربرد هریک در مقابل آن نوشته شده است.

i,d برای نمایش مقادیر صحیح **f** برای نمایش مقادیر اعشاری

u برای نمایش مقادیر صحیح بدون علامت **e** برای نمایش علمی اعداد

o برای نمایش اعداد مبنای هشت **c** برای نمایش یک کاراکتر

x برای نمایش اعداد مبنای شانزده **s** برای نمایش رشته‌ها

- بجای `ch` در `\ch` یکی از مقادیر زیر قرار می‌گیرد.

n برای شکستن سطر **** برای ایجاد Backslash

t برای ایجاد فاصله‌ای به اندازه یک Tab **b** برای ایجاد Backspace

علاوه بر کاراکترهای کنترلی بیان شده، برخی کاراکترهای دیگر نیز وجود دارد که کاربرد کمتری دارند و در صورت نیاز می‌توانید در اینترنت یا در راهنمای MATLAB که در خود نرم‌افزار موجود است به آنها دسترسی پیدا کنید.

کاراکتر کنترلی `%f` را می‌توان به شکل `%m.nf` بکار برد که در این صورت عدد اعشاری با `m` رقم که تعداد ارقام بعد از نقطه اعشار `n` تا است نمایش داده خواهد شد.

مثال ۳.۵. با استفاده از دستورات خروجی بیان شده مثال ۱.۵ را به شکل زیر بازنویسی می‌کنیم،


```

a = input("a: "); b = input("b: "); c = input("c: ");
Delta = b^2 - 4*a*c;
x_1 = (-b + sqrt(Delta))/(2*a);
x_2 = (-b - sqrt(Delta))/(2*a);
fprintf('First root is: %f,\t Second root is: %3.2f\n',x_1,x_2)

```

در صورت اجرای برنامه بالا خروجی به شکل زیر خواهد بود.

```

a: 3
b: 4
c: -3
First root is: 0.535184, Second root is: -1.87

```

📌 در این مثال به چگونگی استفاده از `\t` و `\n` برای ایجاد فاصله در سطر و شکستن سطر توجه کنید.

مثال ۴.۵. برنامه‌ای بنویسید که با داشتن سه راس یک مثلث و با استفاده از فرمول

$$S = \sqrt{p(p-a)(p-b)(p-c)},$$

مساحت مثلث را محاسبه کنید. توجه کنید که a ، b و c اندازه سه ضلع مثلث و p نصف محیط مثلث می‌باشند. در این برنامه مختصات سه راس مثلث باید از کاربر دریافت شود.

ورودی

```

a1 = input('a1: '); a2 = input('a2: ');
b1 = input('b1: '); b2 = input('b2: ');
c1 = input('c1: '); c2 = input('c2: ');
a = sqrt((a1-b1)^2 + (a2-b2)^2);
b = sqrt((a1-c1)^2 + (a2-c2)^2);
c = sqrt((b1-c1)^2 + (b2-c2)^2);
P = (a+b+c)/2;
S = sqrt(P*(P-a)*(P-b)*(P-c));
fprintf('\nArea of triangle is %f\n',S);

```

خروجی

```

a1: 4
a2: -1
b1: 3
b2: 2
c1: 9
c2: 1

Area of triangle
is 8.500000

```

مثال ۵.۵. برنامه‌ای بنویسید که یک عدد طبیعی را دریافت کند و یک دستگاه n معادله با n مجهول را به شکل تصادفی تولید کرده و جواب را محاسبه کنید.

ورودی

```

n = input('Enter n: ');
A = randi(10,n);
b = randi(10,n,1);
x = A\b;
disp('Solution is:')
fprintf('%f\n',x);

```

خروجی

```

Enter n: 5
Solution is:
0.525030
3.000659
-0.230359
1.052635
-2.799042

```

👉 در این برنامه از توابع کتابخانه‌ای استفاده کرده‌ایم. برای مشاهده ماتریس ضرایب می‌توانید سمی‌کالن دستور سطر دوم را حذف کنید.

۳.۵ عملگرهای مقایسه‌ای و دستورات شرطی

با دستوراتی که تا کنون بیان کردیم، کار زیادی نمی‌توان کرد و برنامه‌های زیادی نمی‌توان نوشت. برای نوشتن برنامه‌های کارتر معمولاً نیاز به دستوراتی داریم که بتوانند عمل مقایسه میان دو متغیر انجام دهند. پیش از پرداختن به دستورات شرطی باید نوع دیگری از عملگرها را معرفی کنیم که به عملگرهای مقایسه‌ای موسومند.

عملگرهای مقایسه‌ای

در MATLAB شش عملگر مقایسه‌ای وجود دارد که عبارتند از:

$>$ بزرگتر از	$>=$ بزرگتر یا مساوی با	$==$ مساوی با
$<$ کوچکتر از	$<=$ کوچکتر یا مساوی با	\sim مخالف با

در ارتباط با این شش عملگر به نکات زیر توجه کنید:

- از عملگرهای مقایسه‌ای می‌توان در عبارات ریاضی و به‌عنوان یک عملگر محاسباتی استفاده کرد.
- هنگامی که دو عدد مقایسه می‌شوند، اگر مقایسه درست باشد، مقدار حاصل یک و اگر مقایسه نادرست باشد، مقدار حاصل صفر خواهد بود.
- مقایسه دو آرایه تنها میان دو آرایه هم‌اندازه امکان‌پذیر است، در این صورت عمل مقایسه به‌شکل عنصر به عنصر انجام می‌گیرد و نتیجه کار یک آرایه هم‌اندازه با دو آرایه مورد مقایسه می‌باشد که عناصر آن فقط صفر و یک می‌باشند.
- اگر یک کمیت اسکالری با یک آرایه مقایسه شود، آن کمیت با تک تک عناصر آرایه مقایسه می‌شود و با توجه به نتیجه مقایسه، یک آرایه شامل صفر و یک برگشت داده می‌شود.

برای درک بهتر مفهوم عملگرهای مقایسه‌ای در مثال بعدی تعدادی مقایسه انجام داده‌ایم.

مثال ۶.۵. به چگونگی استفاده از عملگرهای مقایسه در دستورات زیر دقت کنید. پیشنهاد می‌شود

دستورات را در یک m-فایل وارد کنید و آنها را اجرا کنید، سپس در دستورات تغییراتی ایجاد کرده و چندین بار آنها را اجرا کنید تا کاملاً با مفهوم عملگرهای مقایسه‌ای آشنا شوید.

ورودی	خروجی
5 > 10	ans = 0
x = 7 <= 11	x = 1
y = (2<3)+(12/4 == 3)+(3*5 ~= 15)	y = 2
a = [1,2,3]; b = [5,2,7];	ans =
a == b	0 1 0
3 ~= a	ans =
A = [-1,2;3,-4];	1 1 0
A > 0	ans =
	0 1
	1 0

انواع دیگری از عملگرها موجود می‌باشند که به عملگرهای منطقی موسومند. با استفاده از این دسته از عملگرها، می‌توان چند عمل مقایسه را به‌طور همزمان انجام داد.

عملگرهای منطقی

به‌طور کلی در MATLAB سه عملگر منطقی وجود دارد:

AND که به‌صورت A&B مورد استفاده قرار می‌گیرد و زمانی که هر دو عملوند A و B مقدار درست داشته باشند، مقدار آن یک و در غیر این صورت مقدار آن صفر می‌باشد.

OR که به‌صورت A|B مورد استفاده قرار می‌گیرد و زمانی که هر دو عملوند A و B مقدار نادرست داشته باشند، مقدار آن صفر و در غیر این صورت مقدار آن یک می‌باشد.

NOT که به‌صورت ~A مورد استفاده قرار می‌گیرد و اگر A مقدار نادرست داشته باشند، مقدار آن یک و در غیر این صورت مقدار آن صفر می‌باشد.

❏ در عملگرهای مقایسه‌ای A و B می‌توانند مقادیر عددی داشته باشند، در این صورت توجه کنید که تمام اعداد ناصفر از نظر منطقی ارزش درست و عدد صفر ارزش نادرست دارد.

مثال ۷.۵. به روش استفاده از عملگرهای منطقی در دستورات زیر توجه کنید.

ورودی	خروجی
3 & 5	ans = 1
3 & 0	ans = 0
0 0	ans = 0
3 2	ans = 1
~7	ans = 0
x = [1,-1,4,0]; y = [2 -1,3,0];	ans =
x & y	1 1 1 0
x y	ans =
~y	1 1 1 0
	ans =
	0 0 0 1

❏ معمولاً از عملگرهای منطقی و عملگرهای مقایسه‌ای با هم و در دستورات شرطی استفاده می‌شود که در ادامه به آنها می‌پردازیم.

❏ از عملگرهای منطقی بیان شده می‌توان به شکل $\text{and}(A,B)$ ، $\text{or}(A,B)$ و $\text{not}(A)$ نیز استفاده کرد.

در MATLAB علاوه بر سه عملگر منطقی که بیان کردیم چند تابع کتابخانه‌ای نیز وجود دارد که مشابه عملگرهای منطقی کار می‌کنند. در جدول ۱.۵ این توابع به همراه مثالی از روش استفاده از آنها آورده شده است.

۴.۵ دستورات شرطی

معمولاً در هنگام برنامه‌نویسی نیاز به بررسی شرطها و انجام دستورات در صورت برقرار آنها می‌باشد، به این منظور در MATLAB مانند زبان‌های برنامه‌نویسی دیگر دستوراتی به این منظور وجود دارد. در این بخش به معرفی دستورات شرطی می‌پردازیم.

جدول ۱.۵: توابع منطقی در MATLAB

تابع	عمل	مثال
<code>xor(a,b)</code>	اگر یکی از عملوندها درست و دیگری نادرست باشد مقدار یک برگشت داده می‌شود.	<pre>>> xor(5,0) ans = 1 >> xor(5,4) ans = 0</pre>
<code>all(A)</code>	اگر همه عناصر بردار A ناصفر باشند، مقدار یک برگشت داده می‌شود و اگر یک عنصر یا بیشتر صفر باشد، مقدار صفر برگشت داده می‌شود.	<pre>>> A=[2,1,4]; >> all(A) ans = 1 >> B=[2,0,3]; >> all(B) ans = 0</pre>
<code>any(A)</code>	اگر حداقل یک عنصر بردار A ناصفر باشد، مقدار یک برگشت داده می‌شود و اگر همه عناصر بردار صفر باشند، مقدار صفر برگشت داده می‌شود.	<pre>>> A = [2,0,1]; >> any(A) ans = 1 >> B = [0,0,0]; >> any(B) ans = 0</pre>
<code>find(cond)</code>	اندیس‌های عناصری را برمی‌گرداند که شرط نوشته شده برای آنها برقرار باشد، خروجی این دستور یک بردار است. اگر تنها نام بردار نوشته شود، اندیس‌های عناصر ناصفر برگشت داده می‌شود.	<pre>>> A=[2,3,0,10,0,6]; >> find(A) ans = 1 2 4 5 6 >> find(A>4) ans = 4 6</pre>

۱.۴.۵ ساختار if-end

در MATLAB چند ساختار مختلف برای دستورات شرطی وجود دارد، ساده‌ترین نوع ساختارهای شرطی، ساختار if-end می‌باشد. دو واژه if و end از واژه‌های کلیدی در MATLAB هستند، توجه داشته باشید که از این دو واژه نمی‌توان به‌عنوان اسمی متغیرها استفاده کرد.

ساختار if-end

ساده‌ترین دستور شرطی ساختار if-end است که به شکل زیر می‌باشد.

MATLAB CODE	در این ساختار پس از رسیدن کنترل برنامه به شرط
if cond	cond، درستی آن بررسی می‌شود و در صورت درست
.	بودن شرط، ابتدا دستورات بین if و end اجرا
.	می‌شوند و سپس دستورات پس از end اجرا خواهند
end	شد، ولی اگر شرط نادرست باشد از اجرای دستورات
MATLAB CODE	بین if و end صرف‌نظر می‌شود و دستورات پس از
	end اجرا خواهند شد.

مثال ۸.۵. فرض کنید به کارمندی بابت ۴۰ ساعت کار در هفته، ساعتی x تومان پرداخت می‌شود ولی برای هر ساعت کار بیشتر از ساعات موظف ۵۰ درصد بیشتر از دستمزد معمولی پرداخت می‌شود. برنامه‌ای بنویسید که نام کارمند، تعداد ساعت کار او در هفته و دستمزد پایه او را دریافت کند و دستمزد پرداختی به او را مشخص نماید.

```
Name = input('Employee name: ');
H = input('How many hourse: ');
salary_per_hour = input('Base salary: ');
pay = 40*salary_per_hour;
if (H > 40)
pay = pay + (H-40)*salary_per_hour*1.5;
end
fprintf('%s will recieve %5.2f\n',Name,pay);
```

حاصل اجرای برنامه به صورت زیر می‌باشد:

```
Employee name: 'Ali'
```

```
How many hourse: 50
Base salary:5
Ali will recieve 275.00
```

👉 در هنگام اجرای این برنامه باید نام را در میان یک جفت کوتیشن قرار داد، در غیر این صورت با خطا مواجه خواهید شد.

حال فرض کنید می‌خواهیم برنامه‌ای بنویسیم که عددی از کاربر دریافت کند و اگر عدد از ۵ بیشتر باشد با پیام مناسب در خروجی چاپ شود و اگر از ۵ کمتر بود نیز با پیام مناسب در خروجی نمایش داده شود.

کد زیر برای مقادیر کمتر از ۵ به درستی کار می‌کند، ولی اگر مقدار ورودی از ۵ بیشتر باشد، به دلیل ساختار دستور if-end خروجی درست نخواهد بود. برای رفع این مشکل به ساختار دیگری برای if نیاز داریم.

ورودی

```
x = input('Enter a number: ')
if x > 5
fprintf('%i is greater than 5\n',x)
end
fprintf('%i is less than 5\n',x)
```

خروجی

```
Enter a number: 7
7 is greater than 5
7 is less than 5
```

۲.۴.۵ ساختار if-else-end

نوع دیگری از دستورات شرطی که می‌تواند مشکل پیش آمده در بالا را به سادگی حل کند، ساختار if-else-end می‌باشد. در این دستور هر سه واژه if، else و end از واژه‌های کلیدی در MATLAB می‌باشند، توجه داشته باشید که از این سه واژه نمی‌توان به‌عنوان اسامی متغیرها استفاده کرد.

ساختار if-else-end

نوع دیگر دستورات شرطی ساختار if-else-end است که به شکل زیر می باشد.

<p>MATLAB CODE</p> <pre>if cond ... else ... end MATLAB CODE</pre>	<p>در این ساختار پس از رسیدن کنترل برنامه به شرط cond، درستی آن بررسی می شود و در صورت درست بودن شرط، ابتدا دستورات بین if و else اجرا می شوند و سپس دستورات پس از end اجرا خواهند شد، ولی اگر شرط نادرست باشد، ابتدا دستورات بین else و end اجرا می شود، سپس دستورات پس از end اجرا خواهند شد.</p>
--	---

این ساختار نسبت به ساختار if-end توانمندی بیشتری دارد، لذا برنامه نویسان بیشتر استفاده از این ساختار را بر ساختار پیشین ترجیح می دهند.

به عنوان نمونه مشکلی که در انتهای بخش ۱۰.۴.۵ مطرح شد با استفاده از این ساختار به شکل زیر حل می شود.

ورودی

```
x = input('Enter a number: ')
if x > 5
fprintf('%i is greater than 5\n',x)
else
fprintf('%i is less than 5\n',x)
end
```

خروجی

```
Enter a number: 7
7 is greater than 5
```

مثال ۹.۵. برنامه حل معادله درجه دوم $ax^2 + bx + c = 0$ را به گونه تغییر دهید که تنها ریشه های حقیقی معادله را محاسبه کند و در صورت داشتن ریشه های موهومی پیغام مناسبی نمایش دهد.

```
a = input("a: "); b = input("b: "); c = input("c: ");
```

```

Delta = b^2 - 4*a*c;
if Delta >= 0
x_1 = (-b + sqrt(Delta))/(2*a);
x_2 = (-b - sqrt(Delta))/(2*a);
fprintf('First root is: %f,\t Second root is: %3.2f\n',x_1,x_2)
else
fprintf('Equation has not real roots\n');
end

```

خروجی برای $x^2 + 4x - 3 = 0$ و $x^2 + 2x + 3 = 0$ به صورت زیر می‌باشد،

```

First root is: 0.581139, Second root is: -2.58
Equation has not real roots

```

ساختار if-elseif-else-end

if cond1	در این ساختار پس از رسیدن کنترل برنامه به شرط
.	cond1، درستی آن بررسی می‌شود و در صورت درست
elseif cond2	بودن شرط، ابتدا دستورات بین if و elseif اجرا
.	می‌شوند و سپس دستورات پس از end اجرا خواهند
elseif cond3	شد، ولی اگر شرط نادرست باشد، شرط cond2 بررسی
.	می‌شود و در صورت درست بودن شرط، دستورات پس
else	از elseif اجرا می‌شوند و به همین ترتیب اجرا ادامه
.	می‌یابد. اگر تمامی شرط‌ها نادرست باشند، دستورات
end	پس از else اجرا خواهند شد.

👉 با وجود دستوری که در بخش ۳.۴.۵ معرفی خواهیم کرد خیلی به این ساختاری نیازی نخواهیم داشت.

۳.۴.۵ ساختار switch-case

هرگاه تعداد زیادی مقایسه بین یک متغیر با مقادیر مختلف وجود داشته باشد، علاوه بر ساختار if-elseif-else-end می‌توان از ساختار switch-case نیز استفاده کرد.

ساختار switch-case

ساختار switch-case به شکل زیر می‌باشد.

<pre>switch variable case value 1 . case value 2 . case value n . otherwise . end</pre>	<p>در این ساختار پس از رسیدن کنترل برنامه به واژه switch برابری variable با value1، بررسی می‌شود و در صورت برابری، دستورات پس از آن اجرا می‌شوند و در صورت برابر نبودن، حالت بعد بررسی می‌شود. همین روال تا انتها ادامه پیدا می‌کند و اگر برابری در هیچ یک از حالات رخ ندهد، دستورات پس از otherwise اجرا خواهند شد.</p>
---	--

مثال ۱۰.۵. برنامه زیر یک ماشین حساب شبیه‌سازی می‌کند که چهار عمل اصلی را انجام می‌دهد. خروجی نمایش داده شده نتیجه حاصل از چند اجرای متفاوت می‌باشد.

در برنامه به استفاده از تابع error برای چاپ پیغام مناسب در حالت otherwise برای توقف برنامه توجه کنید. اگر بجای این دستور از fprintf استفاده می‌شد، پس از اجرای دستور بخش otherwise دستور پس از end اجرا می‌شد. همچنین توجه کنید که در دستور آخر، fprintf، از سه نقطه برای شکستن دستور به دو سطر استفاده شده است.

ورودی

```

a = input('a: ');
b = input('b: ');
op = input('op: ');
switch op
case '+'
c = a + b;
case '-'
c = a - b;
case '*'
c = a * b;
case '/'
c = a / b;
otherwise
error('Error in data...');
end
fprintf('%3.2f %c %3.2f ...
= %3.2f\n', a, op, b, c);

```

خروجی

```

a: 9
b: 4
op: '/'
9.00 / 4.00 = 2.25
a: 7
b: 3
op: '-'
7.00 - 3.00 = 4.00
a: 8
b: 3
op: '*'
8.00 * 3.00 = 24.00
a: 4
b: 5
op: '#'
Error in data...

```

با توجه به اینکه در ادامه کتاب و در مثال‌های که آورده می‌شود همواره از دستورات شرطی به شکل‌های مختلف استفاده خواهد شد، در این بخش مثال دیگری نخواهیم زد.

۵.۵ حلقه‌های تکرار

در هنگام برنامه‌نویسی با هر زبانی، معمولاً نیاز به انجام برخی کارها به صورت تکراری می‌باشد، به این منظور در تمام زبان‌های برنامه‌نویسی، راهکارهایی برای انجام عملیات تکراری تدارک دیده شده است. در MATLAB دو ساختار به این منظور وجود دارد که در این بخش به معرفی آنها می‌پردازیم.

۱.۵.۵ حلقه for-end

اولین حلقه تکراری که به آن می‌پردازیم، حلقه for می‌باشد. این حلقه زمانی مورد استفاده قرار که تعداد تکرار دستورات داخل حلقه از ابتدا مشخص باشد.

ساختار حلقه for

این حلقه زمانی استفاده می‌شود که تعداد تکرار مورد نیاز از ابتدا و به صورت دقیق مشخص باشد. ساختار کلی حلقه for به شکل زیر می‌باشد.

<pre>for k = first:step:last end</pre>	<p>در این حلقه، به شمارنده k، اولین مقدار، یعنی $first$، داده می‌شود و دستورات داخل حلقه یک بار اجرا می‌شوند، سپس به اندازه طول گام، یعنی $step$، به شمارنده افزوده می‌شود و اگر مقدار حاصل از مقدار نهایی کمتر باشد، دستورات داخل حلقه یک بار دیگر اجرا می‌شود. این روند تا زمانی که مقدار شمارنده از مقدار نهایی، یعنی end، بیشتر نشده است ادامه می‌یابد.</p>
--	---

- در حلقه for مقادیر $first$ ، $step$ و $last$ می‌توانند مثبت یا منفی باشند.
- اگر $step$ نوشته نشود، مقدار طول گام یک در نظر گرفته خواهد شد.
- در حالت خاص حلقه for را می‌توان به شکل $for\ k = A$ نیز بکار برد که در آن A یک بردار می‌باشد. در این صورت به k عناصر بردار A از اولین عنصر تا آخرین عنصر داده می‌شود.

در ادامه این بخش به چند مثال از کاربرد حلقه for در برنامه‌نویسی می‌پردازیم، ولی توجه کنید که در MATLAB و با استفاده از دستورات آن گاهی اوقات می‌توان بدون استفاده از حلقه‌های تکرار نیز نتایج مشابهی گرفت. در صورتی که این کار ممکن باشد در انتهای حل مثال، راه حل بدون استفاده از حلقه نیز بیان خواهد شد.

مثال ۱۱.۵. برنامه‌ای بنویسید که همگرایی سری $\sum_{n=0}^{\infty} \frac{x^n}{n!}$ را بررسی کند. کد زیر را برای $x = 1$ و دو مقدار $n = 5, 20$ اجرا کرده‌ایم.

ورودی	خروجی
<pre>x = input('x: '); n = input('n: '); sum = 1; for k = 1:n sum = sum + x^k/factorial(k); end fprintf('Sum = %10.9f\n',sum)</pre>	<pre>x: 1 n: 5 Sum = 2.716666667 x: 1 n: 20 Sum = 2.718281828</pre>

👉 بدون استفاده از حلقه for و بادستور

```
sum((1.^[0:20])./(factorial([0:20])))
```

نیز می‌توان همین نتیجه را بدست آورد.

مثال ۱۲.۵. بردار زیر را در نظر بگیرید،

$$V = [-5, 7, 10, 13, -10, 15, 9, 3, -21, 35, 21, 40, 23, 60, -1, -13]$$

برنامه‌ای بنویسید که

- بجای تمام اعداد منفی صفر قرار دهد.
- تمام اعداد بخش‌پذیر بر ۳ و ۵ را دو برابر کند.
- تمام اعداد بخش‌پذیر بر ۷ را به توان دو برساند.

برای نوشتن این برنامه از حلقه for و دستور if استفاده کرده‌ایم.

```
V = [-5,7,10,13,-10,15,9,3,-21,35,21,40,23,60,-1,-13]
n = length(V);
```

```

for k = 1:n
    if V(k) < 0
        V(k) = 0;
    elseif (rem(V(k),5) == 0) & (rem(V(k),3) == 0)
        V(k) = 2*V(k);
    elseif rem(V(k),7) == 0
        V(k) = V(k)^2;
    end
end
V

```

خروجی حاصل از این برنامه به شکل زیر است،

```

V =
Columns 1 through 12
-5    7   10   13  -10   15    9    3  -21   35   21   40
Columns 13 through 16
23    60    -1   -13
V =
Columns 1 through 6
0         49         10         13         0         30
Columns 7 through 12
9         3         0        1225        441        40
Columns 13 through 16
23         120         0         0

```

پیشنهاد می شود این کد را در یک m-فایل بنویسید و اجرا کنید. 📄

۲.۵.۵ حلقه while-end

نوع دیگری از حلقه‌ها در MATLAB وجود دارد که در آن تعداد تکرار دستورات آن مشخص نیست و بسته به شرایط می‌تواند تغییر کند. این نوع حلقه‌ها به حلقه while موسومند.

ساختار حلقه while

این حلقه زمانی مورد استفاده قرار می‌گیرد که تعداد تکرار حلقه مشخص نباشد. ساختار کلی حلقه while به شکل زیر است.

<pre>while condition end</pre>	<p>در این حلقه، ابتدا شرط بررسی می‌شود، در صورت درست بودن شرط، دستورات داخل حلقه اجرا می‌شوند. با رسیدن به end دوباره شرط حلقه بررسی می‌شود و همین روند تا زمانی که شرط برقرار نباشد ادامه پیدا می‌کند.</p>
--	---

- باید حتما در داخل بدنه حلقه تغییراتی رخ دهد تا در نهایت شرط حلقه ار درست به نادرست تغییر پیدا کند و حلقه خاتمه پیدا کند.
- اگر شرط حلقه همیشه درست باشد، حلقه هرگز پایان نمی‌یابد.
- اگر شرط حلقه در اولین بررسی نادرست باشد، حلقه هرگز تکرار نخواهد شد.

مثال ۱۳.۵. با توجه به رابطه زیر مقدار $\sin \frac{\pi}{3}$ را با دقت 10^{-4} حساب کنید.

$$\sin x = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{(2k+1)!}$$

```
x = pi/3;
S = 0;
k = 0;
while abs(sin(x) - S) > 0.0001
```



```

S = S + (-1)^k * x^(2*k+1)/factorial(2*k+1);
k = k + 1;
end
fprintf('S( %f) = %10.9f\nsin(%f) = %10.9f\n',x,S,x,sin(x));

```

خروجی برنامه به صورت زیر می باشد،

```

S( 1.047198) = 0.866021272
sin(1.047198) = 0.866025404

```

نکته عملی

از حلقه ها می توان به صورت تودرتو نیز استفاده کرد، به این معنا که می توان یک حلقه را در داخل حلقه دیگر بکار برد. در این صورت توجه داشته باشید که زمان اجرا ممکن است افزایش زیادی داشته باشد، لذا تا جایی که ممکن است باید از استفاده از چندین حلقه به شکل تودرتو پرهیز کرد.

مثال ۱۴.۵. برنامه ای بنویسید که تمام اعداد قیثاغورثی بین ۱ تا ۲۰ را پیدا کند.

ورودی

```

for i = 1:10
for j = 1:10
for k = 1:10
if (i^2+j^2 == k^2)|...
(i^2+k^2 == j^2)|...
(j^2+k^2 == i^2)
fprintf('%i, %i, %i\n',i,j,k)
end
end
end
end

```

خروجی

```

3, 4, 5
3, 5, 4
4, 3, 5
4, 5, 3
5, 3, 4
5, 4, 3
6, 8, 10
6, 10, 8
8, 6, 10
8, 10, 6
10, 6, 8
10, 8, 6

```

مثال ۱۵.۵. فرض کنید کاربر عددی بین یک تا ۱۰۰۰ انتخاب کرده است. برنامه‌ای بنویسید که این عدد را در کمترین تعداد حدس پیدا کند. در این برنامه از یک دستور جدید استفاده شده است که پس از مثال توضیح داده می‌شود. این برنامه در محیط MATLAB وارد و اجرا کنید.

```
clc
U = 1000; L = 0;
Guess = round((U+L)/2);
No_Guess = 1;
while true
    fprintf('Is your number (=), (>) or (<) of %i',Guess);
    op = input(': ','s');
    switch op
    case '='
        fprintf('I found your number in %i Guess, it is %i\n',...
            No_Guess,Guess);
        break
    case '>'
        L = Guess;
        Guess = round((U+L)/2);
        No_Guess = No_Guess + 1;
    case '<'
        U = Guess;
        Guess = round((U+L)/2);
        No_Guess = No_Guess + 1;
    otherwise
        fprintf('Error in data...\n');
        fprintf('Is your number (=), (>) or (<) of %i',Guess);
        op = input(': ','s');
    end
end
```

end

با فرض اینکه عدد ۲۹۳ را انتخاب کرده باشیم، نتیجه اجرای این برنامه به صورت زیر می باشد. توجه کنید برای وارد کردن کاراکترها نیازی به کوتیشن نیست.

```
Is your number (=), (>) or (<) of 500: <
Is your number (=), (>) or (<) of 250: >
Is your number (=), (>) or (<) of 375: <
Is your number (=), (>) or (<) of 313: <
Is your number (=), (>) or (<) of 282: >
Is your number (=), (>) or (<) of 298: <
Is your number (=), (>) or (<) of 290: >
Is your number (=), (>) or (<) of 294: <
Is your number (=), (>) or (<) of 292: >
Is your number (=), (>) or (<) of 293: =
I found your number in 10 Guess, it is 293
```

👉 در این برنامه به عنوان شرط حلقه while عبارت true قرار داده شده است، لذا این حلقه دارای شرط همیشه درست است و یک حلقه بی پایان محسوب می شود.

👉 به چگونگی استفاده از دستور input در دریافت یک کاراکتر از کاربر توجه کنید.

👉 به دلیل بی پایان بودن حلقه مورد استفاده، در این برنامه از دستور break برای شکستن حلقه while استفاده شده است.

دستورات break و continue

در MATLAB دو دستور برای حلقه‌ها وجود دارد که در مواقع خاصی کاربرد دارند.

break هنگام اجرای برنامه و با رسیدن به دستور break، حلقه تکرار در هر مرحله‌ای از اجرا که باشد، شکسته می‌شود و اجرای برنامه به پس از end منتقل می‌شود و اجرای حلقه به‌طور کامل پایان می‌پذیرد.

continue هنگام اجرای برنامه و با رسیدن به دستور continue، دستورات پس از آن اجرا نخواهد شد و اجرا برنامه end حلقه منتقل می‌شود. در واقع این دستور باعث پایان یافتن تکرار فعلی و آغاز تکرار بعدی حلقه می‌باشد.

برنامه‌هایی که در این بخش نوشته شدند بیشتر جنبه آموزشی دارند ولی شایان توجه است که با بیان دستورات بیشتر از MATLAB که در فصل‌های بعدی بیان می‌شوند، شکل برنامه‌نویسی در MATLAB تغییر خواهد کرد و عموماً برنامه‌هایی که نوشته می‌شود به‌گونه‌ای خواهد بود در آنها از دستورات و توابع کتابخانه‌ای MATLAB بیشتر استفاده خواهد شد. لذا پیشنهاد می‌شود در هنگام برنامه‌نویسی تلاش کنید تا از توابع کتابخانه‌ای بیشتر استفاده کنید و از نوشتن برنامه‌های تکراری پرهیز نمایید. این بخش را با یک مثال دیگر از حلقه‌های تودرتو به پایان می‌بریم.

مثال ۱۶.۵. برنامه‌ای بنویسید که تمام اعداد اول بین دو تا N را پیدا کند. در این برنامه از حلقه‌های تودرتو استفاده شده است. همچنین به چگونگی استفاده از متغیر flag، دستور break و دستور rem توجه کنید.

ورودی

```

N = input('N: ');
for k = 2:N
    flag = 0;
    for p = 2:ceil(sqrt(k))
        if rem(k,p) == 0
            flag = 1;
            break
        end
    end
    if flag == 0
        fprintf(' %i ',k);
    end
end
fprintf('\n');

```

خروجی

```

N: 20
3  5  7  11  13  17  19

```

۶.۵ توابع غیر کتابخانه‌ای

در هنگام برنامه‌نویسی در MATLAB گاهی نیاز به انجام چندباره برخی محاسبات می‌شود که هیچ تابع کتابخانه‌ای برای آن وجود ندارد، در این صورت این امکان وجود دارد تا کاربران توابعی تعریف کنند و از آنها در نوشتن برنامه‌های خود کمک بگیرند. در این فصل به چگونگی ایجاد توابع و استفاده از آنها در برنامه‌ها می‌پردازیم.

عملکرد توابع معمولاً به گونه‌ای است که داده‌هایی را از برنامه اصلی، یا زیربرنامه‌های دیگر، دریافت می‌کنند و پس از انجام پردازش روی آنها، مقادیری را به برنامه اصلی بر می‌گردانند.

داده‌های ورودی →

تابع

→ مقادیر خروجی

📌 در برخی توابع کتابخانه‌ای شاید هیچ ورودی به تابع ارسال نشود، یا هیچ مقداری به برنامه اصلی برگشت داده نشود.

۱.۶.۵ ایجاد و استفاده از تابع

برای ایجاد تابع نیاز به انجام کار جدیدی نیست و با اطلاعاتی که تا کنون در زمینه m-فایل‌ها داریم می‌توان تابع را تعریف کرد. در واقع تابع، یک m-فایل است که دارای بخش‌های زیر است،

۱. سطر تعریف تابع

۲. بخش معرفی کاری که تابع انجام می‌دهد، این بخش اختیاری است.

۳. بدنه تابع

۴. مقداردهی به مقادیر برگشتی

👉 پس از معرفی تابع، باید فایل را با نام مشخصی ذخیره کرد که در ادامه بیان خواهیم کرد، لذا از ذخیره‌سازی تابع با نام دلخواه خودداری کنید.

سطر تعریف تابع

سطر تعریف تابع به شکل کلی زیر می‌باشد:

```
function [output argument] = function_name(input argument)
```

که در آن

function تعیین کننده تابع بودن فایل می‌باشد و به همین شکل باید نوشته شود.

output argument اسامی متغیرهایی هستند که مقادیر برگشتی از تابع در آنها ذخیره خواهند شد. اگر تنها یک متغیر برگشت داده شود، نیازی به کروشه‌ها نیست.

function_name نام دلخواهی است که تابع قواعد نامگذاری متغیرهاست ولی بهتر است نام بر اساس کاری که تابع انجام می‌دهد انتخاب شود.

input argument اسامی متغیرهایی هستند که از برنامه اصلی به زیربرنامه ارسال می‌شوند.

👉 پس از نوشتن سطر تعریف تابع باید فایل را با نام **function_name**، یعنی همان نامی که در سطر تعریف تابع انتخاب کرده‌ایم ذخیره کنیم. توجه کنید که در هنگام ذخیره فایل، نام پیشنهادی برای فایل همان **function_name** می‌باشد، لذا همین نام را بپذیرید.

پس از سطر تعریف تابع، پیشنهاد می‌شود در قالب کامنت، توضیحاتی پیرامون کاری که تابع انجام می‌دهد بنویسید. شایان ذکر است که این بخش اختیاری است. برای نوشتن کامنت در یک فایل کافیست ابتدا علامت % را قرار دهید و به دنبال آن هر آنچه می‌خواهید بنویسید. در این صورت تمام نوشته‌های پس از % اجرا نخواهند شد.

بدنه تابع

منظور از بدنه تابع، همان دستورات MATLAB و کدهایی است که پیشتر دیدیم و در قالب m-فایل‌ها استفاده کردیم. این بخش می‌تواند شامل تمامی دستورات MATLAB و دستورات برنامه‌نویسی مانند حلقه‌های تکرار، دستورات شرطی و موارد دیگری باشد که تاکنون استفاده کرده‌ایم.

پس از نوشتن برنامه و به عنوان آخرین سطر یک تابع باید از واژه کلیدی end استفاده کرد.

مثال ۱۷.۵. تابعی بنویسید که ضرایب معادله درجه دوم $ax^2 + bx + c = 0$ را به عنوان ورودی دریافت کند و دو ریشه را محاسبه نماید.

```
function [x1,x2] = rishe(a,b,c)
Delta = b^2 - 4*a*c;
x1 = (-b + sqrt(Delta))/(2*a);
x2 = (-b - sqrt(Delta))/(2*a);
end
```

کد بالا باید در فایلی به نام rishe.m ذخیره شود.

توجه کنید که در صورت اجرای این فایل، با خطا مواجه خواهید شد، پس یک تابع به تنهایی قابل اجرا نیست. برای این مثال، برای اجرای تابع در پنجره فرمان باید به شکل زیر عمل کنید.

```
>> [x1,x2] = rishe(1,3,-4)
x1 = 1
x2 = -4
```

در ادامه توضیحات کامل‌تری پیرامون روش‌های اجرای تابع بیان شده است. همچنین مثال‌های بیشتری در زمینه استفاده از توابع در برنامه‌های اصلی یا توابع دیگر آورده شده است.

اجرای توابع

یک تابع به تنهایی قابل اجرا نیست و نمی‌توان مشابه آنچه تاکنون برای `m`-فایل‌ها انجام دادیم عمل کنیم، بلکه برای اجرای توابع باید آنها را فراخوانی کرد که به دو روش این امکان وجود دارد:

- فراخوانی در پنجره فرمان که به صورت زیر در پنجره فرمان قابل انجام می‌باشد،

```
>>[output arguments] = function_name(input arguments)
```

- فراخوانی در برنامه اصلی یا یک تابع دیگر، در این صورت نیازی به ذخیره‌سازی تابع به شکل جداگانه نیست و می‌توان در یک `m`-فایل، ابتدا برنامه اصلی را نوشت و در انتهای برنامه اصلی تابع را قرار داد.

مثال ۱۸.۵. در برنامه زیر، تابع `rishe` در یک برنامه فراخوانی شده است. پیشنهاد می‌شود که زیر را در یک `m`-فایل بنویسید و با نام دلخواهی ذخیره کرده و آن را اجرا کنید.

```
a = input('a: '); b = input('b: '); c = input('c: ');
[x1,x2] = rishe(a,b,c);
fprintf('x1 = %f\n',x1);
fprintf('x2 = %f\n',x2);
```

```
function [x1,x2] = rishe(a,b,c)
Delta = b^2 - 4*a*c;
x1 = (-b + sqrt(Delta))/(2*a);
x2 = (-b - sqrt(Delta))/(2*a);
end
```

این برنامه قابل اجراست و می‌توانید با زدن `F5` آن را اجرا کنید. نتیجه اجرا به شکل زیر می‌باشد.


```

a: 2
b: 5
c: 3
x1 = -1.000000
x2 = -1.500000

```

مثال ۱۹.۵. برنامه‌ای بنویسید که تعداد درس، نمرات و تعداد واحد هر درس را به شکل یک ماتریس $n \times 2$ بخواند و معدل را محاسبه کند.

```

N = input('Number of courses: ');
Nomreh_Vahed = zeros(N,2);
for k = 1:N
    Nomreh_Vahed(k,1) = input('Nomreh: ');
    Nomreh_Vahed(k,2) = input('Vahed: ');
end
Avg = Average(Nomreh_Vahed,N);
fprintf('Your Average for %i courses is: %f\n',N,Avg);

```

```

function Avg = Average(Nomreh_Vahed,N)
%This function calculate the average of student
sum = 0;
Sum_Vahed = 0;
for k = 1:N
    sum = sum + Nomreh_Vahed(k,1)*Nomreh_Vahed(k,2);
    Sum_Vahed = Sum_Vahed + Nomreh_Vahed(k,2);
end
Avg = sum/Sum_Vahed;
end

```

خروجی این برنامه به صورت زیر می باشد.

```

Number of courses: 2
Nomreh: 14
Vahed: 4
Nomreh: 13
Vahed: 2
Your Average for 2 courses is: 13.666667

```

۲.۶.۵ توابع بدون نام

در بسیاری از برنامه‌ها نیاز به استفاده از توابعی است که بدنه‌های یک خطی دارند، برای مثال توابعی که یک تابع ریاضی را تعریف می‌کنند، در این صورت هم می‌توان از توابع به‌شکلی که بیان کردیم استفاده کنیم و هم می‌توانیم از روش جدید موسوم به توابع بدون نام استفاده کنیم.

مثال ۲.۵. در برنامه زیر و با استفاده از شکل استاندارد تعریف توابع، با استفاده از روش نیوتون و ساختن دنباله $x_{n+1} = x_n - \frac{f'(x_n)}{f(x_n)}$ ریشه معادله $x^3 + 4x - 1 = 0$ را بدست می‌آوریم.

```

x = [0];
k = 1;
x(k+1) = x(k) - f(x(k))/fp(x(k));
while abs(x(k+1) - x(k)) > 0.000001
k = k + 1;
x(k+1) = x(k) - f(x(k))/fp(x(k));
end
fprintf('Root is: %f in %i iterations.\n',x(k),k);

function fx = f(x)
fx = x^3 + 4*x -1;
end

function fpx = fp(x)

```

```
fpx = 3*x^2 + 4;
end
```

حاصل اجرای این برنامه به صورت زیر می باشد،

Root is: 0.246266 in 4 iterations.

👉 در این برنامه از دو تابع به شکل استاندارد برای محاسبه تابع $f(x)$ و مشتق $f'(x)$ استفاده کردیم، ولی برای این گونه مسایل راه ساده تری وجود دارد که در ادامه به آن می پردازیم.

👉 در این برنامه می توان در توابعی که تعریف کردیم، بجای تابع، هر تابعی را قرار داد تا مقدار جواب، در صورت وجود، با استفاده از روش نیوتون محاسبه شود.

توابع بدون نام

این گونه توابع به شکل یک دستور در میان سایر دستورات m -فایل نوشته می شوند و معمولاً برای تعریف توابع ریاضی یا عبارات ریاضی بکار برده می شوند و ساختار کلی آنها به صورت زیر است:

```
name = @(arguments) exprision
```

برای استفاده از آنها کافیست $name$ را به همراه آرگومانهای ورودی بکار ببریم.

مثال ۲۱.۵. مثال ۲۰.۵ را با استفاده از توابع بدون نام بازنویسی کنید.

```
x = [0];
k = 1;
f = @(x) x^3 + 4*x -1;
fp = @(x) 3*x^2 + 4;
x(k+1) = x(k) - f(x(k))/fp(x(k));
while abs(x(k+1) - x(k)) > 0.000000001
k = k + 1;
x(k+1) = x(k) - f(x(k))/fp(x(k));
```

```
end
fprintf('Root is: %f in %i iterations.\n',x(k),k);
```

👉 به چگونگی استفاده از توابع بدون نام برای معرفی تابع $f(x)$ و مشتق آن توجه کنید.

مثال ۲۲.۵. برنامه‌ای بنویسید که فاصله یک نقطه تا خط $2x + y + 2 = 0$ را محاسبه کند. در این برنامه از توابع بدون نام استفاده می‌کنیم.

```
dist = @(x,y) (2*x + y + 2)/sqrt(5);
x = input('x: ');
y = input('y: ');
fprintf('distance of (%i,%i) from line 2x + y + 2 is: %f\n'...
,x,y,dist(x,y))
```

خروجی این برنامه به شکل زیر است:

```
x: 4
y: 5
distance of (4,5) from line 2x + y + 2 is: 6.708204
```

دستور inline

برای تعریف توابع بدون نام به شکل دیگری نیز می‌توان عمل کرد:

```
name = inline('expression')
```

که در آن name نامی دلخواه است و عبارت مورد نظر باید میان یک جفت کوتیشن نوشته شود.

👉 با استفاده از این روش سطر اول برنامه مثال ۲۲.۵ به شکل زیر می‌باشد:

```
dist = inline('(2*x + y + 2)/sqrt(5)');
```

در برنامه‌نویسی می‌توان توابع را به شکل تودرتو نیز بکار برد، به این معنی که در داخل یک تابع، می‌توان تابع دیگری را فراخوانی کرد. در این صورت می‌توان ساختاری برای چگونگی فراخوانی توابع تودرتو به شکل زیر در نظر گرفت.

ساختار توابع تودرتو

```
function y=A(a1,a2)
.....
function z=B(b1,b2)
.....
function w=C(c1,c2)
.....
end
end
function u=D(d1,d2)
.....
function h=E(e1,e2)
.....
end
end
.....
end
```

شرح ساختار

در این ساختار، A تابع اصلی و تابع B در داخل A تعریف شده است، همچنین تابع C در داخل تابع B تعریف شده است. به همین ترتیب برای بقیه توابع نیز می‌توان تودرتو بودن را تشخیص داد.

۷.۵ تمرین

تمرین ۱.۵. عبارات زیر را ابتدا بدون استفاده از MATLAB ارزیابی کنید، سپس درستی ارزیابی خود را با MATLAB بررسی کنید.

$$۳. \quad y = 2 \times (3 > \frac{10}{5}) + (1 > 2)^2$$

$$۱. \quad 5 + 3 > \frac{32}{4}$$

$$۴. \quad 5 \times 3 - 4 \times 4 \leq \sim 2 \times 4 + \sim 0$$

$$۲. \quad y = 2 \times 3 > \frac{10}{5} + 1 > 2^2$$

تمرین ۲.۵. با فرض $a = 6$, $b = 2$, و $c = -5$ عبارات زیر را ابتدا بدون استفاده از MATLAB محاسبه کنید، سپس درستی محاسبات خود را با MATLAB بررسی کنید.

$$1. \quad y = a + b > a - b < c \quad 3. \quad y = b + c > c > \frac{a}{b}$$

$$2. \quad y = -6 < c < -2 \quad 4. \quad y = a + c == \sim (c + a \sim \frac{a}{b})$$

تمرین ۳.۵. با فرض

$$v = (4, -2, -1, 5, \circ, 1, -3, 8, 2), \quad u = (\circ, 2, 1, -1, \circ, -2, 4, 3, 2),$$

عبارات زیر را ابتدا بدون استفاده از MATLAB محاسبه کنید، سپس درستی محاسبات خود را با MATLAB بررسی کنید.

$$1. \quad \sim (\sim v) \quad 3. \quad u - v < u$$

$$2. \quad u == v \quad 4. \quad u - (v < u)$$

تمرین ۴.۵. برنامه‌ای بنویسید که عناصر دنباله فیبوناچی

$$\circ, 1, 1, 2, 3, 5, 8, 13, 21, \dots$$

را تا n جمله محاسبه کند و بدیهی است که تعداد جملات باید در ابتدا از کاربر گرفته شود.

تمرین ۵.۵. با استفاده از حلقه‌های تکرار یک ماتریس 4×3 تولید کنید که مقدار هر عنصر به شکل مجموع شماره سطر و ستون، تقسیم بر مربع شماره ستون آن عنصر باشد.

تمرین ۶.۵. عناصر ماتریس متقارن پاسکال^۱ به شکل زیر می‌باشند،

$$P_{ij} = \frac{(i + j - 2)!}{(i - 1)!(j - 1)!}.$$

برنامه‌ای بنویسید که یک ماتریس پاسکال $n \times n$ تولید کند. برنامه نوشته شده را برای مقادیر مختلف n اجرا کنید. در MATLAB با دستور `pascal(n)` می‌توان یک ماتریس پاسکال $n \times n$ تولید کرد. این دستور را برای مقادیر مختلف n بکار ببرید و نتیجه را با نتیجه حاصل از برنامه مقایسه کنید.

¹Pascal matrix

تمرین ۷.۵. برنامه‌ای بنویسید که کوچکترین عدد فرد که بر ۱۱ بخش‌پذیر باشد و مربع آن از ۱۳۲ بیشتر باشد را پیدا کند.

تمرین ۸.۵. برنامه‌ای بنویسید که مجموع زیر را محاسبه کند. در این برنامه از حلقه‌های تکرار استفاده کنید.

$$\sqrt{12} \sum_{n=0}^m \frac{\left(-\frac{1}{3}\right)^n}{2n+1}.$$

این برنامه را برای m های مختلف اجرا کنید و حاصل را با π مقایسه کنید.

تمرین ۹.۵. برنامه‌ای بنویسید که با استفاده از حلقه‌های تکرار حاصل ضرب زیر را محاسبه کند.

$$2 \prod_{n=1}^m \frac{(2n)^2}{(2n)^2 - 1} = 2 \left(\frac{4}{3} \cdot \frac{16}{15} \cdot \frac{36}{35} \cdot \dots \right).$$

برنامه را برای مقادیر مختلف m اجرا کنید و حاصل را با π مقایسه کنید.

تمرین ۱۰.۵. بردار زیر را در نظر بگیرید،

$$\mathbf{x} = (-3/5, 5, -6/2, 11/1, 0, 7, -9/5, 2, 15, -1, 3, 2, 5).$$

با استفاده از دستورات شرطی، حلقه‌های تکرار و با استفاده از بردار مفروض، برنامه‌ای بنویسید که

• برداری شامل تمام عناصر مثبت بردار \mathbf{x} تولید کند.

• برداری شامل تمام عناصر منفی بردار \mathbf{x} تولید کند.

ترتیب عناصر در هر دو بردار تولید شده، باید با ترتیب عناصر در بردار اولیه یکسان باشد.

تمرین ۱۱.۵. بردار زیر را در نظر بگیرید،

$$\mathbf{x} = (-3/5, 5, -6/2, 11/1, 0, 7, -9/5, 2, 15, -1, 3, 2, 5).$$

با استفاده از دستورات شرطی، حلقه‌های تکرار و با استفاده از بردار \mathbf{x} ، برنامه‌ای بنویسید که عناصر بردار \mathbf{x} را به شکل صعودی مرتب کند. در این برنامه از تابع کتابخانه‌ای `sort` استفاده نکنید.

تمرین ۱۲.۵. اعداد زیر نمرات مربوط به یک درس دانشجویان می‌باشد،

$$73, 91, 37, 81, 63, 66, 50, 90, 75, 43, 88, 80, 79, 69, 26, 82, 89, 99, 71, 59.$$

برنامه‌ای بنویسید که میانگین ۸ نمره برتر را محاسبه کند.

تمرین ۱۳.۵. برنامه‌ای بنویسید که عدد مثبت n را به گونه‌ای پیدا کند که مجموع $1+2+\dots+n$ عددی بین ۱۰۰ تا ۱۰۰۰ باشد و هر سه رقم مجموع یکسان باشند.

تمرین ۱۴.۵. با توجه به اینکه

- هر فوت برابر با ۱۲ اینچ است.
- هر اینچ برابر ۲/۵۴ سانتی‌متر است.
- هر پوند برابر با ۴۵۳/۵۹ گرم می‌باشد،

تابعی بنویسید که قد و وزن فردی را برحسب اینچ و پوند دریافت کند و به سیستم سانتی‌متر، کیلوگرم تبدیل کند. توجه کنید کاربر می‌تواند قد را برحسب فوت و اینچ نیز وارد کند. این تابع را در یک برنامه اصلی فراخوانی کنید و برای داده‌های مختلف از آن استفاده کنید.

تمرین ۱۵.۵. برنامه‌ای بنویسید که با معلوم بودن اضلاع یک مثلث، مساحت مثلث را حساب کند.

تمرین ۱۶.۵. فرض کنید به کاراکترهای A تا E مقادیر 4 تا 0 نسبت داده شده باشد. برنامه‌ای بنویسید که به کمک یک تابع، رشته‌ای حاوی کاراکترها بیان شده را دریافت کند و یک آرایه عددی شامل اعداد متناظر با هر کاراکتر برگشت دهد. برای مثال اگر رشته ACBAEB به عنوان ورودی به تابع ارسال شود، آرایه

[4 2 3 4 0 3]

برگشت داده شود.

۶ رسم نمودارهای دوبعدی

یکی از توانمندی‌های MATLAB، رسم نمودارها در صفحه و در فضا می‌باشد. با استفاده از MATLAB می‌توان انواع مختلف نمودارهایی که در ریاضی، آمار، فیزیک و رشته‌های مهندسی ظاهر می‌شوند را به سادگی و با کیفیت بالا رسم کرد. رسم نمودارها در MATLAB را می‌توان به دو شیوه انجام داد. در این فصل از دستورات استاندارد MATLAB برای رسم نمودارها در صفحه استفاده خواهیم کرد و در فصل‌های آینده به رسم نمودار با استفاده از بسته Symbolic می‌پردازیم. هر کدام از این دو شیوه رسم کاربردهای خاص خود را دارند و عملاً نمی‌توان یکی را جایگزین دیگری کرد، پس پیشنهاد می‌شود هر دو روش را به خوبی فرا بگیرید تا در صورت نیاز بتوانید از بهترین روش در کارهای خود استفاده کنید.

۱.۶ دستورات رسم نمودار

در MATLAB برای رسم نمودار دستورات مختلفی وجود دارد که هر یک قابلیت‌های ویژه خود را دارند. در این بخش با این دستورات آشنا خواهیم شد و مثال‌هایی از هر کدام خواهیم زد. بدیهی است که کاربران MATLAB بسته به نیاز خود می‌توانند از دستوراتی که بیان می‌شوند استفاده کنند.

۱.۱.۶ دستور plot

دستور اصلی رسم نمودار در MATLAB دستور plot می‌باشد. با استفاده از این دستور می‌توان دو گونه از نمودارها را رسم کرد: • نمودار دسته‌ای از نقاط، • نمودار یک تابع مشخص.

ساده‌ترین شکل دستور plot

این دستور در ساده‌ترین شکل خود به شکل

```
plot(x,y)
```

می باشد که در آن x و y دو آرایه یک‌بعدی هستند. این دستور به ازای هر زوج مرتب $(x(i), y(i))$ ، یک نقطه در صفحه مشخص می‌کند و هر دو نقطه متوالی را با یک خط راست به هم وصل می‌کند.

مثال ۱.۶. مجموعه نقاط زیر را رسم کنید،

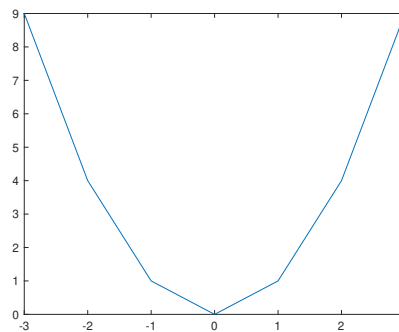
$$\{(-3, 9), (-2, 4), (-1, 1), (0, 0), (1, 1), (2, 4), (3, 9)\}$$

برای رسم این نقاط دو بردار x شامل مولفه‌های اول و y شامل مولفه‌های دوم تعریف می‌کنیم سپس با دستور plot نقاط را رسم می‌کنیم.

ورودی

```
x = [-3:3];  
y = [9,4,1,0,1,4,9];  
plot(x,y);
```

خروجی



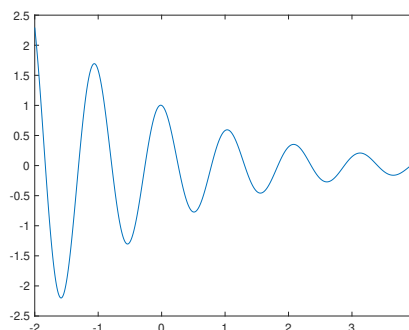
بدیهی است که بجای x و y هر دو برداری را می‌توان قرار داد، فقط توجه کنید که طول دو بردار باید یکسان باشند، در غیر این صورت در هنگام اجرا با خطا برخورد خواهید کرد. اگر بخواهیم نمودار یک تابع را رسم کنیم، باید ابتدا بردار x را با طول گام کوچک ایجاد کنیم، سپس با استفاده از ضابطه تابع و قرار دادن x در آن بردار y را تولید می‌کنیم، و در آخر با استفاده از دستور plot به شکلی که بیان کردیم، نمودار را رسم می‌کنیم.

مثال ۲.۶. نمودار تابع $y = e^{-0.5x} \cos(6x)$ را در بازه $[-2, 4]$ رسم کنید. در کد زیر به چگونگی استفاده از عملگرها به صورت معمولی و عنصر به عنصر توجه کنید.

ورودی

```
x = [-2:0.01:4];
y = exp(-0.5*x).*cos(6*x);
plot(x,y);
```

خروجی



دستور plot دارای گزینه‌هایی است که به کمک آنها می‌توان بر شکل کنترل بیشتری داشت.

شکل کامل دستور plot

دستور plot در حالت کلی به شکل زیر می‌باشد،

```
plot(x,y,'Line Specifiers','Property Name',Property Value)
```

که در آن

Line Specifiers رشته‌ای حداکثر چهار کاراکتری است که تعیین‌کننده مواردی مانند نوع خط، رنگ خط و شکل مورد استفاده برای نقاط می‌باشد. استفاده از این گزینه اختیاری است و اگر از آن استفاده نشود، نوع خط مورد استفاده به شکل پیوسته و به رنگ آبی می‌باشد و نقاط به شکل ویژه نمایش داده نخواهند شد.

Property Name این گزینه نیز اختیاری است ولی در صورت استفاده باید آن را به همراه **Property Value** به هر تعداد که مورد نیاز باشد بکار برد. با استفاده از این گزینه می‌توان مواردی مانند ضخامت خط، اندازه نقاط نمایش داده شده بر روی نمودار، رنگ خط دور نقطه و رنگ داخل نقاط را کنترل کرد.

در ادامه این بخش به معرفی کامل تمام گزینه‌های مورد استفاده در دستور plot خواهیم پرداخت.

Line Specifiers

در صورت استفاده از این گزینه باید یک رشته حداکثر چهار کاراکتری در دستور plot قرار داد. این چهار کاراکتر، کاراکترهای ویژه‌ای هستند که به منظور خاصی بکار برده می‌شوند و باید به شکل پشت‌سرهم و در میان یک جفت کوتیشن نوشته شوند. در حالتی که از این گزینه استفاده نکنیم، نوع خط مورد استفاده در رسم نمودار پیوسته می‌باشد، ولی می‌توانیم با استفاده از یکی از گزینه‌های زیر نوع خط را به یکی از شکل‌های بیان شده در جدول ۱.۶ تغییر دهیم.

جدول ۱.۶: کاراکترهای تعیین نوع خط

نوع خط	کاراکتر	نوع خط	کاراکتر
پیوسته	-	خط چین	--
نقطه چین	:	نقطه خط	-.

تغییر رنگ خط نیز به سادگی و با استفاده از گزینه‌های جدول ۲.۶ امکان پذیر است. در صورت عدم استفاده از این گزینه‌ها رنگ خط آبی خواهد بود.

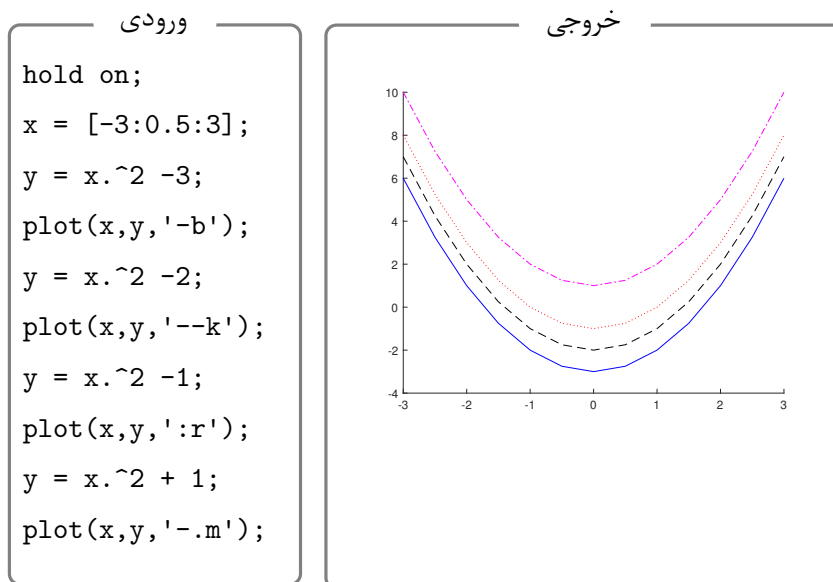
جدول ۲.۶: کاراکترهای تعیین رنگ خط

رنگ خط	کاراکتر	رنگ خط	کاراکتر
red	r	green	g
blue	b	cyan	c
magenta	m	yellow	y
black	k	white	w

مثال ۳.۶.

به چگونگی استفاده از Line Specifiers جداول ۱.۶ و ۲.۶ در نمودارهای زیر توجه کنید. در سطر اول این کد از دستور hold on استفاده شده است. این دستور امکان رسم چند نمودار را

در یک دستگاه مختصات فراهم می‌کند. برای غیر فعال کردن رسم همزمان نمودارها باید از دستور `hold off` استفاده کرد.



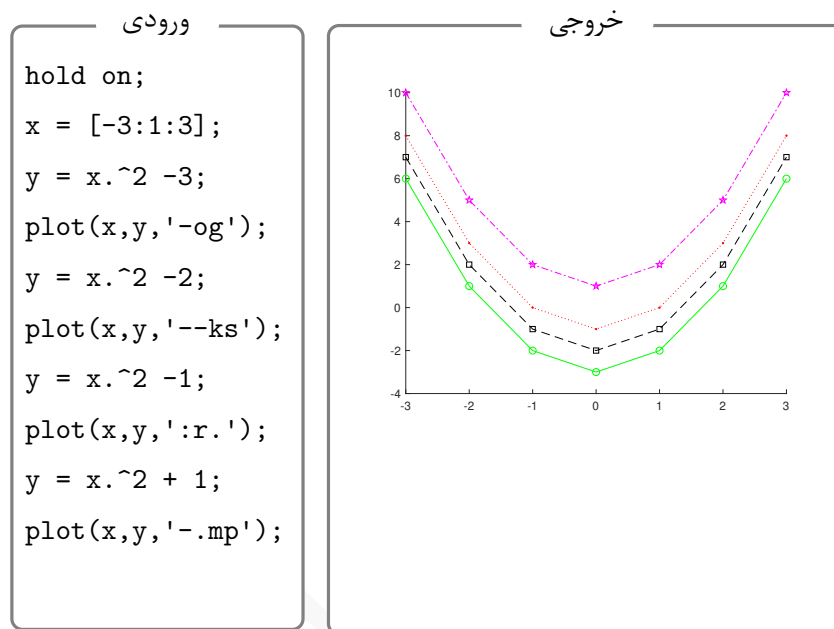
دستور `plot` نمودار توابع را از متصل کردن تعدادی نقطه به یکدیگر تولید می‌کند. در حالت معمولی نقاط مورد استفاده نمایش داده نمی‌شوند و تنها خطوطی که هر دو نقطه مجاور را به هم وصل می‌کنند نمایش داده می‌شود. ولی اگر بخواهیم نقاط هم نمایش داده شوند، می‌توان از یک کاراکتر کنترلی دیگر در `Line Specifiers` استفاده کرد که مشخص‌کننده شکل نمایش نقطه در نمودار می‌باشد. این کاراکترها را می‌توان از جدول ۳.۶ انتخاب نمود.

جدول ۳.۶: کاراکترهای تعیین شکل نقاط

کاراکتر	شکل نقطه	کاراکتر	شکل نقطه
o	دایره	+	علامت جمع
.	نقطه	*	ستاره
s	مربع	x	علامت ضرب
p	ستاره پنج پر	d	لوزی
^	مثلث رو به بالا	v	مثلث رو به پایین
>	مثلث رو به راست	<	مثلث رو به چپ

در ادامه مثال‌هایی در زمینه استفاده ترکیبی از کاراکترهای بیان شده خواهیم آورد.

مثال ۴.۶. در دستورات زیر از تمامی کاراکترهای کنترلی استفاده شده است.



👉 در اولین دستور `plot` اگر رشته کنترلی به شکل `'og'` استفاده شود، فقط نقاط نمایش داده خواهد شد و هیچ خطی رسم نخواهد شد.

👉 ترتیب نوشتن کاراکترها اثری در خروجی ندارد.

پیشنهاد می‌شود دستورات را در یک `m`-فایل بنویسید و با ایجاد تغییرات و استفاده از ترکیب‌های مختلف کاراکترهای کنترلی، خروجی‌های متفاوتی که تولید می‌شود را بررسی کنید. همچنین پیشنهاد می‌شود تا با کوچک کردن طول گام در تولید بردار `x` و اجرای برنامه نمودارهای حاصل را بررسی کنید.

مثال ۵.۶. به دستورات زیر و نوع خروجی که تولید می‌کنند توجه کنید.

`plot(x,y,'-g')` نموداری به شکل نقطه خط و به رنگ سبز رسم می‌شود.

`plot(x,y,':sg')` نموداری به شکل نقطه‌چین، به رنگ سبز و نقاطی به شکل مربع تولید می‌گردد.

`plot(x,y,'gd')` نموداری با خطوط پیوسته و به رنگ سبز رسم می‌شود.

`plot(x,y,'x')` فقط نقاط به صورت علامت ضرب و به رنگ پیش فرض، یعنی آبی، انجام می شود.

برای تعیین نوع خط، رنگ خط و نمایش نقاط از روش دیگری نیز می توان استفاده کرد که در بخش بعد به معرفی آن می پردازیم.

Property Name, Property Value

در دستور `plot` این دو گزینه باید با هم استفاده شوند. Property Name می تواند یکی از مقادیری باشد که در جدول ۴.۶ آورده شده است. این گزینه ها باید دقیقاً به شکلی که نوشته شده اند و در داخل یک جفت کوتیشن نوشته شوند.

جدول ۴.۶: گزینه های کنترل مشخصات نمودار

Property Value	شرح عمل	Property Name
اعداد حقیقی مثبت	تعیین ضخامت خط	LineWidth linewidth
اعداد حقیقی مثبت	تعیین اندازه نقاط	MarkerSize markersize
یکی از مقادیر جدول ۲.۶ به شکل رشته	تعیین رنگ خط نقاط	MarkerEdgeColor markeredgecolor
یکی از مقادیر جدول ۲.۶ به شکل رشته	تعیین رنگ داخل نقاط	MarkerFaceColor markerfacecolor

مثال ۴.۶. به چگونگی استفاده از Property Name و Property Value در دستورات زیر دقت کنید.

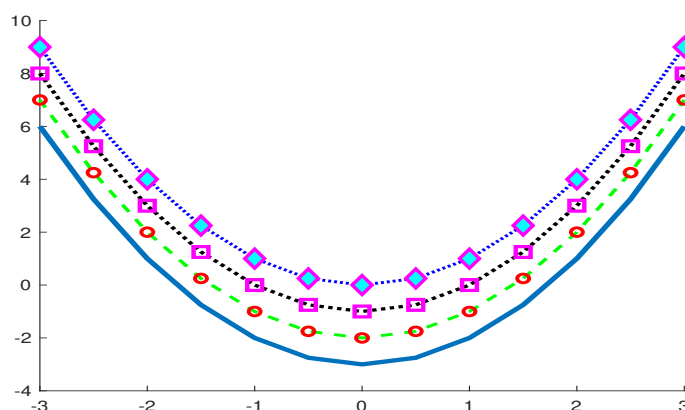
```
hold on;
x = [-3:0.5:3];
y = x.^2 -3;
plot(x,y,'linewidth',3);
y = x.^2 -2;
```

```

plot(x,y,'--og','linewidth',2,'markeredgecolor','r');
y = x.^2 -1;
plot(x,y,':sk','linewidth',2.25,'markersize',10,...
'markeredgecolor','m');
y = x.^2;
plot(x,y,':db','linewidth',2,'markersize',10,...
'markeredgecolor','m','markerfacecolor','c');

```

خروجی کد بالا به صورت زیر می باشد.



👉 برای مشاهده رنگها، پیشنهاد می شود کد را در یک m-فایل بنویسید و اجرا کنید. همچنین می توانید با ایجاد تغییراتی در کد، خروجی های متفاوت را بررسی کنید.

با استفاده از Property Name و Property Value می توان نوع و رنگ خط و شکل نمایش نقاط را تعیین کرد. به این منظور می توانید از جدول ۵.۶ استفاده کنید.

👉 استفاده از روش بیان شده در بخش پیش کمی ساده تر است، لذا پیشنهاد می شود برای کنترل نوع و رنگ خط و شکل نمایش نقاط از آن روش استفاده کنید.

جدول ۵.۶: گزینه‌های تعیین نوع خط

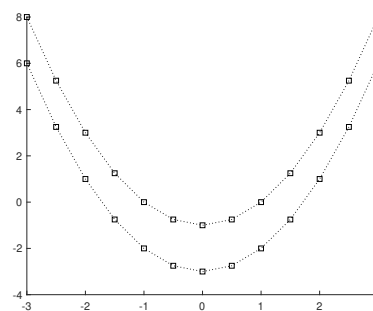
Property Name	شرح عمل	Property Value
LineStyle linestyle	تعیین نوع خط	یکی از مقادیر جدول ۱.۶ به شکل رشته
Color color	تعیین رنگ خط	یکی از مقادیر جدول ۲.۶ به شکل رشته
Marker marker	تعیین شکل نمایش نقاط	یکی از مقادیر جدول ۳.۶ به شکل رشته

مثال ۷.۶. در کد زیر هر دو دستور plot خروجی یکسانی دارند.

ورودی

```
hold on;
x = [-3:0.5:3];
y = x.^2 - 3;
plot(x,y,'ks');
y = x.^2 - 1;
plot(x,y,'linestyle',':',...
'color','k','marker','s');
```

خروجی



مثال ۸.۶. نمودار توابع $y = \sin x + x$ و $y = \cos x - x$ را در یک نمودار و با ویژگی‌های مختلفی برای رنگ، شکل و دیگر موارد رسم کنید. در کد نوشته شده به روش استفاده از گزینه‌های دستور plot برای کنترل نمودار توجه کنید.

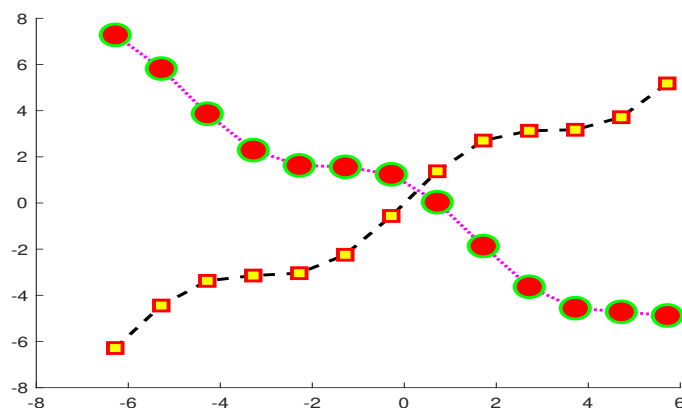
```
hold on;
x = [-2*pi:1:2*pi];
y = sin(x) + x;
plot(x,y,'--ks','LineWidth',2,'MarkerSize',10,...
'MarkerEdgecolor','r','MarkerFaceColor','y');
```

```

y = cos(x) - x;
plot(x,y,':ro');
plot(x,y,'linestyle',':', 'color','m','marker','o',...
'LineWidth',2,'MarkerSize',15,'MarkerEdgecolor','g',...
'MarkerFaceColor','r');

```

خروجی این کد به شکل زیر می باشد.



👉 پیشنهاد می شود کد نوشته شده را در MATLAB وارد کنید و با ایجاد تغییراتی در آن خروجی های متفاوتی که حاصل می شود را بررسی کنید. همچنین سعی کنید با دستور plot و گزینه های اختیاری آن نمودار توابع دیگری را رسم کنید. برای این کار به یک کتاب حساب دیفرانسیل و انتگرال مراجعه کنید.

پیشتر دیدید که با دستور hold on این امکان داشت که خروجی حاصل از چند دستور plot را در یک نمودار رسم کرد، ولی این امکان وجود دارد که در یک دستور plot چندین نمودار را رسم کنیم. به این منظور می توان دستور را به شکل زیر بکار برد،

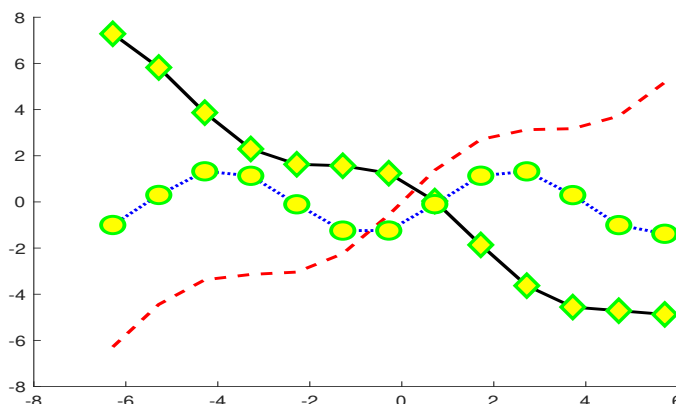
```
plot(x1,y1,'LS',x2,y2,'LS',...,Options)
```

در این دستور ابتدا باید نقاط به همراه Line Specifiers برای هر دسته نقاط نوشته شود و در آخر باید Property Name و Property Value نوشته شود تا برای تمام نمودارها اعمال شود.

مثال ۹.۶. به چگونگی استفاده از دستور plot برای رسم چند نمودار به‌طور همزمان توجه کنید.

```
x = [-2*pi:1:2*pi];
y1 = sin(x) + x;
y2 = cos(x) - x;
y3 = sin(x) - cos(x);
plot(x,y1,'--r',x,y2,'-kd',x,y3,':bo', 'LineWidth',2,...
'markersize',12,'markeredgecolor','g',...
'markerfacecolor','y');
```

خروجی کد بالا به‌صورت زیر می‌باشد.



📌 اگر رنگی برای نمودارها مشخص نشود، پس از اجرای دستور plot به‌شکل خودکار رنگ‌های مختلفی برای نمودارها در نظر گرفته خواهد شد.

📌 در این شکل استفاده از دستور plot گزینه‌های مشخص شده، Options، بر تمام نمودارها به یک شکل اعمال می‌شود، لذا اگر بخواهید گزینه‌های متفاوتی بر نمودارها اعمال شوند، بهتر است به شکلی که پیشتر بیان کردیم با استفاده از hold on و دستورات plot مختلف نسبت به رسم اقدام کنید.

۲.۱.۶ دستور fplot

اگر ضابطه یک تابع به صورت $y = f(x)$ معلوم باشد، بدون آنکه نیاز به تولید آرایه x و ساختن بردار y از روی x باشد، می‌توان با دستور fplot نمودار تابع را رسم کرد.

دستور fplot

دستور fplot برای رسم نمودار تابع $y = f(x)$ در یک بازه مشخص و بدون نیاز به تعیین نقاط بکار می‌رود. شکل کلی این دستور به صورت زیر می‌باشد،

```
fplot('function',Limit,'Line Specifiers',
      'Property Name',Property Value)
```

که در آن

function ضابطه تابع مورد نظر است که باید به صورت یک رشته وارد شود. در هنگام ورود ضابطه نیازی به استفاده از عملگرهای عنصر به عنصر نیست و از عملگرهای معمولی استفاده می‌شود.

Limit بازه‌ای است که می‌خواهیم تابع در آن بازه رسم شود و به صورت دو عدد میان دو کروشه و به شکل $[a, b]$ نوشته می‌شود.

Line Specifiers رشته‌ای است که تعیین کننده نوع و رنگ خط و شکل نمایش نقاط می‌باشد. این رشته دقیقاً به همان صورتی که در دستور plot بیان شده، جداول ۱.۶، ۲.۶ و ۳.۶ را ببینید، نوشته می‌شود. استفاده از این گزینه اختیاری است.

Property Value کاملاً مشابه دستور plot بکار می‌رود، جداول ۴.۶ و ۵.۶ را ببینید، و برای ایجاد تغییرات در مواردی مانند ضخامت خط، اندازه نقاط نمایش داده شده بر نمودار، رنگ خط دور نقطه و رنگ داخل نقاط بکار برده می‌شود. استفاده از این گزینه اختیاری می‌باشد.

📌 در دستور fplot اگر از کاراکترهای کنترلی برای نمایش نقاط استفاده کنیم، تعدادی نقطه به صورت خودکار روی نمودار نمایش داده خواهد شد.

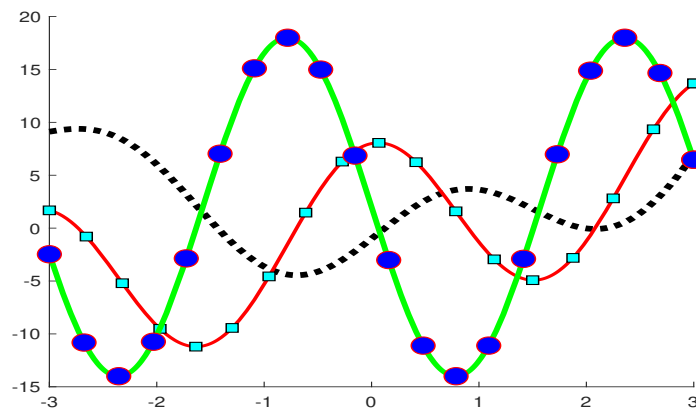
📌 در دستور fplot تابع ورودی را می‌توان با استفاده از توابع بدون نام و به صورت `function @(x)` نیز بکار برد که در این صورت باید ضابطه تابع را به شکلی که در بخش ۲.۶.۵ بیان کردیم تعریف

کنیم. توجه کنید که در این حالت باید برای تعریف تابع از عملگرهای عنصر به عنصر استفاده کنید. پیشنهاد می‌شود در صورت استفاده از `fplot` از این روش برای تعریف تابع استفاده کرد.

مثال ۱۰.۶. نمودار تابع $y = x^2 + 4 \sin 2x - 1$ را به همراه نمودار مشتق اول آن در یک نمودار رسم کنید. در رسم این نمودارها برای نمایش بهتر نمودار از گزینه‌های اختیاری استفاده کرده‌ایم، به چگونگی استفاده از آنها توجه کنید.

```
hold on
fplot(@(x) x.^2+4*sin(2*x)-1,[-3,3],':k','LineWidth',4);
fplot(@(x) 2*x+8*cos(2*x),[-3,3],'-rs','LineWidth',2,...
'markersize',8,'markeredgecolor','k','markerfacecolor','c');
fplot(@(x) 2-16*sin(2*x),[-3,3],'-go','LineWidth',3,...
'markersize',12,'markeredgecolor','r','markerfacecolor','b');
```

خروجی این دستورات به صورت زیر است. حتما دستورات را در محیط MATLAB اجرا کنید.



۳.۱.۶ دستور line

علاوه بر دو دستور `plot` و `fplot`، دستور دیگری برای رسم نمودار در MATLAB وجود دارد که تا اندازه زیادی شبیه دستور `plot` می‌باشد و تنها چند تفاوت کوچک بین آنها وجود دارد.

دستور line

دستور line از نظر ساختاری کاملاً شبیه دستور plot می‌باشد،

```
line(x,y,'Line Specifiers','Property Name',Property Value)
```

ولی دو تفاوت میان دستور line و دستور plot وجود دارد:

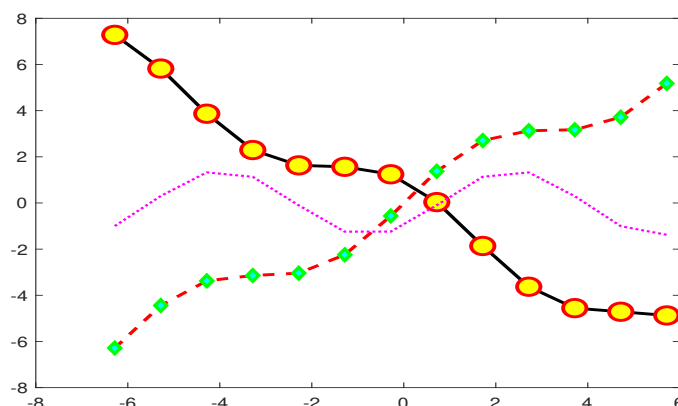
۱. Line Specifiers باید به‌صورتی که در جدول ۵.۶ آورده شده‌اند مورد استفاده قرار بگیرند، لذا امکان استفاده به‌شکل رشته حداکثر چهارکاراکتری در دستور line وجود ندارد.
۲. در صورتی که از دستور line به‌همراه دستور plot یا دستورهای line دیگری استفاده کنیم، نمودارها در صفحه موجود نمودارها رسم خواهند شد و نمودارهای پیشین پاک نخواهند شد. به عبارت بهتر در صورت استفاده از line نیازی به دستور hold on نیست.

توجه کنید که اگر از دستور hold on استفاده نکرده باشید، به محض اولین استفاده از دستور plot نمودارهای موجود پاک خواهند شد و فقط نمودار جدید نمایش داده خواهد شد.

مثال ۱۱.۶. به چگونگی رسم سه نمودار با استفاده از دستورات plot و line در کد زیر دقت کنید.

```
x = [-2*pi:1:2*pi];
y1 = sin(x) + x;
y2 = cos(x) - x;
y3 = sin(x) - cos(x);
plot(x,y1,'--rd','LineWidth',2,...
'Markeredgecolor','g','markerfacecolor','c')
line(x,y2,'linestyle','-','color','k','marker','o',...
'LineWidth',2,'Markeredgecolor','r',...
'markerfacecolor','y','markersize',12)
line(x,y3,'linestyle',':','color','m','LineWidth',1.5)
```

خروجی این دستورات به شکل زیر می باشد.

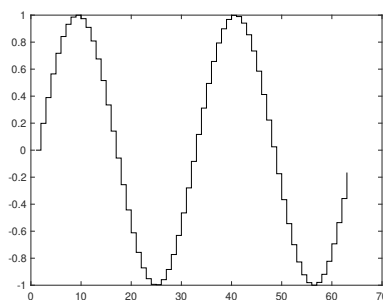


دستور ساده‌ای به صورت `stairs(y, Line Specifiers)` وجود دارد که نمودار پله‌ای تابع $y = f(x)$ را رسم می‌کند. برای مثال دستورات و خروجی متناظر با آن را ببینید.

ورودی

```
x = -2*pi:0.2:2*pi;
y = sin(x);
stairs(y, '-k');
```

خروجی



بدیهی است که امکان استفاده از دستورات مربوط به تعیین نوع و رنگ خطوط مطابق با آنچه در دستورات دیگر مربوط به رسم نمودارها گفتیم در مورد این دستور نیز برقرار می باشد.

۴.۱.۶ رسم نمودارهای قطبی

در MATLAB برای رسم نمودارهای قطبی دستور ویژه‌ای وجود دارد که به سادگی قابل استفاده است.

دستور subplot

برای رسم نمودارهای قطبی می‌توان از دستور polar به شکل زیر استفاده کرد،

```
polar(theta,rho,'Line Specifiers')
```

که در آن

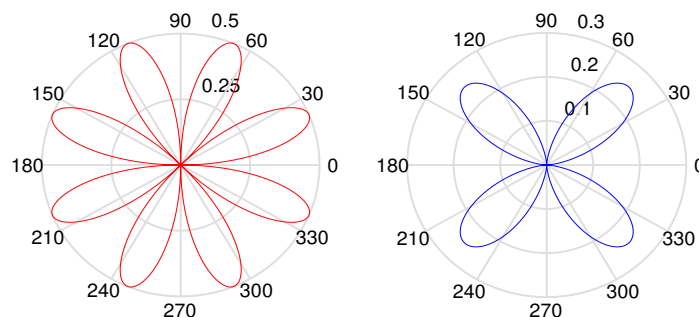
theta یک بردار می‌باشد که از تقسیم‌بندی یک بازه حاصل می‌شود،

rho ضابطه نمایش قطبی تابع است.

مثال ۱۲.۶. به نمودارهای قطبی تولید شده با دستورات زیر توجه کنید. در کدهای زیر از دستور subplot برای تولید دو نمودار در کنار هم استفاده شده است که در بخش ۲.۶ به طور کامل شرح داده شده است.

```
theta = 0:0.01:2*pi;
rho = sin(2*theta).*cos(2*theta);
subplot(1,2,1);
polar(theta,rho,'r-');
rho = sin(theta).^2 .* cos(theta).^2;
subplot(1,2,2);
polar(theta,rho,'-b');
```

خروجی کد بالا به صورت زیر می‌باشد. برای تمرین بهتر است کد بالا را در MATLAB بنویسید و خروجی حاصل را برای توابع قطبی مختلف بررسی کنید.



۲.۶ رسم چند نمودار در کنار هم

در MATLAB این امکان وجود دارد که چند نمودار را که در شکل‌های مختلف رسم شده‌اند، در کنار یکدیگر نمایش دهیم. به این منظور می‌توان از دستور subplot استفاده کرد. از این دستور بیشتر در مقالاتی استفاده می‌شود که دارای تعداد زیادی نمودار هستند و نگارنده قصد مقایسه میان نمودارها را دارد، لذا باید نمودارها به هر تعدادی که نیاز باشد به شکل چند سطر و چند ستون در کنار هم ظاهر شوند تا امکان انجام مقایسه فراهم شود.

دستور subplot

شکل کلی این دستور به صورت subplot(m,n,p) است که در آن

m تعداد سطرهای شکل نهایی است.

n تعداد ستون‌های شکل نهایی است.

p عددی بین ۱ تا mn (حاصل ضرب m در n) است. برای شکل سطر اول و ستون اول عدد ۱ و برای شکل سطر m و ستون n عدد mn در نظر گرفته می‌شود.

📌 برای استفاده از دستور subplot باید ابتدا این دستور را بکار برد، سپس یک دستور مربوط به رسم نمودار را مورد استفاده قرار داد.

مثال ۱۳.۶. نمودار توابع زیر را در چهار نمودار مختلف و به شکل دو سطر و دو ستون رسم کنید.

$$y = x^2 - 1, \quad y = x \sin x - 1, \quad y = e^x \cos(x), \quad y = x^3 - 2x^2 + 2x - 1.$$

دستورات زیر خروجی مطلوب را تولید می‌کنند.

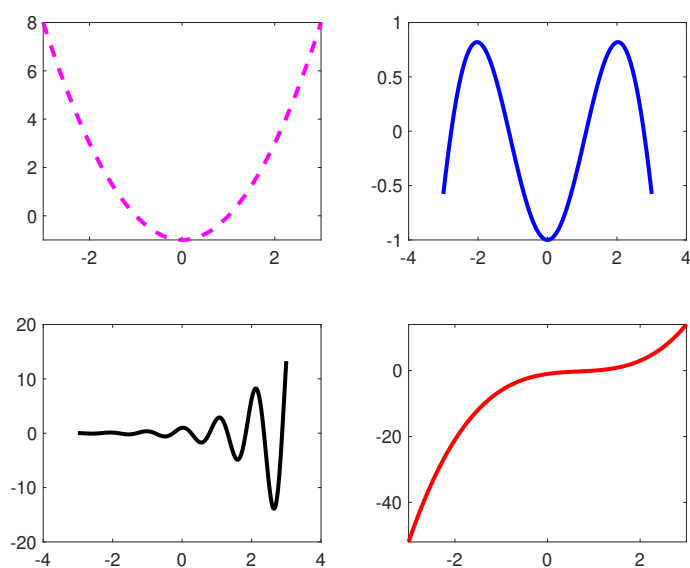
```
t = [-3:0.01:3];
subplot(2,2,1)
fplot(@(x) x.^2 - 1, [-3,3], '--m', 'LineWidth', 2);
y = t.*sin(t)-1;
subplot(2,2,2)
```

```

plot(t,y,'-b','LineWidth',2);
y = exp(t).*cos(6*t);
subplot(2,2,3)
plot(t,y,'-k','LineWidth',2);
subplot(2,2,4)
fplot(@(x) x.^3 - 2*x.^2 + 2*x -1,[-3,3],'-r','LineWidth',2);

```

خروجی این کد به صورت زیر می باشد. حتما کد را در MATLAB اجرا کنید تا تصاویر را به صورت رنگی مشاهده کنید.



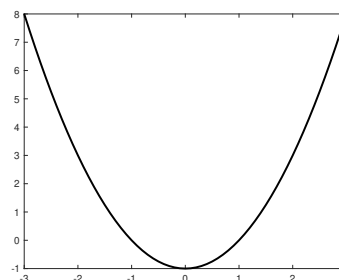
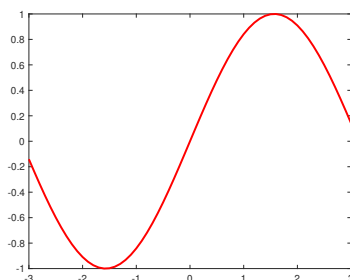
دستور figure

در MATLAB با هر بار استفاده از دستور figure می توان یک پنجره جدید ایجاد کرد که در صورت استفاده از یکی از دستورات رسم نمودار، نمودار حاصل در پنجره جدید رسم خواهد شد.

مثال ۱۴.۶. به خروجی حاصل از کد زیر که با دو دستور figure تولید شده است توجه کنید.

```
x = [-3:0.1:3];
z = sin(x); figure;
plot(x,z,'-r','LineWidth',2);
y = x.^2 - 1; figure;
plot(x,y,'-k','LineWidth',2);
```

خروجی دستورات بالا به شکل زیر است.



۳.۶ قالب بندی نمودارها

در هنگام تولید نمودارها شاید به افزودن مواردی به نمودار نیاز پیدا کنیم که به قابل فهم تر شدن نمودار کمک می کنند، عنوان نمودار، راهنمای نمودار، اسامی محورهای مختصات و ... از این دست موارد هستند که معمولاً به آنها احتیاج پیدا می کنیم. در این بخش به چگونگی افزودن این گونه موارد به نمودارها سخن خواهیم گفت.

به طور کلی قالب بندی نمودارها هم با دستورات MATLAB امکان پذیر است، که پس از هر بار اجرای دستورات بر شکل نهایی اعمال خواهد شد، و هم با استفاده از محیط گرافیکی ممکن است، که پس از هر بار اجرا باید دوباره انجام شوند.

📁 اگر می خواهید در یک m-فایل و در قالب یک برنامه نمودارهای خود را رسم کنید حتماً از روش اول استفاده کنید.

۱.۳.۶ قالب‌بندی نمودارها به کمک دستورات MATLAB

در این روش چند دستور ساده وجود دارد که باید پس از رسم نمودار بسته به نیاز نوشته شوند. استفاده از این دستورات ضروری نیستند ولی می‌توانند به قابل فهم‌تر شدن نمودارها کمک شایانی کنند، لذا استفاده از این دستورات به ویژه در مقالات و پایان‌نامه‌ها توصیه می‌شود.

دستورات تولید عنوان و برچسب‌گذاری محورها

پس از تولید نمودار، معمولاً علاقه‌مند به نوشتن عنوانی برای نمودار هستیم، همچنین در بسیاری از موارد نیاز به نوشتن توضیحاتی پیرامون محورهای مختصات وجود دارد. به این منظور سه دستور ساده وجود دارد که این کارها را به‌سادگی به انجام می‌رسانند.

درج عنوان و برچسب‌گذاری محورها

برای نوشتن عنوان نمودار بر بالای نمودار تولید شده، پس از دستور مربوط به رسم از دستور `title` استفاده کنید. می‌توان هر عنوان دلخواهی را به‌صورت یک رشته در این دستور قرار داد.

```
title('Title Of the Plot ...')
```

برای تولید برچسب‌های محورهای مختصات می‌توان از دو دستور زیر استفاده کرد:

```
xlabel('Label of x axis')
```

```
ylabel('Label of y axis')
```

از دستور اول برای تولید برچسب محور x ها و از دستور دوم برای تولید برچسب محور y ها استفاده می‌شود. بدیهی است که هر آنچه در میان جفت کوتیشن نوشته شود، در خروجی و در محل مناسب نوشته خواهد شد.

👉 دستورات بیان شده پس از دستورات `plot`، `fplot` و `line` قابل استفاده می‌باشند.

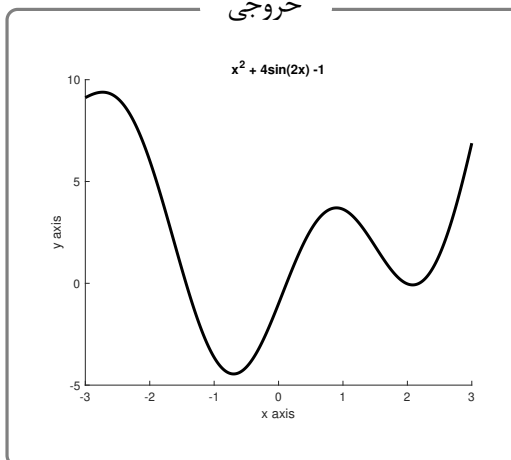
👉 در دستورات بیان شده امکان تغییر اندازه و رنگ نوشتار نیز وجود دارد. همچنین امکان استفاده از فرمول‌نویسی ریاضی، شبیه LATEX ، در نمودارها و در این دستورات نیز وجود دارد که در ادامه همین بخش به آنها خواهیم پرداخت.

مثال ۱۵.۶. در دستورات زیر به چگونگی نوشتن متن مورد نظر در `title` و چگونگی ظاهر شدن آن در بالای نمودار توجه کنید.

ورودی

```
box off;
x = [-3:0.01:3];
y = x.^2+4*sin(2*x)-1;
plot(x,y,'-k',...
'LineWidth',2.5);
title('x^2+4sin(2x)-1');
xlabel('x axis');
ylabel('y axis');
```

خروجی



نکته عملی

در سطر اول از دستور `box off` برای نمایش نمودار به شکل نمایش داده شده استفاده کرده‌ایم. اگر از این دستور استفاده نکنیم، نمودار در داخل یک مربع رسم می‌شود. حالت پیش فرض این دستور `box on` می‌باشد که معادل با نوشتن این دستور است.

تولید راهنمای نمودار

اگر در یک شکل بیش از یک نمودار رسم شده باشد بهتر است تا در گوشه‌ای از شکل راهنمایی برای نمودار تهیه شود تا خواننده بتواند بر اساس نوع خط و رنگ خط مورد استفاده، نمودار و ضابطه مربوط به آن را تشخیص دهد. در MATLAB به سادگی و با یک دستور می‌توان اینکار را انجام داد.

تولید راهنمای نمودار

راهنمای نمودار را می‌توان با دستور زیر تولید کرد،

```
legend('str',... 'str','Location','Pos','Numcolumns',value)
```

که در آن

str ضابطه توابعی هستند که نمودار آنها رسم شده و باید بین یک جفت کوتیشن نوشته شوند.

Location این واژه باید به همین صورت، یا به شکل `location`، و در میان یک جفت کوتیشن نوشته شود و پس از آن، یعنی بجای `Pos`، باید موقعیتی که می‌خواهیم راهنما در آن تولید شود در داخل یک جفت کوتیشن نوشته شود. جهت‌های معتبر عبارتند از:

southeast •	northeast •	east •	north •
southwest •	northwest •	west •	south •

استفاده از این گزینه اختیاری است و در صورت عدم استفاده، راهنما در گوشه بالا سمت راست قرار خواهد گرفت.

NumColumns این واژه باید به همین صورت، یا به شکل `numcolumns` و در میان یک جفت کوتیشن نوشته شود و پس از آن، یعنی بجای `value`، باید تعداد ستونی که می‌خواهیم برای راهنما در نظر گرفته شود به صورت یک عدد طبیعی نوشته شود. استفاده از این گزینه اختیاری است و در صورت عدم استفاده راهنما به شکل یک ستونی تولید می‌شود.

مثال ۱۶.۶. با دستورات زیر ۴ نمودار در یک شکل رسم شده‌اند و با دستور `legend` برای شکل راهنما تولید شده است. این راهنما در شمال شرق شکل و به شکل دوستونی تولید شده است. همچنین در این شکل برای محورها نیز برچسب انتخاب شده است و عنوانی نیز برای شکل در نظر گرفته شده است.

```
hold on;
```

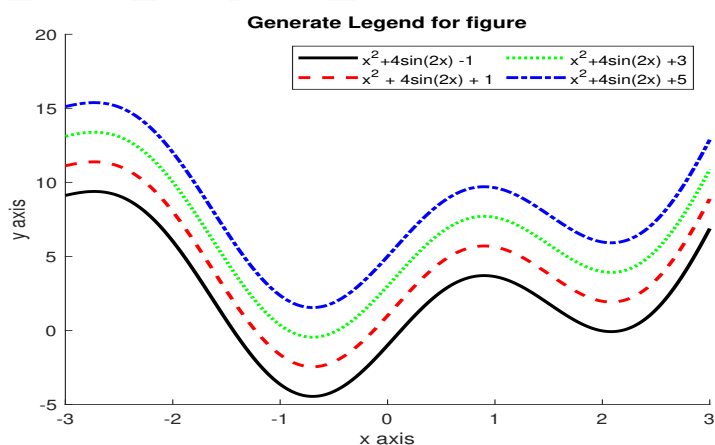
```
x = [-3:0.01:3];
```

```

y1 = x.^2 + 4*sin(2*x) -1;
plot(x,y1,'-k','LineWidth',2);
y2 = x.^2 + 4*sin(2*x) +1;
plot(x,y2,'--r','LineWidth',2);
y3 = x.^2 + 4*sin(2*x) +3;
plot(x,y3,':g','LineWidth',2);
y4 = x.^2 + 4*sin(2*x) +5;
plot(x,y4,'-.b','LineWidth',2);
title('Generate Legend for figure')
xlabel('x axis');
ylabel('y axis');
legend('x^2+4sin(2x) -1','x^2 + 4sin(2x) + 1',...
'x^2+4sin(2x) +3','x^2+4sin(2x) +5',...
'location','northeast','NumColumns',2);
box off

```

خروجی دستورات بالا به صورت زیر می باشد.



📖 به عنوان تمرین پیشنهاد می شود دستورات بالا را در یک MATLAB بنویسید و با ایجاد تغییرات در آن خروجی های مختلف را بررسی کنید. برای مثال بجای عدد ۲ مقادیر دیگری قرار

دهید یا موقعیت راهنما را تغییر دهید.

در MATLAB این امکان وجود دارد که در صورت نیاز، در داخل شکل و در کنار نمودارها چیزی نوشت. این کار با دستورات `text` و `gtext` انجام می‌شود.

دستور `text`

با دستورات `text` و `gtext` می‌توان در محل مشخصی از شکل و در کنار نمودارها مطالبی نوشت. شکل کلی این دستورات به صورت زیر می‌باشند،

```
text(x,y,'string')
gtext('string')
```

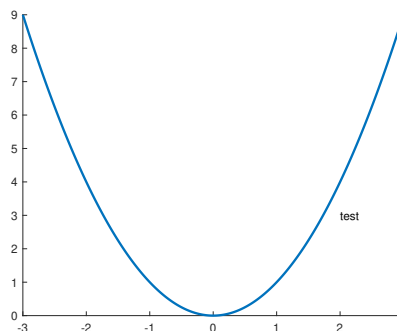
- در دستور `text`، مختصات محلی که می‌خواهیم رشته `string` در آن محل نوشته شود با مقادیر `x` و `y` مشخص می‌شود.
- در دستور `gtext` پس از اجرای دستور، باید با ماوس روی نقطه‌ای از شکل که می‌خواهیم رشته `string` در آن نقطه درج شود کلیک کنیم تا اجرای فرمان کامل شود.

مثال ۱۷.۶. در شکل زیر به روش استفاده از دستور `text` توجه کنید.

ورودی

```
box off;
x = [-3:0.01:3];
y = x.^2;
plot(x,y,'LineWidth',2);
text(2,3,'test');
```

خروجی



🔗 در کد بالا، `text` را به `gtext` تغییر دهید و نتیجه کار را بررسی کنید.

۲.۳.۶ قالب‌بندی متن درون شکل

می‌توان در رشته‌ای که با دستورات بیان شده در بخش ۱.۳.۶ برای عنوان، برجسب محورهای مختصات، راهنمای نمودار و متن داخل شکل تولید کردیم، تغییراتی مانند رنگ متن، اندازه متن، قونت متن و ... ایجاد کرد. همچنین این امکان وجود دارد که با دستوراتی مشابه آنچه در حروف‌چینی با \LaTeX وجود دارد، برخی فرمول‌ها، نمادها و حروف یونانی را در رشته‌های مورد استفاده در شکل تولید کرد. در این بخش به معرفی این قابلیت در MATLAB می‌پردازیم.

تغییر اندازه، تغییر فونت و شکل نمایش متن

اگر بخواهیم در متن مورد استفاده در دستورات `text`، `legend`، `ylabel`، `xlabel`، `title` و `gtext` از نظر نوع فونت و اندازه فونت تغییراتی ایجاد شود، می‌توان از دستوراتی خاصی استفاده کرد.

تغییر شکل نمایش متن

برای نمایش متن در سیستم‌های حروف‌چینی می‌توان از چندین شکل مختلف استفاده کرد. برخی از اشکال موجود را می‌توان در MATLAB نیز مورد استفاده قرار داد،

`\bf{text}` استفاده از این دستور باعث می‌شود تا متن نوشته شده در میان جفت آکولاد به صورت پررنگ نمایش داده شود.

`\it{text}` استفاده از این دستور باعث می‌شود تا متن نوشته شده در میان جفت آکولاد به صورت خوابیده نمایش داده شود.

`\rm{text}` استفاده از این دستور باعث می‌شود تا متن نوشته شده در میان جفت آکولاد به صورت معمولی نمایش داده شود. استفاده از این دستور معادل با ننوشتن آن می‌باشد.

توجه کنید که این دستورات باید در میان یک جفت کوتیشن قرار بگیرند، در این صورت هر آنچه میان جفت آکولاد قرار گرفته است تغییر شکل خواهد یافت.

برای تغییر اندازه فونت و نوع فونت نیز دستوراتی در MATLAB وجود دارد.

^۱ به کتاب خودآموز سریع \LaTeX از همین نویسنده مراجعه کنید.

تغییر اندازه و نوع فونت

با استفاده از دو دستور می‌توان اندازه و نوع فونت مورد استفاده برای نمایش متن را تغییر داد.

\fontname{Font Name} این دستور باعث می‌شود تا فونت مورد استفاده برای متن از جایی که این دستور نوشته شده است تا انتهای متن تغییر کند. بدیهی است که Font Name نام فونتی است که می‌خواهیم مورد استفاده قرار بگیرد. توجه کنید که این فونت باید روی کامپیوتر شما نصب باشد.

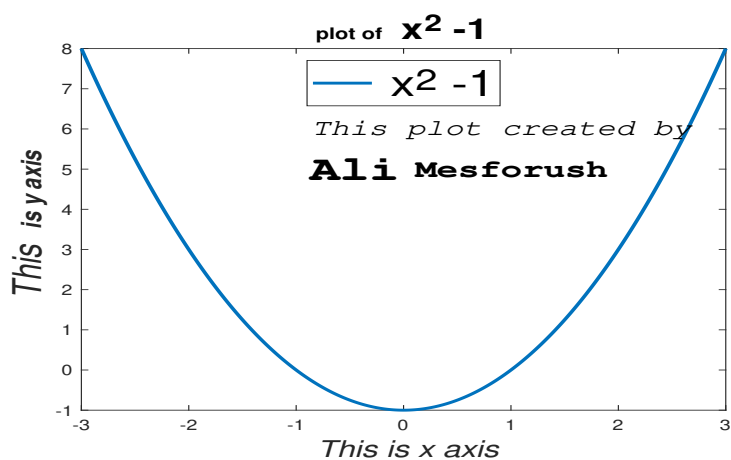
\fontsize{Font Size} این دستور باعث می‌شود تا اندازه فونت مورد استفاده برای متن از جایی که این دستور نوشته شده است تا انتهای متن (یا تا رسیدن به دستور تغییر اندازه فونت بعدی) تغییر کند. توجه کنید که Size Name می‌تواند یک عدد حقیقی مثبت باشد.

👉 توجه کنید که این دستورات نیز باید در میان یک جفت کوتیشن قرار بگیرند.

مثال ۱۸.۶. در کد زیر به چگونگی استفاده از دستورات بیان شده توجه کنید.

```
box off;
x = [-3:0.01:3];
y = x.^2-1;
plot(x,y,'LineWidth',2);
title('\bf{plot of} \fontsize{20} x^2 -1');
xlabel('\fontsize{16} \it{This is x axis}');
ylabel('\fontsize{20} \it{This} \fontsize{14} \bf{is y axis}');
legend('\fontsize{25} x^2 -1','Location','north');
text(-1,6,'\fontname{Yas}\fontsize{16}\it{This plot created by}');
text(-1,5,'\fontname{XB Titre} \fontsize{25} \bf{Ali}');
text(0,5,'\fontname{XB Zar} \fontsize{18} \bf{Mesforush}');
```

خروجی دستورات بالا به‌صورت زیر می‌باشد.



👉 پیشنهاد می‌شود کد بالا را در یک m -فایل بنویسید و با ایجاد تغییراتی در آن خروجی‌های مختلف را بررسی کنید.

علاوه بر موارد بیان شده با استفاده از جدول ۶.۶ نیز می‌توان تغییراتی در نوع فونت، شکل نمایش متن، رنگ متن، اندازه متن و موارد دیگر بوجود آورد.

جدول ۶.۶: گزینه‌های کنترل مشخصات متن

Property Value	شرح عمل	Property Name
اعداد حقیقی	چرخاندن متن با زاویه مشخص	Rotation
normal یا italic	خوابیده یا معمولی بودن متن	FontAngle
یکی از فونتهای نصب شده	تعیین فوع فونت	FontName
کمیت عددی مثبت	اندازه فونت	FontSize
bold، normal، light یا	تعیین شدت رنگ	FontWeight
مقادیر جدول ۲.۶	تعیین رنگ فونت	Color
مقادیر جدول ۲.۶	تعیین رنگ زمینه متن	BackgroundColor
مقادیر جدول ۲.۶	تعیین رنگ دور متن	EdgeColor
کمیت عددی مثبت	تعیین ضخامت رنگ دور متن	LineWidth

نکته عملی

تمام موارد مشخص شده در جدول ۶.۶ باید پس از رشته به شکل زیر استفاده شوند،

```
command(...,'string','Property Name',Property Value)
```

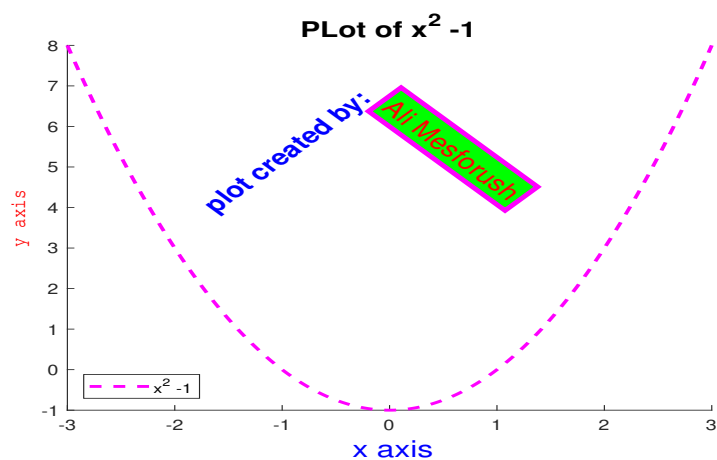
- که در آن command می تواند یکی از دستورات title، xlabel، ylabel، legend، text و gtext باشند.

- Property Name و Property Value باید از جدول ۶.۶ انتخاب شوند.

مثال ۱۹.۶. به چگونگی استفاده از گزینه های اختیاری جدول ۶.۶ در دستورات زیر توجه کنید. پیشنهاد می شود این دستورات را در یک m-فایل بنویسید و با ایجاد تغییرات در آنها خروجی های حاصل را بررسی کنید.

```
x = [-3:0.01:3];
y = x.^2-1;
plot(x,y,'--m','LineWidth',2)
title('Plot of x^2 -1','FontSize',15);
xlabel('\fontsize{16} x axis','color','b');
ylabel('y axis','color','r','FontName','XB Titre');
legend('x^2 -1','Location','southwest');
text(-1.7,4,'\bf{plot crated by:}','rotation',45,...
'FontSize',15,'Color','b');
text(0,6.6,'\fontsize{15}Ali Mesforush','rotation',-45,...
'BackgroundColor','g','EdgeColor','m',...
'LineWidth',3,'color','r');
box off;
```

خروجی این دستورات به صورت شکل زیر می باشد.



استفاده از L^AT_EX در متن

در عنوان شکل، برچسب محورها و نوشته‌های درون شکل می‌توان از برخی از دستورات L^AT_EX استفاده کرد و فرمول‌هایی تولید کرد. استفاده از تمام دستورات L^AT_EX در MATLAB امکان‌پذیر نیست ولی برخی از نمادها و حروف یونانی را دقیقاً با همان دستوراتی که در L^AT_EX تولید می‌شوند، می‌توان مورد استفاده قرار داد^۱. در این بخش هدف آموزش L^AT_EX نیست، لذا تنها به چند مثال از چگونگی استفاده از دستورات L^AT_EX در MATLAB پسندیده می‌کنیم. تمامی دستورات جدول ۷.۶ اگر در

جدول ۷.۶: حروف یونانی قابل استفاده در MATLAB

نماد	دستور	نماد	دستور	نماد	دستور	نماد	دستور
α	<code>\alpha</code>	θ	<code>\theta</code>	o	<code>o</code>	τ	<code>\tau</code>
β	<code>\beta</code>	ϵ	<code>\epsilon</code>	π	<code>\pi</code>	υ	<code>\upsilon</code>
γ	<code>\gamma</code>	ι	<code>\iota</code>	χ	<code>\chi</code>	ϕ	<code>\phi</code>
δ	<code>\delta</code>	κ	<code>\kappa</code>	ρ	<code>\rho</code>	ψ	<code>\psi</code>
ζ	<code>\zeta</code>	ν	<code>\nu</code>	μ	<code>\mu</code>	ω	<code>\omega</code>
η	<code>\eta</code>	ξ	<code>\xi</code>	λ	<code>\lambda</code>	σ	<code>\sigma</code>
Γ	<code>\Gamma</code>	Λ	<code>\Lambda</code>	Σ	<code>\Sigma</code>	Ψ	<code>\Psi</code>
Δ	<code>\Delta</code>	Ξ	<code>\Xi</code>	Υ	<code>\Upsilon</code>	Ω	<code>\Omega</code>
Θ	<code>\Theta</code>	Π	<code>\Pi</code>	Φ	<code>\Phi</code>		

^۱ برای آشنایی با دستورات L^AT_EX می‌توانید به کتاب خودآموز سریع L^AT_EX مراجعه کنید.

در داخل یک جفت کوتیشن و به عنوان رشته در دستوراتی که برای تولید عنوان و برجسب گذاری و ... بیان کردیم، بکار بروند، منجر به تولید حروف یونانی خواهند شد.

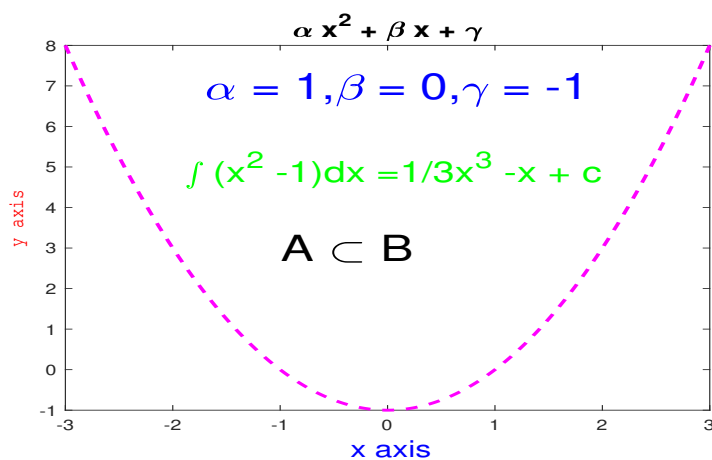
برخی دیگر از نمادها و علایمی که \LaTeX تولید می‌شوند نیز در MATLAB قابل تولید هستند ولی در این بخش به آنها اشاره‌ای نخواهد شد. پیشنهاد می‌شود اگر خوانندگان گرامی نیاز به برخی نمادهای ریاضی در نمودارهای خود شدند با مراجعه به کتاب‌های آموزش \LaTeX مانند کتاب خودآموز سریع \LaTeX ، دستور متناظر با نماد مورد نظر خود را پیدا کنند و در محیط MATLAB بکار ببرید. توجه کنید که همه نمادهای \LaTeX در MATLAB قابل استفاده نیستند.

مثال ۲۰.۶. به چگونگی استفاده از برخی از دستورات \LaTeX در دستورات زیر دقت کنید.

در دستورات زیر از رنگ‌های مختلف استفاده شده است ولی به دلیل چاپ سیاه و سفید کتاب، رنگ‌ها قابل تشخیص نیستند، لذا پیشنهاد می‌شود کدهای نوشته شده را در MATLAB اجرا کنید و با ایجاد تغییرات در آن خروجی‌های مختلف را بررسی کنید.

```
x = [-3:0.01:3];
y = x.^2-1;
plot(x,y,'--m','LineWidth',2)
title('\alpha x^2 + \beta x + \gamma','FontSize',15);
xlabel('\fontsize{16} x axis','color','b');
ylabel('y axis','color','r','FontName','XB Titre');
text(-1.7,7,'\alpha = 1,\beta = 0,\gamma = -1'...
,'FontSize',25,'color','b');
text(-1.9,5,'\int (x^2 -1)dx =1/3x^3 -x + c'...
,'FontSize',20,'color','g');
text(-1,3,'A \subset B','FontSize',25,'Color','k');
```

خروجی دستورات بالا به شکل زیر است.



۳.۳.۶ ایجاد تغییرات در محورهای مختصات

در MATLAB و در هنگام استفاده از دستور plot محورهای مختصات بر اساس دامنه مقادیر x و y تولید و مقداردهی می‌شوند. ولی این امکان وجود دارد که محورها را به شکل دلخواه و با مقادیر دلخواه واقع بر آن تولید کنیم.

تولید محورهای مختصات با اندازه دلخواه

با استفاده از دستور axis می‌توان محورهای مختصات را به شکل دلخواه مقداردهی و تولید کرد. این دستور را می‌توان به شکل `axis([xmin,xmax,ymin,ymax])` بکار برد که در آن

- $xmin$ و $xmax$ به ترتیب کمترین و بیشتر مقدار واقع بر محور x ها می‌باشند.
- $ymin$ و $ymax$ به ترتیب کمترین و بیشتر مقدار واقع بر محور y ها می‌باشند.

مثال ۲۱.۶. با دستورات زیر یک نمودار در سه دستگاه مختصات با اندازه‌ای متفاوت برای محور x ها رسم شده است.

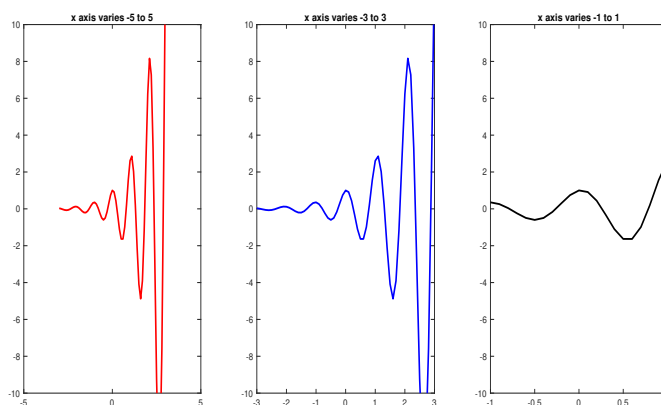
```
x = [-3:0.1:3];
y = exp(x).*cos(6*x);
subplot(1,3,1);
```

```

plot(x,y,'-r','LineWidth',2);
title('x axis varies -5 to 5');
axis([-5,5 -10 10]);
subplot(1,3,2);
plot(x,y,'-b','LineWidth',2);
title('x axis varies -3 to 3');
axis([-3,3 -10 10]);
subplot(1,3,3);
plot(x,y,'-k','LineWidth',2);
title('x axis varies -1 to 1');
axis([-1,1 -10 10]);

```

خروجی دستورات بالا به صورت زیر است.



👉 به محل قرار گرفتن دستور axis که بعد از دستور plot می باشد توجه کنید.

در صورتی که از محورهای مختصات با تقسیم بندی خودکار استفاده شده باشد، می توان با استفاده از چند حالت از پیش تعیین شده تغییراتی در محورهای مختصات ایجاد کرد.

حالات خاص دستور axis

برای دستور axis شش حالت خاص وجود دارد که به سادگی قابل استفاده هستند،

axis normal محورها را به صورت پیش فرض در نظر می گیرد، لذا اگر بخواهیم تغییری در محورها داده نشود بهتر است این گزینه را ننویسیم.

axis equal برای هر دو محور تقسیم بندی یکسانی در نظر می گیرد.

axis tight هر دو محور مختصات را به اطراف داده های نمودار محدود می کند.

axis image کاری مشابه axis equal انجام می دهد، با این تفاوت که چهارچوب اطراف نمودار را کاملاً به نمودار می چسباند.

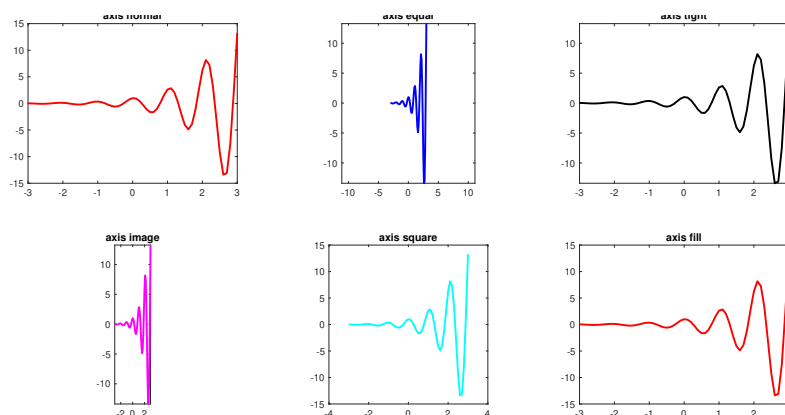
axis square نمودار را در یک چهارچوب مربعی شکل رسم می کند.

axis fill باعث گسترده شدن هر دو محور در سرتاسر بازه و به طور کامل می شود.

مثال ۲۲.۶. دستورات زیر یک نمودار را در شش حالت بیان شده رسم می کنند.

```
x = [-3:0.1:3]; y = exp(x).*cos(6*x);
subplot(2,3,1); plot(x,y,'-r','LineWidth',2);
title('axis normal'); axis normal
subplot(2,3,2); plot(x,y,'-b','LineWidth',2);
title('axis equal'); axis equal;
subplot(2,3,3); plot(x,y,'-k','LineWidth',2);
title('axis tight'); axis tight;
subplot(2,3,4); plot(x,y,'-m','LineWidth',2);
title('axis image'); axis image
subplot(2,3,5); plot(x,y,'-c','LineWidth',2);
title('axis square'); axis square;
subplot(2,3,6); plot(x,y,'-r','LineWidth',2);
title('axis fill'); axis fill;
```

خروجی این دستورات به شکل زیر می باشد. برای یادگیری بیشتر، حتما دستورات را در محیط MATLAB اجرا کنید.



با استفاده از دستور `axis off` می توان محورهای مختصات را حذف کرد. در این صورت تنها نمودار نمایش داده می شود. این دستور را باید بعد از دستور مربوط به رسم نمودار نوشت.

تغییر مقیاس در محورهای مختصات

در حالت عادی مقیاس مورد استفاده در هر دو محور خطی است، ولی در صورت لزوم می توان مقیاس مورد استفاده در هر یک از محورها را به لگاریتمی تغییر داد،

semilogy(x,y) که منجر تغییر محور y ها از خطی به لگاریتمی می گردد.

semilogx(x,y) که منجر تغییر محور x ها از خطی به لگاریتمی می گردد.

loglog(x,y) که منجر تغییر هر دو محور از خطی به لگاریتمی می گردد.

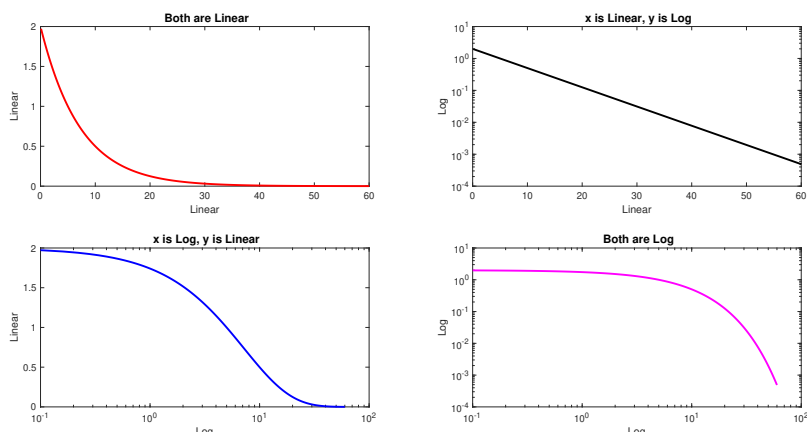
مثال ۲۳.۶. به چهار نمودار زیر که با محورهای مختصات در چهار حالت مخلف رسم شده اند توجه کنید.

```

x = linspace(0.1,60,1000);
y = 2.^(-0.2*x + 1);
subplot(2,2,1); plot(x,y,'r','LineWidth',2);
title('\fontsize{12} Both are Linear');
xlabel('Linear'); ylabel('Linear');
subplot(2,2,2); semilogy(x,y,'-k','LineWidth',2);
xlabel('Linear'); ylabel('Log');
title('\fontsize{12} x is Linear, y is Log');
subplot(2,2,3); semilogx(x,y,'b','LineWidth',2);
xlabel('Log'); ylabel('Linear');
title('\fontsize{12} x is Log, y is Linear');
subplot(2,2,4); loglog(x,y,'-m','LineWidth',2);
xlabel('Log'); ylabel('Log');
title('\fontsize{12} Both are Log')

```

خروجی دستورات بالا به صورت زیر می باشد.



دستور grid

در MATLAB در حالت پیش فرض پس زمینه نمودار شبکه بندی شده نیست، ولی امکان شبکه بندی پس زمینه با دستور grid وجود دارد.

grid on باعث ایجاد شبکه در پس زمینه نمودار می شود.

grid off باعث حذف شبکه در پس زمینه نمودار می شود. حالت پیش فرض رسم نمودارها این وضعیت می باشد.

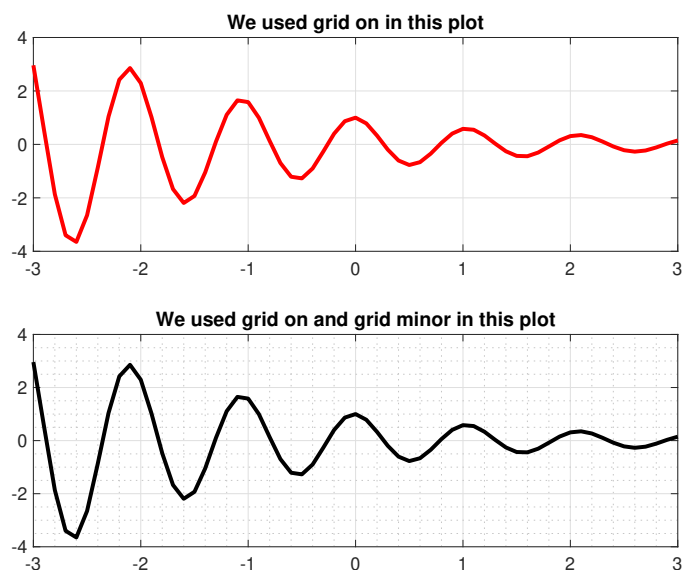
grid minor باعث ریزتر شدن شبکه می شود. این دستور را می توان تنها یا به همراه دستور **grid on** بکار برد.

👉 با دستور **axis on** می توان دوباره محورها را در نمودار ظاهر کرد.

مثال ۲۴.۶. به روش استفاده از دستور grid در کد زیر توجه کنید.

```
x = [-3:0.1:3];
y = exp(-.5*x).*cos(6*x);
subplot(2,1,1);
plot(x,y,'-r','LineWidth',2);
title('We used grid on in this plot');
grid on;
subplot(2,1,2);
plot(x,y,'-k','LineWidth',2);
title('We used grid on and grid minor in this plot');
grid on; grid minor;
```

خروجی دستورات بالا به صورت زیر است. به روش استفاده از دستور subplot برای تولید ستونی نمودارها دقت کنید.



۴.۶ برخی توابع خاص در رسم نمودار

در MATLAB علاوه بر دستورات متداولی که برای رسم نمودارها وجود دارند که در بخش ۱.۶ بیان کردیم، برخی دستورات وجود دارند که موارد استفاده خاصی در رسم نمودارها دارند. در این بخش به برخی از این گونه دستورات می‌پردازیم.

دستور fill

با دستور fill می‌توان نموداری تولید کرد و آن را رنگ‌آمیزی نمود. شکل کلی این دستور به صورت $\text{fill}(x,y,C)$ می‌باشد که در آن بردار x و y دو بردار هستند که به‌شکلی که در دستور plot بیان کردیم تولید می‌شوند و C رنگ مورد نظر است که به شکل جدول ۲.۶ مورد استفاده قرار می‌گیرد.

مثال ۲۵.۶. به دو نمودار تولید شده با دستور fill در کد زیر دقت کنید.

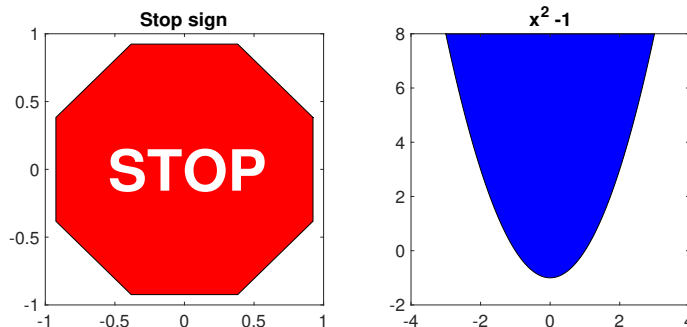
```
t = (1/16:1/8:1)'*2*pi;
```

```

x = cos(t); y = sin(t);
subplot(1,2,1); fill(x,y,'r');
title('Stop sign')
text(-0.55,0,'\bf{STOP}','FontSize',30,'color','w')
axis square
x = -3:0.01:3; y = x.^2 -1;
subplot(1,2,2); fill(x,y,'b');
title('x^2 -1')
axis square

```

خروجی کد بالا به شکل زیر می باشد.



👉 دستور fill شکل‌های استفاده دیگری نیز دارد. برای اطلاعات بیشتر در پنجره فرمان MATLAB دستور help fill را اجرا کنید تا به شکل کلی و کاربردهای این دستور دسترسی پیدا کنید.

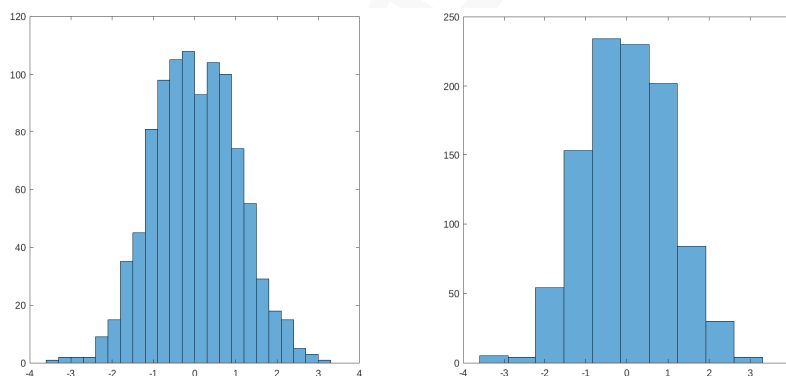
دستور histogram

با دستور histogram می‌توان نموداری تولید کرد که فراوانی داده‌ها را در بازه‌های مختلف نشان می‌دهد. این دستور یک بردار را به عنوان ورودی دریافت می‌کند و با توجه به مقادیر بردار، نموداری تولید می‌کند که نشان دهنده فراوانی عناصر بردار ورودی در بازه‌های مختلف می‌باشد. دستور histogram دارای گزینه‌های زیادی است که در ادامه با برخی از آنها آشنا خواهیم شد.

مثال ۲۶.۶. در دستورات زیر با دستور `randn` یک بردار با مقادیری که دارای توزیع نرمال هستند ایجاد شده است و هیستوگرام آن به دو شکل متفاوت تولید شده است.

```
x = randn(1000,1);
subplot(1,2,1); h = histogram(x); axis square;
nbins = 10;
subplot(1,2,2); h = histogram(x,nbins); axis square;
```

خروجی کد بالا به صورت زیر است.



👉 در تصویر سمت چپ، تعداد ستون‌ها به صورت خودکار تنظیم شده است ولی در تصویر سمت راست، تعداد ستون‌ها^۱، که با `nbins` مشخص شده است، توسط کاربر وارد شده است، که در این مثال عدد ۱۰ می‌باشد. در صورت مشخص کردن تعداد ستون، تقسیم بندی داده‌ها به شکلی انجام می‌شود که تعداد کل بازه‌ها برابر با عدد مشخص شده باشد.

^۱ این ستون‌ها bin نامیده می‌شوند.

نکته عملی

پس از اجرای دستور `histogram` در متغیر `h` مقادیری ذخیره می‌شوند که به صورت `h.Attribute` قابل دسترسی هستند. `Attribute` می‌تواند یکی از مقادیر زیر باشد:

Data یک بردار هم‌اندازه و هم مقدار با بردار ورودی می‌باشد.

Values برداری است که شامل تعداد عناصر بردار ورودی در هر یک از زیربازه‌های مورد استفاده می‌باشد.

NumBins یک کمیت اسکالری است که در آن تعداد `bin`های مورد استفاده ذخیره می‌شود.

BinEdges برداری است که طول آن `NumBins+1` بوده و شامل نقاط مرزی بازه‌های مورد استفاده در هیستوگرام است.

BinWidth یک کمیت اسکالری و شامل طول هر بازه مورد استفاده می‌باشد.

BinLimits یک بردار دوتایی است که شامل نقاط ابتدا و انتهای واقع بر محور افقی است که در هیستوگرام مورد استفاده قرار گرفته است.

FaceColor یک رشته است که رنگ مورد استفاده در رنگ‌آمیزی هیستوگرام در آن ذخیره شده است. حالت پیش‌فرض `auto` می‌باشد.

EdgeColor یک بردار سه‌تایی است که مشخص کننده رنگ مورد استفاده برای رسم خطوط هیستوگرام می‌باشد. حالت پیش‌فرض بردار `[0 0 0]` است که به معنای رنگ سیاه است.

📖 مقادیر دیگری نیز وجود دارد که در صورت تمایل می‌توانید با دستور `help histogram` آنها را مشاهده کنید.

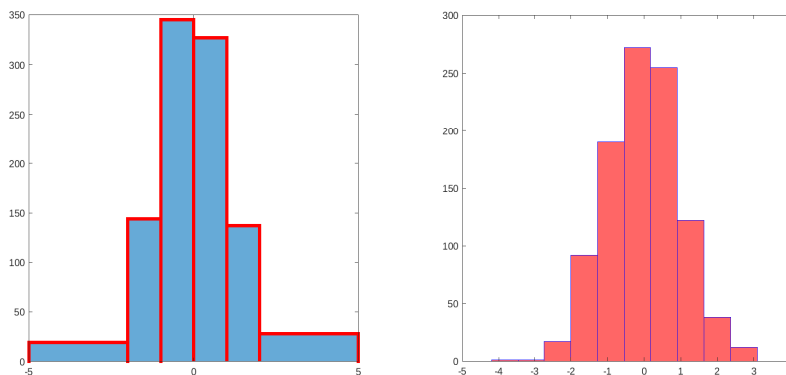
📖 برخی از مقادیری که در بالا بیان شدند قابل تغییر می‌باشند. برای ایجاد تغییر در رنگ نمودار یا رنگ خطوط کافیه نام ویژگی را در میان یک جفت کوتیشن بنویسیم و پس از آن رنگ مورد نظر را براساس جدول ۲.۶ مشخص کنیم.

مثال ۲۷.۶. به چگونگی ایجاد تغییر رنگ در هیستوگرام و خطوط توجه کنید. همچنین به روش ایجاد تغییر در `BinEdges` نیز دقت کنید. به این منظور در هنگام اجرای دستور `histogram`

یک بردار به عنوان بردار معرف عناصر مرزی محور افقی مورد استفاده قرار گرفته است.

```
x = randn(1000,1);
subplot(1,2,1);
h = histogram(x,[-5 -2:1:2 ,5], 'EdgeColor','r','LineWidth',2);
axis square;
nbins = 10;
subplot(1,2,2);
h = histogram(x,nbins,'FaceColor','r','EdgeColor','b');
axis square
```

خروجی دستورات بالا به صورت زیر می باشد.



👉 در دستور اول با استفاده از گزینه `LineWidth` که پیشتر بیان کردیم ضخامت خط هیستوگرام را تغییر دادیم. شایان ذکر است که گزینه های دیگری که برای دستورات مربوط به رسم نمودارها بیان کردیم در دستور `histogram` نیز کار می کنند. همچنین امکان استفاده از دستورات عنوان گذاری و برچسب گذاری نیز وجود دارد.

رسم نمودار هیستوگرام داده‌های طبقه‌بندی شده

با استفاده از دستور `histogram` می‌توان داده‌ها را به شکل طبقه‌بندی شده و به صورت نمودار ستونی تولید کرد. به این منظور باید،

- داده‌ها را به صورت یک بردار سطری معرفی کرد.

- با استفاده از دستور `categorical` و به صورت زیر طبقات را مشخص کرد،

```
C = categorical(Data,[items],{'captions'})
```

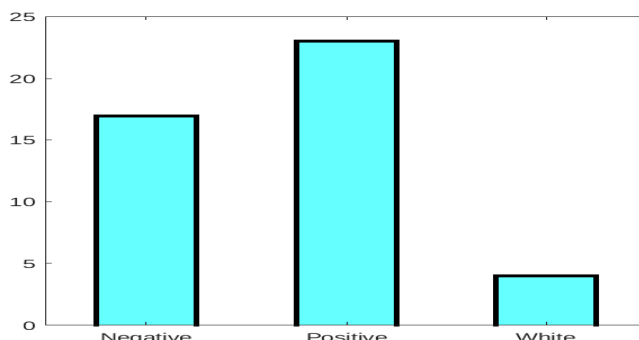
که در آن باید آیتم‌ها را با یک فاصله از هم و در داخل یک جفت کروشه نوشته و به تعداد آیتم‌ها باید برچسب‌ها را به صورت رشته در داخل یک جفت آکولاد مشخص کرد.

- با دستور `histogram(C)` نمودار مطلوب رسم می‌شود.

مثال ۲۸.۶. به چگونگی رسم هیستوگرام مربوط به یک نظرسنجی که به صورت برداری شامل ۰ برای رای منفی، ۱ برای رای مثبت و NaN برای رای سفید، می‌باشد دقت کنید.

```
A = [1 1 0 0 1 0 NaN 1 1 1 0 0 0 0 1 ...
NaN 0 1 0 1 0 1 0 1 1 1 NaN 1 1 ...
0 0 0 1 1 1 NaN 1 1 1 0 1 0 1 0];
C = categorical(A,[0 1 NaN],{'Negative','Positive','White'});
histogram(C,'BarWidth',0.5,'FaceColor','c','LineWidth',2);
axis square;
```

در دستور نوشته شده برای رسم هیستوگرام از گزینه `BarWidth` برای تعیین پهنای ستون‌ها و از گزینه `LineWidth` برای تعیین ضخامت خط در هیستوگرام استفاده شده است. استفاده شده است. خروجی دستورات بالا به صورت هیستوگرام زیر می‌باشد.



در مورد دستور histogram قابلیت‌های دیگری نیز وجود دارد که از حوصله کتاب این خارج است، لذا خوانندگان گرامی در صورت نیاز می‌توانند با مراجعه به help نرم‌افزار MATLAB نیازهای خود را رفع کنند.

دستور bar

در MATLAB برای رسم نمودارهای میله‌ای دستور ساده‌ای وجود دارد. دستور bar در ساده‌ترین شکل خود به صورت `bar(x,y,Options)` می‌باشد که در آن، `x` و `y` دو بردار هم‌اندازه می‌باشند، و `Options` گزینه‌هایی اختیاری برای کنترل ظاهری نمودار خروجی است. برخی از این گزینه‌ها عبارتند از:

width یک عدد حقیقی مثبت است که معمولاً بین صفر تا یک انتخاب می‌شود و میزان پهنای هر ستون را تعیین می‌کند. مقدار پیش‌فرض عدد 0.8 می‌باشد.

color یکی از کاراکترهای جدول ۲.۶ است که رنگ ستون‌ها را تعیین می‌کند.

style یکی از رشته‌های زیر است که تعیین کننده شکل ستون‌ها است.

'grouped', 'stacked', 'histc', 'hist'

حالت پیش‌فرض grouped می‌باشد.

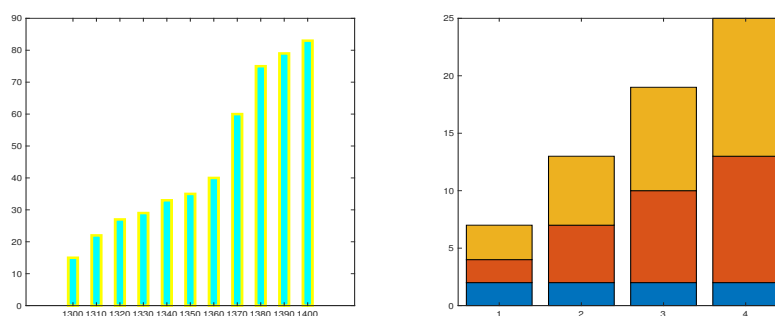
مثال ۲۹.۶. نمودار میله‌ای مربوط به جمعیت ایران از سال ۱۳۰۰ تا ۱۴۰۰ را با مقادیر فرضی می‌توان به شکل زیر تولید کرد. نمودار سمت راست با استایل stacked رسم شده است.

```

x = 1300:10:1400;
y = [15 22 27 29 33 35 40 60 75 79 83];
subplot(1,2,1);
bar(x,y,0.4,'c','EdgeColor','y','LineWidth',3);
axis square
y = [2 2 3; 2 5 6; 2 8 9; 2 11 12];
subplot(1,2,2);
bar(y,'stacked');
axis square;

```

خروجی کد بالا به صورت زیر است. چون تمامی خروجی های زیر رنگی است، لذا پیشنهاد می شود تا کدهای نوشته شده را در یک MATLAB بنویسید و اجرا کنید.



برای تولید نمودارهای گروهی، مشابه نمودار سمت راست شکل بالا، به مثال زیر توجه کنید.

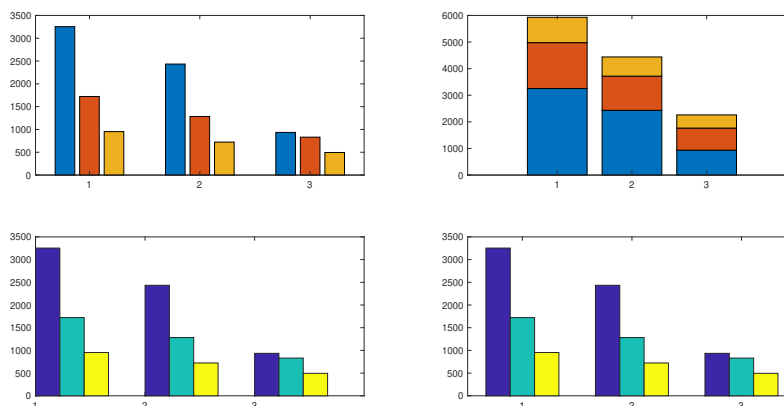
مثال ۳۰.۶. فرض کنید جدول زیر را برای تعداد دانشجویان در سه مقطع کارشناسی، کارشناسی ارشد و دکتری برای سه دانشگاه تهران، مشهد و شاهرود را در اختیار داریم. هدف تولید نمودار میله ای برای مقایسه ورودی های این سه دانشگاه می باشد.

کارشناسی	کارشناسی ارشد	دکتری	
۳۲۵۴	۱۷۲۳	۹۵۳	دانشگاه تهران
۲۴۳۴	۱۲۸۳	۷۲۳	دانشگاه فردوسی مشهد
۹۳۵	۸۳۱	۴۹۵	دانشگاه صنعتی شاهرود

با کد زیر و به چهار شکل مختلف نمودار میله‌ای را تولید کرده‌ایم.

```
y = [3254,1723,953
      2434,1283,723
      935,831,495];
subplot(2,2,1); bar(y,'grouped');
subplot(2,2,2); bar(y,'stacked');
subplot(2,2,3); bar(y,'histc');
subplot(2,2,4); bar(y,'hist');
```

خروجی حاصل به شکل زیر می‌باشد. حتماً کد را در محیط MATLAB اجرا کنید تا رنگ‌ها را به درستی مشاهده کنید.

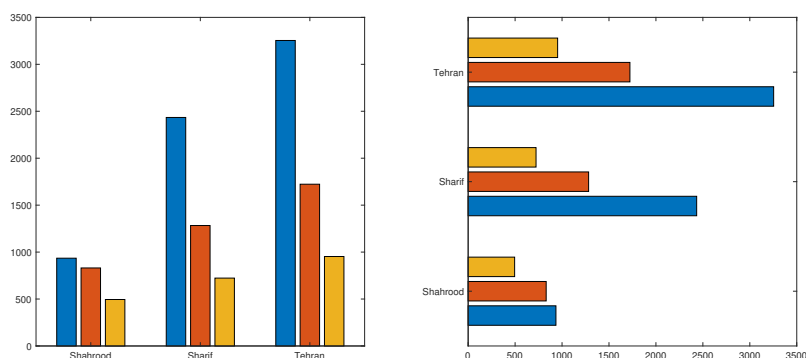


حال اگر بخواهیم در زیر ستون‌های مربوط به هر دانشگاه اسامی دانشگاه‌ها نوشته شود، می‌توان به روش زیر عمل کرد.

مثال ۳۱.۶. به روش ایجاد نوشتار در زیر نمودارهای میله‌ای توجه کنید.

```
y = [3254,1723,953;2434,1283,723;935,831,495];
c = categorical({'Tehran','Sharif','Shahrood'});
subplot(1,2,1); bar(c,y,'grouped'); axis square;
subplot(1,2,2); barh(c,y); axis square;
```

خروجی، کد بالا به صورت زیر می باشد. به صورت زیر می باشد.



👉 در دستور مربوط به نمودار سمت راست، بجای `bar` از دستور `barh` استفاده کردیم که منجر رسم نمودار میله‌ای به شکل افقی شد.

دستور pie

شکل کلی دستور `pie(X, explode, label)` است که در آن

X برداری است که می‌خواهیم نمودار دایره‌ای براساس داده‌های آن ساخته شود.

label مجموعه‌ای از رشته‌هاست که تعداد آن با تعداد عناصر بردار `X` برابر می‌باشد. عناصر این مجموعه باید بین دو آکولاد و به صورت رشته نوشته شوند و با کاما از هم جدا شوند.

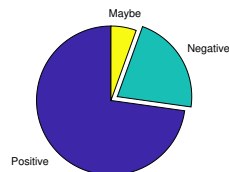
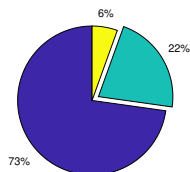
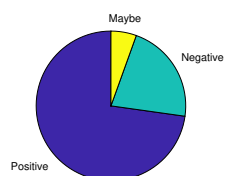
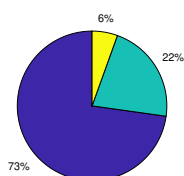
explode برداری است هم‌اندازه با بردار `x` که شامل صفر و یک است. عنصری از این بردار که مقدار یک داشته باشد، در نمودار دایره‌ای، عنصر متناظر با آن از نمودار دایره‌ای جدا می‌شود.

مثال ۳۲.۶. فرض کنید در یک نظرسنجی تعداد ۲۳۵ نفر رای مثبت، ۷۱ نفر رای منفی و ۱۸ نفر رای سفید داده‌اند. نمودار دایره‌ای این نظرسنجی را می‌توان با دستورات زیر رسم کرد.

```
X = [238,71,18];
subplot(2,2,1); pie(X); axis square
label = {'Positive','Negative','Maybe'};
```

```
subplot(2,2,2); pie(X,label); axis square;
explode = [0,1,0];
subplot(2,2,3); pie(X,explode); axis square;
subplot(2,2,4); pie(X,explode,label); axis square
```

خروجی دستورات بالا به صورت زیر می باشد. به دلیل وجود رنگ بندی در شکل حاصل، حتما دستورات را در محیط MATLAB اجرا کنید.



۵.۶ تمرین

تمرین ۱.۶. نمودار توابع زیر را در بازه تعیین شده رسم کنید.

$$f(x) = \frac{(x+5)^2}{4+3x^2}, \quad -3 \leq x \leq 5, \quad \bullet$$

$$f(x) = \frac{\Delta \sin x}{x + e^{-\gamma/\Delta x}} - \frac{2x}{\Delta}, \quad -5 \leq x \leq 10, \quad \bullet$$

$$f(x) = (x+1)(x-2)(2x-0.25) - e^x, \quad 0 \leq x \leq 3, \quad -3 \leq x \leq 6. \quad \bullet$$

تمرین ۲.۶. با استفاده از دستور fplot نمودار توابع زیر را در بازه داده شده رسم کنید.

$$f(x) = \sqrt{|\cos 3x|} + \sin^2 4x, \quad -2 \leq x \leq 2, \quad \bullet$$

$$f(x) = \Delta e^{\gamma \sin 0.4x} \cos 4x, \quad -20 \leq x \leq 30. \quad \bullet$$

تمرین ۳.۶. معادله پارامتری زیر را در نظر بگیرید،

$$x(t) = 1/5 \sin 5t, \quad y(t) = 1/5 \cos 3t,$$

نمودار این تابع را برای $0 \leq t \leq 2\pi$ رسم کنید. نمودار را به شکلی قالب بندی کنید که هر دو محور در بازه $[-2, 2]$ قرار بگیرند.

تمرین ۴.۶. نمودار تابع $f(x) = \frac{x^2 + 3x + 3}{x^2 + 1}$ را برای $-4 \leq x \leq 3$ رسم کنید. توجه کنید که نمودار در $x = -1$ دارای مجانب قائم است، لذا باید نمودار در دو بازه $[-4, -1/2]$ و $[1/2, 3]$ رسم کرد.

تمرین ۵.۶. نمودار تابع پارامتری تعریف شده با

$$x(t) = \frac{3t}{1+t^3}, \quad y(t) = \frac{3t^2}{1+t^3},$$

را در نظر بگیرید و نمودار آن را رسم کنید. توجه کنید که مخرج در $t = -1$ مقدار صفر دارد.

تمرین ۶.۶. نمودار تابع $y = \frac{x^2 - 4x - 7}{x^2 - x - 6}$ را برای $-6 \leq x \leq 6$ رسم کنید. توجه کنید که نمودار این تابع دارای دو مجانب قائم می باشد.

تمرین ۷.۶. نمودار حاصل از اثر یک نقطه واقع بر یک دایره متحرک بر صفحه، سیکلوئید نامیده می شود. معادلات پارامتری سیکلوئید به شکل زیر می باشند،

$$x(t) = r(t - \sin t), \quad y(t) = r(t - \cos t).$$

نمودار سیکلوئید را برای $0 \leq t \leq 4\pi$ و به ازای $r = 1/5$ رسم کنید.

تمرین ۸.۶. نمودار تابع $f(x) = \cos x \sin 2x$ و مشتق آن را در یک نمودار و در بازه $[-\pi, \pi]$ رسم کنید. تابع را با خط معمولی، رنگ قرمز و ضخامت ۲ رسم کنید و نمودار مشتق را به شکل نقطه چین و رنگ آبی رسم کنید. محورها را نام گذاری کنید و راهنمای نمودار را نیز تهیه کنید.

تمرین ۹.۶. تابع مثلثاتی $\sin x$ را به شکل سری مک لورن زیر می توان بیان کرد:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} \dots$$

نمودار تابع $\sin x$ را هم به شکل مستقیم و هم با استفاده از سری در یک بازه مناسب و در یک نمودار برای n های مختلف رسم کنید. همین کار را برای توابع $\cos x$ و e^x نیز که دارای سری های مک لورن به شکل زیر می باشند نیز انجام دهید.

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots,$$

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} \dots$$

نسخه
رایگان

۷ چندجمله‌ای‌ها، برازش منحنی و درونیابی

چندجمله‌ای‌ها عبارات ریاضی هستند که کاربردهای گسترده‌ای در علوم و مهندسی دارند. در MATLAB توابع و دستورات زیادی وجود دارند که به کمک آنها می‌توان چندجمله‌ای‌ها را تولید کرد و عملیات مختلفی روی آنها انجام داد. در بخش اول این فصل به معرفی این دستورات می‌پردازیم. در بخش دوم این فصل به معرفی دستورات مرتبط با برازش منحنی‌ها خواهیم پرداخت. منظور از برازش یک منحنی، یافتن یک چندجمله‌ای‌ای است که نمودار آن با کمترین خطای ممکن بر نمودار یک تابع منطبق شود. در انتهای این فصل به استفاده از MATLAB در درونیابی خواهیم پرداخت. در درونیابی با معلوم بودن مختصات تعدادی نقطه در صفحه، یک چندجمله‌ای خواهیم یافت تا از تمامی نقاط مفروض بگذرد، سپس به کمک چندجمله‌ای یافته شده به برآورد مقدار تابع، که ضابطه آن معلوم نیست، در نقاط دیگری به جز نقاط مفروض خواهیم پرداخت.

۱.۷ چندجمله‌ای‌ها

یک چندجمله‌ای در حالت کلی به صورت

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0,$$

می‌باشد. در MATLAB به سادگی می‌توان چندجمله‌ای‌ها را تعریف کرد و عملیات گوناگون را بروی آنها انجام داد. این بخش از کتاب به این موضوع می‌پردازد. برای یادگیری بهتر پیشنهاد می‌شود تا تمامی دستوراتی و مثال‌هایی که در این بخش آورده می‌شود را در محیط MATLAB اجرا کنید.

۱.۱.۷ تعریف چندجمله‌ای‌ها

در MATLAB چندجمله‌ای‌ها به‌سادگی و با استفاده از آرایه‌ها قابل تعریف هستند.

تعریف چندجمله‌ای‌ها

یک چندجمله به‌شکل کلی

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

را می‌توان به‌صورت آرایه‌ای از ضرایب توان‌های x از زیاد به کم نوشت. برای مثال چندجمله‌ای $P(x) = 8x^5 - 3x^4 + 2x^2 - x + 5$ را به‌شکل زیر تعریف می‌کنیم.

$$P = [8, -3, 0, 2, -1, 5]$$

مثال ۱.۷. به چگونگی تعریف چندجمله‌ای‌های

$$p(x) = x + 2, \quad q(x) = \sqrt{2}x^3 - x^2 + \frac{1}{2}x - 3, \quad r(x) = \ln 5 x^4 + \pi^2 x^3 - 3x^2 + \sin 3$$

در دستورات زیر دقت کنید.

```
p = [1 2];
q = [sqrt(2) -1 1/2 -3];
r = [log(5) pi^2 -3 0 sind(30)];
```

۲.۱.۷ محاسبه مقدار چندجمله‌ای

پس از تعریف چندجمله‌ای به‌صورتی که بیان کردیم و برای محاسبه مقدار یک چندجمله‌ای برای یک مقدار مشخص، می‌توان از دستور polyval استفاده کرد.

دستور polyval

شکل کلی دستور polyval به صورت $\text{polyval}(p,v)$ است که در آن

p آرایه‌ای است که به عنوان چندجمله‌ای معرفی شده است.

v مقداری است که می‌خواهیم چندجمله‌ای به ازای آن محاسبه شود.

مثال ۲.۷. نمودار توابع زیر را در دو دستگاه مختصات جداگانه رسم کنید.

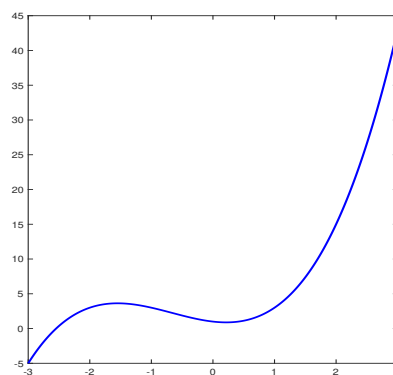
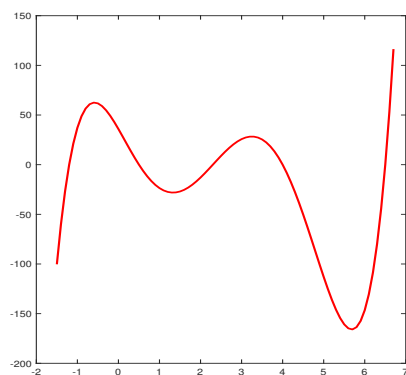
$$f(x) = x^5 - 12.1x^4 + 40.59x^3 - 17.015x^2 - 71.95x + 35.88 + 4x - 1,$$

$$g(x) = x^3 + 2x^2 - x + 1$$

به روش استفاده از polyval در دستورات زیر توجه کنید.

```
f = [1 -12.1 40.59 -17.015 -71.95 35.88];
x = -1.5:0.1:6.7; y = polyval(f,x);
subplot(1,2,1); plot(x,y,'-r','Linewidth',2);
g = [1,2,-1,1];
x1 = -3:0.01:3; y1 = polyval(g,x1);
subplot(1,2,2); plot(x1,y1,'-b','LineWidth',2);
```

خروجی حاصل به صورت زیر می‌باشد.



۳.۱.۷ ریشه چندجمله‌ای‌ها

یکی از مواردی که در هنگام کار با چندجمله‌ای‌ها بسیار مورد نیاز می‌باشد، ریشه‌های یک چندجمله‌ای است. در MATLAB امکان محاسبه تمام ریشه‌های یک چندجمله‌ای با دستور roots امکان پذیر می‌باشد.

دستور roots

برای محاسبه ریشه‌های یک چندجمله‌ای کافیه پس از تعریف چندجمله‌ای از دستور roots(p) استفاده کنید. در این صورت یک بردار n تایی برگشت داده می‌شود که در آن تمام ریشه‌های حقیقی و مختلط چندجمله‌ای p قرار گرفته است. توجه کنید که n درجه چندجمله‌ای می‌باشد.

مثال ۳.۷. ریشه‌های دو چندجمله‌ای

$$2x^2 + 3x + 1 = 0, \quad 4x^5 + 3x^4 - x^3 + 2x^2 + 1 = 0,$$

با دستورات زیر بدست می‌آیند.

ورودی

```
%2x^2 + 3x + 1 = 0
p = [2,3,1];
r = roots(p)
%4x^5+3x^4-x^3+2x^2+1=0
q = [4 3 -1 2 0 1];
s = roots(q)
```

خروجی

```
r =
-1.0000
-0.5000
s =
-1.3138 + 0.0000i
0.4853 + 0.5324i
0.4853 - 0.5324i
-0.2034 + 0.5704i
-0.2034 - 0.5704i
```

👉 در دستورات بالا از کاراکتر درصد برای کامنت گذاری استفاده است، به این معنا که عبارات نوشته شده بعد از علامت درصد اجرا نخواهد شد.

👉 اگر ریشه‌های یک چندجمله‌ای معلوم باشند، با دستور poly(r) ضرایب چندجمله‌ای به صورت یک آرایه تولید می‌شود. برای مثال اگر مقادیر ۱ و ۲ و ۳ ریشه‌های یک چندجمله‌ای باشند آنگاه

ورودی

```
r = [1,2,3];
p = poly(r)
```

خروجی

```
p =
1    -6    11   -6
```

که بردار p نشان دهنده چندجمله‌ای $p(x) = x^3 - 6x^2 + 11x - 6$ می‌باشد.

۴.۱.۷ عملیات جبری روی چندجمله‌ای‌ها

پس از تعریف چندجمله‌ای‌ها در MATLAB، روی چندجمله‌ای‌ها می‌توان عملیات جبری، یعنی جمع، تفریق، ضرب و تقسیم انجام داد. دو عمل جمع و تفریق چندجمله‌ای‌ها دقیقاً مانند جمع و تفریق آرایه‌ها انجام می‌شود، لذا تنها به ذکر یک مثال ساده در این زمینه بسنده می‌کنیم.

مثال ۴.۷. مجموع و تفاضل دو چندجمله‌ای زیر را محاسبه کنید،

$$p(x) = x^6 + 2x^5 - 3x^4 + 2x^2 - x + 1, \quad q(x) = 3x^3 + 2x^2 - 3x + 5.$$

در دستورات زیر توجه کنید چون درجه دو چندجمله‌ای یکسان نیستند برای توان‌های ۴، ۵ و ۶ ضریب صفر در نظر گرفته شده است.

ورودی

```
p = [1,2,-3,0,2,-1,1];
q = [0,0,0,3,2,-3,5];
pplusq = p + q
pminusq = p - q
```

خروجی

```
pplusq =
1    2   -3    3    4   -4    6
pminusq =
1    2   -3   -3    0    2   -4
```

ولی برای ضرب و تقسیم چندجمله‌ای‌ها دستورات ساده‌ای وجود دارند که در ادامه به معرفی آنها می‌پردازیم.

دستور conv

پس از معرفی دو چندجمله‌ای p و q که باید به صورت آرایه تعریف شوند، می‌توان با استفاده از دستور $\text{conv}(p, q)$ حاصل ضرب دو چندجمله‌ای را بدست آورد.

مثال ۵.۷. حاصل ضرب دو چندجمله‌ای زیر را محاسبه کنید،

$$p(x) = 3x^4 + 2x^2 - x + 1, \quad q(x) = 3x^2 + 2x - 5.$$

توجه کنید اگر دو چندجمله‌ای هم‌درجه نباشند، نیازی به گذاشتن صفر در آرایه برای هم‌درجه شدن نیست. دستورات زیر حاصل ضرب را محاسبه می‌کنند.

ورودی	خروجی
<pre>p = [3,0,2,-1,1]; q = [3,2,-5]; conv(p,q)</pre>	<pre>ans = 9 6 -9 1 -9 7 -5</pre>

برای محاسبه خارج‌قسمت و باقیمانده حاصل از تقسیم دو چندجمله‌ای بر یکدیگر نیز دستور ساده‌ای وجود دارد.

دستور deconv

از دستور deconv برای محاسبه خارج‌قسمت و باقیمانده تقسیم دو چندجمله‌ای بر یکدیگر می‌توان استفاده کرد. شکل کلی این دستور به‌صورت زیر است،

$$[K,R] = \text{deconv}(p,q)$$

که در آن K و R دو بردار بوده و به‌ترتیب، خارج قسمت و باقیمانده تقسیم p بر q می‌باشند.

مثال ۶.۷. خارج قسمت و باقیمانده حاصل از تقسیم دو چندجمله‌ای زیر را محاسبه کنید،

$$p(x) = 3x^4 + 2x^2 - x + 1, \quad q(x) = 3x^2 + 2x - 5.$$

توجه کنید اگر دو چندجمله‌ای هم‌درجه نباشند، نیازی به گذاشتن صفر در آرایه برای هم‌درجه شدن نیست.

ورودی	خروجی
<pre>p = [3,0,2,-1,1]; q = [3,2,-5]; [K,R] = deconv(p,q)</pre>	<pre>K = 1.0000 -0.6667 2.7778 R = 0 0 0 -9.8889 14.8889</pre>

👉 در مثال بالا، خارج قسمت یک چندجمله‌ای از درجه ۲ و باقیمانده از درجه اول می‌باشد.

۵.۱.۷ مشتق‌گیری از چندجمله‌ای‌ها

علاوه بر دستوراتی که برای مشتق‌گیری از توابع وجود دارند و آینده به آنها خواهیم پرداخت، سه دستور ساده برای مشتق‌گیری از چندجمله‌ای‌ها وجود دارد که در این بخش به معرفی و روش استفاده از آنها می‌پردازیم.

دستور polyder

در حالت کلی برای مشتق‌گیری از چندجمله‌ای‌ها از دستور polyder استفاده می‌شود، ولی با توجه به روش استفاده از این دستور کارایی آن تفاوت خواهد داشت.

u = polyder(p) این روش استفاده منجر به محاسبه مشتق چندجمله‌ای $p(x)$ می‌شود، در این صورت نتیجه به شکل یک چندجمله‌ای می‌باشد که در آرایه u نگهداری شده است.

u = polyder(p,q) این روش استفاده منجر به محاسبه مشتق چندجمله‌ای حاصل ضرب $p(x)q(x)$ می‌شود، در این صورت نتیجه به شکل یک چندجمله‌ای می‌باشد که در آرایه u نگهداری شده است.

[u,v] = polyder(p,q) این روش استفاده منجر به محاسبه مشتق چندجمله‌ای کسر $\frac{p(x)}{q(x)}$ می‌شود، در این صورت نتیجه به شکل دو چندجمله‌ای می‌باشد که صورت در آرایه u و مخرج در آرایه v ذخیره می‌شوند.

مثال ۷.۷. مشتق عبارات زیر را محاسبه کنید.

$$x^3 + 3x^2 - 2x + 1, \quad (x^2 + 1)(x^3 - 2x^2 + 4x - 2), \quad \frac{x^2 + 2x + 3}{x^3 - x - 1}$$

در دستورات زیر به چگونگی استفاده از دستور `polyder` برای محاسبه مشتق عبارت حاصل ضرب و عبارت کسری توجه کنید.

ورودی

```
s = [1,3,-2,1];
ds = polyder(s)
p1 = [1,0,1];
p2 = [1,-2,4,-2];
dp1p2 = polyder(p1,p2)
q1 = [1,2,3];
q2 = [1,0,-1,-1];
[u,v] = polyder(q1,q2)
```

خروجی

```
ds =
3    6   -2
dp1p2 =
5   -8   15   -8    4
u =
-1   -4  -10   -2    1
v =
1    0   -2   -2    1    2    1
```

۲.۷ برازش منحنی

فرض کنید تعدادی نقطه در صفحه داده شده باشند که دارای نظم خاصی نیستند. هدف از برازش منحنی یافتن یک تابع می‌باشد که یا از تمامی نقاط بگذرد یا از میان نقاط به گونه‌ای عبور کند که منحنی تابع یافته شده کمترین فاصله با نقاط را داشته باشد. تابع مورد نظر می‌تواند خطی، چندجمله‌ای، نمایی و یا از انواع دیگری باشد که در این بخش درباره آن بیشتر خواهیم گفت.

۱.۲.۷ برازش منحنی با چندجمله‌ای‌ها

با استفاده از چندجمله‌ای‌ها می‌توان برازش را به دو صورت انجام داد:

- تابع یافته شده از تمامی نقاط داده شده بگذرد،
- تابع یافته شده از تمام نقاط داده شده نگذرد ولی تقریب قابل قبولی برای داده‌ها ارائه نماید.

در ادامه روش یافتن تابع مورد نظر به هر دو روش می‌پردازیم.

دستور polyfit

در حالت کلی اگر n نقطه به صورت

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n),$$

داده شده باشند با دستور polyfit به شکل کلی $z = \text{polyfit}(x, y, m)$ می توان یک چندجمله ای از درجه m یافت که نقاط داده شده را برازش کند. در این دستور

x برداری شامل تمام مولفه های اول نقاط داده شده می باشد.

y برداری شامل تمام مولفه های دوم نقاط داده شده می باشد.

در این دستور m می تواند عددی بین ۱ تا $n-1$ باشد.

❏ اگر بخواهیم چندجمله ای یافته شده از تمامی نقاط بگذرد باید m را برابر با $n-1$ انتخاب کرد.

در غیر این صورت چند جمله ای از میان نقاط عبور خواهد کرد.

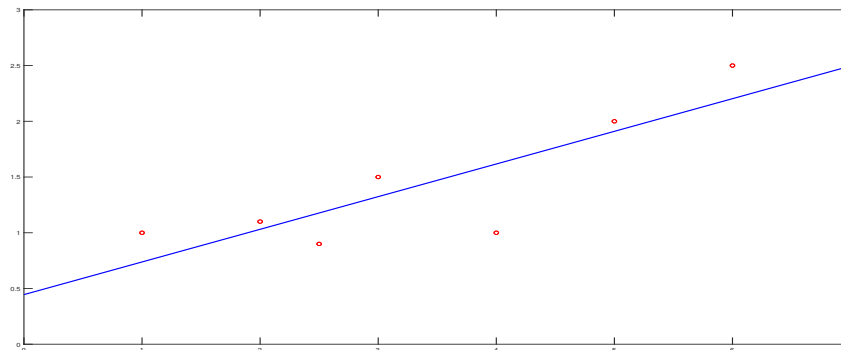
ابتدا یک مثال ساده در زمینه استفاده از دستور polyfit می زنیم، سپس به بیان مثالی خواهیم پرداخت که نیاز به اندکی برنامه نویسی دارد.

مثال ۸.۷. یک خط بیابید که مجموعه نقاط زیر را برازش کند.

$$(1, 1), (2, 1/1), (2.5, 0.9), (3, 1/5), (4, 1), (5, 2), (6, 2/5).$$

خط حاصل را به همراه نقاط در یک نمودار رسم کنید.

```
x = [1,2,2.5,3,4,5,6];
y = [1,1.1,0.9,1.5,1,2,2.5];
plot(x,y,'ro'); axis([0 7 0 3]);
z = polyfit(x,y,1)
x1 = 0:0.1:7; w = polyval(z,x1);
hold on;
plot(x1,w,'-b','LineWidth',2);
```



👉 توجه کنید که برای رسم خط از دستور polyval به همراه تقسیم‌بندی جدید x1 برای یافتن بردار y استفاده کردیم.

👉 مقدار z یک بردار دوتایی است که عنصر اول آن ضریب m و عنصر دوم آن مقدار ثابت b در معادله خط $y = mx + b$ می‌باشد.

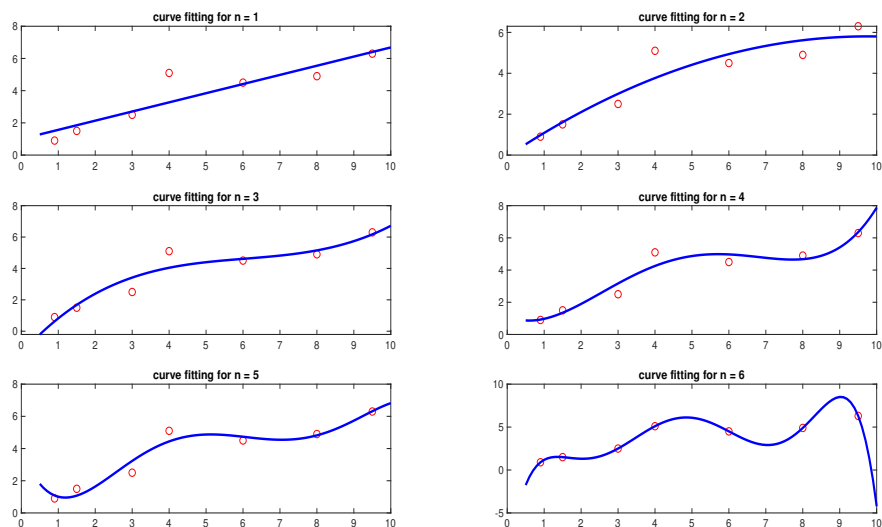
مثال ۹.۷. نقاط زیر را در نظر بگیرید،

$$(0.9, 0.9), (1.5, 1.5), (3, 2.5), (4, 5.1), (6, 4.5), (8, 4.9), (9.5, 6.3).$$

چندجمله‌هایی از درجه‌های اول تا ششم بسازید که داده‌های بالا را برازش کنند. هر چندجمله‌ای را در نمودار جداگانه‌ای رسم کنید. برنامه زیر را در یک m-فایل بنویسید و اجرا کنید.

```
x = [0.9,1.5,3,4,6,8,9.5];
y = [0.9,1.5,2.5,5.1,4.5,4.9,6.3];
x1 = 0.5:0.1:10;
for m=1:6
    z = polyfit(x,y,m);w = polyval(z,x1);
    subplot(3,2,m); plot(x,y,'ro'); hold on;
    plot(x1,w,'-b','Linewidth',2);
    title(['curve fitting for m = ',num2str(m)]);
end
```

خروجی برنامه زیر به شکل زیر می باشد.



👉 در دستور `title` در برنامه بالا از دستور `num2str` برای تبدیل یک کمیت عددی به رشته استفاده شده است.

👉 در نمودارهای بالا تنها برای $m = 6$ نمودار دقیقاً از تمام نقاط می گذرد و برای مقادیر دیگر نمودار، تقریبی برای داده ها می باشد.

👉 به چگونگی استفاده از دستور `polyval` برای تولید بردار w بر اساس بردار $x1$ توجه کنید. با این روش امکان رسم نمودار هموارتری فراهم شد. در صورت استفاده از دو بردار x و y نمودار حاصل به صورت خط های شکسته حاصل می شد.

👉 به چگونگی استفاده از حلقه `for` در این برنامه توجه کنید. اگر از حلقه استفاده نمی کردیم ناچار بودیم دستورات داخل حلقه را شش بار بازنویسی کنیم.

۲.۲.۷ برازش منحنی با توابع دیگر

با استفاده از دستور `polyfit` این امکان وجود دارد تا نقاط مفروضی را با توابع دیگری به جز چندجمله ای ها برازش نمود. شکل کلی دستور به همان صورتی است که در بخش ۱.۲.۷ بیان شد ولی در آرگومان های ورودی تفاوت کوچکی دارد که در ادامه به آن خواهیم پرداخت. شکل تمام برازش های ممکن در جدول ۱.۷ آورده شده است.

جدول ۱۰.۷: برازش با توابع غیر چندجمله‌ای در MATLAB

نام	تابع مورد استفاده	دستور
برازش توانی	$y = b^x$	<code>polyfit(log(x),log(y),1)</code>
برازش نمایی	$y = be^{mx}$ $y = b10^{mx}$	<code>polyfit(x,log(y),1)</code> <code>polyfit(x,log10(y),1)</code>
برازش لگاریتمی	$y = m \ln x + b$ $y = m \log x + b$	<code>polyfit(log(x),y,1)</code> <code>polyfit(log10(x),y,1)</code>
برازش معکوس	$y = \frac{1}{mx+b}$	<code>polyfit(x,1./y,1)</code>

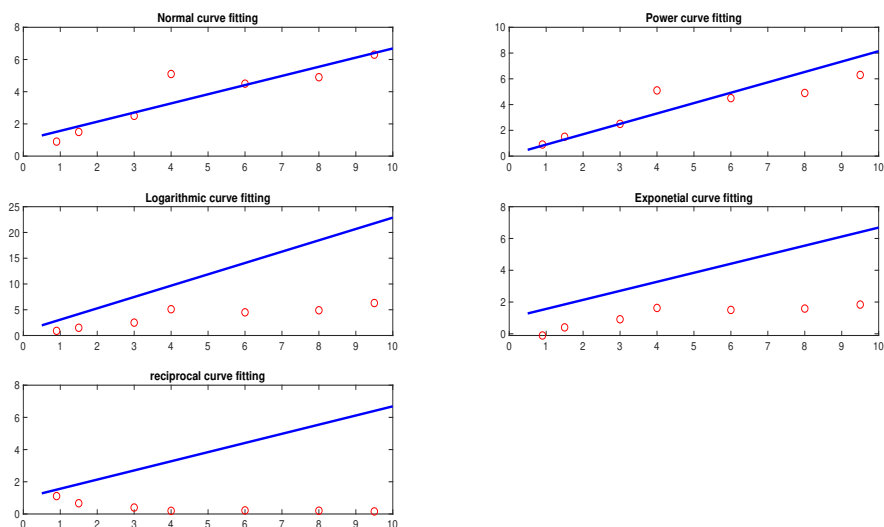
مثال ۱۰.۷. نقاط زیر را در نظر بگیرید،

$$(0.9, 0.9), (1.5, 1.5), (3, 2.5), (4, 5.1), (6, 4.5), (8, 4.9), (9.5, 6.3).$$

توابع غیر چندجمله‌ای را به‌شکلی که در جدول ۱۰.۷ معرفی شدند را بیابید که نقاط بالا را برازش کنند.

```
x = [0.9,1.5,3,4,6,8,9.5];
x1 = 0.5:0.1:10;
y = [0.9,1.5,2.5,5.1,4.5,4.9,6.3];
z = polyfit(log(x),log(y),1);
subplot(3,2,1);plot(x,y,'ro');
hold on;
w = polyval(z,x1);
plot(x1,w,'-b','LineWidth',2);
title('Power curve fitting');
```

خروجی زیر حاصل اجرای دستورات بالاست. توجه کنید که دستورات بالا بخشی از کد می‌باشد که با آن برازش توانی انجام می‌شود. برای سایر حالات، باید این دستورات برای تمام حالت‌های بیان شده در جدول ۱۰.۷ برای دستور `polyfit` تکرار شوند.



۳.۷ درونیابی

فرض کنید $n + 1$ نقطه^۱ داده شده باشند، (یا تابعی با ضابطه مشخص داده شده باشد).

$$(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n),$$

منظور از درونیابی نقاط مفروض، (یا درونیابی تابع مفروض) یافتن یک چندجمله‌ای $P_n(x)$ از درجه n است، به گونه‌ای که

$$P(x_i) = y_i, \quad i = 0, 1, \dots, n.$$

در آنالیز عددی انواع گوناگونی برای چندجمله‌ای‌های درونیاب وجود دارد که برخی از آنها عبارتند از: چندجمله‌ای‌های درونیاب لاگرانژ، نیوتون، هرمیت، اسپلاین و با استفاده از MATLAB برخی از این چندجمله‌ای‌های درونیاب را می‌توان به سادگی یافت. در این بخش به معرفی چگونگی درونیابی نقاط خواهیم پرداخت.

^۱ این نقاط در آنالیز عددی به نقاط گره‌ای موسوم هستند.

دستور `interp1`

شکل کلی دستور `interp1` که برای درونیابی مورد استفاده قرار می‌گیرد به صورت زیر است،

```
yi = interp1(x,y,xi,'method')
```

(توجه کنید که کاراکتر آخر دستور `interp1` عدد یک می‌باشد) که در آن

x برداری شامل مولفه‌های اول نقاط گره‌ای می‌باشد. مقادیر این بردار باید به ترتیب صعودی یا نزولی باشند.

y برداری شامل مولفه‌های دوم نقاط گره‌ای می‌باشد.

xi برداری شامل مولفه‌های اول نقاطی است که چندجمله‌ای درونیاب را تشکیل می‌دهند. تعداد عناصر این بردار می‌تواند از تعداد عناصر بردار **x** بیشتر باشد.

yi برداری شامل مولفه‌های دوم نقاطی است که چندجمله‌ای درونیاب را تشکیل می‌دهند.

method روشی است که چندجمله‌ای درونیاب مبتنی بر آن روش ساخته می‌شود. بجای `method` می‌توان موارد زیر را قرار داد:

nearest مقادیری را برگشت می‌دهد که به نقاط گره‌ای نزدیک‌ترین فاصله را داشته باشد.

linear از روش درونیابی اسپلاین خطی استفاده می‌کند.

spline از روش درونیابی اسپلاین مکعبی استفاده می‌کند.

pchip از روش درونیابی قطعه‌وار مکعبی هرمیت استفاده می‌کند.

علاوه بر این چهار مورد موارد دیگری نیز وجود دارند که عبارتند از: `previous`، `next`، `makima` و `v5cubic` که کاربرد کمتری دارند ولی در صورت نیاز می‌توانید با دستور `help interp1` به اطلاعاتی پیرامون آنها دسترسی پیدا کنید.

در ادامه به بیان چند مثال در زمینه درونیابی می‌پردازیم. پیشنهاد می‌شود تمامی مثال‌ها را در محیط MATLAB اجرا کنید.

مثال ۱۱.۷. نقاط گره‌ای زیر را به هر ۸ روش بیان شده درونیابی کنید و نمودار آنها را جداگانه رسم کنید.

$$(1, -1), (3, 3), (6, 4), (8, 2), (10, 8), (11, -3), (14, 5), (15, 6).$$

شکل اولیه دستورات لازم به صورت زیر می‌باشد و لی به شکل کوتاه‌تر و با استفاده از حلقه for نیز این مسأله را می‌توان حل کرد.

```
x = [1,3,6,8,10,11,14,15]; y = [-1,3,4,2,8,-3,5,6]
; xi = 0:0.01:16;
subplot(4,2,1); yi = interp1(x,y,xi,'linear');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('linear');
subplot(4,2,2); yi = interp1(x,y,xi,'nearest');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('nearest');
subplot(4,2,3); yi = interp1(x,y,xi,'spline');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('spline');
subplot(4,2,4); yi = interp1(x,y,xi,'pchip');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('pchip');
subplot(4,2,5); yi = interp1(x,y,xi,'next');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('next');
subplot(4,2,6); yi = interp1(x,y,xi,'previous');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('previous');
subplot(4,2,7); yi = interp1(x,y,xi,'v5cubic');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('v5cubic');
subplot(4,2,8); yi = interp1(x,y,xi,'makima');
plot(x,y,'ro',xi,yi,'-b','LineWidth',2);title('makima');
```

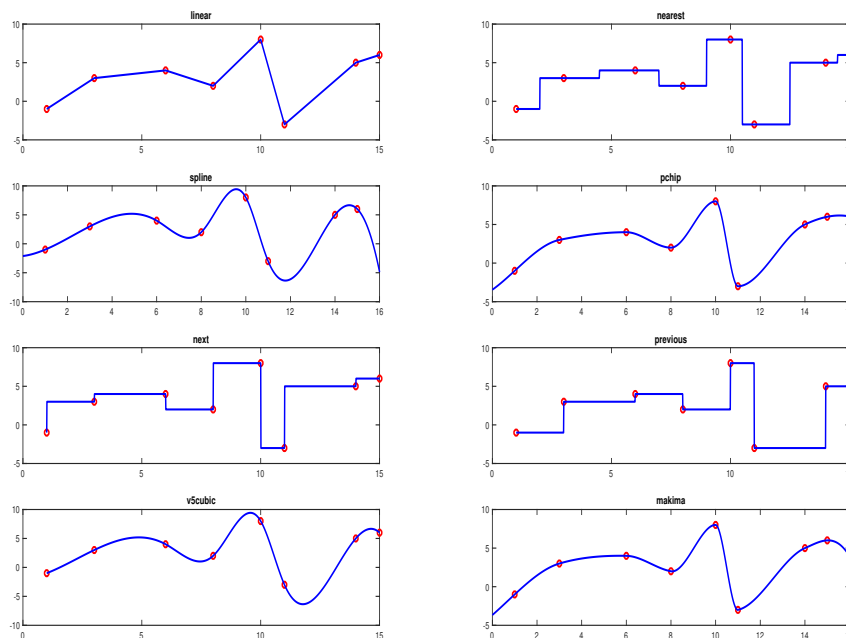
شکل ساده‌تر دستورات بالا و به صورت برنامه زیر به شکل زیر می‌باشد. پیشنهاد می‌شود این گونه مسایل را با استفاده از برنامه نویسی و به صورتی که در زیر آمده است حل کنید تا از بازنویسی چندین باره دستورات جلوگیری شود. در برنامه زیر به چگونگی معرفی method به صورت مجموعه‌ای از رشته‌ها و روش استفاده از آنها در دستور interp1 توجه کنید.

```

x = [1,3,6,8,10,11,14,15]; y = [-1,3,4,2,8,-3,5,6];
xi = 0:0.01:16;
method = {'linear','nearest','spline','pchip','nest','previous',...
'v5cubic','makima'};
for i=1:8
    subplot(4,2,i); yi = interp1(x,y,xi,method{i});
    plot(x,y,'ro',xi,yi,'-b','LineWidth',2);
    title(['interpolation by ',method{i}]);
end

```

کد بالا هر ۸ حالت درونیابی را تولید می‌کند.

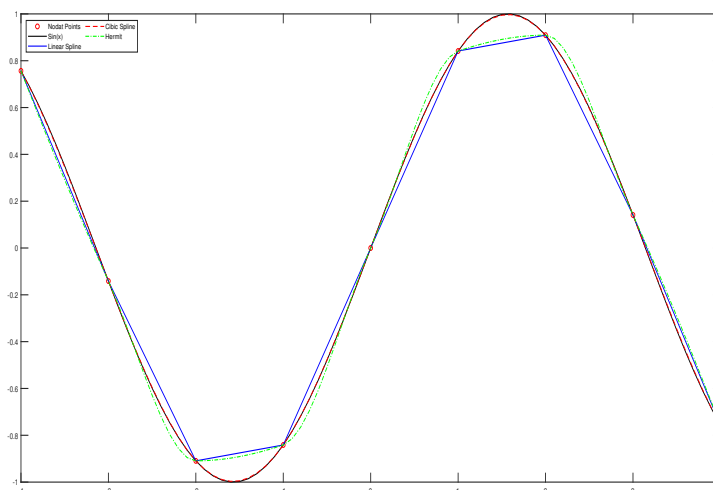


مثال ۱۲.۷. درونیاب تابع $y = \sin(x)$ را در بازه $[-4, 4]$ با استفاده از روش‌های اسپلاین خطی، هرمیت و اسپلاین مکعبی بدست آورید و همه را در یک نمودار رسم کنید. همچنین راهنمای مناسب برای نمودارها تهیه کنید. تعداد نقاط گره‌ای مورد استفاده را ۸ نقطه در نظر بگیرید.

برنامه زیر نمودار مورد نظر را تولید می‌کند. چون در این نمودار از رنگ‌های مختلف استفاده شده است، حتما برنامه در یک m-فایل بنویسید و اجرا کنید.

```
x = -4:1:4; y = sin(x);
xi = -4:0.1:4; yi = sin(xi);
plot(x,y,'ro',xi,yi,'-k','LineWidth',1);
hold on;
yi = interp1(x,y,xi,'linear');
plot(xi,yi,'-b','LineWidth',1);
yi = interp1(x,y,xi,'spline');
plot(xi,yi,'--r','LineWidth',1);
yi = interp1(x,y,xi,'pchip');
plot(xi,yi,'-.g','LineWidth',1);
legend('Nodat Points','Sin(x)','Linear Spline','Cibic Spline',...
'Hermit','Location','NorthWest','NumColumns',2);
```

خروجی برنامه بالا به صورت زیر می‌باشد.



۴.۷ تمرین

تمرین ۱.۷. نمودار چندجمله‌ای‌های زیر را در باز خواسته شده رسم کنید.

$$\bullet \quad y = -\frac{1}{4}x^4 + 7x^2 - \frac{20}{5}x - 28 \quad \text{در بازه } [-5, 4]$$

$$\bullet \quad y = -\frac{1}{10}x^4 + \frac{5}{10}x^3 - \frac{7}{10}x^2 + \frac{3}{8}x - \frac{1}{4} \quad \text{در بازه } [1, 14]$$

برای رسم هر دو نمودار از دستورات مرتبط با تعریف و محاسبه چندجمله‌ای‌ها استفاده کنید.

تمرین ۲.۷. حاصل ضرب عبارات زیر را با استفاده از دستور ضرب چندجمله‌ای‌ها بدست آورید.

$$\bullet \quad (2x^2 + 3)(x^3 + \frac{3}{5}x^2 + 5x - 16)$$

$$\bullet \quad (x + \frac{1}{4})(x - \frac{1}{4})(x + \frac{1}{6})(x - \frac{1}{4})$$

سپس نمودار عبارت دوم را در بازه $[-1/5, 1/5]$ رسم کنید.

تمرین ۳.۷. حاصل تقسیم $\frac{p(x)}{q(x)}$ را در دو حالت زیر محاسبه کنید.

$$\bullet \quad p(x) = -\frac{1}{6}x^5 + \frac{7}{7}x^3 - 8x^2 - \frac{24}{6}x + 48, \quad q(x) = -\frac{1}{6}x^3 + \frac{4}{1}x - 8$$

$$\bullet \quad p(x) = x^4 - 6x^3 + 13x^2 - 12x + 4, \quad q(x) = x^3 - 3x^2 + 2$$

تمرین ۴.۷. زیربرنامه‌ای به نام polyadd بنویسید که دو چندجمله‌ای و یک عملگر که می‌تواند عملگر جمع یا تفریق باشد را دریافت کند و عمل را بر روی چندجمله‌ها انجام داده و حاصل را در خروجی به شکل یک چندجمله‌ای بنویسد. برای مثال خروجی دستورات

```
p1 = [1 2 -1];
p2 = [0 1 2];
op = '+';
p = polyadd(p1,p2,op)
```

باید به صورت x^2+3x+1 باشد. با تغییرات کوچکی در زیربرنامه نوشته شده، زیربرنامه‌ای به نام polymult بنویسید که عمل ضرب دو چندجمله‌ای را انجام دهد. زیربرنامه‌هایی که نوشته‌اید را برای دو چندجمله‌ای زیر امتحان کنید.

$$p_1(x) = x^4 - 3x^3 + 2x - 1, \quad p_2(x) = x^2 - x + 3.$$

تمرین ۵.۷. جمعیت در جهان از سال ۱۷۵۰ تا سال ۲۰۱۹ در جدول زیر آورده شده است، جمعیت برحسب میلیون نفر می باشد.

سال	۱۷۵۰	۱۸۰۰	۱۸۵۰	۱۹۰۰	۱۹۵۰	۱۹۹۰	۲۰۰۰	۲۰۰۹	۲۰۱۹
جمعیت	۷۹۱	۹۸۰	۱۲۶۰	۱۶۵۰	۲۵۲۰	۵۲۷۰	۶۰۶۰	۶۸۰۰	۷۳۰۰

۱. یک تابع نمایی بدست آورید که بهترین تقریب برای داده های جدول باشد. با استفاده از تابع تعیین شده، جمعیت جهان را در سال ۱۹۸۰ برآورد کنید.

۲. داده های جدول را با یک چندجمله ای درجه سوم برازش کنید. با استفاده چندجمله ای یافته شده جمعیت جهان را در سال ۱۹۸۰ تخمین بزنید.

۳. درونیاب های خطی و اسپلاین را برای داده های جدولی بدست آورید و هر دو را در یک نمودار رسم کنید.

تمرین ۶.۷. داده های جدول زیر را در نظر بگیرید.

x	-۵	-۳/۴	۲	-۰/۸	۰	۱/۲	۲/۵	۴	۵/۰	۷	۸/۵
y	۴/۴	۴/۵	۴	۳/۶	۳/۹	۳/۸	۳/۵	۲/۵	۱/۲	۰/۵	-۰/۲

۱. داده های جدول را با چندجمله ای های درجه اول تا درجه یازدهم برازش کنید و نمودارهای حاصل را رسم کنید.

۲. توابع درونیاب داده های جدول را برای هر ۸ حالتی که بیان کردیم بدست آورید و نمودار هر یک را جداگانه رسم کنید.

تمرین ۷.۷. درونیاب های اسپلاین خطی، اسپلاین مکعبی و هرمیت را برای توابع زیر بدست آورید و هر سه را به همراه نمودار خود تابع در یک نمودار با رنگ های مختلف رسم کنید. راهنمای نمودار مناسب برای هر نمودار تهیه کنید. تعداد نقاط گره ای را برای هر تابع ۱۰ نقطه در نظر بگیرید.

$$f(x) = \frac{\sin x}{x^2 + 1}, \quad g(x) = (x^2 + 1)^{\sin x}, \quad h(x) = \left(x + \frac{1}{x}\right)^x$$

نسخه
دیپگان

۸ کاربرد MATLAB در ریاضیات محاسباتی

انجام محاسبات ریاضی همواره در شاخه‌های مختلف علوم و مهندسی مورد نیاز بوده است. در این فصل به جنبه‌های مختلف کاربرد MATLAB در ریاضیات محاسباتی می‌پردازیم.

۱.۸ کاربرد در آنالیز عددی

آنالیز عددی از موضوعاتی است که تقریباً در همه شاخه‌های مهندسی کاربرد دارد. این شاخه از ریاضیات کاربردی از دو منظر مورد توجه ریاضیدانان می‌باشد: تحلیلی و عددی. در این کتاب ما هیچ کاری به بخش تحلیلی آنالیز عددی نداریم و تمام توجه ما به بخش عددی آن معطوف می‌باشد. تقریباً تمام مواردی که در آنالیز عددی مطرح می‌شوند، نیاز به پیاده‌سازی دارند. تا پیش از تولید نرم‌افزارهای محاسباتی، معمولاً مهندسی‌ن و ریاضیدان‌ها برای پیاده‌سازی الگوریتم‌های عددی با استفاده از زبان‌های مختلف برنامه‌نویسی مانند فرترن، C و زبان‌های دیگر، برنامه‌های مورد نیاز خود را می‌نوشتند، ولی با ظهور نرم‌افزارهایی مانند MATLAB کار ساده‌تر شد. با استفاده از MATLAB تقریباً تمام کارهای مورد نیاز در آنالیز عددی قابل انجام است، کارهایی مانند حل معادلات غیرخطی، تعیین نقاط اکسترمم، انتگرال‌گیری عددی، حل عددی معادلات دیفرانسیل، حل دستگاه معادلات خطی و غیرخطی و موارد دیگری که ممکن است به آنها نیاز باشد. هدف ما در این بخش معرفی امکانات MATLAB در حل مسایل آنالیز عددی می‌باشد.

۱.۱.۸ یافتن ریشه‌های معادلات یک متغیره

یکی از پر کاربردترین موضوعات در آنالیز عددی یافتن ریشه‌های یک معادله یک متغیره می‌باشد. همان‌طور که در آنالیز عددی و محاسبات عددی دیده‌اید، در حالت کلی برای یافتن ریشه‌های

معادله $f(x) = 0$ راه حل کلی وجود ندارد و برای حل این معادله باید از روش‌های عددی مانند روش نقطه ثابت، روش نیوتون، روش نصف کردن فاصله و روش‌های دیگری که به این منظور طراحی شده‌اند کمک گرفت. با استفاده از هر زبان برنامه‌نویسی این امکان وجود دارد تا روش‌هایی که بیان کردیم را تبدیل به یک برنامه کنیم و با اجرای آنها یک ریشه معادله را بدست آوریم. ولی با استفاده از MATLAB، بدون نیاز به برنامه‌نویسی و تنها با یک دستور می‌توان به یک ریشه معادله $f(x) = 0$ دسترسی پیدا کرد.

دستور fzero

با استفاده از دستور fzero می‌توان یک ریشه معادله $f(x) = 0$ را که به x_0 نزدیک‌تر است را بدست آورد. شکل کلی این دستور به صورت `fzero(function, x0)` است که در آن **x0** مقداری است که ریشه به آن نزدیک است و باید توسط کاربر مشخص شود. این مقدار را می‌توان با رسم نمودار و به صورت حدودی از روی شکل تعیین کرد.

function تابعی یک متغیره است که می‌خواهیم ریشه آن را بدست آوریم. تابع را به سه شکل می‌توان تعریف کرد:

۱. به صورت یک رشته و در میان یک جفت کوتیشن،

۲. به صورت یک تابع در MATLAB،

۳. با استفاده از توابع بدون نام.

در هر سه حالت نتیجه یکسان خواهد بود و ریشه‌ای از معادله که به مقدار x_0 نزدیک‌تر می‌باشد محاسبه می‌شود.

📌 دستور fzero را می‌توان به یک کمیت اسکالری نسبت داد تا مقدار ریشه در یک متغیر ذخیره شود.

📌 ریشه‌ای که محاسبه می‌شود به x_0 نزدیک می‌باشد، پس اگر معادله‌ای بیش از یک ریشه داشته باشد، ممکن است با تغییر مقدار x_0 مقدار محاسبه شده برای ریشه نیز تغییر کند.

📌 بجای x_0 می‌توان از بازه‌ای که جواب در آن بازه قرار دارد نیز استفاده کرد، یعنی می‌توان $[a, b]$ را بجای x_0 قرار داد که در آن ریشه معادله در بازه $[a, b]$ قرار دارد.

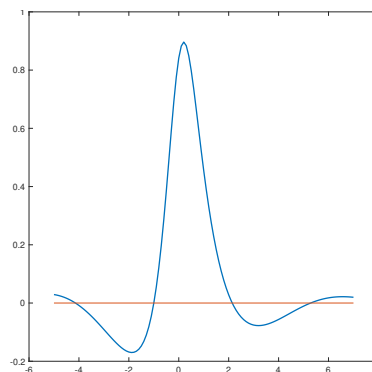
مثال ۱.۸. تمام ریشه‌های معادله $\frac{\sin(x+1)}{x^2+1}$ را بدست آورید.

برای تعیین ریشه‌های معادله ابتدا باید حدود ریشه‌ها را بدانیم، به این منظور ابتدا نموداری از تابع رسم می‌کنیم تا حدود ریشه‌ها را پیدا کنیم،

ورودی

```
x = -5:0.1:7;
y = sin(x + 1)./(x.^2 + 1);
plot(x,y,'LineWidth',1.5);
hold on;
plot(x,zeros(1,length(x)));
axis square;
```

خروجی



با توجه به نمودار، معادله ریشه‌هایی در بازه‌های $[2, 3]$, $[-2, 0]$, $[-5, -4]$ و $[4, 6]$ دارد. لذا می‌توان دستور `fzero` را به‌صورت زیر بکار ببریم.

ورودی

```
f = @(x) sin(x+1)./(x.^2+1);
x1 = fzero(f,-5)
x2 = fzero(f,[-2,0])
x3 = fzero(f,2)
x4 = fzero(f,[4,6])
```

خروجی

```
x1 = -4.1416
x2 = -1
x3 = 2.1416
x4 = 5.2832
```

👉 در دستورات بالا هم از مقدار اولیه نزدیک به ریشه و هم از بازه شامل ریشه استفاده شده است.

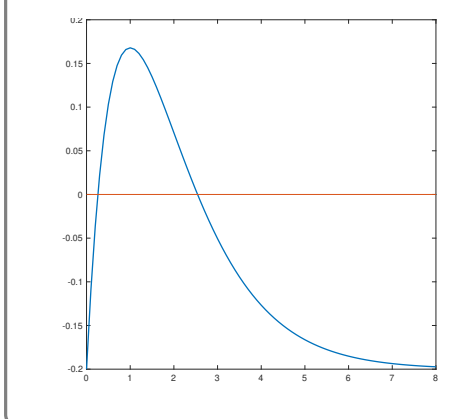
مثال ۲.۸. ریشه‌های معادله $\frac{1}{2} = xe^{-x}$ را بدست آورید. مشابه مثال پیش، ابتدا با رسم نمودار حدود ریشه‌ها را تعیین می‌کنیم. با توجه نمودار معادله دارای دو ریشه در بازه‌های $[0, 1]$ و $[2, 3]$ می‌باشد. توجه کنید برای تعیین محل ریشه می‌توانید از روش‌های دیگری که در کتاب‌های حساب دیفرانسیل و انتگرال آورده شده است نیز استفاده کنید. برای مثال می‌توانید مقدار تابع $f(x) = xe^{-x} - \frac{1}{2}$ را برای دو مقدار x حساب کنید، اگر حاصل ضرب دو مقدار منفی باشد،

یعنی یک ریشه در بازه وجود دارد.

ورودی

```
x = 0:0.1:8;
y = x.*exp(-x) - 0.2;
plot(x,y,'LineWidth',1.5);
hold on;
plot(x,zeros(1,length(x)));
axis square;
```

خروجی



حال با دستورات زیر می‌توانید ریشه‌ها را محاسبه کنید. در دستورات زیر از تابع بدون نام استفاده نشده است و معادله به صورت مستقیم و به شکل رشته به دستور `fzero` ارسال شده است.

ورودی

```
x1 = fzero('x.*exp(-x) - 0.2', 0)
x2 = fzero('x.*exp(-x) - 0.2', [2, 3])
```

خروجی

```
x1 = 0.2592
x2 = 2.5426
```

📌 توجه کنید اگر در دستور `fzero` از بازه استفاده می‌کنید حتما باید مقدار تابع در ابتدای بازه و مقدار تابع در انتهای بازه علامت‌های یکسانی نداشته باشند، در غیر این صورت در هنگام اجرا با خطا مواجه خواهید شد.

دستور `fzero` گزینه‌های دیگری نیز دارد که زیاد مورد استفاده قرار نمی‌گیرند، ولی دو شکل استفاده از این دستور شاید کاربردهایی داشته باشند.

می‌دانیم ریشه یک معادله که با دستور `fzero` حاصل می‌شود یک جواب تقریبی است، پس شاید مقدار تابع در این نقطه دقیقا صفر نشود. با تغییر اندکی در روش استفاده از دستور `fzero` می‌توان مقدار تابع در جواب محاسبه شده را نیز در اختیار داشت.

نکته عملی

اگر بخواهیم پس از یافتن جواب، مقدار تابع در جواب نیز محاسبه شود می‌توان دستور `fzero` را به شکل زیر بکار برد.

```
[x,fx] = fzero(function,x0)
```

در این صورت پس از محاسبه ریشه مقدار آن در `x` و مقدار تابع در جواب در `fx` ذخیره می‌شود.

مثال ۳.۸. یک ریشه معادله $\frac{\sin(x+1)}{x^2+1}$ را به همراه مقدار تابع در جواب یافته شده محاسبه کنید. در مثال ۱.۸ دیدیم که یک جواب در نزدیک ۵- رخ خواهد داد، پس دستور `fzero` را به شکل زیر فراخوانی می‌کنیم تا جواب به همراه مقدار تابع در جواب یافته شده محاسبه شود.

ورودی

```
f = @(x) sin(x+1)./(x.^2+1);  
[x,fx] = fzero(f,-5)
```

خروجی

```
x = -4.1416  
fx = -6.7463e-18
```

نکته عملی

دستور `fzero` ریشه یک معادله را با استفاده از روش‌های تکراری پیدا می‌کند. اگر بخواهیم بدانیم که ریشه با چند تکرار محاسبه شده است، می‌توان این دستور را به شکل زیر بکار برد.

```
x = fzero(function,x0,optimset('display','iter'))
```

در این صورت تمام محاسباتی که برای یافتن ریشه انجام می‌شود نیز نمایش داده خواهد شد.

مثال ۴.۸. نتیجه حاصل از اجرای دستورات

```
f = @(x) sin(x+1)./(x.^2+1);  
x = fzero(f,5,optimset('display','iter'))
```

که برای یافتن ریشه معادله $\frac{\sin(x+1)}{x^2+1}$ که به عدد ۵ نزدیکتر می‌باشد به شکل زیر می‌باشد که در آن تمامی محاسبات، بازه جواب، جواب و تعداد تکرارهای انجام شده برای یافتن جواب آورده شده است. در حل مسایل با MATLAB معمولاً کمتر از این گزینه استفاده می‌شود.

Search for an interval around 5 containing a sign change:

Func-count	a	f(a)	b	f(b)	Procedure
1	5	-0.0107467	5	-0.0107467	initial interval
3	4.85858	-0.0167425	5.14142	-0.00515012	search
5	4.8	-0.0193262	5.2	-0.00296325	search
7	4.71716	-0.0230644	5.28284	-1.1851e-05	search
9	4.6	-0.0284868	5.4	0.00386436	search

Search for a zero in the interval [4.6, 5.4]:

Func-count	x	f(x)	Procedure
9	5.4	0.00386436	initial
10	5.30444	0.000729399	interpolation
11	5.28276	-1.46825e-05	interpolation
12	5.28319	1.15347e-07	interpolation
13	5.28319	1.78881e-11	interpolation
14	5.28319	-8.47153e-18	interpolation
15	5.28319	-8.47153e-18	interpolation

Zero found in the interval [4.6, 5.4]

x = 5.2832

👉 توجه کنید که اگر معادله مورد نظر به صورت یک چندجمله‌ای باشد برای یافتن ریشه‌های آن بهتر است از دستور roots استفاده کرد، زیرا با استفاده از این دستور می‌توان تمام ریشه‌های چندجمله‌ای را یافت، ولی با استفاده از دستور fzero تنها یک ریشه محاسبه می‌شود.

👉 به عنوان آخرین نکته درباره دستور fzero خوب است بدانید که این دستور تنها نقاطی را پیدا می‌کند که از برخورد محور x ها با نمودار تابع حاصل می‌شوند. همچنین اگر معادله هیچ جواب حقیقی در بازه مشخص شده نداشته باشد مقدار NaN برگشت داده می‌شود. برای مثال دستور fzero را برای معادله $x^2 + 1 = 0$ بکار ببرید.

۲.۱.۸ تعیین نقاط ماکزیمم و مینیمم توابع

یافتن نقاط ماکزیمم و مینیمم در بسیاری از شاخه‌های علوم و مهندسی مورد نیاز می‌باشد. در MATLAB به سادگی می‌توان نقطه‌ای از یک بازه مشخص را یافت که تابع در آن نقطه مقدار مینیمم یا ماکزیمم داشته باشد. این نقطه می‌تواند یک اکسترمم نسبی یا اکسترمم مطلق باشد.

دستور fminbnd

با دستور `fminbnd` می‌توان نقطه مینیمم یک تابع را یافت، به این منظور کافیسیت دستور را به شکل زیر بکار برد،

`[x,fx] = fminbnd(function,interval)`

x,fx نقطه مینیمم و مقدار تابع در آن می‌باشد.

text تابع مورد نظر است که مشابه دستور `fzero` باید تعریف شود و مورد استفاده قرار بگیرد.

interval بازه‌ای که نقطه می‌نیمم مورد نظر باید در آن جستجو شود.

اگر هدف یافتن مقدار ماکزیمم در بازه تعیین شده باشد، باید در دستور `fminbnd` کل تابع را در یک منفی ضرب کرد و در دستور قرار داد. در این صورت مقداری که برای f در نقطه x حاصل می‌شود را باید در منفی ضرب کرد تا مقدار ماکزیمم حاصل شود.

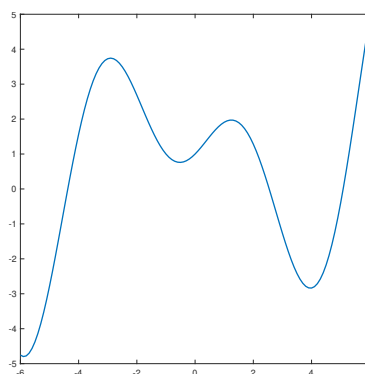
مثال ۵.۸. اکسترمم‌های نسبی تابع $f(x) = x \sin(x+1) + 1$ را در بازه $[-6, 6]$ بدست آورید.

ابتدا نمودار تابع را رسم می‌کنیم تا حدود نقاط اکسترمم مشخص شوند. با توجه به نمودار، تابع در بازه‌های $[-4, -2]$ و $[1, 2]$ دارای دو ماکزیمم نسبی و در بازه‌های $[-2, 0]$ و $[3, 5]$ دارای مینیمم نسبی می‌باشد. هدف تعیین این نقاط است.

ورودی

```
x = -6:0.1:6;
y = x.*sin(x+1) + 1;
plot(x,y,'LineWidth',1.5);
axis square;
```

خروجی



در دستورات زیر، با چهار دستور اول نقاط مینیمم و مقدار تابع در این نقاط و با چهار دستور آخر نقاط ماکزیمم و مقدار تابع در نقاط ماکزیمم محاسبه شده‌اند.

```
[min1,fmin1] = fminbnd('x.*sin(x+1) + 1',-2,0);
[min2,fmin2] = fminbnd('x.*sin(x+1) + 1',3,5);
fprintf('\nFirst Minimum is:%f and f(%f) = %f\n',min1,min1,fmin1);
fprintf('Second Minimum is:%f and f(%f) = %f\n',min2,min2,fmin2);

[max1,fmax1] = fminbnd('-x.*sin(x+1) - 1',-4,-2);
[max2,fmax2] = fminbnd('-x.*sin(x+1) - 1',1,2);
fprintf('\nFirst Maximum is:%f and f(%f) = %f\n',max1,max1,-fmax1);
fprintf('Second Maximum is:%f and f(%f) = %f\n',max2,max2,-fmax2);
```

خروجی دستورات بالا به شکل زیر می‌باشد.

```
First Minimum is:-0.520274 and f(-0.520274) = 0.759875
Second Minimum is:3.959760 and f(3.959760) = -2.839223
```

```
First Maximum is:-2.902586 and f(-2.902586) = 3.744282
Second Maximum is:1.246787 and f(1.246787) = 1.972603
```

در دستورات مربوط به محاسبه نقاط ماکزیمم به ضرب ضابطه تابع در ۱- و منفی کردن مقادیر تابع در نقاط یافته شده در دستور fprintf دقت کنید.

۳.۱.۸ انتگرال گیری عددی

تقریباً در تمامی شاخه‌های علوم و مهندسی، به موارد بر می‌خوریم که نیاز به محاسبه انتگرال‌های معین می‌باشد. برای محاسبه انتگرال همواره امکان استفاده از روش مستقیم نیست، زیرا شاید تابع اولیه قابل محاسبه نباشد، یا محاسبه تابع اولیه به سختی امکان‌پذیر باشد، لذا روش‌های گوناگونی برای محاسبه انتگرال‌های عددی توسط ریاضیدانان به‌وجود آمده است که امکان محاسبه انتگرال معین را بدون نیاز به یافتن تابع اولیه فراهم می‌کند. برای استفاده از روش‌های عددی محاسبه انتگرال‌های معین نیاز به کامپیوتر و برنامه‌نویسی است تا مثلاً روش انتگرال گیری گاوسی ۳ نقطه‌ای را پیاده‌سازی کرد. تمامی روش‌های عددی انتگرال گیری را می‌توان با برنامه‌نویسی به هر زبانی مورد استفاده قرار داد، ولی با استفاده از MATLAB با چند دستور ساده می‌توان مقادیر عددی انتگرال‌های معین را بدون هیچ برنامه‌نویسی ویژه‌ای بدست آورد.

دستور quad

از دستور quad که به شکل کلی زیر مورد استفاده قرار می‌گیرد، می‌توان برای محاسبه انتگرال‌های معین استفاده کرد.

`quad(function,a,b)`

که در آن function ضابطه تابع مورد نظر و a و b کران‌های پایین و بالای انتگرال می‌باشند. توجه کنید که ضابطه تابع را باید به شکل توابع بدون نام یا با استفاده از m-فایل‌ها به صورت یک تابع تعریف کرد. شایان ذکر است که این روش مبتنی بر روش سیمپسون می‌باشد.

مثال ۶.۸. انتگرال‌های معین زیر را محاسبه کنید.

$$I_1 = \int_0^{\pi} \frac{\sin x}{x+1} dx, \quad I_2 = \int_0^5 \frac{\sin x}{x^2 + 2x + 1} dx, \quad I_3 = \int_0^{\infty} \frac{e^{-x^2}}{x+1} dx.$$

انتگرال‌های بالا با دستورات زیر محاسبه می‌شوند.

ورودی

```
f = @(x) sin(x)./(x+1);
I1 = quad(f,0,pi)
f = @(x) sin(x)./(x.^2 + 2*x + 1);
I2 = quad(f,0,5)
format long
f = @(x) exp(-x.^2)./(x+1);
I3 = quad(f,0,100000)
```

خروجی

```
I1 =
0.843810763128514
I2 =
0.343480805529832
I3 =
0.605134334344201
```

📌 در محاسبه انتگرال آخر بجای بی‌نهایت یک عدد بزرگ قرار داده‌ایم.
 📌 در هنگام استفاده از دستور quad تابع نباید در بازه انتگرال‌گیری مجانب قائم داشته باشد.

نکته عملی

دستور دیگری برای محاسبه انتگرال‌های عددی وجود دارد که به شکل quadl مورد استفاده قرار می‌گیرد. این دستور کاملاً مشابه با دستور quad می‌باشد و شکل استفاده و آرگومان‌های ورودی این دو دستور هیچ تفاوتی با هم ندارند. تنها تفاوت میان این دو دستور در روشی است که دستور بر اساس آن ساخته شده است. دستور quadl از روش لوباتو برای محاسبه انتگرال استفاده می‌کند. کاراکتر l در انتهای دستور به همین دلیل قرار داده شده است.

📌 به عنوان تمرین دستورات مثال پیش را با دستور quadl اجرا کنید و خروجی را با دستور quad مقایسه کنید.

مثال ۷.۸. سطح محصور بین نمودار توابع زیر را بیابید،

$$f(x) = x^3 - 3x^2 - 2x - 5, \quad g(x) = 10x^2 - 10x - 5.$$

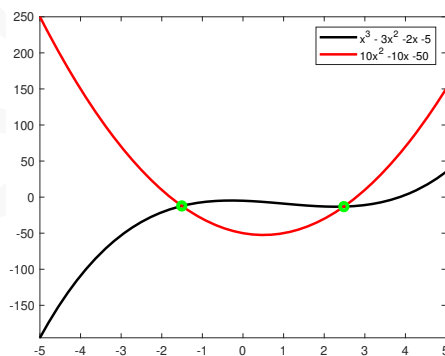
ابتدا باید نقاط برخورد دو نمودار را تعیین کنیم. به این منظور باید معادله $f(x) - g(x) = 0$ را حل کنید. برنامه زیر، ابتدا نقاط برخورد را یافته و نمودار را به همراه نقاط برخورد در یک شکل رسم می‌کند. با دو دستور آخر سطح محصور مورد نظر با استفاده از $\int_a^b (f(x) - g(x)) dx$ محاسبه و چاپ می‌شود.


```

f = @(x) x.^3 - 3*x.^2 - 2*x - 5;
fplot(f, '-k', 'LineWidth', 2);
hold on;
g = @(x) 10*x.^2 - 10*x - 50;
fplot(g, '-r', 'LineWidth', 2);
h = @(x) (x.^3 - 3*x.^2 - 2*x - 5) - (10*x.^2 - 10*x - 50);
a = fzero(h, -2);
plot(a, f(a), 'go', 'LineWidth', 3);
b = fzero(h, 2);
plot(b, f(b), 'go', 'LineWidth', 3);
legend('x^3 - 3x^2 - 2x - 5', '10x^2 - 10x - 50');
A = abs(quad(h, a, b));
fprintf('Area from %f to %f is %f\n', a, b, A);

```

خروجی برنامه بالا به شکل نمودار زیر می باشد.



به علاوه مقدار سطح محصور نیز به شکل زیر نمایش داده می شود.

Area from -1.506961 to 2.483621 is 122.173087

📌 پیشنهاد می شود برنامه بالا را در یک m-فایل بنویسید و خروجی حاصل را مشاهده کنید. همچنین می توانید مسایل مختلفی که در کتاب های حساب دیفرانسیل و انتگرال وجود دارد را به همین شیوه به عنوان تمرین حل کنید.

دستور trapz

اگر بجای ضابطه تابع، تعدادی نقطه در اختیار داشته باشیم، با دستور `trapz(x,y)` می‌توان مقدار انتگرال معین را یافت.

مثال ۸.۸. حاصل انتگرال $\int_0^1 \frac{\sin \sqrt{x+1} + 1}{x^2 + x + e^x} dx$ را با استفاده از هر سه دستوری که برای محاسبه انتگرال بیان کردیم بدست آورید.

با استفاده از دستورات زیر می‌توان مقدار مورد نظر را یافت. این دستورات را در یک فایل `m`- بنویسید و با ایجاد تغییرات در طول گام استفاده شده برای بردار `x` نتایج مختلف را با هم مقایسه کنید. بدیهی است که هر چقدر طول گام کوچک‌تر انتخاب شود، جواب به مقدار دقیق نزدیک‌تر خواهد بود.

```
format long;
x = 0:0.31:3;
y = (sin(sqrt(x+1))+1)./(x.^2 + x + exp(x));
I_trapz = trapz(x,y)
f = @(x) (sin(sqrt(x+1))+1)./(x.^2 + x + exp(x));
I_quad = quad(f,0,3)
I_quadl = quadl(f,0,3)
```

خروجی دستورات بالا به شکل زیر می‌باشد.

```
I_trapz = 1.277246949909653
I_quad = 1.264417962630626
I_quadl = 1.264417913548457
```

۲.۸ کاربرد در معادلات دیفرانسیل با مقدار اولیه

معادلات دیفرانسیل در ریاضیات از دو جنبه مورد توجه می‌باشد: حل معادله دیفرانسیل و یافتن جواب به صورت تحلیلی و حل عددی معادلات دیفرانسیل و یافتن جواب به شکل تقریبی. در این بخش به حل عددی معادلات دیفرانسیل با مقدار اولیه خواهیم پرداخت و یافتن جواب تحلیلی را به فصل‌های بعدی موکول می‌کنیم.

برای حل عددی معادلات دیفرانسیل مرتبه اول با مقدار اولیه، برخلاف استفاده از دستورات دیگر MATLAB نیاز انجام برخی کارها می‌باشد و نمی‌توان از دستورات حل عددی معادلات دیفرانسیل مسقیم استفاده کرد. به منظور استفاده از دستورات حل عددی معادلات دیفرانسیل مرتبه اول باید ابتدا مراحل زیر را به ترتیب انجام داد.

۱. معادله دیفرانسیل را به شکل استاندارد زیر بنویسید،

$$\begin{cases} \frac{dy}{dt} = f(t, y), & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

توجه کنید که هدف ما حل یک مسأله مقدار اولیه می‌باشد. برای مثال معادله دیفرانسیل $t dy - (t^3 - 2y) dt = 0$ ، برای $t \in [1, 3]$ و $y(1) = 4$ را باید به شکل زیر نوشت،

$$\begin{cases} \frac{dy}{dt} = \frac{t^3 - 2y}{t}, & 1 \leq t \leq 3, \\ y(1) = 4. \end{cases}$$

۲. با استفاده از `m`-فایل یک تابع بنویسید که $f(t, y)$ در آن تعریف شده باشد. برای مثال یک `m`-فایل با نام `f.m` و با محتوای به شکل زیر تولید کنید.

```
function dydt = f(t,y)
    dydt = (t.^3 - 2*y)./t;
end
```

همچنین می‌توان با استفاده از توابع بدون نام نیز تابع $f(t, y)$ را تعریف کرد. برای مثال می‌توان این کار را با دستور زیر انجام داد،

```
ode = @(t,y) (t.^3 - 2*y)./t;
```

۳. روش مورد نیاز خود را برای حل معادله دیفرانسیل از جدول ۱.۸ انتخاب کنید.

جدول ۱.۸: روش‌های حل معادلات دیفرانسیل در MATLAB

نام روش	شرح روش
ode45	بهترین گزینه برای شروع محاسبات است که مبتنی بر روش رانگ-کوتا بوده برای حل معادلات غیرسخت بکار می‌رود.
ode23	مبتنی بر روش رانگ-کوتا است که دقت کمتر نسبت به روش ode45 دارد و برای حل معادلات غیرسخت بکار می‌رود.
ode113	مبتنی بر روش چند گامی است که برای حل معادلات غیرسخت بکار می‌رود.
ode15s	مبتنی بر روش چند گامی است و برای حل معادلات سخت بکار می‌رود. اگر روش ode45 با خطا مواجه شد از این روش استفاده کنید.
ode23s	مبتنی بر روش تک گامی است و برای حل معادلات سخت بکار می‌رود. اگر روش ode15s قادر به حل مسأله نبود از این روش استفاده کنید.
ode23t	برای حل معادلات تقریباً سخت مورد استفاده قرار می‌گیرد.
ode23tb	برای حل معادلات سخت بکار می‌رود که از روش ode15s کاراتر است.

👉 پیشنهاد می‌شود نخست روش ode45 را آزمایش کنید، اگر با خطا مواجه شدید روش ode15s مورد استفاده قرار دهید.

روش حل معادلات دیفرانسیل

پس از انجام مراحل بالا با دستور زیر معادله دیفرانسیل را حل کنید،

```
[t,y] = Solver_Name(@func_name,tspan,y0)
[t,y] = Solver_Name(func_name,tspan,y0)
```

توجه کنید اگر از روش اول گام ۲ برای تعریف $f(t,y)$ استفاده کرده‌اید باید دستور اول و اگر از روش دوم استفاده کرده‌اید باید دستور دوم را بکار ببرید. در دستورات بالا

Solver_Name نام روش مورد استفاده است که باید از جدول ۱.۸ انتخاب شود.

func_name نام تابعی است که در گام ۲ تعریف کرده‌ایم.

tspan برداری است که تعیین کننده بازه‌ای است که جواب بر آن تعریف شده است. این بردار باید حداقل شامل دو عنصر باشد، ولی بهتر است بازه جواب را به تعداد بیشتری بخش تقسیم کرد تا دقت جواب حاصل بیشتر شود.

y0 مقدار اولیه‌ای است که در مسأله مشخص شده است.

[t,y] شامل دو بردار ستونی است که در t همان مقادیر **tspan** قرار می‌گیرد و در y مقادیر جواب در بردار t ذخیره می‌شود. پس از تعیین این مقادیر می‌توان با دستور `plot(t,y)` نمودار جواب تقریب را رسم کرد.

مثال ۹.۸. معادله دیفرانسیل $t^3 - 2y) dt = 0$ را برای $t \in [1,3]$ و $y(1) = 4$ حل کنید و نمودار آن را رسم کنید.
ابتدا معادله را به شکل

$$\begin{cases} \frac{dy}{dt} = \frac{t^3 - 2y}{t}, & 1 \leq t \leq 3, \\ y(1) = 4 \end{cases}$$

می‌نویسیم. با استفاده از توابع بدون نام تابع $\frac{t^3 - 2y}{t}$ را تعریف می‌کنیم، سپس بردارهای t و y محاسبه کرده و نمودار را رسم می‌کنیم. کد زیر را به همراه خروجی مشاهده کنید.

ورودی

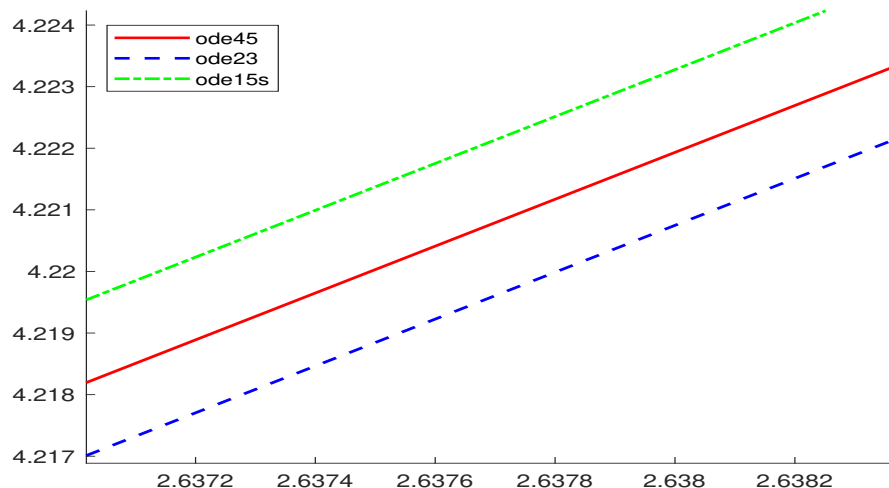
```
ode = @(t,y) (t.^3 - 2*y)./t;
[t,y] = ode45(ode,[1:0.5:3],4)
```

خروجی

```
t =
1.0000
1.5000
2.0000
2.5000
3.0000
y =
4.0000
2.3639
2.5500
3.7330
5.8222
```

در دستورات زیر معادله دیفرانسیل با چند روش مختلف که از جدول ۱۰.۸ انتخاب شده‌اند حل شده است و بخشی از نمودارها در یک شکل رسم شده‌اند. به تفاوت میان جواب‌ها دقت کنید. این تصویر برای بخش کوچکی از جواب بزرگ‌نمایی شده است تا اختلاف دیده شود، در حالت رسم عادی، هر سه نمودار تقریباً بر هم منطبق می‌باشند.

```
ode = @(t,y) (t.^3 - 2*y)./t;
hold on;
[t,y] = ode45(ode,[1:0.1:3],4);
plot(t,y,'-r','LineWidth',1.5);
[t,y] = ode23(ode,[1:0.1:3],4);
plot(t,y,'--b','LineWidth',1.5);
[t,y] = ode15s(ode,[1:0.1:3],4);
plot(t,y,'-.g','LineWidth',1.5);
legend('ode45','ode23','ode15s','Location','Northwest');
```



مثال ۱۰.۸. معادله دیفرانسیل زیر را در نظر بگیرید،

$$\begin{cases} xy' + 2y = \sin x, & \frac{\pi}{4} \leq x \leq 2\pi, \\ y\left(\frac{\pi}{4}\right) = 0. \end{cases}$$

می‌دانیم جواب تحلیلی این معادله دیفرانسیل به صورت

$$y = \frac{1}{x^2} \sin x - \frac{1}{x} \cos x - \frac{1}{x^2},$$

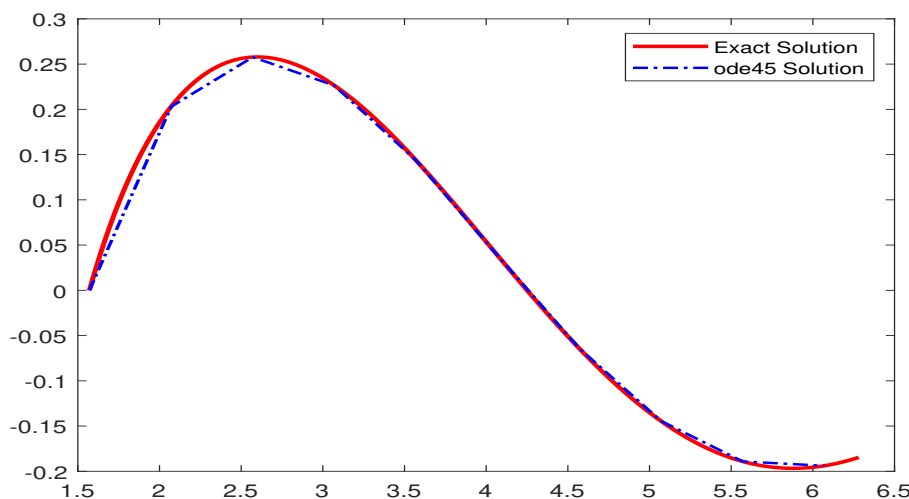
می‌باشد. جواب عددی را با روش ode45 بیابید و هر دو نمودار را در یک شکل رسم کنید.

ابتدا معادله را به شکل استاندارد می‌نویسیم، $y' = \frac{\sin x}{x} - \frac{2}{x}y$. سپس با دستورات زیر جواب عددی را تولید کرده و رسم نمودارها را انجام می‌دهیم.

```
x = pi/2:0.01:2*pi;
y1 = 1./(x.^2).*sin(x) - 1./x .*cos(x) - 1./(x.^2);
plot(x,y1,'-r','LineWidth',2);
ode = @(x,y) sin(x)./x - (2./x).*y;
[t,y] = ode45(ode,[pi/2:0.5:2*pi],0);
hold on;
```

```
plot(t,y,'-b','LineWidth',1.4);
legend('Exact Solution','ode45 Solution')
```

خروجی حاصل از دستورات بالا به شکل زیر می باشد. پیشنهاد می شود در دستور محاسبه جواب عددی، طول گام را از 0.5 به 0.01 تغییر داده. خروجی را مشاهده کنید. خواهید دید که جواب عددی و جواب تحلیلی تقریباً بر یکدیگر منطبق می شوند.



برای حل معادلات دیفرانسیل روش های دیگری نیز وجود دارد که در فصل های بعدی بیان خواهند شد. همچنین امکان حل تحلیلی و حل معادلات دیفرانسیل از مرتبه های بالاتر از یک نیز وجود دارد که به آنها نیز خواهیم پرداخت. به این منظور باید از بسته symbolic استفاده کنیم.

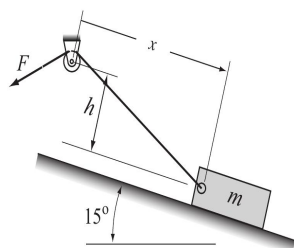
۳.۸ تمرین

تمرین ۱.۸. تمام جواب های معادلات زیر را بدست آورید. به این منظور ابتدا نمودار آنها را رسم کنید و پس از تعیین حدود ریشه ها دستورات لازم را بکار ببرید.

$$e^{\circ/\Delta x} - \sqrt{x} = 3, \quad 3 + 3 \sin x = \circ/\Delta x^3, \quad x^3 - 8x^2 + 17x = \sqrt{x} = 10$$

$$x^2 - 5x \sin 3x + 3 = \circ, \quad x \sin(x+1) + 2\sqrt{x} - 1 = \circ.$$

تمرین ۲.۸.



جسمی به جرم $m = ۲۰ \text{ Kg}$ مطابق شکل مقابل توسط یک کابل کشیده می‌شود. نیروی مورد نیاز برای کشیده شده جرم با فرمول زیر داده شده است،

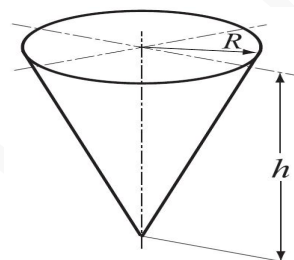
$$F = \frac{(\mu mg \cos 15^\circ + mg \sin 15^\circ) \sqrt{x^2 + h^2}}{x + \mu h}$$

که در آن $h = ۸ \text{ m}$ و $\mu = ۰.۴۵$ ضریب اصطکاک می‌باشد و $g = ۹.۸۱ \text{ m/s}^2$ نیروی جاذبه است. برای حالتی که $F = ۲۳۰$ نیوتون باشد، مقدار x را بدست آورید.

تمرین ۳.۸. مقادیر ماکزیمم و مینیمم تابع زیر را بدست آورید.

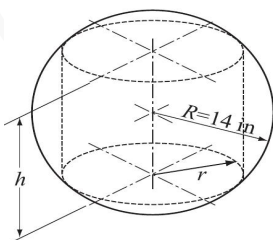
$$f(x) = \frac{x - ۲}{[(x - ۲)^4 + ۲]^{1/8}}.$$

تمرین ۴.۸.



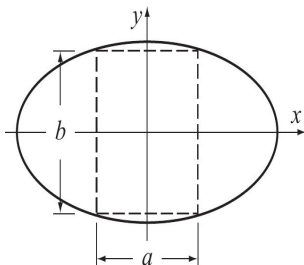
یک کاغذ به که به صورت مخروط مقابل شکل داده شده است دارای حجمی معادل ۲۵ cm^3 می‌باشد. شعاع R و ارتفاع h را به گونه‌ای تعیین کنید که کمترین مقدار کاغذ ممکن برای ساخت مخروط مصرف شود.

تمرین ۵.۸.



مقادیر h و r را به گونه‌ای تعیین کنید که استوانه محاط در کره به شعاع $R = ۱۴$ ، بیشترین حجم ممکن را داشته باشد. این حجم چقدر است؟

تمرین ۶.۸.



بیضی $\frac{x^2}{19^2} + \frac{y^2}{5^2} = 1$ را در نظر بگیرید. مقادیر a و b را به گونه‌ای تعیین کنید که مستطیل محاط در بیضی بیشترین مساحت ممکن را داشته باشد.

تمرین ۷.۸. مقادیر انتگرال‌های معین زیر را حساب کنید.

$$\int_1^6 \frac{x^2}{\sqrt{1+x}} dx, \quad \int_1^2 \frac{\cos^2 x}{x} dx, \quad \int_1^2 \frac{e^{2x}}{x} dx, \quad \int_{-1}^1 e^{-x^2} dx$$

تمرین ۸.۸. سرعت یک ماشین مسابقه در ۷ ثانیه اول حرکت در جدول زیر آورده شده است،

$t(s)$	۰	۱	۲	۳	۴	۵	۶	۷
$v(mi/h)$	۰	۱۴	۳۹	۶۹	۹۵	۱۱۴	۱۲۹	۱۳۹

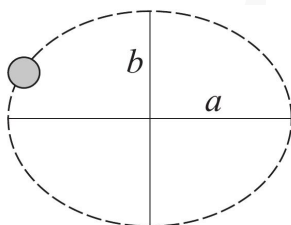
مسافت طی شده توسط اتوموبیل در ۶ ثانیه اول را محاسبه کنید.

تمرین ۹.۸. طول یک منحنی با معادلات پارامتری $x(t)$, $y(t)$ با فرمول زیر محاسبه می‌شود،

$$\int_a^b \sqrt{[x'(t)]^2 + [y'(t)]^2} dt.$$

اگر معادلات پارامتری یک سیکلوئید به شکل $x(t) = R(t - \sin t)$ و $y(t) = R(1 - \cos t)$ باشد، طول سیکلوئید را به ازای $R = 8cm$ برای $0 \leq t \leq 2\pi$ بدست آورید.

تمرین ۱۰.۸.



مدار سیاره پلوتو به شکل بیضی مقابل می‌باشد که در آن $a = 5.9065 \times 10^9 km$ و $b = 5.7208 \times 10^9 km$. اگر محیط بیضی با فرمول زیر محاسبه شود،

$$P = 4a \int_0^{\pi/2} \sqrt{1 - k^2 \sin^2 \theta} d\theta, \quad k = \frac{\sqrt{a^2 - b^2}}{a}$$

مسافتی که سیاره پلوتو در یک دور کامل مدار خود طی می‌کند را بدست آورید.

تمرین ۱۱.۸. انتگرال‌های فرسnel^۱ به شکل زیر می باشند،

$$S(x) = \int_0^x \sin(t^2) dt, \quad C(x) = \int_0^x \cos(t^2) dt.$$

$S(x)$ و $C(x)$ را برای $0 \leq x \leq 4$ محاسبه کنید.

تمرین ۱۲.۸. مسایل مقدار اولیه زیر را حل کنید و نمودار هر یک از جواب‌های حاصل را در یک نمودار جداگانه رسم کنید.

۱. $\frac{dy}{dx} = \sqrt{x} + \frac{x^2\sqrt{y}}{4} \quad 1 \leq x \leq 5, \quad y(1) = 1$

۲. $\frac{dy}{dx} = \sqrt{xy} - 0.5ye^{-0.1x} \quad 0 \leq x \leq 4 \quad y(0) = 6/5$

۳. $\frac{dy}{dt} = 1.0e^{-0.6t} \cos(4t) - 0.4y \quad 0 \leq t \leq 4 \quad y(0) = 0$

¹Fresnel integrals

نسخه
رایگان

۹ کاربرد MATLAB در جبر خطی

در فصل ۳ برخی از کاربردهای MATLAB را در جبر خطی بیان کردیم و با تولید بردار و ماتریس و انجام عملیات روی ماتریس‌ها و آرایه‌ها آشنا شدیم. در این فصل به شکل کامل‌تر به استفاده از MATLAB در جبر خطی عددی می‌پردازیم. پیشنهاد می‌شود پیش از ادامه مطالعه این فصل به فصل‌های ۳ و ۴ مراجعه کنید و مفاهیم پایه‌ای مانند تعریف ماتریس‌ها، انجام چهار عمل اصلی بر ماتریس‌ها، محاسبه دترمینان و وارون ماتریس، حل دستگاه‌های معادلات خطی و برخی ماتریس‌های خاص که در آن‌ها بیان شده است را یک بار دیگر مرور کنید سپس به ادامه مطالعه این فصل بپردازید.

۱.۹ تجزیه ماتریس‌ها

در جبر خطی برای تجزیه یک ماتریس مربعی به حاصل ضرب دو ماتریس روش‌های گوناگونی وجود دارد. از بین روش‌های موجود، سه روش از اهمیت و کاربرد ویژه‌ای برخوردار هستند که هر سه روش را می‌توان به سادگی در MATLAB پیاده‌سازی کرد.

۱.۱.۹ تجزیه LU

یک ماتریس مربعی $A_{n \times n}$ را می‌توان به شکل حاصل ضرب $A = LU$ تجزیه کرد که در آن L یک ماتریس پایین مثلثی با عناصر قطری ۱ و U یک ماتریس بالا مثلثی می‌باشد. برای مثال،

$$\begin{bmatrix} 1 & 1 & -1 \\ 6 & 2 & 2 \\ -3 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -0.5 & 1 & 0 \\ 0.1667 & 0.1333 & 1 \end{bmatrix} \begin{bmatrix} 6 & 2 & 2 \\ 0 & 5 & 2 \\ 0 & 0 & -1.6 \end{bmatrix}$$

تجزیه LU در MATLAB

در MATLAB با دستور `lu` می‌توان ماتریس مربعی A را به شکل $A = LU$ تجزیه کرد. روش اجرای دستور به یکی از دو شکل زیر می‌باشد،

$$[L,U] = lu(A)$$

$$[L,U,P] = lu(A)$$

در هر دو دستور بالا ماتریس U یکسان می‌باشد ولی

- در دستور اول L به شکل ماتریسی است که شاید پایین مثلثی نباشد ولی با تعویض جای برخی سطرها ماتریس پایین مثلثی خواهد شد. در این صورت رابطه $A = LU$ برقرار می‌باشد.

- در دستور دوم، ماتریس L یک ماتریس پایین مثلثی است و ماتریس P یک ماتریس جایگشت خواهد بود. در این صورت رابطه $A = P'LU$ برقرار است.

مثال ۱.۹. ماتریس زیر را با هر دو دستور تجزیه کنید.

$$\begin{bmatrix} 1 & 1 & -1 \\ 6 & 2 & 2 \\ -3 & 4 & 1 \end{bmatrix}$$

با دستورات زیر تجزیه انجام می‌شود.

$$A = [1,1,-1;6,2,2;-3,4,1];$$

$$[L,U] = lu(A)$$

$$[L,U,P] = lu(A)$$

که منجر به خروجی زیر می‌گردد.

$$L =$$

0.1667	0.1333	1.0000
1.0000	0	0
-0.5000	1.0000	0

U =

```
6.0000    2.0000    2.0000
0    5.0000    2.0000
0         0   -1.6000
```

L =

```
1.0000         0         0
-0.5000    1.0000         0
0.1667    0.1333    1.0000
```

U =

```
6.0000    2.0000    2.0000
0    5.0000    2.0000
0         0   -1.6000
```

P =

```
0    1    0
0    0    1
1    0    0
```

👉 در دستور تجزیه دوم، به ماتریس P ماتریس جایگشت گفته می‌شود که با ضرب کردن آن در ماتریس L ، جای برخی از سطرها را ماتریس L عوض می‌شود.

👉 در دستورات بالا بعد از دستور تجزیه اول عبارت $L*U$ و بعد از دستور تجزیه دوم عبارت $P'*L*U$ را وارد کنید و فایل را اجرا کنید. در هر دو حالت حاصل ماتریس A می‌باشد.

در صورت تجزیه ماتریس A به شکل $A = LU$ با دستور تجزیه اول، برای انجام برخی محاسبات می‌توان دستورات زیر را بکار برد،

- برای حل دستگاه $Ax = b$ می‌توان از دستور $x = U \setminus (L \setminus b)$ استفاده کرد.
- برای محاسبه دترمینان A می‌توان از دستور $\det(L) * \det(U)$ استفاده کرد.
- برای محاسبه وارون ماتریس A می‌توان از دستور $\text{inv}(U) * \text{inv}(L)$ استفاده کرد.

۲.۱.۹ تجزیه QR

از جبر خطی می‌دانیم که اگر اندازه ستون‌های ماتریس Q مقدار یک داشته باشند و هر دو ستون متمایز بر هم عمود باشند، آنگاه ماتریس Q متعامد نامیده می‌شود. در این صورت رابطه $QQ' = I$ برقرار است.

در جبر خطی عددی یک نوع تجزیه برای ماتریس مربعی $n \times n$ وجود دارد که به تجزیه QR موسوم است. در این تجزیه ماتریس A به شکل حاصل ضرب $A = QR$ تجزیه می‌شود که در آن Q یک ماتریس متعامد و R یک ماتریس بالا مثلثی می‌باشند.

تجزیه QR

در MATLAB تجزیه QR ماتریس A با دستور

$$[Q,R] = \text{qr}(A)$$

انجام می‌شود که در آن Q یک ماتریس متعامد و R یک ماتریس بالا مثلثی می‌باشد.

مثال ۲.۹. به تجزیه QR مثال پیش توجه کنید.

ورودی

```
A = [1 1 -1
      6 2 2
      -3 4 1];
[Q,R] = qr(A)
```

خروجی

```
Q =
-0.1474    -0.2136    0.9657
-0.8847    -0.4082   -0.2253
 0.4423   -0.8876   -0.1288

R =
-6.7823   -0.1474   -1.1795
 0   -4.5802   -1.4903
 0         0   -1.5452
```

اگر پس تجزیه ماتریس، دستور $Q*Q'$ را اجرا کنید ماتریس همانی حاصل می‌شود.

۳.۱.۹ تجزیه چولسکی

در تجزیه چولسکی^۱ ماتریس معین مثبت $A_{n \times n}$ به شکل حاصل ضرب ماتریس قطری R در ترانهاد R تجزیه می شود، یعنی $A = R'R$.

تجزیه چولسکی

اگر A یک ماتریس معین مثبت باشد با دستور $R = \text{chol}(A)$ می توان ماتریس بالامثلی R را به گونه ای یافت که رابطه $A = R'R$ برقرار باشد.

مثال ۳.۹. به تجزیه چولسکی ماتریس در دستورات زیر توجه کنید. پیشنهاد می شود پس از دستور تجزیه، دستور $R' * R$ را بنویسید و خروجی حاصل را مشاهده کنید.

ورودی	خروجی
<pre>A = [1 1 -1 6 3 2 -3 4 8]; R = chol(A)</pre>	<pre>R = 1.0000 1.0000 -1.0000 0 1.4142 2.1213 0 0 1.5811</pre>

برای ماتریس ها انواع دیگری از تجزیه نیز وجود دارد که نیاز به برخی مفاهیم دارند. در ادامه پس از بیان مطالب مورد نیاز آن ها را بیان خواهیم کرد.

۴.۱.۹ مقادیر ویژه و بردارهای ویژه

در MATLAB می توان با دستوری ساده مقادیر ویژه و بردارهای ویژه یک ماتریس را محاسبه کرد. از جبرخطی می دانیم که اگر λ یک کمیت اسکالری، v یک بردار n تایی و A یک ماتریس $n \times n$ باشند، به گونه ای که رابطه $Av = \lambda v$ برقرار باشد، آنگاه λ مقدار ویژه و v بردار ویژه متناظر با λ برای ماتریس A نامیده می شوند. همچنین می دانیم که یک ماتریس $n \times n$ دارای n مقدار ویژه حقیقی و مختلط است که این مقادیر می توانند تکراری نیز باشند. در شکل کلی اگر $\Lambda_{n \times n}$ یک ماتریس قطری در نظر بگیریم که عناصر روی قطر آن مقادیر ویژه ماتریس a باشند و V یک

¹Cholesky decomposition

ماتریس $n \times n$ باشد که ستون j ام آن بردار ویژه متناظر با λ_j باشد، آنگاه رابطه $AV = V\Lambda$ برقرار می‌باشد.

محاسبه مقادیر ویژه

اگر $A_{n \times n}$ یک ماتریس مربعی باشد آنگاه با دستور

$$[V,D] = \text{eig}(A)$$

تمام مقادیر ویژه و بردارهای ویژه ماتریس A محاسبه می‌شوند. در این دستور

D یک ماتریس قطری $n \times n$ است که عناصر روی قطر آن مقادیر ویژه ماتریس A خواهند بود.

V یک ماتریس $n \times n$ است که ستون j ام آن، بردار ویژه متناظر با مقدار ویژه j ام ماتریس A است.

مثال ۴.۹. مقادیر ویژه و بردارهای ویژه ماتریس $A = \begin{bmatrix} -1 & 2 & 3 \\ 2 & 3 & -4 \\ 0 & -2 & 1 \end{bmatrix}$ را بدست آورید.

ورودی

```
A = [-1,2,3
      2,3,-4
      0,-2,1];
[V,D] = eig(A)
A*V(:,1);
D(1,1)*V(:,1);
```

خروجی

```
V =
0.0796    -0.8604    0.8791
0.8967     0.4527    0.0594
-0.4354     0.2341    0.4729

D =
5.1196     0     0
0    -2.8686     0
0     0    0.7490
```

اگر دستورات فوق را اجرا کنید، حاصل دو دستور آخر یکسان خواهد شد، زیرا این دو دستور دو طرف رابطه $Av_1 = \lambda_1 v_1$ را محاسبه می‌کنند. برای اجرا سمی کالن انتهای دو دستور آخر را بردارید.

با توجه به رابطه $AV = V\Lambda$ و با فرض وارون پذیر بودن V می توان با ضرب طرفین رابطه اخیر در V^{-1} از سمت راست به رابطه $A = V\Lambda V^{-1}$ رسید که به تجزیه مقدار ویژه موسوم می باشد.

۵.۱.۹ مقادیر تکین و بردارهای تکین

فرض کنید $A_{m \times n}$ یک ماتریس مستطیلی باشد و اسکالر σ و بردارهای $u_{m \times 1}$ و $v_{n \times 1}$ به گونه ای موجود باشند که روابط $Av = \sigma u$ یا $A^T u = \sigma v$ برقرار باشند، آنگاه به σ مقدار تکین و به u, v بردارهای تکین ماتریس A گفته می شود.

حال اگر فرض کنید مقادیر تکین ماتریس $A_{m \times n}$ روی قطر ماتریس قطری $\Sigma_{m \times n}$ قرار گرفته باشند و بردارهای تکین متناظر با مقادیر تکین، ستون های دو ماتریس متعامد $U_{m \times m}$ و $V_{n \times n}$ باشند، آنگاه روابط $AV = U\Sigma$ و $A^T U = V\Sigma$ برقرار می باشند. حال چون دو ماتریس U, V متعامد هستند لذا می توان تجزیه $A = U\Sigma V^T$ را برای ماتریس $A_{m \times n}$ بدست آورد.

روش تجزیه مقدار تکین

اگر $A_{m \times n}$ یک ماتریس مستطیلی باشد، با دستور زیر می توان مقادیر تکین و بردارهای تکین متناظر با ماتریس A را یافت،

$$[U, S, V] = \text{svd}(A)$$

که در آن

U یک ماتریس متعامد $m \times m$ است که ستون های آن تشکیل بردارهای تکین متناظر با یک مقدار تکین می دهند.

V یک ماتریس متعامد $n \times n$ است که ستون های آن تشکیل بردارهای تکین متناظر با یک مقدار تکین می دهند.

S یک ماتریس $m \times n$ است که مقادیر تکین بر روی قطر اصلی آن قرار گرفته اند.

مثال ۵.۹. تجزیه مقدار تکین ماتریس $A = \begin{bmatrix} 9 & 4 \\ 6 & 8 \\ 7 & 7 \end{bmatrix}$ را پیدا کنید.

با دستورات زیر می توان تجزیه را انجام داد. توجه کنید در دستور آخر تجزیه مقدار تکین انجام شده آزمایش شده است و با ضرب کردن ماتریس ها ماتریس A محاسبه شده است.

ورودی	خروجی
<pre>A = [9,4 6,8 2,7] [U,S,V] = svd(A) B = U*S*V'</pre>	<pre>U = -0.6105 0.7174 0.3355 -0.6646 -0.2336 -0.7098 -0.4308 -0.6563 0.6194 S = 14.9359 0 0 5.1883 0 0 V = -0.6925 0.7214 -0.7214 -0.6925 B = 9.0000 4.0000 6.0000 8.0000 2.0000 7.0000</pre>

۲.۹ توابع مورد استفاده در ماتریس‌ها

در MATLAB توابع زیادی وجود دارند که در ارتباط با ماتریس‌ها هستند. آگاهی از وجود این توابع و استفاده از آنها باعث می‌شود تا از بازنویسی برخی عملیات که روی ماتریس‌ها و بردارها انجام شود بی‌نیاز شویم. در این بخش به معرفی برخی از توابع پرکاربرد در عملیات ماتریس می‌پردازیم. در انتهای این بخش به معرفی دستورات مرتبط با ماتریس‌های اسپارس می‌پردازیم و با چگونگی تولید، ذخیره‌سازی و کار با ماتریس‌های اسپارس آشنا خواهیم شد. توجه کنید که در این بخش مثال‌های زیادی برای دستورات زده نشده است، لذا پیشنهاد می‌شود تا تمامی دستورات این بخش را برای ورودی‌های مختلف در محیط MATLAB مورد استفاده قرار دهید.

تابع norm

تابع norm را می‌توان هم برای بردارها و هم برای ماتریس‌ها به شکل‌های زیر بکار برد،
 $\text{norm}(X)$
 $\text{norm}(X,p)$
 که در آنها، X می‌تواند یک بردار یا ماتریس باشد.

• اگر $X = v$ و یک بردار فرض شود، آنگاه دستور $\text{norm}(v)$ نرم اقلیدسی است و به شکل
 $\|v\| = \sqrt{\sum_{k=1}^N |v_k|^2}$ محاسبه می‌شود که در آن N تعداد عناصر بردار است.

• اگر $X = v$ و یک بردار فرض شود، آنگاه دستور $\text{norm}(v,p)$ مقدار p -نرم را به شکل
 $\|v\|_p = \left[\sum_{k=1}^N |v_k|^p \right]^{1/p}$ محاسبه می‌کند. p می‌تواند هر عدد حقیقی مثبت یا بی‌نهایت باشد و N تعداد عناصر بردار است.

• اگر X یک ماتریس باشد، آنگاه دستور $\text{norm}(X)$ مقدار نرم دو را محاسبه می‌کند که به شکل بیشترین مقدار تکین ماتریس X می‌باشد، یعنی حاصل این دستور با مقدار دستور $\max(\text{svd}(X))$ یکسان است.

• اگر X یک ماتریس $m \times n$ باشد، آنگاه دستور $\text{norm}(X,p)$ مقدار p -نرم ماتریس X را محاسبه می‌کند که p می‌تواند مقادیر 1، 2 یا Inf داشته باشد، در این صورت از روابط زیر برای محاسبه نرم استفاده می‌شود.

$$\|X\|_1 = \max_{1 \leq j \leq n} \left(\sum_{i=1}^m |a_{ij}| \right), \quad \|X\|_\infty = \max_{1 \leq i \leq m} \left(\sum_{j=1}^n |a_{ij}| \right).$$

حالت $p = 2$ معادل با $\text{norm}(X)$ می‌باشد.

مثال ۶.۹. به خروجی حاصل از دستورات نوشته برای محاسبه نرم توجه کنید.

ورودی

```
A = [9,4;6,8;2,7]; a = norm(A,1)
v = [2,3,-1]; b = norm(v)
```

خروجی

```
a = 19
b = 3.7417
```

📌 بجای دستور $\text{norm}(X, p)$ می‌توان از دستور $\text{normest}(X)$ استفاده کرد. این دستور مقدار نرم دو بردار یا ماتریس X را برمی‌گرداند. دستورات دیگری نیز وجود دارند که به شکل خلاصه در جدول ۱.۹ آورده شده‌اند.

جدول ۱.۹: دستورات مرتبط با ماتریس‌ها در MATLAB

دستور	شرح	مثال
rank	تعیین رتبه ماتریس	<pre>>> A = [1,5;-2,3]; >> rank(A) ans = 2</pre>
det	محاسبه دترمینان ماتریس	<pre>>> A = [1,5;-2,3]; >> det(A) ans = 13</pre>
trace	محاسبه مجموعه عناصر روی قطر	<pre>>> A = [1,5;-2,3]; >> trace(A) ans = 4</pre>
orth	متعامد سازی ماتریس	<pre>>> A = [1,5;-2,3]; >> orth(A) ans = 0.8507 0.5257 0.5257 -0.8507</pre>
inv	محاسبه ماتریس وارون	<pre>>> A = [1,5;-2,3]; >> inv(A) ans = 0.2308 -0.3846 0.1538 0.0769</pre>

همچنین دستورات دیگری نیز وجود دارند که در جدول ۲.۹ آورده شده است. علاوه بر دستورات

این دو جدول، دستورات دیگری نیز وجود دارند که برای مشاهده و استفاده از آنها می‌توانید به کتاب‌های جامعی که در زمینه آموزش MATLAB وجود دارند مراجعه کنید.

جدول ۲.۹: دستورات مرتبط با ماتریس‌ها در MATLAB

دستور	شرح	مثال
cond	محاسبه عدد شرطی	<pre>>> A = [1,5;-2,3]; >> cond(A) ans = 2.6180</pre>
condest	محاسبه عدد شرطی	<pre>>> A = [1,5;-2,3]; >> condest(A) ans = 2.6180</pre>
poly	محاسبه چندجمله‌ای‌های مشخصه	<pre>>> A = [1,5;-2,3]; >> poly(A) ans = 1.0000 -4.0000 13.0000</pre>

در انتهای این بخش به بیان چند دستور پیرامون تولید ماتریس‌های اسپارس خواهیم پرداخت و به چگونگی استفاده از این نوع ماتریس‌ها اشاره می‌کنیم. از جبر خطی می‌دانیم که یک ماتریس را اسپارس گوئیم هرگاه تعداد عناصر صفر آن نسبت به عناصر غیر صفر آن خیلی بیشتر باشد. مثلاً اگر در یک ماتریس 100×100 تنها 50 عنصر ناصفر داشته باشیم با یک ماتریس اسپارس مواجه هستیم.

تولید ماتریس اسپارس

پس از تولید ماتریس اسپارس A برای ذخیره‌سازی آن می‌توان از دستور `sparse(A)` و برای نمایش کامل ماتریس می‌توان از دستور `full(A)` استفاده کرد.

📌 پس از ذخیره‌سازی ماتریس‌های اسپارس می‌توان مانند ماتریس‌های عادی با آنها رفتار کرد و

تمام عملیات، مانند جمع، تفریق، ضرب و ... را روی آنها انجام داد.

مثال ۷.۹. برنامه زیر یک ماتریس اسپارس $N \times N$ تولید می‌کند و آن را به شکل اسپارس ذخیره کرده و نمایش می‌دهد.

```
N = 10; A = zeros(N,N);
for k = 1:N
    m = randi(N,1);
    n = randi(N,1);
    A(m,n) = randi(N,1);
end
B = sparse(A)
full(A)
```

خروجی دستورات بالا به شکل زیر می باشد.

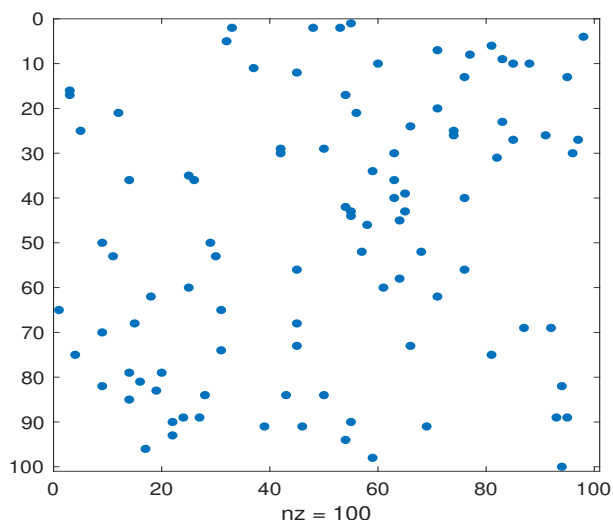
```
B =
(2,2)      5
(3,3)      7
(9,3)      6
(3,4)      6
(8,4)      6
(1,5)      4
(1,6)      5
(7,7)      7
(3,9)     10
(2,10)     9

ans =
0    0    0    0    4    5    0    0    0    0
0    5    0    0    0    0    0    0    0    9
```


0	0	7	6	0	0	0	0	10	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	7	0	0	0
0	0	0	6	0	0	0	0	0	0
0	0	6	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

دستور `spy(A)` یک نمایش گرافیکی از توزیع نقاط ناصفر در سرتاسر ماتریس تولید می‌کند.

برای مثال اگر برای $N = 100$ برنامه بالا را اجرا کنیم نمودار زیر حاصل خواهد شد.



برنامه را برای اعداد بزرگتر اجرا کنید و خروجی گرافیکی را مشاهده کنید.

نسخه
دایگان

۱۰ رسم نمودارهای سه بعدی

در MATLAB امکان رسم نمودار در دستگاه مختصات فضایی نیز وجود دارد. برخی از دستورات مربوط رسم توابع در فضا شبیه به دستوراتی است که در فصل ۶ برای رسم توابع در دستگاه مختصات دوبعدی بیان کردیم و برخی از دستورات که برای رسم رویه‌ها در فضا بکار می‌روند کاملاً جدید هستند. در این فصل به رسم نمودارها و رویه‌ها در فضا می‌پردازیم.

۱.۱۰ رسم منحنی‌های خط

برای رسم یک منحنی خط در فضا، می‌توان از دستوری کاملاً مشابه با دستور plot که در فصل ۶ دیدیم، استفاده کرد.

دستور plot3

برای رسم منحنی خط در فضا از دستور plot3 به شکل کلی زیر می‌توان استفاده کرد.

```
plot3(x,y,z,'Line Specifiers','Property Name',property value)
```

استفاده کرد که در آن

- x ، y و z سه بردار هم‌اندازه هستند.

- `Property Value`، `Line Specifiers` و `property value` همان‌هایی هستند

که در جداول ۱.۶ تا ۵.۶ آورده شده‌اند.

مثال ۱۰.۱. نمودار توابع زیر که به شکل پارامتری در فضا تعریف شده‌اند را در بازه‌های $t \in [0, 10\pi]$ و $[0, 10]$ در دو دستگاه مختصات جداگانه رسم کنید.

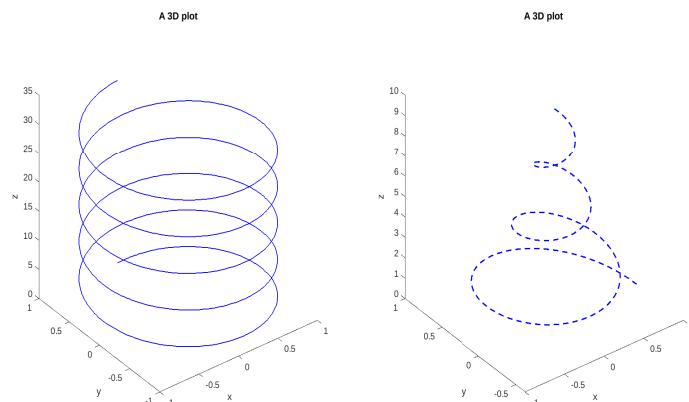
$$\begin{aligned} x(t) &= \sin t, & y(t) &= \cos t, & z(t) &= t, \\ x(t) &= -\cos t, & y(t) &= \sin t, & z(t) &= t \end{aligned}$$

با دستورات زیر می‌توان نمودار را در فضا رسم کرد.

```
t = linspace(0,10*pi,1000);
x = sin(t); y = cos(t); z = t;
subplot(1,2,1)
plot3(x,y,z,'-b','LineWidth',1.25);
title('A 3D plot');
xlabel('x'); ylabel('y'); zlabel('z');

t = 0:0.05:10;
x = exp(-0.2*t).*cos(2*t);
y = exp(-0.2*t).*sin(2*t);
z = t;
subplot(1,2,2)
plot3(x,y,z,'--b','LineWidth',1.5);
title('A 3D plot');
xlabel('x'); ylabel('y'); zlabel('z');
```

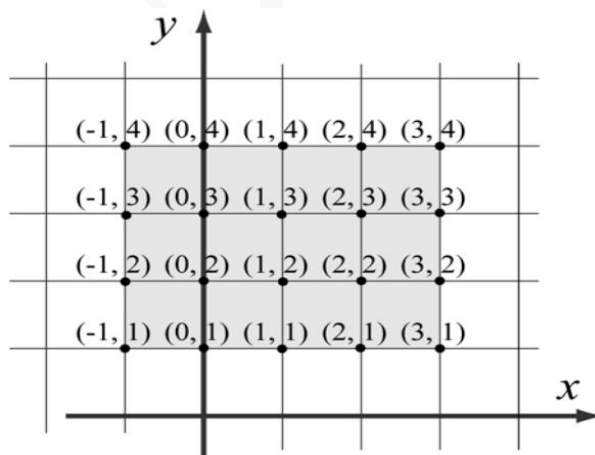
خروجی حاصل به شکل زیر می‌باشد. توجه کنید که تمام دستوراتی که برای قالب‌بندی شکل در دستور plot مورد استفاده قرار می‌گرفت، در این دستور نیز به همان شکل قابل استفاده است.



۲.۱۰ رسم رویه‌ها

در حساب دیفرانسیل و انتگرال چند متغیره دیدیم که تابع $z = f(x, y)$ که در آن x, y متغیرهای مستقل هستند و z متغیر وابسته می‌باشد، دارای نموداری در فضای ۳ بعدی است که به رویه موسوم است. برای رسم رویه‌ها در MATLAB نیاز انجام سه مرحله عملیات می‌باشد.

۱. ابتدا باید یک شبکه در صفحه xy به صورت شکل ۱.۱۰ ایجاد کنیم. این شبکه را می‌توان



شکل ۱.۱۰: نمونه‌ای از یک شبکه در صفحه xy

با دو ماتریس X و Y تولید کرد که ماتریس X شامل تمام مولفه‌های اول مختصات و

ماتریس Y شامل تمام مولفه‌های دوم مختصات می‌باشد. یعنی برای شکل ۱۰.۱ این دو ماتریس به شکل زیر می‌باشند.

$$X = \begin{bmatrix} -1 & 0 & 1 & 2 & 3 \\ -1 & 0 & 1 & 2 & 3 \\ -1 & 0 & 1 & 2 & 3 \\ -1 & 0 & 1 & 2 & 3 \end{bmatrix}, \quad \begin{bmatrix} 4 & 4 & 4 & 4 & 4 \\ 3 & 3 & 3 & 3 & 3 \\ 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

این دو ماتریس را در MATLAB می‌توان با دستور `meshgrid` تولید کرد. این دستور به شکل

$$[X,Y] = \text{meshgrid}(x,y)$$

مورد استفاده قرار می‌گیرد که در آن x و y دو بردار می‌باشند. برای مثال به دستورات زیر و خروجی حاصل از آن توجه کنید.

ورودی

```
x = -1:3;
y = 1:4;
[X,Y] = meshgrid(x,y)
```

خروجی

```
X =
-1    0    1    2    3
-1    0    1    2    3
-1    0    1    2    3
-1    0    1    2    3

Y =
1    1    1    1    1
2    2    2    2    2
3    3    3    3    3
4    4    4    4    4
```

۲. در گام دوم باید مقدار متغیر z را در هر نقطه از شبکه ایجاد شده پیدا کرد. برای مثال اگر $z = \frac{xy^2}{x^2+y^2}$ باشد، می‌توان با دستور زیر محاسبات لازم را انجام داد،

$$Z = X.*Y.^2./(X.^2 + Y.^2)$$

که منجر به خروجی زیر خواهد شد.

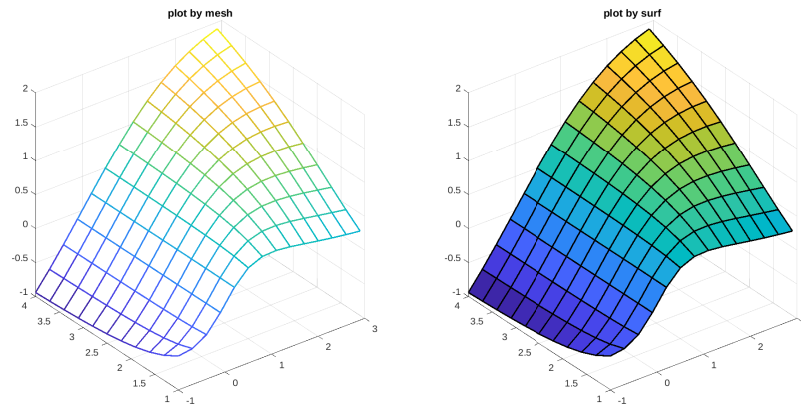
```
Z =
-0.5000      0    0.5000    0.4000    0.3000
-0.8000      0    0.8000    1.0000    0.9231
-0.9000      0    0.9000    1.3846    1.5000
-0.9412      0    0.9412    1.6000    1.9200
```

۳. در گام آخر با یکی از دو دستور $\text{mesh}(X,Y,Z)$ یا $\text{surf}(X,Y,Z)$ می‌توان رویه مورد نظر را رسم کرد.

مثال ۲.۱۰. با دستورات زیر رویه $z = \frac{xy^2}{x^2+y^2}$ برای $-1 \leq x \leq 3$ و $1 \leq y \leq 4$ در دو نمودار جداگانه رسم شده‌اند. به تفاوت میان خروجی دستورات surf و mesh توجه کنید.

```
x = -1:0.3:3; y = 1:0.3:4;
[X,Y] = meshgrid(x,y);
Z = X.*Y.^2./(X.^2 + Y.^2);
subplot(1,2,1); mesh(X,Y,Z,'LineWidth',1.5);
title('plot by mesh')
subplot(1,2,2); surf(X,Y,Z,'LineWidth',1.5);
title('plot by surf')
```

خروجی حاصل از دستورات بالا به شکل زیر است. توجه کنید نتیجه اجرای دستور surf شکلی رنگی خواهد بود ولی حاصل دستور mesh به شکل شبکه‌بندی شده و فاقد رنگ می‌باشد.



👉 در دستورات mesh و surf می‌توان از Property Value به همان شکلی که برای دستور plot در فصل ۶ استفاده کردیم، استفاده کنیم.

تغییر رنگ رویه

در حالت پیش فرض MATLAB از رنگ‌های مختلفی برای تولید رویه استفاده می‌کند ولی با دستور colormap(C) می‌توان رنگ‌ها دلخواهی را بکار برد. در این دستور C یک بردار سه‌تایی شامل سه عدد حقیقی می‌باشد که مولفه‌های اول، دوم و سوم آن به ترتیب نشان دهنده شدت رنگ‌های قرمز، سبز و آبی می‌باشند. این دستور را باید پس از دستورات mesh یا surf بکار برد.

👉 دستور colormap([0.5, 1, 1]) را در انتهای دستورات مثال پیش قرار دهید و نتیجه را بررسی کنید.

👉 توجه کنید که امکان استفاده از دستوراتی مانند hold off یا grid off و دستورات دیگری که رسم نمودارهای دوبعدی مورد استفاده بود نیز در حالت سه‌بعدی وجود دارد.

مثال ۳.۱۰. نمودار رویه

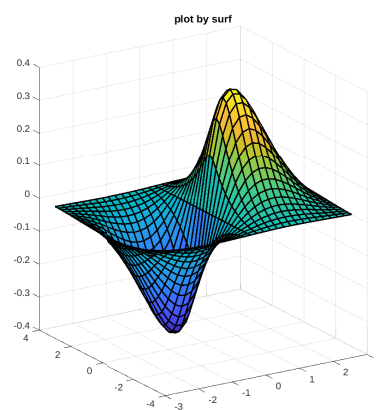
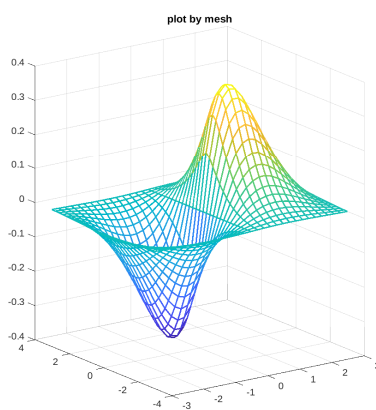
$$z = \frac{1}{\sqrt{8}} - \frac{1}{\sqrt{5}} \sqrt{x^2 + y^2} \sin x \cos(\pi/5 y),$$

را برای $x, y \in [-3, 3]$ رسم کنید.

با دستورات زیر نمودار را با استفاده از هر دو دستور mesh و surf می‌توان تولید کرد.


```
x = -3:0.2:3; y = -3:0.2:3;
[X,Y] = meshgrid(x,y);
Z = 1.8.^(-1.5*sqrt(X.^2 + Y.^2)).*sin(X).*cos(0.5*Y);
subplot(1,2,1); mesh(X,Y,Z,'LineWidth',1.5);
title('plot by mesh')
subplot(1,2,2); surf(X,Y,Z,'LineWidth',1.5);
title('plot by surf')
```

که منجر به نمودارهای زیر می شود.



دستورات mesh و surf دستورات پایه‌ای رسم رویه‌ها می‌باشند. دستوراتی دیگری نیز وجود دارند که با استفاده از آنها می‌توان از آنها برای رسم رویه‌ها به منظور خاصی استفاده کرد.

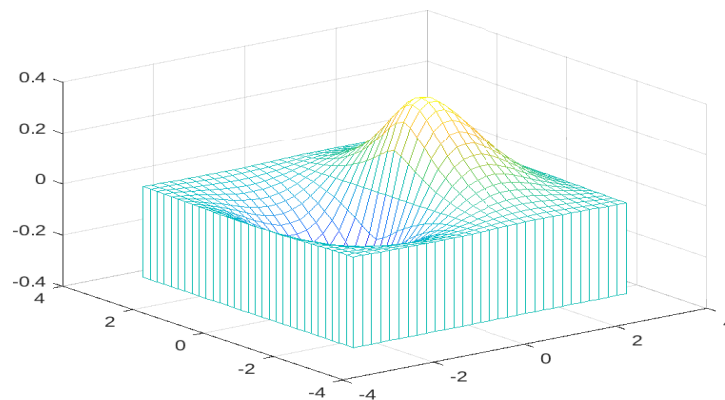
دستور meshz

با دستور meshz نموداری مشابه دستور mesh تولید می‌شود، با این تفاوت که اطراف مش با خطوطی عمودی از بقیه صفحه جدا خواهد شد.

مثال ۴.۱۰. دستورات زیر را در محیط MATLAB بنویسید و اجرا کنید.

```
x = -3:0.2:3; y = -3:0.2:3; [X,Y] = meshgrid(x,y);
Z = 1.8.^(-1.5*sqrt(X.^2 + Y.^2)).*sin(X).*cos(0.5*Y);
meshz(X,Y,Z);
```

خروجی به شکل زیر می‌باشد.



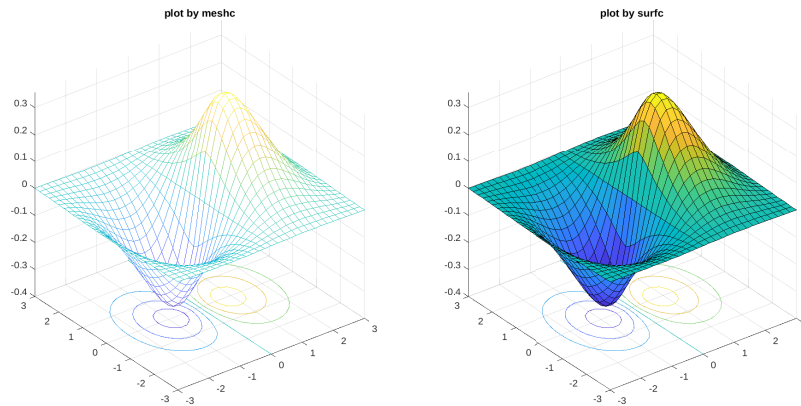
دستورات meshc و surfc

دو دستور meshc و surfc رویه را به همراه منحنی‌های کانتور رسم می‌کند. توجه کنید که نوع نمایش رویه و رنگ‌آمیزی رویه در دستورات اخیر مانند حالت پایه‌ای دستورات می‌باشد.

مثال ۵.۱۰. دستورات زیر را در محیط MATLAB بنویسید و خروجی را بدست آورید.

```
x = -3:0.2:3; y = -3:0.2:3; [X,Y] = meshgrid(x,y);
Z = 1.8.^(-1.5*sqrt(X.^2 + Y.^2)).*sin(X).*cos(0.5*Y);
subplot(1,2,1); meshc(X,Y,Z);
title('plot by meshc')
subplot(1,2,2); surfc(X,Y,Z);
title('plot by surfc')
```

خروجی دستورات بالا به شکل زیر است.



👉 در دو دستور meshc و surfc امکان استفاده از Property Value نیست و در صورت استفاده از آنها با خطا مواجه خواهید شد.

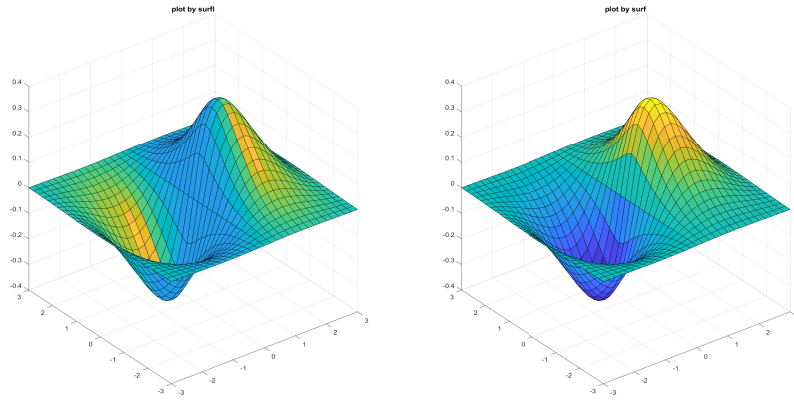
دستور surf

خروجی دستور surf مشابه دستور surfc می‌باشد و تنها نوع رنگ‌آمیزی رویه در دو دستور متفاوت هستند.

مثال ۶.۱۰. کد زیر را در محیط MATLAB بنویسید و نوع رنگ‌آمیزی در شکل تولید شده را با هم مقایسه کنید.

```
x = -3:0.2:3; y = -3:0.2:3; [X,Y] = meshgrid(x,y);
Z = 1.8.^(-1.5*sqrt(X.^2 + Y.^2)).*sin(X).*cos(0.5*Y);
subplot(1,2,1); surf1(X,Y,Z);
title('plot by surf1')
subplot(1,2,2); surf(X,Y,Z);
title('plot by surf')
```

نمودار حاصل از دستورات بالا به شکل زیر می‌باشد. چون خروجی هر دو شکل در چاپ سیاه و سفید، تفاوت ندارد، لذا دو رویه شبیه هم به نظر می‌رسند. برای مشاهده تغییرات دستورات بالا را در محیط MATLAB اجرا کنید.



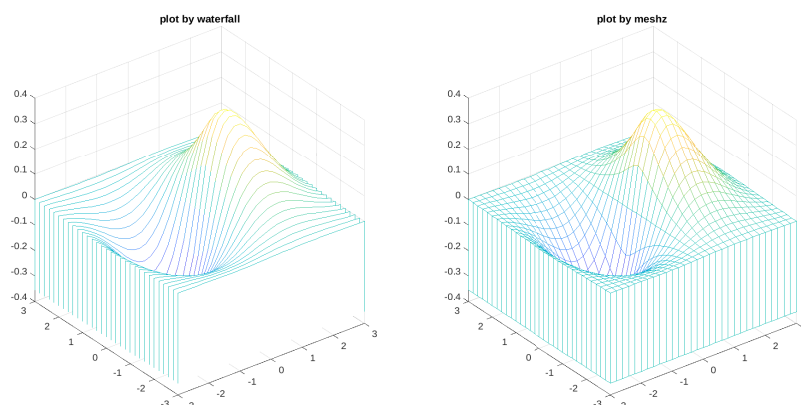
دستور waterfall

خروجی دستور waterfall شبیه خروجی دستور meshz می باشد، با این تفاوت که در دستور waterfall خطوط عمودی رسم نخواهند شد.

مثال ۷.۱۰. دستورات زیر در محیط MATLAB اجرا کنید.

```
x = -3:0.2:3; y = -3:0.2:3; [X,Y] = meshgrid(x,y);
Z = 1.8.^(-1.5*sqrt(X.^2 + Y.^2)).*sin(X).*cos(0.5*Y);
subplot(1,2,1); waterfall(X,Y,Z);
title('plot by waterfall');
subplot(1,2,2); meshz(X,Y,Z);
title('plot by meshz');
```

خروجی دستورات بالا به شکل زیر می باشد.



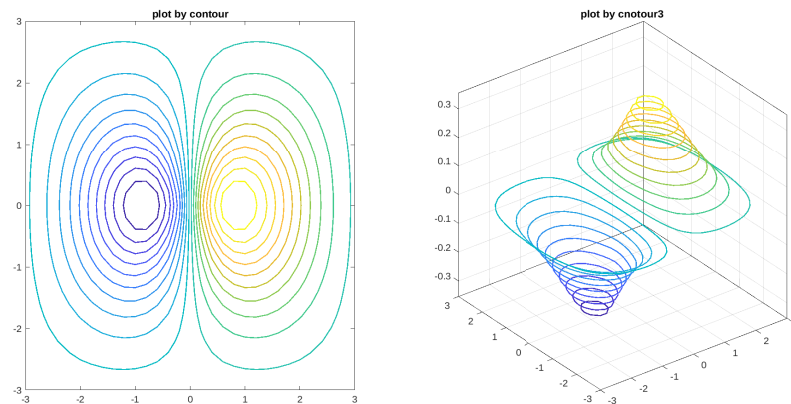
دستورات contour3 و contour

با دو دستور contour و contour3 می‌توان منحنی‌های کانتور را در صفحه و فضا تا n سطح رسم کرد. در این دو دستور پارامتر n به عنوان آرگومان ورودی چهارم به دستورات داده می‌شود.

مثال ۸.۱۰. دستورات زیر را در محیط MATLAB بنویسید و اجرا کنید.

```
x = -3:0.2:3; y = -3:0.2:3; [X,Y] = meshgrid(x,y);
Z = 1.8.^(-1.5*sqrt(X.^2 + Y.^2)).*sin(X).*cos(0.5*Y);
subplot(1,2,1); contour(X,Y,Z,20,'LineWidth',1.25);
title('plot by contour');
subplot(1,2,2); contour3(X,Y,Z,20,'LineWidth',1.25);
title('plot by cnotour3');
```

خروجی دستورات بالا به شکل زیر می باشد.



👉 در این دو دستور امکان استفاده از Property Value وجود دارد. همچنین در دستورات بالا به چگونگی وارد کردن پارامتر n به عنوان آرگومان چهارم توجه کنید.

۳.۱۰ رسم اشکال خاص در فضا

در MATLAB برای رسم برخی از شکل‌های خاص مانند کره و استوانه و برخی نمودارها مانند نمودارهای میله‌ای و دایره‌ای در حالت سه‌بعدی دستورات ساده‌ای وجود دارد که می‌توان از آنها بهره برد.

رسم کره

با دستور `shpere` می‌توان مجموعه نقاط لازم برای یک کره را تولید کرد و سپس با یکی از دستورات رسم رویه مانند دستور `surf` یا دستور `mesh`، کره را رسم نمود.

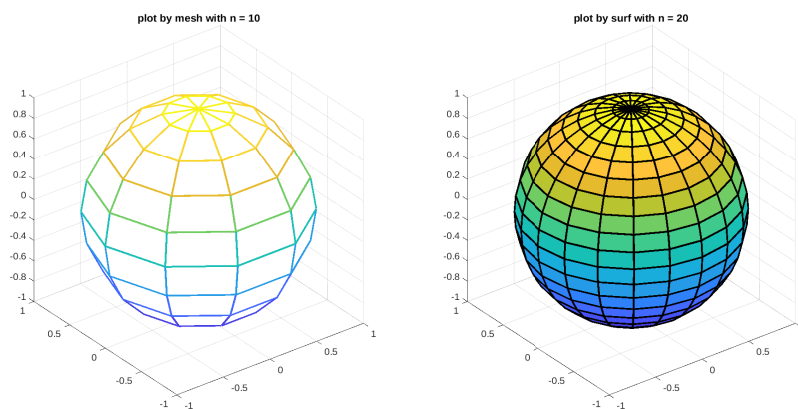
```
[X,Y,Z] = sphere(20)
surf(X,Y,Z)
```

مثال ۹.۱۰. با دستورات زیر دو کره با تعداد نقاط $n = 10$ و $n = 20$ رسم شده‌اند. در این دستورات از دستورات `surf` و `mesh` استفاده شده است.

```
[X,Y,Z] = sphere(10);
```

```
subplot(1,2,1); mesh(X,Y,Z,'LineWidth',2);
title('plot by mesh with n = 10');
[X,Y,Z] = sphere(20);
subplot(1,2,2); surf(X,Y,Z,'LineWidth',2);
title('plot by surf with n = 20');
```

خروجی دستورات بالا به شکل زیر می باشد.



رسم استوانه

برای رسم استوانه می توان از دستور `cylinder` و بدون هیچ آرگومانی استفاده کرد، ولی می توان با شکل دیگری از دستور `cylinder` اشکالی رسم کرد که استوانه استاندارد نیستند ولی شباهت هایی به استوانه می دهند. به این منظور باید از دستورات زیر استفاده کرد.

```
[X,Y,Z] = cylinder(2+cos(t));
surf(X,Y,Z)
```

بدیهی است که بجای $2 + \cos(t)$ هر تابع دیگری بر حسب t می توان قرار داد. این تابع باعث ایجاد تغییر شکل در استوانه می شود.

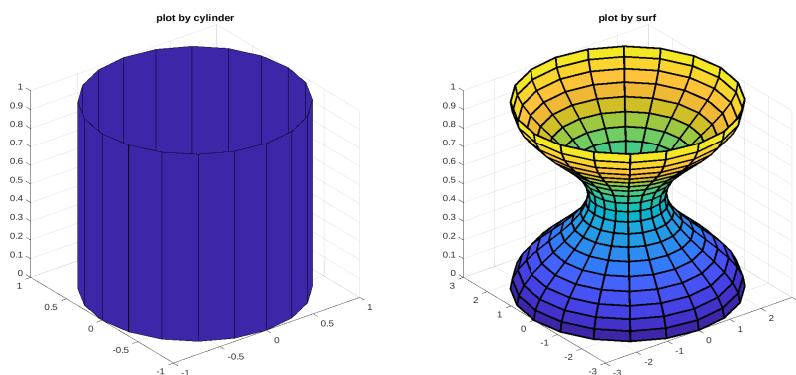
مثال ۱۰.۱۰. به خروجی حاصل از دستورات زیر توجه کنید. در این دستورات از هر دو شکل دستور `cylinder` استفاده شده است.

```

t = 0:pi/10:2*pi;
subplot(1,2,1);
cylinder
title('plot by cylinder');
[X,Y,Z] = cylinder(2+cos(t));
subplot(1,2,2);
surf(X,Y,Z,'LineWidth',2);
title('plot by surf');

```

خروجی این دستورات به شکل زیر می‌باشد. به دلیل وجود رنگ‌بندی در نمودارها حتما دستورات را در محیط MATLAB بنویسید و اجرا کنید.

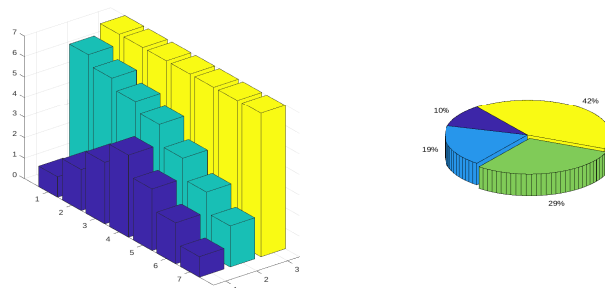


رسم نمودارهای میله‌ای و دایره‌ای

با دو دستور bar3 و pie3 می‌توان نمودارهای میله‌ای و دایره‌ای را در دستگاه سه‌بعدی رسم کرد. در این دو دستور یک بردار به عنوان ورودی داده می‌شود.

مثال ۱۱.۱۰. به چگونگی استفاده از این دو دستور در رسم نمودارهای زیر توجه کنید.


```
Z=[1 6.5 7; 2 6 7; 3 5.5 7; 4 5 7; 3 4 7; 2 3 7; 1 2 7];
subplot(1,2,1); bar3(Z);
subplot(1,2,2);
X=[5 9 14 20]; explode=[0 0 1 0]; pie3(X,explode)
```



۴.۱۰ رسم نمودارهای قطبی در فضا

نمودارهای توابع قطبی در فضا را نیز می‌توان با MATLAB رسم کرد.

رسم نمودارهای قطبی

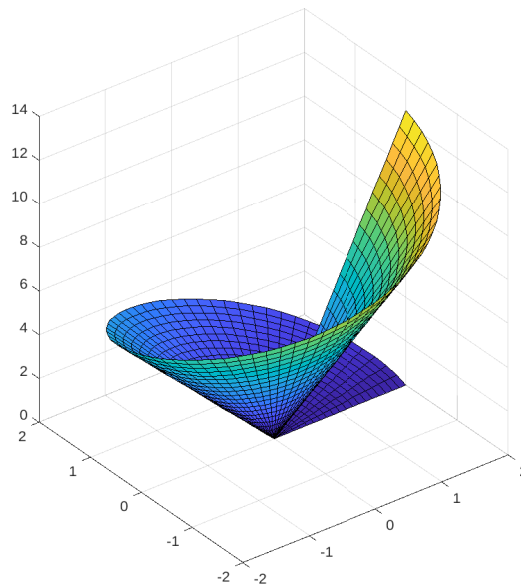
برای رسم نمودار تابع $z = f(r, \theta)$ در فضا باید چهار گام زیر را انجام داد:

۱. با استفاده از تابع `meshgrid` شبکه‌ای برای مقادیر r و θ ایجاد کنید. این کار کاملاً مشابه رسم رویه‌ها انجام می‌شود.
۲. مقدار z را در هر نقطه از شبکه محاسبه کنید.
۳. با دستور `pol2cart` شبکه ایجاد شده برای r و θ را از دستگاه مختصات قطبی به دستگاه مختصات دکارتی تبدیل کنید.
۴. با دستوراتی که برای رسم رویه‌ها داشتیم، مثل دستورات `mesh` یا `surf` می‌توان رویه را رسم کرد.

مثال ۱۲.۱۰. نمودار منحنی قطبی $z = r\theta$ را در فضا و در بازه $0 \leq \theta \leq 2\pi$ و $0 \leq r \leq 2$ رسم کنید.

```
th = [0:5:360]*pi/180;
r = 0:0.1:2;
[T,R] = meshgrid(th,r);
Z = R.*T;
[X,Y] = pol2cart(T,R);
surf(X,Y,Z)
```

خروجی دستورات بالا به شکل زیر می‌باشد.



۵.۱۰ تمرین

تمرین ۱۰.۱. موقعیت یک ذره متحرک به شکل تابعی از زمان و به صورت زیر داده شده است،

$$x(t) = (4 - 0.1t) \sin(0.1t), \quad y(t) = (4 - 0.1t) \cos(0.1t), \quad z(t) = 0.1t^{\frac{3}{2}},$$

موقعیت ذره را برای $0 \leq t \leq 30^\circ$ رسم کنید.

تمرین ۲.۱۰. یک راه‌پله بیضی شکل که اندازه آن از پایین به بالا کاهش می‌یابد با معادلات پارامتری زیر مدل‌بندی می‌شود،

$$x(t) = r \cos t, \quad y(t) = r \sin t, \quad z(t) = \frac{ht}{2\pi n},$$

که در آن

$$r = \frac{ab}{\sqrt{[b \cos(t)]^2 + [a \sin(t)]^2}} e^{-\frac{h}{a} t}.$$

نمودار سه‌بعدی این راه‌پله را برای $20^\circ = a$, $10^\circ = b$, $18 = h$ و $5 = n$ رسم کنید.

تمرین ۳.۱۰. رویه‌های زیر را در بازه‌های خواسته شده رسم کنید.

$$1. \quad z = \frac{x^2}{3} + 2 \sin(3y) \quad \text{در دامنه } -3 \leq x \leq 3 \text{ و } -3 \leq y \leq 3.$$

$$2. \quad z = 0.5|x| + 0.5|y| \quad \text{در دامنه } -2 \leq x \leq 2 \text{ و } -2 \leq y \leq 2.$$

$$3. \quad z = \frac{\sin R}{R} \quad \text{که در آن } R = \sqrt{x^2 + y^2} \quad \text{در دامنه } -10 \leq x \leq 10 \text{ و } -10 \leq y \leq 10.$$

$$4. \quad z = \cos(xy) \cos(\sqrt{x^2 + y^2}) \quad \text{در دامنه } -\pi \leq x \leq \pi \text{ و } -\pi \leq y \leq \pi.$$

نسخه
دایگان

۱۱ جعبه‌ابزار Symbolic

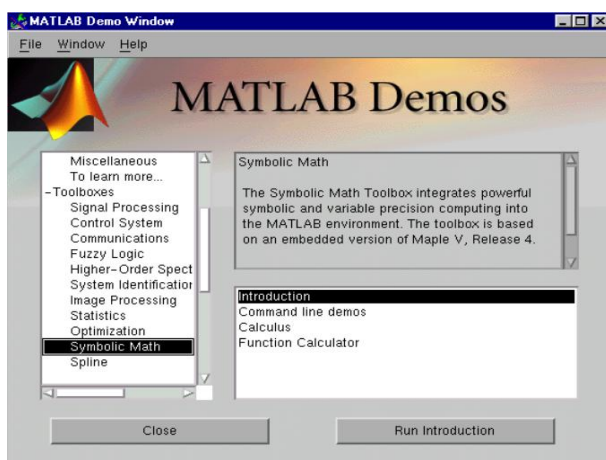
شاید این فصل کتاب مهم‌ترین فصل باشد، زیرا تقریباً تمام کارهایی که تاکنون انجام داده‌ایم را در این فصل به شکل ساده‌تری انجام خواهیم. شاید این پرسش مطرح شود که اگر می‌توان در MATLAB کارها را به شکل ساده‌تر انجام داد، چرا از ابتدا به این بسته و استفاده از آن نپرداختیم. در پاسخ به این پرسش می‌توان گفت، هرچند با بسته Symbolic بسیاری از کارها را می‌توان انجام داد ولی در برخی از مسایل، قدرت و انعطاف دستوراتی که در فصل‌های پیشین بیان شد بیشتر است و از کارایی بیشتری برخوردار هستند، همچنین بسیاری از مسایل را نمی‌توان با دستورات این بسته حل کرد. به همین دلیل با وجود امکان استفاده از بسته Symbolic حل برخی از مسایل، کماکان کاربران MATLAB تمایل به استفاده از دستورات عادی دارند. ولی دانستن دستورات و امکانات بسته Symbolic خالی از لطف نیست و در بسیاری از مسایل به ویژه در رسم نمودارها می‌تواند در حل مسایل کمک زیادی بکند. در سرتاسر این فصل دستورات و امکانات بسته Symbolic شرح داده شده‌اند و استفاده از این بسته در رسم نمودارها، حل معادلات جبری، حل معادلات دیفرانسیل، محاسبه انتگرال‌های معین و نامعین و ... پرداخته شده است.

۱.۱۱ معرفی جعبه‌ابزار Symbolic

در هنگام نصب نرم‌افزار MATLAB یکی از بسته‌هایی که به صورت خودکار نصب می‌شود بسته Symbolic می‌باشد. شکل ۱.۱۱ را ببینید. ولی اگر از Octave استفاده می‌کنید، باید بسته Symbolic را به شکل جداگانه و با دستورات زیر نصب و سپس فراخوانی کرد.

```
pkg install -forge symbolic
pkg load symbolic
```

توجه کنید که پس از هربار خروج از Octave و ورود دوباره به آن باید دستور دوم اجرا شود ولی دستور نخست فقط یک‌بار باید اجرا شود. با استفاده از این جعبه‌ابزار می‌توان متغیرها و عبارات



شکل ۱.۱۱: نصب جعبه‌ابزار Symbolic در هنگام نصب MATLAB

ریاضی را به‌شکل نمادین و به همان شکلی در متون ریاضی آورده می‌شود تعریف کرد و مورد استفاد قرار داد. برای مثال در این جعبه‌ابزار یک چندجمله‌ای‌های به شکل آرایه ذخیره نمی‌شود، بلکه دقیقاً به همان شکلی تعریف می‌شود که در ریاضی نوشته می‌شود. بدیهی است که پس از انجام عملیات روی توابع و عبارات به کمک جعبه‌ابزار Symbolic، حاصل به‌شکل نمادین خواهد و دیگر با بردارها سر و کار نخواهیم داشت. همچنین برخی عملیات مانند انتگرال‌گیری نامعین، یا مشتق‌گیری از توابع، یا حل معادلات دیفرانسیل و یافتن جواب تحلیلی که با دستوراتی که تا به حال بیان کردیم مقدور نبود، در این جعبه‌ابزار امکان‌پذیر می‌باشد.

در این بخش به چگونگی معرفی متغیرهای نمادین و تعریف عبارات ریاضی به‌صورت نمادین خواهیم پرداخت و روش استفاده از آنها را در محاسبات ساده بیان خواهیم کرد.

۱.۱.۱۱ معرفی متغیرهای نمادین

جعبه‌ابزار Symbolic نوع جدیدی از داده‌ها را در MATLAB معرفی می‌کند که به نوع نمادین موسوم هستند. داده‌هایی از نوع نمادین در واقع رشته‌هایی هستند می‌توانند نمادها یا عبارات ریاضی را در خود نگهداری کنند.

ایجاد متغیرهای نمادین

در Symbolic به دو شکل می‌توان متغیر نمادین ایجاد کرد.

۱. با دستور `sym` و به شکل

```
obj_name = sym('string')
```

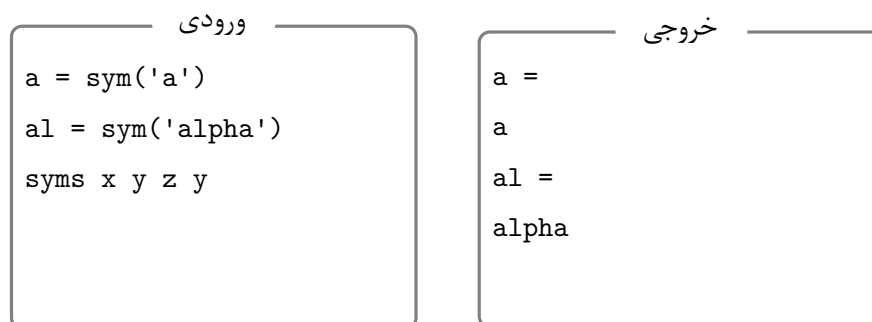
که در آن `obj_name` یک نام دلخواه و `string` یک رشته می‌باشد که در آن نباید از عملگرهای محاسباتی و برخی کاراکترهای خاص استفاده کرد.

۲. با دستور `syms` و به شکل زیر می‌توان تعدادی متغیر نمادین تعریف کرد.

```
syms var1 var2 var3 ... varn
```

که در آن `syms` کلمه کلیدی و `var1` تا `varn` اسامی دلخواه برای متغیرهای نمادین می‌باشند. توجه کنید بین اسامی متغیرها باید یک فاصله باشد.

مثال ۱.۱۱. به چگونگی استفاده از دستورات `sym` و `syms` برای تولید متغیرهای نمادین توجه کنید.



👉 توجه کنید که دستور `syms` هیچ خروجی تولید نمی‌کند، فقط متغیرهای معرفی شده به عنوان متغیر نمادین تعریف کرده و آماده استفاده می‌کند.

۲.۱.۱۱ تعریف عبارات به شکل نمادین

پس از تعریف متغیرها به شکل نمادین می‌توان از آنها برای تعریف عبارات ریاضی استفاده کرد. به این منظور باید شکل زیر عمل کرد.

```
Exp_Name = Math_Exp
```

که در آن سمت چپ تساوی نامی دلخواه می‌باشد و سمت راست یک عبارت ریاضی است. عبارت سمت راست نباید داخل کوتیشن باشد.

👉 هر متغیری که در سمت راست تساوی مورد استفاده قرار می‌گیرد باید پیش از استفاده با استفاده از دستور `syms` به عنوان یک متغیر نمادین تعریف شده باشد.

مثال ۲.۱۱. به چگونگی تعریف برخی توابع و عبارات در دستورات زیر توجه کنید.

ورودی

```
syms x y z t;
E = x^2 - 3*exp(x) + 1
g(x,y) = x^2/(x^2 + y^2)
g(1,2)
h(x,y,z,t) = ...
    x*y*z*z/(x+y+z+t)
h(1,1,1,1)
s(x) = x/sin(x) + exp(x)
s(pi/2)
```

خروجی

```
E =
x^2 - 3*exp(x) + 1
g(x, y) =
x^2/(x^2 + y^2)
ans =
1/5
h(x, y, z, t) =
(x*y*z^2)/(t + x + y + z)
ans =
1/4
s(x) =
exp(x) + x/sin(x)
ans =
pi/2 + exp(pi/2)
```

👉 به چگونگی تعریف توابع و مقداردهی به توابع در دستورات بالا توجه کنید.

👉 توجه کنید که در سمت راست برای اعداد نیازی به تعریف متغیر نمادین نیست ولی اگر بخواهیم این امکان وجود دارد که برای اعداد هم متغیر نمادین تعریف کنیم و بکار ببریم. برای مثال دستورات زیر را در نظر بگیرید.

ورودی

```
syms x y z t;
one = sym('1');
two = sym('2');
three = sym('3');
g(x) = two*x^2 + ...
      three*exp(x) - one
g(5)
```

خروجی

```
g(x) =
3*exp(x) + 2*x^2 - 1
ans =
3*exp(5) + 49
```

دستور symvar

اگر عبارت نمادینی موجود باشد، با استفاده از دستور `symvar(S)` می‌تواند به اسامی تمام متغیرهای نمادین مورد استفاده در عبارت `S` دست یافت.

ورودی

```
syms x y z t;
E = t^2 - 3*exp(t) + 1;
g(x,y) = x^2/(x^2 + y^2);
symvar(E)
symvar(g)
```

خروجی

```
ans =
t

ans =
[ x, y]
```

👉 در نسخه‌های قدیمی MATLAB بجای این دستور از `findsym` استفاده می‌شد ولی در نسخه‌های اخیر، دستور `symvar` جایگزین `findsym` شده است.

۳.۱.۱۱ ساده‌سازی عبارات ریاضی

شاید عبارات ریاضی که به شکل نمادین در MATLAB تولید می‌شوند مورد پسند کاربر نباشد، لذا در جعبه‌ابزار Symbolic این امکان فراهم شده است با برخی دستورات بتوان عبارت ریاضی را برحسب توان‌های یک متغیر خاص مرتب کرد، عبارت تولید شده را بسط داد یا آن را تجزیه کرد. در این بخش به این سه کار می‌پردازیم.

دستور collect

دستور `collect` برای مرتب‌سازی یک عبارت ریاضی نسبت به یک متغیر و برحسب توان‌های نزولی آن متغیر بکار می‌رود. این دستور به یکی از دو شکل زیر مورد استفاده قرار می‌گیرد.

```
collect(S)
```

```
collect(S,var_name)
```

که در آن S یک عبارت ریاضی و `var_name` متغیری است که می‌خواهیم مرتب‌سازی نسبت توان‌های نزولی آن انجام شود. دستور اول زمانی بکار می‌رود که عبارت دارای یک متغیر باشد و دستور دوم برای عبارات چندمتغیره مورد استفاده قرار می‌گیرد.

مثال ۳.۱۱. در دستورات زیر به خروجی حاصل از هر دستور دقت کنید.

```
syms x y;
F = (x^3 - x*exp(x) + x^2 - 1)*(x*sin(x) + x^2 - 1)
F_col = collect(F)
E = (x^2 - y^2 + y)*(x^2 - y^3 - 1)
E_colx = collect(E,x)
E_coly = collect(E,y)
```

خروجی این دستورات به شکل زیر می‌باشد. در هر دستور شکل اصلی عبارت نیز آورده شده است.

```
F =
-(x*sin(x) + x^2 - 1)*(x*exp(x) - x^2 - x^3 + 1)
F_col =
x^5+(sin(x)+1)*x^4+(sin(x)-exp(x)-1)*x^3+(-exp(x)*sin(x)-2)*x^2
+(exp(x)-sin(x))*x+1
E =
-(- x^2 + y^3 + 1)*(x^2 - y^2 + y)
E_colx =
```

```

x^4 + (- y^3 - y^2 + y - 1)*x^2 - (y - y^2)*(y^3 + 1)
E_coly =
y^5-y^4+(-x^2)*y^3+(1-x^2)*y^2+(x^2 - 1)*y+x^2*(x^2-1)

```

دستور expand

با دستور `expand` می‌توان یک عبارت ریاضی را باز کرد، به این معنی که اگر عبارت اولیه به شکل یک اتحاد جبری یا اتحاد مثلثاتی یا حاصل ضرب چند عبارت باشد، با اجرای دستور `expand` روی عبارت، شکل باز شده اتحاد یا ضرب شده عبارات در هم محاسبه می‌شود. تنها ورودی این دستور یک عبارت ریاضی می‌باشد.

مثال ۴.۱۱. به خروجی دستورات زیر توجه کنید.

```

syms x y;
F = (x^3 - 1)^2*(x*sin(x) + 1)
F_expand = expand(F)
E = (x+1)^3
E_expand = collect(E)
expand(tan(2*x))

```

خروجی به صورت زیر می‌باشد.

```

F =
(x*sin(x) + 1)*(x^3 - 1)^2
F_expand =
x^7*sin(x) - 2*x^4*sin(x) + x*sin(x) - 2*x^3 + x^6 + 1
E = (x + 1)^3
E_expand = x^3 + 3*x^2 + 3*x + 1
ans = -(2*tan(x))/(tan(x)^2 - 1)

```

دستور factor

اگر یک چندجمله‌ای به عنوان ورودی به دستور factor داده شود، و اگر چندجمله‌ای قابل تجزیه به حاصل ضرب عوامل اول باشد، خروجی یک بردار است که شامل عوامل اول می‌باشد و اگر قابل تجزیه نباشد، خود عبارت برگشت داده می‌شود.

مثال ۵.۱۱. به چگونگی استفاده از دستور factor در زیر توجه کنید.

ورودی

```
syms x y;
F = x^3 + 4*x^2 - 11*x - 30
F_factor = factor(F)
G = x^2 + 1
G_factor = factor(G)
```

خروجی

```
F =
x^3 + 4*x^2 - 11*x - 30
F_factor =
[ x + 5, x - 3, x + 2]
G = x^2 + 1
G_factor = x^2 + 1
```

دستور simplify

دستور simplify با استفاده از عملگرهای ریاضی مانند جمع، ضرب، تقسیم، توان، لگاریتم و ... شکل ساده‌تری از یک عبارت ریاضی را محاسبه می‌کند. ورودی این دستور یک عبارت ریاضی است.

مثال ۶.۱۱. به دستورات زیر و چگونگی استفاده از دستور simplify دقت کنید.

ورودی

```
syms x y;
F = (x^2 + 5*x + 6)/(x+2)
F_simp = simplify(F)
G = (x+y)/(1/x + 1/y)
G_simp = simplify(G)
```

خروجی

```
F = (x^2 + 5*x + 6)/(x + 2)
F_simp = x + 3
G = (x + y)/(1/x + 1/y)
G_simp = x*y
```

دستور pretty

با دستور pretty عبارت ریاضی که به عنوان ورودی به آن داده می شود، به شکلی تقریباً مشابه نوشتارهای ریاضی نمایش داده می شود.

مثال ۷.۱۱. به دستورات زیر و خروجی حاصل از آن توجه کنید.

ورودی

```
syms x y;
F = (x^2+5*x+3)/(sin(x)+2)
pretty(F)
G = (x^2 + 2*x -1)^(1/2)
pretty(G)
```

خروجی

```
F =
(x^2 + 5*x+3)/(sin(x)+2)
2
x  + 5 x + 3
-----
sin(x) + 2
G =
(x^2 + 2*x - 1)^(1/2)
2
sqrt(x  + 2 x - 1)
```

دستور subs

اگر بخواهیم بخشی از یک عبارت جبری را با چیزی دیگری جایگزین کنیم می توانیم از دستور subs استفاده کنیم. شکل کلی این دستور به صورت `subs(E,old,new)` می باشد که در آن

E عبارتی است که می خواهیم در آن جایگزینی انجام شود. این عبارت باید به شکل نمادین تعریف شود. این عبارت و دو عبارت بعدی نیازی نیست بین جفت کوتیشن نوشته شوند.

old بخشی از عبارت E است که می خواهیم بجای آن مقدار جدیدی جایگزین شود.

new مقداری است که می خواهیم جایگزین `old` در عبارت E شود.

مثال ۸.۱۱. در عبارات جبری زیر تغییرات خواسته شده را اعمال کنید.

• در $(a+b)^2 + 2a^2 - 3b(a+b) + \frac{a-b}{(a+b)^3}$ عبارت $a+b$ را به X تغییر دهید.

• در $ab^2 + 5(a+b)^2(ab)^3 - 3a^2b$ عبارت ab را به Y تغییر دهید

با دستورات زیر می‌توان تغییرات خواسته شده را در عبارات اعمال کرد.

```
syms a b X Y;
E = (a+b)^2 + 2*a^2 - 3*b*(a+b) + (a-b)/(a+b)^3;
Enew = subs(E1,a+b,X)
F = a*b^2 + 5*(a+b)^2*(a*b)^3 - 3*a^2*b;
Fnew = subs(E2,a*b,Y)
```

نتیجه اجرای این دستورات به‌صورت زیر می‌باشد.

```
Enew =
(a - b)/X^3 - 3*X*b + X^2 + 2*a^2
Fnew =
Y*b - 3*a^2*b + 5*Y^3*(a + b)^2
```

۲.۱۱ کاربرد Symbolic در حساب دیفرانسیل و انتگرال

با استفاده از جعبه‌ابزار Symbolic می‌توان تقریباً تمام کارهایی که در حساب دیفرانسیل و انتگرال مورد نیاز است را انجام داد. در این بخش به معرفی چگونگی حل مسایل حساب دیفرانسیل و انتگرال با جعبه‌ابزار Symbolic می‌پردازیم.

۱.۲.۱۱ حل معادلات جبری

در جعبه‌ابزار Symbolic می‌توان معادلات جبری را حل کرد. معادله می‌تواند به شکل چندجمله‌ای، مثلثاتی، گویا، شامل رادیکال یا به هر شکل دیگری باشد. همچنین امکان حل دستگاه معادلات

خطی و غیرخطی نیز در Symbolic وجود دارد. برخلاف دستور fzero که برای حل معادلات در فصل‌های پیش بیان شد، دستورات حل معادلات در بسته Symbolic می‌توانند با یک دستور تمام جواب‌های یک معادله را بدست بیاورند، در حالیکه با دستورات قبل تنها یک ریشه در هر بار استفاده محاسبه می‌شد و برای یافتن تمام ریشه‌ها باید چندین بار دستور fzero را در برای بازه‌های مختلف اجرا کرد.

دستور solve

با دستور solve می‌توان یک معادله جبری را حل کرد. این دستور به شکل کلی زیر می‌باشد،
`solve(eq,var)`

که در آن

eq معادله‌ای است که می‌خواهیم حل شود. این معادله باید پیش از فراخوانی دستور solve به صورت نمادین معرفی شود.

var متغیری است که می‌خواهیم معادله نسبت به آن حل شود. استفاده از این گزینه اختیاری است و زمانی استفاده می‌شود که معادله با بیش از یک متغیر تعریف شده باشد و بخواهیم معادله به شکل پارامتری حل شود. اگر در معادلات با بیش از یک متغیر از این گزینه استفاده نشود، متغیر x در نظر گرفته می‌شود.

مثال ۹.۱۱. به مثال‌هایی که در زمینه استفاده از دستور solve آورده شده است توجه کنید.

```
syms x y a b c;
eq1 = x^3 + 2*x*sin(x) - x + 1; SOL1 = solve(eq1)
eq2 = a*x^2 + b*x + c; SOL2 = solve(eq)
eq2 = x^2*y - y^2*x + x*y + x - y + 1; SOL3 = solve(eq2,y)
```

خروجی دستورات بالا به شکل زیر است.

```
SOL1 =
-1.8589026587005620729836160070795
```

```
SOL2 =
-(b + (b^2 - 4*a*c)^(1/2))/(2*a)
-(b - (b^2 - 4*a*c)^(1/2))/(2*a)
SOL3 =
-1/x
x + 1
```

👉 علاوه بر دستور solve دستور دیگری وجود دارد که مشابه دستور solve عمل می‌کند. این دستور به شکل vpasolve مورد استفاده قرار می‌گیرد. استفاده از این دستور باعث می‌شود تا جواب‌ها به شکل عددی نمایش داده شود، درحالی‌که دستور solve جواب‌ها را به شکل نمادین نمایش می‌دهد.

مثال ۱۰.۱۱. به تفاوت میان خروجی solve و vpasolve توجه کنید.

ورودی	خروجی
<pre>syms x y z a b c; eq = 2*x^2 + 3*x -5; Sol1 = solve(eq) Sol2 = vpasolve(eq)</pre>	<pre>Sol1 = -5/2 1 Sol2 = -2.5 1.0</pre>

👉 برای حل دستگاه معادلات جبری باید ابتدا معادلات را به شکل نمادین تعریف کرد، سپس دستور solve یا vpasolve را روی معادلات اجرا کرد.

مثال ۱۱.۱۱. دستگاه معادلات زیر را با استفاده از solve و vpasolve حل کنید.

$$\begin{cases} 2x + 3y - z = -5 \\ x + 2y - 2z = -3 \\ -x + y + 3z + 2 = 0 \end{cases}$$

با دستورات زیر می‌توان دستگاه را حل کرد. بدیهی است که از این روش می‌توان برای هر تعداد معادله استفاده کرد. پیشنهاد می‌شود به‌عنوان تمرین چند دستگاه معادله مختلف را تعریف و با این روش حل کنید.

ورودی

```
syms x y z;
eq1 = 2*x + 3*y - z + 5;
eq2 = x + 2*y - 2*z + 3;
eq3 = -x + y + 3*z + 2;
[X1,Y1,Z1] = solve(eq1,eq2,eq3)
[X2,Y2,Z2] = vpasolve(eq1,eq2,eq3)
```

خروجی

```
X1 = -1/5
Y1 = -8/5
Z1 = -1/5
X2 = -0.2
Y2 = -1.6
Z2 = -0.2
```

توجه کنید که اگر معادلات به‌شکل پارامتری تعریف شوند، جواب‌ها نیز به‌شکل پارامتری محاسبه خواهند شد. برای مثال دستورات و خروجی حاصل از آنها را در کد زیر ببینید.

ورودی

```
syms x y a1 a2 b1 b2 c1 c2;
eq1 = a1*x + b1*y - c1;
eq2 = a2*x + b2*y - c2;
[XP,YP] = solve(eq1,eq2)
```

خروجی

```
XP =
-(b1*c2-b2*c1)/(a1*b2-a2*b1)
YP =
(a1*c2-a2*c1)/(a1*b2-a2*b1)
```

اما اگر با یک دستگاه معادلات غیرخطی مواجه باشیم، آیا کماکان این دو دستور قادر به محاسبه جواب دستگاه می‌باشند؟ پاسخ مثبت است. با استفاده از دو دستور `solve` و `vpasolve` می‌توان جواب دستگاه معادلات غیرخطی را نیز بدست آورد. به این منظور باید ابتدا معادلات را به شکل نمادین تعریف کرد، سپس با ارسال معادلات به دو یکی از دو دستور بیان شده جواب دستگاه را محاسبه کرد.

مثال ۱۲.۱۱. دستگاه معادلات زیر را حل کنید.

$$\begin{cases} x^2y + 2xy^2 + x - y = 9 \\ x^2 + y^2 + 2x - 3y = 1 \end{cases}$$

ورودی

```
syms x y ;
eq1 = x^2*y + 2*x*y^2 + x - y - 9;
eq2 = x^2 + y^2 + 2*x - 3*y - 1;
[X,Y] = solve(eq1,eq2)
```

خروجی

```
X = 1
Y = 2
```

👉 اگر دستورات بالا در MATLAB اجرا کنید، علاوه بر جواب نمایش داده شده جواب‌های مختلط نیز نمایش داده خواهند شد که به دلیل طولانی بودن از نمایش آنها در کتاب صرف‌نظر شده است.

👉 در دستورات بالا از `vpasolve` بجای `solve` استفاده شده باشد، جواب‌ها به شکل عددی محاسبه می‌شوند.

مثال ۱۳.۱۱. محل برخورد دایره $(x-2)^2 + (y-4)^2 = 9$ را با خط $y = \frac{x}{4} + 1$ بدست آورید.

ورودی

```
syms x y R;
eq1 = (x-2)^2 + (y-4)^2 - 9;
eq2 = y - x/2 - 1;
[X1,Y1] = solve(eq1,eq2)
[X2,Y2] = vpasolve(eq1,eq2)
```

خروجی

```
X1 =
14/5 - (2*29^(1/2))/5
(2*29^(1/2))/5 + 14/5
Y1 =
12/5 - 29^(1/2)/5
29^(1/2)/5 + 12/5
X2 =
4.95406592285380161250
0.64593407714619838749
Y2 =
3.47703296142690080625
1.32296703857309919374
```

👉 به تفاوت میان خروجی `solve` و `vpasolve` توجه کنید.

۲.۲.۱۱ حدگیری

محاسبه حد $\lim_{x \rightarrow a} f(x)$ در جعبه ابزار Symbolic به سادگی امکان پذیر می باشد.

محاسبه حد توابع

محاسبه حد توابع با دستور `limit(f,var,a)` انجام می شود که در آن

f تابعی است که می خواهیم از آن حد بگیریم. این تابع باید به صورت نمادین تعریف شده باشد.

var متغیری است که می خواهیم حد نسبت به آن متغیر گرفته شود. برای توابع یک متغیره می توان این گزینه را حذف کرد.

a مقداری است که می خواهیم حد تابع در آن محاسبه شود.

مثال ۱۴.۱۱. حد توابع زیر را محاسبه کنید.

$$\lim_{x \rightarrow 0} \frac{1 - \cos x}{x^2}, \quad \lim_{x \rightarrow \infty} \frac{3x^2 + 1}{2x^2 + 2x - 1}, \quad \lim_{y \rightarrow a} \frac{x^2 - y^2}{x - a}$$

دستورات زیر حدهای خواسته شده را محاسبه می کنند.

ورودی

```
syms x y a;
eq1 = (1-cos(x))/(x^2);
L1 = limit(eq1,0)
eq2 = (3*x^2 +1)/(2*x^2 + 2*x -1);
L2 = limit(eq2,x,inf)
eq3 = (x^2 - y^2)/(x-a);
L3 = simplify(limit(eq3,y,a))
```

خروجی

```
L1 = 1/2
L2 = 3/2
L3 = a + x
```

📌 در محاسبه حد آخر اگر از دستور `simplify` استفاده نمی شد، جواب به شکل ساده نشده ظاهر می شد.

۳.۲.۱۱ مشتق‌گیری

در جعبه‌ابزار Symbolic امکان مشتق‌گیری نمادین از توابع وجود دارد، یعنی می‌توان تابعی را به‌صورت نمادین تعریف کرد و سپس از آن مشتق‌گیری کرد و شکل نمادین مشتق را مشاهده کرد.

مشتق‌گیری نمادین

با دستور `diff` به‌شکل کلی

`diff(function,var,n)`

می‌توان مشتق‌گیری نمادین انجام داد. در این دستور

function ضابطه تابع است که باید به‌شکل نمادین تعریف شده باشد.

var متغیری است که می‌خواهیم مشتق‌گیری نسبت به آن انجام شود. اگر تابع یک متغیره باشد نیازی به استفاده از این آرگومان نیست.

n یک عدد طبیعی است که نشان‌دهنده مرتبه مشتقی است که می‌خواهیم محاسبه شود. اگر از این گزینه استفاده نشود، مشتق مرتبه اول محاسبه می‌شود.

مثال ۱۵.۱۱. مشتق مرتبه اول تابع $y = e^{x^2+y^2}$ نسبت به x و مشتق سوم نسبت به y را محاسبه کنید.

ورودی

```
syms x y;
eq1=exp(x^2+y^2);
D1 = diff(eq1)
D3 = diff(eq1,y,3)
```

خروجی

```
D1 =
2*x*exp(x^2 + y^2)
D4 =
12*y*exp(x^2+y^2)+8*y^3*exp(x^2+y^2)
```

👉 در صورت تمایل می‌توانید با دستور `pretty` نمایش بهتری برای مشتقات محاسبه شده بدست آورید.

👉 در دستور اول چون مشتق اول نسبت به x خواسته شده است، متغیر و مرتبه مشتق وارد نشده است.

۴.۲.۱۱ انتگرال گیری

در MATLAB و با استفاده از جعبه‌ابزار Symbolic می‌توان انتگرال‌های معین و نامعین را محاسبه کرد. توجه کنید که با روشی که بیان خواهد شد امکان محاسبه هر انتگرال نامعینی وجود ندارد، پس انتظار نداشته باشید که این روش هر انتگرال نامعینی را محاسبه کند.

انتگرال گیری نمادین

برای محاسبه تابع اولیه در جعبه‌ابزار Symbolic می‌توان از دستور `int` به شکل کلی `int(function,var)` استفاده کرد که در آن `function` تابعی است که باید به صورت نمادین تعریف شده باشد و `var` متغیری است که می‌خواهیم انتگرال گیری نسبت به آن انجام شود. اگر تابع یک متغیره باشد، نیاز به نوشتن متغیر نیست.

مثال ۱۶.۱۱. به چگونگی استفاده از دستور `int` برای توابع یک متغیره و دو متغیره در دستورات زیر توجه کنید.

ورودی

```
syms x y;
eq1=x^2*sin(x)+x*exp(x^2+1);
I1 = int(eq1)
eq2 = 2*x*exp(x^2 + y^2);
I2 = int(eq2,x)
```

خروجی

```
I1 =
exp(x^2+1)/2+2*cos(x)
-x^2*cos(x)+2*x*sin(x)
I2 =
exp(x^2 + y^2)
```

👉 امکان استفاده از دستور `pretty` برای نمایش بهتر حاصل انتگرال‌ها وجود دارد.

نکته عملی

امکان محاسبه انتگرال معین نیز با دستور `int` در جعبه‌ابزار Symbolic وجود دارد. به این منظور کفایت دستور `int(function,var,a,b)` را به شکل `int(function,var,a,b)` فراخوانی کرد، که در آن `a` و `b` به ترتیب کران‌های پایین و بالای انتگرال می‌باشند.

مثال ۱۷.۱۱. به دستورات زیر و چگونگی محاسبه انتگرال معین توجه کنید.

ورودی	خروجی
<pre>syms x y; eq1=x^2*sin(x)+x*exp(x^2+1); I1 = int(eq1,0,2) vpa(I1) eq2 = 2*x*exp(x^2 + y^2); I2 = int(eq2,x,1,y)</pre>	<pre>I1 = 4*sin(2)+(exp(1)*(exp(4)-1)) /2 - 4*cos(1)^2 ans = 75.316922017455790648456862 I2 = exp(y^2)*(exp(y^2) - exp(1))</pre>

👉 در دستورات بالا از دستور vpa برای تبدیل یک کمیت نمادین به کمیت عددی استفاده شده است.

۵.۲.۱۱ محاسبه سری

فرض کنید می‌خواهیم سری $S = \sum_{n=1}^N f(n)$ را محاسبه کنیم. با روش‌هایی که در فصل‌های پیش دیدید می‌توان مقدار عددی این سری را حساب کرد، ولی محاسبه سری‌ها در حساب دیفرانسیل و انتگرال با جعبه‌ابزار Symbolic نیز امکان‌پذیر می‌باشد.

محاسبه سری

با دستور `symsum` می‌توان مجموع سری‌ها را محاسبه کرد. شکل کلی این دستور به صورت `symsum(f,var,lb,ub)` که در آن،

f عبارتی است که باید به شکل نمادین تعریف شده باشد.

var متغیری است که می‌خواهیم سری بر اساس آن محاسبه شود. استفاده از این گزینه اختیاری است. اگر سری تنها شامل یک متغیر باشد نیازی به نوشتن آن نیست.

lb,ub کرانه‌های پایین و بالای سیگما می‌باشند. اگر کران پایین و بالا نوشته نشوند، مجموع سری به صورت نمادین محاسبه می‌شود.

مثال ۱۸.۱۱. سری‌های زیر را محاسبه کنید،

$$\sum_{n=1}^{\infty} n, \quad \sum_{n=1}^N n, \quad \sum_{n=1}^{\infty} \frac{1}{n^2}, \quad \sum_{n=1}^{\infty} \frac{1}{n^3}.$$

در دستورات زیر به روش استفاده از حالت‌های مختلف استفاده از دستور `symsum` توجه کنید.

ورودی

```
syms x n N;
f = n;
S1 = symsum(f,1,N)
S2 = symsum(f,1,100)
f = 1/n^2;
S3 = symsum(f,1,10)
vpa(S3)
S4 = symsum(f,1,Inf)
vpa(S4)
```

خروجی

```
S1 =
(N*(N + 1))/2
S2 =
5050
S3 =
1968329/1270080
ans =
1.5497677311665406903
S4 =
pi^2/6
ans =
1.6449340668482264364
```

👉 در دستور سطر سوم از متغیر نمادین N بجای کران بالا استفاده شده است که منجر به محاسبه مجموع سری به شکل نمادین می‌شود.

👉 در سه دستور آخر از `vpa` برای تبدیل مقدار نمادین سری به مقدار عددی استفاده شده است.

مثال ۱۹.۱۱. نشان دهید که سری $\sum_{n=1}^{\infty} \frac{1}{n}$ واگرا و سری $\sum_{n=1}^{\infty} (-1)^{(n+1)} \frac{1}{n}$ همگراست.

با دستورات زیر می‌توان درستی ادعای بالا را بررسی کرد.

ورودی

```
syms n
f1 = (-1)^(n+1)/n;
S1 = symsum(f1,1,Inf)
f2 = 1/n;
S2 = symsum(f2,1,Inf)
```

خروجی

```
S1 =
log(2)

S2 =
Inf
```

محاسبه سری تیلور

برای محاسبه سری تیلور تابع $f(x)$ می‌توان از دستور `taylor` به شکل زیر استفاده کرد،

```
taylor(function,var,a,'order',n)
```

که در آن

function تابعی است که می‌خواهیم سری تیلور آن را حساب کنیم.

var متغیری است که می‌خواهیم سری تیلور نسبت به آن متغیر محاسبه شود. برای توابع یک متغیره نیازی به نوشتن `var` نیست.

a مقداری است که می‌خواهیم سری تیلور حول آن نقطه محاسبه شود. استفاده از این گزینه اختیاری است و اگر نوشته نشود، سری مک‌لورن، یعنی سری تیلور حول نقطه صفر، محاسبه می‌شود.

order این واژه باید به همین شکل و بین دو کوتیشن نوشته شود و به همراه `n` استفاده می‌شود که نشان‌دهنده تعداد جملاتی از سری تیلور است که می‌خواهیم حساب شود. اگر این گزینه نوشته نشود، پنج جمله اول سری محاسبه و نمایش داده می‌شود.

مثال ۲۰.۱۱. به چگونگی تولید سری‌های تیلور و مک‌لورن در دستورات زیر توجه کنید.

```
syms n x; f = exp(x);
T1 = taylor(f)
T2 = taylor(f,x,1,'order',3)
```


خروجی دستورات بالا به شکل زیر می باشد.

```
T1 =
x^5/120 + x^4/24 + x^3/6 + x^2/2 + x + 1
T2 =
exp(1) + exp(1)*(x - 1) + (exp(1)*(x - 1)^2)/2
```

👉 خروجی حاصل از دستور اول سری مک لورن تابع می باشد.

مثال ۲۱.۱۱. چهار جمله اول سری تیلور و مک لورن تابع $g(x) = \sin(x+y)$ را نسبت به y و حول $a = 1$ بدست آورید.
با دستورات زیر خروجی مطلوب حاصل می شود.

```
syms x y; f = sin(x+y);
T = taylor(f,y,1,'order',4)
```

خروجی کد بالا به شکل زیر می باشد.

```
T =
sin(x+1)+cos(x+1)*(y-1)-(cos(x+1)*(y-1)^3)/6-(sin(x+1)*(y-1)^2)/2
```

۳.۱۱ کاربرد Symbolic در حل معادلات دیفرانسیل

در بخش ۲.۸ روش حل عددی معادلات دیفرانسیل را بیان کردیم. در این بخش به معرفی چگونگی حل تحلیلی معادلات دیفرانسیل با استفاده از جعبه ابزار Symbolic می پردازیم. با استفاده از جعبه ابزار Symbolic می توان جواب تحلیلی یک معادله دیفرانسیل را پیدا کرد. شایان ذکر است که دستوراتی که بیان می شوند همواره و برای هر معادله دیفرانسیلی قادر به یافتن جواب نیستند. با استفاده از جعبه ابزار Symbolic می توان معادلات دیفرانسیل معمولی یا دستگاه معادلات دیفرانسیل را حل کرد. در این بخش ابتدا به بیان چگونگی حل معادلات دیفرانسیل معمولی مرتبه اول و دوم می پردازیم، سپس به آموزش روش حل دستگاه های معادلات دیفرانسیل خواهیم پرداخت.

۱.۳.۱۱ حل معادلات دیفرانسیل مرتبه اول

معادله دیفرانسیل خطی مرتبه اول به شکل $\frac{dy}{dt} = f(t, y)$ را در نظر بگیرید. هدف یافتن تابع $y = y(t)$ است به گونه‌ای که در معادله دیفرانسیل مفروض صدق کند.

حل معادلات دیفرانسیل مرتبه اول

برای حل معادله دیفرانسیل $\frac{dy}{dt} = f(t, y)$ گام‌های زیر را به ترتیب اجرا کنید.

۱. معادله دیفرانسیل مفروض را به شکل `ode = 'Dy = f(t,y)'` تعریف کنید. توجه کنید که کاراکتر `D` رزرو شده است و باید به همین شکل مورد استفاده قرار بگیرد. همچنین کل معادله باید در بین یک جفت کوتیشن قرار بگیرد. بجای `ode` می‌توان هر نام دلخواهی قرار داد.

۲. با استفاده از دستور `dsolve` به یکی از دو شکل زیر جواب تحلیلی معادله دیفرانسیل محاسبه می‌شود،

```
dsolve(ode)
```

```
dsolve(ode,var)
```

دستور اول جواب معادله دیفرانسیل را نسبت به متغیر `t` و دستور دوم جواب معادله دیفرانسیل را نسبت به متغیر مشخص شده محاسبه می‌کند.

مثال ۲۲.۱۱. جواب معادلات دیفرانسیل

$$\frac{dy}{dt} = ty, \quad \frac{ds}{dx} = ax^2.$$

معادله دوم را یکبار نسبت به a و یکبار نسبت به x حل کنید.

به روش استفاده از دستور `dsolve` برای یافتن سه جواب خواسته شده توجه کنید.

```
syms t y a x;
```

```
ode1 = 'Dy = t*y'; SOL1 = dsolve(ode1)
```

```
ode2 = 'Ds = a*x^2'; SOL2 = dsolve(ode2,x)
```

```
SOL3 = dsolve(ode2,a)
```

خروجی حاصل از این دستورات به شکل زیر می باشد.

```
SOL1 = C1*exp(t^2/2)
SOL2 = (a*x^3)/3 + C2
SOL3 = (a^2*x^2)/2 + C3
```

مثال ۲۳.۱۱. جواب عمومی معادله دیفرانسیل $x^2 y' = (x-1)e^{x+y}$ را بدست آورید.
ابتدا معادله دیفرانسیل را به شکل استاندارد $\frac{dy}{dx} = \frac{(x-1)e^{x+y}}{x^2}$ می نویسیم. حال دستورات زیر را در محیط MATLAB وارد کنید.

ورودی

```
syms y x;
ode='Dy=(x-1)*exp(x+y)/x^2';
Sol = dsolve(ode,x)
```

خروجی

```
Sol =
log(-x/(exp(x)-C4*x))
```

با دستور dsolve نمی توان هر معادله دیفرانسیلی را حل کرد. در این صورت با پیغامی مبنی بر امکان پذیر نبودن حل معادله مواجه خواهیم شد. برای مثال کد زیر و خروجی حاصل از آن را ببینید. این دستورات برای حل معادله دیفرانسیل $y' = \sqrt{x+y}$ نوشته شده است.

ورودی

```
syms y x;
ode = 'Dy = sqrt(x + y)';
Sol = dsolve(ode,x)
```

خروجی

```
Warning: Unable to find
explicit solution.
> In dsolve (line 201)
In Test (line 3)
Sol =
[ empty sym ]
```

۲.۳.۱۱ معادلات دیفرانسیل مرتبه اول با مقدار اولیه

معادله دیفرانسیل مرتبه اول با مقدار اولیه زیر را در نظر بگیرید،

$$\begin{cases} y' = f(t, y), & a \leq t \leq T, \\ y(a) = y_0. \end{cases}$$

هدف یافتن جوابی است که در معادله و شرط اولیه صدق کند. در این حالت جواب فاقد مقدار ثابت می‌باشد.

جواب مسایل مقدار اولیه

برای یافتن جواب این دسته از معادلات پس از تعریف معادله به شکلی که بیان کردیم، شرط اولیه را به صورت `cond = 'y(a) = y0'` تعریف کرده، سپس از دستور `dsolve` به شکل

```
dsolve(ode, cond)
```

استفاده می‌کنیم. در این صورت معادله نسبت به متغیر پیش فرض t حل خواهد شد و مقادیر ثابت به گونه‌ای تعیین می‌شوند که مقدار اولیه نیز برقرار باشد.

مثال ۲۴.۱۱. معادله دیفرانسیل با شرط اولیه زیر را حل کنید،

$$ty' + 2y = \sin t, \quad y\left(\frac{\pi}{2}\right) = 0.$$

شکل استاندارد معادله به صورت $\frac{dy}{dt} = \frac{\sin t}{t} - 2\frac{y}{t}$ است.

ورودی

```
syms y t;
ode='Dy=sin(t)/t-2*y/t';
cond = 'y(pi/2) = 0';
Sol = dsolve(ode, cond)
```

خروجی

```
Sol =
(sin(t)-t*cos(t))/t^2-1/t^2
```

۳.۳.۱۱ حل معادلات دیفرانسیل مرتبه دوم

برای یافتن جواب تحلیلی معادله دیفرانسیل مرتبه دوم $y'' = f(t, y, y')$ نیز می‌توان از دستور `dsolve` استفاده کرد. تنها تفاوتی که بین حل معادلات دیفرانسیل مرتبه دو و معادلات دیفرانسیل مرتبه اول وجود دارد، در چگونگی تعریف معادله دیفرانسیل می‌باشد.

نکته عملی

برای تعریف معادله دیفرانسیل مرتبه دوم (یا مراتب بالاتر از دو) برای استفاده در دستور `dsolve` کافیست بجای y' از `Dy`، بجای y'' از `D2y` و در حالت کلی بجای $y^{(n)}$ از `Dny` استفاده کنیم.

مثال ۲۵.۱۱. جواب عمومی معادله دیفرانسیل $t^2 y'' - t(t+2)y' + (t+2)y = 0$ را بیابید. شکل استاندارد معادله به صورت $y'' = \frac{t+2}{t}y' - \frac{t+2}{t^2}y$ است.

ورودی

```
syms y t;
ode='D2y=(t+2)/t*Dy-(t+2)/t^2*y';
Sol = dsolve(ode)
```

خروجی

```
Sol =
C4*t + C5*t*exp(t)
```

از معادلات دیفرانسیل می‌دانیم که برای تعیین یک جواب خصوصی برای معادله دیفرانسیل مرتبه دوم باید دو شرط اولیه به صورت

$$y(a) = y_0, \quad y'(a) = y_1,$$

تامین شوند. برای استفاده از دستور `dsolve` برای تعیین جواب خصوصی معادله دیفرانسیل مرتبه دوم باید این دو شرط اولیه در قالب دو دستور `cond1` و `cond2` (اسامی اختیاری هستند) تعریف شوند، سپس به دستور `dsolve` ارسال شوند.

مثال ۲۶.۱۱. جواب معادله دیفرانسیل

$$y'' - 2y' + 2y = 0, \quad y(0) = 1, \quad y'(0) = 0,$$

را بیابید.

با دستورات زیر می‌توان جواب خصوصی را یافت.

ورودی

```
syms y t;
ode='D2y= 2*Dy -2*y';
cond1 = 'y(0) = 1';
cond2 = 'Dy(0) = 0';
Sol=dsolve(ode,cond1,cond2)
```

خروجی

```
Sol =
exp(t)*cos(t)-exp(t)*sin(t)
```

توجه کنید که برای یافتن جواب خصوصی معادلات دیفرانسیل مرتبه n باید n شرط اولیه به شکل زیر تامین شوند،

$$y(a) = y_0, \quad y'(a) = y_1, \quad \dots \quad y^{(n-1)}(a) = y_{n-1}.$$

۴.۳.۱۱ حل دستگاه معادلات دیفرانسیل

منظور از دستگاه معادلات دیفرانسیل، n معادله دیفرانسیل متمایز با n تابع $y_1(t), y_2(t), \dots, y_n(t)$ می‌باشند، به گونه‌ای که توابع y_1, y_2, \dots, y_n به طور همزمان در معادلات دیفرانسیل صدق می‌کنند.

مثال ۲۷.۱۱. دستگاه‌های زیر نمونه‌هایی از دستگاه معادلات دیفرانسیل هستند،

$$\begin{cases} 2x'(t) + x(t) + y'(t) = t, \\ x'(t) - x(t) + y'(t) = 0, \end{cases} \quad \begin{cases} x''(t) + x'(t) + y'(t) - x + y = t - 1, \\ x'(t) + 2y''(t) + 2x(t) - y'(t) = 1. \end{cases}$$

در این بخش هدف یافتن جواب تحلیلی برای این گونه دستگاه‌های معادلات دیفرانسیل با استفاده از جعبه‌ابزار Symbolic می‌باشد.

نکته عملی

روش حل دستگاه‌های معادلات دیفرانسیل مشابه با روش حل معادلات دیفرانسیل معمولی است، با این تفاوت که باید ابتدا هر معادله را به شکلی که در دستور dsolve بیان شد، تعریف کنیم، سپس تمام معادلات را به عنوان آرگومان‌های ورودی به دستور dsolve ارسال کنیم.

❏ اگر دستگاه دارای شرایط اولیه باشد، باید ابتدا شرایط را تعریف کرد، سپس آنها را به همراه معادلات به دستور `dsolve` فرستاد.

مثال ۲۸.۱۱. برای حل دستگاه معادلات دیفرانسیل،

$$\begin{cases} 2x'(t) + x(t) + y'(t) = t, \\ x'(t) - x(t) + y'(t) = 0, \end{cases}$$

به شکل زیر عمل می‌کنیم،

```
syms y x t;
eq1='2*Dx + x + Dy = t'; eq2 = 'Dx -x + Dy =0';
Sol = dsolve(eq1,eq2)
```

با اجرای دستورات بالا جواب دستگاه محاسبه شده و در متغیر `Sol` ذخیره می‌شود. برای دستیابی به جواب‌ها باید عبارات `Sol.x` و `Sol.y` را در سطر فرمان، یا در سطر بعد از دستور `dsolve` در MATLAB نوشت. همچنین می‌توان دو جواب را در دو متغیر جداگانه نیز ذخیره کرد. برای مثال با دستورات زیر می‌توان به دو جواب دست یافت. توجه کنید که برای ساده‌تر شدن عبارات از دستور `simplify` نیز استفاده شده است.

```
syms y x t;
eq1='2*Dx + x + Dy = t'; eq2 = 'Dx -x + Dy =0';
Sol = dsolve(eq1,eq2);
X = simplify(Sol.x)
Y = simplify(Sol.y)
```

خروجی حاصل از دستورات بالا به شکل زیر می‌باشد.

```
X =
t/2 - (2*C1*exp(-2*t))/3 - 1/4
Y =
```

$$C2 - (3*t)/4 + t^2/4 + C1*\exp(-2*t) + 3/8$$

مثال ۲۹.۱۱. دستگاه مثال پیش را برای شرایط اولیه $x(0) = 1$, $y(0) = 0$ بدست آورد، دستورات زیر را در یک MATLAB بنویسید و اجرا کنید.

```
syms y x t;
eq1='2*Dx + x + Dy = t'; eq2 = 'Dx -x + Dy =0';
cond1 = 'x(0) = 0'; cond2 = 'y(0) = 1';
Sol = dsolve(eq1,eq2,cond1,cond2);
X = simplify(Sol.x)
Y = simplify(Sol.y)
```

خروجی حاصل از دستورات بالا جواب خصوصی دستگاه و به شکل زیر می باشد.

```
X =
t/2 + exp(-2*t)/4 - 1/4
Y =
t^2/4 - (3*exp(-2*t))/8 - (3*t)/4 + 11/8
```

۵.۳.۱۱ محاسبه تبدیل لاپلاس و تبدیل معکوس لاپلاس

تبدیلات لاپلاس و تبدیلات معکوس لاپلاس در معادلات دیفرانسیل و ریاضیات مهندسی از اهمیت ویژه ای برخوردارند. در MATLAB و با استفاده از جعبه ابزار Symbolic به سادگی و با دو دستور ساده می تواند تبدیل لاپلاس و تبدیل معکوس لاپلاس را بدست آورد.

تبدیل لاپلاس و تبدیل معکوس لاپلاس

با استفاده از دو دستور `laplace` و `ilaplace` می‌توان تبدیل لاپلاس و تبدیل معکوس لاپلاس را بدست آورد. برای استفاده از این دو دستور باید ابتدا تابعی که می‌خواهیم تبدیل لاپلاس، یا تبدیل معکوس لاپلاس آن را محاسبه کنیم به شکل نمادین تعریف کنیم، سپس تابع تعریف شده را به عنوان آرگومان ورودی به دو دستور مربوطه ارسال نماییم. توجه کنید که توابع نباید در داخل جفت کوتیشن نوشته شوند.

مثال ۳۰.۱۱. تبدیل لاپلاس توابع زیر را محاسبه کنید،

$$f(t) = t^5, \quad g(t) = \sin(2t), \quad h(t) = e^t, \quad s(t) = te^{-2t} \sin(3t)$$

ورودی

```
syms t;
f = t^5; g = sin(2*t);
h = exp(t);
s = t*exp(-2*t)*sin(3*t);
Lf = laplace(f)
Lg = laplace(g)
Lh = laplace(h)
Ls = laplace(s)
```

خروجی

```
Lf =
120/s^6
Lg =
2/(s^2 + 4)
Lh =
1/(s - 1)
Ls =
(3*(2*s+4))/((s+2)^2+9)^2
```

مثال ۳۱.۱۱. تبدیل معکوس لاپلاس توابع زیر را بدست آورید.

$$f(t) = \frac{2t}{t^4 + 1}, \quad g(t) = \frac{t - 1}{t^2 - 6t + 9}.$$

```
syms t;
Lf = 2*t/(t^4 + 1); f = ilaplace(Lf)
Lg = (t-1)/(t^2 - 6*t + 9); g = ilaplace(Lg)
```

خروجی دستورات بالا به شکل زیر است.

```
f = 2*sin((2^(1/2)*x)/2)*sinh((2^(1/2)*x)/2)
```

```
g = exp(3*x) + 2*x*exp(3*x)
```

۴.۱۱ کاربرد جعبه‌ابزار Symbolic در رسم نمودار

رسم نمودارها در MATLAB با استفاده از جعبه‌ابزار Symbolic کار بسیار ساده‌تری نسبت به روش متداول در MATLAB می‌باشد، ولی در استفاده‌های عملی و در حل مسایل به کمک MATLAB به هر دو روش نیاز است، پس پیشنهاد می‌شود هر دو روش را به خوبی یاد بگیرید تا در موارد ضروری بتوانید آنها را بکار ببرید.

۱.۴.۱۱ رسم در فضای دوبعدی

برای رسم نمودار توابع در فضای دوبعدی می‌توان از جعبه‌ابزار Symbolic کمک گرفت. در این بخش به معرفی دستورات لازم برای رسم نمودار و قالب‌بندی نمودار در فضای دوبعدی خواهیم پرداخت.

رسم نمودار با دستور ezplot

برای رسم نمودار از دستور ezplot استفاده می‌شود. شکل کلی این دستور به صورت $\text{ezplot}(\text{function}, [\text{min}, \text{max}])$ می‌باشد که در آن

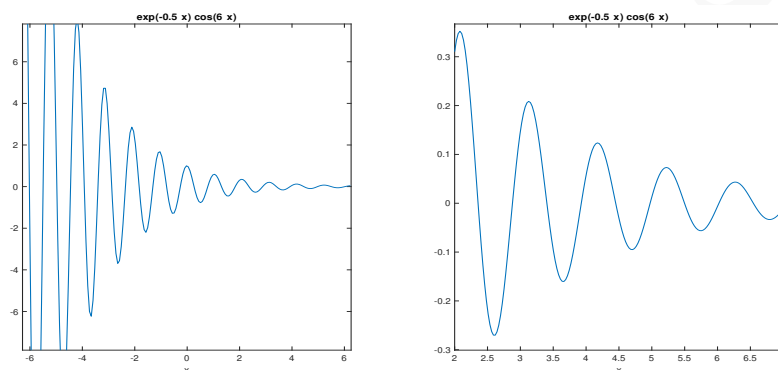
function تابعی است که می‌خواهیم نمودار آن را رسم کنیم. این تابع باید به صورت نمادین تعریف شود.

min, max ابتدا و انتهای بازه‌ای است که می‌خواهیم نمودار در آن بازه رسم شود. اگر این دو مقدار نوشته نشوند، نمودار در بازه $[-2\pi, 2\pi]$ رسم می‌شود.

مثال ۳.۲.۱۱. دستورات زیر نمودار تابع $y = e^{-x/5} \cos(6x)$ در دو حالت بازه خودکار و بازه $[2, 7]$ رسم می‌کند.

```
syms x;
y = 'exp(-0.5*x)*cos(6*x)';
subplot(1,2,1); ezplot(y);
subplot(1,2,2); ezplot(y,[2,7]);
```

خروجی دستورات بالا به شکل زیر می باشد.



در دستور `ezplot` نمی توان از گزینه هایی مانند `Line Style` و `Property Value` که در دستور `plot` بین کردیم استفاده کنیم. برای ایجاد تغییراتی در ظاهر نمودارهای رسم شده با دستور `ezplot` می توان به شکلی که در زیر بیان می شود عمل کنیم.

دستور set

با دستور `set` می توان در نمودارهای رسم شده توسط دستور `ezplot` و همچنین دستور `plot` تغییراتی ایجاد کرد. به این منظور به شکل زیر عمل کنید،

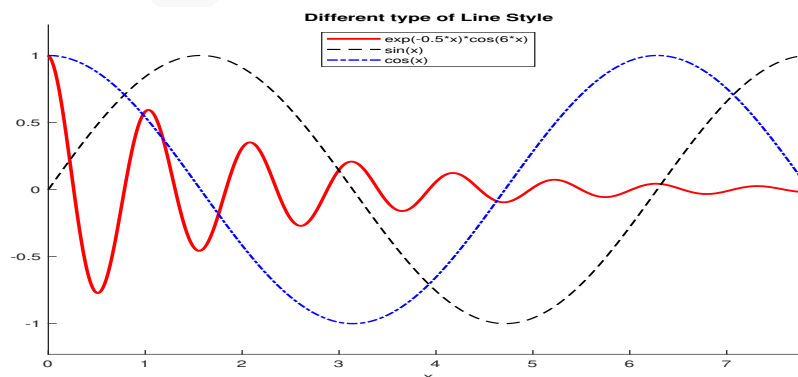
```
fig = ezplot(...);
set(fig,'Option','value','option','value',...);
```

که در دستور `set` بجای `Option` می توان گزینه هایی مانند `color`، `LineStyle` و `LineWidth` قرار داد که کارایی آنها دقیقاً مشابه دستور `plot` می باشد. بدیهی است که بجای `value` نیز باید دقیقاً همان مقادیری را قرار داد که برای دستور `plot` مورد استفاده قرار می گرفت.

مثال ۳۳.۱۱. به چگونگی استفاده از دستور `set` برای ایجاد تغییرات در نوع خط، رنگ خط و ضخامت خط در دستورات زیر دقت کنید. به دلیل وجود تنوع رنگ در دستورات زیر پیشنهاد می‌شود تمام کدهای زیر را در یک فایل `m-` وارد کنید و اجرا کنید. در دو سطر آخر از دستورات `title` و `legend` برای تولید عنوان نمودار و راهنمای نمودار به همان صورتی که برای دستور `plot` استفاده می‌شود، استفاده شده است.

```
syms x;
f = 'exp(-0.5*x)*cos(6*x)';
hold on
figf = ezplot(f,[0,2.5*pi]);
set(figf,'color','red','LineStyle','-','LineWidth',2);
g = sin(x);
figg = ezplot(g,[0,2.5*pi]);
set(figg,'color','black','LineStyle','--','LineWidth',1.25);
h = cos(x);
figh = ezplot(h,[0,2.5*pi]);
set(figh,'color','blue','LineStyle','-.','LineWidth',1.5);
title('Different type of Line Style')
legend('exp(-0.5*x)*cos(6*x)','sin(x)','cos(x)','Location','North');
```

خروجی دستورات بالا به شکل زیر می‌باشد.



نکته عملی

از دستور `ezplot` می‌توان برای رسم نمودار توابعی به شکل $f(x, y) = 0$ استفاده کرد. به این منظور باید از دستور `ezplot` به شکل زیر استفاده کرد. بدیهی است که بازه مورد نظر برای رسم نمودار باید در آرگومان دوم به دستور ارسال شود.

`ezplot(function, [xmin, xmax, ymin, ymax])`

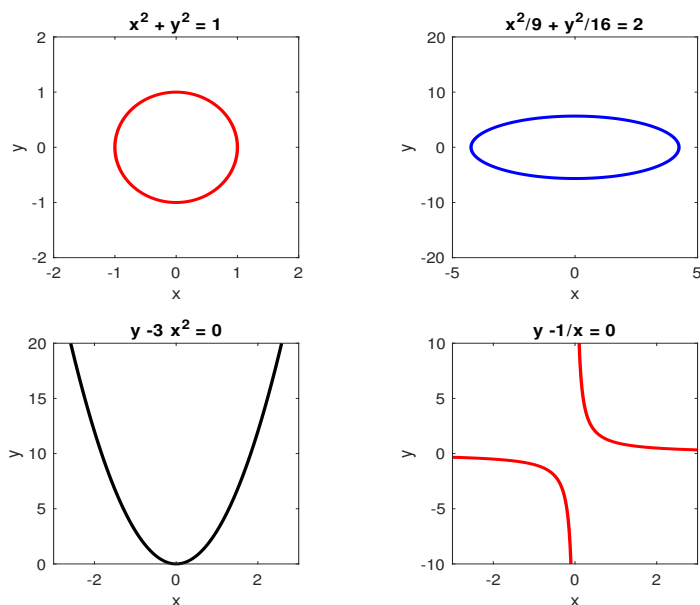
مثال ۳۴.۱۱. نمودار توابع زیر را در شکل‌های جداگانه‌ای و در بازه‌های مناسب رسم کنید،

$$x^2 + y^2 = 1; \quad \frac{x^2}{9} + \frac{y^2}{16} = 2, \quad y = 3 * x^2, \quad y = \frac{1}{x}.$$

```
syms x y;
f = 'x^2 + y^2 = 1';
subplot(2,2,1);
figf = ezplot(f,[-2,2,-2,2]); axis square;
set(figf,'color','red','LineStyle','-','LineWidth',2);
g = 'x^2/9 + y^2/16 = 2';
subplot(2,2,2);
figg = ezplot(g,[-5,5,-20,20]); axis square;
set(figg,'color','blue','LineStyle','-','LineWidth',2);
g = 'y - 3*x^2';
subplot(2,2,3);
figg = ezplot(g,[-3,3,0,20]); axis square;
set(figg,'color','black','LineStyle','-','LineWidth',2);
g = 'y - 1/x';
subplot(2,2,4);
figg = ezplot(g,[-3,3,-10,10]); axis square;
set(figg,'color','red','LineStyle','-','LineWidth',2);
```

دو نمودار آخر با شکل اول دستور `ezplot` نیز قابل رسم می‌باشند. خروجی دستورات بالا

به شکل زیر می باشد.



رسم توابع پارامتری و نمودارهای قطبی

برای رسم توابع پارامتری به شکل زیر عمل کنید. ابتدا ضابطه مولفه های تابع را به صورت نمادین تعریف کنید، سپس تابع `ezplot(x,y)` را به شکل `ezplot(x,y)` فراخوانی کنید که در آن x و y ضابطه مولفه های تابع پارامتری می باشند.

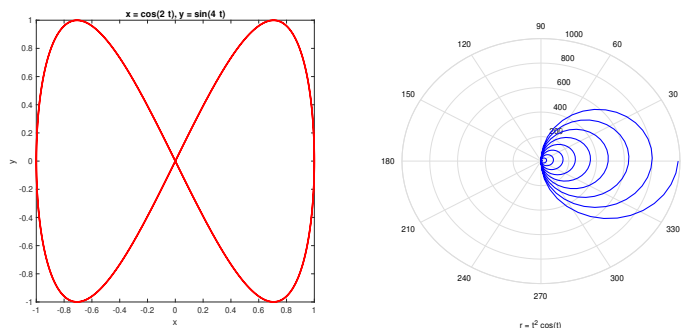
برای رسم نمودارهای قطبی باید پس از تعریف ضابطه تابع $r = f(\theta)$ به صورت نمادین، نمودار را با دستور `ezpolar(z, [min,max])` به صورت `ezpolar(z, [min,max])` رسم کرد.

مثال ۳۵.۱۱. نمودار تابع پارامتری با ضابطه های $x(t) = \cos(2t)$, $y(t) = \sin(4t)$ و نمودار قطبی با ضابطه $r = \theta^2 \cos \theta$ را در بازه های مناسب رسم کنید.

```
syms t;
x = 'cos(2*t)'; y = 'sin(4*t)';
subplot(1,2,1); figpara = ezplot(x,y); axis square;
set(figpara,'color','red','LineWidth',2);
```

```
r = 't^2*cos(t)';
subplot(1,2,2); figpolar = ezpolar(r,[0,10*pi]); axis square;
set(figpolar,'color','blue','LineWidth',1.3);
```

خروجی دستورات بالا به شکل زیر می باشد.



۲.۴.۱۱ رسم نمودار در فضای سه بعدی

با دستوراتی مشابه ezplot این امکان وجود دارد که نمودار توابع را در فضا رسم کنیم. با این دستورات می توان منحنی های خط و رویه ها را به شکل ساده تری نسبت به دستوراتی که در فصل های پیش دیدیم رسم کرد.

رسم منحنی خط با دستور ezplot3

با دستور ezplot3 می توان منحنی های خط را در فضا رسم کرد. به این منظور مشابه رسم توابع پارامتری که در بخش ۱.۴.۱۱ بیان کردیم باید ابتدا مولفه های تابع را به صورت نمادین و برحسب t تعریف کنیم، سپس از دستور زیر استفاده کنیم.

```
ezplot3(x,y,z,[tmin,tmax])
```

که در آن x, y, z مولفه های تابع هستند و بازه مورد نیاز با $tmin, tmax$ مشخص شده است.

مثال ۳۶.۱۱. نمودار تابع زیر را در بازه $[0, 10\pi]$ رسم کنید.

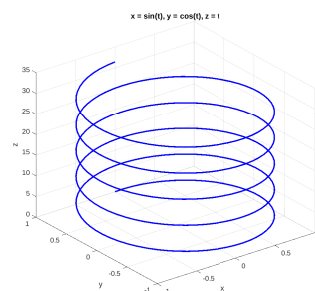
$$x(t) = \sin t, \quad y(t) = \cos t, \quad z(t) = t$$

با دستورات زیر می‌توان نمودار را رسم کرد. توجه کنید می‌توان از دستور `set` برای ایجاد تغییرات در ظاهر منحنی در دستور `ezplot3` نیز استفاده کرد.

ورودی

```
syms t;
x = sin(t);
y = cos(t);
z = t
ezplot3(x,y,z,[0,10*pi]);
```

خروجی



اگر به عنوان آرگومان آخر و پس از تعیین بازه رسم، گزینه `'animate'` قرار داده شود، نمودار به شکل متحرک ترسیم خواهد شد. پیشنهاد می‌شود در محیط MATLAB از این گزینه استفاده کنید و خروجی را مشاهده کنید.

۳.۴.۱۱ رسم رویه‌های فضایی

در فصل ۱۰ با چگونگی رسم رویه‌های فضایی با دستورات `mesh` و `surf` آشنا شدیم. در این بخش روش ساده‌تری برای ترسیم رویه‌ها معرفی خواهد شد. این روش از پیچیدگی کمتری نسبت روش‌های پیشین برخوردار است ولی کماکان هر دو روش دارای کاربردهای خاص خود است.

رسم رویه‌های فضایی

با دستورات `ezsurf` و `ezsurfz` می‌توان یک رویه را رسم کرد. به این منظور تابع $z = f(x, y)$ را به صورت نمادین تعریف کنید، سپس بسته به نیاز خود یکی از دستورات زیر را فراخوانی کنید.

```
ezsurf(z, [min,max])
```

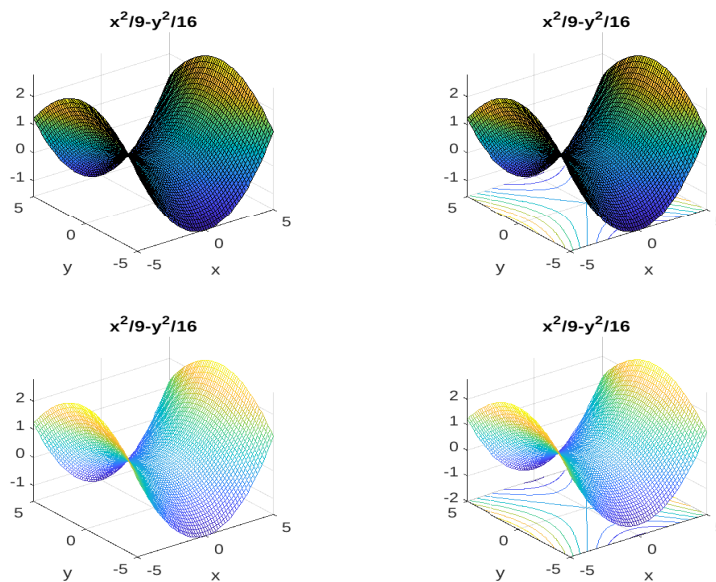
```
ezsurfz(z, [min,max])
```

دستور اول رویه z را در بازه مشخص شده رسم می‌کند و دستور دوم، رویه به همراه منحنی‌های کانتور آن رسم خواهد کرد.

مثال ۳۷.۱۱. رویه $z = \frac{x^2}{9} - \frac{y^2}{16}$ را در بازه $[-5, 5]$ رسم کنید. نمودار شامل منحنی‌های کانتور را در نمودار جداگانه‌ای رسم کنید.

```
syms x y;
z = 'x^2/9-y^2/16';
subplot(1,2,1); ezsurf(z, [-5,5]); axis square;
subplot(1,2,2); ezsurfz(z, [-5,5]); axis square;
subplot(2,2,3); ezmesh(z, [-5,5]); axis square;
subplot(2,2,4); ezmeshc(z, [-5,5]); axis square;
```

خروجی حاصل از دستورات بالا به صورت زیر است.



👉 در دو دستور آخر از `ezmesh` و `ezmeshc` برای رسم رویه به شکل شبکه‌بندی شده و بدون رنگ‌آمیزی استفاده شده است. روش استفاده از این دو دستور کاملاً مشابه دستورات `ezsurf` و `ezsurfz` می‌باشد.

👉 حتماً کدهای بالا را در محیط MATLAB اجرا کنید تا تصاویر را به صورت رنگی مشاهده کنید.

رسم منحنی‌های کانتور

با دو دستور `ezcontour` و `ezcontourf` می‌توان منحنی‌های کانتور را رسم کرد. روش استفاده از این دو دستور به صورت

```
ezcontour(z, [min,max], n)
ezcontourf(z, [min,max], n)
```

می‌باشد که در آن z تابعی است که باید به شکل نمادین تعریف شده باشد، بازه، همانی است که در دستورات مشابه معرفی شدند و n یک عدد طبیعی دلخواه است که اگر در هنگام استفاده از دستور نوشته شود، تعیین‌کننده بعد شبکه‌ای است که برای تولید دامنه در نظر گرفته می‌شود. عدد بزرگتر منجر به شکل دقیق‌تری می‌شود.

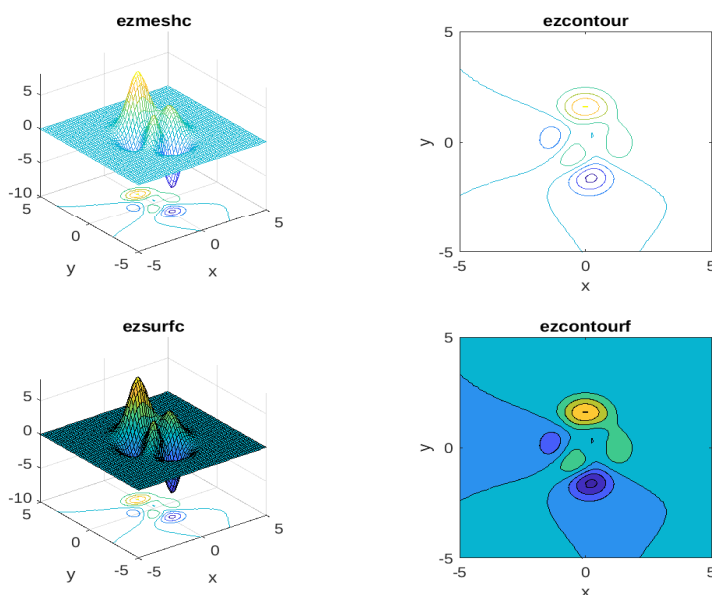
مثال ۳۸.۱۱. به دستورات زیر که برای رسم رویه

$$z = 3(1-x)^2 e^{-x^2} - (y+1)^2 - 10\left(\frac{x}{5} - x^3 - y^5\right) e^{-x^2-y^2} - \frac{1}{3} e^{-(x+1)^2-y^2},$$

بکار رفته است توجه کنید.

```
syms x y;
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);
subplot(2,2,1); ezmeshc(z,[-5,5]);
title('ezmeshc'); axis square;
subplot(2,2,2); ezcontour(z,[-5,5]);
title('ezcontour'); axis square;
subplot(2,2,3); ezsurf(z,[-5,5]);
title('ezsurf'); axis square;
subplot(2,2,4); ezcontourf(z,[-5,5]);
title('ezcontourf'); axis square;
```

خروجی دستورات بالا به شکل زیر می باشد. حتما این دستورات را در محیط MATLAB و در یک m-فایل اجرا کنید.



۵.۱۱ کاربرد جعبه‌ابزار Symbolic در جبر خطی

با استفاده از جعبه‌ابزار Symbolic می‌توان ماتریس‌ها را به شکل نمادین تعریف کرد. روش تعریف ماتریس به همان صورتی است که برای عبارات تعریف عبارات جبری نمادین در بخش‌های پیش بیان کردیم. پس از تعریف ماتریس می‌توان از تمام دستوراتی که برای ماتریس‌ها در فصل ۹ بیان کردیم، استفاده کرد. بدیهی است اگر ماتریس به شکل نمادین تعریف شده باشد خروجی حاصل از دستوراتی مانند `det`، `inv`، `eig`، `diff`، `int`، و دستورات دیگر، نیز به صورت نمادین خواهد بود.

مثال ۳۹.۱۱. ماتریس نمادین زیر را در نظر بگیرید. برخی از دستورات مرتبط با جبر خطی با دستوراتی که در ادامه می‌آید بر این ماتریس اعمال شده است. پیشنهاد می‌شود دستورات دیگری مانند دستورات مربوط به تجزیه ماتریس‌ها را در محیط MATLAB اجرا کنید.

$$A = \begin{bmatrix} 1 & t & t^2 \\ e^t & t-1 & \sin t \\ 2 & \ln t & \frac{1}{t} \end{bmatrix}.$$

به دستورات زیر و چگونگی تعریف ماتریس در دستورات زیر توجه کنید.

```
syms t;
A = [1,t,t^2;exp(t),t-1,sin(t);2,log(t),1/t]
D = det(A)
E = eig(A)
DA = diff(A)
IA = int(A)
```

خروجی دستورات بالا به شکل زیر می باشد.

```
A =
[      1,      t,      t^2]
[ exp(t),  t - 1, sin(t)]
[      2, log(t),   1/t]

D =
(t + 2*t^2*sin(t) - t*exp(t) + 2*t^3 - 2*t^4
 - t*log(t)*sin(t) + t^3*exp(t)*log(t) - 1)/t

DA =
[      0,  1,  2*t]
[ exp(t),  1, cos(t)]
[      0, 1/t, -1/t^2]

IA =
[      t,      t^2/2,  t^3/3]
[ exp(t),  (t*(t - 2))/2, -cos(t)]
[  2*t, t*(log(t) - 1),  log(t)]
```

❗ خروجی دستور eig به دلیل بسیار طولانی بودن آورده نشده است.

۶.۱۱ تمرین

تمرین ۱.۱۱. دو عبارت زیر را به صورت نمادین تعریف کنید،

$$S_1 = x^2(x - 6) + 4(3x - 2), \quad S_2 = (x + 2)^2 - 8x,$$

سپس

• ساده‌ترین شکل را برای عبارات زیر بدست آورید.

۳. $S_1 + S_2$

۲. $\frac{S_1}{S_2}$

۱. $S_1 \cdot S_2$

• با استفاده از دستور subs مقدار عبارات حاصل از بخش قبل را در $x = 5$ محاسبه کنید.

تمرین ۲.۱۱. تمرین ۱.۱۱ را برای دو عبارت زیر تکرار کنید.

$$S_1 = x(x^2 + 6x + 12) + 8, \quad S_2 = (x - 3)^2 + 10x - 5.$$

تمرین ۳.۱۱. دو عبارت زیر را به شکل نمادین تعریف کنید،

$$S = x + \sqrt{x}y^2 + y^4, \quad T = \sqrt{x} - y^2,$$

سپس ساده‌ترین شکل عبارت $S \cdot T$ را محاسبه کنید و با استفاده از دستور subs مقدار آن را برای $x = 9$ و $y = 2$ بدست آورید.

تمرین ۴.۱۱. متغیر x را به صورت نمادین تعریف کنید،

۱. یک چندجمله‌ای بدست آورید که ریشه‌های آن $x = -2$, $x = -5/2$, $x = 2$ و $x = 4/5$ باشند.

۲. ریشه‌های چندجمله‌ای زیر را با استفاده از دستور factor بدست آورید،

$$f(x) = x^6 - 6/5x^5 - 58x^4 + 167/5x^3 + 728x^2 - 890x - 1400.$$

تمرین ۵.۱۱. درستی تساوی‌های زیر را با دستورات مناسب در جعبه‌ابزار Symbolic ثابت کنید.

$$\bullet \sin(4x) = 4 \sin x \cos x - 8 \sin^3 x \cos x$$

$$\bullet \cos x \cos y = \frac{1}{2} [\cos(x-y) + \cos(x+y)]$$

$$\bullet \tan(3x) = \frac{3 \tan x - \tan^3 x}{1 - 3 \tan^2 x}$$

$$\bullet \sin(x+y+z) = \sin x \cos y \cos z + \cos x \sin y \cos z + \cos x \cos y \sin z - \sin x \sin y \sin z$$

تمرین ۶.۱۱. نمودار گلبیگ برای $t \neq -1$ دارای معادلات پارامتری به شکل زیر می باشد،

$$x(t) = \frac{3t}{1+t^3}, \quad y(t) = \frac{3t^2}{1+t^3}.$$

۱. با استفاده از MATLAB نشان دهید که معادله گلبیگ را می توان به شکل $x^3 + y^3 = 3xy$ نیز نوشت.

۲. با استفاده از دستور `ezplot` نمودار معادله گلبیگ را رسم کنید.

تمرین ۷.۱۱. دویضی با معادلات زیر را در نظر بگیرید،

$$\frac{(x-1)^2}{36} + \frac{y^2}{9} = 1, \quad \frac{(x+2)^2}{4} + \frac{(y-5)^2}{16} = 1.$$

۱. با استفاده از دستور `ezplot` نمودار هر دو بیضی را در یک نمودار رسم کنید.

۲. مختصات نقاط برخورد دو بیضی را بدست آورید.

تمرین ۸.۱۱. انتگرال های نامعین زیر را محاسبه کنید،

$$I = \int \frac{x^3}{\sqrt{1-x^2}} dx, \quad I = \int x^3 \cos x dx$$

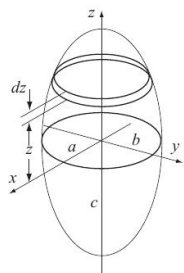
تمرین ۹.۱۱. متغیر x را به عنوان یک متغیر نمادین تعریف کنید، سپس تابع زیر را تعریف کنید،

$$S(x) = \frac{\cos^3 x}{1 + \sin^3 x}.$$

نمودار $S(x)$ را در بازه $0 \leq x \leq \pi$ رسم کنید، سپس $\int S(x) dx$ را محاسبه کنید.

تمرین ۱۰.۱۱.

معادلات پارامتری بیضی گون برای



$$0 \leq u \leq 2\pi, -\pi \leq v \leq 0$$

به صورت زیر می باشد،

$$x = a \cos u \sin v, y = b \cos u \sin v, z = c.$$

نشان دهید دیفرانسیل عنصر حجم بیضی گون به شکل

$$dV = -\pi abc \sin^2 v dv,$$

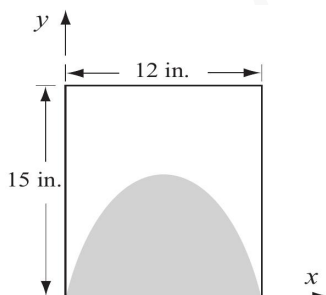
است. با استفاده از MATLAB، انتگرال dV را از $-\pi$ تا صفر و با استفاده از انتگرال گیری نمادین محاسبه کنید و نشان دهید که حجم بیضی با فرمول $V = \frac{4}{3}\pi abc$ بدست می آید.

تمرین ۱۱.۱۱. معادله انتشار در حالت یک بعدی به صورت $m \frac{\partial^2 u}{\partial x^2} = \frac{du}{dt}$ است. نشان دهید که دو تابع u که در زیر آمده است، جواب های معادله انتشار می باشند.

$$1. \quad u = A \frac{1}{\sqrt{t}} e^{\frac{-x^2}{4mt}}$$

۲. $u = Ae^{-\alpha x} \cos(\alpha x - 2m\alpha^2 t + B) + C$ که در آن A, B, C و α مقادیر ثابتی می باشند.

تمرین ۱۲.۱۱.



یک کاشی سرامیکی به صورتی که در شکل مقابل نمایش داده شده، طراحی شده است. ناحیه سایه خورده قرمز و بقیه کاشی سفید می باشد. معادله مرز میان ناحیه قرمز و سفید به صورت $y = -kx^2 + 12kx$ می باشد. مقدار k را به گونه ای تعیین کنید که مساحت دو ناحیه قرمز و سفید برابر باشند.

تمرین ۱۳.۱۱.

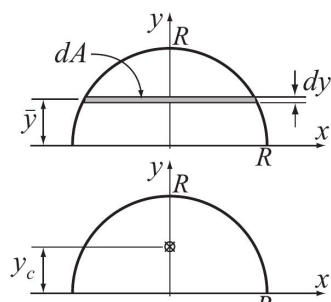
نشان دهید که موقعیت y_c که مرکز مساحت نیم‌دایره نمایش داده شده در شکل مقابل می‌باشد به شکل

$$y_c = \frac{4R}{3\pi}$$

است. مقدار y_c را می‌توان با رابطه زیر حساب کرد،

$$y_c = \frac{\int_A \bar{y} dA}{\int_A dA}.$$

تمرین ۱۴.۱۱. برای نیم‌دایره مسأله قبل، نشان دهید که گشتاور اینرسی حول محور x ها، I_x ، به شکل $I_x = \frac{1}{8}\pi R^4$ بدست می‌آید. این گشتاور را می‌توان به شکل $I_x = \int_A y^2 dA$ محاسبه کرد.



نمایه

بسته، ۴	MATLAB، ۱
سیمبولیک، ۱۱۳	Octave، ۱
ت	آ
تابع بدون نام، ۱۰۶	آرایه
تابع علامت، ۱۷	دوبعدی، ۳۲
تبدیل	یک بعدی، ۲۷
لاپلاس، ۲۶۴	ا
معکوس لاپلاس، ۲۶۴	اعداد مختلط، ۱۴
تجزیه	انتگرال گیری، ۲۵۳
چولسکی، ۲۰۹	اندازه
مقدار تکین، ۲۱۱	ماتریس، ۳۳
مقدار ویژه، ۲۱۱	ب
LU، ۲۰۶	باقیمانده، ۱۷
QR، ۲۰۸	بدنه تابع، ۱۰۲، ۱۰۳
ترانهاده، ۳۵	برچسب محور، ۱۳۲
مزدوج، ۳۵	بردار، ۲۷
تقدم عملیاتی، ۸	تکین، ۲۱۱
تقسیم آرایه ها، ۵۶	ویژه، ۲۰۹
تقسیم چپ، ۵۷	برش، ۱۷
توابع	بزرگترین مقسوم علیه مشترک، ۱۷
مثلثاتی، ۱۵	

- توان، ۵۹
- ج
- جذر، ۱۷
- جمع آرایه‌ها، ۵۱
- جواب
- دستگاه خطی، ۵۷
- چ
- چندجمله‌ای، ۱۶۳
- ح
- حد، ۲۵۱
- حدگیری، ۲۵۱
- حلقه
- تکرار، ۹۲
- for، ۹۳
- while، ۹۶
- خ
- خط پیوسته، ۱۱۶
- خط‌چین، ۱۱۶
- د
- درونیابی، ۱۷۵
- دستگاه معادلات دیفرانسیل، ۲۶۲
- دستورات
- گمارشی، ۱۰
- ر
- راهنمای نمودار، ۱۳۳
- رسم نمودار، ۱۱۳
- رشته، ۶۷
- رنگ خط، ۱۱۶
- ریشه، ۱۷
- س
- سری
- تیلور، ۲۵۶
- مک‌لورن، ۲۵۶
- سمی‌کالن، ۱۱
- سیمبولیک، ۲۳۷
- ض
- ضرب آرایه‌ها، ۵۳
- ضرب داخلی، ۵۴
- ع
- عملگر
- محاسباتی، ۷
- مقایسه‌ای، ۸۳
- منطقی، ۸۴
- عنوان نمودار، ۱۳۲
- ف
- فاکتوریل، ۱۷
- ق
- قدر مطلق، ۱۷
- ک
- کالن، ۳۱
- کاما، ۱۱
- کامنت، ۱۰۳

و	کانتور، ۲۲۶
واژه کلیدی، ۹	کوچکترین مضرب مشترک، ۱۷
A	گ
abs، ۱۵، ۱۷	گرد کردن، ۱۷
acos، ۱۵	ل
acosd، ۱۵	لگاریتم، ۱۷
acosh، ۱۵	لگاریتم طبیعی، ۱۷
acot، ۱۵	م
acotd، ۱۵	ماتریس، ۳۲
acoth، ۱۵	اسپارس، ۲۱۵
acsc، ۱۵	پاسکال، ۱۱۰
acscd، ۱۵	صفر، ۳۵
acsch، ۱۵	متعامد، ۲۰۸
all، ۸۵	همانی، ۳۵
AND، ۸۴	یک، ۳۵
and، ۸۵	ماکزیمم، ۱۸۹
angle، ۱۵	متغیر، ۸
ans، ۸	مشتق گیری، ۲۵۲
any، ۸۵	مقدار
asec، ۱۵	تکین، ۲۱۱
asecd، ۱۵	ویژه، ۲۰۹
asech، ۱۵	مینیمم، ۱۸۹
asin، ۱۵	ن
asind، ۱۵	نقطه خط، ۱۱۶
asinh، ۱۵	نقطه چین، ۱۱۶
atan، ۱۵	نمایش نقاط، ۱۱۷
atand، ۱۵	نوع خط، ۱۱۶
atanh، ۱۵	

۳۰ ،colon

۱۳۹ ،۱۲۱ ،Color

۲۶۷ ،۱۲۱ ،color

۲۲۴ ،colormap

۱۵ ،complex

۲۱۵ ،cond

۲۱۵ ،condest

۱۵ ،conj

۱۰۰ ،continue

۲۲۹ ،contour

۲۲۹ ،۳contour

۱۶۷ ،conv

۱۵ ،cos

۱۵ ،cosd

۱۵ ،cosh

۱۵ ،cot

۱۵ ،cotd

۱۵ ،coth

۶۳ ،cross

۱۵ ،csc

۱۵ ،cscd

۱۵ ،csch

۲۳۱ ،cylinder

D

۱۶۸ ،deconv

۲۷۶ ،۲۱۴ ،۶۳ ،det

۳۶ ،diag

۲۷۶ ،diff

۱۴۳ ،axis

۱۴۵ ،axis equal

۱۴۵ ،axis fill

۱۴۵ ،axis image

۱۴۵ ،axis normal

۱۴۶ ،axis off

۱۴۸ ،axis on

۱۴۵ ،axis square

۱۴۵ ،axis tight

B

۱۳۹ ،BackgroundColor

۱۵۵ ،bar

۲۳۲ ،bar3

۱۵۴ ،BarWidth

۱۳۷ ،\bf

۱۵۲ ،BinEdges

۱۵۲ ،BinLimits

۱۵۲ ،BinWidth

۱۳۹ ،bold

۱۳۳ ،box

۱۰۰ ،break

C

۱۵۴ categorical،

۱۷ ،ceil

۶۷ ،char

۱۳ ،۱۲ ،clc

۱۲ ،۱۱ ،clear

۲۴۲ ،collect

۱۷ ،fix	۷۸ ،disp
۱۷ ،floor	۶۳ ،dot
۱۸۹ ،fminbnd	۲۶۲ ،۲۶۱ ،dsolve
۱۳۹ ،FontAngle	E
۱۳۹ ،FontName	۱۳۴ ،east
۱۳۸ ،\fontname	۱۵۲ ،۱۳۹ ،EdgeColor
۱۳۹ ،FontSize	۲۷۶ ،eig
۱۳۸ ،\fontsize	۳۲ ،end
۱۳۹ ،FontWeight	۹۱ ،error
۱۳ ،format hex	۱۷ ،exp
۱۳ ،format long	۲۴۳ ،expand
۱۳ ،format short	۳۵ ،eye
۱۲۴ ،fplot	۲۷۴ ،ezcontour
۷۸ ،fprintf	۲۷۴ ،ezcontourf
۲۱۵ ،۲۱۴ ،full	۲۷۴ ،ezmesh
۱۰۱ ،function	۲۷۴ ،ezmeshc
۱۸۴ ،fzero	۲۷۱ ،۲۶۶ ،ezplot
G	۲۷۰ ،ezpolar
۱۷ ،gcd	۲۷۳ ،ezsurf
۱۴۸ ،grid minor	۲۷۳ ،ezsurfz
۱۴۸ ،grid off	F
۱۴۸ ،grid on	۱۵۲ ،FaceColor
۱۳۶ ،gtext	۲۴۴ ،factor
H	۱۷ ،factorial
۲۱ ،help	۱۳۰ ،figure
۱۵۰ ،histogram	۱۴۹ ،fill
۱۱۶ ،hold	۸۵ ،find
	۲۴۱ ،findsym

- | | |
|----------------------|--------------------------------------|
| ۱۳۴ ،location | I |
| ۱۷ ،log | ۸۹ ،۸۸ ،if-else-end |
| ۱۷ ،log10 | ۸۷ ،۸۶ ،if-end |
| ۱۴۶ ،loglog | ۲۶۵ ،ilaplace |
| ۳۱ ،۳۰ ،logspace | ۱۵ ،imag |
| ۲۰۶ ،lu | ۱۰۸ inline، |
| M | ۷۷ ،input |
| ۱۷۶ ،makima | ۲۷۶ ،۲۵۳ ،int |
| ۱۲۱ ،Marker | ۱۷۶ ،interpl |
| ۱۲۱ ،marker | ۲۷۶ ،۲۱۴ ،۶۳ ،inv |
| ۱۱۹ ،MarkerEdgeColor | ۹ ،iskeyword |
| ۱۱۹ ،markeredgecolor | ۱۰ ،isvarname |
| ۱۱۹ ،MarkerFaceColor | ۱۳۷ ،\it |
| ۱۱۹ ،markerfacecolor | L |
| ۱۱۹ ،MarkerSize | ۲۶۵ ،laplace |
| ۱۱۹ ،markerSize | ۱۳۷ ،L ^A T _E X |
| ۶۳ ،max | ۱۷ ،lcm |
| ۶۳ ،mean | ۲۶۸ ،۱۳۴ ،legend |
| ۶۳ ،median | ۲۸ ،length |
| ۲۷۲ ،۲۲۳ ،mesh | ۱۳۹ ،light |
| ۲۲۶ ،meshc | ۱۲۶ ،line |
| ۲۲۲ ،meshgrid | ۱۷۶ ،linear |
| ۶۳ ،min | ۲۶۷ ،۱۲۱ ،LineStyle |
| N | ۱۲۱ ،linestyle |
| ۱۷۶ ،nearest | ۲۶۷ ،۱۳۹ ،۱۱۹ ،LineWidth |
| ۱۷۶ ،next | ۱۱۹ ،linewidth |
| ۲۱۳ ،norm | ۳۰ ،linspace |
| ۱۳۹ ،normal | ۱۳۴ ،Location |

۲۳۳ ، pol2cart	۱۳۴ ، north
۱۲۸ ، polar	۱۳۴ ، northeast
۲۱۵ ، ۱۶۶ ، poly	۱۳۴ ، northwest
۱۶۹ ، polyder	۸۴ ، NOT
۱۷۱ ، polyfit	۸۵ ، not
۱۶۵ ، polyval	۱۷ ، nthroot
۲۴۵ ، pretty	۱۷۳ ، str2num
۱۷۶ ، previous	۱۵۲ ، NumBins
	۱۳۴ ، NumColumns
	۱۳۴ ، numcolumns
Q	
۲۰۸ ، qr	
۱۹۱ ، quad	
۱۹۲ ، quadl	
R	
۶۵ ، rand	
۶۶ ، randi	
۶۶ ، randn	
۶۵ ، randperm(n)	
۲۱۴ ، rank	
۱۵ ، real	
۱۷ ، rem	
۱۳۷ ، \rm	
۱۸۸ ، ۱۶۶ ، roots	
۱۳۹ ، Rotation	
۱۷ ، round	
S	
۱۵ ، sec	
۱۵ ، secd	
	O
	۱۹۶ ، ode15s
	۱۹۶ ، ode23
	۱۹۶ ، ode23s
	۱۹۶ ، ode23t
	۱۹۶ ، ode23tb
	۱۹۶ ، ode45
	۱۹۶ ، ode113
	۳۵ ، ones
	۸۴ ، OR
	۸۵ ، or
	۲۱۴ ، orth
	P
	۱۷۶ ، pchip
	۱۵۸ ، pie
	۲۳۲ ، pie3
	۱۲۶ ، ۱۱۴ ، ۱۱۳ ، plot
	۲۱۹ ، plot3

- ۲۳۹ ،sym
۲۳۷ ،۱۱۳ symbolic،
۲۳۹ ،syms
۲۵۴ ،symsum
۲۴۱ ،symvar
- T**
۱۵ ،tan
۱۵ ،tand
۱۵ ،tanh
۲۵۶ ،taylor
۱۳۶ ،text
۲۶۸ ،۱۳۲ ،title
۲۱۴ ،trace
۱۹۴ ،trapz
- V**
۱۷۶ ،cubeΔv
۲۵۴ ،vpa
۲۴۸ ،vpasolve
- W**
۲۲۸ ،waterfall
۱۳۴ ،west
۱۳ ،۱۲ ،who
۱۳ ،۱۲ ،whos
- X**
۱۳۲ ،xlabel
۸۵ ،xor
- ۱۵ ،sech
۱۴۶ ،semilogx
۱۴۶ ،semilogy
۱۷ ،sign
۲۴۴ ،simplify
۱۵ ،sin
۱۵ ،sind
۱۵ ،sinh
۳۳ ،size
۲۴۷ ،solve
۶۳ ،sort
۱۳۴ ،south
۱۳۴ ،southeast
۱۳۴ ،southwest
۲۱۵ ،sparse
۲۳۰ ،sphere
۱۷۶ ،spline
۲۱۷ ،spy
۱۷ ،sqrt
۲۶۷ ،st
۱۲۷ ،stairs
۱۲۹ ،subplot
۲۴۵ ،subs
۶۳ ،sum
۲۷۲ ،۲۲۳ ،surf
۲۲۶ ،surfc
۲۲۷ ،surfl
۲۱۱ ،svd

Y

۱۳۲، ylabel

Z

۳۵، zeros

نسخه
رایگان