```
>>[h,w]=freqs(b,a,n)
>>freqs(b,a)
```

.. ellip

تستخدم لتصميم مرشح تمرير ترددات منخفضة أو عالية أو تمرير مجال ترددي محدد أو حذف كمجال ترددي محدد.

```
>>[b,a]=ellip(N,Rp,Rs,Wn)
>>[b,a]=ellip(N,Rb,Rs,Wn,'high')
>>[b,a]=ellip(N,Rb,Rs,Wn,'stop')
```

N: مرتبة المرشح الذي بسطه b ومقامه a .

Wn: تردد القطع .عند كون تردد القطع مكونا من عددين فهذا يعني ترددي القطع الأدنى والأعلى . وعند عدم ذكر نوع المرشح فهو مرشح تمرير ترددات منخفضة أو تمرير مجال ترددي محدد .

'high': مرشح تمرير ترددات عالية .

'stop' : مرشح حذف مجال ترددي محدد .

Rb : لتحديد مقدار التعرج في مجال التمرير .

Rs : لتحديد مقدار التعرج في مجال الحذف .

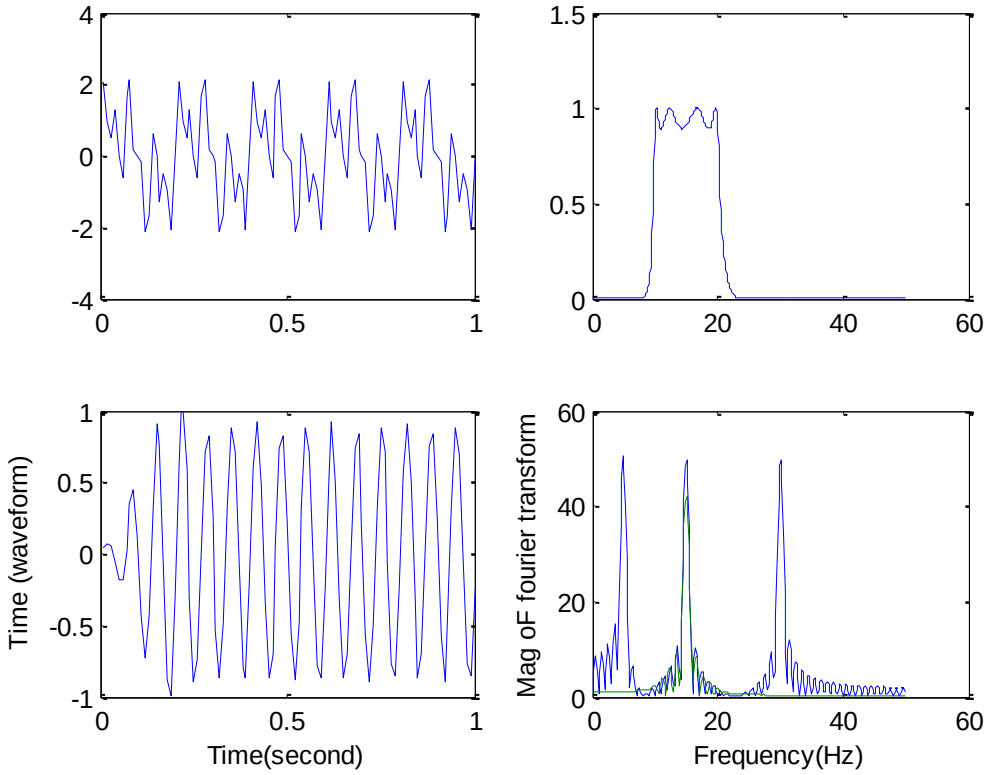
مثال : بفرض لدينا إشارة مكونة من مجموع ثلاثة مركبات ترددية .

نمرر هذه الإشارة على مرشح تمرير ترددات منخفضة بحيث تمر خلاله مركبة وحيدة ويتم حذف باقي المركبات الترددية .

تبين الأشكال الأصلية والاستجابة الترددية للمرشح وإشارة الخرج للمرشح بالإضافة إلى طيف الإشارة الأصلية وطيف الإشارة بعد ترشيحها .

```
>> fs=100;t=[1:100]/fs;
>> s1=sin(2*pi*5*t);s2=sin(2*pi*15*t);s3=sin(2*pi*30*t);
>> s=s1+s2+s3;
>> subplot(2,2,1);plot(t,s)
>> [b,a]=ellip(4,1,40,[10 20]*2/fs);
>> [h,w]=freqz(b,a,512);
>> subplot(2,2,2);plot(w*fs/(2*pi),abs(h))
>> sf=filter(b,a,s);
>> subplot(2,2,3);plot(t,sf)
>> xlabel('Time(second)')
>> ylabel('Time (waveform)')
```

```
>> axis([0 1 -1 1]);
>> sff=fft(s,512);
>> sfft=fft(sf,512);
>> w1=(0:255)/256*(fs/2);
>> subplot(2,2,4);plot(w1,abs([sff(1:256)' sfft(1:256)']));
>> xlabel('Frequency(Hz)');
>> ylabel('Mag oF fourier transform');
```



التـــكامل العـــددي ...

يمكن إيجاد المساحة المحصورة تحت التابع f(x) و يوجد تابعين في MATLAB لإيجاد التكامل الأحادي و هما:

quad, quad8 ,dblquad

- **quad** تستعمل لإيجاد التكامل حسب طريقة سمبسون.
- **quad8** تستعمل لإيجاد التكامل حسب طريقة نيوتن-كوتا.

فمثلاً لإيجاد التكامل للتابع f ضمن المجال 0 إلى 1 $q = \int_0^1 f(x)$ بطريقة سمبسون نكتب:

```
» q=quad('f',0,1)
```

و بطريقة نيوتن-كوتا:

```
» q=quad8('f',0,1)
```

- **Dblquad** تستعمل لإيجاد التكامل الثنائي .

فمثلاً لإيجاد التكامل للتابع f_2 $\mathbf{f_2(x,y)=y\cdot\sin(x)+x\cdot\cos(y)}$ نكتب: $\mathbf{q_1=\int_{p_1}^{p_2}\int_{0}^{p_1}}$

```
» q1=dblquad('f2',xmin,xmax,ymin,ymax)
```

حيث:

x_{min}= p1 , x_{max}=p2 , y_{min}=0 , y_{max}=p1

الاستيفاء **Interpolation ...**

هو عملية إيجاد القيم التي تقع بين نقاط معلومة. يوجد في MATLAB توابع للاستيفاء أحادي البعد و ثنائي البعد و ثلاثي البعد. سنستعرض فيما يلي الاستيفاء أحادي البعد One Dimensional Interpolation حيث يوجد نوعين منه:

- استيفاء كثير الحدود Polynomial interpolation.
- استيفاء التوابع الدورية FFT-based interpolation

أولاً - استيفاء كثير الحدود ...

و يتم باستخدام التابع **interp1** ، و يستعمل هذا التابع تقنيات كثيرات الحدود حيث يقوم هذا التابع بتحديد تابع كثير الحدود و المار من النقاط المفروضة و بعد ذلك يقوم بحساب قيم التابع المقابلة للنقاط المطلوبة.

شكله العام

y_i=**interp1**(x,y,x_i,method)

حيث:

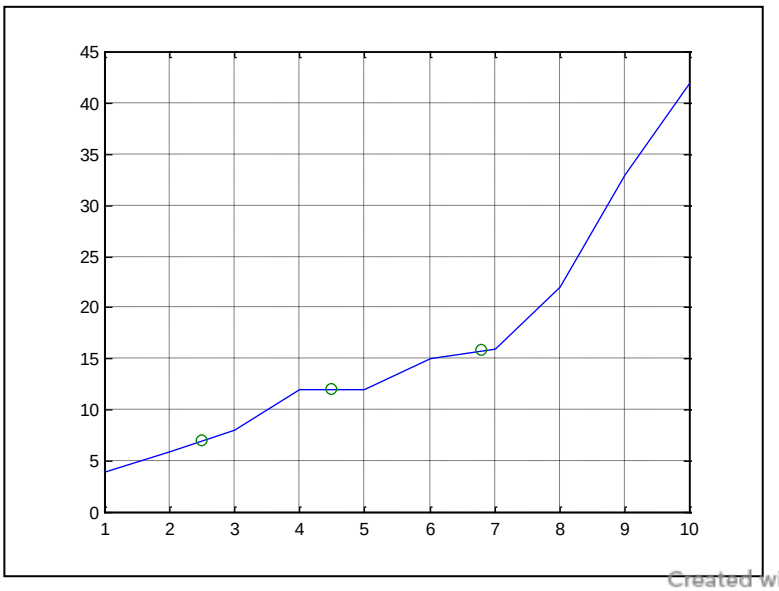
- x شعاع يحوي على قيم المتحول x .
- y شعاع بنفس الطول يحوي على قيم التابع y المقابلة لقيم المتحول x.
- x_i شعاع يحوي على النقاط التي سيتم حساب قيمة التابع y من أجلها (سيتم حساب الاستيفاء).
- Method عبارة عن سلسلة حروف اختيارية تدل على الطريقة التي سيتم فيها الاستيفاء، يوجد أربعة طرق:
  - الاستيفاء المقرب (method='nearest'): تقرب هذه الطريقة القيم المطلوب حساب قيمة التابع فيها إلى أقرب قيمة موجودة للتابع y و تستعمل هذه الطريقة نفس الألغوريتم المستعمل في التابع **round**

فإذا كان القسم الكسري للقيم المطلوبة أصغر من 0.5 يتم إعطاء x_i القيمة السابقة الموجودة و إذا كان القسم الكسري أكبر من 0.5 يتم إعطاء x_i القيمة اللاحقة الموجودة.

- الاستيفاء الخطي (method='liner'): في هذه الطريقة يتم تحديد التوابع المنفصلة بين كل زوج من القيم المعطاة و يتم حساب قيمة التابع المناسب من أجل كل قيمة لـ x_I و هذه هي الطريقة الافتراضية عند عدم تحديد طريقة من التابع interp1 .
- الاستيفاء التجزيئي (method='spline'): في هذه الطريقة يتم استعمال سلسلة من التوابع لإيجاد القيم المطلوب حساب الاستيفاء عندها (القيم المطلوب إيجاد قيمة التابع عندها) كما في طريقة الاستيفاء الخطي حيث يتم تعيين تابع مناسب من كل نقطتين متتاليتين من نقاط x_i و لكن شرط أن يكون لكل تابع نفس المشتق الأول و المشتق الثاني للتابع الذي يليه.
- الاستيفاء التكعيبي (method='cubic'): يتم تحديد التابع التكعيبي بهذه الطريقة بواسطة y و نحسب قيم هذا التابع عند النقاط المحددة لـ x_I و تأخذ النقاط التي تقع خارج المجال _____ nan.

في كل طريقة من هذه الطرق تتطلب أن يكون x مرتبة بشكل تصاعدي أو تنازلي، مثال:

```
» plot(x,y,xi,yi,'o')
» grid on
» x=1:10;
» y=[4 6 8 12 12 15 16 22 33 42];
» xi=[2.5 4.5 6.8];
» yi=interp1(x,y,xi);
» plot(x,y,xi,yi,'o')
» grid on
```



الشكل (7- 6)

ثانياً - استيفاء التوابع الدورية ...

يتم باستعمال التابع **interpft** و الذي يأخذ الشكل العام التالي:

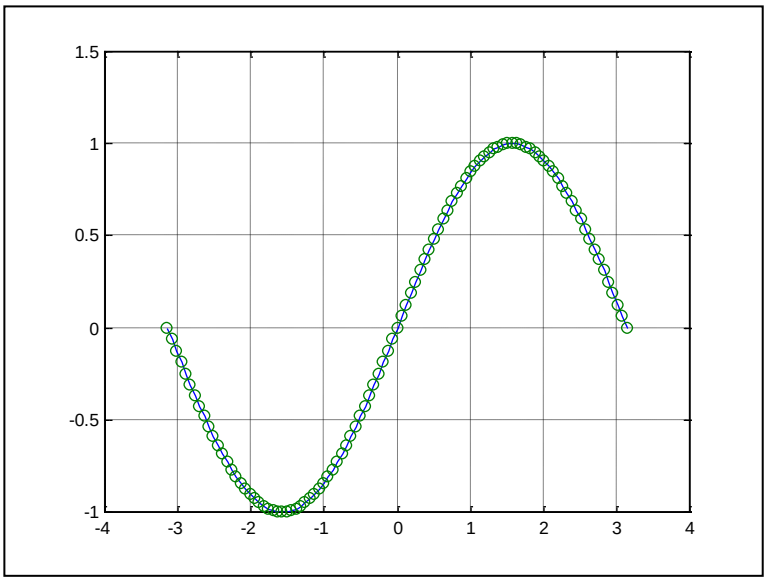
y=**interpft** (x,n)

حيث:

- x شعاع يحتوي على قيم تابع دوري و هي قيم مأخوذة عند زمن متساوي (البعد بينها متساوي).
- n عدد من النقاط (البعد بينها أيضاً متساوي) المطلوب إيجاد قيمة التابع عندها.

مثال:

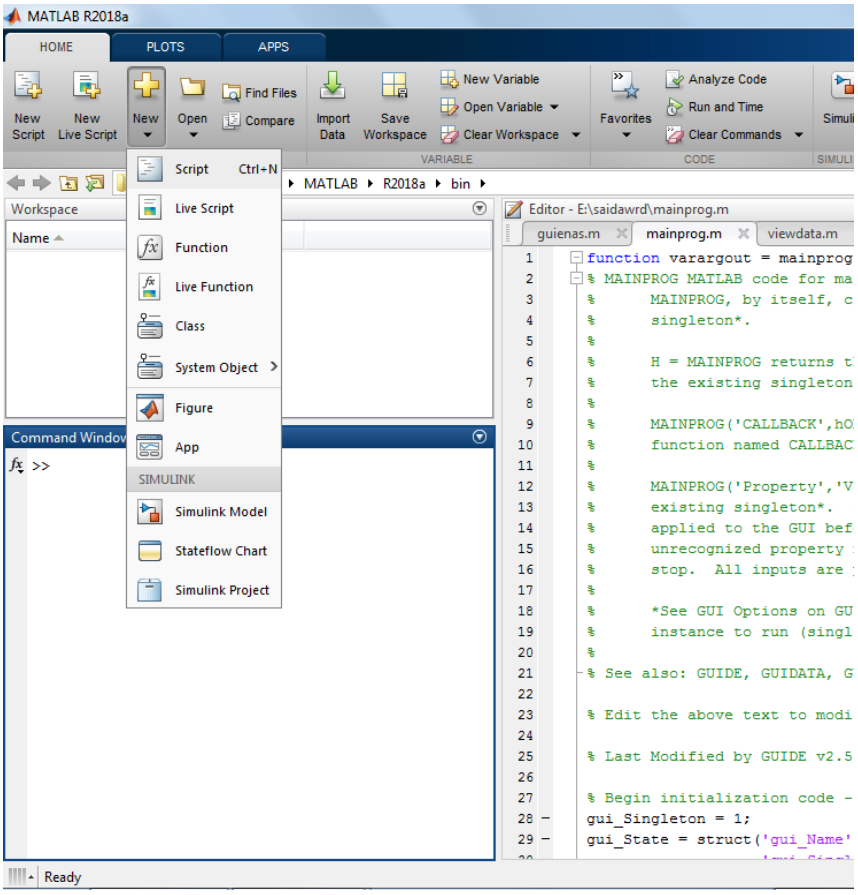
```
» x=-pi:pi/50:+pi;
» y=sin(x);
» yi=interpft(y,length(x));
» plot(x,y,x,yi,'o')
» grid on
```



الشكل (7- 6)

البرمجة في MATLAB ...

الملفات التي تحتوي على شفرة لغة MATLAB تدعى ملفات M يوجد نوعين من هذه الملفات الأول يدعى ملف نص أو مستند **Script** و الثاني يدعى بملف الإجراء **Function** .



توليد ملف M : ملف M هو عبارة عن ملف نص عادي يمكن توليده باستعمال أي محرر نصوص و يحتوي MATLAB على محرر Editor داخلي و لكن MATLAB يسمح باستخدام أي محرر نصوص لتوليد هذه الملفات. لفتح محرر ملفات M نختار New ثم Script او القائمة home ثم Script

يمكن إظهار محتوى الملف M-File ذو الاسم file_name بكتابة type file_name فيتم إظهار محتويات هذا الملف في نافذة الأوامر.

ملف النص ...

ملف النص هو من أبسط أنواع ملفات M حيث لا تحوي ملفات النص على متحولات دخل أو متحولات خرج و هي مفيدة بشكل خاص عند تنفيذ مجموعة من الأوامر أو الخطوات أو العلاقات التي يتكرر تنفيذها عدد من المرات في نافذة الأوامر.

نستخدم ملف النص المتحولات الموجودة في ساحة العمل و يمكنه توليد متحولات جديدة أثناء إجراء الحسابات و تنفيذ الأوامر و أي متحول جديد يتولد بواسطة ملفات النص يبقى مخزناً في ساحة العمل بعد انتهاء عمل ملف النص حيث يمكن استعمال هذه المتحولات في إجراء الحسابات التالية:

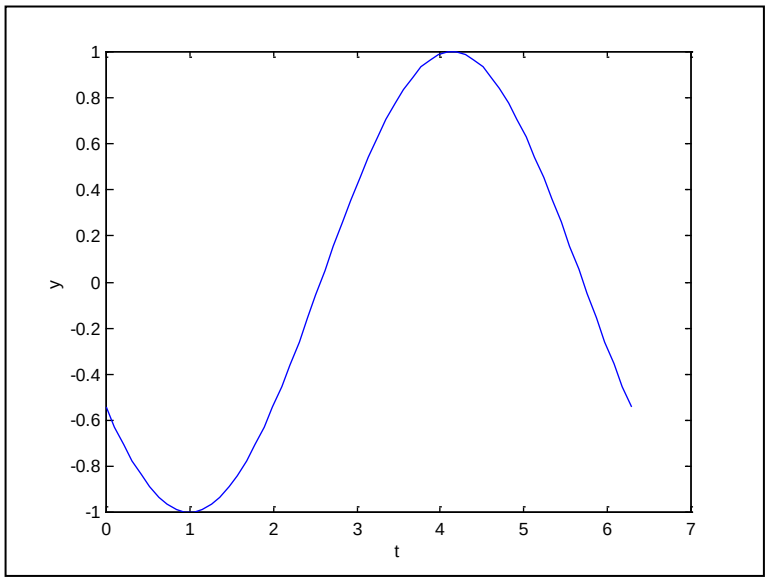
لكتابة ملف نص نفتح في البداية محرر النصوص و نبدأ بكتابة الأوامر. أي سطر يبدأ بإشارة النسبة المئوية % يعتبر سطر ملاحظات أي أن المحرر يتجاهله و وظيفته فقط لتوضيح البرنامج إذا لزم الأمر. فمثلاً لكتابة ملف نص لرسم التابع sin(3x)+5 نكتب الأوامر التالية و يشمله تحت اسم sinus.

```
%plot sin(x+t) , x an integer.
t=0:pi/30:2*pi;
y=sin(x+t);
plot(t,y);
xlabel('t');
ylabel('y');
shg
```

نلاحظ أنه لا يوجد أية متحولات دخل أو خرج و يكفي لاستدعاء ملف النص تعيين قيم المتحولات الداخلة في حسابه (في هذه الحالة تعيين قيمة x) و من ثم كتابة اسمه فقط sinus في نافذة الأوامر فيقوم بتنفيذ مجموعة الأوامر المكتوبة فيه.

```
» x=10
x =
   10
» sinus
```

سيعطي البرنامج الشكل الموضح:



الشكل (4-1)

عند عدم إدخال قيمة x أي عدم تعريف المتحول x سيتم إعلام المستخدم بذلك.

```
» sinus
??? Undefined function or variable 'x'.

Error in ==> D:\MATLABR11\work\sinus.m
On line 3  ==> y=sin(x+t);
```

ملف الإجراء Function M-File ...

ملفات الإجراءات هي عبارة عن ملفات من نوع M تستقبل متحولات دخل و ترجع متحولات خرج. و تستعمل ملفات الإجراءات متحولات ضمن ساحة عمل خاصة بها و هذه الساحة منفصلة عن ساحة عمل أوامر MATLAB . و كمثال بسيط على ملف الإجراء Function نحسب قيمة العاملي (!) لأي عدد n.

```
function f=fact(n)
% Fact Factorial
% Fact(n) Returns the factorial of number n
% Write simply fact(n)
f=prod(1:n); % Actual computation
```

يستعمل الإجراء متجول دخل وحيد و هو n و يرجع متحول خرج واحد هو f . لاستدعاء هذا الإجراء يمكن كتابة اسم الإجراء.

```
» fact(7)
ans =
     5040
```

أو أن نكتب

```
» t=fact(7)
t =
     5040
```

الأجزاء الرئيسية المكونة لملف الإجراء Function M-File

1. سطر تعريف الإجراء.
2. السطر H1.

3. نص التعليمات.
4. جسم الإجراء.
5. ملاحظات.
6. سطر تعريف الإجراء.

سطر تعريف الإجراء: يُعلِم سطر تعريف الإجراء برنامج MATLAB أن ملف M يحتوي على إجراء و يحدد طريقة استدعاء هذا الإجراء فمثلاً سطر تعريف الإجراء fact هو:

```
function f=fact(n)
```

حيث أن f متحول، fact اسم الإجراء و n متحول.

إذا كان للإجراء عدة متحولات خرج يتم إحاطتها بقوسين مربعين أما إذا كان للإجراء أكثر من متحول دخل فيتم إحاطة المتحولات بقوسين عاديين، على سبيل المثال:

```
function [x1,x2]=equation(a,b,c)
```

أما إذا لم يكن هناك متحولات خرج فنترك متحول الخرج فارغاً.

```
function printresults(x)
```

أو تستعمل أقواس مربعة فارغة.

```
function []=printresults(x)
```

إن المتحولات التي ستدخل إلى الإجراء ليس من الضروري أن يكون لها نفس الاسم.

السطر H1 : يسمى السطر الأول بالسطر H1 لأن السطر H1 يعتبر السطر الأول من التعليمات Help التي يمكن استدعاؤها بأمر help و هو عبارة عن سطر ملاحظات يبدأ بإشارة النسبة المئوية (%). يظهر هذا السطر عند كتابة أمر التعليمات help function_name و يجب أن يحوي السطر H1 على ملخص عن محتويات الملف M لأنه يظهر لوحده عند استعمال أمر البحث عن الملفات look for file_name و السطر H1 من أجل الإجراء fact هو:

```
% Fact factorial
```

نص التعليمات: يمكن توليد نص تعليمات (يظهر عند استعمال أمر help file_name) بواسطة إدخال نص يتألف من سطر واحد أو عدة أسطر تبدأ بإشارة النسبة المئوية % و يبدأ هذا النص دائماً من السطر الذي يلي السطر H1 و نص التعليمات في الإجراء fact هو:

```
% Fact(n) Returns the factorial of number n
% Write simply fact(n)
```

عند طلب التعليمات يظهر MATLAB الملاحظات الموجودة في سطر تعريف التابع و بين أول أمر تنفيذي و يهمل أية ملاحظات نأتي بعد ذلك.

```
» help fact

 Fact Factorial
 Fact(n) Returns the factorial of number n
 Write simply fact(n)
```

جسم الإجراء : يحتوي على شفرة MATLAB اللازمة لإنجاز الحسابات و إسناد القيم الناتجة عن هذه الحسابات إلى متحولات الخرج.

العبارات المستخدمة في جسم الأجراء يمكن أن يحتوي على كل أوامر استدعاء لإجراءات أخرى أو برامج أو حسابات أو أوامر دخل و خرج أو ملاحظات أو أسطر فارغة. في مثالنا هذا جسم الإجراء هو:

```
f=prod(1:n); % Actual computation
```

ملاحظات : كما هو مذكور سابقاً يمكن إدراج أية أسطر ملاحظات في الملف M على شرط أن يبدأ سطر الملاحظات بإشارة النسبة المئوية.

مثالنا التالي لإجراء مع استعمال عدة متحولات للدخل و الخرج:

```
function [x1,x2]=equation(a,b,c)
% Solve Equations
% used for solve equations of the form ax^2+bx+c=0
d=(b^12+4*a*c)/(2*a);
x1=(-b+sqrt(a))/(2*a);
```

```
x2=(-b-sqrt(a))/(2*a);
```

يتم استدعاء (على سبيل المثال) كما يلي:

```
» [L,M] = equalion(5,3,8)
L =
  -0.0764
M =
  -0.5236
```

أسماء الإجراءات ...

تخضع أسماء الإجراءات إلى نفس القواعد التي تخضع لها الأسماء في MATLAB أي أن MATLAB يستعمل الرموز الـ 31 الأولى من الاسم و يجب أن يبدأ الاسم بحرف.

اسم ملف النص الذي يحتوي إجراءات MATLAB يتألف من اسم الإجراء و ينتهي بالامتداد m.* أي fact.m مثلاً. إذا كان اسم الملف النص و الاسم الذي عرّف به الإجراء مختلفين فإن MATLAB يهمل الاسم الذي عرف به الإجراء و يجري استدعاء الإجراء بواسطة اسم الملف النص. على سبيل المثال إذا حفظنا الإجراء fact باسمٍ ثانٍ مثل factorial.m فإنه عند استدعاء البرنامج يجب استخدام الاسم factorial و ليس fact الذي هو الاسم الذي عرف به الإجراء أي أن الاسم الذي يحفظ به الإجراء و ليس هو بالضرورة الاسم الذي يعرف به الإجراء.

عند استدعاء إجراء يقوم MATLAB بالخطوات التالية:

1. يفحص فيما إذا كان الاسم هو اسم متحول.
2. يفحص إذا كان الاسم هو اسم إجراء فرعي Sub Function.
3. يفحص إذا كان الاسم هو اسم تابع محلي Private Function.
4. يفحص إذا كان الاسم موجوداً على قائمة الأسماء الموجودة في مسار برنامج MATLAB .

يستعمل MATLAB الملف الأول الذي يصادفه بنفس الاسم و إذا تكررت الأسماء يقوم MATLAB بإنجاز الإجراء الذي يجده أولاً باستخدام القواعد المذكورة سابقاً. عند استدعاء ملف الإجراء M من نافذة الأوامر أو من ملف M آخر يحول MATLAB الإجراء إلى شفرة خاصة به و يحفظه في الذاكرة و هذا ما يوفر على MATLAB إجراء عملية التحويل عند كل استدعاء لهدا الإجراء و يتم الاحتفاظ بهذه الشفرة في الذاكرة حتى يتم مسحها بواسطة أمر **clear**.

توليد ملفات شفرة P – ...

يمكن حفظ شفرة الإجراءات المحولة إلى شفرة MATLAB الخاصة ضمن ملفات P حيث يمكن لـ MATLAB استعمالها مباشرة بدون إجراء عملية التحويل فعلى سبيل المثال عند كتابة fact **pcode** يقوم MATLAB بتحويل الإجراء fact إلى شفرة MATLAB و يحفظه في ملف بامتداد p.* أي fact.p.

ساحة عمل الإجراءات ...

كل إجراء يحجز مساحة من الذاكرة منفصلة عن مساحة عمل MATLAB و تدعى ساحة العمل هذه بساحة عمل الإجراء.

فحص عدد متحولات الإجراء ...

بواسطة التوابع **nargin** و **nargout** يمكن تحديد عدد متحولات الدخل و الخرج عند استدعاء الاجراء على سبيل المثال:

```
function c=testarg(a,b)
if (nargin==1);
  c=a^2;
elseif (nargin==2);
  c=a+b;
end
```

عند إعطاء متحول دخل واحد يقوم الإجراء بتربيعه و عند إعطاء متحولي دخل يقوم التابع بجمعهما، لاحظ ...

```
» testarg(1)
ans =
    1
» testarg(10)
ans =
  100
» testarg(10,10)
ans =
   20
```

إدخال عدد متغير من متحولات الدخل و الخرج ...

يمكن إدخال عدد متغير من متحولات الدخل أو الخرج إلى الإجراء باستخدام التابعين **varargin** و **varargout** يسمح هذين التابعين بإرسال عدد متغير من متحولات الدخل أو إرجاع عدد متغير من متحولات الخرج. يضع MATLAB جميع متحولات الدخل و الخرج ضمن مصفوفة خلايا Cell array و التي هي عبارة عن نوع خاص من مصفوفات MATLAB حيث تحتوي المصفوفات على خلايا بدلاً من العناصر و كل خلية Cell يمكن أن تحتوي على أي حجم أو نوع من المعطيات و يمكن أن تحتوي أحد خلايا المصفوفة على شعاع يحتوي على معطيات رقمية بينما تحتوي خلية أخرى على مصفوفة من المعطيات الحرفية و هكذا ...

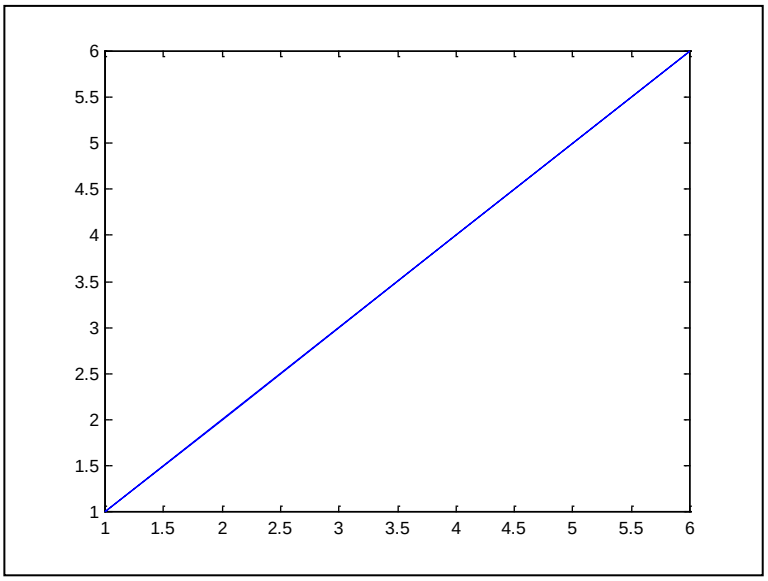
و فيما يلي نعطي مثال على إعطاء عدد متغير من متحولات الدخل :

```
function testvar(varargin)
for i =1: length(varargin)
  x(i)=varargin{i}(1);
  y(i)=varargin{i}(1);
end
xmin=min(0,min(x));
ymin=min(0,min(y));
axis([xmin fix(max(x))+3 , ymin fix(max(y))+3])
plot(x,y)
```

يمكن استعمال عدد من متحولات الدخل باستدعاء ...

```
» testvar([2 3],[1 5],[4 8],[6 5],[4 2],[2 3])
```

سيعطي البرنامج الشكل التالي:

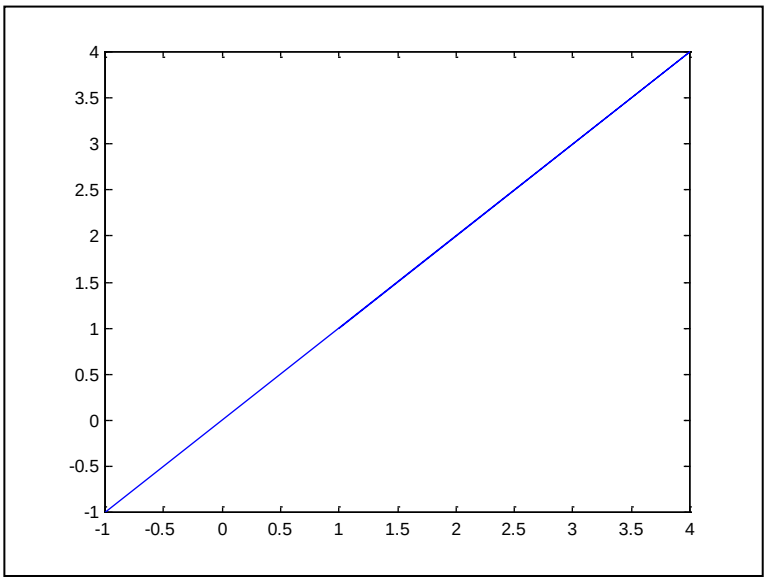


الشكل (2-4)

أو :

```
» testvar([-1 0],[2 -5],[4 2],[1 1])
```

سينتج الشكل التالي (لاحظ حدود المحورين x, y):



المتحولات المحلية و العامة ...

إن قواعد تعريف التوابع هو يعتمد على ملفات M. يحتوي كل إجراء في MATLAB معين بواسطة ملف M على متحولاته المحلية الخاصة به و التي هي مستقلة و منفصلة عن بقية المتحولات المحلية للإجراءات الأخرى و لكن إذا صرحنا عن المتحولات كتوابع عامة Global فإن المتحول المصرح عنه بهذا الشكل سيصبح مشترك بين جميع الإجراءات و يتم التصريح عن المتحول ضمن أحد ملفات الإجراء.

التحكم بمسار البرنامج ...

سنستعرض عبارات التحكم بمسار البرنامج في MATLAB و التي تشبه إلى حد كبير عبارات التحكم في لغات البرمجة الأخرى.

1. أمر التحكم الشرطي **IF** : شكله العام

```
IF logical expression
   statements
END
```

في الحالة الأولى إذا كانت العلاقة المنطقية صحيحة فإن MATLAB ينفذ جميع العبارات الموجودة بين if و end و يتابع تنفيذ البرنامج من السطر الذي يتبع end أما إذا كان الشرط خاطئ (0) فإن MATLAB يتجاوز تنفيذ جميع العبارات الواقعة بين if و end و يتابع التنفيذ من السطر الذي يتبع end. مثال ...

```
if rem(4,2)==0
  disp('a is eren');
  b=a/2
end
```

يمكن ان يحتوي الشرط على مصفوفات و ليس على أعداد فمثلاً إذا كان

```
IF x
   statements
END
```

فهذا الشرط يتحقق إذا كانت جميع عناصر x > 0

يمكن استعمال **else**, **elseif** كشروط إضافية للشرط if.

```
IF logical expression
   statements
ELSEIF
   statements
ELSE
   statements
END
```

عبارة else لا تحتوي على شرط منطقي و لكن يجري تنفيذ العبارات المرتبطة بعبارة else إذا كانت نتيجة الشرط المنطقي في عبارة if (أو عبارة elseif) خطأ أو مساوي للصفر. عبارة elseif تحتوي على شرط منطقي يتم تنفيذه إذا كان الشرط المنطقي عبارة if خطأ أو تساوي الصفر و يتم تنفيذ جميع العبارات المرتبطة بها إذا كانت نتيجة الشرط المنطقي في عبارة elseif تساوي الواحد (لا تساوي الصفر). مثال ...

```
if n<0
  disp('input must be positive');
elseif rem(n,2)==0
  A=n/2;
else
  A=(n+1)/2;
end
```

```
if n<0
  disp('input must be positive');
```

إذا كان n أصغر من الصفر أطبع العبارة

```
elseif rem(n,2)==0
  A=n/2;
```

إذا كان n موجباً و زوجياً قم بالعملية A=n/2

```
else
  A=(n+1)/2;
end
```

إذا كان موجباً و فردياً قم بالعملية A=(n+1)/2

عبارة switch ...

تنفذ عبارة switch مجموعة من العبارات عند قيم معينة للمتحول المراد اختباره و الشكل العام للعبارة:

```
SWITCH expression
CASE value1
    statements
CASE value2
    statements
OTHERWISE
    statements
END
```

تتألف العبارة من كلمة switch متبوعة بالعلاقة المراد اختبارها. تنفذ العلاقة و تقارن النتيجة مع القيم التي تتبع عبارة case و عند حدوث أول تطابق بين نتيجة العلاقة و بين القيمة الموجودة على يمين عبارة case يتم تنفيذ التي تطابقت القيمة الموجودة على يمينها مع نتيجة العلاقة.

ملاحظة: لا يتابع MATLAB المقارنة بعد حدوث أول تطابق.

إذا لم يحدث أي تطابق مع القيم الموجودة على يمين عبارة case و نتيجة العلاقة المراد اختيارها يتم تنفيذ العبارات التي تلي otherwise و عبارة otherwise اختيارية. يجب إنهاء عبارة switch بـ end. مثال ...

```
switch input_num
case -1
  disp('negative');
case 0
  disp('zero');
case
  disp('positive')
otherwise
  disp('other value')
end
```

حلقة while ...

تنفذ حلقة while عبارة أو مجموعة من العبارات بشكل متكرر طالما أن قيمة عبارة التحكم تساوي الواحد.

WHILE expression (عبارة التحكم)

statements

END

إذا كانت العلاقة مصفوفية فإن جميع عناصرها يجب أن تكون مساوية الواحد ليستمر الحساب.

مثال: في هذا البرنامج سيستمر تكرار العبارات بين العبارة while و end حتى يصبح الشرط خطأ أو مساوياً للصفر.

```
b=10
while b>0
  b=b-1
end
```

حلقة For ...

تقوم حلقة for بتكرار عبارات معينة لعدد محدد من المرات.

FOR variable = expression

Statements

END

```
x=[5 4 3 2 5 7 8]
for t=2:6;
  x(t)=2*x(t-1)
end
```

بشكل افتراضي تكون الخطوة مساوية للواحد إذا لم تذكر. بتنفيذ البرنامج و بفرض أسميناه forring نجد:

```
» forring
x =
   5   4   3   2   5   7   8
x =
   5  10   3   2   5   7   8
x =
   5  10  20   2   5   7   8
x =
   5  10  20  40   5   7   8
x =
   5  10  20  40  80   7   8
x =
   5  10  20  40  80 160   8
```

يمكن استعمال المصفوفات مكان علاقة التحكم فعلى سبيل المثال إذا كانت A هي مصفوفة أبعادها $m \times n$ فإن:

FOR i = A
  Statements
END

سوف تعطي i=(:,k) حيث k=1,2,…,n

عبارة الإيقاف Break ...

تقوم هذه العبارة بإيقاف تنفيذ حلقة for أو حلقة while عند وضعها ضمن الحلقة، فمثلاً ...

```
for i=1:100
  e=i^2
  if e>=10e3,disp(e),break,end
end
```

العبارة try catch ...

الشكل العام لهذه العبارة

TRY statement, ..., statement, **CATCH** statement, ..., statement, **END**

يتم بشكل طبيعي تنفيذ العبارات بين try و catch و لكن عند حدوث أي خطأ في تنفيذ أي عبارة من العبارات عند ذلك يتم الانتقال إلى تنفيذ العبارات الواقعة بين catch و end و عند حصول خطأ أيضاً من أحد العبارات يوقف MATLAB تنفيذ الأوامر و يضع عبارة الخطأ في متحول اسمه lasterr. مثال:

```
try
  m=10
  y=n
catch
  disp('error')
end
```

بحفظ الملف باسم trycatch.m و استدعائه من نافذة MATLAB نحصل على التالي:

```
» trycatch
m =
   10
error
```

عبارة return ...

تنهي return تسلسل تنفيذ الأوامر في الإجراء و تحول القيادة إلى البرنامج الرئيسي. بشكل عام عندما تنتهي أوامر الإجراء فإن القيادة تنتقل بشكل آلي إلى البرنامج الرئيسي و لكن يمكن وضع عبارة return في أي مكان من الإجراء لإعادة القيادة قسرياً إلى البرنامج الرئيسي.

الإجراءات الفرعية SubFunctions ...

يمكن أن يحتوي ملف m على شفرة أكثر من إجراء Function واحد يسمى أول إجراء في الملف الإجراء الأولي و تسمى الإجراءات اللاحقة و التي يتم استدعاؤها عن طريق الإجراء الأولي بالإجراءات الفرعية و هذه الإجراءات مرئية فقط ضمن الإجراء الأولي (أي لا يمكن استدعاؤها خارج هذا الإجراء).

لكل إجراء فرعي سطر التعريف الخاص به و تتبع الإجراءات الفرعية بعضها أي ترتب الإجراء تلو الآخر و لكن على شرط أن يكون الإجراء الأولي في البداية. مثال لحساب جذور المعادلة من الدرجة الثانية:

```
function [x1,x2]=equa(a,b,c)
x1=(-b+delta(a,b,c))/2*a;
x2=(-b-delta(a,b,c))/2*a;
function d=delta(l,m,n)
d=sqrt(m^2-4*1*m)
```

لاستدعاء هذا الإجراء عند قيم a=1, b=2, c=3 نكتب:

```
» [o,p]=equa(1,2,3)
d =
      0 + 2.0000i
d =
      0 + 2.0000i
o =
  -1.0000 + 1.0000i
p =
  -1.0000 - 1.0000i
```

أي أنه لحساب جذور المعادلة تم استدعاء البرنامج الفرعي delta مرتين.

حساب السلاسل الحرفية ...

تضيف إمكانية حساب السلاسل الحرفية قوة و مرونة إلى لغة MATLAB. لنستعرض بعض التوابـــــع.

التابع eval وظيفته حساب سلسلة الحروف ضمن أي علاقة و الشكل العام لهذا التابع هو:

EVAL ('string')

[x,y,z,…]=eval(s)

مثال:

```
t=('i^2+2')
for i =1:10
```