الفصل الثالث

حل المعادلات غير الخطية على MATLAB

في هذا الفصل يتم عرض إحدى المسائل الأساسية في الرياضيات، وهي إيجاد الحل العددي لمعادلة غير خطية. معظم المعادلات التي تظهر في التجارب العلمية أو في الطبيعة هي معادلات غير خطية، وقد تكون المعادلات في متغير واحد أو أكثر، وقد تكون معادلة واحدة أو نظاماً من المعادلات. الطرق العددية تقرّب حل المعادلات غير الخطية بطريقة تكرارية، ويكون التقارب في بعضها مضموناً، وفي بعضها الآخر مشروطاً.

إذا فرضنا أن $f(x)$ دالة متصلة، فإن العدد α بحيث إن $f(\alpha) = 0$ يسمى جذراً أو صفراً للمعادلة $f(x) = 0$. و يمكن أن يكون الجذر حقيقياً أو مركباً، كما يمكن أن يوجد أكثر من جذر. نقدم في معالجاتنا في هذا الفصل طرقاً عددية على MATLAB للوصول للحلول ذات قيم حقيقية .

(٣,١) طريقة التنصيف Bisection Method

طريقة التنصيف Bisection method هي من أبسط الطرق العددية التكرارية لإيجاد جذر معادلة، وتحتاج إلى نقطتين ابتدائيتين. تعتمد أساساً على نظرية القيمة

المتوسطة وهي التي تضمن وجود جذر في الفترة التي تتغير فيها إشارة الدالة، بتكرار تنصيف الفترات التي تحتوي على الجذر يتم التقريب للجذر.

لتكن $f(x)$ دالة متصلة على الفترة $[a,b]$ بحيث إن $f(a)$ و $f(b)$ مختلفتان في الإشارة، أي $f(a)\,f(b) < 0$ فإنه يوجد على الأقل عدد $c \in [a,b]$ بحيث $f(c) = 0$. نأخذ في البداية الفترة $[a_1,b_1]$ بحيث إن $f(a_1)$ و $f(b_1)$ مختلفتان في الإشارة، ثم نحسب $c_1=(b_1+a_1)/2$ ونحسب $f(c_1)$. بعدها نقوم باختيار الفترة $[a_1,c]$ أو $[c,\ b_1]$ بحيث تكون $f(a_1)$ و $f(c_1)$ أو $f(b_1)$ و $f(c_1)$ مختلفتين في الإشارة. نكرر العملية حتى التوصل للجذر أو لجذر بالتقريب المطلوب. نحاول حصر جذر واحد فقط في الفترة المنصفة. وإذا كانت الفترة التي تحتوي على الجذر غير معلومة، نستطيع الاستعانة بالرسم *plot* لتحديد تلك الفترة. نخزن في MATLAB m-file والذي يحتوي على الخوارزمية (٣,١) *bisect* [7]، ويتم استخدام البرنامج بمناداته مع تحديد المعادلة، فترة التنصيف والدقة المطلوبة.

مثال رقم (٣,١)

أوجد جذر المعادلة $x^3-2x-1=0$ في الفترة [1.5, 2] بطريقة التنصيف.

الحل:

نخزن المعادلة في m-file يدعى fun :

```
function f=fun(x)
f=x.^3-2*x+ -1;
```

```
function solution=bisect(fun,a,b,acc)
fa=feval(fun,a);
fb=feval(fun,b);
while abs(b-a)>acc
   c=(a+b)/2;
   fc=feval(fun,c);
   if fa*fc<=0;
      b=c;
   else a=c;
   end
end
solution=(a+b)/2;
```

خوارزمية (٣,١)

ثم ندخل في نافذة الأوامر :

```
>>bisect('fun',1.5,2,1e-2)

solution =
   1.6211
```

وتظهر النتيجة بقيمة الجذر. طريقة التنصيف بسيطة و دائمة التقارب ولكن من عيوبها البطء في الوصول للجذر.

(٣,٢) طريقة نيوتن Newton Method

طريقة نيوتن Newton method هي من أشهر وأقوى الطرق التكرارية لإيجاد جذور المعادلة $f(x) = 0$. وتحتاج طريقة نيوتن لفرض نقطة بداية x_0 وتولّد مع التكرار المتتالية x_n التي تتقارب إلى الجذر الفعلي. إذا كانت النقطة الابتدائية قريبةً بما فيه الكفاية من الجذر p، والجذر بسيطاً، (جذر غير مكرر) أي أن تكون مشتقة الدالة عند الجذر p قيمة غير صفرية $(f'(p) \neq 0)$ فإن طريقة نيوتن تتقارب بسرعة. تحتاج الطريقة

لقيم الدالة $f(x)$ و المشتقة الأولى $f'(x)$ ، وتعتمد هندسياً على المماس لـ $f(x)$. تدعى الطريقة أيضاً بطريقة نيوتن– رافسون **Newton-Raphson method**.

إذا كانت $f(x)$ دالة متصلة وقابلة للاشـــــتقاق على $[a,b]$ و $f'(x) \neq 0$ لكل $x \in [a,b]$ ، فإن طريقة نيوتن تولّد متتالية $\{x_n\}$ المعرّفة بالتالي :

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

نبرمج الطريقة باسم *newton* وتخزنها في **m-file** [15] (خوارزمية ٣,٢) ويتطلب الأمر تحديد الدالة، والمشتقة، ونقطة البداية ويمكن تحديد قيمة الدقة المطلوبة tol بحيث يتم التوقف عن حساب قيم جديدة في المتتالية x_n إذا :

$$\left| \frac{f(x_n)}{f'(x_n)} \right| < \text{tol}$$

```
function[r,it]=newton(fun,dfun,x,acc)
it=0;
x0=x;
d=feval(fun,x0)/feval(dfun,x0);
while abs(d)>acc
   x1=x0-d;
   it=it+1;
   x0=x1;
   d=feval(fun,x0)/feval(dfun,x0);
end;
r=x0;
```

خوارزمية (٣,٢) .

مثال رقم (٣,٢)

استخدم البرنامج newton المعطى في خوارزمية (٣,٢) لإيجاد جذر المعادلة $x^3-10x^2+29x-20=0$ بنقطة البداية $x_0=7$.

الحل :

نخزن الدالة و المشتقة في m-files باسم *fun* و *dfun* :

```
function f=fun(x)
f=x.^3-10*x.^2+29*x-20;
```

```
function f=dfun(x)
f=3*x.^2-20*x+29;
```

نطبّق الأمر newton مع تحديد الدالة ، والمشتقة ، و نقطة البداية ، والدقة المطلوبة ، ليعطي نتيجة الحل مع عدد خطوات التكرار :

```
>>[r,it]=newton('fun','dfun',7,.00005)
r =
   5.0000
it =
    6
```

ويمكن تعديل الطريقة إذا كان هناك صعوبة في المشتقة بالتعويض عن المشتقة بقيمة الميل ، وتصبح :

$$x_n = x_{n-1} - \frac{(x_{n-1} - x_{n-2})f(x_{n-1})}{f(x_{n-1}) - f(x_{n-2})} \quad .$$

تسمى هذه الطريقة بطريقة القاطع Secant method وتحتاج إلى نقطتي بداية ،

وتوجد نسخة من البرنامج في الملحق [7] ، عند استخدام برنامج القاطع *secant* على نفس المثال و بنقطتي بداية 4 و 4.5 وبنفس الدقة نحصل على :

```
>>sol=secant('fun',4,4.5,.00005)
sol=
    5.0000
it=
   8
```

التقارب بهذه الطريقة أبطأ من طريقة نيوتن ، و كلا الطريقتين بطيئة في حال كانت الجذور قريبة جداً من بعضها أو إذا كانت جذوراً مضاعفة (تتكرر أكثر من مرة واحدة كجذر للمعادلة). إذا كانت الجذور مضاعفة فيمكن معالجة الأمر وتسريع التقارب باستخدام طريقة نيوتن المعدلة Modified Newton method ومعادلتها :

$$x_{n+1} = x_n - \frac{f(x_n)f'(x_n)}{f'(x_n)^2 - f(x_n)f''(x_n)}$$

العائق الوحيد في طريقة نيوتن المعدلة هو حساب المشتقة الثانية $f''(x)$ وازدياد كمية العمليات الحسابية. يستطيع القارئ استخدام البرنامج الموجود في الملحق [7] باسم *modifiedNewton* :

مثال رقم (٣,٣)

احسب جذر المعادلة :

$$f(x) = (x-1)^2 \ln(x)$$

نلاحظ أن الجذر p=1 مكرر، فنطبق برنامج طريقة نيوتن المعدلة على

المعادلـة، وهـو يحتـاج إلى إدخـال الدالـة، والمـشتقة الأولى، والمـشتقة الثانيـة، ونقطـة البداية، و الدقة المطلوبة.

```
>> [sol,it]=modifiedNewton('fun','dfun','ddfun',.9,.00005)
sol =
   1.0000
it =
    3
```

للمقارنـة، تـستخدم برنـامج نيـوتن علـى نفـس الدالـة، وبـنفس نقطـة البدايـة والدقـة. نلاحـظ أن طريقـة نيـوتن احتاجـت إلى 17 خطـوة ولم تـصل إلى نتيجـة دقيقـة. وبـذلك نـستنتج أن طريقـة نيـوتن تحتـاج إلى خطـوات أكثـر لتـصل للجـذر بـسبب كـون الجذر مكرراً، بينما طريقة نيوتن المعدلة احتاجت فقط ثلاث خطوات:

```
>> [sol,it]=newton('fun','dfun',.9,.00005)
sol =
   0.9999
it =
   17
```

(٣,٣) إيجاد جذور معادلات باستخدام دوال جاهزة في MATLAB

يقـدم MATLAB دوال جـاهزة مختلفـة تـساعد في إيجـاد حلـول لمعـادلات خطيـة وغير خطية منها:

(٣,٣,١) دالة fzero

الدالة الجاهزة *fzero* تقوم بحساب جذور المعادلات، وهي عبارة عن توليفة من الطرق العددية التي تعطي نتائج مضمونة. ويتم استخدام الأمر بعد تعريف الدالـة f

في *inline* أو m-file ثم تحديد الفترة التي تحتوي على الجذر والدقة المطلوبة ، والتي يمكن الحصول عليها برسم الدالة بالأمر *plot* :

```
>> f=inline('x^3-2*x^2-x+2')
f =
   Inline function:
   f(x) = x^3-2*x^2-x+2
>>sol=fzero(f,[0 1.5], 1e-15)
sol =
    1
```

إذا لم يكن للدالة جذرٌ فعلي محدد جبرياً مثل الدالة *cos(x)-x* فيمكن أن نحصل على تقريب للجذر بالأمر *fzero* :

```
>> function z=f(x)
   z = cos(x)-x;
>> format long
>> sol=fzero('f',[0 2],1e-15)
sol =
  0.739085133215l6
```

يمكن استخدام الأمر *fzero* بإدخال الدالة ونقطة البداية فقط ، ولكن قد لا يصل للجذر خاصة إذا كان الجذر قريباً من نقطة غير معرفة بالنسبة للدالة. وإذا لم يتم تحديد الدقة المطلوبة فإن MATLAB يحسب بالدقة $2x10^{-16}$.

```
>> f=inline('sin(x)-.5*x')
f =
   Inline function:
   f(x) = sin(x)-.5*x
>> sol=fzero(f,1)
sol =
  1.89549426703398
```

(٣,٣,٢) دالة roots

توجد دالة أخرى جاهزة في MATLAB تستخدم لإيجاد أصفار كثيرات الحدود تدعى *roots (c)* ، حيث إن المتجه *c* هو متجه معاملات كثيرة الحدود.

مثال رقم (٣,٤)

اوجد جـــــذور $x^3-2x^2-x+2=0$ بالدالة *roots* حيـــث متجه المعامـــلات : c=[1 -2 -1 2]

```
>> roots([1 -2 -1 2])
ans =
  - 1.0000
    2.0000
    1.0000
```

(٣,٣,٣) دالة solve

نستطيع استخدام دالة solve في الجبر الرمزي *Symbolic algebra* لحل المعادلة السابقة (كثيرة الحدود) كالآتي :

```
>> syms x
>> sol=solve(x^3-2*x^2-x+2)
 sol =
  -1
   1
   2
```

أو لتقريب حل للمعادلة غير الخطية $cos(x)-x=0$:

```
>> syms x
>> sol=solve(cos(x)-x)
 sol =
0.73908513321516
```

(٣,٤) حل نظام معادلات غير الخطية

إذا كان لدينا نظام من المعادلات غير الخطية فيمكن حلها باستخدام طريقة تعتمد على طريقة نيوتن. نبدأ بفرض وجود معادلتين غير خطيتين في متغيرين :

$$f_1(x,y)=0$$
$$f_2(x,y)=0$$

بحيث إن $f_1(x,y)$ و $f_2(x,y)$ دوال متصلة وتمثل النظام بالمعادلة $\mathbf{F(x)=0}$ حيث إن $F(x,y)=(f_1(x,y), f_2(x,y))^t$، و نبحث عن الحل $(x,y)^t$ الذي يحقق المعادلتين.

نحتاج في طريقة نيوتن لحل هذا النظام إلى المشتقة باستخدام المصفوفة الجاكوبية Jacobian matrix :

$$J(x)=\begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{pmatrix}$$

لتصبح صيغة المعادلة التكرارية لحل النظام بطريقة نيوتن و بنقطة بداية $x^{(0)}$ مناسبة لكل $n \geq 1$ هي :

$$x^{(n)}=x^{(n-1)}-J(x^{(n-1)})^{-1}F(x^{(n-1)})$$

ولتفادي حساب معكوس مصفوفة الجاكوبية فنستخدم :

$$J(x^{(n-1)})h^{(n-1)}=-F(x^{(n-1)})$$
$$x^{(n)}=x^{(n-1)}+h^{(n-1)}$$

وهي الصيغة التكرارية المستخدمة في الخوارزمية (٣,٣) لطريقة نيوتن لنظام معادلات غير خطية التي تدعى *newton2* [15] . وعند استخدامها تتم المناداة بالمدخلات: نقطة البداية المناسبة (x,y)، والمعادلات، والمشتقات، وعدد المتغيرات، والدقة المطلوبة.

```
function [xx,it]=newton2(x,f,jf,n,tol)
it=0;
xx=x;
fr=feval(f,xx);
while norm(fr)>tol
   jr=feval(jf,xx);
   xx1=xx-jr\fr;
xx=xx1;
   fr=feval(f,xx);
   it=it+1;
end
```

خوارزمية (٣,٣)

مثال رقم (٣,٥)

في نظام معادلات غير خطية كالآتي:

$$x^2 + y^2 = 16$$
$$xy = 1$$

الحل يمثل بنقاط التقاطع المبينة في الشكل رقم (٣,١) الذي حصلنا عليه بالأوامر التالية:

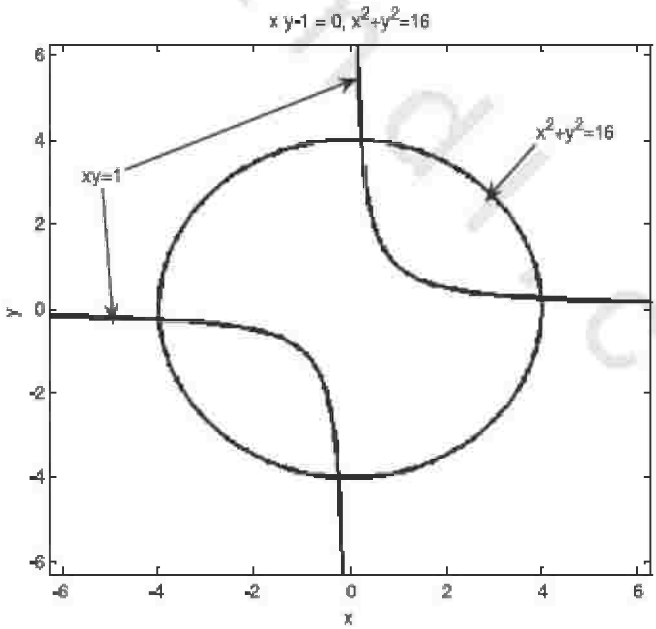
```
>> ezplot('x^2 + y^2 - 16')
>> hold
Current plot held
>> ezplot('x*y-1')
```

نخزن المعادلات في m-file يدعى *f1* :

```
function f=f1(v)
x=v(1);y=v(2);
f=zeros(2,1);
f(1)=x^2+y^2-16;
f(2)=x*y-1;
```

والمشتقة الجاكوبية في m-file آخر يدعى *f2* :

```
function jf=f2(v)
 x=v(1);y=v(2);
jf=zeros(2,2);
 jf(1,:)=[x*2 y*2];
 jf(2,:)=[y x];
```



الشكل رقم (٣,١). نقاط التقاطع لنظام المعادلات غير خطية (مثال رقم ٣,٥).

نستخدم برنامج newton 2 مع تحديد نقطة البداية المناسبة (x,y)، المعادلات من ملف f1 ، المشتقات من ملف f2 ، عدد المتغيرات 2 ، والدقة 0.00005 :

```
>> [sol1,iter]=newton2([4 1]', 'f1','f2',2,0.00005)
sol1 =
   3.9921
   0.2505
iter =
    3
```

ونحصل على أحد الجذور (3.9921 ,0.2505) بعد ثلاثة تكرارات. وباختيار نقاط بداية مختلفة نحصل عليها من الرسم ونستطيع الحصول على باقي الجذور الثلاثة :

```
>> [sol2,iter]=newton2([.5 3]', 'f1','f2',2,0.00005)
Sol2 =
   0.2505
   3.9921
iter =
    4

>> [sol3,iter]=newton2([-4 0]', 'f1','f2',2,0.00005)
Sol3 =
  -3.9921
  -0.2505
iter =
    3
>> [sol4,iter]=newton2([0 -4]', 'f1','f2',2,0.00005)
Sol4 =
  -0.2505
  -3.9921
iter =
    3
```

مثال رقم (٣,٦)

أوجد حلاً للنظام غير الخطي التالي :

$$\sin x + y^2 + \ln z = 7$$
$$3x + 2y - z^3 = -1$$
$$x + y + z = 5$$

الحل :

نخزن المعادلات في m-file يدعى *f101* و المشتقة الجاكوبية في m-file آخر يدعى *f201* :

```
function q=f101(p)
x=p(1);y=p(2);z=p(3);
q=zeros(3,1);
q(1)=sin(x)+y^2+log(z)-7;
q(2)=x*3+2^y-z^3+1;
q(3)=x+y+z-5;
```

```
function jq=f201(p)
x=p(1);y=p(2);z=p(3);
jq=zeros(3,3);
jq(1,:)=[cos(x) 2*y 1/z];
jq(2,:)=[3 (2^y )*log(2) -3*(z^2)];
jq(3,:)=[1 1 1 ];
```

نستخدم برنامج newton 2 مع تحديد نقطة البداية المناسبة (0,2,2)، وعدد المتغيرات ثلاثة، والدقة 0.001 . لنحصل على الجذر(*2.005 , 2.3959 , 0.5991*) وبعد ثلاثة تكرارات :

```
[x,it]=newton2([0,2,2]','f101','f201',3,.001)
x =
   0.5991
   2.3959
   2.0050
it =
    3
```

(٣,٥) تمارين

١- استخدم طريقة التنصيف لإيجاد جذر المعادلة e^x-2-x على الفترة [-3, -2] وبالدقة 10^{-4} .

٢- استخدم طريقة التنصيف لإيجاد جذر المعادلات بالدقة 10^{-4} :

أ) $x-2^{-x}=0 \quad 0 \le x \le 1$

ب) $x^4-2x^3-4x^2+4x+4=0 \quad -1 \le x \le 0\ ,\ 2 \le x \le 3,\ 0 \le x \le 2,\ -2 \le x \le -1$

٣- استخدم طريقة نيوتن لإيجاد جذور المعادلات في التمرين رقم ٢.

٤- استخدم طريقة نيوتن لإيجاد جذر المعادلة x^3-3x+1 على الفترة [1, 3] ونقطة بداية $x_0=1.5$.

٥- أوجد قيمة تقريبية بطريقة نيوتن للجذر $\sqrt{7}$ بنقطة بداية 2.5.

٦- أوجد نقطة التقاطع للدالتين $sin(x)\ ,\ e^{-x}$ في [0,1].

٧- استخدم طريقة رباعية التقارب لإيجاد الجذر $x=0$ للمعادلة $e^x-x-1=0$.

٨- استخدم طريقة القاطع secant method لإيجاد الجذر للمعادلة $f(x)=-x^3-cos(x)$ واستخدم طريقة التنصيف لإيجاد نقطتي البداية.

٩- استخدم طريقة القاطع secant method لإيجاد الجذر الموجب للمعادلة $x^{10}-1=0$ بنقطتي بداية $x_0=1.2\ x_1=1.1$.

١٠- استخدم طريقة نيوتن المعدلة لإيجاد جذر المعادلة المكرر $x\ lnx - lnx$ عند $p=1$.

١١- حل النظام التالي بطريقة نيوتن بنقطة بداية (1,1) و بدقة 10^{-7} :

$$4x^3+y=6$$
$$x^2y=1$$

١٢- استخدم طريقة نيوتن بنقطة بداية $x^{(0)}=0$ لإيجاد $x^{(2)}$ للنظام غير الخطي :

$$4x_1^2 - 20x_1 + \frac{1}{4}x_2^2 + 8 = 0$$
$$\frac{1}{2}x_1x_2^2 + 2x_1 - 5x_2 + 8 = 0$$

١٣- أوجد نقطتي التقاطع بين المعادلتين وارسم للتأكد من موقع التقاطع بدقة 10^{-7}.

$$-x_1(x_1+1) + 2x_2 = 18$$
$$(x_1 - 1)^2 + (x_2^2 - 6)^2 = 25$$