

## برامج MATLAB

### خوارزميات إضافية تم استخدامها في الكتاب

```

*****
function s=bisect(fn,a,b,acc)
fa=feval(fn,a);
fb=feval(fn,b);
if fa*fb>0
    fprintf('endpoints are not of different sign ')
end
while abs(b-a)>acc
    c=(a+b)/2;
    fc=feval(fn,c);
    if fa*fc<=0;
        b=c;
    else a=c;
    end
end
s=(a+b)/2;
*****
function [tvals yvals]=abm(f,start,finish,startval,h)
%Adams Bashfoth Moulton method
%set up matrices for Runge-Kutta methods
b=[ ];c=[ ];d=[ ]; order=4;
b=[ 1/6 1/3 1/3 1/6]; d=[0 .5 .5 1];
c=[0 0 0 0;0.5 0 0 0;0 .5 0 0;0 0 1 0];
s=(finish-start)/h+1;
y=startval; t=start; fval(1)=feval(f,t,y);
ys(1)=startval; yvals=startval;tvals=start;
for j=2:4
    k(1)=h*feval(f,t,y);
    for i=2:order
        k(i)=h*feval(f,t+h*d(i),y+c(i,1:i-1)*k(1:i-1)');
    end;
    y1=y+b*k'; ys(j)=y1; t1=t+h;
    fval(j)=feval(f,t1,y1);
    %collect values together for output
    tvals=[tvals,t1]; yvals=[yvals,y1];
    t=t1; y=y1;
end;
end;

```

```

for i=5:s
    yl=ys(4)+h*(55*fval(4)-59*fval(3)+37*fval(2)-9*fval(1))/24;
    t1=t+h; fval(5)=feval(f,t1,y1);
    yc=ys(4)+h*(9*fval(5)+19*fval(4)-5*fval(3)+fval(2))/24;
    fval(5)=feval(f,t1,yc);
    fval(1:4)=fval(2:5);
    ys(4)=yc;
    tvals=[tvals,t1]; yvals=[yvals,yc];
    t=t1; y=y1;
end;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function q = diffgen(func,n,x,h)
if (n==1) | (n==2) | (n==3) | (n==4)
    c=zeros(4,7);
    c(1,:)=[ 0 1 -8 0 8 -1 0];
    c(2,:)=[ 0 -1 16 -30 16 -1 0];
    c(3,:)=[1.5 -12 19.5 0 -19.5 12 -1.5];
    c(4,:)=[ -2 24 -78 112 -78 24 -2];
    y=feval(func,x+[-3:3]*h);
    q=c(n,:)*y';q=q/(12*h^n);
else
    disp('n must be 1, 2, 3 or 4');
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function D = divdif(x,y) % Construct divided difference table
m = length(x);
D = zeros(m,m);
D(:,1) = y(:);
for j = 2:m
    for i = j:m
        D(i,j) = (D(i,j-1) - D(i-1,j-1))/(x(i)-x(i-j+1));
    end; end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [tvals, yvals]=feuler(f,start,finish,startval,h)
% solves dy/dt=f(t,y).start,finish are initial, final values of t
% startval is initial value of y, h is the increment in t
s=(finish-start)/h+1;
y=startval;t=start;
yvals=startval;tvals=start;
for i=2:s
    yl=y+h*feval(f,t,y); t1=t+h;
    % collect values together for output
    tvals=[tvals, t1]; yvals=[yvals, yl];
    t=t1;y=y1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function c = fgenfit1(func,x,y)
n = length(y);
[p,jj] = feval (func,x(1));
A = zeros(p,p); b = zeros (p,1);
for i = 1:n
    [jj,f] = feval (func,x(i));
    for j = 1:p
        for k = 1:p
            A(j,k) = A(j,k)+f(j)*f(k);
        end
        b(j) = b(j)+y(i)*f(j);
    end;
end
c=A\b;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function x=Gaussian(B)
[n,t]=size(B);G=B;

```

```

for i=1:n-1
    for j=i:n-1
        m=G(j+1,i)/G(i,i);
        for k=1:t
            G(j+1,k)=G(j+1,k)-m*G(i,k);
        end
    end
end
j=n;x(j,1)=G(j,t)/G(j,j);
for j=n-1:-1:1
    s=0;
    for k=n:-1:j+1
        s=s+G(j,k)*x(k,1);
    end
    x(j,1)=(G(j,t)-s)/G(j,j);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function x=GaussSeidel(B,x,acc)
[n,t]=size(B);
b=B(1:n,t);
R=1;k=1;
d(1,1:n+1)=[0 x];
k=k+1;
while R>acc
    for i=1:n
        sum=0;
        for j=1:n
            if j<=i-1
                sum=sum+B(i,j)*d(k,j+1);
            elseif j>=i+1
                sum=sum+B(i,j)*d(k-1,j+1);
            end
        end
        x(1,i)=(1/B(i,i))*(b(i,1)-sum);
        d(k,1)=k-1;d(k,i+1)=x(1,i);
    end
    R=max(abs((d(k,2:n+1)-d(k-1,2:n+1))));
    k=k+1;
    if R>100 & k>10
        ('Gauss-Seidel method is Diverges')
    end;
end;
x=d;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function x=Jaccobi(B,x,tol)
[n,t]=size(B);
b=B(1:n,t);
w=1;k=1;
d(1,1:n+1)=[0 x];
while w>tol
    for i=1:n
        sum=0;
        for j=1:n
            if j~=i
                sum=sum+B(i,j)*d(k,j+1);
            end
        end
        x(1,i)=(1/B(i,i))*(b(i,1)-sum);
    end
    k=k+1;
    d(k,1:n+1)=[k-1 x];
    w=max(abs((d(k,2:n+1)-d(k-1,2:n+1))));
    if w>100 & k>10

```

```

('Jaccobi method is Divergent')
end;
end;
x=d;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Wave equation Backwards-difference method %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear
fprintf(1,'\n%s\n','The Hyperbolic equation of the form ');
fprintf(1,'\n%s\n','d^2u/dt^2 - (alpha^2)*d^2u/dx^2 = 0  0<x<l,
0<t<T');
fprintf(1,'\n%s\n','Subject to the boundary conditions ');
fprintf(1,'\n%s\n','u(0,t)=u(l,t)=0      0<t<T');
fprintf(1,'\n%s\n','u(x,0)=f(x)');
fprintf(1,'\n%s\n','du/dt=g(x)      0<x<l');
% initializations %%
n = input('enter the number of grid sections for the t variable; n=
');
m= input('enter the number of grid sections for the x variable; m =
');
l1 = input('enter the end point of the range for x; l= ');
T = input('enter the end point of the range for t; T= ');
alpha = input('enter the constant alpha= ');
fx0= input('enter the boundary condition f(x)=','s');
gx0= input('enter the boundary condition g(x)=','s');
ee= input('enter the exact solution e(x,t)=','s');
% step sizes
h=l1/m;k=T/n;
l=alpha*k/h;
% initial conditions %%%
for i=1:m+1
x=(i-1)*h;
ff(i)=eval(fx0);
gg(i)=eval(gx0);
end
w(1,1)=ff(1);
w(m+1,1)=ff(m+1);
for ii=2:m
    w(ii,1)=ff(ii);
    w(ii,2)=(1^2/2)*(ff(ii+1)+ff(ii-1))+(1-1^2)*(ff(ii))+
k*(gg(ii));
end
% boundary conditions
for g=2:n+1
    w(1,g)=0;
    w(m+1,g)=0;
end
% matrix multiplication
for j=2:n
for i=2:m
w(i,j+1)=2*(1-1^2)*w(i,j)+(1^2)*(w(i+1,j)+w(i-1,j))-w(i,j-1);
end
end
% exact solution %%
for ic=1:m+1
    for jc=1:n+1
        x=(ic-1)*h;
        t=(jc-1)*k;
        e(ic,jc)=eval(ee);
    end
end
fprintf(1,'\n%s\n','The exact solution is ');
e

```

```

% error calculation
r1=ones((n+1),1);
error1=(e-w).^2*r1;
error=sum(error1);
% plotting exact and the approximated solution %%
mesh(w)
title('approximated solution');
figure
mesh(e)
title('exact solution');
fprintf(1,'\n%s\n','The approximated solution is ');
w
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Poisson equation central-difference method
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear
fprintf(1,'\n%s\n','The Elliptic equation of the form ');
fprintf(1,'\n%s\n','d^2u/dt^2 + d^2u/dx^2 = f(x,y) a<x<b, c<y<d');
fprintf(1,'\n%s\n','Subject to the boundary conditions ');
fprintf(1,'\n%s\n','u(a,y)=g(a,y) c<y<d');
fprintf(1,'\n%s\n','u(b,y)=g(b,y)');
fprintf(1,'\n%s\n','u(x,c)=g(x,c) a<x<b');
fprintf(1,'\n%s\n','u(x,d)=g(x,d)');
%initializations %%
n = input('enter the number of grid sections for the x variable; n = ');
m= input('enter the number of grid sections for the y variable; m = ');
ep_max = input('enter the maximum number of iterations ');
a =input('enter the right end point of the range of x ; a= ');
b=input('enter the left end point of the range of x ; b= ');
c1 = input('enter the right end point of the range of y ; c= ');
d= input('enter the left end point the range of y ; d= ');
tol= input('enter the tolerance ; tol=');
ff= input('enter the function f(x,y)=','s');
ee= input('enter the exact solution e(x,y)=','s');
gxc= input('enter boundary condition u(x,c)=','s');
gxd= input('enter boundary condition u(x,d)=','s');
gat= input('enter boundary condition u(a,y)=','s');
gbt= input('enter boundary condition g(b,y)=','s');
% step sizes
h=(b-a)/n;k=(d-c1)/m;
lm=(h^2)/(k^2);
u=2*(1+lm);
% exact solution %%
for ic=1:n+1
    for jc=1:m+1
        x=a+(ic-1)*h;
        y=c1+(jc-1)*k;
        e(ic,jc)=eval(ee);
    end
end
for ic=1:n+1
    for jc=1:m+1
        x=a+(ic-1)*h;
        y=c1+(jc-1)*k;
        f(ic,jc)=eval(ff);
    end
end
fprintf(1,'\n%s\n','The exact solution is ');
e
w=zeros(n+1,m+1);

```

```

%boundary conditions
for j=1:m+1
y=c1+(j-1)*k;
w(1,j)=eval(gat);
w(n+1,j)=eval(gbt);
end
for i=1:n+1
x=a+(i-1)*h;
w(i,1)=eval(gxc);
w(i,m+1)=eval(gxd);
end
L=1;
% Gauss-Siedel iterations
while L<= ep_max
z=(-f(2,m)*h^2+w(1,m)+lm*w(2,m+1)+lm*w(2,m-1)+w(3,m))/u;
norm=abs(z-w(2,m));
w(2,m)=z;
for i=2:n-2
z=(-f(i+1,m)*h^2+lm*w(i+1,m+1)+lm*w(i+1,m-1)+w(i,m)+w(i+2,m))/u;
if abs(w(i+1,m)-z)>norm
norm=abs(w(i+1,m)-z);
end
w(i+1,m)=z;
end
z=(-f(n,m)*h^2+w(n+1,m)+lm*w(n,m+1)+lm*w(n,m-1)+w(n-1,m))/u;
if abs(w(n,m)-z)>norm
norm=abs(w(n,m)-z);
end
w(n,m)=z;
for j = m-2:-1:2
z=(-f(2,j+1)*h^2+w(1,j+1)+lm*w(2,j+2)+lm*w(2,j)+w(3,j+1))/u;
if abs(w(2,j+1)-z)>norm
norm=abs(w(2,j+1)-z);
end
w(2,j+1)=z;
for i=2:n-2
z=(f(i+1,j+1)*h^2+w(i,j+1)+lm*w(i+1,j+2)+lm*w(i+1,j)+w(i+2,j+1))/u;
if abs(w(i+1,j+1)-z)>norm
norm=abs(w(i+1,j+1)-z);
end
w(i+1,j+1)=z;
end
z=(-f(n,j+1)*h^2+w(n+1,j+1)+lm*w(n,j+2)+lm*w(n,j)+w(n-1,j+1))/u;
if abs(w(n,j+1)-z)>norm
norm=abs(w(n,j+1)-z);
end
w(n,j+1)=z;
end
z=(-f(2,2)*h^2+w(1,2)+lm*w(2,1)+lm*w(2,3)+w(3,2))/u;
if abs(w(2,2)-z)>norm
norm=abs(w(2,2)-z);
end
w(2,2)=z;
for i=2:n-2
z=(-f(i+1,2)*h^2+lm*w(i+1,1)+w(i,2)+lm*w(i+1,3)+w(i+2,2))/u;
if abs(w(i+1,2)-z)>norm
norm=abs(w(i+1,2)-z);
end
w(i+1,2)=z;
end
z=(-f(n,2)*h^2+w(n+1,2)+lm*w(n,1)+lm*w(n,3)+w(n-1,2))/u;
if abs(w(n,2)-z)>norm

```

```

norm=abs(w(n,2)-z);
end
w(n,2)=z;
if norm <= tol
fprintf(1,'\n%s\n','The approximated solution is ');
w
L
L=ep_max;
end
L=L+1;
end
r1=ones((m+1),1);
error1=(e-w).^2*r1;
error=sum(error1)
mesh(w)
title('approximated solution');
figure
mesh(e)
title('exact solution');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Heat equation Backwards-difference method
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear
fprintf(1,'\n%s\n','The Parabolic equation of the form ');
fprintf(1,'\n%s\n','d^2u/dt^2 - (alpha^2)* d^2u/dx^2 =0 0<x<1,
0<t<T');
fprintf(1,'\n%s\n','Subject to the boundary conditions ');
fprintf(1,'\n%s\n','u(0,t)=T1');
fprintf(1,'\n%s\n','u(1,t)=T2 0<t<T');
fprintf(1,'\n%s\n','u(x,0)=f(x) 0<=x<=1');
% initializations %%
n = input('enter the number of grid sections for the x variable; n =
');
m= input('enter the number of grid sections for the y variable; m =
');
l = input('enter the end point of the range for x ; l= ');
T = input('enter the end point of the range for t ; T= ');
ee= input('enter the exact solution e(x,t)=','s');
alpha = input('enter the constant alpha = ');
T1 = input('enter the constant T1 = ');
T2 = input('enter the constant T2 = ');
gx0= input('enter boundary condition u(x,0)=f(x)=','s');
% step sizes
h=l/m;
k=T/n;
lm=k*(alpha/h)^2;
% initial condition
for ii=2:m
x=(ii-1)*h;
w(ii)=eval(gx0);
w1(ii,1)=w(ii);
end
% boundary conditions
for j=1:n+1
w1(1,j)=0;
w1(m+1,j)=0;
end
% exact solution
for ic=1:m+1
for jc=1:n+1
x=(ic-1)*h;
t=(jc-1)*k;
e(ic,jc)=eval(ee);

```

```

end
end
fprintf(1,'\n%s\n','The exact solution is ');
e
% Solving the tridiagonal system Crout reduction
s(2)=1+2*lm;
u(2)=-lm/s(2);
for i=3:m-1
s(i)=1+2*lm+lm*u(i-1);
u(i)=-lm/s(i);
end
s(m)=1+2*lm+lm*u(m-1);
for j=2:n+1
z(2)=w(2)/s(2);
for i=3:m
z(i)=(w(i)+lm*z(i-1))/s(i);
end
w(m)=z(m);
for i=m-1:-1:2
w(i)=z(i)-u(i)*w(i+1);
end
for i=2:m
w1(i,j)=w(i);
end
end
% calculating error
r1=ones((n+1),1);
error1=(e-w1).^2*r1;
error=sum(error1)
% graphing the approximate & exact solutions %%
figure
mesh(w1)
title('approximate solution');
figure
mesh(e)
title('exact solution');
fprintf(1,'\n%s\n','The approximate solution is ');
w1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Heat equation crank-nicholson
method%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear
fprintf(1,'\n%s\n','The Parabolic equation of the form ');
fprintf(1,'\n%s\n','du/dt = (alpha^2)* d^2u/dx^2 =0 0<x<1, 0<t<T');
fprintf(1,'\n%s\n','Subject to the boundary conditions ');
fprintf(1,'\n%s\n','u(0,t)=T1');
fprintf(1,'\n%s\n','u(1,t)=T2 0<t<T');
fprintf(1,'\n%s\n','u(x,0)=f(x) 0<=x<=1');
% initializations %%
n = input('enter the number of grid sections for the x variable; n=
');
m= input('enter the number of grid sections for the y variable; m =
');
l = input('enter the end point of the range for x ; l= ');
T = input('enter the end point of the range for t ; T= ');
ee= input('enter the exact solution e(x,t)=','s');
alpha = input('enter the constant alpha = ');
T1 = input('enter the constant T1 = ');
T2 = input('enter the constant T2 = ');
gx0= input('enter boundary condition u(x,0)=f(x)=','s');
% step sizes
h=l/m;
k=T/n;

```



```

lm=k*(alpha/h)^2;
w(m+1)=0;
% initial condition
for ii=2:m
x=(ii-1)*h;
w(ii)=eval(gx0);
w1(ii,1)=w(ii);
end
% boundary conditions
for j=1:n+1
w1(1,j)=0;
w1(m+1,j)=0;
end
% exact solution
for ic=1:m+1
    for jc=1:n+1
        x=(ic-1)*h;
        t=(jc-1)*k;
        e(ic,jc)=eval(ee);
    end
end
fprintf(1,'\n%s\n','The exact solution is ');
e
% Solving the tridiagonal system Crout reduction
s(2)=1+lm;
u(2)=-lm/(2*s(2));
for i=3:m-1
s(i)=1+lm+lm*u(i-1)/2;
u(i)=-lm/(2*s(i));
end
s(m)=1+lm+lm*u(m-1)/2;
for j=2:n+1
z(2)=( (1-lm)*w(2)+(lm/2)*w(3) )/s(2);
for i=3:m
z(i)=( (1-lm)*w(i)+(lm/2)*(z(i-1)+w(i-1)+w(i+1))) )/s(i);
end
w(m)=z(m);
for i= m-1:-1:2
w(i)=z(i)-u(i)*w(i+1);
end
for i=2:m
w1(i,j)=w(i);
end
end
% calculating error
r1=ones((n+1),1);
error1=(e-w1).^2*r1;
error=sum(error1)
% graphing the approximate & exact solutions %%
figure
mesh(w1)
title('approximate solution');
figure
mesh(e)
title('exact solution');
fprintf(1,'\n%s\n','The approximate solution is '); w1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [r,it]=modifiedNewton(fun,dfun,ddfun,x0,acc)
it=0;o=x0+1;
while abs(x0-o)>acc
    o=x0;
    it=it+1;

```

```

        x0=o-((feval(fun,o)*feval(dfun,o))/((feval(dfun,o).^2)-
(feval(ddfun,o)*feval(fun,o))));
    end;
r=x0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [r,it,p,pp]=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);
    p(it)=x1;pp(it)=feval(fun,x1);
end;
r=x0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [xv,it]=newton2(x,f,jf,n,acc)
it=0;xv=x;fr=feval(f,xv);
while norm(fr)>acc
    jr=feval(jf,xv);
    xv1=xv-jr\fr;xv=xv1;
    fr=feval(f,xv);
    it=it+1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[tvals,yvals]=rkgen(f,start,finish,startval,h,method)
% solves dy/dt=f(t,y). start, finish are initial, final values of t
% startval is initial value of y,h is the increment in t
% method (1, 2 or 3) selects Classical, Butcher or Merson RK.
b=[ ];c=[ ];d=[ ];
if method <1 | method >3
    disp('Method number unknown so using Classical');
    method=1;
end;
if method==1
    order=4;
    b=[1/6 1/3 1/3 1/6]; d=[0 .5 .5 1];
    c=[0 0 0 0;0.5 0 0 0;0 .5 0 0;0 0 1 0];
    disp('Classical method selected');
elseif method ==2
    order=5;
    b=[1/6 0 0 2/3 1/6];
    d=[0 1/3 1/3 1/2 1];
    c=[0 0 0 0 0;1/3 0 0 0 0;1/6 1/6 0 0 0;1/8 0 3/8 0 0;1/2 0 -3/2 2
0];
    disp('Merson method selected');
end;
s=(finish-start)/h+1;
y=startval; t=start;
yvals=startval; tvals=start;
for j=2:s
    k(1)=h*feval(f,t,y);
    for i=2:order
        k(i)=h*feval(f,t+h*d(i),y+c(i,1:i-1)*k(1:i-1)');
    end;
    y1=y+b*k'; t1=t+h;
    % collect values together for output
    tvals=[tvals, t1]; yvals=[yvals, y1];
    t=t1; y=y1;
end;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [r,it]=secant(fun,a,b,acc)
x1=b;x0=a;o=x1+1;it=0;
while abs(x1-o)>acc

```





## المراجع

### References

أولاً: المراجع العربية

- [1] مجبوح ، أسامة أسعد مجبوح. *MATLAB لغة المهندسين* ، دار الكتب العلمية للنشر والتوزيع ، القاهرة، مصر، ١٤٢٥هـ.
- [2] رشوان، أسامة عجمي. *مرشد برنامج ماتلاب للرياضيات الجامعية* ، مكتب الفلاح للنشر والتوزيع ، الكويت ، ١٤٢٣هـ.
- [3] زرتي، عمر. *الطرق العددية باستخدام فورتران*، دار الحكمة ، طرابلس ، ليبيا، ١٩٩٥م.
- [4] نعمة، مازن. *نعمة، محمد. تعلم برمجة 7 MATLAB للمبتدئين* ، دار القلم العربي ، حلب، سوريا، ٢٠٠٦م.

ثانياً: المراجع الأجنبية

- [5] Burden R.L., Faires D., *Numerical Analysis*, Seventh edition, 2001.
- [6] Backstrom G., *Practical Mathematics using Matlab*, Lund, 2000.
- [7] Butt R., Hadid Y., *Introduction to Numerical Analysis with Matlab*, Almunata Printing Press, 2003.
- [8] Cheney W., Kincaid D., *Numerical Mathematics and Computing*, Brooks/Cole publishing company, 1999 .
- [9] Cooper J. , *Matlab companion for Multivariable Calculus*, Academic Press, 2001.

- [10] Davis T. Sigmon K., *Matlab Primer*, Chapman &Hall/CRC, 2005.
- [11] Etter D.M., *Engineering problem Solving with Matlab*, Prentice Hall, 1997.
- [12] Hahn B., *Essentials Matlab for scientists and engineers*, Longmann, 2002.
- [13] Hanselman D., Littlefield B., *Mastering Matlab*, The Mathworks Inc, Printice Hall , 1998.
- [14] Jensen G., *Using Matlab in Calculus*, Prentice Hall, 2000.
- [15] Lindfield G., Penny J. , *Numerical methods using Matlab*, Ellis Horwood Limited, 1995.
- [16] McMahon D., *Matlab Demystified a self teaching guide*, McGraw Hill, 2007.
- [17] Moler C., *The student Edition of Matlab version 5 for windows software and users Guide*, The Mathworks Inc, Printice Hall , 1995.
- [18] Polking J., *Ordinary Differential Equations using Matlab*, The Mathworks Inc, Printice Hall, 1999.
- [19] Moler, Forsyth G., *Numerical Computing with Matlab*, The Mathworks inc., 2004.
- [20] Knight A. , *Basics of Matlab and Beyond*, Chapman of Hcal/crc, 2000.
- [21] [Steven K., *Numerical Analysis using Matlab and Spreadsheet*, 2<sup>nd</sup> edition, Orchard Publishers, 2004.
- [22] Smith G.D., *Numerical Solution of Partial Differential Equations*, Oxford University Press, Oxford; 1965.
- [23] Richard Goering, "Matlab edges closer to electronic design automation world," EE Times, 10/04/2004.
- [24] Patric Marchand, *Graphics and GUIs with Matlab*, 2<sup>nd</sup> edition, CRC Press, Boca Raton, 1999.
- [25] The Mathworks, *Using Matlab Graphics*, v5, The Mathworks Inc., MA, 1996.
- [26] Kermit Sigmon, *Matlab Primer*, 5<sup>th</sup> edition, CRC Press, Boca Raton, 1998.
- [27] Gilat A., *Matlab an introduction with applications*, 3<sup>rd</sup> edition, John Wiley & Sons Inc., USA, 2007.
- [28] Merson R.H. , *An operational method for the study of integration processes*,

Proc. Conf. on data processing and automatic computing machines salisbury, Australia, 1957.

ثالثاً: مواقع الإنترنت

- [29] <http://www.mathworks.com>
- [30] <HTTP://WWW.MATHWORKS.COM/SUPPORT/BOOKS/BOOK2595.HTML>
- [31] [HTTP://WWW.MATHWORKS.COM/ACADEMIA/STUDENT\\_CENTER/TUTORIALS/INTROPAGE.HTML](HTTP://WWW.MATHWORKS.COM/ACADEMIA/STUDENT_CENTER/TUTORIALS/INTROPAGE.HTML)
- [32] <HTTP://WWW.MATHWORKS.COM/PRODUCTS/MATLAB/DEMOS.HTML>
- [33] <HTTP://WWW.ENGIN.UMICH.EDU/GROUP/CTM/>
- [34] [HTTP://WWW.MATHWORKS.COM/COMPANY/NEWSLETTERS/NEWS\\_NOTES/CLEVESCORNER/DEC04.HTML](HTTP://WWW.MATHWORKS.COM/COMPANY/NEWSLETTERS/NEWS_NOTES/CLEVESCORNER/DEC04.HTML)
- [35] [HTTP://WWW.MATHWORKS.COM/COMPANY/NEWSLETTERS/NEWS\\_NOTES/CLEVESCORNER/JAN06.PDF](HTTP://WWW.MATHWORKS.COM/COMPANY/NEWSLETTERS/NEWS_NOTES/CLEVESCORNER/JAN06.PDF)





## ثبتت المصطلحات

أولاً: عربي - إنجليزي

أ

Output	الإخراج
Input	الإدخال
Logic operators	أدوات المنطق
Basis	أساس
Stability	استقرار
Interpolations	استكمال
Linear Independence	استقلال خطي
Standard Deviation	انحراف معياري

ب

Programming	برمجة
Linear Programming	برمجة خطية
Bernoli	برنولي



Co-variance	تباين مصاحب
Doolittle Factorization	تحليلي دوليتل
Cheloski Factorization	تحليل شلوسكي
Fourier Analysis	تحليل فوريير
SVD	تحليل القيم الشاذة
Crout Factorization	تحليل كراوت
Load	تحميل
Fourier Transforms	تحويلات فوريير
Fast Fourier Transforms	تحويلات فوريير السريع
Discrete Fourier Transforms	تحويلات فوريير المتقطعة
Cryptology	تشفير
Differentiation	تفاضل
Integration	تكامل
Multiple Integration	تكامل متعدد



Matrix algebra	جبر المصفوفات
Gramm Schmit	جرام شميت

م

Gaussian Elimination	حذف جاوس
Symbolic Algebra	حساب رمزية
Vector Calculus	حساب المتجهات
Multivariable Calculus	حساب متعدد المتغيرات
Solids of Revolution	حجوم دورانية
Save	حفظ
Loops	حلقات

ا

Functions	دوال
Statistical functions	دوال الإحصاء
Logical functions	دوال المنطق

د

Matrix rank	رتبة المصفوفة
Plot	رسم منحنيات
Plot 3D	رسم منحنيات ثلاثي الأبعاد

ش

Pseudo-inverse

شبه معكوس

Stability condition

شرط الاستقرار

Cubic Spline Interpolation

شريحة تكعيبية للاستكمال

ص

Row reduced echolean form

الصيغة الدرجية الصفية المختزلة

ض

Dot product

الضرب القياسي لمتجهين

Cross product

الضرب الاتجاهي لمتجهين

ط

Print

طباعة

Iterative methods

طرق تكرارية

Optimization methods

طرق مثلى

Euler method

طريقة أويلر

Predictor-Corrector method	طريقة التخمين والتصحيح
Bisection method	طريقة التنصيف
Runge kutta method	طريقة رونج كوتا
Jacobi method	طريقة جاكوبي
Gauss-Seidal method	طريقة جاوس سيدال
Simplex method	طريقة سمبلكس
Secant method	طريقة القاطع
Newton method	طريقة نيوتن
Modified method	طريقة نيوتن المعدلة
Arc length	طول القوس

ع

Condition number	عدد شرطي
Basic operations	عمليات حسابية

ف

Divided Difference	فرق تجزيئي
--------------------	------------

ق

Simpson rule	قاعدة سمبسون
--------------	--------------

Trapazoidal rule	قاعدة شبه المنحرف
Optimum points	قيم قصوى
Eigenvalues	قيم مميزة



Newton interpolating polynomials	كثيرة حدود نيوتن للاستكمال
Lagrange interpolating polynomials	كثيرة حدود لاجرانج للاستكمال



Sequences	متتاليات
Vectors	متجهات
Series	متسلسلات
Taylor series	متسلسلة تايلور
Symmetric	متناظرة
Riemann Integral	مجموع ريمان
Polar coordinates	محاور قطبية
Matrix determinant	محدد المصفوفة
Least squares problems	مسائل أصغر مربعات
Surface of revolution	مساحة سطح دوران
Derivative	مشتقة

Matrix	مصفوفات
Diagonal matrix	مصفوفة قطرية
Augmented matrix	مصفوفة موسعة
strictly diagonally dominant	مصفوفة مسيطرة قطريا بدقة
Identity matrix	مصفوفة الوحدة
Factorial	مضروب
Characteristic equation	معادلة مميزة
Differential equations	معادلات تفاضلية
Parabolic Partial differential equations	معادلات تفاضلية جزئية تكافئية
Hyperbolic Partial differential equations	معادلات تفاضلية جزئية زائدية
Elliptic Partial differential equations	معادلات تفاضلية جزئية ناقصية
Coefficient of variation	معامل تغيير
Posotive definite	معرفة إيجابياً
Matrix inverse	معكوس المصفوفة
files	ملفات
Parametric curves	منحنيات وسيطية
Matrix transpose	منقول مصفوفة
Complex conjugate transpose	منقول مرافق مركب

System of linear equations	نظام معادلات خطية
Nonlinear system of equations	نظام معادلات غير خطية
Consistent system	نظام متسق
Norm	نظيم
Influctuation point	نقطة انقلاب
Limit	نهاية
Windows	نوافذ
Mean	وسط حسابي
Median	وسيط



## ثانياً: إنجليزي - عربي

A

Arc length

طول القوس

Augmented matrix

مصفوفة موسعة

B

Basic operations

عمليات حسابية

Basis

أساس

Bernoli

برنولي

Bisection method

طريقة التنصيف

C

Characteristic equation

معادلة مميزة

Cheloski Factorization

تحليل شلوسكي

Coefficient of variation

معامل تغيير

Complex conjugate transpose

منقول مرافق مركب

Condition number

عدد شرطي

Consistent system

نظام متسق

Co-variance

تباين مصاحب

Cross product

الضرب الاتجاهي لمتجهين

Crout Factorization

تحليل كراوت

Cryptology

تشفير

Cubic Spline Interpolation

شريحة تكعيبية للاستكمال

## D

Derivative

مشتقة

Diagonal matrix

مصفوفة قطرية

Differentiation

تفاضل

Differential equations

معادلات تفاضلية

Discrete Fourier Transforms

تحويلات فوريير المتقطعة

Divided Difference

فرق تجزيئي

Doolittle Factorization

دوليتل تحليل

Dot product

الضرب القياسي لمتجهين

## E

Eigenvalues

قيم مميزة

Elliptic Partial differential equations

معادلات تفاضلية جزئية ناقصية

Euler method

طريقة أولر

## F

Factorial

مضروب

Fast Fourier Transforms

تحويلات فوريير السريع

Files	ملفات
Fourier Analysis	تحليل فوريير
Fourier Transforms	تحويلات فوريير
Functions	دوال

## G

Gauss-Seidal method	طريقة جاوس سيدال
Gaussian Elemination	حذف جاوس
Gramm Schmit	جرام شميت
Hyperbolic Partial differential equations	معادلات تفاضلية جزئية زائدية

## I

Identity matrix	مصفوفة الوحدة
Inconsistent system	نظام غير متسق
Influctuation point	نقطة انقلاب
Input	الإدخال
Integration	تكامل
Interpolations	استكمال
Iterative methods	طرق تكرارية

## J

Jacobi method	طريقة جاكوبي
---------------	--------------

## L

Lagrange interpolating polynomials	كثيرة حدود لاجرانج للاستكمال
Least squares problems	مسائل أصغر مربعات
Limit	نهاية
Linear Independence	استقلال خطي
Linear Programming	برمجة خطية
Load	تحميل
Logic operators	أدوات المنطق
Logical functions	دوال المنطق
Loops	حلقات

## M

Matrix	مصفوفات
Matrix algebra	جبر المصفوفات
Matrix determinant	محدد المصفوفة
Matrix inverse	معكوس المصفوفة
Matrix rank	رتبة المصفوفة
Matrix transpose	منقول مصفوفة
Mean	وسط حسابي
Median	وسيط

Modified method	طريقة نيوتن المعدلة
Multiple Integration	تكامل متعدد
Multivariable Calculus	حساب متعدد المتغيرات

## N

Newton interpolating polynomials	كثيرة حدود نيوتن للاستكمال
Newton method	طريقة نيوتن
Nonlinear system of equations	نظام معادلات غير خطية
Norm	نظيم

## O

Optimization methods	طرق مثلى
Optimum points	قيم قصوى
Output	الإخراج

## P

Parabolic Partial differential equations	معادلات تفاضلية جزئية تكافئية
Parametric curves	منحنيات وسيطية
Plot	رسم منحنيات
Plot 3D	رسم منحنيات ثلاثي الأبعاد
Polar coordinates	محاور قطبية
Posotive definite	معرفة إيجابياً

Predictor-Corrector method	طريقة التخمين والتصحيح
Print	طباعة
Programming	برمجة
Pseudo-inverse	شبه معكوس
Riemann Integral	مجموع ريمان
Row reduced echolean form	الصيغة الدرجية الصفية المختزلة
Runge kutta method	طريقة رونج كوتا

## S

Save	حفظ
Secant method	طريقة القاطع
Sequences	متتاليات
Series	متسلسلات
Simplex method	طريقة سمبلكس
Simpson rule	قاعدة سمبسون
Solids of Revolution	حجوم دورانية
Stability	استقرار
Stability condition	شرط الاستقرار
Standard Deviation	انحراف معياري
Statistical functions	دوال الاحصاء
strictly diagonally dominant	مصنوفة مسيطرة قطريا بدقة
Surface of revolution	مساحة سطح دوران

SVD		تحليل القيم الشاذة
Symbolic Algebra		حساب رمزية
Symmetric		متناظرة
System of linear equations		نظام معادلات خطية
	<b>T</b>	
Taylor series		متسلسلة تايلور
Trapezoidal rule		قاعدة شبه المنحرف
	<b>V</b>	
Vector Calculus		حساب المتجهات
Vectors		متجهات
	<b>W</b>	
Windows		نوافذ





## كشاف الموضوعات



- تباين مصاحب ٢٣٥
- تحليلي دوليتل ٧١
- تحليل شلوسكي ٧٣
- تحليل فورير ٢٠٤
- تحليل القيم الشاذة ٧٥
- تحليل كراوت ٧١
- تحميل ٨، ٣١
- تحويلات فورير ٢٠٧
- تحويلات فورير السريع ٢١٤
- تحويلات فورير المتقطعة ٢٠٨
- تشفير ٢٣٩
- تفاضل ١٠٦
- تكامل ٥٢، ١١٨
- تكامل متعدد ١٢٦



- الإخراج ٣٠
- الإدخال ٣٠
- أدوات المنطق ٤٤
- أساس ٢٢٤
- استقرار ١٠٦
- استكمال ١٨١
- استقلال خطي ٢٢٣
- انحراف معياري ٢٣٤



- برمجة ٤٧
- برمجة خطية ٢٢٧
- برنولي ٢٣٦

د

رتبة المصفوفة ١٢

رسم منحنيات ٣٢

رسم منحنيات ثلاثي الأبعاد ٤٠

ش

شبه معكوس ٢٠

شرط الاستقرار ١٧٢

شريحة تكعيبية للاستكمال ١٩٠

ص

الصيغة الدرجية الصفية المختزلة ٦٨

ض

الضرب القياسي لمتجهين ٢٢١

الضرب الاتجاهي لمتجهين ٢٢٢

ط

ج

جبر المصفوفات ١٦

جرام شميت ٢٢٥

ح

حذف جاوس ٦٨

حساب رمزية ٥١

حساب المتجهات ٢١٩

حساب متعدد المتغيرات ١٣٣

حجوم دورانية ١٢٦

حفظ ٧

حلقات ٤٧

د

دوال ٢٦

دوال الإحصاء ٢٣٢

دوال المنطق ٤٤

فرق تجزيئي ١٠٦

ج

قاعدة سمبسون ١١٩

قاعدة شبه المنحرف ١١٨

قيم قصوى ١٢٩

قيم مميزة ٢١ ، ٨٠

ك

كثيرة حدود نيوتن للاستكمال ١٨٣

كثيرة حدود لاجرانج للاستكمال ١٨٢

م

متتاليات ١٠٢

متجهات ١١

متسلسلات ١٠٣

متسلسلة تايلور ١٠٤

متناظرة ٢٢

طباعة ٣٠

طرق تكرارية ٧٦

طرق مثلى ٢٢٧

طريقة أويلر ١٥٢

طريقة التخمين والتصحيح ١٥٩

طريقة التصنيف ٨٥

طريقة رونج كوتا ١٥٦

طريقة جاكوبي ٧٦

طريقة جاوس سيدال ٧٧

طريقة سمبلكس ٢٢٨

طريقة القاطع ٨٨

طريقة نيوتن ٨٧

طريقة نيوتن المعدلة ٨٩

طول القوس ١٢٦

ع

عدد شرطي ٢٤

عمليات حسابية ٨

ف

- مجموع ريمان ١١٨  
 محاور قطبية ٣٧  
 محدد المصفوفة ١٩  
 مسائل أصغر المربعات ٧٥ ، ١٩٤  
 مساحة سطح دوران ١٣١  
 مشتقة ٥٢ ، ١٠٦  
 مصفوفات ١١  
 مصفوفة قطرية ١٥  
 مصفوفة موسعة ٦٤  
 مصفوفة مسيطرة قطريا بدقة ٧٩  
 مصفوفة الوحدة ١٩  
 مضروب ٤٩  
 معادلة مميزة ٢١ ، ٨٠  
 معادلات تفاضلية ١٥١  
 معادلات تفاضلية جزئية تكافئية ١٧٢  
 معادلات تفاضلية جزئية زائدية ١٧٥  
 معادلات تفاضلية جزئية ناقصية ١٦٦  
 معامل تغيير ٢٣٤  
 معرفة إيجابياً ٧١  
 معكوس المصفوفة ٢٠  
 ملفات ٢٦
- ج**
- منحنيات وسيطية ٣٧  
 منقول مصفوفة ٢٢  
 منقول مرافق مركب ٢٢  
 نظام غير متنسق ٦٣  
 نظام معادلات خطية ٥٩  
 نظام معادلات غير خطية ٩٤  
 نظام متنسق ٦٣  
 تنظيم ٢٣  
 نقطة انقلاب ١١٤  
 نهاية ٥٤  
 نوافذ ٥
- و**
- وسط حسابي ٢٣٣  
 وسيط ٢٣٤