**PRACTICA DEL CAPITULO 4. SISTEMAS DE ECUACIONES NO LINEALES**

**POR**

**LUIS MARIO URREA MURILLO**

**UNIVERSIDAD COOPERATIVA DE COLOMBIA - POPAYAN**

**FACULTAD DE INGENIERIA DE SISTEMAS**

**POPAYÁN – CAUCA**

**2010**

**PRACTICA DEL CAPITULO 4. SISTEMAS DE ECUACIONES NO LINEALES**

**POR**

**LUIS MARIO URREA MURILLO**

Presentado al profesor Ing. Esp. Andrés Escallon, en el programa de Análisis Numérico

**UNIVERSIDAD COOPERATIVA DE COLOMBIA - POPAYAN**

**FACULTAD DE INGENIERIA DE SISTEMAS**

**POPAYÁN – CAUCA**

**2010**

Contenido

1. SISTEMAS DE ECUACIONES NO LINEALES: demostraciones aplicadas y algoritmos en matlab 4

1.1. Método de punto fijo multivariado 4

1.2. Método de Newton Rapson 7

1.3. Método de Broyden 11

**LISTA DE TABLAS Y CUADROS**

1. Tabla de iteraciones y resultados finales de x, y y z 7

2. Tabla de iteraciones y resultados finales de x1 y x2 10

3. Tabla de iteraciones y resultados finales de x1, x2 y x3 14

**NOTA**: Este trabajo fue desarrollado en MATLAB 7.6 por lo tanto, para que funcionen los algoritmos en versiones anteriores de MATLAB tenga en cuenta que:

* Posiblemente las **sentencias** no serán compatibles y deberá cambiar la sintaxis de algunas de ellas.

# SISTEMAS DE ECUACIONES NO LINEALES: demostraciones aplicadas y algoritmos en matlab

## Método de punto fijo multivariado

Utilice el método de punto fijo multivariado para resolver el siguiente sistema de ecuaciones no lineales y compruebe su respuesta.



Solución

Despejando x, y y z se tiene que:

*Para i*

**Iteración 1:**

**Para i=0**

**Errores**

**Iteración 2:**

*Para i=1*

**Errores**

**En MATLAB**

**Algoritmo:**

function [iter,x,ea] = puntofijomultivariado3x3

syms x1 x2 x3 %sistema para tres ecuaciones

x0=input('Digite f(x0)=');%valores iniciales de x1, x2, x3 en un vector fila

tol=input('Digite la Tolerancia del Sistema=');

f(1)= input('digite f(1)='); %ecuación 1

f(2)= input('digite f(2)='); %ecuación 2

f(3)= input('digite f(3)='); %ecuación 3

ecu=[f(1);f(2);f(3)];

iter=0;

ea=[100 100 100];

x1=x0(1); x2=x0(2); x3=x0(3);

disp(x0);

disp(tol);

fprintf('iter x1 x2 x3 ea(1) ea(2) ea(3) \n');

fprintf('%d %f\t %f\t %f\t %f\t %f\t %f\n',iter,x1,x2,x3,ea(1),ea(2),ea(3));

while ((ea(1)>tol)||(ea(2)>tol)||(ea(3)>tol))

iter=iter+1;

x(1)=eval(ecu(1));

x(2)=eval(ecu(2));

x(3)=eval(ecu(3));

x1ante=x1;

x1=x(1);

x2ante=x2;

x2=x(2);

x3ante=x3;

x3=x(3);

ea(1)=abs((x1-x1ante)\*100/x1);

ea(2)=abs((x2-x2ante)\*100/x2);

ea(3)=abs((x3-x3ante)\*100/x3);

fprintf('%d %f\t %f\t %f\t %f\t %f\t %f\n',iter,x1,x2,x3,ea(1),ea(2),ea(3));

end

**Datos de entrada:**

>> puntofijomultivariado3x3

Digite f(x0) = [0.5; 0.5; 0.5;]

Digite la Tolerancia del Sistema = 0.0005

Digite f(1) = sqrt(x2^x3+0.2\*log(x3))

Digite f(2) = sqrt(x3\*(x1-0.3))

Digite f(3) = x1\*x2-log(x2)

**Resultados:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Iteración** | **x1** | **x2** | **x3** | **ea(1)** | **ea(2)** | **ea(3)** |
| **0** | 0.5 | 0.5 | 0.5 | 100 | 100 | 100 |
| **1** | 0.753974 | 0.316228 | 0.943147 | 33.684748 | 58.113883 | 46.986005 |
| **2** | 0.570887 | 0.654343 | 1.389720 | 32.070667 | 51.672470 | 32.134023 |
| **3** | 0.787701 | 0.613561 | 0.797680 | 27.524914 | 6.646739 | 74.220338 |
| **4** | 0.795039 | 0.623722 | 0.971778 | 0.922904 | 1.629092 | 17.915462 |
| **5** | 0.791430 | 0.693590 | 0.967933 | 0.455956 | 10.073428 | 0.397208 |
| **6** | 0.833821 | 0.689689 | 0.914802 | 5.083939 | 0.565613 | 5.807985 |
| **7** | 0.833102 | 0.698814 | 0.946591 | 0.086271 | 1.305659 | 3.358306 |
| **8** | 0.837461 | 0.710373 | 0.940554 | 0.520426 | 1.627246 | 0.641850 |
| **9** | 0.844218 | 0.710993 | 0.936874 | 0.800461 | 0.087174 | 0.392794 |
| **10** | 0.844646 | 0.714048 | 0.941326 | 0.050625 | 0.427814 | 0.472905 |
| **11** | 0.846289 | 0.716023 | 0.939923 | 0.194207 | 0.275903 | 0.149275 |
| **12** | 0.847435 | 0.716568 | 0.940006 | 0.135200 | 0.076045 | 0.008788 |
| **13** | 0.847742 | 0.717351 | 0.940527 | 0.036184 | 0.109090 | 0.055439 |
| **14** | 0.848175 | 0.717751 | 0.940319 | 0.051070 | 0.055721 | 0.022150 |
| **15** | 0.848405 | 0.717955 | 0.940411 | 0.027089 | 0.028447 | 0.009845 |
| **16** | 0.848519 | 0.718141 | 0.940465 | 0.013412 | 0.025878 | 0.005712 |
| **17** | 0.848623 | 0.718236 | 0.940446 | 0.012265 | 0.013230 | 0.002065 |
| **18** | 0.848677 | 0.718296 | 0.940469 | 0.006368 | 0.008454 | 0.002454 |
| **19** | 0.848711 | 0.718341 | 0.940474 | 0.003996 | 0.006152 | 0.000617 |
| **20** | 0.848736 | 0.718365 | 0.940475 | 0.002930 | 0.003399 | 0.000037 |
| **21** | 0.848749 | 0.718381 | 0.940479 | 0.001625 | 0.002284 | 0.000489 |
| **22** | 0.848759 | 0.718392 | 0.940480 | 0.001083 | 0.001501 | 0.000105 |
| **23** | 0.848765 | 0.718399 | 0.940481 | 0.000716 | 0.000890 | 0.000079 |
| **24** | 0.848768 | 0.718403 | 0.940482 | 0.000424 | 0.000593 | 0.000095 |
| **25** | **0.848771** | **0.718406** | **0.940482** | 0.000282 | 0.000375 | 0.000029 |

1. Tabla de iteraciones y resultados finales de x, y y z

## Método de Newton Rapson

Dado el siguiente sistema de ecuaciones no lineales, encuentre los valores de x e y utilizando el método de newton Rapson. Compruebe su respuesta.



Utilice los valores iniciales que usted crea conveniente y una tolerancia de 0.005%

Solución

Igualando las ecuaciones a CERO se tiene que:

Formula de predicción

**Iteración 1:**

*Para i=0*

**Errores**

**Iteración 2:**

*Para i=1*

**Errores**

**En MATLAB**

**Algoritmo:**

function [iter,x,ea] = nr2x2

syms x1 x2

x0=input('Digite un vector fila con los valores iniciales de x0= '); %ecuación x0

tol=input('Tolerancia del sistema= ');

f(1)= input('Digite la ecuación 1 ='); %ecuación 1

f(2)= input('Digite la ecuación 2 ='); %ecuación 2

ecu=[f(1);f(2)];

iter=0;

ea=[100;100];

xx1=x0(1); xx2=x0(2);

fprintf('iter x1 x2 ea(1) ea(2) \n');

fprintf('%d %3.4f\t %3.4f\t %3.4f\t %3.4f\n',iter,xx1,xx2,ea(1),ea(2));

vars = '[';

for i = 1:length(f)

iS = num2str(i);

vars = [vars 'x' iS ' '];

eval(['x' iS ' = sym(''x' iS ''');']);

end

vars = [vars ']' ];

eval(['vars= ' vars ';']);

J = jacobian(ecu, vars);

x = x0;

while ((ea(1)>tol)||(ea(2)>tol))

iter=iter+1;

JJ = double(subs(J, vars, x.'));

FF = double(subs(ecu, vars, x.'));

k=inv(JJ);

disp(k)

x = x - k \* FF;

x1ante=xx1;

xx1=x(1);

x2ante=xx2;

xx2=x(2);

ea(1)=abs((xx1-x1ante)\*100/xx1);

ea(2)=abs((xx2-x2ante)\*100/xx2);

fprintf('%d %3.4f\t %3.4f\t %3.4f\t %3.4f\n',iter,xx1,xx2,ea(1),ea(2));

end

**Datos de entrada:**

Digite un vector fila con los valores iniciales de x0 = [2; 2]

Tolerancia del sistema = 0.005

Digite la ecuación 1 = x1\*x1-x2\*x2-4

Digite la ecuación 2 = exp(-x1)+x1\*x2-1

**Resultados:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iteración** | **x1** | **x2** | **ea(1)** | **ea(2)** |
| **0** | 2.0000 | 2.0000 | 100.0000 | 100.0000 |
| **1** | 1.7062 | 0.7062 | 17.2177 | 183.1952 |
| **2** | 2.0357 | 0.3784 | 16.1864 | 86.6562 |
| **3** | 2.0443 | 0.4260 | 0.4204 | 11.1901 |
| **4** | 2.0448 | 0.4258 | 0.0235 | 0.0640 |
| **5** | **2.0448** | **0.4258** | 0.0000 | 0.0000 |

1. Tabla de iteraciones y resultados finales de x1 y x2

## Método de Broyden

Resuelva el siguiente sistema de ecuaciones no lineales utilizando el método de Broyden para un valor de tolerancia de 0.005%.Utilice los valores inciales que usted crea convenientes.



Solución

Formula de predicción

**Iteración 1:**

*Para i=0*

**Errores**

**Iteración 2:**





es la transpuesta del vector .

*Para i=1*

**Errores**

**En MATLAB**

**Algoritmo:**

function y = Broyden3x3

syms x1 x2 x3

x0=input('Digite un vector fila con los valores iniciales de x0= ');

tol=input('Tolerancia del sistema= ');

f(1)= input('Digite la ecuación 1 ='); %ecuación 1

f(2)= input('Digite la ecuación 2 ='); %ecuación 2

f(3)= input('Digite la ecuación 3 ='); %ecuación 3

F=[f(1);f(2);f(3)];

iter=0;

ea=[100;100;100];

xx1=x0(1); xx2=x0(2); xx3=x0(3);

fprintf('iter x1 x2 x3 ea(1) ea(2) ea(3) \n');

fprintf('%d %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\n',iter,xx1,xx2,xx3,ea(1),ea(2),ea(3));

vars = '[';

for i = 1:length(F)

iS = num2str(i);

vars = [vars 'x' iS ' '];

eval(['x' iS ' = sym(''x' iS ''');']);

end

vars = [vars ']' ];

eval(['vars= ' vars ';']);

J=jacobian(F,vars);

Fold = double(subs(F, vars, x0.'));

Jold=double(subs(J,vars, x0.'));

A0 = inv(Jold);

dx = -A0 \* Fold;

x0 = x0 + dx;

iter=iter+1;

x1ante=xx1;

xx1=x0(1);

x2ante=xx2;

xx2=x0(2);

x3ante=xx3;

xx3=x0(3);

ea(1)=abs((xx1-x1ante)\*100/xx1);

ea(2)=abs((xx2-x2ante)\*100/xx2);

ea(3)=abs((xx3-x3ante)\*100/xx3);

fprintf('%d %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\n',iter,xx1,xx2,xx3,ea(1),ea(2),ea(3));

while ((ea(1)>tol)||(ea(2)>tol)||(ea(3)>tol))

iter=iter+1;

Fnew = double(subs(F,vars, x0.'));

dy = Fnew - Fold;

u = A0 \* dy;

v = dx' \* A0;

denom = dx' \* u;

A0 = A0 + (dx-u) \* v / denom;

dx = -A0 \* Fnew;

x0 = x0 + dx;

x1ante=xx1;

xx1=x0(1);

x2ante=xx2;

xx2=x0(2);

x3ante=xx3;

xx3=x0(3);

ea(1)=abs((xx1-x1ante)\*100/xx1);

ea(2)=abs((xx2-x2ante)\*100/xx2);

ea(3)=abs((xx3-x3ante)\*100/xx3);

fprintf('%d %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\n',iter,xx1,xx2,xx3,ea(1),ea(2),ea(3));

end

**Datos de entrada:**

Digite un vector fila con los valores iniciales de x0 = [3;2;1]

Tolerancia del sistema = 0.005

Digite la ecuación 1 = 2^x1+x2\*x3-0.3\*x3

Digite la ecuación 2 = x1^2-x2^2-0.5\*x3^2

Digite la ecuación 3 = x1^3+2^x2-x3^3

**Resultados:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Iteración** | **x1** | **x2** | **x3** | **ea(1)** | **ea(2)** | **ea(3)** |
| **0** | 3 | 2 | 1 | 100 | 100 | 100 |
| **1** | 1.7640 | 1.6361 | -0.4602 | 70.0664 | 22.2440 | 317.3010 |
| **2** | 1.3047 | 1.1903 | -0.9753 | 35.2088 | 37.4541 | 52.8170 |
| **3** | 1.2687 | 0.8336 | -0.9309 | 2.8329 | 42.7859 | 4.7773 |
| **4** | 1.3498 | 0.7142 | -1.0066 | 6.0087 | 16.7176 | 7.5216 |
| **5** | 1.4283 | 0.7907 | -1.2806 | 5.4920 | 9.6783 | 21.3977 |
| **6** | 1.5513 | 0.7558 | -1.2555 | 7.9325 | 4.6237 | 2.0008 |
| **7** | 1.5985 | 0.7112 | -1.1715 | 2.9524 | 6.2716 | 7.1687 |
| **8** | 1.6736 | 0.7396 | -1.2667 | 4.4844 | 3.8417 | 7.5183 |
| **9** | 1.6524 | 0.7551 | -1.5420 | 1.2803 | 2.0581 | 17.8539 |
| **10** | **1.7330** | **0.7071** | **-1.3280** | 4.6480 | 6.7918 | 16.1162 |

1. Tabla de iteraciones y resultados finales de x1, x2 y x3