Tarea Computacional de Métodos Numéricos.

Primera Parte

Se establecen los distintos métodos de punto fijo, para los cuales se escribe una función por cada uno, de modo que se llama alguna de ellas para determinar el cero en cierta función determinada. La estructura es simple, primero se establece un punto de partida xo, luego se ingresa la función definida, luego la cantidad de iteraciones. El resultado de la función depende del método utilizado y la cantidad de iteraciones, algunos pueden resultar no convergentes o más lentos que otros.

Primero se establece la función Lf(x), su codigo es:

```
function lfx=lefx(f)

g=diff(f);
h=diff(g);
lfx=f*h/g;
```

A continuación el Codigo de cada método, con la condición límite para detener iteración:

$$\frac{|x_{n+1}-x_n|}{|x_n|} \leqslant 10^{-10}$$

Método Newton

```
function [Xc,no,co]=newton(xo,f,n)
g=diff(f);
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(g);
   xo=x;
   co(:,i)=xo;
   syms x;
   x=xo-fo/go;
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
Xc=x;
```

Método Schroder

```
function [sc,no,co]=schroder(xo,f,n)
g=diff(f);
h=diff(g);
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(g);
   ho=eval(h);
   xo=x;
   co(:,i)=xo;
   syms x;
   x=xo-fo*go/(go^2-fo*ho);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
sc=x;
```

Método Whittaker

```
function [wh,no,co]=whittaker(xo,f,n)
g=diff(f);
h=diff(g);
lfx=lefx(f);
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(g);
   ho=eval(h);
   lfxo=eval(lfx);
   xo=x;
   co(:,i)=xo;
   syms x;
   x=xo-(fo/(2*go))*(2-lfxo);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
wh=x;
```

Método Halley

```
function [ha,no,co]=halley(xo,f,n)
g=diff(f);
h=diff(g);
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(g);
   ho=eval(h);
   xo=x;
   co(:,i)=xo;
   syms x;
   x=xo-2*fo*go/(2*go^2 - fo*ho);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
ha=x;
```

Método Chebysev

```
function [ch,no,co]=chebysev(xo,f,n)
g=diff(f);
lfx=lefx(f);
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(g);
   lfxo=eval(lfx);
   xo=x;
   co(:,i)=xo;
   syms x;
   x=xo-(fo/go)*(1+lfxo/2);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
ch=x;
```

Método SuperHalley

```
function [sha,no,co]=superhalley(xo,f,n)
g=diff(f);
lfx=lefx(f);
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(g);
   lfxo=eval(lfx);
   xo=x;
   co(:,i)=xo;
   syms x;
   x=xo-(fo/(2*go))*(2-lfxo)/(1-lfxo);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
sha=x;
```

Método Stirling

```
function [st,no,co]=stirling(xo,f,n)
g=diff(f);
lfx=lefx(f);
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   co(:,i)=xo;
   x1=xo;
   x=xo-fo;
   go=eval(g);
   lfxo=eval(lfx);
   xo=x1;
   syms x;
   x=xo-(fo/go);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
st=x;
```

Método Steffensen

```
function [ste,no,co]=steffensen(xo,f,n)
syms x;
x=xo;
for i=1:1:n
    fo=eval(f);
    x1=xo;
    co(:,i)=xo;
    x=xo-fo;
    go=eval(f);
    lfxo=eval(lfx);
    xo=x1;
    syms x;
    x=xo-(fo^2/(go-fo));
    m=abs(x-xo)/abs(x);
    no=i;
    if m<=10^-10
        break
    end
end
ste=x;
```

Método Punto Medio

```
function [pm,no,co]=puntomedio(xo,f,n)
g=diff(f);
syms x;
x=xo;
for i=1:1:n
    fo=eval(f);
    go=eval(f);
    co(:,i)=xo;
    x1=xo;
    x=xo-fo/(2*go);
    g1=eval(g);
    xo=x1;
    syms x;
    x=xo-(fo^2/(go-fo));
    m=abs(x-xo)/abs(x);
    no=i;
    if m<=10^-10
        break
    end
end
pm=x;
```

Método Trub-Ostrowsku

```
function [to,no,co]=traubostrowski(xo,f,n)
g=diff(f);
u=f/g;
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(f);
   uo=eval(u);
   x1=xo;
   co(:,i)=xo;
   x=xo-uo;
   f1=eval(f);
   xo=x1;
   syms x;
   x=xo-(fo/(2*go))*(f1-fo)/(2*f1-fo);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
to=x;
```

Método Jarrat.

```
function [ja,no,co]=jarrat(xo,f,n)
g=diff(f);
u=f/g;
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(f);
   uo=eval(u);
   x1=xo;
   co(:,i)=xo;
   x=xo-(2/3)*uo;
   g1=eval(g);
   xo=x1;
   syms x;
   x=xo-(.5)*uo-fo/(go-3*g1);
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
```

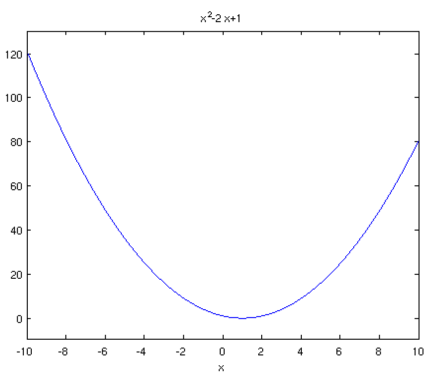
```
end
ja=x;
```

Método Newton Newton

```
function [nn,no,co]=newtonnewton(xo,f,n)
g=diff(f);
u=f/g;
syms x;
x=xo;
for i=1:1:n
   fo=eval(f);
   go=eval(f);
   uo=eval(u);
   x1=xo;
   co(:,i)=xo;
   x=xo-uo;
   f1=eval(f);
   g1=eval(g);
   xo=x1;
   syms x;
   x=xo-uo-f1/g1;
   m=abs(x-xo)/abs(x);
   no=i;
   if m<=10^-10
      break
   end
end
nn=x;
```

A continuación se expresa el algoritmo utilizado para obtener el cero.

1.- Se define la variable x.
   syms x;
2.- Se define una función determinada.
   f=x^2-2*x+1;
3.- Se grafica, para determinar un valor cercano a 0 que será el valor inicial de la iteración.
   ezplot(f,[-10,10])



Dependiendo del valor escogido el método iterativo converge para una cierta cantidad de iteración o diverge según cada caso. Si observamos la función, por intuición debiera converger para cualquier método, escogiendo xo=4, con el método de newton se obtiene.

```
[xo,no,co]=newton(4,f,50)
xo =1.0000

no =30

co =

  Columns 1 through 10

    4.0000    2.5000    1.7500    1.3750    1.1875    1.0938    1.0469    1.0234    1.0117    1.0059

  Columns 11 through 20

    1.0029    1.0015    1.0007    1.0004    1.0002    1.0001    1.0000    1.0000    1.0000    1.0000

  Columns 21 through 30

    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
```

Se observa que el cero encontrado es xo*=1, y la iteración para la cual cumple condición es 30.

De manera similar se procede con el resto de las funciones.

Otra posible opción de programación es reemplazar el "for" del código por un while(true), de esta manera nos evitamos "adivinar" una cantidad de iteraciones que iguale o supere al no.

A continuación se muestra el código programado para obtener en forma fácil el punto de convergencia y la cantidad de iteraciones por método, valores por iteración.

```
function [A,B,C]=analisisf(f,xo,N)

[Xc,no1,co1]=newton(xo,f,N);
[sc,no2,co2]=schroder(xo,f,N);
[wh,no3,co3]=whittaker(xo,f,N);
[ha,no4,co4]=halley(xo,f,N);
[ch,no5,co5]=chebysev(xo,f,N);
[sha,no6,co6]=superhalley(xo,f,N);
[st,no7,co7]=stirling(xo,f,N);
[pm,no8,co8]=puntomedio(xo,f,N);
[to,no9,co9]=traubostrowski(xo,f,N);
[ja,no10,co10]=jarrat(xo,f,N);
[nn,no11,co11]=newtonnewton(xo,f,N);
A(:,1)=[Xc;sc;wh;ha;ch;sha;st;pm;to;ja;nn];
B(:,1)=[no1,no2,no3,no4,no5,no6,no7,no8,no9,no10,no11];

l(:,1)=length(co1);
```

```
l(:,2)=length(co2);
l(:,3)=length(co3);
l(:,4)=length(co4);
l(:,5)=length(co5);
l(:,6)=length(co6);
l(:,7)=length(co7);
l(:,8)=length(co8);
l(:,9)=length(co9);
l(:,10)=length(co10);
l(:,11)=length(co11);
lm=max(l);
Cx=[zeros(1,lm-l(1)) co1 ;zeros(1,lm-l(2)) co2 ;zeros(1,lm-l(3)) co3 ;zeros(1,lm-l(4)) co4
;zeros(1,lm-l(5)) co5 ;zeros(1,lm-l(6)) co6 ;zeros(1,lm-l(7)) co7 ;zeros(1,lm-l(8)) co8;zeros(1,lm-
l(9)) co9 ; zeros(1,lm-l(10)) co10; zeros(1,lm-l(11)) co11 ];
C=Cx;
```

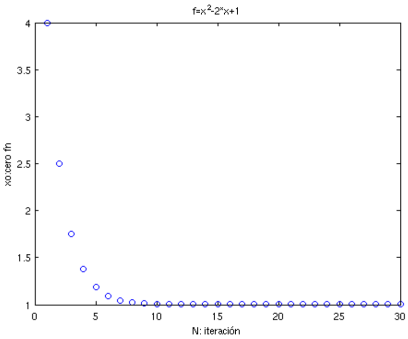
Donde N es la cantidad de veces que se itera con distintos valores de iteración máxima en cada función, determinada por "paso".

Seleccionando xo=2.5, los resultados obtenidos para cada método iterativo son:

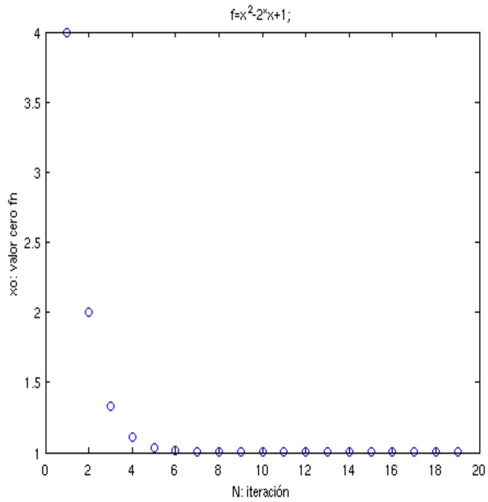
Método	A: Son los valores de Xo cero de función	B: Son los valores de las iteraciones suficientes por método.
1,-Newton	1	29
2,-Schroder	NaN	50
3,-Whittaker	1	32
4,-Halley	1	18
5,-Chebysev	1	26
6,-SuperHalley	3	30
7,-Stirlign	3.219	50
8,-Punto Medio	NaN	50
9,-Traub ostrowski	1.9101	50
10,-Jarrat	-3.2356	50
11,-Newton Newton	1.6	50

Se observa convergencia en los métodos: {1,3,4,5} con cantidad de iteraciones menores a 50:{29,32,18,26} siendo el método más rápido para este caso el Halley. Se indefine con métodos no convergentes para: {2,8} Y para el resto de los métodos la convergencia requiere de mayor cantidad de iteraciones.

Por último se grafica la convergencia al valor cero de la función utilizando método de newton:



Con el método halley se obtuvo:



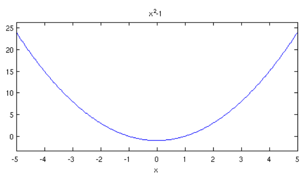
Se observa obviamente una mayor rapidez en la convergencia con el método halley que con el newton.

Para facilitar el estudio se desarrolla un código para graficar todos los métodos:

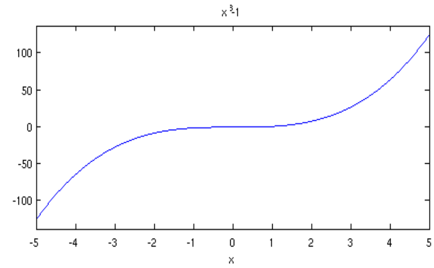
```
function p=plotearf(B,C)
metodos=['    Newton     ';'  Schroder    ';'  Whittaker   ';'    Halley      ';
    '    Chebysev    ';'  Super Halley ';'    Stirling    ';'  Punto Medio  ';
    'Traub ostrowski';'    Jarrat      ';' Newton Newton '];
l=length(B)
p=figure;
for i=1:l
    n=i
    co=length(C(i,:))
    bo=co-B(i,:)+1;
    B(i,1)
    no=[1:1:B(i,1)]
    subplot(3,4,i)
    c=C(i,bo:co)
    plot(no,c,'o')
   title(metodos(i,:))
   xlabel('N: iteraciÃ³n')
   ylabel('Xo: cero de funciÃ³n')
end
```

A continuación se definen las funciones a las cuales se debe calcular el cero, su gráfica y el valor inicial de iteración.

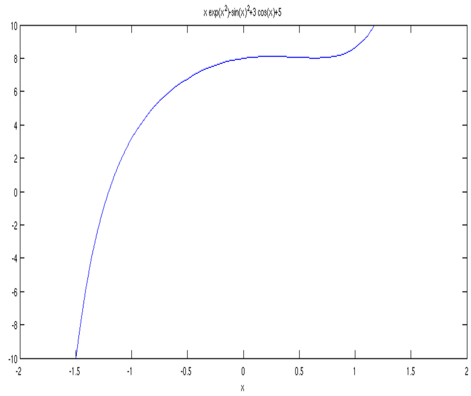
```
syms x;
f1=x^2-1;
```



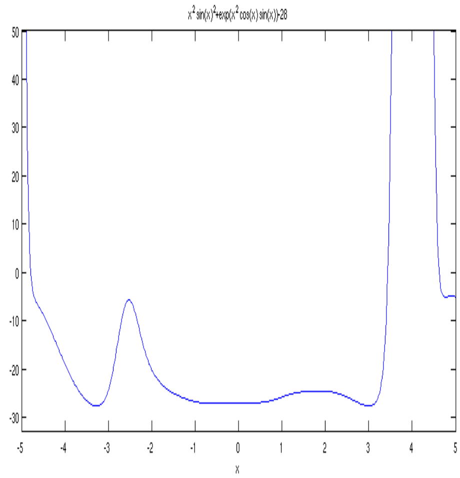
f2=x^3-1;



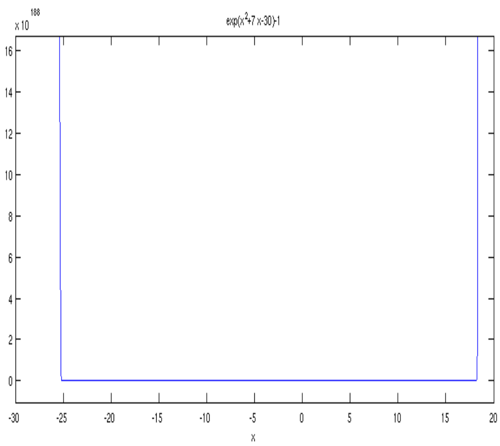
f3=x*exp(x^2)-sin(x)^2+3*cos(x)+5;



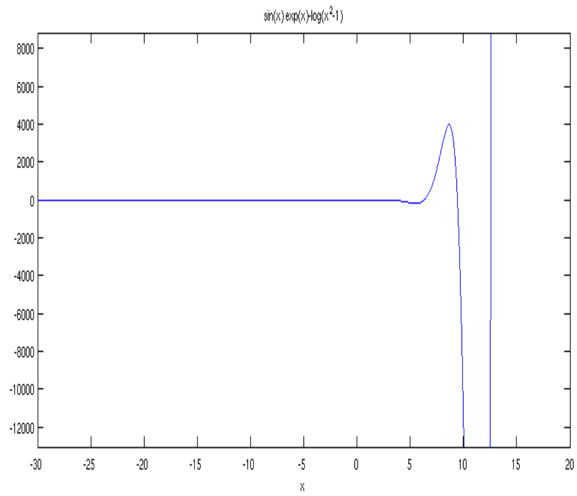
f4=x^2*sin(x)^2+exp(x^2*cos(x)*sin(x))-28;



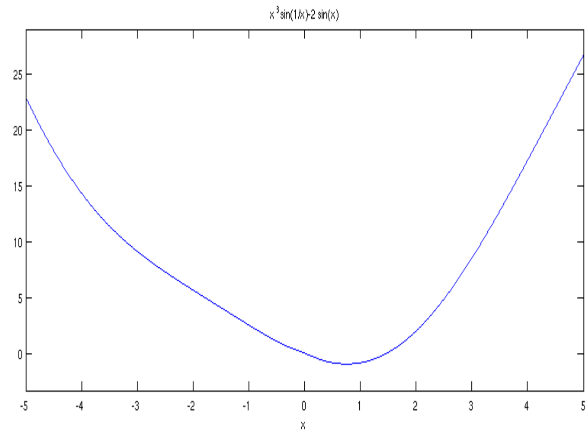
%menor cero positivo
f5=exp(x^2+7*x-30)-1;



f6=sin(x)*exp(x) -log(x^2-1);



f7=x^3 * sin(1/x) -2* sin(x); %continua en 0, equivalente a definida en guia



Se define el vector de condiciones iniciales:

xo=[2, 2.5, -1, 3.6, 3,2, 3.0395 , 1.6]

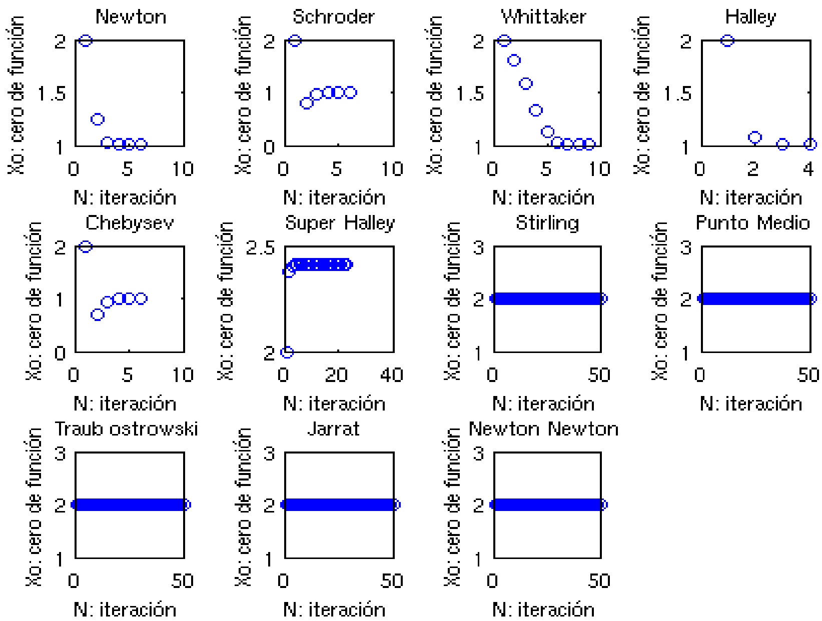
Se definen las siglas:
CR: convergencia rápida, en orden descendente.
CN: No alcanza a converger en la cantidad de iteraciones dispuestas.
D:diverge

Con estas siglas se determinará en forma rápida el nivel de convergencia de cada método por cada función, clasificándolos adecuadamente.

A continuación los resultados y las gráficas de cada función con cada métodos.

1) f1=x^2-1;

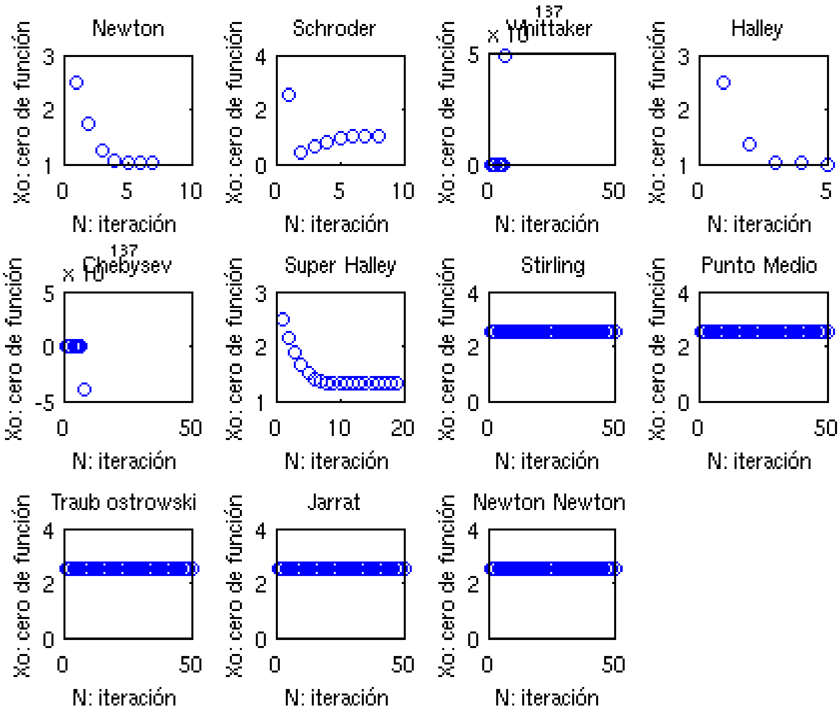
A	B
1	6
1	6
1	9
1	4
1	6
2.4142	23
2.7257	50
NaN	50
1.277	50
3.2646	50
1.1873	50



CR:{4,1,2,3, 5,6}
CN:{7,9,10,11,6}
D:{8}

2) f2=x^3-1;

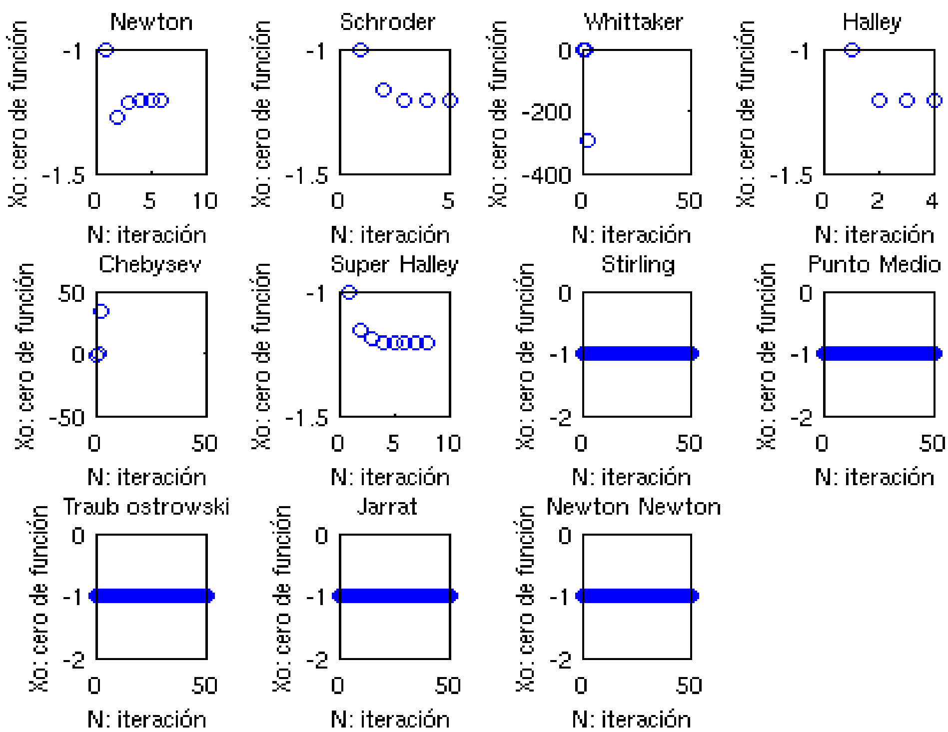
1	7
1	8
NaN	50
1	5
NaN	50
1.3247	19
2.4646	50
NaN	50
1.3246	50
2.9271	50
1.5033	50



CR:{4,1,2,6}
CN:{9,10,11}
D:{3,5,8}

3) f3=x*exp(x^2)-sin(x)^2+3*cos(x)+5;

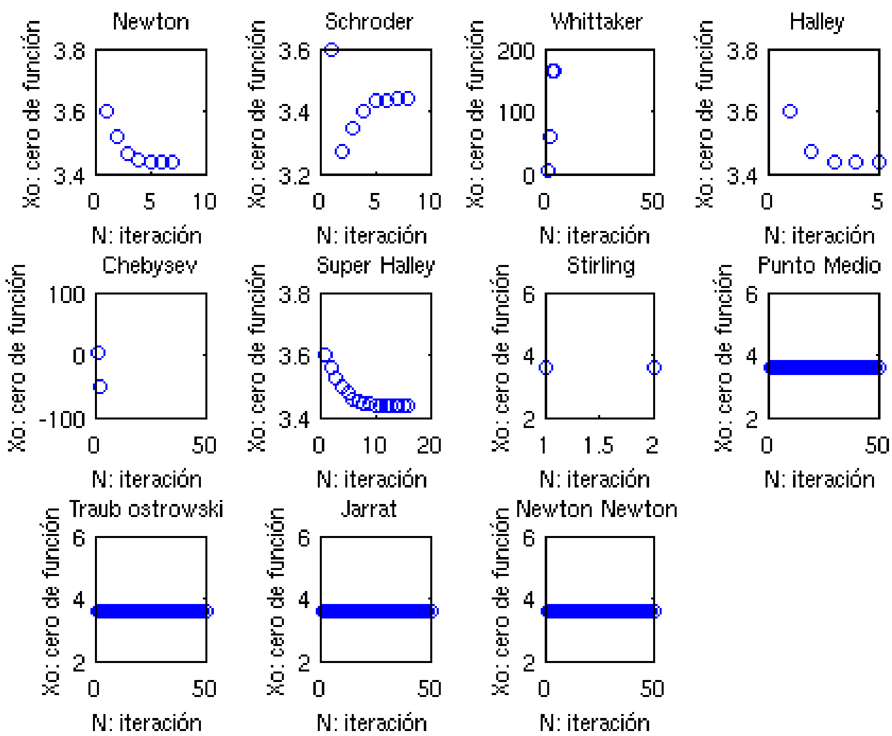
-1.2076	6
-1.2076	5
NaN	50
-1.2076	4
NaN	50
-1.2076	8
-1	50
NaN	50
-1.3053	50
-1.0454	50
-1.4073	50



CR:{1,2,4,6}
CN:{9,10,11}
D:{3,5,8}

4) f4=x^2*sin(x)^2+exp(x^2*cos(x)*sin(x))-28;

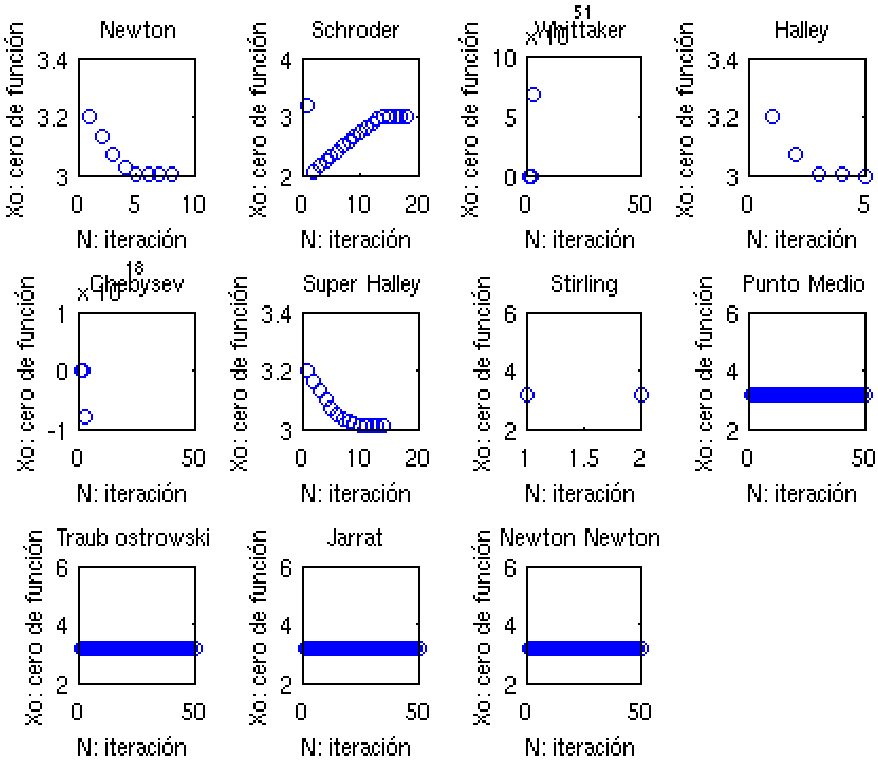
3.4375	7
3.4375	8
NaN	50
3.4375	5
NaN	50
3.4380	16
3.6000	2
NaN	50
3.3471	50
3.6253	50
3.4915	50



CR:{4,1,2,6}
CN:{7,9,10,11}
D:{3,5,8}

5) f5=exp(x^2+7*x-30)-1;

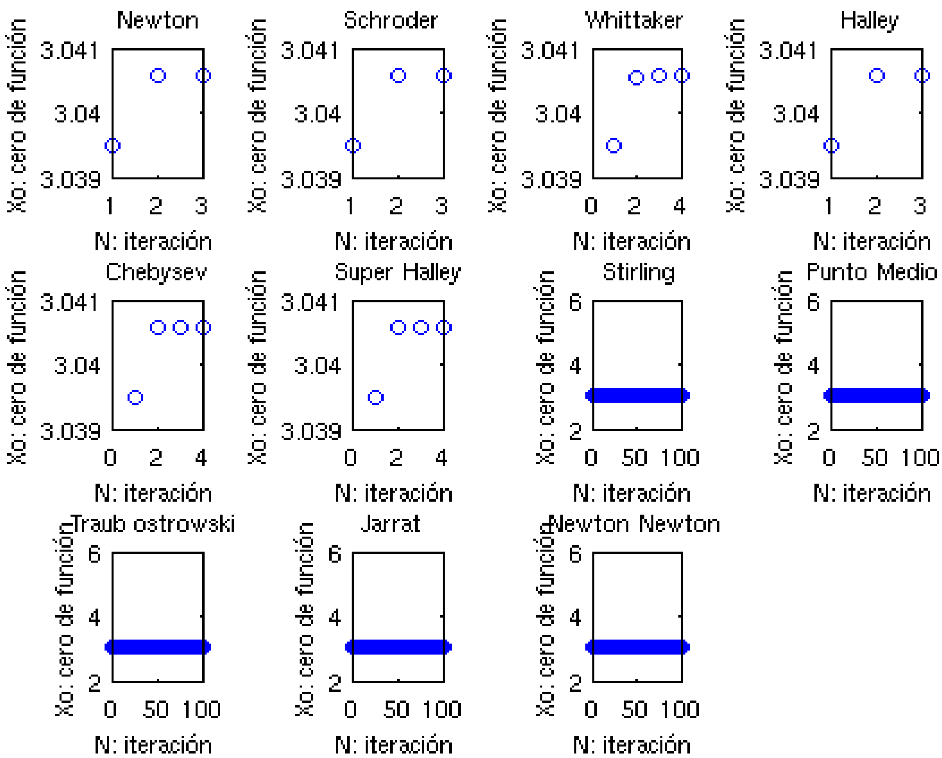
3.0000	8
3.0000	18
NaN	50
3.0000	5
NaN	50
3.0109	14
3.2000	2
NaN	50
NaN	50
NaN	50
3.0846	50



CR:{3,1,2}
CN:{6,7,11}
D:{3,5,8,9,10}

6)f6=sin(x)*exp(x) -log(x^2-1);

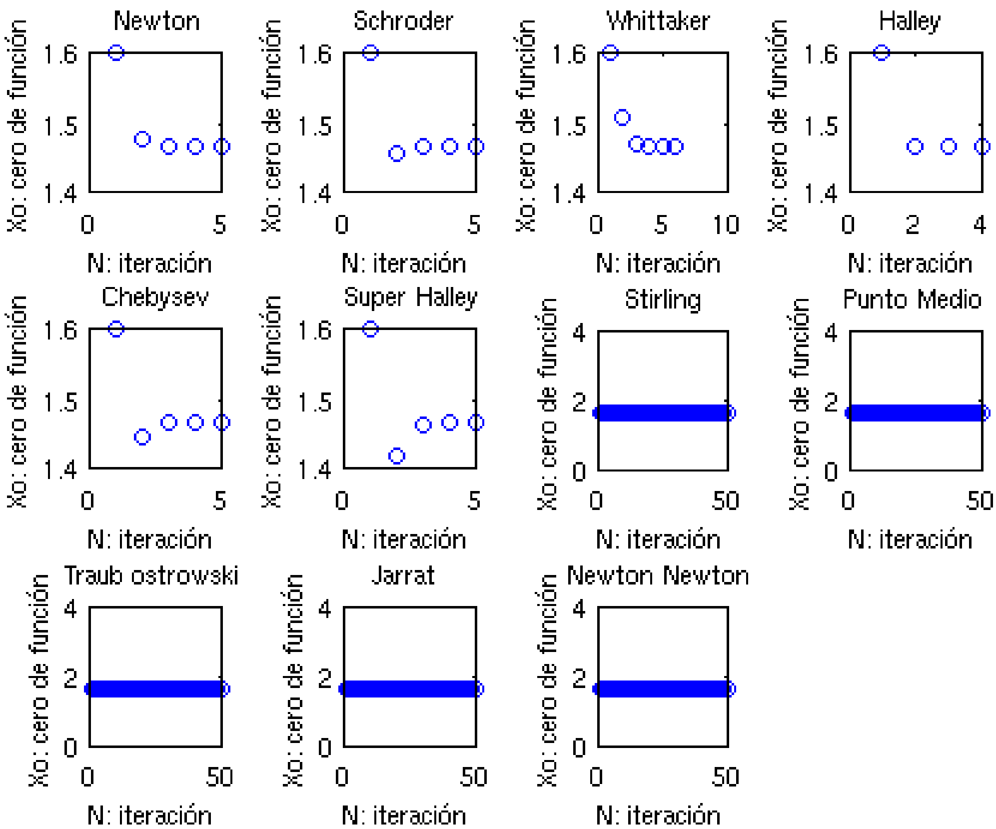
3.0406	3
3.0406	3
3.0406	4
3.0406	3
3.0406	4
3.0406	4
3.0319	50
NaN	50
2.7550	50
3.0397	50
3.0406	50



CR:{1,2,4,3,5,6}
CN:{7,9,10,11}
D:{8}

7) f7=x^3 * sin(1/x) -2* sin(x);

1.4670	5
1.4670	5
1.4670	6
1.4670	4
1.4670	5
1.4670	5
2.3310	50
NaN	50
1.3119	50
1.5862	50
1.4765	50



CR:{4,1,2,5,6,3}
CN:{7,9,10,11}
D:{8}