



Universitat Politècnica de València

FACULTAD DE TELECOMUNICACIONES

TRABAJO ACADÉMICO ANÁLISIS NUMÉRICO



Marc Bridges Fernández, Pau David Brugal
Jesús Gil Olivares, María Jordá Blanes

Junio 2024

Índice de contenidos

Contenidos	Página
Introducción al problema	2
1 Representa gráficamente la función $F(E) = E - e \sin(E) - M$, cuyo cero vamos a buscar numéricamente.	2
2 Utilizando el método de Newton, resuelve la ecuación utilizando los valores, las estimaciones iniciales y el criterio de parada indicados.	3
2.1. Script método de Newton	3
3 Representa las tasas de convergencia lineal y cuadrática utilizando el método de Newton. ¿Qué se puede concluir a partir de ellas?	7
3.1. Script cálculo ACOC	6
4 Utilizando el método de punto fijo, resuelve la ecuación utilizando los valores, las estimaciones iniciales y el criterio de parada indicados.	11
4.1. Script método de punto fijo	10
5 Representa las tasas de convergencia lineal y cuadrática utilizando el método de punto fijo. ¿Qué se puede concluir a partir de ellas?	16
Bibliografía	19

Introducción al problema

Para el estudio del movimiento de cuerpos celestes con órbitas elípticas resulta imprescindible resolver la llamada ecuación de Kepler

$$E - e \sin(E) = M$$

donde la incógnita es la anomalía excéntrica E y son conocidas la excentricidad de la órbita e y la anomalía media M .

Representa gráficamente la función $F(E) = E - e \sin(E) - M$, cuyo cero vamos a buscar numéricamente.

Para la representación gráfica en función de los parámetros M y e se ha utilizado el software Mathematica (orientada al cálculo simbólico). El comando que permite crear una interfaz mediante la cual el usuario puede controlar el valor de tantos parámetros como quiera es "Manipulate". A continuación se muestra el código completo.

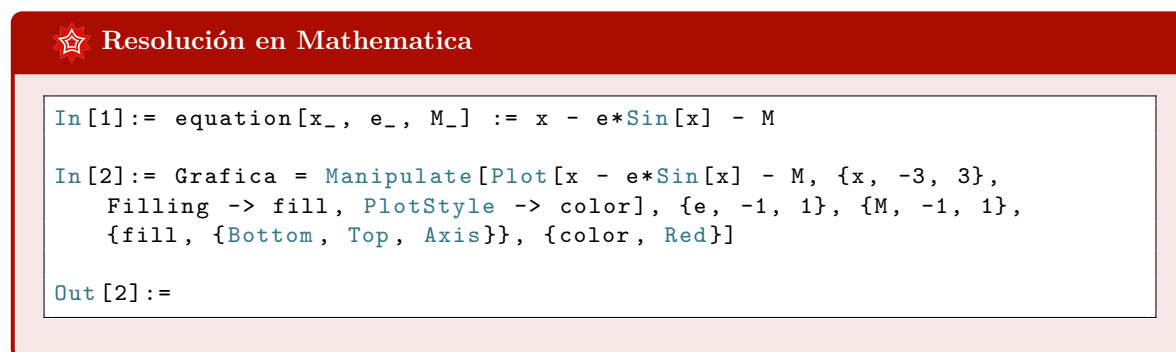


Figure 1: Gráfica interactiva 1

Figure 2: Gráfica interactiva 2

Las imágenes que se muestran a continuación son una muestra de la gráfica interactiva. Para poder apreciar mejor el funcionamiento de la interfaz, así como la variación de la gráfica en función de los parámetros, se ha compartido un vídeo de esta gráfica interactiva vista desde el software Mathematica.

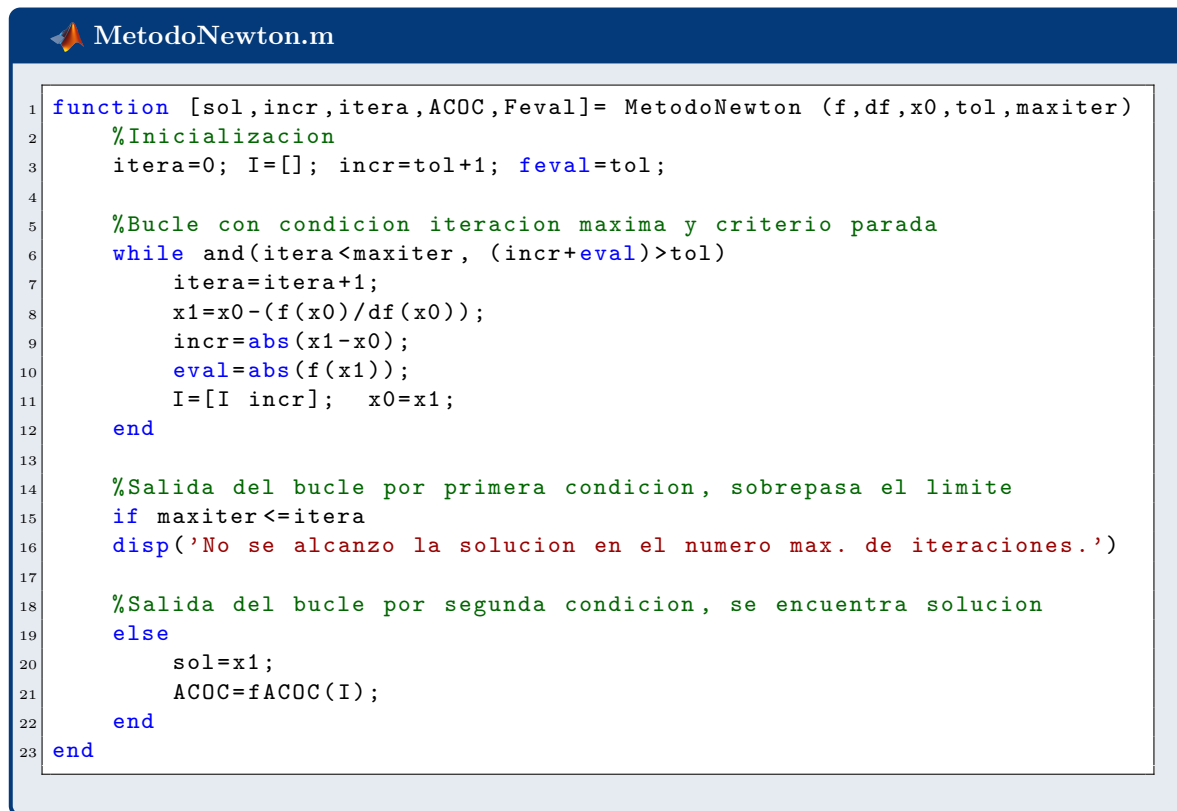
Con esta gráfica, se puede establecer una aproximación inicial bastante certera de la que es la solución a la ecuación para cada e y M posibles. De hecho, si se toman los valores propuestos en los siguientes apartados, y a partir de los que se quiere obtener la solución de la ecuación, se puede comparar el resultado de la representación con los del los cálculos posteriores.

En la figura 2 ya se han tomado los valores especificados en los ejercicios posteriores. A continuación, se vuelve a construir la gráfica, pero esta vez acotando el eje de las x en el intervalo $[-0.2, 0.2]$, para poder observar de forma más exacta el punto en el que se satisface la ecuación.

Figure 3: Gráfica interactiva 3

Utilizando el método de Newton, resuelve la ecuación utilizando los valores, las estimaciones iniciales y el criterio de parada indicados.

Se debe programar un algoritmo que implemente el método de Newton, y que además incorpore el criterio de parada especificado en el ejercicio. A continuación, se presenta el script de este mismo código programado en MatLab.



```
1 function [sol,incr,itera,ACOC,Feval]= MetodoNewton (f,df,x0,tol,maxiter)
2 %Inicializacion
3 itera=0; I=[]; incr=tol+1; feval=tol;
4
5 %Bucle con condicion iteracion maxima y criterio parada
6 while and(itera<maxiter, (incr+eval)>tol)
7     itera=itera+1;
8     x1=x0-(f(x0)/df(x0));
9     incr=abs(x1-x0);
10    eval=abs(f(x1));
11    I=[I incr]; x0=x1;
12 end
13
14 %Salida del bucle por primera condicion, sobrepasa el limite
15 if maxiter<=itera
16     disp('No se alcanzo la solucion en el numero max. de iteraciones.')
```

En el programa, se ingresan los valores de las constantes conocidas, así como la función de la ecuación de Kepler y su derivada.

Para obtener resultados utilizando diferentes estimaciones iniciales, se aplica el algoritmo presentado para cada una de estas estimaciones. Posteriormente, se contruye una tabla que resume los resultados obtenidos durante el proceso. Siendo k el número de iteraciones que se han necesitado para alcanzar la solución ($itera$); E_k la solución a la ecuación (sol); $|E_{k+1} - E_k|$ el incremento entre los dos últimos valores obtenidos ($incr$); y $|F(E_{k+1})|$ la evaluación de la función en el último valor calculado.

Se establece como criterio de parada la condición $|E_{k+1} - E_k| + |F(E_{k+1})| < 5 \cdot 10^{-5}$; que indica que la tolerancia debe fijarse a $5 \cdot 10^{-5}$.

Resolución en MATLAB

```
>> e=0.96727464; M=4.527594e-3; tol=5e-5; maxiter=100;  
>> F = @(E) E - e*sin(E) - M; dF= @(E) 1 - e*cos(E);  
>> E01=1; E02=M; E03=0; E04=(M+e)/2;
```

```

>> %Estimación inicial 1
>> [sol1,incr1,itera1,ACOC1,Feval1]= MetodoNewton (F,dF,E01,tol,maxiter)
sol1 =
    0.1280
incr1 =
    4.9244e-08
itera1 =
     7
ACOC1 =
    1.0987    1.7843    2.3319    2.1486    2.0131
Feval1 =
    1.3444e-16

>> %Estimación inicial 2
>> [sol2,incr2,itera2,ACOC2,Feval2]= MetodoNewton (F,dF,E02,tol,maxiter)
sol2 =
    0.1280
incr2 =
    4.0886e-08
itera2 =
     4
ACOC2 =
    1.5972    2.0124
Feval2 =
    1.2056e-16

>> %Estimación inicial 3
>> [sol3,incr3,itera3,ACOC3,Feval3]= MetodoNewton (F,dF,E03,tol,maxiter)
sol3 =
    0.1280
incr3 =
    4.1547e-08
itera3 =
     4
ACOC3 =
    1.5774    2.0124
Feval3 =
    1.0669e-16

>> %Estimación inicial 4
>> [sol4,incr4, itera4,ACOC4,Feval4]= MetodoNewton (F,dF,E04,tol,maxiter)
sol4 =
    0.1280
incr4 =
    6.9426e-06
itera4 =
     5
ACOC4 =
    2.1608    2.2784    2.0575
Feval4 =
    2.9763e-12

```

Hemos querido añadir una nueva estimación inicial para establecer una última comparación. La aproximación no es nada acertada, pero queríamos poner a prueba la convergencia del método.

```
>> %Estimación extra
>> E0E=300;
>> [solE,incrE, iteraE,ACOCE,FevalE]= MetodoNewton (F,dF,E0E,tol,maxiter)
solE =
    0.1280
incrE =
    2.7993e-08
iteraE =
    11
ACOCE =
    0.3617    0.4052    4.4290    0.1694    0.5255    1.0816
    16.0650    1.3972    1.9879
FevalE =
    3.7297e-17

>> %Comparación de resultados
>> itera=[itera1; itera2; itera3; itera4; iteraE];
>> sol=[sol1; sol2; sol3; sol4; solE];
>> incr=[incr1; incr2; incr3; incr4; incrE];
>> eval=[Feval1; Feval2; Feval3; Feval4; FevalE];
>> E0=[E01; E02; E03; E04; E0E];
>> digits(6)
>> tabla = table(vpa(E0), vpa(itera), vpa(sol), vpa(incr), vpa(eval),
    'VariableNames', 'E0', 'Iteraciones', 'Solución', 'Incremento', 'Evaluación');
>> disp(tabla)
```

E0	Iteraciones	Solución	Incremento	Evaluación
1.0	7.0	0.128023	4.92443e-8	1.34441e-16
0.00452759	4.0	0.128023	4.08858e-8	1.20563e-16
0	4.0	0.128023	4.15473e-8	1.06685e-16
0.485901	5.0	0.128023	0.00000694255	2.97628e-12
300.0	11.0	0.128023	2.79931e-8	3.72966e-17

Tal y como pedía el apartado, finalmente, se recogen los datos en la tabla propuesta, obteniendo los resultados buscados en cada uno de los casos estudiados. Incluso con una estimación inicial $E_0 = 300$ el método de Newton acaba convergiendo a la solución de la ecuación de Kepler.

E_0	k	E_k	$ E_{k+1} - E_k $	$ F(E_{k+1}) $
1	7	0.128023	4.92443e-8	1.34441e-16
M	4	0.128023	4.08858e-8	1.20563e-16
0	4	0.128023	4.15473e-8	1.06685e-16
$\frac{M+e}{2}$	5	0.128023	0.00000694255	2.97628e-12
300	11	0.128023	2.79931e-8	3.72966e-17

Representa las tasas de convergencia lineal y cuadrática utilizando el método de Newton. ¿Qué se puede concluir a partir de ellas?

En el anterior programa, se llamaba a la función fACOC.m. Este script de MatLab está programado para dar como solución el vector del ACOC, así como la representación gráfica de las tasas de convergencia lineal y cuadrática. Al igual que en el apartado anterior, se proporciona el código de la función.

```
fACOC.m

1 function ACOC = fACOC(I)
2 i=1; ACOC=[];
3
4 %Calculo del ACOC
5 while (length(I)>i+1)
6     x0=I(i);
7     x1=I(i+1);
8     x2=I(i+2);
9     sol=log(x2/x1)/log(x1/x0);
10    ACOC=[ACOC sol];
11    i=i+1;
12 end
13
14 %Tasa de convergencia lineal y cuadratica
15 j=1; lineal=[]; cuadratica=[];
16 while (length(I)>j)
17     x0=I(j);
18     x1=I(j+1);
19     sol=x1/x0;
20     lineal=[lineal sol];
21     sol2=x1/x0^2;
22     cuadratica=[cuadratica sol2];
23     j=j+1;
24 end
25
26 %Representacion
27 ejehorizontal = 1:length(lineal);
28 figure,
29 plot(ejehorizontal, lineal, 'o--')
30 hold on
31 plot(ejehorizontal, cuadratica, 'o--')
32 legend('Lineal', 'Cuadratica')
33 xlabel('Iteraciones'), grid
34 axis([ejehorizontal(1) ejehorizontal(end) 0 4])
35 end
```

Al tener todos los datos definidos en Matlab, se vuelve a ejecutar el comando de la función "MetodoNewton" para cada una de las estimaciones iniciales, devolviendo por pantalla la representación gráfica de las tasas de convergencia.

Resolución en MATLAB

```
>> [sol1,incr1,itera1,ACOC1,eval1]= MetodoNewton (F,dF,E01,tol,maxiter);
```

Figure 4: Gráfica $E_0=1$

```
>> [sol2,incr2,itera2,ACOC2,eval2]= MetodoNewton (F,dF,E02,tol,maxiter);
```

Figure 5: Gráfica $E_0=M$

```
>> [sol3,incr3,itera3,ACOC3,eval3]= MetodoNewton (F,dF,E03,tol,maxiter);
```

Figure 6: Gráfica $E_0=0$

```
>> [sol4,incr4,itera4,ACOC4,eval4]= MetodoNewton (F,dF,E04,tol,maxiter);
```

Figure 7: Gráfica $E_0=\frac{M+e}{2}$

```
>> [solE,incrE,iteraE,ACOCE,evalE]= MetodoNewton (F,dF,E0E,tol,maxiter);
```

Figure 8: Gráfica $E_0=300$

CONLUSIONEEEEEES, FALTA POR COMPLETAAAAAR

Como hemos calculado el ACOC en cada una de las aproximaciones y, en cada una de ellas, se ha estabilizado en un valor cercano a 2, se puede reforzar nuestra conclusión acerca del grado de convergencia del método.

Utilizando el método de punto fijo, resuelve la ecuación utilizando los valores, las estimaciones iniciales y el criterio de parada indicados.

En primer lugar, se muestra el script utilizado para calcular la solución mediante el método de punto fijo. Se ha programado el algoritmo teniendo en cuenta el criterio de parada indicado.

```
MetodoPuntoFijo.m

1 function [sol, incr, itera, feval, ACOC] = MetodoPuntoFijo(f, g, x0, tol
  , maxiter)
2 %Inicializacion
3 incr=tol+1; feval=tol; itera=1; I=[x0];
4 while and(itera<maxiter, (incr+feval)>tol)
5     x1=g(x0);
6     incr=abs(x1-x0);
7     itera=itera+1;
8     feval=abs(f(x1));
9     I=[I incr];
10    x0=x1;
11 end
12
13 %Salida del bucle por primera condicion, sobrepasa el limite
14 if maxiter<=itera
15     disp('No se alcanzo la solucion en el numero maximo de
      iteraciones.')
```

Un pequeño inciso sobre el programa. Cuando se implementa el criterio de parada, se utiliza la evaluación sobre f (la función original, utilizada en el Método de Newton), y no sobre la nueva función construida. Esto es porque, si se utiliza la función g , no obtendremos resultados concluyentes. La evaluación de $g(E)$ en x_1 se irá acercando a la solución (pues, como se explica a continuación, para la utilización de este método, contruimos una función $g(E)$ tal que $g(E) = E$; y no a un valor mínimo, que es lo que queremos obtener).

Entonces, la forma de la función del punto fijo es $g(x) = x$, siendo x la variable independiente. Por tanto, debemos elegir una función que, además de cumplir esta condición, converja a la solución. Se recuerda, a continuación, parte del teorema 2, que establece las condiciones de convergencia de dicho método.

Teorema 2

Sea $g : [a, b] \rightarrow \mathbb{R}$ continuamente derivable : $g(x) \in [a, b] \forall x \in [a, b]$. Además, supongamos que $|g'(x)| \leq C < 1 \forall x \in]a, b[$. Entonces, la iteración de punto fijo, converge al único punto fijo \bar{x} para cualquier estimación inicial x_0 .

Se plantean dos posibles funciones para la resolución del problema. Previamente, por el método de Newton, ya se ha obtenido la solución a la ecuación. Entonces, se puede estudiar la convergencia de cada una de ellas.

$$E = e \sin(E) + M \quad E = \arcsin\left(\frac{E - M}{e}\right)$$

Prueba convergencia para la elección de la función

■ $E = e \sin(E) + M \rightarrow g(E) = e \sin(E) + M \rightarrow g'(E) = e \cos(E) \rightarrow g'(E_k) = g'(0.128) \rightarrow g'(E_k) = 0.9594 \rightarrow |g'(E_k)| \leq 1$: convergencia

■ $E = \arcsin\left(\frac{E-M}{e}\right) \rightarrow g(E) = \arcsin\left(\frac{E-M}{e}\right) \rightarrow g'(E) = \frac{e^{-1}}{\sqrt{1-\frac{(E-M)^2}{e^2}}} \rightarrow g'(E_k) = g'(0.128) \rightarrow g'(E_k) = 1.0420 \rightarrow |g'(E_k)| \leq 1$: divergencia

Entonces, se llega a la conclusión de que la primera función es la adecuada para resolver la ecuación mediante el método del punto fijo. Se procede a la obtención de resultados con la utilización de MatLab.

Las constantes, la tolerancia y las estimaciones iniciales no sufren ninguna variación respecto del apartado anterior. Así pues, ingresamos de nuevo en MatLab los mismos datos, junto con la nueva función. A continuación, se utiliza el script para calcular la solución a la ecuación de Kepler mediante el método de punto fijo; y, finalmente, se contruye una tabla comparativa.

Resolución en MATLAB

```
>> e=0.96727464; M=4.527594e-3; tol=5e-5; maxiter=500; E01=1; E02=M; E03=0;
>> E04=(M+e)/2; E0E=300; g = @(E) e*sin(E) + M; f = @(E) E - e*sin(E) - M;

>> %Estimación inicial 1
>> [sol1, incr1, itera1, eval1, ACOC1] = MetodoPuntoFijo(f, g, E01, tol, maxiter)
sol1 =
    0.1286
incr1 =
    2.4770e-05
itera1 =
    148
eval1 =
    2.3761e-05
ACOC1 =
Columns 1 through 13
    0.3058    0.6898    0.7767    0.8237    0.8542    0.8758    ...    0.9429
Columns 14 through 26
    0.9477    0.9519    0.9556    0.9589    0.9620    0.9647    ...    0.9781
Columns 27 through 39
    0.9795    0.9807    0.9819    0.9830    0.9840    0.9849    ...    0.9899

    :

Columns 144 through 146
    0.9595    0.9595    0.9595
```

```

>> %Estimación inicial 2
>> [sol2, incr2, itera2, eval2, ACOC2] = MetodoPuntoFijo(f, g, E02, tol, maxiter)
sol2 =
    0.1274
incr2 =
    2.4990e-05
itera2 =
    134
eval2 =
    2.3976e-05
ACOC2 =
    Columns 1 through 13
         1.0006         1.0011         1.0016         1.0021         1.0025         1.0028         ...         1.0042
    Columns 14 through 26
         1.0043         1.0043         1.0044         1.0044         1.0044         1.0044         ...         1.0041
    Columns 27 through 39
         1.0040         1.0040         1.0039         1.0038         1.0037         1.0036         ...         1.0030
        :

    Columns 118 through 130
         1.0001         1.0001         1.0001         1.0001         1.0001         1.0001         ...         1.0001
    Columns 131 through 132
         1.0001         1.0001

>> %Estimación inicial 3
>> [sol3, incr3, itera3, eval3, ACOC3] = MetodoPuntoFijo(f, g, E03, tol, maxiter)
sol3 =
    0.1274
incr3 =
    2.4990e-05
itera3 =
    135
eval3 =
    2.3976e-05
ACOC3 =
    Columns 1 through 13
         0          1.0006         1.0011         1.0016         1.0021         1.0025         ...         1.0041
    Columns 14 through 26
         1.0042         1.0043         1.0043         1.0044         1.0044         1.0044         ...         1.0042
    Columns 27 through 39
         1.0041         1.0040         1.0040         1.0039         1.0038         1.0037         ...         1.0031
        :

    Columns 118 through 130
         1.0002         1.0001         1.0001         1.0001         1.0001         1.0001         ...         1.0001
    Columns 131 through 133
         1.0001         1.0001         1.0001

```

```

>> %Estimación inicial 4
>> [sol4, incr4, itera4, eval4, ACO4] = MetodoPuntoFijo(f, g, E04, tol, maxiter)
sol4 =
    0.1286
incr4 =
    2.5478e-05
itera4 =
    141
eval4 =
    2.4440e-05
ACO4 =
Columns 1 through 13
    0.0531    0.9085    0.9183    0.9265    0.9334    0.9393    ...    0.9655
Columns 14 through 26
    0.9679    0.9701    0.9721    0.9739    0.9756    0.9771    ...    0.9852
Columns 27 through 39
    0.9861    0.9868    0.9876    0.9883    0.9889    0.9895    ...    0.9928
    :

Columns 118 through 130
    0.9998    0.9998    0.9998    0.9998    0.9998    0.9998    ...    0.9999
Columns 131 through 139
    0.9999    0.9999    0.9999    0.9999    0.9999    0.9999    ...    0.9999

>> %Estimación inicial extra
>> [solE, incrE, iteraE, evalE, ACOE] = MetodoPuntoFijo(f, g, E0E, tol, maxiter)
solE =
    0.1274
incrE =
    2.5506e-05
iteraE =
    180
evalE =
    2.4471e-05
ACOE =
Columns 1 through 13
   -2329.0    0.0644    0.6933    0.7711    0.8142    0.8424    ...    0.9214
Columns 14 through 26
    0.9717    0.9741    0.9763    0.9784    0.9805    0.9824    ...    0.9692
Columns 27 through 39
    0.9741    0.9763    0.9784    0.9805    0.9824    0.9843    ...    0.9954
    :

Columns 157 through 169
    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    ...    1.0000
Columns 170 through 178
    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    ...    1.0000

```

```

>> %Comparación de resultados
>> itera=[itera1; itera2; itera3; itera4; iteraE];
>> sol=[sol1; sol2; sol3; sol4; solE];
>> incr=[incr1; incr2; incr3; incr4; incrE];
>> eval=[eval1; eval2; eval3; eval4; evalE];
>> E0=[E01; E02; E03; E04; E0E];
>> digits(6)
>> tabla = table(vpa(E0), vpa(itera), vpa(sol), vpa(incr), vpa(eval),
    'VariableNames', 'E0', 'Iteraciones', 'Solución', 'Incremento', 'Evaluación');
>> disp(tabla)

```

E0	Iteraciones	Solución	Incremento	Evaluación
1.0	148.0	0.128607	0.0000247695	0.000023761
0.00452759	134.0	0.127433	0.0000249898	0.000023976
0	135.0	0.127433	0.0000249898	0.000023976
0.485901	141.0	0.128624	0.0000254776	0.0000244402
300.0	180.0	0.12742	0.0000255058	0.0000244712

Como en el método de Newton, se recogen en el formato de tabla propuesto en el apartado:

E_0	k	E_k	$ E_{k+1} - E_k $	$ F(E_{k+1}) $
1	148	0.128607	0.0000247695	0.000023761
M	134	0.127433	0.0000249898	0.000023976
0	135	0.127433	0.0000249898	0.000023976
$\frac{M+e}{2}$	141	0.128624	0.0000254776	0.0000244402
300	180	0.12742	0.0000255058	0.0000244712

Representa las tasas de convergencia lineal y cuadrática utilizando el método de punto fijo. ¿Qué se puede concluir a partir de ellas?

Al igual que en el anterior método, la función del método de punto fijo llamaba a la función fACOC. Por tanto, con ella obtenemos el cálculo del vector ACOC, y de las tasas de convergencia lineal y cuadrática. Se vuelve a ejecutar el comando de la función "MetodoPuntoFijo" para cada una de las estimaciones iniciales, devolviendo por pantalla la representación gráfica de las tasas de convergencia.



Resolución en MATLAB

```
>> [sol1, incr1, itera1, eval1, ACOC1] = MetodoPuntoFijo(f, g, E01, tol, maxiter)
```

Figure 9: Gráfica $E_0=1$

```
>> [sol2, incr2, itera2, eval2, ACOC2] = MetodoPuntoFijo(f, g, E02, tol, maxiter)
```

Figure 10: Gráfica $E_0=M$

```
>> [sol3, incr3, itera3, eval3, ACOC3] = MetodoPuntoFijo(f, g, E03, tol, maxiter)
```

Figure 11: Gráfica $E_0=0$

```
>> [sol4, incr4, itera4, eval4, ACOC4] = MetodoPuntoFijo(f, g, E04, tol, maxiter)
```

Figure 12: Gráfica $E_0=\frac{M+e}{2}$

```
>> [solE, incrE, iteraE, evalE, ACOCE] = MetodoPuntoFijo(f, g, E0E, tol, maxiter)
```

Figure 13: Gráfica $E_0=300$

Se ha querido mostrar la totalidad de la representación de las tasas y no se puede apreciar con claridad en qué valor se estabiliza la tasa de convergencia lineal. Sin embargo, si ampliamos las gráficas, se puede observar una situación similar a la observada en la Figura 14. Ésta corresponde a la primera de las estimaciones, pero la situación se repite en cada una de las otras.

Figure 14: Gráfica $E_0=1$

De las gráficas podemos llegar a la conclusión de que el método no presenta convergencia cuadrática para ninguna de las estimaciones. Su tasa de convergencia no se estabiliza.

En cuanto

Bibliografía

Transparencias proporcionadas por Francisco I. Chicharro