



# Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey  
Campus Estado de México

Modalidad Flexible Digital

## **Métodos Numéricos en Ingeniería**

Adolfo Centeno Tellez

Reporte Técnico

Trabajo por:

*Isabel Gonzalez Lezama A01746586*

Septiembre, 2021

## **Introducción**

¿Para qué sirven las ecuaciones? En realidad la aplicación de los polinomios en la vida diaria es de suma importancia. Se puede aplicar en el área de construcción, para el pronóstico del clima, para el cálculo en finanzas, al realizar alguna compra, etc. Emplearemos un ejemplo que se ocupa en la vida diaria: la velocidad. Siendo más específicos, la velocidad constante.

Imagina que eres un astronauta en la Estación Espacial Internacional. Estás arreglando unos paneles solares, cuando de pronto al presionar tu alguna herramienta que estés ocupando sale disparada de tus manos. Si no lo atrapas a tiempo, ésta estará viajando por el espacio en línea recta y a velocidad constante, a menos que algo se interponga en su camino. Esto sucede porque la herramienta se mueve con movimiento rectilíneo uniforme. Esto es uno de los pocos ejemplos que podemos encontrar, así que resolveremos uno que sí pase en la tierra y a mortales como nosotros.

Este problema consistirá en una ecuación por los automóviles igualando a 300km de distancia entre ellos. Se resolverá la ecuación con los tres métodos vistos: Método secante, método bisección y Newton-Raphson. Se obtendrá una una aproximación de la raíz de la ecuación.

## **Descripción del problema a resolver**

Dos ciudades A y B distan 300 km entre sí. A las 9 de la mañana parte de la ciudad A un coche hacia la ciudad B con una velocidad de 90 km/h, y de la ciudad B parte otro hacia la ciudad A con una velocidad de 60 km/h. Hallar el tiempo que tardarán en encontrarse; la hora del encuentro; la distancia recorrida por cada uno.

La ecuación deducida por los datos proporcionados es:

$$e_{AC} = 90t ; e_{CB} = 60t.$$

Sabemos que el espacio recorrido por el primer coche más el espacio recorrido por el segundo es igual a 300 km. Por lo tanto quedará :

$$e_{AC} + e_{CB} = 300;$$

$$90t + 60t = 300 ;$$

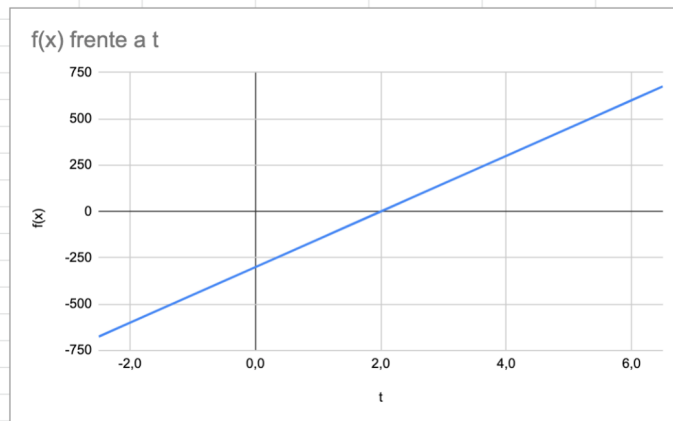
$$F(x): 90t + 60t - 300 = 0,$$

Mediante el método de bisección, se obtendrá las raíces a partir de un intervalo inicial. En el método de secante, se utilizará una serie de raíces para aproximar la raíz de la función. Y por último el método de Newton-Raphson donde se busca un cero en la función.

## Resultados

### Metodo de biseccion (Excel )

f(x) = 90t + 60t - 300									
t	f(x)								
-10,0	-1800								
-9,5	-1725								
-9,0	-1650								
-8,5	-1575	iteracione	xi	xu	xr ( xi + xu ) / 2	f(xi)	f(xr)	f(xi)f(xr)	e
-8,0	-1500	1	1	3	2	-150	0	0,000000	encontrado
-7,5	-1425	2	1	2	1,5	-150	-75	11250,000000	error
-7,0	-1350	3	1,5	2	1,75	-75	-37,5	2812,500000	error
-6,5	-1275	4	1,75	2	1,875	-37,5	-18,75	703,125000	error
-6,0	-1200	5	1,75	1,875	1,8125	-37,5	-28,125	1054,687500	error
-5,5	-1125	6	1,8125	1,8125	1,8125	-28,125	-28,125	791,015625	error
-5,0	-1050	7	1,8125	1,8125	1,8125	-28,125	-28,125	791,015625	error
-4,5	-975								
-4,0	-900								
-3,5	-825								
-3,0	-750								
-2,5	-675								
-2,0	-600								
-1,5	-525								
-1,0	-450								
-0,5	-375								
0,0	-300								
0,5	-225								
x0	1,0	-150							
	1,5	-75							
	2,0	0							
	2,5	75							
xu	3,0	150							
	3,5	225							



## Metodo de biseccion (MatLab)

```

1 %Biseccion PKUYELIU
2 %Isabel Gonzalez Lezama
3 %'90*x+ 60*x - 300'
4 clear, clc
5 h = input('ingrese funcion'); %captura la funcion ''
6 f = inline(h); % la convierte a f(x)
7 a = input('limite inferior '); %captura lim inferior (1)
8 b = input('limite superior '); %captura lim superior (1.6)
9 tol = input ('tolerancia '); %tolerancia 0.0001
10
11 c= 0;
12 n=0;
13 m=(b-a)/2; % calculo de primer punto medio
14
15 fprintf ('\t n \t a ');
16 while (m> tol)
17     c = (a + b ) /2 ; %calculo del punto medio (xr)
18     disp([n, a, c, b, m]); %imprime un set de variables
19     if ((f(a) * f(c)) < 0 ) %se calcula f(xr), y se condiciona <0
20         b = c;
21     else
22         a = c;
23     end
24
25     m= (b-a) / 2; %siguiente punto medio
26     n = n + 1;
27 end

```

ingrese funcion

'90\*x+ 60\*x - 300'

limite inferior

1

limite superior

2

tolerancia

0.0001

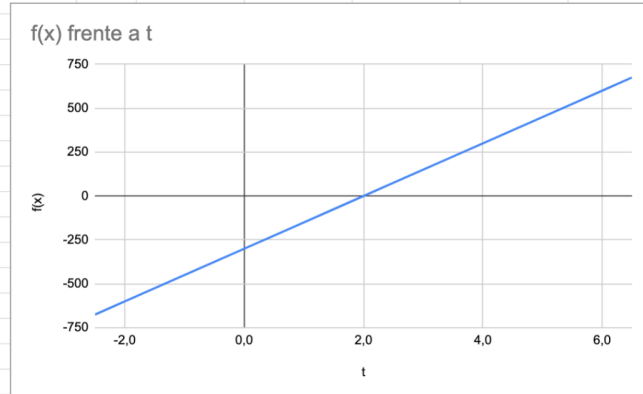
	n	a	0	1.0000	1.5000	2.0000	0.5000
1.0000	1.5000	1.7500	2.0000	0.2500			
2.0000	1.7500	1.8750	2.0000	0.1250			
3.0000	1.8750	1.9375	2.0000	0.0625			
4.0000	1.9375	1.9688	2.0000	0.0312			
5.0000	1.9688	1.9844	2.0000	0.0156			
6.0000	1.9844	1.9922	2.0000	0.0078			
7.0000	1.9922	1.9961	2.0000	0.0039			
8.0000	1.9961	1.9980	2.0000	0.0020			
9.0000	1.9980	1.9990	2.0000	0.0010			
10.0000	1.9990	1.9995	2.0000	0.0005			
11.0000	1.9995	1.9998	2.0000	0.0002			
12.0000	1.9998	1.9999	2.0000	0.0001			

raiz 0.000100

1.999878

## Método Secante(Excel)

A	B	C	D	E	F	G	H	I	J	K	L
		$f(x) = 90t + 60t - 300$									
	t	f(x)								< 0 xu=xr	
	-10,0	-1800								> 0 xi=xr	
	-9,5	-1725		iter	Xi	X0	f(Xi)	f(X0)	X (i+1)	Error f(Xa)	
	-9,0	-1650		1	3	1	150	-150	2,0000	-0,00500000	error
	-8,5	-1575		2	2,0000	3	0	150	2,0000	0,00000000	Encontrado
	-8,0	-1500		3	2,0000	2,0000	0	0	#DIV/0!	#DIV/0!	#ERROR!
	-7,5	-1425									
	-7,0	-1350									
	-6,5	-1275									
	-6,0	-1200									
	-5,5	-1125									
	-5,0	-1050									
	-4,5	-975									
	-4,0	-900									
	-3,5	-825									
	-3,0	-750									
	-2,5	-675									
	-2,0	-600									
	-1,5	-525									
	-1,0	-450									
	-0,5	-375									
	0,0	-300									
	0,5	-225									
x0	1,0	-150									
	1,5	-75									
	2,0	0									
	2,5	75									
xi	3,0	150									
	3,5	225									



## Método Secante(MatLab)

```

secante.m x +
1 %Isabel Gonzalez Lezama Proyecto
2 % '90*x+ 60*x - 300'
3
4 clear, clc
5 cf = input('ingrese funcion = '); %captura funcion en c como string '4*x**2 -5*x'
6 f = inline(cf); %transforma un texto a una funcion que se puede evaluar
7 x0 = input('limite inferior = '); %captura lim inferior (1)
8 x1 = input('limite superior = '); %captura lim superior (1.6)
9 tol = input('tolerancia = '); %tolerancia (0.001)
10
11 error = 100; % variable que almacena el error actual se inicializa
12
13 n=0; %contador de iteraciones
14 fprintf(' n   x0   x1   x2   \t error \n'); %imprime el encabezado de la tabla
15
16 while(error > tol) %hacer mientras el error actual sea mayor para la tolerancia
17
18     x2 = x1 - (x1-x0) * f(x1) / (f(x1) - f(x0));
19
20     error = abs(f(x2));
21     fprintf(' %i   %4.4f   %4.4f   %4.4f   %4.4f \n', n, x0, x1, x2, error);
22
23     x0 = x1;
24     x1 = x2;
25     n = n + 1;
26 end
27
Command Window
0 1.0000 3.0000 2.0000 0.0000
raiz = 2.000000
>>

```

secante.m

Command Window

ingrese funcion =

'90\*x+ 60\*x - 300'

limite inferior =

1

limite superior =

3

tolerancia =

0.0001

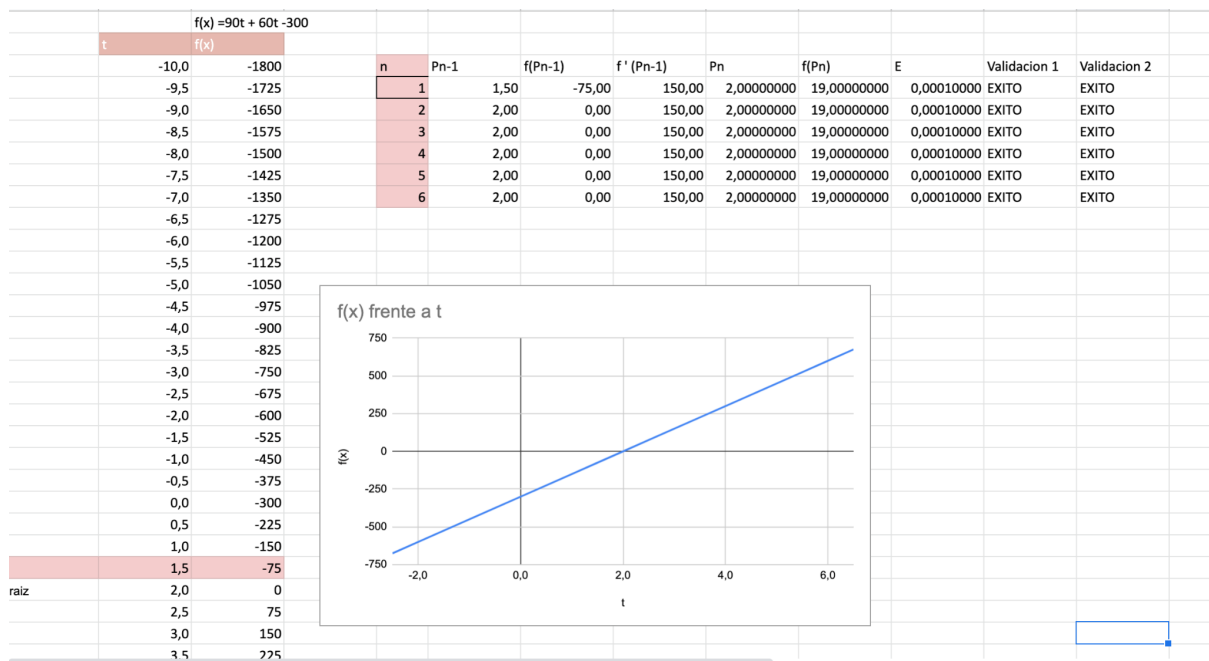
n	x0	x1	x2	error
---	----	----	----	-------

0	1.0000	3.0000	2.0000	0.0000
---	--------	--------	--------	--------

raiz = 2.000000

>>

### Método de Newton-Raphson(Excel)



### Método de Newton-Raphson(MatLab)

```
NewthonRap.m x +
1 clear, clc
2 f = input('f(x)= ', 's'); %'90*x+ 60*x - 300'
3 sf = str2sym(f); %convierte un string en una funcion
4 tol = input('tolerancia del metodo = '); %tolerancia
5 x0 = input('valor inicial = '); %valor inicial, ejemplo: -3
6 v = symvar(sf); %extrae las variables de la funcion
7 f1 = diff(sf); %calcula la derivada de una funcion
8 sf
9 v
10 f1
11
12 y=subs(sf,v,x0)
13 z=subs(f1,v,x0)
14 y
15 z
16
17 sw = 0;
18 while (sw==0)
19     %subs es una funcion que reemplaza valores constantes en las variables
20     %de una funcion
21     x1 = x0 - (subs(sf, v, x0) / subs(f1, v, x0));
22     %x1 = x0 - (f(x0)/f'(x0))
23     %x1 = x0 - (90*x0+60*x0-300)/(90+60)
24     %x1 = x0 - (90*(-3)+60*(-3)-300)/(90+60)
25     %x1 = x0 - (-150-180-300)/150
26     %x1 = x0 - (-430)/150
27     %x1 = x0 + 2.8666666666666667
28     %x1 = -3 + 2.8666666666666667
29     %x1 = -0.13333333333333333
30     %x1 = -0.13333333333333333
31     %x1 = -0.13333333333333333
32     %x1 = -0.13333333333333333
33     %x1 = -0.13333333333333333
34     %x1 = -0.13333333333333333
35     %x1 = -0.13333333333333333
36     %x1 = -0.13333333333333333
37     %x1 = -0.13333333333333333
38     %x1 = -0.13333333333333333
39     %x1 = -0.13333333333333333
40     %x1 = -0.13333333333333333
41     %x1 = -0.13333333333333333
42     %x1 = -0.13333333333333333
43     %x1 = -0.13333333333333333
44     %x1 = -0.13333333333333333
45     %x1 = -0.13333333333333333
46     %x1 = -0.13333333333333333
47     %x1 = -0.13333333333333333
48     %x1 = -0.13333333333333333
49     %x1 = -0.13333333333333333
50     %x1 = -0.13333333333333333
51     %x1 = -0.13333333333333333
52     %x1 = -0.13333333333333333
53     %x1 = -0.13333333333333333
54     %x1 = -0.13333333333333333
55     %x1 = -0.13333333333333333
56     %x1 = -0.13333333333333333
57     %x1 = -0.13333333333333333
58     %x1 = -0.13333333333333333
59     %x1 = -0.13333333333333333
60     %x1 = -0.13333333333333333
61     %x1 = -0.13333333333333333
62     %x1 = -0.13333333333333333
63     %x1 = -0.13333333333333333
64     %x1 = -0.13333333333333333
65     %x1 = -0.13333333333333333
66     %x1 = -0.13333333333333333
67     %x1 = -0.13333333333333333
68     %x1 = -0.13333333333333333
69     %x1 = -0.13333333333333333
70     %x1 = -0.13333333333333333
71     %x1 = -0.13333333333333333
72     %x1 = -0.13333333333333333
73     %x1 = -0.13333333333333333
74     %x1 = -0.13333333333333333
75     %x1 = -0.13333333333333333
76     %x1 = -0.13333333333333333
77     %x1 = -0.13333333333333333
78     %x1 = -0.13333333333333333
79     %x1 = -0.13333333333333333
80     %x1 = -0.13333333333333333
81     %x1 = -0.13333333333333333
82     %x1 = -0.13333333333333333
83     %x1 = -0.13333333333333333
84     %x1 = -0.13333333333333333
85     %x1 = -0.13333333333333333
86     %x1 = -0.13333333333333333
87     %x1 = -0.13333333333333333
88     %x1 = -0.13333333333333333
89     %x1 = -0.13333333333333333
90     %x1 = -0.13333333333333333
91     %x1 = -0.13333333333333333
92     %x1 = -0.13333333333333333
93     %x1 = -0.13333333333333333
94     %x1 = -0.13333333333333333
95     %x1 = -0.13333333333333333
96     %x1 = -0.13333333333333333
97     %x1 = -0.13333333333333333
98     %x1 = -0.13333333333333333
99     %x1 = -0.13333333333333333
100    %x1 = -0.13333333333333333
101    %x1 = -0.13333333333333333
102    %x1 = -0.13333333333333333
103    %x1 = -0.13333333333333333
104    %x1 = -0.13333333333333333
105    %x1 = -0.13333333333333333
106    %x1 = -0.13333333333333333
107    %x1 = -0.13333333333333333
108    %x1 = -0.13333333333333333
109    %x1 = -0.13333333333333333
110    %x1 = -0.13333333333333333
111    %x1 = -0.13333333333333333
112    %x1 = -0.13333333333333333
113    %x1 = -0.13333333333333333
114    %x1 = -0.13333333333333333
115    %x1 = -0.13333333333333333
116    %x1 = -0.13333333333333333
117    %x1 = -0.13333333333333333
118    %x1 = -0.13333333333333333
119    %x1 = -0.13333333333333333
120    %x1 = -0.13333333333333333
121    %x1 = -0.13333333333333333
122    %x1 = -0.13333333333333333
123    %x1 = -0.13333333333333333
124    %x1 = -0.13333333333333333
125    %x1 = -0.13333333333333333
126    %x1 = -0.13333333333333333
127    %x1 = -0.13333333333333333
128    %x1 = -0.13333333333333333
129    %x1 = -0.13333333333333333
130    %x1 = -0.13333333333333333
131    %x1 = -0.13333333333333333
132    %x1 = -0.13333333333333333
133    %x1 = -0.13333333333333333
134    %x1 = -0.13333333333333333
135    %x1 = -0.13333333333333333
136    %x1 = -0.13333333333333333
137    %x1 = -0.13333333333333333
138    %x1 = -0.13333333333333333
139    %x1 = -0.13333333333333333
140    %x1 = -0.13333333333333333
141    %x1 = -0.13333333333333333
142    %x1 = -0.13333333333333333
143    %x1 = -0.13333333333333333
144    %x1 = -0.13333333333333333
145    %x1 = -0.13333333333333333
146    %x1 = -0.13333333333333333
147    %x1 = -0.13333333333333333
148    %x1 = -0.13333333333333333
149    %x1 = -0.13333333333333333
150    %x1 = -0.13333333333333333
151    %x1 = -0.13333333333333333
152    %x1 = -0.13333333333333333
153    %x1 = -0.13333333333333333
154    %x1 = -0.13333333333333333
155    %x1 = -0.13333333333333333
156    %x1 = -0.1333
```

## Conclusiones

Con este proyecto se puede concluir que estos métodos son de gran utilidad para casi cualquier área de la ingeniería. Los tres métodos pueden tener varias aplicaciones en diferentes campos, como en este caso para la ciencia, para problemas sencillos o muy complejos. Los problemas resueltos tiene la ventaja de tener un margen de error mínimo ya que gracias a la programación nosotros mismos determinamos la tolerancia que queramos y eso es sumamente importante, ya que si lo hiciéramos a mano todos estos cálculos tardaremos mucho tiempo y muy posiblemente cometamos muchos errores en su cálculo.

También se pudo concluir que el método y la ingeniería más específicamente en ciencias como la física tiene una relación unida ya que si tomamos en cuenta que cada vez que se desarrolle un nuevo método de cálculo dentro de lo que son los métodos numéricos los cálculos en trayectorias, velocidades en cuestión de tiempo o incluso como el ejemplo de espacial, en alguna trayectoria de algún cohete que resulta interesante saber la relación de masa y velocidad respecto a su función se podrían volver más exactos, más veloces o en algunas ocasiones ambas.