CÁLCULO NUMÉRICO Y COMPUTACIÓN.

MATERIAL PARA CLASES DE TEORÍA.

El Material para Clases de Teoría, es el conjunto de diapositivas con el que se desarrollarán las clases de Teoría. Está ordenado según el cronograma a seguir en e presente año lectivo. En general No incluye las demostraciones a desarrollar. Pero si tiene muchos de los gráficos y tablas con las que se desarrollan las clases de teoría. En general las tablas están asociadas a ejercicios o conceptos de síntesis; y suelen estar en blanco para sean completadas durante las clases.

Los temas desarrollados son los siguientes:

Algoritmia. Conceptos Básicos
Solución Numérica de Raíces de Ecuaciones No Lineales
Sistemas de Ecuaciones Lineales
Valores y Vectores Propios
Interpolación y Aproximación de funciones discretas
Integración Numérica
Derivación Numérica, con aplicación a la solución de Ecuaciones Diferenciales con Valores de Contorno
Solución Numérica de Ecuaciones Diferenciales Ordinarias con Valores Iniciales

El ordenamiento elegido está según el cronograma del año 2016.

ALGORITMOS

- *Definición de Algoritmo*
- *Ejemplo*
- *Pseudo código, Sintaxis de MATLAB*
- *Variables, definición y tipos*
- *Estructuras Algorítmicas* *Secuencial*
  - *Decisión*
  - *Variar*
  - *Mientras*
- *Ejercicios*
- *Tips*
- *Subprogramas en MATLAB*

ALGORITMO

Es la descripción de un procedimiento en forma ordenada y sistemática en un número finito de pasos, líneas o comandos.

Se puede hacer mediante **Pseudo código**; Diagrama de Flujo; Diagrama de Bloques o **Código de Programación**

Ejemplo: Algoritmo para calcular el promedio de dos números

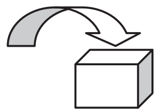
Pseudo código	Elementos	Sintaxis de MATLAB
`Programa` prom_num	Nombre	`function prom_num`
`Reales (xa; xb; prom)`	Declaración de variables	`% Reales (xa; xb; prom)`

Escribir "Ingrese el primer número"
Leer xa
Escribir "Ingrese el segundo número"
Leer xb
	Ingreso de datos	`disp('Ingrese el número 1:');`
		`xa = input('');`
		`disp('Ingrese el número 2:');`
		`xb = input('');`

prom ← (xa+xb)/2	Proceso	`prom = (xa+xb)/2;`
Escribir "el Promedio es: " Escribir prom	Entrega de resultados	`disp('el Promedio es:');` `disp(prom);`
`Fin`	Final	`end`

ALGORITMO. Definición y Tipos de Variables

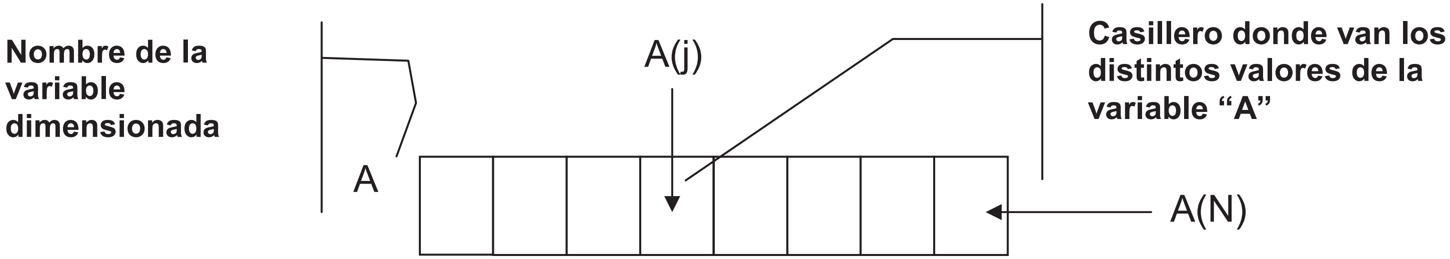
Variable es una dirección, alojada en la memoria de la computadora, para la identificación del CPU
es una especie de recipiente en cuyo interior podemos colocar cierta información
se identifica con un nombre representativo



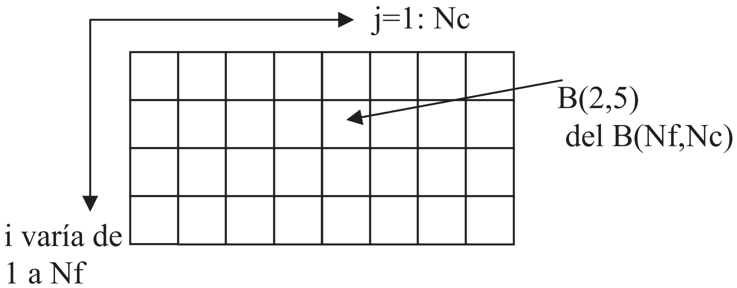
Variables simples solo puede guardar un solo valor y son las variables que ya hemos vistos: altura; base, xa, xb, etc

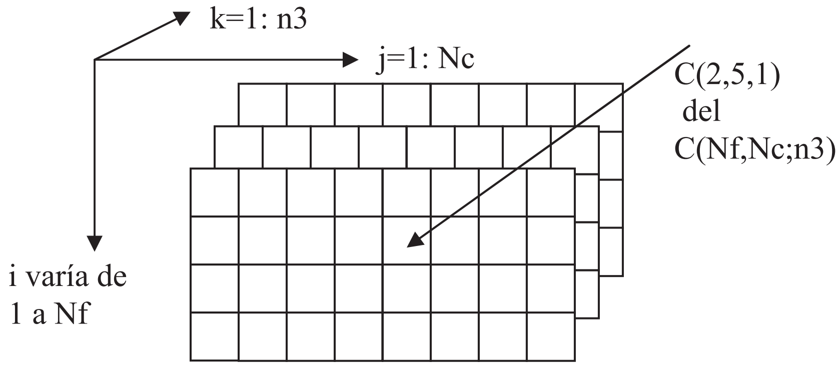
Variables Dimensionadas son las variables que guardan información referida a un mismo dato y que por la cantidad de datos es necesario dimensionarlas. Tipicamente Vectores y Matrices.

Vectores. Se dimensionan con el máximo número de componentes en la forma A(N).Se identifica uno con A(j).



Matrices. Se dimensionan con el máximo número de componentes en la forma A(Nf,Nc).Se identifica uno con A(i,j).





ALGORITMO. Definición y Tipos de Variables

Variable pueden ser clasificadas según el TIPO de información que almacenan:

Reales Enteras Lógicas Caracter

Jerarquía de las operaciones con las distintas variables

Prioridad	**Operador**	*Nombre*	**Resultado**
1	**^**	Potencia	Numérico
2	***** **/**	Producto - Cociente	Numérico
3	**+ - +**	Suma – Resta – Concatenación de caracteres	Suma y resta el resultado es numérico. Concatenación el resultado es Carácter
4	$= \neq < \leq > \geq$	Relación	Lógico
5	**.NOT.**	Negación	Lógico
6	**.AND.**	Conjunción [Y] lógico	Lógico
7	**.OR.**	Disyunción [O] lógico	Lógico

ALGORITMO. Estructura Tipo Secuencia

Es un conjunto finito de líneas, con principio y fin

Ejemplo: Algoritmo para calcular las raíces de la ecuación de segundo grado

Pseudo código	Elementos	Sintaxis de MATLAB
`Programa` raices	Nombre	`function raices`
`Reales (a; b; c; r1; r2)`	Declaración de variables	`% Reales (a; b; c; r1; r2)`

Escribir "Ingrese Coef Cuadrático"
Leer a
Escribir "Ingrese Coef Lineal"
Leer b
Escribir "Ingrese Coef Indep", Leer c
	Ingreso de datos	`disp('Ingrese Coef Cuadrático:');`
		`a = input('');`
		
		`b = input('Ingrese Coef Lineal:');`
		`c = input('Ingrese Coef Indep:');`

Discrim ← (b^2-4*a*c)
r1 ← (-b+Discrim^0.5)/(2*a)
r2 ← (-b-Discrim^0.5)/(2*a)
	Proceso	`Discrim = (b^2-4*a*c);`
		`r1 = (-b+Discrim^0.5)/(2*a);`
		`r2 = (-b-Discrim^0.5)/(2*a);`

Escribir "Raices son: ", r1, r2	Entrega de resultados	`disp('Raices son:');disp(r1);disp(r2);`
`Fin`	Final	`end`

Discriminante puede ser Negativo!!

ALGORITMO. Estructura Tipo Decisión

Es un conjunto finito de líneas, con una bifurcación según la respuesta lógica a una pregunta

Ejemplo: Algoritmo para calcular las raíces de la ecuación de segundo grado

<table>
<tr><th>Pseudo código</th><th>Elementos</th><th>Sintaxis de MATLAB</th></tr>
<tr><td>Programa raices</td><td>Nombre</td><td>function raices</td></tr>
<tr><td>Reales (a; b; c; r1; r2)</td><td>Declaración de variables</td><td>% Reales (a; b; c; r1; r2)</td></tr>
<tr><td>Escribir “Ingrese Coef Cuadrático”, Leer a
Escribir “Ingrese Coef Lineal”, Leer b
Escribir “Ingrese Coef Indep”, Leer c</td><td>Ingreso de datos</td><td>a = input('Ingrese Coef Cuadrático:');
b = input('Ingrese Coef Lineal:');
c = input('Ingrese Coef Indep');</td></tr>
<tr><td>Discrim ← (b^2-4*a*c)
SI (Discrim > 0) entonces
r1 ← (-b+Discrim^0.5)/(2*a)
r2 ← (-b-Discrim^0.5)/(2*a)
Escribir “Raices son: “, r1, r2
Sino
Escribir “Raíces imaginarias“
Fin del SI</td><td>Proceso
y
Entrega de resultados</td><td>Discrim = (b^2-4*a*c);
If (Discrim > 0)
r1 = (-b+Discrim^0.5)/(2*a);
r2 = (-b-Discrim^0.5)/(2*a);
disp('Raices son:');disp(r1);disp(r2);
else
disp('Raices imaginarias:');
end</td></tr>
<tr><td>Fin</td><td>Final</td><td>end</td></tr>
</table>

ALGORITMO. Estructura Tipo Variar

Es un conjunto finito de líneas, que se <u>repite un número definido y conocido de veces</u>

<u>Ejemplo</u>: Algoritmo para leer y escribir las componentes de un vector de dimensión 10

Pseudo código	Elementos	Sintaxis de MATLAB
`Programa leer_vec`	Nombre	`function leer_vec`
`Reales (vec(10))` `Enteros i, j`	Declaración de variables	`% Reales (vec(10))` `% Enteros i, j`

DOFOR i =1 HASTA 10 PASO 1
Escribir “Ingrese componente”,
Leer vec(i)
ENDDO
	Ingreso de datos	`disp(lectura de las componentes');`
		`for i=1:10`
		`disp ('Ingrese Componente:');`
		`vec(i) = input('');`
		`end`

DOFOR j =1 HASTA 10 PASO 1
Escribir “La componente es”,
Escribir vec(j)
ENDDO
	Proceso
	y
	Entrega de resultados
		`disp('Escritura de las componentes');`
		`for j=1:10`
		`disp ('La componente es:');`
		`disp (vec(j));`
		`end`

`Fin`	Final	`end`

ALGORITMO. Estructura Tipo Mientras

Es un conjunto finito de líneas, que se repite un número indefinido y desconocido de veces

Ejemplo: Algoritmo para que:"mientras la función f(x)=(3x²-12) sea distinto de cero, lea un valor de abscisa x0, calcule la función f(x0) en ese valor x0, y sólo entregue un valor de x0 cuando la función f(x) sea cero".

Pseudo código	Elementos	Sintaxis de MATLAB
`Programa` leer_vec	Nombre	`function` leer_vec
`Reales (x0, f)` `Enteros k`	Declaración de variables	`%Real (f, x0)` `%Integer k`

k 1
Escribir "Ingrese x0", Leer x0
f 3*x0^2-12
	Ingreso de datos	`k=1;`
		`x0=input ('íngrese abscisa');`
		`f= 3*x0^2-12;`

DOWHILE
k ← k + 1
Escribir "Ingrese x0", Leer x0
f ← 3*x0^2-12
ENDDO
Escribir "la abscisa que anula f es: ", x0
	Proceso
	y
	Entrega de resultados
		`while (abs(f) > 0)`
		`k =k + 1;`
		`x0=input ('íngrese abscisa');`
		`f= 3*x0^2-12 ;`
		`end`
		`disp('la abscisa es: '),x0`

`Fin`	Final	`end`

¿Cuál es la utilidad de k? Modificar para tener la "historia" de los x0

ALGORITMO. Estructura Tipo Mientras

Es un conjunto finito de líneas, que se <u>repite un número indefinido y desconocido de veces</u>

<u>Ejemplo</u>: Algoritmo para que:"mientras la función $f(x)=(3x^2-12)$ sea distinto de cero, lea un valor de abscisa x0, calcule la función f(x0) en ese valor x0.Y sólo entregue la historia de x0 cuando la función f(x) es cero".

Pseudo código	Elementos	Sintaxis de MATLAB
`Programa` cer_de_f	Nombre	`function` cer_de_f
`Reales (x0, f)` `Enteros k`	Declaración de variables	`%Real (A(2,1000), x0, f)` `%Integer k`

k 1
Escribir "Ingrese x0", Leer x0
f 3*x0^2-12
`A(k,1)=x0;`
`A(k,2)=f;`
	Ingreso de datos	`k=1;`
		`x0=input ('íngrese abscisa');`
		`f= 3*x0^2-12;`
		`A(k,1)=x0;`
		`A(k,2)=f;`

DOWHILE
k ← k + 1
Escribir "Ingrese x0", Leer x0
f ← 3*x0^2-12
`A(k,1)=x0;`
`A(k,2)=f;`
ENDDO
Escribir "la historio de : ", x0
`for i=1:k`
`Escribir, A(k,1), A(k,2)`
`end`
	Proceso
	y
	Entrega de resultados
		`while (abs(f) > 0)`
		`k =k + 1;`
		`x0=input ('íngrese abscisa');`
		`f= 3*x0^2-12 ;`
		`A(k,1)=x0;`
		`A(k,2)=f;`
		`end`
		`disp('la historia de x0 y f es: ')`
		`for i=1:k`
		`disp(' '), A(k,1), A(k,2)`
		`end`

`Fin`	Final	`end`

Modificar para controlar que no se supere la dimensión 1000 de la matriz A y definir <u>*"banda de cero"*</u>

ALGORITMO. Estructura Tipo Mientras

Ejercicio *Algoritmo para que:"mientras la función f(x)=(3x^2-12) sea distinto de cero, lea un valor de abscisa x0, calcule la función f(x0) en ese valor x0. Y sólo entregue la historia de x0 cuando la función f(x) es cero". No se debe superar las 1234 iteraciones.*

```
function cer_de_f
   %Real (A(2,1000), x0, f, tol)
   %Integer k, i
    tol=1e-12;
    max=[        ;]
    k=1;
    x=input ('íngrese abscisa');
    f= 3*x^2-12;
    A(1,k)=x;
    A(2,k)=f;
%
    [.        ] (abs(f) > tol [......] k < max)
        k =k + 1;
        x=input ('íngrese componente');
        f= 3*x^2-12 ;
        A(1,k)=x;
        A(2,k)=f;
    [        .]

    disp('la historia de x0 y f es: ')
    for i=1:k
        disp(' '), A(i,1), A(i,2)
    end
end
```

```
while (comparación o variable lógica)
    Línea 1
    Línea 2
    .........
    .........
end
```

Bloque de líneas que se "ejecuta" sólo si la **comparación o variable lógica** es VERDADERA.
Debe iniciarse antes del while
Debe cambiar dentro del Bloque

Operadores Lógicos

Mayor, Menor,…….….…en MATLAB……>…<

Igual, Distinto……..….…en MATLAB…....==……~=

OR……………………….en MATLAB……|

AND………...................en MATLAB…..&

dan como resultado Verdadero o Falso

ALGORITMO. Subprogramas

Los subprogramas son un **subproceso** constituido por un bloque de órdenes o comandos que realizan dicho subproceso y se los identifica con un Nombre.

Se los llama por su Nombre desde un proceso principal
Pueden necesitar datos de ingreso, y en general entregan un resultado
Los datos de ingreso y resultados pueden intercambiarse con el proceso principal mediante argumentos

En MATLAB se los identifica como "`function`".

Programa Principal	Subprograma
Es un conjunto de líneas de órdenes, y en alguna de ellas se invoca una `function` mediante el Nombre de la misma `function`	las `function` puede estar en el archivo principal o en otro archivo que debe llamarse con el nombre de la `function`

Programa Principal:

```
--------------------------
--------------------------
% ingreso de datos
  [x1, y1, x2, y2]= leer_puntos;
% sin pasaje de argumentos de entrada
% con multiples argumentos de salida
-----------------------------
-----------------------------
% Cálculo del Punto Medio
   ym = Punto_medio(x1, y1, x2, y2);
% con pasaje de argumentos de entrada
% con un único argumento de salida
-----------------------------
```

Subprograma:

```
function [a, b, c, d]= leer_puntos
  a=input('Ingrese x1');
  b=input('Ingrese y1');
  c=input('Ingrese x2');
  d=input('Ingrese y2')
end

function zm = Punto_medio (za, zb, zc , zd)
  xx=(za+zc)/2;
  zm=(zb+zd)/2;
end
```

NOTAR Variables Globales y Locales

ALGORITMO. Ejercicios

Ejercicio 1
Desarrollar un algoritmo que lea un vector de 100 componentes e imprima su módulo (norma cuadrática)

Ejercicio 2
Desarrollar un algoritmo que lea un vector de 45 componentes e imprima su norma infinita

Ejercicio 3
Desarrollar un algoritmo que lea un vector de 45 componentes e imprima su versor, sólo si su norma infnito es no nula. De lo contario exprese que el vector es nulo

ALGORITMO. Tips

- No hay obviedades
- No debe haber ambigüedades
- Cada orden deber ser precisa
- Ordene cosas simples, que sea sólo una tarea
- Deje la optimización de cantidad de órdenes para una segunda etapa
- Proponga el algoritmo y "ejecútelo". Sea estricto en ello.
- Si aparecen errores, corrija y vuelva a ejecutar. Itere hasta que esté conforme.

RAÍCES DE ECUACIONES NO LINEALES

Planteo de Problema

Estrategias Iterativas en general
- Análisis de función
- Acercamiento o Inicialización
- Recurrencia
- Control de Detención
- Actualización

Métodos de Intervalo
- Bisección
- Regula Falsi

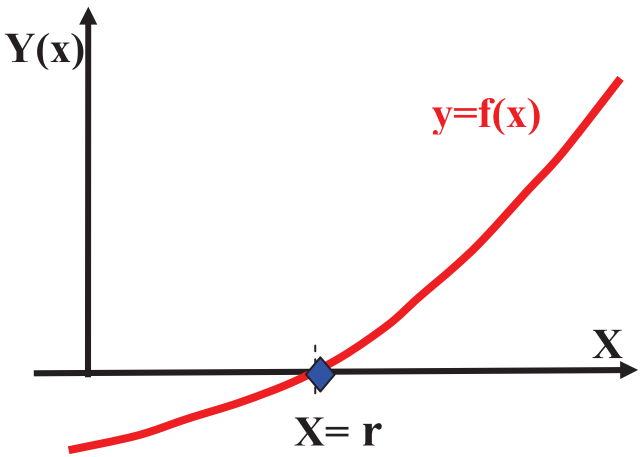
Métodos Abiertos
- Método de la Secante
- Método de Newton Raphson
- Métodos de Punto Fijo

Ejemplos

Resumen

RAÍCES DE ECUACIONES NO LINEALES

Se busca una aproximación de la abscisa que hace nulo el valor de una función, con una precisión deseada. Dicha abscisa se denomina raíz de la ecuación no lineal. Se obtiene de imponer igual a cero la ordenada (imagen) de la función.



Supuestos

La función **y = f(x):** R $\rightarrow$ R, es **no singular, continua,** conocida en forma analítica y tiene al menos una raíz

La *ecuación no lineal* es **f(x)=0**

La abscisa X=r es Raíz si verifica la ecuación no lineal.

Es decir si: **f(r)=0**

Ejemplo 1

Dada la función $y = a \cdot x^2 + b \cdot x + c$

La ecuación no lineal $0 = a \cdot x^2 + b \cdot x + c$

Tiene como raíces $r_{1,2} = (-b \pm \sqrt{b^2 - 4 \cdot a \cdot c})/(2 \cdot a)$

Ejemplo 2

Dada la función $y = x \cdot \tan(x) - c$

La ecuación no lineal $0 = x \cdot \tan(x) - c$

Tiene como raíces…………………..???????

Objetivo

Calcular una aproximación de la raíz mediante ***Métodos Iterativos***.

PROCEDIMIENTO GENERAL

Paso Inicial

Realizar un **análisis de la función**: determinar singularidades, asíntotas, etc.

Se busca toda la información posible a los efectos de elegir adecuadamente las variables iniciales de los métodos iterativos

Paso Acercamiento o Inicialización

Se debe **elegir adecuadamente las variables iniciales** de los métodos iterativos buscando toda la información posible a tal efecto.

Se trata de encontrar un intervalo en el eje X donde exista al menos una raíz de la ecuación no lineal.

Se busca un valor de abscisa cercano a la raíz.

Paso Aproximación mediante Métodos Iterativos

Métodos de Intervalos o Cerrados

- Método de Bisección
- Método de Regula Falsi

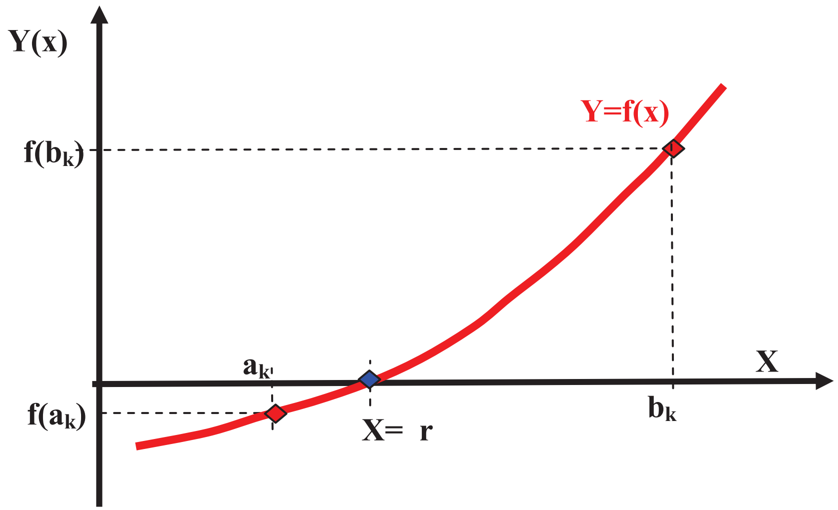
Métodos Abiertos

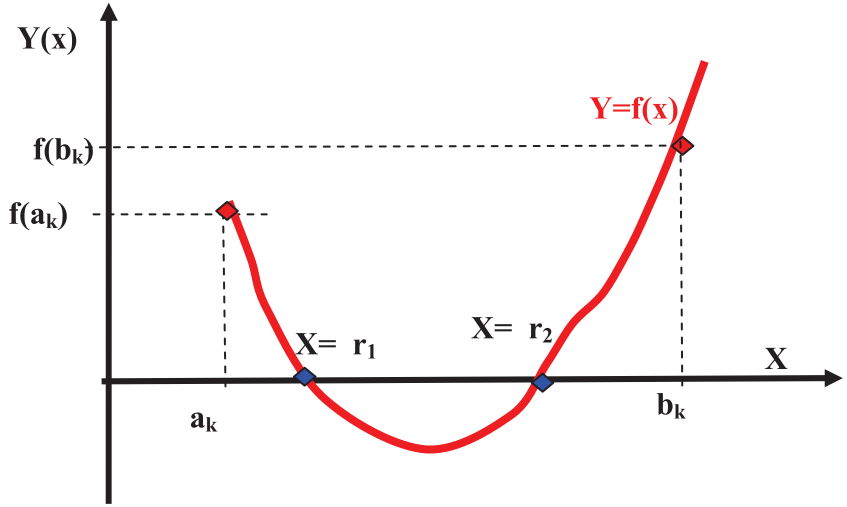
- Método de la Secante o de Newton Lagrange
- Método de Newton
- Métodos de Puntos Fijos

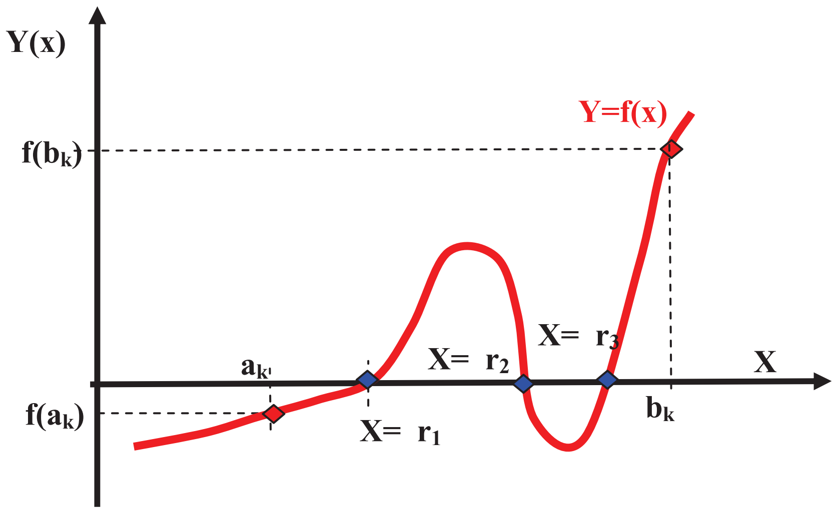
Siempre requieren de alguna condición de inicialización

Paso Acercamiento o Inicialización

Se trata de encontrar un intervalo en el eje X donde exista al menos una raíz de la ecuación no lineal.







$\Rightarrow$ *existe al menos una raìz*

Paso Aproximación mediante Métodos Iterativos

INICIALIZACIÓN
Se deben definir los contenidos de las variables de modo que se cumplan las condiciones de Inicialización del método

HACER MIENTRAS "No Hay Solución" es Verdadero
DOWHILE (NHS)

RECUERRENCIA
Se debe evaluar la nueva solución aproximada correspondiente a la nueva iteración o ciclo

CONTROL DE DETENCIÓN
Si alguna MEDIDA del ERROR es adecuada entonces se ha logrado la solución buscada. Se debe Cambiar NHS a falso

SI (Valor Absoluto de $f(r_{k+1}) <$ Tolerancia)
NHS es FALSO
FINSI

ACTUALIZACIÓN DE VARIABLES
Se reasignan las variable de modo que se cumplan con las condiciones de Inicalización

FIN DEL MIENTRA

MÉTODOS DE INTERVALOS O CERRADOS	*MÉTODOS ABIERTOS*
Método de Bisección	Método de la Secante o de Newton Lagrange
Método de Regula Falsi	Método de Newton o de Newton Raphson
	Métodos de Puntos Fijos

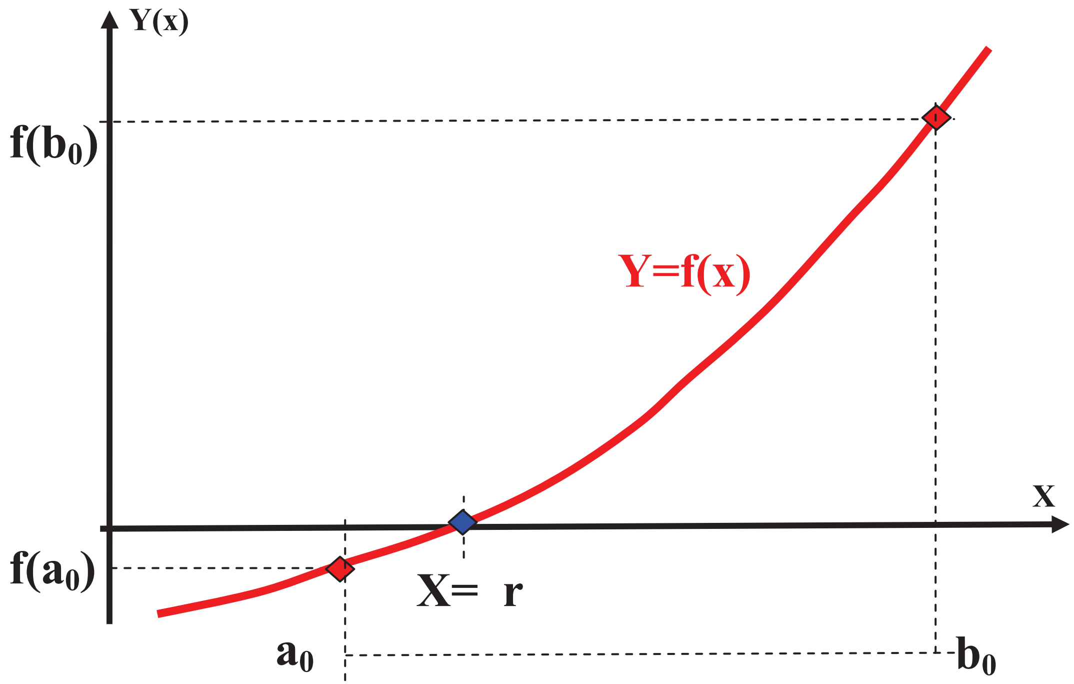
MÉTODO DE BISECCIÓN

Paso Inicial

Un intervalo $[a_0 ; b_0]$ en el eje de las abscisas X en el cuál la función no lineal tenga al menos una raíz.

Esto es abscisas tales que

$$f(a_0).f(b_0) < 0$$



Primera Aproximación

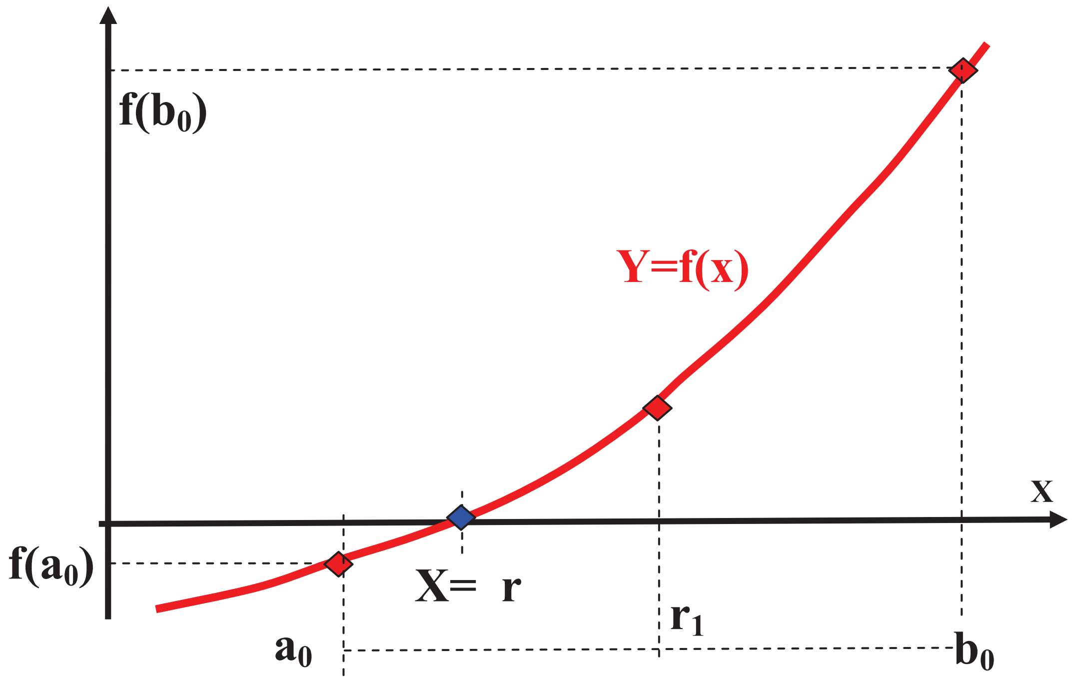
Se debe encontrar la primera aproximación de la raíz; es decir, el primer elemento de la sucesión de soluciones aproximadas.

MÉTODO DE BISECCIÓN

Primera aproximación

Dado el Intervalo Inicial $(a_0; b_0)$;

se aproxima la raíz con
El Punto Medio del Intervalo



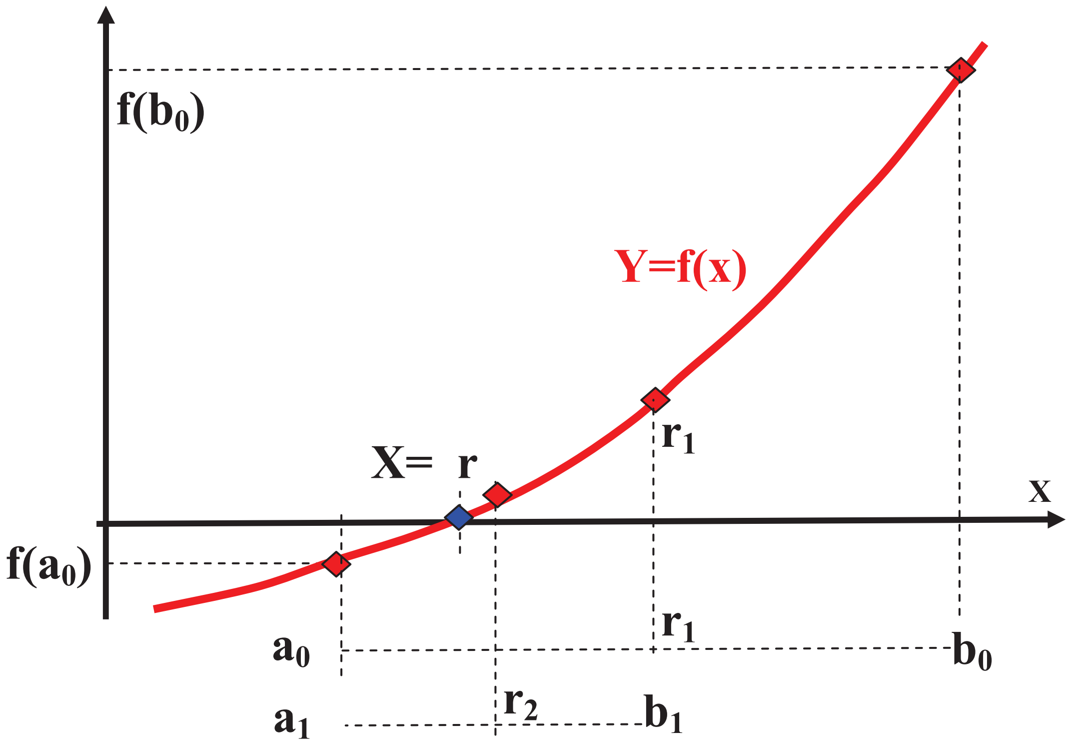
Se tienen dos intervalos Intervalo $(a_0; r_1)$ Intervalo $(r_1; b_0)$

y se debe seleccionar uno de ellos que será el Intervalo 1 $(a_1; b_1)$ en el cual está la raíz. ¿Cuál es?

Segunda aproximación

Dado el Intervalo 1 (a_1; b_1);

se aproxima la raíz con
el Punto Medio del intervalo 1

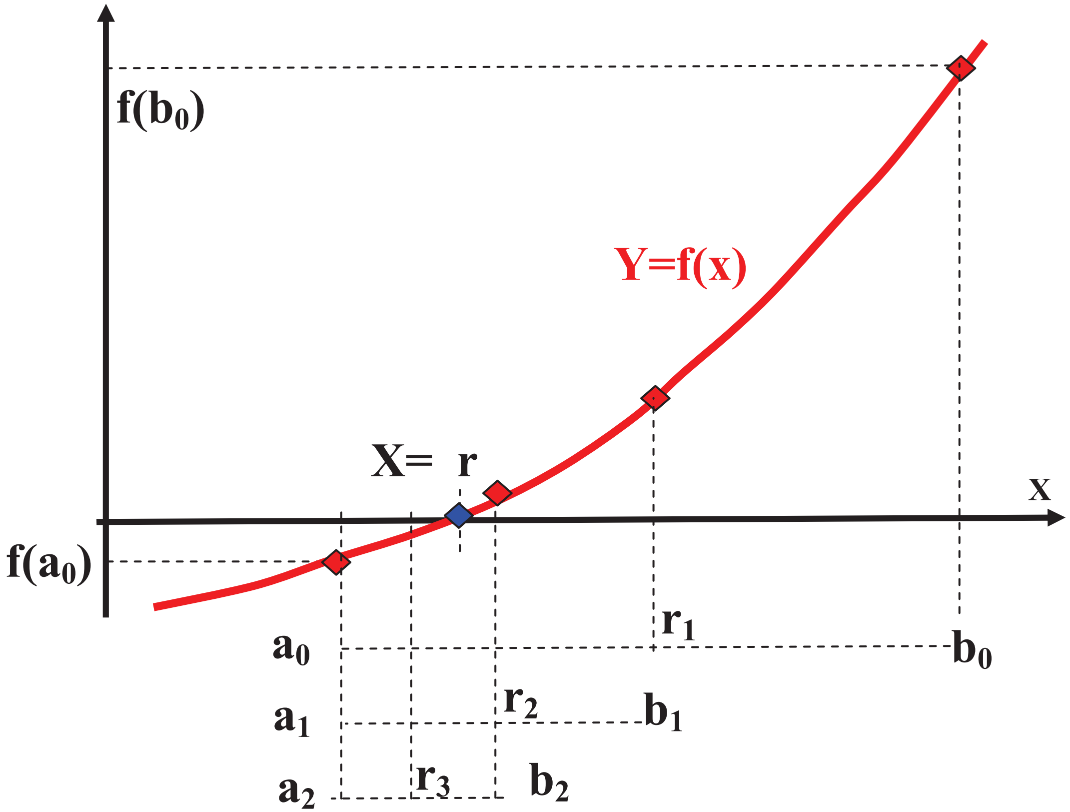


Se tienen dos intervalos: Intervalo (a_1; r_2) Intervalo (r_2; b_1)

y se debe seleccionar uno de ellos que será el Intervalo 2 (a_2; b_2) en el cual está la raíz.

Tercera aproximación

Dado el Intervalo 2 (a_2; b_2);



se aproxima la raíz con
el Punto Medio del intervalo 2

Se tienen dos intervalos Intervalo (a_2; r_3) Intervalo (r_3; b_2)

y se debe seleccionar uno de ellos que será el Intervalo 3 (a_3; b_3) en el cual está la raíz.

Cuarta aproximación

Dado el Intervalo 3 (a_3; b_3);

se aproxima la raíz con el Punto Medio del intervalo 3

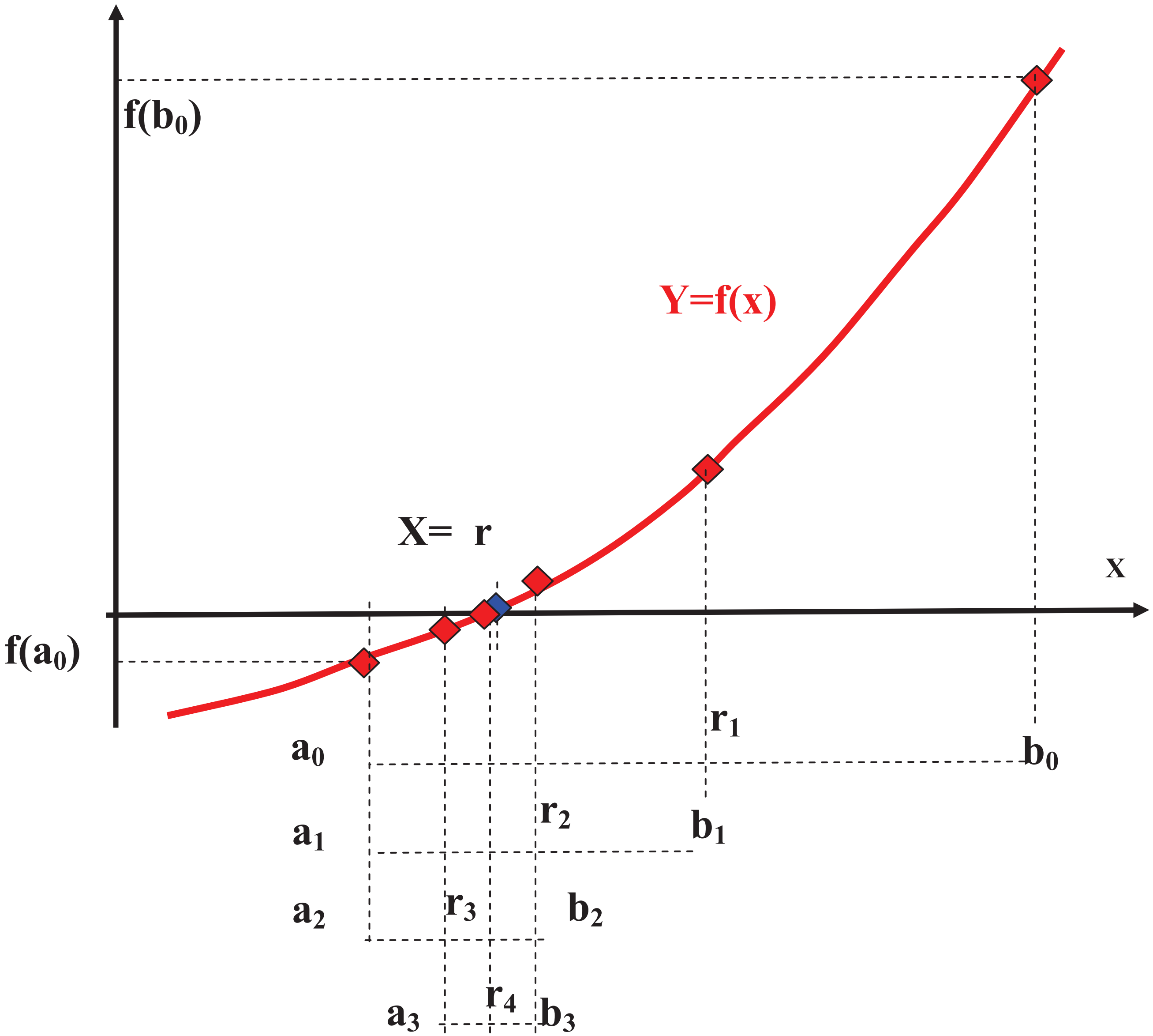
$$\boxed{r_4 = \frac{(a_3 + b_3)}{2}}$$

Se tienen dos intervalos

Intervalo (a_3; r_4)

Intervalo (r_4; b_3)

y se debe seleccionar uno que será
el Intervalo 4 (a_4; b_4)
en el cual está la raíz.



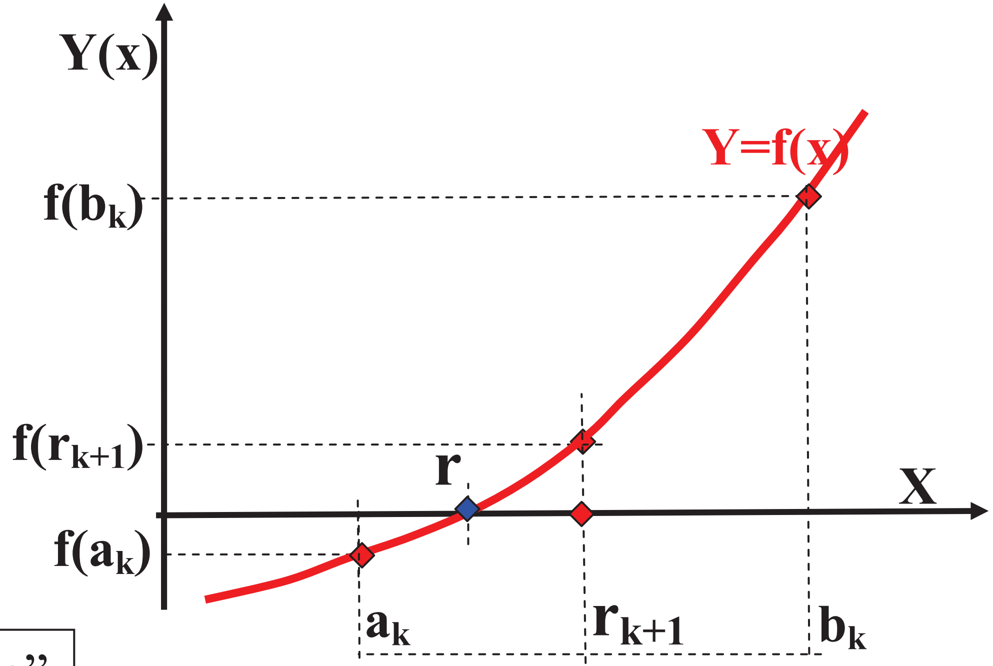
k-ésima aproximación

Dado el Intervalo k $(a_k;\ b_k)$

que cumple la condición de inicialización

$$f(a_k)*f(\ b_k)<0$$

se calcula la nueva raíz aproximada
como la abscisa media del Intervalo k



Se controla “ si r_{k+1} es la solución buscada”.

Se actualizan las variables si es que no se alcanzó la solución buscada, eligiendo de los dos intervalos definidos con $a_k;\ b_k$ y r_{k+1}.

Si $(f(a_k)*f(r_{k+1})<0)$ *entonces* $a_{k+1}= a_k$ $b_{k+1} =r_{k+1}$	*Si* $(f(b_k)*f(r_{k+1})<0)$ *entonces* $a_{k+1}= r_{k+1}$ $b_{k+1} = b_k$

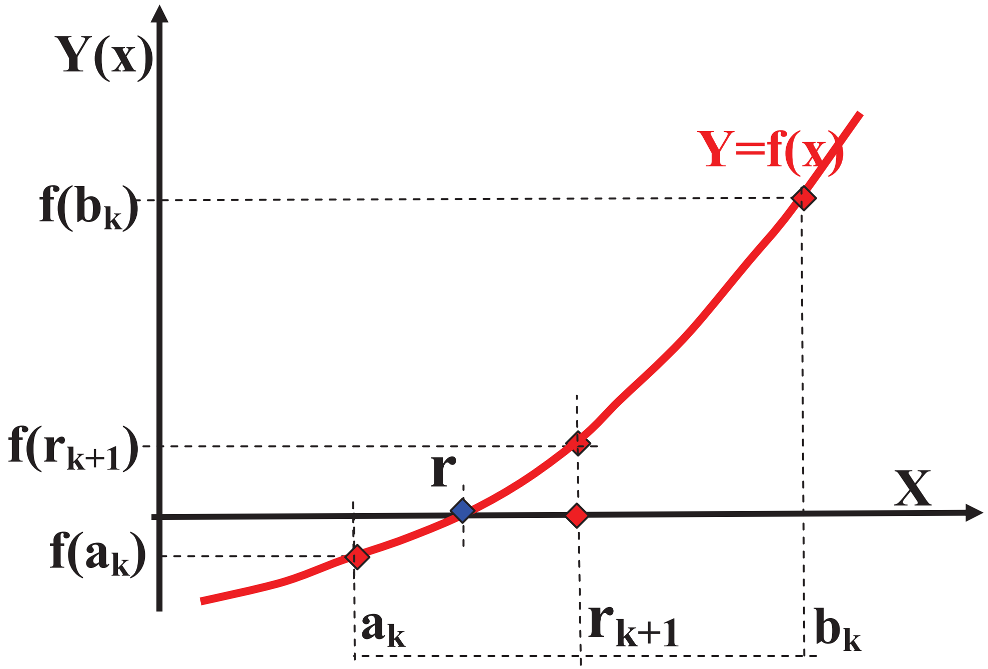
ALGORITMO DEL MÉTODO BISECCIÓN

INICIALIZACIÓN

Con k=0, y el Intervalo $(a_0; b_0)$ tal que

$f(a_0)*f(b_0)<0$

No_Hay_Solu=Verdadero



HACER MIENTRAS (*No_Hay_Solu)*

RECURRENCIA

$r_{k+1} = (a_k + b_k)/2$

CONTROL DE DETENCIÓN

Si ($f(r_{k+1})=0$) entonces

No_Hay_Solu=Falso

Fin SI

ACTUALIZACIÓN $k=k+1$

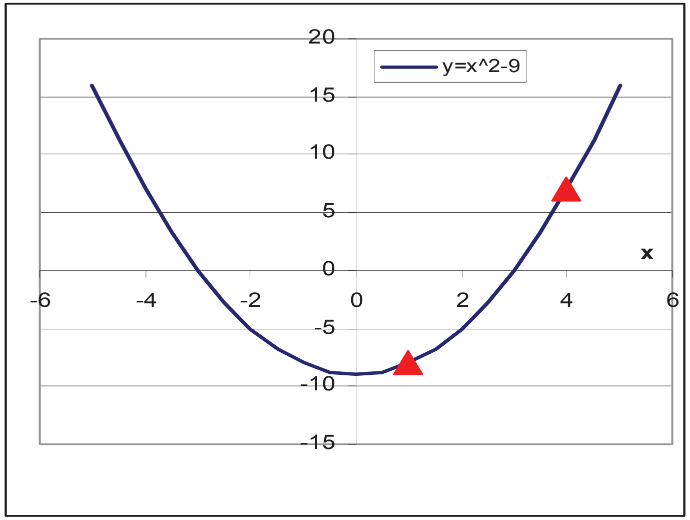
Si $(f(a_k)*f(r_{k+1})<0)$ *entonces* $a_{k+1}= a_k$ $b_k = r_{k+1}$

Si $(f(b_k)*f(r_{k+1})<0)$ *entonces* $a_{k+1}= r_{k+1}$ $b_{k+1}= b_k$

FIN DEL HACER MIENTRAS

MÉTODO DE BISECCIÓN

Ejemplo: Buscar una aproximación de $\sqrt{9}$



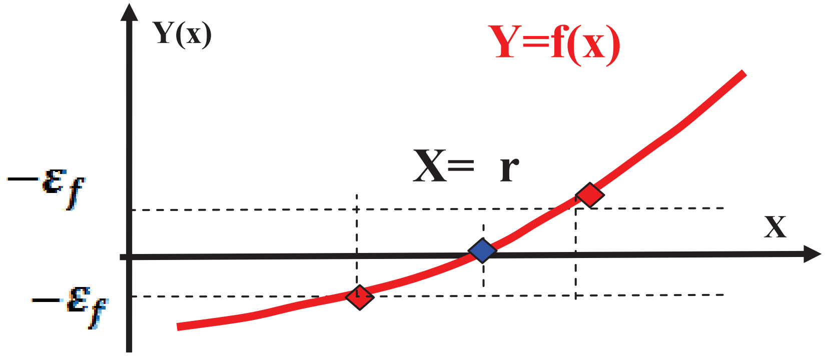
itera	a	b	f(a)	f(b)	r	f(r).
0	1	4	-8	7	2,5	-2,75
1						
2						
3						

CONTROL DE DETENCIÓN

Si ($f(r_{k+1})=0$) entonces
No_Hay_Solu=Falso
Fin SI

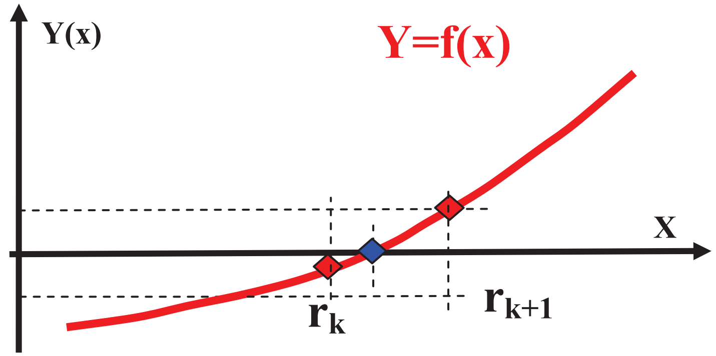
Pero es "muy exigente"

Es conveniente "suavizar"



Si (Valor absoluto de $f(r_{k+1}) < \varepsilon f$) entonces
No_Hay_Solu=Falso
Fin SI

Es conveniente "controlar el cambio de digitos"



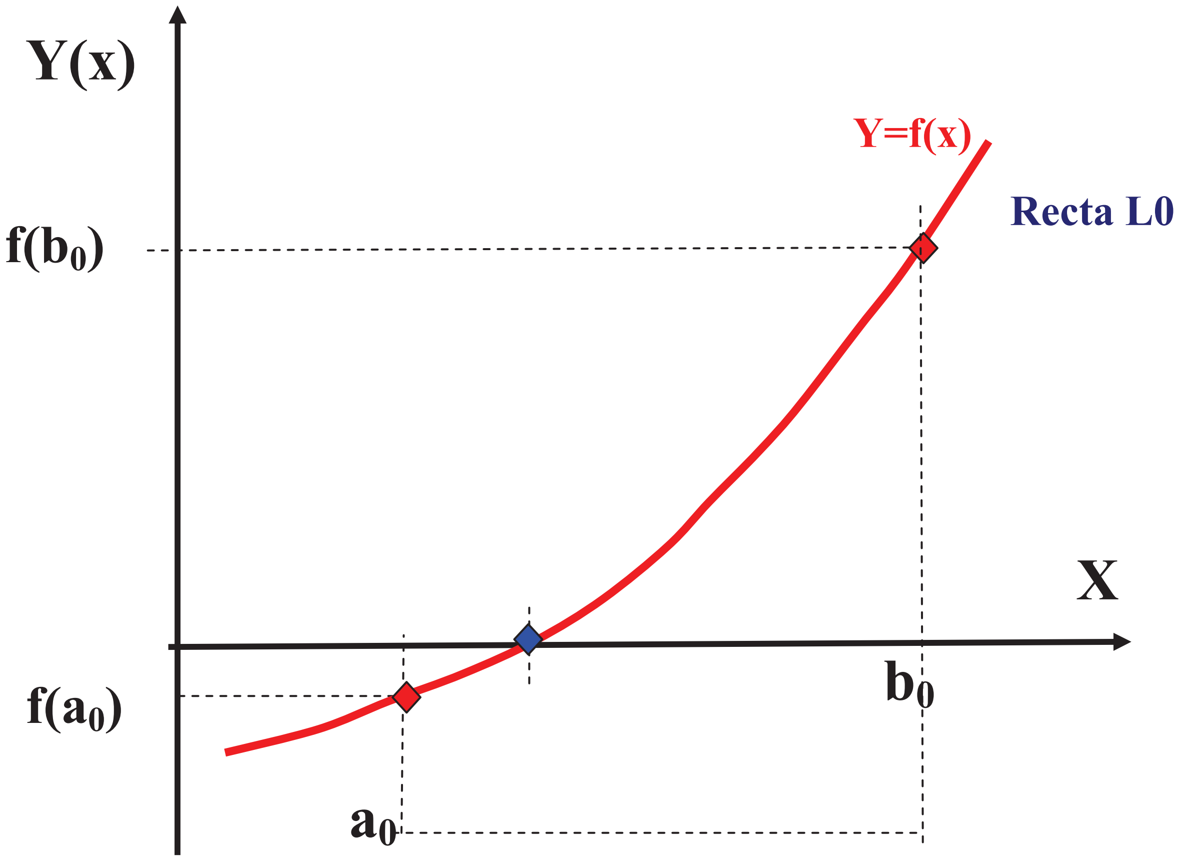
Si (Valor absoluto de $[r_{k+1} - r_k] < \varepsilon 1$) entonces
No_Hay_Solu=Falso
Fin SI

Si (Valor absoluto de $[(r_{k+1} - r_k)/ r_{k+1}] < \varepsilon r$) entonces
No_Hay_Solu=Falso
Fin SI

MÉTODO DE REGULA FALSI

INICIALIZACIÓN

Es igual que BISECCIÓN. Es decir, se necesita un intervalo donde al menos exista uan raíz



ALGORITMO DEL MÉTODO REGULA FALSI

INICIALIZACIÓN

Con k=0, y el Intervalo $(a_0; b_0)$ tal que

$f(a_0)*f(b_0)<0$

No_Hay_Solu=Verdadero

HACER MIENTRAS (*No_Hay_Solu*)

CONTROL DE DETENCIÓN

Si $(f(r_{k+1})=0)$ *entonces*

No_Hay_Solu=Falso

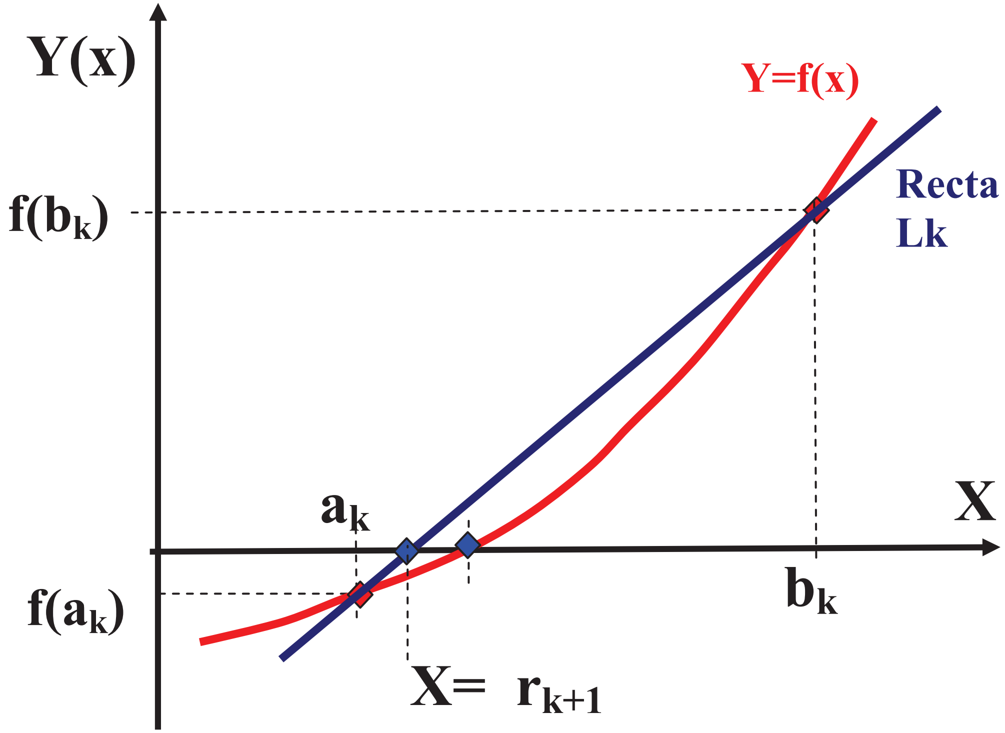
Fin SI

ACTUALIZACIÓN $k=k+1$

Si $(f(a_k)*f(r_{k+1})<0)$ *entonces* $a_{k+1}=a_k$ $b_k=r_{k+1}$

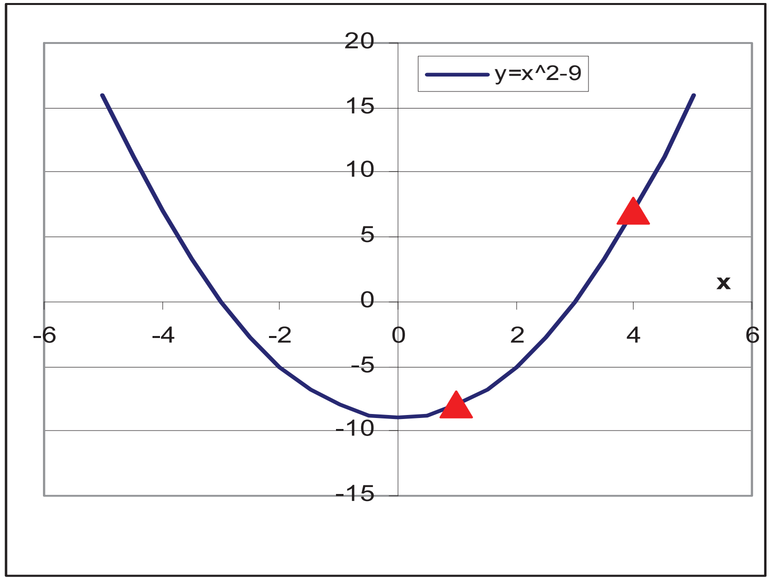
Si $(f(b_k)*f(r_{k+1})<0)$ *entonces* $a_{k+1}=r_{k+1}$ $b_{k+1}=b_k$

FIN DEL HACER MIENTRAS



MÉTODO DE REGULA FALSI

Ejemplo: Buscar una aproximación de $\sqrt{9}$

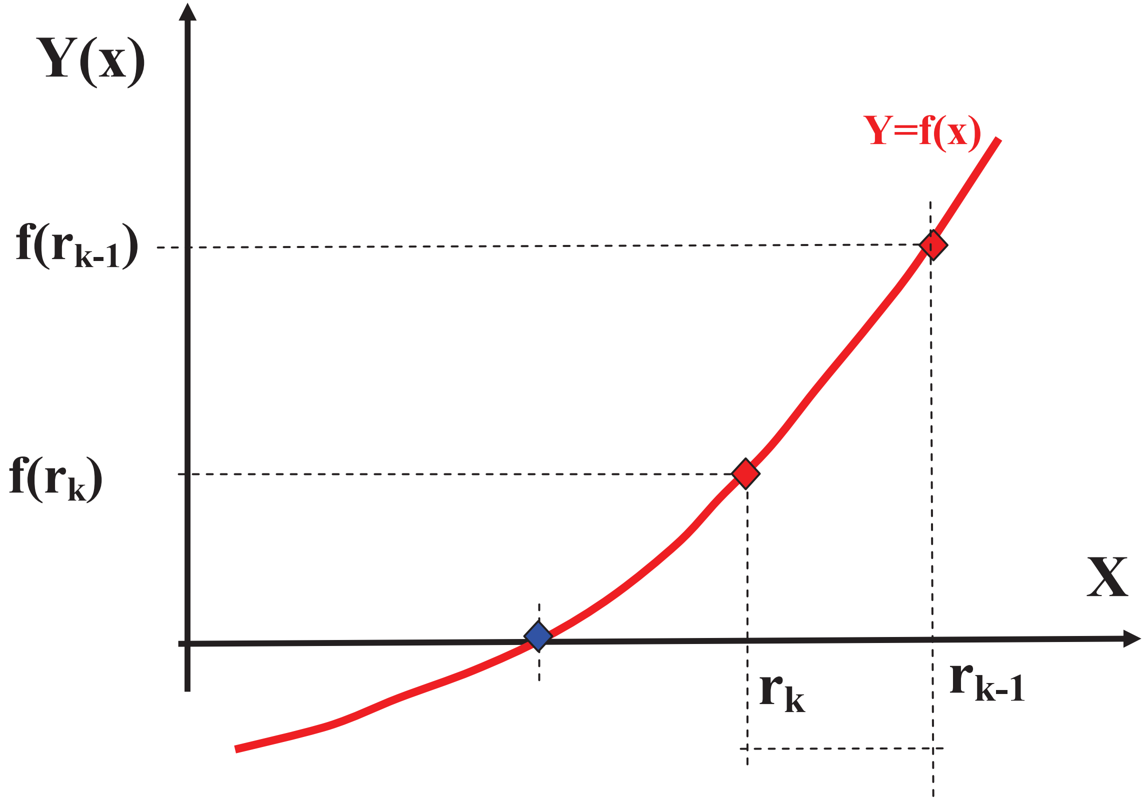


itera	a	b	f(a)	f(b)	m	r	f(r).
0	1	4	-8	7	5	2,6	-2,24
1							
2							
3							
4							

MÉTODO *DE LA SECANTE*

INICIALIZACIÓN

Se necesita *dos puntos cercanos* a una raíz. Pueden ser dos raíces aproximadas con otro método



ALGORITMO DEL MÉTODO DE LA SECANTE

INICIALIZACIÓN

Dos Puntos cercanos *a la raíz buscada*

k=0,

No_Hay_Solu=Verdadero

HACER MIENTRAS (*No_Hay_Solu*)

CONTROL DE DETENCIÓN

Si ($f(r_{k+1})=0$) entonces

No_Hay_Solu=Falso

Fin SI

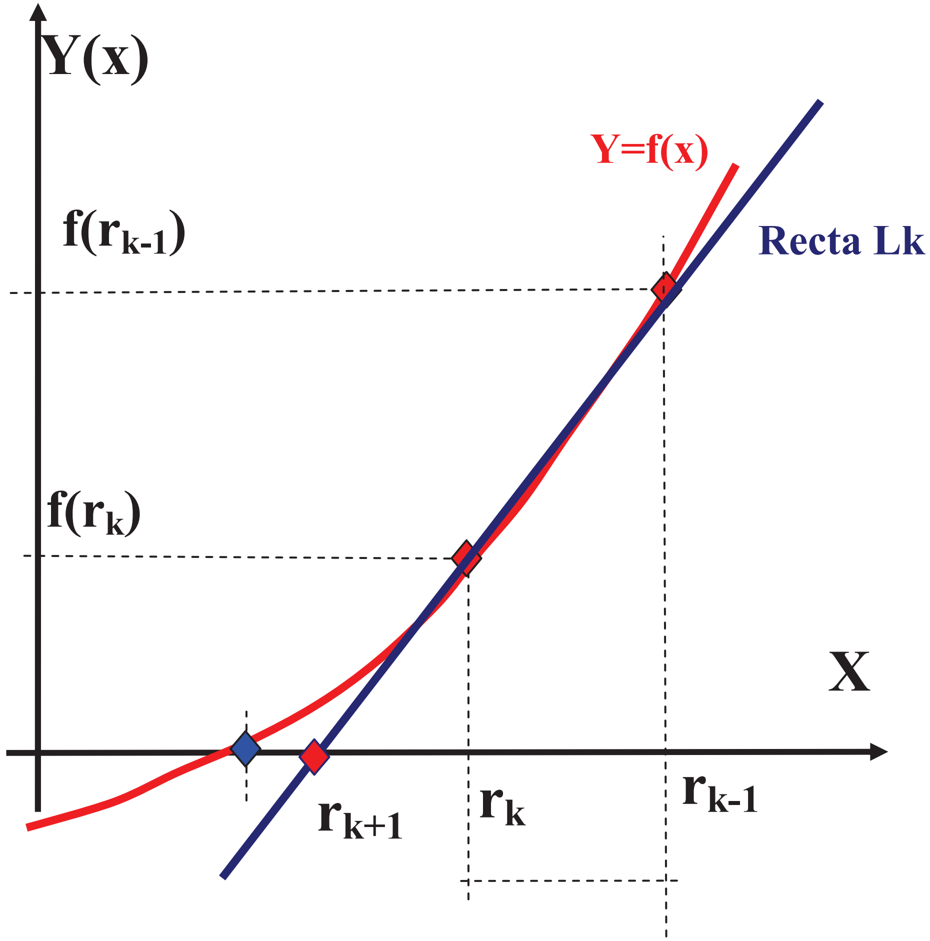
ACTUALIZACIÓN

k =k+1

$r_{k-1} = r_k$

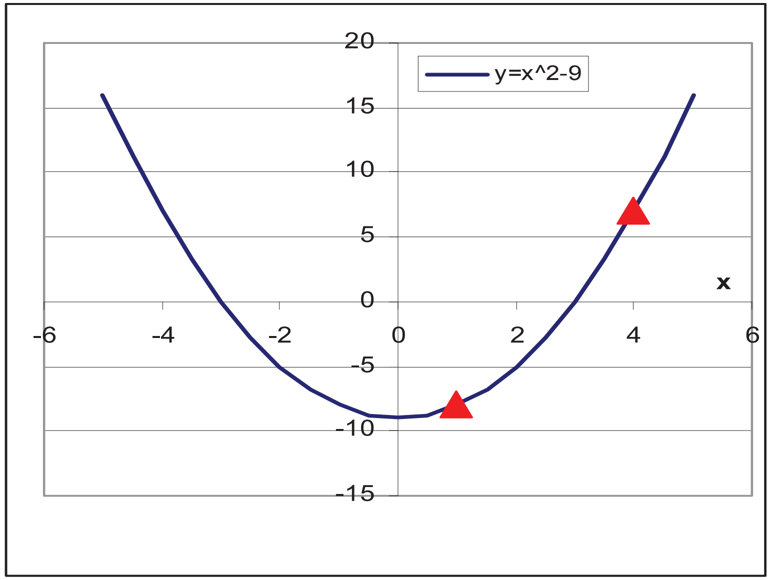
$r_k = r_{k+1}$ Se retienen dos soluciones aproximadas

FIN DEL HACER MIENTRAS



MÉTODO DE LA SECANTE

Ejemplo: Buscar una aproximación de $\sqrt{9}$

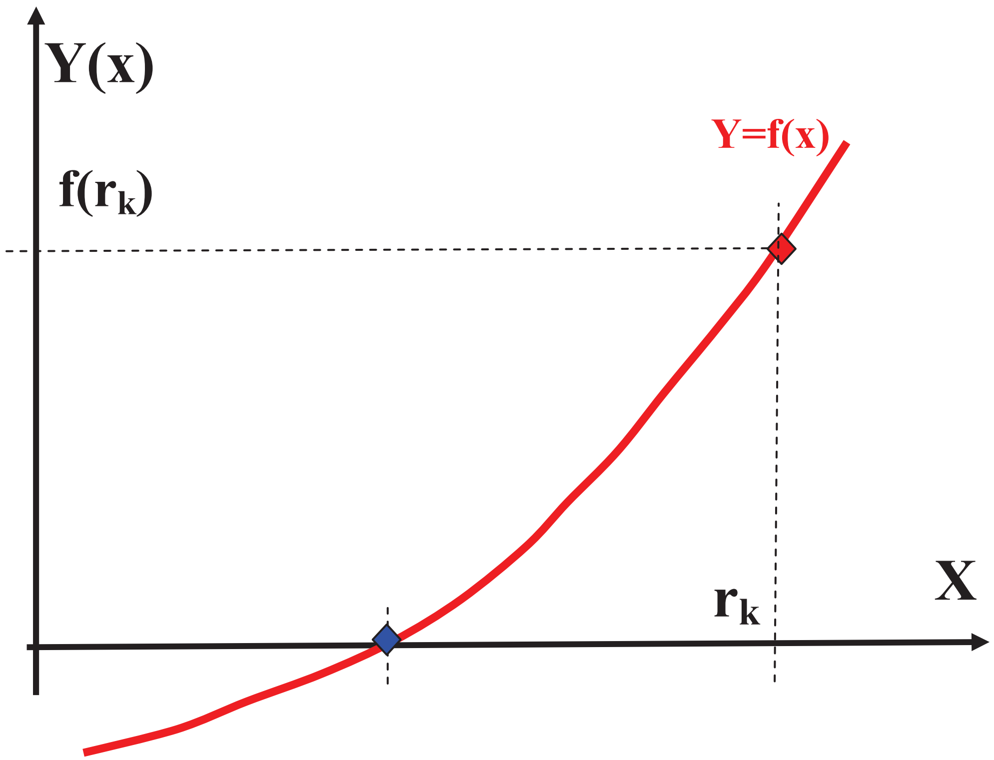


itera	a	b	f(a)	f(b)	m	r	f(r).
0	1	4	-8	7	5	2,6	-2,24
1							
2							
3							
4							

MÉTODO *DE NEWTON RAPHSON*

INICIALIZACIÓN

Se necesita *un punto cercano* a una raíz. Pueden ser una raíz aproximada con otro método, o simplemente una propuesta



ALGORITMO DEL MÉTODO NEWTON RAPHSON

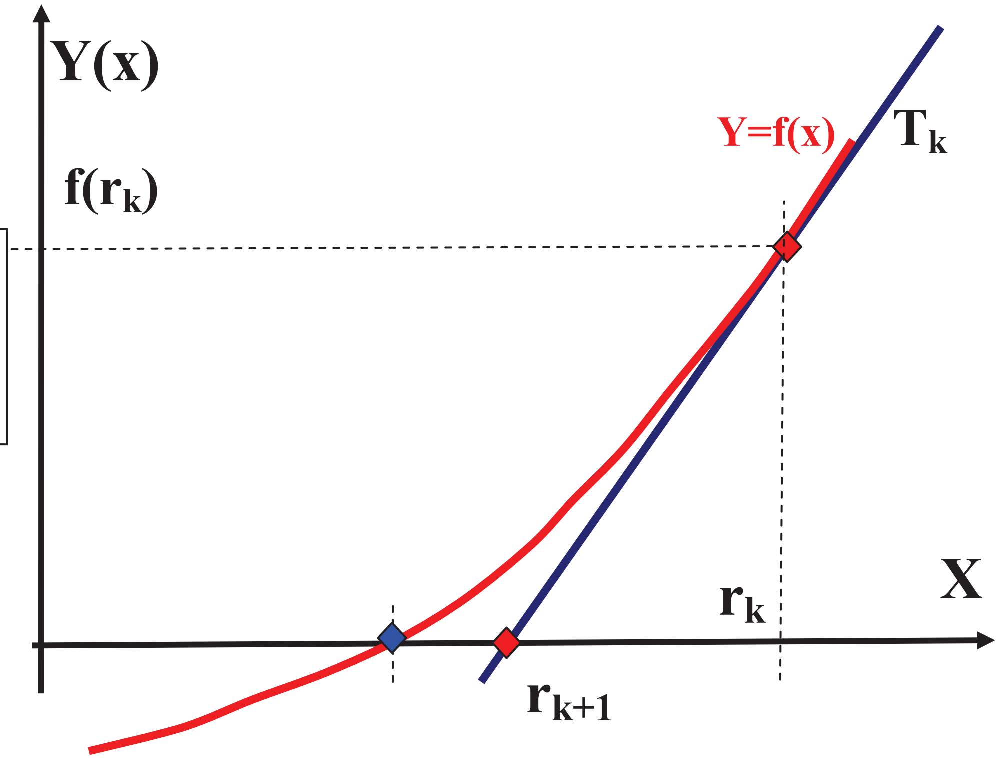
INICIALIZACIÓN

Un Punto cercano *a la raíz buscada*

k=0,

No_Hay_Solu=Verdadero

HACER MIENTRAS (*No_Hay_Solu*)



CONTROL DE DETENCIÓN

Si ($f(r_{k+1})=0$) entonces

No_Hay_Solu=Falso

Fin SI

ACTUALIZACIÓN

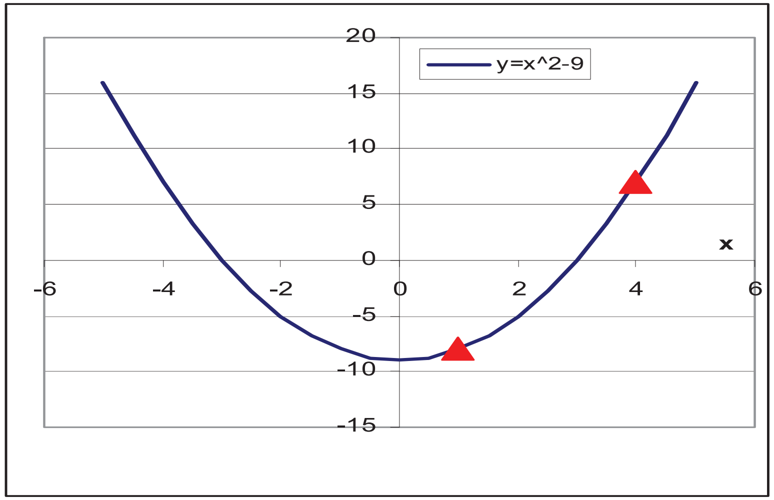
k =k+1

$r_k = r_{k+1}$ Se retienen una solución aproximada

FIN DEL HACER MIENTRAS

MÉTODO DE NEWTON RAPHSON

Ejemplo: Buscar una aproximación de $\sqrt{9}$



itera	rk	f(rk)	m	rk+1	f(rk+1).
0	1	-8	2		
1					
2					
3					
4					
5					

MÉTODO DE PUNTO FIJO

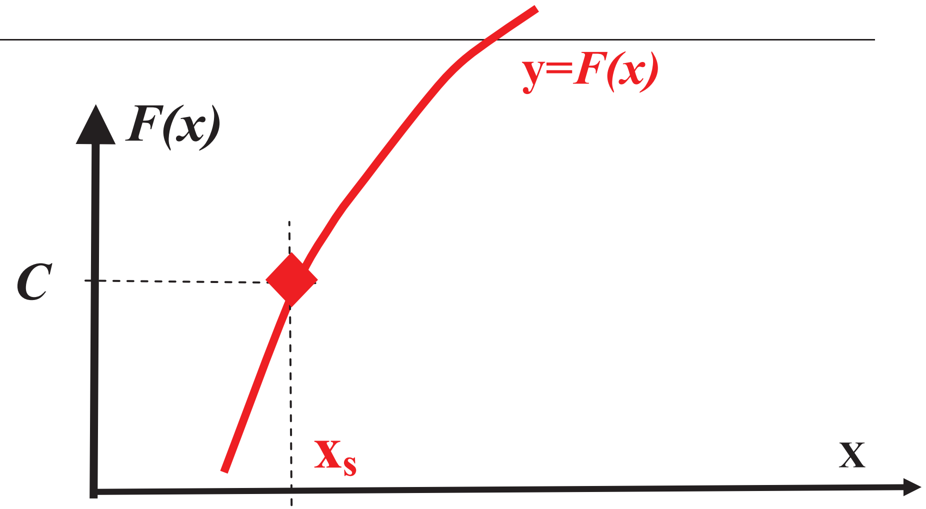
Dada una *función no lineal*

$$y = F(x)$$

se busca el valor de abscisa x_s tal que

$$F(x_s) = C$$

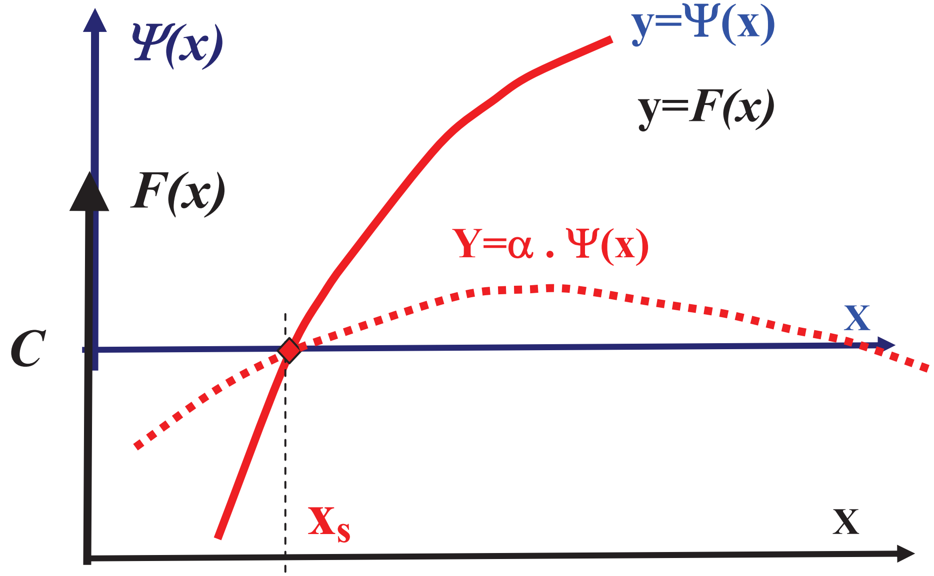
Con *C una constante arbitraria y conocida.*



Es posible
escribir la ecuación no lineal en la forma:

$$\psi(x_s) = F(x_s) - C = 0$$

que es la búsqueda de la raíz de la función



El valor de la abscisa x_s es INVARIANTE
a que la ecuación no lineal se multiplique por
α un ESCALAR NO NULO

$$\alpha \cdot \psi(x_s) = \alpha \cdot (F(x_s) - C) = 0$$

Es posible sumar en ambos miembros el valor de la abscisa x_s

Si a la ecuación no lineal

$$\boxed{\alpha \cdot \psi(x_s) = \alpha \cdot (F(x_s) - C) = 0}$$

Se le suma x_s en ambos miembros resulta

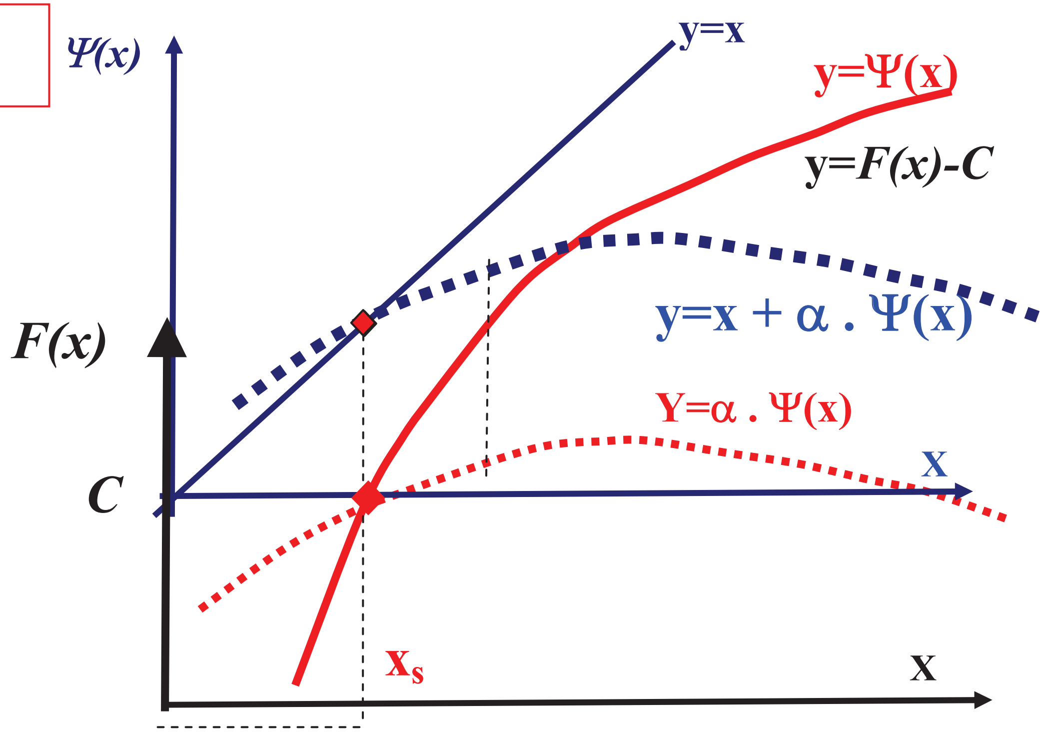
$$\boxed{x_s = x_s + \alpha \cdot \psi(x_s)}$$

que se puede escribir en la forma:

$$\boxed{x_s = g(x_s)}$$

donde

$$g(x) = x + \alpha \cdot (F(x) - C)$$
$$= x + \alpha \cdot \psi(x)$$



La igualdad $x = g(x)$ se puede interpretar como la intersección de las siguientes funciones

	Recta que bisecta el primer cuadrante (y=x) con la curva y=g(x). El punto solución es tal que tiene abscisa y ordenadas de igual valor. Es el *Punto Fijo de a curva g(x)*

Inicialización

Se necesita *un punto cercano* a una raíz. (Igual a Newton Raphson)

Recurrencia

La aproximación de la raíz se obtiene mediante,

$$x_{k+1} = g(x_k)$$

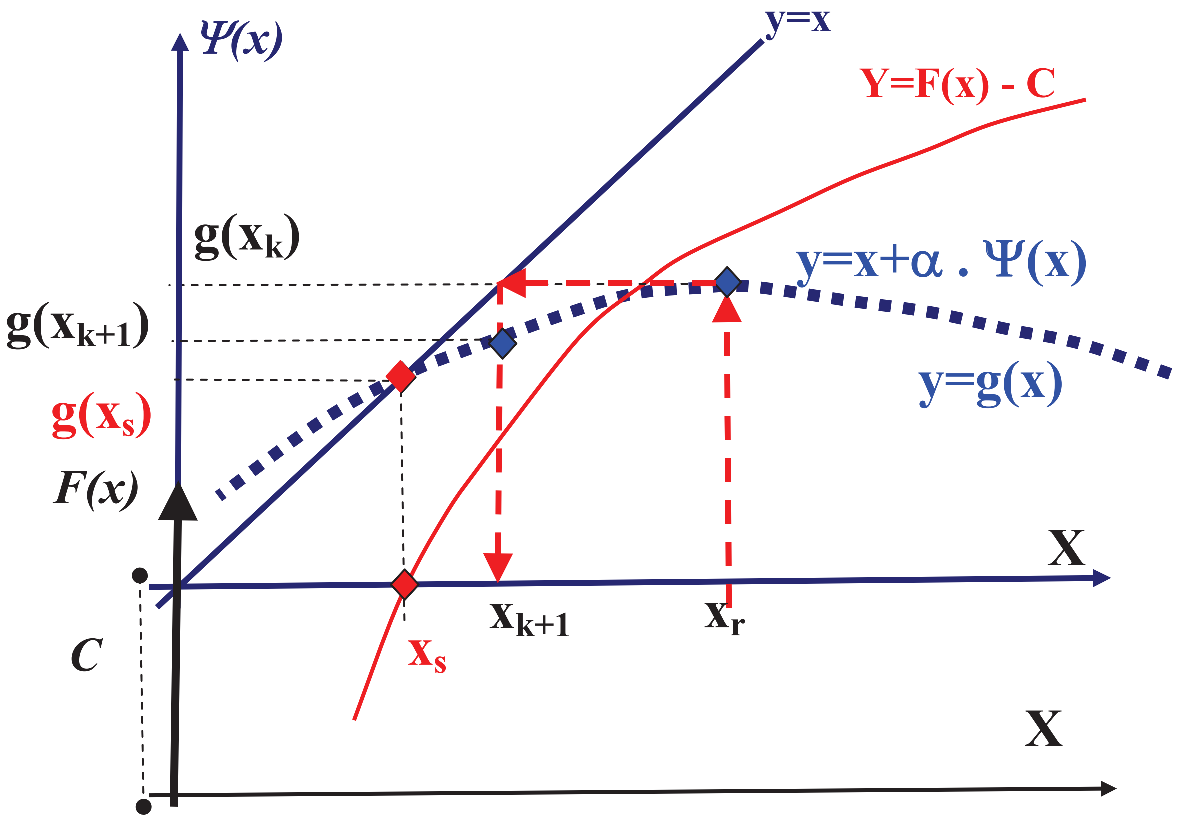
Control de Detención

Igual que métodos anteriores. Alternativamente se debe controlar si se cumple que

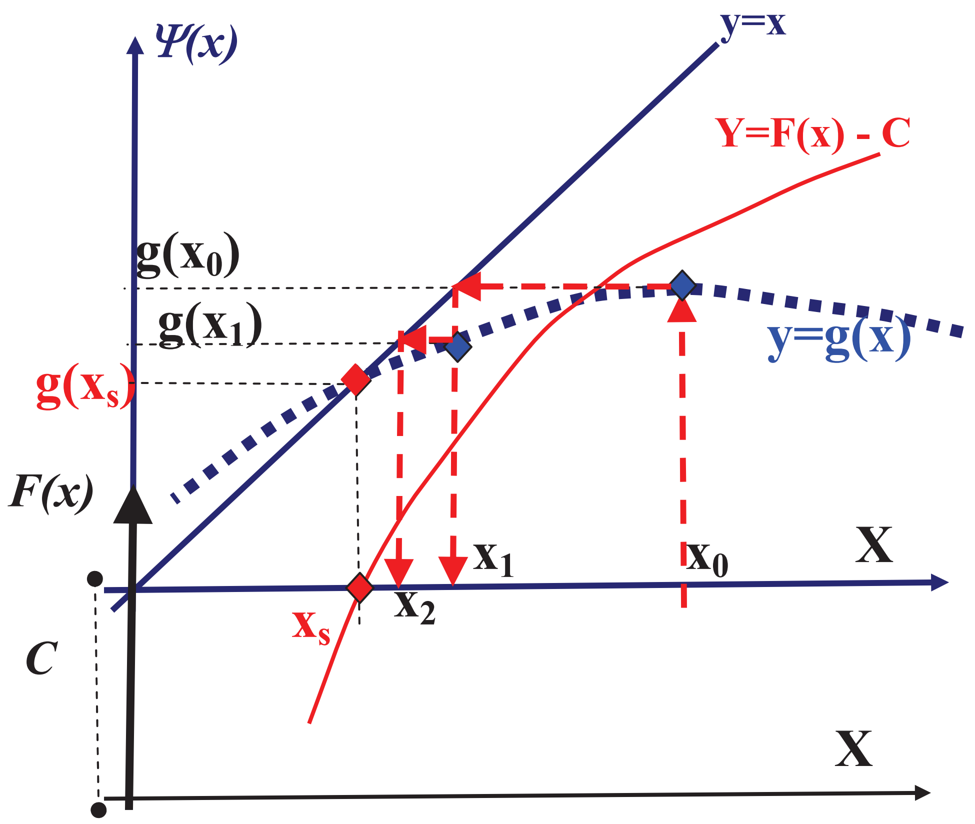
$$\left|x_{k+1} - g(x_{k+1})\right| \leq \varepsilon_f$$

Actualización de Variables

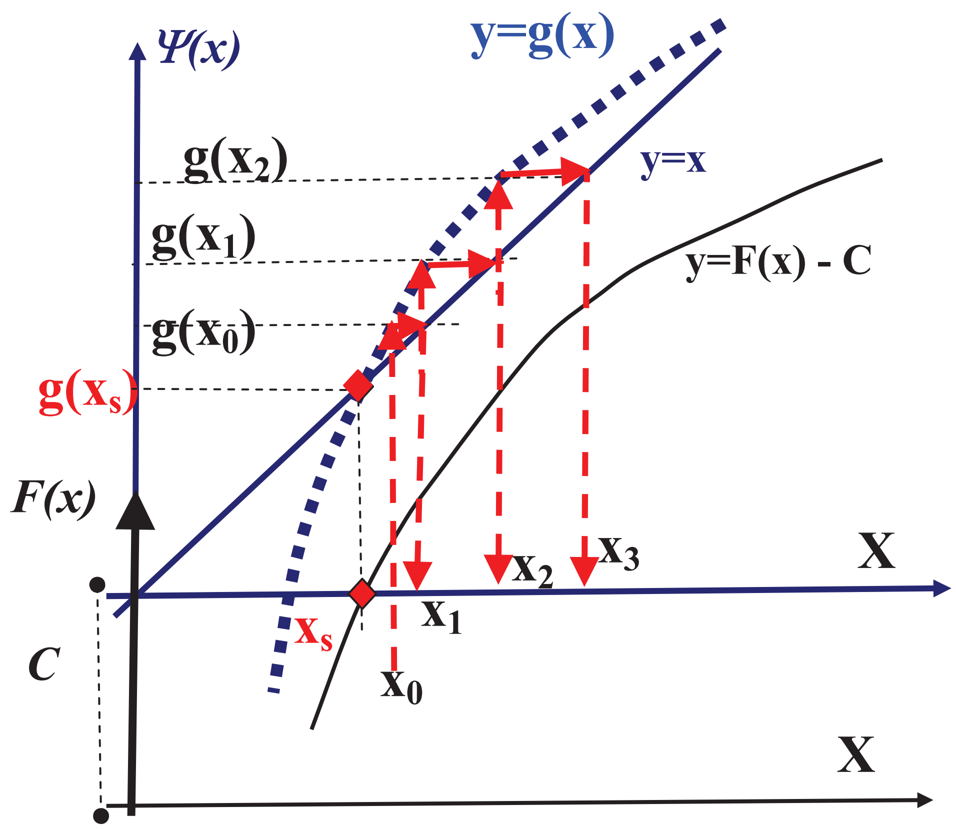
Se deben retener la última aproximación obtenida.



MÉTODO DE PUNTO FIJO



CONVERGE



DIVERGE

MÉTODO DE PUNTO FIJO

Condición de Convergencia del Método de Punto Fijo

La solución de la igualdad $x = g(x)$ es x_s tal que es el punto fijo de *g(x)*; es decir,

$$x_s = g(x_s)$$

La fórmula de recurrencia es

$$x_{k+1} = g(x_k)$$

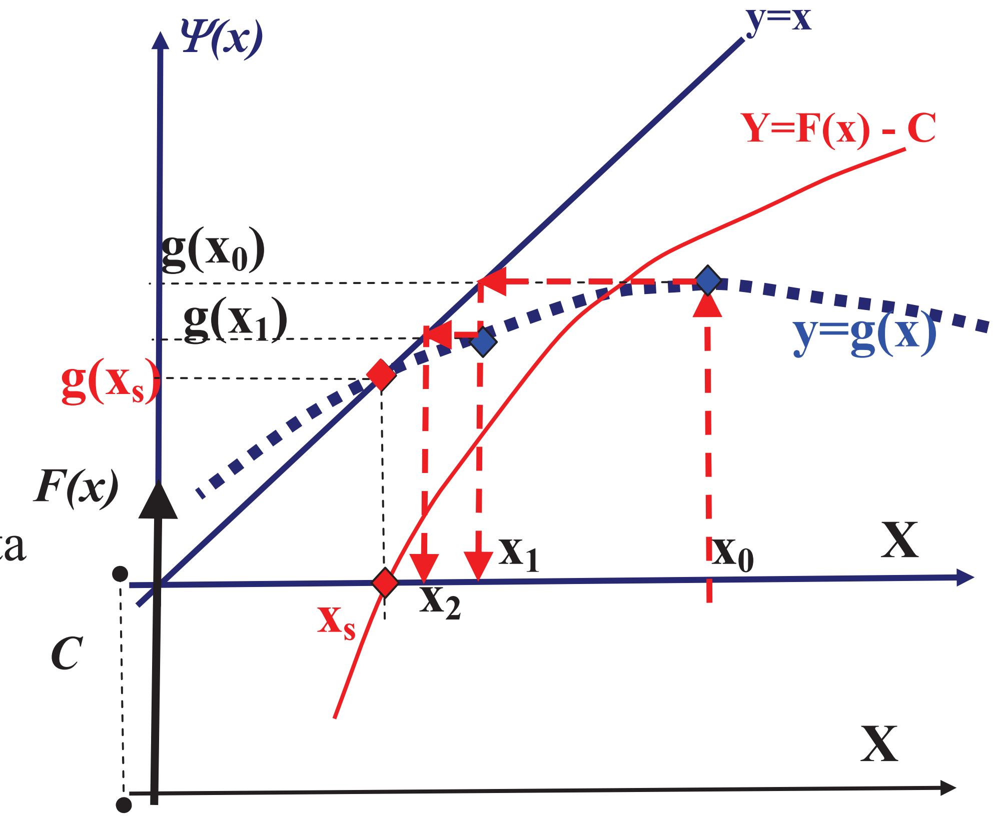
Al restar miembro a miembro, se tiene

$$x_{k+1} - x_s = g(x_k) - g(x_s)$$

Al aplicar el Teorema del Valor Medio, resulta

$$(x_{k+1} - x_s) = \left.\frac{dg(x)}{dx}\right|_{x=\xi} \cdot (x_k - x_s)$$

con ξ una abscisa entre x_k y x_s.

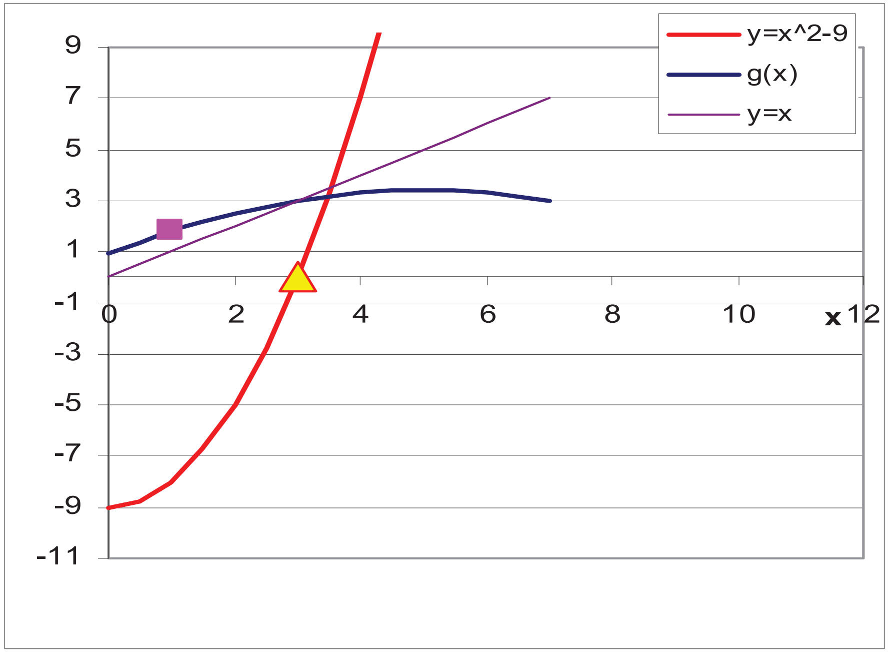


$$\varepsilon_{k+1} = \left.\frac{dg(x)}{dx}\right|_{x=\xi} \cdot \varepsilon_k$$

CONVERGE si $\left| \left.\frac{dg(x)}{dx}\right|_{x=\xi} \right| < 1$

MÉTODO DE PUNTO FIJO

Ejemplo: Buscar una aproximación de $\sqrt{9}$, es buscar que $x^2=9$.



itera	rk	rk+1=g(rk)	f(rk+1).
0	1	1,8	-5,76
1			
2			
3			
4			
5			
6			
7			
8			

$$g(x) = x - (1/10)\cdot(x^2 - 9)$$

$$\frac{dg(x)}{dx} = 1 - (2/10)\cdot x$$

MÉTODO ITERATIVOS PARA OBTENER RAICES DE ECUACIONES NO LINEALES

Resumen de los Métodos			
Método	**Inicialización**	**Recurrencia**	**Actualización**
Bisección	Intervalo con Raíz	$r_{k+1} = (a_k + b_k)/2$	Elije Intervalo con Raíz
Regula Falsi	Intervalo con Raíz	$r_{k+1} = a_k - \frac{f(a_k)}{m_k} \quad con \quad m_k = \left(\frac{f(a_k) - f(b_k)}{(a_k) - (b_k)} \right)$	Elije Intervalo con Raíz
Secante	2 Puntos Cercanos	$r_{k+1} = r_k - \frac{f(r_k)}{m_k} \quad con \quad m_k = \left(\frac{f(r_k) - f(r_{k-1})}{(r_k) - (r_{k-1})} \right)$	Retiene 2 Puntos Cercanos
Newton Raphson	1 Punto Cercano	$r_{k+1} = r_k - \frac{f(r_k)}{m_k} \quad con \quad m_k = \left. \frac{df}{dx} \right\|_{r_k}$	Retiene 1 Punto Cercano
Punto Fijo	1 Punto Cercano	$x_{k+1} = g(x_k)$	Retiene 1 Punto Cercano

MÉTODO ITERATIVOS PARA OBTENER RAICES DE ECUACIONES NO LINEALES

CONTROL DE DETENCIÓN

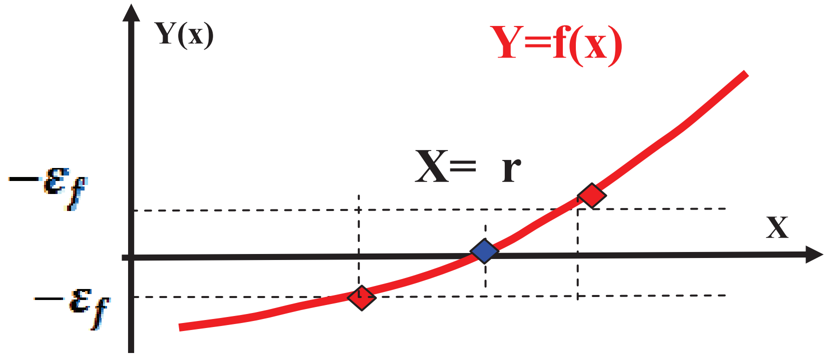
Los siguientes controles son complementarios

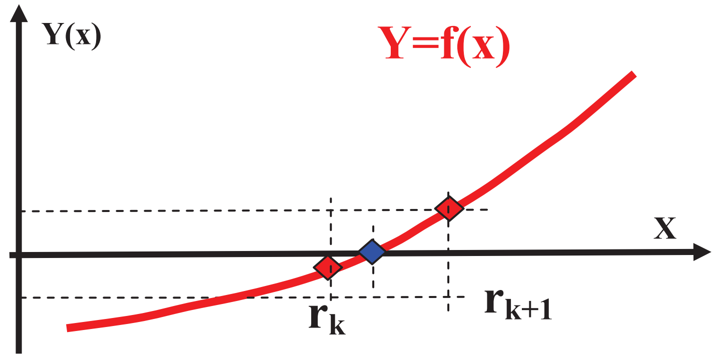
Si (Valor absoluto de $f(r_{k+1}) < \varepsilon f$) entonces
No_Hay_Solu=Falso
Fin SI

Si (Valor absoluto de $[r_{k+1} - r_k] < \varepsilon 1$) entonces
No_Hay_Solu=Falso
Fin SI

Si (Valor absoluto de $[(r_{k+1} - r_k)/ r_{k+1}] < \varepsilon r$) entonces
No_Hay_Solu=Falso
Fin SI

Si ($k >$ MaxIter) entonces
No_Hay_Solu=Falso
Fin SI





MÉTODOS ITERATIVOS EN PROBLEMAS MATRICIALES DE SISTEMAS DE ECUACIONES

Introducción.
Repaso de métodos iterativos para obtener raíces de ecuaciones no lineales

Sistemas de Ecuaciones Lineales (No Homogéneos)

- Método de Jacobi.
  - Algoritmo, Convergencia
- Método de Gauss Seidel.
  - Algoritmo

Autovalores y Autovectores

- Propiedades
- Método de la Potencia. Algoritmo, Convergencia
- Método de la Potencia Inversa
- El cociente de Rayleigh

Aplicaciones

INTRODUCCIÓN

Los **métodos iterativos** generan **una sucesión de soluciones aproximadas**, que debe **converger a la solución exacta** del problema que se pretende resolver.

Cada iteración genera una solución aproximada, y su diferencia respecto de la solución exacta debe ser menor que en la iteración anterior para que exista convergencia

Tienen en general los siguientes componentes

Inicialización o Acercamiento
Recurrencia
Control de Detención
Actualización

y deben cumplir algún
CRITERIOS DE CONVERGENCIA
Para garantizar que se encontrará la solución del problema

MÉTODOS ITERATIVOS PARA RAICES DE ECUACIOENS NO LINEALES

Se busca el valor $x_{k+1} = r$ tal que $f(r) = 0$

Síntesis:

	Método Iterativo	
	Punto Fijo	Newton Raphson
Inicialización	Se propone una solución inicial x_0	
Recurrencia	$x_{k+1} = g(x_k)$	$r_{k+1} = r_k - \frac{f(r_k)}{m_k} \quad con \quad m_k = \left.\frac{df}{dx}\right\|_{r_k}$
Control de Detención	$\left\|f(x_{k+1})\right\| \leq \varepsilon_f$ o $k \leq k\max$ $\left\|x_{k+1} - x_k\right\| \leq \varepsilon_a$ o $\left\|\frac{x_{k+1} - x_k}{x_{k+1}}\right\| \leq \varepsilon_r$	
Actualización	Retiene la última solución aproximada	
CRITERIOS DE CONVERGENCIA	$\left\| \left.\frac{dg(x)}{dx}\right\|_{x=\xi} \right\| < 1$	$\left\| \left.\frac{dg(x)}{dx}\right\|_{x=\xi} \right\| < 1 \quad con \quad g(x) = x - \frac{f(x)}{\frac{df}{dx}}$

MÉTODOS ITERATIVOS PARA RAICES DE ECUACIOENS NO LINEALES

La implementación se puede sintetizar de la siguiente forma:

Se eligen los valores de $x_0 \quad \varepsilon_a \quad \varepsilon_r \quad \varepsilon_f$ Inicialización

Se define **ban =true**

while(ban)

$$x_{k+1} = g(x_k)$$

Recurrencia $r_{k+1} = r_k - \dfrac{f(r_k)}{m_k}$ *con* $m_k = \left.\dfrac{df}{dx}\right|_{r_k}$

k=k+1

if $(|f(x_{k+1})| \leq \varepsilon_f)$

ban =false Control de Detención

end

if $(|x_{k+1} - x_k| \leq \varepsilon_a)$

ban =false **end**

if $(k \leq k\max)$

ban =false

end

$x_k = x_{k+1}$ Actualización de Variables

end

SISTEMAS DE ECUACIONES LINEALES

Dada una Matriz **A** ε R^{NxN}, un vector **b** ε R^{N}, el sistema de ecuaciones lineales asociado es

$$\mathbf{A}\cdot\underline{x}=\underline{b}$$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & .... & a_{1N} \\ a_{21} & a_{22} & a_{23} & .... & a_{2N} \\ a_{31} & a_{32} & a_{33} & .... & a_{3N} \\ .... & .... & .... & .... & .... \\ a_{N1} & a_{N2} & a_{N3} & .... & a_{NN} \end{pmatrix}\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ ... \\ x_N \end{Bmatrix}=\begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ ... \\ b_N \end{Bmatrix}$$

Es posible escribir $$\mathbf{A}=\mathbf{D}+\mathbf{B}$$

$$\begin{pmatrix} a_{11} & 0 & 0 & .... & 0 \\ 0 & a_{22} & 0 & .... & 0 \\ 0 & 0 & a_{33} & .... & 0 \\ .... & .... & .... & .... & .... \\ 0 & 0 & 0 & .... & a_{NN} \end{pmatrix}=\mathbf{D} \qquad \mathbf{B}=\begin{pmatrix} 0 & a_{12} & a_{13} & .... & a_{1N} \\ a_{21} & 0 & a_{23} & .... & a_{2N} \\ a_{31} & a_{32} & 0 & .... & a_{3N} \\ .... & .... & .... & 0 & .... \\ a_{N1} & a_{N2} & a_{N3} & .... & 0 \end{pmatrix}$$

El sistema se puede escribir

$$\mathbf{D}\cdot\underline{x}+\mathbf{B}\cdot\underline{x}=\underline{b}$$

SISTEMAS DE ECUACIONES LINEALES. MÉTODO DE JACOBI

Dada una Matriz **A** ε R^{NxN}, un vector **b** ε R^{N}, el sistema de ecuaciones lineales asociado es

$$\mathbf{A} \cdot \underline{x} = \underline{b}$$

El sistema se puede escribir

$$\mathbf{D} \cdot \underline{x} + \mathbf{B} \cdot \underline{x} = \underline{b}$$

$$\underline{x} = -\mathbf{D}^{-1} \cdot \mathbf{B} \cdot \underline{x} + \mathbf{D}^{-1} \cdot \underline{b}$$

$$\underline{x} = \mathbf{T} \cdot \underline{x} + \underline{c}$$

$$\mathbf{T} = -\mathbf{D}^{-1} \cdot \mathbf{B} \qquad \underline{c} = \mathbf{D}^{-1} \cdot \underline{b}$$

$$\mathbf{T} = \begin{pmatrix} 0 & -a_{12}/a_{11} & -a_{13}/a_{11} & .... & -a_{1N}/a_{11} \\ -a_{21}/a_{22} & 0 & -a_{23}/a_{22} & .... & -a_{2N}/a_{22} \\ -a_{31}/a_{33} & -a_{32}/a_{33} & 0 & .... & -a_{3N}/a_{33} \\ .... & .... & .... & 0 & .... \\ -a_{N1}/a_{NN} & -a_{N2}/a_{NN} & -a_{N3}/a_{NN} & .... & 0 \end{pmatrix} \qquad \underline{c} = \begin{Bmatrix} b_1/a_{11} \\ b_2/a_{22} \\ b_3/a_{33} \\ ... \\ b_N/a_{NN} \end{Bmatrix}$$

Se puede iterar con

$$\underline{x}^{(k+1)} = \mathbf{T} \cdot \underline{x}^{(k)} + \underline{c}$$

Hasta encontrar que el ERROR es tan pequeño como se quiera

EJEMPLO

Se busca la solución del siguiente sistema de ecuaciones lineales

$$\begin{pmatrix} 50 & -25 & 0 & 0 \\ -25 & 50 & -25 & 0 \\ 0 & -25 & 50 & -25 \\ 0 & 0 & -25 & 50 \end{pmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} 10 \\ 20 \\ 20 \\ 10 \end{Bmatrix} \qquad \mathbf{A} \cdot \underline{x} = \underline{b}$$

La matriz de coeficientes se puede escribir

$$\mathbf{A} \quad = \quad \mathbf{D} \quad + \quad \mathbf{B}$$

$$\begin{pmatrix} 50 & -25 & 0 & 0 \\ -25 & 50 & -25 & 0 \\ 0 & -25 & 50 & -25 \\ 0 & 0 & -25 & 50 \end{pmatrix} = \begin{pmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 50 \end{pmatrix} + \begin{pmatrix} 0 & -25 & 0 & 0 \\ -25 & 0 & -25 & 0 \\ 0 & -25 & 0 & -25 \\ 0 & 0 & -25 & 0 \end{pmatrix}$$

$$\underline{x} = -\mathbf{D}^{-1} \cdot \mathbf{B} \cdot \underline{x} + \mathbf{D}^{-1} \cdot \underline{b}$$

$$\underline{x} = \mathbf{T} \cdot \underline{x} + \underline{c}$$

$$\underline{c} = \qquad \mathbf{D}^{-1} \qquad \cdot \qquad \underline{b}$$

$$\underline{c} = \begin{pmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 50 \end{pmatrix}^{-1} \cdot \begin{Bmatrix} 10 \\ 20 \\ 20 \\ 10 \end{Bmatrix} = \begin{pmatrix} & 0 & 0 & 0 \\ 0 & & 0 & 0 \\ 0 & 0 & & 0 \\ 0 & 0 & 0 & \end{pmatrix} \cdot \begin{Bmatrix} 10 \\ 20 \\ 20 \\ 10 \end{Bmatrix} = \begin{Bmatrix} \\ \\ \\ \\ \end{Bmatrix}$$

$$\mathbf{T} = \qquad -\mathbf{D}^{-1} \qquad \cdot \qquad \mathbf{B}$$

$$\mathbf{T} = -\begin{pmatrix} & 0 & 0 & 0 \\ 0 & & 0 & 0 \\ 0 & 0 & & 0 \\ 0 & 0 & 0 & \end{pmatrix} \cdot \begin{pmatrix} 0 & -25 & 0 & 0 \\ -25 & 0 & -25 & 0 \\ 0 & -25 & 0 & -25 \\ 0 & 0 & -25 & 0 \end{pmatrix} = \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

$$\mathbf{A}\cdot\underline{x}=\underline{b}$$

$$\begin{pmatrix} 50 & -25 & 0 & 0 \\ -25 & 50 & -25 & 0 \\ 0 & -25 & 50 & -25 \\ 0 & 0 & -25 & 50 \end{pmatrix}\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix}=\begin{Bmatrix} 10 \\ 20 \\ 20 \\ 10 \end{Bmatrix}$$

$$\underline{x}=-\mathbf{D}^{-1}\cdot\mathbf{B}\cdot\underline{x}+\mathbf{D}^{-1}\cdot\underline{b}$$

$$\underline{x}=\qquad\mathbf{T}\qquad\cdot\underline{x}+\qquad\underline{c}$$

$$\underline{x}=\begin{pmatrix} 0 & 25/50 & 0 & 0 \\ 25/50 & 0 & 25/50 & 0 \\ 0 & 25/50 & 0 & 25/50 \\ 0 & 0 & 25/50 & 0 \end{pmatrix}\cdot\begin{Bmatrix} x_1 \\ x_1 \\ x_3 \\ x_4 \end{Bmatrix}+\begin{Bmatrix} 10/50 \\ 20/50 \\ 20/50 \\ 10/50 \end{Bmatrix}$$

Proceso Iterativo de JACOBI

$$\underline{x}^{(k+1)} = \mathbf{T} \cdot \underline{x}^{(k)} + \underline{c}$$

$$\underline{x}^{(k+1)} = \begin{pmatrix} 0 & 25/50 & 0 & 0 \\ 25/50 & 0 & 25/50 & 0 \\ 0 & 25/50 & 0 & 25/50 \\ 0 & 0 & 25/50 & 0 \end{pmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix}^{(k)} + \begin{Bmatrix} 10/50 \\ 20/50 \\ 20/50 \\ 10/50 \end{Bmatrix}$$

Para un vector inicial $\underline{x}^{(0)}$ arbitrario

Iter.	x_1	x_2	x_3	x_4
0	**1,5**	**2,5**	**2,5**	**1,5**
1	(25/50)*(2,5)+10/50	(25/50)*(1,5)+(25/50)*(2,5)+20/50	(25/50)*(2,5)+(25/50)*(1,5)+20/50	(25/50)*(2,5)+10/50
1	**1,45**	**2,4**	**2,4**	**1,45**
2				
3				
4				

Se puede paralelizar ¡!

Proceso Iterativo de JACOBI

$$\underline{x}^{(k+1)} = \mathbf{T} \cdot \underline{x}^{(k)} + \underline{c}$$

$$\underline{x}^{(K+1)} = \begin{pmatrix} 0 & 25/50 & 0 & 0 \\ 25/50 & 0 & 25/50 & 0 \\ 0 & 25/50 & 0 & 25/50 \\ 0 & 0 & 25/50 & 0 \end{pmatrix} \cdot \begin{Bmatrix} x_1 \\ x_1 \\ x_3 \\ x_4 \end{Bmatrix}^{(K)} + \begin{Bmatrix} 10/50 \\ 20/50 \\ 20/50 \\ 10/50 \end{Bmatrix}$$

Iter.	x_1	x_2	x_3	x_4	***dif_x₁***	***dif_x₂***	***dif_x₃***	***dif_x₄***	*Norma dif*
0	**1,5**	**2,5**	**2,5**	**1,5**	**--------**	**--------**	**---------**	**---------**	**---------**
1	**1,45**	**2,4**	**2,4**	**1,45**	**1,45 -1,5**	**2,4 -2,5**	**-0,1**	**-0,05**	**0,1**
2			**2,325**						**0,075**
3	**1,3625**								**0,0625**

Norma de un vector: Cuadrática o Infinito.

SISTEMAS DE ECUACIONES LINEALES. MÉTODO DE GAUSS SEIDEL

A partir del método iterativo de Jacobi, cuyo fórmula de recurrencia está dada por

$$\underline{x}^{(k+1)} = \mathbf{T} \cdot \underline{x}^{(k)} + \underline{c}$$

Se puede iterar con

$$\underline{x}^{(k+1)} = \mathbf{Tl} \cdot \underline{x}^{(k+1)} + \mathbf{Ts} \cdot \underline{x}^{(k)} + \underline{c}$$

Hasta encontrar que el ERROR es tan pequeño como se quiera. Siendo

$$\mathbf{Tl} = \begin{pmatrix} 0 & 0 & 0 & .... & 0 \\ -a_{21}/a_{22} & 0 & 0 & .... & 0 \\ -a_{31}/a_{33} & -a_{32}/a_{33} & 0 & .... & 0 \\ .... & .... & .... & 0 & .... \\ -a_{N1}/a_{NN} & -a_{N2}/a_{NN} & -a_{N3}/a_{NN} & .... & 0 \end{pmatrix} \quad \underline{x}^{(k+1)} = \begin{Bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \\ x_3^{(k+1)} \\ ... \\ x_N^{(k+1)} \end{Bmatrix}$$

$$\mathbf{Ts} = \begin{pmatrix} 0 & -a_{12}/a_{11} & -a_{13}/a_{11} & .... & -a_{1N}/a_{11} \\ 0 & 0 & -a_{23}/a_{22} & .... & -a_{2N}/a_{22} \\ 0 & 0 & 0 & .... & -a_{3N}/a_{33} \\ .... & .... & .... & 0 & .... \\ 0 & 0 & 0 & .... & 0 \end{pmatrix} \quad \underline{x}^{(k)} = \begin{Bmatrix} x_1^{(k)} \\ x_2^{(k)} \\ x_3^{(k)} \\ ... \\ x_N^{(k)} \end{Bmatrix}$$

Se debe destacar que:

- para calcular x_1, participa todo el vector **x** de la iteración anterior.
- Para calcular x_2, participa x_1 de la iteración actual (recién calculado)y todas las demás componentes del vector **x** de la iteración anterior.
- Para calcular x_3, participa x_1 y x_2 de la iteración actual (recién calculadas)y todas las demás componentes del vector **x** de la iteración anterior.
- Para calcular x_4, participa x_1 ,x_2 y x_3 de la iteración actual (recién calculadas)y todas las demás componentes del vector **x** de la iteración anterior.
- Así se sigue hasta calcular x_N con todas las componentes de la iteración actual del vector x (recién calculadas), desde la 1 hasta la N-1.

SISTEMAS DE ECUACIONES LINEALES. MÉTODO DE GAUSS SEIDEL

Se itera con
$$\underline{x}^{(k+1)} = \mathbf{Tl}\cdot\underline{x}^{(k+1)} + \mathbf{Ts}\cdot\underline{x}^{(k)} + \underline{c}$$

$$\mathbf{Tl} = \begin{pmatrix} 0 & 0 & 0 & .... & 0 \\ -a_{21}/a_{22} & 0 & 0 & .... & 0 \\ -a_{31}/a_{33} & -a_{32}/a_{33} & 0 & .... & 0 \\ .... & .... & .... & 0 & .... \\ -a_{N1}/a_{NN} & -a_{N2}/a_{NN} & -a_{N3}/a_{NN} & .... & 0 \end{pmatrix} \qquad \underline{x}^{(k+1)} = \begin{Bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \\ x_3^{(k+1)} \\ ... \\ x_N^{(k+1)} \end{Bmatrix}$$

$$\mathbf{Ts} = \begin{pmatrix} 0 & -a_{12}/a_{11} & -a_{13}/a_{11} & .... & -a_{1N}/a_{11} \\ 0 & 0 & -a_{23}/a_{22} & .... & -a_{2N}/a_{22} \\ 0 & 0 & 0 & .... & -a_{3N}/a_{33} \\ .... & .... & .... & 0 & .... \\ 0 & 0 & 0 & .... & 0 \end{pmatrix} \qquad \underline{x}^{(k)} = \begin{Bmatrix} x_1^{(k)} \\ x_2^{(k)} \\ x_3^{(k)} \\ ... \\ x_N^{(k)} \end{Bmatrix}$$

Se debe destacar que:

- para calcular x_1, participa todo el vector **x** de la iteración anterior.
- Para calcular x_2, participa x_1 de la iteración actual (recién calculado) y todas las demás componentes del vector **x** de la iteración anterior.
- Así se sigue hasta calcular x_N con todas las componentes de la iteración actual del vector x (recién calculadas), desde la 1 hasta la N-1.

MÉTODOS ITERATIVOS

Síntesis

	Punto Fijo	Sistema de Ecuaciones Lineales	
Se busca	$x_{k+1} = r$ tal que $f(r) = 0$	$\underline{x}\ \varepsilon\ R^N$ tal que $\underline{f} = \mathbf{A}\cdot\underline{x} - \underline{b} = \underline{0}$	
Inicialización	Se propone una solución inicial x_0		
Recurrencia	$x_{k+1} = g(x_k)$	$\underline{x}^{(k+1)} = \mathbf{T}\cdot\underline{x}^{(k)} + \underline{c}$	$\underline{x}^{(k+1)} = \mathbf{Tl}\cdot\underline{x}^{(k+1)} + \mathbf{Ts}\cdot\underline{x}^{(k)} + \underline{c}$
Control de Detención	$\lvert f(x_{k+1})\rvert \le \varepsilon_f$ o $\lvert x_{k+1} - x_k\rvert \le \varepsilon_a$ o $k \le k\max$		
Actualización	Retiene la última solución aproximada		
CRITERIOS DE CONVERGENCIA	$\left\lvert \left.\dfrac{dg(x)}{dx}\right\rvert_{x=\xi} \right\rvert < 1$	Mayor de los Valores Absolutos de **T** debe ser menor a 1 $\rho(\mathbf{T}) < 1$	Matriz **A** estrictamente diagonal dominante

En lugar de **Valor absoluto** de una variable real se usa **Norma de un Vector** de componentes reales

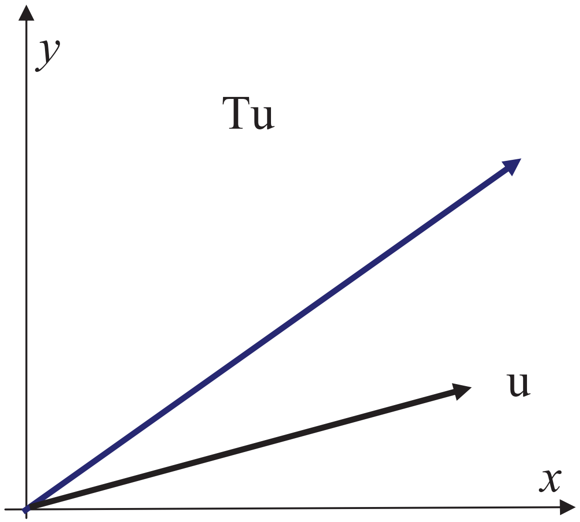
AUTOVALORES Y AUTOVECTORES

Dada una matriz **A,** se le puede asociar una Transformación Lineal, la que tiene *Direcciones Invariantes*, soluciones de

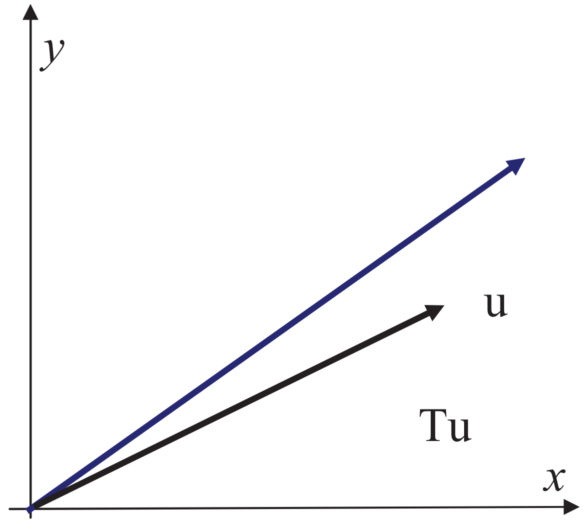
$$(\mathbf{A}-\lambda\cdot\mathbf{I})\underline{x}=\underline{0}$$

Ejemplos

Simetría respecto de una RECTA



Proyección en EJE X en dirección de una RECTA



Dirección Invariante 1
Autovalor

Dirección Invariante 2
Autovalor

AUTOVALORES Y AUTOVECTORES- MÉTODO DE LA POTENCIA

Dada una matriz **A,** se buscan las *Direcciones Invariantes*, soluciones de

$$(\mathbf{A} - \lambda \cdot \mathbf{I})\underline{x} = \underline{0} \quad \text{o bien} \quad \mathbf{A} \cdot \underline{x} = \lambda \cdot \underline{x}$$

Esto es equivalente a encontrar un vector $\underline{y}$ que se puede obtener como $\underline{y} = \mathbf{A} \cdot \underline{x}$

y resulta igual a $\underline{y} = \lambda \cdot \underline{x}$

lo que significa que los vectores $\underline{x}$ e $\underline{y}$ son PARALELOS

El **METODO DE LA POTENCIA** es un algoritmo iterativo que propone un $\underline{y_0}$ arbitrario y

Obtiene un nuevo vector como $\underline{y_{k+1}} = \mathbf{A} \cdot \underline{y_k}$

Sólo se detiene si se cumple que $\underline{y_{k+1}}$ es paralelo a $\underline{y_k}$

EJEMPLO: Dada la matriz **A**

-10	4
-4	0

y0	y1	y2	y3	y4	y5	y6
2	-12	88	-688			
2	-8	48	-352			
iterac	**1**	**2**	**3**	**4**	**5**	**6**
alfa(1)						
alfa(2)						

El **METODO DE LA POTENCIA** es un algoritmo iterativo para resolver $\mathbf{A}\cdot\underline{x}=\lambda\cdot\underline{x}$

que propone un $\underline{y_0}$ arbitrario y se *escala* para obtener el versor $\underline{x_0}$, con el que se inicia el proceso iterativo

Obtener un nuevo vector como $\underline{y_{k+1}}=\mathbf{A}\cdot\underline{x_k}$

Sólo se detiene si se cumple que $\underline{y_{k+1}}$ es paralelo a $\underline{x_k}$

EJEMPLO: Dada la matriz **A**

-10	4
-4	0

	y0	y1	y2	y3	y4	y5	y6
	2	-6					
	2	-4					
norma	2	-6					
	x0	x1	x2	x3	x4	x5	x6
	1	1					
	1	0,66666667					
	iterac	1	2	3	4	5	6
	alfa(1)						
	alfa(2)						
	num						
	den						
	rho						

MÉTODOS ITERATIVOS

Se busca $\underline{x} \ \varepsilon \ R^N$ tal que

	Autovalores	Sistema de Ecuaciones Lineales
	$\underline{f} = \mathbf{A} \cdot \underline{x} - \lambda \cdot \underline{x} = \underline{0}$	$\underline{f} = \mathbf{A} \cdot \underline{x} - \underline{b} = \underline{0}$
Inicialización	Se propone una solución inicial x_0	
Recurrencia	$\underline{y}_{k+1} = \mathbf{A} \cdot \underline{x}_k$ $\underline{x}_{k+1} = \underline{y}_{k+1} \cdot \left(1 / \left\Vert \underline{y}_{k+1} \right\Vert\right)$	$\underline{x}^{(k+1)} = \mathbf{T} \cdot \underline{x}^{(k)} + \underline{c}$
Control de Detención	$\left\vert f(x_{k+1}) \right\vert \leq \varepsilon_f$ o $\left\vert x_{k+1} - x_k \right\vert \leq \varepsilon_a$ o $k \leq k\max$	
Actualización	Retiene la última solución aproximada	
CRITERIOS DE CONVERGENCIA	La matriz **A** debe ser Diagonalizable. Sus autovectores deben ser linealmente independientes	Mayor de los Valores Absolutos de **T** debe ser menor a 1 $\rho(\mathbf{T}) < 1$ Matriz **A** estrictamente diagonal dominante

En lugar de **Valor absoluto** de una variable real se usa **Norma de un Vector** de componentes reales

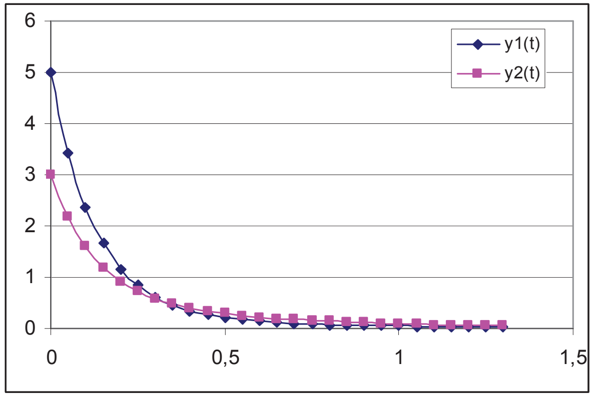
AUTOVALORES Y AUTOVECTORES- APLICACIÓN

Sea el sistema EDO

$$\underline{\dot{\mathbf{y}}}(t)=\mathbf{A}\cdot\underline{\mathbf{y}}(t) \qquad \text{con} \qquad \mathbf{y}(0)=\begin{Bmatrix}5\\3\end{Bmatrix}, \qquad \mathbf{A}=\begin{pmatrix}-10 & 4\\ -4 & 0\end{pmatrix}$$

admite como solución $\underline{\mathbf{y}(t)} \quad = \underline{\mathbf{v}}\cdot e^{\lambda\cdot t}$ siendo $\underline{\mathbf{v}}$, λ la solución del siguiente sistema de autovalores y autovectores

$$\begin{bmatrix}-10 & 4\\ -4 & 0\end{bmatrix}\begin{Bmatrix}v_1\\ v_2\end{Bmatrix}=\lambda\cdot\begin{Bmatrix}v_1\\ v_2\end{Bmatrix}$$

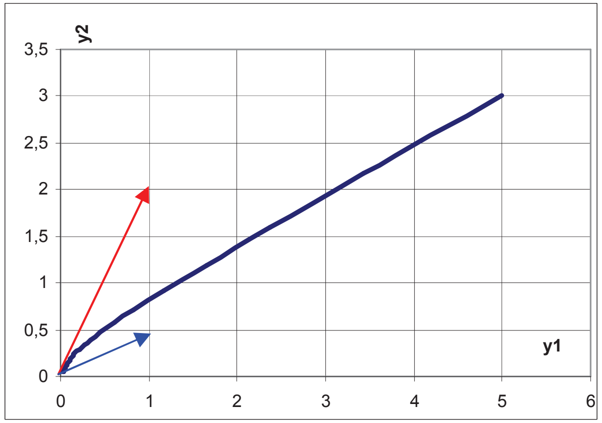


La solución general será la combinación lineal

$$\underline{\mathbf{y}(t)}=\underline{\mathbf{v}_1}\cdot e^{\lambda_1\cdot t}+\underline{\mathbf{v}_2}\cdot e^{\lambda_2\cdot t}$$

que cumple con las condiciones iniciales.

Es decir, $\underline{\mathbf{y}(t)}=\dfrac{1}{3}\begin{Bmatrix}1\\2\end{Bmatrix}\cdot e^{-2\cdot t}+\dfrac{14}{3}\begin{Bmatrix}1\\0.5\end{Bmatrix}\cdot e^{-8\cdot t}$



AUTOVALORES Y AUTOVECTORES- MÉTODO DE LA POTENCIA-CONVERGENCIA

Dada una matriz $\mathbf{A} \, \varepsilon \, \mathbf{R}^{\mathbf{NxN}}$**,** tal que

Sus autovectores $\underline{v}_1, \underline{v}_2, \ldots \underline{v}_N$, y sus autovalores $\lambda_1, \lambda_2, \lambda_3, \ldots. \lambda_N$, verifican que

$$\mathbf{A} \,.\, \underline{v}_k = \lambda_k \,.\, \underline{v}_k \qquad \text{con k=1 a N}$$

Es diagonalizable: es decir tiene N autovectores linealmente independientes que forman base de R^N.

Es posible ordenar los autovalores de mayor a menor considerando sus valores absolutos en la forma:

$$|\lambda_1| > |\lambda_2| \;\; > |\lambda_3| > \ldots\ldots\ldots. |\lambda_N|$$

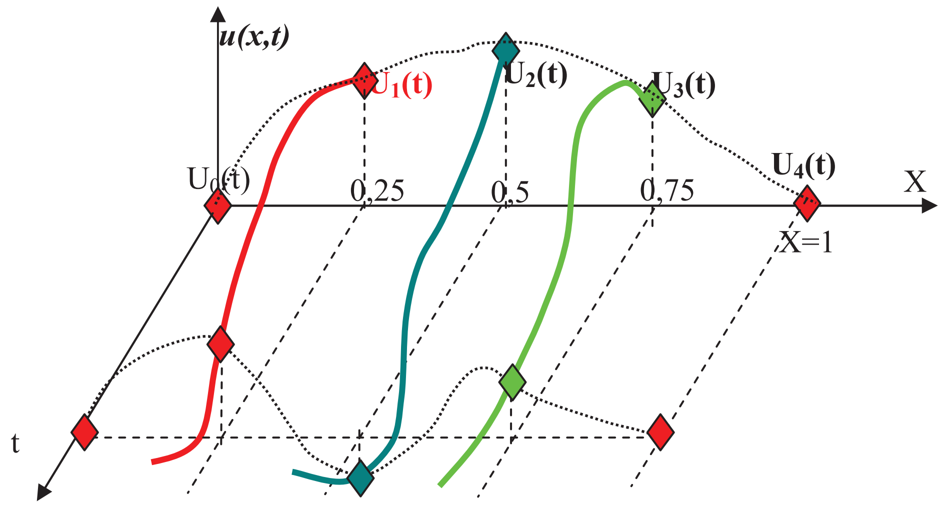
Entonces el algoritmo del método de la potencia converge

AUTOVALORES Y AUTOVECTORES- APLICACIÓN

Se busca *u(x,t)* solución de

$$-12\frac{\partial^2 u(x,t)}{\partial x^2}+x^2\frac{\partial^2 u(x,t)}{\partial t^2}=0 \quad en \quad \Omega=\{x\,\varepsilon\, R:0\le x\le 1\}$$

$$u(0,t)=0 \qquad\qquad u(1,t)=0$$



Se plantea encontrar una ***solución aproximada en forma de función discreta en la variable x, aunque continua en la variable t***. Se pretende encontrar las funciones $U_k(t)=u(X_k,t)$ con k=0,N, en N+1 puntos elegidos del dominio x, identificados con su abscisa X_k.

$$\mathbf{M}\cdot\ddot{\mathbf{u}}(t)+\mathbf{K}\cdot\mathbf{u}(t)=\mathbf{0}$$

con

$$\mathbf{K}=\frac{12}{0{,}25^2}\begin{bmatrix}2 & -1 & 0\\ -1 & 2 & -1\\ 0 & -1 & 2\end{bmatrix};\quad \mathbf{M}=\begin{bmatrix}0{,}25^2 & 0 & 0\\ 0 & 0{,}50^2 & 0\\ 0 & 0 & 0{,}75^2\end{bmatrix}\qquad \mathbf{u}(t)=\begin{Bmatrix}U_1(t)\\ U_2(t)\\ U_3(t)\end{Bmatrix}$$

Que admite una solución del tipo

$$\mathbf{u}(t)=\underline{x}\cdot e^{I\,\omega t}$$

Que resulta en

$$\cdot(\mathbf{K}-\omega^2\mathbf{M})\cdot\underline{x}=\mathbf{0}$$

INTERPOLACIÓN Y APROXIMACIÓN DE FUNCIONES

Planteo de Problema

Interpolación de funciones
- Método Directo
- Método con Polinomios de Lagrange
- Método con Polinomios de Newton

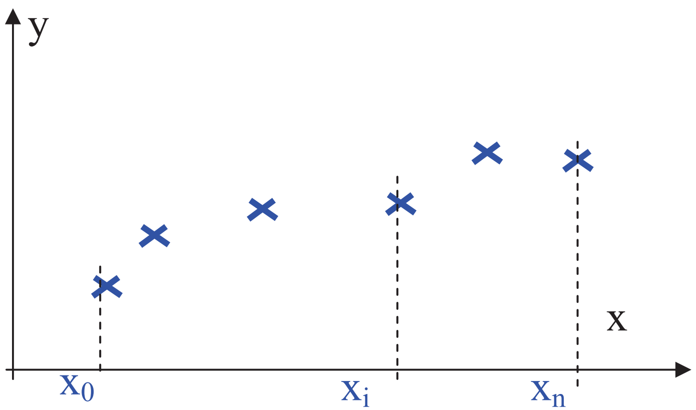
Aproximación de funciones
- Método de Mínimos Cuadrados

Ejemplos

Resumen

INTERPOLACIÓN Y APROXIMACIÓN DE FUNCIONES DISCRETAS

Dada una **función discreta**, **y=f(x)** de $R \rightarrow R$
definida mediante **(n+1) puntos** $(x_i; y_i=f(x_i))$ con **i=0,n**.
Se propone $P_m(x) = \Sigma\, a_k\, \phi_k(x)$ con k=0,m.
Donde a_k son las incógnitas; y $\{\phi_k(x)\}$ es una **Base elegida**



Se define: **Residuo** $r_i = f(x_i) - P_m(x_i)$ con k=0,m; i=0,n

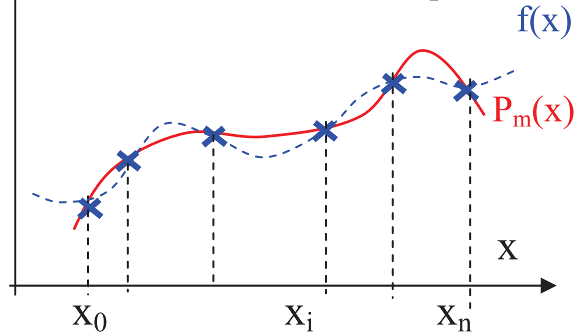
$$r_i = y_i - \Sigma\, a_k\, \phi_k(x_i)$$

$$\begin{Bmatrix} r_0 \\ r_1 \\ r_2 \\ \cdots \\ r_n \end{Bmatrix} = \begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ \cdots \\ y_n \end{Bmatrix} - \begin{pmatrix} \phi_0(x_0) & \phi_1(x_0) & \phi_2(x_0) & \cdots & \phi_m(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_m(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_m(x_2) \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_0(x_n) & \phi_1(x_n) & \phi_2(x_n) & \cdots & \phi_m(x_n) \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \cdots \\ a_m \end{Bmatrix}$$

$$\underline{r} = \underline{y} - \Phi \cdot \underline{a}$$

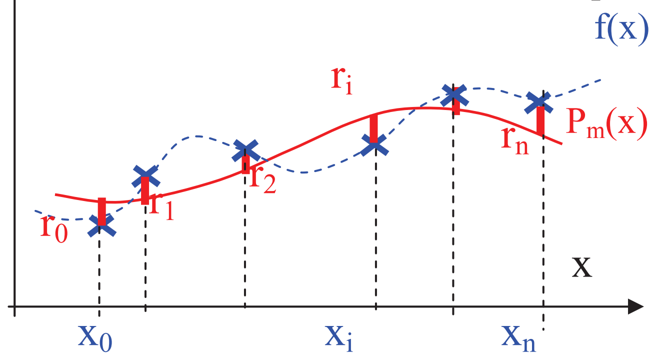
INTERPOLACIÓN

Forma Fuerte: $\mathbf{r_i=0}$, para todo x_i



APROXIMACIÓN

Forma Débil: los r_i “son nulos en promedio



INTERPOLACIÓN: MÉTODO DIRECTO

Dada una **función discreta**, $y=f(x)$ de $R \rightarrow R$ definida mediante **(n+1) puntos** $(x_i; y_i=f(x_i))$ con $i=0,n$.

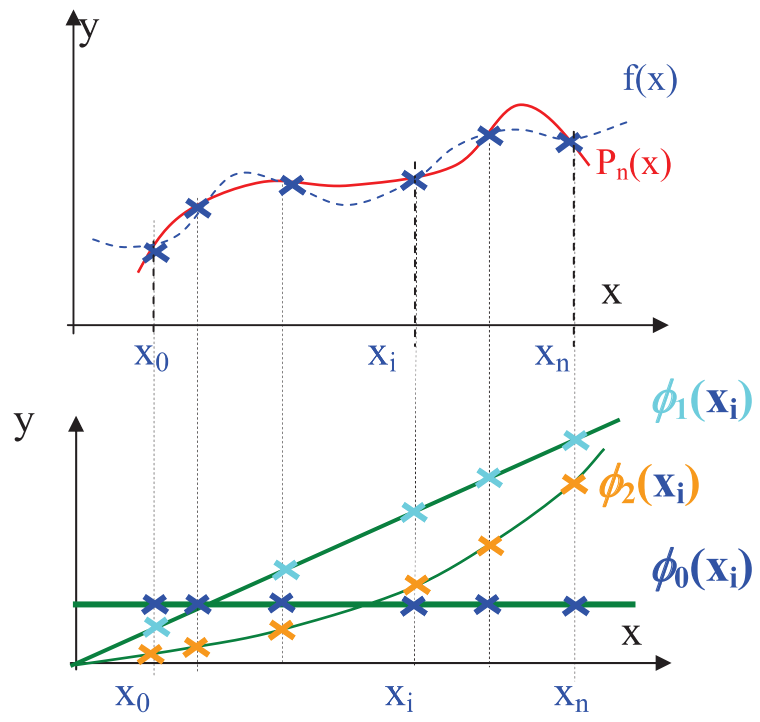
Se adopta como $\boxed{\text{Base} = \{1, x, x^2, x^3, \ldots\ldots, x^n\}}$

Se propone $P_n(x) = \Sigma\, (a_k\ x^k)$ con $k=0,n$

Se determinan los a_k tales que **el Residuo sea nulo**, es decir

$$\begin{Bmatrix} r_0 \\ r_1 \\ r_2 \\ \ldots \\ r_n \end{Bmatrix} = \begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ \ldots \\ y_n \end{Bmatrix} - \begin{pmatrix} \phi_0(x_0) & \phi_1(x_0) & \phi_2(x_0) & \ldots. & \phi_m(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \ldots. & \phi_m(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \phi_2(x_2) & \ldots. & \phi_m(x_2) \\ \ldots. & \ldots. & \ldots. & \ldots. & \ldots. \\ \phi_0(x_n) & \phi_1(x_n) & \phi_2(x_n) & \ldots. & \phi_m(x_n) \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \ldots \\ a_m \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ \ldots \\ 0 \end{Bmatrix}$$

$$\underline{r} = \underline{y} - \Phi \cdot \underline{a} = \underline{0}$$



Que resulta en el sistema de ecuaciones lineales

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots. & x_0^n \\ 1 & x_1 & x_1^2 & \ldots. & x_1^n \\ 1 & x_2 & x_2^2 & \ldots. & x_2^n \\ \ldots. & \ldots. & \ldots. & \ldots. & \ldots. \\ 1 & x_n & x_n^2 & \ldots. & x_n^n \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \ldots \\ a_n \end{Bmatrix} = \begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ \ldots \\ y_n \end{Bmatrix}$$

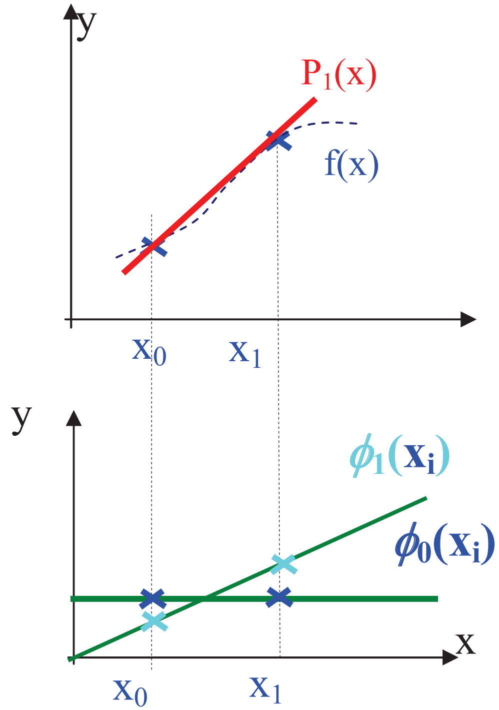
para que sea Solución Única; m=n y todos x distintos!!!!!

Resulta el polinomio interpolante $\boxed{P_n(x) = \Sigma\, (a_k\ x^k) \quad \text{con } k=0,n}$

INTERPOLACIÓN: MÉTODO DIRECTO *EJEMPLO*

Dada una **función discreta**, $y=f(x)$ de $R \to R$ definida mediante **(2) puntos (x_i; $y_i=f(x_i)$)** con $i=0,1$.

x	y=f(x)
1,5	3
3	4



Se adopta como Base = $\{1, x\}$

con lo que la matriz Φ y el sistema a resolver resultan:

$$\begin{pmatrix} \quad & \quad \\ \quad & \quad \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} \quad \\ \quad \end{Bmatrix}$$

de donde se tiene que $a_0 =$ $\qquad$ $a_1 =$

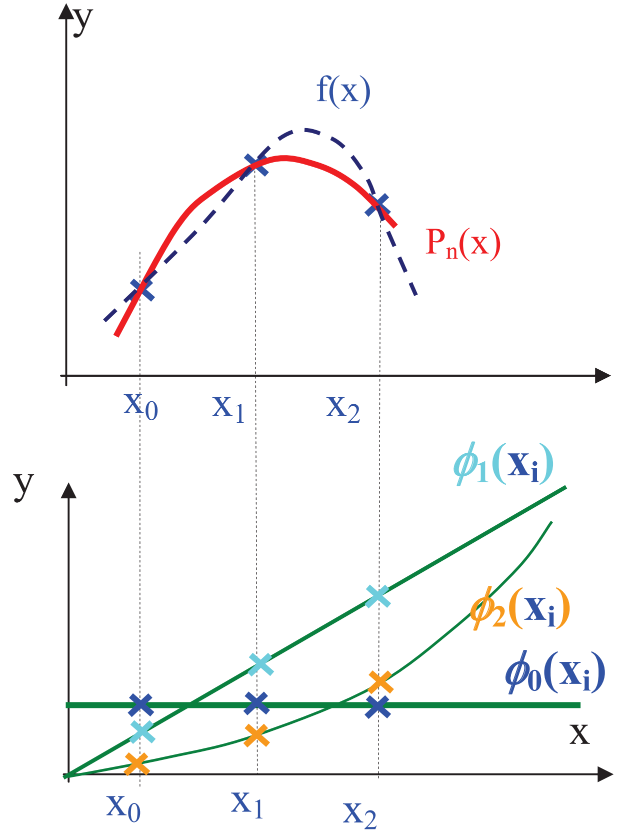
El **Polinomio Interpolante** usando la **Base de Polinomios elementales** $\{1, x\}$ es

$$\boxed{P_1(x) = \qquad 1 + \qquad x}$$

INTERPOLACIÓN: MÉTODO DIRECTO *EJEMPLO*

Dada una **función discreta**, $y=f(x)$ de R → R definida mediante **(3) puntos (x_i; $y_i=f(x_i)$)** con i=0,2.

x	y=f(x)
1,5	3
3	4
4,5	3,5



Se adopta como Base = $\{1, x, x^2\}$

con lo que la matriz Φ y el sistema a resolver resultan:

$$\begin{pmatrix} \quad & \quad & \quad \\ & & \\ & & \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \quad \\ \quad \\ \quad \end{Bmatrix}$$

de donde se tiene que $a_0 =$ $a_1 =$ $a_2 =$

El **Polinomio Interpolante** usando la **Base de Polinomios elementales** $\{1, x, x^2\}$ es

$$\boxed{P_2(x) = \qquad 1 + \qquad x + \qquad + x^2}$$

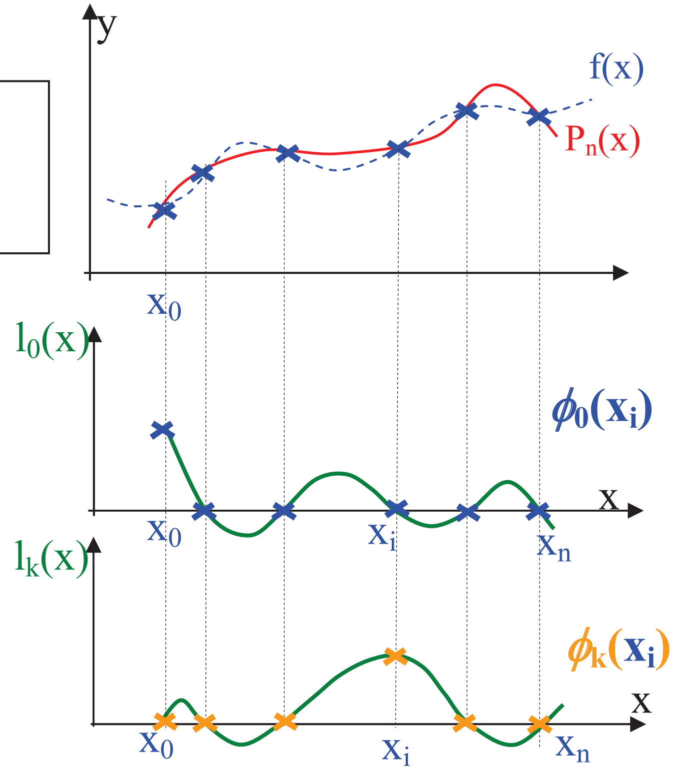
INTERPOLACIÓN: MÉTODO DE POLINOMIOS DE LAGRANGE

Dada una **función discreta**, $y=f(x)$ de $R \to R$ definida mediante **(n+1) puntos** $(x_i; y_i=f(x_i))$ con $i=0,n$.

Se adopta como $\boxed{\text{Base} = \{l_0(x),\ l_1(x),\ l_2(x), \ldots\ldots, l_n(x)\},}$

Los **polinomios de Lagrange** $l_i(x)$ se definen como,

$$l_i(x) = \frac{(x-x_1)(x-x_2)(x-x_3)\ldots\ldots\ldots(x-x_n)}{(x_i-x_1)(x_i-x_2)(x_i-x_3)\ldots\ldots\ldots(x_i-x_n)}, \qquad l_i(x) = \prod_{\substack{k=0 \\ k\neq i}}^{n} \frac{(x-x_k)}{(x_i-x_k)}.$$



Son tales que para los i, k abscisas datos

$l_i(x_i)=1$ y $l_i(x_k)=0$ con $k\neq i$

Se determinan los a_k imponiendo

que el Residuo sea nulo, $\underline{r} = \underline{y} - \Phi \cdot \underline{a} = \underline{0}$

De donde resulta

$$\begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 1 \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \ldots \\ a_n \end{Bmatrix} = \begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ \ldots \\ y_n \end{Bmatrix}$$

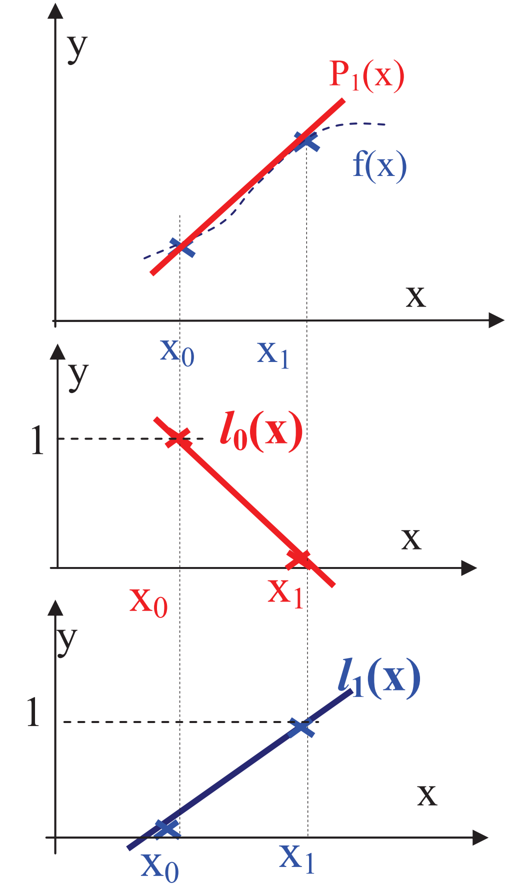
NO HAY QUE RESOLVER SISTEMA DE ECUACIONES!!!!

y resulta el polinomio interpolante $\boxed{P_n(x) = \Sigma\ (y_k\ l_k(x)) \text{ con } k=0,n}$

INTERPOLACIÓN: MÉTODO DE POLINOMIOS DE LAGRANGE *EJEMPLO*

Dada una **función discreta**, y=f(x) de R → R definida mediante **(2) puntos (x_i; $y_i=f(x_i)$)** con i=0,1.

x	y=f(x)
1,5	3
3	4



$$l_0(x) = \underline{\qquad\qquad}$$

Base = $\{l_0(x), l_1(x)\}$ con $l_1(x) = \underline{\qquad\qquad}$

y así la matriz Φ y el sistema a resolver resultan:

$$\begin{pmatrix} & \\ & \end{pmatrix}\begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} \\ \end{Bmatrix}$$

de donde se tiene que $a_0 =$ $\qquad\qquad$ $a_1 =$

El **Polinomio Interpolante** usando la **Base de Polinomios de Lagrange** es

$$\boxed{P_1(x) = \qquad\qquad\qquad\qquad}$$

INTERPOLACIÓN: MÉTODO DE POLINOMIOS DE LAGRANGE *EJEMPLO*

Dada una **función discreta**, y=f(x) de R → R definida mediante **(3) puntos (x_i; $y_i=f(x_i)$)** con i=0,2.

x	y=f(x)
1,5	3
3	4
4,5	3,5

$l_0(x) =$ ____________

$l_1(x) =$ ____________

$l_2(x) =$ ____________

Con la Base = $\{l_0(x), l_1(x), l_2(x)\}$ resulta

$$\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}\begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \\ \\ \end{Bmatrix}$$

de donde se tiene que $a_0 =$ $a_1 =$ $a_2 =$

El **Polinomio Interpolante** usando la **Base de Polinomios de Lagrange** $\{ l_0(x), l_1(x), l_2(x)\}$ es

$P_2(x) =$

AL AGREGAR UN PUNTO SE DEBE HACER TODO DE NUEVO

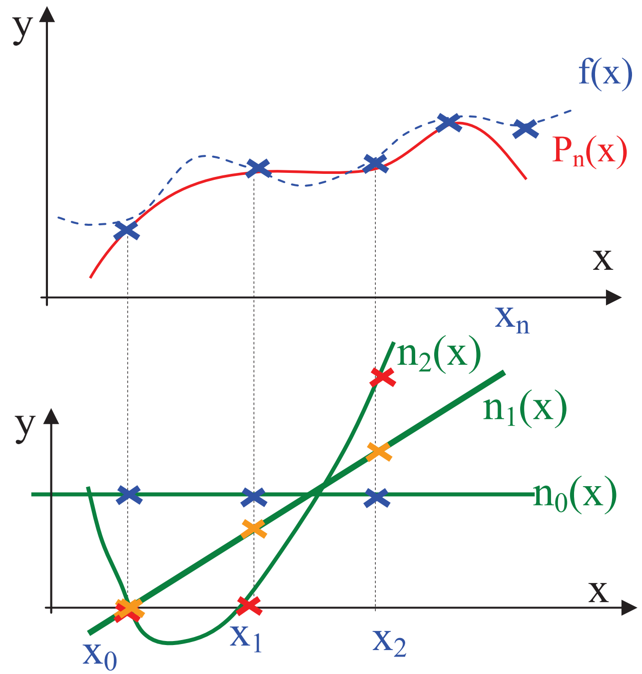
INTERPOLACIÓN: MÉTODO DE POLINOMIOS DE NEWTON

Dada una **función discreta**, $y=f(x)$ de $R \to R$ definida mediante **(n+1) puntos** $(x_i; y_i=f(x_i))$ con $i=0,n$.

Se toma como base a los llamados **polinomios de Newton**.
Estos tienen como particularidad que se basan en los polinomios bases anteriores.

$n_0(x) = 1$	$n_0(x) = 1$
$n_1(x) = n_0(x)\,(x-x_0)$	$n_1(x) = 1(x-x_0)$
$n_2(x) = n_1(x)\,(x-x_1)$	$n_2(x) = 1(x-x_0)\,(x-x_1)$
$n_3(x) = n_2(x)\,(x-x_2)$	$n_3(x) = 1(x-x_0)(x-x_1)(x-x_2)$

$n_k(x) = n_{k-1}(x)\cdot(x-x_{k-1})$, para todo $k\geq 1$



Con la Base = $\{n_0(x),\ n_1(x),\ n_2(x),\ldots\ldots,\ n_n(x)\}$,

Se determinan los a_k tales que **el Residuo sea nulo**, $\underline{r} = \underline{y} - \Phi\cdot\underline{a} = \underline{0}$

y resulta

$$\begin{pmatrix} 1 & 0 & 0 & \ldots\ldots\ldots & 0 \\ 1 & x_1-x_0 & 0 & \ldots\ldots\ldots & 0 \\ 1 & x_2-x_0 & (x_2-x_0)(x_2-x_1) & \ldots\ldots\ldots & 0 \\ .. & \ldots\ldots\ldots & \ldots\ldots\ldots\ldots\ldots\ldots & \ldots\ldots & 0 \\ 1 & x_n-x_0 & (x_n-x_0)(x_n-x_1) & \ldots\ldots\ldots & (x_n-x_0)(x_n-x_1)\ldots..(x_n-x_{n-1}) \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \ldots \\ a_n \end{Bmatrix} = \begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ \ldots \\ y_n \end{Bmatrix}$$

De donde: $a_0 = y_0$, $\quad a_1 = \dfrac{y_1-y_0}{x_1-x_0}$, $\quad a_2 = \dfrac{\dfrac{y_2-y_1}{x_2-x_1} - \dfrac{y_1-y_0}{x_1-x_0}}{x_2-x_0}$

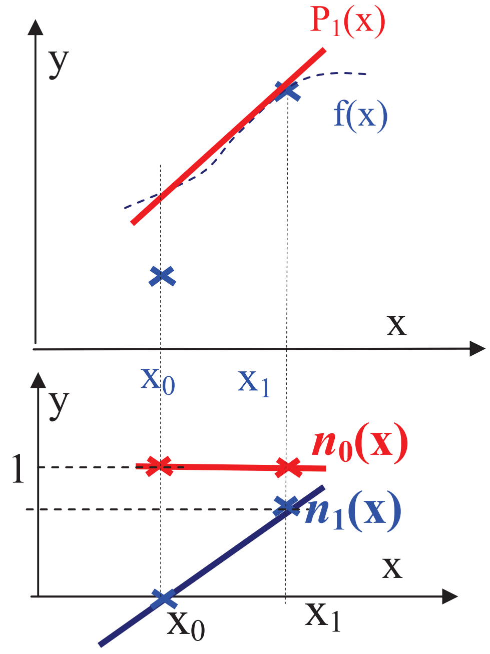
Resulta el polinomio interpolante

$$\boxed{P_n(x) = \Sigma\,(a_k\ n_k(x)) \qquad \text{con } k=0,1,2,\ldots n}$$

INTERPOLACIÓN: MÉTODO DE POLINOMIOS DE NEWTON — *EJEMPLO*

Dada una **función discreta**, $y=f(x)$ de $R \to R$ definida mediante **(2) puntos (x_i; $y_i=f(x_i)$)** con i=0,1.

x	y=f(x)	Δ	Δ^2
1,5	3		
3	4		



Base = $\{\mathbf{n_0(x), n_1(x)}\}$ con $n_0(x) = 1$, $n_1(x) =$

y así la matriz Φ y el sistema a resolver resultan:

$$\begin{pmatrix} & \\ & \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} \\ \end{Bmatrix}$$

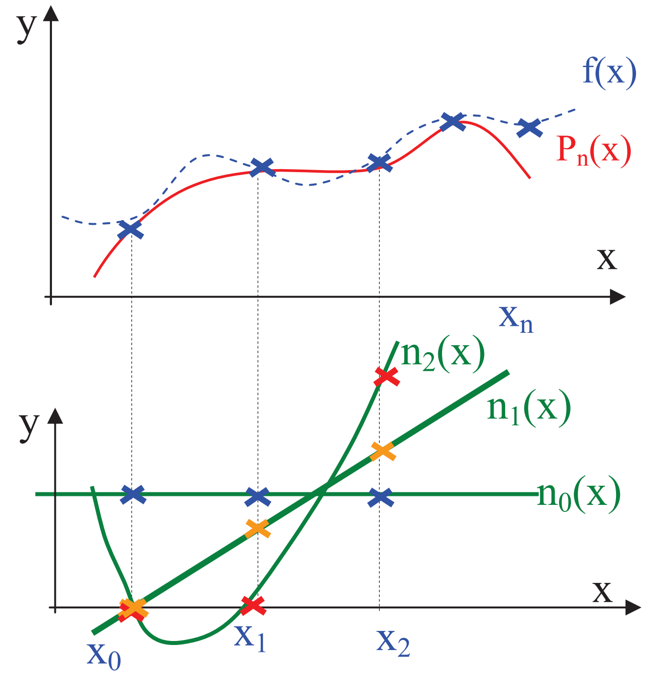
de donde se tiene que $a_0 =$ $a_1 =$

El **Polinomio Interpolante** usando la **Base de Polinomios de Newton** es

$$\boxed{P_1(x) =}$$

Dada una **función discreta**, $y=f(x)$ de R → R definida mediante **(3) puntos (x_i; $y_i=f(x_i)$)** con i=0,2.

x	y=f(x)	Δ	Δ^2
1,5	3		
3	4		
4,5	3,5		



Base = $\{\mathbf{n_0(x), n_1(x), n_2(x)}\}$

con

$$n_0(x) = 1$$

$$n_1(x) =$$

$$n_2(x) =$$

la Φ y el sistema a resolver resultan:

$$\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}\begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \\ \\ \\ \end{Bmatrix}$$

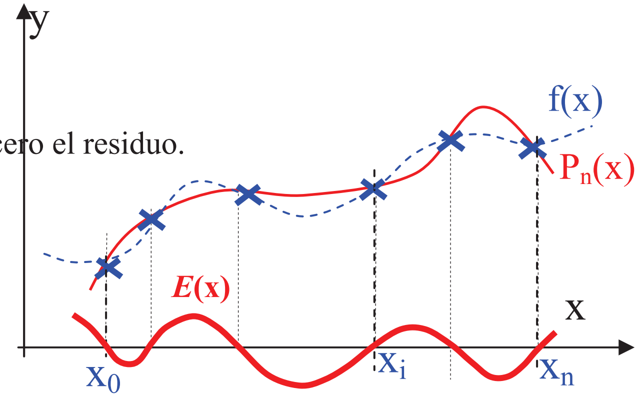
de donde se tiene que $a_0 =$ $a_1 =$ $a_2 =$

El **Polinomio Interpolante** usando la **Base de Polinomios de Newton** es

$$P_2(x) =$$

FUNCIÓN ERROR DE INTERPOLACIÓN

Dados (n+1) puntos de coordenadas **(x_i; $y_i=f(x_i)$)** con i=0, n, la ***Función Error de Interpolación*** ***E*****(x)** es la diferencia entre la función f(x) y el polinomio de interpolación $P_n(x)$, y se puede expresar en la forma:



$$E(x) = f(x) - P_n(x) = f(x) - \Sigma\, a_k\, \phi_k(x),$$

con **ϕ_k(x) n funciones bases conocidas** (Lagrange, Newton, etc); y a_k son los coeficientes que hacen cero el residuo.
La función ***Error de interpolación E*****(x)** tiene **(n+1) ceros**, en cada uno de los x_i datos,

por lo que **se puede expresar como un polinomio de al menos grado (n+1)**, como

$$E(x) = C\,(x-x_0)\,(x-x_1)\,(x-x_2)\ldots\ldots(x-x_n)$$

La C se determinará de modo que la función auxiliar W(x) sea cero para todo valor de x.

$$W(x) = f(x) - P_n(x) - E(x)$$

La constante C se determinará de modo que la igualdad $f(x)=P_n(x)+E(x)$, se cumpla para cualquier valor de x. La función *W* (x) vale cero en cada x_k dato (k=0,n), es decir tiene n+1 ceros. Pero para cualquier otro $x_i \neq x_k$, se elige C de modo que *W* (x_i)=0, entonces

W(x) tiene (n+2) ceros, para cada x_i; → $\dfrac{d^1W}{dx^1}$ tiene (n+2-1) ceros, para cada x_i; → $\dfrac{d^2W}{dx^2}$ tiene (n+2-2) ceros, para cada x_i;

……. $\dfrac{d^{n+1}W}{dx^{n+1}}$ tiene (n+2-(n+1)) cero, para cada x_i. Esta derivada es: $\dfrac{d^{n+1}W}{dx^{n+1}} = \dfrac{d^{n+1}f}{dx^{n+1}} - \dfrac{d^{n+1}P_n}{dx^{n+1}} - \dfrac{d^{n+1}E}{dx^{n+1}}$ →

$$\frac{d^{n+1}W}{dx^{n+1}} = \frac{d^{n+1}f}{dx^{n+1}} - 0 - C\cdot(n+1)!$$

Así para cada valor $x_i \neq x_k$ existe una C que hace el error de interpolación se pueda expresar

$$E(x) = \left.\frac{d^{n+1}f(x)}{dx^{n+1}}\right|_{X=\xi} \frac{1}{(n+1)!}(x-x_0)(x-x_1)(x-x_2)\ldots\ldots\ldots\ldots(x-x_n) \qquad \text{con } \xi\ \varepsilon\ (x_0,\, x_n),$$

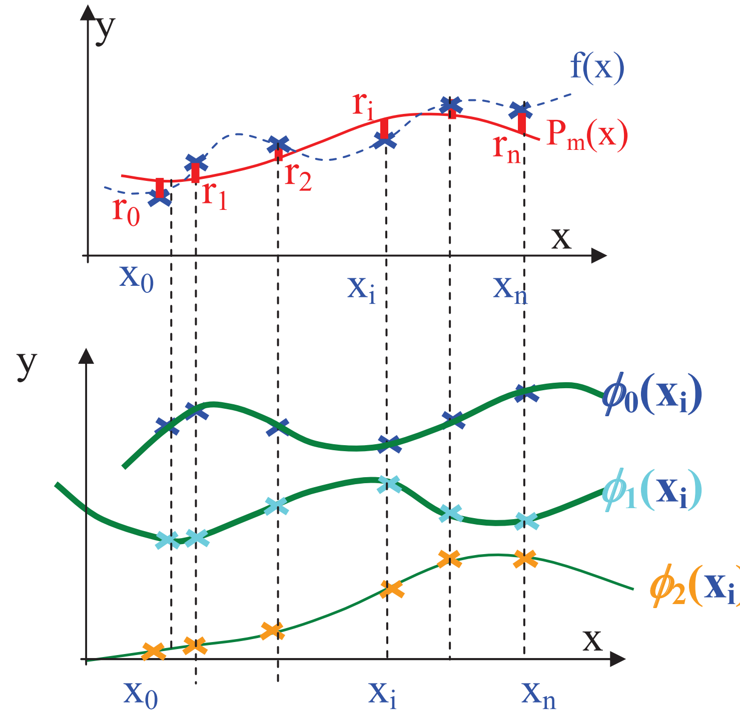
Esta expresión establece que:

- **El error de interpolación en las abscisas datos es cero**
- **La interpolación es exacta si f(x) es un polinomio hasta de grado n.**

Un tratamiento detallado se puede ver en: *Análisis Numérico*, R. Burden y D. Faires (1998). Capitulo 3, Teorema 3.3.; *Análisis Numérico*, W. Smith (1988). Capitulo 7, Teorema 2.

MÉTODO DE MÍNIMOS CUADRADOS

Dada una **función discreta**, $y=f(x)$ de $R \to R$ definida mediante **(n+1) puntos (x_i; $y_i=f(x_i)$)** con i=0,n.



Se propone $\quad \mathbf{P_m(x)} = \Sigma\, \boldsymbol{a_k}\, \boldsymbol{\phi_k(x)}$ con k=0,m.

Donde $\boldsymbol{a_k}$ son las incógnitas; y $\{\boldsymbol{\phi_k(x)}\}$ es una **Base elegida**

Se define: **Residuo** $\quad r_i = f(x_i) - P_m(x_i)$ con k=0,m; i=0,n

$$r_i = y_i - \Sigma\, a_k\, \phi_k(x_i)$$

$$\begin{Bmatrix} r_0 \\ r_1 \\ r_2 \\ \cdots \\ r_n \end{Bmatrix} = \begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ \cdots \\ y_n \end{Bmatrix} - \begin{pmatrix} \phi_0(x_0) & \phi_1(x_0) & \phi_2(x_0) & \cdots & \phi_m(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_m(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_m(x_2) \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_0(x_n) & \phi_1(x_n) & \phi_2(x_n) & \cdots & \phi_m(x_n) \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \cdots \\ a_m \end{Bmatrix}$$

$$\underline{r} = \underline{y} - \Phi \cdot \underline{a}$$

Los coeficientes $\boldsymbol{a_k}$ son tales que minimizan la Suma de los Cuadrados de los Residuos

$$\min \| \mathbf{r} \|_2^2 = \min \left[\Sigma (r_i(a_k))^2 \right],$$

La condición para que la Suma de los Cuadrados de los Residuos tome un valor extremo es $\sum_{i=1,n} 2\, r_i(a_j) \cdot \phi_j(x_i) = \underline{r}^T \cdot \underline{\phi_j} = 0 \qquad j=1,m$

Como se debe cumplir para todo j=1,m; finalmente resultan las siguientes ECUACIONES NORMALES

$$\boxed{\Phi^T \Phi \underline{a} = \Phi^T \underline{y}},$$

Sistema de ecuaciones lineales cuya solución da los coeficientes $\boldsymbol{a_k}$. Así la APROXIMACIÓN resulta $\mathbf{P_m(x)} = \Sigma\, \boldsymbol{a_k}\, \boldsymbol{\phi_k(x)}$ con k=0,m

MÉTODO DE MÍNIMOS CUADRADOS *EJEMPLO*

Dada una **función discreta**, y=f(x) de R $\rightarrow$ R
definida mediante **(2) puntos (x_i; y_i=f(x_i))** con i=0,1.

x	y=f(x)
1,5	3
3	4

Con la Base = $\{1, x\}$ el vector residuo $\underline{r} = \underline{y} - \Phi \cdot \underline{a}$ resulta

$$\begin{Bmatrix} r_1 \\ r_2 \\ r_3 \end{Bmatrix} = \begin{Bmatrix} \; \\ \; \\ \; \end{Bmatrix} - \begin{pmatrix} \; & \; \\ \; & \; \\ \; & \; \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix}$$

El sistema de ecuaciones normales $\Phi^T \Phi \underline{a} = \Phi^T \underline{y}$ resulta

$$\begin{pmatrix} \; & \; \\ \; & \; \end{pmatrix} \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} \; \\ \; \end{Bmatrix}$$ de donde se tiene que $a_0 =$ $a_1 =$

El **Polinomio de Aproximación** usando la **Base de Polinomios elementales** $\{$**1, x**$\}$
es $Pa(x) = \quad 1 + \quad x$

INTEGRACIÓN NUMÉRICA

Planteo de Problema

Integración de Newton Cotes

Método de Trapecios Simple y Múltiple

Regla de Integración y Error

Método de Simpson Simple y Múltiple

Regla de Integración y Error

Extrapolación de Richardson e Integración de Romberg

Integración de Gauss Legendre

Ejemplos

INTEGRACIÓN NUMÉRICA

El propósito es abordar el **cálculo de integrales definidas** de funciones.

Se asume que:

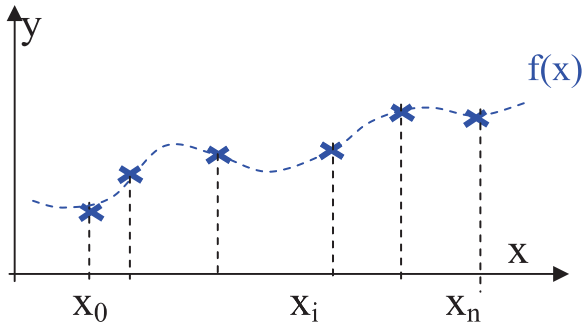
la función **$y = f(x)$:** R → R, **no singular, continua (al menos por tramos),** es conocida en forma

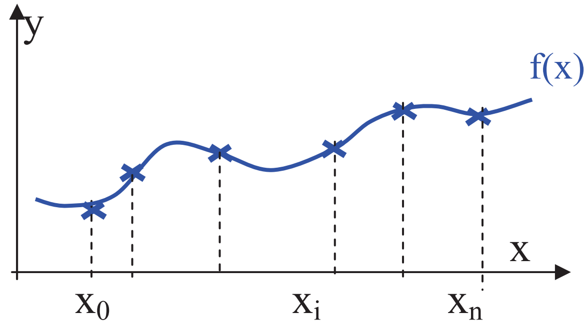
Discreta

f(x) se conoce en x_i, con i = 0,1,2,...,n

Analítica

f(x) se conoce en todo $x \in [x_0;x_n]$.





Se busca mostrar que:

es posible calcular la integral definida de la *y=f(x)* como una combinación lineal:

$$I = \int_a^b f(x)dx = \sum_{k=0,N} w_k \cdot f(x_k) = \sum_{k=0,N} w_k \cdot y_k \ ,$$

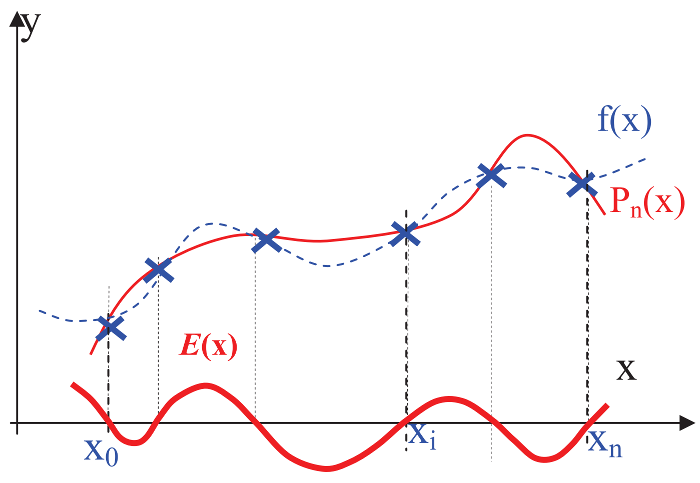
donde los coeficientes w_k son valores particulares para cada regla de integración y los $y_k = f(x_k)$ son los valores de la función discreta.

INTEGRACIÓN NUMÉRICA

Si f(x) está dada en forma ***discreta*** es posible ***interpolar*** f(x) colocando un **polinomio $P_n(x)$,** de grado n, por los (n+1) puntos datos. Si f(x) está dada en forma ***analítica*** se pueden *"extraer"* esos (n+1) puntos, y tener la versión discreta de f(x).

Es posible expresar a f(x) como suma del Polinomio de Interpolación mas la función Error de Interpolación

$$f(x) = P_n(x) + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)(x-x_1)........(x-x_n)$$



Resulta posible obtener la integral en la forma

$$I = \int_{X_0}^{X_n} f(x)dx = \int_{X_0}^{X_n} \left(P_n(x) + \varepsilon_n(x)\right)dx = \int_{X_0}^{X_n} P_n(x)dx + \int_{X_0}^{X_n} \varepsilon_n(x)dx$$

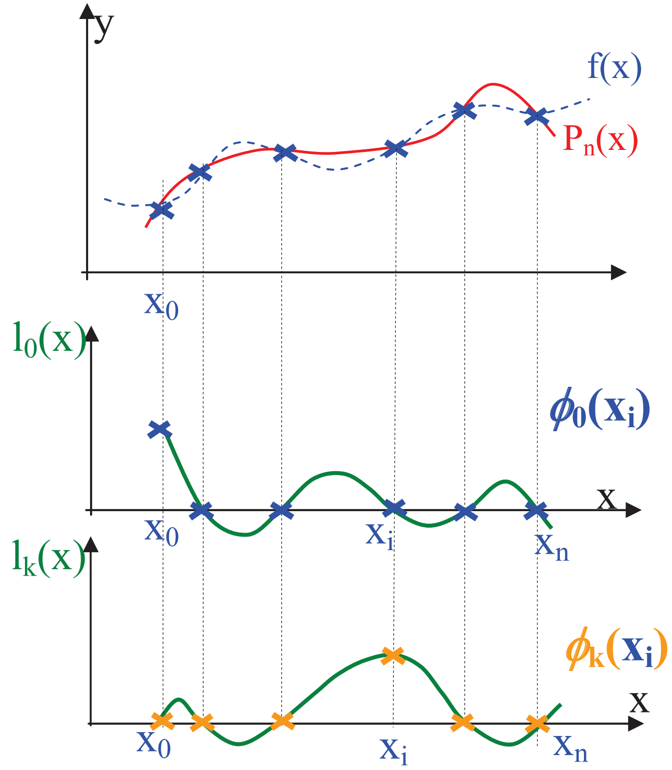
$$I = \int_{X_0}^{X_n} f(x)dx = \sum_{k=0,N} w_k \cdot y_k + \mathcal{E}_n = I_n + \mathcal{E}_n,$$

Resultando, la aproximación de la integral *In* y su Error de Integración *En* en la forma:

$$I_n = \int_{X_0}^{X_n} P_n(x)dx = \sum_{k=0,N} w_k \cdot y_k \qquad \mathcal{E}_n = \int_{X_0}^{X_n} \varepsilon_n(x)dx = \int_{X_0}^{X_n} \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)\cdots(x-x_n)dx$$

INTEGRACIÓN NUMÉRICA DE NEWTON COTES

Las reglas de integración de Newton Cotes se basan en interpolar con **Polinomios de LAGRANGE**. Para los n+1 puntos datos, el polinomio interpolante es



$$P_n(x) = \sum_{i=0}^{n} y_i \cdot l_i(x)$$

Así el valor aproximado de la integral

$$I_n = \int_{X_0}^{X_n} \sum_{i=0}^{n} y_i \cdot l_i(x) dx = \sum_{k=0,N} \int_{X_0}^{X_n} y_k \cdot l_k(x) \cdot dx$$

$$I_n = \sum_{k=0,N} y_k \cdot \int_{X_0}^{X_n} l_k(x) \cdot dx \cdot = \sum_{k=0,N} y_k \cdot w_k$$

resulta

$$w_k = \int_{X_0}^{X_n} l_k(x) \cdot dx = \int_{X_0}^{X_n} \prod_{\substack{j=0 \\ j \neq k}}^{n} \frac{(x - x_j)}{(x_k - x_j)} \cdot dx$$

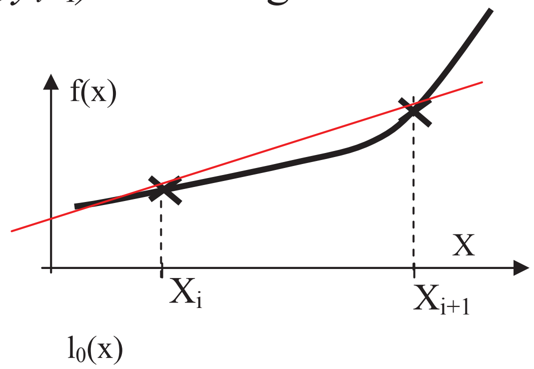
Así el Error de la aproximación de la integral

$$\mathcal{E}_n = \int_{X_0}^{X_n} \varepsilon_n(x) dx = \frac{f^{(n+1)}(\xi)}{(n+1)!} \int_{X_0}^{X_n} (x - x_0) \cdots (x - x_n) dx = \frac{f^{(n+1)}(\xi)}{(n+1)!} \cdot h^{n+2} \cdot \alpha_{n+1}$$

REGLA DE INTEGRACIÓN DE LOS TRAPECIOS

Es una regla de integración de Newton – Cotes por 2 puntos $(x_i;y_i)$, $(x_{i+1};y_{i+1})$. La integral

$$I = \int_{x_i}^{x_{i+1}} f(x) \cdot dx$$



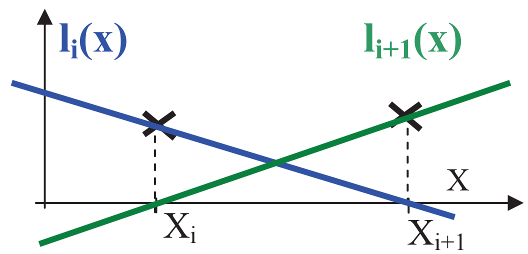
Se resuelve con un polinomio interpolante degrado uno

$$I = \int_{x_i}^{x_{i+1}} \left[P_1(x) + \varepsilon_1(x) \right] dx = \int_{x_i}^{x_{i+1}} \left[y_i \cdot l_i(x) + y_{i+1} \cdot l_{i+1}(x) \right] dx + \int_{x_i}^{x_{i+1}} \varepsilon_1(x) dx,$$

donde

$$l_i(x) = \frac{x - x_{i+1}}{x_i - x_{i+1}} = -\frac{x - x_{i+1}}{x_{i+1} - x_i}$$ es una recta que vale 1 en x_i y 0 en x_{i+1},

$$l_{i+1}(x) = \frac{x - x_i}{x_{i+1} - x_i}$$ es una recta que vale 0 en x_i y 1 en x_{i+1}.



$$I = \int_{x_i}^{x_{i+1}} \left[y_i \cdot \frac{x - x_{i+1}}{x_i - x_{i+1}} + y_{i+1} \cdot \frac{x - x_i}{x_{i+1} - x_i} \right] dx + \int_{x_i}^{x_{i+1}} \varepsilon_1(x) dx,$$

$$I = y_i \cdot \int_{x_i}^{x_{i+1}} \frac{x - x_{i+1}}{x_i - x_{i+1}} dx + y_{i+1} \cdot \int_{x_i}^{x_{i+1}} \frac{x - x_i}{x_{i+1} - x_i} dx + \int_{x_i}^{x_{i+1}} \varepsilon_1(x) dx = y_i \cdot w_i + y_{i+1} \cdot w_{i+1} + E_1$$

y, llamando **paso h_i** a la diferencia x_{i+1}-x_i y operando, se puede llegar a

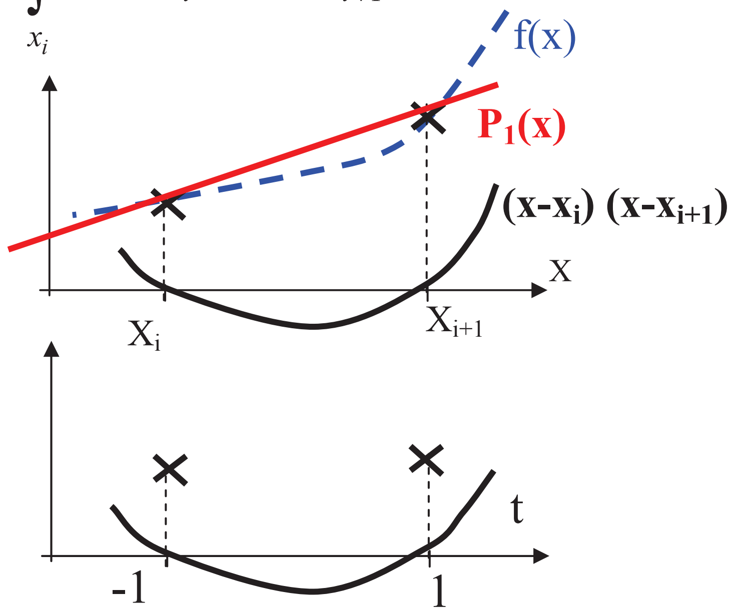
$$I = h_i \left[\frac{y_i}{2} + \frac{y_{i+1}}{2} \right] + \mathcal{E}_1 = I_1 + \mathcal{E}_1,$$

ERROR DE REGLA DE INTEGRACIÓN DE LOS TRAPECIOS

El error de interpolación es

$$\mathcal{E}_1 = \int_{x_a}^{x_b} \varepsilon_1(x)dx = \int_{x_i}^{x_{i+1}} \frac{f^{(2)}(\xi)}{2!}(x-x_i)(x-x_{i+1})dx = \frac{f^{(2)}(\xi)}{2!}\int_{x_i}^{x_{i+1}}(x-x_i)(x-x_{i+1})dx$$

siendo $\xi \in (x_i;x_{i+1})$.



Se propone hacer un cambio de variable mediante

$$\frac{x-x_i}{x_{i+1}-x_i} = \frac{t-(-1)}{1-(-1)},$$

Definiendo h=x_b-x_a, es posible despejar,

$$x - x_i = h(t+1)/2$$

$$x - x_{i+1} = h(t-1)/2$$

$$dx = (h/2)dt$$

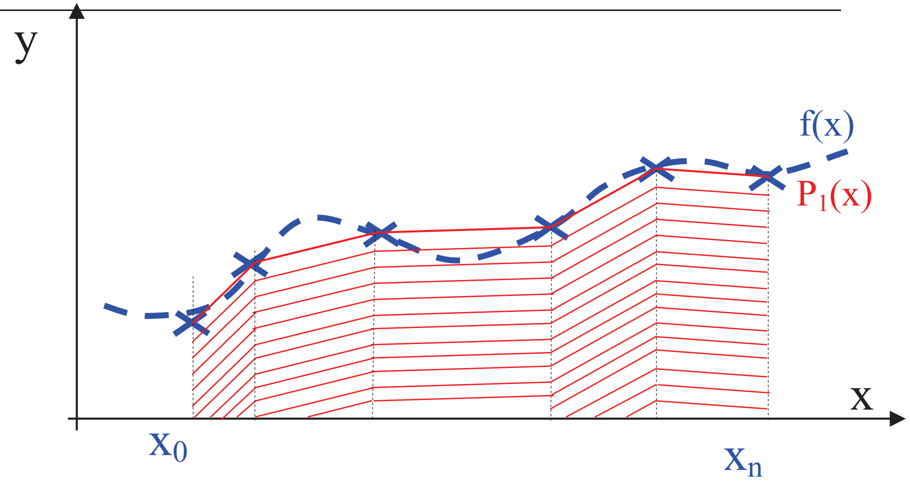
Así se puede hallar

$$\int_{x_i}^{x_{i+1}}(x-x_i)(x-x_{i+1})dx = \int_{-1}^{1}\frac{h(t+1)}{2}\cdot\frac{h(t-1)}{2}\cdot\frac{h}{2}dt = \frac{h^3}{8}\int_{-1}^{1}(t^2-1)dt = -h^3\frac{1}{6}$$

de manera que, sustituyéndola en (5), se obtiene

$$\mathcal{E}_1 = \frac{f^{(2)}(\xi)}{2!}\left(-\frac{1}{6}\right)h^3 = -\frac{1}{12}h^3 f^{(2)}(\xi)$$ para cierto punto $\xi \in (x_a, x_b)$.

REGLA DE INTEGRACIÓN DE LOS TRAPECIOS MÚLTIPLES



Se busca I ∈ R, $$I = \int_{X_0}^{X_n} f(x)dx\,.$$

Se divide el intervalo $[x_0; x_n]$
en subintervalos $[x_0; x_1]$, $[x_1; x_2]$, …, $[x_{n-1}; x_n]$. Así

$$I = \int_{X_0}^{X_1} f(x)\,dx + \int_{X_1}^{X_2} f(x)dx + \cdots + \int_{X_{n-1}}^{X_n} f(x)\,dx\,.$$

En cada uno de los n subintervalos se aplica la regla de los trapecios, se aplica trapecios simple

$$I = h_0\frac{(y_0 + y_1)}{2} + \mathcal{E}_1(h_0^{\,3}) + h_1\frac{(y_1 + y_2)}{2} + \mathcal{E}_1(h_1^{\,3}) + \ldots\ldots\ldots\ldots + h_{n-1}\frac{(y_{n-1} + y_n)}{2} + \mathcal{E}_1(h_{n-1}^{\,3})\,.$$

Si todos los intervalos tienen igual longitud h_i=h, esa fórmula se simplifica y se tiene la **regla de trapecios múltiple**:

$$I = h\left[\frac{y_0}{2} + \sum_{i=1}^{n-1} y_i + \frac{y_n}{2}\right] + \mathcal{E}_{1M} = \frac{h}{2}\left[y_0 + \sum_{i=1}^{n-1} 2\cdot y_i + y_n\right] + \mathcal{E}_{1M}\,,$$

donde $\mathcal{E}_{1M}$ es el error total que se acumula al sumar los n errores provenientes de la aplicación de la regla en cada subintervalo, y está dado por

$$\mathcal{E}_{1M} = -\frac{(x_n - x_0)}{12} h^2 f^{(2)}(\xi)$$

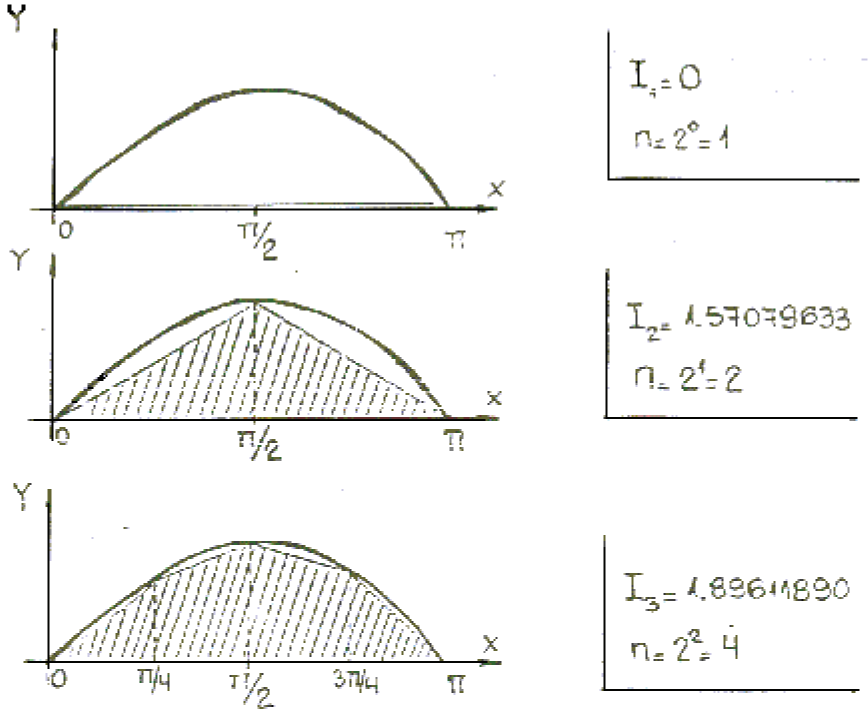
EJEMPLO DE LA REGLA DE INTEGRACIÓN DE LOS TRAPECIOS MÚLTIPLES

Se busca resolver $I = \int_0^{\pi} \text{sen}(x)dx$ cuyo resultado exacto es $I = -\cos x\big|_0^{\pi} = \cos 0 - \cos \pi = 2$

$$I = \int_0^{\pi} \text{sen}(x)dx \approx \frac{h}{2} \cdot \left[y_0 + 2 \cdot \sum_{i=1}^{n-1} y_i + y_n \right] = \frac{h}{2} \cdot S ,$$

donde $y_i = \text{sen}(x_i)$, $i=0,\ldots,n$.

X	Y=sen(x)	$h_1 = \pi$	$h_2 = \pi/2$	$h_3 = \pi/4$
0	0	1	1	1
$\pi/4$	$\sqrt{2}/2$	0	0	2
$\pi/2$	1	0	2	2
$3\pi/4$	$\sqrt{2}/2$	0	0	2
π	0	1	1	1
	0	S_1	S_2	S_3

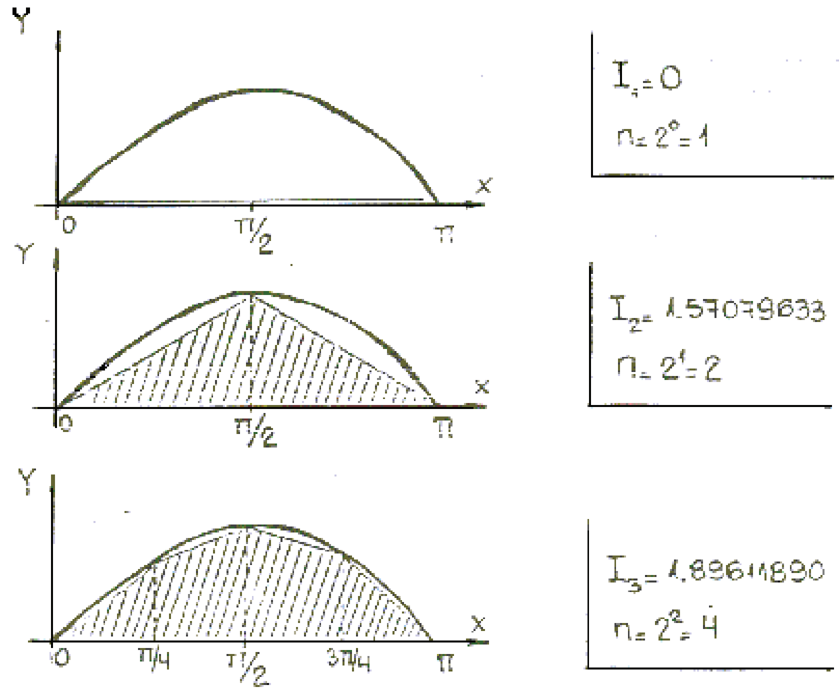


Es fácil concluir que la mejor aproximación es I_3.

Los errores cometidos por cada aplicación de la regla son, respectivamente $O(h_1^3)$, $O(h_2^2)$ y $O(h_3^2)$.

Extrapolación de Richardson para el ejemplo $I = \int_0^{\pi} \text{sen}(x)dx$

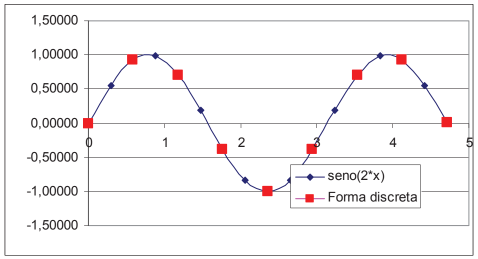
A partir de dos valores aproximados de la Integral, obtenidos con la misma Regla y pasos distintos, es posible encontrar un valor mejorado mediante Extrapolación de Richardson.

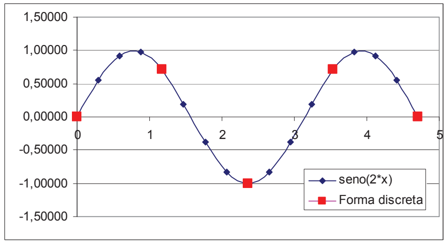


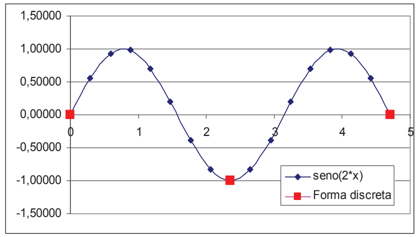
$$I = \frac{\beta\, I(h_2) - I(h_1)}{\beta - 1} \qquad \text{con} \quad \beta = \left(\frac{h_1}{h_2}\right)^n$$

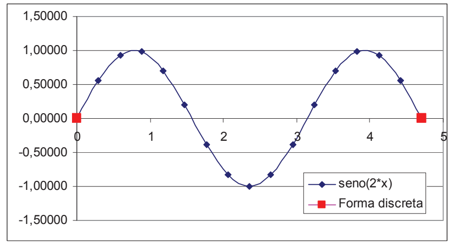
Ejemplo 2 Regla de Integración de los Trapecios Múltiples $Int = \int_0^{3\pi/2} \sin(2x)\,dx = \frac{-1}{2}\cos(2x)\Big|_0^{3\pi/2} = 1$

n		**16**	**n**	**8**	**n**	**4**	**n**	**2**	**n**	**1**
h		0,294524311	h	0,58905	h	1,1781	h	2,35619	h	4,7124
w		w* Sin(2x)	w	w* Sin(2x)	w	w* Sin(2x)	w	w* Sin(2x)	w	w* Sin(2x)
	1	0	1	0	1	0	1	0	1	0
	2	1,111140466								
	2	1,847759065	2	1,84776						
	2	1,961570561								
	2	1,414213562	2	1,41421	2	1,4142				
	2	0,390180644								
	2	-0,765366865	2	-0,7654						
	2	-1,662939225								
	2	-2	2	-2	2	-2	2	-2		
	2	-1,662939225								
	2	-0,765366865	2	-0,7654						
	2	0,390180644								
	2	1,414213562	2	1,41421	2	1,4142				
	2	1,961570561								
	2	1,847759065	2	1,84776						
	2	1,111140466								
	1	2,1439E-15	1	2,1E-15	1	2E-15	1	2,1E-15	1	2E-15
Integral		**0,970916536**		**0,88157**		**0,488**		**-2,3562**		**5E-15**
h		0,294524311		0,58905		1,1781		2,35619		4,7124







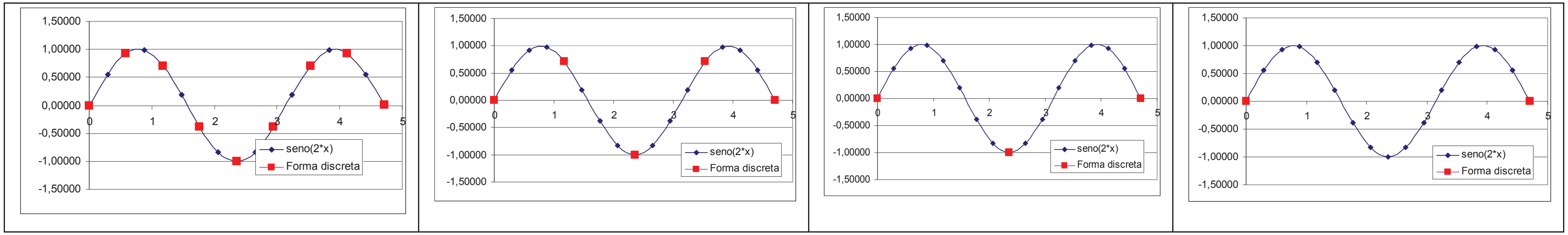


Extrapolación de Richardson para el ejemplo $Int = \int_0^{3\pi/2} \sin(2x)\,dx = \frac{-1}{2}\cos(2x)\Big|_0^{3\pi/2} = 1$

$$I = \frac{\beta\, I(h_2) - I(h_1)}{\beta - 1} \qquad \text{con} \quad \beta = \left(\frac{h_1}{h_2}\right)^n$$

Y la extrapolación de Richardson sucesiva (Método de Romberg) resulta:

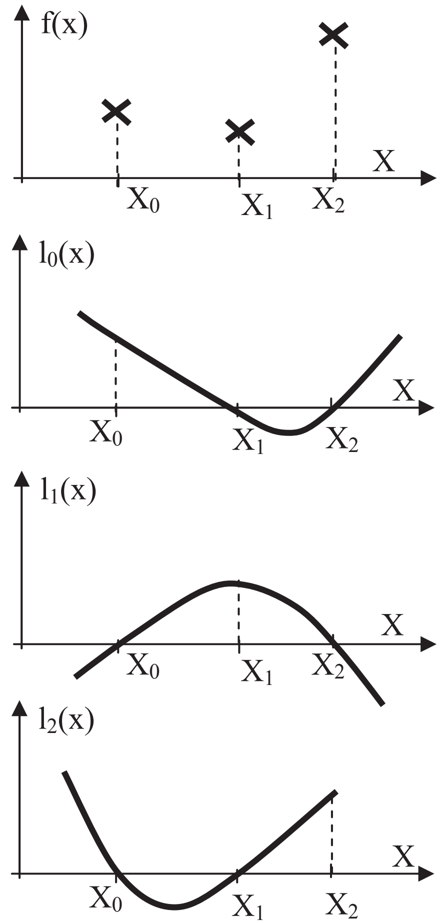
n	Orden	h^2	h^4	h^6	h^8	
Intervalos	**Paso h**	**I(h^2)**	**I(h^4)**	**I(h^6)**	**I(h^8)**	**I(h^10)**
16	0,29452	0,97092				
8	0,58905	0,88157	1,00069753			
4	1,1781	0,48798	1,01277013	0,999892687		
2	2,35619	-2,3562	1,43604331	0,98455192	1,000136191	
1	4,71239	5,1E-15	-3,14159265	1,741219036	0,972541331	1,00024441



REGLA DE INTEGRACIÓN DE SIMPSON

Es una cuadratura de Newton – Cotes con n = 2, es decir con tres puntos. Se interpola mediante un polinomio de Lagrange de grado dos y luego se integra en forma aproximada ese polinomio.

$$I = \int_{X_0}^{X_2} f(x)dx \; ,$$



Si $f(x) = \sum Y_i \, l_i(x) + E_2(x)$, con i = 0,1,2,

$$l_0(x) = \frac{x - x_1}{x_0 - x_1}\frac{x - x_2}{x_0 - x_2} \quad l_1(x) = \frac{x - x_0}{x_1 - x_0}\frac{x - x_2}{x_1 - x_2} \quad l_2(x) = \frac{x - x_0}{x_2 - x_0}\frac{x - x_1}{x_2 - x_1}$$

$$\varepsilon_2(x) = \frac{f^{(3)}}{3!}(x - x_0)(x - x_1)(x - x_2)$$

$$I_2 = \sum_{i=0}^{2} y_i \omega_i \quad \omega_i = \int_{x_0}^{x_2} l_i(x)dx, \, i = 0,1,2. \quad \mathcal{E}_2 = \int_{x_0}^{x_2} \varepsilon_2(x)dx$$

Entonces, si los intervalos son iguales ($h_1 = h_2 = h$), se tiene:

$$I = h\left[\frac{1}{3}y_0 + \frac{4}{3}y_1 + \frac{1}{3}y_2\right] - \frac{h^5}{90} f^{(4)}(\xi)$$

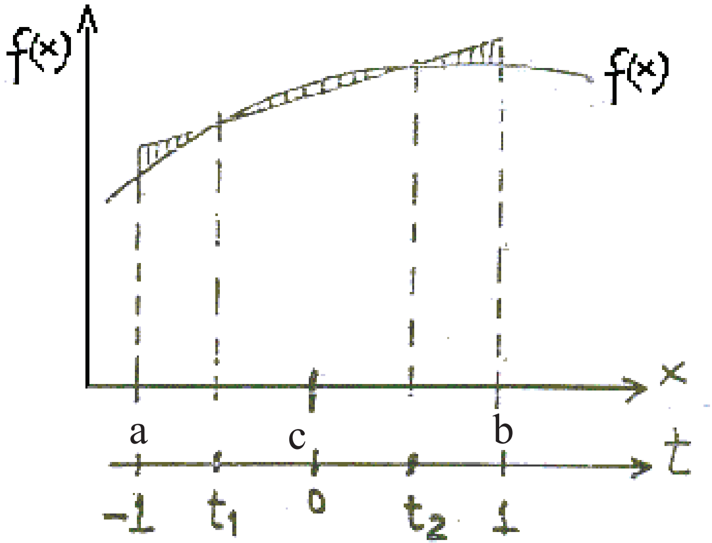
INTEGRACIÓN DE GAUSS LEGENDRE

Se busca resolver la

$$I=\int_a^b f(x)dx = h\int_{-1}^{1} G(t)dt = h\left(\sum_{i=0}^{n}\omega_i G(t_i)+R\right)=h\sum_{i=0}^{n}\omega_i G(t_i)+\mathcal{E}$$

$$G(t)= f(c+(\frac{b-a}{2})\cdot t) \qquad x=c+(\frac{b-a}{2})\cdot t$$

El problema se resuelve en el dominio unitario [-1, 1], mediante Mapeo o Cambio de variables conveniente.



$$\int_{-1}^{1} G(t)dt = \omega_1 G(t_1) + \omega_2 G(t_2) + R$$

ω_1, ω_2, t_1, t_2 deben ser tales que el error de integración sea R = 0 para polinomios de hasta 3º grado. Comparando los resultados exactos de la integral con el resultado propuesto por la regla, se tiene:

$$\left.\begin{aligned}
&\int_{-1}^{1} 1dt = 2 = 1\omega_1 + 1\omega_2\\
&\int_{-1}^{1} tdt = 0 = t_1\omega_1 + t_2\omega_2\\
&\int_{-1}^{1} t^2dt = \frac{2}{3} = t_1^2\omega_1 + t_2^2\omega_2\\
&\int_{-1}^{1} t^3dt = 0 = t_1^3\omega_1 + t_2^3\omega_2
\end{aligned}\right\}
\begin{aligned}
&\omega_1=\omega_2=1\\
&-t_1=t_2=\frac{\sqrt{3}}{3}
\end{aligned}$$

$$I = h\left[G\left(\frac{-\sqrt{3}}{3}\right)+G\left(\frac{\sqrt{3}}{3}\right)\right]+\mathcal{E}$$

Integración de Gauss Legendre. Generalización.

Se busca resolver la

$$\int_{-1}^{1} G(t)dt = \sum_{i=0}^{n} \omega_i G(t_i) + R$$

En una regla de n+1 puntos existen 2n+2 coeficientes a determinar, que son los (n+1) w_k y las (n+1) abscisas t_k. Se exige que la regla de integración sea exacta cuando integra polinomios de hasta grado 2n+1. Dada una función polinómica $G(t)$ de hasta grado (2n+1), existen polinomios $C_n(t)$ y $r_n(t)$ de grado menor o igual a n, tales que

$$G(t)= C_n(t)\, p_{n+1}(t) + r_n(t)$$

Siendo $p_{n+1}(t)$ un polinomio de grado (n+1), que en particular puede ser un polinomio de Legendre. Así es posible expresar la integral

$$I = \int_{-1}^{1} G(t)\,dt = \int_{-1}^{1} (C_n(t)\, p_{n+1}(t) + r_n(t))\,dt = \int_{-1}^{1} C_n(t)\, p_{n+1}(t)dt + \int_{-1}^{1} r_n(t)\,dt = \int_{-1}^{1} r_n(t)\,dt = \sum_{k=0,n} w_k\, r_n(t_k)$$

Ya que **cualquier polinomio de grado n es ortogonal al polinomio de Legendre de grado n+1** en el dominio [-1,1]. Para elegir las abscisas t_k, es posible elegir las (n+1) **raíces del polinomio de Legendre** $\boldsymbol{p_{n+1}(t)}$ de grado (n+1), que se las denomina t_r, y en las que se verifica que

$$G(t_r)= C_n(t_r)\, p_{n+1}(t_r) + r_n(t_r)= r_n(t_r)$$

Así la integral resulta

$$I = \int_{-1}^{1} G(t)\,dt = \int_{-1}^{1} r_n(t)\,dt = \sum_{k=0,n} w_k\, r_n(t_k) = \sum_{k=0,n} w_k\, G(t_k)$$

Para calcular los w_k,

Es posible plantear una **interpolación del polinomio de grado n** $r_n(t)$, con polinomios de Lagrange, tomando como abscisas conocidos las (n+1) raíces t_k del polinomio de Legendre de grado (n+1)

$$r_n(t)= r_n(t_0)\, l_0(t) + r_n(t_1)\, l_1(t) + r_n(t_2)\, l_2(t) +\dots\; r_n(t_n)\, l_n(t)$$

o bien considerando que $G(t_r)= r_n(t_r)$

$$r_n(t)= G(t_0)\, l_0(t) + G(t_1)\, l_1(t) + G(t_2)\, l_2(t) +\dots\; G(t_n)\, l_n(t)$$

$$I = \int_{-1}^{1} G(t)\,dt = \int_{-1}^{1} r_n(t)\,dt = \int_{-1}^{1}\left[\sum_{k=0,n} G(t_k)\, l_k(t)\right]dt = \sum_{k=0,n}\left[\int_{-1}^{1} G(t_k) l_k(t)\,dt\right]$$

$$I = \int_{-1}^{1} G(t)\,dt = \sum_{k=0,n}\left[G(t_k)\int_{-1}^{1} l_k(t)\,dt\right] = \sum_{k=0,n} w_k\, G(t_k) \quad \text{Con} \quad w_k = \int_{-1}^{1} l_k(t)\,dt$$

Es decir que los coeficientes w_k son las integrales de los polinomios de Lagrange de grado n+1, definidos por las (n+1) raíces t_k de los polinomios de Legendre de grado n+1. Así resulta,

$I = \int_{-1}^{1} G(t)\,dt = \sum_{k=0,n} w_k\, G(t_k)$	Siendo $w_k = \int_{-1}^{1} l_k(t)\,dt$	Y (t_r) raíces del Polinomio de Legendre $p_{n+1}(t_r)=0$

Se toma (n+1) puntos de Gauss (raices del polinomio de grado n+1),
y se puede integrar en forma exacta hasta polinomios de grado (2n+1)

(n+1)Puntos de Gauss	1	2	3	4
Grado exacto	1	3	5	7

DERIVACIÓN NUMÉRICA

Planteo de Problema

Derivadas a partir de Interpolación con Polinomios de Newton

- Derivada Primera
- Derivada Segunda

Derivadas a partir de Serie de Taylor

- Derivada Primera
- Derivada Segunda

Extrapolación de Richardson

Ejemplos

DERIVACIÓN NUMÉRICA

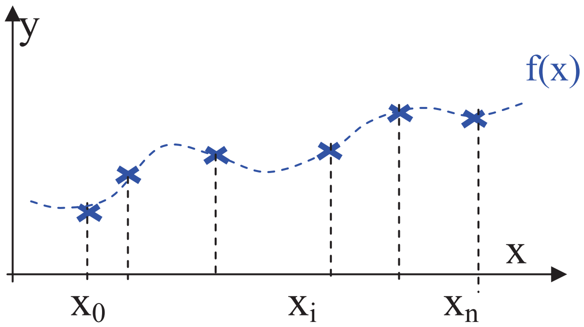
El propósito es abordar el **cálculo de derivadas de distinto orden** de funciones discretas.

Supuestos

La función **y = f(x):** R $\rightarrow$ R, **no singular, continua,** es conocida en forma

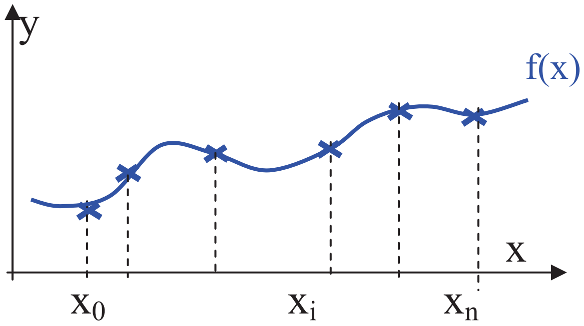
Discreta

f(x) se conoce en x_i, con i = 0,1,2,...,n



Analítica

f(x) se conoce en todo x $\in [x_0;x_n]$.



Hipótesis

Es posible calcular la derivada n-ésima de la *y=f(x)* evaluada en *Xj* como una suma:

$$D = \left.\frac{d^n(f(x))}{dx^n}\right|_{Xj} = \sum_{k=0,N} c_k \cdot f(x_k) = \sum_{k=0,N} c_k \cdot y_k = \vec{c}^{\,T} \cdot \vec{y} \,,$$

Donde los coeficientes c_k son valores particulares para cada regla de derivación y los $y_k=f(x_k)$ son los valores de la función discreta.

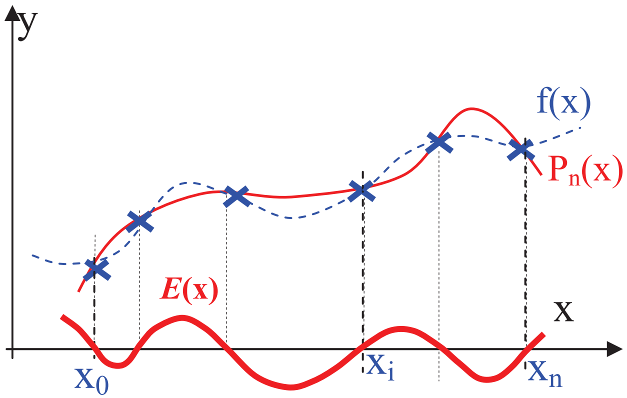
DERIVACIÓN NUMÉRICA

Si f(x) está dada en forma discreta es posible ***interpolar*** f(x) colocando un **polinomio $P_n(x)$,** de grado n, por los (n+1) puntos datos.

Si f(x) está dada en forma ***analítica*** se pueden *"extraer"* esos (n+1) puntos, para la versión discreta de la función f(x).

Es posible expresar:
$$f(x) = P_n(x) + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)(x-x_1)\ldots\ldots.(x-x_n)$$

suma del Polinomio de Interpolación mas la función Error de Interpolación.



Resulta posible

$$D = \left.\frac{d^n(f(x))}{dx^n}\right|_{Xj} = \sum_{k=0,N} c_k \cdot f(x_k) + \mathcal{E}_n = \sum_{k=0,N} c_k \cdot y_k + \mathcal{E}_n = \vec{c}^{\,T} \cdot \vec{y} + \mathcal{E}_n$$

DERIVACIÓN NUMÉRICA

Si f(x) está dada en forma discreta con <u>(n+1) puntos</u>, es posible interpolar f(x) colocando un <u>polinomio de grado n</u>. A partir de

$$f(x) = P_n(x) + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)(x-x_1)........(x-x_n)$$

Es posible obtener <u>derivadas hasta de grado n</u>, y evaluarlas en cualquier punto *Xj*, de la forma:

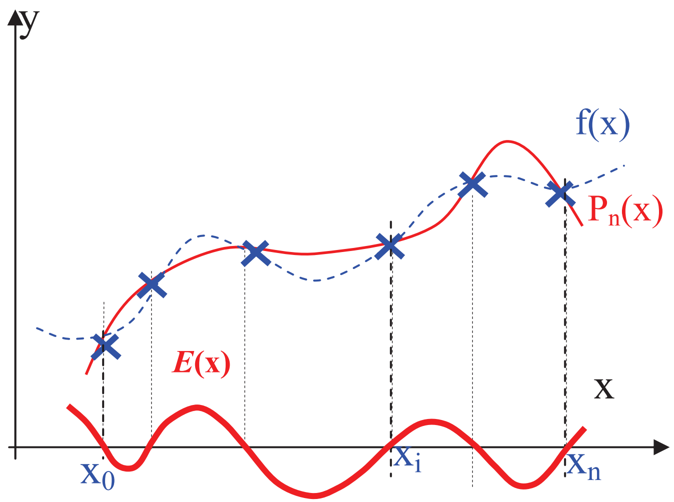
$$D = \left.\frac{d^n(f(x))}{dx^n}\right|_{Xj} = \left.\frac{d^n\left(P_n(x)+\varepsilon_n(x)\right)}{dx^n}\right|_{Xj}$$

$$D = \left.\frac{d^n\left(P_n(x)\right)}{dx^n}\right|_{Xj} + \left.\frac{d^n\left(\varepsilon_n(x)\right)}{dx^n}\right|_{Xj}$$

$$D = \quad D_n \quad + \quad \mathcal{E}_n$$

Resultando

$$D_n = \left.\frac{d^n\left(P_n(x)\right)}{dx^n}\right|_{Xj} = \sum_{k=0,N} c_k \cdot f(x_k) = \sum_{k=0,N} c_k \cdot y_k = \vec{c}^{\,T} \cdot \vec{y}$$



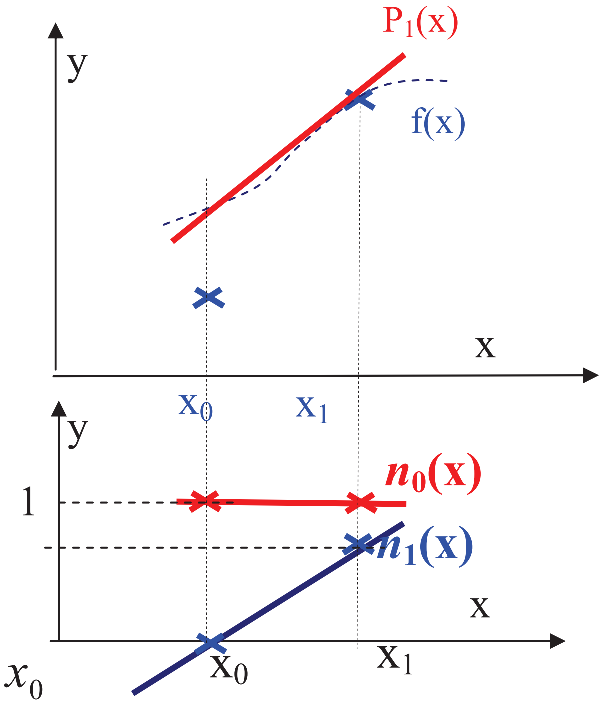
con el Error de Derivación dado por $\mathcal{E}_n = \left.\frac{d^n\left(\varepsilon_n(x)\right)}{dx^n}\right|_{Xj} = \frac{d^n\left(\frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)\cdots(x-x_n)\right)}{dx^n}$

Pregunta: ¿Qué cantidad de puntos se necesita para evaluar una derivada primera?, ¿y para una derivada segunda?

DERIVADA PRIMERA

Si f(x) está dada en forma discreta con 2 puntos $(x_0;y_0)$, $(x_1;y_1)$ es posible interpolar f(x) colocando un polinomio de grado 1, usando polinomios de Newton, en la forma:

$$f(x)\cdot = a_0 + a_1 \cdot (x - x_0) + \frac{f^{(2)}(\xi)}{(2)!}(x - x_0)\cdot(x - x_1)$$



La derivada primera de f(x) es

$$\frac{d(f(x))}{dx} = a_1 + \frac{f^{(2)}(\xi)}{(2)!}\cdot\frac{d((x - x_0)\cdot(x - x_1))}{dx}$$

$$\frac{d(f(x))}{dx} = a_1 + \frac{f^{(2)}(\xi)}{(2)!}\cdot[(x - x_1) + (x - x_0)]$$

Cuando se evalúa la derivada primera de f(x) en x_0 y en x_1 se obtienen:

Derivada Primera Adelante

$$\left.\frac{d(f(x))}{dx}\right|_{X=X0} = a_1 + \frac{f^{(2)}(\xi)}{(2)!}\cdot\Delta x \quad con \quad a_1 = \frac{y_1 - y_0}{\Delta x} \quad y \quad \Delta x = x_1 - x_0$$

Derivada Primera Atrás

$$\left.\frac{d(f(x))}{dx}\right|_{X=X1} = a_1 + \frac{f^{(2)}(\xi)}{(2)!}\cdot\Delta x \quad con \quad a_1 = \frac{y_1 - y_0}{\Delta x} \quad y \quad \Delta x = x_1 - x_0$$

Derivada Primera es constante en el intervalo; y el Error es lineal.

DERIVADA SEGUNDA

Si f(x) está dada en forma discreta con 3 puntos *$(x_0;y_0)$, $(x_1;y_1)$* es posible interpolar f(x) colocando un polinomio de grado 2, usando polinomios de Newton, en la forma:

$$f(x) \cdot = a_0 + a_1 \cdot (x - x_0) + a_2 \cdot (x - x_0) \cdot (x - x_1) + \frac{f^{(3)}(\xi)}{(3)!}(x - x_0) \cdot (x - x_1) \cdot (x - x_2)$$

La derivada segunda de f(x) es

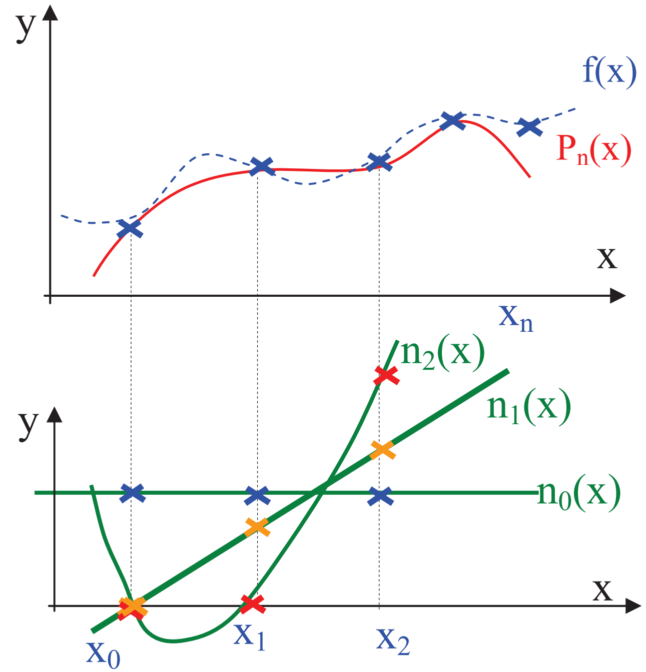
$$\frac{d^2(f(x))}{dx^2} = 2a_2 + \frac{f^{(3)}(\xi)}{(3)!} \cdot \frac{d^2((x - x_0) \cdot (x - x_1) \cdot (x - x_2))}{dx^2}$$

Con $2a_2 = 2\dfrac{\dfrac{y_2 - y_1}{x_2 - x_1} - \dfrac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_0} = 2 \cdot \dfrac{1}{2\Delta x} \cdot \dfrac{y_2 - y_1}{\Delta x} - \dfrac{y_1 - y_0}{\Delta x}$

Con lo que la derivadas segunda es

$$\left.\frac{d^2(f(x))}{dx^2}\right| = \frac{1}{\Delta x^2}(y_0 - 2y_1 + y_2) \quad + \quad E_D(x)$$

$$E_D(x) = \frac{f^{(3)}(\xi)}{(3)!} \cdot \frac{d^2((x - x_0) \cdot (x - x_1) \cdot (x - x_2))}{dx^2}$$



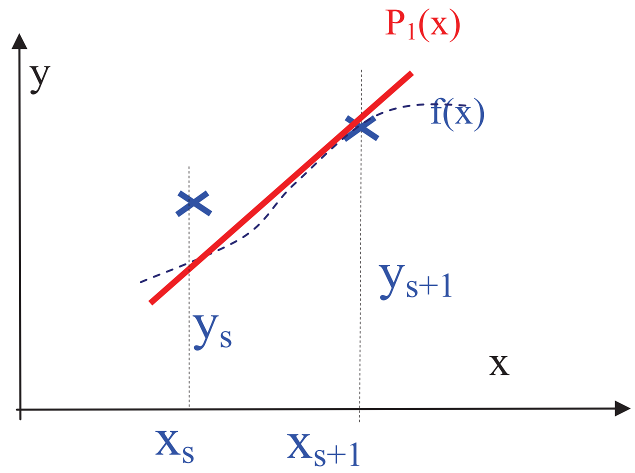
Derivada Primera es constante en el intervalo; y el Error es lineal.
¿El error será igual a cero en alguno de los puntos datos?

DERIVADA PRIMERA EN BASE A SERIE DE TAYLOR

Si f(x) está dada en forma discreta con 2 puntos $(x_s;y_s)$, $(x_{s+1};y_{s+1})$ es posible considerar el desarrollo de Serie de Taylor de $y_{s+1}=f(x_{s+1})$ alrededor de x_s en la forma:

$$f_{s+1} = f_s + h\, f_s' + \frac{h^2}{2!} f_s'' + \frac{h^3}{3!} f_s''' + \frac{h^4}{4!} f_s^{\prime v} + O\left(h^5\right),$$

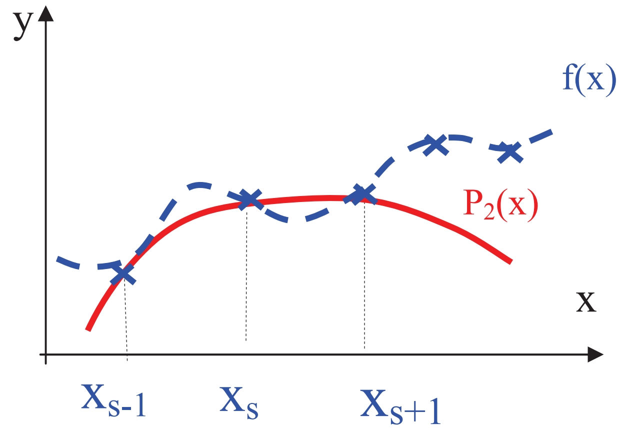
de donde se puede despejar la derivada primera en x_s



$$f\left(x_s + nh\right) = f\left(x_s\right) + n\,h\,f'\left(x_s\right) + \frac{(nh)^2 f''\left(x_s\right)}{2!} + \frac{(nh)^3 f'''\left(x_s\right)}{3!} + \frac{(nh)^4 f^{\prime v}\left(x_s\right)}{4!} + O\left(h^5\right)$$

DERIVADA SEGUNDA EN BASE A SERIE DE TAYLOR

Si f(x) está dada en forma discreta con 3 puntos $(x_{s-1};y_{s-1})$ $(x_s;y_s)$, $(x_{s+1};y_{s+1})$ es posible considerar el desarrollo de Serie de Taylor de $y_{s\pm1}=f(x_{s\pm1})$ alrededor de x_s en la forma:



$$f_{s+1} = f_s + h\, f_s' + \frac{h^2}{2!} f_s'' + \frac{h^3}{3!} f_s''' + \frac{h^4}{4!} f_s^{\prime v} \ldots\ldots,$$

$$f_{s-1} = f_s - h\, f_s' + \frac{h^2}{2!} f_s'' - \frac{h^3}{3!} f_s''' + \frac{h^4}{4!} f_s^{\prime v} \ldots\ldots$$

y con ello se puede plantear en x_s la derivada segunda de f(x) como

$$f_s'' = \left[\alpha \cdot f_{s-1} + \beta \cdot f_s + \gamma \cdot f_{s+1}\right]$$

$$f(x_s + nh) = f(x_s) + n\,h\,f'(x_s) + \frac{(nh)^2 f''(x_s)}{2!} + \frac{(nh)^3 f'''(x_s)}{3!} + \frac{(nh)^4 f'^{v}(x_s)}{4!} + O(h^5)$$

SOLUCIÓN NUMÉRCIA DE ECUACIONES DIFERENCIALES CON VALORES DE CONTORNO

Ecuaciones diferenciales: una clasificación

Ecuaciones Diferenciales Ordinarias Orden Superior
- Problema de Valores de Contorno
  - Obtención de Sistemas de Ecuaciones Lineales No Homogeneos
  - Obtención de Sistemas de Valores y vectores propios
- Problema de Valores Iniciales

Ecuaciones Diferenciales a Derivadas Parciales

Ejemplos
Resumen

SOLUCIÓN NUMÉRCIA DE ECUACIONES DIFERENCIALES ORDINARIAS

Se busca obtener una solución aproximada en forma discreta

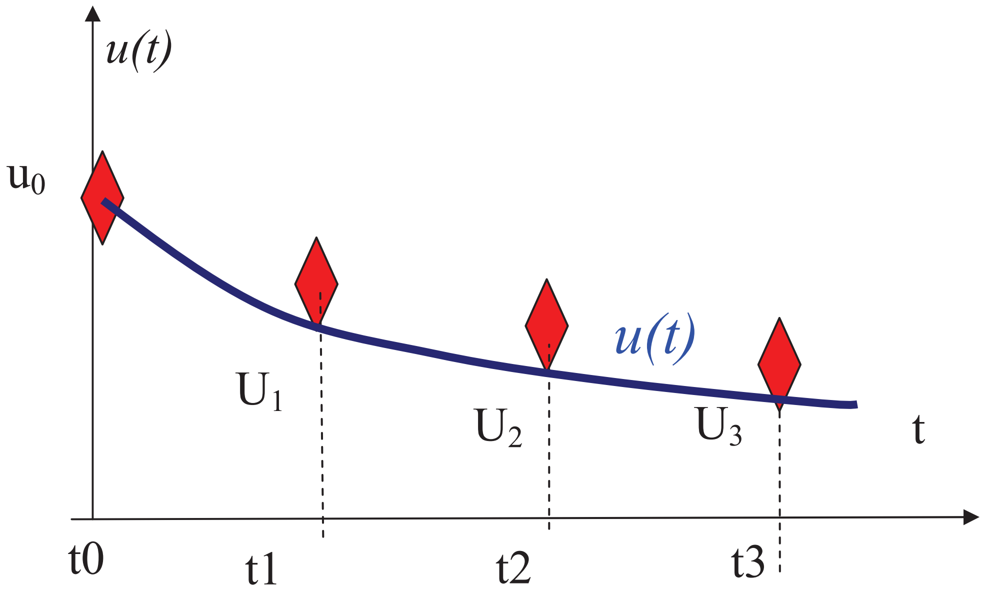
Clasificación de EDO

Primer orden

Siempre son de valor inicial

$$\frac{du(t)}{dt} + A \cdot u(t) = 0, \qquad A \in \mathrm{R},$$

$$u(t_0) = U_0,$$



La solución discreta es tal que aproxima a la solución exacta del problema

$$U_k \cong u(t_k)$$

SOLUCIÓN NUMÉRCIA DE ECUACIONES DIFERENCIALES ORDINARIAS

Se busca obtener una solución aproximada en forma discreta

Orden Superior

Pueden ser

de valores iniciales

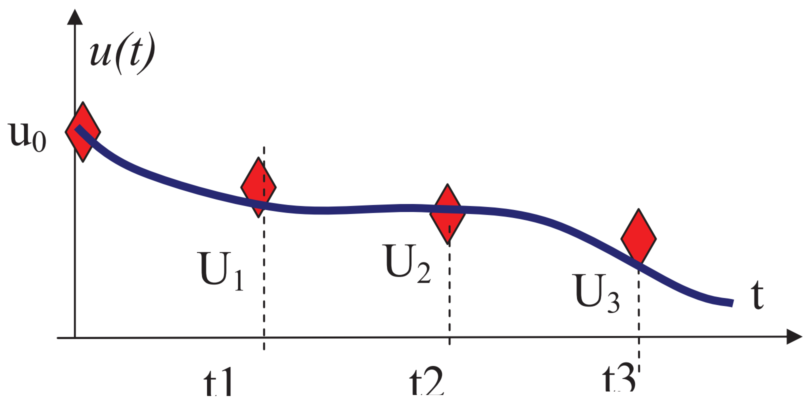
$$\frac{d^2u(t)}{dt^2}+\frac{g}{L}\cdot u(t)=0,$$

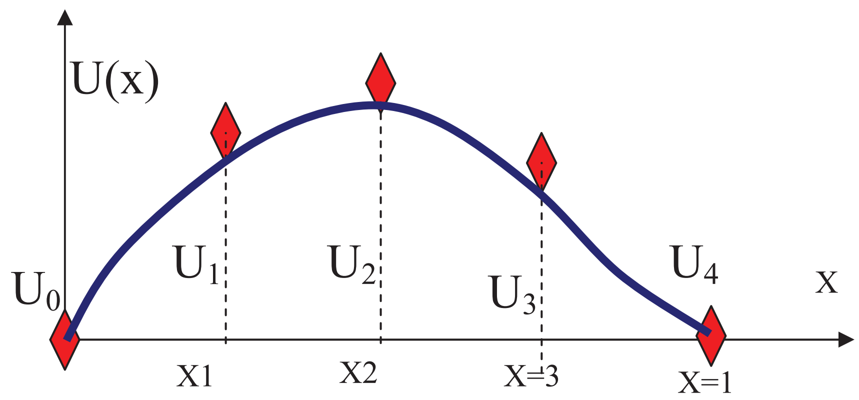
$$u(t_0)=u_0, \qquad \left.\frac{du(t)}{dt}\right|_{t0}=v_0$$

de valores de contorno

$$-\frac{d^2u(x)}{dx^2}+u(x)-x=0 \qquad en \quad \Omega=\{x\,\varepsilon\,R:0\le x\le 1\}$$

$$u(0)=0 \qquad u(1)=0$$





La solución discreta es tal que aproxima a la solución exacta del problema

$$U_k \cong u(t_k) \qquad\qquad U_k \cong u(x_k)$$

DISCRETIZACIÓN DE ECUACIONES DIFERENCIALES ORDINARIAS

Se busca *u(x)* solución de

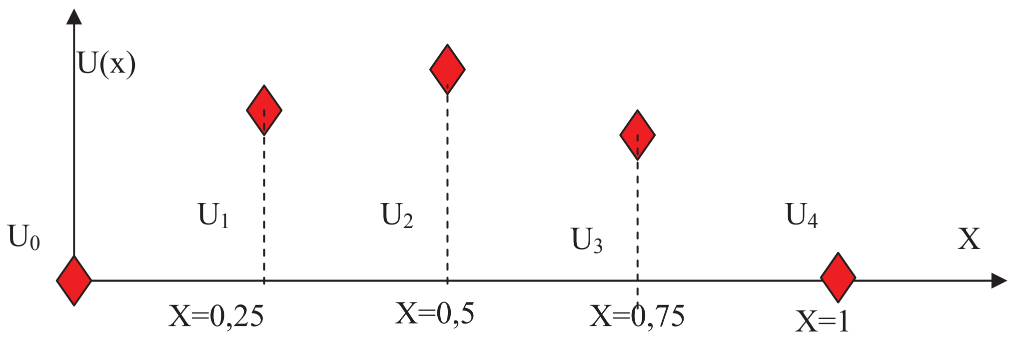
$$-\frac{d^2u(x)}{dx^2}+u(x)-x=0 \quad en \;\; \Omega=\left\{x\,\varepsilon\, R: 0\leq x\leq 1\right\}$$

$$u(0)=0$$

$$u(1)=0$$

se plantea encontrar una ***solución aproximada en forma de función discreta***, sólo en algunos puntos elegidos del dominio Ω, equidistantes entre sí, identificados con su abscisa X_k.

Es decir, se busca ***$U(X_k)=U_k$*** con k=0,N; ***función discreta*** que es una aproximación de la función continua *u(x)*.

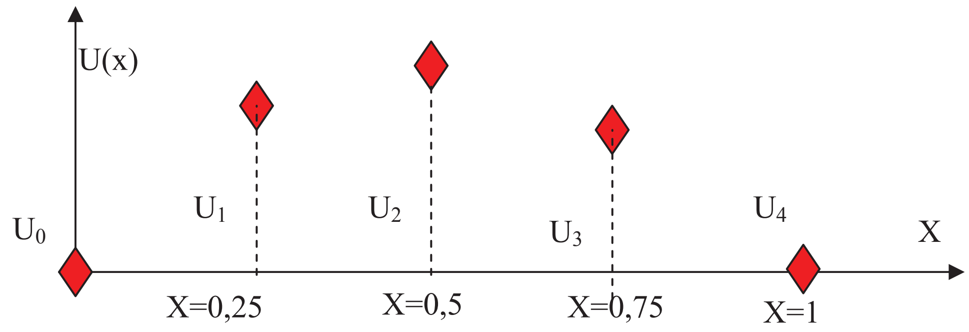


Se busca la ***función discreta $U(X_k)=U_k$*** con k=0,N;
que es una aproximación de la función continua *u(x)*.

$$-\frac{d^2u(x)}{dx^2}+u(x)-x=0 \quad en \ \ \Omega=\{x\,\varepsilon\,R:0\le x\le 1\}$$

$$u(0)=0$$

$$u(1)=0$$



Se divide el dominio Ω en N segmentos iguales

En cada uno de los X_k se puede plantear la ecuación diferencial a resolver pero con una aproximación de la derivada segunda en forma de derivada numérica

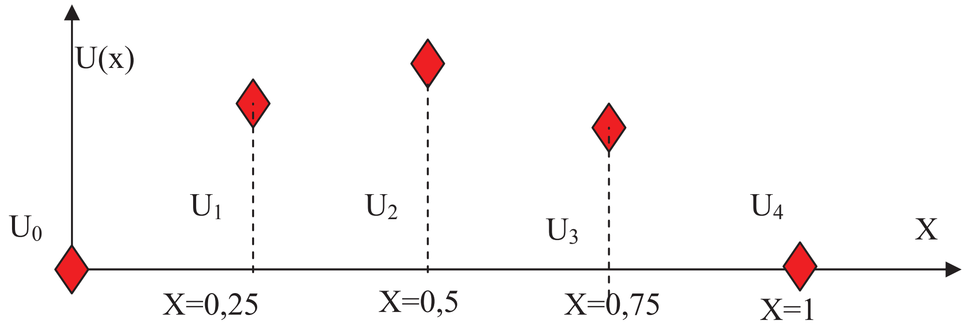
$$-\frac{1}{\Delta x^2}\left[U_{k-1}-2\cdot U_k+U_{k+1}\right]+U_k-X_k=0 \quad en \ \ X_k \qquad con \ \ k=1,(N-1)$$

De estas ecuaciones se pueden plantear tantas como puntos interiores; es decir N-1 ecuaciones y se tienen N+1 incógnitas.

Además se tienen las dos ecuaciones correspondientes a las Condiciones de Contorno, que agregan dos ecuaciones más.

Así se tienen N+1 ecuaciones con N+1 incógnitas.

$$-\frac{d^2u(x)}{dx^2}+u(x)-x=0 \quad en \;\; \Omega=\{x\,\varepsilon\, R: 0\le x\le 1\}$$
$$u(0)=0$$
$$u(1)=0$$



Si se consideran 5 puntos en todo el dominio, que son 3 en el interior y uno en cada uno de los bordes, es posible plantear

en X_0 se debe cumplir que: $$U_0=0$$

en X_1 se debe cumplir que: $$-\frac{1}{0{,}25^2}\left[U_0-2\cdot U_1+U_2\right]+U_1-X_1=0$$

en X_2 se debe cumplir que: $$-\frac{1}{0{,}25^2}\left[U_1-2\cdot U_2+U_3\right]+U_2-X_2=0$$

en X_3 se debe cumplir que: $$-\frac{1}{0{,}25^2}\left[U_2-2\cdot U_3+U_4\right]+U_3-X_3=0$$

en X_4 se debe cumplir que: $$U_4=0$$

O bien en forma de sistema de ecuaciones lineales

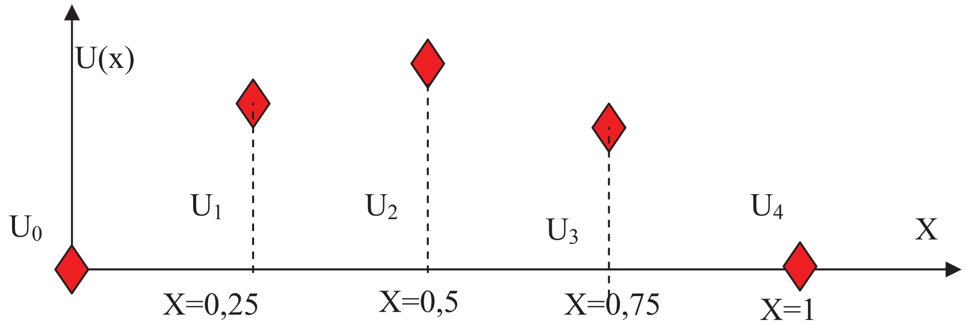
$$\left[\frac{1}{0{,}25^2}\begin{pmatrix}2&-1&0\\-1&2&-1\\0&-1&2\end{pmatrix}+\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}\right]\cdot\begin{Bmatrix}U_1\\U_2\\U_3\end{Bmatrix}=\begin{bmatrix}32+1&-16&0\\-16&32+1&-16\\0&-16&32+1\end{bmatrix}\cdot\begin{Bmatrix}U_1\\U_2\\U_3\end{Bmatrix}=\begin{Bmatrix}0{,}25\\0{,}5\\0{,}75\end{Bmatrix}$$

La solución aproximada resulta $$\begin{Bmatrix}U_1\\U_2\\U_3\end{Bmatrix}=\begin{Bmatrix}0{,}03488525\\0{,}05632582\\0{,}05003676\end{Bmatrix}$$

$$-\frac{d^2u(x)}{dx^2}+u(x)-x=0 \quad en \;\; \Omega=\{x\,\varepsilon\,R:0\le x\le 1\}$$

$$u(0)=0$$

$$u(1)=0$$



La solución aproximada completa, que incluye las condiciones de borde es:

$$\begin{Bmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0{,}03488525 \\ 0{,}05632582 \\ 0{,}05003676 \\ 0 \end{Bmatrix}$$

Justificar que con la solución obtenida una aproximación de la función derivada primera de esta solución se puede calcular mediante:

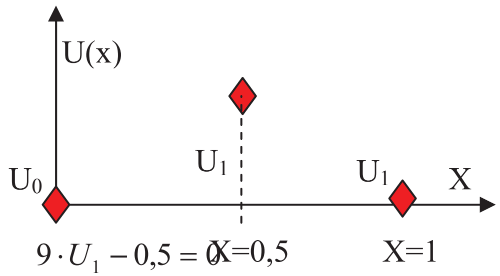
$$\begin{Bmatrix} U_0^{(1)} \\ U_1^{(1)} \\ U_2^{(1)} \\ U_3^{(1)} \\ U_4^{(1)} \end{Bmatrix} = \frac{1}{(2*0{,}25)} \begin{bmatrix} -3 & 4 & -1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & -4 & 3 \end{bmatrix} \begin{Bmatrix} 0 \\ 0{,}03488525 \\ 0{,}05632582 \\ 0{,}05003676 \\ 0 \end{Bmatrix}$$

Caso N=2

$U_0 = 0 \quad en \ X_0$

$-\frac{1}{0{,}5^2}\left[U_0 - 2\cdot U_1 + U_2\right] + U_1 - X_1 = 0 \quad en \ X_1$

$U_1 = 0 \quad en \ X_1$



O bien

$9\cdot U_1 - 0{,}5 = 0$

La solución aproximada es

$$U_1 = 1/18 = 0{,}055555$$

Así el error respecto de la solución exacta en ese punto es

$$E_2 = \left| \frac{1}{18} - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| = 0{,}001035002$$

$$E(abs)_2 = \left| \frac{1}{18} - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| / (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) = 1{,}83\%$$

Caso N=4

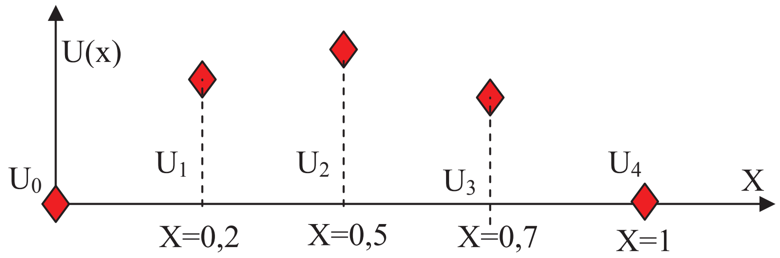
$$U_0 = 0 \quad en\ X_0$$

$$-\frac{1}{0{,}5^2}\left[U_0 - 2\cdot U_1 + U_2\right] + U_1 - X_1 = 0 \quad en\ X_1 = 0{,}25$$

$$-\frac{1}{0{,}5^2}\left[U_1 - 2\cdot U_2 + U_3\right] + U_2 - X_2 = 0 \quad en\ X_2 = 0{,}5$$

$$-\frac{1}{0{,}5^2}\left[U_2 - 2\cdot U_3 + U_4\right] + U_3 - X_3 = 0 \quad en\ X_3 = 0{,}75$$

$$U_4 = 0 \quad en\ X_4$$



O bien

$$\begin{bmatrix} 33 & -16 & 0 \\ -16 & 33 & -16 \\ 0 & -16 & 33 \end{bmatrix} \cdot \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \end{Bmatrix} = \begin{Bmatrix} 0{,}25 \\ 0{,}5 \\ 0{,}75 \end{Bmatrix}$$

La solución aproximada es

$$\begin{Bmatrix} U_1 \\ U_2 \\ U_3 \end{Bmatrix} = \begin{Bmatrix} 0{,}03488525 \\ 0{,}05632582 \\ 0{,}05003676 \end{Bmatrix}$$

Así el error respecto de la solución exacta en ese punto es

$$E_4 = \left| U_2 - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| = 0{,}000264735$$

$$E(abs)_4 = \left| U_2 - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| / (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) = 0{,}47\%$$

Caso N=8

$U_0 = 0 \quad en \ X_0 = 0$

$-\frac{1}{(1/8)^2}\left[U_0 - 2\cdot U_1 + U_2\right] + U_1 - X_1 = 0 \quad en \ X_1 = 1/8$

$-\frac{1}{(1/8)^2}\left[U_1 - 2\cdot U_2 + U_3\right] + U_2 - X_2 = 0 \quad en \ X_2 = 2/8$

$-\frac{1}{(1/8)^2}\left[U_2 - 2\cdot U_3 + U_4\right] + U_3 - X_3 = 0 \quad en \ X_3 = 3/8$

....

$U_8 = 0 \quad en \ X_8 = 1$

O bien

$$\begin{bmatrix} 129 & -64 & 0 & 0 & 0 & 0 & 0 \\ -64 & 129 & -64 & 0 & 0 & 0 & 0 \\ 0 & -64 & 129 & -64 & 0 & 0 & 0 \\ 0 & 0 & -64 & 129 & -64 & 0 & 0 \\ 0 & 0 & 0 & -64 & 129 & -64 & 0 \\ 0 & 0 & 0 & 0 & -64 & 129 & -64 \\ 0 & 0 & 0 & 0 & 0 & -64 & 129 \end{bmatrix} \cdot \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \\ U_7 \end{Bmatrix} = \begin{Bmatrix} 1/8 \\ 2/8 \\ 3/8 \\ 4/8 \\ 5/8 \\ 6/8 \\ 7/8 \end{Bmatrix}$$

La solución aproximada es

$$\mathbf{U}^T = \{0{,}0183367;\ 0{,}03500678;\ 0{,}04831759;\ 0{,}05652399;\ 0{,}05780107;\ 0{,}05021568;\ 0{,}03169615\}$$

Así el error respecto de la solución exacta en ese punto es

$$E_8 = \left| U_4 - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| = 6{,}65711\text{E-}05$$

$$E(abs)_8 = \left| U_4 - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| / (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) = 0{,}12\%$$

Evaluación del Error

Al considerar el error para distintos niveles de *disicretización*; es decir, distinto número de segmentos en que se divide el dominio, se tiene

$$E_N = \left| U_{N/2} - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| \text{ y } E(abs)_N = \left| U_{N/2} - (0{,}5 - \frac{senh(0{,}5)}{senh(1)}) \right| / (0{,}5 - \frac{senh(0{,}5)}{senh(1)})$$

Cuyas evaluaciones se presentan en la siguiente Tabla

N	Δx	$U_{aprox}(0{,}5)$	E_N	$E(abs)_N$
2	0,5	0,05555556	0,001035002	1,83%
4	0,25	0,05632582	0,000264735	0,47%
8	0,125	0,05652399	6,65711E-05	0,12%
16	0,0625	0,05657389	1,66672E-05	0,03%

Si se asume una relación exponencial entre $E(abs)_N$ y Δx, la aproximación por mínimos cuadrados da:

$$E(abs)_N = C \cdot \Delta x^P = e^{-2{,}6168} \cdot \Delta x^{1{,}9861}$$

Que indica una relación del orden de $\Delta x^{1{,}9861} \cong \Delta x^2$ que es el error de truncamiento local de la aproximación de derivada segunda utilizado.

DISCRETIZACIÓN DE ECUACIONES DIFERENCIALES A DERIVADAS PARCIALES

Se busca *u(x,t)* solución de

$$-12\frac{\partial^2 u(x,t)}{\partial x^2}+x^2\frac{\partial^2 u(x,t)}{\partial t^2}=0 \quad en\ \ \Omega=\{x\,\varepsilon\,R:0\le x\le 1\}$$

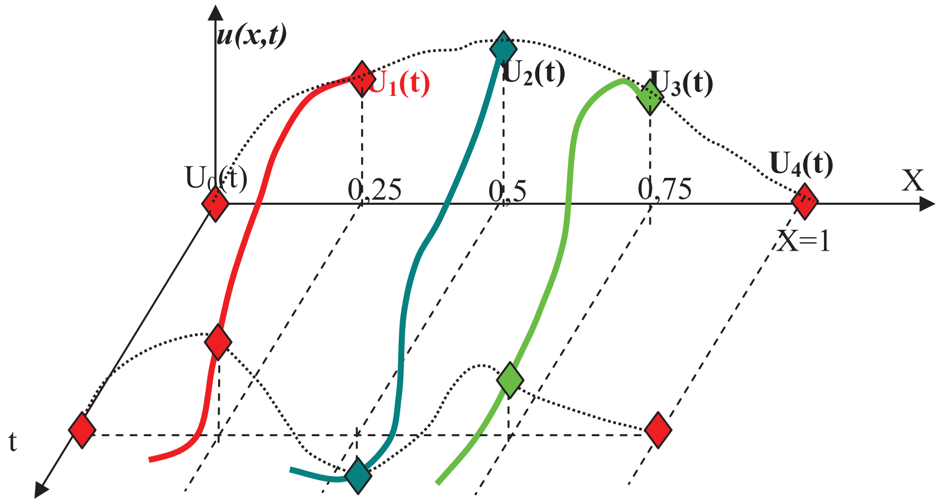
$$u(0,t)=0 \qquad\qquad u(x,0)=\sin(\pi\cdot x)$$

$$u(1,t)=0 \qquad\qquad \frac{\partial u}{\partial t}(x,0)=3$$

Se plantea encontrar una ***solución aproximada en forma de función discreta en la variable x, aunque continua en la variable t***. Se pretende encontrar las funciones $U_k(t)=u(X_k,t)$ con k=0,N, en N+1 puntos elegidos del dominio x, identificados con su abscisa X_k.

Cuando se consideran derivadas parciales respecto a la variable x, para t constante, la función a derivar es una *función discreta* y se puede hacer *derivadas numéricas*.

Cuando se consideran derivadas parciales respecto a la variable t, para x constante, la función a derivar es una *función continua* y se puede hacer *derivadas analíticas*.

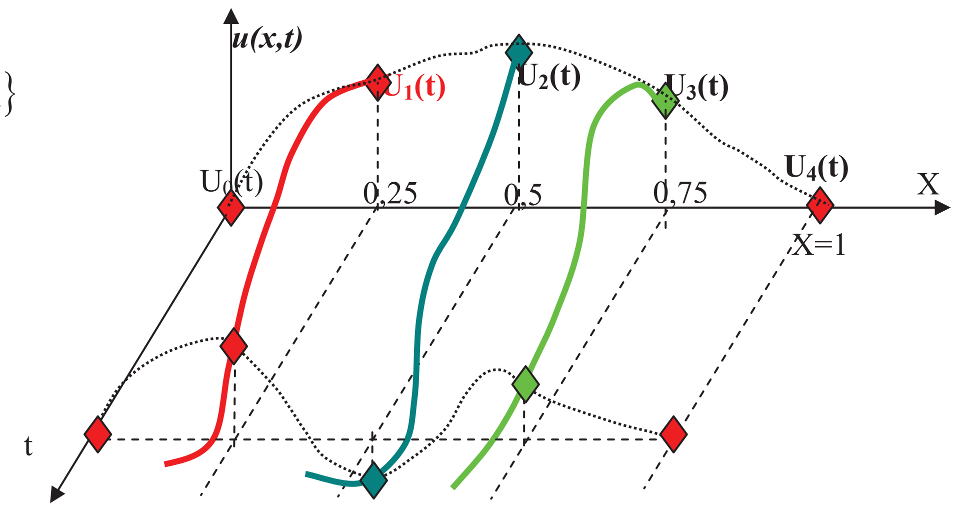


Se busca *u(x,t)* solución de

$$-12\frac{\partial^2 u(x,t)}{\partial x^2}+x^2\frac{\partial^2 u(x,t)}{\partial t^2}=0 \quad en\ \ \Omega=\{x\,\varepsilon\, R:0\le x\le 1\}$$

$$u(0,t)=0 \qquad\qquad u(x,0)=\sin(\pi\cdot x)$$

$$u(1,t)=0 \qquad\qquad \frac{\partial u}{\partial t}(x,0)=3$$



En cada uno de los X_k se puede plantear la ecuación diferencial a resolver pero con una aproximación de la derivada segunda respecto a x en forma de derivada numérica; y con la derivada respecto de la variable t en forma analítica evaluada en esa abscisa X_k. Así se puede escribir:

en X_0 se debe cumplir que: $$U_0(t)=0$$

en X_1 se debe cumplir que: $$-\frac{12}{0{,}25^2}\left[U_0(t)-2\cdot U_1(t)+U_2(t)\right]+(X_1)^2\cdot\frac{d^2U_1(t)}{dt^2}=0$$

en X_2 se debe cumplir que: $$-\frac{12}{0{,}25^2}\left[U_1(t)-2\cdot U_2(t)+U_3(t)\right]+(X_2)^2\cdot\frac{d^2U_2(t)}{dt^2}=0$$

en X_3 se debe cumplir que: $$-\frac{12}{0{,}25^2}\left[U_2(t)-2\cdot U_3(t)+U_4(t)\right]+(X_3)^2\cdot\frac{d^2U_3(t)}{dt^2}=0$$

en X_4 se debe cumplir que: $$U_4(t)=0$$

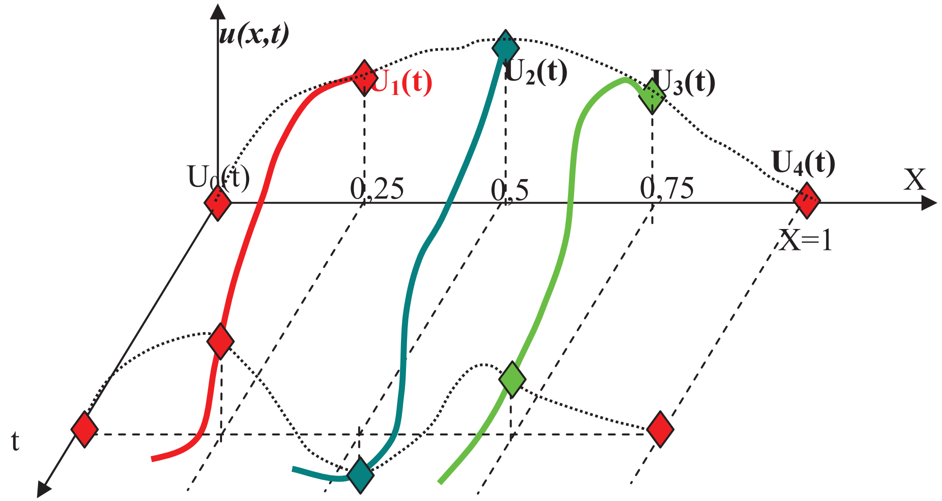
Se busca *u(x,t)* solución de

$$-12\frac{\partial^2 u(x,t)}{\partial x^2}+x^2\frac{\partial^2 u(x,t)}{\partial t^2}=0 \quad en\ \ \Omega=\{x\,\varepsilon\,R: 0\le x\le 1\}$$

$$u(0,t)=0 \qquad\qquad u(x,0)=\sin(\pi\cdot x)$$

$$u(1,t)=0 \qquad\qquad \frac{\partial u}{\partial t}(x,0)=3$$

Así se puede escribir:



$$\frac{12}{0{,}25^2}\begin{bmatrix}2&-1&0\\-1&2&-1\\0&-1&2\end{bmatrix}\begin{Bmatrix}U_1(t)\\U_2(t)\\U_3(t)\end{Bmatrix}+\begin{bmatrix}0{,}25^2&0&0\\0&0{,}50^2&0\\0&0&0{,}75^2\end{bmatrix}\begin{Bmatrix}\frac{d^2U_1(t)}{dt^2}\\\frac{d^2U_2(t)}{dt^2}\\\frac{d^2U_3(t)}{dt^2}\end{Bmatrix}=\begin{Bmatrix}0\\0\\0\end{Bmatrix}$$

con las condiciones iniciales $\begin{Bmatrix}U_1(0)\\U_2(0)\\U_3(0)\end{Bmatrix}=\begin{Bmatrix}sen(\pi\cdot 0{,}25)\\sen(\pi\cdot 0{,}50)\\sen(\pi\cdot 0{,}75)\end{Bmatrix}$ y $\begin{Bmatrix}\frac{dU_1}{dt}(0)\\\frac{dU_2}{dt}(0)\\\frac{dU_3}{dt}(0)\end{Bmatrix}=\begin{Bmatrix}3\\3\\3\end{Bmatrix}$

Este sistema se puede resolver con métodos analíticos o con métodos numéricos.

SOLUCIÓN NUMÉRCIA DE ECUACIONES DIFERENCIALES CON VALORES INICIALES

Ecuaciones Diferenciales Ordinarias de Primer Orden

- Método de Euler
- Método de Runge Kutta de 2do Orden

Sistemas de Ecuaciones Diferenciales Ordinarias de Primer Orden

Ecuaciones Diferenciales Ordinarias de Segundo Orden

- Reducción a Sistemas de primer orden

Sistemas de Ecuaciones Diferenciales Ordinarias de Segundo Orden

- Método de Diferencia Central
- Reducción a Sistemas de primer orden

Ejemplos

Resumen

ECUACIONES DIFERENCIALES ORDINARIAS DE PRIMER ORDEN

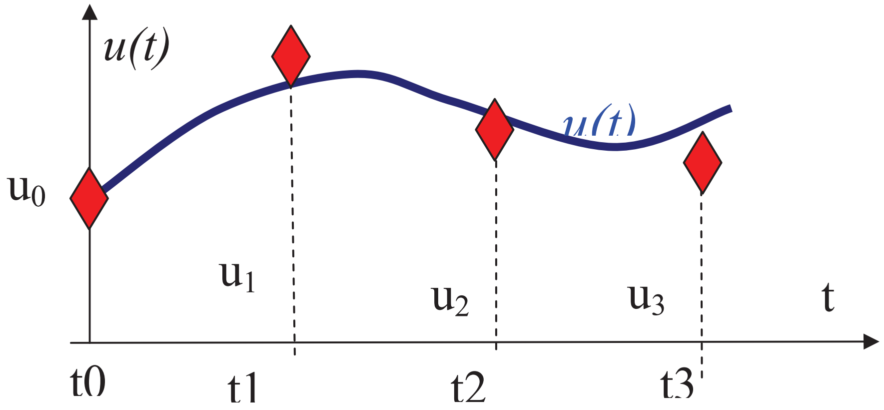
Se busca obtener una solución aproximada en forma discreta de la siguiente EDO

$$\begin{cases} \dfrac{du(t)}{dt} = f(t, u(t)) \\ u(t_0) = U_0 \end{cases}$$

Forma genércia.

$$\begin{cases} \dfrac{du(t)}{dt} = -A \cdot u(t) \\ u(t_0) = U_0 \end{cases}$$

Ejemplo



La solución discreta es tal que aproxima a la solución exacta del problema $u_k \cong u(t_k)$

<u>Solución en base a Serie de Taylor</u>

en t_k se conoce u_k y sus derivadas

$$u(t_k + \Delta t) = u(t_k) + \Delta t \cdot \left.\frac{du}{dt}\right|_{t_k} + \frac{\Delta t}{2} \cdot \left.\frac{d^2u}{dt^2}\right|_{t_k} + \ldots\ldots\ldots$$

<u>Métodos basados en Derivación Numérica</u>

se consideran aproximaciones

$$\left.\frac{du(t)}{dt}\right|_{t_n} = \frac{1}{\Delta t}(u_{n+1} - u_n) + O(\Delta t) \qquad \left.\frac{du(t)}{dt}\right|_{t_{n+1}} = \frac{1}{\Delta t}(u_{n+1} - u_n) + O(\Delta t)$$

$$\left.\frac{du(t)}{dt}\right|_{t_n} = \frac{1}{\Delta t}(u_{n+1} - u_{n-1}) + O(\Delta t^2)$$

<u>Métodos basados en Integración Numérica</u>

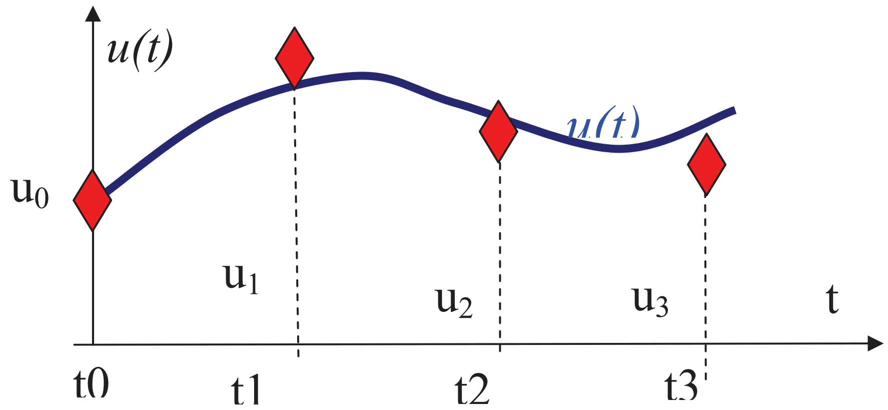
se considera la integral definida

$$\int_{ua}^{ub} du = \int_a^b f(t, u(t))\, dt \qquad ub - ua = \sum_{k=1}^{NP} w_k \cdot f(t_k, u_k) + O(\Delta t^p)$$

ECUACIONES DIFERENCIALES ORDINARIAS DE PRIMER ORDEN

Se busca obtener una solución aproximada en forma discreta de la siguiente EDO

$$\begin{cases} \dfrac{du(t)}{dt} = f(t,u(t)) \\ u(t_0) = U_0 \end{cases} \qquad \begin{cases} \dfrac{du(t)}{dt} = -A \cdot u(t) \\ u(t_0) = U_0 \end{cases}$$



<u>Solución en base a Serie de Taylor</u>

$$\boxed{u(t_k + \Delta t) = u(t_k) + \Delta t \cdot \left.\frac{du}{dt}\right|_{t_k} + \frac{\Delta t^2}{2} \cdot \left.\frac{d^2u}{dt^2}\right|_{t_k} + \ldots\ldots\ldots\ldots}$$

en t_k se conoce u_k y $\left.\dfrac{du(t)}{dt}\right|_{t_k} = f(t_k, u(t_k)) = f(t_k, u_k) = f_k$ $\qquad \left.\dfrac{d^2u(t)}{dt^2}\right|_{t_k} = \left.\dfrac{\partial f(t,u(t))}{\partial t} + \dfrac{\partial f(t,u(t))}{\partial u} \cdot \dfrac{du}{dt}\right|_{t_k}$

Solución de Primer Orden

$$\boxed{u(t_k + \Delta t) = u(t_k) + \Delta t \cdot f(t_k, u_k) + O(\Delta t^2)}$$

Solución de Segundo Orden

$$\boxed{u(t_k + \Delta t) = u(t_k) + \Delta t \cdot \left.\frac{du}{dt}\right|_{t_k} + \frac{\Delta t^2}{2} \cdot \left.\left(\frac{\partial f(t,u(t))}{\partial t} + \frac{\partial f(t,u(t))}{\partial u} \cdot \frac{du}{dt} \right)\right|_{t_k} + O(\Delta t^3)}$$

EDO DE PRIMER ORDEN-MÉTODO DE EULER

Se busca obtener una solución aproximada de la EDO

$$\begin{cases} \dfrac{du(t)}{dt} = f(t, u(t)) \\ u(t_0) = U_0 \end{cases}$$

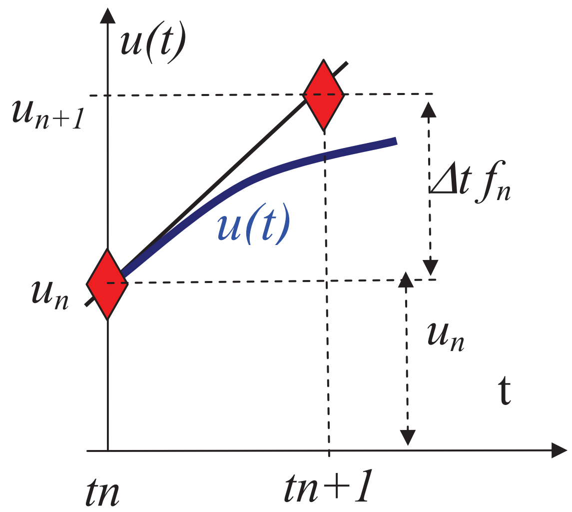
Métodos basados en Derivación Numérica

EULER Adelante considera que $\left.\dfrac{du(t)}{dt}\right|_{t_n} = \dfrac{1}{\Delta t}(u_{n+1} - u_n) + O(\Delta t)$

$$\left.\frac{du(t)}{dt}\right|_{t_n} = f(t_n, u_n)$$

$$\boxed{\begin{aligned} u_{n+1} &= u_n + \Delta t \cdot f(t_n, u_n) + O(\Delta t^2) \\ t_{n+1} &= t_n + \Delta t \end{aligned}}$$

EXPLÍCITO



EULER Atrás considera que $\left.\dfrac{du(t)}{dt}\right|_{t_{n+1}} = \dfrac{1}{\Delta t}(u_{n+1} - u_n) + O(\Delta t)$

$$\left.\frac{du(t)}{dt}\right|_{t_{n+1}} = f(t_{n+1}, u_{n+1})$$

$$\boxed{\begin{aligned} u_{n+1} &= u_n + \Delta t \cdot f(t_{n+1}, u_{n+1}) + O(\Delta t^2) \\ t_{n+1} &= t_n + \Delta t \end{aligned}}$$

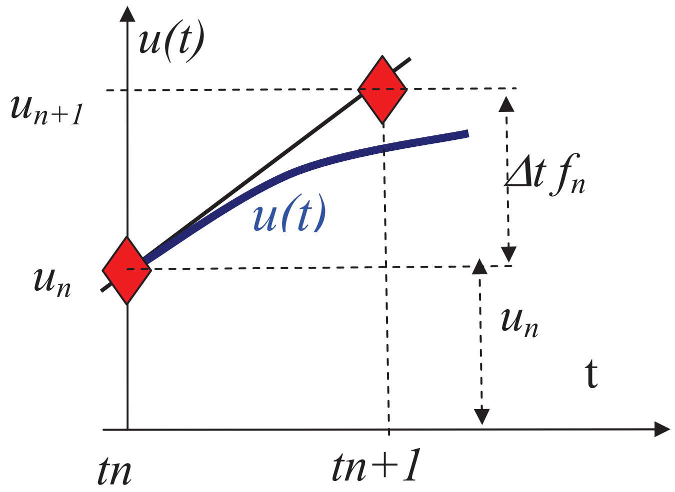
IMPLÍCITO

MÉTODO DE EULER ATRÁS -EJEMPLO

La Ecuación diferencial ordinaria de primer orden

$$\frac{dy(t)}{dt} = 2 \cdot t \cdot y(t)^2 \qquad con \quad y(0) = 1$$

tiene la solución exacta $y_{ex}(t) = 1/(1-t^2)$



Se busca obtener una solución aproximada de la EDO mediante el Método de EULER ATRÁS

Dado (t_m, y_m),la nueva solución (t_{m+1}, y_{m+1}) se calcula con

$$y_{m+1} = y_m + \Delta t \ f(t_{m+1}, y_{m+1})$$

$$t_{m+1} = t_m + \Delta t$$

Identificar la función *f(t,y)*.

Obtener una solución aproximada en el intervalo [0,1), con el método de Euler Atrás y el paso Δt=0.25

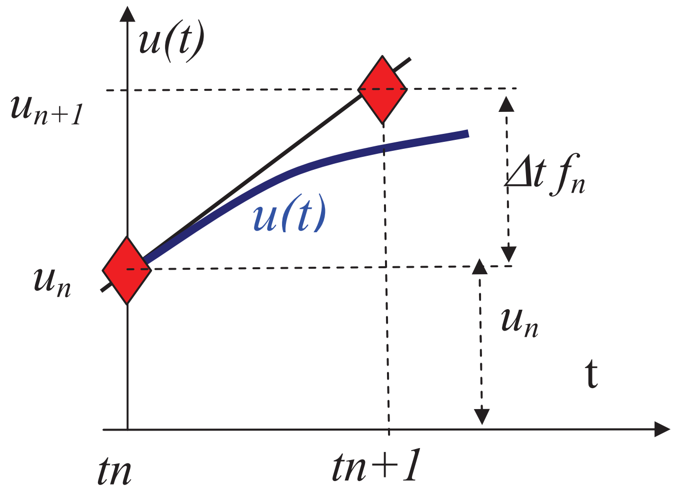
…………….se debe resolver una ecuación no lineal!!!!!!!!

MÉTODO DE EULER ADELANTE -EJEMPLO

La Ecuación diferencial ordinaria de primer orden

$$\frac{dy(t)}{dt} = 2 \cdot t \cdot y(t) \quad con \quad y(0) = 1$$

tiene la solución exacta $y_{ex}(t) = 1/(1-t^2)$



Se busca obtener una solución aproximada de la EDO mediante el Método de EULER ADELANTE

Dado (t_m, y_m),

la nueva solución (t_{m+1}, y_{m+1}) se calcula con

$$k_1 = \Delta t \ f(t_m, y_m)$$

$$y_{m+1} = y_m + k_1$$

$$t_{m+1} = t_m + \Delta t$$

Identificar la función *f(t,y)*.

Obtener una solución aproximada en el intervalo [0,1], con el método de Euler adelante y el paso Δt=0.25:

t	y	k1=Δt *f(t,y)	y(t+Δt)	y_{ex}(t)	Error=abs(y_{ex}(t)-y_n)
0	1				
0,25					
0,5					

MÉTODO DE EULER-EJEMPLO

La Ecuación diferencial ordinaria de primer orden

$$\frac{dy(t)}{dt} + \frac{y(t)}{2} = \frac{t}{2} \quad con \quad y(0) = 4$$ tiene la solución exacta $$y_{ex}(t) = 6 \cdot e^{-t/2} - 2 + t$$

Se busca obtener una solución aproximada de la EDO mediante el Método de EULER (ADELANTE)

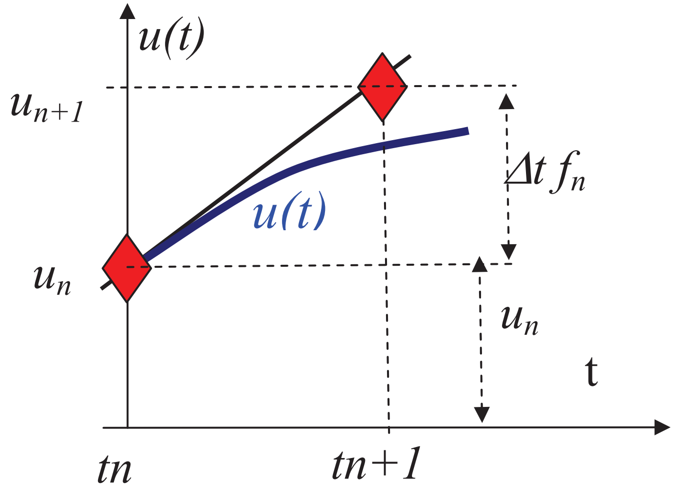
Dado (t_m, y_m),

la nueva solución (t_{m+1}, y_{m+1}) se calcula con

$$k_1 = \Delta t \; f(t_m, y_m)$$

$$y_{m+1} = y_m + k_1$$

$$t_{m+1} = t_m + \Delta t$$



Identificar la función *f(t,y)*.

Obtener una solución aproximada en el intervalo [0,1], con el método de Euler adelante y el paso Δt=0.25:

t	y	k1=Δt *f(t,y)	y(t+Δt)	$y_{ex}(t)$	Error=abs(y_{ex}(t)-y_n)
0	4				
0,25					
0,5					
0,75					
1					

Método de EULER (ADELANTE) en MATLAB

```matlab
function Euler_00
y0=4;       % Valores Iniciales
t0=0;
Dt=0.01;  % incremento de tiempo
NDt=1000;  % cantidad de Dt a realizar
% Dimensionamiento
t=zeros(NDt,1);    % vector para tiempo
y=zeros(NDt,1);    % vector para y(t)
```

$$\frac{dy(t)}{dt}+\frac{y(t)}{2}=\frac{t}{2}$$

$$con \quad y(0)=4$$

```matlab
% Inicialización
t(1)=t0;
y(1)=y0;
% EULER
for j=1:NDt-1
  k1=Dt*(-(1/2)*y(j) + (1/2)*t(j));
   y(j+1)  = y(j)+k1;
   t(j+1)  = t(j)+Dt;
end
% Graficación
figure(1)
plot(t,y(:,1),'b');
end
```

$$Conocido \ (t_m, y_m)$$

$$k_1 = \Delta t \ f(t_m, y_m)$$

$$y_{m+1} = y_m + k_1$$

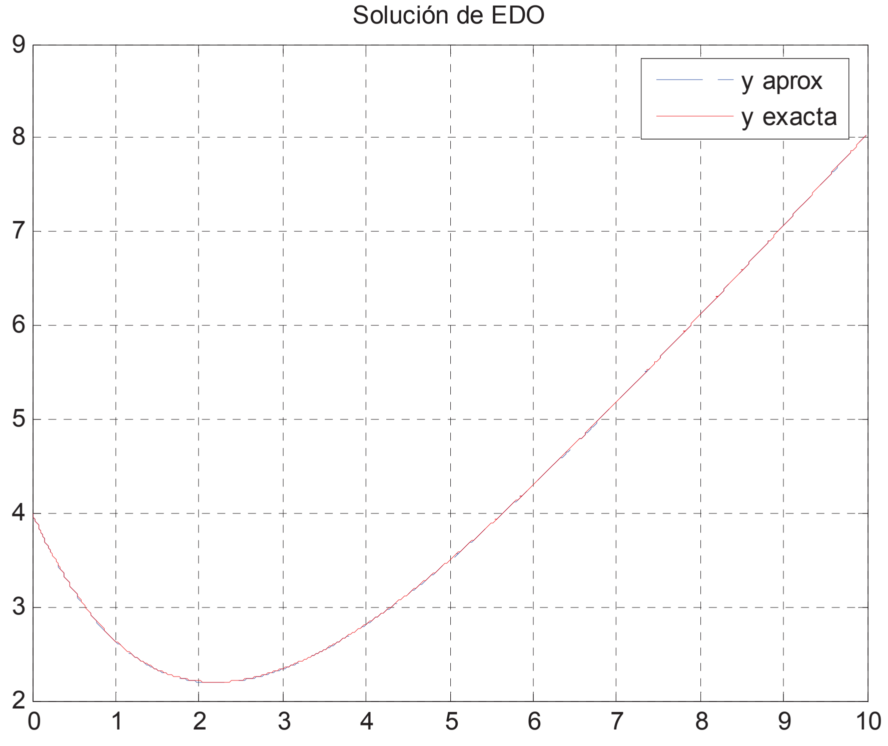
$$t_{m+1} = t_m + \Delta t$$

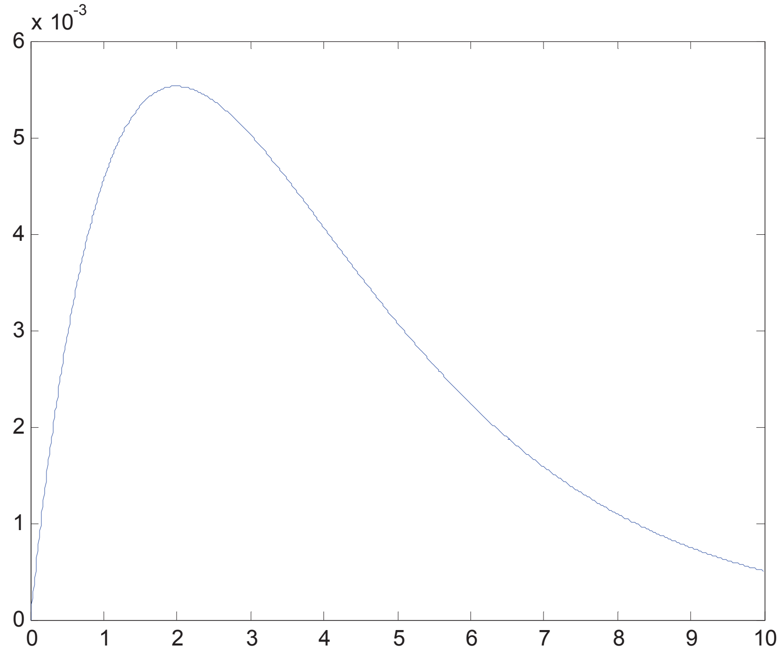
MÉTODO DE EULER-EJEMPLO

La Ecuación diferencial ordinaria de primer orden

$$\frac{dy(t)}{dt}+\frac{y(t)}{2}=\frac{t)}{2} \quad con \quad y(0)=4 \text{ tiene la solución exacta } y_{ex}(t)=6\cdot e^{-t/2}-2+t$$

Con Δt=0.01 se obtienen las siguientes gráficas





EDO DE PRIMER ORDEN-MÉTODO DE PUNTO MEDIO

Se busca obtener una solución aproximada de la EDO

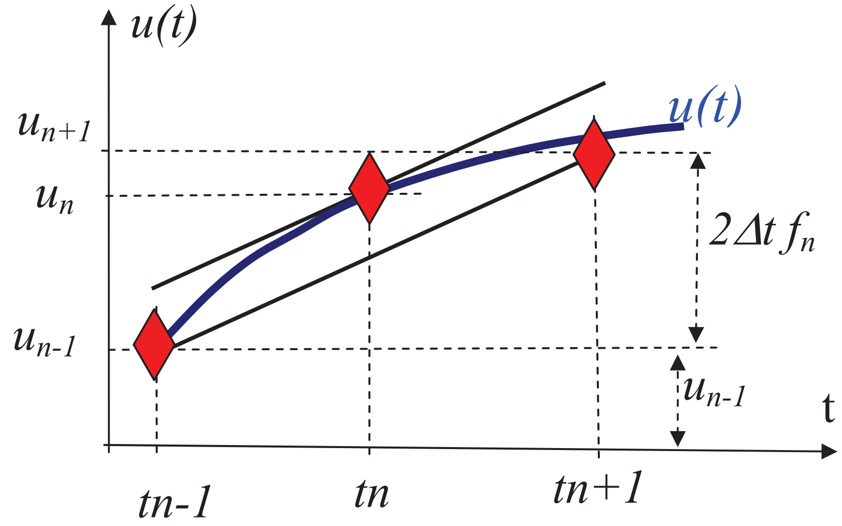
$$\begin{cases} \dfrac{du(t)}{dt} = f(t,u(t)) \\ u(t_0) = U_0 \end{cases}$$

Métodos basados en Derivación Numérica

Punto Medio considera que

$$\left.\frac{du(t)}{dt}\right|_{t_n} = \frac{1}{2\Delta t}(u_{n+1} - u_{n-1}) + O(\Delta t^2)$$

$$\left.\frac{du(t)}{dt}\right|_{t_n} = f(t_n, u_n)$$



$$\boxed{\begin{aligned} u_{n+1} &= u_{n-1} + 2\Delta t \cdot f(t_n, u_n) + O(\Delta t^3) \\ t_{n+1} &= t_n + \Delta t \end{aligned}}$$

Método Multipaso
Orden del Error es 3

EDO DE PRIMER ORDEN-MÉTODOS DE RUNGE KUTTA DE SEGUNDO ORDEN

Dado (t_m, y_m)

se busca calcular la nueva solución (t_{m+1}, y_{m+1})

$$y_{m+1} = y_m + \Delta t\, \Phi(t_m, y_m, \Delta t)$$

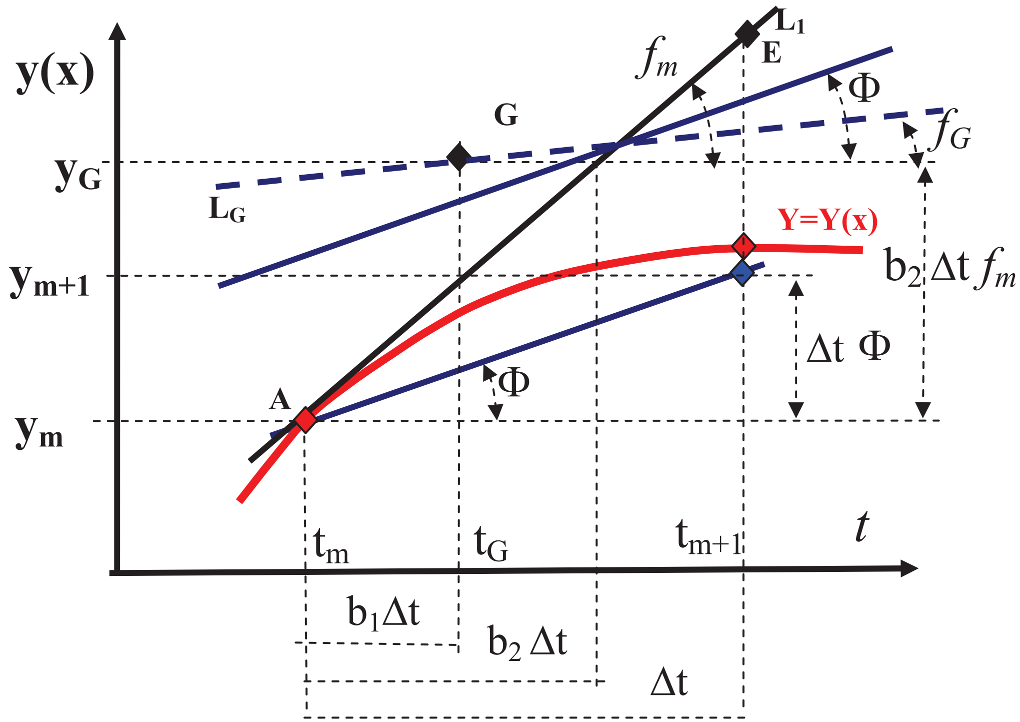
$$t_{m+1} = t_m + \Delta t$$

Siendo

$$\Phi(t_m, y_m, \Delta t) = a_1 \cdot f(t_m, y_m) + a_2 \cdot f(t_G, y_G)$$

$$t_G = (t_m + b_1 \cdot \Delta t)$$

$$y_G = (y_m + b_2 \Delta t \cdot f_m)$$



Se expande en Taylor

$$f(t_G, y_G) = f(t_m, y_m) +$$

$$+ (b_1 \Delta t) \cdot \left.\frac{\partial f}{\partial t}\right|_{t_m} + (b_2 \Delta t \cdot f_m) \cdot \left.\frac{\partial f}{\partial y}\right|_{t_m} + O(\Delta t^2)$$

$$\Delta t \cdot \Phi(t_m, y_m, \Delta t) = \Delta t \cdot \left\{ (a_1 + a_2) \cdot f(t_m, y_m) + a_2 \cdot (b_1 \Delta t) \cdot \left.\frac{\partial f}{\partial t}\right|_{t_m} + a_2 \cdot (b_2 \Delta t \cdot f_m) \cdot \left.\frac{\partial f}{\partial y}\right|_{t_m} + O(\Delta t^2) \right\}$$

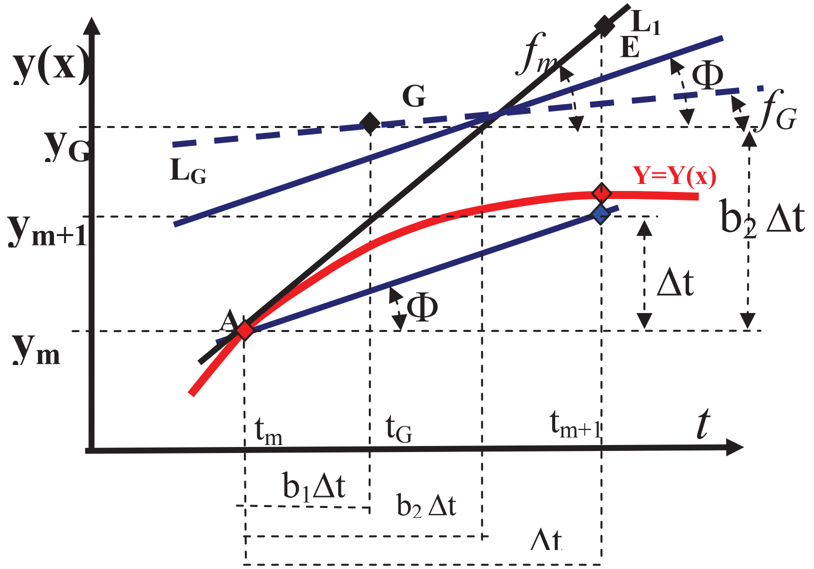
EDO DE PRIMER ORDEN-MÉTODOS DE RUNGE KUTTA DE SEGUNDO ORDEN

Dado (t_m, y_m)

se busca calcular la nueva solución (t_{m+1}, y_{m+1})

$$y_{m+1} = y_m + \Delta t\, \Phi(t_m, y_m, \Delta t)$$

$$t_{m+1} = t_m + \Delta t$$



Siendo

$$\Phi(t_m, y_m, \Delta t) = a_1 \cdot f(t_m, y_m) + a_2 \cdot f(t_G, y_G)$$

$$t_G = (t_m + b_1 \cdot \Delta t)$$

$$y_G = (y_m + b_2 \Delta t \cdot f_m)$$

$$\Delta t \cdot \Phi(t_m, y_m, \Delta t) = \Delta t \cdot \left\{ (a_1 + a_2) \cdot f(t_m, y_m) + a_2 \cdot (b_1 \Delta t) \cdot \left.\frac{\partial f}{\partial t}\right|_{t_m} + a_2 \cdot (b_2 \Delta t \cdot f_m) \cdot \left.\frac{\partial f}{\partial y}\right|_{t_m} + O(\Delta t^2) \right\}$$

$$y_{m+1} = y_m + \Delta t \cdot (a_1 + a_2) \cdot f(t_m, y_m) + a_2 \cdot (b_1 \Delta t^2) \cdot \left.\frac{\partial f}{\partial t}\right|_{t_m} + a_2 \cdot (b_2 \Delta t^2 \cdot f_m) \cdot \left.\frac{\partial f}{\partial y}\right|_{t_m} + O(\Delta t^3)$$

Se compara con la solución por Serie de Taylor

$$u(t_k + \Delta t) = u(t_k) + \Delta t \cdot \left.\frac{du}{dt}\right|_{t_k} + \frac{\Delta t^2}{2} \cdot \left.\left(\frac{\partial f(t, u(t))}{\partial t} + \frac{\partial f(t, u(t))}{\partial u} \cdot \frac{du}{dt} \right)\right|_{t_k} + O(\Delta t^3)$$

Para que las dos Series sean iguales los coeficientes deben ser iguales

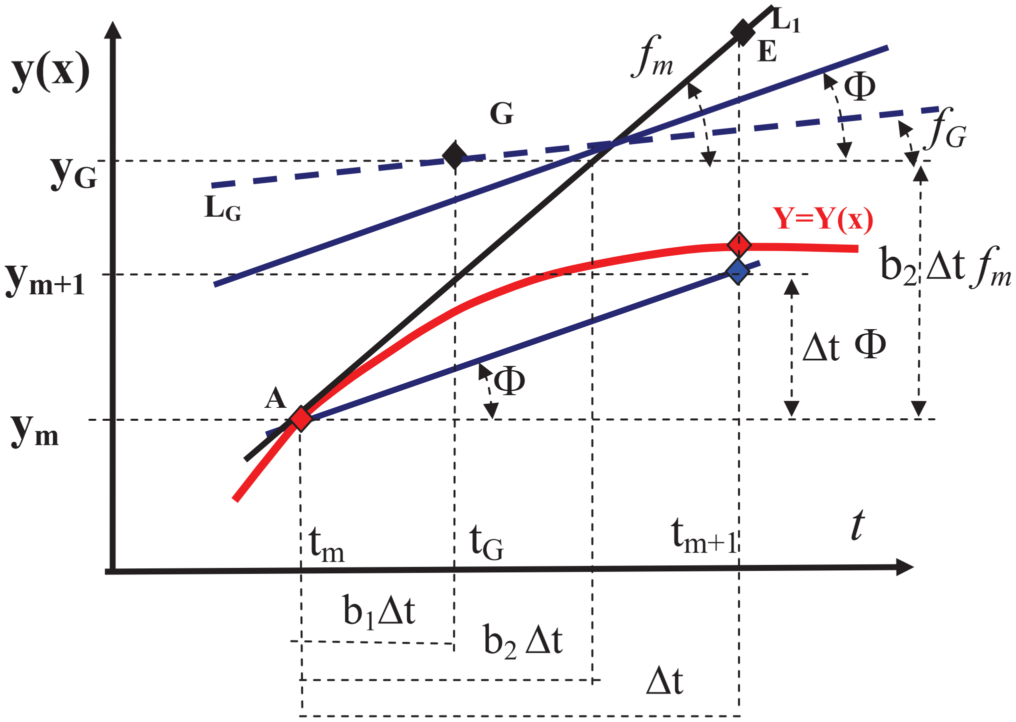
EDO DE PRIMER ORDEN-MÉTODOS DE RUNGE KUTTA DE SEGUNDO ORDEN

De comparar las series se obtiene que

$$a_1 + a_2 = 1 \qquad a_2 = \omega \neq 0$$

$$a_2 \cdot b_1 = 1/2 \quad \text{o bien} \quad a_1 = 1 - \omega$$

$$a_2 \cdot b_2 = 1/2 \qquad b_1 = b_2 = \frac{1}{2\omega}$$



Dado (t_m, y_m) se calcula

la nueva solución (t_{m+1}, y_{m+1})

$$y_{m+1} = y_m + \Delta t\,\Phi(t_m, y_m, \Delta t)$$

$$t_{m+1} = t_m + \Delta t$$

mediante

$$k_1 = \Delta t\ f\left(x_m, y_m\right) \qquad t_G = t_m + \Delta t/(2\omega) \qquad y_G = y_m + \frac{1}{2\omega} k_1$$

$$k_2 = \Delta t\ f\left(t_G, y_G\right)$$

$$y_{m+1} = y_m + (1-\omega)\,k_1 + \omega\,k_2$$

$$t_{m+1} = t_m + \Delta t$$

que es equivalente ha obtener:

$$y_{m+1} = y_m + \Delta t\left\{(1-\omega) f\left(t_m, y_m\right) + \omega f\left[\left(t_m + \frac{\Delta t}{2\omega}\right), \left(y_m + \frac{\Delta t}{2\omega} f\left(t_m, y_m\right)\right)\right]\right\} + O(\Delta t^3)$$

EDO DE PRIMER ORDEN-MÉTODOS DE RUNGE KUTTA DE SEGUNDO ORDEN

Dado (t_m, y_m), la nueva solución (t_{m+1}, y_{m+1}) se calcula con

$$k_1 = \Delta t\ f(x_m, y_m)$$

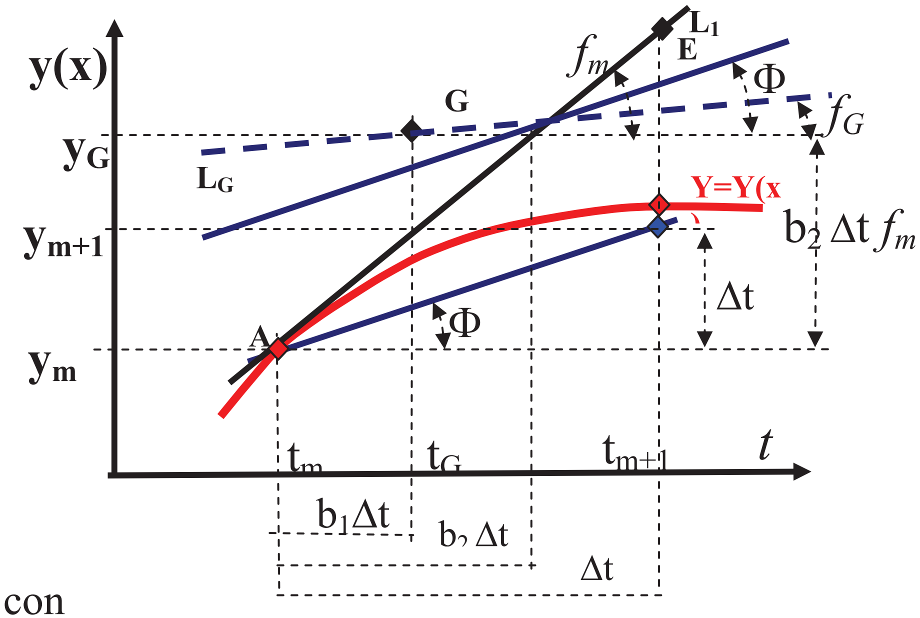
$$t_G = t_m + \Delta t/(2\omega)$$

$$y_G = y_m + \frac{1}{2\omega} k_1$$

$$k_2 = \Delta t\ f(t_G, y_G)$$

$$y_{m+1} = y_m + (1-\omega)\, k_1 + \omega\, k_2$$

$$t_{m+1} = t_m + \Delta t$$



Método de EULER MODIFICADO con ω=1

Dado (t_m, y_m), la nueva solución (t_{m+1}, y_{m+1}) se calcula con

$$k_1 = \Delta t\ f(t_m, y_m)$$

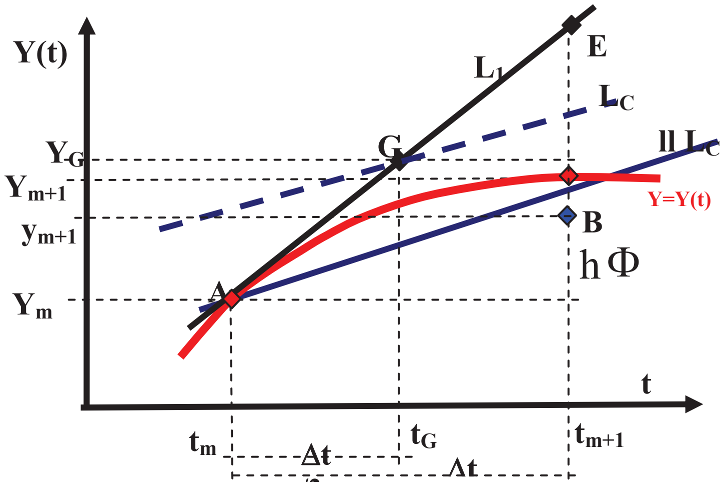
$$t_G = t_m + \Delta t/2$$

$$y_G = y_m + k_1/2$$

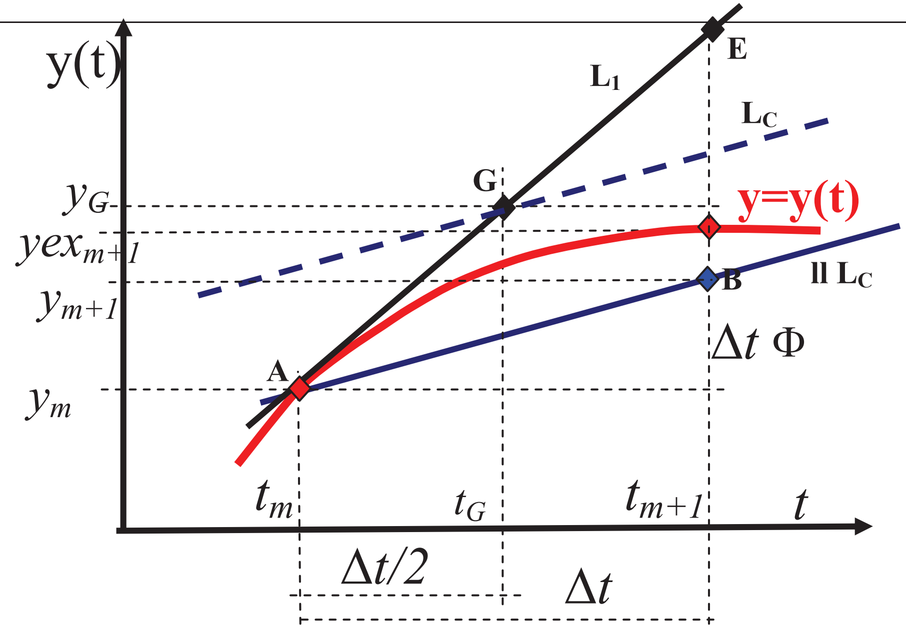
$$k_2 = \Delta t\ f(t_G, y_G)$$

$$y_{m+1} = y_m + k_2$$

$$t_{m+1} = t_m + \Delta t$$



MÉTODO DE EULER MODIFICADO



Dado (t_m, y_m), la nueva solución (t_{m+1}, y_{m+1}) se calcula con $\omega=1$

$$k_1 = \Delta t \; f(t_m, y_m)$$

$$t_G = t_m + \Delta t / 2$$

$$y_G = y_m + k_1 / 2$$

$$k_2 = \Delta t \; f(t_G, y_G)$$

$$y_{m+1} = y_m + k_2$$

$$t_{m+1} = t_m + \Delta t$$

EJEMPLO

La Ecuación diferencial ordinaria de primer orden

$$\frac{dy(t)}{dt} + \frac{y(t)}{2} = \frac{t}{2} \quad con \quad y(0) = 4$$

tiene la solución exacta $y_{ex}(t) = 6 \cdot e^{-t/2} - 2 + t$

Se busca obtener una solución aproximada de la EDOmediante el Método de EULER MODIFICADO

Obtener una solución aproximada en el intervalo [0,1], con el método de Euler adelante y el paso Δt=0.25:

t	y	k1= Δt *f(t,y)	tg= t + Δt /(2w)	yg= y+k1/(2w)	k2= Δt *f(t_G,y_G)	Y(t+Δt)= y+(1-w)k1+w*k2
0	4					
0,25						
0,5						

Método de EULER MODIFICADO en MATLAB

```
function Euler_00
y0=4;       % Valores Iniciales
t0=0;
Dt=0.01;   % incremento de tiempo
NDt=1000;   % cantidad de Dt a realizar
% Dimensionamiento
t=zeros(NDt,1);    % vector para tiempo
y=zeros(NDt,1);    % vector para y(t)
```

$$\frac{dy(t)}{dt} + \frac{y(t)}{2} = \frac{t}{2}$$

$$con \quad y(0) = 4$$

```
% Inicialización
t(1)=t0;
y(1)=y0;         w=1
% EULER
for j=1:NDt-1
    k1=Dt*(-(1/2)*y(j) + (1/2)*t(j));
      yg     = y(j) + k1/(2*w);
      tg     = t(j) + Dt/(2*w);
    k2=Dt*(-(1/2)*yg + (1/2)*tg);
    y(j+1)   = y(j)+(1-w)*k1+w*k2;
    t(j+1)   = t(j)+Dt;
end
% Graficación
figure(1)
plot(t,y(:,1),'b');
end
```

$$k_1 = \Delta t \; f\left(t_m, y_m\right)$$

$$t_G = t_m + \Delta t / 2$$

$$y_G = y_m + k_1 / 2$$

$$k_2 = \Delta t \; f\left(t_G, y_G\right)$$

$$y_{m+1} = y_m + k_2$$

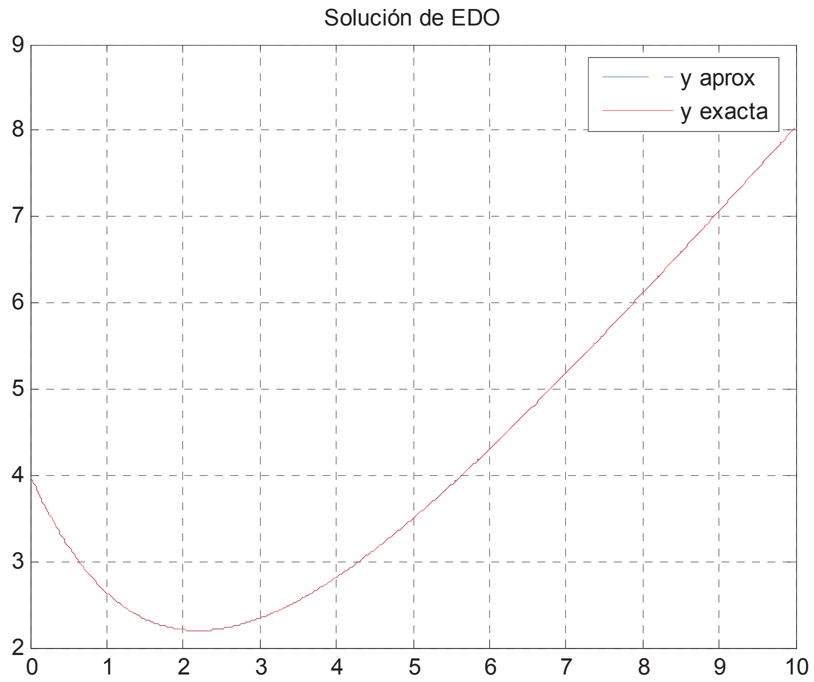
$$t_{m+1} = t_m + \Delta t$$

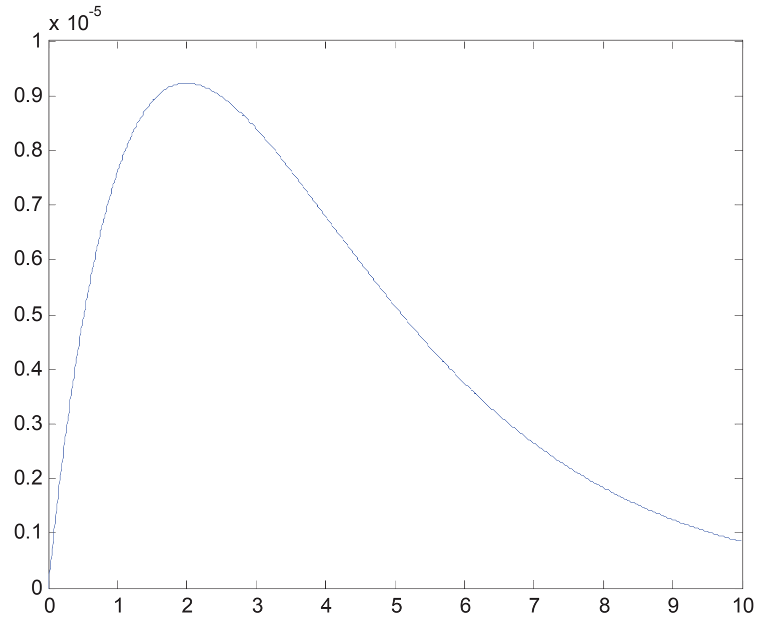
MÉTODO DE EULER-MODIFICADO

La Ecuación diferencial ordinaria de primer orden

$$\frac{dy(t)}{dt}+\frac{y(t)}{2}=\frac{t)}{2} \quad con \quad y(0)=4 \text{ tiene la solución exacta } y_{ex}(t)=6\cdot e^{-t/2}-2+t$$

Con $\Delta t=0.01$ se obtienen las siguientes gráficas





EDO DE PRIMER ORDEN-MÉTODOS DE RUNGE KUTTA DE SEGUNDO ORDEN

Dado (t_m, y_m), la nueva solución (t_{m+1}, y_{m+1}) se calcula con

$$k_1 = \Delta t \ f(x_m, y_m)$$

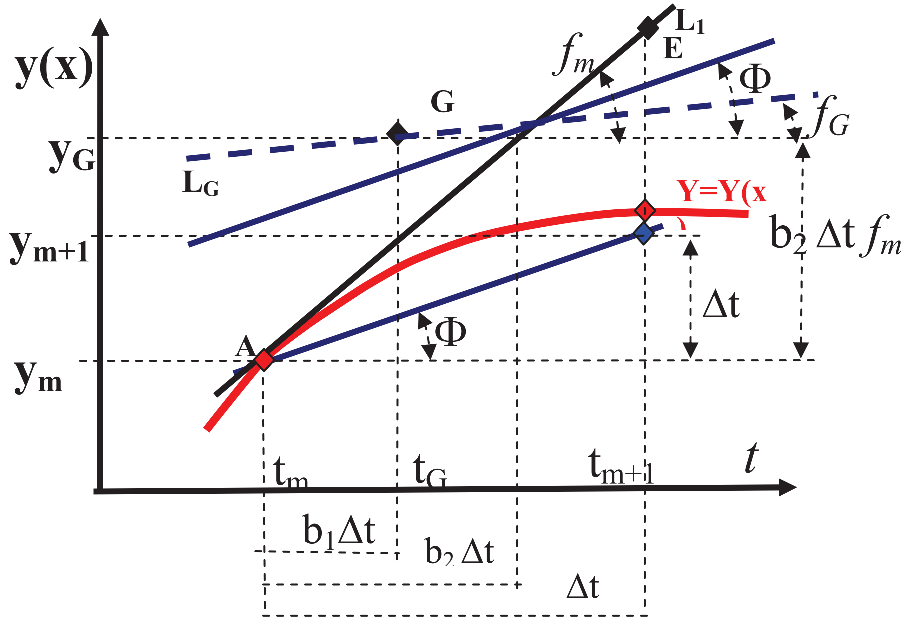
$$t_G = t_m + \Delta t/(2\omega)$$

$$y_G = y_m + \frac{1}{2\omega} k_1$$

$$k_2 = \Delta t \ f(t_G, y_G)$$

$$y_{m+1} = y_m + (1-\omega)\, k_1 + \omega\, k_2$$

$$t_{m+1} = t_m + \Delta t$$



Método de EULER MEJORADO con ω=1/2

Dado (t_m, y_m), la nueva solución (t_{m+1}, y_{m+1}) se calcula con

$$k_1 = \Delta t \ f(x_m, y_m)$$

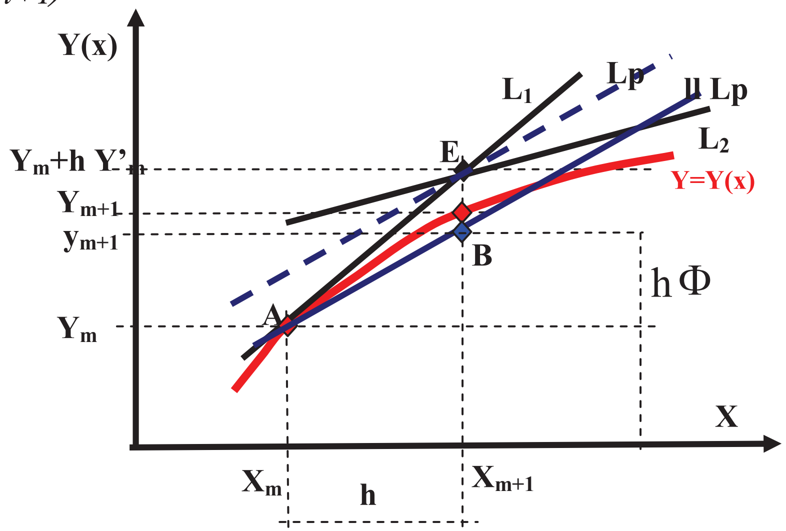
$$t_G = t_m + \Delta t$$

$$y_G = y_m + k_1$$

$$k_2 = \Delta t \ f(t_G, y_G)$$

$$y_{m+1} = y_m + (k_1 + k_2)/2$$

$$t_{m+1} = t_m + \Delta t$$



MÉTODO DE EULER MEJORADO

Dado (t_m, y_m), la nueva solución (t_{m+1}, y_{m+1}) se calcula con $\omega=1/2$

$$k_1 = \Delta t\ f(x_m, y_m)$$

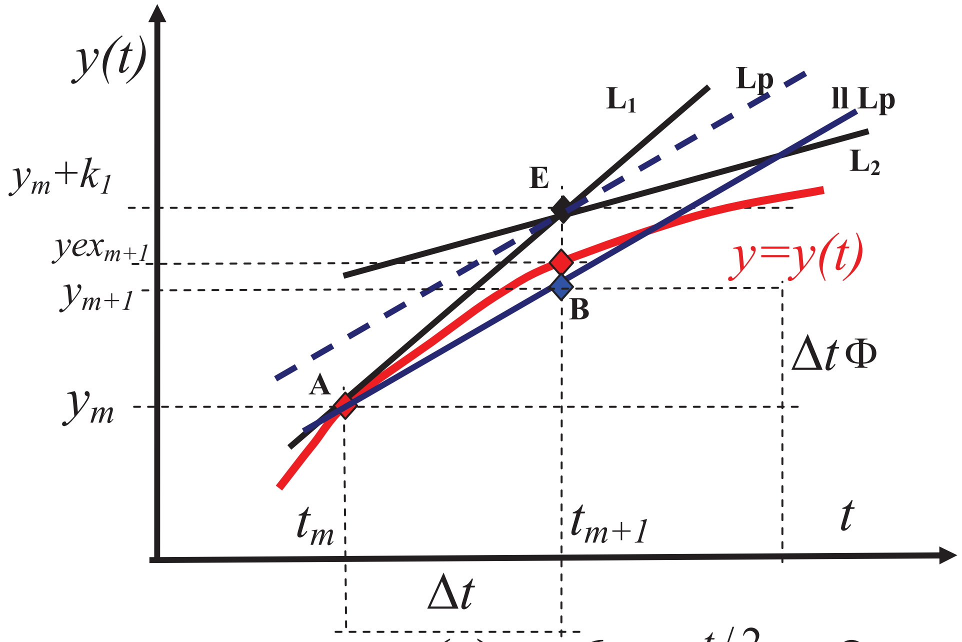
$$t_G = t_m + \Delta t$$

$$y_G = y_m + k_1$$

$$k_2 = \Delta t\ f(t_G, y_G)$$

$$y_{m+1} = y_m + (k_1 + k_2)/2$$

$$t_{m+1} = t_m + \Delta t$$



EJEMPLO la EDO

$$\frac{dy(t)}{dt} + \frac{y(t)}{2} = \frac{t}{2} \qquad con \quad y(0) = 4$$

tiene la solución exacta $y_{ex}(t) = 6 \cdot e^{-t/2} - 2 + t$

Se busca obtener una solución aproximada de la EDOmediante el Método de EULER MEJORADO

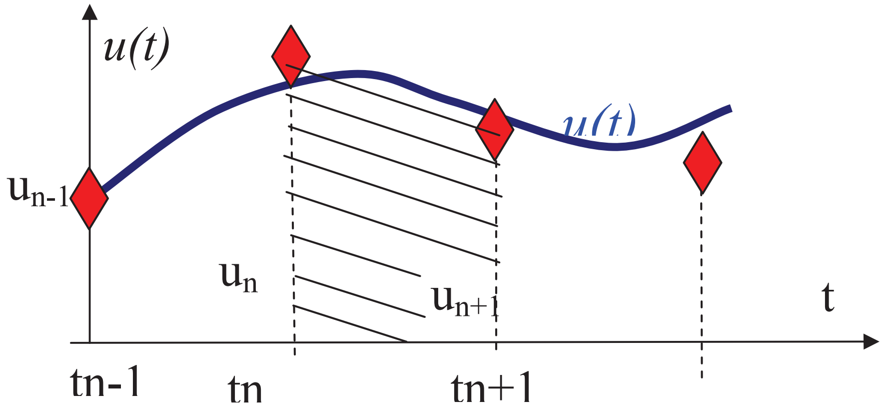
Obtener una solución aproximada en el intervalo [0,1], con el método de Euler adelante y el paso Δt=0.25:

t	y	k1= Δt *f(t,y)	xg= t + Δt /(2w)	yg= y+k1/(2w)	k2= Δt *f(t_G,y_G)	Y(t+Δt)= y+(1-w)k1+w*k2
0	4					
0,25						
0,5						

EDO DE PRIMER ORDEN-MÉTODO DE LOS TRAPECIOS

Se busca obtener una solución aproximada en forma discreta de la siguiente EDO

$$\begin{cases} \dfrac{du(t)}{dt} = f(t, u(t)) \\ u(t_0) = U_0 \end{cases}$$



<u>Métodos basados en Integración Numérica</u>

se considera la integral definida

$$\int_{u_n}^{u_{n+1}} du = \int_{t_n}^{t_{n+1}} f(t, u(t))\, dt$$

$$\boxed{u_{n+1} = u_n + \frac{\Delta t}{2}\left(f(t_n, u_n) + f(t_n, u_{n+1}) \right) + O(\Delta t^3)}$$

Método Implícito de un paso

Solución de ecuación no lineal

Corrector iterativo de una Predicción con algún método explícito

SOLUCIÓN NUMÉRCIA DE ECUACIONES DIFERENCIALES CON VALORES INICIALES

Ecuaciones Diferenciales Ordinarias de Primer Orden

- Método de Euler
- Método de Runge Kutta de 2^{do} Orden

Sistemas de Ecuaciones Diferenciales Ordinarias de Primer Orden

Ecuaciones Diferenciales Ordinarias de Segundo Orden

- Reducción a Sistemas de primer orden

Sistemas de Ecuaciones Diferenciales Ordinarias de Segundo Orden

- Método de Diferencia Central
- Reducción a Sistemas de primer orden

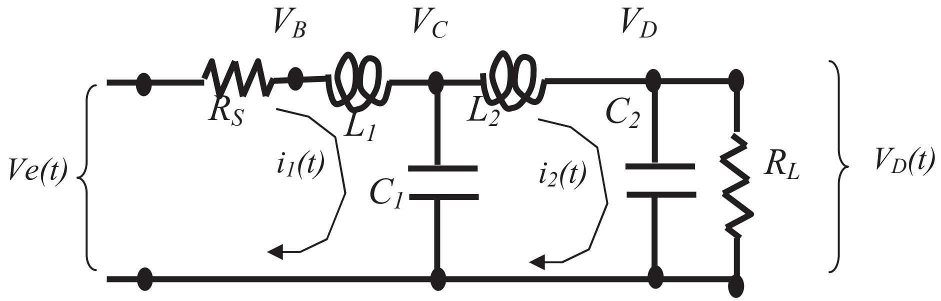
Ejemplos

Resumen

SISTEMAS DE EDO -CIRCUITOS ELÉCTRICOS

Circuito Butterworth para filtros pasa baja (Eronini Umez Eronini, *Dinàmica de Sistemas y Control,* Thomson Learning, 2001).

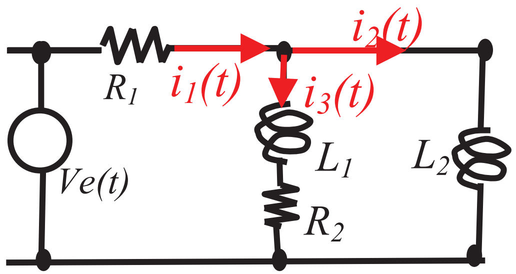
Se trata de un circuito de tipo RLC; es decir, con resistencias R, inductancias L, y capacitares C. Las ecuaciones que describen el comportamiento son las siguientes:



$$\begin{Bmatrix} \dfrac{di_1}{dt} \\ \dfrac{dV_C}{dt} \\ \dfrac{di_2}{dt} \\ \dfrac{dV_D}{dt} \end{Bmatrix} = \begin{bmatrix} -R_S/L_1 & -1/L_1 & 0 & 0 \\ 1/C_1 & 0 & -1/C_2 & 0 \\ 0 & 1/L_2 & 0 & -1/L_2 \\ 0 & 0 & +1/C_2 & -1/R_L C_2 \end{bmatrix} \begin{Bmatrix} i_1 \\ V_C \\ i_2 \\ V_D \end{Bmatrix} + \begin{Bmatrix} V_e/L_1 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

Circuitos RL y RC (Zill,G. Cullen, M. *Ecuaciones Diferenciales con Problemas de Valores de Frontera*, Thomson Learning, 2002).

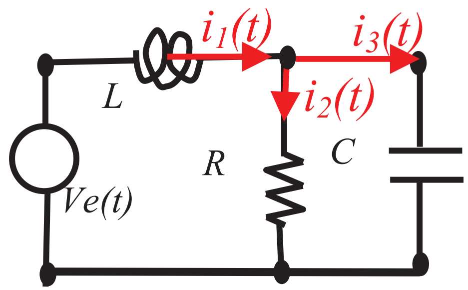
Se trata de circuitos de tipo RL; es decir, con resistencias R, inductancias L, y RC con resistencias y capacitores C. Cuando se consideran como incógnitas las corrientes, se tiene que las ecuaciones resultantes son las siguientes:



$$L_1 \cdot \frac{di_2(t)}{dt} + (R_1 + R_2)\cdot i_2(t) + R_1 \cdot i_3(t) = Ve(t)$$

$$L_2 \cdot \frac{di_3(t)}{dt} + R_1 \cdot i_2(t) + R_1 \cdot i_3(t) = Ve(t)$$

$$i_1(t) = i_2(t) + i_3(t)$$



$$L \cdot \frac{di_1(t)}{dt} + R \cdot i_2(t) = Ve(t)$$

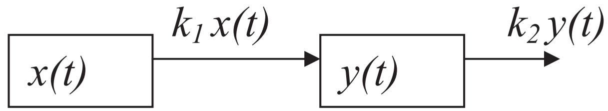
$$RC \cdot \frac{di_2(t)}{dt} + i_2(t) - i_1(t) = 0$$

$$i_1(t) = i_2(t) + i_3(t)$$

SISTEMAS DE EDO-MODELOS DE DISPERSIÓN

Modelo de Dispersión de antihistaminas.
(Borelli, R.; Colleman, C. *Ecuaciones Diferenciales*, Oxford University Press, 2002).
Durante los resfríos es típico ingeniar sustancias para aliviar los malestares. Dichas sustancias, como la antihistaminas, se presentan en el organismo como sustancias a eliminar. En principio las antihistaminas están en el aparato digestivo, y desde allí pasan a la sangre que las circula por todo el cuerpo y son eliminadas de la sangre en los riñones.
Se considera con *x(t)* es la cantidad de antihistamina en el aparato digestivo en el instante de tiempo t; y con *y(t)* se considera la cantidad de antihistamina en la sangre en el mismo instante de tiempo t.
Así un modelo de intercambio de antihistaminas en el organismo está dado por las siguientes dos consideraciones:

- La tasa de cantidad de antihistaminas en el aparato digestivo es proporcional a la cantidad de antihistaminas en el aparato digestivo. Con una cierta cantidad inicial de antihistaminas



$$\frac{dx(t)}{dt} = -k_1 \cdot x(t)$$
$$x(0) = x_0$$

  Al ser una tasa negativa, deja en manifiesto que el proceso es tal que la cantidad decrece.

- La tasa de cantidad de antihistaminas en sangre es proporcional a la cantidad de antihistaminas en sangre y a la cantidad de antihistaminas que ingresan desde el aparato digestivo. Con una cierta cantidad inicial de antihistaminas

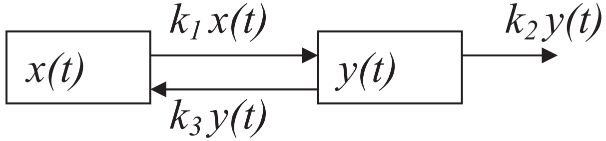
$$\frac{dy(t)}{dt} = -k_2 \cdot y(t) + k1 \cdot x(t)$$
$$y(0) = y_0$$

Otra forma de expresar las ecuaciones diferenciales es en la siguiente forma matricial.

$$\begin{Bmatrix} \dfrac{dx(t)}{dt} \\ \dfrac{dy(t)}{dt} \end{Bmatrix} = \begin{bmatrix} -k_1 & 0 \\ k_1 & -k_2 \end{bmatrix} \begin{Bmatrix} x(t) \\ y(t) \end{Bmatrix} \qquad con \qquad \begin{Bmatrix} x(0) \\ y(0) \end{Bmatrix} = \begin{Bmatrix} x_0 \\ y_0 \end{Bmatrix}$$

Eliminación de contaminante en un líquido

Considérese un proceso por el cual se busca eliminar un contaminante de un liquido que circula entre dos depósitos depuradores. Se trata de dos bloques o sistemas depuradores en los cuales la cantidad de contamínate en el primer bloque es *x(t),* mientras que en el segundo es *y(t)*. En la siguiente figura se muestra el diagrama de bloques del sistema en análisis.



En el primer bloque solo salen $k_1\,x(t)$ cantidad de contaminante, que ingresan en el segundo bloque. Pero también existe una retroalimentación no deseada desde el segundo bloque que hace ingresar al primer bloque $k_3\,y(t)$ contaminante. Desde el segundo bloque salen $k_2\,y(t)$ cantidad de contaminantes residuales. Las cantidades que salen de un bloque se consideran negativas, mientras que las que ingresan, positivas. De esa manera es posible hacer el balance de lo que entra y sale en cada bloque e igualarlo a la tasa de cantidad de sustancia contaminante en el tiempo de dicho bloque.

$$\frac{dx(t)}{dt} = +k_3 \cdot y(t) - k1 \cdot x(t) \qquad \frac{dy(t)}{dt} = -k_3 \cdot y(t) + k1 \cdot x(t) - k_2 \cdot y(t)$$

$$x(0) = x_0 \qquad y(0) = y_0$$

Otra forma de expresar las ecuaciones diferenciales es en la siguiente forma matricial.

$$\begin{Bmatrix} \dfrac{dx(t)}{dt} \\ \dfrac{dy(t)}{dt} \end{Bmatrix} = \begin{bmatrix} -k_1 & k_3 \\ k_1 & -k_2 - k_3 \end{bmatrix} \begin{Bmatrix} x(t) \\ y(t) \end{Bmatrix} \qquad con \qquad \begin{Bmatrix} x(0) \\ y(0) \end{Bmatrix} = \begin{Bmatrix} x_0 \\ y_0 \end{Bmatrix}$$

En este caso se debe resolver las dos incógnitas en forma simultánea con las dos ecuaciones.

Otros problemas descriptos por SISTEMAS de EDO son:

- Flujos entre depósitos;
- Flujos en procesos químicos;
- Evolución de poblaciones y/o especies
- Evolución de "Romeo y Julieta"

SISTEMAS DE ECUACIONES DIFERENCIALES ORDINARIAS DE PRIMER ORDEN

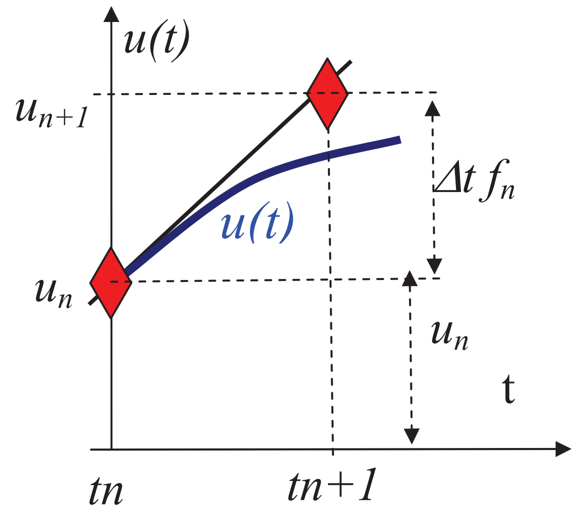
Se busca obtener una solución aproximada en forma discreta del siguiente SISTEMA de EDO

$$\frac{dy_1}{dt} = -10 \cdot y_1(t) + 4 \cdot y_2(t)$$
$$\frac{dy_2}{dt} = -4 \cdot y_1(t) + 0 \cdot y_2(t)$$

, con: $y_1(0) = 5$, $y_2(0) = 3$, que tiene solución exacta $\mathbf{y}(t) = \frac{1}{3}\begin{Bmatrix} 1 \\ 2 \end{Bmatrix} e^{-2*t} + \frac{14}{3}\begin{Bmatrix} 1 \\ 1/2 \end{Bmatrix} e^{-8*t}$

Aplicar el Método de Euler y obtener la siguiente aproximación

t	**0**	0,01	0,02	0,03	0,04	0,05
y1	**5**					
y2	**3**					
k1 _y1	0,01*(-10***5**+4***3**)					
_y2	0,01*(-4***5**+0***3**)					
y1_(n+1)	**5**+(-0,38)					
y2_(n+1)	**3**+(-0,2)					
tn+1	**0**+0,1					



Método de Euler: Dado (t_m, y_m)

$$k_1 = \Delta t \; f\left(t_m, y_m\right)$$
$$y_{m+1} = y_m + k_1$$
$$t_{m+1} = t_m + \Delta t$$

SISTEMAS DE ECUACIONES DIFERENCIALES ORDINARIAS DE PRIMER ORDEN

Se busca obtener una solución aproximada en forma discreta del siguiente SISTEMA de EDO

$$\frac{dy_1}{dt} = -10 \cdot y_1(t) + 4 \cdot y_2(t)$$

$$\frac{dy_2}{dt} = -4 \cdot y_1(t) + 0 \cdot y_2(t)$$

, con: $y_1(0) = 5$, $y_2(0) = 3$,

Aplicar el Método de Euler Modificado (w=1) y obtener la siguiente aproximación

t	**0**	0,01	0,02	0,03	0,04
y1	**5**	4,635	4,2977		
y2	**3**	2,8076	2,6292		
k1 _y1	0,01*(-10*5+4*3)=-0,38				
_y2	0,01*(-4*5+0*3)=-0,2				
tg	0+0,01/2=0,005				
yg1	5 -0,38/2= **4,81**				
yg2	3 -0,2/2= **2,9**				
k2 _y1	0,01*(-10*4,81+4*2,9)= -0,365				
_y2	0,01*(-4*4,81+0*2,9)= -0,1924				
tg	0+0,01				
y1_(n+1)	5 -0,365= **4,635**				
y2_(n+1)	3 -0,1924= **2,8076**				

$$k_1 = \Delta t \; f\left(t_m, y_m\right)$$

$$t_G = t_m + \Delta t/(2\omega)$$

$$y_G = y_m + \frac{1}{2\omega} k_1$$

$$k_2 = \Delta t \; f\left(t_G, y_G\right)$$

$$y_{m+1} = y_m + \left(1-\omega\right) k_1 + \omega\, k_2$$

$$t_{m+1} = t_m + \Delta t$$

```
% Datos
y10=5;     % Valores Iniciales
y20=3;
t0=0;
Dt=0.01;  % incremento de tiempo
NDt=200;  % cantidad de Dt a realizar
% Dimensionamiento
t=zeros(1,NDt);    % vector fila para el tiempo
y=zeros(2,NDt);    % Matriz para el vector solución y(t)
      yg=zeros(2,1);    % Vector columna auxiliar
ya=zeros(2,1);
k1=zeros(2,1);
      k2=zeros(2,1);
% Inicialización del primer estado solución
t(1)=t0;
y(1,1)=y10;
y(2,1)=y20;
```

```
% Euler
for j=1:NDt-1
   ya=y(:,j);
   ta=t(j);

   k1(1,1)= Dt* (-10*ya(1) + 4*ya(2));
   k1(2,1)= Dt* (-4*ya(1)+0*ya(2));

   y(:,j+1)  = ya + k1;
   t(j+1)    = ta + Dt;
end

figure(1)
plot(t,y(1,:),'b');grid on
end
```

```
% Euler Modificado
for j=1:NDt-1
   ya=y(:,j);
   ta=t(j);

   k1(1,1)= Dt* (-10*ya(1) + 4*ya(2));
   k1(2,1)= Dt* (-4*ya(1)+0*ya(2));

   yg    = ya + k1/(2);
   tg    = ta + Dt/(2);

   k2(1,1)= Dt* (-10*yg(1) + 4*yg(2));
   k2(2,1)= Dt* (-4*yg(1)+0*yg(2));

   y(:,j+1)  = ya + k2;
   t(j+1)    = ta + Dt;
end
```

```
for i=1:NDt
    yex(i)=(1/3)*exp(-2*t(i))+(14/3)*exp(-8*t(i));    er(i)=abs(yex(i)-y(1,i));
end
normer=norm(er,inf)
figure(1)
plot(t,y(1,:),'b', t,yex(1,:),'r');grid on
end
```

```
% Datos
y10=5;     % Valores Iniciales
y20=3;
t0=0;
Dt=0.01;  % incremento de tiempo
NDt=200;  % cantidad de Dt a realizar
% Dimensionamiento
t=zeros(1,NDt);    % vector fila para el tiempo
y=zeros(2,NDt);    % Matriz para el vector solución y(t)
      yg=zeros(2,1);     % Vector columna auxiliar
ya=zeros(2,1);
k1=zeros(2,1);
      k2=zeros(2,1);
% Inicialización del primer estado solución
t(1)=t0;
y(1,1)=y10;
y(2,1)=y20;
```

```
% Euler
for j=1:NDt-1
   ya=y(:,j);
   ta=t(j);
   k1=Dt*f_sist_1(ya,ta)
   k1(1,1)= Dt* (-10*ya(1) + 4*ya(2));
   k1(2,1)= Dt* (-4*ya(1)+0*ya(2));
   y(:,j+1)  = ya + k1;
   t(j+1)    = ta + Dt;
end

figure(1)
plot(t,y(1,:),'b');grid on
end
```

```
% Euler Modificado
for j=1:NDt-1
   ya=y(:,j);
   ta=t(j);
   k1=Dt*f_sist_1(ya,ta)
   k1(1,1)= Dt* (-10*ya(1) + 4*ya(2));
   k1(2,1)= Dt* (-4*ya(1)+0*ya(2));
   yg     = ya + k1/(2);
   tg     = ta + Dt/(2);
   k2=Dt*f_sist_1(yg,tg);
   k2(1,1)= Dt* (-10*yg(1) + 4*yg(2));
   k2(2,1)= Dt* (-4*yg(1)+0*yg(2));
   y(:,j+1)   = ya + k2;
   t(j+1)     = ta + Dt;
end
```

```
% Euler y RK
for j=1:NDt-1
   ya=y(:,j);
   ta=t(j);
   k1=Dt*f_sist_1(ya,ta)
   if (w==0)
     k2=zeros(2,1)
   else
     yg     = ya + k1/(2*w);
     tg     = ta + Dt/(2*w);
     k2=Dt*f_sist_1(yg,tg);
   end
   y(:,j+1)   = ya + (1-w)*k1+w* k2;
   t(j+1)     = ta + Dt;
end
```

```
end
function [fy]=f_sist_1(z,x)
  fy(1,1)=(-10*z(1) + 4*z(2));
  fy(2,1)=(-4*z(1)+0*z(2));
end
```

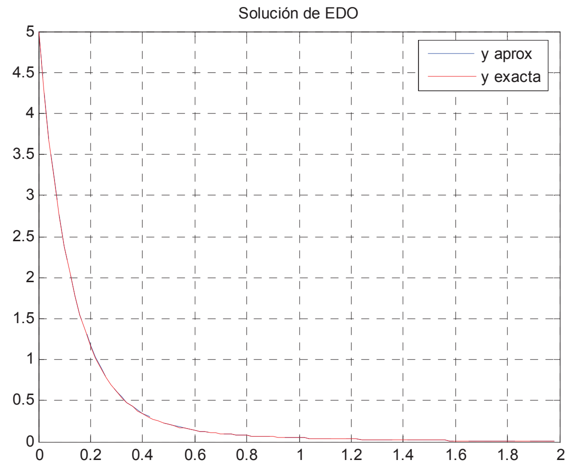
Se busca obtener una solución aproximada en forma discreta del siguiente SISTEMA de EDO

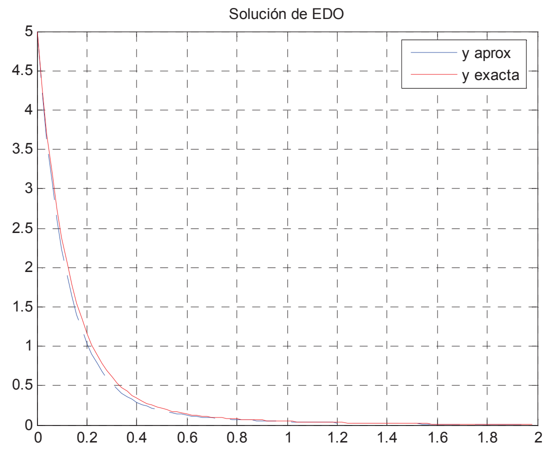
$$\frac{dy_1}{dt} = -10 \cdot y_1(t) + 4 \cdot y_2(t)$$

$$\frac{dy_2}{dt} = -4 \cdot y_1(t) + 0 \cdot y_2(t)$$

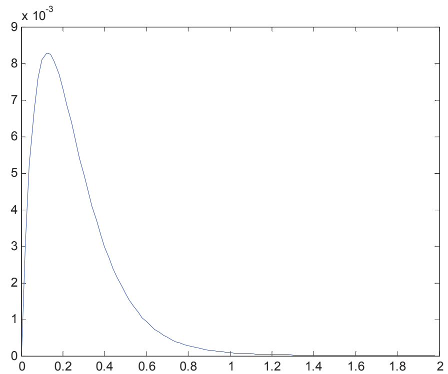
, con: $y_1(0) = 5$, $y_2(0) = 3$, que tiene solución exacta $\mathbf{y}(t) = \frac{1}{3}\begin{Bmatrix} 1 \\ 2 \end{Bmatrix} e^{-2*t} + \frac{14}{3}\begin{Bmatrix} 1 \\ 1/2 \end{Bmatrix} e^{-8*t}$

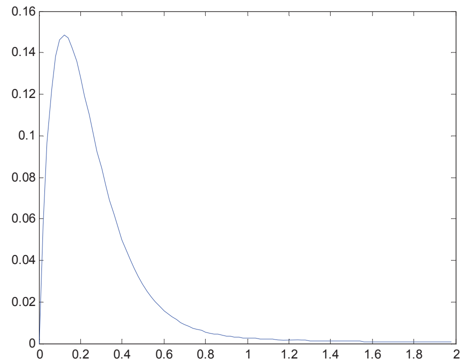
$y_1(t)$ con Dt=0.02 EULER MODIFICADO EULER





Soluciones Aproximadas





Funciones Error

ECUACIONES DIFERENCIALES ORDINARIAS DE SEGUNDO ORDEN

REDUCCIÓN A SISTEMAS DE PRIMER ORDEN

Considérese la EDO de segundo orden del péndulo simple,

$$\frac{d^2\theta(t)}{dt^2}+\frac{g}{L}\theta(t)=0 \qquad \theta(t_0)=\theta_0$$

$$\frac{d\theta(t)}{dt}=\beta_0 \ \ en\ t=t_0$$

donde $\theta(t)$ es la posición angular medida respecto de la vertical; θ_0 la posición inicial y β_0 la velocidad inicial. Si se plantea un cambio de variables tal que

$y_1(t)=\theta(t)$ $\qquad$ $\dot{y}_1(t)=y_2(t)$

$y_2(t)=\dfrac{d\theta(t)}{dt}$ entonces $\dot{y}_2(t)=\dfrac{d^2\theta(t)}{dt^2}$ con lo que la EDO es $\dot{y}_2(t)+\dfrac{g}{L}\theta(t)=0$

Se tiene

$\begin{array}{l}\dot{y}_1(t)=y_2(t)\\ \dot{y}_2(t)=-(g/L)y_1(t)\end{array}$ o bien $\begin{array}{l}\dot{y}_1(t)= \quad 0\cdot y_1(t)\ +\ 1\cdot y_2(t)\\ \dot{y}_2(t)=-(g/L)\cdot y_1(t)\ +\ 0\cdot y_2(t)\end{array}$

con lo que $\vec{y}(t)=\begin{Bmatrix}\theta(t)\\ \dot{\theta}(t)\end{Bmatrix}=\begin{Bmatrix}y_1(t)\\ y_2(t)\end{Bmatrix}$

y $\vec{f}(t,\vec{y})=\begin{Bmatrix}\dot{y}_1(t)\\ \dot{y}_2(t)\end{Bmatrix}=\begin{Bmatrix}0\cdot y_1(t)\ +\ 1\cdot y_2(t)\\ -(g/L)\cdot y_1(t)\ +\ 0\cdot y_2(t)\end{Bmatrix}$ con $\vec{y}(0)=\begin{Bmatrix}\theta(0)\\ \dot{\theta}(0)\end{Bmatrix}=\begin{Bmatrix}y_1(0)\\ y_2(0)\end{Bmatrix}=\begin{Bmatrix}\theta_0\\ \beta_0\end{Bmatrix}$

En un sistema masa resorte los desplazamientos son la solución de la siguiente EDO de segundo orden:
Buscar *u(t)*, para $t \; \varepsilon \; [0;+\infty)$, tal que

$$m\frac{d^2u(t)}{dt^2}+c\cdot\frac{du(t)}{dt}+k\cdot u(t)=p\cdot sen(\Omega\cdot t) \qquad con \quad \begin{Bmatrix} u(0) \\ \dot{u}(0) \end{Bmatrix}=\begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

Si se plantea un cambio de variables tal que

$$y_1(t)=u(t)$$

$$y_2(t)=\frac{du(t)}{dt} \text{ lo que implica que } \qquad \frac{dy_1(t)}{dt}=y_2(t)$$

y la EDO se puede escribir

$$m\frac{dy_2(t)}{dt}+c\cdot y_2(t)+k\cdot y_1(t)=p\cdot sen(\Omega\cdot t) \qquad con \quad \begin{Bmatrix} y_1(0) \\ y_2(t) \end{Bmatrix}=\begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

El problema original se puede reemplazar por
Buscar $y_1(t)$, $y_2(t)$, para $t \; \varepsilon \; [0;+\infty)$, tal que

$$\frac{dy_1(t)}{dt}=y_2(t) \qquad con \;\; y_1(0)=0$$

$$\frac{dy_2(t)}{dt}=(p/m)sen(\Omega\cdot t)-(k/m)\cdot u(t) \qquad y_2(0)=0$$

$$\frac{d}{dt}\begin{Bmatrix} u(t) \\ v(t) \end{Bmatrix}=\begin{Bmatrix} v(t) \\ (p/m)sen(\Omega\cdot t)-(k/m)\cdot u(t) \end{Bmatrix} \quad con \quad \begin{Bmatrix} u(0) \\ v(0) \end{Bmatrix}=\begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

La solución exacta de este problema es

$$\begin{Bmatrix} u(t) \\ v(t) \end{Bmatrix}=\frac{p}{k}\cdot\frac{1}{1-(\Omega/\omega)^2}\cdot\begin{Bmatrix} -(\Omega/\omega)sen(\omega\cdot t)+sen(\Omega\cdot t) \\ \Omega\cdot(-\cos(\omega\cdot t)+\cos(\Omega\cdot t)) \end{Bmatrix} \quad con \quad \omega^2=k/m \qquad t \; \varepsilon \; [0;+\infty)$$

En un sistema la incógnita primaria es la solución de la siguiente EDO de tercer orden:
Buscar *u(t)*, para $t \; \varepsilon \; [0;+\infty)$, tal que

$$a_3 \frac{d^3u(t)}{dt^3} + a_2 \frac{d^2u(t)}{dt^2} + a_1 \cdot \frac{du(t)}{dt} + a_0 \cdot u(t) = b(t) \qquad con \begin{Bmatrix} u(0) \\ \dot{u}(0) \\ \ddot{u}(0) \end{Bmatrix} = \begin{Bmatrix} A \\ B \\ C \end{Bmatrix}$$

Si se plantea un cambio de variables tal que

$$y_1(t) = u(t) \qquad \qquad \frac{dy_1(t)}{dt} = y_2(t)$$

$$y_2(t) = \frac{du(t)}{dt} \quad \text{lo que implica que} \quad \frac{dy_2(t)}{dt} = y_3(t)$$

$$y_3(t) = \frac{d^2u(t)}{dt^2}$$

y la EDO se puede escribir $a_3 \dfrac{dy_3(t)}{dt} + a_2 \cdot y_3(t) + a_1 \cdot y_2(t) + a_0 \cdot y_1(t) = b(t)$ $\quad con \;\; u(0) = A;\; \dot{u}(0) = B;\; \ddot{u}(0) = C$

El problema original se puede reemplazar por:
Buscar $y_1(t)$, $y_2(t)$, $y_3(t)$ para $t \; \varepsilon \; [0;+\infty)$, tal que

$$\frac{dy_1(t)}{dt} = y_2(t)$$

$$\frac{dy_2(t)}{dt} = y_3(t) \qquad con \begin{Bmatrix} y_1(0) \\ y_2(0) \\ y_3(0) \end{Bmatrix} = \begin{Bmatrix} A \\ B \\ C \end{Bmatrix}$$

$$\frac{dy_3(t)}{dt} = -(a_2/a_3) \cdot y_3(t) - (a_1/a_3) \cdot y_2(t) - (a_0/a_3) \cdot y_1(t) + b(t)/a_3$$

Buscar $y_1(t)$, $y_2(t)$, $y_3(t)$ para $t \;\varepsilon\; [0;+\infty)$, tal que

$$\begin{Bmatrix} \dfrac{dy_1(t)}{dt} \\ \dfrac{dy_2(t)}{dt} \\ \dfrac{dy_3(t)}{dt} \end{Bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -(a_0/a_3) & -(a_1/a_3) & -(a_2/a_3) \end{bmatrix} \begin{Bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{Bmatrix} + \begin{Bmatrix} 0 \\ 0 \\ b(t)/a_3 \end{Bmatrix} \quad con \quad \begin{Bmatrix} y_1(0) \\ y_2(0) \\ y_3(0) \end{Bmatrix} = \begin{Bmatrix} A \\ B \\ C \end{Bmatrix}$$

Ejemplo: Pag. 149 de Zill,G. Cullen, M. *Ecuaciones Diferenciales con Problemas de Valores de Frontera*, Thomson Learning, 2002.
Buscar u(t) solución de

$$\frac{d^3u(t)}{dt^3} - 6\cdot\frac{d^2u(t)}{dt^2} + 11\cdot\frac{du(t)}{dt} - 6\cdot u(t) = 3\cdot t \qquad con \;\; u(0) = A\;;\;\; \dot{u}(0) = B \quad \ddot{u}(0) = C$$

La solución exacta es:

$$u_{ex}(t) = c_1\cdot e^{t} + c_2\cdot e^{2t} + c_3\cdot e^{3t} - (11/2) - (1/2)\cdot t$$

SOLUCIÓN NUMÉRCIA DE ECUACIONES DIFERENCIALES CON VALORES INICIALES

Ecuaciones Diferenciales Ordinarias de Primer Orden

- Método de Euler
- Método de Runge Kutta de 2do Orden

Sistemas de Ecuaciones Diferenciales Ordinarias de Primer Orden

Ecuaciones Diferenciales Ordinarias de Segundo Orden

- Reducción a Sistemas de primer orden

Sistemas de Ecuaciones Diferenciales Ordinarias de Segundo Orden

- Método de Diferencia Central
- Reducción a Sistemas de primer orden

Ejemplos

Resumen

ECUACIONES DIFERENCIALES ORDINARIAS DE SEGUNDO ORDEN

REDUCCIÓN A SISTEMAS DE PRIMER ORDEN

En un **sistema la incógnita primaria es** la solución de la siguiente EDO de segundo orden:
Buscar *u(t)*, para $t \; \varepsilon \; [0;+\infty)$, tal que

$$12\frac{d^2u(t)}{dt^2}+6\cdot\frac{du(t)}{dt}+48\cdot u(t)=36\cdot sen(5\cdot t) \qquad con \quad \begin{Bmatrix} u(0) \\ \dot{u}(0) \end{Bmatrix}=\begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

La solución exacta de este problema es

$$\begin{Bmatrix} u(t) \\ v(t) \end{Bmatrix}=\frac{36}{48}\cdot\frac{1}{1-(5/2)^2}\cdot\begin{Bmatrix} -(5/2)sen(2\cdot t)+sen(5\cdot t) \\ 5\cdot(-\cos(2\cdot t)+\cos(5\cdot t)) \end{Bmatrix} \qquad t \; \varepsilon \; [0;+\infty)$$

Reducir a un sistema de EDO de primer orden de la forma:

$$\frac{d\vec{y}(t)}{dt}=\vec{f}(\vec{y}(t),t) \quad con \quad \vec{y}(0)=\vec{y}_0$$

expresar

$$\vec{y}(t)=\begin{Bmatrix} \\ \\ \end{Bmatrix} \qquad \vec{y}(0)=\vec{y}_0=\begin{Bmatrix} \\ \\ \end{Bmatrix} \qquad \vec{f}(\vec{y}(t),t)=\begin{Bmatrix} \\ \\ \end{Bmatrix}$$

y resolver con un método de Runge Kutta de segundo orden. Comparar con la solución exacta.

REDUCCIÓN A SISTEMAS DE PRIMER ORDEN

En un **sistema no lineal** la incógnita primaria es la solución de la siguiente EDO de segundo orden:
Buscar *u(t)*, para $t \; \varepsilon \; [0;+\infty)$, tal que

$$m\frac{d^2u(t)}{dt^2} + c \cdot \frac{du(t)}{dt} + k \cdot (1 + a \cdot u(t)) \cdot u(t) = p \cdot sen(\Omega \cdot t) \qquad con \quad \begin{Bmatrix} u(0) \\ \dot{u}(0) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

Reducir a un sistema de EDO de primer orden de la forma:

$$\frac{d\vec{y}(t)}{dt} = \vec{f}(\vec{y}(t), t) \quad con \quad \vec{y}(0) = \vec{y}_0$$

expresar

$$\vec{y}(t) = \begin{Bmatrix} \quad \end{Bmatrix} \qquad \vec{y}(0) = \vec{y}_0 = \begin{Bmatrix} \quad \end{Bmatrix} \qquad \vec{f}(\vec{y}(t), t) = \begin{Bmatrix} \quad \end{Bmatrix}$$

y resolver con un método de Runge Kutta de segundo orden. Comparar con la solución exacta.

REDUCCIÓN A SISTEMAS DE PRIMER ORDEN

En un **sistema la incógnita primaria** es la solución de la siguiente EDO de tercer orden:
Buscar *u(t)*, para $t\ \varepsilon\ [0;+\infty)$, tal que

$$\frac{d^3u(t)}{dt^3} - 6\cdot\frac{d^2u(t)}{dt^2} + 11\cdot\frac{du(t)}{dt} - 6\cdot u(t) = 3\cdot t \qquad con\quad u(0)=A\ ;\ \ \dot{u}(0)=B \quad \ddot{u}(0)=C$$

La solución exacta de este problema es

$$u_{ex}(t) = c_1\cdot e^t + c_2\cdot e^{2t} + c_3\cdot e^{3t} - (11/2) - (1/2)\cdot t$$

Ejemplo: Pag. 149 de Zill,G. Cullen, M. *Ecuaciones Diferenciales con Problemas de Valores de Frontera*, Thomson Learning, 2002.
Buscar u(t) solución de

Reducir a un sistema de EDO de primer orden de la forma:

$$\frac{d\vec{y}(t)}{dt} = \vec{f}(\vec{y}(t),t) \quad con \quad \vec{y}(0) = \vec{y}_0$$

expresar

$$\vec{y}(t) = \left\{ \qquad \right\} \qquad \vec{y}(0) = \vec{y}_0 = \left\{ \qquad \right\} \quad \vec{f}(\vec{y}(t),t) = \left\{ \qquad \qquad \right\}$$ y

resolver con un método de Runge Kutta de segundo orden, eligiendo Dt y el coeficiente w. Comparar con la solución exacta.

REDUCCIÓN A SISTEMAS DE PRIMER ORDEN

En un **sistema las incógnitas primarias** son la solución del siguiente sistema de EDO de segundo orden:
Buscar $\theta_1(t)$, $\theta_2(t)$ para $t \ \varepsilon \ [0;+\infty)$, tal que

$$16\frac{d^2\theta_1(t)}{dt^2}+4(\theta_2+\theta_1)=5t$$

$$32\frac{d^2\theta_2(t)}{dt^2}+9\theta_1=-3t^2$$

con $\begin{Bmatrix}\theta_1\\ \theta_2\end{Bmatrix}_{t0}=\begin{Bmatrix}\pi/8\\ -\pi/4\end{Bmatrix}$ y $\begin{Bmatrix}\dfrac{d\theta_1}{dt}\\ \dfrac{d\theta_2}{dt}\end{Bmatrix}_{t0}=\begin{Bmatrix}0,1\\ -0,2\end{Bmatrix}$

Reducir a un sistema de EDO de primer orden de la forma:

$$\frac{d\vec{y}(t)}{dt}=\vec{f}(\vec{y}(t),t) \quad con \quad \vec{y}(0)=\vec{y}_0$$

Expresar

$\vec{y}(t)=$ $\qquad$ $\vec{y}(0)=\vec{y}_0=$ $\qquad$ $\vec{f}(\vec{y}(t),t)=$

y resolver con un método de Runge Kutta de segundo orden. Comparar con la solución exacta.

REDUCCIÓN A SISTEMAS DE PRIMER ORDEN

En un **sistema las incógnitas primarias son** la solución del siguiente sistema de EDO de segundo orden: Buscar $x_1(t)$, $x_2(t)$ $x_3(t)$ para $t \ \varepsilon \ [0;+\infty)$, tal que

$$\mathbf{M}\,\ddot{\mathbf{x}}(t)+\mathbf{C}\,\dot{\mathbf{x}}(t)+\mathbf{K}\,\mathbf{x}(t)=\mathbf{R}(t),$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{bmatrix}\begin{Bmatrix} \ddot{x}_1(t) \\ \ddot{x}_2(t) \\ \ddot{x}_3(t) \end{Bmatrix}+\begin{bmatrix} 4 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 3 \end{bmatrix}\begin{Bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{Bmatrix}+\begin{bmatrix} 0 & 4 & 1 \\ 4 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix}\begin{Bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{Bmatrix}=\begin{Bmatrix} 5e^{t}+8e^{2t}+\cos t \\ -8e^{2t}+4\,e^{t} \\ -\cos t-3sent+e^{t} \end{Bmatrix}$$

$$con \quad \begin{Bmatrix} x_1(0) \\ x_2(0) \\ x_3(0) \end{Bmatrix}=\begin{Bmatrix} 1 \\ 2 \\ 1 \end{Bmatrix} \qquad \begin{Bmatrix} \dot{x}_1(0) \\ \dot{x}_2(0) \\ \dot{x}_3(0) \end{Bmatrix}=\begin{Bmatrix} 1 \\ 4 \\ 0 \end{Bmatrix}$$

Reducir a un sistema de EDO de primer orden de la forma:

$$\frac{d\vec{y}(t)}{dt}=\vec{f}(\vec{y}(t),t) \quad con \quad \vec{y}(0)=\vec{y}_0$$

Expresar

$$\vec{y}(t)= \qquad\qquad \vec{y}(0)=\vec{y}_0= \qquad\qquad \vec{f}(\vec{y}(t),t)=$$

y resolver con un método de Runge Kutta de segundo orden. Comparar con la solución exacta.

SISTEMA DE EDO DE PRIMER ORDEN -MATLAB

```
% Datos
Dim=  2            % Dimensión del Sistema de EDO
w= 1               % 0 es Euler; 1 es E-Modificado; ½ es E-Mejorado
y0=[5; 3];     % Valores Iniciales deben ser Dim
t0=0;
Dt=0.01;  % incremento de tiempo
NDt=200;  % cantidad de Dt a realizar
% Dimensionamiento
t=zeros(1,NDt);    % vector fila para el tiempo
y=zeros(Dim,NDt);    % Matriz para el vector solución y(t)
yg=zeros(2,1);    % Vector columna auxiliar
ya=zeros(Dim,1);
k1=zeros(Dim,1);
k2=zeros(2,1);
% Inicialización del primer estado solución
t(1)=t0;
y(:,1)=y0;
% Euler y RK
for j=1:NDt-1
   ya=y(:,j);
   ta=t(j);
   k1=Dt*f_sist_1(ya,ta)
   if (w==0)
     k2=zeros(Dim,1)
   else
     yg    = ya + k1/(2*w);
     tg    = ta + Dt/(2*w);
     k2=Dt*f_sist_1(yg,tg);
   end
   y(:,j+1)  = ya + (1-w)*k1+w* k2;
   t(j+1)    = ta + Dt;
end
end
function [fy]=f_sist_1(z,x)           % deben ser Dim componentes
  fy(1,1)=(-10*z(1) + 4*z(2));
  fy(2,1)=(-4*z(1)+0*z(2));
end
```

ECUACIONES DIFERENCIALES ORDINARIAS DE SEGUNDO ORDEN

MÉTODO DE DIFERENCIA CENTRAL

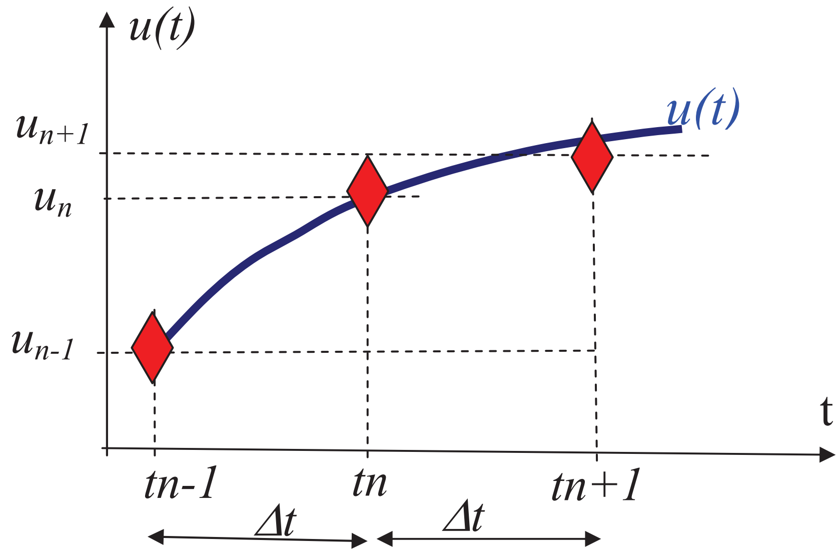
En un **sistema la incógnita primaria es** la solución de la siguiente EDO de segundo orden:
Buscar *u(t)*, para $t \; \varepsilon \; [0;+\infty)$, tal que

$$m\frac{d^2u(t)}{dt^2}+c\cdot\frac{du(t)}{dt}+k\cdot u(t)=p\cdot g(t) \qquad con \quad \begin{Bmatrix} u(0) \\ \dot{u}(0) \end{Bmatrix}=\begin{Bmatrix} \alpha \\ \beta \end{Bmatrix}$$

En base a las reglas de derivadas centrales

$$\left.\frac{d^2u(t)}{dt^2}\right|_{t_n}=\frac{1}{\Delta t^2}(u_{n-1}-2\cdot u_n+u_{n+1})+O(\Delta t^2)$$

$$\left.\frac{du(t)}{dt}\right|_{t_n}=\frac{1}{2\cdot\Delta t^2}(-u_{n-1}+u_{n+1})+O(\Delta t^2)$$



Se puede escribir la EDO en t_n en la forma:

$$\boxed{u_{n+1}= b_g(t_n) + D_E\cdot u_n + H_E\cdot u_{n-1}}$$

Con

$$G_E=\left(m+\frac{\Delta t}{2}c\right)^{-1}; \quad D_E=G_E\cdot(2m-\Delta t^2k); \quad H_E=G_E\cdot\left(\frac{\Delta t}{2}c-m\right); \quad b_g(t)=\Delta t^2\cdot G_E\cdot p\cdot g(t)$$

Es necesario conocer dos estados solución (u_{n-1}; u_n) para calcular el nuevo estado u_{n+1} en t_{n+1}.

EJEMPLO

Buscar con el **método de diferencia central** la incógnita *u(t)*, para $t \; \varepsilon \; [0;+\infty)$, tal que

$$65\frac{d^2u(t)}{dt^2}+400\cdot u(t)=80\cdot sen(5\cdot t) \qquad con \quad \begin{Bmatrix} u(0) \\ \dot{u}(0) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

La solución exacta de este problema es

$$u(t)=\frac{80}{400}\cdot\frac{1}{1-(5/2.48)^2}\cdot\left\{-(5/2.48)sen(2.48\cdot t)+sen(5\cdot t)\right\}$$

MÉTODO DE DIFERENCIA CENTRAL

Para un sistema EDO que en forma generalizada se puede escribir

$$\mathbf{M}\,\ddot{\mathbf{u}}(t)+\mathbf{C}\dot{\mathbf{u}}(t)+\mathbf{K}\,\mathbf{u}(t)=\mathbf{R}(t),$$

con valores iniciales conocidos $\mathbf{u}(0), \dot{\mathbf{u}}(0),$ y con $\ddot{\mathbf{u}}(0)$ que se puede obtener de satisfacer la ecuación diferencial vectorial en el instante inicial t=0.

Se aproximan la $\ddot{\mathbf{u}}(t)$ y la $\dot{\mathbf{u}}(t)$ con derivadas numéricas centrales y se reemplazan en el sistema original.

$$\mathbf{M}\frac{1}{\Delta t^2}\big(\mathbf{u}(t-\Delta t)-2\mathbf{u}(t)+\mathbf{u}(t+\Delta t)\big)+\mathbf{C}\frac{1}{2\Delta t}\big(-\mathbf{u}(t-\Delta t)+\mathbf{u}(t+\Delta t)\big)+\mathbf{K}\mathbf{u}(t)=\mathbf{R}(t)$$

Es posible aproximar $\mathbf{u}(t+\Delta t)$, para cualquier $t \geq t_0$ en la forma:

$$\boxed{\mathbf{u}(t+\Delta t)= \quad \mathbf{b}_g(t) \;+\mathbf{D}_E(\Delta t)\cdot\mathbf{u}(t) \;+\mathbf{H}_E(\Delta t)\mathbf{u}(t-\Delta t)}$$

con

$$\mathbf{G}_E=\left(\mathbf{M}+\frac{\Delta t}{2}\mathbf{C}\right)^{-1}; \quad \mathbf{D}_E=\mathbf{G}_E\cdot\left(2\mathbf{M}-\Delta t^2\mathbf{K}\right); \quad \mathbf{H}_E=\mathbf{G}_E\cdot\left(\frac{\Delta t}{2}\mathbf{C}-\mathbf{M}\right); \quad \mathbf{b}_g(t)=\Delta t^2\cdot\mathbf{G}_E\cdot\mathbf{R}(t)$$

Se debe destacar que las matrices $\mathbf{G_E}$ y $\mathbf{H_E}$, como así también la matriz inversa que las define, se calcula sólo una vez al inicio del método.

Para el valor inicial de t (t=t0), primero se halla una aproximación de $\mathbf{u}(t_0-\Delta t)$ mediante Serie de Taylor:

$$\mathbf{u}(t_0-\Delta t)=\left(\mathbf{u}(t_0)-\Delta t\,\dot{\mathbf{u}}(t_0)+\frac{1}{2}\Delta t^2\,\ddot{\mathbf{u}}(t_0)\right).$$

EJEMPLO

Buscar **con el método de diferencia central** las incógnitas $x_1(t)$, $x_2(t)$ $x_3(t)$ para $t \ \varepsilon \ [0;+\infty)$, tal que

$$\mathbf{M}\,\ddot{\mathbf{x}}(t)+\mathbf{C}\,\dot{\mathbf{x}}(t)+\mathbf{K}\,\mathbf{x}(t)=\mathbf{R}(t),$$

$$\begin{bmatrix}1 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 2\end{bmatrix}\begin{Bmatrix}\ddot{x}_1(t)\\ \ddot{x}_2(t)\\ \ddot{x}_3(t)\end{Bmatrix}+\begin{bmatrix}4 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 3\end{bmatrix}\begin{Bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \dot{x}_3(t)\end{Bmatrix}+\begin{bmatrix}0 & 4 & 1\\ 4 & 2 & 0\\ 1 & 0 & 1\end{bmatrix}\begin{Bmatrix}x_1(t)\\ x_2(t)\\ x_3(t)\end{Bmatrix}=\begin{Bmatrix}5e^t+8e^{2t}+\cos t\\ -8e^{2t}+4\,e^t\\ -\cos t-3sent+e^t\end{Bmatrix}$$

$$con \quad \begin{Bmatrix}x_1(0)\\ x_2(0)\\ x_3(0)\end{Bmatrix}=\begin{Bmatrix}1\\ 2\\ 1\end{Bmatrix} \qquad \begin{Bmatrix}\dot{x}_1(0)\\ \dot{x}_2(0)\\ \dot{x}_3(0)\end{Bmatrix}=\begin{Bmatrix}1\\ 4\\ 0\end{Bmatrix}$$

```
function Dif_cen
clc,clear
 % Datos
Dt=0.003;  % incremento de tiempo
NDt=2500;  % cantidad de Dt a realizar
dim=3;     % cantidad de funciones incógnitas

% Dimensionamiento
%t=zeros(1,NDt);      % vector fila para guardar el tiempo
%y=zeros(dim,NDt);    % Matriz para guardar el vector solución y(t)
%yan=zeros(dim,1);    % Vector de solución anterior
%yac=zeros(dim,1);    % Vector de solución actual
%ynu=zeros(dim,1);    % Vector de solución nueva

M= [1  0  0
    0  -1 0
    0  0  2];
C= [4  0  0
    0  -1 0
    0  0  3];
K= [0  4  1
    4  2  0
    1  0  1];

% Valores Iniciales o actuales
tac=0;
yac= [1;  2;   1]
vac= [1;  4;   0]
fua(1,1)=5*exp(tac)+8*exp(2*tac)+cos(tac);
fua(2,1)=-8*exp(2*tac)+4*exp(tac);
fua(3,1)=-cos(tac)-3*sin(tac)+exp(tac);

% Inicialización
G=inv(M+(Dt/2)*C)
D=G*(2*M-Dt^2*K)
H=G*((Dt/2)*C-M)

yan=yac-Dt*vac+((Dt^2)/2)*inv(M)*(fua-K*yac-C*vac);

t(1)=tac;            % Almacenamiento para luego graficar
y(:,1)=yac;
```

```
% Diferencia Central
for j=2:NDt
  bac=fun_ind(tac,G,Dt); % actualización de término independiente

  ynu=bac + D*yac + H*yan;   % Calculo con Dif Central
  tnu=tac+Dt;
  t(j)=tnu;             % Almacenamiento para luego graficar
  y(:,j)=ynu;

  yan=yac;              % actualización de estado anterior
  yac=ynu;              % actualización de estado actual
  tac=tnu;
end
end

function [fy]=fun_ind(x,G,Dt)
  r(1,1)= 5*exp(x)  +8*exp(2*x) +cos(x);
  r(2,1)=-8*exp(2*x)+4*exp(x);
  r(3,1)=-cos(x)    -3*sin(x) +exp(x);
  fy=Dt^2*G*r;
end
```