FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 1

[75.12] Análisis Numérico I
Segundo cuatrimestre de 2018

Alumno	Padrón	Mail
del Mazo, Federico	100029	delmazofederico@gmail.com
Kristal, Juan Ignacio	99779	kristaljuanignacio@gmail.com

Curso 07:

- Dr Daniel Fabian Rodriguez
- Valeria Machiunas
- Federico Balzarotti
- Michael Portocarrero

ANÁLISIS NUMÉRICO I - 75.12 – 95.04

Curso: Rodríguez- Balzarotti - Machiunas- Portocarrero

2º cuatrimestre de 2018

TRABAJO PRÁCTICO DE MÁQUINA Nº 1

Desarrollo del práctico:

1) Programar un algoritmo para estimar la unidad de máquina (μ), en simple y en doble precisión.
2) Implementar el método de trapecios compuestos para evaluar la integral:

a) $$F(\alpha,\beta)=\int_{1}^{240}\frac{\sin(Px)+\beta x^2}{\alpha\ x}dx$$

donde P = (N° de padrón de integrante 1 + N° de padrón de integrante 2) /50

o bien P = Nº de padrón / 25

$\alpha=0.17$ y $\beta=0.41$

de tal forma que el módulo del error absoluto de truncamiento sea menor que 10^{-5}.

Informar qué valor de n (cantidad de trapecios) se ha utilizado, justificando la elección.

Considere P exacto, y α y β bien redondeados.

3) Fijado dicho valor de n, luego:

I. Calcular la condición del problema mediante la técnica de perturbaciones experimentales.
II. Estimar experimentalmente el término de estabilidad.
III. Utilizando los resultados anteriores que sean necesarios, y suponiendo nulo el error inherente, acotar el error total.
IV. Repetir III suponiendo que el error inherente relativo está acotado por $0.5\ 10^{-4}$
V. Indicar la fuente más importante de error en los dos casos anteriores.

La entrega del presente trabajo práctico deberá realizarse de acuerdo al reglamento del curso, en la fecha informada en clase.

Índice

1. Introducción

El trabajo práctico tiene como objetivo el cálculo y acotamiento de errores de la siguiente integral:

$$F(\alpha, \beta) = \int_{1}^{240} \frac{\sin(Px) + \beta x^2}{\alpha x} dx$$

Siendo:

- $P = \frac{\sum_{padrones}}{50} = \frac{99779+100029}{50} = 3996{,}16$ exacto
- $\alpha = 0{,}17$ bien redondeando
- $\beta = 0{,}41$ bien redondeando

Específicamente:

- Se estimará el valor de la unidad de maquina μ.
- Se evaluará la integral con el método de trapecios compuestos teniendo con objetivo en mente que el error absoluto de truncamiento sea menor a 1×10^{-5}.
- Se calculará computacionalmente la cantidad de trapecios utilizada en el método descrito anteriormente.
- Se calculará la condición del problema mediante perturbaciones experimentales.
- Se estimará experimentalmente el término de estabilidad.
- Se acotará el error total.

2. Desarrollo

2.1. Estimación de μ

Para los cálculos del μ se utilizó el algoritmo del ejemplo 6.4 del libro de Hernan Gonzales [1].

2.2. Función y sus derivadas

Siempre teniendo en cuenta los valores de P, α, β previamente utilizados, definimos la funcion $f(x)$ como:

$$f(x) = \frac{\sin(Px) + \beta x^2}{\alpha x} dx$$

Graficamos la función para saber un poco más de ella en la figura 1

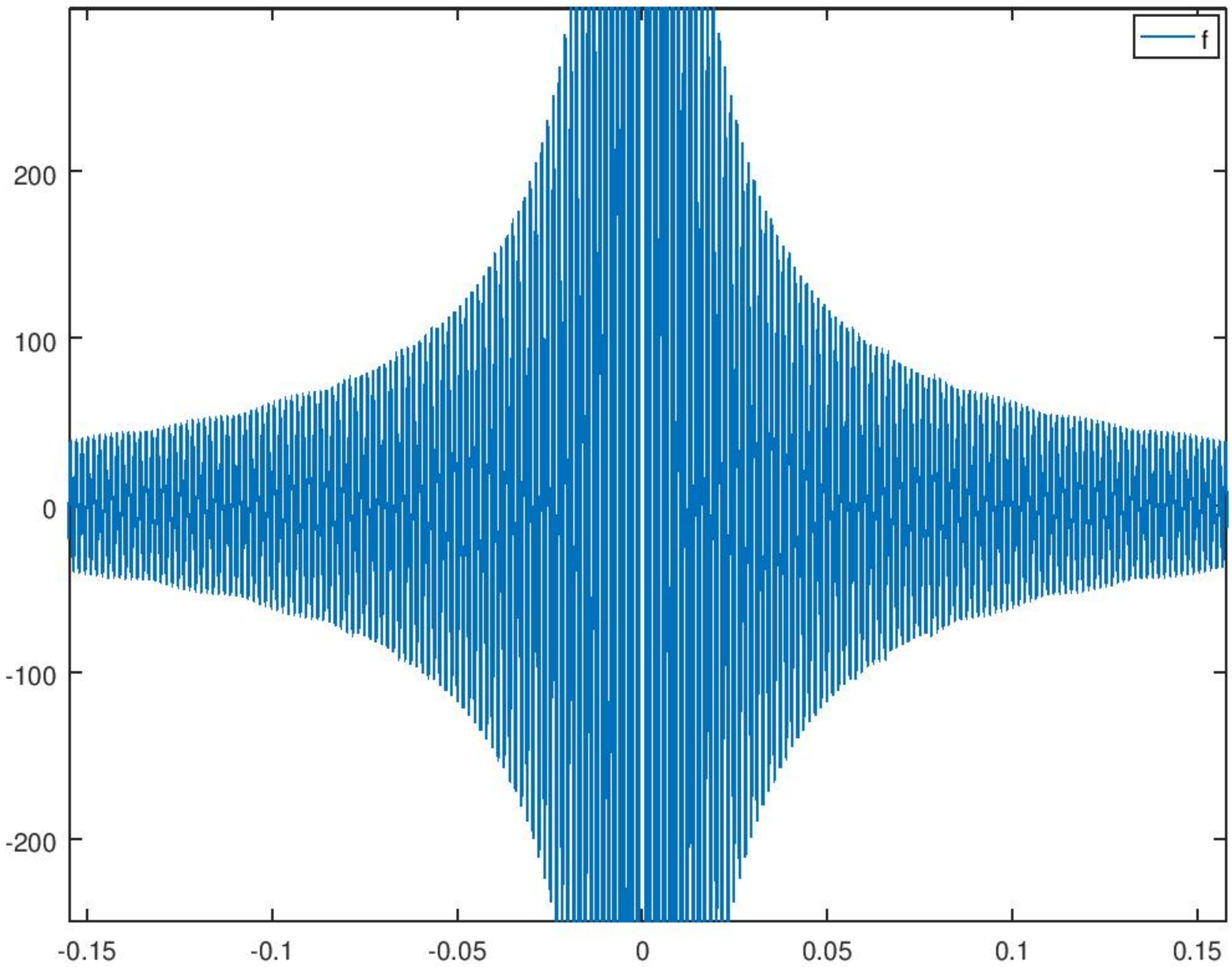

Figura 1: $f(x)$

De esta función calculamos sus derivadas y las graficamos en las figuras 2 y 3, para utilizar en cálculos posteriores.

$$f'(x) = \frac{P\cos(Px)}{\alpha x} - \frac{\sin(Px)}{\alpha x^2} + \frac{\beta}{\alpha}$$

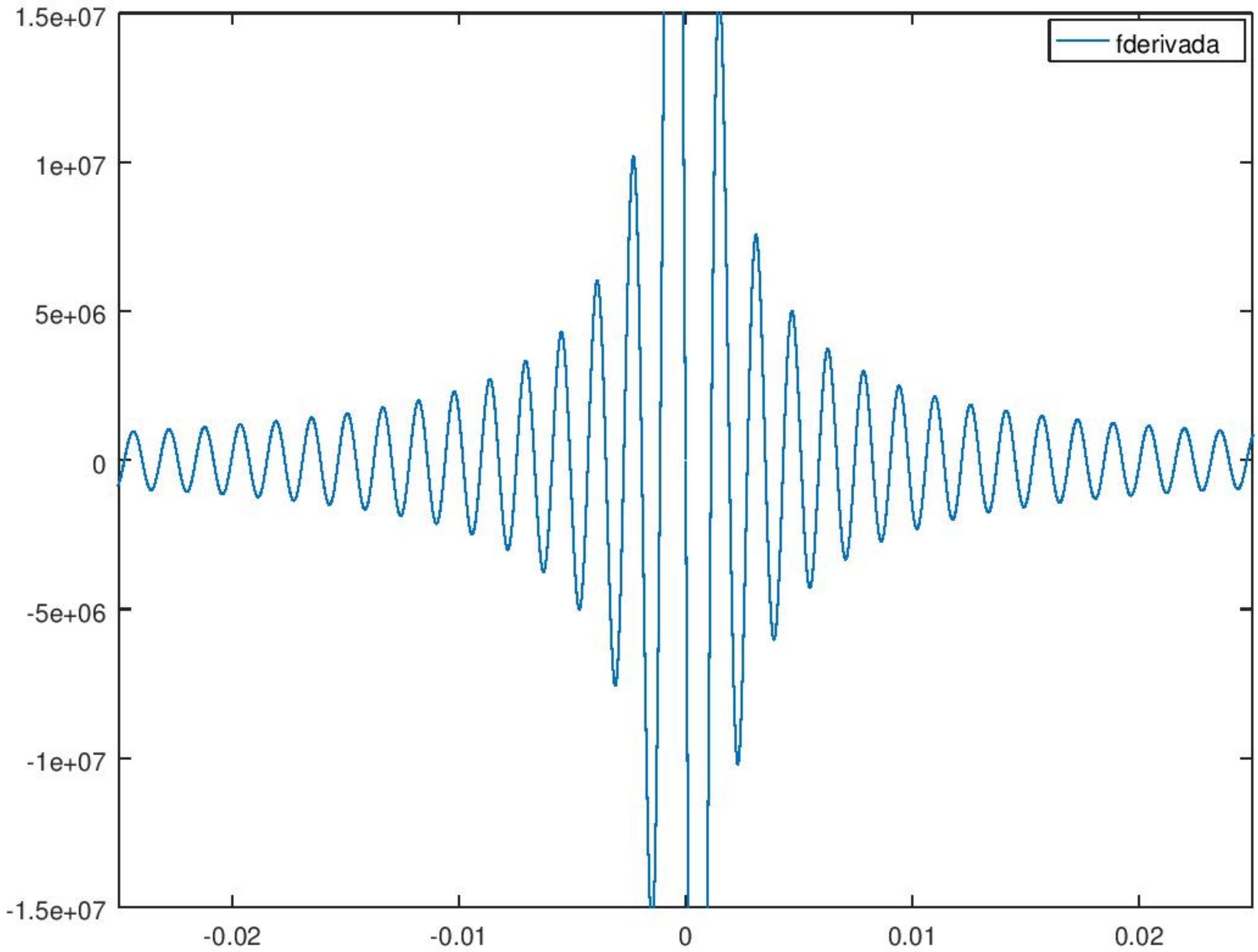

Figura 2: $f'(x)$

$$f''(x) = -\frac{2P\cos(Px)}{\alpha x^2} + \frac{2\sin(Px)}{\alpha x^3} - \frac{P^2\sin(Px)}{\alpha x}$$

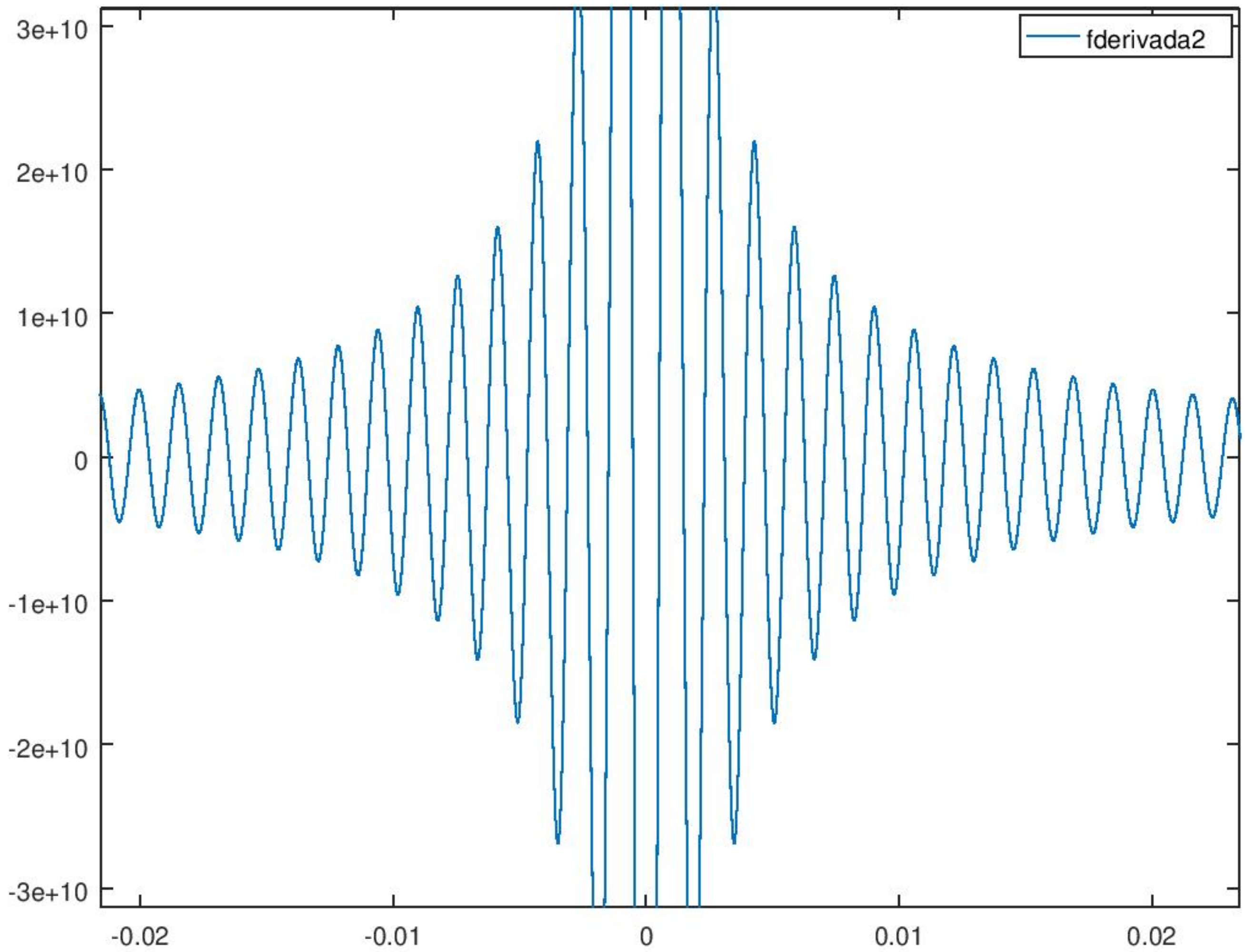

Figura 3: $f''(x)$

2.3. Método de trapecios compuestos

Sabemos que el error de truncamiento producido por el método de trapecios compuestos es:

$$\epsilon_t = -\frac{(b-a)^3}{12n^2} * f''(\xi)$$

Donde b, a son los limites de integración y n es la cantidad de trapecios. Como lo que queremos es acotar el error de truncamiento, debemos evaluar a la segunda derivada en su imagen máxima, es decir $\xi = 1$

Por lo tanto, y ahora con el error de truncamiento al que queremos llegar, despejamos la cantidad de trapecios:

$$n = \sqrt{\left| -\frac{(b-a)^3 * f''(1)}{12\epsilon_t} \right|}$$

2.4. Condición del problema y término de estabilidad

3. Resultados

4. Conclusiones

A. Anexo I: Código Fuente

mu.m

```
function mu
  mu_simple
  mu_doble
end

function mu_doble
  mu_doble=1; digitos=1; x=2;
  while (x>1)
    digitos = digitos+1;
    mu_doble = mu_doble/10;
    x = 1+mu_doble;
  endwhile
  mu_doble
end

function mu_simple
  mu_simple=single(1); digitos=single(1); x=single(2);
  while (x>1)
    digitos = digitos+1;
    mu_simple = mu_simple/10;
    x = 1+mu_simple;
  endwhile
  mu_simple
end
```

main.m

```
function integral = main
  padron1 = 100029; padron2 = 99779;
  global P = (padron1 + padron2) / 50
  global ALPHA = 0.17; global BETA = 0.41;
  global ERR_MAX = 10e-5;
  global LIM_INF = 1; global LIM_SUP = 240;
  n = calcular_n(ERR_MAX)
  integral = calcular_todas_las_areas(n)
end

function y = f(x)
  global P ALPHA BETA

  y = ( sin(x.*P) + BETA * (x.^2) ) ./ (x.*ALPHA);
end

function y = fderivada(x)
  global P ALPHA BETA

  primer_term = (P./(ALPHA.*x)) .* cos(P.*x);
  segundo_term = - ( ( sin(P.*x) ) ./ ( ALPHA .* (x.^2) ) );
  tercer_term = BETA / ALPHA;
  y = primer_term + segundo_term + tercer_term;
end

function y = fderivada2(x)
  global P ALPHA BETA

  primer_term = - (2.*P.*cos(P.*x) ) ./ (ALPHA .* (x.^2) );
  segundo_term = 2* sin(P.*x) ./ (ALPHA .* (x.^3) );
  tercer_term = - ( ( (P^2)*sin(P.*x) ) ./ (ALPHA .* x) );
  y = primer_term + segundo_term + tercer_term;
end

function n = calcular_n(error_maximo_truncamiento)
  global LIM_SUP LIM_INF
  num = - ( (LIM_SUP - LIM_INF)^3 ) * fderivada2(1);
  denom = error_maximo_truncamiento * 12;
  n = sqrt(abs(num/denom));
end

function a = calcular_todas_las_areas(n)
```

```
  global LIM_INF LIM_SUP;
  resultado = 0;
  base = ( LIM_SUP - LIM_INF ) / n;
  ini = LIM_INF;
  fin = LIM_INF + base;
  for i = 0:n;
    resultado += calcular_area_trapecio(base,ini,fin);
    ini+=base;
    fin+=base;
  end
  a = resultado;
end

function a_trap = calcular_area_trapecio(base,ini,fin)
    if (f(ini)<f(fin)); altura_rectangulo = (f(ini)); else; altura_rectangulo = (f(
       ↪ fin)); end;
    if (f(ini)>f(fin)); altura_triangulo = (f(ini)); else; altura_triangulo = (f(fin
       ↪ )); end;
    area_rectangulo = base * altura_rectangulo;
    area_triangulo = (base * altura_triangulo) / 2;
    a_trap = area_rectangulo + area_triangulo;
end
```

B. Anexo II: Resultados Numéricos

Resultados de `mu.m`

```
>> mu
mu_simple = 1.0000e-08
mu_doble = 1.0000e-16
```

Bibliografía

[1] Gonzales, Hernan: *Análisis Numérico, Primer Curso* Buenos Aires: Nueva Librería, 2002.