Análisis Matemático I

Tema 1

Software matemático aplicado a las funciones de variable real

Autor: Víctor Gayoso Martínez

Curso: 2024-2025

Versión: 1.0

Centro Universitario U-tad

Doble Grado en Ingeniería del Software y Matemática Computacional

Índice

1 Introducción

El objetivo de este tema es recoger los comandos que pueden emplearse para el cálculo de límites, derivadas, interpolación polinómica e integración con los siguientes programas/motores de cálculo matemático: Mathematica/WolframAlpha, GeoGebra, Maxima y Sagemath. Adicionalmente, se incluyen los comandos a utilizar con el lenguaje de programación Python.

Es posible utilizar todas las herramientas presentadas en este tema sin necesidad de instalar ningún software, ya que existen diversas páginas web donde el código puede ejecutarse en remoto. A continuación se incluyen algunas de esas páginas:

- Mathematica/WolframAlpha: `https://www.wolframalpha.com`
- GeoGebra: `https://www.geogebra.org/classic`
- Maxima: `http://www.dma.ufv.br/maxima/`
- SageMath: `https://sagecell.sagemath.org/`
- Python: `https://colab.research.google.com` o bien `https://jupyter.org/try` (seleccionar Jupyter Notebook)

El código Python incluido en este documento está basado en el repositorio de Javier García Algarra "PythonMatematicas" disponible en `https://github.com/jgalgarra/PythonMatematicas`.

2 Límites

2.1 Mathematica/WolframAlpha

- $\lim_{x\to 0} \frac{\sin(x)}{x}$: `Limit[Sin[x]/x, x -> 0]`
- $\lim_{x\to \pi} x^2 + \cos(x)$: `Limit[x^2 + Cos[x], x -> Pi]`
- $\lim_{x\to \infty} (1 + 1/x)^x$: `Limit[(1 + 1/x)^x, x -> Infinity]`
- $\lim_{x\to 0} \frac{|x|}{x}$: `Limit[Abs[x]/x, x -> 0]`
- $\lim_{x\to 0^-} \frac{|x|}{x}$: `Limit[Abs[x]/x, x -> 0, Direction -> 1]`
- $\lim_{x\to 0^+} \frac{|x|}{x}$: `Limit[Abs[x]/x, x -> 0, Direction -> -1]`

2.2 GeoGebra

- $\lim_{x\to 0} \frac{\sin(x)}{x}$: `Limit(sin(x)/x, 0)`
- $\lim_{x\to \pi} x^2 + \cos(x)$: `Limit(x^2 + cos(x), pi)`
- $\lim_{x\to \infty} (1 + 1/x)^x$: `Limit((1 + 1/x)^x, infinity)`
- $\lim_{x\to 0} \frac{|x|}{x}$: `Limit(abs(x)/x, 0)`
- $\lim_{x\to 0^-} \frac{|x|}{x}$: `LimitBelow(abs(x)/x,0)`
- $\lim_{x\to 0^+} \frac{|x|}{x}$: `LimitAbove(abs(x)/x,0)`

2.3 Maxima

- $\lim_{x\to 0} \frac{\sin(x)}{x}$: `limit(sin(x)/x,x,0);`
- $\lim_{x\to \pi} x^2+\cos(x)$: `limit(x^2 + cos(x),x,%pi);` o `limit(x^2 + cos(x),x,%pi),numer;`
- $\lim_{x\to \infty} (1+1/x)^x$: `limit((1+1/x)^x,x,infinity);`
- $\lim_{x\to 0} \frac{|x|}{x}$: `limit(abs(x)/x,x,0);`
- $\lim_{x\to 0^-} \frac{|x|}{x}$: `limit(abs(x)/x,x,0,minus);`
- $\lim_{x\to 0^+} \frac{|x|}{x}$: `limit(abs(x)/x,x,0,plus);`

2.4 SageMath

- $\lim_{x\to \pi} x^2+\cos(x)$: `limit(sin(x)/x,x=0)`
- $\lim_{x\to \pi} x^2+\cos(x)$: `limit(x^2 + cos(x),x=pi)` o `limit(x^2 + cos(x),x=pi).n()`
- $\lim_{x\to \infty} (1+1/x)^x$: `limit((1+1/x)^(x),x=infinity)`
- $\lim_{x\to 0} \frac{|x|}{x}$: `limit(abs(x)/x,x=0)`
- $\lim_{x\to 0^-} \frac{|x|}{x}$: `limit(abs(x)/x,x=0,dir='minus')`
- $\lim_{x\to 0^+} \frac{|x|}{x}$: `limit(abs(x)/x,x=0,dir='plus')`

2.5 Python

```
import numpy as np
import sympy as sp

x, expresion = sp.symbols('x expresion')

expresion = sp.sin(x)/x
print("f(x)=",expresion)
sp.pprint(expresion)
limite = sp.limit(expresion,x,0)
print("El límite cuando f(x) tiende a 0 es",limite)

expresion = x**2 + sp.cos(x)
print("f(x)=",expresion)
sp.pprint(expresion)
limite = sp.limit(expresion,x,sp.pi)
print("El límite cuando f(x) tiende a pi es",limite)
print("El límite cuando f(x) tiende a pi es",limite.n())

expresion = (1+1/x)**x
print("f(x)=",expresion)
sp.pprint(expresion)
limite = sp.limit(expresion,x,sp.oo)
print("El límite cuando f(x) tiende a infinito es",limite)
print("El límite cuando f(x) tiende a infinito es",limite.n())

expresion = abs(x)/x
print("f(x)=",expresion)
sp.pprint(expresion)
limite = sp.limit(expresion,x,0)
print("El límite cuando f(x) tiende a cero es",limite)
```

```
limite_minus = sp.limit(expresion,x,0,'-')
print("El límite cuando x tiende a 0 por la izquierda es",limite_minus)

limite_plus = sp.limit(expresion,x,0,'+')
print("El límite cuando x tiende a 0 por la derecha es",limite_plus)
```

3 Derivación

3.1 Mathematica/WolframAlpha

- Derivada de $f(x) = -3x + x^2\cos(x)$: `D[-3x+x^2Cos[x],x]`
- Derivada de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `ReplaceAll[D[-3x+x^2Cos[x],x->Pi]`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$: `D[-3 x + x^2Cos[x],x,2]`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `ReplaceAll[D[-3x+x^2Cos[x],x,2],x->Pi]`
- Representación gráfica de $f(x) = -x^3 + 6x + 2$: `Plot[-x^3+6x+2]`
- Representación gráfica de $f(x) = -x^3 + 6x + 2$ en $[-5,5]$: `Plot[-x^3+6x+2,x,-5,5]`
- Mínimos de $f(x) = -x^3 + 6x + 2$: `Minimize[-x^3+6x+2,x]`
- Máximos de $f(x) = -x^3 + 6x + 2$: `Maximize[-x^3+6x+2,x]`
- Puntos de inflexión de $f(x) = -x^3 + 6x + 2$: `InflectionPoint[-x^3+6x+2]`
- Asíntotas de $f(x) = \frac{1}{x}$: `Asymptote[1/x]`

3.2 GeoGebra

- Derivada de $f(x) = -3x + x^2\cos(x)$: `Derivative(x^2*cos(x)-3x,1)`
- Derivada de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `Iteration(Derivative(x^2*cos(x)-3x),pi,1)`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$: `Derivative(-3 x + x^2Cos[x],x,2)`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `Iteration(Derivative(x^2*cos(x)-3x,2),pi,1)`
  También se pueden obtener las derivadas anteriores mediante esta cadena de comandos:
  ```
  > f(x)=x^2*cos(x)-3x
  > f'
  > f'(pi)
  > f"
  > f"(pi)
  ```
- Representación gráfica de $f(x) = -x^3 + 6x + 2$: `-x^3+6x+2`
- Representación gráfica de $f(x) = -x^3 + 6x + 2$ en $[-5,5]$: `If(-5<=x<=5,-x^3+6x+2)`
- Maximos y mínimos de $f(x) = -x^3 + 6x + 2$: `Extremum(-x^3 + 6x + 2)`
- Puntos de inflexión de $f(x) = -x^3 + 6x + 2$: `InflectionPoint(-x^3+6x+2)`
- Asíntotas de $f(x) = \frac{1}{x}$: `Asymptote(1/x)`

3.3 Maxima

- Derivada de $f(x) = -3x + x^2\cos(x)$: `diff(-3*x+(x^2)*cos(x),x,1);`
- Derivada de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `at(diff(-3*x+(x^2)*cos(x),x,1),[x=%pi]),numer;`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$: `diff(-3*x+(x^2)*cos(x),x,2);`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `at(diff(-3*x+(x^2)*cos(x),x,2),[x=%pi]),numer;`
- Representación gráfica de $f(x) = -x^3 + 6x + 2$:
  `draw2d(explicit(-x^3+6*x+2,x,-1e+06,1e+06));`
- Representación gráfica de $f(x) = -x^3 + 6x + 2$ en $[-5, 5]$:
  `draw2d(explicit(-x^3+6*x+2,x,-5,5));`

3.4 SageMath

- Derivada de $f(x) = -3x + x^2\cos(x)$: `diff(-3*x+(x^2)*cos(x),x)`
- Derivada de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `numerical_approx(diff(-3*x+(x^2)*cos(x),x)(x=pi))`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$: `diff(-3*x+(x^2)*cos(x),x,2)`
- Derivada segunda de $f(x) = -3x + x^2\cos(x)$ en $x = \pi$:
  `numerical_approx(diff(-3*x+(x^2)*cos(x),x,2)(x=pi))`
- Representación gráfica de $f(x) = -x^3 + 6x + 2$ desde $x = -5$ hasta $x = 5$:
  `plot(-x^3+6*x+2,(x,-5,5))`
- Mínimo de $f(x) = -x^3 + 6x + 2$ en el intervalo $(-5, 5)$:
  `find_local_minimum(-x^3+6*x+2,-5,5)`
- Maximo de $f(x) = -x^3 + 6x + 2$ en el intervalo $(-5, 5)$:
  `find_local_maximum(-x^3+6*x+2,-5,5)`

3.5 Python

```python
import numpy as np
import sympy as sp
import matplotlib.pyplot as plt

x, expresion = sp.symbols('x expresion')

expresion = -3*x+(x**2)*sp.cos(x)
print("f(x)=",expresion)
sp.pprint(expresion)
deriv = sp.diff(expresion,x,1)
print("f'(x)=",deriv)

print("f'(pi)=",(deriv.subs(x,sp.pi)).evalf(3))

deriv=sp.diff(expresion,x,2)
print("f''(x)=",deriv)

print("f''(pi)=",(deriv.subs(x,sp.pi)).evalf(3))

t = np.linspace(-5,5,1000)
expresion = -x**3+6*x+2
f = sp.lambdify(x, expresion, "numpy")
plt.plot(t,f(t),label="f(t)")
```

4 Interpolación de funciones

4.1 Mathematica/WolframAlpha

- Polinomio de Taylor de segundo orden de $f(x) = \cos(x)$ en $c = 0$: `Series[Cos[x],x,0,2]`
- Polinomio de Taylor de cuarto orden de $f(x) = \cos(x)$ en $c = \pi/2$: `Series[Cos[x], x, Pi/2, 4]`
- Polinomio interpolador de Lagrange que pasa por los puntos $(-1,3)$, $(2,1)$ y $(3,2)$: `InterpolatingPolynomial[-1,3,2,1,3,2,x]`

4.2 GeoGebra

- Polinomio de Taylor de segundo orden de $f(x) = \cos(x)$ en $c = 0$: `TaylorPolynomial(cos(x), 0, 2)`
- Polinomio de Taylor de cuarto orden de $f(x) = \cos(x)$ en $c = \pi/2$: `TaylorPolynomial(cos(x), pi/2, 4)`
- Polinomio interpolador de Lagrange que pasa por los puntos $(-1,3)$, $(2,1)$ y $(3,2)$: `Polynomial((-1,3),(2,1),(3,2))`

4.3 Maxima

- Polinomio de Taylor de segundo orden de $f(x) = \cos(x)$ en $c = 0$: `taylor(cos(x),x,0,2);`
- Polinomio de Taylor de cuarto orden de $f(x) = \cos(x)$ en $c = \pi/2$: `taylor(cos(x),x,%pi/2,4);`
- Polinomio interpolador de Lagrange que pasa por los puntos $(-1,3)$, $(2,1)$ y $(3,2)$: `expand(lagrange([[-1,3],[2,1],[3,2]]));`

  Nota: En este último caso, es necesario primero ejecutar el siguiente comando: `load(interpol)$`

4.4 SageMath

- Polinomio de Taylor de segundo orden de $f(x) = \cos(x)$ en $c = 0$: `(cos(x)).taylor(x,0,2)`
- Polinomio de Taylor de cuarto orden de $f(x) = \cos(x)$ en $c = \pi/2$: `(cos(x)).taylor(x,pi/2,4)`
- Polinomio interpolador de Lagrange que pasa por los puntos $(-1,3)$, $(2,1)$ y $3,2$: `R.lagrange_polynomial([(-1,3),(2,1),(3,2)])`

  Nota: En este último caso, es necesario primero ejecutar el siguiente comando: `R = PolynomialRing(RR,'x')`

4.5 Python

```python
import numpy as np
import sympy as sp
from scipy.interpolate import lagrange

x, expresion = sp.symbols('x expresion')

expresion=sp.cos(x)
taylor = expresion.series(x, 0, 2)
print(taylor)
```

```
taylor = expresion.series(x, 0, 3)
print(taylor)

taylor = expresion.series(x, sp.pi/2, 5)
print(taylor)

x = np.array([-1, 2, 3])
y = np.array([3, 1, 2])
poly = lagrange(x, y)
print(poly)
```

5 Integración

5.1 Mathematica/WolframAlpha

- $\int x\,e^{2x}$: `Integrate[xE^(2x),x]`
- $\int_0^{3\pi/2} \text{sen}(2x-1)$: `Integrate[Sin[2x-1],x,0,3(Pi/2)]`
- Área geométrica de $f(x) = \text{sen}(2x-1)$ entre $x = 0$ y $x = 3\pi/2$:
  `Integrate[Abs[Sin[2x-1]],x,0,3(Pi/2)]`
- Área entre las funciones $f(x) = x^2$ y $g(x) = \sqrt{(x)}$ entre $x = 0$ y $x = 2$:
  `Integrate[Abs[x^2-Sqrt[x]],x,0,2]`
- Longitud de arco de la curva $f(x) = x^3 - 1$ entre $x = -2$ y $x = 1$:
  `ArcLengthIntegral[x^3-1,x,-2,1]`

5.2 GeoGebra

- $\int x\,e^{2x}$: `Integral(x*e^(2x))`
- $\int_0^{3\pi/2} \text{sen}(2x-1)$: `Integral(sin(2x-1),0,3*(Pi/2))`
- Área geométrica de $f(x) = \text{sen}(2x-1)$ entre $x = 0$ y $x = 3\pi/2$:
  `Integral(|sin(2x-1)|,0,3(Pi/2))`
- Área entre las funciones $f(x) = x^2$ y $g(x) = \sqrt{x}$ entre $x = 0$ y $x = 2$:
  `Integral(|x^2-sqrt(x)|,0,2)`
- Longitud de arco de la curva $f(x) = x^3 - 1$ entre $x = -2$ y $x = 1$: `Length(x^3-1,-2,1)`

5.3 Maxima

- $\int x\,e^{2x}$: `integrate (x*%e**(2*x), x);`
- $\int_0^{3\pi/2} \text{sen}(2x-1)$: `integrate (sin(2*x-1), x, 0,3*(%pi/2));`

5.4 SageMath

Antes de realizar los cálculos, es necesario incluir este comando:

```
from sage.symbolic.integration.integral import definite_integral
```

- $\int x\,e^{2x}$: `integral(x*e^(2*x),x)`
- $\int_0^{3\pi/2} \text{sen}(2x-1)$: `integral(sin(2*x-1),x,0,3*pi/2)`
- Área geométrica de $f(x) = \text{sen}(2x-1)$ entre $x = 0$ y $x = 3\pi/2$: `integral(abs(sin(2*x-1)),x,0,3*pi/2)`
- Área entre las funciones $f(x) = x^2$ y $g(x) = \sqrt{(x)}$ entre $x = 0$ y $x = 2$: `integral(abs(x^2-sqrt(x)),x,0,2)`
- Longitud de arco de la curva $f(x) = x^3 - 1$ entre $x = -2$ y $x = 1$: `show((integral(sqrt(1+(derivative(x^3-1,x))^2),x,-2,1)).n())`

5.5 Python

```python
import numpy as np
import sympy as sp

x, expresion = sp.symbols('x expresion')

# Integral indefinida

integral = sp.symbols('integral')
expresion = x*sp.exp(2*x)
integral = sp.integrate(expresion, x)
print ("La integral indefinida de")
sp.pprint(expresion)
print ("es")
sp.pprint(integral)

# Integral definida

expresion = sp.sin(2*x-1)
sp.pprint(expresion)
print ("La integral entre 0 y 3*(sp.pi/2) ", (sp.integrate(expresion,(x,0,3*(sp.pi/2)))).n())
```

Bibliografía

En la elaboración de estos apuntes se ha utilizado el siguiente material:

- Mathematica/WolframAlpha. https://www.wolframalpha.com
- GeoGebra. https://www.geogebra.org
- Maxima. https://maxima.sourceforge.io
- SageMath: https://www.sagemath.org
- Python: https://es.python.org
- Javier García Algarra. https://github.com/jgalgarra/PythonMatematicas. Repositorio PythonMatematicas.