ITAM

DATA SCIENCE AND MACHINE LEARNING APPLIED TO FINANCIAL MARKETS¶

Modulo III

Tarea: Métodos Numéricos en Python

Objetivo:

El estudiante será capaz de aplicar herramientas de scipy y numpy para resolver problemas prácticos relacionados con integración numérica, interpolación, ajuste polinómico, búsqueda de raíces y optimización mediante diferentes métodos, incluyendo descenso de gradiente.

Instrucciones

1. Crea un archivo en Python o Jupyter Notebook donde desarrolles todos los ejercicios listados abajo.
2. Para cada ejercicio:
    a. Explica brevemente el procedimiento seguido.
    b. Escribe el código en Python con los métodos adecuados (scipy.integrate, scipy.interpolate, numpy.polyfit, scipy.optimize, etc.).
    c. Muestra los resultados en consola o mediante gráficas cuando corresponda.
3. Incluye al final una conclusión breve de lo aprendido en la tarea.

Ejercicios a realizar (Bloque 1 – Básico)

1. **Integración numérica:** Calcula la integral definida $f(x) = e^{-x^2}$ en el intervalo [0, 1] usando scipy.integrate.quad. Compara el resultado con la integral aproximada mediante la regla del trapecio con numpy.trapz (investigar).

2. **Interpolación 1D:** Dada la tabla de puntos:

x=[0,1,2,3,4]

y=[1,2.7,5.8,6.6,7.5]

Construye una interpolación cúbica con interp1d y evalúa el valor aproximado de f(2.5). Grafica los puntos originales y la curva interpolada.

3. **Ajuste polinómico:** Con los mismos puntos anteriores, ajusta un polinomio de grado 2 con numpy.polyfit. Grafica los puntos, la curva interpolada y el polinomio ajustado en la misma figura.

4. **Interpolación polinómica completa:** Usa numpy.polyfit con grado 4 para obtener el polinomio interpolador que pasa por todos los puntos anteriores. Evalúa el polinomio en x=2.5 y compáralo con el valor de la interpolación cúbica.

5. **Búsqueda de raíces:** Encuentra una raíz de la función $f(x) = x^3 - 6x^2 + 11x - 6$ usando numpy.roots y luego con el método de Newton-Raphson (scipy.optimize.newton).

Ejercicios a realizar (Bloque 2 – Intermedio)

1. **Integral aplicada:** Calcula el área bajo la curva de f$f(x) = \sin(x)$ en el intervalo [0, π] usando quad y compáralo con la solución analítica.

2. **Interpolación de datos ruidosos:** Genera 20 puntos de $f(x) = cos(x) + \varepsilon$ en el intervalo [0, 10], donde ϵ∼N(0,0.1) Aplica interpolación lineal y cúbica, y compara gráficamente cuál se ajusta mejor.

3. **Ajuste polinómico aplicado:** Ajusta un polinomio de grado 3 a los datos ruidosos anteriores. Evalúa el polinomio en 100 puntos uniformemente espaciados y grafica.

4. **Raíces con búsqueda numérica:** Encuentra las raíces de $f(x) = cos(x) - x$ en el intervalo [0, 2] usando Newton Raphson (implementa una rutina propia) . Compara los resultados.

5. **Optimización con scipy:** Encuentra el mínimo de la función $f(x) = x^4 - 3x^3 + 2$ usando scipy.optimize.

6. **Descenso de gradiente:** Implementa una función en Python que aplique descenso de gradiente para encontrar el mínimo de $f(x) = (x - 3)^2 + 4$

    - Usa una tasa de aprendizaje α de 0.1, 0.5 y .9 y realiza lo sig:

        1. Inicia en X0=0.
        2. Haz 20 iteraciones y muestra la evolución de x y de f(x).
        3. Da tus conclusiones.

Formato de Entrega

- **Archivo principal:** un script en Python (.py) o un notebook (.ipynb) con todas las soluciones, comentarios y gráficas.
- **PDF adjunto:** explicación breve (1–2 páginas) de cada ejercicio, justificando el método usado y la interpretación de los resultados.
- **Nomenclatura de archivos:**
    - Tarea_MetodosNumericos_NombreApellidoAA.py
    - Tarea_MetodosNumericos_NombreApellidoAA.pdf

- **Entrega:** por correo en una carpeta comprimida .zip con ambos archivos, a más tardar el **13/10/2025 12:00 a.m.**