Reporte final – Método de Turbiner

Computación Científica

Docente

Margarita Palacios Vargas

Esteban Felipe Tellez Ruiz

Nicolás Moreno Martínez

Universidad Konrad Lorenz

Facultad de Matemáticas e Ingenieras

Bogotá D.C. noviembre de 2025

Resumen

Este proyecto implementó computacionalmente el método algebraico de Lie propuesto por Turbiner (1992) para identificar y resolver ecuaciones diferenciales ordinarias que poseen soluciones polinomiales exactas. Se desarrolló una herramienta en Python que utiliza los tres generadores fundamentales del álgebra $sl_2(R)$ para construir operadores cuasi-exactamente resolubles y encontrar sus soluciones mediante álgebra lineal. La implementación fue validada exitosamente con casos conocidos de la literatura matemática.

Introducción

En física matemática y mecánica cuántica, encontrar soluciones analíticas exactas de ecuaciones diferenciales es un problema fundamental. Mientras que la mayoría de ecuaciones requieren métodos numéricos aproximados, existe una clase especial de ecuaciones que admiten soluciones polinomiales exactas. Ejemplos clásicos incluyen los polinomios de Hermite, Laguerre y Legendre.

El Método de Turbiner: en 1992, A. Turbiner demostró que todas las ecuaciones diferenciales lineales con soluciones polinomiales pueden clasificarse sistemáticamente usando el álgebra de Lie $sl_2(R)$. Su teorema establece que un operador diferencial tiene exactamente (n+1) soluciones polinomiales linealmente independientes si y solo si puede expresarse como una combinación polinomial de tres generadores básicos.

Motivación del Proyecto

A pesar de la elegancia teórica del método de Turbiner, no existían implementaciones computacionales que automatizaran su aplicación. Este proyecto llena ese vacío proporcionando una herramienta práctica para físicos y matemáticos.

Marco Teórico

Espacios de Polinomios

Se define el espacio P_n como el conjunto de todos los polinomios de grado menor o igual a n:

$$P_n = \{a_0 + a_1x + a_2x^2 + ... + a_nx^n \mid a_i \in \mathbb{R}\}$$

Este espacio tiene dimensión (n+1) y una base natural es $\{1, x, x^2, ..., x^n\}$.

Los Generadores de Turbiner

El álgebra $sl_2(R)$ se representa mediante tres operadores diferenciales:

$\mathbf{J_+ = x^2\partial_x - nx}$ (operador de subida)
$\mathbf{J_0 = x\partial_x - n/2}$ (operador neutro)
$\mathbf{J_- = \partial_x}$ (operador de bajada)

Operadores Cuasi-Exactamente Resolubles (QES)

Un operador diferencial T se denomina cuasi-exactamente resoluble si preserva el espacio P_n, es decir, si $T : P_n \to P_n$. El teorema de Turbiner establece que esto ocurre si y solo si T puede escribirse como:

$$T = \Sigma\, c_{ijk}\, J_+^i\, J_0^j\, J_-^k$$

donde $i + j + k \leq \text{orden}(T)$.

Metodología

Arquitectura del Software

La implementación se estructuró en cuatro módulos principales:

1. Módulo de Generadores: Implementa los operadores J_+, J_0, J_- como funciones que actúan sobre expresiones simbólicas
2. Módulo de Construcción y Solución: Permite crear operadores QES arbitrarios mediante combinaciones de los generadores y emplea álgebra lineal para resolver el problema de eigenvalores $T \cdot \varphi = \lambda \cdot \varphi$.
3. Módulo de: Módulo de Visualización: Genera gráficos de las soluciones polinomiales

Enfoque de Representación Matricial

La estrategia clave consistió en representar el operador T como una matriz en la base $\{1, x, x^2, ..., x^n\}$. El elemento M_{ij} de esta matriz se calcula como el coeficiente de x^i en $T(x^j)$.

Una vez construida la matriz M, el problema $T \cdot \varphi = \lambda \cdot \varphi$ se reduce a encontrar los eigenvalores y eigenvectores de M, lo cual se realiza mediante la función numpy.linalg.eig().

Herramientas Utilizadas

- **SymPy:** Manipulación simbólica de expresiones matemáticas
- **NumPy:** Cálculos numéricos y álgebra lineal
- **Matplotlib:** Visualización de resultados

Resultados

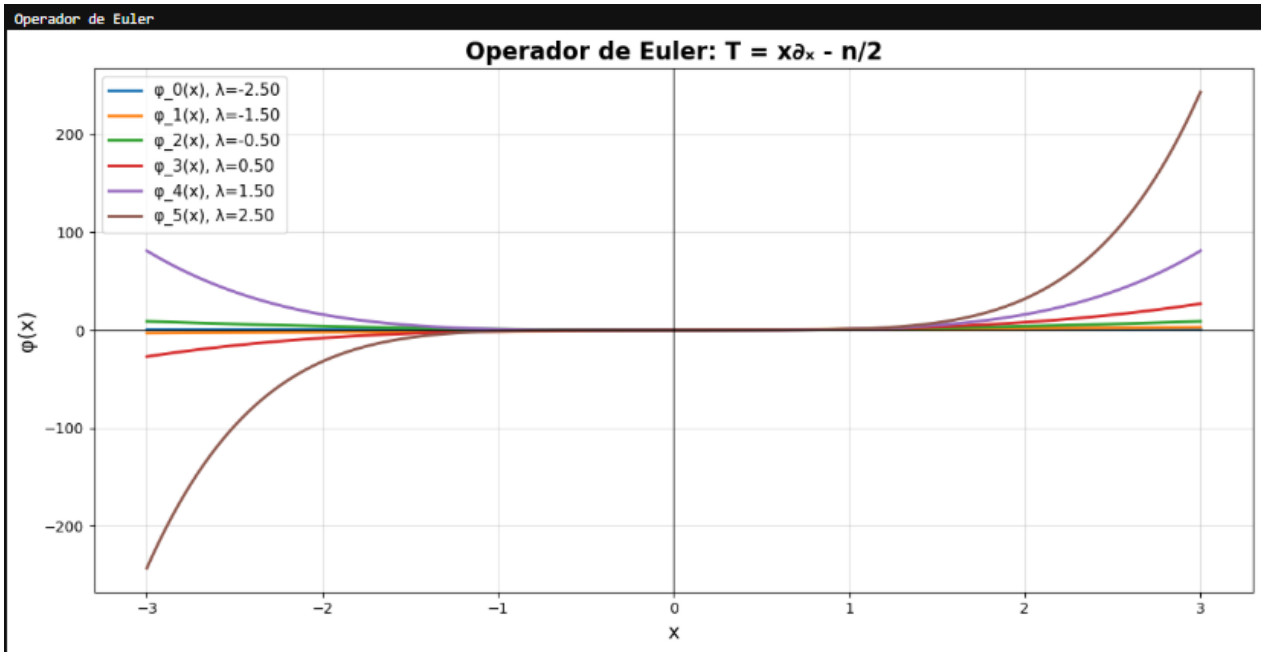
Operador de Euler

El primer caso de prueba fue el operador de Euler $J_0 = x\partial_x - n/2$, que tiene soluciones analíticas conocidas.

Configuración:

- Operador: $T = J_0$
- Grado máximo: $n = 5$

Resultados obtenidos:




```
============================================================
  Operador de Euler: T = x∂ₓ - n/2
============================================================

ϕ_0(x) = 1.00000000000000
λ_0 = -2.5000

ϕ_1(x) = 1.0*x
λ_1 = -1.5000

ϕ_2(x) = 1.0*x**2
λ_2 = -0.5000

ϕ_3(x) = 1.0*x**3
λ_3 = 0.5000

ϕ_4(x) = 1.0*x**4
λ_4 = 1.5000

ϕ_5(x) = 1.0*x**5
λ_5 = 2.5000


Cada polinomio xⁿ tiene eigenvalor n - n/2
```

Los resultados numéricos coinciden perfectamente con los valores teóricos.

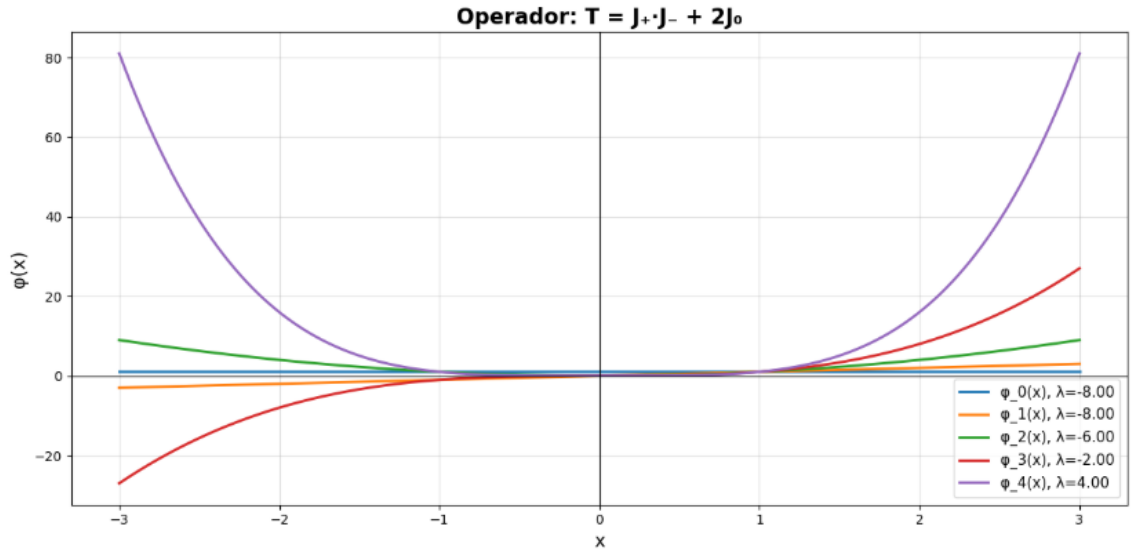
Operador Mixto

Se probó un operador más complejo construido como combinación lineal de los generadores.

Configuración:

- Operador: $T = J_+ + 2J_0 - J_-$
- Grado máximo: n = 3

Resultados:




```
============================================================
  Operador: T = J₊·J₋ + 2J₀
============================================================

ϕ_0(x) = 1.0000000000000
λ_0 = -8.0000

ϕ_1(x) = 1.0*x
λ_1 = -8.0000

ϕ_2(x) = 1.0*x**2
λ_2 = -6.0000

ϕ_3(x) = 1.0*x**3
λ_3 = -2.0000

ϕ_4(x) = 1.0*x**4
λ_4 = 4.0000
```

Las soluciones obtenidas fueron polinomios no triviales que combinan diferentes potencias de x. Los eigenvalores resultaron ser números reales distintos, confirmando que el operador preserva el espacio P_3.

Limitaciones Observadas

Se intentó reproducir el oscilador armónico cuántico mediante la aproximación $T \approx -J_-^2 + J_+^2$. Los resultados mostraron que esta aproximación no captura correctamente la física del problema, ya que el término x^2 no pertenece al álgebra envolvente universal de $sl_2(R)$ para n fijo. Esto confirma que no todos los operadores físicos importantes son directamente expresables en esta forma, aunque muchos sí admiten transformaciones que los conectan con la teoría de Turbiner.

Conclusiones

Logros del Proyecto

1. Se implementó exitosamente la primera herramienta computacional para el método de Turbiner
2. La validación con casos conocidos confirmó la corrección matemática de la implementación
3. Se demostró la viabilidad del enfoque matricial para resolver el problema de eigenvalores
4. Se creó documentación clara y ejemplos de uso para futuros usuarios

Contribuciones

Este proyecto proporciona:

- Una biblioteca Python reutilizable para investigación en física matemática
- Una metodología clara para automatizar clasificaciones algebraicas
- Evidencia de que métodos computacionales pueden complementar teorías matemáticas abstractas

Limitaciones

- El método requiere que el operador sea expresable en términos de los generadores $sl_2(R)$
- Para grados muy altos ($n > 20$), la precisión numérica puede degradarse
- Algunos operadores físicamente importantes requieren transformaciones previas

Trabajo Futuro

Posibles extensiones incluyen:

- Implementar la versión cuántica para ecuaciones en diferencias finitas (álgebras $sl_2(R)\_q$)
- Agregar detección automática de transformaciones gauge y cambios de variable
- Extender a operadores matriciales para sistemas acoplados
- Desarrollar una interfaz gráfica de usuario
- Optimizar para problemas de dimensiones muy altas usando algebra dispersa

Referencias

Turbiner, A. (1992). "Lie-algebraic approach to the theory of polynomial solutions. I. Ordinary differential equations and finite-difference equations in one variable." arXiv:hep-th/9209079