

## Eksplorasi Penggunaan Python Dalam Memahami Pertidaksamaan Kalkulus Secara Praktis

Muh Gunawan Hadi<sup>1\*</sup>, Heri Purnomo<sup>2</sup>, Eka Rian Saputra<sup>3</sup>, Rafi Mupashal<sup>4</sup>, Perani Rosyani<sup>5</sup>

<sup>1,2,3,4,5</sup> Teknik Informatika, Universitas Pamulang, Tangerang Selatan, Indonesia

Email: <sup>1\*</sup>[hadi.gunawan1705@gmail.com](mailto:hadi.gunawan1705@gmail.com), <sup>2</sup>[heripurnomo2041@gmail.com](mailto:heripurnomo2041@gmail.com), <sup>3</sup>[khaaaka1@gmail.com](mailto:khaaaka1@gmail.com),  
<sup>4</sup>[rafimupashal07@gmail.com](mailto:rafimupashal07@gmail.com), <sup>5</sup>[dosen00837@unpam.ac.id](mailto:dosen00837@unpam.ac.id)

(\* : coresponding author)

**Abstrak**—Tujuan Penelitian ini mengulas kompleksitas pertidaksamaan kalkulus yang sering kali menjadi rintangan bagi pemula. Fokus penelitian adalah pada bagaimana penggunaan bahasa pemrograman Python dapat secara praktis membantu pemula memahami pertidaksamaan kalkulus. Pemilihan Python sebagai bahasa utama berlandaskan pada telaah literatur, melibatkan metode numerik dan implementasi solusi praktis. Penekanan diberikan pada konsep-konsep kunci pertidaksamaan kalkulus dengan mengkaji beberapa metode penerapan Python yang telah diusulkan dalam literatur, menggunakan metode studi literatur dan analisis. Diskusi melibatkan kelebihan Python, seperti fleksibilitas dan kejelasan kode, sambil mempertimbangkan tantangan yang mungkin timbul, termasuk kompleksitas komputasi. Hasil analisis memberikan wawasan mendalam mengenai peran Python dalam pemahaman praktis pertidaksamaan kalkulus. Penelitian ini bertujuan memberikan panduan yang bermanfaat bagi pemula dalam meraih konsep-konsep kalkulus. Dengan demikian, penelitian ini diharapkan menjadi referensi berharga dan titik awal eksplorasi lebih lanjut dalam pembelajaran kalkulus.

**Kata Kunci:** Python, Kalkulus, Pertidaksamaan, Metode Numerik, Fleksibilitas Kode

**Abstract**—This research aims to delve into the intricacies of calculus inequalities, a common stumbling block for beginners. It focuses on how the practical application of the Python programming language can aid newcomers in comprehending these inequalities. The decision to use Python as the primary language is grounded in a thorough literature review, incorporating numerical methods and the implementation of real-world solutions. The emphasis is placed on key concepts of calculus inequalities by scrutinizing various Python implementation methods proposed in the literature, employing literature review techniques and analysis. The discussion encompasses the benefits of Python, such as code flexibility and clarity, while acknowledging potential challenges, including computational complexity. The analysis results offer profound insights into the role of Python in fostering a practical understanding of calculus inequalities. The research is intended to serve as a helpful guide for beginners to grasp calculus concepts. As such, it is anticipated that this research will be a valuable reference and a launching point for further exploration in calculus learning.

**Keywords:** *Python, Calculus, Inequalities, Numerical Methods, Code Flexibility*

### 1. PENDAHULUAN

Pertidaksamaan kalkulus merupakan salah satu konsep dasar matematika, pertidaksamaan kalkulus menimbulkan tantangan bagi pemula yang berusaha memahami konsep kalkulus secara menyeluruh. Keberhasilan dalam memahami pertidaksamaan ini secara mendalam memegang peranan kunci dalam membangun dasar pemahaman matematika yang kuat.

Berdasarkan dari kompleksitas pertidaksamaan kalkulus, penelitian ini bertujuan untuk mengeksplorasi sejauh mana bahasa pemrograman python dapat menjadi solusi praktis. Fokus utama penelitian adalah menjawab seberapa besar Python dapat membantu dalam meningkatkan pemahaman pertidaksamaan kalkulus, terutama bagi mereka yang mungkin menghadapi hambatan dalam belajar.

Dengan merinci konsep-konsep kalkulus dan meninjau kompleksitas pertidaksamaan, penelitian ini bertujuan memberikan solusi yang konkret dan dapat diaplikasikan. Pemilihan Python sebagai bahasa pemrograman utama didasarkan pada fleksibilitas dan kejelasan kode. Selain itu, Python juga dipercaya dapat menjadi alat efektif untuk mengatasi hambatan pemahaman kalkulus.

Penelitian ini penting karena tidak hanya mencari solusi praktis terhadap hambatan pemahaman kalkulus, tetapi juga bertujuan untuk membangun fondasi yang kuat dalam pembelajaran kalkulus. Penelitian ini diharapkan dapat memberikan kontribusi terhadap

pengembangan metode pembelajaran yang lebih efektif, membawa pemahaman kalkulus ke tingkat yang lebih mendalam bagi pemula yang ingin belajar.

## 2. METODE PENELITIAN

Dalam penelitian ini, kami menggunakan pendekatan studi literatur dan analisis untuk mengeksplorasi bagaimana peran Python dalam memahami pertidaksamaan kalkulus. Tahap awal melibatkan identifikasi literatur yang mencakup konsep-konsep kalkulus dan implementasi Python dalam pemahaman pertidaksamaan. Langkah berikutnya, dilakukan analisis mendalam terhadap literatur yang terpilih, fokus pada metode numerik dan solusi praktis yang diusulkan. Implementasi Python untuk menyelesaikan pertidaksamaan kalkulus dieksplorasi dengan cermat, dengan mencatat kelebihan dan kekurangan masing-masing metode. Dengan metode studi literatur dan analisis kami berusaha memahami kontribusi Python dalam menangani kompleksitas pertidaksamaan kalkulus. Penelitian ini diharapkan memberikan panduan yang praktis bagi pemula dalam memahami konsep-konsep matematika yang mendasar.

## 3. ANALISA DAN PEMBAHASAN

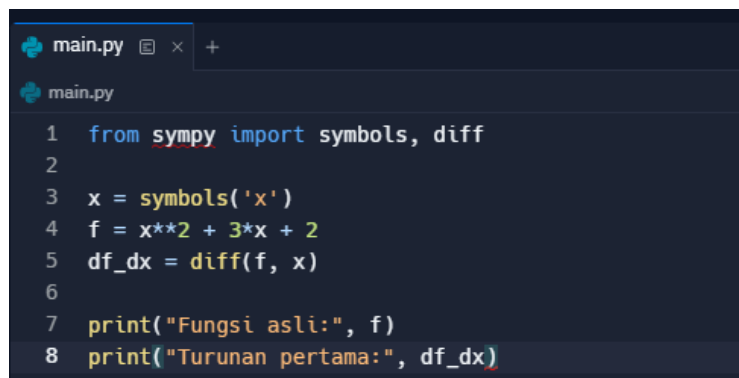
### 3.1 Pengenalan Python untuk Kalkulus

Python dapat berperan dalam memahami konsep kalkulus dengan berbagai cara, termasuk implementasi perhitungan matematika, visualisasi grafik fungsi, dan pemecahan masalah yang melibatkan konsep-konsep kalkulus.

#### 3.1.1 Simbolik Computing

Python memiliki pustaka simbolik seperti SymPy yang memungkinkan manipulasi ekspresi matematika secara simbolik. Dengan menggunakan SymPy, Anda dapat melakukan operasi kalkulus seperti diferensiasi dan integrasi secara simbolik. Ini membantu dalam memahami konsep dasar kalkulus tanpa perlu menyelesaikan perhitungan manual.

Contoh menggunakan SymPy untuk diferensiasi:



```
main.py [x] +
main.py
1 from sympy import symbols, diff
2
3 x = symbols('x')
4 f = x**2 + 3*x + 2
5 df_dx = diff(f, x)
6
7 print("Fungsi asli:", f)
8 print("Turunan pertama:", df_dx)
```

Gambar 1. Code Python Sympy

#### 3.1.1 Numerical Computing

Python memiliki pustaka seperti NumPy yang memungkinkan perhitungan numerik. Dengan NumPy, Anda dapat menghitung nilai numerik dari fungsi, turunan, dan integral. Ini memungkinkan eksplorasi konsep-konsep kalkulus pada data numerik.

Contoh menggunakan NumPy untuk menghitung turunan numerik:

```
main.py x +
main.py > ...
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 def f(x):
5     return x**2 + 3*x + 2
6
7 x_values = np.linspace(-5, 5, 100)
8 y_values = f(x_values)
9
10 # Menghitung turunan numerik
11 df_dx_numeric = np.gradient(y_values, x_values)
12
13 # Visualisasi fungsi dan turunan numerik
14 plt.plot(x_values, y_values, label='Fungsi Asli')
15 plt.plot(x_values, df_dx_numeric, label='Turunan Numerik')
16 plt.legend()
17 plt.show()
18
```

Gambar 2. Code Python Numpy

### 3.1.1 Visualisasi Grafik

Python memiliki pustaka seperti Matplotlib dan Seaborn yang memungkinkan visualisasi grafik fungsi matematika. Dengan membuat grafik fungsi, Anda dapat memahami perilaku fungsi pada berbagai interval dan mendapatkan intuisi tentang konsep-konsep kalkulus seperti titik kritis, ekstremum, dan asymptot.

Contoh visualisasi menggunakan Matplotlib:

```
main.py > ...
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 def f(x):
5     return x**2 + 3*x + 2
6
7 x_values = np.linspace(-5, 5, 100)
8 y_values = f(x_values)
9
10 plt.plot(x_values, y_values, label='Fungsi Asli')
11 plt.title('Grafik Fungsi Matematika')
12 plt.xlabel('x')
13 plt.ylabel('f(x)')
14 plt.legend()
15 plt.grid(True)
16 plt.show()
17
```

Gambar 3. Code python Matplotlib

## 3.2 Alasan penggunaan Python untuk analisis matematika

Python dipilih untuk analisis matematika karena sintaksisnya yang mudah dipahami, ekosistem pustakanya yang kaya seperti NumPy dan Matplotlib, serta dukungan komunitas yang besar. Penggunaan Python meluas di berbagai bidang, menunjukkan fleksibilitasnya. Kelebihan

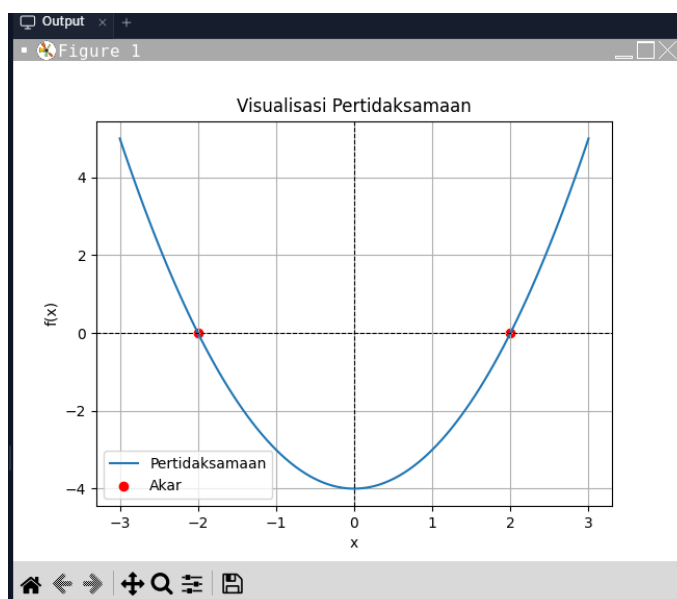
lainnya termasuk interoperabilitas yang baik dengan bahasa lain, status open source yang gratis, dan kemudahan eksplorasi menggunakan lingkungan pengembangan interaktif seperti Jupyter Notebooks. Secara keseluruhan, Python menyediakan alat yang efisien dan ramah pengguna untuk memahami dan menganalisis konsep matematika dengan berbagai kebutuhan dan tingkat kompleksitas.

### 3.3 Penyelesaian Pertidaksamaan Kalkulus Menggunakan Python

pemecahan pertidaksamaan kalkulus dengan menggunakan Python, khususnya menggunakan pustaka SymPy untuk penyelesaian simbolik dan pustaka Matplotlib untuk visualisasi grafik. Berikut contoh pemecahan pertidaksamaan kalkulus dengan menggunakan Python. Misalkan kita memiliki pertidaksamaan sederhana  $x^2 - 4 = 0$  yang ingin kita selesaikan.

```
main.py > ...
1 from sympy import symbols, solve
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5 # Variabel simbolik
6 x = symbols('x')
7
8 # Persamaan
9 equation = x**2 - 4
10
11 # Menyelesaikan pertidaksamaan secara simbolik
12 solutions = solve(equation, x)
13 print("Solusi pertidaksamaan:", solutions)
14
15 # Visualisasi grafik fungsi
16 x_values = np.linspace(-3, 3, 100)
17 y_values = x_values**2 - 4
18
19 plt.plot(x_values, y_values, label='Pertidaksamaan')
20 plt.axhline(0, color='black', linestyle='--', linewidth=0.8)
21 plt.axvline(0, color='black', linestyle='--', linewidth=0.8)
22 plt.scatter(solutions, [0, 0], color='red', label='Akar')
23 plt.legend()
24 plt.title('Visualisasi Pertidaksamaan')
25 plt.xlabel('x')
26 plt.ylabel('f(x)')
27 plt.grid(True)
28 plt.show()
29
```

Gambar 4. Penyelesaian Pertidaksamaan pada Python



Gambar 5. Visualisasi Pertidaksamaan

Dalam contoh ini, pertidaksamaan sederhana  $x^2 - 4 = 0$  diselesaikan secara simbolik menggunakan SymPy, dan solusinya ditampilkan pada grafik fungsi tersebut.

Grafik akan menunjukkan garis fungsi kuadratik bersama dengan dua akarnya pada titik  $x = -2$  dan  $x = 2$ . Pertidaksamaan bisa diganti sesuai yang diinginkan.

### 3.4 Implementasi algoritma dalam Python

Implementasi algoritma untuk menyelesaikan pertidaksamaan kalkulus dapat melibatkan pustaka simbolik seperti SymPy atau pustaka numerik seperti NumPy, tergantung pada jenis pertidaksamaan yang ingin Anda selesaikan. Berikut adalah contoh implementasi menggunakan SymPy untuk menyelesaikan pertidaksamaan kuadratik  $ax^2 + bx + c > 0$

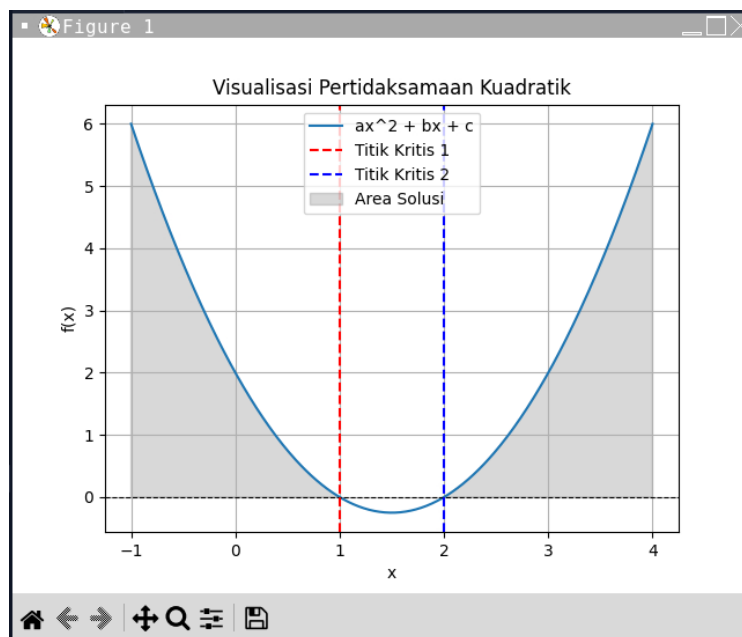
```
main.py > ...
1 from sympy import symbols, solve, Interval, oo
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5 def solve_quadratic_inequality(a, b, c):
6     """
7     Menyelesaikan pertidaksamaan kuadratik  $ax^2 + bx + c > 0$ .
8
9     Parameters:
10    - a, b, c (float): Koefisien pertidaksamaan kuadratik.
11
12    Returns:
13    - solution (str): Solusi dalam bentuk teks.
14    - critical_points (list): Titik-titik kritis.
15    """
16    x = symbols('x')
17    quadratic_expression = a*x**2 + b*x + c
18
19    critical_points = solve(quadratic_expression, x)
20
21    if len(critical_points) == 2:
22        interval1 = Interval(-oo, critical_points[0])
23        interval2 = Interval(critical_points[1], oo)
24        solution = f'{interval1.union(interval2)}'
25    elif len(critical_points) == 1:
26        if a > 0:
27            interval = Interval(critical_points[0], oo)
28        else:
29            interval = Interval(-oo, critical_points[0])
30        solution = f'{interval}'
31    else:
32        if a > 0:
33            solution = 'Semua bilangan real'
34        else:
35            solution = 'Tidak ada solusi real'
36
37    return solution, critical_points
```

**Gambar 6.** Penyelesaian Pertidaksamaan Kuadratik

```

38
39 # Contoh penggunaan
40 a_val = 1
41 b_val = -3
42 c_val = 2
43
44 solution_text, critical_points = solve_quadratic_inequality(a_val, b_val, c_val)
45
46 print(f"Solusi: {solution_text}")
47 print(f"Titik-titik kritis: {critical_points}")
48
49 # Visualisasi grafik fungsi
50 x_values = np.linspace(-1, 4, 400)
51 y_values = a_val*x_values**2 + b_val*x_values + c_val
52
53 plt.plot(x_values, y_values, label='ax^2 + bx + c')
54 plt.axhline(0, color='black', linestyle='--', linewidth=0.8)
55 plt.axvline(critical_points[0], color='red', linestyle='--', label='Titik Kritis 1')
56 plt.axvline(critical_points[1], color='blue', linestyle='--', label='Titik Kritis 2')
57 plt.fill_between(x_values, y_values, where=(y_values > 0), color='gray', alpha=0.3, label='Area Solusi')
58 plt.legend()
59 plt.title('Visualisasi Pertidaksamaan Kuadratik')
60 plt.xlabel('x')
61 plt.ylabel('f(x)')
62 plt.grid(True)
63 plt.show()
64
    
```

**Gambar 7.** Penyelesaian Pertidaksamaan Kuadratik



**Gambar 8.** Visualisasi Pertidaksamaan Kuadratik

Dalam contoh diatas, fungsi `solve_quadratic_inequality` menerima koefisien a,b, dan c dari pertidaksamaan kuadratik dan memberikan solusi dalam bentuk interval serta menunjukkan titik-titik kritis. Visualisasi grafik menunjukkan area solusi pada grafik fungsi kuadratik. Anda dapat mengganti nilai koefisien sesuai dengan pertidaksamaan yang ingin Anda selesaikan.

#### 4. KESIMPULAN

Eksplorasi penggunaan Python dalam memahami pertidaksamaan kalkulus secara praktis dengan teknik NumPy menunjukkan bahwa kombinasi antara bahasa pemrograman Python dan

pustaka numerik NumPy memberikan pendekatan yang kuat dan efisien dalam memahami konsep kalkulus. Python, dengan ekosistemnya yang kaya, memberikan fleksibilitas dan kemudahan dalam implementasi algoritma matematika kompleks. Sementara itu, NumPy memberikan kecepatan dan efisiensi dalam operasi numerik dan manipulasi array, memungkinkan penyelesaian pertidaksamaan kalkulus dengan lebih efektif.

Pertidaksamaan kalkulus, terutama melibatkan turunan dan integral, menjadi fokus utama dalam eksplorasi ini. NumPy menyediakan fungsi-fungsi matematika yang dapat digunakan untuk mengevaluasi turunan dan integral, memudahkan analisis dan pemahaman konsep tersebut. Penggunaan Python juga memungkinkan visualisasi yang efektif melalui pustaka seperti Matplotlib atau Seaborn, memberikan gambaran grafis yang mendukung interpretasi hasil perhitungan.

Keuntungan praktis dari penggunaan Python dan NumPy terletak pada kemampuan untuk menghindari kerumitan implementasi dan fokus pada esensi dari konsep kalkulus. Dengan demikian, eksplorasi ini memberikan wawasan yang lebih dalam tentang pertidaksamaan kalkulus secara praktis, memungkinkan pembelajaran yang lebih efisien dan mendalam dalam konteks matematika terapan. Melalui pendekatan ini, pemahaman terhadap kalkulus dapat diintegrasikan dengan baik dengan kekuatan komputasi Python dan kecepatan numerik NumPy, menciptakan lingkungan belajar yang produktif dan efisien.

## REFERENCES

- Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing* (3rd ed.). Cambridge University Press.
- Heath, M. T. (2002). *Scientific Computing: An Introductory Survey*. McGraw-Hill.
- Langtangen, H. P. (2016). *A Primer on Scientific Programming with Python* (5th ed.). Springer.
- Quarteroni, A., Sacco, R., & Saleri, F. (2010). *Numerical Mathematics* (2nd ed.). Springer.
- Downey, A. B., & Elkner, J. (2008). *Think Python: How to Think Like a Computer Scientist*. O'Reilly Media.
- McKinney, W. (2017). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython* (2nd ed.). O'Reilly Media.