iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

ADOPTION OF PROGRAMMING CODES IN THE DESIGN OF GRAVITY DAMS

E.C. Nwadike,
S.C. Anyanwu,
N.P. Ogbonna,
I.C. Nwosu
Lecturer, Civil Engineering Department, Federal Polytechnic Nekede, Owerri, Nigeria
nopsoftinc@yahoo.com

ABSTRACT
The advancement of computational tools has revolutionized structural engineering, allowing for precise, efficient, and cost-effective design methodologies. This paper explores the adoption of programming codes in the design of gravity dams, specifically focusing on rectangular dams, trapezoidal dams with a vertical water face, and trapezoidal dams with an inclined water face. The study includes mathematical formulations, stability checks, and a Python-based computational model for the design and verification of these dams. The proposed methodology enhances accuracy, reduces human errors, and provides engineers with an automated approach to stability assessment.

Keywords:
Programming, codes, python, dams, gravity, design

INTRODUCTION

Gravity dams are massive structures that resist external forces primarily through their own weight. These dams are commonly constructed from concrete or masonry and are widely used for hydroelectric power generation, flood control, and water storage [1],[2]. The stability of such dams is evaluated based on sliding, overturning, and bearing capacity criteria. The traditional hand-calculation methods are prone to human errors and inefficiencies. This paper presents a computational approach to designing gravity dams using Python programming, providing an efficient and replicable framework for engineers. By integrating automation into the design process, engineers can achieve rapid design validation while ensuring compliance with safety standards [3].

TYPES OF GRAVITY DAMS CONSIDERED

1. **Rectangular Dams** - Simple in shape, these dams rely solely on their weight for stability. While easy to construct, they may require excessive material to ensure safety [4].
2. **Trapezoidal Dams with Vertical Water Face** - This type provides better stability and economy compared to rectangular dams. The inclined downstream face reduces material consumption while maintaining strength.
3. **Trapezoidal Dams with Inclined Water Face** - Designed to reduce uplift pressure and enhance stability, these dams optimize structural efficiency by modifying the upstream profile [5].

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

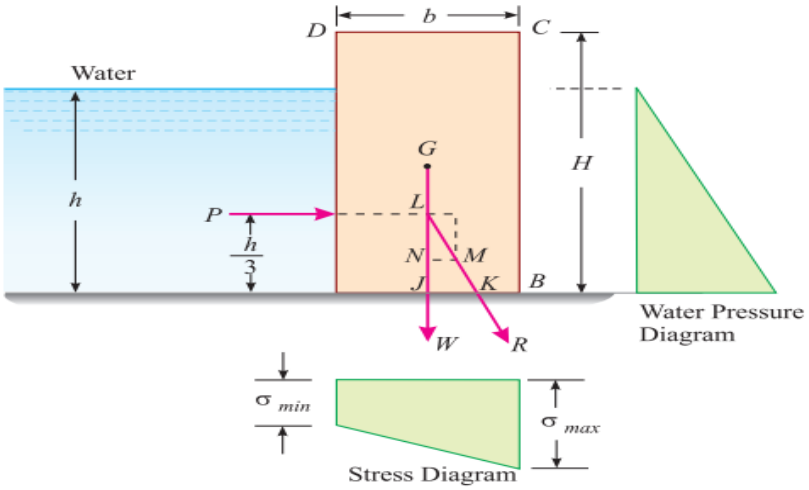

Figure 1 Rectangular Dam [6]

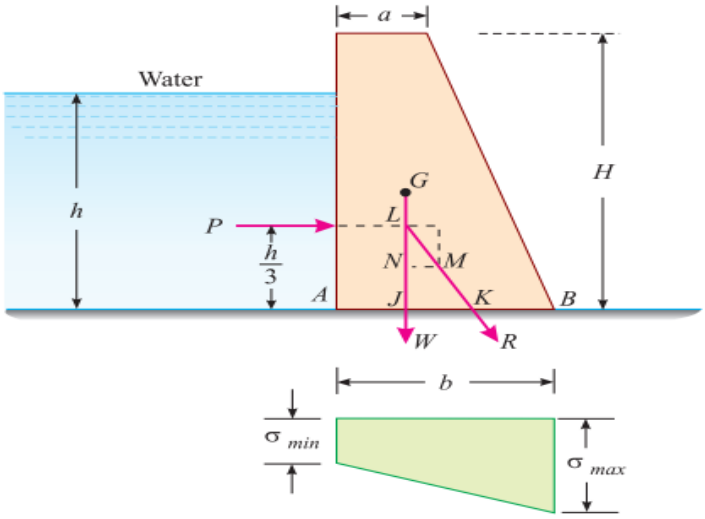

Figure 2 Trapezoidal dam with water face vertical [6]

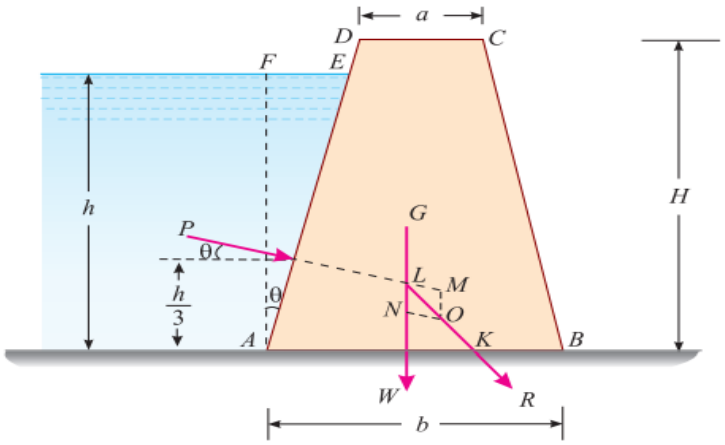

Figure 3 Trapezoidal dam with water face inclined [6]

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

Where:
a = Top width of the dam
b = Bottom width of the dam
H = Height of the dam
ρ = Specific weight of the dam masonry
h = Height of water retained by the dam
w = Specific weight of the water
θ = Inclination of the water face with the vertical

STABILITY ANALYSIS AND DESIGN FORMULATIONS

1. Forces Acting on Gravity Dams:

- **Self-weight of the dam (W)**: The weight of the dam acts vertically downward and contributes to stability.
- **Hydrostatic Water Pressure (P)**: Acts horizontally against the upstream face and is given by:
  $P = \frac{1}{2}\gamma_w h^2$, where γ_w
  where is the unit weight of water and is the water height.
- **Uplift Pressure (U)**: Acts as an upward force due to water infiltration at the base.
- **Seismic Forces (if applicable)**: Accounted for in seismic-prone regions.
- **Silt Pressure and Ice Pressure (if applicable)**: Considered in special cases for comprehensive analysis [7].[8].

2. Stability Checks:

(a) Overturning Stability Check: $FOS = \frac{\text{Sum of Restoring Moments}}{\text{Sum of Overturning Moments}}$ (FOS should be > 1.5 for safety, per Eurocode 7: EN 1997-1)

(b) Sliding Stability Check: $FOS = \frac{\text{Resisting Forces}}{\text{Driving Forces}}$ (FOS should be > 1.5, ensuring frictional resistance is adequate)

(c) Bearing Capacity Check: $\sigma = \frac{P}{A} \leq \sigma_{\text{safe}}$ where is the total vertical load, is the base area, and is the allowable bearing stress.

ABOUT THE PROGRAMMING LANGUAGE: PYTHON

Python is a high-level, interpreted programming language known for its simplicity and readability. Created by Guido van Rossum and first released in 1991, Python follows a design philosophy that emphasizes code clarity and ease of use. Its syntax allows developers to express concepts in fewer lines of code while maintaining readability. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming, making it highly versatile.

One of Python's key strengths is its dynamic type system and automatic memory management, which enhance developer productivity. Additionally, Python boasts a comprehensive standard library that provides built-in functionalities for various applications, including scientific computing, data analysis, and structural engineering computations.

Python is widely used across multiple operating systems, with CPython being its reference implementation. CPython is open-source software, managed by the non-profit Python Software Foundation, and supported by a vast global community of developers.

Advantages of Python in Structural Engineering:

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

1. **Open Source and Community Support** - Python is freely available, and its active community continuously contributes to its development.
2. **Ease of Learning** - The language's simple syntax and readability make it accessible to engineers with limited programming experience.
3. **User-Friendly Data Structures** - Built-in data structures, such as lists and dictionaries, enable efficient handling of engineering data.
4. **Productivity and Speed** - Python facilitates rapid prototyping and execution, streamlining computational analyses in structural engineering.

METHODOLOGY

Example Problem:

A gravity dam has the following properties:

- Height of Dam (h) = 30 m
- Base Width (B) = 20 m
- Top Width (T) = 5 m
- Unit Weight of Concrete (γ_c) = 24 kN/m³
- Unit Weight of Water (γ_w) = 9.81 kN/m³
- Coefficient of Friction (μ) = 0.65
- Allowable Bearing Stress (σ_{safe}) = 1200 kN/m²

Python Program for the design of Retaining Walls

```
import tkinter as tk
from tkinter import messagebox
import math
def calculate_stability():
    # Get user inputs
    height = float(height_entry.get())
    base_width = float(base_width_entry.get())
    top_width = float(top_width_entry.get())
    gamma_c = float(gamma_c_entry.get())
    gamma_w = float(gamma_w_entry.get())
    uplift_coeff = float(uplift_coeff_entry.get())
    water_height = float(water_height_entry.get())
    dam_type = dam_type_var.get()
    # Compute Hydrostatic Force
    hydrostatic_force = 0.5 * gamma_w * water_height ** 2
    hydrostatic_moment = hydrostatic_force * water_height / 3  # Moment about the base

    # Compute Uplift Force
    base_area = base_width * height
    uplift_force = uplift_coeff * base_area * gamma_w
    uplift_moment = uplift_force * (base_width / 2)  # Assumed at centroid of base
    # Compute Weight of Dam
    if dam_type == "Rectangular":
        dam_area = base_width * height
        centroid_x = base_width / 2
```

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

```
    elif dam_type == "Trapezoidal (Vertical Water Face)":
        dam_area = 0.5 * (base_width + top_width) * height
        centroid_x = (base_width + 2 * top_width) / 3  # Approximate centroid location
    elif dam_type == "Trapezoidal (Inclined Water Face)":
        slope = (base_width - top_width) / height  # Slope of the inclined face
        dam_area = 0.5 * (base_width + top_width) * height
        centroid_x = (base_width * 2 + top_width) / 3  # Adjust centroid considering the slope
    weight_dam = gamma_c * dam_area
    weight_moment = weight_dam * centroid_x
    # Compute Factor of Safety (Overturning and Sliding)
    resisting_moment = weight_moment - uplift_moment
    overturning_moment = hydrostatic_moment
    factor_safety_overturning = resisting_moment / overturning_moment
    resisting_force = weight_dam - uplift_force
    driving_force = hydrostatic_force
    factor_safety_sliding = resisting_force / driving_force
    # Compute Bearing Stress
    bearing_stress = resisting_force / base_width
    # Display results
    result_text = f"Factor of Safety (Overturning): {factor_safety_overturning:.2f}\n"
    result_text += f"Factor of Safety (Sliding): {factor_safety_sliding:.2f}\n"
    result_text += f"Bearing Stress: {bearing_stress:.2f} kN/m²\n"
    messagebox.showinfo("Results", result_text)
# GUI setup
root = tk.Tk()
root.title("Gravity Dam Stability Analysis")
tk.Label(root, text="Height of Dam (m):").pack()
height_entry = tk.Entry(root)
height_entry.pack()
tk.Label(root, text="Base Width (m):").pack()
base_width_entry = tk.Entry(root)
base_width_entry.pack()
tk.Label(root, text="Top Width (m):").pack()
top_width_entry = tk.Entry(root)
top_width_entry.pack()
tk.Label(root, text="Height of Water (m):").pack()
water_height_entry = tk.Entry(root)
water_height_entry.pack()
tk.Label(root, text="Concrete Density (kN/m³):").pack()
gamma_c_entry = tk.Entry(root)
gamma_c_entry.pack()
tk.Label(root, text="Water Density (kN/m³):").pack()
gamma_w_entry = tk.Entry(root)
gamma_w_entry.pack()

tk.Label(root, text="Uplift Coefficient:").pack()
```

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

```
uplift_coeff_entry = tk.Entry(root)
uplift_coeff_entry.pack()
dam_type_var = tk.StringVar()
dam_type_var.set("Rectangular")
tk.Label(root, text="Select Dam Type:").pack()
dam_type_menu = tk.OptionMenu(root, dam_type_var, "Rectangular", "Trapezoidal (Vertical Water Face)",
"Trapezoidal (Inclined Water Face)")
dam_type_menu.pack()
tk.Button(root, text="Calculate Stability", command=calculate_stability).pack()
root.mainloop()
```

RESULTS AND DISCUSSION

The input data and the results of the design of the gravity dams in rectangular and trapezoidal shapes are shown in figure 4, 5 and 6 respectively

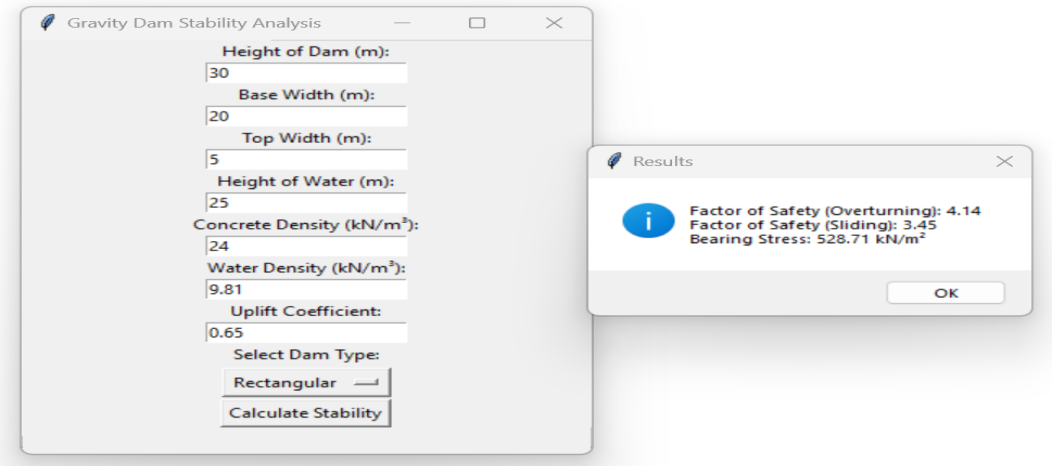

Figure 4 Results of the design of rectangular gravity dam

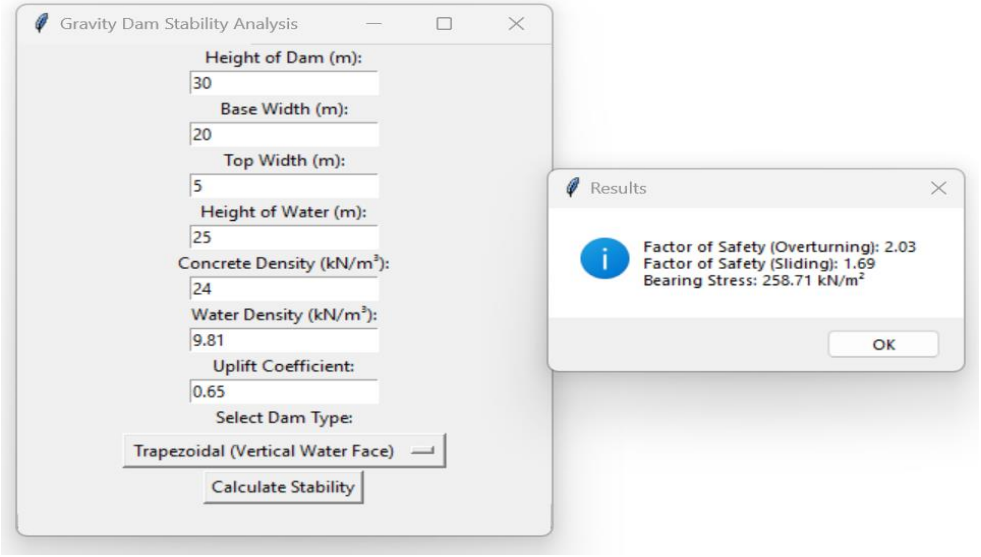

Figure 5 Results of the design of trapezoidal gravity dam with vertical waterface

iJETRM
International Journal of Engineering Technology Research & Management
Published By:
https://www.ijetrm.com/

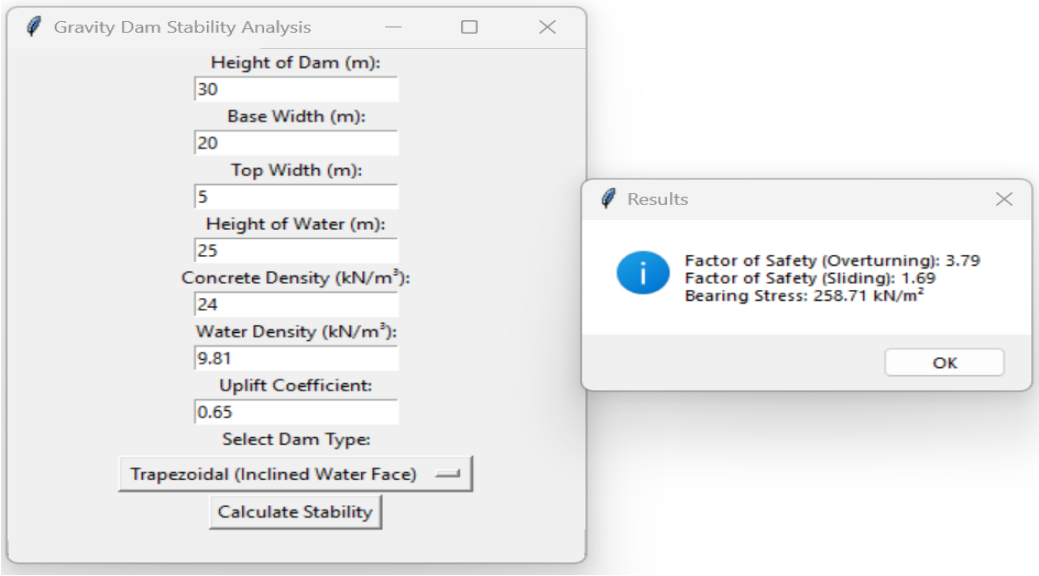

Figure 6 Results of the design of trapezoidal gravity dam with vertical waterface

CONCLUSION

The adoption of programming in the design of gravity dams enhances accuracy, efficiency, and repeatability. This study provides a computational framework using Python, demonstrating its application in stability assessment. Future work may integrate finite element analysis for more refined modeling and optimization. By leveraging programming-based automation, structural engineers can significantly improve design reliability while reducing manual errors (BS 6349-6:1989, Eurocode 7: EN 1997-1).

REFERENCES

[1] Chow, V.T., "Open-Channel Hydraulics," McGraw-Hill, 1959.
[2] Das, B.M., "Principles of Foundation Engineering," Cengage Learning, 2011.
[3] Bureau of Reclamation (USBR), "Design of Small Dams," 3rd Edition, U.S. Department of the Interior, 2012.
[4] Garg, S.K., "Irrigation Engineering and Hydraulic Structures," Khanna Publishers, 2009.
[5] Chanson, H., "Hydraulics of Open Channel Flow," 2nd Edition, Elsevier, 2015.
[6] Khurmi, R.S., Strength of Materials," S.Chand & Company, 2008.
[7] British Standards (BS) 6349-6:1989, "Maritime Structures – Code of Practice for Design of Quay Walls, Jetties and Dolphins."
[8] Eurocode 7: EN 1997-1, "Geotechnical Design – General Rules."