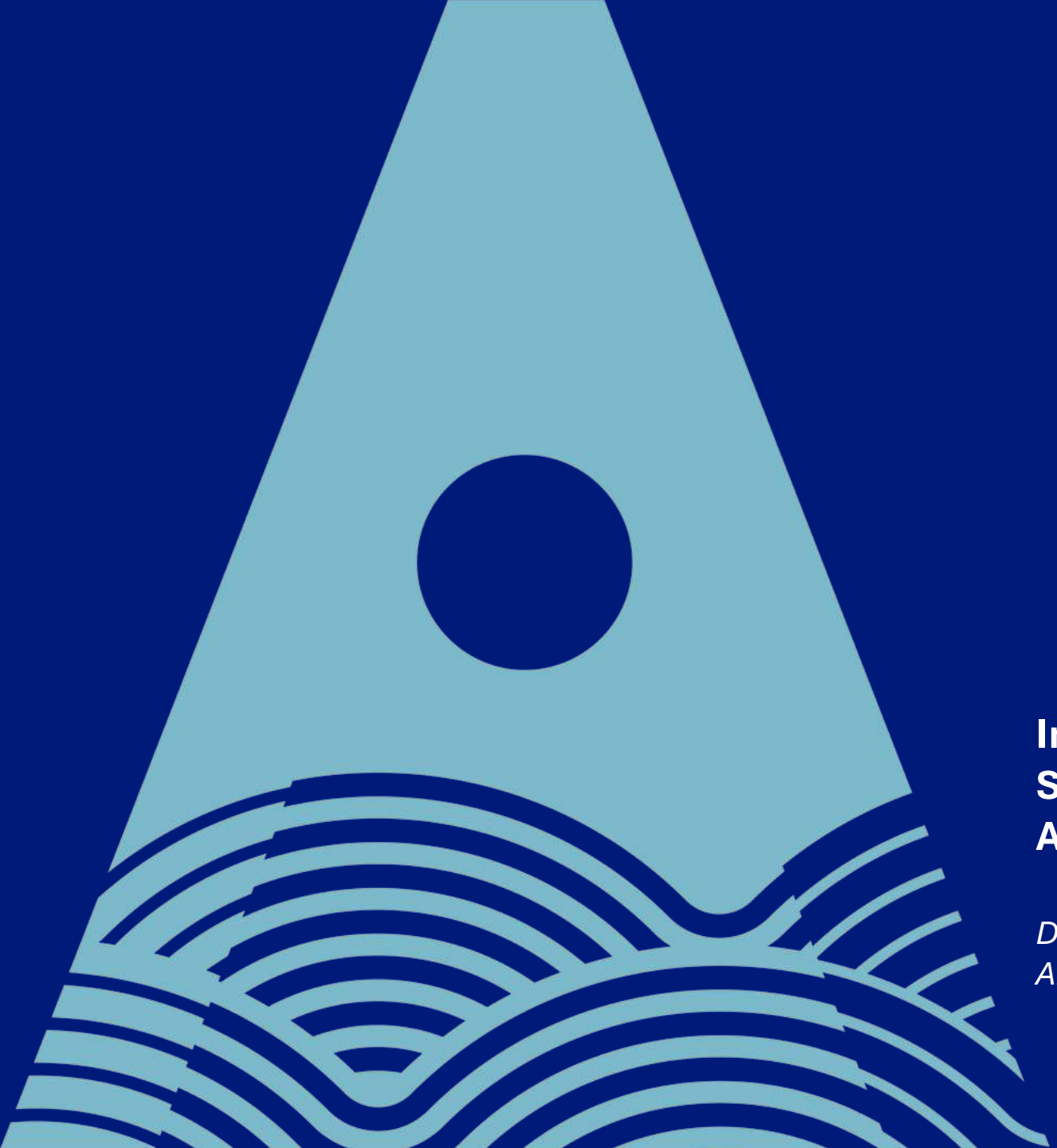


Ollscoil Teicneolaíochta an Atlantaigh

Atlantic Technological University

Interactive Python Notebooks: Scaffolding Mathematical Intuition Through AI-Assisted Design

Dr Fionn Downes
AL in Mathematics & Information Technology

Introduction

About Me

- AL in Mathematics & Information Technology
- Faculty of Science & Health, ATU Sligo

My Focus:

- Building inclusive, accessible STEM education.
- Supporting the next generation of STEM graduates.

The Challenge

- Equity Gap evident in first year STEM students.
- Functions as a Threshold concept in Mathematics

The Result

- Students with weaker preparation experience immediate maths anxiety, face higher failure risk, and disproportionately consume limited remedial resources.

Today I'll show you

- Interactive notebooks that let students **explore ideas through live manipulation**
- **Evidence of impact** – usage data, student voices, assessment gains
- An **AI-assisted workflow** that made sophisticated resources possible with minimal technical expertise
- A practical framework you can **adapt for your own discipline**

Threshold Concepts Exist in Every Discipline

Definition: A gateway idea that, once understood, transforms how you see a whole subject.

Discipline	The Threshold Concept	What Changes
English Literature	*"The author is not the narrator."*	Reading becomes analysis, not autobiography.
Business	*"Sunk costs are irrelevant to future decisions."*	Money already spent should not influence what you do next.
Art & Design	*"Negative space is as important as the object."*	What you leave out shapes meaning as much as what you put in.
Education	*"Teaching is not telling."*	A student can hear a perfect explanation and learn nothing.

Mathematics: *"An equation is not a rule to memorise. It is a relationship you can explore, predict, and see."*

Before: (foreign language)

$$y = 2(x - 3)^2 + 1$$

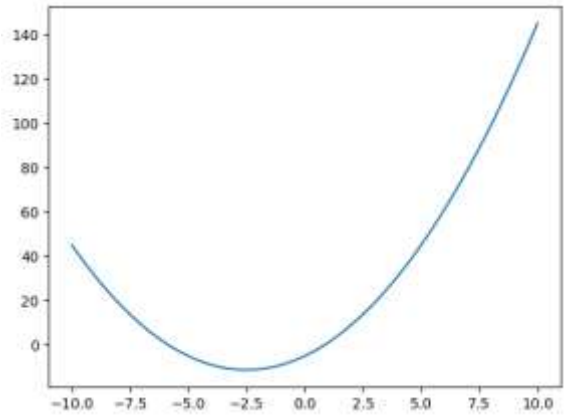
After: (sees the shape)

A parabola that stretches, shifts, and flips

Why Traditional Approaches Fall Short

Static graphs

No connection → no insight



No connection between numbers you change and shape you see

Symbols alone

Steps without meaning

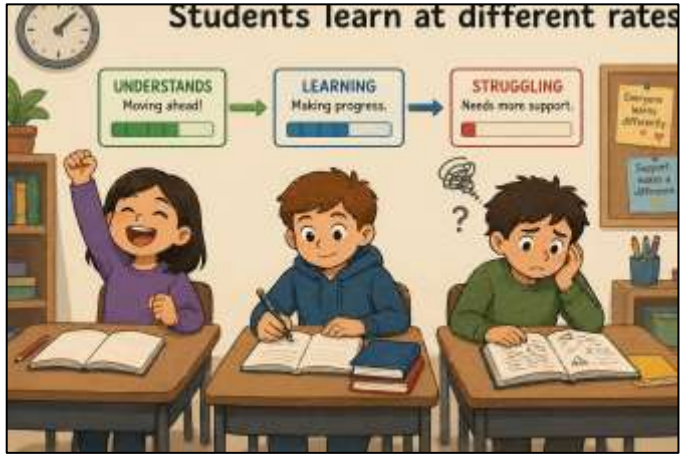
$$ax^2 + bx + c = 0$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Memorise steps without understanding *why*

One pace

Some wait. Some drown.



Doesn't work when backgrounds are diverse

Question: *How do we build intuition before abstraction?*

Static Plots – Isolating One Variable at a Time

What They Do:

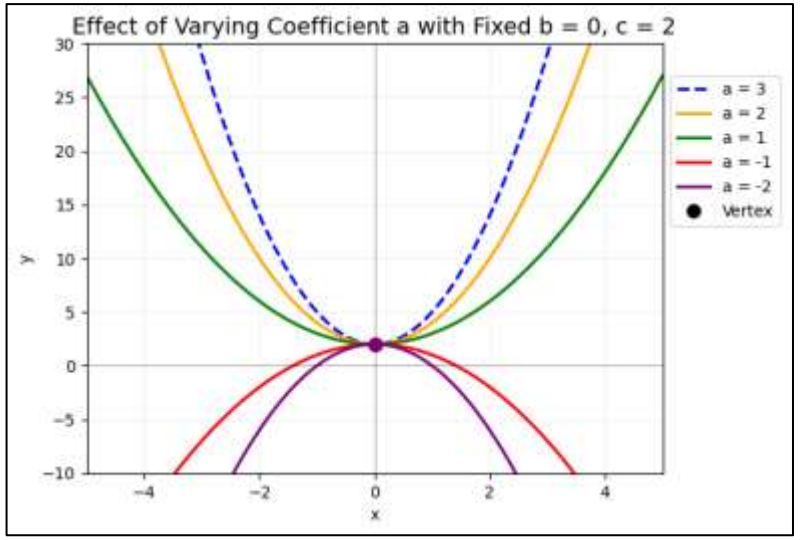
- Show multiple versions side-by-side
- Change **only one parameter** per comparison
- Remove motion – allow unhurried comparison

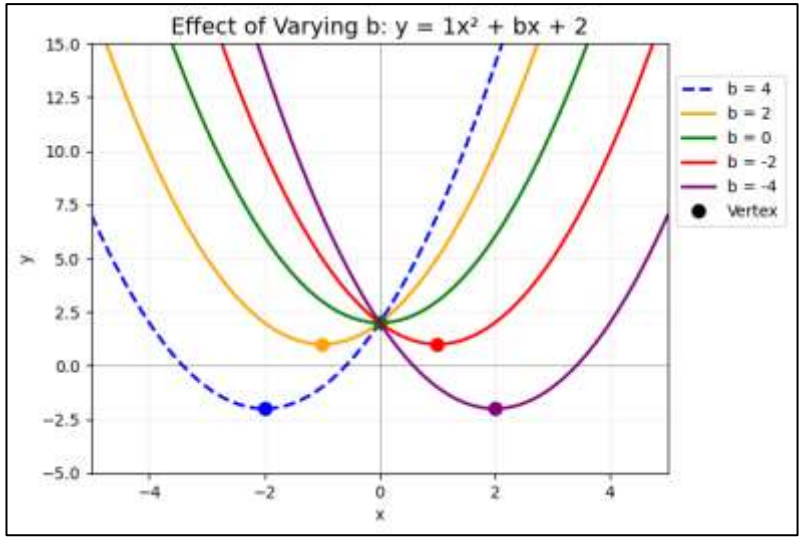
Vary One Parameter at a time:

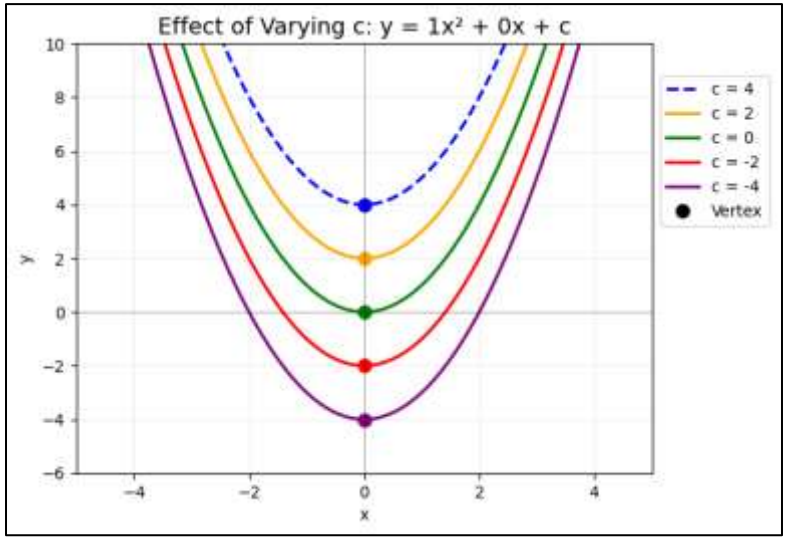
$$y = ax^2 + bx + c$$

Why it Helps Learning:

- Reduces cognitive load
- Supports *contrast* – students see the difference directly
- Works for **reflective learners** who process slowly







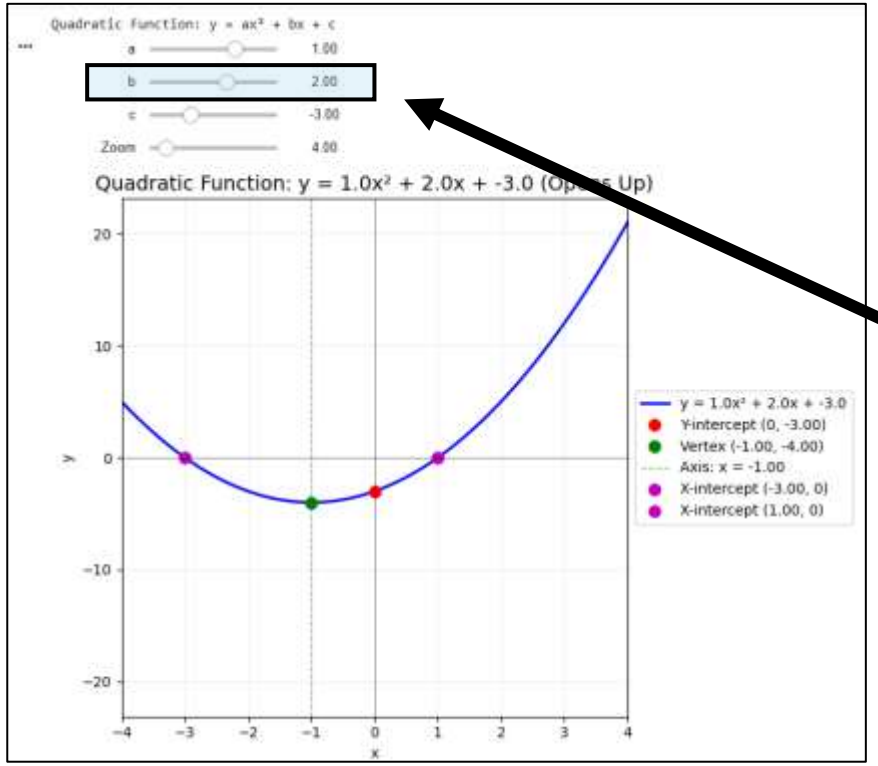
Interactive Sliders – Transforming Passive Viewing into Active Discovery

What They Do:

- Drag a slider → graph updates instantly
- Key features (roots, vertex, asymptotes) auto-calculated and highlighted

Why it Helps Learning:

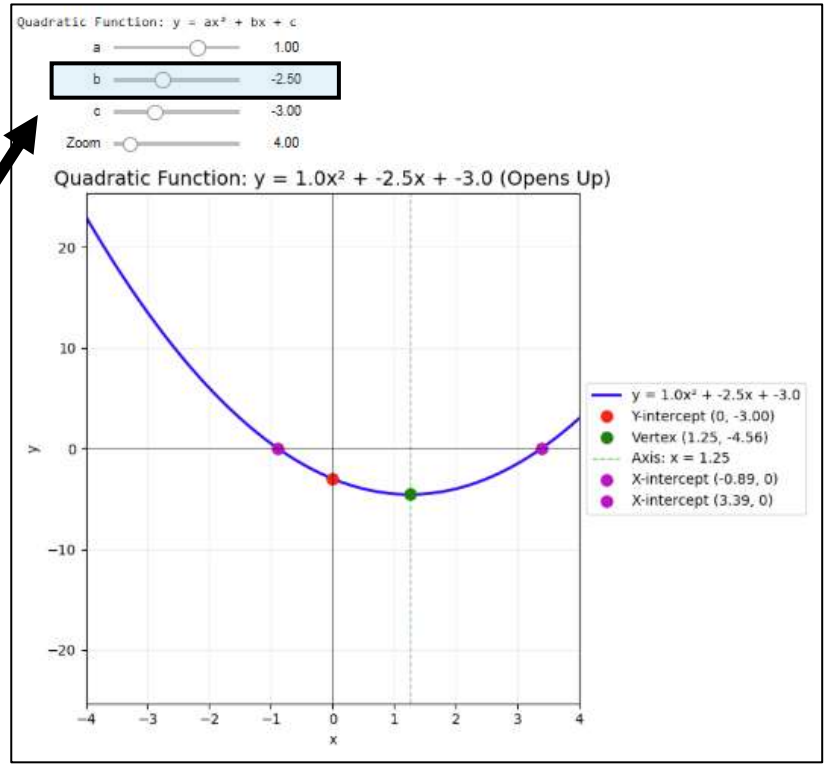
- Builds *causal intuition* – "when I move a, the graph stretches"
- Lowers cost of experimentation – try extreme values safely
- Works for **active learners** who learn by doing



Adjust slider

. . . . .

Plot Updates



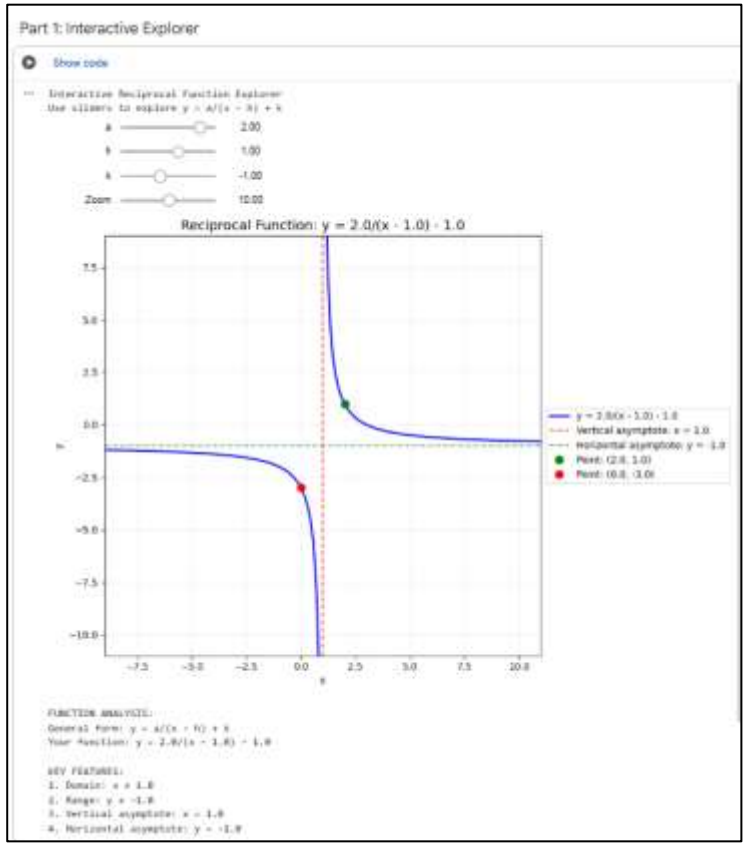
Design Features – Removing Barriers, Preserving Depth

Python Google Colab Notebooks

- **Google Colab** – completely free, no cost barrier for students
- **Python –** Language of Data Science, career relevant
- **9 interactive figures** – students adjust sliders and watch functions transform in real time
- **30 static plots** – isolate the effect of changing one variable at a time
- **Code hidden by default** – reduces intimidation
- **Code available with one click** – deeper exploration

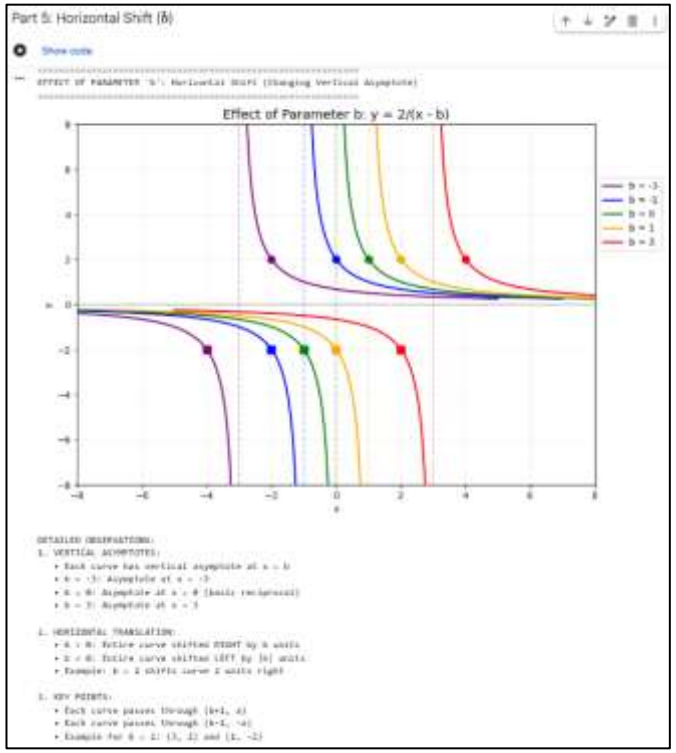
"We didn't remove the depth. We just made it optional."

Interactive Slider View



Scan to Explore this Notebook!

Static Plot View



Generative AI as Development Accelerator

Traditional Approach

- Limited by individual programming skill
- Weeks of manual coding
- Difficult to iterate and refine

AI-Assisted Approach

- Accessible to non-specialist educators
- Rapid prototyping with AI assistance
- Quick adjustments based on feedback

*This is a **Human-AI Collaborative Model***

Meyer, J. H. F., & Land, R. (2003),
Bennett, V., & Royce, C. A. (2025)

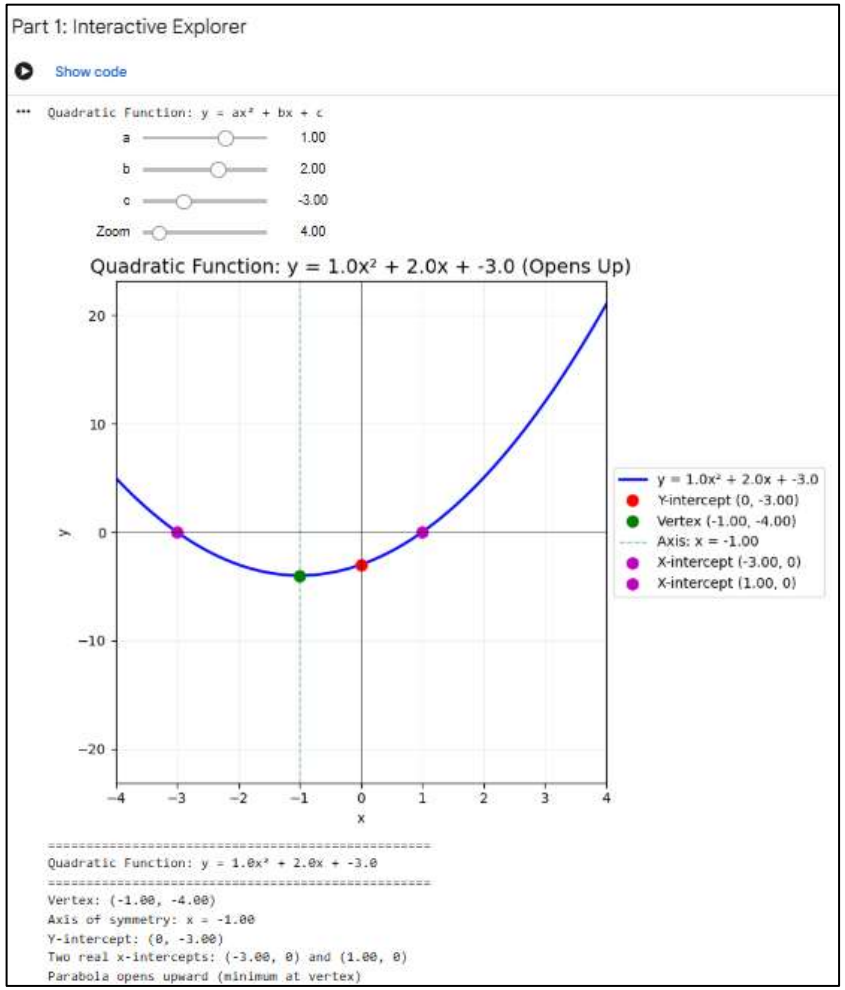
Evidence of Effectiveness

Initial Analytics

- Promising usage metrics from practical tutorial sessions
- Active engagement during mathematics journal completion

Integration

- Live demonstrations during tutorials
- Embedded in mathematics journal workflow
- Not bolt-on extras—core to learning



$y = 1x^2 + 2x - 3$???

"Now I can actually see what the numbers do."

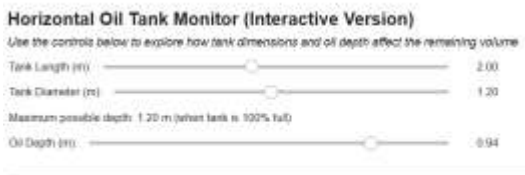
The Oil Tank Problem – Real-World Monitoring

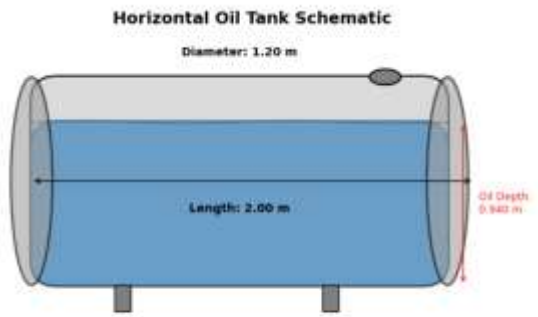
Tools: Measuring Tape, A Stick.

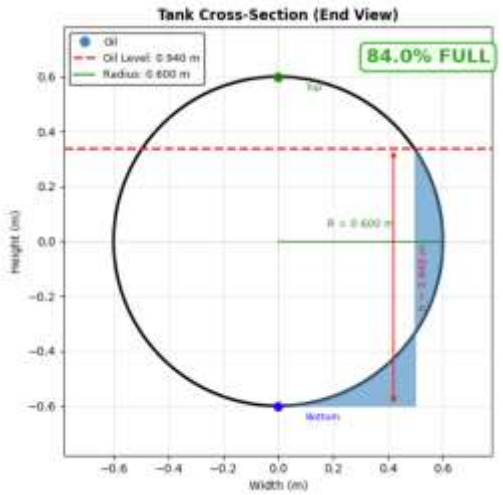
Goal: How much oil is there?

Process: Derivation, Implementation, Visualization.

Scan to use the Interactive Oil Tank Monitor







ATU Sligo - Maths Enrichment Programme:

- **Senior Cycle:** https://www.atu.ie/student-life/student-support/academic-supports/maths-enrichment-programme
- **Junior Cycle:** https://www.atu.ie/student-life/student-support/academic-supports/maths-enrichment-programme/junior-maths-enrichment-programme-sligo

More Information:

- **Downes, F. (2024)**, Physics Ed, Quantifying residual oil in a household oil tank: DOI: 10.1088/1361-6552/ad424d
- **Downes, F. (2026)**, Zendo, From Derivation to Dashboard: Applying Mathematical Modelling to Real-World Problems: DOI: 10.5281/zenodo.19665461

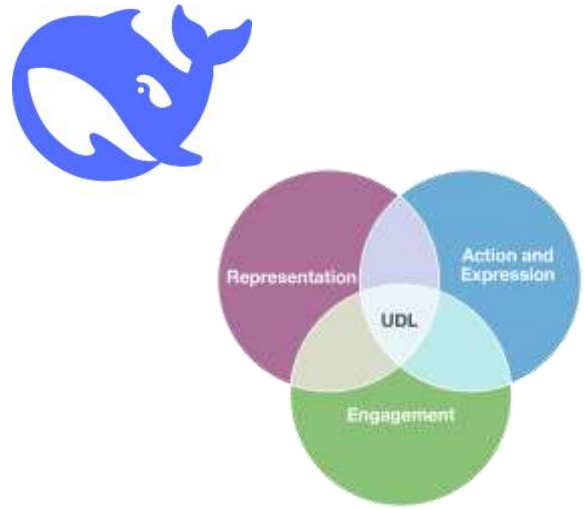
Summary – Innovation & Application

What We Showed

- Interactive Colab notebooks
- Features plot automatically
- Students see the maths happen

Why It's Innovative

- AI-accelerated development
- UDL - depth is optional
- Free + Open + Browser



What You Can Take Away

- Transferable framework
- Identify your skills gap
- Deliver accessibly

Ollscoil Teicneolaíochta an Atlantaigh

Atlantic Technological University

Additional Slides

Links To Google Python Colab Notebooks

Notebooks

- 1_Linear
- 2_Quadratic
- 3_Cubic
- 4_Reciprocal
- 5_Logarithmic
- 6_Exponential
- 7_Sin
- 8_Cos
- 9_Tan

Code Example: Interactive Tan Function

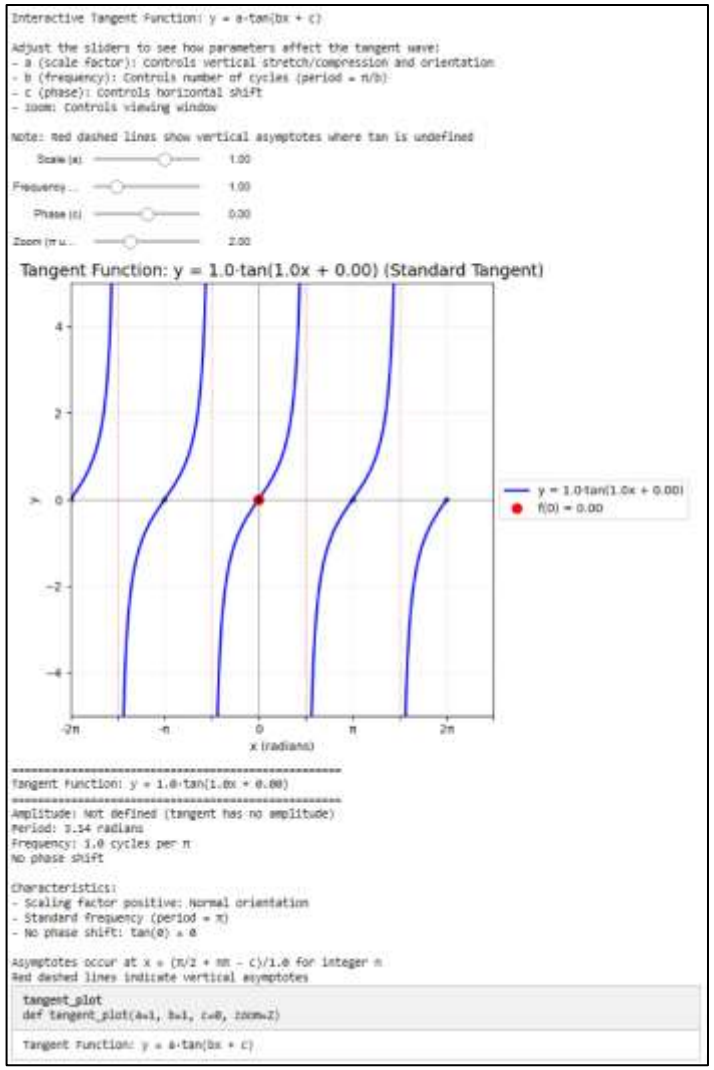
Part 1: Interactive Explorer

```
# @title Part 1: Interactive Explorer
def tangent_plot(a=1, b=1, c=0, zoom=2):
    """
    Tangent function: y = a·tan(bx + c)
    """
    # Setup plot
    plt.figure(figsize=(8, 6))

    # Create x values - need to avoid asymptotes
    x = np.linspace(-zoom*np.pi, zoom*np.pi, 2000)  # More points for better as

    # Calculate y, handling asymptotes
    with np.errstate(divide='ignore', invalid='ignore'):
        y = a * np.tan(b * x + c)

    # Plot the tangent curve with asymptote handling
    # Split at asymptotes to avoid connecting lines across them
    asymptotes = []
    if b != 0:
        # Asymptotes occur when bx + c = π/2 + nπ
        for n in range(-int(zoom*b*2)-2, int(zoom*b*2)+3):
            x_asy = (np.pi/2 + n*np.pi - c) / b
            if -zoom*np.pi <= x_asy <= zoom*np.pi:
                asymptotes.append(x_asy)
                plt.axvline(x=x_asy, color='red', linestyle='--', alpha=0.3, li

    # Plot the curve in segments between asymptotes
    asymptotes.sort()
    segments = [-zoom*np.pi] + asymptotes + [zoom*np.pi]

    for i in range(len(segments)-1):
        x_seg = np.linspace(segments[i], segments[i+1], 200)
        mask = (x_seg > segments[i]) & (x_seg < segments[i+1])
        x_seg = x_seg[mask]
        if len(x_seg) > 0:
            y_seg = a * np.tan(b * x_seg + c)
            plt.plot(x_seg, y_seg, 'b-', linewidth=2)

    # Add a single label for the legend
    plt.plot([], [], 'b-', linewidth=2, label=f'y = {a}·tan({b}x + {c:.2f})')

    # Mark key points
    # Mark y-intercept (x=0)
    y_intercept = a * np.tan(c)
    if abs(y_intercept) < 10:  # Only plot if not near asymptote
        plt.plot(0, y_intercept, 'ro', markersize=8, label=f'f(0) = {y_intercept
```

```
    # Mark zero crossings (tan = 0 when argument = nπ)
    if b != 0:
        for n in range(-int(zoom*b*2)-1, int(zoom*b*2)+2):
            x_zero = (n*np.pi - c) / b
            if -zoom*np.pi <= x_zero <= zoom*np.pi:
                # Check if not too close to an asymptote
                is_asymptote = False
                for x_asy in asymptotes:
                    if abs(x_zero - x_asy) < 0.01:
                        is_asymptote = True
                        break
                if not is_asymptote:
                    plt.plot(x_zero, 0, 'ko', markersize=4, alpha=0.5)

    # Basic formatting
    plt.axhline(y=0, color='k', linewidth=0.5, alpha=0.7)
    plt.axvline(x=0, color='k', linewidth=0.5, alpha=0.7)
    plt.grid(True, alpha=0.3)

    # Set zoom with appropriate y-limits for tangent
    plt.xlim(-zoom*np.pi, zoom*np.pi)
    plt.ylim(-5, 5)  # Fixed y-limits to avoid extreme values near asymptotes

    # Custom x-ticks with π labels
    xticks = np.arange(-zoom, zoom+1, 0.5) * np.pi
    xtick_labels = []
    for tick in xticks:
        if tick == 0:
            xtick_labels.append('0')
        elif abs(tick/np.pi - int(tick/np.pi)) < 0.01:
            n = int(round(tick/np.pi))
            if n == 1:
                xtick_labels.append('π')
            elif n == -1:
                xtick_labels.append('-π')
            else:
                xtick_labels.append(f'{n}π')
        else:
            xtick_labels.append('')
    plt.xticks(xticks, xtick_labels)

    # Title with function details
    title_str = f'Tangent Function: y = {a}·tan({b}x + {c:.2f})'
    if b == 1 and c == 0:
        title_str += ' (Standard Tangent)'
    elif a < 0:
        title_str += ' (Inverted)'
    plt.title(title_str, fontsize=14)

    plt.xlabel('x (radians)')
    plt.ylabel('y')
```

```
    # Add legend
    plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))

    plt.tight_layout()
    plt.show()

    # Output information
    print("="*50)
    print(f"Tangent Function: y = {a}·tan({b}x + {c:.2f})")
    print("="*50)

    print(f"Amplitude: Not defined (tangent has no amplitude)")
    print(f"Period: {np.pi/abs(b):.2f} radians")
    print(f"Frequency: {abs(b):.2f} cycles per π")
    print(f"Phase shift: {-c/b:.2f} radians (to the right)" if c != 0 else "No phase shift")

    # Determine special cases
    print("\nCharacteristics:")
    if a > 0:
        print("- Scaling factor positive: Normal orientation")
    elif a < 0:
        print("- Scaling factor negative: Inverted orientation")

    if b == 1:
        print("- Standard frequency (period = π)")
    elif b > 1:
        print(f"- Higher frequency: {b} times faster than standard")
    elif b < 1:
        print(f"- Lower frequency: {1/b} times slower than standard")

    if c == 0:
        print("- No phase shift: tan(0) = 0")
    elif c == np.pi/4:
        print("- Phase shift of π/4: tan(x + π/4) = (1 + tan x)/(1 - tan x)")

    # Show asymptote information
    print(f"\nAsymptotes occur at x = (π/2 + nπ - c)/{b} for integer n")
    print("Red dashed lines indicate vertical asymptotes")

print("Interactive Tangent Function: y = a·tan(bx + c)")
print("\nAdjust the sliders to see how parameters affect the tangent wave:")
print("- a (scale factor): Controls vertical stretch/compression and orientation")
print("- b (frequency): Controls number of cycles (period = π/b)")
print("- c (phase): Controls horizontal shift")
print("- zoom: Controls viewing window")
print("\nNote: Red dashed lines show vertical asymptotes where tan is undefined")

interact(tangent_plot,
         a=widgets.FloatSlider(min=-3, max=3, step=0.1, value=1, description='Scale (a)'),
         b=widgets.FloatSlider(min=0.25, max=4, step=0.25, value=1, description='Frequency (b)'),
         c=widgets.FloatSlider(min=-np.pi, max=np.pi, step=np.pi/4, value=0, description='Phase (c)'),
         zoom=widgets.FloatSlider(min=1, max=4, step=0.5, value=2, description='Zoom (π units)'))
```

Interactive Tan Explorer Syntax



Output: Interactive Tan function with adjustable parameters.

References

- Bennett, V., & Royce, C. A. (2025). AI Meets Three-Dimensional Learning: Guiding Scientific Thinking With New Features of NotebookLM. NSTA Blog.
- Downes, F. (2024), Physics Ed, Quantifying residual oil in a household oil tank: DOI: 10.1088/1361-6552/ad424d
- Downes, F. (2026), Zendo, From Derivation to Dashboard: Applying Mathematical Modelling to Real-World Problems: DOI: 10.5281/zenodo.19665461
- Meyer, J. H. F., & Land, R. (2003). Threshold concepts and troublesome knowledge: Linkages to ways of thinking and practising. In C. Rust (Ed.), Improving Student Learning: Theory and Practice – 10 Years On (pp. 412–424). Oxford: Oxford Centre for Staff and Learning Development.
- Vygotsky, L. S. (1978). Mind in society: The development of higher psychological processes. Cambridge, MA: Harvard University Press.