ECEN 377: Engineering Applications of AI

Dr. Mahmoud Nabil Mahmoud
mnmahmoud@ncat.edu

North Carolina A & T State University

September 4, 2024

Maximizing Tax Revenue

Imagine that you're elected prime minister of a small country. You have ambitious goals, but you don't feel like you have the budget to achieve them.

- Your first order of business is to maximize the tax revenues your government brings in.
- It's not obvious what taxation rate you should choose to maximize revenues.

Challenges in Setting Tax Rates

- If your tax rate is 0 percent, you will get zero revenue.
- At 100 percent, taxpayers would likely avoid productive activity, leading to near-zero revenue.
- Optimizing revenue requires balancing between:
  - Rates that are too high and discourage productive activity.
  - Rates that are too low and undercollect revenue.
- To achieve this balance, you need to understand how tax rates relate to revenue.

Steps in the Right Direction

Your team of economists returns after research, using various methods, to determine the relationship between tax rates and revenues.

They've derived a function that relates taxation rate to revenue and even provided it in Python.

The function might look like this:

Code:

```
import math
def revenue(tax):
    return(100 * (math.log(tax+1) - (tax - 0.2)**2 + 0.04))
```

Tax Revenue

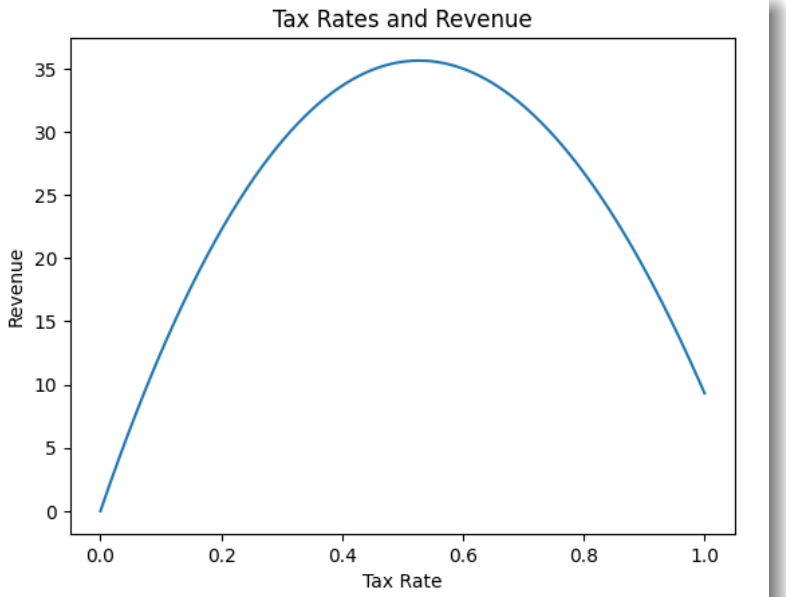
Plotting the Tax vs Revenue

- zero tax means zero revenue
- 100 percent tax means zero revenue as well

Code:

```
import matplotlib.pyplot as plt
xs = [x/1000 for x in range(1001)]
ys = [revenue(x) for x in xs]
plt.plot(xs,ys)
plt.title('Tax Rates and Revenue')
plt.xlabel('Tax Rate')
plt.ylabel('Revenue')
plt.show()
```

Output:



Tax Revenue

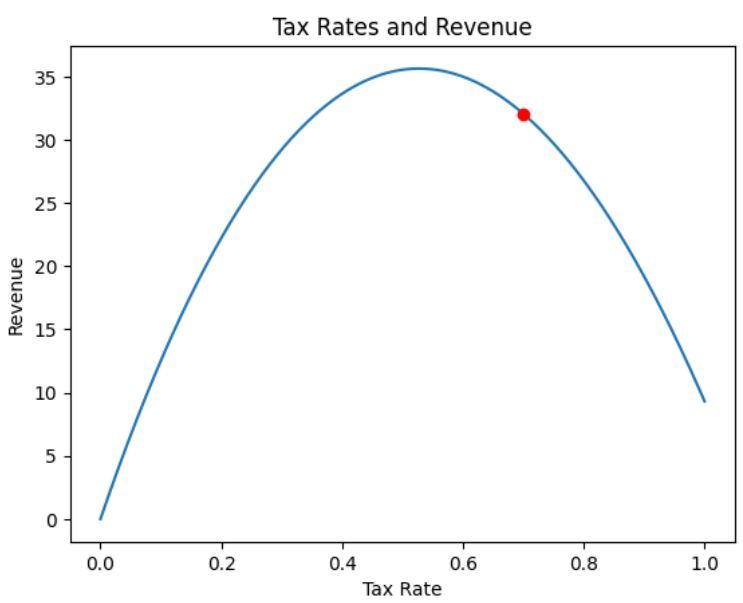
Plotting the Tax vs Revenue

- If your country currently has a flat 70% tax on income, we can plot that point on the curve by adding these lines revenue as well

Code:

```
import matplotlib.pyplot as plt
xs = [x/1000 for x in range(1001)]
ys = [revenue(x) for x in xs]
plt.plot(xs,ys)
current_rate = 0.7
plt.plot(current_rate,revenue(current_rate
plt.title('Tax Rates and Revenue')
plt.xlabel('Tax Rate')
plt.ylabel('Revenue')
plt.show()
```

Output:



Optimizing the Tax Rate

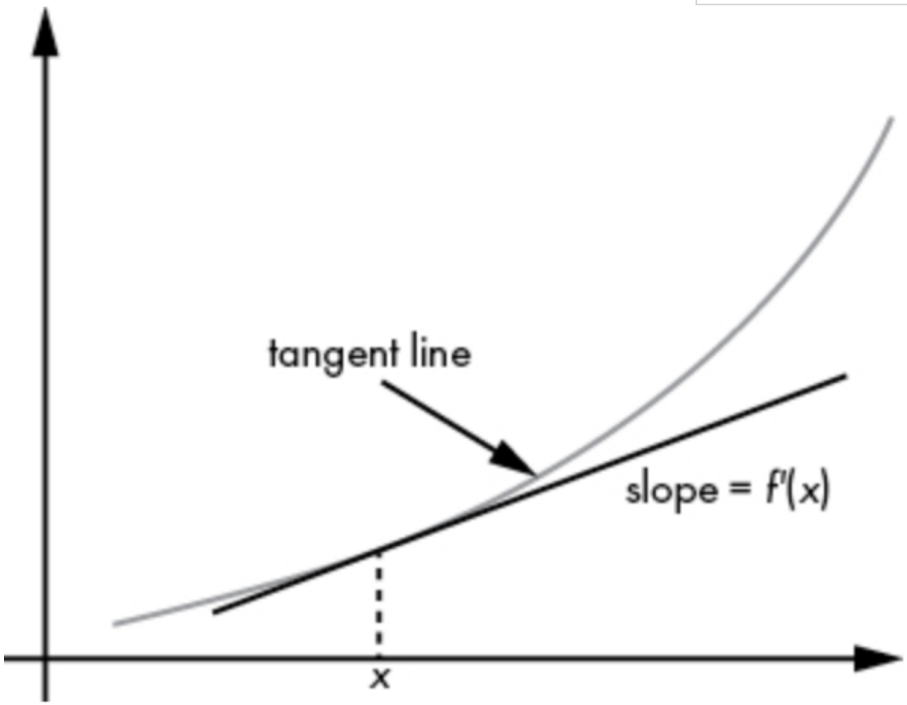
Your country's current 70% tax rate is not maximizing revenue according to the economists' formula.

- A visual inspection of the plot shows that a lower tax rate might increase revenues.
- You seek a precise figure for the optimal tax rate, not just an approximation.

The curve suggests that decreasing the tax rate could increase revenue. To confirm this, we can formally analyze the derivative of the revenue formula.

A derivative measures the slope of the tangent line—steepness is indicated by large values, and a downward trend by negative values.

Derivative

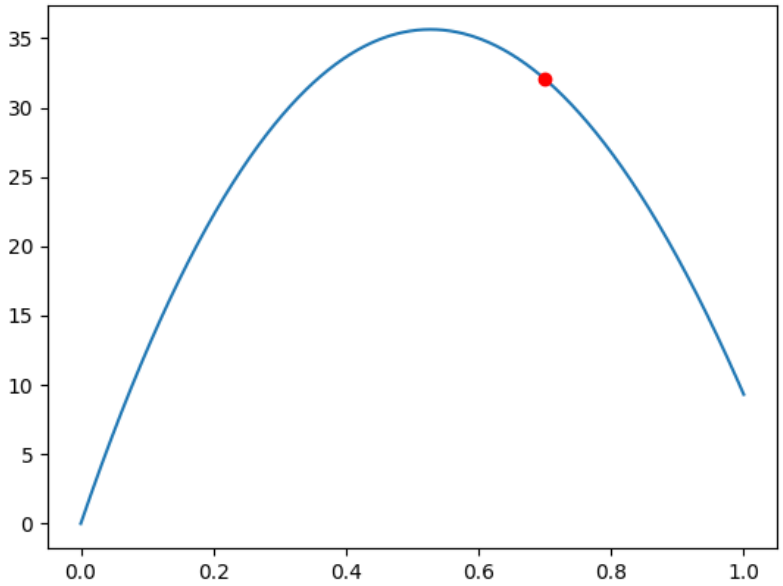


A derivative measures the slope of the tangent. We approximate the derivative using:

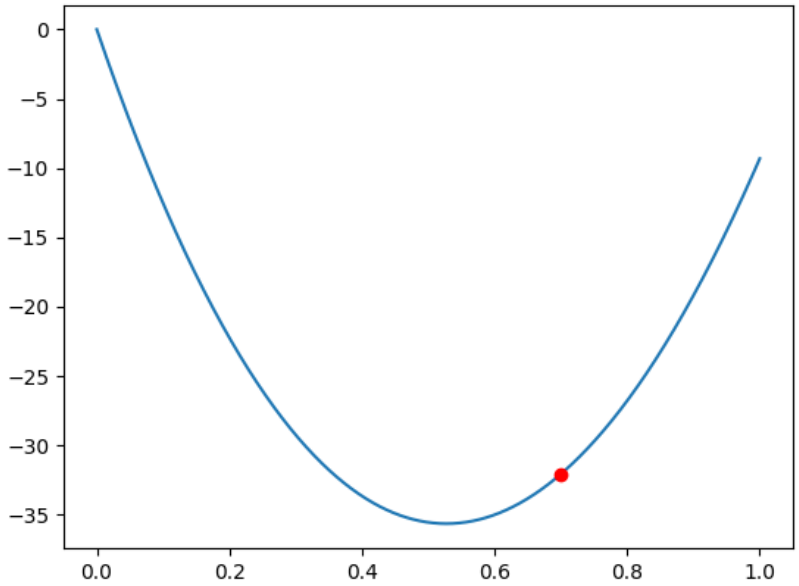
$$f'(x) \approx \frac{f(x+\delta) - f(x-\delta)}{2\delta}$$

where δ is a small increment.

What is the derivative sign here (at the red point)?

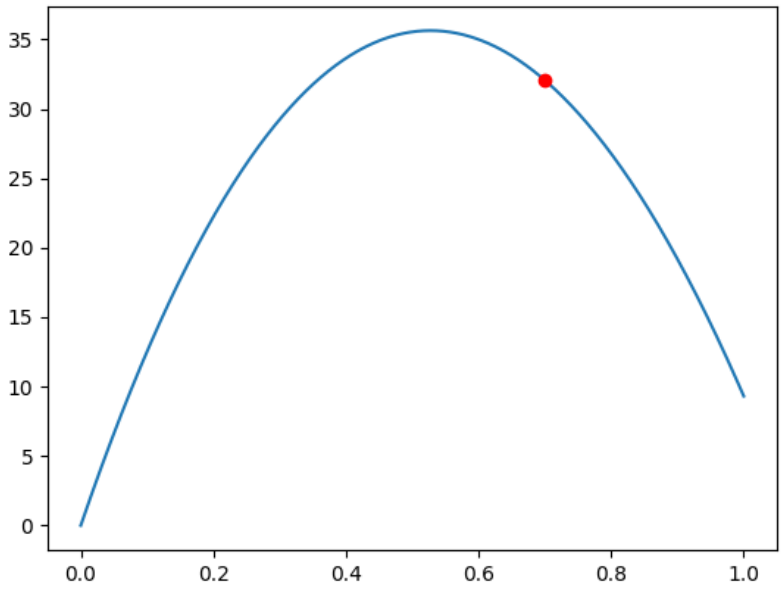


-ve derivative



+ve derivative

Using calculus rules to verify




```
def revenue_derivative(tax):
    return(100 * (1/(tax + 1) - 2
     ↪  * (tax - 0.2)))

print(revenue_derivative(0.7))

-41.17647058823528
```

Step Towards Maximizing Revenue

A negative derivative means that an increase in tax rate leads to a decrease in revenue. Conversely, a decrease in tax rate should lead to an increase in revenue.

While the precise tax rate for maximum revenue is not yet known, we can be sure that taking a small step in the direction of decreased taxation should increase revenue.

```
current_rate = current_rate + step_size * revenue_deriv(current_rate)
print(current_rate)
print(revenue(current_rate))

0.5711139389253913
35.40593273476592
```

Step Towards Maximizing Revenue

Running this code again

```
current_rate = current_rate + step_size * revenue_deriv(current_rate)
print(current_rate)
print(revenue(current_rate))

0.5282576597923916
35.63752108348475
```

Turning the Steps into an Algorithm (Gradient Ascent)

To maximize revenue, follow these steps:

1. Start with a `current_rate` and a `step_size`.
2. Calculate the derivative of the revenue function at the `current_rate`.
3. Update `current_rate` by adding `step_size` multiplied by the revenue derivative.
4. Repeat steps 2 and 3.

The only thing missing is a rule to stop when we are sufficiently close to the maximum.

We can specify a threshold to determine when changes are too small.

Gradient Ascent Update Rule

$$x_{new} = x_{current} + \text{step\_size} \times \nabla f(x_{current})$$

- step_size is a small number that determines the size of the step.
- $x\_current$ is the current value of the variable we are optimizing.
- $f(x\_current)$ is the value of the function at the current point.
- $\nabla f(x\_current)$ is the gradient of the function at the current point.
  - If the gradient is positive, the function is increasing, so we want to move in the positive direction.
  - If the gradient is negative, the function is decreasing, so we want to move in the negative direction.

Gradient Ascent in Python

```
threshold = 0.0001
maximum_iterations = 100000

keep_going = True
iterations = 0
while(keep_going):
    rate_change = step_size * revenue_derivative(current_rate)
    current_rate = current_rate + rate_change

    if(abs(rate_change) < threshold):
        keep_going = False

    if(iterations >= maximum_iterations):
        keep_going = False

    iterations = iterations+1
```

Objections to Gradient Ascent

Gradient ascent has practical applications for maximizing revenues, but there are objections to using it:

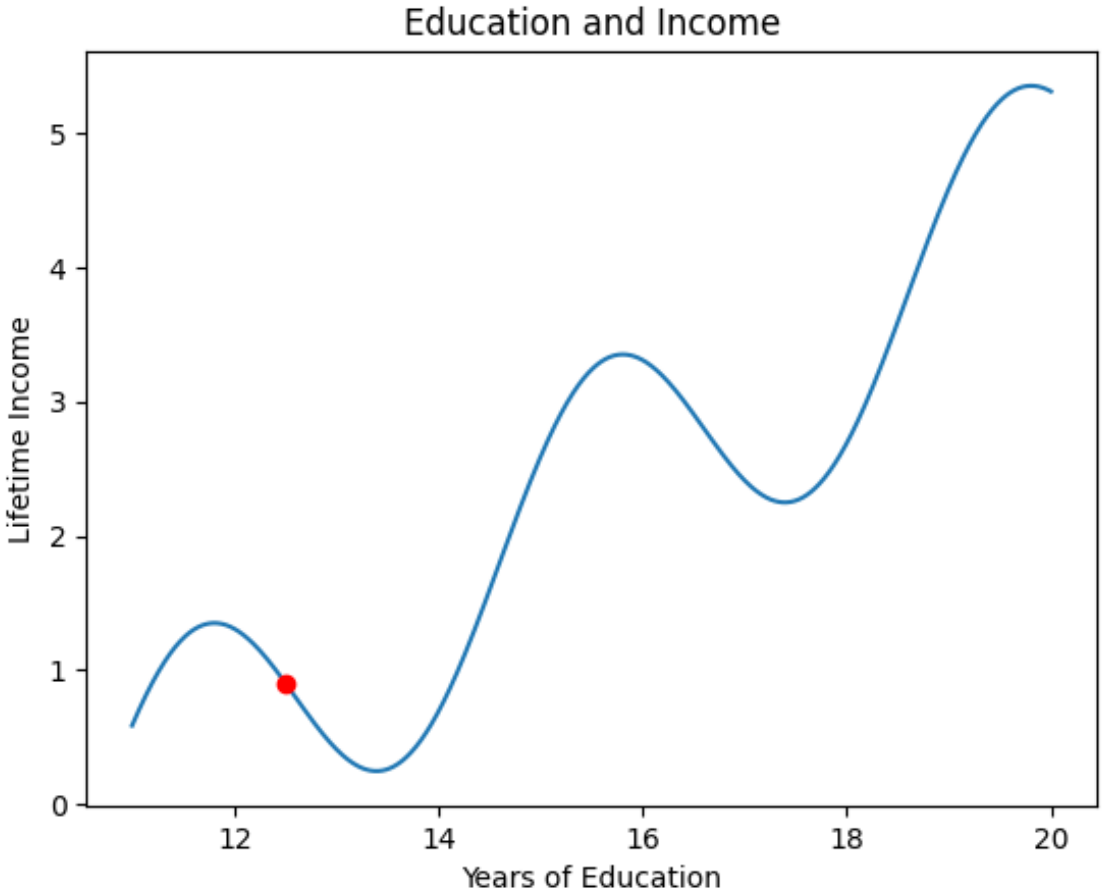
- **Visual Inspection:** It's argued that a simple visual inspection can be enough to find the maximum. Is that true??
- **Guess-and-Check:** Some prefer a repeated guess-and-check strategy to determine the maximum. Is that better??
- **First-Order Conditions:** Others believe that solving the first-order conditions directly is a more straightforward approach. Is that true??

The Problem of Local Extrema

Algorithms like gradient ascent may face local extrema:

- **Local Peak:** Higher than immediate surroundings, but not the global maximum.
- **Mountain Analogy:** Reaching a foothill summit might require descending to find a higher peak.
- **Naive Strategy:** May miss the global maximum by only moving locally.

Education and Income



Gradient ascent is limited by its local perspective. No transition to higher peaks by temporarily descending. Real-life analogies people quitting education for immediate earnings without considering long-term benefits.

From Maximization to Minimization

To switch from maximizing to minimizing a function, we can do either

- Invert Function: Minimize $-f(x)$ if you want to use the same maximization algorithm.
- Algorithm Adjustment: modify the Update rule as follows

Example: To minimize a function, apply gradient descent:

$$x_{new} = x_{current} - \text{step\_size} \times \nabla f(x_{current})$$

Exercise

1. If a function has a known global maximum, describe how you could verify that a solution found by gradient ascent is correct.
2. How would you determine the appropriate step size for gradient ascent to ensure that the algorithm converges to the maximum?

Questions