The Algorithms of Creating Money and their Impacts on Inflation and Wealth Distribution

scuti

2025/11/12

If you had a chance to start your own parallel economy, which **algorithmic supply model** would you chose?

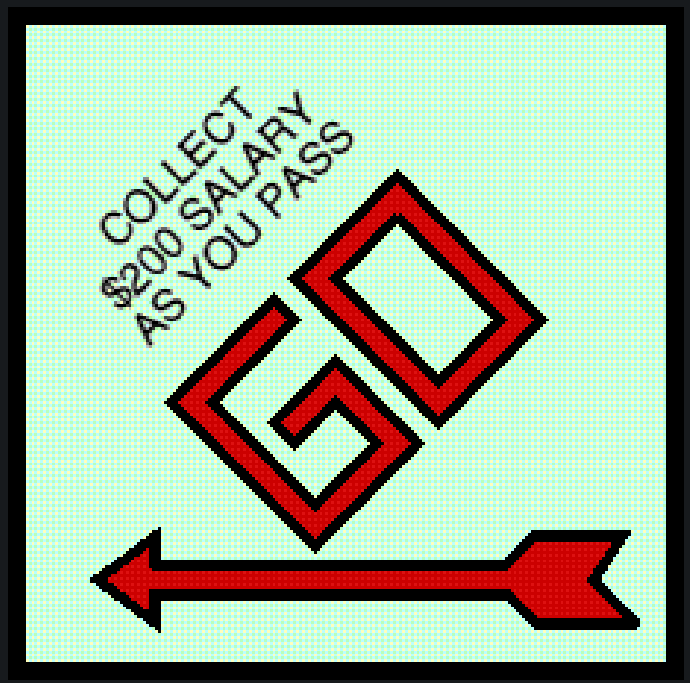
What is an ‘algorithmic supply model’?

algorithmic supply model — the means by which a financial instrument such as currency is created.

- the **monetary rule** in classical economics
- the **drop algorithm** in video gaming.
- **issuance algorithm** or **emission schedule** in cryptocurrencies
  - proof-of-work mining and halving
  - staking, (delegated) proof-of-stake

A Supply Model in Board Games

Players receive $200 when they complete a lap around the board in Monopoly.



Supply Models in Video Gaming

Weighted RNG — players have a chance of acquiring currency (gold, credits) or resources for questing and adventuring.

Log-in Reward — players receive an item once per day by logging in.

Dailies — players can complete a set of repeatable quests once each day; the number of quests and the reward is typically fixed.

- Gold, the Isle of Quel'Danas in *The Burning Crusade*

Real Money Transaction (RMT) — currency is created when a player performs a real-money transaction.

- PLEX in *EVE Online*, Platinum in *Warframe*

So what's the supply model of the dollar ($)?

fractional reserve banking – money is created by taking out a loan from a bank. Money is likewise destroyed when the loan is repaid.

- The central bank determines the interest rate.
- Interest accrues on the principal given by the **annual percentage rate** (APR).

What Game Economies Never Do

Very few (if any at all) online games or space sims implement fractional reserve banking or facilitate lending between players.

- no game mechanic or interface that sets up loan payment schedules or garnishes someone's quest rewards.
- the closest I found was a single player game with a starting scenario of taking out a mortgage for a spaceship at 4% APR.
  - Endless Sky by Michael Zahniser
  - optimal strategy was to pay it off asap
- *Let's see the results (inflation and wealth distribution) of fractional reserve banking. . .*

Simulating Compounding Interest

```
def compounding_interest(balances, rate=0.05, terms=3):
    '''
    balances : list of numerical values where
    the index correlates to a participant.
    '''
    if terms <= 0:
        print("Number of terms must be > 0!")
        return
    new_bal = []
    for balance in balances:
        b = balance
        for i in range(0,terms):
            b = b + (b * rate)
        new_bal.append(b)
    return new_bal
```

Simple Demo Results: Wealth Distribution

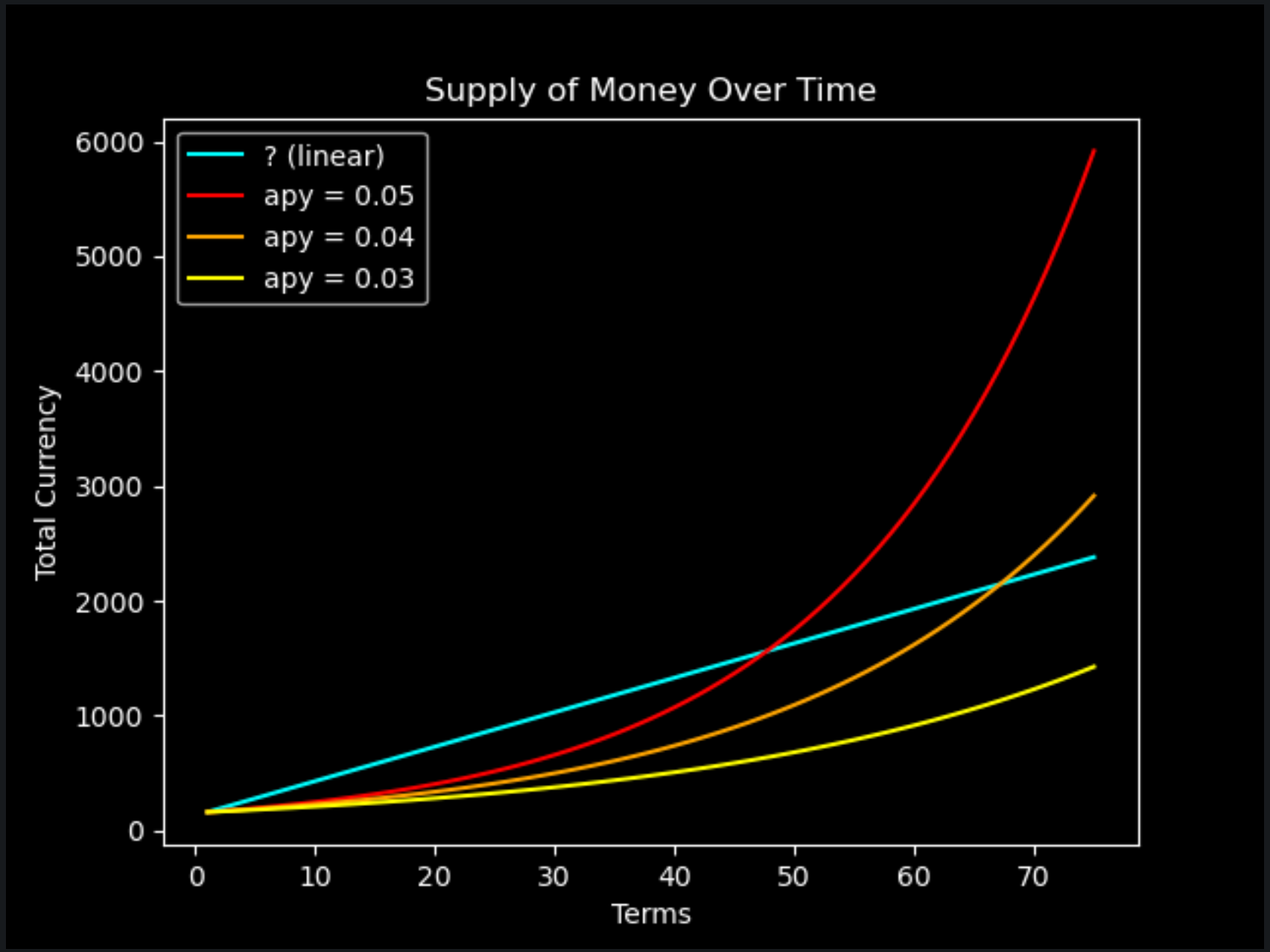
Just to see how compounding interest influences share of wealth, everyone here is a creditor that earns 5% APY and there are no bankers or debtors.

```
participants = ["Alice", "Bob", "Charlie"]
balances = [100,40,20]
```

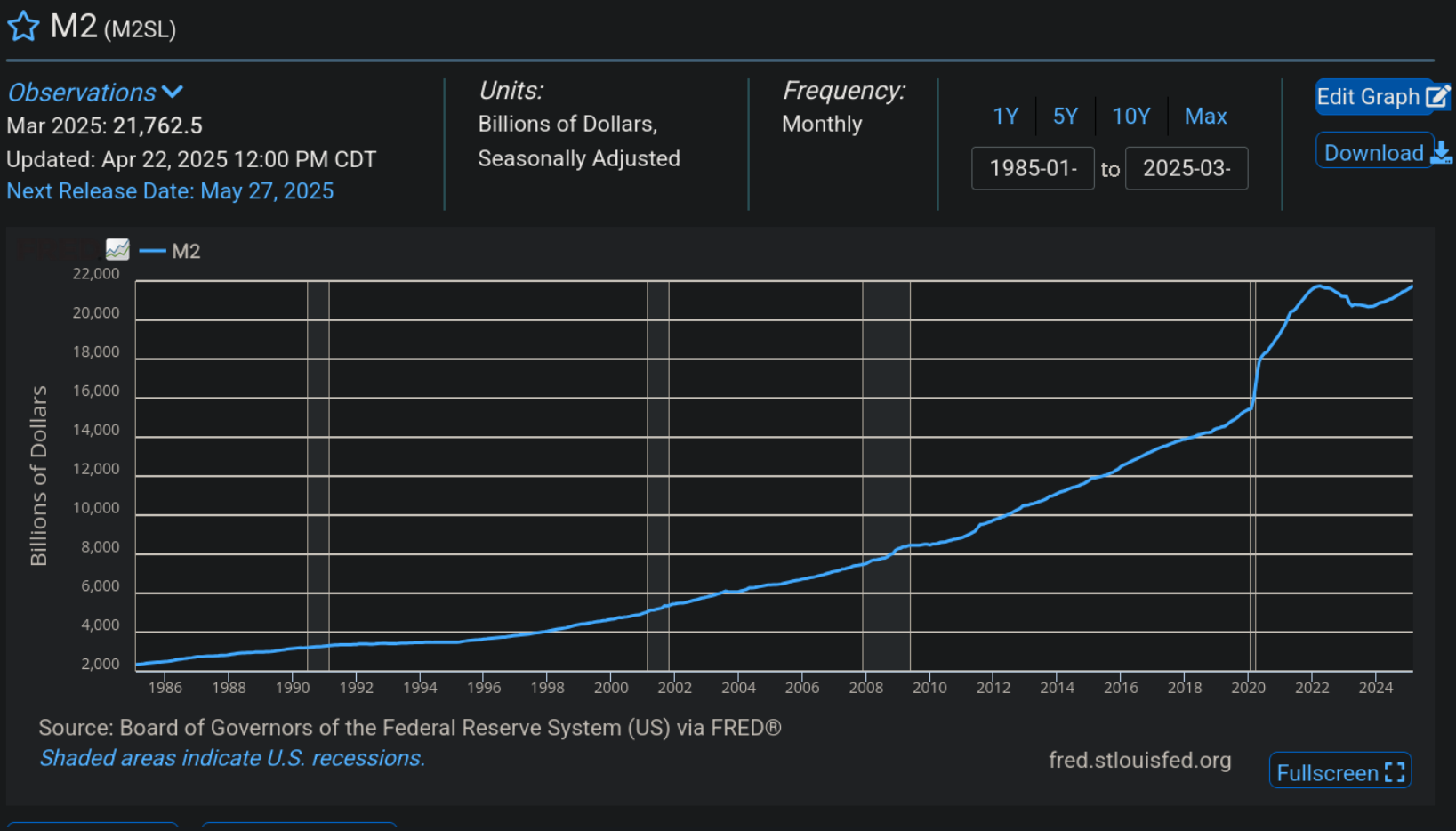
#	name	initial	1	2	3	share of wealth
0	Alice	100	105.0	110.25	115.7625	62.5
1	Bob	40	42.0	44.10	46.3050	25.0
2	Charlie	20	21.0	22.05	23.1525	12.5

- Here, the wealth distribution does not change.
- Each term Alice earns $5; Bob: $2; and Charlie: $1.
- *There is a linear relationship between the balance and the dividend.*

Simple Demo Results: Inflation



IRL Inflation of Δ 40 years



Back to Monopoly

Every player receives a constant amount of money each time they complete a lap around the board.

In Monopoly, in-game time is abstracted from dice rolls.

Even if a player gets lucky by rolling 6, with the **Law of Large Numbers** and a fair d6, repeated rolls will converge to the true value of 3.5.

$$\frac{1+2+3+4+5+6}{6} = 3.5$$

Simulating Game Currencies

```
def constant_dividend(balances, dividend=10, terms=3):
    '''
    dividend : numeric, constant value added
    to balances each term.
    '''
    new_bal = []
    for balance in balances:
        b = balance
        for i in range(0,terms):
            b = b + dividend
        new_bal.append(b)
    return new_bal
```

Simple Demo Results

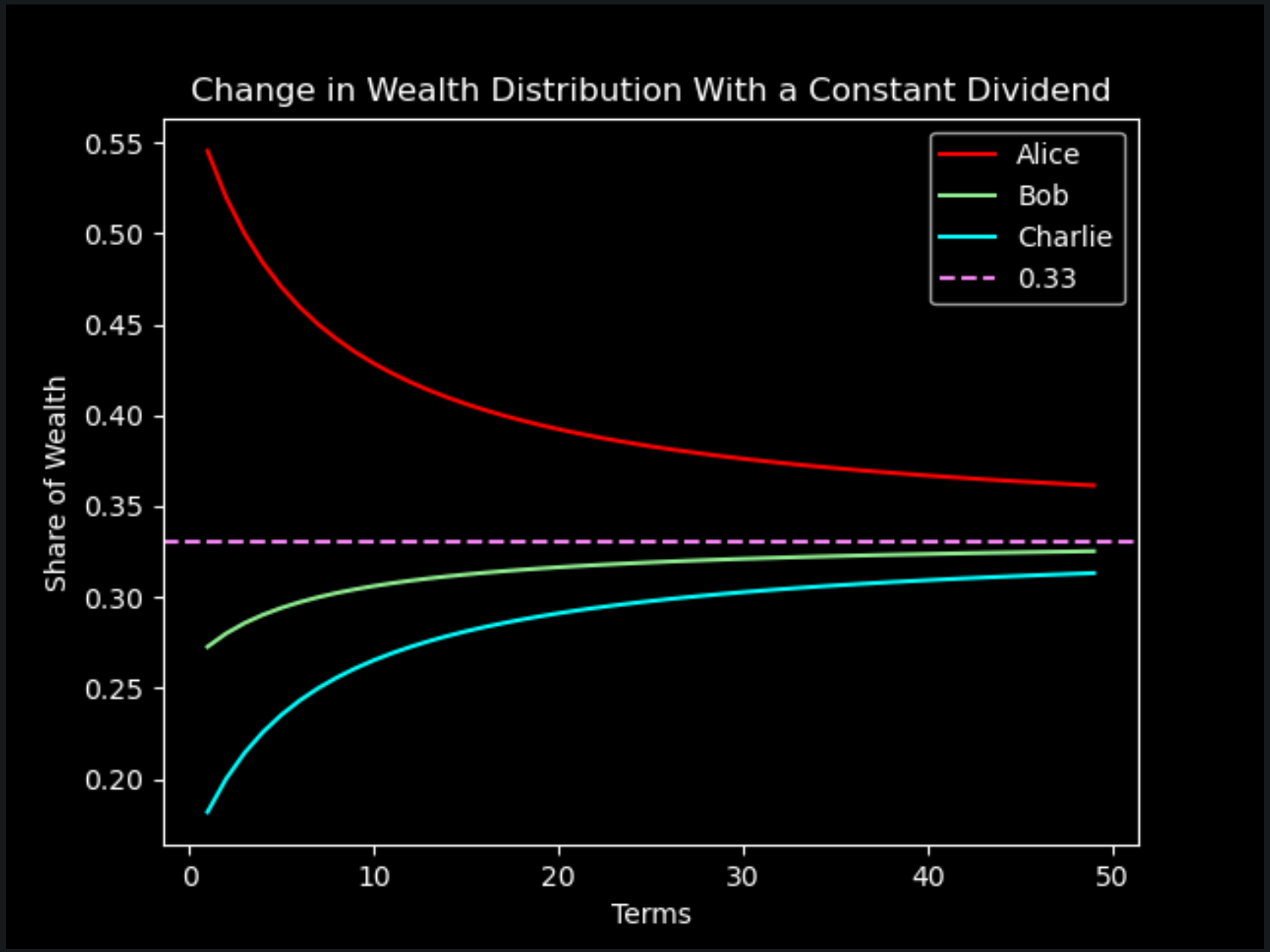
If this was a game of Monopoly, Alice starts with an advantage. Everyone earns a dividend of 10 each term.

```
participants = ["Alice", "Bob", "Charlie"]
balances = [100,40,20]
```

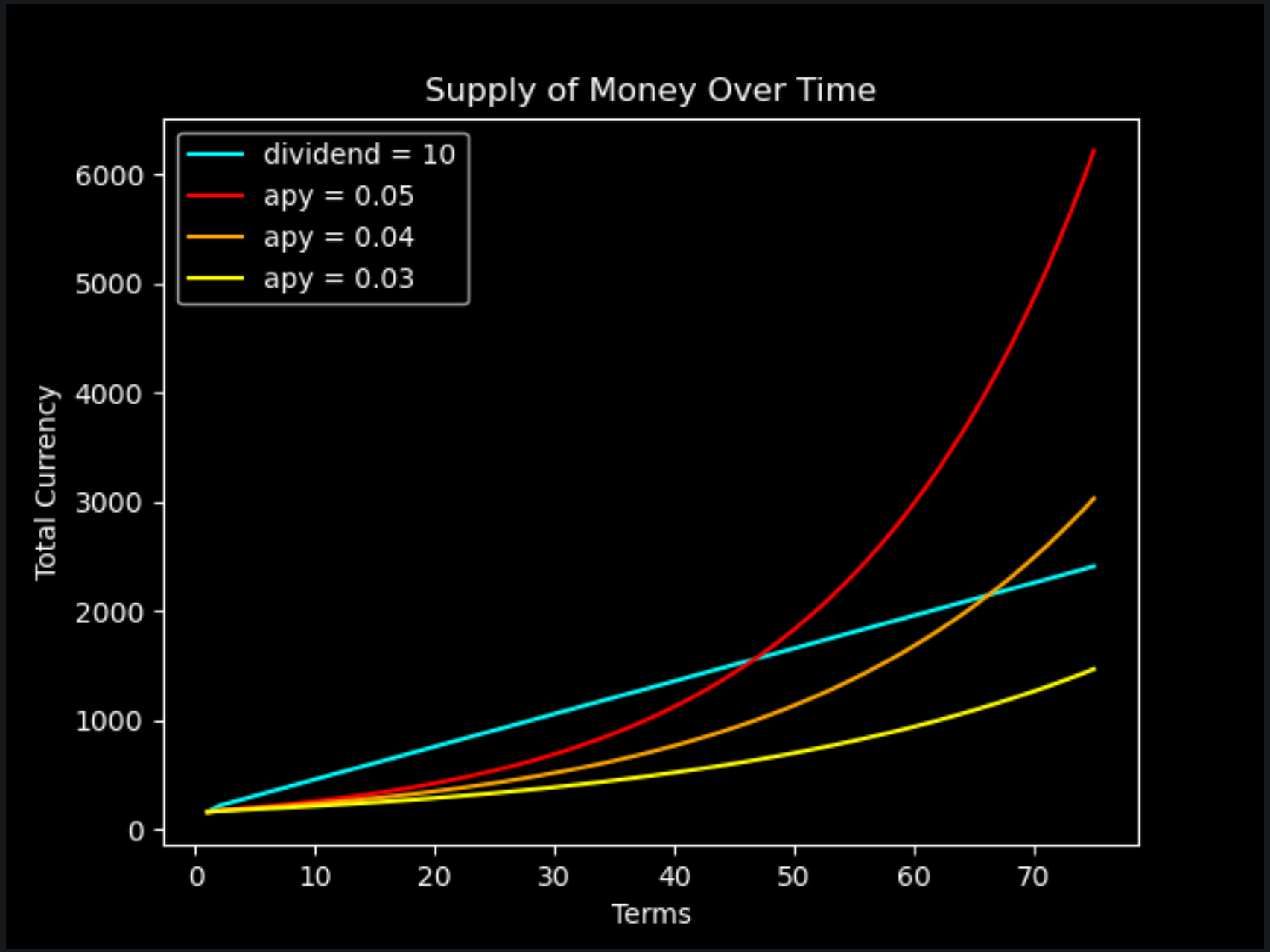
#	name	initial	share of wealth	1	2	3
0	Alice	100	62.5	110	120	130
1	Bob	40	25	50	60	70
2	Charlie	20	12.5	30	40	50

What is the share of wealth after n turns?

A trend to 1/n where n = number of participants



Inflation: it was the cyan line from before



Inflation with different constants

How about a logarithmic function?

```
def mining(balances,
           halving_frequency=1,
           reward=10,
           terms=3):
    new_bal = []
    for balance in balances:
        b = balance
        for i in range(0,terms):
            b = b + reward
            if i % halving_frequency == 0:
                reward = reward/2
        new_bal.append(b)
    return new_bal
```

Simple Demo Results

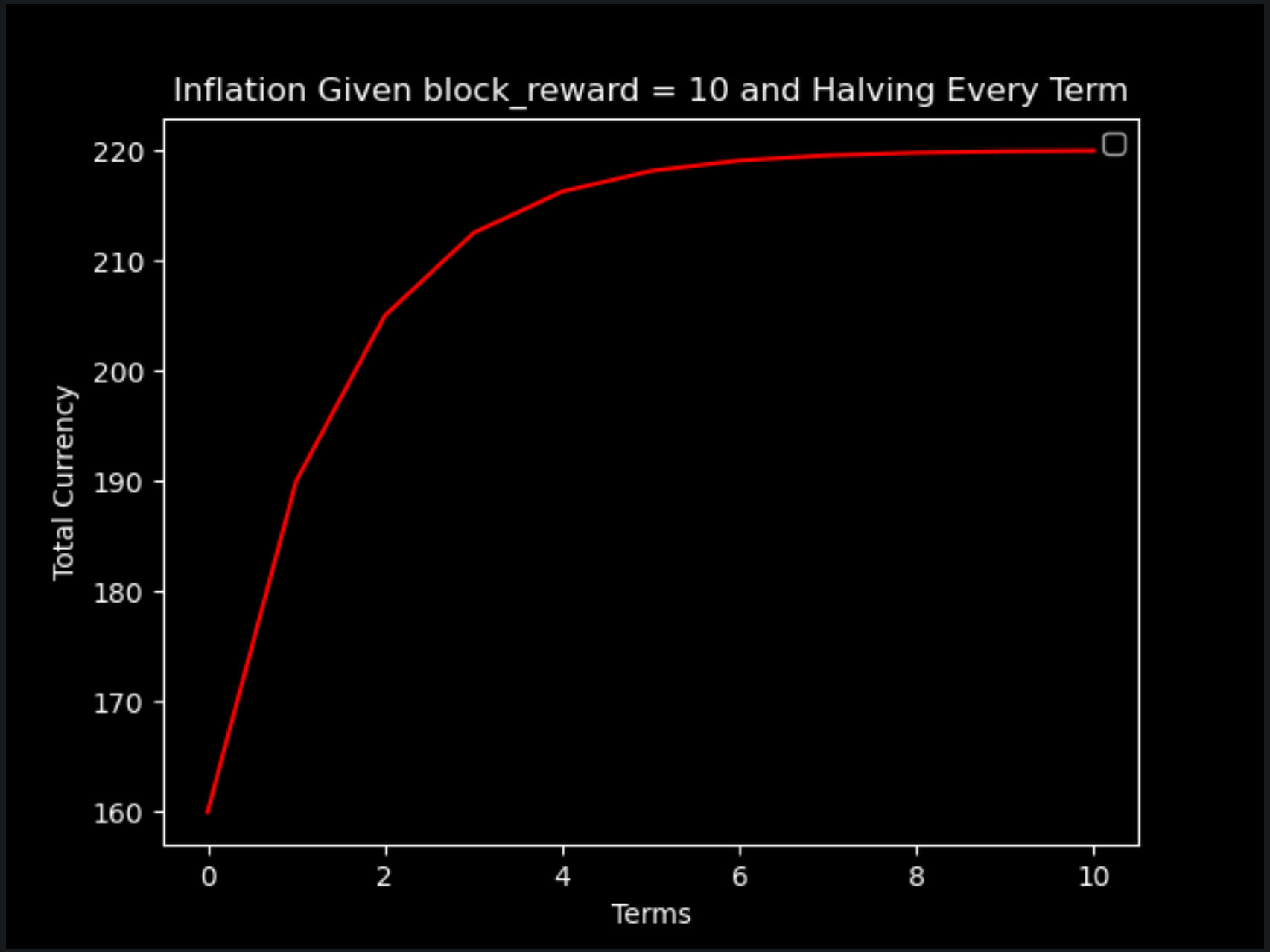
For simplicity, every user is a miner with the same hash rate who will claim a block reward within a term.

```
participants = ["Alice", "Bob", "Charlie"]
balances = [100,40,20]
```

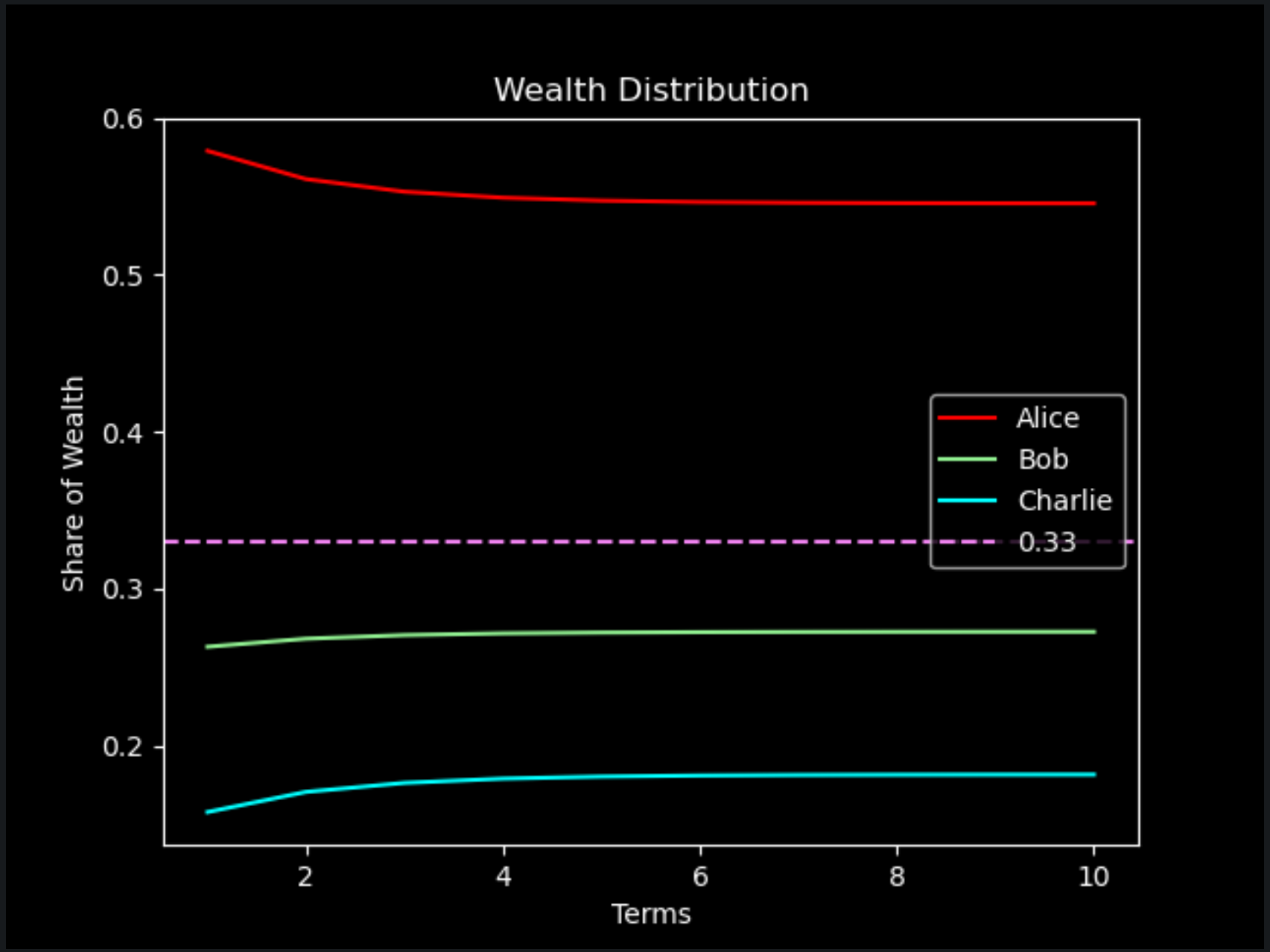
To illustrate halving within a small chart, it occurs each term in this demo, and the initial block reward is 10.

#	name	initial	1	2	3
0	Alice	100	110	115	117.5
1	Bob	40	50	55	57.5
2	Charlie	20	30	35	37.5

Inflation: the most happens at the beginning



One Nudge Toward 1/n then Diminishing Returns. . .



Recap

compounding interest *(banking)*

- inflation is represented by an exponential function.
- if everyone has the same APY, wealth distribution does not change.

constant dividend aka Universal Basic Income *(gaming)*

- inflation is represented by a linear function
- wealth distribution inclines towards 1/n, where *n* is the number of participants

mining w/ halving *(PoW cryptocurrencies)*

- inflation is represented by a logarithmic function
- if everyone has the same hash rate, wealth distribution initially bumps towards 1/n, then flattens as diminishing returns incur

Temporal Asymmetry

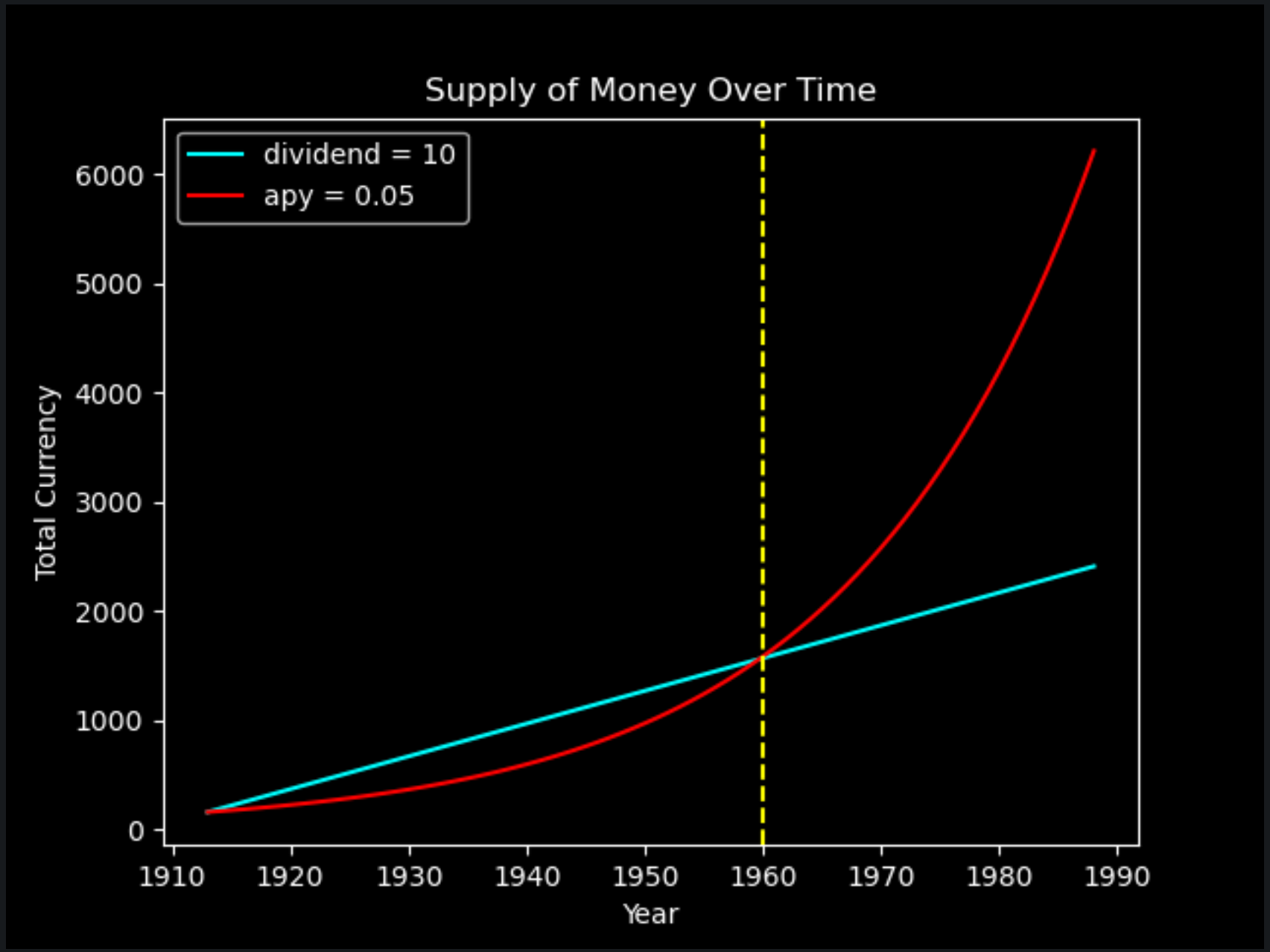
Under exponential and logarithmic functions, different generations of people, players, or users will experience inflation differently.

$$exponential : \frac{d}{dt}(x^t) = x^t \ln(x)$$

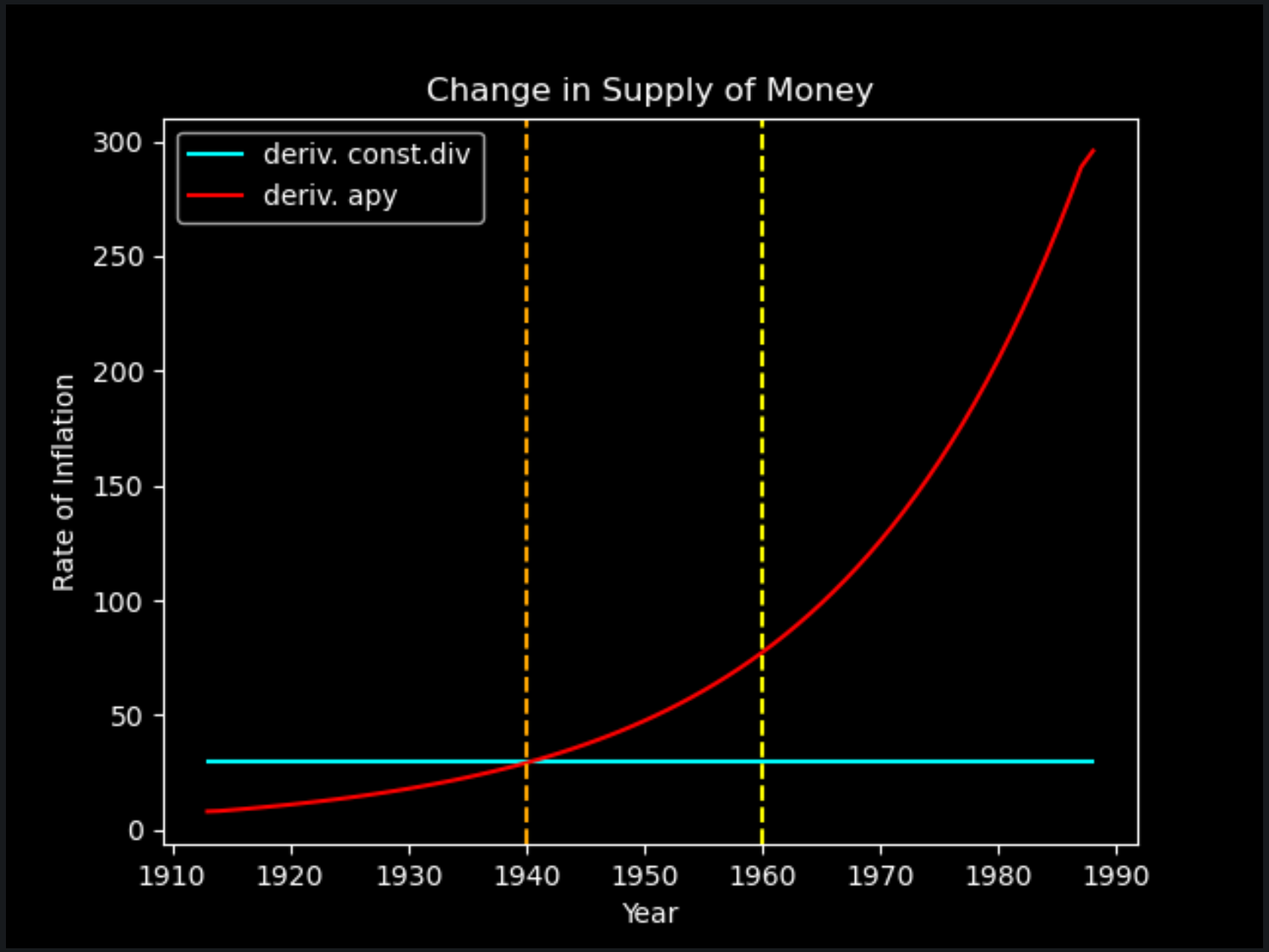
$$logarithmic : \frac{d}{dt}(xlog(t)) = \frac{x}{t}$$

$$linear : \frac{d}{dt}(xt) = x$$

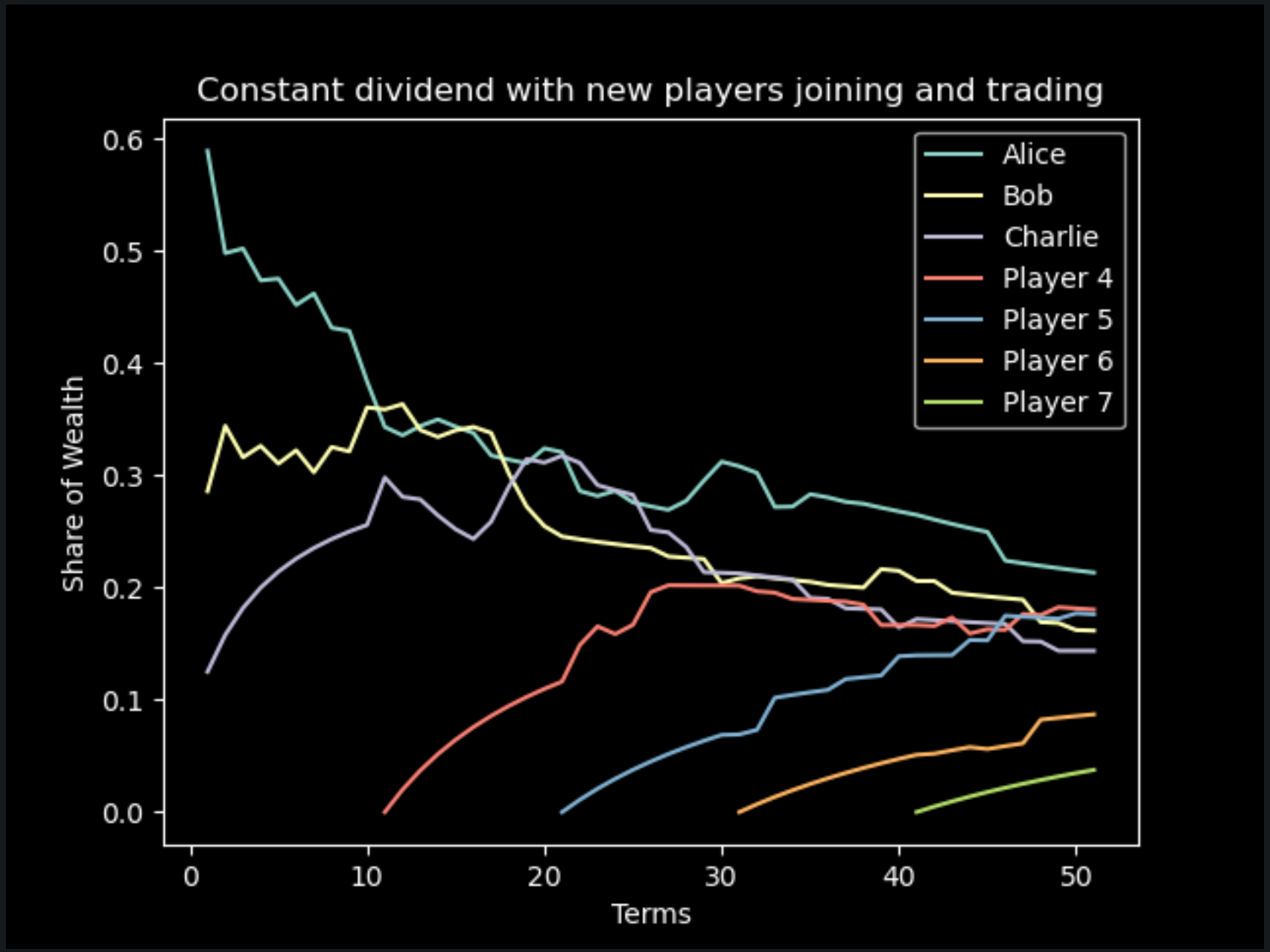
Temporal Asymmetry Example



Temporal Asymmetry Example (Derivatives)

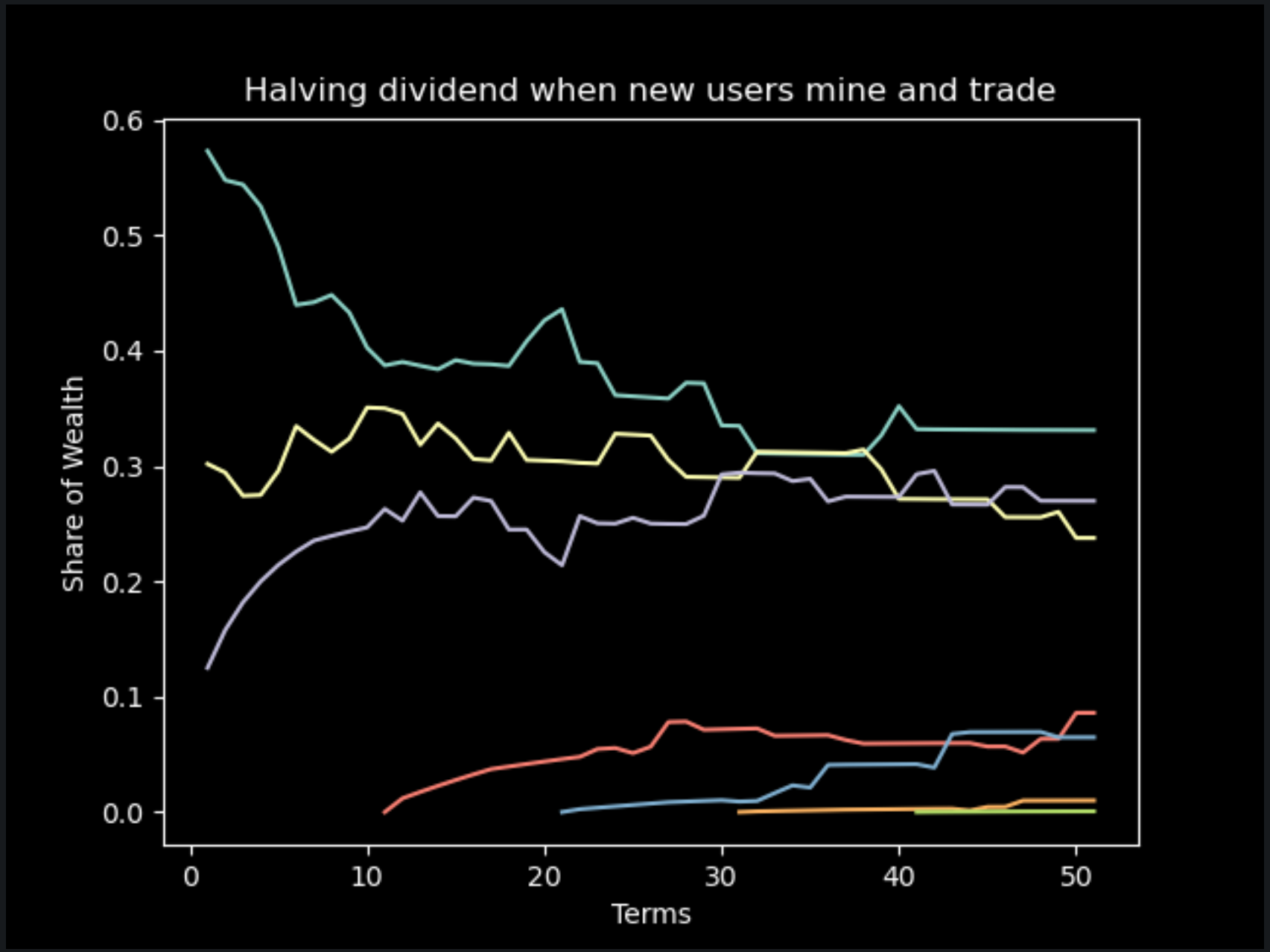


Wealth Distribution When New Players Join



Wealth Distribution When New Bank Accounts Open

Wealth Distribution When New Users Mine



Any questions?

Head Down the Rabbit Hole?

This talk is based on. . .

The Relative Theory of Money by Stephane Laborde

https://en.trm.creationmonetaire.info/

Pourquoi la monnaie libre n'est justement pas un Ponzi ?

"Why free currency isn't just a Ponzi Scheme?"

https://www.youtube.com/watch?v=WRq2197FIMw

Geconomicus

The economy simulation (card) game designed by Laborde and implemented for the web by Nicolas "diablade" Markovic.

https://geconomicus.fr/

https://github.com/diablade/geconomicus

More of My Stuff

My website

https://scuti.neocities.org/

The Code That Generated the Graphs For this Talk

https://codeberg.org/scuti/econ-demo

Where's the money? C Programming or Python?

My logarithmic regression analysis of annual compensation of embedded systems developers and data scientists over a span of 40 years.

https://scuti.neocities.org/pages/where-is-the-money