Modeling Banks: Easier Said than Done

Charlie Webb and James Spillmann

Goal

Modeling a mini economy

Interest Rates in real life are set by the federal treasury to balance spending. Put simply, interest rates are the price you pay to borrow money.

If interest rates are low, borrowing will be cheaper as you lock in at lower rates meaning consumers and businesses are more willing to make larger purchases and loan money from the bank. If spending/economic growth gets too high, interest rates will rise.

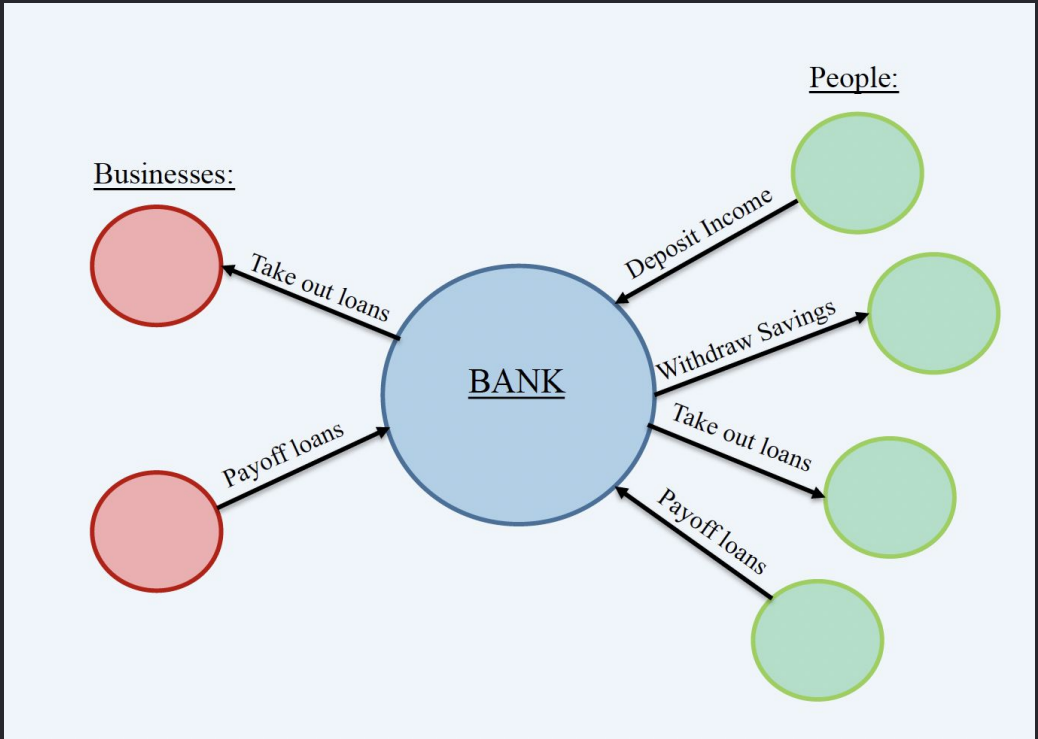
If interest rates are higher, borrowing money is more expensive meaning consumer spending will slow down . This may limit a bank's lending capacity and impact their profitability. In an economy with slow credit, there will be a lower demand for credit, leading to lower interest rates.

10 Years of interest rates

Treasuries solution to low spending during COVID

Overcompensation to curb inflation

Our Model:


People:
Businesses:
Deposit Income
Take out loans
Withdraw Savings
BANK
Take out loans
Payoff loans
Payoff loans

Our code solution:

```
G.add_node(bank_name,
           type='bank',
           cash = bank_cash,
           loaned_money=1,
           accounts = [],
           loans = [],
           crash = False)
```

Our code solution:

```
G.add_node(person_name,
           type='person',
           income=income,
           bank_accounts=[initial_bank_account],
           net_worth=net_worth,
           loans = [],
           cash = initial_cash,
           initial_worth=net_worth)
```

Our code solution:

```
G.add_node(business_name,
           type='a_business',
           cash=cash,
           total_worth=total_evaluation,
           loans=[])
```

Our code solution:

```python
class bank_account():
    def __init__(self, bank, person, amount, interest_rate, confidence):
        self.bank = bank
        self.person = person
        self.amount = amount
        self.interest_rate = interest_rate
        self.confidence = confidence
```

You, 2 days ago | 1 author (You)

```python
class loan():
    def __init__(self, bank, recipient, type, amount, interest_rate, months):
        # can add an amount paid? or does that not matter.
        self.bank = bank
        self.recipient = recipient
        self.recipient_type = type
        self.amount = amount + (amount * interest_rate)
        self.interest_rate = interest_rate
        self.monthly_payment = (amount + (amount * interest_rate)) / months
        self.monthly_payment_no_interest = amount / months
```

Every “game tick”

People:

- Get paid
- Buy things
- Withdraw money if needed
- Check their accounts to ensure 10-30% cash on hand
- Take out a loan?

Every “game tick”

Businesses:

- Buy supplies
- Take out a loan?

Every “game tick”

Banks:

- Collect on their loans
  - Pay or else
- Maintain a 10-30% cash/loan ratio
  - If too much cash - make loans!
  - If not enough cash - loan from other banks.
  - If STILL not enough cash, crash.

Every “game tick”

Update interest rate based on average cash in each entity.

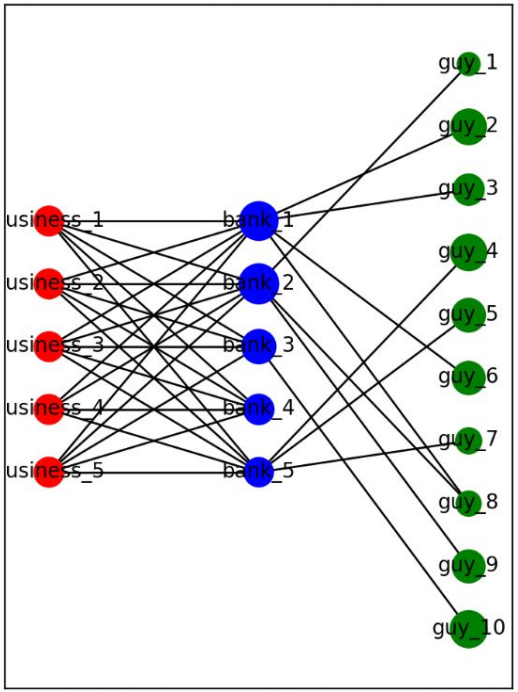
Banks have too much money? Speed up the economy! Lower the rate!

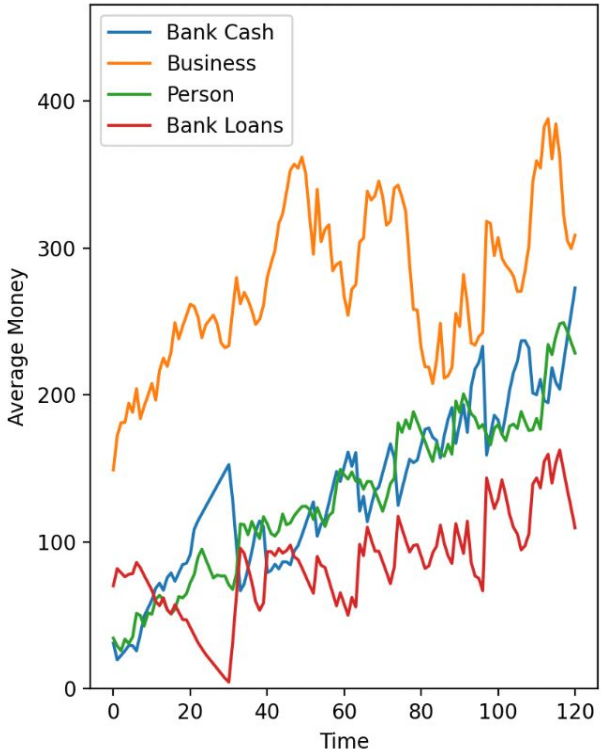
Businesses and people have too much money? Raise the interest rate! Slow down the economy!

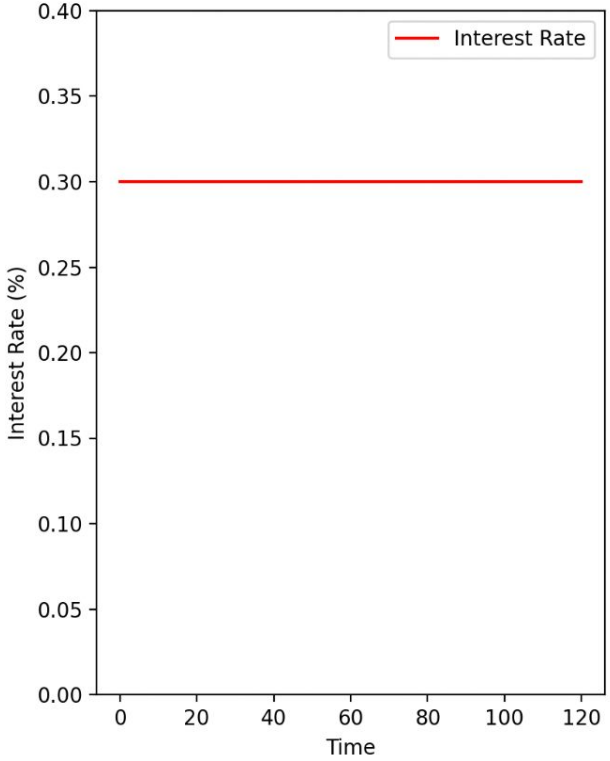
Every “game tick”

Plot data.

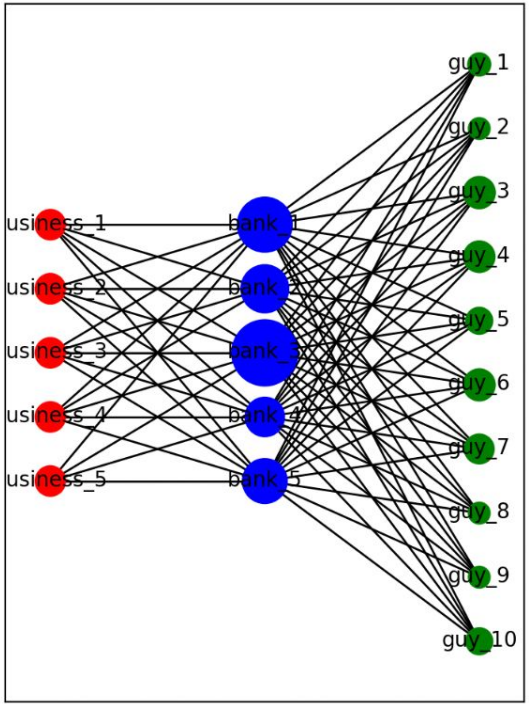
Results

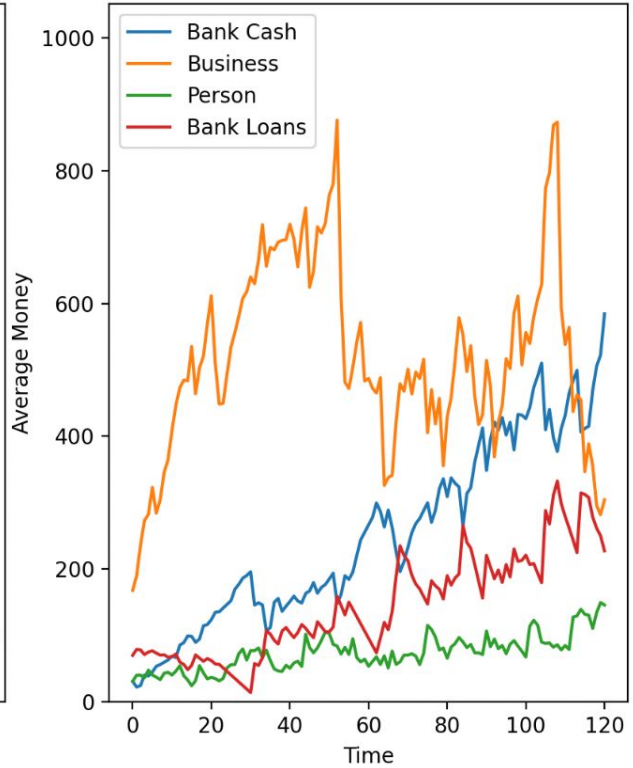


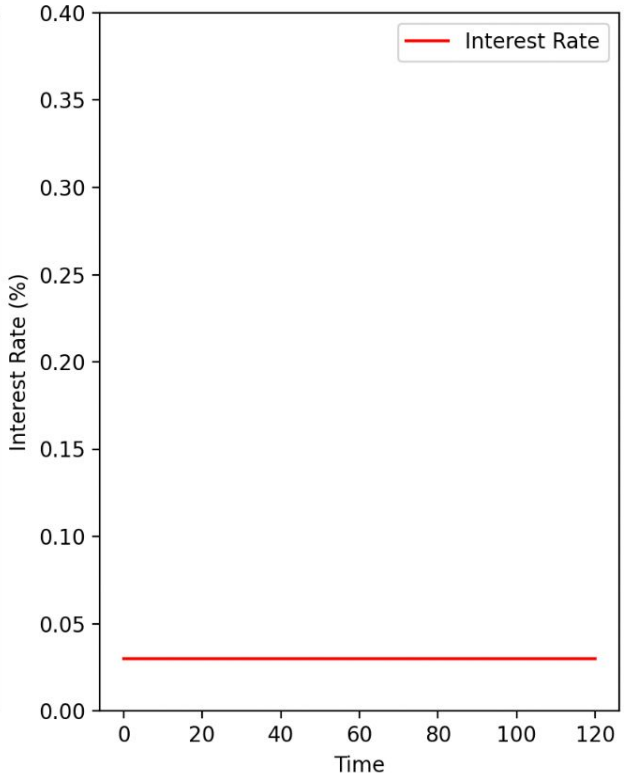




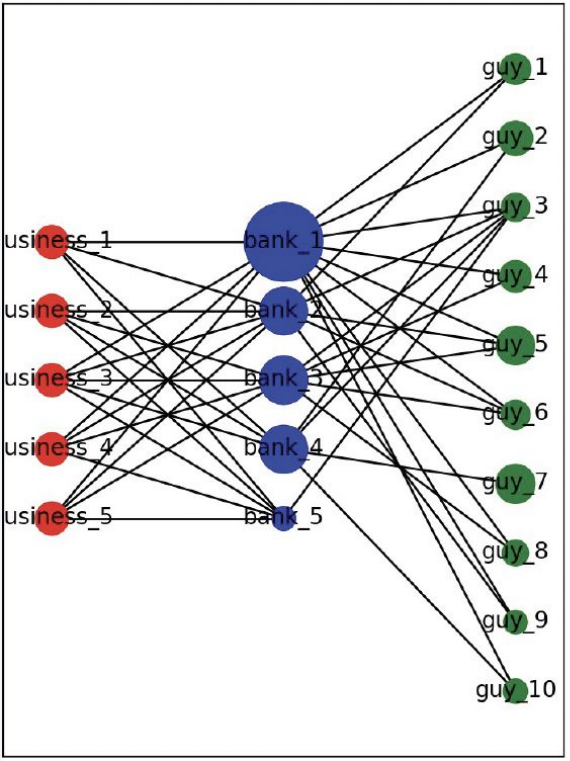
If we fix interest rates high

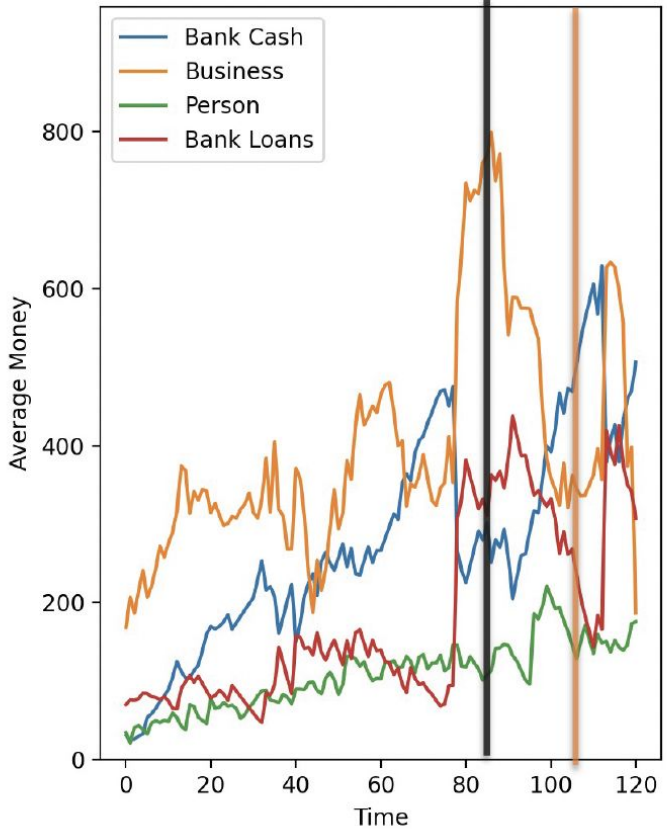


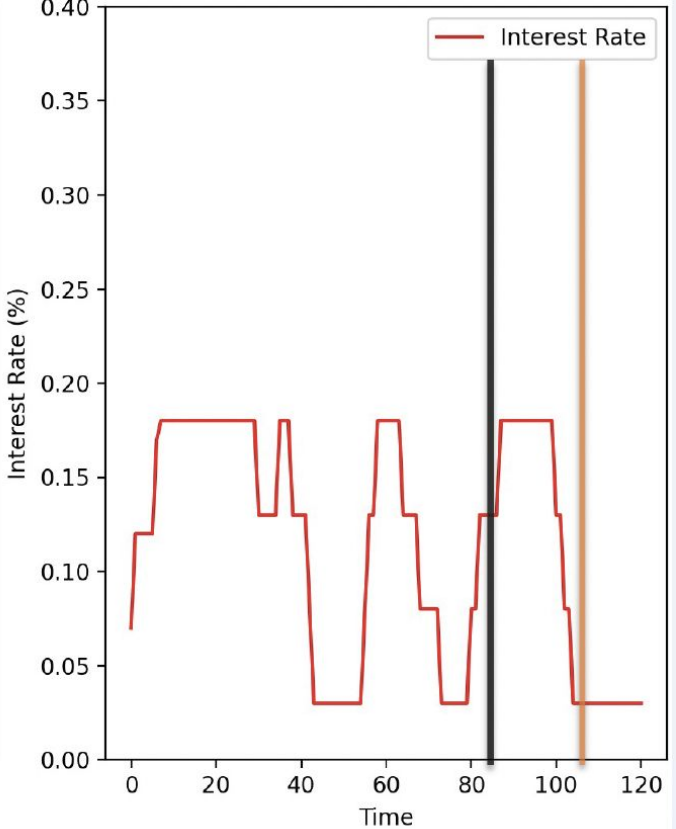




If we fix interest rates low







If we let interest rates fluctuate:

Crashing behavior

Uncertain without a federal treasury to back lost money.

In this model, all money disappears when a bank crashes.

People have too much cash and put it all in new bank accounts.

Banks cannot double loan to businesses, so if there are no remaining businesses to loan to, we do nothing.

If loaned money goes negative, we crash.

etc.

Successes

Markets react to interest rates properly.

Banks are stable.

Banks maintain a 10-30% cash/loan ratio in line with real US solvency laws.

They do this without “cheating”. Money is deposited, withdrew and loaned.

Versus:

People get income out of thin air. Money spent by businesses goes nowhere

Failures

Disclaimer:

In our model, the interest rates are purely reactive to the economic state.

This model predicts nothing about actual banking behavior because of this.

Interest rates fluctuate very quickly. Flipping the market one way or another rather than finding an equilibrium. The model reacts almost instantly to changing rates so it flips fast.

This is an open system versus in the actual economy all money goes somewhere. Money is spawned in by the government though.

Our ideas to expand our model

Close the economy.

People are employees of businesses. People spend money to businesses.

Add logic for the ‘FED’ node to bail out crashing banks.

```
59      # add the fed
60      # G.add_node("THE FED",
61      #            type='fed',
62      #            )
```

Our ideas to expand our model

More stable interest rates. This entails:

Getting a more accurate view of the state of the economy.

Remembering past interest rates.

Defining what a good economy looks like. Use this as a goal state and try to maintain it.

Another bank crash model

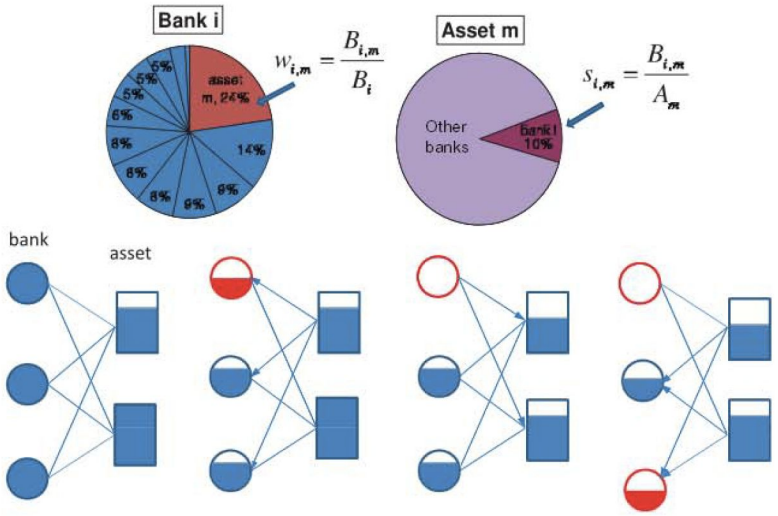


Figure 1 | Bank-asset bipartite network model with banks as one node type and assets as the other node type. Link between a bank and an asset exists if the bank has the asset on its balance sheet. Upper panel: illustration of bank-node and asset-node. $B_{i,m}$ is the amount of asset m that bank i owns. Thus, a bank i with total asset value B_i has $w_{i,m}$ fraction of its total asset value in asset m. $s_{i,m}$ is the fraction of asset m that the bank holds out. Lower panel: illustration of the cascading failure process. The rectangles represent the assets and the circles represent the banks. From left to right, initially, an asset suffers loss in value which causes all the related banks' total assets to shrink. When a bank's remaining asset value is below certain threshold (e.g. the bank's total liability), the bank fails. Failure of the bank elicits disposal of bank assets which further affects the market value of the assets. This adversely affects other banks that hold this asset and the total value of their assets may drop below the threshold which may result in further bank failures. This cascading failure process propagates back and forth between banks and assets until no more banks fail.

Source:

Huang, Xuqing & Vodenska, Irena & Havlin, Shlomo & Stanley, H.. (2013). Cascading Failures in Bi-partite Graphs: Model for Systemic Risk Propagation. Scientific reports. 3. 1219. 10.1038/srep01219.

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4394553

https://www.researchgate.net/figure/Bank-asset-bipartite-network-model-with-banks-as-one-node-type-and-assets-as-the-other_fig4_235406706