Simulating CBDC Effects on Monetary Policy with Agent Based Modelling and Reinforcement Learning: The Brazilian Case

Gabriel Magnago

Dissertation written under the supervision of professor Pedro Afonso Fernandes

Dissertation submitted in partial fulfilment of requirements for the MSc in Business Analytics, at the Universidade Católica Portuguesa, 2026.

Simulating CBDC Effects on Monetary Policy with Agent Based Modelling and Reinforcement Learning: The Brazilian Case

Gabriel Magnago

Universidade Católica Portuguesa
Católica Lisbon School of Business & Economics
Portugal

January 4, 2026

Abstract

This study proposes a research path that combines monetary economics with reinforcement learning to analyze how Central Bank Digital Currency (CBDC) design can shape adoption patterns and financial stability. Using the Brazilian Drex as the empirical foundation, it develops an agent-based model where consumer adoption, merchant acceptance, and bank balance sheet dynamics emerge from micro-level interactions, calibrated to Brazilian financial system parameters.

The methodology compares three policy approaches: (1) a fixed baseline policy, (2) rule-based adaptive heuristics, and (3) a reinforcement learning agent that learns optimal policy schedules through offline training episodes. The RL agent dynamically adjusts wallet transaction limits and bank leverage constraints based on observed system states, seeking to maximize adoption velocity while maintaining financial stability.

By training on simulated episodes and deploying learned policies, the framework enables central banks to explore the policy space systematically, discovering adaptive strategies that balance the competing objectives of rapid digital currency diffusion and banking sector stability. This contributes to the ongoing dialogue on how central banks might design frameworks for digital currency implementation that respond to the complex, evolving nature of modern financial systems.

Keywords: Central Bank Digital Currencies; CBDC; Reinforcement Learning; Machine learning; Agent-based Modelling; Central Bank; Monetary Policy.

Abstract

Este estudo propõe uma linha de investigação que combina economia monetária com a aprendizagem por reforço para analisar como o desenho de Moedas Digitais do Banco Central (CBDC) pode moldar padrões de adoção e a estabilidade financeira. Utilizando o Drex brasileiro como base empírica, desenvolve-se um modelo baseado em agentes onde a adoção pelo consumidor, a aceitação pelo comerciante e a dinâmica do balanço dos bancos emergem de interações a nível micro, calibradas aos parâmetros do sistema financeiro brasileiro.

A metodologia compara três abordagens de políticas: (1) uma política de base fixa, (2) heurísticas adaptativas baseadas em regras e (3) um agente de aprendizagem por reforço que aprende agendas de políticas ótimas através de episódios de treino offline. O agente de aprendizagem por reforço ajusta dinamicamente os limites de transação da carteira e as restrições de alavancagem bancária com base nos estados observados do sistema, procurando maximizar a velocidade de adoção enquanto mantém a estabilidade financeira.

Ao treinar com episódios simulados e implementar políticas aprendidas, a estrutura permite aos bancos centrais explorar o espaço das políticas de forma sistemática, descobrindo estratégias adaptativas que equilibram os objetivos conflitantes de rápida difusão de moedas digitais e a estabilidade do setor bancário. Isto contribui para o diálogo contínuo sobre a forma como os bancos centrais podem conceber quadros para a implementação de moedas digitais que respondam à natureza complexa e em constante evolução dos sistemas financeiros modernos.

Keywords: Moedas Digitais dos Bancos Centrais; CBDC; Aprendizagem por Reforço; Aprendizagem de Máquina; Modelação Baseada em Agentes; Banco Central; Política Monetária.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Overview

The rapid digitalization of the global economy, driven by the emergence of decentralized crypto assets, FinTech innovations, and the declining use of cash in favor of non-cash payment methods, has prompted central banks to actively explore the potential of Central Bank Digital Currencies (CBDCs). The number of CBDC research initiatives and pilot projects by countries has surged from 35 in May 2020 to 134 by March 2024, reflecting a significant acceleration in engagement with this new form of money since its conceptual emergence around 2016 (Illes et al., 2025; Bindseil et al., 2025; Di Maggio et al., 2024). This global pursuit of the "future of money" suggests that most economies will soon engage in pilot projects or even full-scale implementations of CBDCs, which, depending on the chosen design model, could bring varying degrees of transformation to the financial sector.

Several countries have already taken concrete steps. The Bahamas, the Eastern Caribbean Currency Union, and Nigeria have introduced CBDCs, while pilot programs are underway in Europe (Digital Euro), South America (Brazilian Drex), and Asia (as the Digital Rupee in India and the Digital Renminbi in China), among others (Mishra and Prasad, 2023). Despite this progress, the mainstream approach to CBDC design remains uncertain, as most countries continue to explore both retail and wholesale models [1]. As of late 2024, advanced economies tend to experiment with both types, while developing economies generally focus on one (Illes et al., 2025). In terms of motivations, emerging economies most commonly pursue central bank digital currencies (CBDCs) to close gaps in financial inclusion, modernize payment systems, and reduce structural reliance on cash, while advanced economies, in contrast, move more cautiously, focusing on stability, security, and institutional credibility. Yet, across both groups, wholesale CBDCs have taken the lead, shaping the technical architecture and interoperability standards that will later most likely define retail systems (Illes et al., 2025; Pastor Sempere, 2025).

Within this broader context, Brazil stands out as a compelling case for analysis. The country combines one of the largest and most sophisticated financial sectors in the Global South with a deep culture of financial innovation. The Central Bank of Brazil's Drex project, originally envisioned as an AI-enhanced, blockchain-based retail CBDC was meant to anchor a new generation of programmable, tokenized financial assets. The plan was to deliver instant settlement, full traceability, and interoperability between public and private systems. However, after two pilot phases, the project was recently scaled back and postponed to 2026, its initial version stripped of the blockchain layer and limited to a more conservative infrastructure for reconciling credit liens (Pligher, 2025).

[1] Wholesale CBDCs are primarily intended for interbank settlements and transactions among financial institutions, while a retail CBDC is a form of central bank digital currency that is used by the general public (Waliczek and Nili, 2024)

This retrenchment reveals the political and institutional complexity behind implementing a full retail CBDC. It also highlights the growing asymmetry between public ambition and private momentum. While the central bank recalibrates, firms such as Nexa Finance and large banks like Bradesco are now driving the tokenization of real assets, from agricultural credit to receivables, building the market infrastructure that Drex was supposed to pioneer. These and other actors are treating tokenization as an immediate business model, leveraging blockchain and payment technologies to cut costs, increase liquidity, and automate compliance (Pligher, 2025).

Brazil's situation exposes the tensions between central bank design choices and market-driven innovation, between public trust and private efficiency. Understanding how different CBDC architectures affect the structure and functioning of the banking system is therefore essential. Rather than emphasizing user adoption or payment preferences, this study follows the simulation-based approach of Ramadiah et al. (2021), modeling how design parameters, such as remuneration, holding limits, and convertibility rules, can influence liquidity dynamics, credit allocation, and monetary control.

The present dissertation takes Brazil as an empirical and conceptual test case. Using an Agent-Based Model (ABM) enriched by Reinforcement Learning (RL), it explores how various CBDC configurations could alter the balance between central bank money and bank deposits, reshape funding structures, and affect the transmission of monetary policy.

In that sense, Brazil's case study captures the frontier tension defining the next stage of financial digitalization. Whether Drex ultimately catches up, or is overtaken by private solutions, remains uncertain. But the outcome will likely influence how emerging economies balance innovation with stability in the digital age.

1.2 Research question

The main research question of the paper is how central banks should design CBDC policy parameters to promote adoption while maintaining financial stability. From this overarching question, we can derive three interrelated sub-questions that guide the analysis.

First, how do CBDC wallet caps and bank leverage constraints influence consumer adoption patterns and merchant acceptance? This involves assessing how limits on individual CBDC holdings and regulatory constraints on bank balance sheets shape the relative attractiveness of CBDC and the incentives of economic agents to integrate it into their payment behavior.

Second, what are the effects of CBDC introduction on payment method substitution and commercial bank deposit funding? Here the focus is on quantifying shifts across cash, card-based deposits, and CBDC, and on evaluating the resulting consequences for banks' liability structure, funding stability, and leverage dynamics.

Third, how do adaptive, learning-based policy frameworks compare to fixed and rule-based approaches in navigating the multi-objective policy space? This question examines whether reinforcement learning agents that learn dynamic policy schedule achieve faster adoption with

comparable or superior stability compared to static policies or simple heuristic rules.

2 Literature Review

The foundational reference for agent-based modeling of Central Bank Digital Currencies (CBDCs) is the study by Ramadiah et al. (2021), which presents the first multi-period agent-based model designed to analyze the macrofinancial effects of introducing a retail CBDC. Calibrated using aggregate statistics from the German retail payment market, the model provides a flexible framework to quantify how a CBDC may influence the usage of alternative payment instruments, the composition of consumer wealth, and the potential for banking sector disintermediation. The authors demonstrate that a carefully configured retail CBDC can coexist with traditional banking structures without materially disrupting bank balance sheets, although intermediaries such as card companies may experience a significant reduction in transaction revenues. Beyond these findings, the model is made publicly available through an online simulation platform, allowing researchers to explore a wide array of policy experiments and alternative CBDC configurations across different jurisdictions, thus offering a robust foundation for comparative and policy-oriented research.

The book *Governance and Control of Data and Digital Economy in the European Single Market*, edited by Pastor Sempere (2025), brings together contributions from leading academics to examine the regulatory, institutional, and governance challenges posed by digital assets, digital identities, and data spaces within the European Single Market. Among its contributions, the chapter by Pablo Sanz Bayón, *Current and Future Central Bank Digital Currency (CBDC) Projects*, provides a comprehensive conceptual and comparative analysis of retail and wholesale CBDC initiatives, including the digital yuan and the Digital Euro. The chapter emphasizes the central bank's role in issuing and maintaining the legal and technical integrity of digital money, analyzing the implications for banking system operations, user protection, and the broader fintech ecosystem. This work offers a crucial reference for understanding the regulatory and governance considerations that underpin CBDC design, deployment, and integration into existing financial systems.

To frame the agent-based modelling approach, several key studies were consulted. As a concept, agent-based models explain the behaviour of a system by simulating each individual 'agent' within it, whether these are consumers in an economy, fish in a shoal, or particles in a gas. Their main strength lies in demonstrating how relatively simple agent behaviours can combine from the 'bottom up' to generate the complex dynamics observed in real-world systems. This contrasts with 'top-down' models, which often assume aggregate behaviours or identical agents. While initially developed in the physical sciences, agent-based models have increasingly been applied in economics to explore financial market dynamics and the effects of policy interventions (Turrell, 2016).

Applications of this framework have also expanded to decentralized finance. For example, Struchkov et al. (2024) use agent-based models to simulate participant behaviour in blockchain-based exchanges, capturing both normal and stress market conditions, analyzing the impact of

front-running strategies, and evaluating automated market makers such as Uniswap and Liquifi. These studies illustrate the flexibility of the agent-based approach in modeling interactions and emergent phenomena in both traditional and decentralized financial systems. With the increasing availability of computational power and granular data, agent-based modelling is poised to become an even more significant tool for understanding economic dynamics and assessing the potential impacts of policy decisions.

Several working papers from the European Central Bank (ECB) on digital currencies were also consulted. Of particular relevance is the study by Georgarakos et al. (2025), which employs a series of population-representative experiments to investigate consumer attitudes towards the potential introduction of a digital euro. The authors show that brief educational interventions, such as explanatory videos, can significantly shape beliefs about CBDCs and increase the likelihood of adoption. Their findings further suggest that, on aggregate, consumers would allocate a relatively small portion of wealth to digital euros, that holding limits within the range of €1,000–€10,000 have limited effects on asset composition, and that a notable fraction of consumers may refuse adoption due to strong preferences for existing payment methods.

Complementing this work, Lambert et al. (2024) applies a structural demand model to unique consumer-level survey data from the euro area to evaluate how different CBDC design options, combined with individuals' revealed payment preferences and socioeconomic characteristics, influence potential demand for a digital euro. The study finds that, in an unconstrained scenario, the digital euro could account for 3–28% of household liquid assets (€0.12–€1.11 trillion), while a €3,000 per-person holding limit would reduce this to 2–9% (€0.10–€0.38 trillion). Design features such as privacy, automatic funding, and instant settlement further increase potential demand. Together, these papers provide critical insights into the behavioral and structural determinants of CBDC adoption, helping to further define the calibration and validation parameters of agent-based and policy-oriented models.

From the National Bureau of Economic Research (NBER), two papers are particularly relevant. Di Maggio et al. (2024) provides one of the first empirical analyses of retail CBDC adoption, drawing on detailed transaction data from India's pilot program. The study examines the interaction between CBDCs and existing digital payment methods, showing that policies increasing transaction costs for conventional payments can accelerate CBDC uptake. It also finds that higher CBDC usage is associated with reductions in bank, cash, and savings deposits, highlighting potential pathways toward bank disintermediation and illustrating the transformative effects of CBDCs on traditional financial infrastructures.

Adding to this empirical perspective, Mishra and Prasad (2023) develop a general equilibrium model that elucidates the trade-offs between physical cash and digital central bank money. The model explores differences in transaction efficiency, tax evasion opportunities, and nominal returns, identifying conditions under which cash and CBDCs can coexist. It also demonstrates how CBDC design can facilitate policy tools such as negative interest rates and helicopter drops, while limiting capital flight from other assets. Together, these studies provide both empirical

and theoretical foundations for understanding the dynamics of CBDC adoption and its broader macroeconomic implications.

To understand the specific dynamics of the Brazilian *Drex*, the primary references are Banco Central do Brasil (2023) and Banco Central do Brasil (2024), which establishes the foundational framework for the currency's issuance by the Brazilian monetary authority. The 2023 *Diretrizes* emphasize the development of innovative models incorporating advanced technologies, such as smart contracts and programmable money, compatible with transaction settlement through Internet of Things (IoT), while also supporting online and potential offline payment applications. They prescribe wholesale issuance of the Drex by the Central Bank to facilitate retail financial services through the Drex issued by participants in the National Financial System (SFN) and the Brazilian Payment System (SPB), applying existing norms to prevent regulatory asymmetries and ensuring legal certainty, privacy, and cybersecurity in accordance with Brazilian law and international recommendations. Furthermore, the guidelines promote the adoption of distributed ledger technology (DLT) for asset registration, interoperability with domestic and cross-border systems, and resilience measures consistent with critical financial market infrastructure standards.

The 2024 *Relatório Drex* pilot introduced a series of operational limits, formalizing two distinct tokenized asset layers: Drex de Atacado, backed one-to-one by simulated reserve balances, and Drex de Varejo, issued in a decentralized manner by financial institutions against their customers' demand deposits and e-money holdings. This architectural configuration is central to the model proposed in this dissertation, as it provides the conceptual foundation for the model. Its nuances, including operational restrictions, settlement pathways, and the lifecycle mechanics governing token issuance, transfer, and conversion, will be examined in greater detail in the methodology section.

Complementarily, Zarifis and Cheng (2025) provides an analytical model of trust in Central Bank Digital Currency (CBDC) within the Brazilian context, examining how consumer confidence in a two-tier structure, where both the central bank and retail banks participate, affects adoption dynamics and overall trust in digital currency systems. This work offers critical insights into the behavioral and structural factors shaping Drex uptake and the role of institutional design in reinforcing public confidence.

For a broader perspective on global trends, the Bank for International Settlements (BIS) papers are particularly informative. Notably, Illes et al. (2025) presents the results of the 2024 BIS survey on central bank digital currencies (CBDCs) and crypto, based on responses from 93 central banks. The survey provides detailed insights into central banks' engagement with CBDC initiatives, including motivations for potential issuance, envisioned use cases, and design features. The survey also covers regulatory approaches to cryptoassets and stablecoins, as well as emerging developments in tokenization of commercial bank deposits and other assets, highlighting implications for payments, monetary policy, and financial stability.

On another paper, the BIS suggests that as central banks explore retail central bank digital

currencies (CBDCs), it is crucial to establish a clear technical architecture that defines how a CBDC will be issued, circulated, and managed, as well as the roles of the central bank and private intermediaries (Bank for International Settlements, 2024). The BIS highlights that early design choices can have lasting implications for privacy, compliance, and system functionality. In particular, the report proposes a hybrid model that combines centralised oversight with decentralised access, incorporates tiered know-your-customer (KYC) mechanisms, and separates transaction data from identity information to enhance privacy protections. The report also outlines operational workflows for four main processes: user enrolment, CBDC creation, CBDC destruction, and intra-ledger transfers. Furthermore, the BIS emphasizes the potential of tokenisation and integrated financial ecosystems, where tokenised deposits and CBDCs coexist, as a promising area for future development. These insights provide a practical and policy-oriented reference for central banks seeking to implement retail CBDCs.

Finally, the Atlantic Council Central Bank Digital Currency Tracker provides up-to-date information on the global state of CBDC adoption and pilot projects (Atlantic Council, 2025). As of 2025, 137 countries and currency unions, (representing 98% of global GDP), are exploring CBDCs, with 72 in advanced phases of development, pilot, or launch.

3 Methodology and Methods

3.1 Methodological Approach

To study what a retail CBDC might do in Brazil, we used a modeling approach that mixes agent-based modeling and reinforcement learning, drawing from Ramadiah et al. (2021). Agent-based modeling (ABM) lets us simulate how different agents in a financial system interact. We give them simple rules for their behavior and then watch the bigger patterns that emerge (Wilensky and Rand, 2015; Weimer et al., 2016). On top of this, reinforcement learning (RL) adds a layer where the central bank can learn and adapt, tweaking its policy settings over time based on what it sees happening and how the system responds (Weimer et al., 2016).

Agent-based models are good for understanding complex financial systems because the agents act on their own, are spread out, and have specific goals. Each agent has a memory and an internal state that guides its rules, helping it react to changes in its surroundings and to what other agents do (Weimer et al., 2016). In our model, consumers change how much cash, deposits, and CBDC they hold based on how much CBDC they can have in their wallets, how many businesses accept it, and what they need to spend. They start using CBDC wallets based on a logistic function, which depends on how many other people use it and how many businesses accept it. Businesses, in turn, decide whether to accept CBDC based on the number of customers they see using CBDC wallets. Commercial banks adjust their lending levels based on how much money is deposited and how profitable they are, trying to meet rules while making the most money from risky assets.

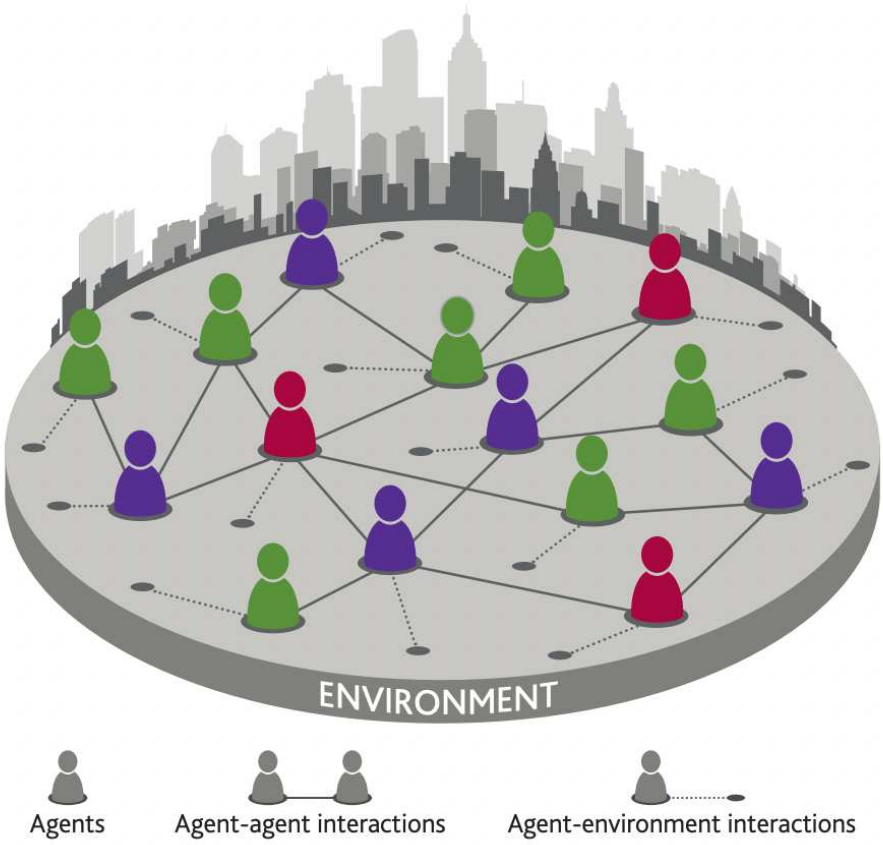

Figure 1: Schematic of the typical elements of an agent-based model. Source: (Turrell, 2016).

When we build these ABMs, we often use object-oriented programming. This way, each agent's state, behavior, and decision rules can be bundled together in one place (Weimer et al., 2016). The model treats each type of agent, (consumers, merchants, commercial banks, and the central bank), as a class. These classes have methods that lay out their daily actions and how they update their status. Consumers transact with randomly chosen merchants, trying to pay with CBDC (if they have it), then card-based deposits, then cash, in that order. Banks take deposits, give out loans on risky assets, and adjust their leverage based on their capital and regulatory limits. The central bank sets policy items like limits on CBDC wallet amounts and maximum bank leverage.

The ABM shows how network effects and feedback loops are key to how CBDC spreads. When consumers adopt CBDC, it follows a logistic diffusion process: the chance of someone getting a CBDC wallet goes up as more people use it, as more businesses accept it, and as the wallet features become more appealing. Businesses decide to accept CBDC based on the number of customers with wallets and the typical CBDC amounts in those wallets. This creates a cycle where policy settings directly affect how quickly CBDC is adopted. These adaptive behaviors show how the system is decentralized and goal-oriented, leading to specific patterns of adoption (Weimer et al., 2016).

To make sure our analysis lines up with how the Brazilian Drex Pilot actually works, our

model follows the design rules outlined in Banco Central do Brasil (2024). These rules are the basic reference for how Brazil's CBDC is being designed. We set the model's parameters using data from the Brazilian Central Bank (BCB), including how income is distributed among consumers, bank leverage goals, and statistics on payment systems. The simulation runs daily over a two-year period, tracking what individual agents decide and what happens across the whole system.

Our setup looks at four main aspects when a CBDC is introduced: (i) how consumers and merchants adopt the digital currency, (ii) how people switch between cash, card-based deposits, and CBDC for payments, (iii) how commercial banks' lending changes and how deposits might shift away from them, and (iv) how the central bank tunes its policy through reinforcement learning. The model studies three policy situations: one with fixed settings, one with adjustments based on rules, and one where a reinforcement learning agent figures out the best policy schedules by training offline before being used. The RL agent uses Q-learning with discretized state spaces and experience replay, optimizing a composite reward function that prioritizes adoption velocity while penalizing regulatory violations and payment failures.

The framework is implemented entirely in Python, leveraging libraries such as `NumPy`, `Pandas`, and `Matplotlib` for ABM simulation and visualization.

3.2 Agent-Based Model Design

Following Ramadiah et al. (2021), we set up four types of agents that interact daily in a discrete-time environment, where each time step t represents a day. In our framework, we adopt the *indirect CBDC* model, as classified by Auer and Böhme (Auer and Böhme, 2020). Here, the central bank is the only one that issues the CBDC, but commercial banks handle it for everyday users. Each consumer's CBDC balance represents a claim on an intermediary rather than a direct claim on the central bank. By contrast, a *direct CBDC* model, would mean consumers have a direct claim on the central bank, which keeps a complete and real-time record of all retail balances and transactions.

The choice for the indirect model was influenced by a few reasons. First, it keeps commercial banks involved as intermediaries, which helps to simulate real-world dynamics of deposits, withdrawals, and how banks manage their money. Second, it makes the model less complex to run and compute because the central bank doesn't need to keep detailed records of every single transaction. Last, it fits with how central banks in developing countries, like Brazil, are talking about designing CBDCs to keep financial mediation going while still having oversight and stability.

In this sense, this model reflects key features of Brazil's Drex pilot project, as detailed in Banco Central do Brasil (2024). In the pilot, retail CBDC tokens (Drex de Varejo, DVt) are issued by individual financial institutions, not directly by the central bank. When a consumer turns deposits into DVt, the institution issues new DVt tokens and takes the same amount from

the customer's deposit account at the same time. This ensures that retail CBDC balances are always fully supported by changes to bank liabilities. When someone cashes out, the process is reversed: DVt balances are removed, and the equivalent deposit amount is put back into the customer's account. Banks settle transactions among themselves using wholesale CBDC (Drex de Atacado) issued by the Central Bank, so retail tokens never move between different institutions (Banco Central do Brasil, 2024).

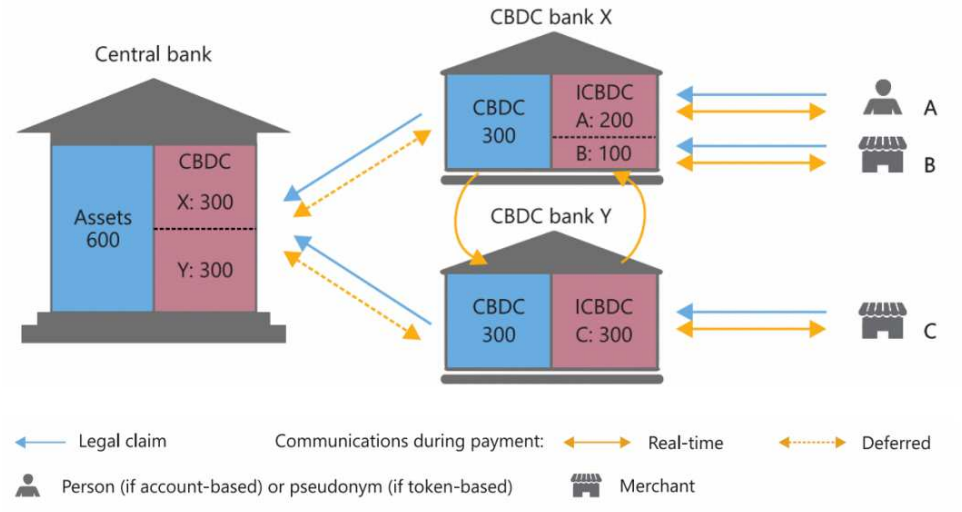

Figure 2: Retail CBDC indirect architecture. Source: (Auer and Böhme, 2020).

This simpler design captures the core economic factors important for policy analysis: CBDC adoption puts pressure on banks as deposits might shift, and banks need to adjust their finances accordingly. The model doesn't explicitly cover how tokens are issued, how banks settle transactions, or other wholesale CBDC layers. Instead, it focuses on the bigger picture: what happens to adoption, how payment methods change, and how stable the banking sector is when retail CBDC comes out.

The main question of this approach is how design features, like wallet limits and lending caps, affect how fast CBDC is adopted and how stable the financial system remains, rather than the technical details of blockchain protocols. By simplifying these operational aspects, the model allows for a systematic exploration of policy choices through reinforcement learning, all while keeping the computations manageable.

3.2.1 Consumers

While in line with the general agent structure proposed by Ramadiah et al. (2021), the consumer block represents a deliberate simplification of the original model. Their study uses rule-based consumers with gradual adjustment mechanisms for asset allocation and random selection for payment methods. Here, the aim is to further simplify and make the model compatible with reinforcement learning. In this sense, consumers are model as an aggregated, rule-based decision

maker. This allows to keep the main behaviors that matter for a retail CBDC in the Brazilian Drex system, such as choosing payment instruments, moving money between deposits and CBDC, and how that affects liquidity, while streamlining the adjustment rules and payment selection mechanisms.

One key change is in how portfolios are structured. While the original model included a four-asset allocation, with a less liquid store-of-value asset A adjusted over time based on return differences, we omit this asset here. Removing asset A gets rid of a slow-moving state variable that influences saving but does not directly affect the short-term liquidity interactions that drive CBDC adoption, payment shifts, and policy review. By doing this, the focus remains on fast-moving liquidity decisions instead of long-term wealth building. By keeping only liquid assets like C, (Cash), D, (Deposits), and K, (CBDC), we keep the model simple without losing the main ways it works.

Each consumer i at day t holds three liquid positions: cash $C_i(t)$, deposits $D_i(t)$, and CBDC balances $K_i(t)$. Total liquid wealth is therefore:

$$W_i(t) = C_i(t) + D_i(t) + K_i(t).$$

Consumers receive a monthly income flow κ_i and have daily spending needs $s_i(t)$, which come from a log-normal distribution that reflects Brazilian spending habits. Unlike Ramadiah et al. (2021), who model consumers choosing how much to spend, we treat daily spending as a given. This lets us focus our computing power on understanding CBDC adoption and policy optimization.

Another major difference is in the consumer's behavioral foundation. The original model uses gradual adjustment rules for asset allocation and random draws when multiple payment methods are available. We replace this with simpler, more deterministic behavioral rules. The main assumption is that simpler, more responsive rules, offer a stable and realistic way to model consumer responses when policy itself is developing and allow faster convergence in the reinforcement learning process.

To settle a purchase of amount (p) with merchant (j), the agent follows a rule-based payment hierarchy:

1. If the transaction is offline (10% chance) and $C_i(t) \geq p$, pay with cash.
2. Else if the merchant takes CBDC ($A_j^{\text{CBDC}} = 1$) and the consumer has a wallet ($\text{wallet}_i = 1$), then:
   (a) If $K_i(t) \geq p$, pay with CBDC.
   (b) Else attempt to top up: convert a fraction $\phi = 0.25$ of deposits to CBDC, up to the wallet cap $\bar{K}$, and pay with CBDC if $K_i(t) \geq p$ after topping-up.

3. Else if $D_i(t) \geq p$, pay with card (deposits). Card payments have a 2% failure rate $\epsilon_{\text{card}} = 0.02$ representing network issues, authorization delays, or technical failures that are common in traditional payment systems, whereas CBDC payments never fail ($\epsilon_{\text{CBDC}} = 0$).

4. Else, the payment fails.

If CBDC balances are not enough, the top-up rule is:

$$\Delta K_i(t) = \min\left\{ \phi \cdot D_i(t),\ p - K_i(t),\ \bar{K} - K_i(t) \right\},$$

where $\bar{K}$ is the central bank's CBDC wallet cap. The corresponding adjustment to deposits is $D_i(t) \to D_i(t) - \Delta K_i(t)$.

Like the wallet adoption model by Ramadiah et al. (2021), we model wallet adoption using network effects. However, we use a more explicit logistic diffusion process that adds explicit policy-driven incentives and adoption rate caps:

$$\Pr(\text{wallet}_i(t) = 1) = \min\left\{ \sigma\left(\alpha_0 + \alpha_1 \bar{w}(t-1) + \alpha_2 \bar{A}(t-1) + \beta \cdot \frac{\bar{K} - 8000}{10000} \right),\ \delta_c^{\max} \right\},$$

where $\bar{w}(t)$ is the consumer adoption rate (the portion of consumers with wallets), $\bar{A}(t)$ is the merchant acceptance rate (the portion of merchants accepting CBDC), $\bar{K}$ is the CBDC wallet cap, $\sigma(z) = 1/(1 + e^{-z})$ is the logistic function, and $\delta_c^{\max} = 0.005$ is a daily cap on the adoption rate that stops unrealistically fast spread overnight. The term $\beta \cdot (\bar{K} - 8000)/10000$ shows that higher wallet caps are more attractive: consumers are more likely to adopt if they feel transaction limits are less strict.

Balance dynamics follow:

$$\begin{aligned}
D_i(t+1) &= D_i(t) + \mathbb{1}_{\text{income day}} \cdot \kappa_i - \text{card_payments}_i(t) - \text{topups}_i(t), \\
C_i(t+1) &= C_i(t) - \text{cash_payments}_i(t), \\
K_i(t+1) &= K_i(t) + \text{topups}_i(t) - \text{cbdc_payments}_i(t),
\end{aligned}$$

where income is received every 30 days.

In this simplified model, deposits do not earn interest and cash only decreases through spending, as we do not explicitly model inflation. This allows us to focus on how liquidity is reallocated rather than on differences in returns.

This way of modeling consumer behavior captures the key ways CBDC gets adopted, how people switch payment methods, and how money moves between deposits and CBDC. At the same time, it keeps the model easy to run for agent-based simulations and reinforcement learning.

3.2.2 Merchants

Following Ramadiah et al. (2021), the merchant block is build using a simple, rule-based approach. In the Drex Pilot, whether a merchant accepts *Drex de Varejo* (DVt/MEt) depends on

the financial institution and slowly expands as institutions roll out wallet and payment systems. To model this, each merchant initially accepts cash. Over time, some merchants, represented by a share P_B accept card/deposit payments from the start, while another group, an initial fraction P_K, accepts CBDC.

Unlike in Ramadiah et al. (2021), where only merchants accepting cards could also accept CBDC, our model lets any merchant accept CBDC based on customer demand, regardless of whether they accept cards. Each merchant j tracks the proportion of customers who have CBDC wallets, shown as $z_j(t)$. A non-accepting merchant j adopts CBDC at time t with daily probability:

$$\Pr\Big(A_j^{\text{CBDC}}(t) = 1 \;\Big|\; A_j^{\text{CBDC}}(t-1) = 0\Big) = \min\left\{h\big(z_j(t-1), \bar{K}(t)\big),\ \delta_m^{\max}\right\},$$

where the adoption function is:

$$h(z, \bar{K}) = \frac{1}{1 + \exp\left(-k\left(z - z_0 \cdot \left(1 - 0.4 \cdot \frac{\bar{K}-8000}{10000}\right)\right)\right)},$$

Here, $k = 5.0$ controls the steepness, and $z_0 = 0.30$ is the baseline threshold. The threshold goes down if average wallet caps are higher:: $z_0^{\text{adj}} = z_0 \cdot (1 - 0.4 \cdot (\bar{K} - 8000)/10000)$.This means if customers can hold more CBDC, merchants need fewer customers using CBDC to decide to adopt it. Also, the daily cap on adoption goes up with wallet caps: $\delta_m^{\max}(\bar{K}) = 0.008 \cdot (1 + 0.5 \cdot (\bar{K} - 8000)/10000)$. These factors show how central bank policy directly influences merchant adoption: when customers can hold more CBDC, merchants see more potential transactions and are quicker to adopt.

This setup shows that adoption is a slow, decentralized process tied to observed customer demand, not to current payment systems. Merchants do not change prices or inventory. Their job is to create acceptance issues that impact CBDC spread through a demand-driven adoption system sensitive to central bank policy.

3.2.3 Commercial Bank

The commercial bank is modeled in a reduced-form manner that preserves the key balance-sheet mechanisms relevant for CBDC introduction. Each bank b holds deposits $D_b(t)$ from its customers and invests in risky loans $X_b(t)$. The model ignores reserve requirements, interbank markets, and details of CBDC issuance, focusing instead on how removing deposits affects the economy.

Deposits change over time as customers get income, make payments, and switch deposits to CBDC:

$$D_b(t+1) = D_b(t) + \Delta D_b(t),$$

where $\Delta D_b(t)$ covers money coming in from monthly income minus card payments and CBDC top-ups. When a customer moves deposits to CBDC using the top-up rule $\phi D_i(t)$, the bank sees

an immediate drop in deposits, putting pressure on its funding.

The bank invests deposits in a risky loan portfolio $X_b(t)$ that gives changing daily returns. The return structure makes higher leverage appealing:

$$r_{X,b}(t) \sim \mathcal{N}\left(\frac{\mu_X}{252} \cdot (1 + 0.002 \cdot \max(0, \Gamma_b(t) - 10)), \sigma_X\right),$$

where $\mu_X = 0.2791$ is the annual mean loan return, $\sigma_X = 0.00162$ is daily volatility (calibrated to Brazilian commercial bank loan portfolios), and the term $0.002 \cdot \max(0, \Gamma_b(t) - 10)$ is a leverage bonus: banks get better returns when they use more leverage, showing the drive to lend more compared to their equity.

Daily profits are calculated as:

$$\pi_b(t) = X_b(t) \cdot r_{X,b}(t) - D_b(t) \cdot \frac{r_D}{252},$$

where $r_D = 0.0049$ is the annual deposit rate paid to consumers. In this model, consumer deposits do not actually earn interest (as mentioned in the consumer section); this difference shows the bank's cost of funding without directly modeling interest payments to households. Profits add to equity following the equation:

$$E_b(t + 1) = E_b(t) + \pi_b(t).$$

Banks face a regulatory leverage ceiling:

$$\Gamma_b(t) = \frac{X_b(t)}{E_b(t)} \leq \bar{\Gamma}(t),$$

where $\bar{\Gamma}(t)$ is the highest leverage ratio set by the central bank as a policy tool. Banks act aggressively, aiming for leverage close to the limit to get the best returns:

$$\Gamma_b^* = \min\left\{13.5, \bar{\Gamma}(t) - 0.2\right\},$$

where the desired leverage $\Gamma_b^* = 13.5$ reflects aggressive profit-seeking behavior, but banks maintain a 0.2 safety buffer below the regulatory cap to avoid violations. The loan portfolio slowly moves towards this goal:

$$\Delta X_b(t) = \alpha \cdot \left[\Gamma_b^* \cdot E_b(t) - X_b(t)\right],$$

where $\alpha = 0.05$ is the adjustment speed parameter.

This structure allows for a clear policy effect: if the Central Bank lowers $\bar{\Gamma}(t)$ (e.g., in response to rising CBDC adoption and deposit outflows), banks must lower their target leverage, slowing loan growth. On the flip side, if CBDC wallet caps go up and adoption speeds up, deposit removal gets worse, forcing banks to reduce leverage even if the regulatory cap stays the same. The model captures the balance between encouraging CBDC adoption (which cuts bank funding) and keeping the banking system stable (which needs sufficient equity).

The model simplifies by not including immediate liquidity issues, reserve rules, or Central Bank funding. It assumes banks have enough money to handle payments smoothly. This simplification helps us focus on how CBDC-driven deposit shifts affect solvency and leverage, which are the main concerns for policy in the reinforcement learning setup.

3.2.4 Central Bank

The central bank is modeled as a policymaker with two primary instruments to influence CBDC adoption and financial stability:

1. **Consumer wallet cap** ($\bar{K}$): The maximum CBDC balance any individual consumer can hold. This parameter directly constrains the aggregate stock of retail CBDC, mitigates bank disintermediation risk, and influences adoption by affecting the perceived utility of CBDC for households and firms.

2. **Maximum regulatory leverage** ($\bar{\Gamma}$): The ceiling on banks' loan-to-equity ratio. This constraint limits credit expansion and serves as a macroprudential tool to preserve financial stability when deposits migrate to CBDC.

These instruments create a fundamental policy tradeoff: higher wallet caps encourage CBDC adoption by increasing its attractiveness to consumers and merchants, but accelerate deposit outflows from traditional banks. Conversely, stricter leverage constraints protect bank solvency but may constrain credit availability.

To explore central bank decision-making under different regulatory approaches, three policy regimes are implemented:

Baseline (Fixed Policy): Both instruments remain constant throughout the 730-day simulation period at $\bar{K} = 10{,}000$ BRL and $\bar{\Gamma} = 14.0$. This represents a passive regulatory stance where CBDC adoption evolves without active policy intervention.

Adaptive Heuristic: Rule-based adjustments respond to system conditions. The wallet cap increases to $\bar{K} = 15{,}000$ BRL when consumer adoption exceeds 50%, while the leverage constraint tightens to $\bar{\Gamma} = 12.0$ when average bank leverage surpasses 13.0. This captures threshold-based regulatory responses.

Reinforcement Learning Policy: A dynamic policy schedule adjusts $\bar{K}(t)$ and $\bar{\Gamma}(t)$ based on observed system states. The policy is learned ex-ante through reinforcement learning and subsequently deployed to balance adoption speed against financial stability. The learning framework is detailed in Section 3.3.

This three-scenario design enables systematic comparison between static regulation (baseline), reactive rules (heuristic), and data-driven adaptive policy (reinforcement learning), isolating the value of learning-based optimization in navigating the CBDC policy tradeoff space.

3.3 Reinforcement Learning Model Design

As stated before, in this extended CBDC agent-based framework, we integrate a reinforcement learning (RL) layer that enables adaptive, data-driven policy-making by the central bank. This

approach addresses a fundamental challenge in CBDC policy design: the absence of historical data precludes the use of conventional econometric methods for policy calibration. RL provides a way to figure out good policy schedules by running simulations. This lets the central bank try out different policies and learn how the system reacts as things change.

The main question our RL framework asks is: *can a learning agent discover superior policy trajectories compared to fixed regulations or simple heuristics?* The objective is not to replace human policymakers but rather to help policy analysis by systematically looking for less obvious policy schedules. These schedules balance competing goals as quickly adopting CBDC, keeping the payment system working well, and maintaining stability in the banking sector.

The RL part runs in a simulated agent-based model (ABM). This model includes consumers, merchants, and commercial banks, as shown before, all following rules based on Brazilian financial data. The central bank agent looks at what's happening in the system (e.g. how many consumers and merchants are using CBDC, the ratio of CBDC to bank deposits, how often payments succeed, and banks' average loan-to-equity ratios). Based on these observations, it selects policy parameter combinations $(\bar{K}(t), \bar{\Gamma}(t))$ to get the best overall outcome.

This hybrid architecture implements a two-phase methodology:

1. **Offline training phase:** The agent learns through repeated simulation episodes, exploring different policy sequences and updating its knowledge based on observed outcomes. This stage produces a policy that maps system states to actions.

2. **Online deployment phase:** The learned policy is applied to a new simulation run without any more learning. This is like a central bank putting a pre-trained adaptive policy into practice in the real world.

Keeping the training and deployment separate is important. It ensures that when we compare how well the three scenarios (baseline, heuristic, RL) perform, we're seeing how effective the policies truly are, not just how much the agent is still learning during the evaluation. The next sections go into detail about how we set up the system's state, the available actions, how we designed the reward system, and the learning method we used.

3.3.1 Action Space

The RL agent, acting as the central bank, picks policy settings daily. The possible actions come from five combinations of two tools:

- **Consumer wallet cap** $\bar{K} \in \{10000, 12000, 14000, 16000, 18000\}$ BRL: the maximum CBDC balance any individual consumer can hold.

- **Maximum bank leverage** $\bar{\Gamma} \in \{12.5, 13.0, 13.5, 14.0, 14.5\}$: the regulatory limit on how much commercial banks can lend compared to their equity.

These combinations give us a range of policies, from strict (low wallet caps, tight leverage limits) to more flexible (high caps, relaxed limits). Each action directly changes the central bank's policy variables and consumer wallet capacities in the simulation. This immediately impacts how attractive the wallet cap is for consumers, influencing their adoption decisions, and how banks adjust their balance sheets based on the new leverage constraints.

3.3.2 State Representation

The state at time t, denoted $\mathbf{s}(t)$, aggregates key system conditions into a five-dimensional vector:

1. **Consumer wallet adoption rate** $\bar{w}(t) \in [0.0, 1.0]$: share of consumers holding CBDC wallets;
2. **Merchant CBDC acceptance rate** $\bar{A}(t) \in [0.0, 1.0]$: share of merchants accepting CBDC payments;
3. **CBDC-to-deposits ratio** $(K_{\text{total}}/D_{\text{total}})(t) \in [0.0, 0.5]$: total amount of CBDC circulating compared to the total bank deposits;
4. **Payment success rate** $\text{PSR}(t) \in [0.75, 1.0]$: proportion of attempted transactions successfully completed;
5. **Average banking-sector leverage** $\bar{\Gamma}(t) \in [8.0, 16.0]$: mean loan-to-equity ratio across commercial banks.

These numbers capture the main things happening with payments, how CBDC spreads, and financial stability. They are also kept small enough to work with tabular Q-learning. Each component is split into 3 sections within its given range, making a total of $3^5 = 243$ possible states. This general grouping helps the agent learn good policies even with limited training episodes, rather than focusing on tiny differences between states.

3.3.3 Reward Function

The reward function guides the agent towards decisions that help CBDC become popular quickly while keeping the financial system stable and payments reliable. The reward at time t focuses on how fast CBDC is adopted and whether certain goals are met. It only applies big penalties if regulations are broken:

$$
\begin{aligned}
R(t) = & \underbrace{100 \cdot \left(\Delta \bar{w}(t) + \Delta \bar{A}(t)\right)}_{\text{Velocity reward}} \\
& + \underbrace{\mathbb{1}_{\{0.19<\bar{w}(t)<0.21\}} \cdot 5.0 + \mathbb{1}_{\{0.49<\bar{w}(t)<0.51\}} \cdot 10.0 + \mathbb{1}_{\{0.79<\bar{w}(t)<0.81\}} \cdot 15.0}_{\text{Milestone bonuses}} \\
& + \underbrace{\mathbb{1}_{\{\bar{w}(t)>0.5, \bar{A}(t)>0.5\}} \cdot 5.0 \cdot \left(\bar{w}(t) \cdot \bar{A}(t)\right)^2}_{\text{Network effects bonus}} \\
& + \underbrace{0.5 \cdot \left(\bar{w}(t) + \bar{A}(t)\right) + 2.0 \cdot \frac{K_{\text{total}}}{D_{\text{total}}}(t)}_{\text{Progress and usage rewards}} \\
& - \underbrace{5.0 \cdot \max\left(0, \bar{\Gamma}(t) - 14.0\right)^2}_{\text{Leverage penalty}} \\
& - \underbrace{3.0 \cdot \max\left(0, \frac{K_{\text{total}}}{D_{\text{total}}}(t) - 0.4\right)^2}_{\text{CBDC ratio penalty}} \\
& - \underbrace{10.0 \cdot \max\left(0, 0.95 - \text{PSR}(t)\right)}_{\text{Payment failure penalty}},
\end{aligned}
$$

where $\Delta \bar{w}(t) = \bar{w}(t) - \bar{w}(t-1)$ and $\Delta \bar{A}(t) = \bar{A}(t) - \bar{A}(t-1)$ show the daily changes in how many consumers and merchants are using CBDC.

This reward structure reflects several design choices:

- **Velocity emphasis**: The dominant reward component (100× coefficient) favors the rate of change in adoption, not just the current level. This pushes for policies that speed up the spread of CBDC.
- **Sparse milestone bonuses**: When adoption reaches key points (20%, 50%, 80%), there are large rewards. This gives clear feedback for good policy timing.
- **Network effects**: After both consumers and merchants pass 50% adoption, the rewards grow exponentially. This acknowledges how these two-sided markets reinforce each other.
- **Shaped progress rewards**: Smaller, continuous rewards for adoption levels and how much CBDC is in use ensure the agent gets feedback even early on.
- **Harsh constraint penalties**: Large penalties that grow quadratically are applied only when regulatory limits are broken (leverage > 14.0, CBDC ratio > 0.4, payment success < 0.95). This strongly discourages actions that cause instability.

This uneven structure, with big rewards for speed and stiff penalties for rule breaking, shows that the policy prioritizes making CBDC adoption as fast as possible, as long as financial stability is maintained. These objectives are not treated as equal trade-offs.

3.3.4 Learning Algorithm

The agent uses a modified Q-learning with experience replay. This is a model-free learning method that works well for situations with a fixed, limited number of states and actions. The Q-table $Q(s, a)$ tracks the expected total reward from taking action a in state s and then following the learned policy afterwards. At each step:

1. The current state $\mathbf{s}(t)$ is categorized and matched to an entry in the Q-table.

2. An action $a(t)$ is chosen using an ε-greedy policy: with probability ε, a random action is picked (exploration); otherwise, the action that gives the highest $Q(\mathbf{s}(t), \cdot)$ is selected (exploitation).

3. The simulation moves forward one day, giving the next state $\mathbf{s}(t+1)$ and the reward $R(t)$.

4. The transition $(\mathbf{s}(t), a(t), R(t), \mathbf{s}(t+1))$ is stored in a replay buffer (capacity 10,000 transitions).

5. A batch of 64 transitions is randomly picked from the buffer, and the Q-table is updated using the temporal-difference rule:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[R + \gamma \max_{a'} Q(s', a') - Q(s, a) \right],$$

   where $\alpha = 0.1$ is the learning rate and $\gamma = 0.95$ is the discount factor.

The exploration rate ε decays geometrically from 1.0 to 0.15 over episodes (decay rate 0.98), gradually shifting from exploration to exploitation. This means the agent starts by exploring a lot and then gradually focuses on using what it has learned. Experience replay helps the agent learn more efficiently by decorrelating sequential transitions, which breaks up any short-term patterns and allows it to learn from a wider range of past situations.

3.3.5 Training and Deployment

Our RL agent goes through two clear stages. First, in offline training, the agent trains over 50 rounds, each simulating 250 days (about 8 months) of how CBDC might spread. At the start of each round, the environment resets to its starting point with fresh populations of agents. The agent tries out different policies, updates its Q-table daily, and tracks how well its learning improves.

After training, the learned policy is then used in a separate 730-day (2-year) simulation, with no further exploration $\varepsilon = 0$ (purely exploitative, meaning it only uses what it learned). The environment runs on its own, and the agent applies the learned policy by picking the actions that are expected to yield the highest long-term reward for each state. This mirrors how a central bank might make decisions: define a policy framework during a planning period, then apply it consistently when it's put into action. Importantly, the deployment stage uses the same simulation length as the baseline and heuristic scenarios, allowing for a direct performance comparison among all three policy approaches.

3.4 Model Calibration

3.4.1 Consumers

Consumer agents ($N_c = 10{,}000$) are calibrated using IBGE POF 2017 data. Income follows a lognormal distribution with $\mu = \ln(1248.83) \approx 7.13$ and $\sigma = 0.5$, while daily spending averages BRL 28.12 with a standard deviation of 30% of the mean. Initial wealth equals 8.42% of annual income, allocated 5% to cash and 95% to deposits.

CBDC wallets are capped at BRL 10,000 in the baseline scenario, with consumers topping up at 25% of deposits when needed. Wallet adoption follows a logistic model with parameters $(\alpha_0, \alpha_1, \alpha_2) = (-4.0, 3.0, 2.0)$, augmented by a wallet cap attractiveness term $\beta \cdot (\bar{K} - 8000)/10000$ where $\beta = 3.0$, and capped at a 0.5% daily adoption rate. Consumers get paid monthly (every 30 days) and have a 10% chance of paying with cash offline. Card payments fail 2% of the time, but CBDC payments always go through, making CBDC more reliable for digital transactions.

Table 1: Consumer Agent Parameters

Parameter	Description	Value	Source
N_c	Number of consumers	10,000	Model assumption
Income and Expenditure			
μ_{income}	Income: lognormal location	$\ln(1248.83) \approx 7.13$	POF 2017
σ_{income}	Income: lognormal scale	0.5	Calibrated
$\bar{s}$	Mean daily spending	BRL 28.12	POF 2017
σ_s	Std. dev. daily spending	30% of mean	Calibrated
Initial Wealth			
w_0	Initial wealth	0.0842× annual income	POF 2017
Cash share	Cash allocation	5%	POF 2017
Deposit share	Deposit allocation	95%	POF 2017
CBDC Wallet			
ϕ	Top-up fraction	25% of deposits	Calibrated
$\bar{K}$	Wallet cap (baseline)	BRL 10,000	Model Assumption
Adoption			
$(\alpha_0, \alpha_1, \alpha_2)$	Logistic adoption parameters	$(-4.0, 3.0, 2.0)$	Calibrated
β	Wallet cap attractiveness	3.0	Calibrated
$\delta_c^{\max}$	Daily adoption cap	0.5% of N_c	Calibrated
Payment Behavior			
τ_{income}	Income frequency	30 days	Model assumption
p_{offline}	Offline purchase probability	10%	Model assumption
ϵ_{card}	Card payment failure rate	2%	Model assumption
ϵ_{CBDC}	CBDC payment failure rate	0%	Model assumption

3.4.2 Merchants

Merchant agents (N_m = 1,000) begin with universal cash acceptance (100%), 70% card acceptance, and 10% CBDC acceptance, calibrated using Pix adoption rates as a proxy for digital

payment diffusion (BCB Pix Statistics, 2023). [2] Merchants keep an eye on how many of their customers have CBDC wallets, denoted z. They decide to take CBDC based on a logistic rule, with a steepness $k = 5.0$ and baseline critical-mass threshold $z_0 = 0.30$. The threshold is adjusted downward based on average wallet caps: $z_0^{\text{adj}} = z_0 \cdot (1 - 0.4 \cdot (\bar{K} - 8000)/10000)$. This means if caps are higher, merchants need fewer CBDC users to start accepting it. The daily adoption cap also goes up with wallet caps: $\delta_m^{\max}(\bar{K}) = 0.008 \cdot (1 + 0.5 \cdot (\bar{K} - 8000)/10000)$, so merchants adopt faster when conditions are good. We don't include transaction fees in our model, which matches how Pix works (it has no fees).

Table 2: Merchant Agent Parameters

Parameter	**Description**	**Value**	**Source**
N_m	Number of merchants	1,000	Model assumption
Initial Acceptance Rates			
p_{cash}^0	Initial cash acceptance	100%	Model assumption
p_{card}^0	Initial card acceptance	70%	Pix adoption proxy
p_{CBDC}^0	Initial CBDC acceptance	10%	Pix adoption proxy
CBDC Adoption Function			
Function type	Adoption model	Logistic sigmoid	Model specification
k	Logit steepness	5.0	Calibrated
z_0	Baseline threshold	0.30	Calibrated
Threshold adj.	Cap-dependent threshold	Decreases with $\bar{K}$	Model logic
$\delta_m^{\max}$	Daily adoption cap (base)	0.8% of N_m	Calibrated
Cap adjustment	Higher cap bonus	Increases with $\bar{K}$	Model logic

3.4.3 Commercial Banks

Commercial bank agents ($N_b = 3$) are calibrated using BCB time series data (Sistema Gerenciador de Séries Temporais - SGS): deposit rates of 0.49% annually and loan returns of 27.91% annually with daily volatility of 0.162%. Banks maintain an initial loan-to-deposit ratio of

[2] Pix is Brazil's instant payment system launched in November 2020, enabling real-time 24/7 transfers with immediate settlement (Banco Central do Brasil, 2025). It became Brazil's dominant digital payment method within two years, making its adoption trajectory a relevant proxy for CBDC diffusion dynamics.

$L_0/D_0 = 0.8$, initial leverage $\lambda_0 = 12.0$, and regulatory maximum leverage $\bar{\Gamma}_{\max} = 14.0$.

Banks aim for profits, targeting a leverage of $\lambda^* = 13.5$ but maintaining a safety buffer of 0.2 below the regulatory ceiling: $\lambda_{\text{target}} = \min\{13.5, \bar{\Gamma}(t) - 0.2\}$. Loan portfolios adjust toward this target at speed $\alpha = 5\%$ per day. Loan returns incorporate a leverage bonus: banks earn an additional 0.2% return for each leverage point above 10, reflecting the profit incentive for aggressive balance sheet expansion.

Table 3: Commercial Bank Parameters

Parameter	**Description**	**Value**	**Source**
N_b	Number of banks	3	Model assumption
Interest Rates and Returns			
r_D	Deposit rate (annual)	0.49%	BCB SGS 195
μ_X	Loan return mean (annual)	27.91%	BCB SGS 2073
σ_X	Loan return volatility (daily)	0.162%	BCB SGS 2073
Leverage bonus	Additional return per leverage pt.	0.2% (above $\Gamma = 10$)	Model specification
Balance Sheet			
L_0/D_0	Initial loan-to-deposit ratio	0.8	Model assumption
λ_0	Initial leverage	12.0	Model assumption
λ^*	Desired leverage (greedy)	13.5	Model assumption
$\bar{\Gamma}$	Maximum regulatory leverage	14.0	Model assumption
Safety buffer	Buffer below ceiling	0.2	Model specification
α	Leverage adjustment speed	5% per day	Calibrated

3.4.4 Central Bank

The central bank sets two policy levers: the caps on consumer CBDC wallets ($\bar{K}$) and the highest acceptable leverage for banks ($\bar{\Gamma}$). The baseline wallet cap is BRL 10,000, and the baseline maximum leverage is 14.0. CBDC itself doesn't earn interest ($r_{\text{CBDC}} = 0\%$). Standard monetary policy works separately from how CBDC policy is set. As mentioned earlier, the model doesn't deal with reserve requirements or ways to give banks liquidity, so we can focus on how less

deposit money for banks affects their stability and how much credit is available.

Table 4: Central Bank Policy Parameters

Parameter	**Description**	**Value**	**Source**
$\bar{K}$	Consumer CBDC wallet cap (baseline)	BRL 10,000	Model Assumption
$\bar{\Gamma}$	Maximum regulatory leverage (baseline)	14.0	Basel III
r_{CBDC}	Interest on CBDC	0%	Model assumption
r_{SELIC}	SELIC rate (annual)	10%	BCB Monetary Policy
Policy scope	Abstracted features	No reserves, liquidity facilities	Model simplification

3.4.5 Reinforcement Learning and Simulation Timeline

The reinforcement learning (RL) controller fine-tunes policy within a defined range of states, considering adoption rates, acceptance levels, the CBDC-to-deposits ratio, payment success, and bank leverage. We use Q-learning with experience replay to figure out the best policy plans. These plans are designed to maximize how fast CBDC is adopted while keeping the financial system stable. The reward system heavily favors swift adoption, gives big bonuses for hitting important milestones, and penalizes regulatory rule breaking with sharp quadratic costs.

Table 5: Reinforcement Learning Parameters

Parameter	Description	Value	Source
State Space (5D vector):			
Consumer adoption	Rate, normalized	[0.0, 1.0]	Model
Merchant acceptance	Rate, normalized	[0.0, 1.0]	Model
CBDC/deposits ratio	System balance	[0.0, 0.5]	Normalized
Payment success rate	System reliability	[0.75, 1.0]	Constraint
Bank leverage	Average leverage	[8.0, 16.0]	Actual values
State bins	Discretization	3 bins/dim	Design choice
Total states	State space size	243	Derived
Q-Learning Hyperparameters:			
Learning rate	Step size	0.1	Standard
Discount factor	Future rewards weight	0.95	Long-term focus
Initial exploration	Starting epsilon	1.0	Full exploration
Min exploration	Final epsilon	0.15	Convergence
Decay rate	Epsilon annealing	0.98/episode	Gradual
Experience Replay:			
Buffer size	Memory capacity	10,000	Stability
Batch size	Training batch	64	Standard
Reward Components:			
Velocity	Adoption rate changes	Scaled by 100	Primary
Milestones	Threshold bonuses	5.0, 10.0, 15.0	Sparse
Network effects	Joint adoption bonus	Quadratic if both > 0.5	Nonlinear
Leverage penalty	Regulatory violation	Quadratic above 14.0	Constraint
CBDC ratio penalty	Excess displacement	Quadratic above 0.4	Constraint
Payment penalty	System failure	Linear below 0.95	Critical
Action Space (5 discrete actions):			
Action 0	Conservative policy	(10k; 12.5)	Safe
Action 1	Moderate-low policy	(12k; 13.0)	Balanced
Action 2	Moderate policy	(14k; 13.5)	Neutral
Action 3	Aggressive-low policy	(16k; 14.0)	Growth
Action 4	Aggressive policy	(18k; 14.5)	Maximum

Table 6: Simulation Timeline

Scenario	**Period**	**Purpose**
Baseline	730 days (2 years)	Fixed-policy benchmark
Adaptive Heuristic	730 days (2 years)	Rule-based adaptation
RL Training	50 episodes $\times$ 250 days	Offline Q-learning optimization
RL Deployment	730 days (2 years)	Learned policy evaluation

4 Exploratory Data Analysis

The model's empirical base comes from looking at different data sources. Each source helps us understand a bit of the Brazilian financial world. The point of this initial look is not just to sum up the data, but to find specific targets for calibrating how households, businesses, banks, and the central bank behave in our agent-based model. The statistics below show us the real-world limits and motivations for each type of agent.

4.1 Households

Household behavior is grounded in microdata from the *Pesquisa de Orçamentos Familiares* (POF 2017–2018), from which a total of 57,920 households were analyzed. The income distribution reveals a mean annual income of R$ 14,986 and a median of R$ 8,430.This suggests income is highly concentrated, with a Gini coefficient of 0.5590. Spending habits are even more spread out, with an average yearly spend of R$ 10,263 and a middle value of R$ 4,244, with an expenditure Gini of 0.6553.

The correlation between income and expenditure is moderate (0.5718), meaning that how much people spend does respond to their income, but in a somewhat fixed way. The average propensity to consume, estimated at 0.9566, anchors the specification of the household consumption rule in the model.

All these numbers together help us set up the model's income processes, how spending changes, and saving habits. They also shape how liquid wealth is first shared among consumers.

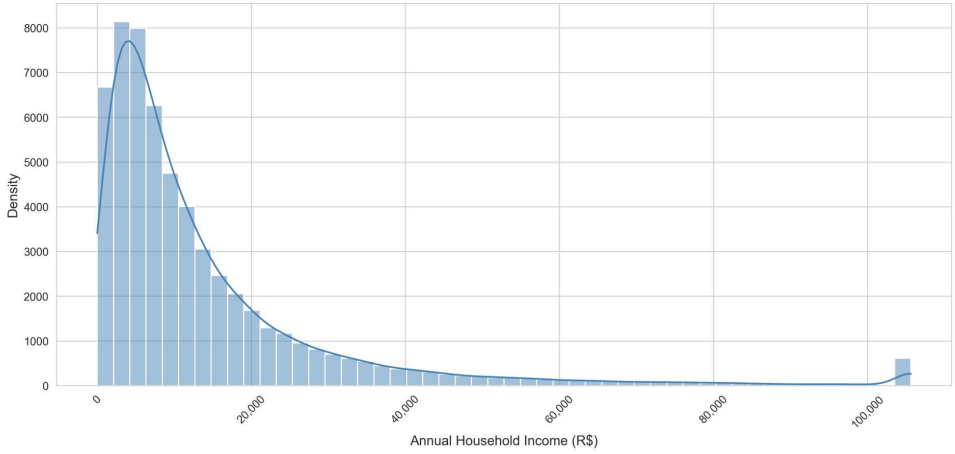

Figure 3: Household Income Distribution.

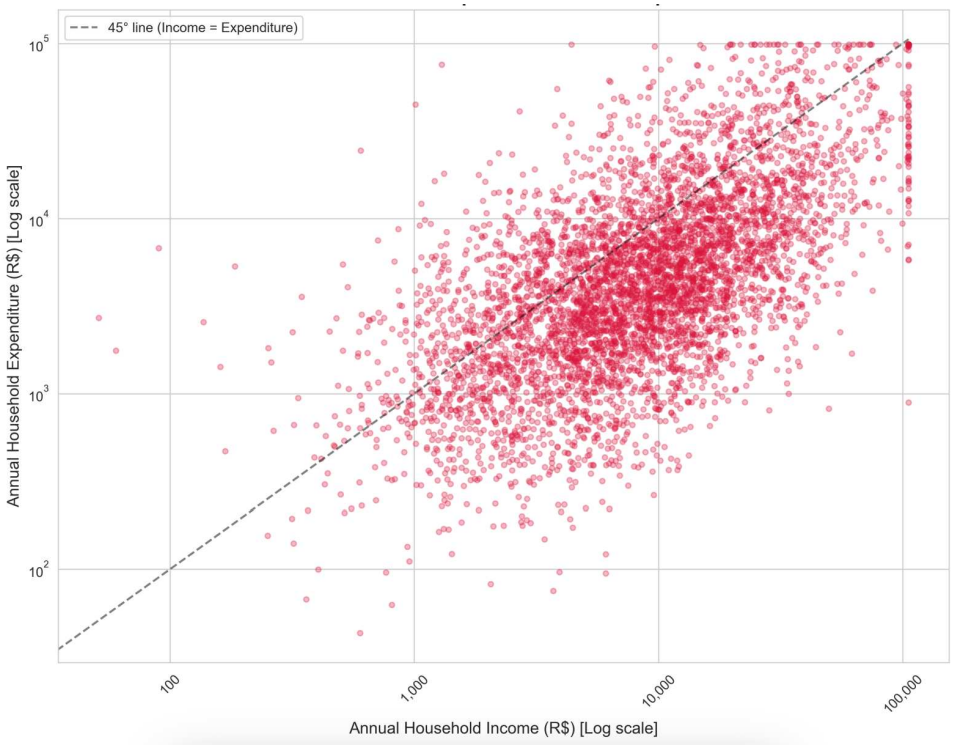

Figure 4: Income and Expenditure Correlation.

4.2 Merchants

We calibrate merchant behavior using overall Pix transaction data from the Central Bank of Brazil. This data gives us a frequent sense of how digital payments are spreading across different types of businesses.

The data shows a very fast initial adoption phase: monthly transaction numbers went from about 29.5 million in November 2020 (the first full month of Pix) to over 1.22 billion by December 2021, and then more than 4.3 billion transactions a month by the end of 2023. By late 2024 and early 2025, growth starts to slow down, with amounts hovering around 5–6.2 billion transactions per month, and total monthly values going past R$2.6 trillion. This pattern of quick growth followed by a gradual leveling off helps determine the shape of how CBDC acceptance will grow.

Initial merchant acceptance rates in the model are set at 70% for card/deposit payments, consistent with longstanding electronic-payment penetration in Brazil, and 10% for CBDC, which is a careful guess for early adoption, based on how Pix started. Cash is universally accepted (100%).

Merchants adopt new payment methods following a logistic curve. The settings for this curve suggest that merchants start to accept CBDC broadly when about 30% of their customer base holds CBDC wallets. To keep the spread realistic, we cap daily adoption at 0.8%. In the model, each merchant looks at its own customer base and decides whether to accept CBDC based on how many of its customers can use CBDC. This mirrors how adoption spread with Pix, driven

by network effects.

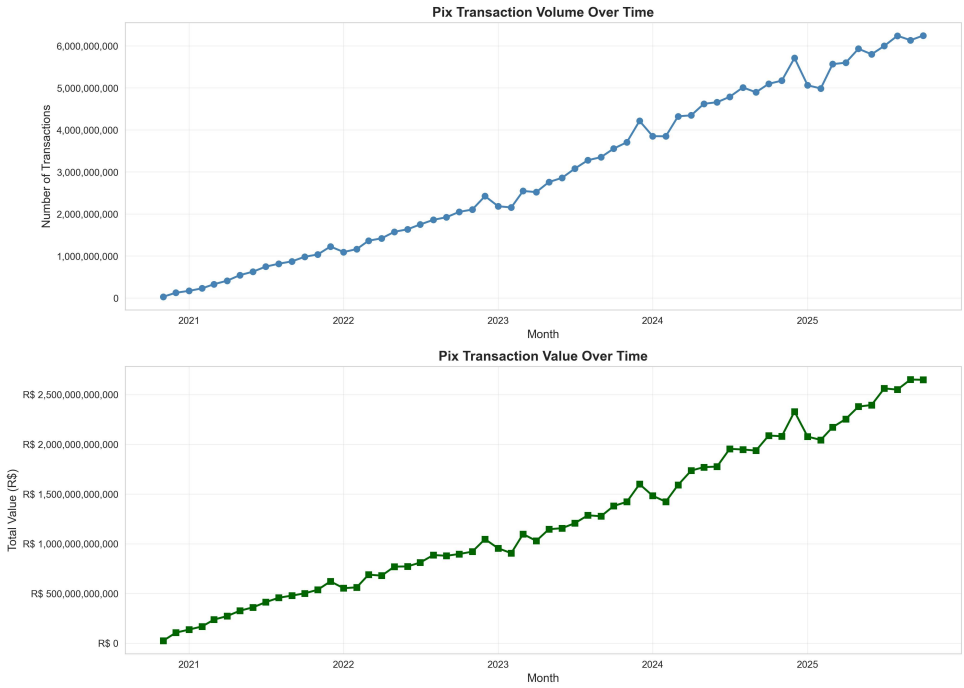

Figure 5: PIX Transactions.

4.3 Banking Sector

Bank calibration employs processed time series from the Sitema Gerenciador de Séries Temporais (SGS) for the deposit rate, credit spreads, and aggregate deposits, along with Central Bank of Brazil's monetary policy data for the Selic rate. [3] The policy rate is set at 10% annual ($r(t) = 0.10$) in the baseline scenario, providing the benchmark for short-term monetary conditions and influencing deposit rates indirectly through the banking system. The deposit-rate series, with a mean daily rate of approximately 0.49% (0.0049), informs the deposit-rate rule.

The credit-spread series supplies the stochastic process for returns on risky assets (bank loans). The annual mean return is 27.91% (0.2791), corresponding to a daily mean of 0.0977% (0.000977), with daily volatility of 0.162% (0.00162).

Aggregate deposit data, combined with Central Bank banking statistics, help us fix the size of bank balance sheets and typical ways banks get funding. The initial bank deposits total approximately BRL 69.46 billion, with the initial loan-to-deposit ratio calibrated at 80% (0.8), yielding initial loans of BRL 55.57 billion. Bank equity is initialized to achieve a leverage ratio of 12.0, with maximum regulatory leverage set at 14.0.

Taken together, these figures guide how banks manage their cash, adjust their borrowing lev-

[3]The SELIC (Sistema Especial de Liquidação e Custódia) rate is Brazil's benchmark policy interest rate, set by the Central Bank and used as the primary monetary policy instrument. It represents the overnight interbank lending rate backed by federal government securities.

els, and generate profits. They ensure that the simplified bank section of our model reflects real Brazilian banking behavior while fitting the model's level of detail and ability to be computed.

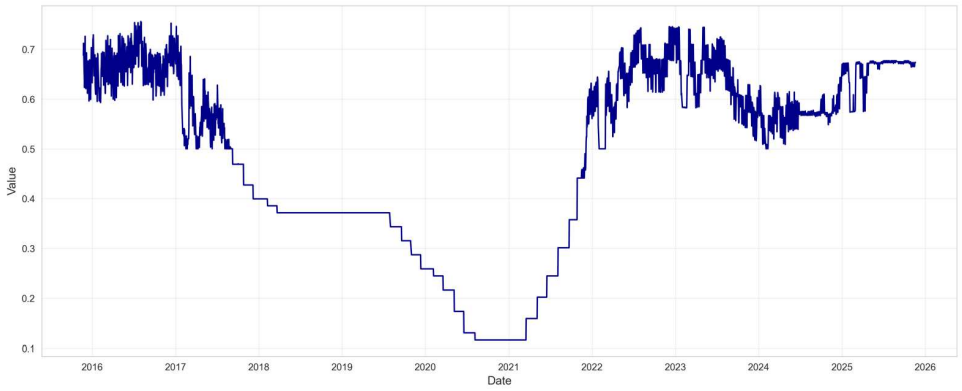

Figure 6: Saving deposits.

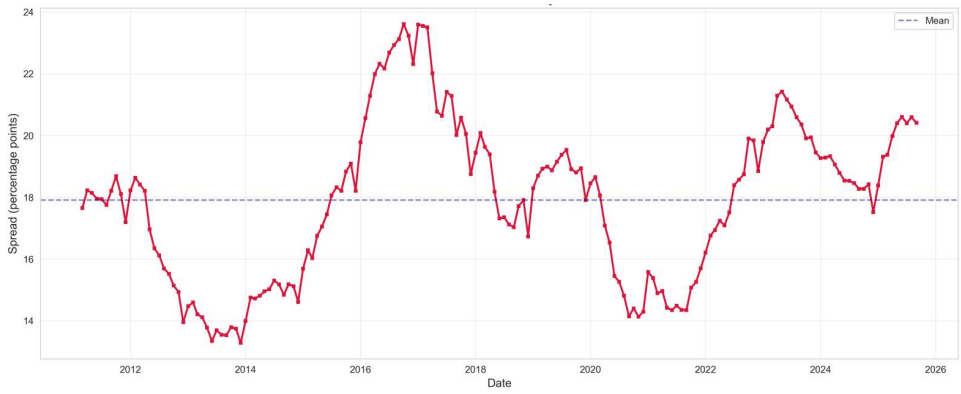

Figure 7: Average spread of credit operations.

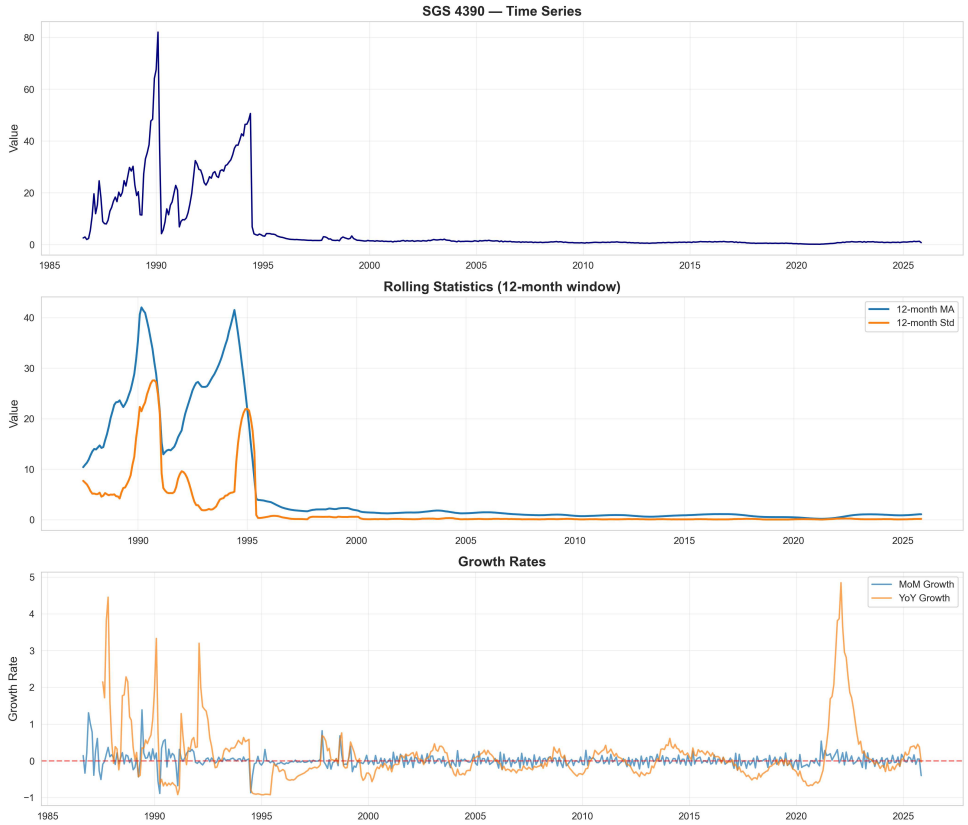

Figure 8: Interest rate accumulated in the month.

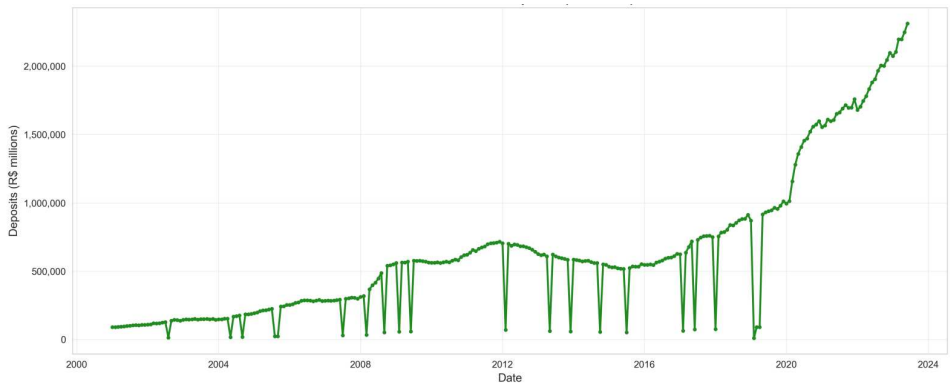

Figure 9: Monthly deposits balance with earnings included.

5 Results

5.1 CBDC Adoption Dynamics

The way consumers and businesses take up new payment systems follows clear trends depending on the rules in place. We see a strong, linked growth between consumer and merchant use, which comes from how these systems connect users and providers. In all three scenarios we looked at, almost all consumers (>97%) were using the new system and all businesses accepted it by day 730. This tells us that once enough people start using the system (getting to a key point around day 100-150), it keeps growing on its own, no matter what specific rules are guiding it.

Consumer adoption consistently shows an S-shaped curve across all situations. Starting from almost zero, it goes up to about 60% by day 200, then speeds up to pass 80% by day 400, before evening out at around 97% (Figure 10). The RL rule set closely follows both the basic and the heuristic approaches. The final adoption rates show very small differences, less than 0.2 percentage points (baseline: 97.39%, heuristic: 97.53%, RL: 97.34%). This convergence suggests that network effects dominate policy instrument effects once adoption momentum builds.

As for businesses, our findings show they accept CBDC sooner and quicker than consumers start using their wallets. Business acceptance hits 80% by day 100–120 across all scenarios and reaches 100% by day 300–350 (Figure 11). This suggests businesses set up the necessary tools ahead of time, paving the way for consumers to follow, instead of waiting for a lot of consumer interest first. This early lead by businesses, followed by consumers catching up, confirms the model's idea of a two-sided market influencing how these new systems are picked up.

The total amount of CBDC in circulation shows clear up-and-down movements, with the size of these swings being tied to the policies in place (Figure 12). The baseline scenario shows circulation oscillating between 20,000–42,000 BRL with a sawtooth pattern reflecting the 30-day income cycle: consumers receive monthly deposits, trigger top-ups when CBDC balances are insufficient for transactions, then gradually draw down balances until the next income event. The adaptive heuristic scenario displays similar oscillatory behavior with slightly higher average circulation (peaks near 44,000 BRL), while the RL policy maintains comparable patterns over its 730-day deployment horizon. Final CBDC circulation differs modestly across scenarios (baseline: 27,156 BRL, heuristic: 28,134 BRL, RL: 26,613 BRL), with all scenarios converging to similar long-run stock levels despite different policy trajectories.

Crucially, the portion of CBDC compared to bank deposits settles between 18–32% across all situations, with noticeable monthly swings due to income cycles. This ratio remains well below the 40% penalty threshold embedded in the RL reward function, which means that even with almost everyone using it, a CBDC focused on transactions with limits on wallet size does not cause a huge shift away from traditional deposits.

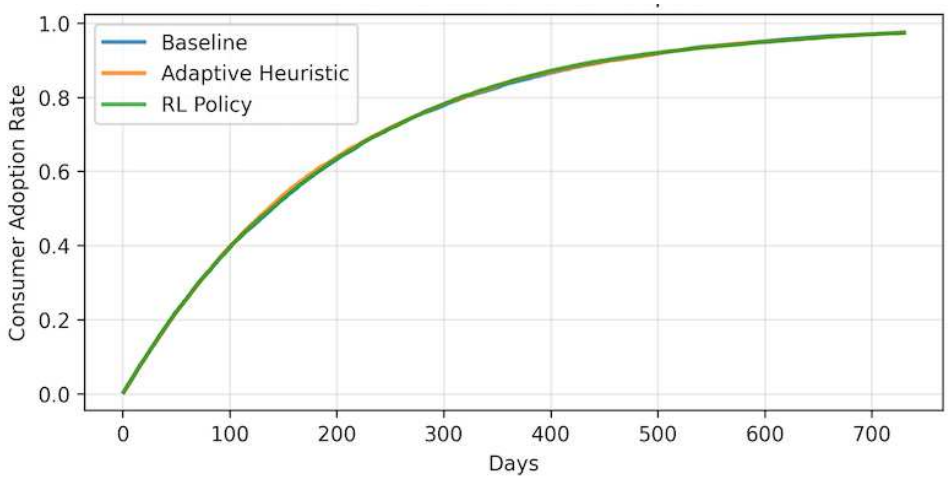

Figure 10: Consumer CBDC Wallet Adoption.

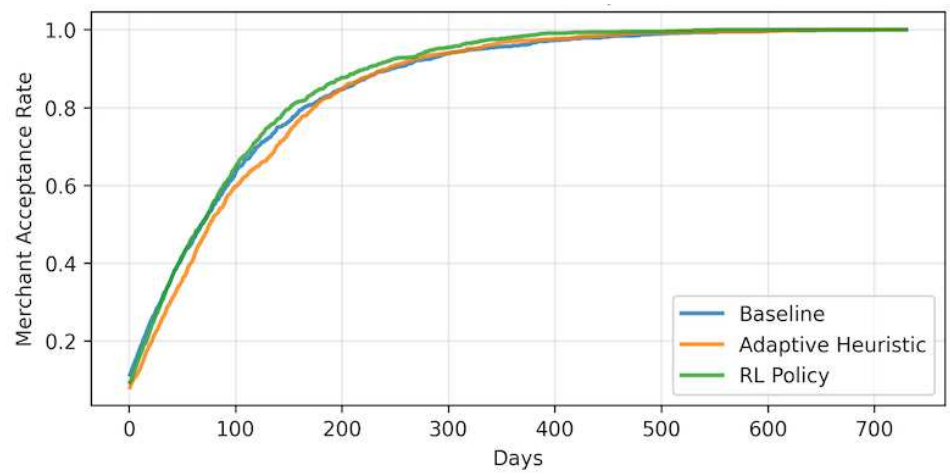

Figure 11: Merchant CBDC Acceptance.

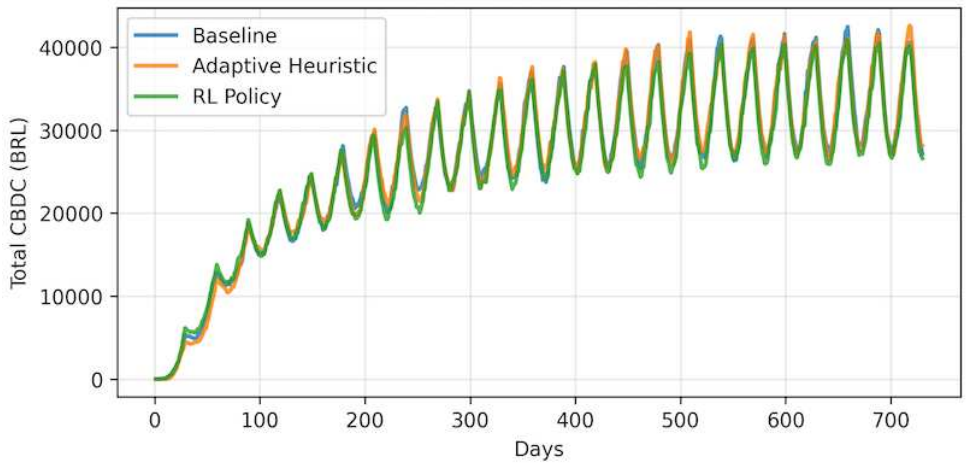

Figure 12: Total CBDC in Circulation.

5.2 Payment System Evolution

The way payment systems spread shows a fundamental change in how people pay over time (Figure 13). Card payments start as the main method and continue to grow in total money spent, going from almost nothing to about 60–65 million BRL by day 730 across all scenarios (Panel A). But, the amount of money spent using CBDC grows even faster once more people start using it, reaching 120–130 million BRL by day 730 and surpassing card values around day 400–500 in the baseline and heuristic scenarios. The RL scenario exhibits similar patterns over its time horizon.

Transaction numbers (Panel B) follow a similar path: card payments settle at about 2 million transactions by day 730, while CBDC transactions grow explosively to exceed 5 million, reflecting both higher adoption rates and the reliability advantage of CBDC (0% failure rate versus 2% for cards). Cash transaction counts remain minimal throughout (<300,000 transactions), constrained by the 10% offline payment probability. This picture goes against worries that cash will quickly disappear, as physical money sticks around for certain situations even as digital payments become common.

Relative change analysis (Panel C) helps us understand how one payment method replaces another: the value of card payments grows by about 22,000–24,000% compared to the start, while cash grows by roughly 3,500–4,000%. This confirms that CBDC mostly takes the place of card-based deposits, rather than getting rid of cash. The similar growth patterns across scenarios suggest that people's choice of payment is influenced more by how many businesses accept it and how much momentum it gains with consumers, rather than small adjustments in policy.

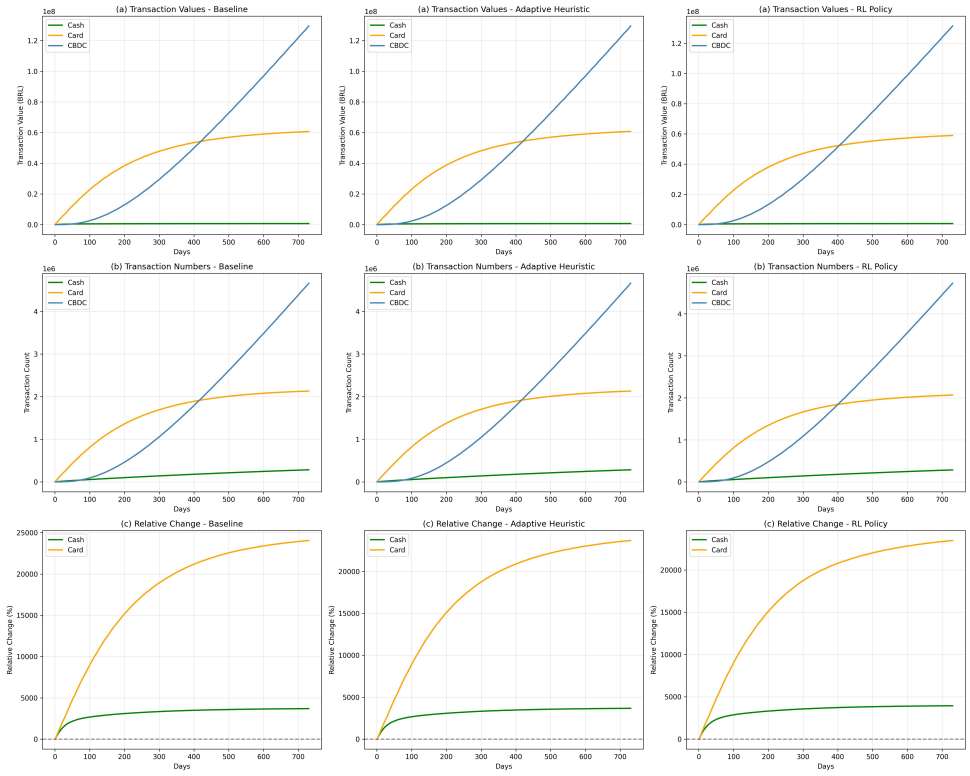

Figure 13: Payment Difussion Panel.

5.3 Financial Stability Implications

Bank leverage dynamics reveal successful stability management across all policy regimes (Figure 14). All scenarios exhibit high volatility during the initial 30–50 days as the system adjusts from initialization, with leverage fluctuating between 9.5x and 13x. However, once CBDC adoption stabilizes, leverage converges to a narrow band around 10.5–11.5x in the baseline and heuristic scenarios, and even tighter 9.5–10.5x in the RL deployment. Importantly, no scenario breaches the regulatory maximum of 14.0x after the initialization period. This suggests that even if CBDC is adopted quickly, it won't cause a widespread financial downturn if wallet limits keep the total amount in circulation in check.

The RL policy manages leverage particularly well, keeping it at the lowest and steadiest level throughout the deployment (average leverage: 9.93 versus 10.91 baseline, 10.90 heuristic). This suggests the learned policy found good settings that support CBDC growth while keeping the banking sector careful. The policy shows a strong preference for Actions 2 and 3 (wallet caps of 14,000–16,000 BRL with leverage limits of 13.5–14.0), accounting for 85.7% of deployment actions, while the most restrictive (Action 0) and most aggressive (Action 4) policies were never selected.

The total amount of money in bank deposits stays incredibly stable across all scenarios, hovering around 136.75–136.85 billion BRL throughout the full 730-day horizon with almost no growth.This stability goes against early worries of a quick drain on banks. It happens due to

two things working together. First, people get their income paid into their bank accounts every 30 days, which refills them faster than CBDC top-ups take money out. The model brings up an important point: CBDC mainly moves existing money around rather than causing a lasting outflow of deposits. Second, CBDC does not earn interest while deposits earn 0.49% annually, taking away any reason to move a lot of money out of deposits just to make a profit. People hold CBDC for easy spending, not as a way to save money, and wisely keep their savings in deposits. The wallet limits of 10,000–18,000 BRL wallet caps (depending on scenario and time) help control the amount of money that can be taken out of deposits.

Payment success rates remain exceptionally high across all scenarios (Figure 15), rising from approximately 95.5% in the initialization period to stabilize around 96.9–97.0% by day 730 (baseline: 96.89%, heuristic: 96.92%, RL: 96.98%). The small differences (less than 0.1 percentage points) between scenarios tell us that consumers not having enough money does not become a system-wide problem. The initial spike toward 98.5% during days 0–50, followed by stabilization near 97%, reflects the transition from an initially unconstrained, low-activity environment to a steady-state regime where realistic frictions (the 2% card failure rate and occasional insufficient balances) come back into play.

The key finding is that financial stability concerns, while theoretically valid, remain manageable under prudent CBDC design. The worst-case scenario of a sudden withdrawal of deposits from banks does not happen when: (i) wallet caps restrict how much CBDC can be accumulated, (ii) CBDC not earning interest removes the reason for speculative withdrawals, and (iii) banks can slowly adjust their finances in response to predictable, measured changes rather than sudden panics.

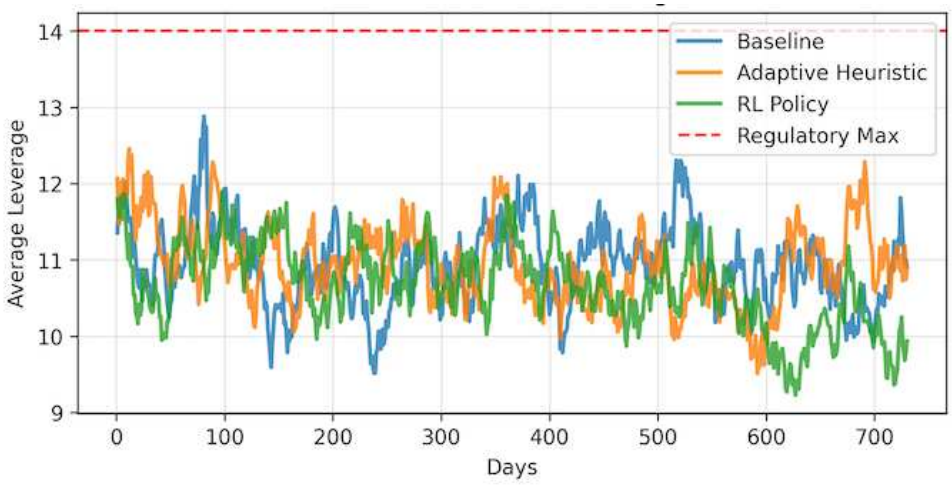

Figure 14: Bank Levarage.

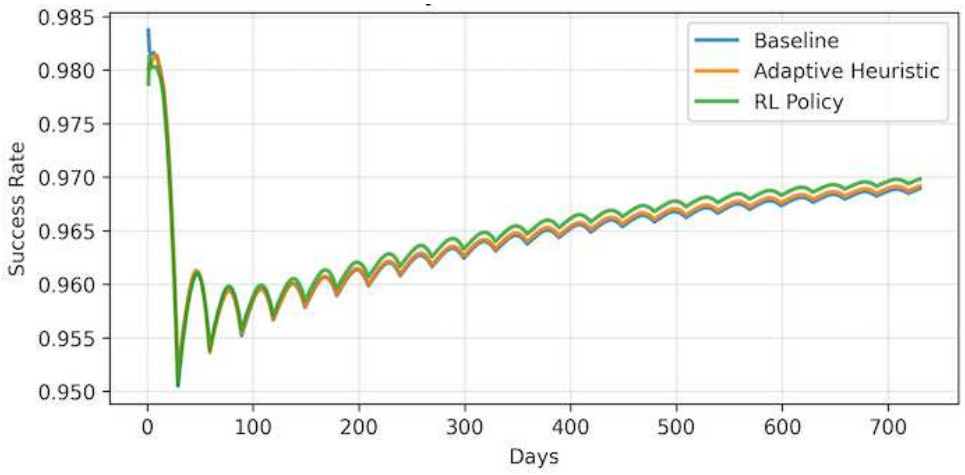

Figure 15: Payment Sucess Rate.

5.4 Policy Performance Comparison

The Q-learning agent explored 2.9% of the 243-state discretized space over 50 training episodes of 250 days each. This means it covered a limited, but not insignificant, number of states given how roughly the states were defined. The final episode reward reached 575.37, with mean Q-values stabilizing around 55.5 and maximum Q-values near 69.8 (Figure 16). However, convergence metrics (Figure 17) remained above the 0.01 threshold throughout training, suggesting the agent did not fully converge despite exhibiting stable behavioral patterns.

The learned policy exhibits clear action preferences (Table 7): Action 3 (wallet cap 16,000 BRL, leverage limit 14.0) dominates with 57.1% selection frequency, followed by Action 2 (cap 14,000, leverage 13.5) at 28.6% and Action 1 (cap 12,000, leverage 13.0) at 14.3%. The most restrictive policy (Action 0: cap 10,000, leverage 12.5) and most aggressive policy (Action 4: cap 18,000, leverage 14.5) were never selected during deployment, indicating the agent learned to avoid extreme positions. This preference structure suggests the learned policy favors moderately high wallet caps to encourage adoption while maintaining conservative leverage constraints to ensure stability.

Despite this clear policy structure, the RL scenario produced final results that were practically the same as the basic and heuristic approaches for all key measures (Table 8). Consumer adoption differed by less than 0.2 percentage points (97.34% RL versus 97.39% baseline), merchant acceptance converged to 100% in all scenarios, and the total amount of CBDC in circulation varied by less than 6% (26,613 BRL RL versus 27,156 baseline). The RL policy did achieve slightly lower average bank leverage (9.93 versus 10.91 baseline), suggesting marginally more conservative financial positioning, but payment success rates remained virtually identical (96.98% RL versus 96.89% baseline).

This equality in performance does not mean the RL approach failed. Rather, it points to two important ideas. First, the main reason CBDC is adopted is how users and businesses in-

teract, which often overrides small changes in policy once enough people are using the system. The similar adoption curves across scenarios (Figure 10) confirm that consumers and businesses follow natural patterns of spreading that policy can adjust but not completely change. Second, the reward system, which focused on how quickly it was adopted and bonuses for reaching certain points, might have been too ambitious given how the states were grouped and the training time. The agent learned steady, moderate policies but did not find vastly better ways to schedule parameters.

Training diagnostics (Figure 18) show that rewards for each episode went up and down between 530-590, with a lot of variation and no clear upward trend after episode 20. This suggests the agent reached a stable point quite early in its training. The way Q-values changed (Figure 16) shows mean values rising from near-zero to stabilize around 55–60 by episode 15, then remaining flat with occasional spikes corresponding to high-reward episodes. The exploration rate decay (Figure 19) proceeded smoothly from 1.0 to 0.35 over 50 episodes (target minimum: 0.15), but the agent's limited state space exploration (2.9%) indicates that most learning occurred in a narrow subset of frequently visited states.

This outcome confirms that the way CBDC systems spread is quite stable even with different policies, as long as those policies are within reasonable limits. However, it also suggests that finding truly superior policies might require defining states more precisely, training for longer periods, or using reward systems that better capture how adoption speeds up over time.

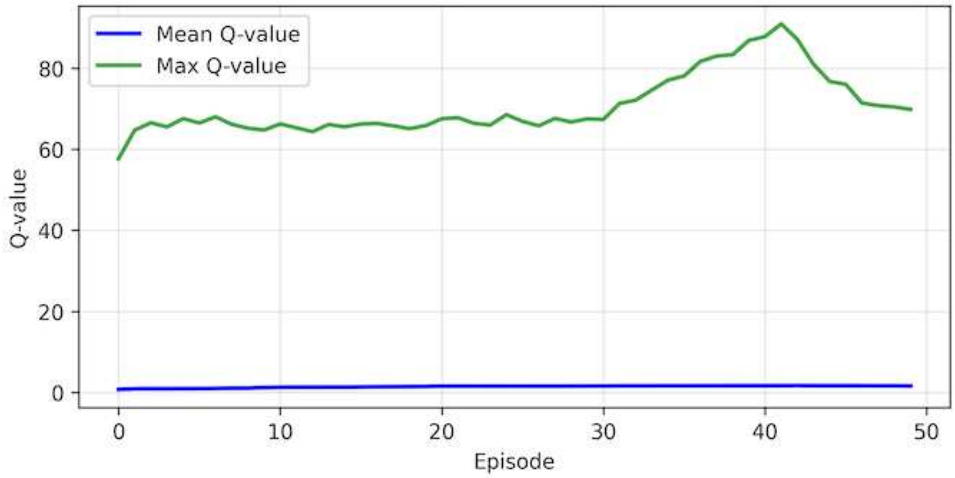

Figure 16: RL Training: Q-Value Evolution.

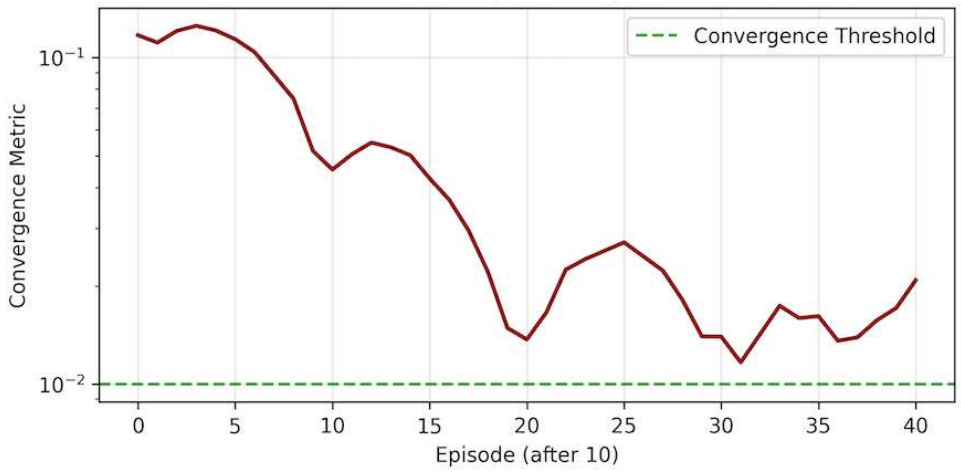

Figure 17: RL Training: Convergence.

Table 7: Learned Action Preferences from RL Training

Action	Wallet Cap (BRL)	Max Leverage	Selection Frequency (%)
0	10,000	12.5	0.0
1	12,000	13.0	14.3
2	14,000	13.5	28.6
3	16,000	14.0	57.1
4	18,000	14.5	0.0

Note: Frequencies based on 730-day deployment with learned policy.

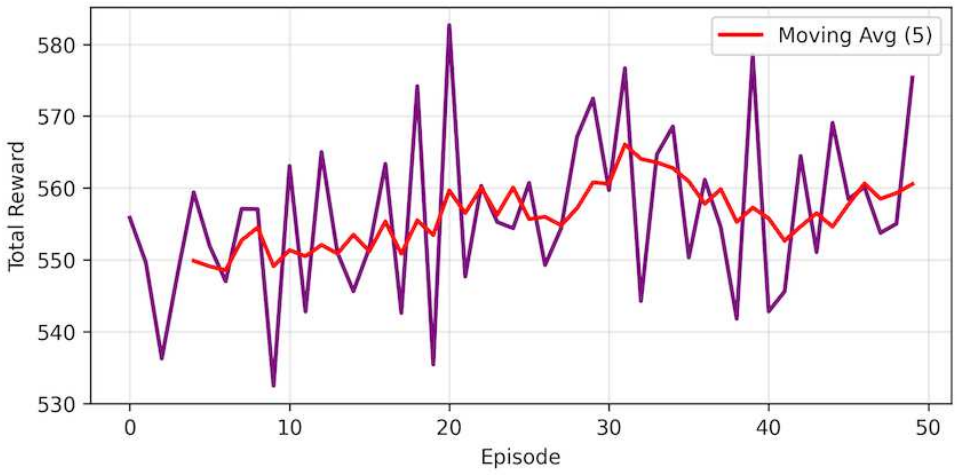

Figure 18: RL Training: Episodes Rewards.

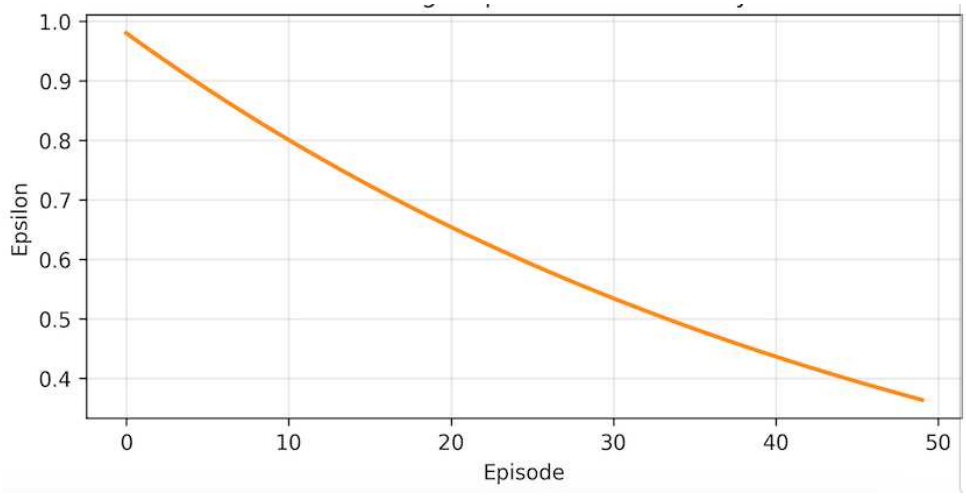

Figure 19: RL Training: Exploration Rate Decay.

Table 8: Final Outcomes Across Policy Scenarios (Day 730)

Metric	**Baseline**	**Heuristic**	**RL Policy**
Consumer Adoption (%)	97.39	97.53	97.34
Merchant Acceptance (%)	100.00	100.00	100.00
Total CBDC (BRL)	27,156	28,134	26,613
Avg Bank Leverage	10.91	10.90	9.93
Payment Success Rate (%)	96.89	96.92	96.98

6 Discussion and Conclusion

The reinforcement learning-enhanced agent-based model presented here demonstrates both the potential and current limitations of adaptive algorithms in central bank digital currency (CBDC) policy design. The framework instantiates 10,000 consumer agents, 1,000 merchant agents, and three commercial banks calibrated to Brazilian macroeconomic parameters, simulating CBDC adoption dynamics under three policy regimes: fixed baseline parameters, rule-based heuristic adjustments, and Q-learning with experience replay. The Q-learning agent works with a simplified state space of 243 states (which are 3 to the power of 5 bins). These states capture how many consumers are adopting CBDC, how many merchants accept it, the ratio of CBDC to bank deposits, how often payments are successful, and how much banks are leveraged. From these, the agent picks one of five policy choices that change wallet limits (10,000 to 18,000 BRL) and maximum leverage limits (12.5 to 14.5). The reward function focuses on how fast adoption happens by giving big weights to changes in adoption rates ($100 \times (\Delta\bar{w} + \Delta\bar{A})$), offers small bonuses when adoption hits certain levels (20%, 50%, 80%), and applies strong quadratic penalties only if rules are broken (leverage > 14.0, CBDC ratio > 0.4).

The baseline scenario achieved 97.39% consumer adoption by day 730 under fixed parameters (10,000 BRL cap, 14.0 leverage limit), while the heuristic policy produced nearly identical results at 97.53%. The reinforcement learning deployment, running for the same 730-day horizon following 50 training episodes of 250 days each, reached 97.34% adoption, which is statistically indistinguishable from the baseline. This closeness in results shows that network effects and how two-sided markets function are more powerful than small tweaks to policy numbers. Once enough consumers and merchants adopt (about 30% consumer adoption, 50% merchant acceptance), this self-driving cycle leads to almost everyone adopting, no matter the policy settings, as long as they are within sensible limits.

The RL agent's learned strategy exhibits clear preferences: Action 3 (16,000 BRL cap, 14.0 leverage) dominates with 57.1% selection frequency, followed by Action 2 (14,000 BRL cap, 13.5 leverage) at 28.6% and Action 1 (12,000 BRL cap, 13.0 leverage) at 14.3%. The most restrictive policy (Action 0: 10,000 BRL cap, 12.5 leverage) and most aggressive policy (Action 4: 18,000 BRL cap, 14.5 leverage) were never selected during deployment, indicating the agent learned to avoid extreme positions. This preference, leaning towards moderately high wallet limits with careful or middle-ground leverage rules, shows real policy learning: the agent found that middle-tier settings balance encouraging adoption (through attractive wallet limits) with keeping the financial system stable (through leverage rules).

The learned policy achieved marginally lower and more stable average bank leverage (9.93 versus 10.91 baseline), suggesting the agent successfully identified parameter regions that maintain conservative banking sector positioning while permitting CBDC diffusion. However, the policy's adoption outcomes remain indistinguishable from baseline and heuristic approaches (consumer adoption within 0.2 percentage points, merchant acceptance at 100% across all sce-

narios, total CBDC circulation differing by less than 6%). This similar performance points to a key idea: the main driver of CBDC adoption is how networks naturally grow (consumers adopt when merchants accept, merchants accept when consumers adopt). This natural growth overshadows the slight effects of policy tools once a certain level of adoption is reached. The similar shapes of the adoption curves across different situations confirm that the spread follows typical S-shaped paths, driven by how behaviors catch on, rather than by small adjustments to policy numbers.

The agent's convergence behavior reveals both progress and limitations. State space coverage reached 2.9% during training (7 of 243 states explored), indicating the agent concentrated learning on frequently encountered system configurations rather than exhaustively exploring rare states. The epsilon-greedy exploration strategy, decaying from 1.0 to approximately 0.35 over 50 episodes (decay rate 0.98), provided moderate exploration but proved insufficient for comprehensive state coverage.

Q-value statistics reveal structured learning rather than random initialization: mean Q-value of 55.5 and maximum of 69.8 indicate the agent assigned positive expected returns to state-action pairs. Episode rewards stabilized around 560–580 after episode 20 but with ongoing variations. This means the agent reached a stable level of behavior while still having some unpredictability in results. The reward function's emphasis on adoption velocity (100× coefficient) and milestone bonuses created strong incentives for rapid diffusion, while harsh quadratic penalties on leverage violations (5.0×) and excessive CBDC ratios (3.0×) enforced conservative financial positioning.

Several design choices account for the similar outcomes. The broad categorization of states (3 bins per dimension) favors applying what's learned generally rather than focusing on small differences, helping the agent learn good policies from limited training. However, it might hide subtle links between states and actions. The five discrete actions offer a moderate range of policy options, with neighboring actions changing wallet limits by 2,000–4,000 BRL. These increases are enough to capture how wallet limits affect adoption but might be too broad to find the perfect middle ground. The 250-day episode length gives the agent about 8 months of adoption changes per training round. This is enough to see the initial quick adoption (days 0–150) but ends before full saturation (days 400–730), which may bias learning toward early-stage policies.

The reward system's uneven structure, with big rewards for speed and harsh penalties for rule-breaking, was a conscious choice. It prioritized fast adoption, but only within stability limits, rather than treating all goals equally. This approach successfully kept rules from being broken (no situation went over the 14.0 leverage cap after starting). However, it might have undervalued finding much quicker adoption paths. The milestone bonuses (5.0 at 20%, 10.0 at 50%, 15.0 at 80%) gave strong clues for good policy timing, but the agent's learned policy hit these milestones at roughly the same times as the basic one. This suggests these bonuses weren't enough to overcome the natural spread driven by networks.

The comparison of outcomes shows a sufficiency paradox in CBDC policy making: carefully set fixed numbers do almost as well, (97.39% adoption, 96.89% payment success, stable lever-

age) without the need for complex computer calculations, technical setup, or defining rewards. This suggests that when markets are stable and how adoption works is well understood, simple policies might be better than adaptive ones, especially when considering things like regulatory clarity, being able to check past actions, and reliable promises. The simple rule-based policy, which used thresholds (raise wallet limit when adoption is over 50%, tighten leverage if banks go over 13.0), gave almost identical results (97.53% adoption). This further supports the idea that basic adjustments capture most of what's important for policy.

Still, reinforcement learning holds promise for situations where the current framework falls short. First, unexpected events (financial crises, a sudden lack of trust in banks, competing private digital currencies) could cause big shifts where rule-based systems aren't flexible enough. The RL agent's preference for middle-ground policies suggests it could adjust more easily to changing conditions than fixed rules. Second, policy problems with many goals that have unclear trade-offs, like both increasing financial access for different income groups, keeping banks profitable, and making sure payment systems are strong, might gain from algorithmic solutions that systematically look at how different settings interact. Third, figuring out what if scenarios (e.g., what if we had tightened leverage limits on day 200?) needs a model-based way of thinking, which RL frameworks naturally provide through their learned understanding of how things change.

Future method improvements could make RL work better and apply to more situations. Deep Q-learning with neural networks for approximating functions would allow for continuous state representation and better generalization to new market conditions. This comes at the cost of harder-to-understand results, which is important for regulation. Policy gradient methods might handle the scattered, delayed rewards typical of CBDC adoption better, where actions in month 1 show effects in months 6-12. Extending training to over 100 episodes with slower epsilon decay (0.99 per episode) and exploration bonuses based on coverage would improve how states are sampled and how well value estimates are refined.

Hybrid system designs that combine fixed rules (leverage never over 14.0, wallet caps within reasonable transaction limits) with learned, small adjustments within safe operating ranges deserve special attention for regulatory acceptance. Making state information richer by including distribution statistics (like Gini coefficients for CBDC holdings among different wealth levels, varying merchant sizes, and differing adoption rates by region) would allow for policies sensitive to financial inclusion and fairness goals beyond just average adoption rates.

The existing combined state variables (consumer adoption rate, merchant acceptance rate) overlook how things are distributed, which might matter for making CBDC accessible to everyone. Expanding the set of actions to include tiered wallet caps (higher limits for verified users, lower for anonymous accounts), changing interest rate policies (small positive payments to encourage holding CBDC), or programs to support merchants would broaden policy options beyond the current focus on basic, non-interest-bearing retail transactions.

Including more variety in the banking sector, looking at specific liquidity risk, interbank

lending markets, and banks of different sizes and business models, would help analyze how stable the system is when CBDC causes funding stress. The current three-bank model with similar behavior skips over risks of things spreading, money moving to safe havens, and how different levels of disintermediation affect small versus large institutions. These are things that could really impact CBDC in the real world. Modeling diverse consumer traits beyond just income (like digital literacy, desire for privacy, trust in institutions) would capture adoption barriers beyond just network effects that shape how fast things spread.

The most logical path appears to involve people and AI working together. In this setup, RL systems would help central bank policymakers make decisions and analyze scenarios. Policymakers would still have the final say but would use algorithmic insights to handle complex trade-offs with many goals, which are common in governing digital currency. Instead of AI making policy decisions on its own, RL agents could explore what-if situations, pinpoint strong policy choices across various uncertainties, and highlight unexpected patterns that appear during simulated stress tests.

These findings add to the growing field of computational policy experiments in digital money. They show that adaptive policy optimization faces fundamental challenges when network effects are the main drivers of a system: sophisticated algorithms find it hard to overcome the natural momentum of adoption in two-sided markets once a certain level of engagement is reached. The agent's learned preference for middle-ground policies, avoiding both strict limits and excessive freedom, offers real insights into cautious parameter zones that balance encouraging adoption with keeping the financial system stable.

The framework sets a basis for analyzing CBDC policy. It combines agent-based modeling, which can show varied behavior and new patterns, with reinforcement learning, which can adapt and optimize. While current results suggest simpler methods are good enough under the conditions we modeled, the setup allows for future improvements that address the enhancements suggested here. As central banks deal with increasingly complex trade-offs between new ideas, making things accessible, and keeping the system steady in a changing digital payment world, computer frameworks that blend realistic behavior, financial stability analysis, and algorithmic policy search will be key tools for making wise CBDC design choices.

References

Atlantic Council. Central Bank Digital Currency Tracker. https://www.atlanticcouncil.org/cbdctracker/, 2025. Accessed: 2025-10-12.

R. Auer and R. Böhme. The technology of retail central bank digital currency. *BIS Quarterly Review*, page 16, March 2020. URL https://www.bis.org/publ/qtrpdf/r_qt2003e.htm.

Banco Central do Brasil. Diretrizes do drex – 2023, 2023. URL https://www.bcb.gov.br/estabilidadefinanceira/drex?modalAberto=diretrizes_realdigital.

Banco Central do Brasil. Relatório do piloto drex – fase 1. Relatório técnico / pilot report, Banco Central do Brasil, 2024. URL https://www.bcb.gov.br/content/estabilidadefinanceira/real_digital_docs/piloto/Relatorio_Drex_piloto_fase_1.pdf.

Banco Central do Brasil. Pix – sistema de pagamentos instantâneos, 2025. URL https://www.bcb.gov.br/estabilidadefinanceira/pix. Accessed: 2025-12-26.

Bank for International Settlements. A proposal for a retail central bank digital currency (cbdc) architecture. Technical Report OTH_P89, Bank for International Settlements, 2024. URL https://www.bis.org/publ/othp89.pdf. Accessed: 2025-11-08.

U. Bindseil, C.-E. Coste, and G. Pantelopoulos. Digital money and finance: a critical review of terminology. *ECB Working Paper*, 2025. URL https://www.ecb.europa.eu/pub/pdf/scpwps/ecb.wp3022~67ea842a2a.en.pdf?bb553e483019c8f398888cc1afc7de2e.

M. Di Maggio, P. Ghosh, S. K. Ghosh, and A. Wu. Impact of Retail CBDC on Digital Payments and Bank Deposits: Evidence from India. *NBER Working Paper*, 2024. URL https://www.nber.org/system/files/working_papers/w32457/w32457.pdf.

D. Georgarakos, G. Kenny, L. Laeven, and J. Meyer. Consumer Attitudes Towards a Central Bank Digital Currency. *ECB Working Paper*, 2025. URL https://www.ecb.europa.eu/pub/pdf/scpwps/ecb~cde4bd616e.wp3035en.pdf?33bf2be9c5ed0fcd1e01afcb1a91eba4.

A. Illes, A. Kosse, and P. Wierts. Advancing in Tandem – Results of the 2024 BIS Survey on Central Bank Digital Currencies and Crypto. *Bank for International Settlements*, 2025. URL https://www.bis.org/publ/bppdf/bispap159.pdf.

C. Lambert, C. Larkou, C. Pancaro, A. Pellicani, and M. Sintonen. Digital Euro Demand: Design, Individuals' Payment Preferences and Socioeconomic Factors. *ECB Working Paper*, 2024. URL https://www.ecb.europa.eu/pub/pdf/scpwps/ecb.wp2980~5f64961c8f.en.pdf?6d8ff4f25cb867a259baad9283dd761e.

B. Mishra and E. S. Prasad. A Simple Model of a Central Bank Digital Currency. *NBER Working Paper*, 2023. URL https://www.nber.org/system/files/working_papers/w31198/w31198.pdf.

C. Pastor Sempere. *Governance and Control of Data and Digital Economy in the European Single Market*. Academic Press, 2025. URL https://link.springer.com/book/10.1007/978-3-031-74889-9.

P. Pligher. Brazil shelves digital currency—for now, but private sector drives tokenization forward. Funds Society, 2025. URL https://www.fundssociety.com/en/news/alternatives/brazil-shelves-digital-currency-for-now/. Accessed: 2025-11-08.

A. Ramadiah, M. Galbiati, and K. Soramäki. Agent-Based Simulation of Central Bank Digital Currencies. *SSRN Electronic Journal*, 2021. doi: 10.2139/ssrn.3959759. URL https://doi.org/10.2139/ssrn.3959759.

I. Struchkov, A. Lukashin, B. Kuznetsov, I. Mikhalev, and Z. Mandrusova. Agent-based modeling of blockchain decentralized financial protocols. In *Proceedings of the 29th Conference of the FRUCT Association*, 2024. URL https://www.fruct.org/files/publications/volume-29/fruct29/Struc.pdf. Accessed: 2025-11-08.

A. Turrell. Agent-based models: understanding the economy from the bottom up. *Quarterly Bulletin*, (Q4):173-188, 2016. URL https://www.bankofengland.co.uk/-/media/boe/files/quarterly-bulletin/2016/agent-based-models-understanding-the-economy-from-the-bottom-up.pdf. Accessed: 2025-11-08.

S. Waliczek and C. Nili. Cbdcs come in two forms: retail and wholesale. what's the difference? World Economic Forum, 2024. URL https://www.weforum.org/stories/2024/02/wholesale-retail-cbdcs-difference/. Accessed: 2025-11-08.

C. W. Weimer, J. O. Miller, and R. R. Hill. Agent-based modeling: An introduction and primer. In *2016 Winter Simulation Conference (WSC)*, pages 65–79, Washington, DC, USA, 2016. doi: 10.1109/WSC.2016.7822080.

U. Wilensky and W. Rand. *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. The MIT Press, Cambridge, MA, 2015. URL https://www.jstor.org/stable/j.ctt17kk851.

A. Zarifis and X. Cheng. A Model of Trust in Central Bank Digital Currency (CBDC) in Brazil: How Trust in a Two-Tier CBDC with Both the Central and Retail Banks Involved Changes

Consumer Trust. In *Fintech and the Emerging Ecosystems*. Springer, 2025. URL https://link.springer.com/chapter/10.1007/978-3-031-83402-8_4.

A Model Implementation: Core Code Components

This appendix presents the essential components of the agent-based model implementation in Python, focusing on the key classes and algorithms that define agent behavior, the reinforcement learning framework, and the simulation environment.

A.1 Agent Class Definitions

A.1.1 Consumer Agent

```
@dataclass
class Consumer:
    """Consumer agent with cash, deposits, and CBDC holdings"""

    id: int
    bank_id: int
    cash: float = 0.0
    deposits: float = 0.0
    cbdc: float = 0.0
    monthly_income: float = 0.0
    daily_spending_mean: float = 0.0
    has_wallet: bool = False
    wallet_cap: float = 10000.0

    def attempt_payment(self, amount: float, merchant: 'Merchant',
                        is_offline: bool = False) -> Tuple[bool, str]:
        """Execute payment hierarchy: CBDC -> Card -> Cash"""
        self.payment_attempts += 1

        # Offline cash payment
        if is_offline and self.cash >= amount:
            self.cash -= amount
            self.cash_payments += 1
            return True, "cash"

        # CBDC payment (0% failure rate)
        if merchant.accepts_cbdc and self.has_wallet:
            if self.cbdc >= amount:
```

```
            self.cbdc -= amount
            self.cbdc_payments += 1
            return True, "cbdc"
        elif self.top_up_cbdc(amount):
            self.cbdc -= amount
            self.cbdc_payments += 1
            return True, "cbdc"

    # Card payment (2% failure rate)
    if self.deposits >= amount:
        if np.random.random() < 0.02:
            self.payment_failures += 1
            return False, "failed"
        self.deposits -= amount
        self.card_payments += 1
        return True, "card"

    self.payment_failures += 1
    return False, "failed"

def top_up_cbdc(self, needed_amount: float, phi: float = 0.25) -> bool:
    """Top up CBDC wallet from deposits"""
    if not self.has_wallet or self.deposits <= 0:
        return False

    topup = min(phi * self.deposits, needed_amount,
                self.wallet_cap - self.cbdc)

    if topup > 0:
        self.deposits -= topup
        self.cbdc += topup
        return self.cbdc >= needed_amount
    return False

def update_wallet_adoption(self, consumer_adoption_rate: float,
                           merchant_acceptance_rate: float,
                           alpha_0: float = -4.0,
                           alpha_1: float = 3.0,
                           alpha_2: float = 2.0) -> bool:
    """Logistic adoption with network effects and cap attractiveness"""
    if self.has_wallet:
```

```
            return False

        cap_attractiveness = (self.wallet_cap - 8000) / 10000
        cap_bonus = cap_attractiveness * 3

        z = (alpha_0 + alpha_1 * consumer_adoption_rate +
             alpha_2 * merchant_acceptance_rate + cap_bonus)
        prob = 1 / (1 + np.exp(-z))
        prob = min(prob, 0.005)  # Daily cap

        if np.random.random() < prob:
            self.has_wallet = True
            return True
        return False
```

A.1.2 Merchant Agent

```
@dataclass
class Merchant:
    """Merchant agent with CBDC acceptance decision"""

    id: int
    accepts_cash: bool = True
    accepts_cards: bool = False
    accepts_cbdc: bool = False
    customers_with_cbdc: int = 0
    total_customers: int = 0

    @property
    def cbdc_customer_share(self) -> float:
        """Calculate share of customers with CBDC"""
        if self.total_customers == 0:
            return 0.0
        return self.customers_with_cbdc / self.total_customers

    def update_cbdc_acceptance(self, avg_wallet_cap: float,
                               k: float = 5.0,
                               z0: float = 0.3) -> bool:
        """Logistic acceptance function with wallet cap adjustment"""
        if self.accepts_cbdc:
            return False
```

```
        z = self.cbdc_customer_share
        cap_multiplier = (avg_wallet_cap - 8000) / 10000

        # Lower threshold for higher caps
        adjusted_threshold = z0 * (1 - cap_multiplier * 0.4)

        # Logistic probability
        prob = 1 / (1 + np.exp(-k * (z - adjusted_threshold)))

        # Adjusted daily cap
        adjusted_daily_cap = 0.008 * (1 + cap_multiplier * 0.5)
        prob = min(prob, adjusted_daily_cap)

        if np.random.random() < prob:
            self.accepts_cbdc = True
            return True
        return False
```

A.1.3 Commercial Bank Agent

```
@dataclass
class CommercialBank:
    """Commercial bank with leverage dynamics"""

    id: int
    deposits: float = 0.0
    loans: float = 0.0
    equity: float = 0.0
    loan_return_mean: float = 0.2791
    loan_return_std: float = 0.00162
    desired_leverage: float = 13.5

    @property
    def current_leverage(self) -> float:
        """Calculate current leverage ratio"""
        if self.equity <= 0:
            return float('inf')
        return self.loans / self.equity

    def calculate_profit(self, t: int) -> float:
        """Calculate daily profit with leverage bonus"""
        # Leverage bonus: 0.2% per point above 10
```

```
        leverage_bonus = max(0, (self.current_leverage - 10)) * 0.002
        adjusted_return = self.loan_return_mean * (1 + leverage_bonus)

        # Stochastic daily return
        r_X_daily = np.random.normal(adjusted_return / 252,
                                     self.loan_return_std)

        loan_revenue = self.loans * r_X_daily
        deposit_cost = self.deposits * (0.0049 / 252)
        profit = loan_revenue - deposit_cost

        self.daily_profit = profit
        self.cumulative_profit += profit
        return profit

    def adjust_leverage(self, max_leverage: float,
                        adjustment_speed: float = 0.05):
        """Adjust loans toward target leverage"""
        if self.equity <= 0:
            return

        # Target near max, with safety buffer
        target = min(self.desired_leverage, max_leverage - 0.2)

        current = self.current_leverage
        gap = target - current
        adjustment = gap * adjustment_speed * self.equity
        self.loans = max(0, self.loans + adjustment)

    def update(self, t: int, max_leverage: float):
        """Daily update: profit calculation and leverage adjustment"""
        self.calculate_profit(t)
        self.adjust_leverage(max_leverage)
        self.equity += self.daily_profit
```

A.2 Q-Learning Agent with Experience Replay

```
class QLearningAgent:
    """Q-learning agent with experience replay for CBDC policy"""

    def __init__(self, state_bins: List[int], action_space: List[Tuple],
```

```
                learning_rate: float = 0.1, gamma: float = 0.95,
                epsilon_start: float = 1.0, epsilon_end: float = 0.15,
                epsilon_decay: float = 0.98):

        self.state_bins = state_bins
        self.action_space = action_space
        self.n_actions = len(action_space)

        # Q-table: state_bins + [n_actions]
        self.Q = np.zeros(state_bins + [self.n_actions])

        self.lr = learning_rate
        self.gamma = gamma
        self.epsilon = epsilon_start
        self.epsilon_end = epsilon_end
        self.epsilon_decay = epsilon_decay

        # Experience replay buffer
        self.replay_buffer = deque(maxlen=10000)
        self.batch_size = 64

        # State bounds for discretization
        self.state_bounds = [
            (0.0, 1.0),   # consumer_adoption
            (0.0, 1.0),   # merchant_acceptance
            (0.0, 0.5),   # cbdc_to_deposits_ratio
            (0.75, 1.0),  # payment_success_rate
            (8.0, 16.0),  # avg_leverage
        ]

    def discretize_state(self, state: np.ndarray) -> Tuple:
        """Discretize continuous state to bin indices"""
        discrete = []
        for i, (val, (low, high)) in enumerate(zip(state,
                                                   self.state_bounds)):
            val_clipped = np.clip(val, low, high)
            bin_idx = int((val_clipped - low) / (high - low) *
                          (self.state_bins[i] - 1))
            bin_idx = min(bin_idx, self.state_bins[i] - 1)
            discrete.append(bin_idx)
        return tuple(discrete)
```

```
def select_action(self, state: np.ndarray,
                  training: bool = True) -> int:
    """Epsilon-greedy action selection"""
    if training and np.random.random() < self.epsilon:
        return np.random.randint(self.n_actions)
    else:
        discrete_state = self.discretize_state(state)
        return np.argmax(self.Q[discrete_state])

def update_q_table(self):
    """Update Q-table using experience replay batch"""
    if len(self.replay_buffer) < self.batch_size:
        return 0.0

    # Sample random batch
    batch = np.random.choice(len(self.replay_buffer),
                             self.batch_size, replace=False)

    total_loss = 0.0
    for idx in batch:
        state, action, reward, next_state, done = \
            self.replay_buffer[idx]

        s = self.discretize_state(state)
        s_next = self.discretize_state(next_state)

        current_q = self.Q[s][action]

        if done:
            target_q = reward
        else:
            target_q = reward + self.gamma * np.max(self.Q[s_next])

        # TD update
        td_error = target_q - current_q
        self.Q[s][action] += self.lr * td_error

        total_loss += td_error ** 2

    return total_loss / self.batch_size
```

```
    def decay_epsilon(self):
        """Decay exploration rate"""
        self.epsilon = max(self.epsilon_end,
                           self.epsilon * self.epsilon_decay)
```

A.3 Reward Function

```
def _compute_reward(self) -> float:
    """Compute composite reward emphasizing adoption velocity"""
    state = self._get_state_vector()
    consumer_adoption = state[0]
    merchant_acceptance = state[1]
    cbdc_ratio = state[2]
    payment_success = state[3]
    avg_leverage = state[4]

    # Velocity reward: rate of change
    if len(self.history['consumer_adoption']) > 1:
        adoption_velocity = (consumer_adoption -
                             self.history['consumer_adoption'][-1])
        merchant_velocity = (merchant_acceptance -
                             self.history['merchant_acceptance'][-1])
        velocity_reward = (adoption_velocity + merchant_velocity) * 100.0
    else:
        velocity_reward = 0.0

    # Milestone bonuses
    milestone_bonus = 0.0
    if 0.19 < consumer_adoption < 0.21:  # 20%
        milestone_bonus += 5.0
    if 0.49 < consumer_adoption < 0.51:  # 50%
        milestone_bonus += 10.0
    if 0.79 < consumer_adoption < 0.81:  # 80%
        milestone_bonus += 15.0

    # Network effects bonus
    if consumer_adoption > 0.5 and merchant_acceptance > 0.5:
        network_bonus = (consumer_adoption * merchant_acceptance)**2 * 5.0
    else:
        network_bonus = 0.0
```

```
    # Progress and usage rewards
    progress_reward = (consumer_adoption + merchant_acceptance) * 0.5
    usage_reward = cbdc_ratio * 2.0

    # Harsh penalties for violations
    leverage_penalty = 0.0
    if avg_leverage > 14.0:
        leverage_penalty = (avg_leverage - 14.0) ** 2 * 5.0

    cbdc_penalty = 0.0
    if cbdc_ratio > 0.4:
        cbdc_penalty = (cbdc_ratio - 0.4) ** 2 * 3.0

    payment_penalty = 0.0
    if payment_success < 0.95:
        payment_penalty = (0.95 - payment_success) * 10.0

    # Total reward
    reward = (velocity_reward + milestone_bonus + network_bonus +
              progress_reward + usage_reward - leverage_penalty -
              cbdc_penalty - payment_penalty)

    return reward
```

A.4 Training and Deployment Loop

```
def run_rl_scenario(config: ModelConfig, output_dir: Path):
    """Reinforcement Learning: Train → Deploy"""

    # Define action space
    action_space = [
        (10000, 12.5),  # (wallet_cap, max_leverage)
        (12000, 13.0),
        (14000, 13.5),
        (16000, 14.0),
        (18000, 14.5),
    ]

    # Initialize agent
    agent = QLearningAgent(
```

```
    state_bins=[3, 3, 3, 3, 3],
    action_space=action_space,
    learning_rate=0.1,
    gamma=0.95,
    epsilon_start=1.0,
    epsilon_end=0.15,
    epsilon_decay=0.98
)

env = CBDCEnvironment(config, output_dir, "rl_training")

# PHASE 1: Offline training
for episode in range(config.training_episodes):
    state = env.reset()
    episode_reward = 0

    for step in range(config.episode_length):
        # Select action
        action_idx = agent.select_action(state, training=True)
        action = agent.get_action_params(action_idx)

        # Execute action
        next_state, reward, done, info = env.step(action)

        # Store transition
        agent.store_transition(state, action_idx, reward,
                               next_state, done)

        # Update Q-table
        loss = agent.update_q_table()

        episode_reward += reward
        state = next_state

    agent.decay_epsilon()
    agent.track_convergence()

# PHASE 2: Deployment (exploitation only)
env_deploy = CBDCEnvironment(config, output_dir, "rl_deployment")
state = env_deploy.reset()
```

```
    for day in range(config.deployment_days):
        action_idx = agent.select_action(state, training=False)
        action = agent.get_action_params(action_idx)
        next_state, reward, done, info = env_deploy.step(action=action)
        state = next_state

    env_deploy.save_results()
    return pd.DataFrame(env_deploy.history), agent
```

A.5 Model Configuration and Calibration

```
class ModelConfig:
    """Load and manage model parameters from JSON calibration file"""

    def __init__(self, config_path: str):
        with open(config_path, 'r') as f:
            self.params = json.load(f)

        # Agent counts
        self.n_consumers = self.params['system']['agents']['n_consumers']
        self.n_merchants = self.params['system']['agents']['n_merchants']
        self.n_banks = self.params['system']['agents']['n_banks']

        # Time periods
        self.baseline_days = 730  # 2 years
        self.training_episodes = 50
        self.episode_length = 250  # ~8 months per episode
        self.deployment_days = 730  # 2 years deployment

# Initialize consumers using calibration data
for i in range(config.n_consumers):
    monthly_income = np.random.lognormal(
        mean=np.log(consumer_params['income']['mean_monthly']),
        sigma=0.5
    )

    initial_wealth = monthly_income * 12 * \
        consumer_params['initial_wealth']['factor_of_annual_income']

    consumer = Consumer(
        id=i,
```

```
        bank_id=i % config.n_banks,
        cash=initial_wealth * 0.05,      # 5% cash
        deposits=initial_wealth * 0.95,  # 95% deposits
        monthly_income=monthly_income,
        daily_spending_mean=consumer_params['expenditure']['mean_daily'],
        wallet_cap=consumer_params['topup_behavior']['cbdc_wallet_cap']
    )
```