# CRISIS

## An Economics Simulation Platform and the Crisis Standard Model

J. Kieran Phillips and Balzs Adamcsek, Christoph Aymanns, Tibor Barany, Balázs Bálint, Olaf Bochmann, Fabio Caccioli, Ermanno Catullo, Bob De Caux, Fulvio Corsi, Viktor Erdlyi, J. Doyne Farmer, Gábor Ferschl, Luke Friendshuh, Domenico Delli Gatti, Anatolij Gelimson, Jakob Grazzini, Alessandro Gobbi, László Gulys, Gerald Gurtner, Ariel Y. Hoffman, Giulia Iori, Márton, Iványi, Peter Klimek, Zsolt Kúti, Richard O. Legendi, Milan Lovric, Marotta Luca, Tamás Máhr, Róbert Mészáros, Ervin Mikus, Sebastian Poledna, James Porter, Bocsi Rajmund, Ross Richardson, Victor Spirin, Attila Szabó, Gábor Szemes, Daniel Tang, Stefan Thurner, Aurélien Vermeir.

June 11, 2015

draft.

EUROPEAN COMMISSION

SEVENTH FRAMEWORK PROGRAMME

**Abstract**

A summary and updated description of the CRISIS simulation platform. This document furthers the European Commission CRISIS FP7 deliverables 8.2 and 8.3. We give a high level description of the initiative and integrated simulator.

The integrated simulator is a product of contributions by many researchers across the world, and collaborations between universities and financial institutions including the Bank of England.

**Partners**

Università Cattolica del Sacro Cuore (UCSC)

The Chancellor, Masters and Scholars of the University of Oxford

Universiteit van Amsterdam (UvA)

Centre de Recerca en Economia Internacional (CREI)

Medizinische Universitaet Wien (MUV)

The City University (CITY)

AITIA International, Inc. (AITIA)

Università degli Studi di Palermo (UNIPA)

Commisariat à l'Energie Atomique et aux Energies Alternatives (CEA)

Università Politecnica delle Marche (UPM)

Scuola Normale Superiore di Pisa (SNS)

**FP7 Commission Packages**

WP1 Database construction

WP2 Agent-based model of the financial system (FABM)

WP3 Agent-based model of the macroeconomy (MABM)

WP4 Economic experiments

WP5 Online economic game

WP6 A user-friendly economic simulator

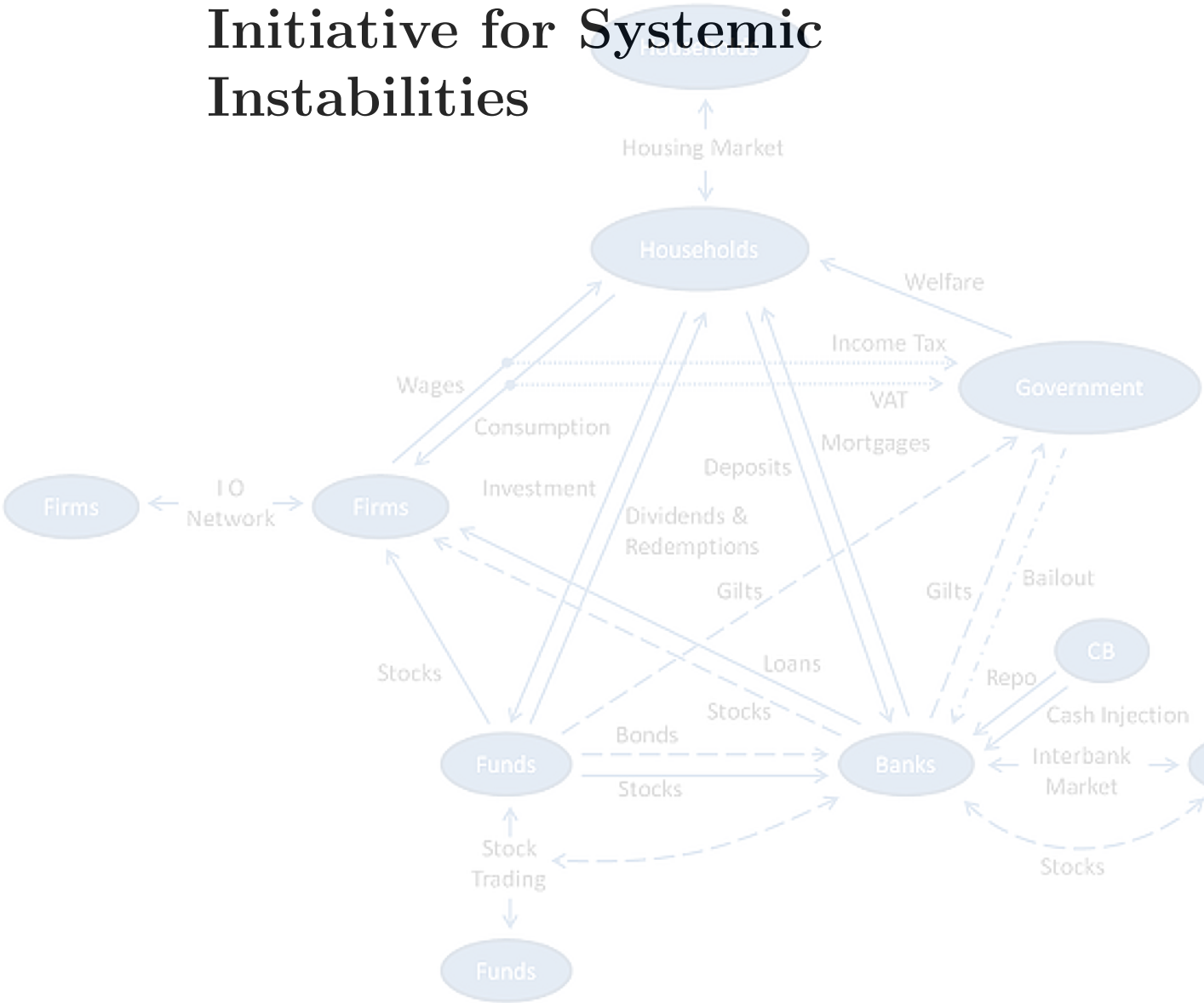
WP7 Stylized complex systems models

WP8 Integration and coordination of Agent Based models

# Contents

# Chapter 1

# The Complexity Research Initiative for Systemic Instabilities



# 1 Introduction

CRISIS is the Complexity Research Initiative for Systemic Instabilities, a special consortium of university scholars, private firms and policymakers that aims to build a new model of the economy and financial system based on how people and institutions truly behave. The consortium was established in the wake of the global financial crisis, which demonstrated that existing models that had been adequate for 'good times' were entirely inadequate for the purpose of predicting and understanding major crises.

CRISIS is a Seventh Framework partnership of researchers from 11 leading European academic and private sector institutions, supported by an advisory board of senior current and former policymakers and financiers. The three-year project (2011–2014) was funded by the European Commission and continues both under the auspices of the Institute for New Economic Thinking, Oxford, and via contributions and guidance from its founders, members and their peers.

The goal of CRISIS and its satellite projects has been to build new models of modern economies, which, viewed as a complex systems, are unpredictable and poorly understood. The components of CRISIS are transformed into integrated software for use by central banks and governments. For that reason, CRISIS works in collaboration with major state central banks, government economic and financial ministries, and with several multilateral institutions.

CRISIS aims to develop tools to deepen policymakers' understanding of the economic and financial system and give them realistic options for modelling the economy and designing policies and regulations. The FP7 initiative delivers three products:

- A model of the European Union financial system and macroeconomy, with a user friendly graphical simulator interface and, in addition, a web–based gaming mode.
- A granular database of households, firms, and financial institutions.
- Analyses of critical European Union financial and economic issues based on the model.

The Project Coordinator for CRISIS was Domenico Delli Gatti, Milan, Italy. The Scientific Coordinator was Doyne Farmer, University of Oxford, UK. The Chief Engineer was J. Kieran Phillips, University of Oxford, UK.

The members of the CRISIS Advisory Board were:

- Andrea Enria, Chairman, the European Banking Authority
- Diana Farrell, Director, the McKinsey Center for Government; former Deputy Director of the National Economic Council, the US White House
- Andrew Haldane, Executive Director, Financial Stability, the Bank of England

- Lex Hoogduin, Professor of Economics, the University of Gronigen, former Board Member, Bank of the Netherlands
- Paul Jenkins, Distinguished Fellow, the Centre for International Governance Innovation, former Deputy-Governor, Bank of Canada
- Mitchell Julis, Co-Chairman and co-CEO, Canyon Capital Advisers
- Hans-Helmut Kotz, Visiting Professor, Harvard, former Executive Board Member for Financial Stability, Deutsche Bundesbank
- Ewald Nowotny, Governor, the National Bank of Austria
- Lord Gus O'Donnell, Former Cabinet Secretary and former Permanent Treasury Secretary, British Government
- Svein Harald Oygard Former Deputy Finance Minister, Norway, Former Governor Bank of Iceland
- Simon Potter, Executive Vice President, Markets Group, Federal Reserve Bank of New York
- George Soros, Chairman, Soros Fund Management
- Coen Teulings, Director, the Netherlands Bureau of Economic Policy Analysis
- Jean-Claude Trichet Chairman of Bruegel and former President of the European Central Bank
- William White, Chairman, Economic Development and Review Committee of the OECD, former Executive Committee Member, Bank for International Settlements (BIS)
- Michael Wilens Vice Chairman, Fidelity

# 2 Why is CRISIS needed?

The global financial crisis destroyed the faith that policymakers and the general public had in existing economic models, and the economic thinking that had failed to foresee the disaster.

The CRISIS initiative aims to fill that gap by developing a new approach to economic modelling and by understanding risks and instabilities in the global economy and the financial system from their roots.

In his opening address to the European Central Bank's annual conference on 18 November, 2010, ECB President Jean-Claude Trichet said 'macro models failed to predict the crisis' and added: 'In the face of the crisis, we felt abandoned by conventional tools'. The models and tools that central banks, finance ministries, and regulators still use today generally rely on three key assumptions: (a) that households, firms and governments are perfectly rational entities and that these tend to behave in similar ways to each other; (b) that markets always

ensure supply and demand are balanced and that the economy does settle into a balanced 'equilibrium' state, and (c) that detailed institutional structures and the interconnections of the financial system (the 'plumbing' of financial markets) do not generally matter for macroeconomic policies. One lesson from the crisis was that the models failed to match how people and institutions behaved. This meant that all these assumptions were fundamentally wrong: markets failed to clear; major economic imbalances emerged; and the plumbing connections had systemic implications for real economies.

The CRISIS initiative is very different from current models in that it will:

- be a 'bottom-up' rather than 'top-down' model that takes account of the different ways that households, firms and governments behave; incorporating the latest evidence from behavioural economics;
- collect new data on how people make decisions using experimental economic techniques;
- explicitly look at the network and institutional structure of the financial system and how that impacts on the economy; and ultimately test the model using empirical data.

# 3 License



# 4 The Purpose of this Document

This deliverable expands upon several earlier European Commission deliverables, including

1. *A Simulation Platform of a Model Incorporating the Real Economy* (delivered November 22nd 2013)
2. Deliverable 8.2 *Description of Integrated Model*
3. Deliverable 8.3 *Simulator*, and
4. The European Commission Final Review presentations, Brussels, November 2014.

The above are superseded by this document. This document outlines our progress toward making the overall CRISIS economics system. The integrated simulator was highlighted to the Bank of England in late 2014 and received very positive feedback.

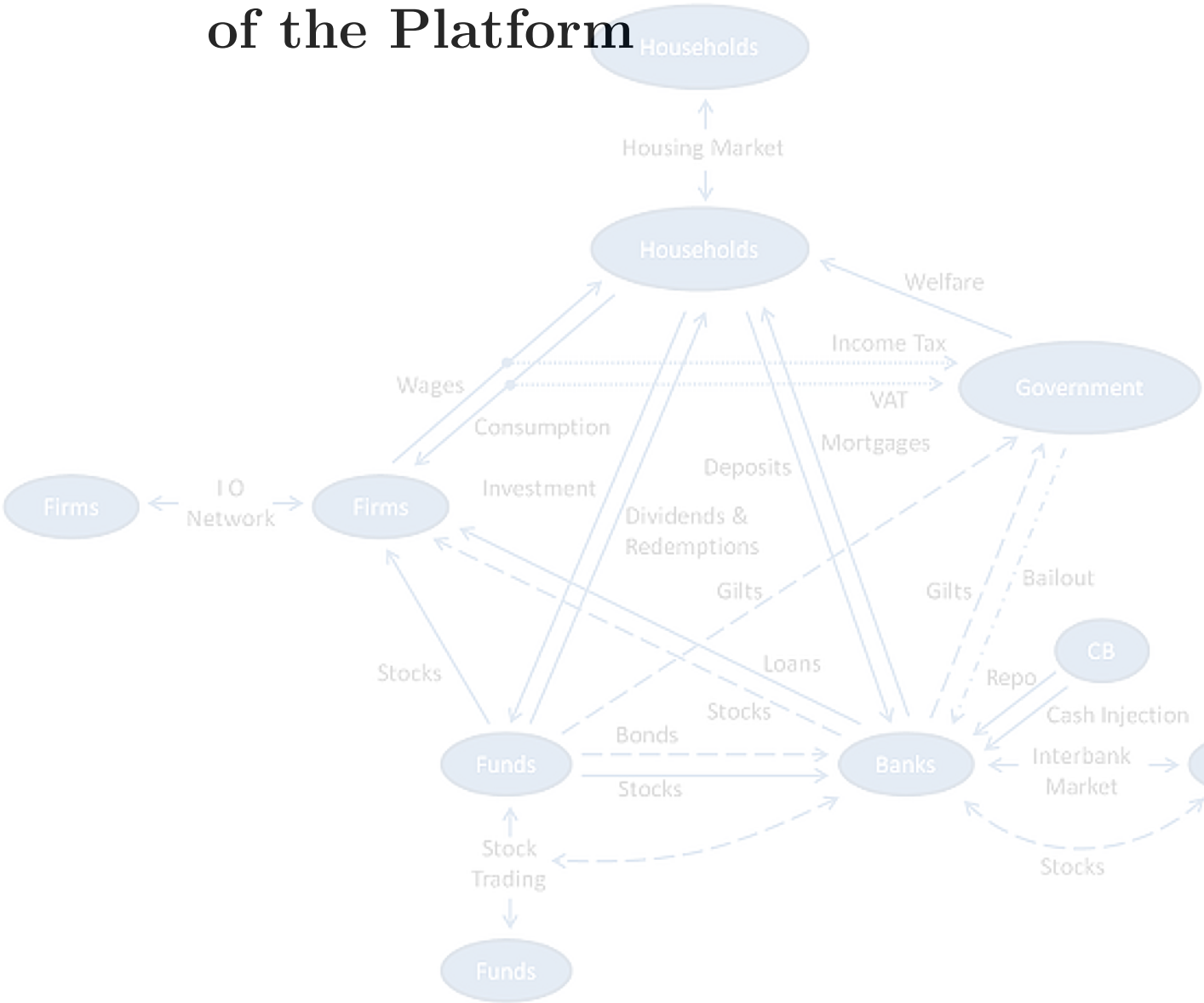
# 5 Contributors to the Integrated Codebase

Contributors to the integrated codebase include, but are not limited to:

J. Kieran Phillips
Ross Richardson
Milan Lovric
Olaf Bochmann
Bob De Caux
Sebastian Poledna
Christoph Aymanns
Gerald Gurtner
Ermanno Catullo
Jakob Grazzini
James Porter
Richard O. Legendi
Fulvio Corsi
Gábor Ferschl
László Gulys
Gábor Szemes
Tibor Baranya
Márton Iványi
Ervin Mikus
Tamás Máhr
Daniel Tang
Victor Spirin
Aurélien Vermeir
Marotta Luca
Bocsi Rajmund
Anatolij Gelimson
Luke Friendshuh
Ariel Y. Hoffman
Peter Klimek
Alessandro Gobbi
Fabio Caccioli
Balázs Bálint
Attila Szabó
Zsolt Kúti
Balzs Adamcsek
Viktor Erdlyi
Róbert Mészáros

# Chapter 2

# Description and Discussion of the Platform



# 1 Recent Updates and Improvements

2014–15 was an especially active period of development for the simulator. This period culminated in the initial public release under the GNU GPL license (Version 3).

The integrated codebase is currently hosted on a public github repository located at `github.com/crisis-economics/CRISIS`. Prerequisites for the use of the simulator include the Java Virtual Machine (available from Oracle) and `git` (a source version control system).

When using a Linux operating system, such a Debian or Ubuntu, `git` can be installed via a terminal:

```
sudo apt-get install git
```

A copy of the codebase can then be downloaded using:

```
git clone https://github.com/crisis-economics/CRISIS.git .
```

This instruction will download a copy of the repository and place it in the current directory (namely, `.`). Depending on the speed of your internet connection, this download may take time.

On Windows/Macintosh computers `git` may be available via powershell/-mac terminals, respectively. Windows and Macintosh systems also support a variety of github desktop clients.

The repository can also be downloaded as a zip file directly from:

```
https://github.com/crisis-economics/CRISIS/archive/master.zip
```

## 1.1 New Components of the Model

Among other deliveries in 2014/2015, new components and features in the integrated codebase include:

1. General stabilisation and repairs for the financial subsystem, including repairs for Basel II regulations and policies. Realistic, permanent procyclical leverage cycles are now observed in all sectors of the economy when Value at Risk leverage constraints are enabled.

2. Various improvements and new features for the macroeconomic subsystem including (a) customizable Input–Output (IO) production networks, (b) capital and durable goods economics, (c) some differentiation between retailer/manufacturer type firms (constrained consumption) and (d) yield shocks (non–constant firm productivity factors), including lognormal random walk productivities and Ornstein–Uhlenbeck productivities.

3. Management and configuration tools for model parameters.

4. Simulations have improved numerical reproducibility for otherwise identical initial conditions.

5. Modularisation and the ongoing effort to produce interchangeable (namely 'lego brick') components has been very effective. The model contains several hundred distinct configuration options and a graphical user interface dedicated to managing these components. GUIs have been improved, both in terms of their aesthetic appearance and their usability.

6. Sporadic unrecoverable runtime exceptions raised by the legacy codebase have, in the large part, been addressed. The codebase is at this stage sufficiently robust that no runtime failures were observed using any reasonable initial conditions for a large part of a year. Spontaneous bank respawning behaviour following bank liquidation (the creation of new bank agents at times other than at initialization) remains unimplemented for technical reasons. Therefor one of the only ways for a simulation to terminate naturally is for the financial system to collapse as a result of a chain of bank liquidations.

7. Unit tests and code coverage have been significantly expanded. The test suite now includes agent behavioural tests, numerical market tests and random–configuration based stress testing (neverending tests running randomly configured simulations in order to identify runtime failures).

8. Sophisticated general purpose market processing algorithms have been installed, including heterogeneous commercial loan markets, bond and gilt markets and some support for risk–graded commercial loans.

9. Agent/market/decision rule initial conditions have been modified in such a way that virtually any customizable feature in the codebase (for instance household cash endowments, public stock emission values, firm decision rules, bank bankruptcy policies etc.) can be initialised heterogeneously at $t = 0$. Many simulation parameters can be configured to evolve steadily or change in time (eg. as lognormal random walks) or can be drawn from user–supplied statistical distributions.

10. Bank bankruptcy resolution processes (eg. bailin, bailout) are more realistic than has been the case previously and are implemented to a higher quality standard. These policy modules have significantly more prolonged effects.

11. A gilt-issuing government entity with support for welfare, benefits, public sector employment and government consumption has been added.

12. The central bank (base rate) Taylor rule has been patched. In the legacy model the central bank base rate algorithm was responsible for producing 'not–a–number' floating point values that gradually spread through the simulation world. The central bank agent now conducts its business according to plugin financial policies and interchangeable financial decision rules, including rules that regulate whether or not to issue uncollateralized loans. The central bank is able to process share transactions. The government agent may be used in the absence of a central bank agent, but the central bank may not be used in the absence of a government agent.

13. Repo loan activity is nonzero, sustained and more meaningful for appropriate initial conditions. A numerical money sink in which the central bank could drain cash flows from the rest of the economy after a prolonged period of use has been patched.

14. By default commercial bank stock market capitalisation has been reduced to more reasonable levels. Optionally, banks may be prohibited from buying stock in other banks (due to professional competitiveness).

15. Contract classes have been diversified to include (a) extended commercial loans, (b) graded commercial loans, (c) extended commercial bank bonds, (d) extended government bonds (namely, gilts), (e) uncollateralized cash injection loans (UCILs), (f) working repo loans, (g) interbank loans, (h) mortgages, (i) capital and durable inventory assets, and (j) loans with no accumulating interest.

16. Settlements have been extended to include automatic VAT, PAYE and taxable labour wages, customizable (or disabled) dividend taxes, welfare and benefits.

17. Stocks are no longer transient assets. Shares are traceable tradeable cash–flow consistent assets that can be retained until the underlying stock instrument is erased.

18. An interbank market has been integrated into the platform. The interbank market is a residual synchronous financial market over which (notionally expensive) central bank repos are converted to (notionally cheap) interbank loans. The interbank feature can be disabled.

19. Support for synchronous and asynchronous signals (also known as event busses) have been integrated as a general simulation service to which any agent of algorithm may subscribe.

20. Interchangeable plumbing configurations (agent $\leftrightarrow$ agent and agent $\leftrightarrow$ market relationship diagrams) have been created and integrated with the main simulation GUI.

21. Simplified alternative implementations for several agents have been added to the integrated codebase. These agents are largely interchangeable with their existing macroeconomy counterparts. New agents include (a) exogenous agents (actors whose deposits are isolated from the regular financial system, or whose deposits are partially modelled) and (b) remote–control type actors, whose behaviours and whose future decisions are governed by timeseries that are specified by the user at $t = 0$.

22. Some support is provided for the use of real FTSE100 firm/sector and commercial bank names, input–output tables, stock market capitalisations, and relative agent wealths.

23. Mutual fund agents and their interactions with households have been stabilised. So–called noise–trader banks have been converted to noise–trader funds (NTFs) as per the intent of their original design. Various bank behavioural decision rules have been converted to fund behavioural decision rules.

24. A number of models that depended on duplicated common code were unified as special cases of a flexible master model. Interesting preset use cases for this skeletal model, including endogenous macroeconomic business cycles, artificial/forced bank bankruptcies (bailins, bailouts, liquidation), Basel II policies and procyclical leverage, gilts/bonds/graded commercial loans and simplified economies with no input–output networks are provided via the user GUIs.

25. More intuitive and flexible control over simulation time, relative event scheduling orders, and event priorities.

26. The sequence of named events in the simulation cycle has been revised, owing to questionable choices in the original program.

27. Fixes for some long/short term 'money sink' and 'money fountain' behaviours, including cash loss caused by central bank hoarding.

28. A longstanding issue with the codebase in which loans requested by borrowers accounted for low (tiny) fractions of borrower equities is believed to have been solved. Macroeconomic firms/establishments now typically borrow a large fraction of their equities from the credit markets. Aggregate commercial loans have been increased by several orders of magnitude.

29. The extent and detail level of inline documentation and javadocs has been increased.

30. Module usage instructions and formal parameter descriptions have been integrated with the user GUIs and can be changed or added (in the case of new modules) by the user.

31. The codebase is Java 8 compatible.

## 1.2 Modularisation and Plugins

In early 2014 is became apparent that end users wanted to modify the standard model in order to install their own features. The legacy model thus consisted of a set of disjoint model files numerously copied and slightly modified in each use case. The resulting array of model files became difficult to maintain because a change in the simulator that was a breaking change with respect to one model file was invariably a breaking change with respect to all other model files. In many cases it was of interest, for development and staging purposes, to modify the decisions made by an existing agent and observe the end result of this change in the context of a full simulation. However in the legacy code many agent decisions and behavioural instructions were aggregated with and hardcoded alongside numerous other such decision rules inside agent class file bodies, meaning that multiple slightly different implementations of the same agent all sharing substantial common code seemed inevitable.

To address this problem an amount of effort was dedicated to refactoring existing agent types in such a way that it is possible to specify agent behaviours and agent decision rules as dependencies (to be supplied to the agent when it is constructed). As such it is possible in many cases to implement a new decision rule without needing to copy and coerce an existing monolithic agent class

file. These interchangeable decision rules and modules are loosely described as 'plugins'. As far as the existing legacy codebase structure would allow, large parts of the core standard model macroeconomy and some parts of the financial system have been modularised and converted to a plugin architecture.

Many actors in the standard model therefor delegate their behavioural decisions to standalone and customisable plugin algorithms. As a result it is often possible to work on a small or isolated subsection of the simulator in order to introduce new behaviour (for instance the algorithms governing household consumption budgets). This approach has greatly increased the scope of possible behaviours for stock agents in the standard model, has empowered the user to create their own behavioural programmes for study, and has also unified a large number of otherwise virtually unmaintainable and disparate model files.

Plugin aspects of the standard model include, but are not limited to:

(i) Decisions made by firms when computing the target production ($q^*$).

(ii) Goods market (bid/ask) price selection algorithms for all market participants.

(iii) The form of the firm production function (eg. Cobb Douglas, Leontief) and its optimisation procedure.

(iv) Labour wage bidding algorithms (expected and offered wage $\omega$ per unit labour).

(v) Valuation algorithms for inventory.

(vi) Domestic consumption functions for households.

(vii) The domestic consumption budget algorithm (a cash value $B \geq 0$ to be spent on goods).

(viii) The implementation of the goods and labour markets.

(ix) The distribution of initial household cash endowments.

Almost any algorithm, decision rule or set of behavioural instructions that are encapsulated in an object whose API is governed by an interface can be treated as an interchangeable component in the context of the standard model.

It may be the case that otherwise identical agents delegate the their decision rules to algorithms of different types. For instance two firms belonging to the same simulation may be heterogeneous with respect to their production functions.

## 1.3 Lessons

Broadly speaking the codebase is far more stable and useable than was the case in 2013–2014. A variety of new, innovative and potentially powerful features have been integrated successfully. Unfortunately the project continues to evolve around legacy code designs, structural issues and some coding issues inherited

from its earliest versions. Some standard features, for instance an effective logger, are either absent or underutilized. It is conceivable that this problem arose because insufficient emphasis was placed on professional engineering principles for much of the project, namely that the capabilities of academic and enthusiast contributors were assumed to suffice. The importance of profiling, static analysis, professional software design and integration must not be underestimated in future. Hacking the codebase in order to insert new features and/or repeatedly installing quick fixes has a cumulative detrimental effect and these actions can require much time and effort to undo.

At length a large number of repairs, bug fixes and general improvements (aesthetic and otherwise) have been committed to the codebase. In particular the core CRISIS code now contains no Java compilation errors and no warnings at the standard level of inspection. This entailed corrections to many hundred coding problems.

In future care should be exercised in the following areas: (a) events should not be scheduled to occur repeatedly instead of once, as required, as eventually the master simulation event queue becomes congested with defunct simulation events; (b) where standard data structures and/or standard algorithms can be utilised, this should be the case. For instance, searching for an element in an array by sequentially testing each element in the array could very probably be improved by utilising a stock hash data structure; (c) it should be noted that some model GUIs have quadratic $\mathcal{O}(t^2)$ or worse time–complexity after approximately $t \geq 500$. The computational cost of using a GUI can ultimately dominate the total computational cost of the simulation; (d) unnecessary function calls and/or unnecessarily synchronised methods; (e) excessive or unintelligible reporting and console output; (f) convoluted or circuitous code design.

Based on wall–clock timings using a modern Macintosh computer, the overall performance of the macroeconomic simulator has been sped up by a factor of $\approx 100$ up to $t = 2000$ simulation timesteps. The codebase appears to have negligible cumulative slowdown up to $t = 3000$ simulation cycles (750 years simulation time), and similarly the simulator appears to have no memory leaks when run with regular settings. These improvements greatly increase the readiness of the platform for use as an interactive game engine.

It should further be noted that the use of the simulation engine as a game platform entails configuring the simulator to operate in a way that is suitable for the game experience. The timing and the validity of all measurements displayed to the user via the game interface screen must be treated with care.

# 2 Overview of the current CRISIS model

## 2.1 Fundamentals

The concept of future event scheduling is important to the simulation platform. Future event scheduling is a request made that a piece of simulation code should be executed (evaluated) at a certain future time in the virtual world. The proce-

dure used for future event scheduling has been hugely simplified. The following code snippets indicate two routine examples of scheduling operations.

In this first example a class called `Schedulable` intends to schedule one of its methods (`methodToProcess`) for future evaluation. `methodToProcess` should be evaluated exactly once per simulation cycle at time `AFTER_ALL`. The platform automatically establishes a lock to `methodToProcess` and then uses this lock to reflect and execute the method every time it becomes due in the schedule:

```
/**
  * Repeatedly schedule an event
  */
public class Schedulable {

   public Schedulable() {
      // Process the function "methodToProcess" once every simulation cycle,
      // at time AFTER_ALL.
      Simulation.repeat(this, "methodToProcess", NamedEventOrderings.AFTER_ALL);
   }

   @SuppressWarnings("unused")   // Scheduled
   private void methodToProcess() {
      // The method to process.
   }
}
```

Despite its obvious reliance on reflection services this approach does appear to be as efficient as is required. The underlying reflection process is compiled (streamlined) after a set number of invocations. This approach can be further improved, but changes are postponed until the next edition of the codebase.

Use cases were quickly identified in which events occurring at the same simulation time nevertheless required special mutual orderings, or priorities, over one another (for instance two different methods scheduled at `AFTER_ALL` for which the relative evaluation order is important). An example of this is the need for all stock-releasing agents to compute their intended dividend payments before any such payment is actually made: otherwise the liquidity (cash at hand) of at least one agent is polluted by incoming dividends before the size of the dividend payment is decided, producing a circuitous dividend stream.

For this purposes we added the option of relative scheduling priorities to otherwise concurrent events in the simulation schedule, as indicated by the following example:

```
   /**
     * Schedule two events, with relative priorities
```

```
    */
  public class Schedulable {

    public Schedulable() {
      Simulation.repeat(this, "first",
        CustomSimulationCycleOrdering.create(
          NamedEventOrderings.BANK_SHARE_PAYMENTS, 1));
      Simulation.repeat(this, "second",
        CustomSimulationCycleOrdering.create(
          NamedEventOrderings.BANK_SHARE_PAYMENTS, 2));
    }

    private void first() { }   // Executes at BANK_SHARE_PAYMENTS (1)

    private void second() { }  // Executes at BANK_SHARE_PAYMENTS (2)
  }
}
```

In the above example `first` and `second` are both evaluated at time `BANK_SHARE_PAYMENTS`, but `first` will always be evaluated before `second`.

## 2.2 Subeconomies

### 2.2.1 Groups of Agents

One aim of the integrated platform described in this document is to provide a model containing the key elements of both the macro (real) economy and the financial system, making it possible to study interactions between the two.

CRISIS comes with a flexible Standard Model. The standard model is segregated into subeconomies of agents with particular types and aims. The standard model provides a financial subeconomy of commercial bank and/or shadowbanking agents and financial intermediation actors. The standard model also contains an industrial subeconomy with firms and establishments (which may be macro– or micro– scale actors) grouped into productive sectors. The classification of agents according to subeconomies is notional. End users may wish to construct a model in which firms provide financial products and therefor belong wholly in the financial subeconomy or in both the financial and industrial subeconomies. Subeconomies, like agents in the standard model, are interchangeable components with multiple possible implementations. The default implemetation of a subeconomy (a null subeconomy) is empty and does nothing.

A list of subeconomies of the standard model is as follows:

1. *The Banking Subeconomy/Banking Financial Institutions (The Financial Sector)*

   A financial subeconomy with multiple differentiated commercial banks, customizable bank bankruptcy resolution policies, optional interbank trading activity, and customizable bank market behaviours. The default im-

plementation of this subeconomy consists of fundamentalist and chartist commercial banks and commercial banks whose behaviour is a mixture of fundamentalist and chartist states.

*Fundamentalist* banks use a dividend/price ratio when gauging the attractiveness of stock investments in the following way: $r = d/(p \cdot \sigma)$ where $r$ is the expected return rate per unit cash invested in stock, $d$ is the dividend per share payable on the stock, $p$ is the price per share, and $\sigma$ is a divisor called the 'stock risk premium' or the 'stock return adjustment factor'. The dividend per share $d$ and the stock price $p$ are taken to be the most recently observed values. The stock return adjustment factor $\sigma$ is customizable. Returns on other investments, for instance commercial loans, are taken to be their respective values from the end of the previous business cycle.

*Chartist (Trend Follower)* banks instead measure the change in stock price over time when calculating the expected return rate of stocks in the following way: $r = (p(t) - p(t-\ell)) \ / \ p(t-\ell)$ where $p(t)$ is the most recently observed stock price and $\ell$ is a customizable nonzero lag parameter specifying the number of simulation cycles elapsed between price observations. Returns on other investments, for instance commercial loans, are taken to be their respective values from the end of the previous business cycle.

Commercial banks can hold deposits from households, funds and firms, and banks can issue mortgages to households, borrow from funds, and trade in firm (and optionally bank) stocks on the stock market. If a central bank agent (lender of last resort) is enabled, commercial banks can also apply for collateralised central bank loans (repo) and uncollateralized central bank cash injection loans (UCILS). If the interbank market is enabled, commercial banks can form interbank loans via the interbank market.

Typically, at the start of the simulation, in order to raise capital to finance their operations, commercial banks will issue stock. The initial stock offering is held by funds, and funds subsequently trade these shares on the stock market.

It is possible for commercial banks to go bankrupt and/or experience negative equity. This may or may not result in intervention by the government or a bridge bank entity.

2. *The Central Bank/Lender of Last Resort*

   The central bank ordinarily has a large finite cash reserve at its disposal. The central bank lends to commercial banks whose cash reserves are low. Commercial banks apply to the central bank for collateralised loans (repo loans) and for uncollateralized cash injection loans (UCILS).

The willingness or otherwise of the central bank to issue repos or UCILs is governed by its policy decision rules, a set of customizable interchangeable decision rules. Repo/UCIL interest rates are decided by the central bank.

By default the central bank uses Taylor rules to select the repo loan interest rate. These Taylor rule are linked to trends in gross (overall) production among firms.

3. *Industrial sector (Non-Financial Establishments and Corporations)*

   By default this subeconomy is implemented as a collection of macroeconomic firms. Firms produce goods and goods are differentiated by sector and by selling price. Firms have customizable heterogeneous stock emission values, customizable credit demands, goods pricing and target production decision rules, wage bid prices and dividends (Ie. liquidity targets). The firm production function, and structure of the IO network and its technological weight components are customizable.

   Commercial banks are the initial stockholders of firms. Goods can be durable or non-durable, and firms may or may not require input goods for production (as opposed to just labour).

   This subeconomy is sometimes called 'the real economy' or 'the macroeconomy'. Firms demand credit (commercial loans) from the financial sector and use this money to pay labour wages and finance production.

   Configurations for the industrial sector include, but are not limited to:

   - *Cobb Douglas*
     A simple economy in which many competing firms have Cobb Douglas production functions depending only on labour. Firms do not require input goods for production. The labour supply is fixed and firms produce nondurable goods (which may or may not be differentiated). In earlier times this implementation was described as the 'mark one' economy (a deprecated term).

   - *Input Output (IO) Network*
     A more sophisticated economy in which competing firms require both input goods and labour for production. The structure of the IO network (how firms depend on one another's outputs) is defined by a graph and can in principle have any configuration. This is typically a completely closed economy in which households are employed by firms and households deposit their money in banks. Bank receive income from labour wages and from investment accounts.

4. *Households*

   The household subeconomy is usually the driving force of the simulation world. Households supply labour (either in fixed amounts per unit time, or in variable amounts, depending one some criteria) and consume goods

produced by firms.

The total quantity of goods consumed by households is determined by interchangeable household decision rules. Ordinarily aggregate household consumption is in proportion to the total wealth of households. By default the household labour supply is fixed in time (though the labour supply may itself be heterogeneous among household agents) and consumption is taken to be a fixed fraction of liquidity. AR1–type consumption is available and buffer-stock consumption is in development. When deciding what type of goods to buy, households select a consumption budget and then divide this budget among differentiated goods according to customizable decision rules. By default households prefer to buy cheaper goods according to a CES algorithm.

5. *Bad Bank*

   The bad bank is a financial entity that assumes the assets of failing commercial banks in order to liquidate these assets and compensate creditors. The bad bank is typically active when (and after) a commercial bank is liquidated due to serious bankruptcy.

6. *Non-Banking Financial Institutions (Funds)*

   Funds provide investment opportunities to households and are not banks. Funds include mutual funds and repo–issuing funds that lend money to banks collateralised by existing bank assets. Mutual funds can lend money to banks and can invest in the stock markets. Typically mutual funds aim to stay at zero equity and maximise their assets.

7. *Government*

   By default the government subeconomy contains one unique central government agent. Local government and public institutions may be possible as extensions to the standard model. The government has a domain of authority. In all standard configurations the domain of government authority is all other agents in the simulation world (including firms, households and banks). The government might make welfare payments to unemployed households, employ a public sector workforce and/or consume goods and services offered by firms. The government is ultimately responsible for financing commercial bank bailouts. Bailouts may prove to be costly, and the government may be required to reduce spending and consumption as a result of an expensive bank failure. Following a bailout the government may be compensated with shares as a result of intervening in the activities of a failing bank, and the government may seek to gradually sell these shares.

8. *Agencies*

   Agencies are agents whose activities do not directly affect the simulation world. Agencies are observers that provide on-demand information about economic activities eg. data about profits made by firms. An agency may be able to supply objective risk measurements about firms, or monitor the equity of a firms over an extended period so that this information can be used by potential investors.

9. *Markets*

   Markets are general structures over which agents exchange goods and services or otherwise conduct trade. Standard markets include an implementation of a goods market (durable goods or otherwise), a labour market, interbank loans, repo loans and UCILs, secondary commercial loan markets, bank bonds, primary or secondary government gilt markets or stock markets. These markets may be continuously processed, processed at as a result of triggers/signals or processed at fixed customizable times in the simulation schedule.

   Many markets in the standard model provide peer to peer trade. Market participants respond not to the overall state of the market but rather to all opportunities offered by all other participants: a firm is inclined to secure the cheapest possible commercial loans, so the firm is not only interested in the average commercial loan interest rate but is also interested in the relative attractiveness of all commercial loans offered to it by all known banks.

   Until recently the simulator offered a simplified interbank market which implicitly assumed that all firms and all commercial banks shared a single common loan interest rate. An interbank market not subject to this restriction has been designed and integrated. Markets are outlined further in §2.2.2

Many of the individual components illustrated here have been discussed in other documents in particular *Deliverable 2.2 The Financial ABM* and *Deliverable 3.3 The Macro ABM*. These components typically require a limited amount of modification and coercion to achieve integration. In due course the simulator should be calibrated against real world data. Ross Richardson (Oxford) was involved in creating model calibration objectives.

Among many objectives one key aim of CRISIS is the ability to study interactions between the financial subsystem and the real economy. We begin with a discussion of various possible models for the macroeconomy and the background of the standard model. In earlier versions of the simulator the so-called 'mark one' economy was known to be incompatible with certain aspects of the financial system including market processing algorithms. Instead of coercing the mark one model to be compatible with the modern simulator it was instead observed that the mark one economy is similar to a special case of the IO model.

### 2.2.2 Markets

Three conceptually different types of market are implemented. These are: (1) the limit order book market, (2) the clearing markets and (3) bipartite graph/simple matching markets. Broadly speaking markets are classified as 'homogeneous' if the underlying traded resource (including its price) does not depend in any way on the underlying buyer/seller pair and the market is described as 'heterogeneous' if the underlying traded resource depends strongly on the buyer/seller pair, for example if the seller has quoted a price to the buyer and this quote differs for other buyers.

Clearing markets are more complex than other types of market in that participants do not post fixed orders to the marketplace. Clearing market participants state what amount of trade they are prepared to admit *given* a set of parameters $P$ (for instance the prices of goods, or the interest rates due on commercial loans). The market is responsible for selecting the values of $P$. The mechanism used to select $P$ is arbitrary. $P$ is typically chosen so that the supply and demand for each market instrument are equal. Depending on the value of $P$, a participant in a clearing market may behave as a seller or as a buyer or both.

Bipartite graph matching markets are markets composed of seller nodes and buyer nodes. Nodes are either sellers or buyers and not both. In this market the primary concern of each buyer is to secure a quantity $q$ of a resource at no more than price $p$ per unit. Likewise the primary concern of each seller is to sell a quantity of goods at no less than the ask price.

Some markets implemented in the codebase are outlined below.

**Stock Markets**

The trading partners on the stock market are funds, commercial banks, bridge banks, the government and financial intermediaries. The stock market is a clearing market: the response of each stock trader depends on the price of the stock.

Shares are transferable assets and stock traders can, at their option, decline to participate in stock markets and retain the shares they already own. The value of shares can go down as well as up, meaning that stock traders who do not engage in the stock market can nonetheless make capital gains and capital losses.

Shares are often redistributed, created (Ie. issued) or purposefully destroyed as a result of bankruptcy resolution processes.

**Commercial Loan Market**

The trading partners on the commercial loan market are commercial banks and firms/establishments. The commercial loan market may optionally be implemented as (a) a limit order book (LoB) market or (b) as a clearing market. The standard model provides a clearing market based on (b).

In case (a) the limit order book loan market is processed as a price–time priority LoB once per simulation cycle. Firms place buy orders and commercial banks place limit sell orders.

In case (b) the market may be processed either in heterogeneous or in homogeneous mode. In heterogeneous mode firms may be quoted different interest rates by each bank and banks may offer different interest rates to each firm. In this configuration there are as many distinct commercial loan interest rates as there are borrower-lender pairs. Banks may, at their option, use risk measurements and/or other historical data when formulating interest rates quoted to each firm. In homogeneous mode all commercial loans have the same interest

rate irrespective of the borrower–lender pair.

In the standard model firms compute their ideal credit demands $L^*$ (the size of the commercial loan required by the firm in order to have sufficient liquidity to meet all production targets) and then ration the size of the commercial loan requested $L \leq L^*$ based on the quoted interest rates. A firm will not request any new commercial loans if the interest rate on the loan exceeds the production markup rate because such a loan would inevitably result in a marginal loss.

**Interbank Market**

Banks (including commercial banks and the central bank) trade loans on the interbank market. Borrower place buy orders and lenders place limit sell orders. Lenders may quote different prices to each counterparty. The market is executed as a price–time priority LoB once per simulation cycle.

**Labour Market**

The labour market is a bipartite market in which firms and government are employers (labour buyers) and households supply employees (sell labour). The labour supply among households may be heterogeneous or fixed. Households may divide their labour among employers. Double employment is not permitted.

The price per unit labour per unit time is formulated by the employer (the bid price) and by the employee (the ask price). The market is responsible for matching labour contracts once prices and desired volumes are posted by employers and employees. Typically the labour market executes this matching by first rationing either the employers (in the event that demand exceeds supply) or by rationing the employees (in the event that supply exceeds demand) and then allowing labour contracts to flow over the resulting balanced graph.

Labour ask/bid prices need not be fixed in time or homogeneous among agents. By default the standard model uses a fixed customizable wage $w > 0$ shared by all employers and employees.

**Goods Market**

Firms and households trade durable and non-durable goods via the goods market. The commercial loan market may optionally be implemented as (a) a limit order book (LoB) market or (b) as a bipartite matching market. The standard model provides a matching market based on (b).

In case (a) firms submit limit sell orders for goods they produce. Households submit limit buy orders quoting a bid price. The market sorts these orders and selects a price that maximises the traded volume. Any unfulfilled or partially traded sell orders are cancelled at the end of the simulation cycle.

In case (b) firms submit a maximum saleable quantity $q$ and an ask price $p$ per unit. Buyers (including household agents, government, and other firms)

submit desired consumption quantities $d$ and bid prices $p^*$ per unit. The market is responsible for matching buyers and sellers in such a way that the sum of all goods sold does not exceed $q$ for any seller and the sum of all goods bought does not exceed $d$ for any buyer. All resulting contracts must have a price $\pi$ satisfying $p \leq \pi \leq p^*$.

Buyers and sellers are required to maintain an inventory of goods. In the standard model households do not resell durable goods after consumption. Capital (durable) goods are partially depleted by production activities and also decay in time.

### 2.2.3 Implementation

The following is an overview of the key constructs used to implement the markets described above. All markets have instruments and participants and some markets have a concept of orders.

#### Instruments

Instruments represent items and resources being traded on the market (eg. deposits, goods, labour, loans, stocks). Instruments for the same underlying resource may be differentiated by maturity: eg. a labour market may offer distinct instruments for labour contracts of length 1 year and 2 years.

#### Orders

Market participants submit 'orders' to the market via an instrument. Orders are labelled 'buy' or 'sell' depending on the intentions of the participant. There are specific order implementations for deposits, goods, labour, loans and stocks.

#### Participants

For each limit order book market instrument there are a pair of matching participants whose responsibility is to generate buy and sell orders. Examples of these matching participants include:

| Buyer | Seller |
|---|---|
| GoodsBuyer | GoodsSeller |
| Employer | Employee |
| StockHolder | StockReleaser |
| StockBuyer | StockSeller |
| Borrower | Lender |
| DepositHolder | Depositor |

## 2.3 Recorder Sources

### 2.3.1 Introduction

The CRISIS Software Library features a data collection and recording framework that allows modellers to handle the cross-cutting aspects of collecting and

outputting data during a simulation in a declarative manner. Ideally, the model code should only be concerned with the behavior of the modelled entities or phenomena, and should not be littered with code writing out data to files. The separation of this concern is achieved by employing Java annotations to allow modellers to define what should be recorded into files and when. In this section, we describe how the recording annotations are used, what kind of data we can record from a simulation model and how.

An annotation, in the Java computer programming language, is a form of syntactic metadata that can be added to Java source code. Classes, methods, variables, parameters and packages may be annotated. Unlike Javadoc tags, Java annotations can be reflective in that they can be embedded in class files generated by the compiler and may be retained by the Java VM to be made retrievable at run-time. The data collection and recording framework in the CRISIS Software Library searches the model being run for annotations and processes them if found.

There are two different annotations that are interpreted by the framework: `@Recorder` and `@RecorderSource`. The `@Recorder` annotation should be attached to a Mason model class, that is to a class that descends (perhaps indirectly) from `eu.crisis\_economics.abm.Simulation`.

Listing 2.1: A @Recorder annotation example

```
@Recorder(value="results.txt",
    recordAt=RecordTime.END_OF_ITERATION, sources={"someValue",
    "someOtherValue"})
public class MyModel extends Simulation {
  @RecorderSource("someValue")
  protected double someField;

  @RecorderSource("someOtherValue")
  protected int someMethod(){
    ...
  }
}
```

The three attributes of the annotation in Listing 2.1 are `value`, `recordAt`, and `sources`. The `value` attribute denotes the file name the recorded values are saved in. The `recordAt` attribute specifies when the requested values should be saved. Finally, the `sources` attribute lists data source names, names that denote values that should be saved.

In the simplest case, data source names refer to members of the model class that are annotated by the `@RecorderSource` annotation. In the above example, `@RecorderSource` annotations have only a single attribute, the `value` (in Java, every annotation has the `value` attribute, and when it is the only attribute, it doesn't have to be written out). The use of the `value` attribute is twofold. First, the `sources` attribute of the `@Recorder` annotation refers to the data sources using it. Second, when the values are written to a coma-separated file, these names are used as column names.

The result of the annotations in Listing 2.1 is a file called 'results.txt' which starts with a time stamp and the parameter values the model has been run with, and continues with a table with the recorded data, as shown in Listing 2.2

Listing 2.2: Pipe separated results

```
Timestamp: 2013.12.09. 15:56:35

...

"run"|"tick"|"someValue"|"someOtherValue"
1|1.0|3.467|34545
1|2.0|4.565|5775
```

Beyond this very simple example, there are many more ways the two annotations can be used to record values. It is possible to record multi-column values (e.g. arrays), it is possible to record statistics instead of raw values, and it is possible to define data sources of other than primitive types. Full details on the recording framework is presented in the following subsections.

### 2.3.2 Recording

Recording data to files during a simulation is handled by two annotations in the CRISIS Software Library: the `@Recorder` and `@RecorderSource` annotations. This chapter describes the usage of these annotations, and discusses the implemented use cases.

Via the annotations, the recording framework provides a declarative way to specify what should be recorded. During the simulation, the framework makes sure that the specified data is written to a results file. The data is recorded in a comma-separated style, although the separator character is the pipe symbol ('|'). The type of data that can be recorded is limited to numbers and strings. Numbers can be represented either by primitive types (`int`, `float`, etc.) or by their object variant (`java.lang.Integer`, `java.lang.Float`, etc.), including `java.lang.Number`.

The name of the results file is specified by the `value` attribute of the `@Recorder` annotation. If a simple file name is given (e.g. '`results.txt`'), the results are saved in a file with the given name in the system temporary directory (e.g. '`/tmp/results.txt`' on Unix). If an absolute path is given (e.g. '`/simulations/results.txt`'), the file is saved in the directory specified by the path. If a relative path is given (e.g. '`./results.txt`'), it is interpreted from the current working directory, i.e. from the directory the simulation is started from or which is set by the '`-Duser.dir=somedir`' command line switch. Specifically, '`./results.txt`' saves the file in the current directory. Normally all data is saved in a single file. If, however, multi-column data is recorded (see Section 2.3.5), and the number of columns in a mult-column data source changes (e.g. between runs in a parameter-sweep experiment), the data is output in multiple files. Every time the number of columns changes, a new file is started with a name that is derived from the original file name by inserting a '`-partX`' string right before the file extension ('`X`' stands for a number incremented every time

a new file is created).

Recording the specified values and writing them to the file is handled separately by the framework. The '`recordAt`' attribute of the `@Recorder` annotation specifies when to record the specified values in memory. This can happen at every iteration, at every n-th iteration, at the end of runs, or when a condition evaluates to true. The '`outputAt`' attribute, however, specifies when to write the recorded values into the file. It is possible to output the values directly when they are recorded, or at the end of runs, or in every n-th iteration within a run. The default is to record at the end of a run, and to output at the same time as record.

What should actually be recorded is specified by the '`sources`' attribute of the '`@Recorder`' annotation. Each member of the `sources` list refer to a data source that is specified by a `@RecorderSource` annotation. `@RecorderSource` annotations can be attached to public or protected members (fields or methods) of the model class or its ancestor classes, and also to public members of related classes (classes the model class uses in collections or arrays). There are a number of different ways members can be annotated by `@RecorderSource`. The following sections introduce the different usages of this annotation.

### 2.3.3 Recording simple values

In the simplest case, model members of type number (`int`, `long`, `float`, `double`, `java.lang.Integer`, `java.lang.Long`, `java.lang.Float`, `java.lang.Double` or `java.lang.Number`), `boolean` (or `java.lang.Boolean`), `java.lang.String` or enum can be directly annotated by the `@RecorderSource` annotation.

Listing 2.3: Recording simple values

```
@Recorder(value="results.txt",
    recordAt=RecordTime.END_OF_ITERATION, sources={"someValue",
    "someOtherValue", "enumValue"})
public class MyModel extends Simulation {
  @RecorderSource("someValue")
  protected double someField;

  @RecorderSource("someOtherValue")
  protected Number someMethod(){
    ...
  }

  public enum SomeEnum {
    SOME_ENUM_VALUE,
    ANOTHER_ENUM_VALUE
  }

  @RecorderSource("enumValue")
  protected SomeEnum someEnumValue = SOME_ENUM_VALUE;

}
```

The value attribute of the `@RecorderSource` annotation specifies the data source name. This name is referred to in the `sources` attribute of the `@Recorder` annotation, and this will also be the header of the corresponding column in the recorder file (`results.txt` in the example).

The above annotations record simple values, a `double`, a `java.lang.Number`, and an enum value into separate columns of the results file. In addition to such scalar values, it is possible to record certain statistics of multiple values. This is detailed in the next section.

### 2.3.4 Recording statistics

To be able to record statistics on a set of values, we have to be able to denote a set of values as data source. To achieve this, you can annotate array or collection members (fields or methods returning array or collection types) by the `@RecorderSource` annotation. For example in Listing 2.4, the `someField` member of the `MyModel` class (on line 4) is annotated to be a data source under the name "`someValue`". Similarly, the `List<Double> someMethod()` method, returning a collection of `java.lang.Double` values, is annotated under the name "`someOtherValue`".

When the elements in the annotated array or collection are not primitive types or objects equivalents, you have to further specify an actual member of the given type. For example on line 7, `myClassList` is annotated to be a data source. `myClassList` is a list of `MyClass` objects, which is not immediately recordable. Therefore, the "`member`" attribute is used in the `@RecorderSource` annotation to let the framework know which member of the `MyClass` instances should be accessed. The denoted member should be a public member of `MyClass`, and it should return a number of some type. If the string assigned to the `member` attribute ends in "`()`", then a method is searched for, otherwise a field. In the example below, the "`myMethod()`" member is specified, therefore to record values from this data source, the framework will call this method on each element of the `myClassList` collection to retrieve a set of values. Finally, if a raw collection is annotated to be a data source, as on line 10, the framework cannot infer the type of the elements. Therefore, you have to further specify this by the `innerType` attribute of the `@RecorderSource` annotation.

Listing 2.4: Recording statistics on collecions

```
@Recorder(value="results.txt",
    recordAt=RecordTime.END_OF_ITERATION,
    sources={"sum(someValue)", "avg(someOtherValue)",
    "sd(myClassList)", "median(myBag)"})
public class MyModel extends Simulation {
  @RecorderSource("someValue")
  protected double[] someField;

  @RecorderSource(value="myClassList", member="myMethod()")
  protected List<MyClass> myClassList = new ArrayList<MyClass>();
```

```
  @RecorderSource(value="myBag", innerType=MyClass.class,
      member="myMethod()")
  protected Bag myBag = new Bag();

  @RecorderSource("someOtherValue")
  protected List<Double> someMethod(){
    ...
  }
}

public class MyClass {

  protected int myValue;

  public int myMethod(){
    return myValue;
  }
}
```

Once a collection or array member is annotated as data source, the second part of recording statistics is to specify the kind of statistics to be applied. Statistics are specified in the `sources` attribute of the `@Recorder` annotation. Instead of simply putting data source names into the list, you can surround the names with a statistical *function*, as seen above. The currently available statistics are the following.

| | |
|---|---|
| mean(), avg(): | the average of the provided values |
| min(): | the minimum of the provided values |
| max(): | the maximum of the provided values |
| sum(): | the sum of the provided values |
| kurtosis(): | the kurtosis ("peakedness") of the provided values |
| median(): | the median of the provided values |
| product(): | the product of the provided values |
| variance(): | the variance of the provided values |
| sd(): | the standard deviation in the provided values |
| skew(): | the skew of the provided values |
| se(), stde(): | the standard error in the provided values |

These statistics are computed using the Colt[1] software library. Not all, in Colt available, statistics have been integrated into the framework yet. Nevertheless, it is easy to include any of the static methods from the `cert.jet.stat.Descriptive` class, when requested.

[1] `http://acs.lbl.gov/software/colt/`

The specified statistics take the set of values provided by the data source as input, and output a single double value. When recording happens, this single double value gets recorded. In addition to channel a set of values into statistical functions, it is also possible to record the collections or arrays directly. This feature is explained in the next section.

### 2.3.5 Recording collections of simple values

When collections or arrays are recorded instead of scalar values, a fixed number of columns are generated in the results file for one collection, instead of the otherwise usual single column. Recording collections of simple values is rather similar to recording statistics, with a few notable differences. First, in the `sources` attribute of the `@Recorder` annotation, you simply have to specify the name of the data source, without any function. Second, you have to tell the framework the length of the collection to be recorded. Finally, you can specify a default value to be recorded if the collection is not of the specified length (using the `NAFiller attribute`). This value defaults to "`NaN`", which can be parsed as a `double`.

To define the length of a collection to be recorded, you can either specify a fixed number or a member of the model class that should be accessed to retrieve a number. To provide a fix number, you can use the `collectionLength` attribute of the `@RecorderSource` annotation, as shown on line 3 in Listing 2.5. To specify a model member, you can use the `collectionLengthMember` attribute (see line 6). If the provided string ends in "`()`", then the named method is invoked to retrieve the length of the collection. Otherwise, a field with the given name is searched for the value.

An important thing to keep in mind when specifying a model member as `collectionLengthMember` is that its value is evaluated only once during a simulation run, at the first time recording happens. The retrieved number is used to generate the columns in the results file. If the collection or array has less elements than this number, then the `NAFiller` value is used to fill the empty columns. If the collection or array contains more elements than this value, then elements with an index larger than this value are omitted.

Listing 2.5: Recording collecions of simple values

```
@Recorder(value="results.txt",
    recordAt=RecordTime.END_OF_ITERATION, sources={"someValue",
    "someOtherValue"})
public class MyModel extends Simulation {
  @RecorderSource(value="someValue", collectionLength=10)
  protected double[] someField = new double[10];

  @RecorderSource(value="someOtherValue",
      collectionLengthMember="someOtherMethod()", NAFiller="-1")
  protected List<Integer> someMethod(){
    ...
  }
```

```
  protected int someOtherMethod(){
    ...
  }
}
```

If more than one run is performed (e.g. in a parameter-sweep experiment), the length of collections and arrays are evaluated in each run. If any of the lengths change between two runs, the actual result file is closed and a new file is started. This can lead to multiple results file, with the first file having the original name as given in the `@Recorder` annotation, and the rest having a name derived from this name by inserting a '`-partX`' fragment right before the file extension, where `X` is a number incremented for each new file. If multiple files have been generated, each file will have a different number of columns. Note, that it is possible that one result file contains values recorded in multiple different runs. A new result file is started only if the length of a recorded collection or array changes.

The names (headers) of the columns belonging to a collection are derived from the data source name as defined in the `@RecorderSource` annotation. That name is appended by the string '`Multi_X`', where '`X`' is the index of the element in the collection or array. For instance, the numbers in the member `someField` in Listings 2.5, are recorded in columns with header "`someValueMulti_0|someValueMulti_1|...`".

In addition to recording collections or arrays of simple values, it is, of course, possible to record values from collections containing arbitrary classes. We have already seen in Listing 2.4, how one can specify the class of the elements in the collection, and the member of that class that should be invoked to retrieve the value. In the next section, we introduce an alternative option, where the member of the element class is automatically reflected by the framework.

### 2.3.6 Recording collections of classes

It is possible to define a collection or array as a data source without fully specifying where the data comes from. In Listing 2.6, the `myBag` member is annotated as data source under the name "`myBag`". It is specified that it contains objects of `MyClass` type, but the member to invoke to retrieve the value to be recorded is unspecified. In such cases, the framework inspects the `MyClass` class for members annotated with the `@RecorderSource` annotation. Any such member is found, it is enlisted as a possible data source. Which one is chosen depends on what is requested in the `sources` attribute of the `@Recorder` annotation. To select the appropriate member, the framework tries to match the requested data-source name and the concatenation of the data-source names of the collection and the `MyClass` members (in this order). In the example below, "`myBagValue`" is requested in the `sources` attribute, therefore in the bag annotated as "`myBag`" a `MyClass` member annotated as "`Value`" is selected.

Listing 2.6: Recording collecions of classes

```
@Recorder(value="results.txt",
    recordAt=RecordTime.END_OF_ITERATION, sources={"myBagValue"})
```

```
public class MyModel extends Simulation {
  @RecorderSource(value="myBag", innerType=MyClass.class)
  protected Bag myBag = new Bag();
  ...
}

public class MyClass {

  protected int myValue;

  @RecorderSource("Value")
  public int myMethod(){
    return myValue;
  }
}
```

This is a very flexible approach, as it allows for the reuse of `@RecorderSource` annotations in classes. It is possible to annotate any method in any class as data source, and depending on the `@Recorder` annotation, some may or may not be used, or even used multiple times (if the same class is used in multiple collections)!

### 2.3.7 Summary

The recording framework in the CRISIS Software Library was designed to provide a declarative way of defining what should be recorded during simulation runs. Introducing two annotation types, it allows recording any number, string or enum type of data from the model or from collections or arrays of the model. It can be configured to record per iteration, per run or when a condition evaluates to true, and it can record scalar values, collections, or statistics computed on a collection of values. This is a flexible and simple to use framework that can greatly simplify the task of recording.

## 2.4 Genetic Search

The Dashboard application is capable of performing a search over the parameter space of a model by utilizing a genetic algorithm (GA). The GA search functionality is implemented by integrating JGAP, a Java Genetic Algorithms Package[2]. To run a GA search in the Dashboard, after a model has been selected (on the first screen), the user can navigate to the 'Genetic Algotihm Search' tab, where the GA can be set up. In the following, we discuss the contents of this screen, and provide details on how the GA search is performed.

### 2.4.1 Chromosomes

The GA operates on *chromosomes* that consist of a fixed number of *genes*. In our ABM simulation application, a gene is a parameter of the selected simulation model. A chromosome is a set of genes that are selected by the user to

[2] `http://jgap.sourceforge.net/`

participate in the search. To designate a model parameter as a gene, the user has to define a range of possible values for that parameter. This can be done on the GA search screen by selecting the desired parameter in the list titled as 'Chromosome' in the middle of the screen. Once a parameter is selected, the allowed range can be configured on the right-hand side of the screen.

During the search, the application maintains a population of chromosomes. Each chromosome can be thought of as concrete a parameter-configuration of the simulation model, in which each gene has a concrete value (*allele*). To evaluate a chromosome, the application, indeed, runs the model with the parameter configuration as defined by the genes. The GA handles the simulation as a black box. It feeds in the parameters as prescribed by the genes, and it receives a single output, which is the fitness of the corresponding chromosome.v

From generation to generation, the algorithm repeats the following steps.

1. *Evaluation.* It runs the simulation with each configuration in the population (each chromosome) one-by-one. This provides the fitness values of the chromosomes used in the selection step.

2. *Selection.* It applies the configured selection algorithms, which results in a new population with selected chromosomes.

3. *Recombination.* It applies the configured genetic operators on the new population. Each operator creates new chromosomes, offsprings of the selected ones, which are added to the new population.

4. *Housekeeping.* If the new population contains more chromosomes as configured for a population, then chromosomes from the old generation are discarded as needed. If the new population contains less chromosomes as configured, then random chromosomes are added as needed.

The above steps are executed repeatedly until the stopping criterion is satisfied. The whole process is initialized with a random population, that is a population of chromosomes, in which genes have random values within the specified ranges.

### 2.4.2 GA configuration

Different aspects of the GA can be configured on the left-hand side of the GA search screen. On the top, the number of turns each simulation should run through can be defined in the category of 'General parameters'. This setting is analogous to the 'Maximum number of time steps' settings of the single-run or parameter-sweep screens. Below, the first setting specific to the GA search controls the populations of chromosomes during the search. The user can specify the size of the populations, as well as a random seed that is used to initialize the random-number generator applied when random chromosomes are created. The next box configures the stopping criterion of the search. The user can select between a fixed number of generations and a fitness limit that is used as a threshold of the fitness. Then, the fitness function can be selected from a list. This list is pre-populated from the list of scalar (double or int/long) recorded values. That is, the user can select only between values that are recorded in the simulation, excluding multi-column recordings. This means, that the objective

function must be implemented in the model, and its value should be recorded in order to govern the search. It is important, that this function should never return a negative number! In addition to selecting the objective function, the user can define the direction of the search as minimize or maximize. Finally, in the left-bottom corner of the screen, the user can select and configure the desired selection algorithms and genetic operators.

### 2.4.3 Selection operators

Currently, there are three selection operators implemented in the application. These are provided by JGAP, and integrated into the Dashboard. It is possible, however, to extend the list of selection operators by implementing a few simple classes. How this can be done is out of the scope of this document. In the following we describe the functionality of the three selection operators already present.

1. **The best chromosome selector** selects the best $n$ chromosomes from the old generation into the new generation. $n$ can be specified as a ratio: the 'rate of original population to survive'.

2. **The weighted roulette selector** models a roulette wheel. When a chromosome is added, it gets a number of 'slots' on the wheel equal to its fitness value.When the operator is invoked, the wheel is 'spun' and chromosome occupying the spot on which it lands is selected. Then the wheel is spun again and again until the requested number of chromosomes have been selected. Since chromosomes with higher fitness values get more slots on the wheel, there is a higher statistical probability that they will be chosen, but it is not guaranteed.

3. **The tournament selector** plays tournaments to determine the chromosomes to be taken to the next generation. The tournament size can be adjusted as well as the probability for selecting an individual.

It is possible to configure multiple selection operators to be applied. Each selection operator is invoked with a target number of chromosomes to generate. If there is only one selection operator, this target is the population size. If there are multiple operators configured, the target number is divided by their number.

After the selection operators are applied, the new generation is taken to be the parents of the new chromosomes generated in the next step.

### 2.4.4 Genetic operators

Currently, there are three genetic operators in the application. Similarly to the selection operators, it is possible to add new operators by creating a few simple classes. The currently available operators are as follows.

1. **Gene averaging crossover** an operator that randomly selects two chromosomes from the population and 'mates' them by averaging the values of the numerical genes. The chromosome with the average gene values is then added to the list of candidate chromosomes.

2. **Crossover** an operator that randomly selects two chromosomes and 'mates' them by randomly picking a gene and then swapping that gene and all subsequent genes between the two chromosomes. The two modified chromosomes are then added to the list of candidate chromosomes.

3. **Mutation** an operator that runs through the genes in each of the chromosomes in the population and mutates them in statistical accordance tot he given mutation rate. Mutated chromosomes are then added to the list of candidate chromosomes.

All genetic operators are applied on the population generated by the selectors. They work independently of each other, the order in which they are applied is irrelevant. After all configured operators are applied, the chromosomes they created are added to the new generation created by the selectors. If the size of this new population is not as configured, it is adjusted either by removing 'old' chromosomes, survivors of the previous generations, from the new population, or by adding new randomly generated chromosomes.

### 2.4.5 Summary

The genetic algorithm functionality of the Dashboard is capable of running a GA search over the defined parameter space of the selected simulation model. The GA search GUI allows the user to configure the search, define the genes, and the genetic operators. Once all set up, the Dashboard runs the GA search which includes repeatedly running batches of simulations. The progress bar on the 'Run' screen has been extended by an iteration counter that allows the user to monitor the process. After the GA stopped, the user is presented a table where each row describes a concrete chromosome. The columns of the table are the genes of the chromosomes (the parameters) and the fitness value. This table details the fitness of each chromosome in each generation. The chromosome with the best fitness value is highlighted in the table, and a button is presented to the user that starts the simulation in single-run mode configured with parameter values corresponding to the winning chromosome. This allows the user to study the behavior of the model in the 'optimal' situation, and to observe the recorded results. In addition, the table presenting the chromosome values is also written to a file ('population.txt') withing the Dashboard's workspace, in a folder corresponding to the simulation model and the time the simulation was started.

## 2.5 Dashboards

The dashboard is a frontend desktop-based GUI application for the simulator. The dashboard can be launched by running `Dashboard.java` as a Java application (Java 6 or above is assumed). The dashboard:

1. enables the user to select from a variety of preconfigured or partly configured customizable models;

2. explore these models before launching a simulation;

3. run a customised simulation (the simulator is capable of running models without the aid of the dashboard application, however the dashboard application provides user interface controls to do so);

4. displays a progress bar indicating the completeness of the running simulation;

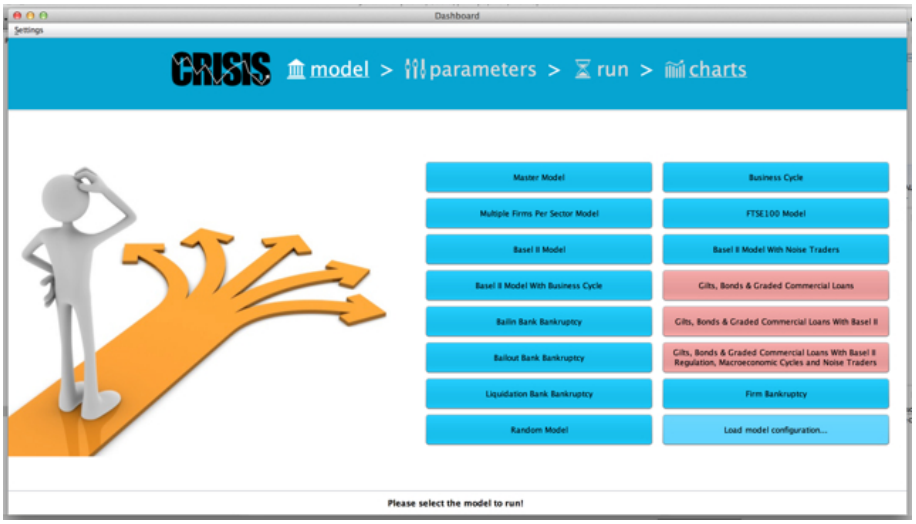
5. provides a set of interactive charts to explore results when the simulation is complete.

Other facets of the dashboard application include simplifying the collection of recorder source data (§2.3) and integrated support for parameter search programs (§2.4).

The dashboard requires a simulation model to be described in terms of a hierarchical tree structure of `submodels`. `submodels` are the components of a simulation that can be configured independently: for instance, the subeconomies of the standard model (§2.2.1) are `submodels` according to the requirements of the dashboard.

When `Dashboard.java` runs with the standard model, the user is displayed with a selection of preconfigured or partly configured models.

At the time of writing the appearance of the dashboard front page is as follows:




The buttons appearing on the right hand side are entry points for the following models:

1. **Business Cycle**

   This is a model with an endogenous macroeconomic business cycle. The business cycle affects bank leverages, production, household consumption and stock prices among other timeseries. This model does not use Basel II financial regulation. The business cycle is as a result of firm and household decision rules.

2. **Multiple Firms Per Sector**

   This model supplements the default model configuration with 2+ competing firm agents per sector (namely, two or more firm agents producing goods differentiated only by selling price).

3. **FTSE1000 Model**

   This model uses real FTSE100 data to initialize agents. This includes the names of the agents, their stock emission values, liquidity targets, sectors and other initial states. This model runs but at the time of writing some assumptions made about household consumption in particular are not entirely compatible with real UK companies.

4. **Basel II Model**

   The Basel II model is a default model configuration supplemented with Basel II financial regulation (commercial bank leverage targets).

5. **Basel II Model With Noise Traders**

   This model supplements the Basel II model with a customizable number of mean-reverting noise trader funds (NTFs). By default there are 4 NTFs but this number can be changed. The aggressiveness of the fund stock investment decision noise can also be customized. If the noise in the stock investment system is increased to very high levels, the financial leverage cycle is suppressed.

6. **Basel II Model With Business Cycle**

   This model is a combination of the procyclical leverage features and the macroeconomic business cycle.

7. **Bailin/Bailout/Liquidation bank bankruptcy**

   These models create an artificial bank bankruptcy event. The method used to create the bankruptcy can be specified (eg. a substantial commercial loan default episode or a stock market price shock/equity exposure incident for banks). You can also specify the number of banks to be affected. Bank liquidation appears to work, however the underlying implementation of commercial bank liquidation is likely to change in the near future.

8. **Firm bankruptcy**

   Firm bankruptcies will be caused artificially at some time during the simulation. The severity and nature of the firm bankruptcy event is customizable.

9. **Gilts, Bonds and Graded Commercial Loans**

   This is a model running the GBCl suite. The GBCl is a clearing market over which commercial banks attempt to maximize their expected returns in risk-graded commercial loans and gilts. Cash flows from mutual funds to banks via bonds. The banks consider the market impact and risk of investing in any particular instrument (eg. medium risk commercial loans) when maximizing their portfolios. This has been optimized to run at an acceptable speed and is now quite competitive with non-BCL settings.

10. **Gilts, Bonds and Graded Commercial Loans With Basel II**

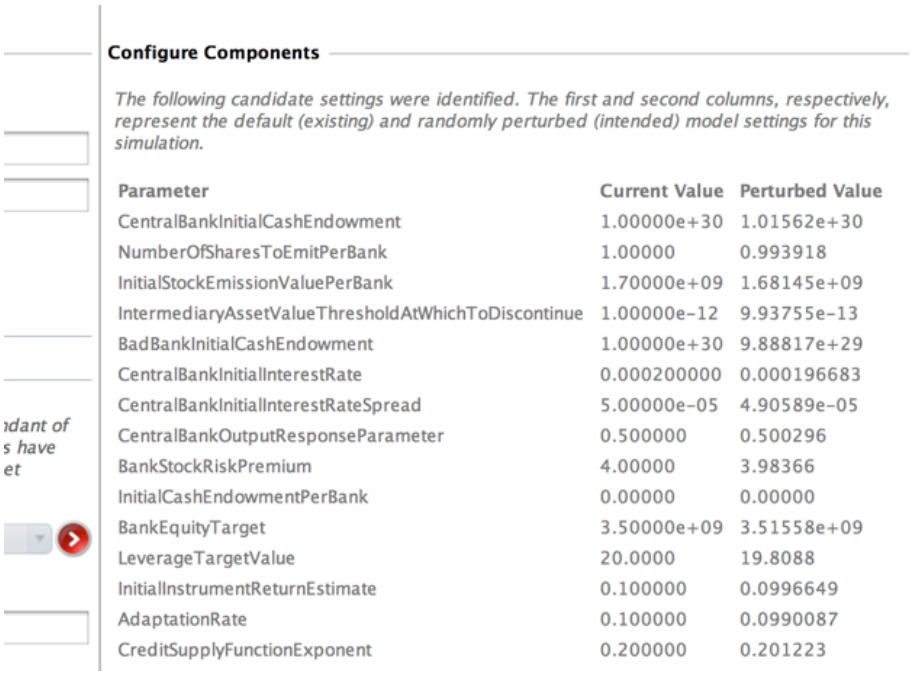
    This model is a combination of the GBCl and Basel II. Heterogeneous clearing is compatible with a financial leverage cycle.

11. **Gilts, Bonds and Graded Commercial Loans With Basel II and Noise Traders**

    This model is as above, supplemented with a customizable number of noise-trader funds.

12. **Random Model**

    'Random Model' is a special type of model that perturbs all of the default model parameters and then runs a simulation using these perturbed parameters. This model searches the entire simulation configuration for parameters to perturb and lists all such parameters (potentially hundreds) on the dashboard:



**Configure Components**

*The following candidate settings were identified. The first and second columns, respectively, represent the default (existing) and randomly perturbed (intended) model settings for this simulation.*

| Parameter | Current Value | Perturbed Value |
|---|---|---|
| CentralBankInitialCashEndowment | 1.00000e+30 | 1.01562e+30 |
| NumberOfSharesToEmitPerBank | 1.00000 | 0.993918 |
| InitialStockEmissionValuePerBank | 1.70000e+09 | 1.68145e+09 |
| IntermediaryAssetValueThresholdAtWhichToDiscontinue | 1.00000e-12 | 9.93755e-13 |
| BadBankInitialCashEndowment | 1.00000e+30 | 9.88817e+29 |
| CentralBankInitialInterestRate | 0.000200000 | 0.000196683 |
| CentralBankInitialInterestRateSpread | 5.00000e-05 | 4.90589e-05 |
| CentralBankOutputResponseParameter | 0.500000 | 0.500296 |
| BankStockRiskPremium | 4.00000 | 3.98366 |
| InitialCashEndowmentPerBank | 0.00000 | 0.00000 |
| BankEquityTarget | 3.50000e+09 | 3.51558e+09 |
| LeverageTargetValue | 20.0000 | 19.8088 |
| InitialInstrumentReturnEstimate | 0.100000 | 0.0996649 |
| AdaptationRate | 0.100000 | 0.0990087 |
| CreditSupplyFunctionExponent | 0.200000 | 0.201223 |

13. **Master Model**

    This is the backbone from which all other states of the standard model currently specialize. With effort this model can be configured to behave as any of the other models listed above.

It is a straightforward exercise to add new models to the dashboard.

# 3 Background and Mark One

The standard model is borne out of a simple Cobb-Douglas economy called the 'mark one' (M1) legacy model. The M1 is simple and follows a structure outlined by Adrian and Boyarchenko (2012). The M1 real economy consists of a productive sector populated by firms who employ capital to produce a single

undifferentiated nondurable consumption good. The real economy is coupled to a financial system consisting of financial intermediaries investing in stocks and commercial loans. Despite being highly stylised and partly exogenous, to some extent the M1 model allowed us to quantify production activities among firms as well the resulting dividend stream. The original M1 model had a variety of shortcomings. The M1 economy contained no model for labour and capital was an abstract quantity; prices of consumer and capital goods were not simulated; household decision making processes were simplified or not present, and stocks were transient assets destroyed at the end of each timestep.

## 3.1 Productivity Fluctuations

Important to the M1 legacy model and its descendants, including the standard model, is the concept of productivity fluctuations among firms. The general form for the firm production function in the standard model is:

$$Y = Z \cdot f(L, g_1, g_2, ...) \quad (*)$$

Where $Y$ is the yield (production) of new goods, $Z$ is the Total Factor Productivity (the TFP factor, a multiplier) and $f$ is a function combining labour inputs $L$ and other goods $g_i$ in some way (subscripts $i$ denote different types of good).

Productivity fluctuations are typically implemented as $Z = Z(\lfloor t \rfloor)$ where $t$ is the simulation time and $\lfloor\rfloor$ is the floor function. Ie. $Z$ is a function of the simulation cycle index. A simple implementation of $(*)$ is the 'labour–only' production function $Y = Z \cdot L^{\alpha}$ where the exponent $\alpha$ of the production function is defined over the interval $0 \leq \alpha \leq 1$. The process $Z(t)$ may be implemented as:

1. A mean–reverting random walk of the form:

   $$Z(t) = \rho \cdot Z(t-1) + \rho \cdot (\mu - Z(t-1)) + \sigma \cdot \epsilon_t$$

   where $\rho$ the mean–reversion rate, $\mu$ is the mean–reversion level (the long-term mean sample value $\mathbb{E}[Z]$), $\sigma$ is the standard deviation (the volatility) of noise, and $\epsilon_t$ is a random shock;

2. A lognormal mean reverting random walk;

3. An autoregressive (discrete Orstein-Uhlenbeck) process of the form:

   $$Z(t) = \rho \cdot Z(t-1) + \mathcal{N}(\mu, \sigma)$$

   where $\mathcal{N}(\mu, \sigma)$ is a normal distribution with mean $\mu$ and variance $\sigma^2$ and $0 \leq \rho \leq 1$ is a customizable parameter. Note that the true mean $\mathbb{E}[Z]$ of this process is $\mu/(1-\rho)$ and the true volatility is $\sigma(Z) = \sigma/\sqrt{2-2\rho}$

4. A lognormal transformation of (3);

5. Any of the above including explicit range saturation:

   $$Z(t) = \min(\max(Z^*(t), Z_{\min}), Z_{\max})$$

   where $Z^*$ is one of the above series, $Z_{\min}$ is a customizable lower bound and $Z_{\max}$ is a customizable upper bound;

or virtually any other random process. Note that productivity fluctuations can be disabled entirely, such that $Z(t) \sim$ const. or $Z(t) = g(t)$ for an arbitrary closed-form function $g(t)$.

## 3.2 Production

The M1 real economy consists of a set of firms. Firms are characterized by their capital stock $K(t)$, equity $E(t)$ and production function $Y(t)$. Firms are myopic profit maximisers who solve:

$$\max_{K(t+1)} \mathbb{E}\left[\text{profit at time } t+1\right]$$

Firms have a labour-only production function depending on capital:

$$Y(K) = Z(t)K(t)^{\alpha},$$

The M1 model assumes that the firm sells all its production at a fixed price $p = 1$. There are no unsold goods. The capital the firm employs for production is subject to depreciation and investment according to:

$$\Delta K = K(t+1) - K(t) = -dK(t) + I(t),$$

where $d < 1$ is depreciation. One unit of cash $I$ invested gives the firm one unit of capital $K$. Capital becomes available for production in the same period as investment. Models with a stronger focus on modelling the real economy eg. Dosi et al. (2010) or Dawid et al. (2011) tend to introduce delays between investment and the availability of capital.

## 3.3 Credit demand

The M1 model assumes investment is financed exclusively by loans. In particular:

$$\text{profit at time } t+1 = \mathbb{E}\left[A(t+1)\right]\left((1-d)K_{i,t} + I_{i,t+1}\right)^{\alpha} - C,$$

where $C$ represents outgoings. The future profit maximisation condition is:

$$\mathbb{E}\left[A(t+1)\right]\alpha\left((1-d)K(t) + I(t+1)\right)^{\alpha-1} - \frac{\partial C}{\partial I(t+1)} = 0$$

In the M1 model commercial loans are classified by their maturity $\tau$ and interest rate $r$ as fixed rate mortgages:

$$C = \frac{rI}{1-(1+r)^{-\tau}}$$

Consequently the firm optimal investment $I^*$ is:

$$I(t+1)^* = \left(\frac{\mathbb{E}\left[r(t+1)\right]}{\mathbb{E}\left[A(t+1)\right]\alpha\left(1-\left(1+\mathbb{E}\left[r(t+1)\right]\right)^{-\tau}\right)}\right)^{\frac{1}{\alpha-1}} - (1-d)K(t).$$

The number $I^*$ if the firm credit demand. Note that $I^* = I^*(r)$ and the above working does not specify how $r$ should be calculated.

### 3.4 M1 Simplifying Assumptions

The M1 model assumed that all firm profits were paid to shareholders as dividends. Profits generated in a given period are paid out in the same period as dividends. The maturity of loans is $\tau = 1$ for all firms, interest rates are homogenous (undifferentiated), and $d = 1$. These simplifications made the analysis easier at the expense of removing some potentially interesting features. In particular the assumption ($\tau = 1, d = 1$) removes all path dependence from the firm dynamics except for the autocorrelations in the stochastic productivity of capital. In reality capital adjustment is a slow process introducing strong persistence in firm output. By making the above simplifications we abstract from this effect.

Firms in the M1 model ought to face two types of uncertainty: uncertainty about $Z(t+1)$ and uncertainty about $r(t+1)$. The M1 model circumvents both uncertainties because firms are told of $Z(t+1)$ in advance and $r$ is a homogeneous market clearing variable. The market mechanism will be explained in more detail in section 5.8. The M1 model contains no price or demand risk and no inventory costs. Interest rate risk may reemerge should the M1 model be implemented with an alternative market eg. an LoB market.

Allowing only one interest rate per firm can potentially be justified when credit risk among firms is very low. Given the above simplifications (including no unsold goods) credit risk is low for all firms. Firm bankruptcy is considered to be meaningless in the M1 model.

## 4 Standard Model and the IO Economy

### 4.1 Introduction

The Input-Output macroeconomy is based on the model described in *Deliverable 3.3*. This IO structure has been integrated with the financial agent based model and has subsequently grown.

Some characteristics of the financial submodel and the macroeconomy were modified so that their key features could be reconciled into the integrated Java framework. There are a variety of differences including:

1. Firm profitability is based on sales as well as access to credit. Firm sales are subject to no guarantee. Firms can become insolvent as a result of failing to sell goods. A mechanism must be introduced to resolve firm bankruptcies.

2. Fundamentally, the integrated macro model has a financial system whereas the standalone macro model does not. Firms may decide whether to fund production from internal cash reserves or new commercial loans or both. It is not necessarily the case that the quantity of goods produced by firms in the integrated simulator is the same as the standalone version.

3. Firms select their own sales prices. Firms should expect to sell fewer goods if their products are expensive.

4. The possibility of firm insolvency entails the possibility of bank bankruptcy as a result of commercial loan default. A mechanism must be provided to resolve bank insolvency.

The following section §4.2 illustrates the synthetic model structure. §4.3 illustrates the conceptual structure of the standard model in which virtually every subcomponent is realised as a model parameter with a customizable value. Section §4.4 focuses on the canonical implementation of firm agents (macrofirms). In the section §4.6 the basic worker/consumer behaviour is outlined. §4.4.6 outlines the credit demand and commercial loan processing subsystem. We also outline the failure conditions for firms in §4.4.7 and accordingly the firm bankruptcy resolution procedure.

## 4.2 The Model

The macro model is populated by firms and consumers/workers.

Consumers have a CES utility function (see section **??**), their nominal wage is constant and they supply inelastically one unit of labour in each period. In fact, the behaviour of households is kept very simple since the analysis is mostly concerned with the effect of firms' behaviour.

Firms produce perishable differentiated goods through a Cobb-Douglas production function (see section **??**), using intermediate goods and labour as inputs.

They have limited market power and decide price and quantity using adaptive heuristics. To avoid simultaneity problems, we assume that firms can sell their production in time $t$ only in time $t + 1$. Therefore, in any period, the supply of good is fixed and depend on previous periods decisions.

When the period starts, each firm has a quantity supplied on the market and a price at which they wish to sell their products. Given the amount sold in the last period, firms decide the quantity to be produced and given the prices of the goods on the market, they choose the combination of inputs to be used and compute the relative amount of liquidity needed for production, i.e the liquidity needed by the firm to buy on the markets the inputs required for production. The difference between the needed liquidity and the available internal liquidity determines the credit demand (see section **??**).

The financial system and the macroeconomic system interact constantly since there is a flow of money continuously moving from one system to the other. In particular, the financial system is crucial to provide liquidity to the production sector through loans and to redistribute profits through the dividends.

## 4.3 Model Parameters

### 4.3.1 General Parameters

Broadly speaking a 'model parameter' is a customizable argument or field describing the behaviour of agents, markets, or other structures in the simulation

world. Model parameters have values and these values may or may not be constants, numerical primitives, strings or more complex layered objects.

The original M1 model §3.4 was subject to configuration challenges as many model parameters were hardcoded, fixed and unmodifiable, not documented as explicit dependencies, or otherwise inaccessible or difficult to access when specifying the model. This has been revised extensively by overhauling the standard model in such a way that the majority of model parameters are listed in object class constructors. This formulation allows a technique called dependency injection (DI) to be used to interchange model parameters and specify alternative implementations with relative ease.

Dependency injection is currently provided as a framework via the Google Guice library. This approach was especially compatible with the limitations and requirements of the dashboard GUI: the dashboard requires java classes specifying the 'submodel' or hierarchical 'nested model' structure of the model[3] in a way that is compatible with Guice submodules.

Conceptually model parameters are divided into two use cases as follows:

1. Model parameters $P(i)$ whose values depend on the index $i$ of the sample taken from them (which we refer to as *sample parameters*).
2. Model parameters $P(t)$ depending on the simulation time $t$ (which we refer to as *timeseries parameters*). Timeseries parameters include parameters whose values depend on the simulation cycle index $\lfloor t \rfloor$: $P(\lfloor t \rfloor)$.

Concrete examples of model parameters could be:

1. The initial cash endowment $C$ of each newly created macrohousehold agent is a sampled model parameter. $C$ has meaning only when a new household agent is created. If all households are created at time $t = 0$ then $C$ is not required or referenced for any subsequent $t > 0$. The initial cash endowment may however differ among households: $C(i)$ is potentially heterogeneous depending on the index $i$ of the household created.
2. Similarly, assuming all firms are instantiated at $t = 0$, the initial stock market capitalisation (stock emission value) $S$ of each firm is not referenced for any $t > 0$. Perhaps the initial firm stock capitalisation data is drawn from FTSE100 reference tables. In this context $S = S(\eta)$ is a model parameter depending on the name of the firm, $\eta$, where $\eta$ is drawn from an array of FTSE100 firm names and $P = S$ is a model parameter underwritten by an indexed column of empirical data.
3. By default the household labour supply is not a function of time, $L \neq L(t)$ in the standard model. However it may be the case that household agents are initialised with heterogeneous labour supplies. For each household $L$ may be drawn independently $L \sim \mathcal{N}(\mu, \sigma)$ form a normal distribution with mean $\mu$ and volatility $\sigma$.

[3] A 'submodel' in the context of the dashboard is a set of parameters existing in drilldown later of a tree of hierarchical parameters

4. It is desirable for firm Total Factor Productivity $Z$ to be subject to shocks and random fluctuations. $Z(t)$ can be realised as a timeseries parameter whose initial value is randomly generated for each firm ($Z(0) \sim \mathcal{N}(\mu, \sigma)$). $Z(t)$ must evolve in time as a random series. $Z(t)$ may be implemented as a normal or lognormal random walk or a discrete AR1 Ornstein-Uhlenbeck process according to the prescriptions in §3.1.

5. Firms require a characteristic production function $Y$ as described in §3.1. $Y$ may differ among firms. In a production chain model wholesaler firms who sell goods to other firms but not to households may require a variety of input goods from other firms but relatively little labour. Contrariwise root producers (eg. quarrying firms) may require comparatively huge amounts of labour and no or few input goods. In this context, it may be appropriate to model a root producer with a production function $Y \sim Z(t)L^{\alpha}$ where $L$ is labour and to model a retailer with a Cobb-Douglas IO or Leontief IO production function. $P = Y$ is a sampled model parameter whose values do not change in time. At initialisation, each firm receives an implementation of $Y$ and $Y$ may be selected differently for each firm.

6. In an industrial subeconomy the implementation of the firm agent is interchangeable subject to retributions imposed by the firm interface from which all firm agent implementations descend. It is therefor possible to produce new implementations of the firm agent types with entirely customizable behaviour. As long as all implementations of the firm confirm to a set of standards outlined by the Java interfaces, these firm implementations are largely interchangeable. In this context the agent type used to populate an industrial subeconomy is a sampled model parameter $F$ whose values are firms.

7. In the standard model it is expected that banks can be bankrupted. The scheme used to resolve bank bankruptcies is an ordered chain of candidate bankruptcy resolution techniques $R_1$, $R_2$... such that resolution $R_1$ is applied initially, and, should $R_1$ fail, then resolution $R_2$ is applied, and so on. This chain terminates with an infallible technique $R = \{$ liquidate the bank $\}$. The resolution chain $\{R_1, R_2 \ldots\}$ is a model parameter, invariant in time, whose value depends on the commercial bank to which the resolution chain is assigned. The structure of the resolution chain need not be the same for each commercial bank.

8. The central bank repo loan refinancing decision is a yes/no question the central bank must decide whenever a commercial bank applies for a new repo loan. It may be the case the the central bank declines a new repo loan if the applicant does not satisfy certain thresholds, for instance a minimum capital adequacy ratio. There is only one central bank agent, and this yes/no decision rule is an object whose value is interchangeable with other implementations. The central bank refinancing decision rule is accordingly a sampled model parameter.

#### 4.3.2 Numerical Model Parameters

Numerical model parameters (model parameters $P$ whose values are integers, double, long etc.) differ slightly from general non-numerical model parameter

(object-valued model parameters) because numerical model parameters typically define timeseries with values $P(0), P(1)...$ and so on. The simulator includes a set of stock numerical model parameter implementations including:

1. **Fixed Value Parameters**
   Fixed-value numerical parameters are parameters $P$ whose values depend on nothing. The result $P(t) = x$ is returned every time $P$ is evaluated, where $x$ is a customisable constant, no matter what.

2. **Exponential Distribution**
   The exponential distribution is the probability distribution that describes the time between events in a Poisson process, i.e. a process in which events occur continuously and independently at a constant average rate. It is the continuous analogue of the geometric distribution, and it has the key property of being memoryless. In addition to being used for the analysis of Poisson processes, it is found in various other contexts. See `en.wikipedia.org/wiki/Exponential_distribution`.

   The probability density $\mathbb{P}$ of the exponential distribution is:

   $$\mathbb{P}(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases},$$

   where $\lambda$ is customizable. The value of $P$ is additionally clamped in the range $a \leq P \leq b$ where $a$, $b$ ($b \geq a$) are customizable parameters (which may take the values $\pm\infty$).

3. **Data Column**
   A model parameter whose numerical values are specified explicitly by rows in a text file. This implementations reads a column of numerical data from a file. The file contains all possible values, in sequence, for the parameter $P$. The file should be a single-column data table with exactly one record per line.

4. **Expression**
   It is possible to specify a model parameter as a closed mathematical expression. The value of $P$ is delegated to a mathematical expression $S$. Provided $S$ can be parsed as a univariate function of an argument $X$, then the values returned by $P$ are $S(X = 0), S(X = 1)...$ in that order.

5. **Lognormal Distribution**
   A lognormal distribution is a continuous probability distribution of a random variable whose logarithm is normally distributed. Thus, if the random variable $X$ is log-normally distributed, then $Y = \log(X)$ has a normal distribution. Likewise, if $Y$ has a normal distribution, then $X = \exp(Y)$ has a log-normal distribution. A random variable which is log-normally distributed takes only positive real values. See `en.wikipedia.org/wiki/Log-normal_distribution`.

6. **Gaussian Distribution**
   The values of $P$ are drawn independently from a normal distribution

$\mathcal{N}(\mu, \sigma)$. The probability density function for $P = P(\mu, \sigma)$ is:

$$\mathbb{P}\{P(\mu, \sigma) = x\} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

7. **Uniform Distribution**
   The values of $P$ are drawn independently from a uniform distribution $U[a, b]$ $(b \geq a)$. The probability density function for $P = P(a, b)$ is:

$$\mathbb{P}\{P(a, b) = x\} = \{\frac{1}{b-a} \text{ if x } \in [a, b],\ 0 \text{ otherwise}\}.$$

8. **Random Series**
   $P$ can alternatively be implemented as a random series Ie. successive samples $P(0), P(1)...$ conform to the specifications of a stochastic random process. This random series can take many possible forms, including but not limited to:

   (a) **A (Discrete) Orstein–Uhlenbeck Process**
   Samples $P(i)$ of this model parameter obey:

$$P(i+1) = \rho \cdot P(i) + \mathcal{N}(\mu, \sigma),$$

   where $\mathcal{N}(\mu, \sigma)$ is sampled from a Gaussian distribution with customizable standard deviation $\sigma$ and mean $\mu$. The expected value $\mathbb{E}[P]$ of this process satisfies:

$$\mathbb{E}[P] = \frac{\mu}{(1-\rho)}.$$

   The standard deviation $\sigma(P)$ of this process satisfies:

$$\sigma(P) = \frac{\sigma}{\sqrt{2 \cdot (1-\rho)}}.$$

   (b) **Mean–Reverting Random Walk**
   An implementation for which the value $V$ of the parameter $P$ evolves as follows:

$$V_{\text{next}} = V_{\text{last}} + a \cdot (V^* - V_{\text{last}}) + \sigma \cdot \epsilon_t,$$

   where:

   i. $a$ is the mean reversion rate;
   ii. $V^*$ is the mean reversion level (the long-run mean sample value);
   iii. $s$ is the standard deviation (the volatility) of noise;
   iv. $\epsilon_t$ is a random shock, and
   v. $V_{\text{next}}$ is the next value of the parameter.

   The initial value of this parameter is $P = V^*$.

   (c) Virtually any other random series.

9. **Beta Distribution**
   The beta distribution is a family of continuous probability distributions defined on the interval $[0, 1]$ parametrized by two positive shape parameters, denoted by $\alpha$ and $\beta$, that appear as exponents of the random variable and control the shape of the distribution. See `en.wikipedia.org/wiki/Beta_distribution`. This implementation may be linearly rescaled such that $P$ takes values on $[a, b]$ $(b > a)$ as opposed to $[0, 1]$.

10. **Timeseries**
    For any numerical model parameter $P$ not depending the simulation cycle $\lfloor t \rfloor$, a forwarding model parameter $T(P)$ can be formed such that the value of $T$ does not change during simulation cycle $\lfloor t \rfloor$. $T$ samples $P$ once and only once at the beginning of cycle $\lfloor t \rfloor$ and returns the resulting value for all subsequent evaluation until cycle $\lfloor t + 1 \rfloor$ begins. All of the above stock numerical parameters can be realised as timeseries in this way.

The above stock features mean that virtually any model parameter whose value is numerical be converted to a heterogeneous distribution or a time–sensitive evolutionary process with ease.

## 4.4 Macroeconomic Firms

One of the first motivations for the standard model macroeconomy was to analyse the behaviour of macroeconomic systems in which boundedly rational firms are mutually connected by an input-output network. Acemoglu et al. (2012) studied the influence of such an input-output structure on the macroeconomic behaviour of the system in an equilibrium setting. Firms in Acemoglu et al. were perfectly rational, implying that the system is always in equilibrium and that the only randomness within the simulation world is as a result of productivity shocks. We intend to analyse a model in which firms are boundedly rational, which implies that the simulation should exhibit continuous out-of-equilibrium dynamics and adaptive behaviour. The noise in the model should originate from the behavioural uncertainty of firms as well as from productivity shocks.

Firms are interacting directly via the input-output network and indirectly via the markets. Firms compete for household consumption, and the result of this competition is price interaction: firm price decisions influence the environment of the other firms. The production decisions (quantity of goods/services) of each firm influences the demand for good in the economic system, both through the consumption market and via the input market. This adaptive behaviour implies that each firm will react to the decisions of the other firms, amplifying the individual noise and spreading the behavioural uncertainty throughout the system, and allowing for extensive out-of-equilibrium dynamics.

Macroeconomic firms, the default implementation of the firm agent type in the standard model, are skeletal agents whose market activities and decisions are delegated to a set of plugin decision rules. These plugin decision rules determine almost all of the activities of the firm save for its internal bookkeeping, market interaction mechanics and its inventories.

Firm decision rules are objects and are parameters in the meaning of §4.3. These decision rules are intended to be interchangeable, but it may be the case that some combinations of decision rule are expected. By default in the standard model the firm decision rules are grouped into five categories as follows:

1. **Target Production** Decisions
   Firms produce goods in business cycle $\lfloor t \rfloor$ and attempt to sell these goods in business cycle $\lfloor t \rfloor + 1$ or later (a 'produce today sell tomorrow' model). This means that all firms are subject to a forward-looking planning problem, and this planning problem includes uncertainty about the future state of the market. It is possible for firms to overproduce/saturate or undersupply the market.

   The target production decision rule must select, by any means, the volume of goods/services a firm *ideally* wishes to produce in business cycle $\lfloor t \rfloor$. Possibilties for the taget production decision rule are outlined in §4.4.1.

2. **Labour Wage Bid Price** Decisions
   Firms propose wages for new labour contracts and must decide at what level to set the wage proposition. It may be the case, depending on the implementation of the labour market, that low wage propositions (as compared to the market average) result in a firm workforce/productivity crisis and that high wages unnecessarily or unsustainable increase firm costs. Possibilities for the labour wage bid price decision rule are outlined in §4.4.2.

3. **Goods Ask Price** Decisions
   Firms must select the price per unit for goods offered to the market (the ask price). Firms will refuse to sell goods at a lesser price per unit than the ask price. If goods are durable (in the sense that goods last for multiple business cycles) then the selling price for goods produced at time $\lfloor t \rfloor$ may be different in business cycles $\lfloor t \rfloor + 1$, $\lfloor t \rfloor + 2$ and so on.

   The labour wage proposition, combined with (a) the goods ask prices selected by other firms and (b) the size of the existing firm cash reserve (firm liquidity at hand) form an upper bound on the commercial loan credit required by the firm for production activities. Firm ask price decisions are illustrated in §4.4.2.

4. **Commercial Loan (Credit) Demand** Decisions
   When a firm has decided what quantity of goods it would *ideally* like to produce (the target production decision) the firm must take into account its capabilities, including capital and labour, and also consider the future profitability of production. For instance if it is the case that the interest rate $r$ on commercial loans is greater than $p/c$ where $p$ is the goods selling price and $c$ is the overall unit cost to produce new goods, the firm would be unwise to take out any new commercial loans at all because the marginal profit on the full production cycle would be negative after loan liabilities no matter how the firm performs on the market.

The firm commercial loan decision rule is called the Credit Demand Function (the CDF) and is an object and a model parameter (§4.3). The firm CDF specifies how much commercial loan credit the firm should apply for given (a) the state of the firm, including its existing liquidity at hand, and (b) the quoted loan interest rate $r$. The mechanics of the firm credit demand decision and a variety of possible implementations are outlined in §4.4.6.

5. **Liquidity Targets/Dividend** Decisions
   At the end of the business cycle when firms have paid their labour wages, bought input goods, executed their production cycles and interacted with the financial system, firms must consider what to do with their most recent profits after sales. At this stage the firm must decide how much liquidity (cash at hand) it intends to retain for the next business cycle.

   In the current implementation of the standard model the firm must choose between cash hoarding and paying dividends: the target liquidity decision is equivalent to a dividend per share decision rule. Some possible implementations of this decision rule are outlined in §4.4.5.

In future this list is likely to grow. Additionally there is:

1. The structure of the firm production function $Y$ (§3.1), and;
2. The firm bankruptcy handler. The firm bankruptcy handler is the bankruptcy/insolvency resolution process to apply to the firm when its debts cannot be paid and when the terms of its contracts cannot be met. The firm bankruptcy handler is not a decision rule (in the sense that the bankruptcy handler is not a process decided by the firm manager) but the bankruptcy handler is nonetheless interchangeable with multiple possible implementations and is a model parameter. This component is illustrated in §4.4.7.

### 4.4.1 Target Production

The target production decision rule ($T$) selects the quantity of goods, $q$, that the firm intends to produce. $q$ is the *ideal* quantity of goods to produce: $q$ may or may not be realised. $T$ free to query the state of any object in the simulation world when deciding $q$. The algorithm used by $T$ to compute $q$ is arbitrary; for instance it is valid, but not necessarily useful, for $T$ to output random values.

It is expected that $T$ should represent the firms' beliefs about the quantity of goods it can sell to the market in the next business cycle (the demand $d$). The information taken into account by $T$ when formulating an estimate of the market demand $d$ is up to the decision rule. The following three implementations are commonly used:

1. **Heuristic Expansion**
   Heuristic expansion is a simple adaptive firm decision rule in which the firm will decide to increase production if and only if there were zero unsold goods in the last business cycle. Otherwise the firm will reduce production. The rate at which the firm modifies its production targets in response to

these two outcomes is customisable. This decision rule is illustrated in §4.4.1.1.

2. **Optimistic Producers**
   The optimistic producer target production decision rule supplements the Heuristic Expansion rule with an overproduction ('optimism') factor. This decision rule describes a process in which firms will increase production if unsold goods were less than a fixed percentage, $f$, of total production. A firm may not need to sell 100% of its goods in order to decide to increase production: 99.9% may be a sufficiently strong indicator of market demand. This threshold for expansion is called the 'firm optimism' or the 'grace factor'. $f = 0\%$ is equivalent to the Heuristic Expansion rule. This decision rule is outlined in §4.4.1.2.

3. **Follow Market Demand**
   This decision rule is an adaptive decision rule in which a firm will repeatedly adjust its target production $q$ in order to match market demand $d$. The rate of adjustment depends on the difference $|q - d|$. This algorithm is described in §4.4.1.3.

4. **Remote Control**
   A remote control target production decision rule is a decision rule in which the firm target production $q$ is a predetermined function of the business cycle $\lfloor t \rfloor$. The timeseries $\{q(0), q(1)...\}$ is specified by the user. See §4.4.1.4.

#### 4.4.1.1 Heuristic Expansion

The behaviour of the firms is very simple and described by the following stochastic equation.

Define $\Delta Q_{t-1}$ to be the difference between quantity of goods produced and sold in business cycle $\lfloor t \rfloor - 1$. The firm will select $q(\lfloor t \rfloor)$ as follows:

$$q(\lfloor t \rfloor) = \begin{cases} q(\lfloor t \rfloor - 1) \cdot (1 + \mu) & \text{if } \Delta Q_{t-1} = 0 \\ q(\lfloor t \rfloor - 1) \cdot (1 - \mu) & \text{if } \Delta Q_{t-1} > 0 \end{cases}$$

where $q(\lfloor t \rfloor - 1)$ is the last period production and $\mu$ is a positive random variable. Typically $\mu \sim \epsilon U[0, 1]$ where $\epsilon \geq 0$ is the customizable 'adaptation rate' and $U[0, 1]$ is a uniform distribution on the unit interval.

#### 4.4.1.2 Optimistic Producers

Optimistic Producer (OP) decision rules were originally introduced to increase commercial loans. In this program, productive agents have an intrinsic notion of optimism about their future market performance. Optimistic producers will increase production if all but a fraction of their yield[4] was sold.

For instance a producer might choose to increase production $q$ and raise prices $p$ if all but 2% of the yield was sold. This threshold (2%) is the so–called

[4] total production

'optimism' or 'grace factor' $\phi$ ($0 < \phi \leq 1$).

Producers select $q$ adaptively. Write $u$ for the unsold yield and $q(\lfloor t \rfloor - 1)$ for the production target in the last business cycle. The desired future production $q(\lfloor t \rfloor)$ is:

$$q(\lfloor t \rfloor) = \begin{cases} q(\lfloor t \rfloor - 1) \cdot (1 + \mu) & \text{if } \dfrac{u}{q(\lfloor t \rfloor - 1)} < \phi \\ q(\lfloor t \rfloor - 1) \cdot (1 - \mu) & \text{if } \dfrac{u}{q(\lfloor t \rfloor - 1)} \geq \phi \ , \end{cases}$$

where $q(\lfloor t \rfloor - 1)$ is the last period production target and $\mu$ is a positive random variable. Typically $\mu \sim \epsilon U[0, 1]$ where $\epsilon \geq 0$ is the customizable 'adaptation rate', $U[0, 1]$ is a uniform distribution on the unit interval and $\phi \geq 0$ is optimism.

Terms $\mu, \sigma, \phi$ appearing in this formula have the same meanings as above, but the numerical values of these terms may be different.

#### 4.4.1.3 Follow Market Demand

Goods markets are able to collect demand/supply information from past business cycles. Overall demand information, $d$, is particularly useful to firms.

One potential problem with the approaches outlined in §4.4.1.1 and §4.4.1.2 is that firms will decide to increase production only when unsold goods are low or zero. If unsold goods are zero it may be the case that $d \gg q$. Concretely, imagine a scenario in which the true market demand is $d = 100$ and the existing production target is $q = 1$. With a Heuristic Expansion decision rule this setup could result in $q \ll d$ for many business cycles because $q$ is incremented by a relatively small factor each cycle: $q(0) = 1, q(1) = (1+\epsilon), q(3) = (1+\epsilon)^2, q(4) = (1+\epsilon)^3$... and so on.

For this decision rule, firms gauge the (unknowable) current market demand $d(\lfloor t \rfloor)$ by increasing/decreasing $q(\lfloor t \rfloor)$ according to $d(\lfloor t \rfloor - 1)$:

$$q(\lfloor t \rfloor) \quad = \quad q(\lfloor t \rfloor - 1) + \mu \cdot \Big( d(\lfloor t \rfloor - 1) - q(\lfloor t \rfloor - 1) \cdot (1 - \phi) \Big),$$

where $\mu \sim \epsilon U[0, 1]$, $U[0, 1]$ is a random variable drawn from a uniform distribution in the unit interval $[0, 1]$, $\epsilon$ is a customisable parameter (the 'adaptation rate') governing the adaptiveness of the firm, and $\phi \geq 0$ is the optimism (as per §4.4.1.2).

Note that $d$ is the total demand for goods. It may be the case that multiple firms compete to produce the same type of goods. In this use case (namely, multiple firms per sector) $d$ is quoted on a per-firm market-share basis $d = d(F)$ where $F$ the name of the firm. This is important, as otherwise all firms will ultimately attempt to produce $q = dN(F)$ where $N(F)$ is the number of distinct firm agents competing to produce goods of the same type.

Convergence between demand and supply is efficient in this approach, and this behavioural plugin is used in the standard model by default.

#### 4.4.1.4 Remote Control

Remote control decision rules are the most customizable rules available. In this implementation, the user specifies the value of $q(\lfloor t \rfloor)$ before the simulation begins. $q(\lfloor t \rfloor)$ is specified as a timeseries or as a closed form function.

### 4.4.2 Goods Ask Price

The ask price decision rule ($P$) selects the selling price $p$ per unit goods. $p$ is the *minimum* selling price: the true selling price realised by the market may be greater than but not less than $p$. $P$ is free to query the state of any object in the simulation world when deciding $p$. The algorithm used by $P$ to compute $p$ is arbitrary; for instance it is valid, but not necessarily useful, for $P$ to output random values.

Increasing $p$ while keeping $q$ (target production, §4.4.1) fixed is expected to reduce sales. For this reason the price decision rule $P$ and the target production decision rule $T$ should at least be harmonious, if not mutually interacting. The following three implementations are commonly used:

1. **Optimistic Producers**
   The Optimistic Producer price decision rule complements the Optimistic Producer target production decision rule §4.4.1.2. It is expected but not required that these rules operate in tandem.

   This module describes a process in which firms will raise prices if unsold goods were less than a fixed percentage, $f$, of total production. A firm may not need to sell 100% of its goods in order to decide to raise prices: 99.9% may be a sufficiently strong indicator of market demand. This threshold for expansion is called the 'firm optimism' or the 'grace factor'. $f = 0\%$ describes a process in which firms will not raise prices unless unsold goods are zero.

   Note that the limit $f \longrightarrow 0$ may be numerically unstable if residual unsold goods are small but nonzero due to residuals in floating point arithmetic. This decision rule is outlined in §4.4.2.1.

2. **Production–Price Coupling**
   The production–price coupling rule slaves the firm selling price $p$ to target production $q$ (§4.4.1). The selling price $p$ is a function of $q$. The obvious generalisation of the 'follow market demand' decision rule §4.4.1.3 to $P$ is not ideal because it entails that the following question must be answered: 'if demand $d$ increases by 1 unit then by how much should price $p$ be increased?'.

   The production-price decision rule approaches this problem by instead asking: 'if target production $q$ is increased by a factor of $\alpha$, by what factor should price $p$ be changed?'. This rule is outlined in §4.4.2.2.

3. **Remote Control**
   A remote-control price decision rule is a decision rule in which $p$ is a predetermined function of the business cycle $\lfloor t \rfloor$. The timeseries $\{p(0), p(1)...\}$ is specified by the user. See §4.4.2.3.

#### 4.4.2.1 Optimistic Producer

The ask price $p(\lfloor t \rfloor)$ in business cycle $\lfloor t \rfloor$ is decided according to:

$$p(\lfloor t \rfloor) = \begin{cases} p(\lfloor t \rfloor - 1) \cdot (1 + \mu) & \text{if } \dfrac{u}{q(\lfloor t \rfloor - 1)} < \phi \\ p(\lfloor t \rfloor - 1) \cdot (1 - \mu) & \text{if } \dfrac{u}{q(\lfloor t \rfloor - 1)} \geq \phi \ . \end{cases}$$

where $p(\lfloor t \rfloor - 1)$ is the ask price in the last business cycle, $q(\lfloor t \rfloor - 1)$ is target production in the last business cycle, $\mu$ is a random variable distributed uniformly over the interval $[0, \epsilon]$ where $\epsilon \geq 0$ is the 'adaptation rate', $u$ is the quantity of unsold goods from the last production cycle, and $\phi$ is the 'optimism' or 'grace factor'.

Note that this prescription is potentially numerically unstable in the limit $\phi \longrightarrow 0$. This is because $u$ may be a small nonzero numerical residual due to finite-precision arithmetic. Note that $p$ is never zero if $p(0) \neq 0$ (except possibly due to long term rounding errors). Note also that the firm may be pessimistic: $\phi < 0$.

It is expected, but not required, that the Optimistic Producer price decision rule should operate in tandem with the Optimistic Producer target production decision rule §4.4.1.2.

#### 4.4.2.2 Production–Price Coupling

The Optimistic Producer price rule §4.4.2.1 is an adaptive decision rule. The price $p$ changes slowly and the total demand for goods/services $d$ is unimportant whenever $d > q$. Incorporating information about $d$ into §4.4.2.1 requires knowledge about the conversion between $\Delta d$ (change in demand) and $\Delta p$ (change in price).

One approach to this problem is to slave the selling price $p$ to the production target $q$. $q$ is decided before $p$ (for macroeconomic firm agents) so it is possible to express $p = p(q)$ as a function of $q$. We would expect that an increase in $q$ would result in an increase in $p$ and contrariwise.

One of the simplest couplings between $p$ and $q$ is:

$$\frac{p(\lfloor t \rfloor)}{p(\lfloor t \rfloor - 1)} = \frac{q(\lfloor t \rfloor)}{q(\lfloor t \rfloor - 1)},$$

Ie. 'double price when target production is doubled'. In order to reduce the sensitivity of the price response $\Delta p$ the $q$ quotient on the right hand side can be exponentiated:

$$\frac{p(\lfloor t \rfloor)}{p(\lfloor t \rfloor - 1)} = \left( \frac{q(\lfloor t \rfloor)}{q(\lfloor t \rfloor - 1)} \right)^{\alpha}.$$

If $\alpha \ll 1$ then $\Delta q = 2$ results in $p(\lfloor t \rfloor) = 2^{\alpha} \cdot p(\lfloor t \rfloor - 1)$. $2^{\alpha}$ is small so the price response $\Delta p$ is small when $\alpha \longrightarrow 0$. In general $p(\lfloor t \rfloor) = (1 + \Delta q / q(\lfloor t \rfloor - 1))^{\alpha} \cdot p(\lfloor t \rfloor - 1) \approx (1 + \alpha \Delta q / q(\lfloor t \rfloor - 1)) \cdot p(\lfloor t \rfloor - 1)$ for small $\alpha$.

A further ingredient is to ensure well-behavedness of the price $p$ when $q \sim 0$ for extended periods. $q \geq 0$ so it is possible to add a small nonzero constant $\xi$ satisfying $0 < \xi \ll \min(q, 1)$ to the right hand side:

$$\frac{p(\lfloor t \rfloor)}{p(\lfloor t \rfloor - 1)} \quad = \quad \left( \frac{q(\lfloor t \rfloor) + \xi}{q(\lfloor t \rfloor - 1) + \xi} \right)^{\alpha}.$$

This decision rule involves no intrinsic noise or random processes and has been known to exhibit stabilising properties.

#### 4.4.2.3 Remote Control

As for §4.4.1.4, remote control decisions are the most customizable firm ask price rules. In this implementation the user specifies the value of $p(\lfloor t \rfloor)$ before the simulation begins. $p(\lfloor t \rfloor)$ is specified as a timeseries or as a closed form function.

#### 4.4.2.4 Desired: Polling

None of the rules described in §4.4.2.1, §4.4.2.2 or §4.4.2.3 involve intra-cycle sampling. All of the decision rules outlined above increment the ask price $p$ once per business cycle based on past information; no knowledge about the *current* state of the market is involved. Real firms use marketing and market research agencies to assess the proposed price/profit relationship before committing to an ask price.

The experiment described in Assenza et al. 2015 (delivered during the CRISIS project) shows that subjects acting as firms are neither perfectly rational nor perfectly irrational. Subjects, and firms, are able to learn about their environment and take decisions following the profit incentive. Put differently, firms try to optimize their profits conditional on their understanding of the economic environment. The same behaviour has been used in a multi–country model, and drafted in a deliverable for the CRISIS project by the macroeconomic group at the Catholic University of Milan.

The proposed modification is to take into account profits when deciding prices and production by quantifying the increase in sales that would result from changes in price. In business cycle $\lfloor t \rfloor$, with a given probability, firms ask a market survey to understand the tastes of consumers. The survey agency provides the firm with information about the relationship between price $p$ (markup rate $\mu$) and expected sales/profits by randomly choosing a set of prices around the current ask price and polling consumer intentions at the perturbed price. Firms use this information to select $p$ such that expected profits are maximised. This behavioural model has already been experimented with in Dawid and Harting (2012).

This approach potentially has a few problems, though none are thought to be difficult. Using this approach, firms will choose the price that maximises profits without taking into account possible input constraints. Either the survey agency or the firm should take into account resource constraints. Another way to tackle the problem without changing the heuristic would be to introduce a labour wage $w$ that changes as a response to the tightness of the labour market.

### 4.4.3 Production Structure

The general structure of the firm production function $Y$ is as indicated in §3.1.

The default firm agent implementation in the standard model is a macroeconomic firm agent, and the macrofirm agent provides the following stock production functions $Y$:

1. **Labour Only**
   A simple production function not depending on input goods/services. Firms using this production function only require labour to produce goods.The conversion rate between labour and capital production need not be linear. This production function is outlined in §4.4.3.1.

2. **IO Cobb–Douglas**
   A Cobb-Douglas production function depending on multiple differentiated goods/service inputs as well as labour. The IO Cobb–Douglas system describes a production process for which there is a degree of substitutability between input goods and also substitutability between labour and non-labour inputs. See §4.4.3.2

3. **von–Neumann Leontieff**
   The von–Neumann Leontieff production function ('Fixed Proportions Production Function') is a limiting case of the IO Cobb–Douglas production function for which there is zero substitutability between differentiated input goods/services. Unavailability of one input good will result in zero overall production. This production function is outlined in §4.4.3.3.

#### 4.4.3.1 Labour Only

The empty Input-Output network in which firms do not buy input goods from other firms is equivalent to a labour-only production function:

$$\begin{aligned} Y(L, g_i) &= Z(t) \cdot L^{\alpha}, \\ c &= wL, \end{aligned}$$

where $Y$ is production, $L$ is labour employed, $g_i$ is the quantity of input goods of type $i$, $Z(t)$ is the firm Total Factor Productivity, $\alpha$ is a numerical model parameter, $c$ is the total cost of production (assuming inputs $L$ and $g_i$) and $w$ is wage per unit labour. If the firm intends to produce $Y = q$ units of goods then the optimal solution is:

$$
\begin{aligned}
L &= \left(\frac{q}{Z(t)}\right)^{1/\alpha}, \\
c_{\min} &= w\left(\frac{q}{Z(t)}\right)^{1/\alpha}, \\
g_i &= 0,
\end{aligned}
$$

where $c_{\min}$ is the minimal cost incurred to produce $q$. Note that $\alpha = 1$ has linear returns to scale.

#### 4.4.3.2 IO Cobb–Douglas

For an Input-Output network the Cobb–Douglas production function is formulated as:

$$
\begin{aligned}
Y(L, g_i) &= Z(t)^{\alpha} L^{\alpha} \prod_{j=1}^{N_S} g_j^{(1-\alpha)w_j}, \\
c &= wL + \sum_{j=1}^{N_S} p_i g_i,
\end{aligned}
$$

where $Y$ is production, $L$ is labour employed, $g_i$ is the quantity of input goods of type $i$, $p_i$ is the price per unit paid for goods of type $i$, $N_S$ is the number of sectors, $Z(t)$ is the firm Total Factor Productivity, $\alpha$ and $w_j$ $(j = 0, 1...)$ are numerical model parameters, $w$ is wage per unit labour and $c$ is the total cost of production. The parameters $w_j$ are called 'technological weights'.

Supposing that firms have supplied their existing goods to the market and that firms now decide their production levels for the current period, the object of the firm is to produce $Y = q$ at the minimum possible cost $c$. The solution to the firm planning problem is:

$$
\begin{aligned}
L &= \frac{q}{Z(t)^{\alpha}\left[\frac{(1-\alpha)}{\alpha}h\right]^{(1-\alpha)}\prod_{j=1}^{N_S}\left[\frac{w_{ij}}{p_j}\right]^{(1-\alpha)w_j}} \\
g_i &= \frac{q}{Z(t)^{\alpha}\left[\frac{(1-\alpha)}{\alpha}h\right]^{-\alpha}\left[\frac{w_i}{p_i}\right]^{-1}\prod_{j=1}^{N_S}\left[\frac{w_j}{p_j}\right]^{(1-\alpha)w_j}}
\end{aligned}
$$

Rationing can occur on both on the goods markets and on the labour market. If the demand is higher than the supply, firms might not be able to buy all the inputs they need for production or hire all the workers they desire.

Note that the parameters $w_j$ need not be fixed in time and are different for each firm. For firm $i$ there are $N_S$ technological weights $w_{ij}$. The technological weight matrix (of size $N_S \times N_S$) $w_{ij}$ is of special interest.

The matrix $W$ containing all the elements $w_{ij}$ is an adjacency matrix representing a *directed* and *weighted* network, where the nodes are the firms and the

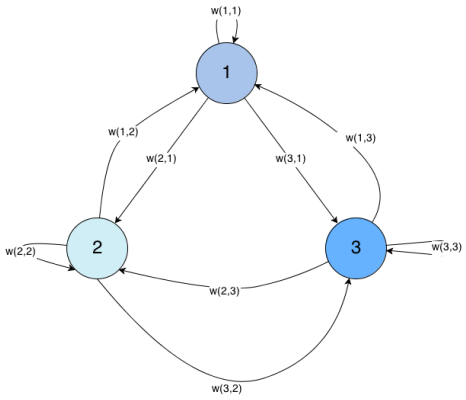



Figure 2.1: The network defined by $W$ with 3 firms

links represent the technological links between the production function. Figure 2.1 shows an example of the network defined by the matrix $W$.

We assume $w_{ij} \geq 0$ and that the technological weight matrix has constant return to scale ie. $\sum_j w_{ij} = 1$ for all $i$. The values $w_{ij}$ define the input-output network in the economic system. If $w_{ij} > 0$ there is a link (production dependency) between firm $i$ and firm $j$: firm $i$ is is able to utilize good $j$ for production. If $w_{ij} = 0$ then no link is present and firm $i$ is not using good $j$ for production. Note that $w_{ij} > 0$ does not necessary imply that goods and services will flow from firm $i$ to firm $j$; it is merely the case that goods/services of type $j$ can in principle be used as ingredients in the production structure of firm $j$.

#### 4.4.3.3 Leontief

The Cobb–Douglas Input–Output network has been supplemented with a Leontief Input-Output economy. In general, firms aim to produce a target quantity $q$ of goods whilst minimising total economic costs incurred. Recall that the production target $q$ is computed individually by all firms based on their financial position and their sales expectations. The process used to do this has been updated to accommodate arbitrary production functions, including the Leontief case.

The Leontief production function takes the form:

$$\begin{aligned} Y(L, g_i) &= Z(t) \min(\alpha L,\, (1-\alpha) w_1 g_1,\, (1-\alpha) w_2 g_2 \ldots), \\ c &= wL + \sum_{j=1}^{N_S} p_i g_i, \end{aligned}$$

where $L$ is labour employed, $Z(t)$ is the Total Factor Productivity, $w_i$ are customizable weights ('technological weights') and $g_i$ is the quantity of goods/services of type $i$ utilised for production.

The optimal solution to the firm planning problem is:

$$\begin{aligned} L &= \frac{q}{Z(t)\alpha}, \\ g_i &= \frac{q}{Z(t)(1-\alpha)w_i}. \end{aligned}$$

#### 4.4.3.3.1 Autoconfiguration for Leontief Economies

The standard model has hundreds of customisable macroeconomic parameters. Normally these parameters are selected by the user via GUIs.

In the Leontief case it is possible in principle to autoconfigure the firm production function to encourage (a) specific labour employment levels, $\epsilon_\ell$, and (b) specific levels of domestic goods consumption, $\epsilon_x$.

$\epsilon_\ell$ is the overall labour employment proportion ($0 \leq \epsilon_\ell \leq 1$). In Cobb–Douglas simulations with 50 productive sectors and 1000 household agents we tend to observe $\epsilon_\ell \approx 0.95$ (namely 5% unemployment). $\epsilon_x$ is the proportion of all goods ($0 \leq \epsilon_x \leq 1$) used in production activities (as opposed to domestic consumption). In Cobb–Douglas simulations we tend to observe $\epsilon_x \approx 0.25$ (Ie. 75% of all goods are consumed by households).

If $\epsilon_\ell$ and $\epsilon_x$ are decided in advance by the user then the Leontief economy can be configured as follows:

1. $q = Z(t)\min(\alpha\ell, (1-\alpha)w_1 g_1, (1-\alpha)w_2 g_2 \ldots)$ is optimised (costs minimized) by $q/L = Z(t)\alpha$;
2. $L/g_i \approx (1/\alpha - 1)w_i$ for each $w_i$;
3. Select:

$$\begin{aligned} L &\approx \epsilon_\ell \frac{N_H}{N_F}, \\ q &\approx q_{\text{init}}, \\ x &= \epsilon_x \frac{q_{\text{init}}}{N_F}, \\ \alpha &\sim w\frac{\epsilon_x q_{\text{init}}}{\epsilon_\ell N_H}, \\ Z(t) &\sim q/L\alpha, \end{aligned}$$

where $q_{\text{init}}$ is the initial firm production target, $N_H$ is the number of households (each endowed with 1 unit of labour), $N_F$ is the number of firms in the macroeconomy, L is labour employed, and $g_i$ is the quantity of good type $i$ utilised for production.

### 4.4.4 Wage Bid Prices (Per Unit Labour)

Firms must select the price per unit for goods offered to the market (the ask price). Firms will refuse to sell goods at a lesser price per unit than the ask

price. If goods are durable (in the sense that goods last for multiple business cycles) then the selling price for goods produced at time $\lfloor t \rfloor$ may be different in business cycles $\lfloor t \rfloor + 1$, $\lfloor t \rfloor + 2$ and so on.

The ask price per unit labour is the maximum amount a firm is willing to pay for new labour contracts. Depending on the implementation of the labour market and the heterogeneity of labour contracts, it may be the case that that firm pays less than the desired as price. As a result the firm planning problem must assume the worst case (most expensive wage) scenario.

By default there is no notion of inflation in the standard model and wages $w$ are fixed. This is realised by fixing the bid and ask prices per unit labour for all market participants.

1. **Fixed Bid Price** Decision Rule
   The ask price per unit labour $w$ is fixed in time and inflexible (§4.4.4.1).
2. **Supply Competition** Decision Rule
   Wages are high when competition for labour is high (unemployment is low) and contrariwise (§4.4.4.2).
3. **Remote Control** Decision Rule
   Any of the stock numerical parameters described in §4.3.2 apply to the wage ask price decision rule (§4.4.4.3).

#### 4.4.4.1 Fixed Bid Price

The Fixed Bid Price decision rule is the simplest implementation of the firm wage decision rule.

In this implementation the firm will bid an amount $w$ per unit labour no matter what the simulation time $t$ and no matter what the state of the simulation world. $w$ is customizable and must be specified by the user. It is expected but not required that $w > 0$.

Typically this decision rule would be used in tandem with complementary 'Fixed Ask Price' decision rules for household agents (namely, decision rules that ask the exact same price $w$ per unit labour).

#### 4.4.4.2 Supply Competition

The labour market provides information about total employment. This information includes:

1. the proportion $e$ of the workforce in employment ($0 \leq e \leq 1$)
2. the proportion of the workforce $u$ that is unemployed ($u = 1 - e$), and
3. the total labour supply $s$, and the total demand for labour $d$.

Using this information it is possible to formulate a simple adaptive wage decision rule as follows:

$$w(\lfloor t \rfloor + 1) = \begin{cases} (1+\mu) \cdot w(\lfloor t \rfloor) & \text{if } d > s, \\ (1-\mu) \cdot w(\lfloor t \rfloor) & \text{otherwise} \end{cases},$$

where $w(\lfloor t \rfloor)$ is the wage ask price in business cycle $\lfloor t \rfloor$, $\mu$ is a random variable distributed uniformly over $[0, \epsilon]$ where $\epsilon > 0$ is the 'adaptation rate', $d$ is the total demand for labour (observed in the last business cycle) and $s$ is the total supply of labour (observed in the last business cycle).

This decision rule describes an adaptive process for which high labour demand results in increasing wages. In the limit $w \longrightarrow \infty$ it is expected that the total demand $d$ for labour will decrease due to cost, which in turn reduces $w$. The user is required to specify the initial ask price $w(0)$.

#### 4.4.4.3 Remote Control

In this implementation the wage ask price $w$ is realized as a numerical model parameter in the meaning of §4.3.2. Any of the realizations listed in §4.3.2 are acceptable.

### 4.4.5 Liquidity Targets and Dividends

Firms must select the price per unit for goods offered to the market (the ask price). Firms will refuse to sell goods at a lesser price per unit than the ask price. If goods are durable (in the sense that goods last for multiple business cycles) then the selling price for goods produced at time $\lfloor t \rfloor$ may be different in business cycles $\lfloor t \rfloor + 1$, $\lfloor t \rfloor + 2$ and so on.

The ask price per unit labour is the maximum amount a firm is willing to pay for new labour contracts. Depending on the implementation of the labour market and the heterogeneity of labour contracts, it may be the case that that firm pays less than the desired as price. As a result the firm planning problem must assume the worst case (most expensive wage) scenario.

By default there is no notion of inflation in the standard model and wages $w$ are fixed. This is realised by fixing the bid and ask prices per unit labour for all market participants.

1. **Absolute Liquidity Target** Decision Rule
   The ask price per unit labour $w$ is fixed in time and inflexible (§4.4.4.1).
2. **Forwarding AR1** Decision Rule
   Wages are high when competition for labour is high (unemployment is low) and contrariwise (§4.4.4.2).
3. **Sticky Dividends** Decision Rule
   Any of the stock numerical parameters described in §4.3.2 apply to the wage ask price decision rule (§4.4.4.3).
4. **Remote Control** Decision Rule
   Any of the stock numerical parameters described in §4.3.2 apply to the wage ask price decision rule (§4.4.4.3).

#### 4.4.5.1 Absolute Liquidity Target

This implementation is the simplest possible specification for the firm liquidity decision rule $L$. This decision returns a fixed number $\xi$ no matter what the simulation time $t$ and no matter what the state of the simulation world. It is expected but not required that $\xi > 0$.

#### 4.4.5.2 Forwarding AR1

This implementation of the firm liquidity decision $L$ is a forwarding decision rule: a decision $L$ which depends on an independent decision made by another algorithm $L'$. The forwarding rule $L'$ is also an instance of a firm liquidity decision algorithm §4.4.5.

The decision made by $L$ is as follows:

1. $L'$ is asked (at time $t$) to compute a firm liquidity target $\xi'(t)$. $L$ has no influence over the decision made by $L'$.
2. $L$ accepts $\xi'(t)$ as input and commits this number to memory
3. The decision $\xi(t)$ made by $L$ is $\alpha \cdot \xi(t) + (1 - \alpha) \cdot \xi(t - 1)$ where $\alpha$ is a model parameter.

The parameter $\alpha$ ($0 \leq \alpha \leq 1$) is the 'adaptation rate'. If $\alpha = 1$ then $L$ is transparent and returns exactly the decisions made by $L'$. If $\alpha = 0$ then $L$ delays the decisions made by $L'$ by one simulation cycle. All other values of $\alpha$ describe an exponential smoothing process in which past decisions made by $L'$ are smoothed against its most recent decisions.

Note that it is acceptable for $L'$ to also be a forwarding decision rule.

#### 4.4.5.3 Sticky Dividends

Assuming that firms are stock releasers, that the number of shares issued by the firm is nonzero and that the firm does not own any of its own shares, then the liquidity decision $L$ made by the firm is equivalent to a dividend decision $D$ where $D = \max(0, C - \min(L, C))$ and $C$ is the existing firm liquidity before $L$ is decided (§4.4.5).

The Sticky Dividend decision rule is an adaptive implementation of $D$. The Sticky Dividend decision rule accepts a delegate liquidity decision rule $L$ and, using $L$, computes a smoothed dividend target $D$. Concretely:

1. $L$ computes a liquidity target, $\xi(t)$.
2. $\xi(t)$ is considered to be a liquidity lower bound for the firm after dividends.
3. The maximum *affordable* dividend $d_{\max}$ given $\xi(t)$ is computed:

$$d_{\max} = C - \xi(t),$$

where $C$ is existing liquidity.

4. The dividend envelope $e(t)$ is computed:

$$e(t) = \alpha \cdot d_{\max} + (1 - \alpha) \cdot e(t - 1)$$

5. $e(t)$ is not allowed to exceed $(1 + \mu_+) \cdot e(t - 1)$ where $0 \leq \mu_+$ is a model parameter.

6. $e(t)$ is not allowed to fall below $(1 - \mu_-) \cdot e(t - 1)$ where $0 \leq \mu_- \leq 1$ is a model parameter.

7. The actual dividend $d$ is set to be $d = \min(d_{\max}, e)$, and the value of $L$ is computed accordingly.

Typically $L$ will be implemented as an absolute/fixed liquidity target §4.4.5.1. The above process reduces shocks in the firm dividend stream. Real firms are highly reluctant to increase dividends and are also reluctant to reduce dividends below levels for which there are established shareholder expectations (for fear of communicating internal financial difficulties to the market).

For this reason firm dividends tend to change significantly only when firm profits undergo substantial and sustainable change. This is the purpose of the dividend envelope $e(t)$. The envelope $e(t)$ increases somewhat every time the affordable dividend $d_{\max}$ increases, and conversely $e(t)$ falls somewhat if the affordable dividend is reduced. When using this decision rule, high profits tend to remain on the balance sheet and unexpectedly low profits result in low dividend payments until profits return to normal.

#### 4.4.5.4 Remote Control

In this case the liquidity target $L$ is realized as a numerical model parameter in the meaning of §4.3.2. Any of the realizations listed in §4.3.2 are acceptable.

### 4.4.6 Credit Demand

Once a firm has decided the production target $q$ (§4.4.1) and the wage bid price $w$ (§4.4.4) the total liquidity required for production is bounded above by the cost of input goods, input services and labour.

The primary external source of new firm liquidity is commercial loans. Firms should not take out commercial loans without considering the future cost of doing so, and importantly, it is the case that firms should completely refrain from borrowing if the loan interest rate $r$ is sufficiently large ($r \longrightarrow \infty$).

A firm must therefor not only decide $q$ and $w$, but must also decide how to reduce ('ration') $q$ as a result of its inability or unwillingness to borrow at prohibitive interest rates. The firm must specify the loan demand $L$ as a function of $r$. This function $L(r)$ (the 'Credit Demand Function' or 'CDF') is a plugin decision rule and is a model parameter in the meaning of §4.3.2. The standard model provides several stock implementations of the CDF. Before describing these implementations, it is necessary to outline the debt problem faced by the firm as a result of commercial loans.

Given the desired production and the price of inputs, the firm can compute the desired liquidity. The difference between the desired liquidity and the internal resources of the firm represents the maximum/desired amount of loans $L_d$ that the firm will apply for.

Define $c$ as the current[5] and immediate[6] cost to produce one unit of new goods. Define $\Lambda$ as the internal liquid resources available for production (Ie. the equity of the firm that can be used for production). We can write:

$$\frac{\Lambda}{c} + \frac{L_d}{c} = q,$$

and

$$L_d = qc - \Lambda,$$

where $qc$ is the desired liquidity (the resources needed to produce $q$). The firm desires to borrow $L_d$ to bridge te gap between the desired liquidity $qc$ and the internal liquidity $\Lambda$.

To understand the overall profitability after commercial loans, we define the maximum profit $\Pi$ after borrowing (profits as a result of the possibility of increased production thanks to new liquidity) as:

$$\Pi = py - cy - rL,$$

where $p$ is the ask price per unit goods to sell, $y$ is the amount of good the firm can produce with the borrowed resources, $r$ is the loan yield to maturity (overall interest rate) and $L$ is the principal value of the loan ($L \leq L_d$). This relationship assumes a best-case scenario in which the firm will succeed to sell the whole production $y$. $y = L/c$ so we have:

$$\Pi = p\frac{L}{c} - c\frac{L}{c} - rL.$$

By dividing both sides of the equation by $L$, we get the expected profits per unit of borrowed liquidity:

$$\frac{\Pi}{L} = \frac{p}{c} - 1 - r.$$

The maximum interest rate $r_{max}$ that is acceptable to the firm (the interest rate at which profits from taking the loan are zero) is therefor:

$$r_{max} = \frac{p}{c} - 1.$$

Note that the expression $p/c - 1$ is the definition of the *production markup rate* $\mu$. There is a possibility, no matter how slim, that a loan to be profitable to the firm if the interest rate $r$ is lower than the expected markup rate $\mu$.

We assume that the firm CDF has the following form:

[5]Note that the unit production cost $c$ depends on the prices of all input goods/services and on the cost of labour.

[6]The cost incurred in cash terms to carry out production; not including possible future charges such as accrued commercial loan interest.

$$D(r) = \ell(r)L_d,$$

where $D(r)$ is the size of the commercial loan requested by the firm, $0 \leq \ell(r) \leq 1$ is the proportion of $L_d$ requested at interest rate $r$, and $L_d$ is the maximum (ideal) commercial loan size described above. Note that $\ell$ may depend on other factors such as the credit history of the firm, historical records of default and/or insolvency, the markup rate $\mu$ etc.

The CDF is a firm decision rule and a model parameter in the meaning of §4.3. The standard model provides several stock implementations of $\ell(r)$ (§4.4.6). These implementations include:

1. The **Risk Neutral CDF**
   This CDF describes a firm that is indifferent to risk. The firm will apply for the maximum amount $D = L_d$ in commercial loans provided there is any possibility (no matter how small) that the loan will ultimately be profitable. This decision rule is described in §4.4.6.1.

2. The **Custom Risk Affinity CDF**
   The custom risk affinity CDF is the general-purpose CDF for macroeconomic firms in the standard model. This decision rule is illustrated in §4.4.6.2. Other CDFs described here are limiting or special cases of this CDF.

3. The **Risk Averse CDF**
   A credit profile for a firm with very low tolerance for credit risk. This decision rule is outlined in §4.4.6.3.

#### 4.4.6.1 Risk Neutral Borrower

This decision rule describes a firm that is indifferent to risk. The profile $\ell(r)$ of the CDF is described by:

$$\ell(r) = \begin{cases} 1 & \text{if } r \leq \mu \\ 0 & \text{otherwise} \end{cases}$$

The firm will borrow the maximum amount required without rationing whenever the loan interest rate $r$ is less than the markup rate $\mu$. At greater interest rates the loan will inevitably result in a marginal loss (§4.4.6) and the firm will not borrow.

#### 4.4.6.2 Custom Risk Affinity Borrower

In order to introduce the general–purpose nonlinear firm CDF we first begin with a discussion of the linear case.

##### 4.4.6.2.1 The Linear Case

Initially we will assume that:

1. $\ell(r)$ is a linear function, and

2. $\ell(r) = 0$ for $r \geq \mu$ and $\ell(\nu) = 1$ for $r \leq \nu$ for some $\nu \geq 0$.

$\mu$ is the production markup rate and $\nu$ is a model parameter denoting the interest below which firms are willing to borrow the whole amount of desired liquidity ($L_d$).

These assumptions completely specify $\ell(r)$:

$$\ell(r) = \begin{cases} 1 & r \leq \nu \\ \dfrac{\mu}{\mu - \nu} - \dfrac{r}{\mu - \nu} & \text{otherwise} \\ 0 & r \geq \mu, \end{cases}$$

in which case $D(r)$ can be written as:

$$D(r) = \begin{cases} L_d & r \leq \nu \\ L_d \left( \dfrac{\mu - r}{\mu - \nu} \right) & \text{otherwise} \\ 0 & r \geq \mu. \end{cases}$$

Depending on the interest rate $r$ the firm will demand a share of the desired loan ($L_d$). If the interest rate $r$ is higher than the markup rate $\mu$ the firms do not ask for loans because the credit cost is higher than the sum of all profits they may receive from increasing production with the aid of the loan. The lower the interest rate $r$ the higher the loan demand $D(r)$. In practice the value $\nu = 0$ is chosen because this is the only value for which $\sum_{\text{firms}} D(r)$ is a strictly monotonic decreasing function near $r \sim 0$.

The linearity of the credit demand function has been assumed for convenience to illustrate one particularly simple case. This assumption is ultimately arbitrary, but one might argue that firms know that their sales expectations might be wrong. The lower the profits, and higher the risk of incurring a loss. Reducing the credit demand when the interest rate increases reduces the risk of losses given uncertainty about future sales of goods and services and, depending on the relationship between $D(r)$ and risk, the underlying credit profile $\ell(r)$ may be linear.

In the next section we will describe the default macrofirm credit demand function for which this linear decision rule is a special case.

#### 4.4.6.2.2 The Nonlinear Case

Assuming that the firm CDF has the following properties:

1. $\ell(r)$ is a piecewise polynomial function, and
2. $\ell(r) = 0$ for $r \geq \mu$ and $\ell(\nu) = 1$ for $r \leq \nu$ for some $\nu \geq 0$,

where $\mu$ is the production markup rate and $\nu$ is a model parameter denoting the interest below which firms are willing to borrow the whole amount of desired liquidity ($L_d$), then $\ell(r)$ has the form:

$$\ell(r) = \begin{cases} 1 & r \leq \nu \\ \left(\dfrac{\mu - r}{\mu - \nu}\right)^{\alpha} & \text{otherwise} \\ 0 & r \geq \mu, \end{cases}$$

for a customizable model parameter $\alpha \geq 0$ (the 'credit risk affinity'). In this case $D(r)$ can be written as:

$$D(r) = \begin{cases} L_d & r \leq \nu \\ L_d \left(\dfrac{\mu - r}{\mu - \nu}\right)^{\alpha} & \text{otherwise} \\ 0 & r \geq \mu. \end{cases}$$

Note that $D(r)$ is strictly monotonic decreasing for $\nu < r < \mu$ and is constant for all other values. Commercial banks (the providers of commercial loans) are subject to a similar decision problem. The bank Credit Supply Function (CSF) $L^*(r)$ is expected to be zero for $r = 0$ (as there is no profit in lending at zero interest) and rise sharply with $r$. The only circumstances in which $L(r) = L_d$ for the firm is if $\nu > 0$ is larger than the interest rate on the loan.

This can lead to a semi-equilibrium state in which yield is consistently less than the firm production target even if the production target correctly reflects the total demand for goods/services over the market. It is therefor desirable to express $\nu$ in terms of $\mu$ as a fraction $\nu(\mu) = \kappa\nu$ $(0 \leq \kappa \leq 1)$ where $\kappa$ is a model parameter in the meaning of §4.3.

The general-purpose custom risk affinity CDF is therefor:

$$\ell(r) = \begin{cases} 1 & r \leq \kappa\mu \\ (\mu - r)^{\alpha}\, \mu^{-\alpha} (1 - \kappa)^{-\alpha} & \text{otherwise} \\ 0 & r \geq \mu, \end{cases}$$

where $\kappa$ is customizable.

#### 4.4.6.3 Risk Averse Borrower

This decision rule describes a firm with zero risk tolerance. The profile of the CDF is described by:

$$\ell(r) = \begin{cases} 1 & \text{if } r \leq \xi \\ 0 & \text{otherwise} \end{cases},$$

where $r_{\min}$ is a model parameter specified by the user. It is expected but not required that $\xi \ll 1$ to ensure that $\xi < \mu$ (the production markup rate) at all times. This CDF is a limiting case of §4.4.6.2 for which $\nu$ is fixed and $\alpha \longrightarrow \infty$.

For convenience (to avoid numerical problems) we also impose a minimum interest rate $r_{textmin}$ which is close to zero. We assume for the moment that

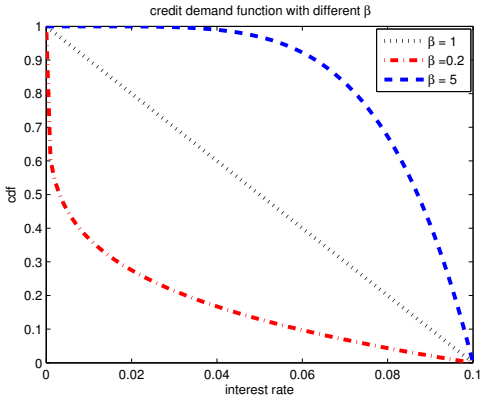



Figure 2.2: The general purpose custom risk affinity firm credit demand function for various values of $\alpha$; $\nu = 0$, $\mu = 1/10$.

the demand function for credit can then be written in the form

### 4.4.7 Firm Bankruptcy Procedure

In the standard model the number of firms does not change in time. Rather, when a firm fails it is made subject to a bankruptcy resolution process (the 'firm bankruptcy handler') which will either save the firm, allowing it to continue as a going concern, or destroy the firm and regenerate it.

The means through which the firm bankruptcy handler resolves the firm bankruptcy/insolvency/debt crisis is up to the implementation: firm bankruptcy handlers are objects and are model parameters in the meaning of §4.3.

#### 4.4.7.1 Default Condition

Firms fail when either:

1. The firm fails to pay its debts or its obligations when these become due, and/or
2. The firm equity is negative: $A < L \implies E < 0$ (§4.4.8)

Using (1) without (2) is considered to be a stronger modelling assumption.

When a firm fails it is made subject to a bankruptcy handler whose purpose is to restore (or begin to restore) the financial position of the firm so that it may continue as a going concern. It is not unusual for the firm bankruptcy handler to be required to address outstanding creditor debts and the bankruptcy handler may be required to do this even if the firm has insufficient liquid or illiquid assets to finance the debt. In these circumstances the firm may be wound down and it may be the case that creditors are required to accept losses.

Firm equity includes inventory evaluated at the market price, so firms can become insolvent as a result of sudden changes in the value of goods/services. The most likely trigger for the firm bankruptcy handler is cash flow insolvency in the form of failure to repay commercial loans, loan interest instalments and/or labour wages. Particular care must be taken when processing goods and labour settlements to avoid cases in which the underlying cash transaction fails as a result of otherwise negligible numerical rounding errors: such a failure could result in the firm bankruptcy handler being triggered inappropriately.

The standard model provides several possible implementations of the firm bankruptcy handler. These implementations derive from a common framework as follows.

#### 4.4.7.2 Bankruptcy Handler Framework

Firm bankruptcy handlers in the standard model consist of three ordered steps:

1. A cash compensation algorithm object is asked to compute the maximum amount of firm liquidity, $C$, that can be used to pay existing creditor debts. In this implementation it is expected that creditor debts consist of unpaid commercial loan instalments only. Up to $C$ units of firm liquidity are paid to creditors in proportion according to the outstanding debt to each creditor.

2. Should $C$ be insufficient to repay the outstanding creditor debt, this bankruptcy resolution process resorts to a stock redistribution. A stock redistribution algorithm is triggered. If this algorithm does not fail then the bankruptcy resolution is taken to be complete and no further action is taken.

3. If stock redistribution is insufficient to compensate creditors for the outstanding firm debt, then this bankruptcy resolution algorithm liquidates the firm by erasing its existing stock instrument and cancelling all outstanding commercial loans. Firm shares are created anew with a stock price equal to the value of the total firm assets, and all shares are distributed to the firm's creditors in proportion to the debt owed to each creditor.

This process if somewhat analogous to a commercial bank bailin procedure, and is referred to as such.

The first step of this procedure (1) is to check whether the firm is financially capable of paying its outstanding debt. The cash compensation algorithm in described is required to decide how much of the existing firm liquidity can be used to pay creditors who have suffered default. The cash compensation algorithm is an object and a model parameter in the meaning of §4.3.2. This component is illustrated in §4.4.7.2.1.

The second step in the resolution procedure is a stock redistribution. The stock redistribution policy is an object and a model parameter in the meaning

of §4.3.2. This step is executed only if the cash compensation process (1) is insufficient to compensate firm creditors. This component is outlined in §4.4.7.2.2.

The final step (3) is applied only if (2) still fails to compensate creditors. In this case the firm is wound down (liquidated), its stock instrument is erased, and a new firm is regenerated in its place. The initial equity of the new firm is equal to the total asset value of the liquidated firm, and shares in the regenerated firm are issued for free and exclusively to the creditors of the liquidated firm.

#### 4.4.7.2.1 Cash Compensation

The cash compensation algorithm must select the maximum amount of cash $C$ that the firm is to pay to its creditors in an attempt to resolve the outstanding debt. In the standard model the cash compensation algorithm sets $C = 90\%$ of all unallocated firm deposits. If the firm has nonzero allocated cash then the compensation algorithm does not undo these allocations as this could generate additional financial problems for the firm after the bankruptcy handler has exited.

The sum $C^* = \min(C, D)$ where $D$ is the total debt owed to creditors is withdrawn from the firm deposit account and distributed to creditors. Each creditor receives a proportion of $C^*$ according to the relative size of the debt owed to it. This process either satisfies the creditors (who have been repaid late) or partly satisfies all creditors. If creditors are repaid then no further action is taken and the firm bankruptcy handler terminates.

Note that many goods/services are nondurable (transient), that durable goods are subject to depreciation in time, and that firm production in business cycle $\lfloor t \rfloor$ will not be realized as new capital until business cycle $\lfloor t \rfloor + 1$. For this reason auctioning firm inventory as a means of compensating creditors is not implemented (though this feature may be implemented in future).

#### 4.4.7.2.2 Stock Redistribution

If firm creditors cannot be compensated out of existing firm liquidity §4.4.7.2.1 and if no other capital is available then at this stage few options remain to the administrator. The equity of the firm is insufficient to compensate its creditors and if the firm is liquidated then all shares owned by stockholders will become worthless. If possible it is therefor prefereable for non-creditor stockholders to suffer partial stock losses and for the firm to not be liquidated: the alternative is for non-creditor stockholders to lose all of the value of their shares.

The purpose of the stock redistribution algorithm is to redistribute firm shares from non-creditor stockholders to creditor stockholders in an effort to compensate creditors at the expense of other shareholders. This process is numerically equivalent to issuing new shares and diluting the firm share price.

In the standard model the stock redistribution algorithm is a follows:

1. Ask the stock exchange for a complete list $S$ of firm stockholders.

2. Partition $S$ into two disjoint subsets $C$ and $\neg C$. $C \subset S$ is the subset of stockholders who are also firm creditors and who have suffered losses as a result of default. $\neg C$ is the subset of firm stockholders who are not creditors suffering default.

3. Add to $C$ all creditors who have suffered losses as a result of default but who are not firm stockholders. If any such creditor is incapable of holding shares in their own right, create an intermediary whose responsibility is to hold shares on behalf of the creditor. The intermediary will gradually sell any shares it comes to own in order to raise cash for the creditor.

4. Compute the total value $V$ of all shares held by agents in $\neg C$ (evaluated at the current price per share).

5. If $V \geq D$ where $D$ is the remaining debt owed to creditors after cash compensation (§4.4.7.2.1) then remove, without compensation, $\alpha = (100 \cdot D/V)$ percent of all shares from all non-creditor stockholders $\neg C$. Give all these shares to creditors $C$ at no cost. Each creditor in $C$ receives shares in proportion to amount of debt owed by the firm to the creditor.

6. Otherwise (if $V < D$) then the total value of shares owned by non-creditor stockholders $\neg C$ is insufficient to compensate the firm creditors. In this eventuality the equity of the firm combined with the value of its stock market capitalization is still inadequate to resolve the default. The firm is now liquidated as per §4.4.7.2.3.

#### 4.4.7.2.3 Liquidation

If after cash compensation §4.4.7.2.1 and stock redistribution §4.4.7.2.2 firm creditors have still not been compensated then the firm is liquidated and a new firm is regenerated in its place. The firm liquidation process has much in common with the commercial bank bailin procedure and involves two steps: stock elimination and loans consolidation.

After liquidation the stock of the bankrupted firm will have zero value. All shares issued by the firm are terminated and the firm stock instrument is erased at the exchange. Existing shareholders incur capital losses (equity shocks) as a result of this erasure. All of the liabilities of the liquidated form are terminated and a new firm is regenerated. The newly regenerated firm inherits the balance sheet of the liquidated firm, so the equity of the regenerated firm is equal to the total asset value of the bankrupt firm. New shares in the regenerated firm are issued to the creditors of the liquidated firm at no cost to its creditors. Each creditor receives a number of shares proportional to the value of the debt owed to it by the liquidated firm.

The liquidation/regeneration cycle is therefor a loans-to-equities conversion. The balance sheets of the lenders are modified. The lender may thereafter be required to change its portfolio composition, triggering a reaction on the markets. Creditors are likely to suffer capital losses due to the difference in value of deleted loan assets compared to newly issued stocks.

The stock price of the newly regenerated firm is $E/n$ where $E$ is the equity of the new firm (the total asset value after bakruptcy resolution of the liquidated firm) and $n$ is the number of shares to issue. In the standard model, stockholders are typically commercial banks and mutual funds. Depending on the financial power of mutual funds as compared to banks, it may be the case (unlike at time $t = 0$) that the dominant initial owners of the regenerated firm are funds.

### 4.4.8 Balance Sheet

The industrial subeconomy consists of $N > 0$ firms labelled $F_1, F_2 \ ... \ F_N$. Each firm has an individual balance sheet in which assets and liabilities are recorded. The balance sheet of firm $F_i$ is:

| Assets | Liabilities |
|---|---|
| $A_{i0}, A_{i1}, A_{i2} \ ...$ | $L_{i0}, L_{i1}, L_{i2} \ ...$ |

where $A_{ij}$ is the $j$th asset of firm $i$ and $L_{ik}$ is the $k$th liability of firm $i$. Note that either column may be empty (nil assets and/or nil liabilities). Write $A_i$ for the total asset value $\sum_j A_{ij}$ and $L_i$ for the total liability value $\sum_j L_{ij}$. The firm equity is written $E_i = A_i - L_i$. Each element of the balance sheet is potentially a time-dependent quantity: $A_{ij} = A_{ij}(t), L_i = L_i(t)$ and so on.

Balance sheets are data structures, continuously updated depending on the actions of the firms. When firms buy input goods/services the value of the capital (if any) is recorded as an inventory asset on the balance sheet. After production, spent input goods are deducted from inventory assets and newly produced goods are added in their place. Note that production is not necessarily a constant equity process, as value is added when input goods are converted to production goods. Firm liabilities include commercial loans, which are recoded at the instant new loans are taken out by the firm.

### 4.4.9 Accounting

The assumption of the standard model is that commercial banks buy all shares in newly created firms. Banks are the initial owners of firms.

At time $t = 0$ it may be helpful to give newly created firms more cash than the value of their initial stock emission. This is because the simulation is meant to appear at time $t = 0$ as though it had existed at earlier times. The creation of the firm therefor involves shareholder cash as well and (optionally) supplementary cash creation from exogenous sources. If a firm is created at any subsequent time $t > 0$, all cash should be raised via endogenous sources (stock emissions) to avoid money creation. Banks may in due course sell their shares other stock-trading agents, meaning that mutual funds and/or other non-banking agents may eventually become the owners of firms.

Once new firms begin production and sell their goods/services via the market, the firm balance sheet will be continuously updated. In particular since the firms produces goods in business cycle $\lfloor t \rfloor$ and sells these goods in cycle $\lfloor t \rfloor + 1$, the assets side of the balance sheet will include the value of the produced good

| Assets | Liabilities |
|---|---|
| $C_i^E + C_i^S$ | nil. |

Table 2.1: A firm balance sheet at time $t = 0$. $C_i^E$ is the initial firm cash endowment not sourced from shares. $C_i^S$ is the cash paid by commercial banks for the initial firm share emission. If a firm is created at time $t > 0$, $C_i^E$ should be equal to zero in order to avoid money creation.

evaluated at the ask price.

| Assets | Liabilities |
|---|---|
| $C_i$ | $L_i$ |
| $p_i q_i$ | |

Table 2.2: A firm balance sheet at time $t > 0$. $C_i(t)$ is the size of the firm cash reserve (deposits and any other liquid holdings). $L_i$ is the total firm liability, which is likely to include commercial loans of various maturities and interest rates. $p_i$ is the firm ask price per unit goods to be sold (the minimum selling price) and $q_i$ is the stock of saleable goods (inventory).

Any inflow and outflow of money and relative changes of the values in the balance sheet are recorded.

#### 4.4.10 Observations and Desirable Features

Macroeconomic business cycles in the simulator are pronounced. With appropriate initial conditions and configuration, heterogeneous commercial loan interest rates, firm equities, and the value of dividend streams are cyclical in time. The amplitude of these endogenous business cycles can be substantial. By default the behaviour of the simulator is that mutual fund agents invest exclusively in firm and bank stocks (Ie. funds do not invest in bonds or government gilts as an alternative to stocks, although this feature can be enabled) and commercial banks have the option to invest in stocks or commercial loans. Commercial banks do not, by default, buy stock in other commercial banks due to professional competitiveness.

Commercial loans are formed via a clearing market in which aggregate demand and relative aggregate supply, not the agents themselves, govern the interest rate. No agent in the system decides the interest rate for commercial loans, but rather, the commercial loan interest rate is as a result of the sum of all agent behaviours in the model. One future aim is to use a base model with an endogenous business cycle and, by extending this base model, to investigate how the business can be affected by agent decision rules, by financial regulation, by macroeconomic structures such as Input-Output (IO) networks and by other

additions and modifications. We consider it to be of key interest if we find that by extending this base model with reasonable and interesting new behaviour we discover that commercial loan interest rates are less variable and that the endogenous business cycle is suppressed. This work is ongoing at the Catholic University of Milan.

Fundamentalist stock return rates are less variable than commercial loan interest rates in the model. More diverse bank investment options (for instance, mortgages) have been in development but are still not available at the time of writing. In the interests of realism we consider it to be important that commercial loans are heterogeneous and that firms should exhibit reluctance to modify their dividend payments, as is known to be the case in real economies[7]. We consider risk and reluctance to be important to commercial bank loan activities and we intend to examine this direction. We ultimately consider it to be important that bank investment options are diversified.

Although the IO network is an obvious candidate for study, we do expect that the IO network is likely to exacerbate existing endogenous business cycles and contagion events rather than stabilising these behaviours. We note that we have a variety of different IO networks at our disposal. By default the IO network structure employed in the model is an all-to-all (namely, a complete graph) Cobb-Douglas network and, owing to its flexibility, this structure is likely to be the most stable configuration. Notwithstanding this, we now briefly consider what aspects of the IO structure we might like to investigate in future:

- Degree Distribution Networks (DDNs). DDNs are IO networks for which the edge set of each vertex is randomly generated. The number of edges connected to each vertex is drawn from a distribution. DDNs are thought to be reasonable and realistic representations of real economies, and we do in future intend to introduce this as standard for the simulator. The Pareto Distributed Degree Network[8] is the most interesting such structure and we intend to add this. We are of the opinion that the capabilities of the codebase are more than adequate for this feature.
- Production Chains. Production chain networks are special types of IO network in which a distinguished group of sectors are identified as root producers (namely, manufacturers whose outputs are fundamental to the whole economy). All other sectors in the IO network are ultimately dependent on the production activities of the root producers. Root producers sell primitive (Ie. unrefined) products that are bought by intermediate firms, which, for the sake of simplicity, we will call wholesalers. Wholesalers in turn buy from other groups of wholesalers until the last refinement step in the chain: a distinguished group of sectors, which, for the sake of simplicity, we will call retailers. Household consumption is limited to retailer firms. This model is currently in a proof of concept state with simplified agent decision rules.
- We intend to introduce a non-IO type firm production function (no input goods) of the form $P = Z \cdot L^a$ where $Z$ is the total factor productivity, $L$

[7] Brav. et al., 'Payout policy in the 21st century'
[8] Acemoglu et. al., The Network Origins Of Aggregate Fluctuations

is labour employed by the firm, $a$ is a customizable fixed exponent and $P$ is production. $Z = Z(t)$ should be subject to a random walk or lognormal random walk or discrete Ornstein Uhlenbeck (AR1) process in $t$. This has been completed.

We also note that:

- The tax setup in the existing model is such that government taxation on inter-firm input goods settlements does not result in firm tax credits. For this reason, government taxation on goods cannot quite be referred to as VAT in the model. We should consider means through which we can remove tax on input goods settlements whilst preserving the tax paid on consumption goods settlements. Although more realistic, we expect that this change will not significantly affect the model.

- Introducing buffer-stock theory for household consumption would be of interest.

- Financial regulation by the central bank agent in the model is currently very indirect. The central bank agent provides repo loans (typically fixed-rate interest loans collateralised by firm stocks) and, optionally, uncollateralized cash injection loans (UCILS) depending on the lending policy of the central bank. We note that, although they are available for use, the model does not by default contain gilts, and that gilts and government bonds are tools actively utilised by central banks to regulate economies. Without gilts moreover the central bank acts as a slow money sink with respect to repo interest payments, because the interest received by the central bank on commercial bank repo loans does not reenter the economy. For this reason in consultation with Southampton we think that the model should, at least temporarily, be structured in such a way that central bank profits are transferred to government and that the lender of last resort should not exist in the simulation unless the government agent is also present. This has been done at the time of writing.

- More attention should be given to inflation and financial regulation generally in the model.

- The model offers an AR1 TFP process with customizable standard deviation. We tested this feature and noted that for low AR1 TFP variance the system evolved to a stead state equilibrium, for moderate AR1 TFP variance the system generated mild firm insolvency issues resolved by stock redistributions and late debt payments, and that for higher AR1 TFP variances the system generated firm liquidations and bank bankruptcies (bailouts or bailins).

- Stock market capitalisation by commercial banks is too high by default in the existing model. The capitalisation of the largest UK banks is around 16% of the market but this figure is exceptionally high when compared to non-UK eurozone states and the US. This issue is linked to commercial bank PAYE and workforce expenses (or, more precisely, the lack thereof in our model).

Labour expenses are high for UK high street banks; in the case of Barclays and Santander, labour wages have at times accounted for roughly 50 pence per pound of all operating expenses[9] [10]. The issue of commercial bank payroll modelling has been discussed extensively. A proposed but as yet undecided relationship $L = L(P)$ between the target bank portfolio size $P$ and required workforce $L$ would seem to be needed to ensure that shortfalls in bank labour translate into reduced bank activities. This modelling assumption is however (a) countercyclical in boom times and would (b) result in forced stock selling activities whenever commercial banks secure unexpectedly small workforces. To solve this issue we intend to increase the interest rate payable on deposit accounts for household deposit holders. The resulting interest rate payable on bank deposit account liabilities in the model should be interpreted as including the cost of premises, labour and other bank operating expenses we do not model directly.

A profitable area of study would be the relationship between central bank interest rates and commercial bank leverage. The connection between collateralised central bank loan interest rates and economic activity is well established and important to state economies, particularly the UK economy.

## 4.5 Mutual Funds

The primary motivations for introducing funds are: (a) to provide a vehicle to trade shares owned by households and (b) to reduce the overall stock trading activity of commercial banks. In the legacy model commercial banks were the only investors in the stock market, whereas in the standard model funds typically have at least as much financial power as banks in this respect. For reasons for convenience the legacy system involved an unrealistic market structure in which households owned shares in commercial banks and banks owned all shares in firms.

In the standard model commercial bank participation in the stock market is optional (and enabled by default). Commercial banks and other firms are both owned by households, but with investment decisions delegated to funds. Households partition their savings between their deposit accounts and a mutual fund investment accounts. Each household agent invests in one and only one mutual fund and households will be able to periodically revisit their fund choice. Funds will be able to invest in bank stocks and firm stocks and they are able to lend to banks and/or government via bonds.

Pension funds and insurance companies are large players in the UK financial system and these agents play significant roles in stock and bond markets. Funds are non-banking establishments that provide financial services and investment opportunities to clients. In the standard model, households are the only such clients[11] and all fund assets are owned by households. The terms 'mutual fund'

[9] `http://reports.barclays.com/ar12/financialstatements/consolidatedfinancialstatements/incomestatement.html`

[10] `http://www.santanderannualreport.com/2013/en`

[11] this restriction is not strictly necessary.

and 'fund' are interchangeable in the parlance of the standard model. Pension funds and repo–loan issing funds are planned but not yet implemented.

Funds aim to: (a) maximize returns on existing assets and (b) balance liquid assets versus illiquid assets in such a way that client withdrawal requests can always be satisfied. Client–fund interactions are characterized as 'contributions' if the client intends to make a new or supplementary investment and as 'withdrawals' if the client wishes to liquidate an existing investment.

At this point it would be beneficial to outline the similarities and differences between (a) `Deposit Account` assets §4.5.1 and (b) `Investment Account` §4.5.2 assets.

### 4.5.1 Deposit Accounts

Deposit accounts are bank (deposit holder) liabilties and client (depositor) assets. Deposit accounts are implemented as instant-access synchronous transferable balance sheet contracts with no withdrawal fees. Deposit account transactions are underwritten by the bank cash reserve if a depositor requires a cash withdrawal. A deposit account transaction can fail as a result of the depositor having a low account balance or as a result of the deposit holder having insufficient liquidity. Overdraft states are not currently allowed and interest is (optionally) payable on deposits. Depending on the bank bankruptcy resolution policy, deposit accounts may or may not be protected by financial regulations: commercial bank bailins will (in the event that the bank failure is sufficiently severe) result in deposit accounts being written down, whereas bank bailouts will not.

#### 4.5.1.1 Interest Rates

Interest (whether zero or otherwise) is due on all deposit accounts once per simulation cycle at time `DEPOSIT_PAYMENT` in the standard model. Deposit account interest payments are not cash transactions. It is possible for commercial banks to enter negative equity states as a result of interest payments.

#### 4.5.1.2 Background, and the Importance of Deposit Interest

The original 'Mark One' model §3 and early iterations of the standard model §4 were subject to unrealistic stock market capitalization dynamics. In the standard model commercial bank agents do not[12] employ a workforce to deliver financial services in the same way that firm agents employ (and depend on) labour for production. Labour wages are a substantial fraction of all firm expenses[13]. For this reason aggregate commercial bank profits have the potential to be far larger than aggregate firm profits and, as cumulative bank dividends reflect cumulative bank profits, the long-term fundamentalist return rate (stock dividend attractiveness) of commercial banks has the potential to be far higher than for firms. In turn this means that the expected long-term total value of

[12] this restriction is actively discussed and is subject to change.

[13] neglecting mutual fund capital gains, domestic consumption is entirely financed by labour wages.

bank stocks is expected to be significantly higher than firm stocks in the standard model, whereas in reality the combined market capitalizations of major UK banks is roughly 17% of the market, and this figure is exceptionally high in the UK as compared to other eurozone states.

In an effort to extend the legacy model to produce a simulation with (a) reasonable aggregate commercial bank stock market capitalisation, (b) increased bank operating expenses and (c) lower bank margins, we began by inspecting the deposit account code itself and noted that the following logic was responsible for the interest rate payment:

```
@SuppressWarnings("unused")   // Scheduled
private void payInterest() {
   if(!isTerminated) {
      final double
         interestToPay = getValue() * getInterestRate();
      if(interestToPay == 0.0)
         return;

      try {
         getDepositHolder().decreaseCashReserves(interestToPay);
      } catch (final LiquidityException e) {
         getDepositHolder().cashFlowInjection(interestToPay);
         try {
            getDepositHolder().decreaseCashReserves(interestToPay);
         } catch (final LiquidityException e1) {
            throw new IllegalStateException(e1);
         }
      }
      super.incrementValue(interestToPay);

      Simulation.enqueue(interestPaymentEvent);
   }
```

The above effectively deducts the interest payment from the deposit holder twice: the first deduction is a cash deduction from the bank cash reserve (namely, a bank asset) and the second deduction is as a result of the increase in the value of the deposit account (namely, a bank liability). Is is not necessary to make a deduction from the bank cash reserve as the interest payment is not a cash transaction, and moreover the change in the value of the deposit account affects only the deposit holder and the depositor directly. It is also the case that no cash flow appeals need to be made by the bank.

The deposit account framework has accordingly been modified. It is not the responsibility of the deposit account to select the interest rate payable on existing deposits: as it stands, it is only the responsibility of the deposit account to increase the liability of the provider at regular intervals. This means that the interest rate payable on deposit accounts is a decision to be made by the bank,

and this decision rule should ideally have multiple interchangeable implementations.

We are content to offer two very simple deposit account interest rate decision rules for the time being. The first implementation is a trivial "no interest" decision rule which is equivalent to the legacy behaviour. The second implementation is a fully customizable decision rule in which the interest rate payable on a deposit account is a function of both the simulation time $t$ and the type (eg. household, mutual fund, firm) of the deposit holder:

```
/**
  * A lightweight decision rule interface for {@link DepositAccount} interest rate selection
  * algorithms.<br><br>
  *
  * Implementations of this interface provide financial decisions about the interest rates
  * payable on {@link DepositAccount} objects. The {@link DepositAccount} for which to decide
  * the interest rate is provided as an argument, and the chosen interest rate is applied
  * to it.<br><br>
  *
  * See also {@link
      DepositAccountInterestRateDecisionAlgorithm#setInterestRate(DepositAccount)}.
  *
  * @author phillips
  */
public interface DepositAccountInterestRateDecisionAlgorithm {

   /**
     * Set the interest rate for the specified {@link DepositAccount} object. The argument
     * must be non-<code>null</code>. Implementations of this interface need not make the
     * interest rate decision during this method call; the interest rate decision could be
     * made at another time in a dedicated scheduled event. The implementing class should
     * document the timing of the decision process.<br><br>
     *
     * The {@link DepositAccount} in the argument will be modified only by adjusting its
     * interest rate. No other modifications to the {@link DepositAccount} will be made
     * by this method.
     *
     * It is expected that this method should be called once for each {@link DepositAccount},
     * as required, before {@link NamedEventOrderings#DEPOSIT_PAYMENT}<code>(2)</code> in the
     * simulation cycle.
     */
   public void setInterestRate(DepositAccount account);
}
```

The above interface was considered to be the simplest possible contract for a deposit account interest rate decision algorithm. The responsibility of the implementing class is to modify the interest rate of the argument (a deposit account).

The trivial implementation of this decision rule simply returns zero for all stimuli. The alternative implementation is as follows. This implementation delegates the interest rate decision $r$ to a predetermined timeseries $r(t)$ supplied by the user:

```
package eu.crisis_economics.abm.bank.strategies;

import com.google.common.base.Preconditions;
import com.google.inject.Inject;
import com.google.inject.name.Named;

import eu.crisis_economics.abm.contracts.DepositAccount;
import eu.crisis_economics.abm.contracts.Depositor;
import eu.crisis_economics.abm.firm.Firm;
import eu.crisis_economics.abm.fund.Fund;
import eu.crisis_economics.abm.household.Household;
import eu.crisis_economics.abm.model.parameters.TimeseriesParameter;

/**
  * A simple customizable implementation of the {@link
      DepositAccountInterestRateDecisionAlgorithm}
  * interface.<br><br>
  *
  * This implementation delegates the {@link DepositAccount} interest rate selection decision
  * to an instance of a {@link TimeseriesParameter}. One such {@link TimeseriesParameter}
      must
  * be provided for the three primary types of {@link DepositAccount} {@link Depositor}:
      namely,
  * {@link Household}{@code s}, {@link Firm}{@code s}, {@link Fund}{@code s}, and one further
  * {@link TimeseriesParameter} for all other types of {@link Depositor} not falling into the
  * above categories.<br><br>
  *
  * See also {@link CustomInterestRateDecisionAlgorithm#CustomInterestRateDecisionAlgorithm(
  * TimeseriesParameter, TimeseriesParameter, TimeseriesParameter, TimeseriesParameter)} and
  * {@link DepositAccountInterestRateDecisionAlgorithm}.
  *
  * @author phillips
  */
public final class CustomInterestRateDecisionAlgorithm
   implements DepositAccountInterestRateDecisionAlgorithm {

   private TimeseriesParameter<Double>
      householdInterestRate,
      firmInterestRate,
      fundInterestRate,
      otherAgentInterestRate;

   /**
     * Create a {@link CustomInterestRateDecisionAlgorithm} object.<br><br>
```

```
  *
  * See also {@link DepositAccountInterestRateDecisionAlgorithm}.
  *
  * @param householdInterestRate <br>
  *        A {@link TimeseriesParameter} whose values define the interest rate payable
  *        on all {@link Household} {@link DepositAccount}{@code s} manipulated by this
  *        object at time <code>t</code>.
  * @param firmInterestRate <br>
  *        A {@link TimeseriesParameter} whose values define the interest rate payable
  *        on all {@link Firm} {@link DepositAccount}{@code s} manipulated by this
  *        object at time <code>t</code>.
  * @param fundInterestRate <br>
  *        A {@link TimeseriesParameter} whose values define the interest rate payable
  *        on all {@link Fund} {@link DepositAccount}{@code s} manipulated by this
  *        object at time <code>t</code>.
  * @param otherAgentInterestRate <br>
  *        A {@link TimeseriesParameter} whose values define the interest rate payable
  *        on all {@link DepositAccount} objects not held by {@link Firm}, {@link Fund} or
  *        {@link Household} {@link Depositor}{@code s} at time <code>t</code>
  */
@Inject
public CustomInterestRateDecisionAlgorithm(
@Named("HOUSEHOLD_INTEREST_RATE")
   TimeseriesParameter<Double> householdInterestRate,
@Named("FIRM_INTEREST_RATE")
   TimeseriesParameter<Double> firmInterestRate,
@Named("FUND_INTEREST_RATE")
   TimeseriesParameter<Double> fundInterestRate,
@Named("OTHER_AGENT_TYPE_INTEREST_RATE")
   TimeseriesParameter<Double> otherAgentInterestRate
   ) {
   this.householdInterestRate = Preconditions.checkNotNull(householdInterestRate);
   this.firmInterestRate = Preconditions.checkNotNull(firmInterestRate);
   this.fundInterestRate = Preconditions.checkNotNull(fundInterestRate);
   this.otherAgentInterestRate = Preconditions.checkNotNull(otherAgentInterestRate);
}

@Override
public void setInterestRate(DepositAccount account) {
   final Depositor
      depositor = account.getDepositor();
   if(depositor instanceof Household)
      account.setInterestRate(householdInterestRate.get());
   else if(depositor instanceof Firm)
      account.setInterestRate(firmInterestRate.get());
   else if(depositor instanceof Fund)
      account.setInterestRate(fundInterestRate.get());
   else
      account.setInterestRate(otherAgentInterestRate.get());
}
```

```
}
```

Bank deposit account liabilities in the standard model are dominated by household and mutual fund deposit holders[14]. We intend to increase the deposit account interest rate as a substitute for unmodelled operating expenses (bank workforce, premises) in such a way that the increased interest paid on deposits should return to households. For this reason we elected not to increase the interest rate payable on firm deposit accounts. We are also mindful that the simulation is subject to a transient period: the deposit account interest payment (which does not depend on bank profit margins) should be initialised after the transient period has passed.

The standard nontrivial nonadaptive bank deposit account interest rate decision rule is therefor as follows:

- The interest rate paid by banks on household deposit accounts is $r(t) = \frac{1}{2} r_H \cdot (\tanh[(t - t_E)/w] + 1)$, where: $r_H$ is a fixed long-term interest rate to be paid to household depositors, $t_E > 0$ is a (post-transient) timescale at which the interest rate payments should begin, and $w > 0$ is the length of the time window around $t_E$ in which to smoothly increase the interest rate from 0 to $r_H$. We found $r_H = 0.05$ (5%), $t_E = 500$ and $w = 50$ to be convenient parameters.
- The interest rate paid for mutual fund deposits is as the above expression with $r_M$ (a long-term fund deposit interest rate) substituted for $r_H$. We again found $r_M = 0.05$ (5%), $t_E = 500$ and $w = 50$ to be convenient parameters.
- The interest rate payable to firms and other types of deposit holder is zero.

### 4.5.2 Investment Accounts

Investments accounts differ from deposit accounts in that withdrawals are not necessarily immediate, allowed, or without fees. The change in value of an investment account reflects the performance an underlying financial product that has been bought, and the value of an investment account can decrease if the performance of the asset was poor.

In the standard model all investment accounts are currently managed by mutual funds. The value of an investment account is the value of the proportion of total mutual fund assets claimed by the account. As the value of mutual fund assets changes every simulation cycle, so does the value of all investment accounts.

Concretely: if a mutual fund agent has exactly two investment account liabilities of value £0.50 each and the fund equity is zero, then assuming no further investment accounts are opened or closed in future, thereafter the value of each investment account is $A/2$ where $A(t)$ is the combined value of all assets owned

[14] mutual funds make financial investments on behalf of households

by the fund at time $t$. Investment accounts correspond to shares in the fund, with the price per share of the fund being $A(t)/n(t)$ where $n(t)$ is the total number of fund shares in existence at time $t$. Note that $n(t)$ is arbitrary.

### 4.5.3 Withdrawal Planning Problem

Many markets in the standard model are synchronous clearing markets §2.2.2. Trades on synchronous markets are processed at specific times in the schedule (eg. $t \bmod 1 =$ `CLEARING_MARKET_MATCHING`), not continuously ('on-demand'). Funds must wait until market processing in order to liquidate existing illiquid portfolio assets, so funds are subject to forward planning problems with uncertainty about their future performance on the market.

Note that funds do not know their clients' future withdrawal plans. Clients are free to request withdrawals and/or make contributions at any time in the simulation cycle. For this reason the behaviour of mutual fund agents in the standard model is as follows:

1. If a client makes a new contribution to a fund (and can affort the proposed investment) then the transaction takes place immediately. This is a constant equity process for both the client (who transfers cash to the fund in exchange for an equivalent value in fund shares) and for the fund (who recieves cash from the client and increases the value of its investment account liabilities).

2. If a client requests a withdrawal, this is not processed immediately. The withdrawal is executed only after markets in which the fund is involved have been processed (in order to give the fund an opportunity to raise liquidity to satisfy the withdrawal).

3. If after the markets have been processed the fund still has insufficient liquidity to honor all client withdrawals, the available fund liquidity is rationed homogeneously among withdrawees. Concretely: if two clients request a withdrawal of £1 each and the fund has £1 in liquidity after market processing then both clients will recieve £0.50 (and may apply to withdraw the remainder in subsequent business cycles).

The fund must monitor the size of its cash reserve to ensure that it does not form illiquid assets with any cash set aside to fulfill withdrawals.

### 4.5.4 Portfolio Planning Problem

A fundamental decision the fund must make is the following: given that the intended illiquid asset portfolio size is $£C \geq 0$, in what ratio should $C$ be distributed among possible instruments? Equivalently: for a set of instruments $\mathcal{I} = \{I\}$ with expected returns $\{\mathbb{E}[r(I)] : I \in \mathcal{I}\}$, how should a fund portfolio of size $C$ be divided among $\mathcal{I}$?

Further questions must be answered:

1. How should $\{\mathbb{E}[r(I)] : I \in \mathcal{I}\}$ be computed?

2. How should $P$ be distributed given (1)? (what are the weights $w(I)$ such that the planned investment in instrument $I$ is $w(I)P$?)

In answer to (1): for stocks the estimated return on rate is precisely the same calculation used by chartist and fundamentalist banks (as outlined in §2.2.1 and detailed in §5.7). For bank bond and government gilt investments (including extended maturity implementations of the same) the expected return rate $\mathbb{E}[r]$ is taken to be the trade-weighted mean yield-to-maturity (book yield) among all contracts formed in the most recently observed business cycle:

$$\mathbb{E}[r] = \frac{\sum_{1 \leq i \leq |\mathcal{V}|} r_i v_i}{\sum_{1 \leq i \leq |\mathcal{V}|} v_i},$$

where $\mathcal{V}$ is the set of all such contracts formed in the last business cycle, $v_i$ is the principal value of the $i$th element of $\mathcal{V}$, and $r_i$ is the yield to maturity of the $i$th element of $\mathcal{V}$. Note that this formula assumes $\mathcal{V}$ is nonempty. If $\mathcal{V}$ is empty then the same estimate is formed for the next-most-recent nonempty set of contracts $\mathcal{V}$'. For other types of investment (namely, non-loan and non-stock instruments) the user must specify the means by which $\mathbb{E}[r]$ is calculated.

Write $r$ for $\mathbb{E}[r]$ for the remainder of §4.5.4. The algorithm used to compute $\{w(I)\}$ given $\{r(I)\}$ (and potentially some other supplementary measurements as well as $r$) is referred to interchangeably as the 'portfolio allocation strategy', the 'portfolio weighting strategy' or the 'investments decision rule'. The standard implementations of the portfolio allocation strategy include but are not limited to:

1. **Fixed Distribution**
   Arguably the simplest allocation rule (§4.5.4.1). When using this portfolio allocation rule, all instruments receive the same priority regardless of risk and regardless or actual or expected return rates.

2. **Fixed Loan Weight**
   A simple allocation strategy in which all instruments deemed to be 'outgoing loans' (eg. bank bonds) receive a fixed weight $w_L$ and all other types of instrument receive a (different) weight $w_{\neg L}$ (§4.5.4.2).

3. **Linear Adaptive**
   An adaptive (incremental) allocation strategy in which $w(I)$ is increased or decreased according to the expected performance of $I$ as compared to the mean performance of all other instruments. This allocation strategy can result in $w(I) = 0$ in a finite number of business cycles (§4.5.4.3).

4. **Exponential Adaptive**
   A modified formulation of the Linear Adaptive allocation strategy. This implementation has the property that $w(I) \longrightarrow 0$ but remains nonzero if $I$ performs poorly over a prolonged period. If $I$ has performed poorly for $n$ business cycles then $\approx n$ business cycles of good performance are required to restore confidence in instrument $I$ (§4.5.4.4).

5. **Logit**
   An efficient closed-form non-adaptive (non-incremental) allocation strategy (§4.5.4.5).

6. **Noise**
   A noisy portfolio investment decision rule for which each $w(I)$ is determined by a mean reverting random process. Funds implementing this decision rule rule are called Noise–Trader Funds or NTFs. See §4.5.4.6.

#### 4.5.4.1 Fixed Distribution

Arguably the simplest implementation of a portfolio allocation strategy. For a portfolio considering $N$ possible instruments, the decision of this strategy is to assign weight $w = 1/N$ to each instrument.

#### 4.5.4.2 Fixed Loan Weight

A minor generalization of the fixed distribution portfolio strategy §4.5.4.1. For a portfolio considering $N$ instruments of which $M \leq N$ such instruments are deemed to be 'outgoing loans' (eg. bank bonds, commercial loans), the decision of this strategy is to assign weight $w_L = \xi/M$ to outgoing loan instruments and $w_{\neg L} = (1 - \xi)/(N - M)$ to all other instruments[15].

#### 4.5.4.3 Linear Adaptive

This portfolio weighting algorithm describes an investor who will in due course favour the instruments with highest returns. This portfolio weighting algorithm will gradually increase weights for instruments whose return rates are higher than the group average and will gradually decrease weights for instruments whose return rates are below the group average.

Once per business cycle the behaviour of the linear adaptive portfolio weighting algorithm is as follows:

1. For each instrument $I$ known to the portfolio, the return rate $r(I)$ is queried and stored in memory.

2. The mean instrument return rate $\overline{r} = \sum_{I} r(I)$ is computed.

3. The existing portfolio weight $w(I)$ assigned to instrument $I$ is updated as follows:
$$w(I) \mapsto \max(w(I) + \epsilon \cdot (r(I) - \overline{r}), 0)$$

4. Each weight $w$ is renormalized according to:
$$w(I) \mapsto \frac{w(I)}{\alpha},$$
where $\alpha = \sum_{\mathcal{I}} w(\mathcal{I})$ is the sum of all investment weights immediately after step (3).

[15] $w_{\neg L}$ is taken to be zero if $M = N$.

The parameter $\epsilon$ is called the 'adaptation rate' and is a model parameter in the meaning of §4.3. $\epsilon \geq 0$ is required in order for the algorithm to behave as expected.

The fundamental motivation for the linear adaptive weighting algorithm is that return rates are expected to diminish as overall investment increases. For instance if the total investment in commercial loan instruments is high then the availability of commercial loans is high, competition for loans is low, and on average the interest rate realized on each commercial loan is low. Contrariwise if the commercial loan interest rate $r$ is high then not only does the instrument represent an attractive investment opportunity (and merits an larger fraction of the portfolio) but an increased investment in the commercial loan instrument is expected to reduce $r$. The linear adaptive portfolio weighting algorithm is therefor expected to reach a dynamic equilibrium state in which the returns on all instruments are similar. Not that in such an equilibrium it is not necessarily the case the the income/profit per unit investment in each instrument is equal: the return rate $r(I)$ is an abstract measurement which may include (eg.) risk premia.

Note that it is possible for the weight $w(I)$ to drop to zero for any instrument with poor returns.

The initial weight configuration (at time $t = 0$) is typically $w(I)_{t=0} = 1/N$ where $N$ is the number of instruments known to the portfolio.

#### 4.5.4.4 Exponential Adaptive

In practice the linear adaptive rule §4.5.4.3 is subject to a number of shortcomings:

1. The adaptation rate $\epsilon$ does not depend on the distribution of returns $r(I)$ and does not depend on the variance (polarization) of existing weights $w$. For a portfolio with heterogeneous return rates/instrument weights, in practice the linear adaptive rule can be subject to a dichotomy in which it is either too slow to adapt or it will systematically overreact to changes in the market. This overreaction by the portfolio can result in disruptive financial oscillations with period 2 cycles. l

2. For instruments whose weights $w(I) \ll 1$ are small the overall change in the size of the investment resulting from $w(I) \mapsto \max(w(I) + \epsilon(\cdot(r(I) - \overline{r}), 0)$ can be large compared to the size of the existing investment. If the system is in a steady state for which all returns $r(I)$ are roughly equal and this state is achieved as a result of keeping one or more weights $w(I)$ small, then this would imply that $r(I)$ is likely to be sensitive to $w$ for some instruments. The linear adaptive rule tends to cause intercycle oscillations as a result of perturbing small weights excessively.

3. The linear rule tends to quickly forget about past instrument performance.

4. Portfolio weights can drop to zero rapidly if one instrument is especially attractive.

None of these shortcomings are necessarily very poor and the linear adaptive scheme may be suitable for some applications. However a more recent portfolio weighting scheme is the exponential adaptive weighting. The exponential weight algorithm is as follows:

1. For each instrument $I$ known to the portfolio, the return rate $r(I)$ is queried and stored in memory.

2. The mean instrument return rate $\overline{r} = \sum_I r(I)$ is computed.

3. The existing portfolio weight $w(I)$ assigned to instrument $I$ is updated as follows:

$$w(I) \mapsto \min(\max(w(I) \cdot (1 + ||\epsilon \cdot (r(I) - \overline{r})\,||_{.99}), \xi), 1 - \xi),$$

   where $||x||_y$ is shorthand for $\max(\min(x, |y|), -|y|)$ and $1 \geq \xi \geq 0$ is a parameter.

4. Each weight $w$ is renormalized according to:

$$w(I) \mapsto \frac{w(I)}{\alpha},$$

   where $\alpha = \sum_{\mathcal{I}} w(\mathcal{I})$ is the sum of all investment weights immediately after step (3).

As for the linear rule §4.5.4.3 the parameter $\epsilon$ is called the 'adaptation rate' and is a model parameter §4.3. $\epsilon \geq 0$ is required in order for the algorithm to behave as expected.

$\xi > 0$ is desirable but not required. The exponential adaptive rule has the property that $w(I)$ cannot drop to zero (other than artificially as a result of finite precision arithmetic) if its initial value $w(I)_{t=0}$ is nonzero. For a portfolio with two instruments $I_1, I_2$ whose return rates $r_1 < r_2$ are fixed, the response of the exponential ruke is to reduce $w(I_1)$ by a factor of $\approx 1 + \frac{1}{2}\epsilon(r_1 - r_2) < 1$ (neglecting renormalization) once per simulation cycle. If after $N$ business cycles have elapsed the values $r_1$ and $r_2$ are swapped then for $\xi = 0$ it will take a further $2N$ business cycles for the portfolio weights $w(I_1), w(I_2)$ to invert. If $\xi > 0$ then the number of business cycles required to invert the portfolio in this way is bounded above as $N \longrightarrow \infty$ because $w(I)$ may not be less than $\xi$.

In practice the exponential rule tends to perform better than the linear rule. Instruments whose portfolio weights are small and whose existing investments are small tend to be perturbed by small amounts and likewise instruments with large existing portfolio weights tend to be perturbed by larger amounts. This has the effect of reducing financial oscillations in the system.

The initial weight configuration (at time $t = 0$) is typically $w(I)_{t=0} = 1/N$ where $N$ is the number of instruments known to the portfolio.

#### 4.5.4.5 Logit

The logit rule is a closed–form nonadaptive portfolio weighting rule. For instruments $I$ with returns $r(I)$ the weight $w(I)$ applied to instrument $I$ is:

$$w(I) = \frac{e^{\beta r(I)}}{\sum_J e^{\beta r(J)}},$$

where $J$ runs over all instruments in the portfolio and $\beta \geq 0$ is a model parameter called the 'intensity of choice'.

In the limit $\beta \to 0$ we have $w(I) \to \frac{1}{N}$ where $N$ is the number of instruments known to the portfolio, and for $\beta \to \infty$ we have:

$$w(I) = \begin{cases} 1 & \text{if } r(I) = \max_J\{r(J)\} \\ 0 & \text{otherwise.} \end{cases}$$

The numerical cost of the logit rule is deceptive. It is not atypical for a logit rule to be connected to a clearing market whose purpose is to identify interest rates $r(I)$ with specific properties by evaluating many possible $\{r(I)\}$. The cumulative cost of evaluating the above exponential expressions can be nontrivial. Particular care should be taken when choosing $\beta$ in the above expression. The return rates $r(I)$ are expected to be of the order $r \sim 10^{-1}$ so $\beta \sim 100$ would result in exponentials of the order $e^{10}$.

#### 4.5.4.6 Noise

Mutual funds using the noisy portfolio weighting rule are called 'noise traders' or 'noise-trader funds' (NTFs). The noisy portfolio rule is a mean–reverting lognormal process in which the weight $w(I)$ applied to each instrument $I$ does not follow the profit incentive, but rather the weight fluctuates randomly.

The internal randomness of the above portfolio rules §4.5.4 is limited. In the standard model funds tend to invest heavily in the stock markets, and for this reason the volatility of the stock markets may be low when using the above rules. It is common for $N$ non-NTF mutual funds to be complemented by $\epsilon N$ $(0 \leq \epsilon \leq 1)$ NTFs in order to increase the volatility of stock prices. Note that for a simulation running with Basel II regulation/procyclical leverage enabled it is expected but not required that at least one NTF be present[16]

The behaviour of the noisy portfolio rule is as follows:

1. For each instrument $I$ known to the portfolio, return rate $r(I)$ is queried and stored in memory.
2. The existing weights $w(I)$ are perturbed as follows:

$$w(I) \mapsto \exp\left(\rho \log(w(I)) + \mathcal{N}(\sigma^2)\right),$$

[16]The procylical leverage features converts the volatility of portfolio instruments into a regulatory upper bound on commercial bank leverage. In the absence of NTFs the volatility of the stock market must be sourced from elsewhere, for instance rational firm decision rules or noisy firm production functions.

where $\mathcal{N}(\sigma^2)$ is a Gaussian distribution with mean zero and variance $\sigma^2$. $0 \leq \rho < 1$ is a customizable model parameter.

3. Each weight $w$ is renormalized according to:

$$w(I) \mapsto \frac{w(I)}{\alpha},$$

where $\alpha = \sum_{\mathcal{I}} w(\mathcal{I})$ is the sum of all investment weights immediately after step (2).

The initial weight configuration (at time $t = 0$) is typically $w(I)_{t=0} = 1/N$ where $N$ is the number of instruments known to the portfolio.

#### 4.5.5 Observations and Desirable Features

The immediate issues requiring attention for the fund agent type are (a) complementary fund implementations and (b) bank bond instruments. Only one fully endogenous implementation of the fund agent type (namely, mutual funds) is provided by the standard model. Bank bonds, including bank bonds of extended maturity, are supported by the standard model however these instruments are disabled by default. The primary investment opportunity for funds is therefor the stock market.

##### 4.5.5.1 Pension Funds

Pension funds share some features in common with mutual funds. Money flows to pension funds via a levy on salaries and only flows out of pension funds after a household (namely, a client) retires. Pension funds are funds whose contributions are derived from labour PAYE. The pension fund receives contributions as a result of deductions made from employee wage packets, as opposed to discretionary investments made by households using surplus cash after consumption. The pension fund is ultimately required to pay households and/or individuals from this funding stream as well as from profits made form investments.

Pension funds present an interesting problem for the codebase because pension funds naturally receive money from employees and pay out to unemployable individuals when their useful labour supply is at an end, which requires the standard model to provide a concept of an employee lifecycle for which penision funds are a natural companion. This in turn is likely to require households to be subdivided into households and individuals: households are static properties whereas individuals are subject to a lifecycle in which their labour supply depands on their age and their entitlement to pension fund support begins at a fixed time in the lifecycle. Under the life cycle model households become economically active at a certain age (when they start working and receiving wages) and eventually take mandatory retirement (when they start receiving a pension).

Mutual funds and pension funds differ in several key respects: firstly unlike mutual funds there is no money flow from pension funds to households before the household retires. Household contributions to pension funds are also mandatory

and these contributions amount to regular cash flow during periods of employment. In contrast to mutual funds, once assigned to a pension fund, households cannot move. These differences may be economically important. Further investigation will be needed in order to determine what happens to leftover pension funds upon the death of the client.

#### 4.5.5.2 Repo–Issuing Funds

Repo loans (repos) are loans supplemented with default insurance collateral. Repos are issued on the contractual understanding that, in the event of the borrower's failure or inability to repay the loan, an agreed collateral asset must be transferred from the borrower to the lender as compensation. The collateral asset need not be liquid. Typically the repo collateral must be a high quality saleable asset (with government gilts being a prime example and, to a lesser extent, stocks in consistently profitable highly regarded firms). In the standard model mutual funds do not require collateral when purchasing bank bonds and funds are treated as any other creditor in the event of commercial bank default.

Some (particularly American) types of fund lend to banks on a collateral repo basis. Commercial banks borrow from funds and make guarantees about existing collateral when doing so. The standard model provides repo loans between the central bank (lender of last resort) and commercial banks by default and it is expected that this feature can be extended.

#### 4.5.5.3 Bond Investments By Default

Bond investments are currently implemented as part of a broad optional market called the BCL system ('bonds gilts and commercial lonas'). There are many markets in the codebase but at present the only market that provides bank bonds is one whose purpose is to clear risk-graded commercial loans, government gilts and bank bonds simultaneously. It is planned to offer an alternative implementation in which each instrument in the BCL structure is realized as a separate dedicated clearing market. In this way it will be possible to allow mutual/pension/repo funds to be involved in bond markets without requiring other classes of agents to issue or invest in government gilts as a prerequisite. This arrangement entails compromises; in particular banks are not necessarily able to efficiently optimize portfolio positions because optimization decision are not a part of the clearing process itself: the portfolio division is a decision made in advance of market processing and not during market processing.

## 4.6 Macroeconomic Households

Households are the root owners of the economy and are the driving force behind consumption in the standard model. Collectively, households have very strong financial power. Households exchange labour services for wages and with these wages they (a) buy goods and services from the market and (b) save money in deposits accounts and in financial investment products.

The maximum amount of labour available per household is a model parameter in the meaning of §4.3. This includes the possibility that the labour supply is

a quantity that changes in time or is linked to other elements of the economy, including the state of the household. By default workers (employees) inelastically supply one unit of labour at a constant nominal ask wage $w$. The household is responsible for formulating the ask price per unit labour whereas the employer (eg. firms, government) is responsible for formulating the bid price. Employee compensation is ultimately selected by the market and is required to lie between the ask and bid wages.

Households are typically implemented as macro (aggregated) agents as the true number of domestic properties in the UK runs into the several tens of million. The default household type in the standard model is the `Macrohousehold` agent. Households are able to supply labour to multiple employers concurrently. The household labour supply is a divisible finite non-transferable non-duplicable quantity and is conserved throughout the simulation cycle. The total labour supply in the system may increase or decrease between successive business cycles as new labour becomes available or as the labour supply shrinks. The labour supply is a household asset not listed on the asset side of the household balance sheet and (can be interpreted as) an employer liability not listed on the liability side of the employer balance sheet.

Households and employers are free[17] to select whatever labour ask price $w$ they please. Labour assets are currently indistinguishable except in terms of wage and contract duration: labour contracts do not have an internal notion of quality, specialization, utility or productivity. Extended labour contracts (contracts lasting for more than one business cycle) are both supported by and tested for use with the standard model however these decision rules are not currently in use by any default agent types. Enabling this feature for use with the `Macrofirm` and `Macrohousehold` types would entail slight modfications to the decision rules of both.

Households are free to hold their deposits with any known commercial bank. The standard model provides facilities for transferring deposit accounts between banks if a deposit holder so wishes. In practice however this feature is underutilized because there is no dedicated decision rule assocated with a household's preferred deposit account provider. This is a potentially area of interest as a major risk factor for troubled UK commercial banks is the possibility of a liquidity crisis caused by a run on existing deposit accounts ('a run on the bank').

### 4.6.1 Legacy Model and Background

In the legacy 'Mark One' model (§3) the amount of labour supplied per household per business cycle was an indivisible integer with value 1, and accordingly in the legacy model there were as many units of labour on the market as there were households agents. Labour was an indivisible asset so at any given time households were either fully employed by a unique employer or were fully unemployed.

In the legacy model households held all shares in all commercial banks and

[17] Regulatory minimum wage is a feature of some firm and household decision rules.

these shared empowered households to exclusively collect the entire dividend stream issued by the financial system. This is an undesirable setup as not only is this structure unreaslistic but it also results in bank shares being withdrawn from households in the event of a commercial bank bailout/bailin. In this eventuality, as households have no ability to trade shares themselves, households are unable to rebuy assets lost to them as a result of the bank bankrupty process, meaning that the system is not permanently described by its initial setup unless bankruptcy and insolvency never occurs in the financial system.

### 4.6.2 Decision Rules

Decisions made by household agents include:

1. **The Consumption Budget**
   Households must decide the maximum amount of cash to set aside for buying new goods/services. The total nonzero sum set aside by the household for this purpose is called the 'consumption budget'. The full consumption budget is never to be exceeded, but is not necessarily fully spent. Possible implementions for this decision rule are illustrated in §4.6.2.1.

2. **Consumption Preferences**
   Once a consumption budget is chosen the household must decide how to divide the budget among goods/services. Goods are differentiated both by the sector from which they are sourced and by their durability characteristics[18]. Households may be motivated by competitive prices, by buffer-stock theory, utility maximization or otherwise.

3. **Investments**
   Households must decide how much of their deposits not spent on consumption should be invested. Money invested in mutual funds is principally illiquid until maturity.

4. **Labour Wage Ask Price**
   Households must decide the ask price per unit labour when seeking employment. Depending on the implementation of the labour market, high wage expectations may result in reduced employment prospects.

5. **Labour Pool Size**
   The size of the labour supply per household is a model parameter in the meaning of §4.3. The labour pool size has the potential to be influenced both by the state of the household and by the state of the economy.

#### 4.6.2.1 Consumption Budget

The household consumption budget rule is the household decision rule governing the amount of cash households are willing to spend, in each business cycle, on goods/services. The productivity and profitability of firms depends strongly on the combined size of domestic consumption budgets, and the GDP of the simulation world is reflected by the cash value of consumption.

[18] goods may provide durable capital and households balance sheets include inventory assets.

The standard model provides the following implementations of the household budget rule:

1. **Consumption Propensity**
   A budget rule describing a power-law relationship between domestic wealth and the size of the consumption budget. This rule is outlined in §4.6.2.1.1.

2. **Fixed Consumption**
   A simple budget rule in which households will spend a fixed amount (without borrowing) whenever deposits suffice. This rule is discussed in §4.6.2.1.2.

3. **Noisy Consumption**
   A budget rule with bounded normally distributed noise. This rule is highlighted in §4.6.2.1.3.

4. **AR1 Consumption**
   A forwarding implementation of the budget decision rule in which one of the above rules is smoothed in time according to a simple memory process. This rule is illustrated in §4.6.2.1.4.

#### 4.6.2.1.1 Consumption Propensity

The 'consumption propensity' household budget rule is used by default for households in the standard model. The value $B$ of the consumption budget decision is:

$$B(\lfloor t \rfloor) = \min(C \cdot W^{\alpha}, d),$$

where $\lfloor t \rfloor$ is the index of the simulation cycle in which the budget decision is made, $C$ is a customizable parameter (the 'consumption propensity') and $\alpha \geq 0$ ('the wealth exponent') is a customizable parameter and $d$ is the value of the unallocated household deposits at the time the budget decision is made.

Note that $\alpha = 1$ is required in order for this decision rule to scale linearly in monetary units.

#### 4.6.2.1.2 Fixed Consumption

The homogeneous consumption budget algorithm is potentially the simplest nontrivial household budget rule. In this implementation the household will set aside a fixed sum $b$ of its deposits for consumption unless the household cannot afford $b$. The final consumption budget is:

$$B = \min(b, d),$$

where $d$ is the value of the unallocated household deposits. The value of $b$ is customizable.

#### 4.6.2.1.3 Noisy Consumption

The noisy consumption budget decision rule is a simple rule for which the underlying budget decision process $B$ is driven by a normal distribution:

$$B(\lfloor t \rfloor) = {}_L|\mathcal{N}(\mu, \sigma^2)|_U,$$

where $\lfloor t \rfloor$ is the index of the simulation cycle in which the budget decision is made, ${}_L|x|_U$ is shorthand for $\min(U, \max(L, x))$ and $\mathcal{N}(\mu, \sigma^2)$ is a normal distribution with mean $\mu$ and variance $\sigma^2$. The values of $\mu, \sigma \geq 0, L$ and $U \geq L$ are customizable. Note that $U = +\infty$, $L = -\infty$ are acceptable.

#### 4.6.2.1.4 AR1 Consumption

The AR1 household consumption budget rule describes a budget process with memory. The size of the household consumption budget ($B$) is exponentially smoothed with reference to a delegate consumption budget decision rule ($B^*$).

This decision rule accepts any budget rule $B^*$ indicated in §4.6.2.1 and yields a new decision $B$ with the following value:

$$B(\lfloor t \rfloor) = B^*(\lfloor t \rfloor) \cdot m + B^*(\lfloor t \rfloor - 1) \cdot (1 - m),$$

where $\lfloor t \rfloor$ is the index of the simulation cycle in which the budget decision is made and $m$ ($0 \leq m \leq 1$) is a customizable parameter describing the memory of the process. In each simulation cycle $B$ is a linear combination of the decision made by $B^*$ and the last decision made by $B^*$. As $B^*$ is required to be positive, so is $B$.

It is acceptable for $B^*$ to itself be an AR1 budget decision rule.

#### 4.6.2.1.5 Desired: 'Permanent Income' Budgets

The AR1 'permanent income' budget strategy is an algorithm that selects the domestic consumption budget $B$ adaptively, as a fraction of household wealth $W$, based on past observations:

$$\begin{aligned}
y_t &= \psi y_{t-1} + (1 - \psi)\frac{Y_t}{\overline{p}_{t-1}}, \\
\overline{p}_t &= \frac{1}{N_S}\sum_{i=1}^{N_S} p_{i,t}, \\
b_t &= \alpha y_t + \beta w_t, \\
B_t &= b_t \overline{p}_{t-1},
\end{aligned}$$

where $y_t$ is the 'real permanent income', $\overline{p}$ is the mean goods price over sectors, $Y_t$ is the nominal income at time step $t$, $N_S$ is the number of sectors,

$\psi\,(0 \leq \psi \leq 1), \alpha$ and $\beta$ are customisable behavioural parameters, and $b_t$ is the 'real consumption budget'.

The permanent income budget algorithm reduces the volatility (variability in time) of domestic consumption, and increases the autocorrelation of budget timeseries.

#### 4.6.2.2 Consumption Preferences

Household (consumer) consumption preferences are the relative demands for goods/services when households go to market. Consumption preferences are a set of cash values assigned to each distinct type of goods/services households can buy. In order to decide how much of each product to buy, a household must first decide how much it is willing to spend in total: this budget decision rule is discussed in §4.6.2.1.

The standard model provides several implementations of the household consumption preferences decision rule. These implementations include:

1. **CES Aggregator**
   A Dixit/Stiglitz–type constant elasticity of substitution consumer utility function. Low prices represent attractive buying opportunities for consumers. This module is illustrated in §4.6.2.2.6.

2. **Homogeneous Consumption**
   Potentially the simplest implementation of the consumer preference decision rule. The homogeneous consumption rule exhibits no biases of any kind and assigns the same weight to all possible buying opportunities. This implementation is outlined in §4.6.2.2.7.

##### 4.6.2.2.6 CES Aggregator

The CES Aggregator rule is a Dixit/Stiglitz–type constant elasticity of substitution consumer utility function with no product quality assessment and no memory. If the market provides $N$ differentiated classes of goods/services with prices $p_1$, $p_2$ ... (per unit) then the value $D_i \geq 0$ a household intends to spend on product $i$ $(1 \leq i \leq N)$ is:

$$D_i = \frac{p_i}{\sum_{i=1}^{N} p_i} B,$$

where $B \geq 0$ is the total value of the household consumption budget (§4.6.2.2.7) and $\rho$ $(0 \leq \rho \leq 1)$ is a customizable model parameter.

In the limit $\rho \longrightarrow 0$ this decision rule is equivalent to homogeneous consumption (§4.6.2.2.7) and in the limit $\rho \longrightarrow \infty$ this rule is equivalent to an extreme price–competitive model in which the product with the lowest price receives the entire budget.

Note that the cash sum $D_j$ must be converted into a quantity $q_j$ of goods/services when ordered from the market, namely $q_j = D_j/p_j$.

#### 4.6.2.2.7 Homogeneous Consumption

The homogeneous consumption preference rule is the extreme limit $\rho \longrightarrow 0$ of the CES Aggregator/Dixit–Stiglitz consumption preference rule §4.6.2.2.6. This implementation is potentially the simplest possible realization of the household consumption preference rule.

For $N$ differentiated goods/services, the cash sum $D_i \geq 0$ a household intends to spend on product type $i$ ($1 \leq i \leq N$) is:

$$D_i = B/N,$$

where $B \geq 0$ is the total value of the household consumption budget (§4.6.2.2.7).

### 4.6.2.3 Restricted Consumption

In some cases it is useful to restrict household consumption to a subset of firms. When studying the macroeconomy the structure of the Input-Output network is important, and for nontrivial networks it may be the case that the macroeconomy is subdivided into root producer firms (who may require large amounts of labour but relatively few input goods for production), intermediate/supply-chain firms who buy inputs from root producers and/or other intermediate firms (and who may require large quantities of input goods and relatively little labour) and retailer firms who sell directory to consumer households. In many economies there is a legal expectation that producers, distributors and retailers should be segregated, and the standard model provides for this configuration.

The standard model has a configuration option that allows the user to specify which of the existing firms a household is allowed to buy from when processing the consumption preference decision rule §4.6.2.2. The structure of the algorithms discussed in §4.6.2.2 is unchanged with the exception that $N$ includes only retailer firms who may sell to households.

### 4.6.2.4 Investment Accounts

Once per simulation cycle households must decide how much cash to invest in/withdraw from funds. The household investment account (held by the fund) has definite value at all times except when the fund is trading and building new illiquid assets via the market[19]. Fund investment accounts are listed on the fund balance sheet as abstract liabilities: funds modify the value of all investment accounts in proportion when their total asset value changes as a result of market activities.

The household investment account decision is as follows: given unallocated deposits of value $d \geq 0$ and an investment account of value $I \geq 0$, decide a flow of funds $\Delta I$ satisfying $-I \leq \Delta I \leq d$ such that the household will apply for the

[19] including when stock prices are adjusted.

value of the investment account to be changed from $I$ to $I + \Delta I$. If $\Delta I < 0$ then money will be withdrawn from the fund investment account and deposited with the household bank. If $\Delta I > 0$ then cash is withdrawn from the household deposit account and transferred to the fund. Note that these restrictions ensure the household cannot enter an overdraft state as a result of a fund investment.

Withdrawals $\Delta I < 0$ are (a) not guaranteed to succeed if the fund has insufficient liquidity and (b) take time to process. New contributions $\Delta I > 0$ are instantaneous (§4.5.2).

The standard model provides the following implementations of the household investment account contribution rule:

1. **Withdrawal Threshold**
   This decision rule will attempt to maintain a fixed investment account value $I^*$. If the instantaneous value of the fund investment account is $I < I^*$ then the household will invest a fraction of its unallocated deposits into the account in order to top it up. If the value of the fund investment account is $I > I^*$ then the household will withdraw a fraction of the excess cash $(I - I^*)$. This decision rule is illustrated in §4.6.2.4.8.

2. **Zero Investment**
   This decision rule is perhaps the simplest possible investment account decision rule. This module describes a forgetful household that has no desire to invest additional cash in funds and never applies to withdraw existing money form the fund. This implementation is outlined in §4.6.2.4.9.

#### 4.6.2.4.8 Withdrawal Threshold

This module requires the household to choose a target investment account value $V \geq 0$. Once $V$ is selected, the steps taken by the household when formulating $\Delta I$ (§4.6.2.4) are as follows:

1. If the value $I$ of the investment account is greater than $V$ then the household will apply to withdraw $\Delta I = \epsilon_+(I - V)$ units of cash from the fund, where $0 \leq \epsilon_+ \leq 1$ is a model parameter describing the rate of withdrawal of excess investments.

2. If the value $I$ of the investment account is less than $V$ then the household will contribute $\Delta I = \min(\epsilon_- D, V - I)$ units of cash, where $D \geq 0$ is the value of unallocated household deposits and $0 \leq \epsilon_- \leq 1$ is a model parameter describing the rate of investment of unallocated deposits.

The result of this process is that the household will attempt to maintain an investment account of value $V$ by withdrawing excess investments and by contributing a fraction of its unallocated deposits to the fund when the value of the account is low.

Note that $\epsilon_- = \epsilon_+ = 1$ describes a process with no contribution/withdrawal smoothing. Investments and withdrawals in this configuration are very abrupt and can induce fund liquidity crises.

#### 4.6.2.4.9 Zero Investment

The 'zero investment' contribution rule is perhaps the simplest possible household investment account decision rule. In this implementation the household will select $\Delta I = 0$ no matter what (Ie. the household will never attempt to modify the value of the investment account).

This decision rule describes a model in which all changes to the value of the household investment account (if the value of the investment account is nonzero) are as a result of profits or losses made by the fund.

### 4.6.2.5 Labour Wage Ask Price

Households must gauge the value of their labour by formulating an ask price $w \geq 0$ per unit labour. A household is free[20] to propose any level of remuneration for its labour services, however a high wage proposal $w$ is likely to reduce the household employment prospects. Depending on the implementation of the labour market and the mechanism used by the market to penalize households with high wage demands, households may need to balance their wage threshold against the likelihood of finding any work.

The underlying wage decision made by the household is called the 'wage ask price decision rule'. The standard model provides several possible implementations as follows:

1. **Fixed Nominal Wage Expectation**
   Perhaps the simplest possible household wage ask price decision rule, this implementation causes the household agent to post a fixed nominal wage ask price $w$ not depending on time and not depending on the state of any other entity in the economy. This module is descibed briefly in §4.6.2.5.10.

2. **Wage Confidence**
   This experimental implementation describes a households whose confidence in its ability to command high wages increases with experience. Wage ask prices steadily rise when the household is in employment, until unemployment, at which point the household loses confidence in its ability to command high wages and reduces its ask price. This implementation is outlined in §4.6.2.5.11.

#### 4.6.2.5.10 Fixed Nominal Wage Expectation

Potentially the simplest possible implementation of the household wage ask price decision rule, this module selects a fixed ask price $w$ per unit labour no matter what the state of the household/simulation world. $w$ is nonzero but may or may not be zero. Note that this module prohibits wage competition among households, and either restricts or entirely precludes inflation.

[20] subject to eg. possible minimum wage constraints

#### 4.6.2.5.11 Wage Confidence

This module describes a household whose confidence in its ability to command high wages increases or decreases depending on its recent employment history. The household becomes confident in its ability to command high wages if it has been fully employed in recent business cycles and the household loses confidence in its labour price if it has recently been unemployed. Note that (depending on the resolution of the simulator) one household may represent millions or thousands of individual domestic properties: a macro scale simulation provides little or no information about the distribution of individual unemployed households contained within and represented by a partly unemployed `Macrohousehold` agent.

The steps taken by the household agent when deciding the ask price $w$ are as follows:

1. The labour market is asked about overall employment/unemployment in the last labour market processing session. The labour market computes the fraction of the total labour supply that was in employment, $e$ ($0 \leq e \leq 1$), for the most recent market session.

2. If the household is fully employed (all labour supplied by the household is currently under contract) then the household computes a wage perturbation $\Delta w = \mu w_0$ where $\mu = \epsilon(\frac{1}{e} - 1)U[0, 1]$, $0 \leq \epsilon \leq 1$ is a customizable model parameter, $U[0, 1]$ is a uniform distribution on the unit interval, and $w_0$ is a model parameter describing the characteristic scale on which to perturb the existing ask price.

3. If the household is not fully employed (it is not the case that all labour supplied by the household is currently under contract) then the household computes a wage perturbation $\Delta w = \epsilon e U[0, 1] w_0$ (where $\epsilon$ is as above, and $U[0, 1]$ is a uniform distribution on the unit interval).

4. The perturbations $\Delta w_t, \Delta w_{t-1} \ldots \Delta w_{t-T}$ (where $\Delta w_x$ is the value of the above perturbation computed in simulation cycle $x$) are summed and averaged for a number of past cycles $T$, where $T \geq 0$ is a model parameter. The resulting smoothed wage perturbation $\Delta^*$ is applied to $w$: $w \mapsto w + \Delta^*$.

The expression $(\frac{1}{e} - 1)$ is chosen because for a simple labour model with $L$ fixed units of labour supply and $L^* \leq L$ fixed units of labour demand (which are guaranteed to be realized, no matter what the labour price $w$) a random assignment of households to vacancies will result in a household being unemployed once every $(1 - \frac{L^*}{L}) = 1 - e$ business cycles. If a variable $x$ is perturbed by $+\alpha\delta$ whenever the household is in employment and perturbed by $-\delta$ whenever the household is not fully employed then the expected change in $x$ per business cycle is $e\alpha\delta - (1 - e)\delta = (\alpha e + e - 1)\delta$, which is equal to zero when $\alpha = (\frac{1}{e} - 1)$. Put differently, when using this balancing factor a biasless labour market is expected to have the property that the mean upward wage perturbation as a result of full household employment is equal to the mean downward wage perturbation as a result of unemployment.

Note that the user must specify the initial wage ask price $w(t = 0)$. This experimental model can be improved by removing the 'fully employed' versus 'not fully employed' dichotomy and replacing this with (eg.) wage perturbation factors associated with each unit of unemployment.

#### 4.6.2.6 Labour Supply

In each simulation cycle households are endowed with a supply of $L$ units of labour. $L$ is the maximum amount of new labour the household may put under contract in the business cycle in which it becomes available.

$L$ is not strictly a decision rule because the household agent does not necessarily control $L$ or decide $L$ at its discretion. $L$ may be an exogenous quantity selected by the user, or a quantity that is derived from the utility of consumption. $L$ must be selected by an algorithm, and the standard model provides a number of possible implementation of this algorithm:

1. **Fixed Labour Supply**
   Potentially the simplest possible implementation of the labour supply algorithm, this module provides a fixed amount of labour in each business cycle. $L$ has a fixed value in time not depending on the state of the household and not depending on the state of the simulation world. The value of $L$ is customizable by the user.

2. **Model Parameter**
   The labour supply algorithm is an object and a model parameter in the meaning of §4.3. Any of the stock model parameters described in §4.3 are acceptable.

Note that, by manually specifying the size of the labour pool, it is possible to stimulate a depression in the economy.

#### 4.6.2.7 Household Bankruptcy

Deposit account overdraft states are currently not permitted in the standard model. Households may suffer deposit account losses as a result of bank bankruptcies, but this state will not result in a negative deposit account value. Cash transactions that require more money than is available to the household are rejected. Households cannot enter a net debt state and are therefor not subject to bankruptcy resolution procedures. There is currently no domestic bailiff service in the standard model.

### 4.6.3 Observations and Desirable Features

Households are potentially a fruitful area of development and there are many possible extensions and generalizations of the existing household agent. The default agent implementation used by the standard model is called `Macrohousehold` and this agent is (as its name suggest) macro scale.

#### 4.6.3.1 Household Lifecycle and State Transition

Individuals and properties evolve through a lifecycle. New dwellings begin at high value and slowly deprecate, requiring increasing labour and inputs to maintain. Unrenewed dwellings become uninhabitable and ultimately exit the market. Individuals have a birth/death lifecycle beginning as dependents, eventually maturing to working age, at which point their labour contribution increases and then declines. At retirement individuals require pension finance and state services. Some studies purposefully include marital/cohabitant states or more granular states when simulating this evolution process.

It is of interest to add (eg.) a Markov transition design pattern in which an object evolves through a set of predetermined, described states. Each state of the agent expresses various characteristics, for instance consumption propensities and the labour supply. Given that the simulator is capable of discrete time event simulation, it would not be unduly difficult to add a Markov process for a specialized household agent.

#### 4.6.3.2 Household Bankruptcy

All cash payments in the standard model are debit transactions. The standard model does not simulate credit cards. It is not possible for a household to become cash flow insolvent as it cannot realize an overdraft state. Introducing overdraft states would require household bankruptcy handlers to be created, and these bankruptcy handlers may involve bailiffs and durable goods auctions.

In the standard model households do not sign extended contracts/repayment plans that would require them to make periodic instalments and consumption is not financed by loans. With mortgages and property collateral, repossession may become a necessary feature of household bankruptcy.

#### 4.6.3.3 Buffer–Stock Theory

'Buffer–Stock' theory is a model of consumption by C. D. Carroll et al[21]. Household decision rules are modular and it would not be difficult to install this feature as a supplementary rule.

#### 4.6.3.4 Product Attachment

Experience has shown that the CES Aggregator consumption function §4.6.2.2.6 has some undesirable properties. For Multiple Firms Per Sector (MFPS) models in which several firms compete to produce otherwise undifferentiated goods, the CES consumption function describes a process in which nothing but retail price is important to households. Households are unable to differentiate identically priced goods of the same class when using CES, and households do not care from which firm such goods/services are sourced. This is not a realistic model of household consumption because it does not capture firm reputation or consumer brand loyalty. It is believed that there should be a quality measure attached to goods/services and that this quality measure should reflect the effort required to produce the good, as well as the total investment the went into its development.

[21]The buffer-stock theory of saving: Some macroeconomic evidence, C. D. Carroll et al.

#### 4.6.3.5 Consumption Polling

In order to implement a polling ask price decision rule §4.4.2.4 for firms it will be necessary to modify the household interface. The household agent must expose a method that (a) accepts a set of goods prices and (b) returns a set of intended consumption budgets for each goods class given the quoted prices.

## 4.7 Government

### 4.7.1 Motivation and Discussion

Two primary motivations for the inclusion of a government agent in the simulator are to:

1. include welfare/benefits (and create the possibility of a welfare state economy),
2. properly represent gilts[22] and government debt,
3. introduce regulatory restrictions such as minimum wage,
4. tax transactions and profits in the economy,
5. provide a commercial bank bailout service,
6. allow for the possibility of public sector consumption and government as a buyer of goods/services, and in due course to
7. facilitate a multi-country/multi-currently model in which households belong to a dedicated government's sphere of authority.

Government welfare and benefits spending in Eurozone states can be significant: in 2014 the UK central and local governments spent in total 112.4 billion Sterling on welfare, or roughly £1 of every £7 spent by government. This sum was dominated by social inclusion and protection expenses and was more than double the national defence budget. The welfare spending of the UK government was second only to healthcare and pension expenses in that order[23][24].

In order to quantify the extent of government borrowing: at the end of 2013 the value of all gilts issued by the UK government to national banking institutions was around £500 billion in total. Approximately £413 billion in gilts has been issued to foreign entities. About 16% of the value of all foreign issue gilts are owned by foreign central banks. According to the UKEA data, in Q4 2013 non-foreign gilts were mainly issued to monetary/financial institutions including the Bank of England. The next largest non-foreign consumers of government gilts are insurance companies and pension funds. In the UK the central government liability to pension funds and insurers seems is roughly 70% of the liability to financial institutions including commercial and shadow banks.

[22] *gilts* are a United Kingdom parlance for government bonds.
[23] `gov.uk/government/collections/public-expenditure-statistical-analyses-pesa`
[24] `budgetresponsibility.org.uk/economic-fiscal-outlook-march-2015/`

In the UK the average weekly wage (assuming employment) is slightly above minimum wage, so minimum wage regulations are likely to be important for realistic simulations of price-competitive labour markets. The simulator offers a number of simple decision rules for wages, and wrapping any of these in a forwarding decision rule with minimum wage regulation (at a level specified by government) is not unduly difficult. Choosing the minimum wage threshold may however involve political as well as economic considerations.

Consumption is taxed by VAT (currently 20%) in the UK. All goods bought at retail are subject to VAT. Goods bought as part of the input-output process (namely, goods that are ingredients in production) are not subject to VAT. PAYE (labour wages) in the UK are subject to income tax, which is deductible from final pay (Ie. individuals typically never receive the taxable component of their income). In the simulator, a government agent may need to decide the level at which goods transactions and income are taxed.

The simulator includes a framework for goods/services payments ('goods for cash') and PAYE ('labour for cash'), meaning that taxation can be implemented realistically as part of the cash transaction subsystem with reference to tax levels set by government. Again choosing the level of VAT/income tax may be a political as well as an economic decision. Some bonds and financial investments in the UK are taxable whereas other investments (eg. ISAs) are tax exempt.

Commercial bank bailout is a rare and potentially very expensive bank insolvency resolution process. Following the 2007/08 episode the UK government recapitalized Lloyds Banking Group (Lloyds) and the Royal Bank of Scotland (RBS) and nationalized Northern Rock and Bradford and Bingley. The UK government provided guarantees for deposits up to £50k as well as special liquidity, creditor guarantee and asset protection schemes which at their peak amounted to £1.2 trillion in support and a cash outlay of £133 billion; seven years later in 2005 the UK government still provides around £120 billion in support.

When large financially important commercial banks fail the government may decide to provide public funds to settle the debt and/or restore bank equities to acceptable levels. In exchange the government is expected to receive shares in the failing bank and/or take majority ownership and control of its affairs. If and when the commercial bank has been restored to a state that it can continue to operate effectively, the government may choose to sell its shares and relinquish control of the bank to private concerns. The standard model provides government bailin services, government share retention and some welfare austerity measures that may result from such a significant national expense.

### 4.7.2 Decision Rules

In the standard model the decisions made by the government agent include the following:

1. The VAT (goods/services) tax percentage,
2. The government consumption function,

3. The government (government bond) gilt debt target,
4. The income tax percentage,
5. The government cash reserve target,
6. The stock market participation (share resale) policy,
7. The total expenditure budget (labour, consumption and welfare/benefits) not including debt interest,
8. The wage bid price (per unit labour) for any public sector labour to employ,
9. How (in what proportion) to subdivide the expenses budget between consumption, welfare/benefits and public sector wages.

These decisions are outlined the following subsections.

#### 4.7.2.1 VAT (goods/services) tax

When agents in the codebase buy goods/services a proportion of the underlying flow of funds is forked from the transaction and transferred to the government agent, if the government agent is present. If the government agent is not present in the simulation then VAT–type taxation does not occur and 'goods for cash' transactions are subject to no deductions.

When a transaction of value $V \geq 0$ takes place (for the acquisition of $q$ units of durable/non-durable goods) the settlement system asks the government agent for a number $T_{\mathrm{g}}$ ($0 \leq T_g \leq 1$) representing the tax deduction proportion. $T_{\mathrm{g}}$ is selected by government is not necessarily constant between business cycles. The sum paid by the consider is $V$, the sum received by the supplier is $(1 - T_g) \cdot V$ and the sum received by government is $T_g \cdot V$. In the event that the consumer cannot afford to pay $V$ then the transaction is either rejected (and is not processed further) or is rationed in such a way that the consumer can afford a modified sum $V' \leq V$ (for $q' \leq q$ units of goods/services) based on its liquidity. The government therefor cannot fail to receive the appropriate tax sum if the transaction is processed. The tax income from the transaction is received immediately by the government agent.

#### 4.7.2.2 Consumption Function

The consumption preferences problem faced by the government agent is, mathematically, the same as the consumption preferences problem faced by household agents (§4.6.2.2). The government must decide an overall consumption budget $B$ and then subdivide this budget among differentiated goods types. Any of the consumption preference rules used by households (as indicated in §4.6.2.2) are compatible with the government agent in principle. By default the standard model provides an inelastic government consumption rule with fixed preference weights.

The government agent decides its preferences for goods/services according to a pseudo von-Neumann Leontief utility function with reference to a set of

consumption weights $\{w_g\}_g$ where $g$ runs over all sectors in the economy. The weight terms $w$ are non-negative and are normalizable such that $\sum_g w_g = 1$. For a government consumption budget of size $B \geq 0$ the amount of money $B_g \geq 0$ the government intends to spend consuming goods of type $g$ is $w_g^* B$, where $w^*$ are the normalized weights.

Note that it is acceptable for some weights $w$ to be zero. The government does not buy goods/services from sectors $g$ for which $w = 0$.

Government interaction with the goods market can be disabled entirely by the user. In this configuration, all government money set aside for consumption is unused.

#### 4.7.2.3 Gilt Issuance

If the government agent sells bonds then it is required to decide the maximum amount of cash it is willing to borrow (the value of the gilt issue) in each business cycle. Government bonds and gilts are implemented as fixed term fixed rate coupon bonds.

The standard model provides a simple decision rule in which the government will attempt to reach a target debt/gilt issue liability. The value of the gilt debt target is customizable and is specified by the user. At each market processing session the government requests to borrow up to $D \geq 0$ units of cash in gilts where $D = \max(G^* - G, 0)$, $G^*$ is the target debt value in gilts, and $G$ is the current government gilt debt value at the time of market.

Note that it is acceptable for $G^*$ to be zero. In this case the government never borrows money and will not participate in bond markets.

#### 4.7.2.4 Income Tax

When employers in the codebase buy labour services a proportion of the underlying flow of funds if forked from the PAYE transaction to the government agent, if the government agent is present. if the government agent is not present in the simulation then labour wages and household income are not subject to tax: labour wages are subject to no deductions and households receive the entire employer labour wage expense as income.

The government is required to select a number $T_w$ ($0 \leq T_w \leq 1$) describing the proportion of the wage packet flow of funds that should be forked to government. When an employer pays labour wages of value $V \geq 0$ the settlement system first asks the government for the value of $T_w$. The employer receives $(1 - T_w)V$ units of untaxable income and the government receives a sum $T_w V$.

In the event that the employer cannot pay $V$ then the transaction is cancelled and no labour is remunerated. If the transaction is processed then the government agent cannot fail to receive the sum $T_w V$ in tax. Income tax enters the government cash reserve immediately.

#### 4.7.2.5 Cash Reserve

The government cash reserve is a fully liquid government balance sheet asset endowed to the government agent at the time the simulation is initialized. Government consumption (if any) §4.7.2.2, public sector remuneration (if any) §4.7.2.9 and government bond interest payments (if any) §4.7.2.3 are paid from the government cash reserve. The size of the initial government cash reserve is customizable and is specified by the user.

#### 4.7.2.6 Stock Market Participation

The government agent may receive stock assets as compensation in lieu of cash for bankruptcy resolution and commercial bank bailout. In the absence of a stock market participation strategy the government will retain these assets permanently.

The standard model provides a simple model of gradual government stock resale. If the government owns a total value $V \geq 0$ in stock (potentially distributed over many instruments/types of shares) then the government strategy is to plan to own $\alpha V$ in shares after the next stock market processing session. $\alpha$ ($0 \leq \alpha \leq 1$) is a model parameter and is specified by the user.

In order to realize this asset value the government must respond to changes in the stock market price per share $p$. If the government owns $N$ shares with value $pN$ before stock market processing and if the revised price per share selected by the market is $p^*$ then the intended government stock exchange transaction is as follows:

1. If $p^* > p$ then the value of the government stock asset before share transactions is $p^*N$. This represents a capital gain of $(p^* - p)N$. The government aims to retain a stock portfolio of value $\alpha pN$, so the government agent bids to sell $(p^*N - \alpha pN)/p$ shares in order to realize this. The resulting flow of funds enters the government cash reserve.

2. If $p^* < p$ then the capital value of existing government shares has decreased as a result of price changes in the market. If $p^*N < \alpha pN$ then the government would be required to buy additional shares in order to realize the target stock portfolio value $\alpha pN$. In this case the government (a) takes no action if $\alpha > \frac{p^*}{p}$, and otherwise (b) sells $(p^*N - \alpha pN)/p$ shares. The resulting flow of funds, if any, enters the government cash reserve.

This policy describes a simple stock market participation strategy in which (a) the government agent will never buy any new shares, and (b) the value of all shares owned by the government will decrease exponentially (at a rate $\alpha$) if the government is able to realize this after capital value changes: otherwise the government will retain all its shares and accept the underlying capital loss.

Note, hypothetically, that if the stock price $p$ is never revised by the market then the value of the government stock portfolio evolves as the series $V, \alpha V, \alpha^2 V ....$ When $n$ simulation cycles have elapsed the value of the portfolio is $V(1 - \alpha^{n+1})/(1 - \alpha)$, a quantity that is never zero. For this reason the user is asked

to specify a cutoff value threshold $V^* > 0$ at which point the government will attempt to sell its entire outstanding stock portfolio without regard for further exponential decreases in value: when $V < V^*$ then the target government portfolio investment value is zero rather than $\alpha V$. If this numerical cutoff $V^* > 0$ is strictly positive then in principle the government agent is able to sell the entire stock portfolio in a finite number of business cycles. It is acceptable for the user to select $V^* = 0$.

#### 4.7.2.7 Expenses

In the same way that household agents are required to select a consumption budget $B \geq 0$ before buying goods/services (§4.6.2.1), the government agent is required to select an expenses budget $B \geq 0$ to be divided between welfare/benefits, public sector wages and consumption.

In the standard model the default government expenses budget decision rule is as follows:

1. At the end of the business cycle the government examines its cash reserve and finds $C \geq 0$ units of unallocated cash.
2. If $C > C^*$, the target cash reserve size, then the total government expenses budget is $C - C^*$.
3. Otherwise if $C < C^*$ (the value of the cash reserve is less than the target value), then the government takes the following steps:
   (a) The combined cost $D$ of all bank bailouts in the timeframe $[t, t - T]$, where $T$ is a customizable time window and $t$ is the current simulation time, is computed.
   (b) The total tax revenue $R \geq 0$ from the last business cycle is computed.
   (c) The government expenses budget is set to $B = \max(\frac{9}{10}R - \frac{D}{T}, 0)$.

$T \geq 0$ is a model parameter representing the characteristic timescale over which bailout losses should be recuperated from tax revenue. If $T = 10$ years and the government immediately pays $X$ units of cash to finance a bailout, then the government will attempt to set aside $X/10$ units of cash per year from the incoming tax stream in order to restore its cash reserve to pre-bailout levels within 10 years. If bailout expenses are low then the government will spend the entire or nearly the entire tax revenue stream on public sector workforce, consumption and/or welfare and benefits.

$C^*$ is a model parameter representing the ideal or intended size of the government cash reserve. In the standard model $C^*$ is equal to the size of the initial government cash endowment (the amount of cash with which the government is initialized at startup).

Note that although it is acceptable to configure a simulation in such a way that the government has no outgoings (no public sector workforce, no requirement for capital or goods/services, no welfare expenses and no bond market participation) this configuration will result in a money sink because tax revenue

collected by government has no means of re-entering the economy.

Once the expenses budget $B$ is decided the government must further decide how to distribute $B$ between (a) public sector wages (§4.7.2.8), (b) new goods/services (§4.7.2.2) and (c) welfare and benefits (§4.7.2.9). In the standard model the government uses a fixed partition function defined by three weights $w_\ell \geq 0$, $w_g \geq 0$, $w_b \geq 0$ such that $w_s + w_b + w_g = 1$. The sum $w_\ell B$ is set aside to salary the public sector workforce, the sum $w_g B$ is set aside to finance consumption and the sum $w_b B$ is allocated to welfare and benefits (§4.7.4).

$w_\ell, w_g$ and $w_b$ are model parameters and are customizable by the user. Note that it is acceptable for one or more of $w_\ell, w_g, w_b$ to be zero.

#### 4.7.2.8 Wage Ask Price and Public Sector Workforce

Once the public sector workforce budget $B_\ell$ has been decided (§4.7.2.7) the government must further decide at what rate to salary its workforce. To this end the government must select a wage bid price $w \geq 0$ per unit public sector labour.

By default the government has no priority over the labour market. The government is subject to minimum wage regulation if minimum wage regulation is in effect. Depending on the implementation of the labour market it may be the case that high ask prices $w$ per unit labour result in many candidates and low wages $w$ result in an inability to fill vacancies.

#### 4.7.2.9 Welfare/Benefits and Cost Breakdown

When the government agent decides the maximum welfare/benefits budget $w_b B$ it must also decide how to distribute this sum among welfare recipients in its authority. Welfare recipients (households) are required to register themselves with the government agent at instantiation (time $t = 0$) so that the government knows to whom it must distribute these benefits. Given a list of $N$ such welfare recipients $J = \{j_0, j_1 \dots j_{N-1}\}$ the government welfare distribution decision is as follows:

1. The maximum welfare/benefits budget per welfare recipient is computed based on a fairness criterion as $w_b B/N$ (§4.7.2.7), unless the welfare/benefits budget has size zero, or there are no welfare recipients in the authority of the government, in which case this sum is zero.

2. For each welfare recipient $j$ the government queries (a) the amount (total supply) of labour $L$ that $j$ could have provided in this business cycle and (b) the amount this labour $U \leq L$ that is unemployed/not under contract.

3. For each welfare recipient $j$ the government computes a benefits budget $w_j$ of size:
$$w_j = \frac{U w_b B}{LN}$$

4. $w_j$ is transferred to $j$ as an untaxable settlement.

Under normal operating conditions the above welfare settlements cannot fail because the government has already budgeted for the worst case benefits scenario $\sum_j w_j = w_b B$.

Note that in general the amount paid by the government in welfare/benefits is less than $w_b B$. The entire welfare budget will only be paid by the government agent if the economy is at 0% employment.

#### 4.7.2.10 Background

In one of the earliest iterations of the standard model the government welfare budget was computed according to the following recipe:

1. The maximum welfare/benefits budget is set to $w_b B$.
2. For each welfare recipient $j$ whose unemployment is $U_j$ and whose total labour supply is $L_j$, the sum received by $j$ in welfare/benefits is:

$$w_j = \frac{U_j}{\sum_k U_k} w_b B.$$

Namely, welfare recipients receive benefits according to their relative degrees of unemployment. Unforuntately it is quite possible (particularly during a VaR leverage cycle) for the total unemployment $\sum_j U_j \longrightarrow 0$. This meant that any welfare recipient with a numerically residual unemployment $U_j \approx 0$ will receive virtually the entire welfare budget when using an approach such as this.

### 4.7.3 Unconventional Policy Operations

Interventions in times of financial crisis (oft referred to as 'unconventional' monetary policies) are implemented in the standard model primarily in terms of bank bankruptcy resolution policy modules. Whereas the failure of non-financial sector firms does not require any substantial action to be taken by the government agent or by the central bank agent[25] the failure of systemically important banks may result in specific, and potentially very expensive, actions taken by the government agent. The government can use public funds (taxes) to finance unconventional policy operations.

In the standard model the government is involved in the following bank bankruptcy resolution policies:

1. **Bailout**
   Bailout is the process in which a government effectively pays a troubled bank $b$ in order to restore it to positive equity and ensure the continuation of its business. The government uses public funds in order to recapitalize $b$, and the government may receive shares or other forms of compensation as a result of this action. This compensation results in a short term expense to the taxpayer and may or may not result in a long term loss to the public (either in the sense that the government makes a long term profit on its transactions, or in the stronger sense that a profit is made and the public

[25] if present

is compensated for the risk assumed and the opportunity cost of lending, or in the weaker sense that a worse or more costly scenario was averted as a result of the actions of the government).

The amount of money the government requires in order to finance a bailout is equal to the negative equity of $b$ plus an overhead to ensure the functioning of $b$ after the bailout (restoring $b$ to zero equity is not sufficient to ensure its survival). If the government does not have access to sufficient bailout funds or if the government declines to fund the bailout then another, different bankruptcy resolution technique is applied to $b$. Note that this failover resolution attempt may result in $b$ being discontinued or liquidated.

Note that a successful bailout will result in the continuation of $b$.

2. **Purchase & Assumption**
   Purchase and assumption (P&A) is a bankruptcy resolution technique in which a troubled bank's operations are transferred to other, healthier peer banks. Write $b$ for the bankrupt bank and $P = \{p_1, p_2 \dots p_N\}$ for its peers. In the purchase and assumption scheme, the banks $p \in P$ take on (assume) assets and liabilities from $b$. The total value of assets assumed by each peer $p$ is proportional to its liquidity position.

   When assets have been assumed by $P$, $b$'s liabilities are also distributed to peers according to the value of the assets taken over by each. Finally, these assets need to be 'trued up' as their combined value is smaller than the combined liability value (as $b$ is assumed to have negative equity). This finance gap is bridged by the government agent who collects the required funds via taxes.

   Note that the P&A scheme is terminal for $b$. $b$ is wound down following the redistribution of its balance sheet assets and liabilities and its business does not continue.

Note that it is unnecessary for the government agent to spend public funds in order to process a bank bailin as the required funds are ultimately written down by bank creditors and depositors.

### 4.7.4 Observations and Desirable Features

The relationship between government and large public sector organisations may be modelled in more detail through the use of dedicated public sector agents than by grouping public sector bodies with existing firms (much as the Bank of England is not modelled as a high street bank). In a model for which government goods/service consumption is directed largely at or on behalf of a specific national service healthcare provider (namely the NHS in the United Kingdom) special mechanics may be necessary. The healthcare and medical service budget of the United Kingdom was £133bn in 2015: more than twice as large as the cost of debt servicing and second only to the cost of public sector pensions (£149.7bn). The precise relationship between government and pension funds is also likely to be significant to third–generation models of European economies.

It should be noted that the structure of public pension arrangements in European and US economies is heterogeneous. In Germany the mandatory state

pension insurance scheme (gesetzliche Rentenversicherung) is a redistributive pay as you go system in which new contributions (money paid into the fund by non-retirees) is not invested but is instead used to settle ongoing pension fund outgoings[26].

It would be helpful to explore the exact nature of the state regulated minimum wage decision processes, and it would also be helpful to add a mechanism to compute inflation through sampling the basket prices of representative goods. Some decisions made by the government agent, for instance the growth or otherwise of the national defence budget, the size and payment policy of the welfare budget and any central/local government intervention in the housing market and/or support for first time buyers may be motivated by political considerations as well as economic outcomes. The government's objectives are not only to serve the interests of the economy and maximize social benefits in its sphere of influence, but also to be re-elected. In may ultimately be necessary to add at least a two-party election system to the government agent and observe the economic impact of differing governance policies.

## 4.8 Goods Markets

Goods markets facilitate interactions between parties who sell goods/services (producers, typically firms/establishments, and perhaps resellers, intermediaries, firesale organizers) and parties who intend to buy the same (consumers, typically households, perhaps government and/or public sector bodies, other firms) with a view to executing trades. By default the standard model is configured so that firms sell and households buy differentiated goods in exchange for cash and no other parties are involved. Government and other agent types may optionally participate in the bidding process. By default all goods contracts in the standard model are executed immediately after their inception (ie. a goods–for–cash transaction is attempted as soon as the contract is established). There is currently no notion of extended supply contracts in which producers are required to furnish consumers with agreed or prepaid goods/services at regular intervals: consumers are required to bid for goods anew as required.

The goods market in the standard model is a batch processed complete bipartite graph whose disjoint node groups consist of `GoodsSellers` and `GoodsBuyers`, respectively. The `GoodsSeller` class represents many possible different types of agent, all of which possess inventory to sell. The `GoodsSeller` group is typically populated exclusively by firms. `GoodsBuyers`, who intend to buy new goods and services, submit bid orders to the market and specify the maximum price per unit they are willing to pay for the privilege. `GoodsSellers` submit ask orders to the market and specify the minimum price per unit at which they are willing to do business.

The goods market matches these orders twice per simulation cycle: once in order to process the input-output (firm-to-firm) consumption/production stage, and then again to process the firm-household consumption stage. Goods not

[26]Rüdiger Blaich, Aegon, *Pension provision in Germany: the first and second pillars in focus*

sold during the first session remain available for household consumption during the second session. The amount of trade processed during the household consumption stage typically dominates the amount of trade processed during the input-output stage by an order of magnitude or more.

### 4.8.1 Goods Market Entry Point

When submitting orders the behaviour of the goods market is as follows:

1. If a party submits an ask order then the party must be an `GoodsSeller`, otherwise the order is rejected by the market (an exception is raised).
2. If the party submits a bid order then the party must be a `CashAllocating`[27] `GoodsBuyer`, otherwise the order cannot be processed.
3. `GoodsSellers` must be able to allocate sufficient inventory for the order, otherwise the order is rejected by the market on the basis that the ask party cannot deliver the promised resources.
4. `GoodsBuyers` are required to allocate sufficient cash at the price specified by their bid order (the maximum price to pay per unit new goods/services). If the `GoodsBuyer` fails to allocate this sum but can allocate a lesser nonzero sum then the order size is reduced proportionally. If the `GoodsBuyer` cannot allocate any cash then the order is rejected by the market.

The bid and ask price (price per unit goods/services) must be positive for all market participants.

### 4.8.2 Background and Legacy Model

The legacy mark one (§3) model had no input-output network. $L \geq 0$ units of labour employed at time $t$ were worth $\alpha L^{\beta}$ units of abstract durable capital for some numbers $\alpha$, $\beta$. This abstract durable capital was used for production and decayed exponentially with each passing simulation cycle.

The legacy model was believed to be subject to numerical problems caused by floating point rounding errors in cash and goods allocation. Many markets in the standard model require participants to allocate (set aside) all resources they intend to trade at the time the order is submitted: in this case buyer cash and seller inventory. In some cases these allocated resources are ultimately represented by encapsulated floating point numbers. For this reason the size of the allocation and the quantity of tradeable resources owned by the participant may be different to the order of the unit of least machine precision. In the legacy model this discrepancy was believed to be sufficient to cause the market to fully reject an order and therefor sporadically exlude a participant, which in the case of the goods market, could lead to artificial firm bankruptcies. The current implementation of the goods market is such that order sizes are automatically scaled down in the event that a participant cannot allocate the required resources but can allocate a lesser nonzero quantity.

[27] An agent with the ability to allocate cash to use for a specific future purpose, in this case buying goods/services.

### 4.8.3 Contract Matching

The labour market is processed by an `GoodsBuyer-GoodsSeller` matching algorithm. The matching algorithm is required to do the following:

1. accept sets of bipartite input nodes $F$ (`GoodsSellers`) and $H$ (`GoodsBuyers`) who have submitted ask orders $A$ and bid orders $B$ (respectively) as follows:

   (a) bid orders $B$ of the form $(h, p, d)$ where $h \in H$ is a `GoodsBuyer`, $p \geq 0$ is the bid price per unit goods/services to buy and $d > 0$ is the quantity required by the `GoodsBuyer` (the demand). Write $p = p(h)$ for the bid price specified by $h$ and $d(h)$ for the demand.

   (b) ask orders $A$ of the form $(f, p^*, s)$ where $f \in F$ is a `GoodsSeller` (eg. a firm), $p^* \geq 0$ is the ask price per unit goods/services and $s > 0$ is the saleable quantity offered by the `GoodsSeller`. Write $p^*(f)$ for the ask price specified by $f$ and $s(f)$ for the supply.

   Note that `GoodsSellers` $f \in F$ submit goods orders to the market, not orders that are required to be fulfilled by specific agents $h \in H$. Likewise `GoodsBuyers` $h \in H$ submit goods orders to the market, not orders that are required to be fulfilled by particular $f \in F$.

2. formulate a set $X$ of matchings $X = \{(f, h, q, \varrho)\}$ where $f \in F$, $h \in H$, $q > 0$ such that:

   (a) $\varrho \leq p(h)$ (the `GoodsBuyer` pays no more than the bid price),

   (b) optionally: $\varrho \geq p^*(f)$ (the `GoodsSeller` receives no less then the ask price).

   $X$ describes a set of possible trades between `GoodsBuyers` and `GoodsSellers`. When processing $X$ it is often of greater importance that the `GoodsBuyer` pays no more than the bid price than that the `GoodsSeller` receives no less than the ask price. This is because the buyer is very often required to set aside (allocate) cash resources at the time the order is submitted to the market. If the execution price $\varrho$ selected by the market is greater than the bid price $p(h)$ then the transaction may fail as a result of insufficient allocated buyer cash.

3. the sum of the sizes of all goods/services contracts in which any $h \in H$ is involved is less than its demand, $d(h)$:

$$\sum_{x \in X \,:\, h \in x} q(x) \leq d(h)$$

4. the sum of the sizes of all goods/services contracts in which any $f \in F$ is involved is less than its saleable inventory, $s(f)$:

$$\sum_{x \in X \,:\, f \in x} q(x) \leq s(f)$$

Note that it is acceptable for a `GoodsSeller` (which may be modelled on the macro scale, for instance an entire sector of competing firms) to provide goods to multiple `GoodsSellers`. The latter two conditions in the about specification ensure that no individual unit of goods/services is sold twice during the same market processing session.

Other than that the goods market is s(a) required to find a solution satisfying the above specification, and (b) is not allowed to modify the states of the market participants, the goods market may compute a solution by any means. The means through which the goods market computes a solution are to be documented be the implementation.

### 4.8.4 Rationing Precondition

Most goods markets in the standard model operate on the premise that a preconditioning step designed to reduce aggregate supply and/or reduce aggregate demand should first be applied to $B$ and/or $A$. This preconditioning step is required to reduce aggregate supply among nodes $f \in F$ and/or reduce aggregate demand among nodes $h \in H$ so that the resulting total supply is equal to total demand. It is expected, but not required, that this rationing should be applied to one but not both groups.

This preconditioning step is loosely referred to as 'rationing'. The rationing algorithm is therefor required to accept sets of disjoint nodes $F$ and $H$ (with bid/ask orders $B$, $A$) and either exclude some nodes from the market and/or specify limits on the proportions of orders that can be fulfilled for each market participant. The goods market rationing algorithm is an object and is a model parameter in the meaning of §4.3.

Concretely, the rationing algorithm is required to coerce the existing bid/ask orders $B$, $A$ to a new set of rationed orders $A^*$, $B^*$ satisfying the following constraints:

1. For all $x \in B$ there exists a modified order $x^* \in B^*$ of the form $\{h, p, d\}$ such that $0 \leq d^*(x) \leq d(x)$.
2. For all $x \in A$ there exists a modified order $x^* \in A^*$ of the form $\{f, p^*, s\}$ such that $0 \leq s^*(x) \leq s(x)$.
3. Aggregate supply and aggregate demand are equal for $B^*$ and $A^*$:

$$\sum_{x \in B^*} d^*(x) = \sum_{x \in A^*} s^*(x)$$

Other than that the rationing algorithm is required to find a solution $A^*, B^*$ satisfying the above constraints, the algorithm may compute a solution by any means. The means through which the rationing algorithm computes the solution is documented by the implementation.

The standard model provides the following stock implementations of the goods market rationing algorithm:

1. **Homogeneous Rationing**
   Perhaps the simplest biasless rationing algorithm. This algorithm reduces the total supply of or the total demand for goods, whichever is greater, through applying a fixed multiplier to the sizes of all bid orders or to the sizes of all ask orders (but not both). This algorithm is detailed in §4.8.4.1.

2. **Inhomogeneous Rationing**
   An efficient heterogeneous rationing algorithm in which some orders are curtailed more than others in order to match supply and demand. This algorithm is illustrated in §4.8.4.2.

3. **Greed Rationing**
   A rationing algorithm that penalizes participants who pose the lowest bid prices or the highest ask prices. This algorithm is outlined in §4.8.4.3.

#### 4.8.4.1 Homogeneous Rationing

Homogeneous rationing is perhaps the simplest possible biasless rationing algorithm. This algorithm reduces aggregate goods supply or reduces aggregate goods demand (but not both) by applying a fixed multiplier to exactly one of the `GoodsBuyer`/`GoodsSeller` node groups as follows:

1. Compute $d = \sum_{h \in H} d(h)$ and $s = \sum_{f \in F} s(f)$.
2. If $d = s$, then stop.
3. If $d > s$ then set $\xi = s/d$ and set $X = B$.
4. Otherwise if $d < s$ then set $\xi = d/s$ and set $X = A$.
5. For each order $x \in X$, replace $x$ with $\{\mu, p, \xi o\}$ where $\mu$ is the participant in $x$, $p$ is the bid/ask price specified by $x$, and $o(x)$ is the size of the order ($o = s(x)$ or $o = d(x)$).

The resulting order sets $(B, X)$ or $(X, A)$ are rationed.

Note that no internal random processes are in effect and therefor this process is deterministic.

Homogeneous rationing can be implemented as a simple double pass operation over $A$ and $B$. The total computational cost is therefor $O(n)$ where $n$ is the number of market participants (in particular very nearly $2n$ total arithmetic cost).

#### 4.8.4.2 Inhomogeneous Rationing

Inhomogeneous rationing is a generalization of the homogeneous rationing algorithm §4.8.4.1 with uneven rationing of order sizes. This algorithm reduces the aggregate goods supply or reduces the aggregate goods demand (but not both) by randomly reducing the sizes of a sufficient number of orders in exactly one of the `GoodsBuyer`/ `GoodsSeller` node group as follows:

For $x \in A \cup B$ write $r(x, \xi)$ for the function $r(x, \xi) = \{\mu, p, {}_0|\xi|_1 o\}$ where $\mu$ is the participant in $x$, $p$ is the bid/ask price specified by $x$, $o$ is the order size ($o = d(s)$ or $o = s(x)$), and ${}_a|q|_b$ is shorthand for $\min(\max(q, a), b)$.

1. Compute $d = \sum_{h \in H} d(h)$ and $s = \sum_{f \in F} s(f)$.
2. If $d = s$, then stop.
3. If $d > s$ then set $X = B$, $t = s$, $a = d$. Otherwise set $X = A$, $t = d$, $a = s$.
4. Set $\xi = t/a$.
5. For each $x \in X$ apply $x \mapsto r(x, \xi + \Omega\epsilon \cdot (1 - \xi))$ where $\epsilon \sim U[0, 1)$ is drawn from a random uniform distribution on the unit interval and $\Omega \geq 0$ is a customizable model parameter.
6. Compute $a^* = \sum_{x \in X} o(x)$.
7. Set $\xi = t/a^*$.
8. For each $x \in X$ apply $x \mapsto r(x, \xi)$.

The resulting order sets $(B, X)$ or $(X, A)$ are rationed.

Note that the expression $\epsilon \sim U[0, 1)$ depends on a random number generator and therefor in general this process will require a random number seed in order to behave deterministically.

The parameter $\Omega > 0$ is a model parameter in the meaning of §4.3. The setting $\Omega = 0$ is equivalent to the homogeneous rationing algorithm §4.8.4.1 because steps $(6 - 8)$ are degenerate in this use case. $\Omega \sim 1$ is likely to yield a rationing in which some market participants are more severely penalized than others, whereas $\Omega \sim 0$ is likely to generate a mildly inhomogeneous rationing.

Note that this process is a three–pass algorithm with $O(n)$ complexity where $n$ is the total number of market participants.

#### 4.8.4.3 Greed Rationing

The greed rationing algorithm was originally devised as a mechanism to regulate wages in the labour market. The aim of this rationing algorithm is to penalize nodes whose price offerings are in excess of the mean price observed when processing the market. In the context of the goods market, this means that either `GoodsBuyers` who bid low prices or `GoodsSellers` with high selling prices are most likely to be excluded from the marketplace. This algorithm is an experimental work in progress and is included for completeness only.

For $x \in A \cup B$ write $r(x, \xi)$ for the function $r(x, \xi) = \{\mu, p, {}_0|\xi|_1 o\}$ where $\mu$ is the participant in $x$, $p$ is the bid/ask price specified by $x$, $o$ is the order size ($o = d(s)$ or $o = s(x)$), and ${}_a|q|_b$ is shorthand for $\min(\max(q, a), b)$.

For an array of nodes $X$, write $S^+$ for a sorting algorithm whose objective is $p(x_i) \leq p(x_{i+1})$. $S^+$ sorts $X$ ascending with respect to the order price $p(x)$. Write $S^-$ for a sorting algorithm whose objective is $p(x_i) \geq p(x_{i+1})$. $S^-$ sorts $X$ descending with respect to the order price $p(x)$.

1. Compute $d = \sum_{h \in H} d(e)$ and $s = \sum_{f \in F} s(f)$.
2. If $d = s$, then stop.
3. If $d > s$ then set $X = B$, $t = s$, $a = d$ and $S = S^-$. Otherwise set $X = A$, $t = d$, $a = s$ and $S = S^+$.
4. Set $\xi = t/a$.
5. Apply $S$ to $X$ (sort $X$ with respect to its order prices). Write $X_i$ for the $i$th element ($1 \leq i \leq |X|$) of $X$ after sorting.
6. Set $p = |X|$ and $R = (1 - \xi)a$.
7. If $o(X_p) \leq R$ then map $R \mapsto R - o(x)$, set $o(x)$ to zero and go to (8). Otherwise, map $o(x) \mapsto o(x) - R$, set $R = 0$ and stop.
8. Decrement $p$. Go to (7).

The resulting order sets $(B, X)$ or $(X, A)$ are rationed.

This rationing algorithm systematically trims order sizes, starting with the least reasonable order (with lowest bid price or highest ask price) and then continues to the next least reasonable order until the rationing is complete.

Note that there are no internal random processes in effect in this algorithm and therefor it is deterministic.

In general the cost of sorting an array of $n$ comparable datums is $O(n \ln n)$ flops. The cost of the instruction loop in steps $(6 - 8)$ is $O(n)$ flops. The total cost of this rationing algorithm is therefore $O(n \ln n)$ flops.

### 4.8.5 Matching

After rationing has been applied to the goods market participants (§4.8.4) the nodes $H$, $F$ will satisfy:

$$\sum_{h \in H} d(h) = \sum_{f \in F} s(f) = \Lambda$$

where $\Lambda \geq 0$ is the total combined trade (quantity of goods) required by both `GoodsBuyers` and `GoodsSellers`.

In the standard model most implementations of the goods market require a rationing preconditioning step followed by contract matching. The contract matching step is required to create a set a set $X = \{(h, f, q, p)\}$ of matchings where $h \in H$, $f \in F$, $q > 0$ such that:

1. $p \leq p(h)$ (the `GoodsBuyer` pays no more than the bid price),
2. $p \geq p^*(f)$ (the `GoodsSeller` receives no less then the ask price),
3. $\sum_{x \in X : h \in x} q(h) = d(h)$ (goods demand is satisfied for all `GoodsBuyers`),
4. $\sum_{x \in X : f \in x} q(f) = s(f)$ (goods supply is satisfied for all `GoodsSellers`),
5. $\sum_{x \in X} q(x) = \Lambda$ (trade is maximized).

Note that condition (5) is equivalent to (3) and (4).

The matching step is an object and is a model parameter in the meaning of §4.3.2. Other than that the contract matching step is required to identify a set $X$ satisfying the above constraints, matching algorithms may identify $X$ by any means. The means used to derive $X$ are documented by the implementation.

The standard model uses a matching step called 'forager'.

#### 4.8.5.1 Forager

The forager node matching algorithm is a simple efficient random matching process. This algorithms assumes that a rationing preconditioning step has been applied to the market participants (§4.11.4) beforehand.

This matching algorithm abstractly interprets the first (left) bipartite node group as a set of buyers and the second (right) bipartite node group as a set of sellers. The buyer node group is initially shuffled. Trades are now executed between seller/buyer pairs, starting at the top of both columns in the following table and working downwards systematically:

| shuffled buyers (intent) | sellers (intent) |
|---|---|
| (a) buy quantity $q_1$ at price $p_1$ | (i.) sell quantity $q'_1$ at price $p'_1$ |
| (b) buy quantity $q_2$ at price $p_2$ | (ii.) sell quantity $q'_2$ at price $p'_2$ |
| $\vdots$ | $\vdots$ |

The order of iteration is as follows: beginning with buyer (a) try to establish a trade with seller (i.), then by seller (ii.), and so on. Remove sellers from the system whenever a seller's supply is exhausted by this process. When the buyer demand is exhausted, remove the buyer for the system. Repeat this process with buyer (b), and so on in the list of remaining buyers. Whenever a potential trade $(\min(q, q'), p, p')$ is identified the matching algorithm makes the following assumptions:

1. if the buyer price is less than the seller price, $p < p'$, then the execution price is taken to be $p$ (favouring the buyer)[28]

2. if the seller price $p' < p$, then negotiation between the buyer and the seller is assumed. The resulting execution price is taken to be $(p' + p)/2$ (favouring the seller).

Note that for matching problems with homogeneous prices $p = p' = \rho$, where $\rho \geq 0$ is a constant, the execution price of all contracts is equal to $\rho$.

The name of this algorithm is borne out of its conceptual similarity to the behaviour of hunter gatherers, who randomly encounter natural resources and accept, out of necessity, whatever resources they find.

Note that the shuffling operation depends on a random number generator and therefor in general this process will require a random number seed in order to behave deterministically.

In general the cost of shuffling an array of $n$ datums is $O(n)$ flops or worse. The standard model uses a single-pass in-place biasless Fisher–Yates (Knuth) shuffle. The cost of the shuffle operation is $O(n)$. The formation of $X$ can be implemented as a single-ass double-cursor iteration visiting each node no more than once, therefore the total cost of this matching is $O(n)$.

### 4.8.6 Call Auction

The standard model provides an alternative contract matching algorithm that uses the rationing §4.11.4 step differently. This matching algorithm is normally referred to as the 'call auction'.

For a set of `GoodsBuyer` nodes $H$ with bid orders $B$ and a set of `GoodsSeller` nodes $F$ with ask orders $A$ the call auction processes the goods market as follows:

1. Create a function $v^- : \mathbb{R} \mapsto \mathbb{R}$ as follows:

$$v^-(w) = \sum_{f \in F : w(f) \leq w} s(f)$$

   ($v^-$ is the sum of the supply volumes of all ask orders whose prices are not greater than $w$, namely the total quantity of goods in the system that can be employed at price $w$ without undercutting the ask price).

2. Create a function $v^+ : \mathbb{R} \mapsto \mathbb{R}$ as follows:

$$v^+(w) = \sum_{h \in H : w(h) \geq w} d(h)$$

[28] As observed in §4.11.3 it is often of greater importance that the buyer (the `Employer`) pays no more than the bid price than that the seller (the `Employee`) receives no less than the ask price. This is because the buyer is very often required to set aside (allocate) cash resources at the time the order is submitted to the market. If the execution price $w$ selected by the market is greater than the bid price $\omega(e)$ then the transaction may fail as a result of insufficient allocated buyer cash. Note that in general it is not possible to satisfy the matching conditions unless the bid price is exceeded or the ask price is not met for some `Employer`–`Employee` pairs.

($v^+$ is the sum of the demand volumes of all bid orders whose prices are not less than $w$, namely the total labour in the system that can be employed at price $w$ without exceeding any bid price).

3. Identify the least execution price $w \geq 0$ such that $v^-(w) \geq v^+(w)$:

$$w = \inf_x \{x : v^-(x) - v^+(x) \geq 0, x \geq 0\}.$$

4. Exclude all `GoodsSellers` from the market whose ask prices are greater than $w$.

5. Exclude all `GoodsBuyers` from the market whose bid prices are less than $w$.

6. Compute $d = \sum_{h \in H} d(h)$ and $s = \sum_{f \in F} s(f)$ for all outstanding participants.

7. If $d \neq e$ then apply a rationing step (§4.8.4) to all remaining market orders (such that $d = e$ after the application of the rationing).

8. Apply a node matching step (§4.8.5) to the remaining market participants (generate a set $X$ of matchings).

Note that unlike other implementations of the labour market in the standard model, this market processing algorithm does not in general have the property that

$$T = \sum_{x \in X} \ell(x) = \min \left( \sum_{h \in H} d(h), \sum_{f \in F} s(f) \right),$$

where the right hand side is evaluated over the original orders $A$, $B$: the total trade described by the matching $X$ is in general not maximized with respect to the bid and ask volumes stated in the original orders. The total trade $T \geq 0$ executed by the call auction is expected to be significantly less than the maximum admissible demand and supply, which in turn means that many market participants are likely to be wholly excluded from the market by the final matching $X$.

The goods market call auction operates in the same way as the labour market call auction (with the clear exception that the underlying tradeable resource is a quantity of goods/services and not employee labour). See §4.11.6 for a discussion of the call auction in the context of the labour market.

### 4.8.7 Instruments

In the standard model the goods market is implemented as a collection of disjoint instruments labelled by sector. These instruments process mutually differentiated goods types (the type of goods produced by the corresponding sector). The goods market is composed of several such instruments, and these instruments may in principle involve different processing algorithms.

Orders are routed to the appropriate instrument according to the type of goods sold by or required by the participant. There is currently no notion of

extended goods/services contacts in which a supplier (a firm) is required to furnish a consumer (a household) with goods for an extended period. All contracts decided by the market are executed immediately after their inception and participants must post orders anew every simulation cycle as required.

It is not a strict requirement that goods markets should be implemented in such a way that each instrument is processed independently.

#### 4.8.8 Order Cancellation

In general the market will be unable to satisfy all orders even if the supply of goods from a particular sector exceeds the demand for goods from that sector. Unsatisfied or partly satisfied orders not resolved by the market must be cancelled in a timely fashion.

In the standard model, goods market contracts are processed twice per simulation cycle in batch. The first goods market processing session is dedicated to the input-output network (typically trades between firms) and the section session is dedicated to domestic consumption (typically trades between firms and households).

Orders are processed in the first instance for the input-output step at time `GOODS_INPUT_MARKET_MATCHING`. The consumption market is processed at time `GOODS_CONSUMPTION_MARKET_MATCHING`. Any outstanding or unfulfilled orders are cancelled immediately after the consumption market processing session at time `POST_CONSUMPTION_MARKET_MATCHING`.

Order cancellation entails the removal of the order from the participant order book (if applicable), the disallocation of any allocated liquidity associated with bid orders, and the disallocation of any inventory associated with ask orders.

### 4.9 Durable Goods

#### 4.9.1 Discussion

A basic assumption of input-output (IO) simulations is that firms/establishments are partitioned into $N_S > 0$ sectors. A 'sector' is a collection of competing and interacting agents producing exactly one of $N_S$ differentiated goods classes. Each firm (be it a `Macrofirm` §4.4 implementation or otherwise) is expected to belong to at least one sector for the purposes of processing the goods market.

The differentiation of goods types will be indicated by subscripts: a quantity $q$ of goods from sector $i$ ($1 \leq i \leq N_S$) is denoted $q_i$. The IO network facilitates the exchange of goods and services for cash over the markets. Physical examples of UK sectors include Construction, Wholesale and Retail Trade, Agriculture, and resource extraction businesses such as Mining. With the increase in model richness associated with customizable IO networks and sectors, it becomes necessary to differentiate the capital properties of the goods sold in each sector.

CRISIS has therefore been upgraded to include customizable durability (persistence) and reusability characteristics for goods. Write $\{g_i\}_{i=1}^{N_S} = \{g_1, g_2,$

$\ldots g_{N_S}\}$ for the differentiated goods types sourced from each sector. The characteristics of durable goods are defined to be: (a) damage, depreciation and/or losses suffered by the good as a result of its use as an ingredient in a manufacturing process, or as a result of domestic use (the overall reusability of the good both as an ingredient in production and its longevity as a used retail product) and (b) the evolution of the good in time (its decay properties). These properties are described by two customizable model parameters: $\delta_C$ ($0 \leq \delta_C \leq 1$), the 'depreciation rate', and $\delta_D$ ($0 \leq \delta_D \leq 1$), the 'decay rate', respectively.

We assume that $\delta_D$ and $\delta_C$ describe exponential loss processes in the quantity $q$ of owned goods. The following table illustrates the physical meanings of these parameters:

| | |
|---|---|
| $\delta_C$ | The decay rate associated with the depreciation of goods due to their use as ingredients in the production process (manufacturing) or as a result of their use by consumers. $\delta_C = 0.4$ means that 40% of goods will be destroyed when used as inputs in a production process. Note that depreciation is modelled as a decrease in the quantity of goods, as opposed to the value of owned goods: a good that has been used in production lessens in value not because the market price (per unit) is changed, but because a smaller quantity of the good remains. |
| $\delta_D$ | The decay rate associated with the natural depreciation of goods due to the passage of time. Decay occurs whenever the simulation advances from cycle $n$ to cycle $n+1$. $\delta_D = 0.8$ means that 80% of goods will be destroyed when the simulation cycle advances. |

For a quantity $q$ of goods left unused in inventory, the quantity of goods remaining in simulation cycle $n$ is $(1-\delta_D)^{n-1}q$. For a quantity $q$ of capital goods with no decay in time ($\delta_D = 0$), fully utilized every time production occurs, the quantity of goods remaining in simulation cycle $n$ is $(1-\delta_C)^{n-1}q$. For a quantity $q$ of goods decaying in time and also depreciating as a result of use in production, the quantity remaining in simulation cycle $n$ is $((1-\delta_D)(1-\delta_C))^{n-1}\, q$.

Note that the parameters $\delta_C$ and $\delta_D$ may differ between sectors. The setting $\delta_D = 1$, $\delta_C = 1$ is equivalent to the legacy CRISIS model.

The quantities $\delta_D$ and $\delta_C$ are referred to as the 'goods characteristics'.

### 4.9.2 Naming Conventions and Edge Cases

A good is referred to as 'durable' if it is not the case that $\delta_D = 1$ and $\delta_C = 1$ (it is not the case that the good exists for exactly one business cycle and it is not the case that the good can be used exactly once). A good is referred to as 'non–durable' if it is the case that $\delta_C = 1$ and $\delta_D = 1$ (it is the case that the good exists for exactly one business cycle and it is the case that the good can be used, either by firms or households, only once). A good is 'persistent' if $\delta_D \neq 1$ (it is not the case that the good lasts for exactly one business cycle).

The extremal configurations $\delta_C \in \{0, 1\}$, $\delta_D \in \{0, 1\}$ describe the following edge cases:

| | |
|---|---|
| $\delta_C = 1, \delta_D = 1$ | Non–durable goods. Goods produced by manufacturers exist only in the simulation cycle in which they appear. Input goods are completely destroyed by the production process and cannot be reused. Non-durable goods represent one-off/on-demand services (eg. legal and healthcare services), perishable foods, non longlife food ingredients, reagents, single-use disposable products, and potentially also early-access services. |
| $\delta_C = 1, \delta_D = 0$ | Durable goods that are destroyed when used as inputs (are non-reusable) and which (virtually) do not decay in time. This class of goods represents eg. longlife foodstuffs and food ingredients, certain chemical ingredients, and medicines. |
| $\delta_C = 0, \delta_D = 1$ | Strongly time-limited durable goods. Any quantity $q$ of these goods disappears after one business cycle has elapsed. In the meantime, these goods are unaffected when used as ingredients in the production process or by households. This class of goods represents eg. licensed products such as software, batteries, and perhaps some catalysts. |
| $\delta_C = 0, \delta_D = 0$ | Unphysical durable goods that are both (a) absolutely unaffected by the passage of time and (b) can be reused in the production process an unlimited number of times. Catalysts and (perhaps) some hard-wearing machining tools have $\delta_D \approx 0$, $\delta_C \approx 0$, $\delta_D \delta_C \neq 0$ – but not $\delta_D = 0$, $\delta_C = 0$. |

### 4.9.3 Inventory

With the addition of persistent goods (goods that do not disappear at the end of each simulation cycle) it is necessary for firms and consumers to store existing goods in inventory for extended periods. Inventory is a balance sheet asset and (by default) the value of the inventory is determined by the mean ask price of goods at market.

In a macroeconomic simulation with $N_G > 0$ sectors, $N_D$ ($0 \leq N_D \leq N_G$) of these sectors may produce durable goods. A firm should be able to buy goods

from the market and retain these goods in a dedicated inventory structure until either: (a) the product decays to negligible levels as a result of the passage of time or (b) the goods are exhausted owing to use in the production process. Similarly, households should have the ability to store and to differentiate between types of durable goods that they buy.

Inventory is a potentially powerful feature that enables:

1. Extended production chains involving retailers,
2. Goods resale and firesales (to firm closeout or otherwise),
3. Insolvency resolution by foreclosure,
4. Capital goods as a component of equity. In particular, firms can develop negative equities as a result of changes in the market price of goods.

Note that *quantities* (namely, amounts of an abstract resource) in the simulator are almost invariably represented by decorated integers or, more commonly, floating point numbers. For this reason the concept of allocation[29] applies equally to resources that are not goods eg. shares, units of labour, cash, securities, and collateral assets, among others.

For this reason an inventory can be interpreted as a general purpose object whose responsibility is to (a) provide allocation and disallocation[30] services while also (b) quoting the total volume of abstract resources that it gives access to. An inventory can be interpreted as either a physical storage medium, for instance a warehouse, in the case of durable goods, or as an abstract façade regulating access to a virtual medium, for instance a bank account. An inventory that is a view of liquid bank accounts (one or more deposit accounts supplemented by cash allocation and disallocation services) is typically referred to as a `CashLedger`.

In the standard model `Inventories` tend to provide lightweight flow recording and measurement services. A flow measurement in this context is a signed representation of the aggregate quantity of resources that have entered or left the underlying storage medium in a given timeframe. In the standard model, this flow measurement information is tallied automatically whenever certain `Inventory` services are invoked (for instance, the extraction of resources).

Flow data that is tallied by inventories is reset inbetween simulation cycles. This service enables various data reporting and debugging features to inspect the aggregate movement of resources into and out of storage media without having to inspect (potentially many tens of thousands of) individual transactions involving the inventory.

In the standard model an `Inventory` structure therefor consists of (a) allocation and disallocation services, (b) an interface to an abstract storage medium

[29] sometimes, interchangeably, referred to as 'reservation' or 'blocking' in the phraseology of the standard model.

[30] interchangeably: 'unreservation' or 'unblocking'.

(that can be manipulated by inserting or removing resources) and (c) some transaction flow recording and memory. These constituent parts are outlined in the following subsections.

#### 4.9.3.1 Allocation

Inventories provide allocation and disallocation services. Allocated resources are 'set aside' for specific future purposes. These purposes may never be realized, in which case a disallocation exercise must in due course be performed by the agent that allocated the resource in the first place[31].

If a resource was allocated as part of market order, then typically the disallocation exercise is processed automatically as a result of callbacks made from the market to which the order was submitted. For instance firms set aside their production (either as inputs or consumption goods), and any unsold goods are disallocated automatically when the goods market cancels unsatisfied/unprocessed orders.

Structures that provide allocation and disallocation services implement the `Allocating` interface. These implementations may incorporate `Allocating` as a mixin.

`Allocating` objects provide the following methods:

1. `getMaximumAllocatable()`

   Compute the maximum non–negative volume of the underlying resource that can be allocated. This method returns zero if the underlying medium is empty or if the allocation policy of the inventory will not permit additional resources to be allocated.

2. `getAllocated()`

   Compute the total quantity of the underlying resource that is currently allocated (no matter for what purpose). Note that this method returns aggregate allocations: if $x$ units of a resource were allocated for purpose $A$ and $y$ units of a resource were allocated for purpose $B$ then the total allocation is $x + y$.

3. `setAllocated(double positiveAmount)`

   Reset the total allocated quantity. The argument must be non–negative, otherwise it is silently trimmed to zero. If the value of the argument is greater than the maximum allocatable quantity (`getMaximumAllocatable`) then the size of the resulting allocation may be less than desired.

[31] failure to disallocate unused allocated resources is a potent source of modelling and agent behavioural bugs.

The value returned by this method is the quantity that has been allocated as a result of the call. The allocation policy of the `Inventory` determines the maximum amount that can be allocated.

4. `setUnallocated(double positiveAmount)`

   When called with argument $x$, this method is behaviourally equivalent to `setAllocated(V - x)` where `V` is the quantity currently stored in the underlying medium.

5. `changeAllocatedBy(double volume)`

   Increment the allocated quantity by a signed amount. This method is behaviourally equivalent to `setAllocated(getAllocated() + x)`, where `x` is the argument (which may be negative).

6. `getUnallocated()`

   This method is behaviourally equivalent to `V - getAllocated()`, where `V` is the quantity of the resource that is currently stored in the underlying medium.

7. `allocateAll()`

   This method is behaviourally equivalent to `setAllocated( getMaximumAllocatable())`.

8. `disallocateAll()`

   This method is behaviourally equivalent to `setAllocated(0)`.

Any specializations, supplementary services and/or limitations that may apply to the above over and above those listed, and which depend on the implementing class, are to be documented by the implementation.

When new resources (eg. goods, labour) are added to an inventory, it is understood that these resources are initially in an unallocated state. If resources are withdrawn from an inventory, and as a result of this withdrawal the total quantity of resources in the inventory is less than the allocated amount, then the allocated amount is reduced to the level that can be allocated.

Note that the above interface is subject to change. The following features are desired and would not be unduly difficult to implement:

1. **Lapse conditions**
   Failure to disallocate agent resources has proven to be one of the most fruitful sources of bugs in economic agent based modelling. This issue is characterized by a sudden or gradual reduction in the availability of resources, perhaps leading to a recession, and ultimately to extremal macroeconmic and financial system behaviour.

Agents very often assume that new resources are initialized in unallocated states. However if unsold goods from previous business cycles remain allocated in inventory then these goods will not be posted as new market orders and therefor not sold. This incurs a financial penalty to the agent that failed to properly disallocate its inventory. Similar effects have been observed in stock markets (note that any action that reduces the number of shares in circulation reduces aggregate supply, which in turn increases price, as demand is concentrated over a smaller supply).

In order to address this problem it is thought that inventories should provide timestamped allocation services (concretely: agents must indicate for how long they intend to allocate resources). Allocations should be subject to natural expiry dates, meaning that behavioural errors and/or defects in agent strategies resulting from their failure to disallocate resources may be more limited.

2. **Allocation receipts**
   At present all allocations within a particular inventory are aggregated, in the sense that one allocation of size $x > 0$ followed by another allocation of size $y > 0$ results in a memoryless allocation of size $x + y$ that has no recollection of the fact that it was formed of two pieces (and likewise for disallocations).

   For this reason if the volume of the resource stored in the inventory (which may be cash) reduces unexpectedly, then the inventory is unable to modify existing allocated resources pro–rata/selectively according to the purposes for which these resources were allocated. This complicates forclosure activities, and bankruptcy resolution policies, and results in race conditions in which the soonest event to claim allocated resources is most likely to receive the promised amount (at the expense of later claims). A simple solution to this issue is to label resource allocations by purpose, so that future claims against allocated resources are binned within the inventory structure. This modification would however noticeably alter the above specification.

#### 4.9.3.2 Production Goods

Inventory structures provide goods allocation (reservation) and disallocation services. Goods set aside for sale can be marked as allocated in the inventory store that contains them. Firms typically have dedicated inventories for each distinct type of production good.

#### 4.9.3.3 Inventory Valuation

The equity value $V(I)$ of an inventory asset $I$ is taken to be:

$$V(I) = \sum_{1 \leq i \leq N_G} \bar{p}_i q_i$$

where $\bar{p}(i)$ is the mean ask price (weighted by order size) of goods of type $i$ observed during the last goods market processing session and $q$ is the quantity

of goods of type $i$ stored in the inventory asset. This valuation mechanism is an algorithm and a model parameter in the meaning of §4.3.

### 4.9.4 Economic Cost

The future utility of durables should be discounted against the ask price when deciding whether or not to buy durable goods. For instance: one unit of single-use non-durable goods is not expected to have as much utility as one unit of otherwise interchangeable durable goods for which $\delta_D \approx 0$ and $\delta_C \approx 0$ (which will last for a very long time and will remain usable throughout their lifecycle). If a firm can substitute durable for non-durable goods at comparable prices then buying durables is likely to be profitable in the long term. Firms should take this into account when optimizing their production problem and deciding which inputs to buy. How should buyers calculate the effective economic cost (the generalised price) of a durable good given its promised future utility? How should firms discount the future utility of durable products from the market ask price?

In order to answer this question the starting point is to define the economic cost $p^*$ as the value of a durable good lost in one production period. The evaluation of the durable good is a modeller choice: it can happen either at market price (or last market price) or at book value. Both methods are simple and the difference in sensible only if the market price of the durable good is highly volatile. For the moment we chose a market evaluation.

Let us suppose for simplicity that the market price of one unit of durable goods is $p = 1$. The quantity of durable goods consumed during a simulation cycle depends on how the decay rate is coded. Suppose, following the above definitions of exponential decay, that the good has a time decay $\delta_D$ and a usage decay $\delta_C$. The time decay applies on the whole stock of durable good and the usage decay applies only the part of the good actually used in production. The timing at which apply the decays is important. The time decay happens at the end of the period and the usage decay happens during production.

Suppose $q \geq 0$ is the available stock of durable goods and $0 \leq \xi \leq q$ is the amount of the durable good utilised during production. The usage (consumption) decay implies that after production the firm will be left with $q - \delta_C \xi$ units of stock. Defining $\omega = \xi/q$ to be the rate of utilisation of the durable good, we have:

$$q \quad \stackrel{\text{production}}{\longrightarrow} \quad (1 - \delta_C \omega) q.$$

Durable goods will decay in time at the end of the simulation timestep, implying that the remaining stock $q_{t+1}$ of durable good at the onset of time $t + 1$ is:

$$\begin{aligned} q_{t+1} &= (1 - \delta_D) \cdot (1 - \delta_C \omega) q_t \\ &= (1 - \delta_D) \cdot (q_t - q_t \delta_C \omega) \\ &= q_t - q_t \delta_C \omega - q_t \delta_D + q_t \delta_C \delta_D \omega. \end{aligned}$$

Thus the amount of durable good lost during the business cycle $[t, t+1)$ is:

$$q_{t+1} - q_t \quad = \quad q_t\delta_C\omega + q_t\delta_D - q_t\delta_C\delta_D\omega.$$

We realize the total economic cost (the generalized market price) of the durable good as:

$$p_t^* = (\delta_C\omega + \delta_D - \delta_C\delta_D\omega) \cdot p_t,$$

where $p_t$ is the market selling price. In the standard model $\omega \equiv 1$ (all input goods are utilized for production).

Note that the economic cost of goods with characteristics $\delta_C = 0$, $\delta_D = 0$ is $p^* = 0$. The unphysical regime $\delta_C = 0$, $\delta_D = 0$ is typically prohibited by the standard model because, in principle, one unit of goods of this special type empowers a firm to never buy from at least one sector again: this has the effect of undermining the macroeconomic input-output network.

An alternative method to compute the the economic cost $p^*$ of durables is as follows: if a firm buys $q_t$ units of a durable good at time $t$ then, assuming complete utilisation forever after, the firm effectively gets to use:

$$\begin{aligned} Q &= q_t + (1-\delta_D)(1-\delta_C\omega)q_t + (1-\delta_D)^2(1-\delta_C\omega)^2 q_t + \ldots \\ &= \frac{q_t}{1-(1-\delta_D)(1-\delta_C\omega)}, \end{aligned}$$

units in future, bought at market price $p_t$ per unit. The purchase cost was $p_t q_t$, therefore the effective unit price $p^*$ (the economic cost of the acquisition) was

$$\begin{aligned} p^* &= \frac{p_t q_t}{\frac{q_t}{1-(1-\delta_D)(1-\delta_C\omega)}} \\ &= p_t(\delta_D + \delta_C\omega - \delta_D\delta_C\omega). \end{aligned}$$

#### 4.9.4.1 A Numerical Example

Suppose that $q_t = 100$, $\omega = 0.9$, $\delta_C = 0.02$, $\delta_D = 0.01$. During the production process the usage decay reduces the stock to $(1 - 0.02 \cdot 0.9) \cdot 100 = 98.2$ units. Decay in time subsequently applies to 98.2, so the remaining stock at the end of the business cycle is $q_{t+1} = (1 - 0.01) \cdot 98.2 = 97.218$. The reduction in durable stock is $100 - 97.218 = 2.782$. From the formula above: $q_{t+1} - q_t = 100 \cdot 0.02 \cdot 0.9$ + $100 \cdot 0.01$ - $100 \cdot 0.02 \cdot 0.01 \cdot 0.9 = 2.782$.

### 4.9.5 Scheduling

The order of durable goods processing (including labour market activities and production) for the `Macrofirm` agent type is as indicated by the following subsections. `Macrofirm` is the default implementation of the firm agent type in the standard model.

CAPTIALIZED headers indicate the onset of named intervals in the simulation schedule order. The sequential order of the following subsections indicates the

chronological order of event/decision processing for the `Macrofirm` agent.

Common terms (subscripted by goods type $i$) include:

| | |
|---|---|
| $N_S$ | The number of differentiated goods types (the number of sectors in the simulation). The subscript $i$ is confined to the range $1 \leq i \leq N_S$. |
| $t$ | The simulation schedule time, $t \geq 0$. |
| $c$ | The cycle index of the simulation, $c = \lfloor t \rfloor$. |
| $q_t^*$ | The firm target (ideal) goods production: a quantity the firm aims produce in cycle $t + 1$. |
| $q_t$ | The actual firm production yield (quantity) at time $t$. |
| $g$ | A quantity of goods stored in inventory. |
| $d$ | The demand for new goods from the markets. |
| $\ell$ | A quantity of labour. |
| $\omega$ | The wage to be paid per unit labour. |
| $p$ | A market price of a good (per unit). |
| $(\delta_D, \delta_C)$ | Durability characteristics of a good. |
| $p^*$ | The economic cost per unit good (the generalized price, §4.9.4). |

#### 4.9.5.1 Simulation Begins

At time $t = 0$ all firms are endowed with a quantity $q_{\text{init.}} = q_0$ of production goods. $q_0 \geq 0$ is a model parameter in the meaning of §4.3 and may be heterogeneous among firms.

#### 4.9.5.2 Processing Loop

Once per simulation cycle $c = 0, 1, 2...$ the following events and firm decisions are executed sequentially in order:

#### 4.9.5.3 SELL_ORDERS

Producers offer all available goods to the markets for sale.

#### 4.9.5.4 FIRM_DECIDE_PRODUCTION

All firms:

1. Compute a new target production $q_t^*$.
2. Compute a new goods unit selling price $p_t$ (bid price per unit).
3. Compute the economic cost $p_{i,t}^*$ for all types of good ($1 \leq i \leq N_S$):

$$p_{i,t}^* = \Big(1 - (1 - \delta_{Di})(1 - \delta_{Ci})\Big) \cdot p_{i,t} \quad = \quad \Big(\delta_{Di} + \delta_{Ci} - \delta_{Di}\delta_{Ci}\Big) \cdot p_{i,t},$$

where (a) $p_{i,t}$ is the market ask price of good $i$ at time $t$, and (b) $(\delta_{Ci}, \delta_{Di})$ are the durability characteristics of that good.

4. Decide labour $\ell$ and goods input demands $(d_1, d_2, \ldots d_{N_S})$ required for production, by:

   (i) Optimising the firm production function such that total economic costs $\overline{c}$ are minimized:

   $$\overline{c}(\ell, d_1, d_2 \ldots d_{N_S}) \quad = \quad \sum_{i}^{N_S} p_i^* d_i + \omega\ell.$$

   This optimisation yields (a) a labour input demand $\ell \geq 0$, and (b) a tuple of demands $D_{\text{goods}} = (d_i)_{i=1}^{N_S}$ for goods of all available types. Note that the optimal labour input $\ell$ may be less than the existing firm workforce size, if labour contracts are extended, and therefor the firm may need to fire staff in order to realize its efficiency targets.

   (ii) Deducting existing inventory from goods demands:

   $$D_{\text{goods}} = \{d_i\}_{i=1}^{N_S} \quad \longrightarrow \quad \{\max(d_i - g_{i,t}, 0)\}_{i=1}^{N_S},$$

   where $g_{i,t}$ is the quantity of good $i$ already owned by the firm.

   (iii) Computing the total cash cost $L$, where

   $$L = \sum_{i}^{N_S} p_i d_i + \ell\omega,$$

   to purchase $D_{\text{goods}}$ and labour $\ell$, (note that $\ell$ and $D_{\text{goods}}$ are optimal solutions minimising the *economic* costs $\overline{c}$ incurred, not the acquisition cost to the firm. It is not necessarily the case that $L$ is the minimum cash cost required in order to realize the current firm production target).

   The producer may only have a fraction of the cash $L$ at its disposal. In order to raise the remaining cash, the producer applies for new loans on the credit markets.

#### 4.9.5.5 FIRM_BUY_INPUTS

Firms seek to buy new input goods and new labour by:

1. Computing the liquidity $L^* \leq L$ available for production (the desired/ideal commercial loan sum may not have been secured on the credit markets).
2. Reducing $D_{\text{goods}}$ and $\ell$ by the factor of $L^*/L$ (assuming linear returns to scale).
3. Submitting labour and goods market bid orders.

#### 4.9.5.6 PRODUCTION

Firms produce new goods as follows:

1. All quantities $g_{i,t}$ of goods of all types in inventory are utilised for production.

2. Following production, $(1 - \delta_C)g_{i,t}$ units of each good remain in inventory.

3. Production yields $q_{c+1} \geq 0$ units of new goods. $q_{c+1}$ is inaccessible in inventory until simulation cycle $c + 1$.

4. Existing goods market ask orders are updated in preparation for domestic consumption. All remaining production goods not sold via the input-output network (namely, sold to other firms for production) are allocated for sale to households.

#### 4.9.5.7 GOODS_CONSUMPTION_MARKET

Households buy goods of all types from the market.

#### 4.9.5.8 AFTER_ALL

Every stock $g_{i,t}$ in every inventory is reduced to $(1 - \delta_D)g_{i,t}$.

With respect to firm event/decision processing, the simulation schedule now repeats from §4.9.5.3.

## 4.10 Input–Output Networks

For Cobb Douglas and Leontief input-output network economies, it is necessary for the user to specify a matrix of 'technological weights' $\{w_{ij}\}$ (one weight per sector *for each* peer sector with which to trade). Ordinarily the technological weights $w_{ij}$ are invariant in time, but this assumption is neither necessary nor binding.

In the Cobb Douglas case it is strictly necessary that $\sum_j w_{ji} \equiv 1$ for all columns $i$. This normalization ensures proper scaling behaviour for the Cobb Douglas production function. In the Leontief case, normalisation is not strictly necessary.

In both these cases the user must specify a square matrix $w = (w_{ij})$, however the use of and the interpretation of the components $w_{ij}$ differs according to the underlying production function used. In the Cobb Douglas case these weights are hints as to the preferences firms/establishments in each sector should have when buying from other firms/sectors. In the Leontief case these weights describe strict ratios to be observed when ordering goods from the market. It is not difficult to conceive of other or modified production functions for which the matrix $w$ is utilized in different ways, but ultimately there is a requirement for a square matrix $w$ whose elements describe (in some way) relative propensities for inter-sector trade.

The form of the matrix $w = w_{ij}$ is of great interest to users. By default we offer the following strategies to generate internal (and import external) input-output matrices into the standard model:

1. **Random Matrices**
   Random input-output networks whose elements are drawn from independent identically distributed random variables. This module is illustrated in §4.10.1.

2. **Diagonal Matrices**
   Trivial (degenerate) input-output networks for which inter-sector trade is forbidden. This module is outlined in §4.10.2.

3. **Homogeneous Matrices**
   An all-to-all (complete graph) input-output network in which no sector is biased toward buying goods for one sector over another. This network describes the most flexible input-output network configuration and also has the least substructure. This setup is described in §4.10.3.

4. **Self–Biasing Matrices**
   A parametrized perturbation of the homogeneous all-to-all input-output network configuration (§4.10.3). This network configuration describes an economy for which intra-sector trade is either preferred to, or less likely than, inter-sector trade. This scheme is illustrated in §4.10.4.

5. **Production Chains**
   An input-output network with a natural differentiation between 'root producers', who do not depend directly on the outputs of other firms in the economy (eg. mining and quarrying firms and firms whose businesses are to extract raw materials), production chain firms (who require unrefined inputs and convert these into increasingly specialized products), and retailers (who sell to consumers). This system is discussed in §4.10.5.

6. **Matrix Files**
   Fully customizable matrices specified by text files. This module is discussed in §4.10.6.

### 4.10.1 Random Matrices

$w$ is a matrix whose elements are all independent identically distributed strictly positive random variables. The underlying distribution is customizable. $w$ is normalized only after each element of the matrix has been computed.

### 4.10.2 Diagonal Matrices

The simplest possible form for the input-output matrix, this matrix describes a network in which no sector buys goods/services from any other sector:

$$w = \begin{bmatrix} 1 & 0 & 0 & \dots \\ 0 & 1 & 0 & \dots \\ 0 & 0 & 1 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} = I_{N_S},$$

where $I_n$ is the identity matrix of dimension $n$.

This configuration describes a degenerate input-output network in which each firm/establishment may only use its own class of goods (and labour) for production.

### 4.10.3 Homogeneous Matrices

$w$ is a homogeneous matrix whose elements are all identical. This input-output matrix represents an 'all-to-all' configuration for which any member of the network can buy goods/services from any other member. No member of the network prefers one peer over another.

$w_{ij} = 1$ for each pair $(i, j)$ before normalization:

$$w \;=\; \begin{bmatrix} 1 & 1 & 1 & \dots \\ 1 & 1 & 1 & \dots \\ 1 & 1 & 1 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} / N_S \;=\; \mathbb{1}/N_S,$$

where $N_S$ is the number of sectors (the dimension of the matrix) and $\mathbb{1}$ is the square matrix of ones.

### 4.10.4 Self–Biasing Matrices

UK trade observations (eg. the official ONS Combined Use Input–Output Analytical Tables[32]) indicate that some sectors (notably manufacturing, and financial intermediation) are composed of firms/establishments that prefer to trade amongst themselves moreso than other sectors. The simple self-biasing (SSB) input-output matrix represents a network for which intra-sector trade is treated differently from, and potentially dominates, inter-sector trade:

$$w \;=\; \begin{bmatrix} \alpha & \beta & \beta & \dots \\ \beta & \alpha & \beta & \dots \\ \beta & \beta & \alpha & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \cdot \frac{1}{\alpha + (N_S - 1)\beta} \;=\; \frac{\mathrm{diag}(\alpha - \beta) + \beta\mathbb{1}}{\alpha + (N_S - 1)\beta},$$

for customisable non-negative numbers ($\alpha$, $\beta$) not both zero. $\mathbb{1}$ is the square matrix of ones and $N_S$ is the number of sectors (the dimension of the matrix). The expression $\alpha/\beta - 1$ is called the 'sector self-bias'.

Note that (a) $\alpha = \beta = 1$ is equivalent to a homogeneous input-output network §4.10.3 with zero bias and that (b) $\beta = 0$, $\alpha = 1$ is a diagonal matrix describing a trivial network in which no sector buys goods/services from any other sector.

[32] http://www.ons.gov.uk/ons/rel/input-output/input-output-analytical-tables/2005/index.html. Crown Copyright.

### 4.10.5 Production Chain Matrices

Real economies tend to be described by a production chain in which a set of distinguished firms/establishments produce primitive raw materials (for instance quarrying and mining firms) that are modified to increasing degrees of specialization by other firms whose businesses increasingly depend on inputs from their peers.

A natural example of this process is a car manufacturing subeconomy in which the root producers are metal and oil extraction firms, followed by oil refiners (plastic granulate and petroleum products) and metal smelting/refiners, machine tool producers, production lines and robotic assemblies and then vehicle distribution and car retailer firms from which consumers buy the final product. Production chain economies can sometimes be realized as a set of root producer firms, who employ a relatively huge amount of labour, followed by a sequence of firms whose businesses employ relatively less labour and relatively more input goods/services. Consumers are limited to buying goods/services from retailers at the end of the production chain.

In this case $w$ can be realized as a banded matrix as follows:

$$w = \begin{bmatrix} \mathbf{1} & \frac{1}{N_R} & 0 & 0 & 0 & 0 & \frac{1}{N_R} & 0 & \dots \\ 0 & 0 & \frac{1}{N_R} & 0 & 0 & 0 & 0 & \frac{1}{N_R} & \dots \\ 0 & 0 & 0 & \frac{1}{N_R} & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & \frac{1}{N_R} & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & \frac{1}{N_R} & 0 & 0 & 0 & \mathbf{1} & \frac{1}{N_R} & 0 & \dots \\ 0 & 0 & \frac{1}{N_R} & 0 & 0 & 0 & 0 & \frac{1}{N_R} & \dots \\ 0 & 0 & 0 & \frac{1}{N_R} & 0 & 0 & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

where $N_R > 0$ is the number of root producers in the production chain (highlighted in bold).

### 4.10.6 Fully Customizable Matrices

Custom input-output matrices may also be parsed from a text file. This text file may be delimited by commas, spaces, or several other form of English punctuation, with one matrix row given per line. Column normalization is applied to the resulting matrix automatically.

### 4.10.7 Desired: Degree Distribution Networks

Degree Distribution Networks (DDNs) are input-output networks for which the edge set of each vertex is randomly generated. The number of edges connected to each vertex is drawn from a distribution. DDNs are thought to be reasonable and realistic representations of real economies. We intend to introduce this as standard for the simulator. The Pareto Distributed Degree Network[33] is

[33] Acemoglu et. al., The Network Origins Of Aggregate Fluctuations

potentially the most interesting such structure. This feature requires one or a few hours to implement.

## 4.11 Labour Markets

Labour markets are structures whose responsibility is to facilitate interactions between employers and employees with a view to forming labour contracts. In the standard model, firms and households buy and sell units of labour (respectively) in exchange for wages. Government, commercial banks and other agent types may optionally be involved in the labour market bidding process. By default all labour contracts in the standard model are renewed one per simulation cycle. Labour contracts in the standard model do support extended maturities (namely, employment contracts lasting for multiple simulation cycles) however this feature is not currently used by any of the agents provided with the simulator.

The labour market in the standard model is a batch processed complete bipartite graph whose disjoint node groups consist of `Employers` and `Employees`, respectively. `Employers`, who intend to buy labour, submit ask orders to the labour market and specify the maximum wage per unit labour they are willing to pay for the privilege. `Employees` submit bid orders to the market and specify the minimum wage per unit labour at which they are willing to work. The market matches these bid and ask orders once per simulation cycle and attempts to execute the resulting contracts.

### 4.11.1 Labour Market Entry Point

When submitting orders the behaviour of the labour market is as follows:

1. If a party submits an ask order then the party must be an `Employee`, otherwise the order is rejected by the market (an exception is raised).
2. If the party submits a bid order then the party must be a `CashAllocating`[34] `Employer`, otherwise the order cannot be processed.
3. `Employees` must be able to allocate sufficient labour for the order, otherwise the order is rejected by the market on the basis that the ask party cannot deliver the promised resources.
4. `Employers` are required to allocate sufficient cash at the wage specified by their bid price (the maximum price to pay per unit new labour). If the `Employer` fails to allocate this sum but can allocate a lesser nonzero sum then the order size is reduced proportionally. If the `Employer` cannot allocate any cash then the order is rejected by the market.

The bid and ask wage (price per unit labour) must be positive for all market participants.

[34] An agent with the ability to allocate cash to use for a specific future purpose, in this case employee remuneration.

### 4.11.2 Background and Legacy Model

The legacy mark one (§3) model was believed to be subject to numerical problems caused by floating point rounding errors in cash and labour allocation. Many markets in the standard model require participants to allocate (set aside) all resources they intend to trade at the time the order is submitted. These allocated resources are in some cases ultimately represented by encapsulated floating point numbers. For this reason the size of the allocation and the quantity of tradeable resources owned by the participant may be different to the order of the unit of least machine precision. In the legacy model this discrepancy was believed to be sufficient to cause the market to fully reject an order. The current implementation of the labour market is such that order sizes are automatically scaled down in the event that a participant cannot allocate the required resources but can allocate a lesser nonzero quantity.

### 4.11.3 Contract Matching

The labour market is processed by an `Employer-Employee` matching algorithm. The matching algorithm is required to do the following:

1. accept sets of bipartite input nodes $E$ (`Employers`) and $W$ (`Employees`) who have submitted bid orders $B$ and ask orders $A$ (respectively) as follows:

   (a) bid orders $B$ of the form $(e, \omega, d)$ where $e \in E$ is an `Employer`, $\omega \geq 0$ is the bid price per unit labour and $d > 0$ is the amount of labour required by the `Employer` (the demand). Write $\omega = \omega(e)$ for the ask price specified by $e$ and $d(e)$ for the demand.

   (b) ask orders $A$ of the form $(e^*, \omega^*, s)$ where $e^* \in W$ is an `Employee` (a worker), $\omega^* \geq 0$ is the ask price per unit labour and $s > 0$ labour supply offered by the `Employee`. Write $\omega^*(e^*)$ for the ask price specified by $e^*$ and $s(e^*)$ for the labour supply.

   Note that `Employers` $e \in E$ submit labour orders to the market, not orders that are required to be fulfilled by specific agents $w \in W$. Likewise `Employees` $w \in W$ submit labour orders to the market, not orders that are required to be fulfilled by particular $e \in E$.

2. formulate a set $X$ of matchings $X = \{(e, e^*, \ell, w)\}$ where $e \in E$, $e^* \in W$, $\ell > 0$ such that:

   (a) $w \leq \omega(e)$ (the `Employer` pays no more than the bid price),

   (b) optionally: $w \geq \omega^*(e^*)$ (the `Employee` receives no less then the ask price).

   $X$ describes a set of possible trades between `Employers` and `Employees`. When processing $X$ it is often of greater importance that the buyer (the `Employer`) pays no more than the bid price than that the seller (the `Employee`) receives no less than the ask price. This is because the buyer is very often required to set aside (allocate) cash resources at the time the order is submitted to the market. If the execution price $w$ selected by the market is greater than the bid price $\omega(e)$ then the transaction may fail as a result of insufficient allocated buyer cash.

3. the sum of the sizes of all labour contracts in which any $e \in E$ is involved is less than its labour demand, $d(e)$:

$$\sum_{x \in X \,:\, e \in x} \ell(x) \leq d(e)$$

4. the sum of the sizes of all labour contracts in which any $e^* \in W$ is involved is less than its labour supply, $s(e)$:

$$\sum_{x \in X \,:\, e^* \in x} \ell(x) \leq s(e^*)$$

Note that it is acceptable for an `Employee` (which may be modelled on the macro scale) to provide labour to multiple `Employers`. The latter two conditions in the about specification ensure that no individual unit of labour is employed twice.

Other than that the labour market is (a) required to find a solution satisfying the above specification, and (b) is not allowed to modify the states of the market participants, the labour market may compute a solution by any means. The means through which the labour market computes a solution are to be documented be the implementation.

### 4.11.4 Rationing Precondition

Most labour market implementations in the standard model currently operate on the premise that a preconditioning step designed to reduce aggregate supply and/or reduce aggregate demand should first be applied to $B$ and/or $A$. This preconditioning step is required to reduce aggregate supply among nodes $w \in W$ and/or reduce aggregate demand among nodes $e \in E$ in such a way that the resulting total supply is equal to the resulting total demand. It is expected, but not required, that this rationing should be applied to one but not both groups.

This preconditioning is loosely referred to as 'rationing'. The rationing algorithm is therefor required to accept sets of disjoint nodes $E$ and $W$ with bid/ask orders $B$, $A$ and either exclude some nodes from the market and/or specify limits on the size of the orders that will be fulfilled for each market participant. The labour market rationing algorithm is an object and is a model parameter in the meaning of §4.3.

Concretely, the rationing algorithm is required to coerce the bid/ask orders $B$, $A$ to a new set of rationed orders $A^*$, $B^*$ satisfying the following constraints:

1. For all $x \in B$ there exists a modified order $x^* \in B^*$ of the form $\{e, \omega, d\}$ such that $0 \leq d^*(x) \leq d(x)$.
2. For all $x \in A$ there exists a modified order $x^* \in A^*$ of the form $\{e^*, \omega, s\}$ such that $0 \leq s^*(x) \leq s(x)$.
3. Aggregate supply and aggregate demand are equal for $B^*$ and $A^*$:

$$\sum_{x \in B^*} d^*(x) = \sum_{x \in A^*} s^*(x)$$

Other than that the rationing algorithm is required to find a solution $A^*, B^*$ satisfying the above constraints, the algorithm may compute a solution by any means. The means through which the rationing algorithm computes the solution is documented by the implementation.

The standard model provides the following stock implementations of the labour rationing algorithm:

1. **Homogeneous Rationing**
   Perhaps the simplest possible biasless rationing algorithm. This algorithm reduces the total labour supply or the total labour demand specified by existing orders, whichever is greater, by applying a fixed multiplier to the sizes of all bid orders or to the sizes of all ask orders (but not both). This algorithm is detailed in §4.11.4.1.

2. **Inhomogeneous Rationing**
   A rationing process in which some orders are randomly curtailed in order to match supply and demand. This algorithm is illustrated in §4.11.4.2.

3. **Greed Rationing**
   A rationing algorithm for which participants who bid the lowest wages or who set the highest ask prices are penalized. This algorithm is outlined in §4.11.4.3.

#### 4.11.4.1 Homogeneous Rationing

Homogeneous rationing is perhaps the simplest possible biasless rationing algorithm. This algorithm reduces aggregate labour supply or reduces aggregate labour demand (but not both) by applying a fixed multiplier to exactly one of the `Employee`/`Employer` node groups as follows:

1. Compute $d = \sum_{e \in E} d(e)$ and $s = \sum_{e^* \in W} s(e^*)$.
2. If $d = s$, then stop.
3. If $d > s$ then set $\xi = s/d$ and set $X = B$.
4. Otherwise if $d < s$ then set $\xi = d/s$ and set $X = A$.
5. For each order $x \in X$, replace $x$ with $\{p, \omega, \xi o\}$ where $p$ is the participant in $x$, $\omega$ is the bid/ask price specified by $x$, and $o(x)$ is the size of the order ($o = s(x)$ or $o = d(x)$).

The resulting order sets $(B, X)$ or $(X, A)$ are rationed.

Note that there are no internal random processes in effect in this algorithm and therefor it is deterministic.

#### 4.11.4.2 Inhomogeneous Rationing

Inhomogeneous rationing is a generalization of the homogeneous rationing algorithm §4.11.4.1 with uneven rationing of order sizes. This algorithm reduces the aggregate labour supply or reduces the aggregate labour demand (but not both) by randomly reducing the sizes of a sufficient number of orders in exactly one of the `Employer`/ `Employee` node group as follows:

For $x \in A \cup B$ write $r(x, \xi)$ for the function $r(x, \xi) = \{p, \omega, {}_0|\xi|_1 o\}$ where $p$ is the participant in $x$, $\omega$ is the bid/ask price specified by $x$, $o$ is the order size ($o = d(s)$ or $o = s(x)$), and ${}_0|q|_1$ is shorthand for $\min(\max(q, 0), 1)$.

1. Compute $d = \sum_{e \in E} d(e)$ and $s = \sum_{e^* \in W} s(e^*)$.
2. If $d = s$, then stop.
3. If $d > s$ then set $X = B$, $t = s$, $a = d$. Otherwise set $X = A$, $t = d$, $a = s$.
4. Set $\xi = t/a$.
5. For each $x \in X$ apply $x \mapsto r(x, \xi + \Omega\epsilon \cdot (1 - \xi))$ where $\epsilon \sim U[0, 1)$ is drawn from a random uniform distribution on the unit interval and $\Omega \geq 0$ is a customizable model parameter.
6. Compute $a^* = \sum_{x \in X} o(x)$.
7. Set $\xi = t/a^*$.
8. For each $x \in X$ apply $x \mapsto r(x, \xi)$.

The resulting order sets $(B, X)$ or $(X, A)$ are rationed.

Note that the expression $\epsilon \sim U[0, 1)$ depends on a random number generator and therefor in general this process will require a random number seed in order to behave deterministically.

The parameter $\Omega > 0$ is a model parameter in the meaning of §4.3. The setting $\Omega = 0$ is equivalent to the homogeneous rationing algorithm §4.11.4.1 because steps $(6-8)$ are degenerate in this edge case. $\Omega \sim 1$ is likely to generate a rationing in which some market participants are more severely penalized than others, whereas $\Omega \sim 0$ is likely to generate a mildly inhomogeneous rationing.

Note that this process is a two–pass algorithm with $O(\max(|\mathcal{A}|, |\mathcal{L}|))$ complexity.

#### 4.11.4.3 Greed Rationing

The greed rationing algorithm was originally devised as a mechanism to regulate wages in the labour market. The aim of this rationing algorithm is to penalize nodes whose price offerings are in excess of the mean price observed when processing the market. In the context of the labour market, this means that either

`Employers` who offer low wages per unit labour or `Employees` who require high wages per unit labour are most likely to be excluded from the marketplace. This algorithm is an experimental work in progress and is included for completeness only.

For $x \in A \cup B$ write $r(x, \xi)$ for the function $r(x, \xi) = \{p, \omega, {}_0|\xi|_1 o\}$ where $p$ is the participant in $x$, $\omega$ is the bid/ask price specified by $x$, $o$ is the order size ($o = d(s)$ or $o = s(x)$), and ${}_0|q|_1$ is shorthand for $\min(\max(q, 0), 1)$.

For an array of nodes $X$, write $S^+$ for a sorting algorithm whose objective is $\omega(x_i) \leq \omega(x_{i+1})$. $S^+$ sorts $X$ ascending with respect to the order price $\omega(x)$. Write $S^-$ for a sorting algorithm whose objective is $\omega(x_i) \geq \omega(x_{i+1})$. $S^-$ sorts $X$ descending with respect to the order price $\omega(x)$.

1. Compute $d = \sum_{e \in E} d(e)$ and $s = \sum_{e^* \in W} s(e^*)$.
2. If $d = s$, then stop.
3. If $d > s$ then set $X = B$, $t = s$, $a = d$ and $S = S^-$. Otherwise set $X = A$, $t = d$, $a = s$ and $S = S^+$.
4. Set $\xi = t/a$.
5. Apply $S$ to $X$ (sort $X$ with respect to its order prices). Write $X_i$ for the $i$th element ($1 \leq i \leq |X|$) of $X$ after sorting.
6. Set $p = |X|$ and $R = (1 - \xi)a$.
7. If $o(X_p) \leq R$ then map $R \mapsto R - o(x)$, set $o(x)$ to zero and go to (8). Otherwise, map $o(x) \mapsto o(x) - R$, set $R = 0$ and stop.
8. Decrement $p$. Go to (7).

The resulting order sets $(B, X)$ or $(X, A)$ are rationed.

This rationing algorithm systematically trims order sizes, starting with the least reasonable order (with lowest bid price or highest ask price) and then continues to the next least reasonable order until the rationing is complete.

Note that there are no internal random processes in effect in this algorithm and therefor it is deterministic.

In general the cost of sorting an array of $n$ comparable datums is $O(n \ln n)$ flops. The cost of the instruction loop in steps $(6 - 8)$ is $O(n)$ flops. The total cost of this rationing algorithm is therefore $O(n \ln n)$ flops.

### 4.11.5 Matching

After rationing has been applied to the labour market participants (§4.11.4) the nodes $E$, $W$ satisfy:

$$\sum_{e \in E} d(e) = \sum_{e^* \in W} s(e^*) = \Lambda$$

where $\Lambda \geq 0$ is the total trade (in units of labour) required by both `Employers` and `Employees`.

At present most implementations of the labour market in the standard model require a rationing preconditioning step followed by contract matching. The contract matching step is required to create a set a set $X = \{(e, e^*, \ell, w)\}$ of matchings where $e \in E$, $e^* \in W$, $\ell > 0$ such that:

1. $w \leq \omega(e)$ (the `Employer` pays no more than the bid price),
2. $w \geq \omega^*(e^*)$ (the `Employee` receives no less then the ask price),
3. $\sum_{x \in X : e \in x} \ell(x) = d(e)$ (labour demand is satisfied for all `Employers`),
4. $\sum_{x \in X : e^* \in x} \ell(x) = s(e)$ (labour supply is satisfied for all `Employees`),
5. $\sum_{x \in X} \ell(x) = \Lambda$ (trade is maximized).

Note that condition (5) is equivalent to (3) and (4).

The matching step is an object and is a model parameter in the meaning of §4.3.2. Other than that the contract matching step is required to identify a set $X$ satisfying the above constraints, matching algorithms may identify $X$ by any means. The means used to derive $X$ are documented by the implementation.

The standard model uses a matching step named 'forager'.

#### 4.11.5.1 Forager

The forager node matching algorithm is a simple efficient random matching process. This algorithms assumes that a rationing preconditioning step has been applied to the market participants (§4.11.4) beforehand.

This matching algorithm abstractly interprets the first (left) bipartite node group as a set of buyers and the second (right) bipartite node group as a set of sellers. The buyer node group is initially shuffled. Trades are now executed between seller/buyer pairs, starting at the top of both columns in the following table and working downwards systematically:

| shuffled buyers (intent) | sellers (intent) |
|---|---|
| (a) buy quantity $q_1$ at price $p_1$ | (i.) sell quantity $q'_1$ at price $p'_1$ |
| (b) buy quantity $q_2$ at price $p_2$ | (ii.) sell quantity $q'_2$ at price $p'_2$ |
| $\vdots$ | $\vdots$ |

The order of iteration is as follows: beginning with buyer (a) try to establish a trade with seller (i.), then by seller (ii.), and so on. Remove sellers from the system whenever a seller's supply is exhausted by this process. When the buyer

demand is exhausted, remove the buyer for the system. Repeat this process with buyer (b), and so on in the list of remaining buyers. Whenever a potential trade $(\min(q, q'), p, p')$ is identified the matching algorithm makes the following assumptions:

1. if the buyer price is less than the seller price, $p < p'$, then the execution price is taken to be $p$ (favouring the buyer)[35]
2. if the seller price $p' < p$, then negotiation between the buyer and the seller is assumed. The resulting execution price is taken to be $(p' + p)/2$ (favouring the seller).

Note that for matching problems with homogeneous prices $p = p' = \rho$, where $\rho \geq 0$ is a constant, the execution price of all contracts is equal to $\rho$.

The name of this algorithm is borne out of its conceptual similarity to the behaviour of hunter gatherers, who randomly encounter natural resources and accept, out of necessity, whatever resources they find.

Note that the shuffling operation depends on a random number generator and therefor in general this process will require a random number seed in order to behave deterministically.

In general the cost of shuffling an array of $n$ datums is $O(n)$ flops or worse. The standard model uses a single-pass in-place biasless Fisher–Yates (Knuth) shuffle. The cost of the shuffle operation is $O(n)$. The formation of $X$ can be implemented as a single-ass double-cursor iteration visiting each node no more than once, therefore the total cost of this matching is $O(n)$.

### 4.11.6 Call Auction

The standard model provides an alternative contract matching algorithm that uses the rationing §4.11.4 step differently. This matching algorithm is normally referred to as the 'call auction'.

For a set of `Employer` nodes $E$ with bid orders $B$ and a set of `Employee` nodes $W$ with ask orders $A$ the call auction processes the labour market as follows:

1. Create a function $v^- : \mathbb{R} \mapsto \mathbb{R}$ as follows:

$$v^-(w) = \sum_{e^* \in W : w(e^*) \leq w} s(e^*)$$

($v^-$ is the sum of the supply volumes of all ask orders whose prices are not greater than $w$, namely the total amount of labour in the system that can be employed at price $w$ without undercutting the ask price).

[35] As observed in §4.11.3 it is often of greater importance that the buyer (the `Employer`) pays no more than the bid price than that the seller (the `Employee`) receives no less than the ask price. This is because the buyer is very often required to set aside (allocate) cash resources at the time the order is submitted to the market. If the execution price $w$ selected by the market is greater than the bid price $\omega(e)$ then the transaction may fail as a result of insufficient allocated buyer cash. Note that in general it is not possible to satisfy the matching conditions unless the bid price is exceeded or the ask price is not met for some `Employer`–`Employee` pairs.

2. Create a function $v^+ : \mathbb{R} \mapsto \mathbb{R}$ as follows:

$$v^+(w) = \sum_{e \in E : w(e) \geq w} d(e)$$

($v^+$ is the sum of the demand volumes of all bid orders whose prices are not less than $w$, namely the total labour in the system that can be employed at price $w$ without exceeding any bid price).

3. Identify the least execution price $w \geq 0$ such that $v^-(w) \geq v^+(w)$:

$$w = \inf_x \{x : v^-(x) - v^+(x) \geq 0, x \geq 0\}.$$

4. Exclude all sellers (`Employees`) from the market whose ask prices are greater than $w$.

5. Exclude all buyers (`Employers`) from the market whose bid prices are less than $w$.

6. Compute $d = \sum_{e \in E} d(e)$ and $s = \sum_{e^* \in W} s(e^*)$ for all outstanding participants.

7. If $d \neq e$ then apply a rationing step (§4.11.4) to all remaining market orders (such that $d = e$ after the application of the rationing).

8. Apply a node matching step (§4.11.5) to the remaining market participants (generate a set $X$ of matchings).

Note that unlike other implementations of the labour market in the standard model, this market processing algorithm does not in general have the property that

$$T = \sum_{x \in X} \ell(x) = \min \left( \sum_{e \in E} d(e), \sum_{e^* \in W} s(e^*) \right),$$

where the right hand side is evaluated over the original orders $A$, $B$: the total trade described by the matching $X$ is in general not maximized with respect to the bid and ask volumes stated in the original orders. The total trade $T \geq 0$ executed by the call auction is expected to be significantly less than the maximum admissible demand and supply, which in turn means that many market participants are likely to be wholly excluded from the market by the final matching $X$.

The call auction does however guarantee that $w(x) \leq \omega(e(x))$ for all $x \in X$ (the execution price of all contracts is not greater then the `Employee` bid price for any contract) and that $w(x) \geq \omega(e^*(x))$ for all $x \in X$ (the execution price of all contracts is not less then the `Employer` ask price for any contract). The call auction algorithm seeks to maximize $T$ subject to these two constraints.

Note that as $|E| < \infty$ and $|W| < \infty$ there are a finite number of distinct bid/ask prices in the system. In general it is not possible to find a number $w$ that satisfies $v^-(w) = v^+(w)$ and the function $f = v^- - v^+$ is piecewise constant discontinuous. The maximum trade $T$ processed by the system is $v^+(w)$ where $w = \inf_x \{x : v^-(x) - v^+(x) \geq 0, x \geq 0\}$.

Note that the matching and/or rationing steps $(7-8)$ may involve random number generators and therefor in general this process will require random number seeds in order to behave deterministically.

As the number of market participants and the number of distinct bid/ask orders is finite, (3) can be performed in a single pass if the set $E$ of `Employers` is sorted ascending with respect to ask price and if the set $W$ of `Employees` is sorted descending with respect to bid price. If both groups are sorted then the cost of step (3) is $O(n)$ where $n = |E| + |W|$ is the total number of markets participants. The cost of the sorting operation is bounded by $O(n \ln n)$ flops. The total cost of the call auction is therefore $O(\min(n \ln n, c_R, c_M))$ where $c_R$ is the flop complexity of the rationing step (7) and $c_M$ is the flop complexity of the matching step (8).

Note that for a labour market with homogeneous bid/ask prices (a single price shared by all market participants) the call auction is equivalent to any of the implementations illustrated in §4.11.3 (with a single-pass $O(n)$ overhead associated with verifying that $E$ and $W$ are sorted).

### 4.11.7 Instruments

In the standard model the labour market consists of a set of disjoint instruments labelled by labour contract maturity (the length of labour contracts processed by each instrument). The labour market may be composed of several such instruments individually dedicated to contracts of length $1Q$, $2Q$, one year, many years and so on. When submitted to the labour market each order is routed to the appropriate instrument according to the desired maturity. Instruments in the labour market may in principle involve different matching processes.

It is not a strict requirement that labour markets should be implemented in such a way that each instrument is processed independently.

### 4.11.8 Order Cancellation

In general the market will be unable to satisfy all orders even if the number of units of labour in supply exceeds the total demand. Unsatisfied or partly satisfied orders not resolved by the market must be cancelled in a timely fashion.

In the standard model, labour market contracts are processed in batch once per simulation cycle at time `LABOUR_MARKET_MATCHING`. Any outstanding or unfulfilled orders are cancelled immediately after this processing session at time `POST_LABOUR_MARKET_MATCHING`. Order cancellation entails the removal of the labour market order from the participant order book (if applicable) and the disallocation of any allocated liquidity associated with bid orders.

## 4.12 Stock Exchange

Stock exchanges are places of activity for traders and brokers who buy and sell listed company stocks, bonds, derivatives and other types of securities. Almost

all major economies have at least one stock exchange: the British London Stock Exchange (LSE), the American NASDAQ (originally an acronym for the *National Association of Securities Dealers Automated Quotations*) and New York Exchanges (NYSE), the Chinese Shanghai Stock (SSE) and Shenzhen Exchanges (SZSE) and the German Deutsche Börse AG in Frankfurt are prominent examples. As of January 2015 the largest stock exchange by market capitalization (accounting for foreign exchange rates) was the NYSE (with a total market cap of around $19.2 trillion) by nearly a factor of 3, with the next most capitalized exchange being the American NASDAQ (≈ $6.8 trillion). The most capitalized non-American exchange is the British LSE, followed by the Tokyo (JPX) Exchange (≈ ¥529 trillion Japanese Yen) and Shanghai (≈ $24.5 trillion Renminbi). Measured according to the value of all trades on a monthly basis, the NYSE (≈ $1.5 trillion) is also the most active exchange, with Shanghai (≈ $1.3 trillion) being comparably active.[36]

The rate and amount of trading that takes place at these exchanges is high. Business on modern stock exchanges is increasingly electronic, and many major exchanges provide electronic interfaces for submitting orders directly to the appropriate trading platform. These electronic trading services use order matching algorithms to compute which seller/buyer pairs ultimately interact with one another. Since at least the turn of the millennium, some participants have utilized computer algorithms to optimize their transactions and decide their order prices, timings and sizes. Hardware and software can naturally perform these tasks many orders of magnitude faster than has been possible previously, and some participants have developed algorithmic trading policies that use the resulting very short period holding times to derive profits. This High Frequency Trading (HFT) is highly significant in some modern markets: it has been estimated as of 2010 that around 60% of US equities trades are HFT[37].

Share price formation is ultimately as a result of differences between aggregate stock supply and demand. The mechanics of the underlying stock market processing algorithm differs among states. Broadly speaking if the combined value of stock sell orders is greater than the combined value of stock buy orders among all market participants then the price per share is expected to fall and vice versa. Modern stock exchanges tend to provide Limit Order Book (LOB) systems to facilitate trades; structures that require sellers to specify the minimum price at which they are willing to sell (and likewise the maximum price at which buyers are willing to buy) and that process trades according to these restrictions. Price emerges from the distribution of these orders, as well as their sizes and timings.

A simplified description of an LOB system is as follows: participants $x$ submit orders, $o$, which amount to triplets $o = (p, \omega, t)$ where $p$ is a price per share, $\omega$ is a quantity of shares and $t$ is a timestamp indicating the order submission time. If $x$ is submitting a buy order then $p$ is the maximum price at which $x$ is willing to buy shares and $-\omega$ is the number of shares desired by $x$. If $x$ is submitting a sell order then $p$ is the minimum price at which $x$ is willing to sell

[36] Domestic Market Capitalization, *World Federation of Exchanges Monthly Reports*, January 2015, `world-exchanges.org/statistics/monthly-reports`.

[37] *High Frequency Trading*, Bruno Biais and Paul Woolley, March 2011.

shares and $+\omega$ is the number of shares offered for sale. The LOB is processed continuously by a trade–matching algorithm whose responsibility is to match pairs of buy and sell orders in such a way that the constraints of both parties' orders are satisfied. When a new order $o$ is submitted, the trade matching algorithm attempts to match $o$ to an outstanding order. If such a match is possible then $o$ is processed immediately, otherwise $o$ is labelled 'active' and it will continue to be monitored for potential matches as new orders are incoming. $o$ can be cancelled either by the submitter or as a result of the rules of the exchange (eg. due to an order timeout policy). Incoming orders which can be satisfied immediately but that are subject to a peer order ambiguity (a buy order that can be fulfilled by multiple sellers and contrariwise) are resolved according to a priority system. Typically this policy is the so–called 'price–time' policy system such that sell orders are matched to buy orders with the maximum available price, with the earliest buy orders gaining priority in case of ties (and similarly for the opposite case), but alternative policies exist.[38]

The standard model currently does not provide a time resolution that would be suitable for modelling HFT activities on the stock market. It is thought, at the present time, that this modelling feature would not be useful. In the standard model stock trades are typically processed once per simulation cycle and these trades represent batch flows of shares due to transactions on smaller and unmodelled timescales. Exceptions to this rule include stock issuing activities and initial public offerings, stock erasure (the deletion and unlisting of a stock instrument from an exchange), over–the–counter stock trades, repo loans that involve stock accounts as collateral, business interactions that amount to compensation by stock assets, liquidation, and certain aspects of the relationship between funds and their clients. The implementation of the stock transaction system is based either on a clearing market process (namely a market in which participants may specify their order sizes and intentions as a function of price) or LOB–like structures for which market orders are processed in batch once per simulation cycle.

### 4.12.1 Background and Legacy Model

In the legacy Mark One model §3 the stock market was implemented in such a way that (typically) all shares were erased and then reissued after one simulation cycle had elapsed. Put differently, a stock holder $x$ that owned $n(x) > 0$ shares before stock market processing and who wanted to retain $n(x) - 1$ shares was required to bid for an additional $n-1$ new shares so that, after stock market processing, the number of shares owned by $x$ would decrease by one[39]. As a result of this procedure it was either not possible or difficult for agents to selectively sell shares and decline to participate in stock markets to which they were subscribed.

In the legacy model shares were implemented as encapsulated integers. Stock market participants were restricted to own an integer number $n = 0, 1, 2 \ldots$ of

[38] *Limit order books*, Guold, Porter, Williams, McDonald, Fenn, Howison. Quantitative Finance 2013, Vol. 13 No. 11.

[39] the market interface in fact accepted orders for a value of shares, not orders for a number of shares to own following market processing; see discussion at the end of this section.

shares in any particular stock instrument and accordingly the range of possible values for stock account assets was discrete. Although this modelling assumption is physically truthful it is nonetheless problematic when implemented in scientific code, as it entails the use of integer solvers and discrete optimization, operations that are often more involved and prone to programming errors than their floating point counterparts (in addition as of May 2015 the market capitalization of Apple Inc. was $\approx$ \$740bn at a price per share of $\approx$ \$130, so the total number of Apple shares in circulation was of the order 5.7 billion, which is larger than the maximum value of an unsigned integer on many 32-bit computer systems). The legacy stock instrument processing algorithm was as follows:

1. If the set of market orders $O$ is nonempty, compute the total value $V' = \sum_{o \in O} v'(o)$ where $v'(o)$ is the value of stocks that the participant $x$ who submitted $o$ intends to buy.
2. Set $N = 0$.
3. For each stock market participant $x$ known to the stock instrument, ask the participant how many stocks $n(x)$ they currently own. Increment $N \mapsto N + n(x)$.
4. Set $p' = V'/N$ (the revised price per share) and set $N' = 0$.
5. For each stock market participant $x$ known to the stock instrument, set $n'(x) = \lfloor V'/p' \rfloor$ where $\lfloor\rfloor$ denotes the integer floor function. Increment $N' \mapsto N' + n'(x)$.
6. Compute $\Delta N = N - N' \geq 0$.
7. Randomly assign $\Delta N$ shares to a subset of the participants (and adjust $n'$ accordingly).

When the participant stock account assets had been updated according to this recipe, the value of all participant cash reserves was modified to account for the (signed) change in the capital value of stock assets, namely:

1. For each stock market participant $x$:
   (a) record the value $v(x)$ of the participant stock account before stock market processing (price adjustment).
   (b) Compute the updated value $v'(x)$ of the participant stock account, where $v'(x) = p'n'(x)$ according to the above procedure.
   (c) Record a planned participant cash balance perturbation, $\Delta v(x)$, with value $\Delta v(x) = v(x) - v'(x)$.
2. Following the processing of all other instruments, for each stock market participant $x$:
   (a) If $\Delta v(x) < 0$ then withdraw $-\Delta v(x)$ units of cash from $x$. If this withdrawal fails, $x$ is instructed to appeal for a cash flow injection and the withdrawal is attempted again. If on this second occasion the withdrawal fails, then flag $x$ for liquidation and unsubscribe $x$ from the market.

(b) Otherwise, if $\Delta v(x) \geq 0$, then debit $x$ with $\Delta(x)$ units of cash.

The cash flow associated with stock account gains and losses was therefore modelled abstractly.

In time this approach presented numerical and modelling challenges. If all participants were to decline to participate then stock market processing would yield non-numerical results, and the processing of the legacy stock market did not necessarily conserve cash (money could be created or destroyed as a result of changes in the stock price, a figure that was governed by aggregate demand for the stock).

Note that throughout the above process the market participants $x$ cannot control the number of shares they will own after stock market processing. The stock market interface (order format) is such that the participants must bid for a value of shares they intend to own, not a quantity of the same. If the price per share is $p$ immediately before market processing, an order for stocks of value $pn'$ where $n'$ is the intended number of shares to own will not in general result in $n'$ shares, because the new price per share $p'$ selected by the market depends on the behaviour of all other participants in addition to $x$ (in particular $x$ would receive $\sim pn'/p'$ shares plus a small random endowment where $p'$ depends on the combined intentions of all participants).

Note that no central object tallies the number of shares in existence (on a per–instrument basis) in this setup, so that the number of shares must be explicitly recounted by querying stock holder participant balance sheets once per simulation cycle at price formation.

The current and revised implementation of the standard model stock market has some spiritual features in common with the above legacy processing scheme. Shares are ultimately encapsulated floating point numbers and are therefor amenable to 'continuous' optimization and arithmetic; cash cannot be created as a result of price per share movements in the stock markets; the mechanism though when stock market participants (peers) interact with one another is modularized; an explicit stock exchange object exists for the purpose of tallying and keeping track of issued shares; stock market transactions do not necessarily result in liquidations in the event that planned transactions cannot be financed; no shares are dolled randomly, and market participants may easily either decline to participate in the market or offer only part of their stock account assets for sale. Shares are conserved quantities (except by explicit instruction to the stock exchange) and are not erased at the end of the simulation cycle, meaning that agents can in principle retain shares for as long as the shares remain valid.

### 4.12.2 Exchange Services and Utilities

The stock exchange is an interfaced object, and as such it is possible for the user to provide a customized implementation. The standard model provides an implementation with (a) high level share transaction processing facilities and (b) central logging and stock instrument snapshots. The stock exchange in the

standard model is statically (ie. globally) accessible.

The types of agent in the remit of the exchange are:

a) `StockHolders` (agents that can administer `StockAccount` balance sheet assets, and thus retain shares), and

b) `StockReleasers` (agents that can emit shares, and optionally pay dividends). Note that in the standard model the primary implementations of `StockReleaser` are commercial banks, shadow banks and firms/establishments.

The stock exchange in the standard model provides the following services:

1. `addStock(StockReleaser, inititalPricePerShare)`
   `removeStock(StockReleaser)`

   These operations enlist/delist `StockReleasers` from the exchange (create and subsequently remove stock instruments, begin and end stock price and share ownership tracking services).

2. `double getNumberOfEmittedSharesIn(stockName)`
   `double getNumberOfEmittedSharesIn(StockReleaser)`

   Query the total number of emitted shares (if any) in a given `StockReleaser`, either with reference to the stock releaser itself or with reference to the identifying name of its stock. If the specified agent or its stock symbol are unknown to the exchange, `IllegalArgumentException` is raised.

3. `disownShares(StockHolder, stockName, numberOfSharesToDisown)`

   Share disownership services. If a `StockHolder` owns a nonzero number of shares in the stock instrument with the specified name, and if there is need or wish for some or all of these shares to be disowned by the `StockHolder` (removed from the `StockHolder` balance sheet without transfer to a counterparty), the exchange is able to accept these. Any actions taken by the exchange following the disownership of shares are documented by the implementation. In the standard model implementation, the exchange will distribute disowned shares to other registered `StockHolders` without bias (if there are any such `StockHolders`). If the disowning party has given away all of the shares it owns, then the `StockAccount` asset of the disowning party is terminated. Voluntary and involuntary stock disownership may occur in several possible behavioural edge cases, including the closure of a financial intermediary who has sold all but a small number of shares on behalf of a non-`StockHolder` beneficiary. Other possible implementation policies include pro-rata share redistribution and explicit deletion of shares (an action that will affect the stock price).

4. `StockReleaser getStockReleaser(String uniqueStockName)`
   `List<StockReleaser> getStockReleasers()`

   List all agents, or retrieve a single `StockReleaser` agent, whose stocks

are traded on the exchange. The return value of the latter method (catalogue retrieval) is a defensive copy.

5. `double getStockPrice(StockReleaser stockReleaser)`
   `double getStockPrice(stockName)`

   Return the current (most recently updated) price per share of a stock instrument, with reference either to the `StockReleaser` that emitted the shares or with reference to the identifying name of its stock.

6. `setStockPrice(stockName, positivePricePerShare)`

   Modify the price per share of stocks in the specified `StockReleaser`. Note that this action will adjust the balance sheet equities of all market participants who currently hold shares of this type, and this action may in turn cause one or more market participants to suffer negative equities.

   The revised price per share must be positive. Otherwise, no action is taken and `IllegalArgumentException` is raised.

7. `eraseAndReplaceStockWithoutCompensation(`
   `stockName, newStockHolders, newPricePerShare)`

   Erase and then reissue a stock instrument without compensating existing `StockHolders`. This method is an entry point for an operation that purges all shares in a `StockReleaser` without offering any cash or any alternative assets to market participants who owned the shares. All shares in the specified `StockReleaser` will be immediately removed form the balance sheets of all such `StockHolders` and subsequently all `StockAccount` assets will be terminated. Existing `StockHolder` participants may suffer negative equities due to the resulting capital loss. The stock instrument is then reissued to a collection of new `StockHolders`, as specified by the second argument, at a new price per share, as specified by the third argument, which must be positive. If the specified price is negative or if no new `StockHolders` are specified, `IllegalArgumentException` is raised and no action is taken.

   This method does not require any of the replacement `StockHolders` to pay for shares (`newStockHolders` enter into no cash transactions during the processing of this method).

   Stock erasure may happen as a result of a firm or a bank suffering significant insolvency or debt default leading to its shares being transferred exclusively to its creditors as compensation (at the expense of non–creditor stockholders). This operation may also model part of the process of firm liquidation followed by the spawning of a new firm in its place with a new public offering and new stakeholders.

8. `generateSharesFor(`
   `stockName,`

```
StockHolder,
numberOfShares,
priceToPayPerShare
) throws InsufficientFundsException
```

Generate new shares for a `StockHolder`. The `StockHolder` is required to pay for the newly generated shares. If the `StockHolder` cannot pay for the shares at the specified price, no action is taken, the stock instrument is unchanged, and `InsufficientFundsException` is raised. The `StockHolder` need not have an existing `StockAccount` in the specified instrument. The number of shares to issue and the price per share to pay must be positive. If the number of shares to issue is zero no action is taken and this method does nothing. If the price per share is zero then the underlying transaction cannot fail and no flow of funds occurs. If the price per share is negative, `IllegalArgumentException` is raised, this method does nothing, and the state of the stock instrument is unchanged. In any eventuality the current price of the stock is unchanged: the `StockHolder` need not pay the price indicated by `getStockPrice(stockName)`.

9. `generateFreeSharesFor(stockName, StockHolder, numberOfShares)`

   Generate free shares of the specified type for a `StockHolder`. If the type of shares specified does not correspond to a stock instrument known to this exchange, `IllegalArgumentException` is raised. The number of shares must be positive. If the number of shares specified is negative this method does nothing. No flow of funds occurs as this method is processed.

In the standard model the exchange provides a set of per–instrument services called 'stock ownership trackers'. These tracker services centrally and automatically record the distribution of owned shares and any change that might occur to existing or new `StockHolder` positions. References to all `StockHolder` agents that currently own shares in a stock instrument are exposed by the stock ownership tracker.

Stock ownership trackers additionally provide the following utilities:

1. Iteration over and/or enumeration of all `StockHolders` who currently hold shares;
2. Explicit recounts of all shares owned by all known `StockHolders`, and
3. Computation of the (instantaneous) value of all shares owned by a particular `StockHolder` (the instantaneous stock price per share multiplied by the number of shares held by the participant).

In the standard model the exchange additionally provides a data logging (market summary snapshot) service whose purpose is to record the following observations on a per–instrument basis once per simulation cycle immediately after all clearing markets have been processed:

1. The price per share of the stock;
2. The number of shares in existence, and

3. The most recent total value of dividends paid by the `StockReleaser` whose stock is observed.

These market snapshots are stored in a circular buffer with customizable length.

### 4.12.3 Access and Registration

In the standard model the stock exchange is implemented as an eager singleton. The exchange is globally accessible via `UniqueStockExchange.Instance`. this setup is intended to streamline the application, as for a large part of the FP7 initiative there has been no functional need for two distinct stock exchanges. This picture may or may not change in future with the introduction of multi–country models. This arrangement also facilitates automatic bookkeeping for `StockAccount` (§4.12.4) objects (namely keeping track of who owns which shares at any particular time).

Stock market registration (listing) is provided by `StockExchange.addStock` (§4.12.2). Whether or not (a) `Firm` agents and/or (b) `Bank` agents are to be listed on the stock exchange is a yes/no configuration option and is a model parameter in the meaning of §4.3.

### 4.12.4 Stock Accounts

`StockAccounts` are balance sheet assets that represent the value of shares owned by a stock market participant. Agents who administer and hold `StockAccount` balance sheet assets are by definition `StockHolders`. `StockAccount` objects are linked to (and interact with) the stock exchange. These accounts provide (a) some automatic bookkeeping services, (b) stock price tracking services, and (c) mark–to–market valuation. Note that `StockAccount` valuation is performed on a mark–to–market basis and is not the book value, by default, meaning that stock market participants can in principle receive negative equities and become insolvent as a result of movements in the stock price. Some type of agent, for instance `Household` agents, cannot retain or trade shares on their own and require the services of a second party who can administer accounts on their behalf.

Each `StockAccount` corresponds to an instrument at an exchange: a `StockHolder` balance sheet may contain as many `StockAccounts` as there are distinct instruments traded on exchanges to which that party has access.

`StockAccount` assets in the standard model provide the following services:

1. `StockAccount.create(StockHolder, StockReleaser, numberOfShares)`

   A static `StockAccount` creation utility. This service creates a new `StockAccount` and installs this asset on the balance sheet of the `StockHolder`. The newly created account is registered with the `StockReleaser` so that dividends may be paid, if and when these become due. The number of shares with which the account is initialized must be strictly positive, otherwise no action is taken and this service does nothing. If the balance sheet of the

`StockHolder` already contains a `StockAccount` in the given instrument, then method fails and no parties are modified.[40].

2. `getReleaser()`

   Provide a reference to the `StockReleaser` whose shares are contained in this account. Note that this agent may be eg. a firm, a bank, or another type of stock–emitting agent.

3. `getHolder()`

   Provide a reference to the `StockHolder` for whom this account is a balance sheet asset.

4. `getInstrumentName()`

   Provide the identifying name of the stock instrument to which shares in this account belong, for instance "Apple Inc.".

5. `getQuantity()`

   State the total number of shares (a floating point number) contained in this stock account.

6. `setQuantity(numberOfShares)`

   Modify the number of shares in the account. The argument to this method, the revised number of shares to exist in the account, must be non-negative. Otherwise, the argument is silently trimmed to zero. If the argument is zero, then the stock account is emptied and terminated.

7. `disownShares(numberOfShares)`
   `disownShares()`

   Cause the `StockHolder` whose account this is to disown some or all of their shares. This service exists in case a `StockHolder` is required to dispose of some or all of its balance sheet assets, but does not have a suitable counterparty to which existing shares could be transferred (for instance if a financial intermediary whose stock trading services on behalf of a beneficiary come to a close before the intermediary has sold all of its shares, or during a liquidation process for some types of agent).

   The argument `numberOfShares` must be non–negative and not greater than the total number of shares in the account, otherwise the argument is silently clamped in the range $[0, \texttt{getQuantity()}]$.

   This method delegates to the stock exchange service `StockExchange.disownShares(StockHolder, stockName, numberOfShares)` (§4.12.2). The disownership policy of the stock exchange is documented by the implementation.

[40] In general a `StockHolder` may not simultaneously own two distinct `StockAccount` assets in the same instrument.

8. `getPricePerShare()`

   Provide the (instantaneous) price per share of the stock to which this account corresponds, as quoted by the stock exchange managing the instrument.

9. `terminateAccountWithoutCompensatingStockHolder()`

   Terminate this account and immediately remove it from the `StockHolder` balance sheet. This method unregisters the account from the `StockReleaser` so that future dividends, if any, will no longer be paid to the holder. The account is removed from the balance sheet of the `StockHolder` and the exchange is instructed to discontinue share ownership tracking for this party for this instrument.

10. ```
    sellSharesTo(StockHolder) throws InsufficientFundsException
    sellSharesTo(StockHolder, numberOfShares)
      throws InsufficientFundsException
    sellSharesTo(StockHolder, numberOfShares, pricePerShare)
      throws InsufficientFundsException
    ```

    Perform a cash–for–shares over the counter transaction. If the number of shares to sell is not specified, it is assumed that all shares contained in the stock account are to be sold. If the sale price per share is not specified, it is understood that the instantaneous mark–to–market value is the sale price. If the number of shares to sell is negative or zero then no action is taken and this method does nothing. Negative sale prices are silently trimmed to zero, and the number of shares to sell is silently capped at `getQuantity()`. If the cash transaction fails (the seller cannot pay for the transaction at the specified price) then no action is taken and the balance sheet of both the holder and the counterparty is unchanged. If the cash transaction succeeds and the counterparty does not possess a `StockAccount` in the appropriate instrument then an account is automatically created. If the seller stock account contains zero shares following the sale, the seller account is terminated and removed from its balance sheet. If the sale price specified is zero then no flow of funds occurs.

11. ```
    transferSharesTo(StockHolder)
    transferSharesTo(StockHolder, numberOfShares)
    ```

    Transfer shares from one `StockHolder` to another. This service is equivalent to selling a number shares at price zero.

12. ```
    makeDividendPayment(sumToPay)
      throws InsufficientFundsException
    ```

    Transfer funds of value `sumToPay` from `StockReleaser` to `StockHolder`. This service executes a dividend settlement transaction between the stock issuer and the owner of the `StockAccount`. If the sum to pay is negative or zero, this method does nothing and no action is taken.
    ```

Note that this flow of funds may fail if the issuer is insufficiently solvent. In this eventuality `InsufficientFundsException` is raised and the state of both parties is unchanged.

13. `getValue()`

    Compute the instantaneous mark–to–market value $v$ of the `StockAccount`, where $v = pn$, $p$ is the instantaneous price per share for the corresponding instrument (as quoted by the exchange) and $n$ is the number of shares in the account.

A small number of nonfunctional methods inherited from `Contract` persist in the `StockAccount` class owing to the structure of the legacy codebase. These methods, which include `setValue()`, `setInterestRate()` and `setFaceValue()`, have no meaning because (a) the interest payable on `StockAccounts` is approximately equal to the dividends per share per unit stock price, a quantity that is unknown in advance except to the `StockReleaser` (who decides the dividend payable on the stock) and (b) the value of a mark–to–market `StockAccount` cannot be modified without first modifying the price per share, a figure that is decided by the stock exchange and over which the account holder has no power. These methods do nothing, raise `UnsupportedOperationException` and are subject to change in future.

### 4.12.5 Stock Market Processing

By default the stock market is subdivided into independent, synchronously processed stock instruments. There is one such instrument per `StockReleaser` whose shares are listed at the exchange. Stock instruments provide their own processing policies and logic (and may be heterogeneous in this respect).

Notionally the processing of each instrument is subdivided into the following steps:

1. Price formation (adjustments to the existing instrument price per share), and
2. Peer–to–peer stock trade processing (the graph of stock buyer/seller pairs and the number of shares traded between each such pair).

When stock instruments are processed as clearing instruments (namely, all participants are given the opportunity to state their stock buying/selling intentions as a function of the stock price, rather than posting market orders with fixed sizes in advance) then all `StockHolders` registered with the exchange are asked to provide univariate functions describing their buying/selling intentions in terms of price per share. These statements of intent are used to select the revised price per share and then execute peer–to–peer transactions according to the following recipe:

1. If no shares are in circulation, no further action is taken.
2. The clearing instrument records its existing price per share, $p$.

3. For all `StockHolders` $H$ registered with the stock exchange on which the instrument is listed, $h \in H$ is asked whether it intends to participate in the next processing session. If any $h \in H$ declines to participate, then it is understood that $h$ neither wants to buy or sell shares no matter how the stock price may change. Write $H^*$ for the collection $\{h \in H : h$ intends to participate in the next instrument processing session$\}$. Note that $H^*$ may consist of many heterogeneous agent types: funds, banks, firms, the state central bank, government and/or bad banks, among others.

4. If at this stage $|H^*| \leq 1$, no further action is taken by the instrument in this session and the stock price is not adjusted.

5. Otherwise, for all participants $h \in H^*$, insert $h$ into the left node group of a pure homogeneous network market and query its share buying/selling intentions $\nu_h$. $v_h(x)$ is a univariate function of the revised stock price per share, $x$ ($x \geq 0$), yet to be chosen by the market:

   (a) If $\nu_h(x) < 0$ it is understood that, at price $x$, $h$ intends to sell shares of value $-\nu(x)$.

   (b) If $\nu_h(x) > 0$ it is understood that, at price $x$, $h$ intends to buy shares of value $+\nu(x)$.

   (c) If $\nu_h(x) = 0$ then $h$ does not wish to buy or sell shares any at price $x$. Note that the trivial function $v_h \equiv 0$ is equivalent to $h$ declining to participate in the next instrument processing session.

   For each $h \in H^*$ it is understood, on the grounds of physical realism, that $\nu_h(x) \longrightarrow -\infty$ as $x \longrightarrow \infty$ (namely, if the stock price abruptly becomes infinite, all participant intend to realize their unbounded capital gains).

   Note that $\nu_h$ represents a desired change in value of the stock portfolio of participant $h$. The (signed) number of shares that $h$ wishes to aquire at price $x$ is $\nu_h(x)/x$.

6. Add the stock exchange itself to the right node group of the same pure homogeneous network. The stock exchange posts the order function $\mu(x) \equiv 0$.

7. Identify a revised price per share $p' \geq 0$ such the aggregate value of shares sold by all market participants is equal to the aggregate value of shares bought by the same (cash flow consistency):

$$\sum_{h \in H^*} \nu_h(p') = \mu(p') = 0 \qquad (\dagger)$$

8. Aggregate the resulting orders, $O$, selected by the network at price $p'$.

9. Modify the stock price $p \mapsto p'$ (note that this action changes the asset value of all existing stock accounts by a factor of $p'/p$).

10. Submit the order set $O$ to a resource distribution algorithm, $X$, whose responsibility is to implement trades between buyers and sellers.

The share distribution algorithm $X$ is responsible for (a) executing trades between peer `StockHolders` and (b) processing the resulting flow of funds and share transactions. $X$ is an object and is a model parameter in the meaning of §4.3.

Note that the price $p'$ satisfying the cash flow consistency constraint † is not necessarily unique. In general it is understood that there exists a sufficiently large price $p^* \gg 1$ such that $\sum_{h \in H^*} \nu_h(p^*) \leq 0$ and that $\sum_{h \in H^*} \nu_h(0) \geq 0$. The value $p'$ is then identified by bisection.

It is expected, but technically not required, that $\sum_h \nu_h(x) \longrightarrow \epsilon$ for some $\epsilon \geq 0$ as $x \longrightarrow 0$, because of all participants $H^*$ it is assumed that at least one participant will invest an amount, no matter how small, in the stock if its price suddenly falls (on the basis that a large profit would be derived from subsequent rebounds in the stock price). If the market buy/sell intentions $\nu_h$ are continuous and monotonic[41] then $p'$ is expected to be unique, because the sum of continuous monotonic functions is also continuous and monotonic.

Note that the existing price per share $p$ will not be revised if either (a) all but one or fewer `StockHolders` registered with the exchange declined to participate, or (b) the number of shares in circulation is zero. If a single `StockHolder` elected to participate in the market, no transactions can take place.

For numerical reasons the exchange will reject the solution $p' = 0$. Note that there may be edge cases, particularly in low resolution simulations, for which the solution $p' = 0$ satisfies †. The price $p' = 0$ could result in singular arithmetic for some price processing algorithms elsewhere in the codebase and in client code. For this reason the exchange specifies a customizable small minimal price per share $p_{\min}$. The value $p'$ is clamped according to $p' \mapsto \max(p', p_{\min})$ as long as a the appropriate `StockReleaser` continues to do business.

In this model all `StockHolders` are grouped on the left hand side of a homogeneous market, whereas the stock exchange is the sole participant on the right hand side of the same. For this reason, $O$, the solution identified by the market processing algorithm, is a set of trades of the form $\{h, s, \nu(p'), p'\}$ where $h \in H^*$ is a participant, $s$ is the exchange, $\nu(p')$ is the intended (signed) share transaction value of $h$ and $p'$ is the transaction price. Thus the solution to the market processing problem results in stock price formation $p'$ but not the selection of any buyer/seller peers among $H^* \times H^*$. The share distribution algorithm $X$ is responsible for identifying how the resulting shares are transferred between parties in $H^*$. Note that the structure of the peer interaction graph selected by $X$ matters, because (among other issues) it may be the case that one or more of $h \in H^*$ proves to be insolvent at the instant the transaction is executed.

The distribution algorithm $X$ is required to select a peer–to–peer transaction graph $P$ with the following properties:

1. $P$ is a collection of transactions of the form $\{b, s, \nu, p'\}$ such that $b \in H^*$ (a buyer), $s \in H^*$ (a seller), $b \neq s$, $\nu > 0$ and $p'$ is the price per share

[41] In the standard model this is typically the case

selected by the above price formation recipe. For $\eta \in P$, write $b(\eta)$ for $b \in \eta$, $s(\eta)$ for $s \in \eta$ and so on.

2. For each $h \in H^*$ such that $\nu_h(p') \geq 0$,

$$\sum_{\eta \in P \,:\, b(\eta)=h} \nu(\eta) \leq \nu_h(p').$$

   Ie. for all participants $h$ that are buyers at price $p$, the sum of all volumes in $P$ for which $h$ is the buyer is not greater than the intended buy size $\nu_h(p')$ stated by $h$.

3. For each $h \in H^*$ such that $\nu_h(p') \leq 0$

$$\sum_{\eta \in P : s(\eta)=h} \nu(\eta) \leq -\nu_h(p').$$

   Ie. for all participants $h$ that are sellers at price $p$, the sum of all volumes in $P$ for which $h$ is the seller is not greater than the intended sell size $|\nu_h(p')|$ stated by $h$.

4. If $h \in H^*$ is a buyer at price $p'$, then there does not exist an element $\eta \in P$ such that $s(\eta) = h$ ($h$ is not realized as a seller with any counterparty in $P$).

5. If $h \in H^*$ is a seller at price $p'$, then there does not exist an element $\eta \in P$ such that $b(\eta) = h$ ($h$ is not realized as a buyer with any counterparty in $P$).

Other than that $X$ is required to identify a solution $P$ satisfying the above constraints, and that $X$ is not allowed to change the market intentions $\nu_h$, the structure of $P$ and the means used by $X$ to decide $P$ are not constrained and are specified by the implementation.

The standard model provides several stock implementations of $X$. These algorithms include:

1. **Central Payment Stock Redistribution**

   A biasless algorithm that calculates the overall effect of participant insolvencies before redistributing any shares. This transaction policy is fair to all market participants in the sense that the insolvency of any particular $h \in H^*$ affects all other participants equally. This redistribution scheme does not require a random number generator and is outlined in §4.12.5.1.

2. **Erase and Replace Stock Redistribution**

   An emulation of the Mark One legacy processing policy (§4.12.1). In this implementation the stock instrument is erased and then reissued to all participants according to their desired investments $\nu_h(p')$. As per the legacy processing scheme, `StockHolder` participants are not required to pay for their transactions. This scheme does not require a random number generator and is discussed in §4.12.5.2.

#### 4.12.5.1 Central Payment Stock Redistribution

By default, stock instruments are processed synchronously in an order that is defined by a clearing house. A given instrument, $I$, may be processed before or after other securities traded at its own exchange (for instance primary and/or secondary gilt markets) and the processing priority of $I$ relative to assets that are not traded at its exchange (eg. commercial loans) is unknown. For this reason, stock instrument processing policies (§4.12.5) cannot make the assumption that all market participants have ensured their allocated cash reserves are sufficient to cover all possible market outcomes (though depending on the decision rules employed by these participants, this may indeed be case).

Concretely: if a bank or a fund agent maintains a portfolio of many stocks and if this agent is caused to invest as a result of unexpected price changes among certain instruments, then this agent may need to borrow funds in order to realize the necessary liquidity: the strategy of the agent may even depend on the assumption that it is able to appeal for liquidity via repo loans. There can be no guarantee that these appeals will succeed, so stock instrumentss cannot assume that the transactions $\nu_h$ (§4.12.5) specified by their `StockHolder` participants are realizable.

The possibility that one or more market participants will prove to be insolvent when trades are executed means that some parties may be unable to sell their shares, which may in turn result in liquidity issues for these sellers. This outcome must be treated with care because, depending on the structure of the peer–to–peer transaction graph (who sells shares to whom) systematic biases in the market processing algorithm may result in one particular agent $X$ repeatedly or disproportionately affecting another party $Y$ for numerical reasons.

The so–called central payment share redistribution algorithm addresses this issue by homogeneously redistributing shares after price formation. The illiquidity of a stock buyer affects all stock sellers equally, so no systematic effects are possible. This approach is appropriate for (a) macroeconomic simulations in which one agent represents an aggregation of many smaller agents and (b) simulations that do not involve HFT or continuous asynchronous trading activities. Share transactions are processed as follows:

1. If the price formation solution $O(p')$ is empty, stop and do nothing.
2. Set $n_b = 0$ and $n_s = 0$.
3. For each order $\{b, s, \nu, p'\} \in O$, do the following:
   (a) If $b = e$ where $e$ is the stock exchange, swap the values of $b$ and $s$ and invert the sign of $\nu$.
   (b) If $\nu > 0$, increment $n_b \longrightarrow n_b + \nu/p'$.
   (c) Otherwise, if $\nu < 0$, then increment $n_s \longrightarrow n_s + (-\nu)/p'$.
4. If $n_s = 0$ or $n_b = 0$, stop and do nothing.
5. If $n_b > n_s$ then set $r_b = n_s/n_b$. Otherwise, set $r_b = 1$.

6. Set $\mu = 0$ and $n_b = 0$.
7. For each order $\{b, s, \nu, p'\} \in O$ such that $\nu > 0$, do the following:
   (a) Ask $b$ to pay the sum $r_b\nu$ (to a virtual counterparty);
   (b) If $b$ fails to pay this sum, take no further action and continue to the next order in $O$.
   (c) Otherwise, if $b$ succeeds in paying this sum, then:
   (d) Add $r_b\nu/p'$ shares to the `StockAccount` belonging to $b$, unless $b$ does not have a `StockAccount` in the required instrument, in which case create an account for $b$ containing this number of shares.
   (e) Increment $n_b \longrightarrow n_b + r_b\nu/p'$.
   (f) Post a `ShareTransactionEvent` detailing this acquisition to the simulation bus.
8. Set $r_s = n_b/n_s$.
9. For each element $\{b, s, \nu, p'\} \in O$ such that $\nu < 0$, do the following:
   (a) Provide $b$ with $-r_s\nu$ in funds (from a virtual counterparty).
   (b) Remove $-r_s\nu/p'$ shares from the `StockAccount` belonging to $b$.
   (c) If $b$ has no remaining shares, terminate its `StockAccount`.
   (d) Post a `ShareTransactionEvent` detailing this change to the simulation bus.

If the simulation is operating in debug or in pedantic mode, it will be asserted that the number of shares in circulation before and after processing is unchanged.

Note that in the above processing scheme, no buyer $b$ is able to tell from which seller $s$ their shares were sourced. The truest interpretation of the above transaction mechanism is that $b$ bought an equal number of shares from all available sellers.

A `ShareTransactionEvent` is a simple immutable event bus signal detailing (a) the `StockHolder` who bought or sold shares, (b) the (signed) cash value of the transaction and (c) the instrument to which the transaction corresponded.

#### 4.12.5.2 Erase and Replace Stock Redistribution

The erase–and–replace share redistribution algorithm is an emulation of the legacy model (§3) stock market processing policy. The primary purpose of this algorithm it to preserve backward–compatibility. This implementation is also distinguished insomuch as that the stock market is decoupled from all cash flows in the simulation.

Stock instrument processing is as follows:

1. If the price formation solution $O(p')$ is empty, stop and do nothing.

2. Create a table $P$ whose keys are the participants listed in $O$ and whose values are all zero. Write $P(p)$ for the value of participant $p$ in this table. Initially $P(p) = 0$ for all participants $p$.

3. For each order $\{b, s, \nu, p'\} \in O$, do the following:

   (a) Map $P(b) \longrightarrow P(b) + \nu$,

   (b) Map $P(s) \longrightarrow P(s) - \nu$.

4. For each participant $p$, compute the value $V(p)$ of shares already owned by $p$ at the revised price per share. Map $P(p) \longrightarrow P(p) + V(p)$.

5. Compute the number $T = \sum_p P(p)$.

6. For each participant $p$, map $P(p) \longrightarrow P(p)/T$.

7. Erase the stock instrument (namely delete, without compensation, all existing shares in all participant `StockAccounts`).

8. Reissue new shares to the participants $p$ in the ratios specified by $P$. The participants $p$ are not required to pay for these assets.

Note that this process is singular if $T = 0$ ie. if there is no instantaneous demand to buy stock. For this reason it is important that stock–trader portfolios assign nonzero weights to the stock instrument no matter what its estimated rate of return may be.

Note that this share transaction processing policy is not cash flow consistent. No flow of funds takes place at any stage during the execution of this policy. Participants are not required to pay for any stock assets that they may receive.

### 4.12.6 Financial Intermediation

#### 4.12.6.1 Background and Legacy Model

In the legacy model (§3) households were the direct, exclusive owners of banks. All household agents were implicit stock traders, and at time $t = 0$ the initial stock emission of each bank would be distributed homogeneously among them. The relationship between firms/establishments and banks was not dissimilar: firms agents, when created, emitted their shares exclusively to banks. The resulting distribution of bank shares among households was static in time whereas the distribution of firm shares among banks was dynamic, meaning that the flow of funds corresponding to dividend payments was from firms to banks in unpredictable proportions and then from banks to households in fixed proportions (ie. dividend incomes were nearly identical among households). Banks were unable to buy stock in other banks even if the simulation policy would otherwise have allowed this.

The household–bank relationship was therefor modelled abstractly. This formula was justified somewhat by the macroeconomic interpretation that households ultimately own all economic resources. In reality it is unusual in the UK for households to process dividends independently and for households to directly

own shares in large firms that they do not direct. Shares may be managed by an investment portfolio provider who has the ability to process dividend streams and provide market analysis for fees, and/or retirement plan providers. More uncommon still is that a household could be said to directly participate on a stock exchange.

In due course it was decided that this ersatz approach should be replaced by a more truthful representation in which most households secure investment services with second parties (typically funds) who hold and trade large numbers of shares. However, this representation is still programmatically incomplete because the normal operating behaviours of many agents do not include strategic stock investment decisions and/or the liquidation of stock appreciation gains. Concretely, a party that receives shares as compensation in lieu of cash for counterparty default may not be a `StockHolder`, so the compensated party may not implement the necessary minimal strategic stock trading behaviours.

In this case it is necessary to identify a financial intermediary whose purpose is to manage the stocks that were received by its non-`StockHolder` beneficiary. The role of the financial intermediary is to execute a stock trading strategy on behalf of its beneficiary in order to liquidate shares. The intermediary is not allowed to create beneficiary debt (ie. the intermediary cannot bill its beneficiary for its services and/or losses) and any funds received by the intermediary should ultimately be transferred, either in whole or with deductions, to a bank account nominated by its beneficiary. When the portfolio of the intermediary is empty or nearly empty as a result of it having sold all of the shares that it managed, the role of the intermediary expires and its relationship with the beneficiary is discontinued. This action may or may not terminate the intermediary itself.

Note that it is improper to attempt to identify an agent that has a preexisting relationship with the beneficiary (for instance a pension fund) and then supplement the stock portfolio of that agent so as to create a virtual financial intermediary. This is likely to entail guesswork about the strategic behaviour of the agent that was hijacked.

In the standard model, financial intermediaries are typically single–beneficiary intermediaries (SBIs) whose services are dedicated exclusively to one beneficiary at a time. These agents are spawned on an on–demand basis and any requisite market subscriptions are processed automatically. Intermediaries persist until their services are of no further financial value, at which stage their market value is numerically zero, they are destroyed and removed form the simulation world. The canonical use case for financial intermediation in the standard model is `StockHolder` intermediaries. Firesale, commercial loan intermediaries and other agents have been employed at various times.

#### 4.12.6.2 Intermediary

In the standard model a financial intermediary is a `StockHolder Agent` that has the ability to participate in (a) fixed order book markets of some kind and/or (b) clearing markets of some kind. The intermediary will, when it is asked, provide a list of beneficiaries on whose behalf it is currently operating. The

intermediary agent can be implemented as a view of an existing agent, as an establishment that has a business relationship with existing private/public sector agents, or as a standalone entity.

Few assumptions are made about the characteristics of the beneficiary. The beneficiary is required to be a `SettlementParty` insomuch as that it is required to provide an entry point into which funds can be paid to it. It is not permitted for the intermediary to withdraw or attempt to withdraw funds from its beneficiary at any time. Note that this restriction does not preclude the possibility that the market activities of the intermediary are financed by public funds (eg. government) or by a bank or a firm, and neither does this preclude the possibility that the intermediary uses some liquidity or profits gained from selling beneficiary assets in order to support its own activities.

#### 4.12.6.3 Single–Beneficiary Intermediary

In the standard model the stock–trading intermediary has a purposefully minimal implementation.

The intermediary is defined by the following behaviours:

1. The intermediary acts on behalf of one exclusive beneficiary at a time.
2. When created, the intermediary is automatically registered as a stock–trading participant on the local exchange.
3. Once per simulation cycle the intermediary reviews its market participation status. If the aggregate value of intermediary stock assets is not greater than a user–specified threshold $T$ ($T \geq 0$) then the intermediary will disown any and all remaining unsold shares and will discontinue its relationship with its beneficiary and unsubscribe itself from the local exchange. This action results in the agent being disconnected from the simulation world and eligible for garbage collection.

In addition, the intermediary:

1. `credit` function (outgoing flow of funds) raises `InsufficientFundsException` for any nonzero flow.
2. `debit` function (incoming flow of funds) immediately delegates to a bank account nominated by its beneficiary.
3. cannot appeal for liquidity. No cash flow injection service is available to the intermediary.
4. stock market strategy is identical to the default central bank (§4.13.3) and government (§4.7.2.6) stock resale strategies.
5. does not require a workforce (is not an employer).
6. is subject to zero operating expenses.
7. does not have a preexisting relationship with any other agent in the simulation world (in particular, it is not a subsidiary or a public body),

8. does not draw on public funds.

9. retains no profits.

The user is required to specify a factory[42] service for the creation of intermediary agents. This factory may or may not produce a standalone agent for each invocation; it may instead be a façade returning a reference to an eagerly/lazily initialized shared intermediary.

Note that the stock market is processed as a pure homogeneous market (ie. at any given time there exists a global price per share) by default. The scaling behaviour of this market processing algorithm is linear in the number of active stock market participants, which is appropriate for an on–demand intermediary spawning policy, even if the total number of intermediaries spawned is of the same order as the population of the financial sector.

## 4.13 Central Bank

The central bank agent is a unique, optional actor in the standard model. For scientific and technical reasons the use of a central bank agent without government is disallowed, whereas the use of a government agent without a central bank is permitted. If present the central bank services the financial sector by providing loans to commercial banks and shadow banks. The central bank is also known as the 'lender of last resort'.

Monetary policy operations in the standard model are provided by this central bank agent and (at present) the central bank represents all monetary authorities, with the exception of any spontaneously spawned financial intermediation agents. This agent regulates liquidity in the model via management of interest rates and acts as a lender of last resort in times of financial turmoil. The management of interest rates is often referred to as 'conventional' monetary policy operations, whereas interventions in times of financial crisis are referred to as 'unconventional' monetary policies. The objective of the central bank is to foster price stability and maximize employment over time (ie. economic stability and welfare). The central bank agent manages interest rates according to a set of formal policy rules, which may include Taylor rules. The central bank also provides certain unconventional policy operations during bank crisis resolutions.

The central bank is distinguished in the simulation world in that it is expected to implement policy frameworks that directly regulate the economy without necessarily perusing any substantial profit objectives: beyond breaking even, the central bank is not necessarily as concerned with aggregate interest accrued on its loans as it is with the long term stability of the economy in which it resides. In the UK the central bank (the Bank of England) influences the development of the economy via three main policy channels: the monetary policy

[42] An algorithm whose purpose is to create or return an instance of an intermediary agent acting on behalf of a beneficiary. By injecting a factory instead of allocating a certain type of intermediary on the heap whenever a new intermediary is required, the implementation of the intermediary (and its stock resale policies etc.) can be changed by supplying a different factory, rather than modifying existing source code that is tied to a particular implementation of the intermediary.

framework, which is aimed at maintaining price stability, the macroprudential policy framework which aims to maintain the stability of the financial system, and the microprudential policy, whose purpose is to increase the resilience of individual financial institutions[43].

In the standard model these policies and regulatory mechanisms are implemented as:

1. modules to which affected agents must subscribe (eg. agent decision rules whose states are constrained by the central bank), or
2. internal decisions made by the central bank agent that affect the current or future state of the economy, or
3. internal policies to be applied on a case by case basis whenever the central bank interacts with other agents.

Note that this is a non exhaustive list and that the user is free to devise other regulatory mechanisms. An example of a policy module affecting all commercial banks is an upper bound on bank balance sheet leverage (with the upper bound leverage being selected by the central bank agent according to some algorithm). A VaR Basel II bank target leverage decision rule could be interpreted as such a policy module. An example of an internal decision made by the central bank agent is the repo loan interest rate. An example of an internal policy applied by the central bank on a case by case basis is whether to issue uncollateralized (non repo) loans to a troubled financial institution or whether to allow the institution to fail and be made subject to bankruptcy resolution processes.

The importance of central bank regulation, intervention and macroprudential and microprudential policies in the simulation should not be underestimated. If financial agents are sufficiently aware of their profit margins then lowering central bank interest rates ought to incentivize commercial banks to take on more debt and increase their leverage. If leverage increases then the financial instability of the economy may also increase, as banks that operate at high leverage are intolerant to substantial losses (whether as a result of stock shocks, capital losses or defaults on loans). On the other hand low central bank interest rates may reduce the real debt burden of some agents in the economy, increase the value of long term company balance sheet assets and facilitate an overall increase in demand for goods and services.

Mandatory minimum leverage ratios are popular tools among financial policy committees in several jurisdictions. For instance in the UK the eight largest banks and building societies have been expected to meet or begin to meet a 3% leverage ratio via Tier 1 capital since January 2014. In the US a minimal regulatory leverage ratio of 4% has been in effect since at least 1 January 2014. In late 2013 the finance ministry of the Netherlands proposed a 4% leverage ratio for key banks, in Canada the minimal bank leverage ratio is around 5%, and in Switzerland the minimal ratio for systemically important banks has been in the

[43] *One Bank Research Agenda*, Discussion Paper, the Bank of England.

range $3.1\% - 4.56\%$ (depending on the institution) since the turn of 2013[44]. In the standard model all banks delegate their target leverage calculations to decision rules. In this context, regulatory minimal leverage ratios can be realized as bounded bank leverage targets (the ratio of illiquid assets to bank equity, or similar).

The Bank of England lends against counterparty collateral and publishes collateral eligibility criteria (and associated haircuts) as part of the sterling monetary framework. Collateral must be of high quality and liquidity in order to qualify for intraday liquidity, operational standing facilities or short term repo (short term monetary policy operations). This 'Level A' collateral amounts to sovereign debt including gilts, sterling treasury bills and also standard sovereign and central bank debts issued by the governments and central banks of Canada, France, Germany, the Netherlands and the US[45]. Somewhat lower quality collateral is admissible for indexed long term repos and other operations. The Bank of England assesses the riskiness of collateral assets independently[46]. This lending policy could be implemented in the simulator if agent portfolios are sufficiently diversified. A more general policy for the central bank agent would include case by case counterparty risk and credit worthiness assessments.

### 4.13.1 Background and Legacy Model

In the legacy Mark One model (§3) it was impossible for borrowers to fail to secure loans from the central bank. Central bank loan applications were processed as follows: (a) if the balance sheet of the borrower contains sufficient stock assets not already collateralizing existing central bank loans, grant a new repo using these stock assets as collateral; otherwise (b) give the counterparty an uncollateralized loan. In the legacy model step (b) was a fallback funding mechanism to ensure that bank cash transactions could not fail. This unfortunately generated a feedback process in which commercial bank debt could be serviced by further debt to the central bank virtually ad infinitum, as option (b) amounted to an unlimited rollover facility. This could lead to hyperinflation among other deleterious effects. If the central bank agent was not present, the model created as many units of cash as were required by the borrower and deposited this sum in the bank cash reserve for free.

A model parameter that requires special attention is the initial size of the central bank cash reserve (the central bank cash endowment). It is incorrect to assume that Java floating point primitives are not subject to arithmetic rounding errors. In 2014 a simulation configuration was identified in which (a) the total cash (the sum of the sizes of all agent cash reserves) decreased gradually over a period of several thousand simulation cycles, (b) substantial central bank repo loan activity was observed, but (c) the missing cash was not accumulated in the long term by the central bank. It was found the user had set the size of

[44] *The Financial Policy Committee's review of the leverage ratio*, July 2014. Note that the calculation of the leverage ratio may differs somewhat between jurisdictions and that these ratios may be adjoined by other regulatory requirements based on the state.

[45] *Level A Collateral*, Bank of England, `bankofengland.co.uk/markets/Documents/money/publications/levelacollateral.pdf`.

[46] *The Red Book: The Bank's current operations in the sterling money markets.* January 2014, `bankofengland.co.uk/markets/Documents/money/publications/redbookcollateral.pdf`

the central bank cash reserve to $10^{303}$ monetary units so as to effectively endow the central bank with unlimited reserves. The combined size of all non-central bank reserves was (by way of example) of the order $10^{12}$ cash units, meaning that no individual repo was likely to have a principal value greater than $10^{10}$ moneytary units. At $10^{303}$ the unit of least precision for a regular desktop computer is of the order $10^{287}$, so in this case all interest payments on existing repo loans had no effect on the central bank cash reserve and were lost to the system even if the central bank had access to a mechanism to redistribute its profits. In the long term this numerical effect was sufficient to drain a noticeable amount of money from the rest of the economy.

As the central bank agent is (a) wealthy and (b) one of a kind and not one of many, it is straightforward to formulate bank–to–central bank loan applications as synchronous on–demand transactions. Put differently, central bank loans need not be made subject to delays or processed in batch at prearranged times in the simulation schedule. Agents may simply apply for central bank loans whenever they do not have enough liquidity to satisfy a settlement. A central bank loan application is thus a last resort when processing a settlement, in the sense that the settlement will not fail until a bank has been told it cannot secure a central bank loan. For historical reasons this synchronous bank–central bank relationship is called the bank cash flow injection step. The cash flow injection step is the last resort of a bank when processing a settlement. This setup has persisted as the codebase has evolved, and despite the clear advantage of its simplicity, this structure may need to be overhauled substantially in order to revise the interbank loan market.

Is is also instructive to note that in the legacy model central bank loan applications were always processed with surpluses: if the borrower required $x$ units of liquidity in order to satisfy a settlement and these funds were to be sourced from the central bank, then the borrower would in fact apply for repo loans of value $\frac{3}{2}x$. This multiplier was found to cause oscillations in the whole economy as it induced periodic excesses in bank repo loan interest repayments.

### 4.13.2 Conventional Policy Operations

The management of interest rates (oft referred to as 'conventional' monetary policy operations) is implemented in the standard model primarily via Taylor rules and repo lending policies. Note that degenerate Taylor rules with zero response parameters result in fixed central bank interest rates.

#### 4.13.2.1 Taylor rules

Taylor rules (named after John B. Taylor, Stanford) are monetary policy rules that specify how central banks should change their interest rates as a function of certain incoming economic measurements, for instance inflation and real GDP. They are reactive formulae that adjust interest rates according to trends in the economy.

In the interest of a very simple formulation, let $i$ be the short term interest rate and $i^*$ be some baseline path (eg. the federal funds target rate in the USA).

The idea behind Taylor rules is to formulate deviations $(i - i^*)$ in proportion to deviations of another variable, say $z$, from its target value $z^*$:

$$i - i^* = \theta_z \cdot (z - z^*),$$

where $\theta_z$ is a response parameter (a number).

A large variety of Taylor rules and associated target variables and response parameters have been studied in the literature. However, we opt for an implementation using rules of the above simple form as this allows us to implement the most important classes of policy rules using economic measurements that are already provided by the model. We want to make it possible in principle for users to formulate monetary policy experiments involving several different target variables $z$ so that they may observe the economic consequences of their choices.

Of particular importance as target variables are:

1. the inflation rate $\pi$, and
2. (the logarithm of the) real output $q$, Bryant et al. (1993).

Together these measurements yield a policy rule of the form:

$$i - i^* = \theta_\pi \cdot (\pi - \pi^*) + \theta_q \cdot (q - q^*). \qquad (*)$$

The quantity $\Delta q = q - q^*$ is called the 'output gap'. We can measure the output gap $\Delta q$ as the deviation of $q = q(t)$ at time $t$ from the linear trend established by $q(t')$ at earlier times $t' < t$.

Write $i^* = r^* + \pi$ where $r^*$ is the equilibrium interest rate (and $\pi$ is inflation). Assume that $\pi^* = 2, r^* = 2$ and set the response parameters $\theta_\pi = \theta_q = \frac{1}{2}$ to one half. This yields the classical Taylor rule Taylor (1999):

$$i = 2 + \pi + \frac{1}{2}(\pi - 2) + \frac{1}{2}(q - q^*). \qquad (**)$$

It has been noted that this rule approximately described the behaviour of the US Federal Reserve in the 1980s and 1990s – Taylor (1999).

To conduct more sophisticated policy experiments, we may want to extend $(*)$ to a more general rule of the form:

$$\begin{aligned} i = (1 - \theta_i)(r^* + \pi^*) + \theta_i \cdot i_{-1} + \theta_\pi \cdot (\pi - \pi^*) + \\ \theta_q \cdot (q - q^*) + \theta_{\Delta q} \cdot (\Delta q - \Delta q^*), \end{aligned} \qquad (\dagger)$$

where $i_{-1}$ is the interest rate in the previous timestep Orphanides (2003). Note that $(**)$ is a special case of $(\dagger)$ when using $\theta_i = \theta_{\Delta q} = 0$.

The generalized Taylor rule $(\dagger)$ introduces two additional mechanisms. Firstly, it implements an exponential moving average for the interest rates and thus adds inertial behaviour to the policy instrument. Secondly, it allows for a response to a change in economic activity as measured by the difference between actual

output growth and its potential: $(\Delta q - \Delta q^*)$. This second feature is of interest, because when $\theta_i = 1$ and $\theta_q = 0$ we recover another family of important policy rules of the form:

$$i - i^* = \theta_\pi \cdot (\pi - \pi^*) + \theta_{\Delta q} \cdot (\Delta q - \Delta q^*).$$

These rules are particularly relevant if one does not (or cannot) target the real output $q$, but only the potential growth in $q$. In addition one can specify a parameter $\Delta i > 0$ to obtain the 'discount window':

$$\max(i - \Delta i, i_{\min}) < i < i + \Delta i$$

where $i_{\min} > 0$ is the smallest acceptable interest rate.

#### 4.13.2.1.1 Implementation

The Taylor–rule like output tracking central bank interest rate decision rule used by the standard model is as follows:

1. Compute the sum $(q)$ of all production yields among firms:

   $$q = \sum_{f \in F} q(f),$$

   where $F$ is the collection of all firms $f$ in the authority of the government in whose state the central bank operates.

   Note that $q(f) \geq 0$ is the most recently recorded production for firm $f$. Depending on the simulation time $T$, this may represent production in the current simulation cycle $(\lfloor T \rfloor)$ or in the last simulation cycle $(\lfloor T \rfloor - 1)$.

2. Record $q$ as the newest element in a circular buffer $Q$ of length $w \geq 1$ (whose elements are indexed by simulation cycle). $w$ is a customizable integer model parameter describing the duration of the memory of the central bank with respect to aggregate firm outputs.

3. Set $\alpha = \dfrac{2}{\beta + 1}$, where $\beta$ is a customizable model parameter (the 'central bank moving average memory').

4. Set $q^* = Q(T - w)$, and for each $x = T - w, T - w + 1 \ldots T - 1$ repeat:

   $$q^* = \alpha Q(x) + (1 - \alpha) q^*.$$

   Equivalently, for $w > 1$, set

   $$q^* = \sum_{x = T - w + 1}^{T - 1} \alpha (1 - \alpha)^{T - x} Q(x) + (1 - \alpha)^w Q(T - w).$$

   Note that for $w = 1$ this expression is $Q(T - 1)$, and for $w = 2$ it is a first order smoothing $\alpha Q(T - 1) + (1 - \alpha) Q(T - 2)$.

5. Set
$$\overline{q} = \frac{1}{w} \sum_{x=T-w}^{T-1} Q(x).$$

6. Revise the existing interest rate $r$ according to
$$r \mapsto r + \Delta r,$$
where
$$\Delta r = \eta \cdot \log\left(\frac{q^*}{\overline{q}}\right).$$
In this expression $\eta \geq 0$ is a customizable model parameter (the 'output response parameter').

7. If $r - r_S < \epsilon$, where $r_S$ is the interest rate spread and $\epsilon$ is a small customizable number, then set $r = \epsilon + r_S$.

Note that this formulation does not intrinsically guarantee that $r \geq 0$. The decision rule applies $r \mapsto \max(r, r_{\min})$ as a final instruction, where $r_{\min}$ is a customizable parameter, to ensure this. This edge case must be explicitly checked as $r_S$ (the interest rate spread) is decided by other means.

In 2014 it was observed that 'not a number' floating point values were being propagated through the limit order book structures during some simulations. The source of these numerical errors was eventually traced to step (6) in the above interest rate decision rule. Note that $-\Delta r$ is unbounded as $\overline{q} \longrightarrow 0$ for $q^* > 0$. For this reason a sharp decrease in aggregate firm production $Q(T) \approx 0$ can be translated into an interest rate perturbation $(r + \Delta)r \ll 0$. The resulting interest rate was incompatible with the computation of interest sums payable on new repo loans. For small $w$ (in periods of extreme firm duress and in some input-output network economies) it may also be the case that $\overline{q}$ and/or $q^* = 0$. Both of these edge cases resulted in unrecoverable arithmetic errors that subsequently propagated through the simulator.

For this reason the implementation of the central bank Taylor rule uses
$$\Delta r = \eta \log\left(\frac{\max(q^*, \varepsilon)}{\max(\overline{q}, \varepsilon)}\right)$$
where $1 \gg \varepsilon > 0$ is a number that is small compared to the expected central bank rate ($\varepsilon = 10^{-10}$ is used by default).

If the Taylor rule is disabled by the user then the central bank interest rate is fixed according to $r = r^*$, where $r^*$ is a customizable model parameter. $r^*$ is also the initial value of the central bank interest rate and the value of the interest rate selected by the central bank for $T < w$ (namely, at times $T$ when the moving average window has not yet collected enough data to implement the above process).

#### 4.13.2.2 Uncollateralized Lending Policies

The central bank agent may offer non–repo uncollateralized loans ('cash injection loans') to other banks at its discretion. This facility is risky for the central bank as it provides no collateral or counterparty default insurance: counterparty default will result in the central bank losing the loan principal, and although the central bank may receive stocks or other assets as compensation in the event of counterparty bankruptcy, in general the loss should be considered unrecoverable. The risk position of the central bank is far higher with respect to uncollateralized injection loans than repo loans.

In the standard model banks will apply for uncollateralized cash injection loans (UCILS) if they are insolvent and either (a) they have insufficient assets to offer for repo loans or (b) their existing assets are already serving as collateral. The central bank is required to decide whether or not to permit or deny the UCIL application on a case by case basis, and this decision may depend on the state of the counterparty and/or the states of the central bank agent and the economy as a whole. As the counterparty either has low assets or its balance sheet is saturated by collateral at the time of the UCIL application, uncollateralized loan applications should be considered more likely when the counterparty is troubled and when the economy is experiencing a liquidity crisis.

The central bank UCIL decision rule is an object and is a model parameter in the meaning of §4.3. These decision rules are intended to be interchangeable. The standard model provides the following implementations of the UCIL decision rule:

1. **Always Deny**
   Potentially the simplest possible central bank UCIL decision rule. This module always rejects the counterparty loan application. This module is outlined in §4.13.2.2.2.

2. **Risk Assessment**
   This decision rule provides a yes/no outcome of the form $(r(c) < R)$ where $R$ is the maximum acceptable risk of the counterparty and $r(c)$ is the estimated actual counterparty risk. This module is illustrated in §4.13.2.2.3. Note that $R \leq 0$ is equivalent to an 'always approve' UCIL decision rule.

It should be noted that an injudicious central bank UCIL policy may amount to an unlimited rollover facility for troubled commercial banks. Put differently, if a central bank has a large cash reserve and always approves uncollateralized loans, then any bank can finance its debt by taking out additional UCILs at any time. This debt may include interest on existing UCILs.

##### 4.13.2.2.2 Always Deny

The central bank provides no UCILs. Note that this implementation may be a reasonable representation of the Bank of England post–2008 lending policy.

#### 4.13.2.2.3 Risk Assessment

The UCIL application is approved when and only when $r(c) < R$, where $r(c)$ is the (believed) risk of the counterparty $c$ and $R$ is a customizable risk threshold. $R$ and $r(c)$ are expected but not required to be positive. $R(c)$ is computed by a dedicated delegate decision rule whose purpose is to measure the risk of $c$. The means by which this delegate decision rule measures the risk of $c$ is specified by the implementation. Examples include risk measurements based on the leverage of the borrowing bank and risk measurements based on the borrower's exposure.

The risk assessment UCIL decision rule has the additional feature that the counterparty is not permitted to take out two UCILs at once. If the counterpart already has a central bank UCIL then it is not allowed to take out another until the original is either fully repaid or written off. This (perhaps excessively simple) condition is intended to prevent debt rollover for banks with otherwise reasonably healthy balance sheets as it is impossible for a bank to secure a second UCIL in order to pay interest on the first.

Note that $R(c)$ can be implemented as a noisy risk measurement or a random number generator. In the latter case UCILs are approved or rejected with a fixed probability (independently of $c$).

### 4.13.3 Stock Market Participation

The central bank may receive stock assets as compensation in lieu of cash for bank bankruptcy resolution (if the central bank has provided uncollateralized cash injection loans §4.13.2.2.3) or if collateral is seized following repo default. In the absence of a stock market participation strategy (and assuming that the underlying stock instrument is never erased or diluted) the central bank will retain these shares permanently.

The standard model provides an elementary strategy for gradual central bank stock resale. If the central bank owns a total value $V \geq 0$ in stock (potentially distributed over many instruments/types of shares) then the central bank strategy is to plan to own $\alpha V$ in shares after the next stock market processing session. $\alpha$ ($0 \leq \alpha \leq 1$) is a model parameter and is specified by the user.

In order to realize this asset value the central bank must respond to changes in the stock market price per share $p$. If the central bank owns $N$ shares with value $pN$ before stock market processing, and if the revised price per share selected by the market is $p^*$, then the intended central bank transaction at the stock exchange is as follows:

1. If $p^* > p$ then the value of the central bank stock asset before any transactions is $p^*N$. This represents a capital gain of $(p^* - p)N$. The central bank aims to retain a stock portfolio of value $\alpha pN$, so the central bank agent bids to sell $(p^*N - \alpha pN)/p$ shares in order to realize this. The resulting flow of funds enters the central bank cash reserve as new unallocated liquidity.

2. If $p^* < p$ then the capital value of existing shares has decreased as a result of price changes in the market. If $p^*N < \alpha pN$ then the central bank would be required to buy additional shares in order to realize the target stock portfolio value $\alpha pN$. In this case the bank (a) takes no action if $\alpha > \frac{p^*}{p}$, and otherwise (b) sells $(p^*N - \alpha pN)/p$ shares. The resulting flow of funds, if any, enters the central bank cash reserve.

This policy describes a simple stock market participation strategy in which (a) the central agent will never buy any new shares, and (b) the value of all shares owned by the central bank will decrease exponentially (at a rate $\alpha$) if this can be realized after the change in capital value: otherwise the central bank will retain all its shares and accept the underlying capital loss.

Note, hypothetically, that if the stock price $p$ is never revised by the market then the value of the central bank stock portfolio evolves as the series $V, \alpha V, \alpha^2 V....$ When $n$ simulation cycles have elapsed the value of the portfolio is $V(1 - \alpha^{n+1})/(1 - \alpha)$, a quantity that is never zero. For this reason the user is asked to specify a cutoff value threshold $V^* > 0$ at which point the central bank will attempt to sell its entire outstanding stock portfolio without regard for further exponential decreases in value: when $V < V^*$ the target central bank stock portfolio value is zero rather than $\alpha V$. If this numerical cutoff $V^*$ is strictly positive then in principle the central bank is able to sell the entire stock portfolio in a finite number of business cycles. It is acceptable for the user to select $V^* = 0$.

### 4.13.4 Simplifying Assumptions

The central bank agent in the standard model is a highly simplified representation of the operations of real state central banks and the Bank of England. For legacy reasons, and in the interest of maintaining backward compatibility with earlier models and versions of the codebase, all types of stock account (firm and bank stocks) are currently eligible as repo loan collateral with the central bank[47].

The central bank repo loan policy is a yes/no decision (whether or not to issue a repo loan to the counterparty) based on microprudential considerations and/or the overall state of the economy. At present the central bank will accept any stock assets not already collateralizing existing repo loans when borrowers apply to it for new repos. This implementation is subject to change in future. Stocks cannot be collateralized twice and are recorded in central bank registers in order to avoid this. Dividends received on stock assets (and any capital gains associated with changes in share price) remain the property of the repo loan counterparty (the original owner of the stock) and not the central bank agent that holds these assets in collateral. This implementation is also subject to change.

Although repo loans are currently renewed every simulation cycle it is technically possible for a borrower to receive a repo and for the value of the associated collateral to decrease (as a result of stock market processing) before maturity.

[47]Mutual funds stocks are not traded on the exchange, they are owned by fund investors (households) and redeemed with the issuing fund as necessary.

In reality the counterparty might be required to supplement the existing collateral in these circumstances, however this possibility is currently neglected by the standard model.

At present the standard model has no secondary gilt/government bond market (no agents in the standard model make use of extended maturity coupon bonds, therefor a secondary gilt market has no utility, as government bonds have homogeneous terms and synchronous maturities). The absence of a secondary gilt market means that the central bank can potentially behave as a numerical money sink with respect to profits accrued on central bank repos. Concretely, if the central bank repo interest rate is nonzero and no repo loan borrowers ever fail, then the value of the central bank balance sheet will increase in time as a result of the central bank having received the repo loan principals plus interest at maturity. In simulations with significant, sustained repo loan activity the resulting money drain may prove to be noticeable (in the form of liquidity crises of increasing severity) after several hundred years have elapsed on the simulation clock.

The only viable solution to this liquidity circulation problem is for the central bank agent to periodically transfer its profits to the government agent at the end of the simulation cycle (unlike the central bank the government agent has the ability to recirculate this money back into the economy via several different channels). Long term central bank profits, $\pi = c - c^*$, are considered to be cash reserve sums $c$ in excess of the original central bank cash endowment $c^*$ (the size of the central bank cash reserve at its initialization, $T = 0$). This relationship between the central bank agent and the government is a short term solution to the circulation problem and is subject to change in future.

The step interval (clock resolution) of the simulator is currently interpreted as one financial quarter and the above central bank Taylor rule §4.13.2.1.1 adjusts the interest rate as often as the clock ticks. Real central banks may adjust interest rates more or less frequently than this and/or on fixed announcement dates (for instance the central bank of Canada has a system or eight pre-specified official interest rate change announcement dates per year). It is straightforward to reduce the frequency of central bank rate adjustments in the standard model, but naturally it is is not helpful to adjust the central bank rate in subdivisions of the clock resolution of the rest of the simulation world (namely, more often than once per quarter).

Bank–to–bank financial transactions (ie. credit/debit cycles between banks) ultimately take place at the central bank. Real commercial banks typically have deposit accounts with the state central bank and use these accounts to transfer money to one another. Concretely, if a depositor at bank $A$ transfers money to a depositor at bank $B \neq A$ then the transaction may in fact be executed as a transfer between central bank deposit accounts followed by abstract changes to the value of client liabilities held by $A$ and $B$. In the United Kingdom the payer ($A$) is not allowed to enter a central bank overdraft state as a result of the transfer.

Unfortunately, realizing this exact mechanism in the simulator would require

all financial transactions anywhere in the economy (a large number of transactions) to be transmitted to the central bank agent in order to be processed. This requirement is incompatible with the optional nature of the central bank agent and may also complicate any future concurrent (eg. MPI) event scheduling techniques because thread pool bottlenecks may be unavoidable in parallel applications. For this reason banks in the standard model keep their cash in local dedicated balance sheet assets ('cash reserves') which represent central bank deposits. The cash reserve asset is not implemented as a deposit account. Cash reserve assets are not allowed to have negative value (central bank overdrafts).

In the standard model the central bank is subject to no expenses of its own and does not currently employ a workforce. Loan contracts cannot be offered as collateral for repos. Central bank repo loans and uncollateralized cash injection loans currently have homogeneous maturities.

## 4.14 Bad Bank

A 'bad bank' is a technical term for a vehicle that takes ownership of (or partial responsibility for) toxic balance sheet assets in order to isolate risky, illiquid securities from an existing bank's healthiest assets. The bad bank is an establishment in its own right, and its creation often entails the division of an existing bank into distinct 'good bank' and 'bad bank' entities with different concerns. Investors may have greater confidence in the survivability and stability of the 'good bank' as a result of it having unloaded low quality assets onto the 'bad bank'.

The formation of the bad bank and the details of the relationship between the bad bank and the good bank, the continuing relationship (if any) between the good bank and the assets it has discharged, and the relationship between both banks and the local government differs among jurisdictions. The bad bank may be a publicly sponsored agency formed as a result of a national rescue plan, and the good bank may retain some (eg. capped) liability for losses on the assets it has transferred. The remit of the bad bank may be to simply administer these low quality assets until their maturity, and it should be noted that the bad bank may in fact be an asset management company rather than a true bank.

This so–called good bank/bad bank strategy has been employed repeatedly since at least the 1980s. An early example was the branching off of Grant Street National Bank (GSNB) from Mellon Financial (a high net–worth individual and institutional asset management bank) in 1988. GSM was initialized with $1.4bn dollars worth of low quality loans. Grant Street Bank satisfied its purpose and was dissolved seven years later in 1995. Similar programs were undertaken by the government of Finland during the early–1990s Finnish banking crisis, though the creation of an asset management company called Arsenal, in Belgium around 2008, in the Baltic new European Union states in the period 2008–2011, in Spain in 2012 and in Portugal in 2014[48].

In 2008 UBS volunteered for a state–sponsored bad bank program (due to

[48] eg. `en.wikipedia.org/wiki/Bad_bank`

exposure to subprime mortgage securities) and was permitted to transfer up to $60bn in toxic assets to it, or about 3% of its total assets at the time[49]. The UK government founded UK Asset Resolution in October 2010, a limited holding company based in West Yorkshire, which continues to manage the closed mortgage books of Bradford and Bingley and Northern Rock Assset Management, and assets worth about £60bn spread across nearly half a million customers[50].

In the standard model the bad bank is an agent of last resort whose purpose is to hold or sell assets that belonged to failed commercial banks. The bad bank is therefore instrumental in commercial bank liquidation, a process that will ultimately result in the removal of the failed bank from the simulation world. The bad bank is a unique optional agent in the simulation and therefore its services are shared among all other banks. The bad bank agent should be viewed as a public sector body.

### 4.14.1 Background and Legacy Model

In the Mark One model §3 the bad bank was a regular bank with a null strategy. Once per simulation cycle, when the stock market was processed, the bad bank would offer any and all shares it owned up for sale. As its purpose was mainly to sit on existing assets until these assets matured, the bad bank interacted with no markets other than the stock market.

The original implementation of the bad bank agent was complicated by the processing policy of the legacy stock market. In the legacy stock market shares would disappear from participant balance sheets unless they were bought anew every simulation cycle. For this reason the bad bank had exactly one opportunity to sell any shares it owned before these shares were erased. With the introduction of stock flow consistency (shares are conserved by all stock market transactions and are only created or destroyed by special instruction) the bad bank and other stock market participants gained the ability to retain shares in the long term. For this reason the bad bank share resale policy was updated to mirror the government and central bank share resale policies (4.13.3) in that that bad bank will not oversupply the market by selling its entire portfolio at once. Rather the bad bank will try to sell its stock assets gradually.

A model parameter that requires special attention is the initial size of the bad bank cash reserve (the bad bank cash endowment). It is incorrect to assume that Java floating point primitives are not subject to arithmetic rounding errors. If the user selects a huge bad bank cash reserve (in the interest of never allowing the bad bank to fail, say $10^{100}$) then it is expected that all loan interest and principal sum repayments received by the bad bank on assets it is administering will be lost due to the difference in size between the unit of least precision of the bad bank cash reserve and the unit of least precision of the size of a typical loan principal. This observation is perhaps less important than the corresponding observation for the central bank agent, as the bad bank provides an upfront sum (totalling the face value of all assets transferred to it) immediately after commercial bank liquidation. The subsequent flow of funds from the rest of the

[49] *The Economics of Bank Restructuring: Understanding the Options*, IMF, August 2012.
[50] `http://www.ukar.co.uk/`

economy into the bad bank serves only to allow the bad bank to restore its cash reserve following the initial expense, and since the bad bank has no other means of injecting cash into the economy, the extent to which the bad bank can act as a numerical money fountain is perhaps more limited than is the case of a central bank with an equally huge cash reserve.

### 4.14.2 Unconventional Policy Operations

Interventions during financial crises (oft referred to as 'unconventional' monetary policies) are implemented in the standard model primarily in terms of bank bankruptcy policy modules. Whereas the failure of non-financial sector firms does not require any substantial action to be taken by the government, the central bank[51] or the bad bank agent, the failure and subsequent liquidation of banks may result in specific actions to be taken by the bad bank.

In the standard model the bad bank is involved in the following bank bankruptcy resolution policies:

1. **Liquidation**
   A last resort bankruptcy resolution technique in which all efforts to continue the operations of a failing bank are abandonned. The bank is shut down as soon as possible and its assets are redistributed to secured debtors and then unsecured debtors on a pro rata basis. The creditors of the failed bank are (in aggregate) expected to suffer losses at least as large as the negative equity of the failed bank. This process is outlined in §4.14.2.1.

#### 4.14.2.1 Bank Liquidation

When an institution is liquidated it is permanently closed and its balance sheet assets are sold, over time, to pay outstanding liabilities to depositors and creditors. This process is modelled by transferring the institution's assets to the bad bank.

The bad bank immediately provides funds totalling the face value of the assets that were transferred to it. These funds are used to pay off and compensate the institution's debtors as much as possible. Secured debtors take priority, and any remaining funds are distributed on a pro rata basis to unsecured debtors. The negative equity of the failing bank at the time of its liquidation is the (combined) capital loss suffered by debtors and depositors. Households and firms that held their deposits at the liquidated bank are instructed to open up new deposit accounts at randomly chosen competing banks, if there are any such banks. All outstanding balance sheet contracts are then summarily terminated without compensation and the bank is isolated from the financial system. The liquidated bank is unsubscribed from any active clearing houses and markets.

Note that the bank liquidation process is a permanent change in the running simulation as it results in the loss of at least one bank agent. As there are a finite number of bank agents in the simulation and as spontaneous bank respawning is currently not modelled, bank liquidation can only occur a finite number of

[51] if present

times before the financial system becomes inoperative. This 'financial system collapse' is a state from which the simulation cannot continue.

#### 4.14.2.2 Financial System Collapse

Bank liquidation can result in simulation termination. The standard model is (ordinarily) required to have a functional financial system because (a) households, funds and firms/establishments are unable to store significant amounts of cash outside banks and (b) because these agents cannot process financial transfers to one another without financial services. If, as a result of a chain of bank liquidations, there are no banks in the financial system, then the simulation will raise an instance of `FinancialSystemHasCollapsedException` and terminate. This state is indicated by the console.

Note that liquidation is a last resort bankruptcy resolution technique. Liquidation may be called, for instance, if bailout or bailin bank bankruptcy resolution policies have failed. For this reason it cannot in general be assumed that any simulation will continue forever.

### 4.14.3 Stock Market Participation

The bank bank may receive stock assets as compensation in lieu of cash for loan counterparty bankruptcy resolution or if collateral on existing assets is seized. In the absence of a stock market participation strategy (and assuming that the underlying stock instrument is never erased or diluted) the bad bank will retain these shares permanently.

Note therefor that the bad bank can receive stock assets via several different channels:

1. as a direct transfer of assets following the liquidation of a commercial bank (stocks on the balance sheet of the liquidated bank);

2. as compensation for the failure of a counterparty to honor the terms of a contract that the bad bank is currently administering, or

3. as a result the bad bank having seized collateral.

The bad bank uses the same share resale policy as the central bank agent. See §4.13.3 for a discussion of the central bank stock market participation strategy. The rate at which the bad bank sells its shares may be different from the corresponding rate at which the central bank (and/or government) sells shares.

### 4.14.4 Simplifying Assumptions

The role of the bad bank is currently limited to bank liquidation processes. The bad bank agent is not a truthful representation of a real bad bank because a good bank does not continue to exist alongside it. In this sense the bad bank is closer in terms of its implementation to a bridge bank, however there is no intention that the failed bank should be bought by another institution. The bad bank is a publically sponsored vehicle whose services may be shared by several

failed bank agents at once.

As the bad bank is active only after a commercial bank has been liquidated, it is currently not possible (not meaningful) for (a) profits on assets transferred to the bad bank to be partly returned to the good bank, (b) for the good bank to have partial liability for losses suffered on the assets it has transferred, or for (c) the original bank to pay for the initial equity of the bad bank and then transfer this equity to the government.

Liquidation represents a permanent change to the simulation world in that the balance sheet of at least one bank is sealed and this bank is removed from the financial system and conducts no further business. Unfortunately a model involving the spontaneous creation of a (non fixed) number of banks is currently at odds with the implementation of the CSV data recorder, as the creation of new bank agents requires the dimensions of a preexisting CSV file to be changed (and these dimensions are not modified by the recorder in the lifetime of the simulation).

# 5 Financial sector

An important component of this model is the financial sector. The version presented below is still a rather abstract representation of the financial system which will be elaborated later; the discussion of the interbank market is deferred to section 6.

An earlier version of the financial sector was based around a limit order book mechanism. As described in an earlier report, we ran a contest in which different groups designed their own banks, which uncovered some failings in this model. In particular, even under idealized conditions, there was a failure to reach a proper equilibrium between interest rates and stock prices, and typically returns has strong and unrealistic autocorrelations. This motivated us to convert to a system based on market clearing, as described here.

## 5.1 Balance sheets

The financial sector is comprised of a set of financial intermediaries indexed by $j \in \{0, ..., N_b\}$. Intermediaries are characterized by their balance sheet, their investment strategy, and potential regulatory requirements. Intermediaries hold a portfolio of three types of assets: cash, commercial loans to firms in the productive sector (in the following simply referred to as "loans") and shares in the stock of firms in the productive sector (in the following simply referred to as "stocks"). In general cash is a non-interest bearing risk-free asset, whilst both stocks and loans are risky assets. In the special case in which the default probability on a loan is zero, loans can be considered an interest bearing risk-free asset comparable to treasury bonds or highly ranked commercial paper. The

asset side of the intermediary balance sheet is therefore given by:

$$\begin{aligned}\mathcal{A}_{j,t} &= c_{j,t} + \mathbf{s}_{j,t}^T \mathbf{p}_t + \sum_{i=1}^{N_f} l_{ji,t}, \\ \mathbf{s}_{j,t} &= \left(s_{j1,t}, ..., s_{jN_f,t}\right)^T, \\ \mathbf{p}_t &= \left(p_{1,t}, ..., p_{N_f,t}\right)^T,\end{aligned}$$

where $c_{j,t}$ is the cash investment, $\mathbf{s}_{j,t}$ the vector of stock ownership, $\mathbf{p}_t$ is vector of share prices, $N_f$ is the number firms in the productive sector and $l_{ji,t}$ is the book value at time $t$ of a loan extended from intermediary $j$ to firm $i$. Note that the total number of shares of a given firm is normalized to 1 such that:

$$\sum_{j=1}^{N_b} s_{ji,t} = 1 \quad \forall i, t.$$

Also note that intermediaries face a long only constraint, i.e. $s_{ji,t} \geq 0$. This implies that intermediaries cannot short an asset which they consider overpriced. This limits the extent of arbitrage that is possible in this setting. Mispricings may therefore be able to persists longer than in the case of permitted short selling.

By definition the equity of the intermediary is given by:

$$\mathcal{E}_{j,t} = \mathcal{A}_{j,t} - \mathcal{L}_{j,t},$$

where $\mathcal{L}_{j,t}$ generically represents all liabilities of intermediary $j$ at time $t$. We abstract from modelling these liabilities in any more detail. Furthermore we assume, for simplicity, that the intermediaries do not face funding restrictions, i.e. should the intermediary decide to increase its liabilities it always finds willing lenders. In a future, more realistic version of this model, it will be crucial to model funding constraints explicitly due to the importance of liquidity and roll-over risk for systemic stability, see Shin (2010) or Caccioli et al. (2013).

Finally, we define the intermediary's leverage as:

$$\lambda_{j,t} = \frac{\mathcal{A}_{j,t} - c_{j,t}}{\mathcal{E}_{j,t}}.$$

## 5.2 Unconventional policy operations

Central bank interventions in times of financial crisis (oft referred to as 'unconventional' monetary policies) are implemented in the standard model primarily via bankruptcy resolution policy modules.

### 5.2.1 Crisis Resolution

The central bank agent encapsulated four default strategies for bank insolvencies. Two of them ('liquidation' and 'purchase & assumption') provide an ordered way to terminate a failed bank. The other two crisis resolution techniques lead to a continuation of the failed bank's operations – 'bailin' and 'bailout'. Two other agents are introduced to execute these default strategies, the *government* and a *bad bank*. The government collects taxes from firms, households,

and other banks. These funds are then used to assist failing banks. The bad bank agent assumes a failing bank's foul assets and sells them off in an ordered way. More details on these two agents will be provided below.

1. **Liquidate**
   When an institution is liquidated, it is closed and its assets are sold over time to pay its liabilities to depositors or other creditors. This is modelled by transferring the institution's assets to the bad bank and receiving immediately the funds equaling the face value of the sold assets. These funds are then used to pay off the institution's debtors as far as possible. Secured debtors (e.g. through repurchase agreements) take priority, the remaining funds are distributed *pro rata* among unsecured debtors. The negative equity of the bank leads to a capital loss of the bank's debtors and depositors. Households and firms, who held their deposits at the liquidated bank, open up a new deposit at a randomly chosen competing bank.

2. **Bailin**
   A bailin is a balance sheet restructuring of the troubled institution. Loans and deposits are converted into bank equity. To implement this debt-to-equity conversion, we compute the size of the bailin as the negative equity of the failing bank, plus an overhead ensuring continuity of operations after the bailin. These funds are then subtracted from each deposit or loan *pro rata* and converted into capital.

## 5.3 Budget constraint

Following LeBaron (2012) and Hommes and Wagener (2009), the budget constraint of the intermediary is given by:

$$\begin{aligned}\mathcal{E}_{j,t} &= c_{j,t} + \mathbf{s}_{j,t}^T \mathbf{p}_t + \sum_{i=1}^{N_f} l_{ji,t} - \mathcal{L}_{j,t} \\ &= c_{j,t-1} + \mathbf{s}_{j,t-1}^T (\mathbf{p}_t + \boldsymbol{\pi}_{t-1}) + \sum_{i=1}^{N_f} l_{ji,t-1}(1 + r_{ji,t-1}) - \mathcal{L}_{j,t-1}(1 + r_{L,t-1}),\end{aligned}$$

where $\boldsymbol{\pi}_{t-1}$ is the $N_f \times 1$ vector of profits generated by the firms in the productive sector at time $t-1$, $r_{ji,t-1}$ is the interest rate at which intermediary $j$ lent to firm $i$ at time $t-1$ and finally $r_{L,t-1}$ is a generic funding cost for the intermediary.

We postulate that, on average, the intermediaries' income through dividends and interest payments should be comparable to its funding cost. This is a strong assumption and simplifies the decision problem of the intermediary significantly. Since we do not model the intermediaries liabilities it is difficult to model its funding cost. Hence this assumption is very useful. In the long run this assumption can be justified by requiring that money flows in the economy are conserved, i.e. the financial system cannot be a sink of money flows. Instead, payments from the real economy to the financial system have to be recycled into the real economy. We satisfy this requirement on every time step by assuming that dividend and interest income are equal to the intermediary's funding cost.

This implies that equity only changes via changes in the prices of the stocks in which the intermediary holds long positions. In particular, this assumption yields the following simplified budget constraint:

$$\begin{aligned}\mathcal{E}_{j,t} &= c_{j,t} + \mathbf{s}_{j,t}^T \mathbf{p}_t + \sum_{i=1}^{N_f} l_{ji,t} - \mathcal{L}_{j,t} \\ &= c_{j,t-1} + \mathbf{s}_{j,t-1}^T \mathbf{p}_t + \sum_{i=1}^{N_f} l_{ji,t-1} - \mathcal{L}_{j,t-1}.\end{aligned} \tag{2.1}$$

## 5.4 Leverage dynamics

Leverage, in particular the so called leverage cycle, has played an important role in the past financial crisis. The existence of the leverage cycle has been documented in, among others, Adrian and Shin (2010) and Shin (2012). Theoretical models of the leverage cycle have been developed in Adrian and Boyarchenko (2012) and Geanakoplos (2009). Despite not modelling leverage explicitly in the results that are presented in the simulator deliverable 8.3, we have explored three approaches to modelling leverage dynamics:

- We impose no constraints or targets on leverage (as in the results presented for deliverable 8.3). This is an unrealistic assumption but serves as a good benchmark case.
- We impose a target leverage. The intermediary will expand or contract its balance sheet to achieve the target leverage by taking on more debt or liquidating assets. This is confirmed for commercial banks in an empirical study done by Adrian and Shin (2010).
- We impose a Value-at-Risk constraint. It is widely believed that adherence to a Value-at-Risk (VaR) constraint leads to the procyclical leverage observed for investment banks as described in Adrian and Shin (2010).

The third approach is most interesting and will be briefly outlined below. The $VaR$ of a portfolio of size $\mathcal{A}$ at a certain confidence level $c$ is defined as the smallest non-negative number such that probability that the value of the portfolio falls below a particular threshold $\mathcal{A}_0 - VaR$ is less than $1 - c$, i.e.

$$P(\mathcal{A} < \mathcal{A}_0 - VaR) \leq 1 - c.$$

As outlined in Shin (2010) and Adrian and Shin (2010), due to regulatory requirements an intermediary's $VaR$ is typically related to its economic capital or equity by some factor of proportionality; in particular intermediaries must ensure that their $VaR$ does not exceed their equity. Effectively this leads to a VaR constraint of the form $VaR \leq \mathcal{E}$. Assuming normally distributed asset returns[52] we have for the $VaR$ of intermediary $j$ at time $t$:

$$VaR_{j,t} = a\mathbf{w}_{j,t}^T \Sigma_t \mathbf{w}_{j,t} \mathcal{A}_{j,t} \leq \mathcal{E}_{j,t},$$

[52] Of course, it is well known that asset returns are heavy tailed, see for example Cont (2001) for a review.

where $a$ is a parameter. $\mathbf{w}_{j,t}$ is the vector of portfolio weights and $\Sigma_t$ is the covariance matrix of asset returns. Therefore, ignoring cash holdings and assuming that the VaR constraint is binding[53], the per-dollar $VaR$ is simply the inverse of the intermediary's leverage, i.e.

$$\frac{VaR_{j,t}}{\mathcal{A}_{j,t}} = a\mathbf{w}_{j,t}^T \Sigma_t \mathbf{w}_{j,t} \approx \frac{1}{\lambda_{j,t}}. \tag{2.2}$$

Therefore, an intermediary with VaR constraint will reduce its leverage as its portfolio variance increases and increase its leverage in the opposite case. This behaviour is generally referred to as procyclical leverage.

## 5.5 Portfolio Optimization

Problem: for asset types $\mathcal{A} = \{A_i\}$ (instruments in which to invest) and liabilities $\mathcal{L} = \{L_j\}$ (eg. borrowing options) identify (a) optimal investments $\{I_i^A\}$ and (b) optimal debts $\left\{I_j^L\right\}$ such that net returns over thes portfolio are maximized. Total investment is subject to a capital constraint (an upper bound, $\Omega$, on $\sum_i I_i^A$) and the equity of the investor should remain unchanged before and after the formation of the portfolio (namely, assets are paid for).

$$\begin{aligned}
&\max_I \left( \sum_i r_i^A(I_i^A) I_i^A - \sum_j r_j^L(I_j^L) I_j^L \right) \qquad (*) \\
\text{s.t.} \quad & I_i^A \geq I_{i,\,\min}^A, I_j^L \geq I_{j,\,\min}^L \quad \forall I_i^A, I_j^L, \\
& \sum_i I_i^A \leq \Omega, \\
& \sum_i I_i^A - \sum_j I_j^L \;=\; \Delta E \\
& r_i^A(x) = c_i^A + \alpha_A \cdot x \qquad c_i^A \geq 0, \alpha_i^A \leq 0, \\
& r_j^L(x) = c_j'^L + \alpha_L' \cdot x \qquad c_j^L \geq 0, \alpha_j^L \geq 0,
\end{aligned}$$

where $\Delta E$ might be the cash $C \geq 0$ already availiable to the investor. In practice the lower bounds $I_{i,\min}^A$ can be subtracted from $\Omega$ and the return rates $r^A, r^L$ adjusted to account for the mandatory sums $I_{i,\min}^A$, $I_{j,\min}^L$. In which case wlog. $I_{\min}^A = 0$, $I_{j\min}^L = 0$ and $\Delta E = C$.

## 5.6 Portfolio decision

In the following we present simple portfolio allocation strategies for risk neutral and risk averse investors. These allocation strategies are coupled to two distinct expectation formation methods: fundamentalist and trend following expectations. Note that expectations are adaptive and investors are myopic. In general a combination of portfolio allocation strategy and expectation formation method yields a set of portfolio weights for the investment assets (stocks,

[53]For a profit maximising intermediary the VaR constraint is binding as shown in Corsi et al. (2013).

loans - we assume that cash is, if present, a residual). We have for the vector of portfolio weights:

$$\mathbf{w}_{j,t} = \mathbf{F}\left(\mathbb{E}\left[\mathbf{r}_{t+1}\right]\right),$$

where $\mathbf{r}_{t+1}$ is the vector of asset returns. Both $\mathbf{r}$ and $\mathbf{w}$ are $N_f + 1 \times 1$ vectors. The definition of the asset returns may differ between allocation strategy and will be further explained below. Note that the expectation $\mathbb{E}\left[\mathbf{r}_{t+1}\right]$ is defined as the conditional mean of the return process:

$$\mathbb{E}\left[\mathbf{r}_{t+1}\right] = \mathbb{E}\left[\mathbf{r}_{t+1}|\mathcal{F}_t\right],$$

where $\mathcal{F}_t$ is information set including all relevant information up to time $t$. We use $\mathbb{E}\left[\mathbf{r}_{t+1}\right]$ for notational simplicity.

**5.6.0.0.1 Risk neutral investors** We define two risk neutral investors: a so-called "logit" investor and a so-called "mean-deviation" investor. The logit investor allocates its assets according to:

$$w_{kj,t} = \frac{e^{\beta\mathbb{E}[r_{k,t+1}]}}{\sum_l^{N_f+1} e^{\beta\mathbb{E}[r_{l,t+1}]}}, \tag{2.3}$$

where $\beta \geq 0$ is the intensity of choice parameter. $w_{kj,t}$ and $\mathbb{E}\left[r_{l,t+1}\right]$ are the portfolio weight and expected return of the $k$th asset, respectively. In the limit $\beta \to 0$ we have $w_{kj,t} \to \frac{1}{N_f+1}$ and for $\beta \to \infty$ we have:

$$w_{kj,t} = \begin{cases} 1 & \text{if } k = \arg\max_l \mathbb{E}\left[r_{l,t+1}\right]; \\ 0 & \text{otherwise.} \end{cases}$$

The no-shorting constraint that intermediaries face is automatically enforced. The mean-deviation investor compares every asset's return to the average return. It then increases or decreases the portfolio weight relative to the difference to the average return. In particular we have:

$$\mathbf{w}_{j,t} = \min\left[\mathbf{w}_{j,t-1} + \gamma\left(\mathbb{E}\left[\mathbf{r}_{t+1}\right] - \frac{1}{N_f+1}\sum_l^{N_f+1}\mathbb{E}\left[r_{l,t+1}\right]\right), w_{min}\mathbf{1}\right], \tag{2.4}$$

where $\gamma > 0$ is a learning rate, $w_{min} < 1$ is the minimum investment weight, "min" is the element wise minimum operator and $\mathbf{1}$ the $N_f + 1 \times 1$ vector of 1s. The minimum operator enforces the no-shorting constraint. The portfolio weight adjustment does not ensure normalization, therefore we normalize:

$$\mathbf{w}_{j,t} \leftarrow \frac{\mathbf{w}_{j,t}}{\sum_l^{N_f+1} w_{lj,t}}.$$

The idea of the mean-deviation investor is that it incrementally shifts his position towards the asset with the higher return. Eventually this incremental portfolio update pushes the returns to convergence. However, in order to achieve a parity between the dividend-price ratio and the interest rate we have to "iterate" the portfolio adjustment of the mean-deviation trader in a given time step to reach. The iteration is necessary to make the step size of the portfolio weight sufficiently large for the returns to balance.

**5.6.0.0.2 Risk averse investors** The standard approach to modelling risk averse investors follows a mean-variance approach, see for example Campbell and Viceira (2002). A prototype version of a mean variance investor was implemented. However, stability problems in the portfolio allocation have not yet been brought under control; this is reflective of fundamental problems in estimating risk. While we still aim to develop the mean variance investor further, we have also implemented a prototype risk averse investor that takes a more simple form. Below we briefly present the mean-variance and the "simple risk-averse" investor for completeness:

**Mean-variance investor:** In general intermediaries solve:

$$\begin{aligned} \max_{\mathbf{w}_{j,t}} \quad & \mathbf{w}_{j,t}^T \left( \mathbb{E}\left[\mathbf{r}_{t+1}\right] - r_f \mathbf{1} \right) - \frac{k}{2} \mathbf{w}_{j,t}^T \mathbf{\Sigma}_t \mathbf{w}_{j,t} \\ \text{subject to} \quad & w_{ji,t} > 0, \text{ for all } i \\ & \sum_i w_{ji,t} \leq 1 \end{aligned} \tag{2.5}$$

where $\mathbf{r}_{t+1}$ is the vector of risky asset return, $r_f$ is the return of the risk free asset, $k$ is a parameter controlling the investor's level of risk aversion and $\mathbf{\Sigma}_t$ is the sample covariance matrix of risky asset returns. Note that we approximate the expected sample covariance matrix at time $t+1$ by the sample covariance matrix at time $t$ conditional on all information available up to that point, see below for more detail. The accompanying constraints impose the no-shorting constraint. For our purposes we distinguish between two cases:

- Loans to the productive sector are considered risk free. In this case the vector of portfolio weights $\mathbf{w}$ has $N_f$ elements, $r_f$ is simply the interest rate paid on these loans and the investment into loans is the residual investment given by: $w_{j,t}^{loan} = 1 - \sum_i w_{ji,t}$.
- Loans are risky. In this case the risk free asset is cash and the vector of portfolio weights $\mathbf{w}$ has $N_f + 1$ elements. $r_f = 0$ and investor's cash holding is again given by the residual.

In the absence of no-shorting constraints the above maximisation problem can be solved analytically, Campbell and Viceira (2002). The optimal portfolio choice is given by:

$$\mathbf{w}_{j,t} = \frac{1}{k} \mathbf{\Sigma}_t^{-1} \left( \mathbb{E}\left[\mathbf{r}_{t+1}\right] - r_f \mathbf{1} \right),$$

where $\mathbf{\Sigma}_t^{-1}$ is the inverse of the sample covariance matrix of returns. However due to the no-shorting constraint we need solve the portfolio problem in eq. 2.5 numerically. This is a quadratic optimization problem for which we can use efficient, pre-implemented algorithms. We use the truncated Newton Conjugate-Gradient minimizer of the `minimize` function in Scipy - a scientific software package implemented in Python, Jones et al. (2001).

**Simple risk averse investor:** The idea was to implement an investor that responds to perceived risk in a straightforward and controllable way. For

simplicity we assume that the investor is only interested in the risk of each individual asset and treats each asset as independent[54]

The investor uses a portfolio allocation rule as defined in Eq. 2.4. However the investor discounts the expected asset returns by their estimated variance. In particular we have for an expected asset return $r$ and corresponding estimated variance $\sigma^2$:

$$r \leftarrow re^{-\theta\sigma^2}, \tag{2.6}$$

where $\theta$ is a risk aversion parameter. The variance is either estimated as outlined below or taken as constant assuming that the investors have fixed beliefs of the assets' risk. In the report on Deliverable 8.3 we briefly discuss the results produced by this investment strategy.

**5.6.0.0.3 Estimation of the sample covariance matrix** There are many techniques for volatility and correlation forecasting ranging from simple data filtering such as exponential moving averages to GARCH models, Andersen et al. (2006). Here we follow a simple approach comparable to the RiskMetrics described in Longerstaey (1996) and Andersen et al. (2006). We estimate the sample covariance matrix as follows: We estimate the conditional sample mean of the return process by:

$$\hat{\boldsymbol{\mu}}_t = \mathbb{E}\left[\mathbf{r}_t\right] = \alpha \mathbf{r}_{t-1} + (1-\alpha)\hat{\boldsymbol{\mu}}_{t-1}, \tag{2.7}$$

where $\alpha < 1$ determines the horizon of this exponential moving average. We now estimate the conditional sample covariance matrix of returns by:

$$\boldsymbol{\Sigma}_t = \delta\left(\mathbf{r}_{t-1} - \hat{\boldsymbol{\mu}}_t\right)\left(\mathbf{r}_{t-1} - \hat{\boldsymbol{\mu}}_t\right)^T + (1-\delta)\boldsymbol{\Sigma}_{t-1}, \tag{2.8}$$

again with $\delta < 1$. We now approximate the sample covariance matrix at time $t+1$ by our estimate, i.e.

$$\boldsymbol{\Sigma}_{t+1} \approx \boldsymbol{\Sigma}_t.$$

## 5.7 Return forecasts

We consider two different strategies for future return forecasts. These strategies differ in how the stock return is defined reflecting the intermediaries' underlying assumptions about the asset return process. However, all strategies take the return on loans equal to the interest rate paid on the these loans. The strategies considered can be classified as: (1) fundamentalist and (2) trend following. These strategies, or slight variations of them, have been discussed in Hommes and Wagener (2009), Hommes (2006), Farmer and Joshi (2002) and LeBaron (2012). The authors demonstrate that these stylized representations of investment strategies of real market participants can lead to rich stock market dynamics and have been selected for this purpose.

[54] In fact this assumption is not unrealistic in a financial system only populated with fundamentalist investors. In this case all noise in the return time series originates in the production noise of the individual firms which is uncorrelated.

**Fundamentalist** In this context fundamentalist investors can be understood as investors who hold long positions in stock in order to benefit from the dividend stream; income due to stock price movements is not relevant to their decision making. A real world equivalent of such an investor could be a pension fund. The fundamentalist investor defines the return of a stock $i$ as the dividend-price ratio, i.e.:

$$r_{i,t} = \frac{\Pi_{i,t}}{p_{i,t}}, \tag{2.9}$$

where $\Pi_{i,t}$ is the profit generated by firm $i$ at time $t$.

**Trend follower** Trend followers believe that past price movements will continue in the future, i.e. they believe prices have inertia. The trend follower estimates the return on a stock over a certain time horizon $\tau$. Thus we have for the return of a stock $i$:

$$r_{i,t} = \frac{p_{i,t} - p_{i,t-\tau}}{p_{i,t-\tau}}. \tag{2.10}$$

Future expected returns are estimated as in eq. 2.7. For risk neutral investors we simplify eq. 2.7 even further and take $\alpha = 1$, i.e. the risk neutral investor expects the one period ahead return to be well approximated by the current return.

## 5.8 Market mechanism

In the following we will outline a simple market clearing approach that determines stock prices and interest rates. Our approach follows closely the market mechanisms outlined in Hommes and Wagener (2009) and LeBaron (2012), however we extend it to multiple assets. In assuming market clearing we abstract from a number of potentially important features such as rationing on the commercial loan markets. Using more sophisticated price formation mechanisms, such as modelling stock markets via limit order books, would allow modelling the microstructure of financial markets and any resulting implications for systemic risk. However, the macro-perspective of this work and the analytical tractability that the market clearing approach provides in certain cases justify this approach at this stage. Later versions of this model may well use limit order books for price formation.

### 5.8.1 Stock market

In the market clearing approach we equate supply and demand and solve for the stock price. First consider a single stock. As mentioned earlier the total supply of shares of a given stock is normalised to 1. In monetary units the supply of stocks, i.e. the number of shares in the market times their price, is therefore simply given by the market clearing price. The demand, in monetary units, for stock $i$ is the sum over all intermediaries' portfolio allocations to this stock, i.e.

$$\text{Demand}_{i,t} = \sum_{j=1}^{N_b} w^s_{ji,t} \lambda_{j,t} \mathcal{E}_{j,t} = \sum_{j=1}^{N_b} w^s_{ji,t} \mathcal{A}^{ill}_{j,t},$$

where $w^s_{ji,t}$ are portfolio weights for stocks only and $\mathcal{A}^{ill}_{j,t}$ represents the amount of illiquid assets (stocks and loans) owned by intermediary $j$ at time $t$. This equation tells us that the total demand for a stock $i$ is equal to the sum of demands of all intermediaries. The demand of a given intermediary is just given by how much of his illiquid asset portfolio he wants to allocate to that particular stock. We have therefore for the stock price:

$$p_{i,t+1} = \sum_{j=1}^{N_b} w^s_{ji,t+1} \left( \mathbf{s}^T_{j,t} \mathbf{p}_{t+1} + \sum_{i=1}^{N_f} l_{ji,t} \right),$$

where we have re-expressed $\mathcal{A}^{ill}_{j,t}$ as a sum over its asset classes. Note that the intermediaries' demand depends on the price of the stock as it determines the wealth of the intermediary. We assume that the investor chooses the portfolio weights prior to the market clearing, i.e. weights are not a function of the price computed in market clearing. This approximation linearises the problem.

Expressed in matrix form we have:

$$\mathbf{p}_{t+1} = \mathbf{W}^s_{t+1} \left( \mathbf{S}_t \mathbf{p}_{t+1} + \mathbf{l}_t \right), \tag{2.11}$$

where $\mathbf{W}^s$ is the $N_f \times N_b$ matrix of stock portfolio weights, $\mathbf{S}$ is the $N_b \times N_f$ stock ownership matrix and $\mathbf{l}$ is the vector of loan investments such that $l_{j,t} = \sum_i l_{ji,t}$. Thus the market clearing vector of stock prices is given by:

$$\begin{aligned} \mathbf{p}_{t+1} &= \mathbf{X}^{-1}_{t+1} \mathbf{W}^s_{t+1} \mathbf{l}_t \\ \mathbf{X}_{t+1} &= \mathbb{1} - \mathbf{W}^s_{t+1} \mathbf{S}_t \end{aligned} \tag{2.12}$$

Note that for the stock prices to be well defined two conditions have to be satisfied: (1) There must exist a bank $i$ such that $l_{i,t} \neq 0$ and (2) the matrix $\mathbf{X}$ must be invertible. In the following we will assume that both these conditions are satisfied.

Also note that if we are assuming fixed intermediary equity we have simply:

$$\mathbf{p}_{t+1} = \mathbf{W}^s_{t+1} \lambda_{j,t} \mathcal{E}_{j,t}. \tag{2.13}$$

### 5.8.2 Loan market

We perform the loan market clearing in a similar fashion to the stock market. The credit supply is given by:

$$\text{Supply}_t = \sum_{j=1}^{N_b} w^l_{j,t} \lambda_{j,t} \mathcal{E}_{j,t} = \sum_{j=1}^{N_b} w^{loan}_{j,t} \mathcal{A}^{ill}_{j,t},$$

where $w^l_{j,t}$ is the investment into commercial loans of intermediary $j$ at time $t$.

**5.8.2.0.1 Clearing for Cobb-Douglas economy:** Recall the credit demand function of the firms in the productive sector given by Eq. 3.3 and the simplifying assumptions in section **??**. The total demand of the productive sector for loans of maturity $\tau$ and at interest rate $r$ is then given by:

$$\text{Demand}_t = D(r_t) = \sum_{i=1}^{N_f} \left[ \left( \frac{r_t}{A_{i,t} \alpha \left( 1 - (1 + r_t)^{-\tau} \right)} \right)^{\frac{1}{\alpha - 1}} - (1 - d) K_{i,t} \right].$$

The market clearing interest rate is hence given by:

$$r_{t+1} = D^{-1}\left(\sum_{j=1}^{N_b} w^l_{j,t+1}\left(\mathbf{s}^T_{j,t}\mathbf{p}_{t+1} + \sum_{i=1}^{N_f} l_{ji,t}\right)\right), \tag{2.14}$$

where $D^{-1}(\cdot)$ is the inverse aggregate credit demand function. For $\tau = 1$ we can solve analytically for $r_t$; in general however we need to solve numerically for the market clearing interest rate through a simple iterative procedure. Note that we may restrict the range of permissible interest rates such that $r_t \in [r_{min}, r_{max}]$.

**5.8.2.0.2 Clearing for Input-Output economy:** Recall the credit demand function of the firms. The total demand for credit will be given by:

$$\text{Demand}_t = D(r_t) = \sum_{i=1}^{N_f} \frac{L^i_d}{r^i_{max} - r_{min}}(r^i_{max} - r), \tag{2.15}$$

where $L^i_d$ and $r^i_{max}$ are the firm specific liquidity shortfall and the reservation interest rate respectively. $r_{min}$ is constant across firms. Note that this demand function is insensitive to different loan maturities. As above the clearing interest rate is then given by: The market clearing interest rate is hence given by:

$$r_{t+1} = D^{-1}\left(\sum_{j=1}^{N_b} w^l_{j,t+1}\left(\mathbf{s}^T_{j,t}\mathbf{p}_{t+1} + \sum_{i=1}^{N_f} l_{ji,t}\right)\right), \tag{2.16}$$

where $D^{-1}(\cdot)$ is the inverse aggregate credit demand function. We solve this equation using a simple numerical solution algorithm.

## 5.9 Value At Risk (VaR) Constraints and Procyclical Leverage

CRISIS can now demonstrate procyclical leverage dynamics as a result of Value–at–Risk (VaR) type financial constraints. In our model, financial institutions estimate their Value-at-Risk position based on historical portfolio returns. Thus the banks' perceived portfolio risk determines the VaR. In turn, the VaR constrains leverage as we will demonstrate below. In times of low perceived risk banks will try to achieve high levels of leverage. Conversely, at times of high perceived risk, banks will aim to reduce leverage.

We have introduced VaR constraints in order to (a) reflect Basel II risk regulation and (b) apply destabilising feedback to the financial sector. The VaR constraint is not enabled by default in the simulator. This feature can be switched on if desired by the researcher.

In order to derive an expression for the banks' VaR we begin by outlining an estimation strategy for the portfolio covariance matrix. Based on the covariance matrix and the banks' beliefs about the return distribution (we assume Gaussian returns for simplicity) we can then derive a simple expression for the VaR.

Given the stock price $p_{i,t}$ at time $t$ for each stock type $i$, a (vector) estimate $\Omega_t$ is made of the stock return for each possible stock investment. When the return estimate $\Omega_t$ is known, a moving mean $\Omega^*$ of this quantity is updated:

$$\begin{aligned} \Omega_t &= \begin{pmatrix} \log(\frac{p_{1,t}}{p_{1,t-1}}) \\ \log(\frac{p_{2,t}}{p_{2,t-1}}) \\ \vdots \end{pmatrix}, \\ \Omega^* &= m \cdot \Omega_t + (1-m) \cdot \Omega^*, \end{aligned}$$

where $m$ is a customisable memory parameter ($0 \leq m \leq 1$) indicating the sensitivity of the moving mean $\Omega^*$ to the most recent sample $\Omega_t$.

A covariance matrix $\Sigma_t$ of stock returns in excess of the mean return $\Omega^*$ is computed, and a moving (matrix) mean $\Sigma^*$ is updated similarly:

$$\begin{aligned} \Sigma_t &= (\Omega_t - \Omega^*)(\Omega_t - \Omega^*)^{\mathrm{T}}, \\ \Sigma^* &= m \cdot \Sigma_t + (1-m) \cdot \Sigma^*. \end{aligned}$$

Write $w_{\mathrm{L}}$ for the proportion ($0 \leq w_{\mathrm{L}} \leq 1$) of the available portfolio that a bank chooses to allocate to new loan investments. Write $w_{\mathrm{S},i}$ ($0 \leq w_{\mathrm{S},i} \leq 1$ for all $i$) for the same proportion that the bank allocates to new stock investments of type $i$. Typically $w_{\mathrm{L}} + \sum_i w_{S,i} = 1$.

The variance $\sigma_{\mathrm{S}}^2$ of stock returns is computed as:

$$\sigma_{\mathrm{S}}^2 = w_{\mathrm{S}}^{\mathrm{T}} \, \Sigma^* \, w_{\mathrm{S}}.$$

For simplicity we assume that the variance $\sigma_{\mathrm{L}}^2$ of loan returns is $\sigma_{\mathrm{L}}^2 = w_{\mathrm{L}}^2 \sigma_{\mathrm{S}}^2$. As a result the standard devision $\sigma$ of the entire bank portfolio is

$$\sigma = \sigma_{\mathrm{S}} \sqrt{1 + w_{\mathrm{L}}^2}.$$

Now that we have determined the portfolio variance $\sigma^2$, we can compute the Value-at-Risk. Assuming Gaussian beliefs about the return distribution we have for the per-dollar VaR:

$$VaR_\alpha = \sqrt{2}\sigma \, \mathrm{erf}^{-1}(2\alpha - 1), \tag{2.17}$$

where $\alpha$ is the VaR quantile. The VaR constraint implies that

$$VaR_\alpha A \leq E \quad (*) \tag{2.18}$$

where $A$ is the total size of the banks' assets and $E$ is its equity. For a bank that maximizes returns on equity $(*)$ will be binding. The bank then has the following target leverage:

$$\lambda = VaR_\alpha^{-1} = \left( \sqrt{2}\sigma \, \mathrm{erf}^{-1}(2\alpha - 1) \right)^{-1} = \frac{z}{\sigma}, \tag{2.19}$$

where we collect all terms except the standard deviation in the multiplier $z$. We calibrate the multiplier $z$ in order establish a reasonable numerical relationship between the banks leverage and the perceived portfolio variance. The implied confidence level is not necessarily the same as is typically used in bank regulation because the Crisis model may operate in regimes where the return variance is not calibrated to actual market variances.

Our simulations show that the VaR constraint, and the resulting inverse relationship between perceived portfolio risk and bank leverage, can result in cyclical behaviour in which banks increase their leverage in times of low risk. Once leverage has reached relatively high levels the system becomes susceptible to small fluctuations leading to sudden increases in perceived portfolio risk and decreases in leverage. This triggers a downward spiral in which falling prices lead to higher perceived risk and decreasing leverage, which in turn further depresses prices. At some point the market recovers and perceived risk begins to decline. This turning point initiates a new the leverage cycle.

The multiplier $z$ plays an important role in the cyclical dynamics resulting from the VaR constraint. A large value of $z$ will result in hyperaggressive leverage fluctuations with $\lambda \gg 1$. Conversely a small value of $z$ will result in low average bank leverage $\lambda \sim 0$ with low variability. In practice we compute $z$ by disabling the VaR constraint until the simulation transient has passed. In the period in which the VaR constant is not active, the simulator computes $\sigma$ and observes the existing bank leverage $\lambda$, forming a tentative multiplier $z_t = \lambda_t \sigma_t$ for each time step $t$. $z$ is subsequently selected as the median of the measurements $\{z_t\}$. This ensures that a reasonable numerical relationship exists between the bank leverage and the endogenous return variance that the simulation generates.

A schematic overview of the flows in this integrated model (without crisis resolution) is given in Fig. 2.3.

# 6 Interbank market

As before banks have strategies which aim at maximising profits from capital allocation in the credit, stock and interbank market and are subject to the balance sheet identity and some regulatory constraints. Banks set lending rates to firms and borrowing rates to other banks in the interbank market. Banks prefer to lend to firms and invest in the stock market, thus lending on the interbank market is residual.

Note that the interbank lending scheme here is the one that we are planning on implementing in the integrated model, not the one that is already implemented. In particular there are extensions to the credit market described here that allow banks to quote different interest rates to different firms, as well as different interest rates to each other. While the ability to quote different interest rates to different firms is available in the Mark I macro model, it is not available in the other two. We plan to implement and make the interbank lending model described here fully compatible with the rest of the integrated model at a later date. The currently implemented interbank facility is much simpler, and only allows banks to lend and borrow at a single interest rate.

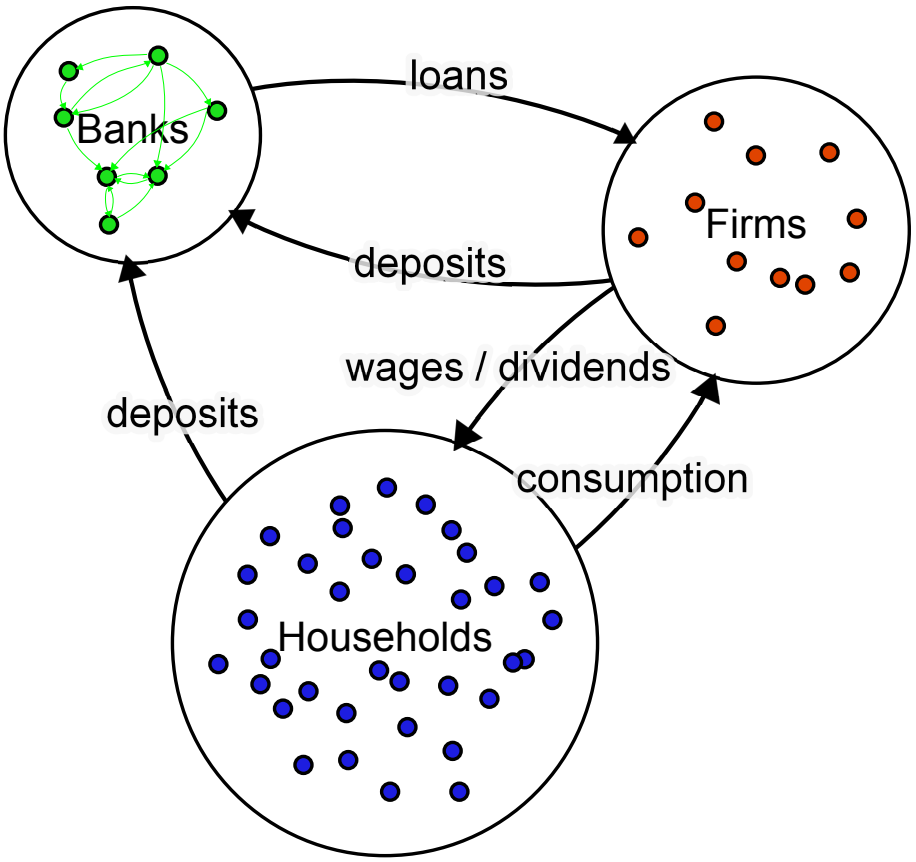



Figure 2.3: A schematic overview of the integrated model structure shows the three sets of agents – banks, firms, and households – and their interactions. Firms pay their owners dividends and their workers loans, the households consume goods produced by the firms. Households and firms deposit money with the banks, the banks grant loans to the firms.

## 6.1 Definitions

- $E_j(t)$ : bank equity
- $D_j(t)$ : bank deposits
- $C_j(t)$ : bank cash
- $L_j(t)$ : total bank liabilities
- $L_j^f(t)$ : total bank loans to firms
- $L_{i,j}^f(t)$ : bank $j$ loans to firm $i$
- $I_j^l(t)$ : bank total interbank lending
- $I_j^b(t)$ : bank total interbank borrowing (including from central bank)
- $T_f$ : maturity of loans to firms.
- $T_b$ : maturity of interbank loans.
- $r_D$ : interest rate to depositors
- $r_E$ : interest rate to banks shareholder $r_E$
- $r_f$ : the risk-free rate in the economy. This is the rate at which the central banks borrows from other banks
- $r_H$ : marginal lending rate (fixed rate at which the central bank lends to other banks)
- $r_{i,j}^f(t)$ : interest rate offered by bank $j$ to firm $i$,
- $r_{i,j}^l(t)$ : interest rate asked by bank $j$ to bank $i$,
- $r_{i,j}^b(t)$ : interest rate paid by bank $j$ to bank $i$,
- $r_j^l(t)$ and $r_j^b(t)$ : respectively the average interest rate at which bank $j$ lends and borrows in the interbank market at time $t$
- $r_j^f(t)$ : average interest rate at which bank $j$ lends to firms at time $t$
- $p_i$ is the probability of default of firm $i$.
- $\lambda_j = \frac{A_j(t)}{E_j(t)}$ : bank leverage ratio
- $w_{jk}(t)$ : bank target level of exposure to stock $k$
- $K_j(t)$ : bank target level of exposure to private sector loans

## 6.2 Sequence of Events

At the beginning of each trading day

- Banks pay interest to depositors
- Banks repay their interbank loans and loans to the Central Bank
- Banks receive dividend payments on the stocks
- Banks receive interest (and principal) back on their loans to firms (unless firm defaults)
- Banks receive interest (and principal) back on their loans to other banks (unless bank defaults)
- Banks receive interest on the cash deposited in central bank accounts
- Banks pay interest to shareholder, if appropriate, at rate $r_E$
- Bank's equity is adjusted
- Regulatory checks are performed and banks with negative equity are deemed to be in default

After checks on banks' solvability have been performed:

- Banks choose their leverage ratio $\lambda$
- Banks choose the weights of their asset portfolios (loans + stocks)
- Banks receive firms demand for loans and information about firms probability of bankruptcy $\rho$ (firms demand is determined in the MABM, in the tests the demand is random and $\rho$ exogenous and the same for all firms).
- Banks lend to firms and invest in the stock market.

After investment has taken place:

- Banks receive an exogenous shock on deposits (deposit fluctuations are determined in the MABM, in the tests the shocks are random).
- In case banks end up with negative positions on their central bank accounts after the shock, they enter the interbank market. If they cannot lend all they need on the IB market they borrow from the central bank at $r_H$

## 6.3 Balance sheets

The asset side of the bank balance sheet is given by:

$$\begin{aligned} A_j(t) &= C_j(t) + \mathbf{s}_{j,t}^T \mathbf{p}(t) + \sum_{i=1}^{N_f} L_{j,i}^f(t) + \sum_{i=1}^{N_b} I_{j,i}^l(t), \\ \mathbf{s}_j(t) &= \left(s_{j1}(t), ..., s_{jN_f}(t)\right)^T, \\ \mathbf{p}(t) &= \left(p_1(t), ..., p_{N_f}(t)\right)^T, \end{aligned}$$

where $C_j(t)$ is the cash investment, $\mathbf{s}_j(t)$ the vector of stock ownership, $\mathbf{p}(t)$ the vector of share prices, $N_f$ is the number of shares and $L_{j,i}(t)$ is the book value at time $t$ of a loan extended from bank $j$ to firm $i$, and $I^l_{j,i}(t)$ is the book value at time $t$ of a loan extended from bank $j$ to bank $i$. Intermediaries face a long only constraint, i.e. $s_{j,i}(t) \geq 0$. This implies that intermediaries cannot short an asset which they consider overpriced. This limits the extent of arbitrage that is possible in this setting. Mispricings may therefore be able to persists longer than in the case of permitted short selling.

The liability side of the bank balance sheet is given by:

$$L_j(t) = D_j(t) + \sum_{i=1}^{N_b} I^b_{j,i}(t),$$

where $D_j(t)$ is the size of deposits of bank $j$, and $I^b_{j,i}(t)$ is the book value at time $t$ of a loan extended from bank $i$ to bank $j$. By definition the equity of the intermediary is given by:

$$E_j(t) = A_j(t) - L_j(t),$$

where $L_j(t)$ generically represents all liabilities of intermediary $j$ at time $t$. We assume, for simplicity, that the intermediaries do not face funding restrictions, i.e. should the intermediary decide to increase its liabilities it always finds willing lenders.

## 6.4 Beginning of day regulatory check

At the beginning of the trading day banks pay interest to depositors, repay their interbank loans and loans to the Central Bank, receive dividend payments on the stocks, receive interest (and principal) back on their loans to firms (unless firm defaults), receive interest (and principal) back on their loans to other banks (unless bank defaults), receive interest on the cash deposited in central bank accounts. Bank's equity is adjusted as follows

$$E_j(t) = E_j(t-1) + dE_j(t)$$

where

$$dE_j(t) = C_j(t-1)r_f + \sum_{k=1}^{N_f} s_{j,k}(t-1)d_k(t) + \sum_{i=1}^{N_f} L^f_{j,i}(t)r^f_{j,i}(t-1) +$$

$$\sum_{i=1}^{N_b} I^l_{j,i}(t)r^l_{j,i}(t-1) - \sum_{i=1}^{N_b} I^b_{j,i}(t)r^l_{j,i}(t-1) - D_j(t-1)r_D.$$

If Bank equity is negative at this point the bank goes into default.

## 6.5 Credit Market

We present here a possible model for the credit market that allows for cross-sectional interest rate dispersion in both the loans and interbank market. The model allows for a predetermined connectivity network between firms and banks

and in the interbank market. The modelling choice for the interbank credit market mimics the behaviour of the e-Mid interbank market which is a transparent trading platform. Alternative mechanisms, based on over the counter trading (OTC) are of course possible.

## 6.6 Lending to firms

Once banks have determined the proportion of loans in their portfolio $w_{j,L}$ they submit loan offers to firms for a total of $\hat{L}_j^f(t) = w_{j,L}\lambda_j E_j(t)$. Banks take into account the their financing costs and the firm probability of default when determining their lending rates (the probability of firm default is an input from the MABM). We assume here that firms are equally risky (extension to heterogenous firms will be consider at a later stage). The bank to firm rate is bounded from below by the interest rate to depositors (that we assume to be higher than the risk-free rate). Additionally the reservation (lowest) price at which bank $j$ is willing to lend to firm $i$ take into account the risk of firm default and is determined by the condition

$$(1 + r^{min}ij(t))(1 - p_i) + \gamma p_i = (1 + E[r_j^b(t)]) > (1 + r_f) \quad \forall j, \tag{2.20}$$

or

$$r_{ij}^{min}(t) = \frac{1 + E[r_j^b(t)] - \gamma p_i}{1 - p_i} - 1, \tag{2.21}$$

where $p_i$ is the probability of default of firm $i$, $\gamma$ is the recovery rate in case of firm default (we currently assume that $\gamma = 0$, in Basel $\gamma = 45\%$).

$E[r_j^b(t)]$ is the rate at which bank $j$ expects to borrow in the interbank market at day $t$. $E[r_j^b(t)]$ calculated as a volume weighted average of past rates obtained in the interbank market

$$E[r_j^b(t)] = \sum_{\tau=1}^{n} r_j^b(t-\tau)\frac{I_j^b(t-\tau)}{\sum_{\tau=1}^{n} I_j^b(t-\tau)}. \tag{2.22}$$

This rule reflects the assumption that the bank lends to firms at least at the rate it expects to borrow in the interbank market in case of a negative liquidity shock. The expected rate incorporates the liquidity risk. In the above equation, if the bank has not borrowed in one of the previous period we set $r_j^b(t-\tau) = 0$ in the sum. If a bank receives shocks that are small compared to the exposure level, so that the bank does not need to borrow in the interbank market, it can afford to set lower rates to firms. As a result (the precise mechanism is explained later), such bank will also try and increase its leverage target $\lambda_j$. Vice versa a bank that has large deposit fluctuations given its level of chosen leverage, has to charge higher rates to firms. Eventually bank learn to choose $\lambda_j$ compatibly with the volatility of their deposits.

We assume that firms are price takers but have a stepwise demand function and do not accept rates higher that $r_{max}$ (linked to the CB policy rate). Note that the bank may not achieve the desired level of lending to firms (for example if $w_{j,L}$ is too high or the rates offered to firms are too high). $L_j^f(t)$ represents the actual amount of loans to firms provided by banks $j$ at time $t$.

Firms approach first banks (possibly only a subset to which the are connected) that offer them the best rate. Banks lending to firms occurs on a first come first served basis.

## 6.7 Shocks to deposits

After banks have committed to lend to firms and invest in the stock market, a shock, $dD_j(t)$, on banks' deposit occurs (this is determined by the macro model, though for testing purposes we replace this by exogenous shocks).

In case the shock on deposit on bank $j$ is such that $D_j((t) > C_j((t)$ then bank $j$ enters the interbank market and requests an interbank loan of size

$$I_j^b((t) = dD_j(t) - C_j((t) \tag{2.23}$$

On the contrary, if the shock on deposits is positive, or if it is sufficiently small that the bank is left with cash reserves, $C_j(t) > 0$, the bank will try and lend $C_j(t)$ in the interbank market.

## 6.8 The interbank market

Given the amount of loans committed by banks to firms and the bank equity (information as we mentioned is public) the probability of defaults of each banks can calculated from the probably of defaults of loans to firms (i.e. probability that losses on firm lending exceed equity). Currently we use a normal approximation to the binomial distribution of losses which assumes each loan is the same size, has the same probability of default, and the defaults are independent.

In the case of correlated defaults, we plan to use the one factor Vasicek model used to calculate the Internal Ratings-Based (IRB) approach of Basel II. To this purpose we assume that banks make loans that are all the same in size and to firms who have the same probability of default. Denote by $p^f$ the (time independent) fraction of loans to banks that may default. The cumulative distribution of the random variable $p^f$ under the Vasicek model is

$$F(p^f) = N\left(\frac{\sqrt{1-\rho}N^{-1}(p^f) - N^{-1}(p)}{\sqrt{\rho}}\right) \tag{2.24}$$

where $p$ is the unconditional probability of each loan (or firm) defaulting (we assume $p_i = p$ for all firms) and $\rho$ is the loans exposure to the systematic risk factor in the Vasicek model.

Thus the probability of a bank $i$ default is

$$p_i^b = P\left(p^f \geq \frac{E_j(t)}{L_j(t)}\right) = 1 - N\left(\frac{\sqrt{1-\rho}N^{-1}\left(\frac{E_j(t)}{L_j(t)}\right) - N^{-1}(p_i)}{\sqrt{\rho}}\right) \tag{2.25}$$

The assumption in the model is that the probability of default of each bank is common knowledge in the market. The implicit assumption is that a credit rating agency provides this information to the market, or CDS spreads are available for any potential counterpart. Various weakened and more realistic versions could be considered. For example banks may know other banks probability of default with some uncertainty, or may be looking at the historical probability of defaults.

Once the shock to deposits has been realised, lending on the interbank market does not bear liquidity risk and banks are willing to lend all their excess cash, as long as they can make an expected profit (accounting for the risk of

default of borrowers) at least comparable to leave money in central banks deposits. This is a strong assumption that limit the possibility of strategic liquid it hoarding of liquidity. In the future we can generalise to deposit shocks and demand for loans happening at random times during the day so that banks have to take their interbank lending decision while facing this uncertainty. Initially, for simplicity, interbank lending is left until the end of the business day.

Lenders reservation prices are determined by the conditions

$$(1 + r^l_{i,j}((t))(1 - p^b_i) + \gamma p^b_i = (1 + r_f) \quad \forall i, j, \tag{2.26}$$

Lender $i$ accepts any offer from a borrower $j$ as long as its reservation rice is lower that the rate offered by the borrower:

$$r^l_{i,j}((t) = \frac{(1 + r_f - \gamma p^b_j)}{1 - p^b_j} - 1 < r^b_j((t)) \tag{2.27}$$

Borrowers compete for liquidity and they post bids at the rate

$$r^b_j(t) = \frac{(1 + r_f - \gamma p^b_j)}{1 - p^b_j} - 1 + \Delta r_j((t) \tag{2.28}$$

Borrowers try and learn which is the lowest spread they can offer in order to satisfy their demand for loans.

We assume that lending is restricted to an exogenous pre-determined network (to model market frictions). This is currently done via a fixed (for each simulation) Erdos-Reyni random graph. This allows us to go from the extreme of an anonymous fully connected market (i.e. $p = 1$) to a very weakly connected market.

Borrowers post their bids and lenders, in random order: observe their neighbours borrowers bids, compute the profitability of each bid as

$$\pi_{i,j}(t) = [r^b_j(t) - r^l_{i,j}(t)] = \Delta r_j(t), \tag{2.29}$$

rank them and fill them in order of profitability.

## 6.9 Learning

### Bank rates to firms

The lending rate to firm is updated so to try and achieve the target level $\hat{L}^f_j(t)$. If the bank has not been able to lend all the desired level to firms it reduces the rate following the rule

$$r^f_{i,j}(t) = \min\left\{r^{max}, \max\left\{r_D, r^{min}_{i,j}(t-1), r^f_{i,j}(t-1)\left[1 + \epsilon^f \frac{L^f_j(t-1) - \hat{L}_j(t-1)}{\hat{L}_j(t-1)}\right]\right\}\right\} \tag{2.30}$$

where $\epsilon_r > 0$. If the bank has been able to lend all the desired level to firms, with probability $p$ it increases the rate to

$$r^f_{i,j}(t) = \min\left\{r^{max}, r^f_{i,j}(t-1)\left[1 + \epsilon^f\right]\right\} \tag{2.31}$$

## 6.10 Interbank market spreads

Spreads on interbank market are updated following a similar rule as to bank rates to firms. Borrowers calculate how much they have managed to borrow in the previous period (without recurring to CB), $\tilde{I}_j^b(t-1)$ and compare it to their liquidity demand $I_j^b(t-1)$ and adjust the rate as follows

$$\Delta r_j^b(t) = \Delta r_j^b(t-1)\left[1+\epsilon^b \frac{I_j^b(t-1)-\tilde{I}_j^b(t-1)}{I_j^b(t-1)}\right] \tag{2.32}$$

If the bank has been able to borrow all the desired liquidity, with probability $p$ it decreases its interbank borrowing rate to

$$\Delta r_j^b(t) = \max\left\{0, \Delta r_j^b(t-1)\left[1-\epsilon^b\right]\right\} \tag{2.33}$$

## 6.11 Bank leverage target

After $P$ steps the banks revise their leverage target in the following way. Different updating strategies are possible. Here we illustrate a possible one. They calculate the average profit over the $P/2$ previous steps (to allow time to learn the rate) as follow:

$$\Pi(\lambda_j) = \frac{2}{P}\sum_{\tau=1}^{P/2}[\sum_{k=1}^{N_f} s_{j,k}(t-\tau)r_k(t-\tau) + \sum_{k=1}^{N_f} L_{j,i}^f(t-\tau) * r_{j,i}^f(t-\tau) -$$

$$\sum_{k=1}^{N_b} I_{j,i}^l(t-\tau) * r_{j,i}^l(t-\tau) - \sum_{k=1}^{N_b} I_{j,i}^b(t-\tau) * r_{j,i}^b(t-\tau)]$$

Each banks they select a new value for $\lambda_j$ (from a discrete possible set of values) with probability

$$p(\lambda_j) = \frac{e^{\beta\Pi(\lambda_j)}}{\sum_i e^{\beta\Pi(\lambda_i)}} \tag{2.34}$$

This rule may possibly lead to unrealistic large strategy jumps which we may control by adding an inertia term.

Overall the strategy optimisation is against fixed historical data, and does not incorporate any strategic consideration regarding how other banks would react. Further development in this direction will be considered.

# Bibliography


Adrian, T. and N. Boyarchenko (2012). Intermediary Leverage Cycles and Financial Stability.

Adrian, T. and H. S. Shin (2010). Liquidity and leverage. *Journal of Financial Intermediation 19*(3), 418–437.

Andersen, T. G., T. Bollerslev, P. F. Christoffersen, and F. X. Diebold (2006). Volatility and correlation forecasting. In G. Elliott, C. Granger, and A. Timmermann (Eds.), *Handbook of Economic Forecasting*, Volume 1, Chapter 15, pp. 777–878. Elsevier.

Bryant, R. C., P. Hooper, and C. L. Mann (1993). *Evaluating Policy Regimes: New Research in Empirical Macroeconomics*. Brookings Institution.

Caccioli, F., J. Farmer, N. Foti, and D. Rockmore (2013). How interbank lending amplifies overlapping portfolio contagion: A case study of the Austrian banking network.

Campbell, J. and L. Viceira (2002). *Strategic asset allocation: portfolio choice for long-term investors* (Clarendon Lectures on Economics ed.). Oxford: Oxford University Press.

Cont, R. (2001). Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 37–41.

Corsi, F., S. Marmi, and F. Lillo (2013). When Micro Prudence Increases Macro Risk: The Destabilizing Effects of Financial Innovation, Leverage, and Diversification.

Dawid, H., S. Gemkow, and P. Harting (2011). The Eurace@ Unibi Model: An Agent-Based Macroeconomic Model for Economic Policy Analysis.

Dosi, G., G. Fagiolo, and A. Roventini (2010). Schumpeter meeting Keynes: A policy-friendly model of endogenous growth and business cycles. *Journal of Economic Dynamics and Control 34*, 1748–1767.

Farmer, J. and S. Joshi (2002). The price dynamics of common trading strategies. *Journal of Economic Behavior & Organization 49*, 149–171.

Geanakoplos, J. (2009). The leverage cycle.

Hommes, C. and F. Wagener (2009). Complex Evolutionary Systems in Behavioral Finance. In T. Hens and K. Schenk-Hoppe (Eds.), *Handbook of Financial Markets: Dynamics and Evolution*, Chapter 4, pp. 217 – 276. San Diego: North-Holland.

Hommes, C. H. (2006). Heterogeneous Agent Models in Economics and Finance. In K. L. Judd, Leigh Tesfatsion (Ed.), *Handbook of Computational Economics*, Chapter 23, pp. 1110–1146. Elsevier.

Jones, E., T. Oliphant, P. Peterson, et al. (2001). SciPy: Open source scientific tools for Python.

LeBaron, B. (2012). Heterogeneous gain learning and the dynamics of asset prices. *Journal of Economic Behavior & Organization 83*(3), 424–445.

Longerstaey, J. (1996). Riskmetrics: technical document. Technical report, J.P. Morgan.

Orphanides, A. (2003, July). Historical monetary policy analysis and the taylor rule. *Journal of Monetary Economics 50*(5), 983–1022.

Shin, H. (2010). *Risk and liquidity* (Clarendon Lectures in Finance ed.). Oxford: Oxford University Press.

Shin, H. (2012). Global Banking Glut and Loan Risk Premium. *IMF Economic Review 60*, 155–192.

Taylor, J. B. (1999, October). *Monetary Policy Rules*. Number tayl99-1 in NBER Books. National Bureau of Economic Research, Inc.

# Chapter 3

# Planned Future Direction

# 1 Introduction

In this document we outline a draft proposal for the development of the CRISIS framework stimulated by our meetings and discussions with the Bank of England. It incorporates inputs from the following internal CRISIS documents:

- Strawman Specs (Ross, Milan, Olaf, Doyne) **?**
- Specification for CRISIS Development (Ross, Doyne) **?**
- BoE Action Plan (Christoph, David, Doyne) **?**
- Shadow banking proposal (Gerald) **?**
- Bank strategy proposal (Christoph) **?**
- Proposal for securitized lending to households (David, Olaf) **?**
- Integration Report.

The primary goal of these changes is to allow our model to capture a more complete and accurate description of the (UK) economy based on the feedback we got on the current model (see **?**).

## 1.1 Summary of Proposed Changes

In the following we organize changes into three categories, corresponding to which of the three basic types of agents are most affected: households, firms, and financial agents (i.e. banks, funds and shadow banks). There are also structural changes corresponding to heterogeneity in risk and timescales. Figure 3.1 illustrates the money flows and interactions in the new scheme.

1. ***Changes primarily affecting households***.
   (a) *Housing market.* We introduce home ownership so that the household decision involves how much to save, consume, or spend on housing. Banks lend to households to allow them to purchase housing.
   (b) *Ownership of the economy.* Households will be the ultimate owners of both firms and banks. This ownership is mediated through funds. Households split their savings between bank deposits and mutual funds. Each household is initially randomly matched to a fund, but afterward can change funds based on performance. Dividends and loan payments are passed back automatically through the mutual funds to the households owning their shares. Households can redeem their mutual fund investments to meet their consumption needs.
   (c) *Life cycle.* We will allow for the possibility that households have ages. Households can die and new households are injected, and decision making may depend on the age of the household.

   ***Changes primarily affecting the financial sector***

2. (a) *Stock trading* will be primarily done by funds. It can also be done by banks, but we will be able to control the fraction of stocks held in a bank's portfolio, including the possibility that the fraction is zero.

(b) *Mutual funds* trade shares in stocks and also issue loans (which should be thought of as bonds) to banks. Each household will be associated with a mutual fund; households put their savings into mutual funds and can remove them if they need them, e.g. to buy a house.

(c) *Pension funds*

(d) *Banks borrow* from funds. This is modeled as a bond. New bonds can be issued and existing bonds can be sold.

(e) *Risk weighted capital constraints* as done in Basel III will be introduced in order to make risk management more realistic.

(f) *Banks' portfolio allocation decisions* will be handled via an improved algorithm.

(g) *Liquidity constraints* will be introduced as well as the current solvency constraints.

(h) *Interbank market.* A project has been underway for some time to add an interbank market. That should reach completion soon. Interbank trading will focus on overnight lending and interactions with the central bank.

(i) *Secondary market for loans.* One of the open issues we have struggled with is whether to implement a secondary market for loans. The argument against this is that it involves pricing loans with a continuous range of maturities and variable risks; the arguments in favor are that otherwise both funds and banks may be constrained from liquidating their assets. This is particularly problematic for banks as it makes it very difficult for them to deleverage (and we may miss important contagion effects if we cannot do this). We are deferring a decision on this point while seeking more advice.

3. ***Changes primarily affecting firms.***
The decision making rules for firms will act over a longer horizon and will change more slowly, corresponding to longer period optimization.

4. ***Heterogeneity***

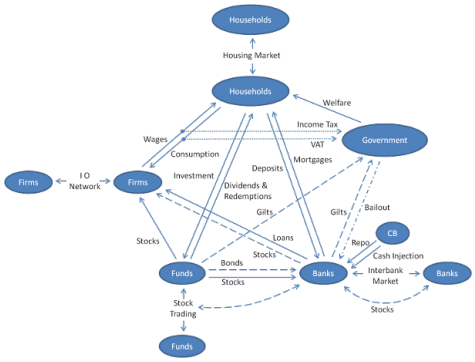
(a) *Heterogeneous risks.* We will allow both firms and banks idiosyncratic risks yielding heterogenous interest rates.

(b) *Heterogenous timescales.* We will create *realistic timescales for contracts*, both in terms of their maturity and repayment schemes. We will do this by matching the profile of loans actually given, with the goal of obtaining an accurate representation of *maturity transformation.* Furthermore, the horizons of agents' decision-making will be adjusted based on the typical timescales for agents to act.

(c) *Heterogeneous market clearing.* We will use a method developed by our collaborator Gerald Gurtner **?** as way to achieve heterogenous interest rate pricing, and clear the bond market and the stock market simultaneously once a day.

5. ***Scale and the use of heterogeneous representative agents.***
Due to computational limitations agents will be modeled on different scales with varying degrees of heterogeneity. In particular:

Figure 3.1: The basic elements of CRISIS 3.0. Arrows illustrate the direction of money flow in the new scheme to pay for goods, services and financial assets. For contracts such as bonds, where money initially flows in one direction and then returns, the arrow indicates the initial flow of money. The dashed arrow indicates an optional flow that can be switched on and off.




(a) *Financial agents.* Given that there are less than 20 important banks in the UK and the number of funds is relatively small, we should be able to model them at one-to-one scale.

(b) *Households.* There are roughly 20 million households in the UK and so it is computationally infeasible to model them all. Since the number of households we can reasonably simulate is in the range of $1,000-10,000$, we will therefore plan on having each simulated household represent $2,000-20,000$ real households. These households will be heterogeneous, in that they will correspond to households with different wealth, income and age (and in the future possibly other variables).

(c) *Firms.* There are the order of three million firms in the UK. However most of these are small, so we propose to model a few hundred of the largest firms on one-to-one scale and then model the remaining firms in terms of representative agents. The precise breakdown remains to be determined after gathering more data. The number of firms must also be chosen in concert with the number of goods as in some cases we want to allow multiple firms per good.

Figure 3.1 illustrates the money flows and interactions in the new scheme.

## 1.2 Plumbing vs. decisions

When reading this document it is sometimes useful to distinguish *Plumbing* vs. *Decisions*:

***Plumbing*** refers to features of the framework that are primarily structural and do not depend on behavioral assumptions. The plumbing is intended to relatively static and not expected to change significantly over timescales of six

months to a year. Examples include the introduction of new markets, new types of agents and new types of contracts that can be held on the balance sheets. Plumbing also encompasses necessary changes to introduce heterogeneous timescales into the framework. For example, we want contracts such as bonds to have multiple maturities and variable repayment schemes.

***Decisions*** on the other hand are encapsulated plug-ins allowing us to specify and vary the behaviour of agents in the model. This includes the behavioural rules of agents and typical timing of their decisions. Such changes are often experimental, and determining the most appropriate decision rules typically involves open research questions, and we plan to be changing these on a regular basis. The decision rules given here should we regarded as default examples, which at this stage are just plausible choices to allow us to make simulations and test the models. Refinement of the decision rules will be one of the main activities during the remainder of the project.

While we do not spell these out, the difference should be clear and it is useful to bear in mind that the decision rules are in general quite malleable.

We now give further details on the proposed changes.

# 2 Changes primarily affecting households

There are roughly 20 million households in the UK and it is computationally infeasible to model them all. Since the number of households we can reasonably simulate is in the range of $1,000 - 10,000$ each simulated household represents $2,000 - 20,000$ real households. Note that each of the households in our model are heterogeneous in that, at a minimum, have different levels of wealth and income.[1]

In previous versions of the CRISIS model household behavior was incredibly simple: each household devoted a constant fraction of its wealth to consumption expenditures and saved the remaining fraction as bank deposits. The most recent version of the CRISIS extends the household behavior along a number of dimensions.

In addition to bank deposits, households now have access to two additional savings vehicles: housing and mutual funds. In most developed countries a house is the most valuable asset of the majority of households. Empirically, secured lending to households accounts for between 2/3 and 3/4 of all bank lending in the UK. While households are the ultimate owners of both firms and banks, this ownership is mediated through mutual funds. Our primary justification for including mutual funds is institutional realism: households commonly own shares in firms and banks indirectly via mututal funds and mututal funds are an important source of medium to long term financing for banks.

In what follows, we sketch a basic framework for introducing mutual funds as well as secured lending to households. Our model of secured lending is inspired by **???**.[2] The primary research focus of our stylised model is on the role of loan-to-value (LTV) and income-to-value (ITV) constraints in generating housing price cycles in a model with substantial household heterogeneity. Thoughout we assume a fixed supply of housing and a fixed population of households.[3]

[1] Households will also likely have different behavioral parameters.

[2] The gory details of the model are discussed in **?**.

[3] An extended version of our model will incorporate a construction sector which will allow

## 2.1 Household preferences

We begin by supposing that household $i$ wishes to choose levels of borrowing, $b_{i,t}$, deposits, $d_{i,t}$, aggregate non-durable consumption, $C_{i,t}$, housing, $h_{i,t}$, and mutual funds, $m_{i,t}$ in order to maximize

$$V_t = E_t \left\{ \sum_{s=0}^{\infty} \beta_i^s \min \left( \omega_i C_{i,t+s}, (1-\omega_i) h_{i,t+s} \right) \right\} \tag{3.1}$$

where $0 < \omega_i < 1$ is a utility weight for consumption relative to ownership of housing stock, and $0 < \beta_i < 1$ is the household discount factor and the expecations operator $E_t$ is a subjective expectations operator.[4] Motivated by empirical work by John Muellbauer, we assume that household expectations of future variables are formed using either adaptive or extrapolative rules. Our households do not have rational expectations.

Note that our approach to housing assumes that housing is a fluid good, so that a given"household" can purchase a little more or sell a little housing at any given point in time. To understand why this makes sense recall that our households actually represent a bin with thousands of individual households who have similar values of income, wealth and possibly other characteristics (such as age). Thus we expect that at any given time a few houses may be buying or selling, and the overall amount of "housing" contained in the bin may change incrementally on a day by day basis.

## 2.2 Household constraints

Household $i$ faces a number of constraints that substantially restricts the levels of borrowing (i.e., mortgage debt), deposits, and housing that it can acquire. Consider the balance sheet of household $i$ at the start of date $t$. From the house-

| Assets | Liabilities |
|---|---|
| $q_{h,t} h_{i,t-1}$ | $R_{t-1} b_{i,t-1}$ |
| $q_{m,t} m_{i,t-1}$ | |
| $R_{f,t-1} d_{i,t-1}$ | Equity |

hold balance sheet we have three sources of funds: housing wealth, $q_{h,t} h_{i,t-1}$, mutual fund wealth, $q_{m,t} m_{i,t-1}$, and bank deposits, $R_{f,t-1} d_{i,t-1}$. Household

for variable housing supply and population. Our extended model will allow us to study credit refinancing cycles and first time home owner transitions induced by age dependent consumption and generation-driven turnover of assets.

[4] The upshot of our assumption that aggregate consumption goods and housing are perfect complements is that households will prefer to hold consumption goods and housing in fixed proportions. Recall that CRISIS households have Dixit-Stiglitz preferences over individual consumption goods and that the decision of how best to allocate resources across the various consumption goods can be treated separately from the housing decision discussed here.

$$C_i(h_t) = \left( \frac{1-\omega_i}{\omega_i} \right) h_{i,t}$$

$i$ also has two additional sources of funds: its nominal income $y_{i,t}$, and new borrowing $b_{i,t}$. Together total household available resources (i.e., "sources" of funds) are as follows.[5]

$$y_{i,t} + q_{h,t}h_{i,t-1} + q_{m,t}m_{i,t-1} + R_{f,t-1}d_{i,t-1} + b_{i,t} \tag{3.2}$$

Total household expenditures consist of new housing purchases $q_{h,t}h_{i,t}$, repayment of accumulated debt of $R_{t-1}b_{i,t-1}$, the purchase of consumption goods, $P_tC_{i,t}$, the purchase of additional mutual fund shares, $q_{m,t}m_{i,t}$, and any desired residual deposits $d_{i,t}$. The following summarizes total household expenditures (i.e., "uses" of funds).

$$P_tC_{i,t} + q_{h,t}h_{i,t} + q_{m,t}m_{i,t} + R_{t-1}b_{i,t-1} + d_{i,t} \tag{3.3}$$

A household's flow-of-funds constraint simply requires that total household expenditures in a given period equal a household's total available resources.

$$\begin{aligned}\text{Uses} \equiv & P_tC_{i,t} + q_{h,t}h_{i,t} + q_{m,t}m_{i,t} + R_{t-1}b_{i,t-1} + d_{i,t} = \\ & y_{i,t} + q_{h,t}h_{i,t-1} + q_{m,t}m_{i,t-1} + R_{f,t-1}d_{i,t-1} + b_{i,t} \equiv \text{Sources}\end{aligned} \tag{3.4}$$

In addition to the flow of funds constraint, we assume that household $i$ faces both loan-to-value (LTV) and income-to-value (ITV) constraints which further restrict its borrowing and housing purchases.[6] We implement an LTV constraint similar to the one used by **?** and **?**. Specifically, we assume that a bank will take care to insure that the total value of the loan (i.e., principle and interest) to household $i$ is less than some fraction of the market value of its stock of durable housing at end of date $t$. Note that our formulation allows the asset being purchased to function as the collateral for the loan.

$$R_tb_{i,t} \leq (1 - \theta_{i,t})q_th_{i,t} \tag{3.5}$$

The parameter $0 < \theta_{i,t} < 1$ denotes the 'haircut" which in principle can vary across both households and time. Similarly, our implementation of the income-to-value (ITV) constraint assumes that the lender will take care to insure that the value of housing purchased by household $i$ does not exceed some multiple $\mu_{i,t} > 1$ of its income.

$$q_th_{i,t} \leq \mu_{i,t}y_{i,t} \tag{3.6}$$

The ability to analyze the endogenous interaction between the LTV and ITV constraints in a model with substantial household heterogeneity is a key feature of our model.

An additional feature of housing stock, one that differentiates it from deposits and mutual funds, is that housing stock is a relatively illiquid asset. Whilst a household can always demand its deposits and sell shares in mutual

[5] The way that I have currently specified the decision problem assumes that mutual fund shares, $m_{i,t}$, is a choice variable for the household. An alternative specification would to assume that the quantity of mutual fund shares is effectively fixed in which case total household resources are

$$y_{i,t} + q_{h,t}h_{i,t-1} + \Pi_{i,t} + R_{f,t-1}d_{i,t-1} + b_{i,t}$$

where $\Pi_{i,t}$ represents the value of mutual funds shares owned by household $i$.

[6] Empirical relevance of both LTV and ITV constraints is documented by the BoE data from Katy's talk. Need more formal citation.

funds it is decidedly more difficult to sell a house. To capture the relative illiquidity of money we follow **?** and make a reduced form assumption that a household can only sell a fraction $0 < \phi < 1$ of its housing stock in any given period.

$$h_{i,t} \geq \phi h_{i,t-1} \tag{3.7}$$

We assume that $\phi$ is a constant. The overall liquidity of the housing market does not vary across time or households.

Finally, we assume that household $i$ faces non-negativity constraints on borrowing (i.e. we do not allow households to lend) and mutual fund holdings (i.e., we do not all households to short mutual funds); and positive lower bounds on both deposits and housing stock. Note that the assumed lower bound on housing stock implies a lower bound on consumption as well.

$$b_{i,t} \geq 0 \tag{3.8}$$

$$C_{i,t} \geq \left(\frac{1-\omega_1}{\omega_i}\right)\underline{h} \tag{3.9}$$

$$d_{i,t} \geq \underline{d} \tag{3.10}$$

$$h_{i,t} \geq \underline{h} \tag{3.11}$$

## 2.3 Household decision rules

Our model of household, though stylised, is capable of generating rich behavioral dynamics. Key state variables for households in our model are income and net worth (i.e., equity). In this section, in order to give some intuition on the type of behavioral rules being used by households in the model, we discuss four behavioral regimes that the Bank of England mentioned as being particularly relevant given their data: loan-to-value (LTV) constrained household, income-to-value (ITV) constrained households, renters, and cash-buyers.[7]

### 2.3.1 Loan-to-value (LTV) constrained households

Loan-to-value (LTV) constrained households are leveraged home-owners. In fact, LTV constrained households are sufficiently leveraged that they will have upward sloping demand curves for housing![8] With upward sloping demand curves, LTV constrained households will want to expand their holdings of housing as much as possible meaning that they will choose to run down their deposit and mutual fund accounts.

Formally, LTV constrained households use the following decision rules for

[7] In general, households will coexist in each of these behavioral regimes simultaneously.

[8] In her presentation, Katy specifically mentioned that they can observe these upward sloping demand curves in the data.

choosing borrowing, consumption, deposits, housing, mutual fund holdings:

$$b_i(n_{i,t-1}, y_{i,t}) = \left(\frac{1-\theta_{i,t}}{R_t}\right) q_{h,t} h_i(n_{i,t-1}, y_{i,t}) \tag{3.12}$$

$$C_i(n_{i,t-1}, y_{i,t}) = \left(\frac{1-\omega_i}{\omega_i}\right) h_i(n_{i,t-1}, y_{i,t}) \tag{3.13}$$

$$d_i(n_{i,t-1}, y_{i,t}) = \underline{d} \tag{3.14}$$

$$h_i(n_{i,t-1}, y_{i,t}) = \left(\frac{1}{P_t\left(\frac{1-\omega_i}{\omega_i}\right) + \left(\frac{(R_t-1)+\theta_{i,t}}{R_t}\right) q_{h,t}}\right) \left[y_{i,t} + n_{i,t-1} - \underline{d}\right] \tag{3.15}$$

$$m_i(n_{i,t-1}, y_{i,t}) = 0 \tag{3.16}$$

where $n_{i,t-1}$ denotes the net worth (i.e, equity) of household $i$.

$$n_{i,t-1} = q_{h,t} h_{i,t-1} + R_{f,t-1} d_{i,t-1} + q_{m,t} m_{i,t-1} - R_{t-1} b_{i,t-1} \tag{3.17}$$

### 2.3.2 ITV constrained households

Income-to-value (LTV) constrained households are also leveraged home-owners. However, unlike LTV constrained households, ITV constrained households do not have upward sloping demand curves for housing.[9]

Formally, ITV constrained households use the following decision rules for choosing borrowing, consumption, deposits, housing, mutual fund holdings:

$$b_i(n_{i,t-1}, y_{i,t}) = \left(P_t\left(\frac{1-\omega_i}{\omega_i}\right) + q_{h,t}\right)(n_{i,t-1}, y_{i,t}) - \left[y_{i,t} + n_{i,t-1} - \underline{d}\right] \tag{3.18}$$

$$C_i(n_{i,t-1}, y_{i,t}) = \left(\frac{1-\omega_i}{\omega_i}\right) h_i(n_{i,t-1}, y_{i,t}) \tag{3.19}$$

$$d_i(n_{i,t-1}, y_{i,t}) = \underline{d} \tag{3.20}$$

$$h_i(n_{i,t-1}, y_{i,t}) = \frac{\mu_{i,t} y_{i,t}}{q_{h,t}} \tag{3.21}$$

$$m_i(n_{i,t-1}, y_{i,t}) = 0 \tag{3.22}$$

Note that here we are assuming, for ease of exposition, that ITV households choose to hold minimal deposits and no mutual funds shares.[10]

### 2.3.3 "Renters"

In our model we define "renters" to be those households who choose to purchase the minimum amount of housing stock and build up stocks of either deposits or mutual funds in anticipation of entering the housing market more aggressively in the future.

[9] Again, Katy's presentation said that they can observe ITV constrained households in the data and that the data support a textbook downward sloping demand curve.

[10] With a downward sloping demand curve it is possible that ITV constrained households might also choose to save by holding significant deposits and/or mutual funds (it all depends on behavioral parameters!).

Formally, "renters" use the following decision rules for choosing borrowing, consumption, deposits, housing, mutual fund holdings:

$$b_i(n_{i,t-1}, y_{i,t}) = 0 \tag{3.23}$$

$$C_i(n_{i,t-1}, y_{i,t}) = \left(\frac{1-\omega_i}{\omega_i}\right)\underline{h} \tag{3.24}$$

$$d_i(n_{i,t-1}, y_{i,t}) = \underline{d} \tag{3.25}$$

$$h_i(n_{i,t-1}, y_{i,t}) = \underline{h} \tag{3.26}$$

$$m_i(n_{i,t-1}, y_{i,t}) = \frac{1}{q_{m,t}}\left[y_{i,t} + n_{i,t-1} - \underline{d}\right] - \left(\frac{P_t}{q_{m,t}}\left(\frac{1-\omega_i}{\omega_i}\right) + \frac{q_{h,t}}{q_{m,t}}\right)\underline{h} \tag{3.27}$$

Note that here we are assuming, for ease of exposition, that "renters" choose to hold minimal deposits and significant shares in the mutual funds.[11]

### 2.3.4 "Cash buyers"

In our model "cash buyers" are households who purchase substantial housing stock using only deposits and selling shares in mutual funds. By definition "cash buyers" do not borrow funds from banks to purchase housing.

Formally, "cash buyers" use the following decision rules for choosing borrowing, consumption, deposits, housing, mutual fund holdings:

$$b_i(n_{i,t-1}, y_{i,t}) = 0 \tag{3.28}$$

$$C_i(n_{i,t-1}, y_{i,t}) = \left(\frac{1-\omega_i}{\omega_i}\right) h_i(n_{i,t-1}, y_{i,t}) \tag{3.29}$$

$$d_i(n_{i,t-1}, y_{i,t}) = \underline{d} \tag{3.30}$$

$$h_i(n_{i,t-1}, y_{i,t}) = \left(\frac{1}{P_t\left(\frac{1-\omega_i}{\omega_i}\right) + q_{h,t}}\right)\left[y_{i,t} + n_{i,t-1} - \underline{d}\right] \tag{3.31}$$

$$m_i(n_{i,t-1}, y_{i,t}) = 0 \tag{3.32}$$

## 2.4 Markets

The addition of housing and mutual fund decisions to the CRISIS model introduces three new markets whose structure needs to be specified: the mortgage market, the housing market, and the market for mutual fund shares.

Where possible we try to implement some form of market clearing. In the housing market for example, where we assume fixed aggregate supply of housing (i.e., no construction sector), the market price for housing equates aggregate demand across households with aggregate supply. For the purposes of clearing the housing market we take the interest rate on mortgages as fixed at a temporary value. Specifically, a bank chooses the interest rate offered on home mortgages to be a referenced to five year commercial lending rates. We will do further research to understand what the right markup is.

[11] An alternative formulation would have "renters" choose to hold significant deposits and no mutual fund shares. I do not think that "renters" will choose to hold both deposits and mutual funds simultaneously.

## 2.5 Extension: Household life-cycle model

In our current model we do not keep track of the age of households and there is no life-cycle: households are effectively immortal. This is fine for some purposes, but in discussions at the Bank of England it became clear that they believe that the life-cycle is an important part of the dynamics of housing markets. There are also other situations where life-cycle dynamics may be important and so we will add a version of the household model that includes a life cycle. This will be an optional plug-in, to be used depending on whether or not it is needed.

To implement a life cycle we will first initialize households to match observed demography. This will include the age of the household, but also income, wealth and employment. The life cycle will be implemented as follows:

- *Age.* The age of each household will be incremented once a year.
- *Death.* A fraction of older households will be removed every year, to simulate the death of the head of household. The fraction eliminated and the ages of the houses that are eliminated will be chosen to match demographic statistics. On death the house will be sold and its assets will be inherited.
- *Household creation.* New households will be created to correspond roughly to children leaving and creating their own households.
- *Retirement.* At retirement age a household will retire, in which case it becomes unemployed and draws from its pension fund.
- *Inheritance.* As a simple approximation, as households die we can create new households inheriting the assets of the old households. However this is certainly a poor approximation, as the descendants receiving inheritance are typically of a much older age than those starting a household. A better approximation will be to randomly choose household to receive inheritance by matching to the ages that receive inheritance according to demographic data.

Olaf has proposed an alternative life cycle model presented in Du and Kamakura **?**. This model was calibrated for the US using data from the census. However, this model is quite complicated, involving 13 different stages of life cycle development, and treating states such as divorce. While it would be more accurate than the crude model given above, I suggest that we defer it to the future.

## 2.6 Household bankruptcy

Households have no regulatory constraints on solvency. However, there can be liquidity situations that may force a household to default on loans. Household bankruptcy handling is limited to unwinding loans and liquidating collateral. For simplicity we assume houses can be liquidated immediately and consumption/housing can be scaled down to zero.

# 3 Changes primarily affecting the financial sector

## 3.1 Funds

### 3.1.1 Structure of funds

At the beginning of the simulation households will be assigned fund shares. (The assignments of fund shares and housing will be set in order to match demography). The total number of shares will be fixed to be one, so that household $i$'s share at time $t$ is a simple fraction $m_{it}$, where $\sum_i m_{it} = 1$. Mutual funds will not automatically pay dividends; however, when a firm pays dividends the mutual funds will pass these through to their clients on a pro-rata basis, i.e. proportional to the number of shares for each household who is their client.

Funds will be unleveraged and will use trading rules to attempt to maximize their returns. (In the future we can easily extend this to allow hedge funds that can use leverage, but for now we will not do this). Funds are long only investors whose assets consist of stocks, bonds and cash.

Funds can be either trend followers or fundamentalists with the respective standard forms for expectation formation. The presence of trend following investment strategies and endogenous switching between strategies can then lead to booms and busts on the stock market.

### 3.1.2 Valuation and liquidation of shares

At any point in time the value of a share can be determined by the aggregate value of the assets. This can be done using mark to market accounting based on the last traded share price. For bonds this will be done by looking at a corresponding bond with a similar maturity and risk weighting. (In practice the maturity of a bond may involve fractions of a year; we will round this as needed when bonds are traded). If a household requests a redemption the fund will pay it out of cash if there is sufficient cash available, and if not will sell shares of its stocks or bonds as required in order to raise the necessary capital. On redemptions or contributions the fractional share of each shareholder will be adjusted accordingly. (If we do not have a secondary market then there are potentially problematic situations in which a fund cannot fully redeem shares due to the lack of liquidity of its bonds).

### 3.1.3 Portfolio allocation

In principle we could use an off the shelf algorithm for the fund's portfolio problem such as mean-variance. Unfortunately due to the long only constraint the optimisation problem would have to be solved numerically making it an unattractive choice. Instead we suggest a heuristic allocation, where the demand function for a particular asset $i$ is given by:

$$S(\hat{r}_i) = (1 - w_c)\frac{e^{\beta \hat{r}_i}}{\sum_j e^{\beta \hat{r}_j}} A', \tag{3.33}$$

where $\hat{r}_i$ is defined either as the expected return on asset $i$ or the corresponding Sharpe ratio where $\hat{r} \to \hat{r}/\sigma$. $A'$ is size of the balance sheet that can be reallo-

cated. $w_c$ is the fund's cash reserve weight. Given this heuristic rule the fund reallocates the assets that are not locked up due to longer maturities. It reserves a certain fraction of its assets in cash in order to be able to make dividend payments and redeem shares from its investors. The logit investment allocation will favour assets with higher returns/Sharpe ratio, doing so more aggressively for larger $\beta$. In fact, $\beta$ can be calibrated such that the portfolio weights resulting from the logit rule and Sharpe ratio input resemble the qualitative behaviour of the weights obtained from mean variance portfolio allocation.

In practice the portfolio of funds will consist of stocks, bonds and cash. Since both banks and funds reconsider their portfolio decision on a daily basis the new bonds issued at any given time may be relatively small in value as they reflect the incremental adjustments of both fund and bank portfolios. We restrict the banks' and funds' decision cycle to daily frequencies in order to simplify their portfolio problems. However, once a fund has committed to issue a bond to a bank the funds are locked up for the entire maturity of the bond.

### 3.1.4 Pension funds

## 3.2 Banks

Banks face a portfolio allocation problem that has to satisfy both regulatory and internal constraints. The main regulatory constraint is the capital adequacy requirement. Internal constraints are related to liquidity and maturities of the bank's assets. In the following we will assume that banks invest either exclusively or primarily into loans, which may also be sold in secondary markets (including possibly to funds), and thus can be considered to be bonds.

### 3.2.1 Heuristic rule

A benchmark portfolio strategy for bank that invests only in loans is based on a simple heuristic rule similar to the rule proposed for funds. In this case the supply function for a particular loan $i$ is given by:

$$S(\hat{r}_i) = (1 - w_c)\frac{e^{\beta \hat{r}_i}}{\sum_j e^{\beta \hat{r}_j}}(A^* - A)^+, \tag{3.34}$$

where $\hat{r}_i$ is defined either as the expected return on the bond $i$ or the corresponding Sharpe ratio where $\hat{r} \to \hat{r}/\sigma$. $A^*$ is the target size of the bank balance sheet. The $+$ operator is simply: $\max(x, 0) = x^+$. The means that the bank only expands its loan portfolio if it is below its target portfolio size. If it is above its target portfolio size the bank will not issue new loans and wait until its balance sheet has shrunk below its target (due to repayment of loans). $w_c$ is the cash reserve requirement of the bank. The bank holds a cash reserve in order to accommodate unanticipated deposit withdrawals.

While this allocation strategy only works for loans, extensions of this strategy can be developed for stocks. In this case the banks would be able to reduce its portfolio by selling stocks. The strategy would then have to specify how much to sell of each stock. Also, this strategy does not specify a demand function for bank funding (e.g. deposits, loans from funds). These questions will be addressed in the following bank strategy which is derived from profit maximization considerations.

### 3.2.2 Profit maximization

In the following we will outline a strategy for the management of the bank's balance sheet, i.e. for *both assets and liabilities*. At a given point in time, the bank takes its balance sheet as given. Then, subject to a capital adequacy constraint and a liquidity constraint, it will maximise its expected profit over the maturity of the newly purchased assets. I.e. the bank chooses the assets and liabilities that maximize its expected profit. The bank's decision variables are changes in its liquid portfolio rather than its actual portfolio position as typically done in one period models of the bank problem. This is necessary since at any given time, part of the bank's balance sheet cannot be changed due to long term maturities.

Note that banks only consider the profit over the maturity of the assets it is investing into at this time step rather than maximizing over an infinite horizon. Further note that banks are risk neutral in this decision problem.

Below we define a Lagrangian that sums up the bank's decision problem. The bank's objective function is the expected incremental profit it can earn by altering its portfolio today. As mentioned above the bank is subject to a number of constraints:

- Liquidity constraint: The liquidity constraint is always binding and ensures that all assets that the bank purchases are funded in some way - either through additional borrowing or through the sale of other assets (note that $a_i$ may be negative).
- Capital constraint: The capital constraint forces banks to have a ratio of risk weighted assets to equity below a certain threshold. The capital constraint may be slack.
- Lower bounds: Finally the bank is subject to a number of lower bound constraints which reflect the fact that parts of its balance sheet (such as loans) cannot be altered on a given times step as they have not yet matured.

These constraints result naturally from the decision problem and the regulatory environment. Given these constraints and the objective function the full Lagrangian is:

$$\mathcal{L} = \underbrace{\mathbf{a}\cdot\hat{\mathbf{r}}_A - \mathbf{l}\cdot\hat{\mathbf{r}}_L}_{\text{Expected profit}} + \underbrace{\mu\left(\overline{\lambda} - \frac{\mathbf{A}\cdot\mathbf{w}}{E}\right)}_{\text{Capital constraint}} + \underbrace{\nu\left(\sum_i a_i - \sum_j l_j\right)}_{\text{Liquidity constraint}} + \underbrace{\boldsymbol{\gamma}\cdot(\underline{\mathbf{a}} - \mathbf{a}) + \boldsymbol{\xi}\cdot(\underline{\mathbf{l}} - \mathbf{l})}_{\text{Lower bounds}}. \tag{3.35}$$

Let's define a few terms:

- Let $A_i'(t)$ and $L_j'(t)$ be the assets and liabilities at the beginning of time step $t$. At this time all assets maturing at time $t$ will have disappeared from the bank's balance sheet. Similarly any associated deposits will have disappeared (e.g. consider a firm paying back its loan with money in its deposit account). This is our starting point of the bank's decision problem.
- $a_i = A_i(t) - A_i'(t)$: change in asset $i$. These are the components of the vector $\mathbf{a}$. Similarly we collect the $A_i$'s in the vector $\mathbf{A}$. There is a lower

bound on the change in asset: $a_i \geq \underline{a}_i$, where $\underline{a}_i \in [-A_i'(t), 0]$. We collect the lower bounds in the vector $\underline{\mathbf{a}}$. The lower bound will depend on the fraction of the asset that can be traded. For example for loans, this lower bound will be 0 since all maturing loans have already been taken into account in $A_i'(t)$.

- $l_j = L_j(t) - L_j'(t)$: change in liability $j$. These are the components of the vector $\mathbf{l}$. There is a lower bound on the change in liability: $l_j \geq \underline{l}_j$, where $\underline{l}_j \in [-L_j'(t), 0]$. Again, the lower bound will depend on the type of asset. We collect the lower bounds in the vector $\underline{\mathbf{l}}$.

- $r_{Ai}$ is the return on asset $i$ over its maturity. $\hat{r}_{Ai}$ is the expected return on asset $i$ over its maturity. $\hat{\mathbf{r}}_A$ is the vector of these returns.

- $r_{Lj}$ is the interest rate paid on liability $j$. $\hat{r}_{Lj}$ is the expected interest rate paid on liability $j$ over some characteristic maturity (e.g. the average maturity of the asset side of the balance sheet). $\hat{\mathbf{r}}_D L$ is the vector of these interest rates.

- Capital ratio: $\lambda = \sum_i A_i(t) w_i / E = \mathbf{A} \cdot \mathbf{w}/E$, where $w_i$ are risk weights and $E$ is the bank's equity. $w_i$ are the components of $\mathbf{w}$.

- Capital adequacy constraint: $\overline{\lambda}$.

The decision problem is linear and can be solved analytically. For brevity we will not present the explicit solution (demand functions for assets and liabilities) here. The full solution to the problem can be found in **?**. However, to develop some intuition on the outcome of this decision problem note the following (based on simple numerical test cases):

- If all assets have the same risk weights and expected return the bank will distribute its investment equally across assets.

- If an asset has a low risk weight relative to other assets, it will be favoured in the allocation.

- If the margin between asset return and funding cost is very small the bank may choose to remain below its maximal capital ratio.

- Shocks to the funding cost spill over into the equilibrium returns on assets. E.g. if liabilities become more expensive the bank supplies less loans to the real economy which drives interest rates up.

Alternative specifications of the decision problem are available in **?** that take into account portfolio risk explicitly. However due to the non-linearity of these problems (and the existence of no shorting constraints), we cannot solve these problems in closed form.

In practice the majority of the bank's asset will consist of long term loans to firm and households, overnight interbank loans and potentially a small amount of stocks. We can tune the relative investments into the different assets by adjusting the corresponding risk weights. Even without tuning however, the investment into stocks is likely to be low due their high risk weight assigned by the regulator.

An open question remains whether banks will have the ability to adjust the risk weights based on internal risk models.

The bank's liabilities will consist of deposits, relatively long term loans from funds and ovenight interbank loans. While the changes in the amount of deposits and long term loans will be determined by bank's decision problem, we take interbank loans as residual. All banks will be subject to idiosyncratic deposit fluctuations leaving some banks either long or short in liquidity at the end of the day. The interbank market then redistributes the liquidity in the banking system via unsecured overnight loans. Should a bank not be able to find liquidity on the interbank market it will be able to access an overnight lending facility at the central bank at a penalty rate provided the bank can provide collateral. If it fails to provide collateral the bank will fail.

As outlined above, banks will transform relatively short maturity liabilities to longer term assets. The maturities of assets and liabilities will be set exogenously as outlined below. It can of course occur that liabilities cannot be rolled over. Then, if the bank does not have sufficient liquid assets to compensate for such a decline in funding and cannot find alternative sources of funding, the bank will fail. This reflects the perils of maturity transformation.

In principle we could also include short term secured interbank lending. The supply and demand on this market could be determined via the bank's decision problem. However, for simplicity we restrict us to these two sources of bank funding and defer secured interbank lending to a later version of this model.

### 3.2.3 Credit risk

Despite the fact that banks are risk neutral we can still incorporate credit risk into the bank's decision problem via the expected return on loans. The idea is that default risk plays a more important role in longer maturity loans. Therefore banks will charge a premium for longer maturity loans to compensate for the increased credit risk. This should naturally give rise to a yield curve. Below we outline a simple model of credit risk that takes maturity into account.

Suppose firms fail with some probability $p$ per time step. Let $L$ be the principal of the loan, $r$ the per period interest rate and $\tau$ the maturity. Then, if the firm does not fail, by the end of the maturity a payment of $(1+r\tau)L$ was made. Furthermore suppose that in the failure case a fraction $0 \leq z \leq 1+\tau r$ of the principal can be recovered - independent of when the failure occurs. Unsecured loans probably have $z \sim 0$ while secured loans will have a $z$ value determined by the expected liquidation value of the collateral. The per period expected return for a loan of maturity $\tau$ is simply:

$$\hat{r}(r,\tau,p,z) = \left((1-p)^{\tau}(1+\tau r) + (1-(1-p)^{\tau})z - 1\right)/\tau. \tag{3.36}$$

More sophisticated models to compute the expected return are conceivable that take into account the dependency of consecutive states of the world (i.e. failure would no longer be result of some iid random variable). However the main conclusion, namely that the expected return decreases in $\tau$ (for constant $r$), should remain unchanged. Thus the bank will charge a premium for longer maturity loans, i.e. it will be indifferent between two loans 1 and 2 with $\tau_1 > \tau_2$ if $r_1 > r_2$.

Of course the default probability is non-trivial to determine in the model. In particular it will probably be tough for default probabilities to be model

consistent. Therefore we should think of the default probability as a perceived quantity, i.e. a belief rather than an actual property of the firm. In this case we can postulate some map $F(o) \to [0,1]$ that maps a vector of observables $o$ to the unit interval. This map then encapsulates how the bank uses its available information to estimate a firm's default probability. An obvious choice for an observable would be the firm's leverage.

#### 3.2.4 Heterogeneity

Banks may differ along several dimensions. Below we list a few aspects of bank heterogeneity.

- Risk aversion: A simple way to parametrise a bank's risk aversion is via its perception of default probability of firms. Suppose for example that the bank bases its estimate of the firm's default probability on the firm's leverage $\lambda$ in the following way:

  $$p(\lambda) = \frac{1}{1 + \exp(-\beta\lambda + \gamma)}.$$

  The offset $\gamma$ and the intensity parameter $\beta$ will then determine the bank's risk aversion.

- Expectations: If banks invest into stocks they may form expectations based on different rules (e.g. fundamentalist, trend-follower).

- Relationship banking: Banks have access to a fixed or slow changing subset of the population of firms and depositors. Then heterogeneity in the population of firms and depositors will affect the banks lending and funding opportunities. This will naturally introduce heterogeneity in the banks balance sheets.

- Risk weights: Under certain circumstances the regulator may want to allow banks to choose their own risk weights in the capital constraint.

#### 3.2.5 Dividend payment

A proper dividend payment scheme remains to be developed. Dividend payments play an important role in the bank's leverage management. In particular, if the banks do not have any salable assets, then in order to deleverage they must do a combination of refraining from issuing new loans and refraining from paying dividends. K.P. is developing strategies for dividend payment schemes that maximize dividend payment taking into account constraints on leverage and the trade between current dividends and future profits. Note that the need to deleverage may be another factor forcing us to implement a secondary market for loans.

Note Gerald **?** has been experimenting with other alternatives involving simple heuristics for some of these decisions that we will not describe here.

### 3.3 Loans

Loan contracts will be generalized to provide multiple maturities and flexible repayment schemes. Loan maturities will range from short-term to long-term

and will be calibrated based on the empirical distributions of loan maturities. Long-term loans will be used in particular for funds' long-term investments into banks; such loans can be viewed as "bonds" – this becomes literally true if we implement a secondary market. Housing loans will be handled differently because in our scheme the changes in lending every day should be thought of as adjustments in the quantity of long-term loans.

Loans will have flexible *repayment schemes*, ranging from amortised payments to bullet payments. Loan contracts are currently designed as fixed-rate mortgages, where the loan is paid off in equal amounts at multiple steps throughout the maturity of the loan, such that the earlier payments primarily cover the interest on the principal, whilst the latter payments consist of more principal repayment than interest. We will add a repayment scheme more typical of bonds in which the entire principal of loan is due at the end of the term (i.e., bullet payment) and there are also interim (e.g., semi-annual) payments of interest (coupon) throughout the maturity.

## 3.4 Secondary market for loans?

One of the questions that at this point remains open is whether we need to implement a secondary market. If we did implement a secondary market, this would mean that we would allow a bank or a fund to sell its loans to other parties. It is not clear to us who those parties should be: The most natural thing would seem to be to allow both banks and funds to buy each others' loans, however, it is not clear if that is realistic.

The pros and cons of implementing a secondary market are:

- *Pros:* If we do not implement a secondary market then all loans are completely illiquid. Given that banks will (generally) have only loans as assets, this means that banks will only be able to deleverage very slowly. This may be unrealistic – real banks own a significant fraction of liquid assets like bonds. This also creates a problem for funds.
- *Cons:* Pricing bonds is a bit tricky. In order to implement a secondary market we have to be able to price loans of variable maturities, which can be fractions of a year. We may need to lump bonds into bins based on their maturities.

Note that if we implement secondary markets, this leaves the questions of which assets are eligible, e.g. do we want to allow home loans? Also who can participate, e.g. can funds buy home loans? (This could be a way to effectively have mortgage securities, but this may not be realistic for the UK).

## 3.5 Summary of maturities of loans

- **Banks:** Various maturity loans to firms and long-term loans to households (mortgages). Long-term loans from funds ('bonds').
- **Firms:** Various maturity loans from banks.
- **Funds:** Long to medium term loans to banks ('bonds').
- **Households:** Long-term loans from banks, but with the quantity adjusted daily.

### 3.6 Interbank market

For now the interbank market will be a residual market that redistributes liquidity in the banking system after idiosyncratic shocks to bank deposits. The interbank market will be modelled as an OTC market in which banks charge a risk premium based on the counter-party's leverage ratio.

# 4 Changes primarily affecting firms

## 4.1 Firm model

### 4.1.1 Firm decisions

The main change to the existing model of firm's production will be in the planning horizon. We want firms to make strategic decisions about production over longer, more realistic time horizons (e.g., a quarter or longer), which will be linked to the heterogenous maturities of their production loans. At the same time, the firms will need to continue producing and supplying their goods to the market every day to ensure the constant flow of goods to the households and other firms that need them as production inputs.

We have recently made several changes to the firms' decision making; however since these were not driven by discussions with Bank of England, and since they are already completed, we will not discuss them here but rather refer the reader to **?**. As part of our future changes we will be developing new algorithms for firm decision making with longer time horizons. One line of decision rules will be based around multi-period optimisation. These are not yet fully developed and will be described elsewhere, and since they can be implemented by plug-ins it is not important to specify them at this stage.

# 5 Heterogeneity

## 5.1 Heterogeneous risks

We will allow both firms and banks idiosyncratic risks yielding heterogenous interest rates. These heterogenous risks will be taken into account in the clearing mechanism explained in Section 5.4.

## 5.2 Heterogeneous timescales

One of the major extensions of the existing model involves a departure from the current myopic (one-step-ahead) model into a model with more realistic and heterogeneous timescales. These timescales are related to:

- Multi-period contracts of various maturities and repayment schemes (see Section 5.2.1).
- Maturity transformation (see Section 5.2.2).
- Agents' decision-making – the financial part of the model works mostly on a daily scale, while macro decisions typically occur less frequently, e.g. quarterly (see Section 5.3).

The model currently works on a single homogeneous timescale, where one time-step is defined to correspond to one day. For example, all labour and loan contracts last for one day, banks' investment decisions are updated daily, firms' production is updated daily, and the cycle repeats. To build a more realistic model, we need to allow processes to operate on different characteristic timescales and allow contracts to have maturities on different timescales. In general, the financial system (financial markets, banks, funds) will operate on a daily timescale, while the macroeconomic sectors (firms and households) typically operate on longer timescales ranging from a month, to a quarter and even up to several years.

**Calibration**. To calibrate the model, we will try to match the profile of loans observed in reality, both in terms of volume and yields. To calibrate for the volume of loans per maturity, we will assign maturities to loans by pulling random numbers from the empirical distribution of loan maturities. We will include maturity into the decision-making of lenders, so that the endogenous yield curve (obtained by the clearing mechanism) corresponds qualitatively (if not quantitatively) to the empirical yield curve.

We intend to capture *maturity transformation*, where there is interaction and transmission between economic information (prices and yields) at different timescales, with lending at long timescales supported by borrowing at short time scales.

### 5.2.1 Contract timescales

We will incorporate realistic timescales for contracts, not only in terms of their maturities (e.g., house mortgages have multi-year maturities) but also in terms of the frequency of the cash-flow they generate. Note this depends on the level of aggregation, e.g. since our households are aggregates of households making transfers every day makes sense. Various maturities are important to capture the concept of maturity transformation, while realistic repayment schemes (cash-flow frequencies) are relevant from a liquidity perspective.

Typical maturities and cash-flow frequencies of the main contracts are provided below:

- **Loans:** Multiple maturities. Various repayment schemes.
- **Labour:** Employment contract yearly and longer. Wage payments monthly.
- **Stocks:** Non-expiring contract. Dividend payments quarterly.
- **Deposit:** Pays interest monthly.

### 5.2.2 Maturity transformation

Banks lend out money to firms at a variety of maturities. Banks are involved in both money creation and maturity transformation - they try to make a profit from the spread between long-term and short-term debt by borrowing short-term (from other banks and money market funds) and lending long-term (to firms).

## 5.3 Timescales of agent decisions

Each agent can perform strategic operations and decision-making on multiple timescales – some decisions may be short-term and are updated every day (one time-step in our model), whereas other decisions can be mid to long-term e.g. updated monthly, quarterly, yearly, all the way up to multi-yearly. When referring to heterogeneous timescales and multiple maturities, we are mainly considering the timing of regular decision-making operations by the agents and the length of related contracts such as loans.

The flow of money in the simulation is encapsulated by contracts of different maturities and repayment schemes. Once set up, contracts control the transfers between agents (e.g., a lender and a borrower) according to their own schedule and without additional actions required from agents. This means that the focus of agent decision-making is on deciding when and what type of contracts to create.

For example consider a firm's decision about whether or not to borrow to finance production. Firms' strategic decisions about borrowing will not occur every day, but less frequently (e.g., quarterly). However, we will not schedule those decision to occur with an exact frequency. Instead, in each period, the firms will be queried if they want to take new loans, and their decision will depend on a set of rules, such as: Does the firm need more money? Has the firm already repaid the existing loan? Does the firm want to expand its production? In most time periods this will result in a negative answer. However, when the answer is positive, a new loan will be issued with a maturity pulled from an empirical distribution of loan maturities.

If a firm at any time runs into financial troubles (e.g., insufficient funds to repay loans) this will be dealt with separately, through *bankruptcy handling* procedures (i.e., this is not part of the regular firm's decision-making process).

## 5.4 Heterogenous market clearing

Many of the changes we are contemplating require heterogeneous markets, in which it is necessary to set prices for related but non-identical goods. Following **?** we will use a simultaneous market clearing approach which allows heterogeneous interest rates. Agents are represented by nodes in a multiplex network such that each directed edge represents a borrower-lender/buyer-seller relationship between agents, and each layer in the network represents a particular contract type (e.g., a loan, a repo, a stock). The clearing minimises the following function:

$$f(\mathbf{x}) = \sum_{i,j} (D_{ij}(\mathbf{x}) - S_{ij}(\mathbf{x}))^2, \tag{3.37}$$

where $\mathbf{x}$ is the vector of returns (interest rates or stock returns) on all edges and layers, $D_{ij}(\mathbf{x})$ is the demand of agent/borrower $i$ and $S_{ij}(\mathbf{x})$ is the supply of agent/lender $j$.

# 6 Scale and the use of heterogeneous representative agents

Due to computational limitations agents will be modeled on different scales with varying degrees of heterogeneity. In particular:

1. *Financial agents.* Given that there are less than 20 important banks in the UK and the number of funds is relatively small, we should be able to model the number of financial agents at one-to-one scale.

2. *Households.* There are roughly 20 million households in the UK and so it is computationally infeasible to model them all. Since the number of households we can reasonably simulate is in the range of $1,000-10,000$, we will therefore plan on having each simulated household represent $2,000-20,000$ real households. These households will be heterogeneous, in that they will correspond to households with different wealth, income and age (and in the future possibly other variables).

3. *Firms.* There are the order of three million firms in the UK. However most of these are small, so we propose to model a few hundred of the largest firms on one-to-one scale and then model the remaining firms in terms of representative agents. The precise breakdown remains to be determined after gathering more data. The number of firms must also be chosen in concert with the number of goods. as we will want to allow multiple firms per good.

# 7 Open issues

We still have the following open issues:

1. Should we implement a secondary market for loans and if so which types of loans and which agents can participate?

2. In order to get the proper maturity transformation, do we need to include short term loans for banks other than those they can obtain in the inter-bank market? If so, who is lending to banks on short timescales? (e.g. funds?)

3. How do we decide when a bank asks for a loan from a fund?

# GNU Free Documentation License

Version 1.3, 3 November 2008


<http://fsf.org/>



## Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

# 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related

matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "**Invariant Sections**" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "**Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "**Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "**Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "**Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "**publisher**" means any person or entity that distributes copies of the Document to the public.

A section "**Entitled XYZ**" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "**Acknowledgements**", "**Dedications**", "**Endorsements**", or "**History**".) To "**Preserve the Title**" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy

of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

# 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and

publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the

Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new

problems or concerns. See `http://www.gnu.org/copyleft/`.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

## 11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

## ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU

> Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with ... Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.