Essays In Macroeconomics

Abstract
This dissertation is composed of three chapters. In the first two chapters, I study the welfare and distributional consequences of government policies in economies with credit constraints and heterogeneous agents. Given the complexity to compute such models, the third chapter reviews the basics of parallel computing in macroeconomics.

In the first chapter, I quantitatively assess the welfare maximizing policy during the Great Recession, when the government had access to two policy instruments: a) offer bailouts to banks, and b) subsidize the mortgage refinancing of households. The implementation of these instruments involves a trade-off, shaped by a deadweight loss caused by foreclosures and an information friction on house prices. The main finding is that a subsidy-only policy would have generated welfare gains of up to 0.4% in consumption equivalent terms when compared to the HAMP and TARP benchmark. In the second chapter, I study the general equilibrium effects of government-supplied student loans in the educational markets of developing countries. Our main finding suggests that subsidized student loans can lead to a widening gap in the quality of education offered by top-10 versus top-50 educational institutions. Given the complexity to compute models with heterogeneous agents, the third chapter describes the basics of parallel computing in economics, reviews widely-used implementation routines and compares performance gains using as a test bed a standard life-cycle problem.

Degree Type
Dissertation

Degree Name
Doctor of Philosophy (PhD)

Graduate Group
Economics

First Advisor
Jesús Fernández-Villaverde

Second Advisor
Dirk Krueger

Subject Categories
Economics

This dissertation is available at ScholarlyCommons: https://repository.upenn.edu/edissertations/2940
To Laura, María Helena, Carlos and Salomón
Acknowledgements

First, I would like to thank my advisors Jesús Fernández-Villaverde and Dirk Krueger for guiding me throughout these years. Their patience, support, and guidance shape the way I think about research and macroeconomics. I owe them having taught me to enjoy what I do and to tackle problems in a pragmatic way. All of this dissertation is a consequence of that. I also thank Jesús for giving me the opportunity to work as his Research Assistant, and to join the lectures on parallel computing in the Quantitative Macro Theory course. The third chapter of this dissertation is a consequence of our interaction during this course.

For their helpful guidance and comments on the first chapter of this dissertation, I would like to thank Hal Cole, Guillermo Ordóñez and participants at the Macro Workshops at the University of Pennsylvania. I also thank the Federal Reserve Bank of San Francisco for the Dissertation Fellowship during the Summer 2017, which allowed me to significantly improve my job market paper.

I would like to thank Rodrigo, María José, Eugenio, Minsu, Juan Herreño, and Camilo Hernández for their unconditional support and friendship during these years. In particular, I am extremely grateful to Rodrigo Azuero, who has been a great friend and co-author. His energy, advice, and example helped me overcome the difficult moments during the program and made my time at Penn more enjoyable.

Este trabajo se lo debo, ante todo, a Laura. Su amor, paciencia, comprensión y apoyo incondicional fueron fundamentales para culminar esta etapa. Su sonrisa y alegría me
ayudaron a sobrellevar los momentos difíciles, a ver la luz al final del túnel y a agradecerle
a la vida cada día por todas las oportunidades que he tenido. Ella me empujó a dar mucho
más de lo que jamás podría haber dado solo. A mis papás les agradezco el apoyo en cada
paso que di. Además de ser un ejemplo, me han enseñado a proceder con rectitud y a
poner en perspectiva cada situación de la vida. Esto me ayudó a mantener el norte y a
disfrutar cada etapa del proceso. De Salomón sé que, sin importar las decisiones que yo
tome, siempre recibiré apoyo incondicional. Esto me dio confianza y fuerza para seguir
adelante.
This dissertation is composed of three chapters. In the first two chapters, I study the welfare and distributional consequences of government policies in economies with credit constraints and heterogeneous agents. Given the complexity to compute such models, the third chapter reviews the basics of parallel computing in macroeconomics.

In the first chapter, I quantitatively assess the welfare maximizing policy during the Great Recession, when the government had access to two policy instruments: a) offer bailouts to banks, and b) subsidize the mortgage refinancing of households. The implementation of these instruments involves a trade-off, shaped by a dead-weight loss caused by foreclosures and an information friction on house prices. The main finding is that a subsidy-only policy would have generated welfare gains of up to 0.4% in consumption equivalent terms when compared to the HAMP and TARP benchmark. In the second chapter, I study the general equilibrium effects of government-supplied student loans in the educational markets of developing countries. Our main finding suggests that subsidized student loans can lead to a widening gap in the quality of education offered by top-10 versus top-50 educational institutions. Given the complexity to compute models with heterogeneous agents, the third chapter describes the basics of parallel computing in economics, reviews widely-used implementation routines and compares performance gains using as a test bed a standard life-cycle problem.
1.7  The Great Recession ......................................................... 42
  1.7.1  HAMP ................................................................. 43
  1.7.2  The Information Parameter $p$ ....................................... 44
  1.7.3  TARP ................................................................. 45
  1.7.4  Results and Counter-factual Experiments .......................... 45
1.8  Concluding Remarks .......................................................... 51

2  General Equilibrium Effects of Student Loans on the Provision and Demand
for Higher Education ................................................................... 55
  2.1  Introduction ................................................................. 56
  2.2  Related Literature .......................................................... 61
    2.2.1  The Role of Borrowing Constraints ............................... 62
    2.2.2  General Equilibrium Effects and the Bennett Hypothesis .... 64
    2.2.3  The Education firms .................................................. 65
  2.3  A Model of the Market for Higher Education with Credit Constraints ... 67
    2.3.1  Households ............................................................ 68
    2.3.2  Universities ............................................................ 77
    2.3.3  Discussion .............................................................. 79
    2.3.4  Government ............................................................ 81
  2.4  The “Revolución Educativa” of Colombia ................................ 83
    2.4.1  Enrollment and inequality ........................................... 84
    2.4.2  Higher Education institutions ....................................... 85
    2.4.3  The ACCES Program ............................................... 87
### 2.4.4 Product Differentiation in the Market for Higher Education

2.5 Numerical Analysis

2.5.1 Evolution of Quality

2.5.2 Calibration of the Model

2.5.3 Parameterization

2.5.4 Results

2.6 Conclusion

3 A Practical Guide to Parallelization in Economics

3.1 Introduction

3.2 Why Parallel?

3.3 What Is Parallel Programming?

3.4 When Do We Parallelize?

3.4.1 An application where parallelization works: Value function iteration

3.4.2 An application where parallelization struggles: A random walk Metropolis-Hastings

3.4.3 The road ahead

3.5 A Life-Cycle Model

3.6 Different Computers

3.7 Parallelization Schemes on CPUs

3.7.1 Julia - Parallel for

3.7.2 Julia - MapReduce

3.7.3 Matlab
3.7.4 R ................................................................. 147
3.7.5 Python .......................................................... 150
3.7.6 C++ and OpenMP .......................................... 153
3.7.7 Rcpp and OpenMP .......................................... 159
3.7.8 MPI ............................................................... 162

3.8 Parallelization Schemes on GPUs ............................ 169
3.8.1 CUDA .......................................................... 172
3.8.2 OpenACC ..................................................... 178

3.9 Beyond CPUs and GPUs ....................................... 181

3.10 Conclusion ....................................................... 182

Appendices ......................................................... 184

.1 Wall Street or Main Street: Who to Bail Out? .............. 185
   .1.1 Steady State Equilibrium ................................ 185
   .1.2 Computation of the Model .............................. 188
   .1.3 Proves of Propositions .................................. 192
   .1.4 Other Figures .............................................. 194

.2 General Equilibrium Effects of Student Loans on the Provision and Demand for Higher Education .......................... 195
   .2.1 Appendix A .................................................. 195

.3 A Practical Guide to Parallelization in Economics .......... 215
List of Tables

Wall Street or Main Street: Who to Bail Out? 3
Table 1 Parameter values 34
Table 2 Model and Data Moments 35
Table 3 % of defaulters with income and price shocks on the period of default 38
Table 4 Productivity Shock 43

General Equilibrium Effects of Student Loans on the Provision and Demand for Higher Education 55
Table 1 Estimates for the quality production function 98
Table 2 Parameter values 101
Table 3 Equilibria computations 102

A Practical Guide to Parallelization in Economics 105
Table 1 Number of CUDA cores and memory in NVIDIA GPUs 170
# List of Figures

**Wall Street or Main Street: Who to Bail Out?**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Foreclosure Starts (Mortgage Bankers Association) and House Price Index (Core Logic). Source: Chatterjee and Eyigungor (2015)</td>
<td>5</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Subsidy vs. Bailout policy</td>
<td>13</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Home Equity Cumulative Distribution - Model vs. Data (SCF 2007)</td>
<td>36</td>
</tr>
<tr>
<td>Figure 4</td>
<td>State space distributions</td>
<td>37</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Home Equity Cumulative Distribution - Model vs. Data (SCF 2007)</td>
<td>38</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Model average default rate (%) (a) by age, and (b) home equity (%)</td>
<td>39</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Mortgage refinancing as a % of home-owners by age and equity</td>
<td>41</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Mortgage refinancing as a (%) of home-owners by age. Solid line: refinancing to increase mortgage size. Dashed line: refinancing to decrease mortgage size</td>
<td>41</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Transitional dynamics of (a) house prices and (b) foreclosure rate</td>
<td>47</td>
</tr>
<tr>
<td>Figure 10</td>
<td>Equity distribution during transitional dynamics</td>
<td>48</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Welfare gains in consumption equivalent terms (%) under a subsidy-only policy, by home equity (%)</td>
<td>49</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Default rate by age, before and after the Great Recession</td>
<td>50</td>
</tr>
</tbody>
</table>
### General Equilibrium Effects of Student Loans on the Provision and Demand for Higher Education

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>% of enrollment in private institutions by country</td>
<td>59</td>
</tr>
<tr>
<td>2</td>
<td>Representation of the education decisions on the state space</td>
<td>71</td>
</tr>
<tr>
<td>3</td>
<td>Number of students that change their study decision when borrowing constraints change from $\bar{A} = 0$ to $\bar{A}$, by ability $\theta$</td>
<td>78</td>
</tr>
<tr>
<td>4</td>
<td>Enrollment, income and financial aid</td>
<td>84</td>
</tr>
<tr>
<td>5</td>
<td>Evolution of Higher Education in Colombia</td>
<td>86</td>
</tr>
<tr>
<td>6</td>
<td>Average decile of ability of entering student body, measured by test scores</td>
<td>87</td>
</tr>
<tr>
<td>7</td>
<td>Differences in quality of student body</td>
<td>89</td>
</tr>
<tr>
<td>8</td>
<td>Quality supplied by colleges</td>
<td>90</td>
</tr>
<tr>
<td>9</td>
<td>Difference in professors per student</td>
<td>91</td>
</tr>
<tr>
<td>10</td>
<td>Evolution of Faculty composition. Professors per Student</td>
<td>92</td>
</tr>
<tr>
<td>11</td>
<td>Gap in academic production</td>
<td>93</td>
</tr>
<tr>
<td>12</td>
<td>Evolution of average tuition prices over the decade</td>
<td>94</td>
</tr>
</tbody>
</table>

### A Practical Guide to Parallelization in Economics

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Number of transistors</td>
<td>110</td>
</tr>
<tr>
<td>2</td>
<td>Cray-1, 1975</td>
<td>111</td>
</tr>
<tr>
<td>3</td>
<td>Sunway TaihuLight, 2016</td>
<td>112</td>
</tr>
<tr>
<td>4</td>
<td>Granularity</td>
<td>116</td>
</tr>
<tr>
<td>5</td>
<td>Fork-join parallel programming</td>
<td>120</td>
</tr>
<tr>
<td>6</td>
<td>1 Core used for computation</td>
<td>130</td>
</tr>
</tbody>
</table>
Figure 7 8 Cores used for computation ........................................ 131
Figure 8 Results in Julia with different number of processors. Number of experiments: 20. ........................................ 141
Figure 9 Results with \texttt{map} function in Julia with different number of processors. Number of experiments: 20. ........................................ 144
Figure 10 Results in Matlab with different number of processors. Number of experiments: 20. ........................................ 146
Figure 11 Results in R with different number of processors. Number of experiments: 5. ........................................ 150
Figure 12 Results in Python with different number of processors. Number of experiments: 20. ........................................ 153
Figure 13 OpenMP computing ........................................ 155
Figure 14 Results in C++ with OpenMP with different number of processors. Number of experiments: 20. ........................................ 158
Figure 15 Results in Rcpp with OpenMP with different number of processors. Number of experiments: 20. ........................................ 162
Figure 16 MPI computing ........................................ 163
Figure 17 MPI Functions ........................................ 164
Figure 18 MPI synchronization with barrier function ........................................ 166
Figure 19 Results in C++ with MPI with different number of processors. Number of experiments: 20. ........................................ 168
Figure 20 CUDA GPU architecture. ........................................ 171
Figure 21 Computing time (s) by language with different number of processors. 182
Figure 22  Life cycle averages of (a) Owner-occupied housing, (b) Mortgage size, (c) Home equity, and (d) Savings .......................... 194

Figure 23  Representation of the education decisions on the state space. .... 206

Figure 24  Computing time(s) in C++ with different number of processors and optimization flags. ...................................................... 215
Introduction

Recent computational developments have made it possible to study in a greater detail the design of optimal government policy in economies with heterogeneous agents, to which distributional concerns are inherent. This dissertation takes advantage of such developments to study the effect of government policies in two seemingly-different contexts: the mortgage markets in the U.S. during the recent financial crisis, and the market for college education in developing countries. In both settings, governments have implemented policies to alleviate credit constraints. In this sense, financial frictions are at the core of this dissertation.

In the first chapter, I study the welfare consequences of government policies in mortgage markets during the Great Recession. Mortgage markets lied in the heart of the Great Recession. After a drop in house prices, foreclosures tripled from their historical average and generated large losses to financial institutions. When many of the global systemically important banks faced solvency problems, the U.S. government implemented a bailout program to mitigate the losses generated by foreclosures, and a mortgage modification program to avoid further default from households. The first chapter of this dissertation studies whether the way in which the bailout and subsidy programs were implemented was welfare-maximizing. The main finding is that the government could have generated welfare improvements by expanding the subsidy program to households and eliminating the bailout program to banks.
The second chapter studies the design of subsidized student loan policies for college education in developing countries. The literature in the economics of education has identified the existence of a large inefficiency when credit markets are imperfect, as many high-ability households that face credit constraints are left out of the market. During the last decade, the Colombian government made an effort to alleviate this inefficiency by providing subsidized student loans to high-ability households. The literature has identified a positive impact of these programs on college enrolment. However, there are general equilibrium consequences of this program that have not been studied. In particular, the evidence from Colombia suggests that, as the government expanded the supply of subsidized student loans, the gap in the quality of education offered by top-10 versus top-50 institutions widened. We propose a model in which the divergence in the quality offered is a consequence of the subsidized student loan program offered by the government.

The last chapter of the dissertation is an effort to advance the use of parallel computing in macroeconomics. The computation of most models with heterogeneous agents, such as the two described above, requires great computational power. The use of parallel computing has improved the performance achieved by serial computing, by exploiting the multi-core architecture of modern computers. In this paper, we describe the basics of parallel computing, discuss the methods to implement it across different programming languages, and provide the reader with multiple example codes for the purpose of exposition.
Chapter 1

Wall Street or Main Street: Who to Bail Out?

Housing crises are characterized by a sharp increase in foreclosure rates that generates losses to mortgage investors. To preserve the solvency of these investors, governments have historically implemented two policies: a) offer them bailouts (Wall Street), and b) subsidize the mortgage refinancing of households to prevent additional foreclosures (Main Street). The implementation of these instruments involves a trade-off, shaped by two frictions. On one hand, houses lose 20% of their value during the foreclosure process because of larger depreciation due to vacancy and vandalism. If the government offers complete bailouts to investors rather than subsidies to households, the economy has to bear the dead-weight loss of the value lost by foreclosed houses. On the other hand, house prices have an idiosyncratic component, so the government does not have perfect information on individual households’ decision to default. Households have incentives to engage in strategic default to qualify for benefits. A subsidy policy transfers resources to households that were not planning to default in the absence of the policy but avoids the dead-weight loss of foreclosures. I quantitatively assess the welfare-maximizing policy in a heterogeneous
agents’ economy and find that a subsidy-only policy would have generated welfare gains of up to 0.4%, measured as the consumption equivalent variation, as compared to the baseline calibration that matches the TARP and HAMP programs implemented during the Great Recession. Households on the left tail of the equity distribution, which benefit the most from a subsidy program, obtain the largest welfare gains. In contrast, a bailout-only policy would have generated a welfare loss of 0.8%.

1.1 Introduction

In the U.S., mortgages are the main source of funding for house purchases, used by two out of three homeowners to finance their home acquisition. Mortgages are collateralized debt contracts in which borrowers pledge their house as collateral. In case of default, the property is foreclosed upon and is legally transferred to the lender, to cover for the losses of the loan. With some exceptions, such as the Great Depression and the Great Recession, the average yearly foreclosure rate has oscillated around 1.5% of total outstanding mortgages. During the recent financial crisis, after house prices plummeted more than 20%, the foreclosure rate almost tripled from an average of 1.52% during 1991-2006, to 4.36% in 2007-2010. This sharp increase in foreclosures generated significant losses to financial institutions, which are the largest investors in the mortgage market (Antoniades, 2015). Figure 1 illustrates the behavior of foreclosure starts since 1992 and the Core Logic housing price index (HPI).

During mortgage crises, governments have historically implemented two sets of policies to preserve the solvency of financial institutions: 1. offer bailouts to cover for the losses of these institutions, and 2. subsidize the mortgage refinancing of households to prevent additional foreclosures. During the Great Recession, through the Emergency Economic Stabilization Act (EESA) of 2008, the Congress authorized the Secretary of the Treasury to
implement both types of policies. First, it approved the Troubled Assets Relief Program (TARP), which allocated up to $700 billion for investments in institutions that were in financial distress, through the purchase of troubled assets. These investments implicitly bailed out the recipient institutions; of the $243 billion invested by 2008, the Congressional Budget Office (CBO) estimated a risk-adjusted net present value return of −$61 billion, or 0.5% of the 2007 GDP. Second, the Treasury committed $75 billion to subsidize mortgage refinancing through the Home Affordable Modification Program (HAMP), in order to reduce payments-to-income (PTI) ratios and prevent default.

In a frictionless, full-information and representative agent economy, offering bailouts to mortgage investors or subsidies for the mortgage refinancing of households would have exactly the same welfare consequences. If the government could observe which households are planning to default after a house price shock, it could offer them the

---

1. These investments do not include loans to automobile companies.
2. Under Section 202 of the EESA, the Office of Management and Budget (OMB) and the Congressional Budget Office (CBO) were required to estimate a semiannual risk-adjusted net present value of TARP. The first of such reports published by the CBO in January 2009 estimated a risk-adjusted net present value of the Capital Purchase Program (CPP) equal to −$61 billion. Although TARP was not specifically directed to cover the losses generated by mortgage investments, the primary driver of the crisis was the exposure to the housing sector (Antoniades, 2015). See Calomiris and Khan (2015) for further details.
exact amount that would make them indifferent between defaulting or not. This would prevent the foreclosure rate from rising and financial intermediaries from experiencing losses. Otherwise, the government could allow foreclosures to rise and transfer to banks an amount exactly equal to their losses. In the absence of any friction, both strategies would yield the same welfare outcome.

However, at least two frictions make this policy design non-trivial. On one hand, wide empirical evidence documents the existence of a 20% – 22% price discount on foreclosed houses. Using hedonic regression methods, Campbell et al. (2011) find that in the State of Massachusetts houses that have been foreclosed upon are sold at a price 27% lower, as compared to houses with similar characteristics that are sold by homeowners rather than banks. Out of this 27%, they estimate that 7 percentage points correspond to the discount of banks engaging in fire sales,3 while the remaining 20 percentage points correspond to a loss of value throughout the foreclosure process, mainly because of a larger depreciation due to vacancy and vandalism.4 Similarly, using a repeat-sales methodology Pennington-Cross (2006) follows a panel of houses over time and finds that when a house is sold after a foreclosure, its price is 22% lower.5 If the government implements a policy that offers bailouts to mortgage holders and does not prevent foreclosures, this 20 – 22% value lost on foreclosed houses represents a dead-weight loss to the economy.

On the other hand, the literature has identified an uninsurable idiosyncratic component in house prices, which is not perfectly observable by the government.6 Given that the

---

3. Mayer (1995) argues that in markets with search frictions, urgent sales are made at lower prices because the quality of the match is lower. This supports the fact that houses in foreclosure are sold at discount, as the real estate market is highly illiquid and banks have incentives to hold the properties for a short time.

4. Part of this price discount can also be explained by lower investments of households prior to default. Melzer (2017) finds that homeowners with a high risk of mortgage default cut down substantially on home improvements, consistent with the debt overhang theory. When households cannot appropriate the returns on their investment at every state of the world, investment falls. In particular, households with negative home equity spend 30% less per quarter on home investments, compared to households with positive equity. Similarly, Harding et al. (2000) find evidence that households with high loan-to-value (LTV) ratios invest less in their properties.

5. See Pennington-Cross (2006) for a review of the literature.

6. See Case and Shiller (1989); Flavin and Yamashita (2002) and Piazzesi and Landvoigt (2015). There are at least two different sources of idiosyncratic house price risk: a) there are idiosyncratic quality shocks that
default decision strongly depends on house prices, the government cannot perfectly observe which households plan to default at every point in time. As a consequence, any policy that subsidizes the mortgage refinancing of households at risk of default generates a moral hazard problem, as households that are not in financial distress might engage in “strategic default” to obtain the benefits. Mayer et al. (2014) estimate that strategic default can be as high as 10% of total default after the announcement of a mortgage modification program. Therefore, any policy that subsidizes mortgage refinancing will avoid the dead-weight loss of foreclosures, but has to cover the costs of subsidizing strategic defaulters, as pointed out by Foote et al. (2008) and Mayer et al. (2014). If taxation is distortionary, levying taxes to subsidize strategic defaulters represents a welfare loss to the economy, through a distortion of the labor decisions of households.

The purpose of this paper is to quantitatively study the welfare-maximizing policy to preserve the solvency of financial institutions during mortgage crises, when governments have the two policy instruments described above. In particular, I study the case of the Great Recession: I assess the welfare costs of the dead-weight loss and strategic default, and evaluate counter-factual policies that yield higher welfare outcomes, when compared to the TARP-HAMP baseline. I use a heterogeneous agents’ life-cycle model with housing and long-term mortgage markets, calibrated in the initial steady state to match the pre-crisis U.S. economy. The model is disciplined by using micro estimates on the dead-weight loss of foreclosures, the magnitude of strategic default and the Frisch elasticity of demand estimated in the macro literature for the calibration. Then, I replicate the financial crisis after unexpected shocks to labor productivity and a loan-to-value restriction at origination hit the economy in 2007, assuming the government implements a policy analogous to TARP and HAMP. This laboratory allows me to perform counter-factual experiments to assess the welfare gains of different policies.

scale over time, and b) the housing market is illiquid, so there are one-time shocks that are realized at the time of sale reflecting the quality of the match. Giacoletti (2016) finds that the idiosyncratic component of house prices represents 80% of house price variance.
A heterogeneous agents’ economy is the appropriate laboratory in which to study the
design of a welfare-maximizing policy for two reasons. First, in addition to the frictions
already described, offering subsidies to households or bailouts to banks has redistributional
consequences that affect welfare and have to be taken into consideration. The welfare
consequences of either policy are mainly driven by marginal propensities to consume and
insurance motives, which are fully considered in a heterogeneous agents’ environment
(Mian et al., 2013; Kaplan and Violante, 2014). Second, as pointed out by Foote et al.
(2008), the cost of strategic default in a subsidy policy depends on the fraction of eligible
households that plan to default after an aggregate house price shock. If a large fraction of
the eligible households finds it optimal to default after the aggregate house price shock,
the cost of the information friction will not be large, as the fraction of households that
can potentially engage in strategic default is small. In contrast, if only a small fraction of
eligible households find it optimal to default after the shock, the total amount of subsidies
distributed to strategic defaulters is potentially large. In a heterogeneous agents’ economy,
the welfare-maximizing policy will offer subsidies to certain groups of the population,
based on observable characteristics.

The main result of my paper is that a subsidy-only policy dominates the TARP-HAMP
combination implemented through the EESA of 2008. The reason is that the distortion
generated by the taxes levied to finance strategic default, given the macro estimates of
the Frisch elasticity of labor supply, is smaller than the dead-weight loss of foreclosure.
For this reason, completely replacing a bailout policy such as TARP with a subsidy policy
analogous to HAMP would yield welfare gains equal to 0.25% in consumption equivalent
terms. The welfare gains are concentrated on the lower tail of the equity distribution, as
households with the highest debt levels receive the largest subsidies. However, households
that do not receive benefits also experience welfare gains, as labor taxes are lower under a
subsidy-only policy. If, in addition, the government implements a better eligibility rule
that targets the households that would not default in the absence of the crisis, and adds an
age component to the subsidy, the welfare gains rise to 0.4%. Finally, eliminating HAMP and extending TARP would yield a welfare loss of 0.8%.

This paper abstracts from two additional frictions that arise in this environment. First, by assuming that the crisis happens after a completely unexpected shock, I abstract from the moral hazard problem that exists when financial institutions and households anticipate the government’s policy and engage in overly-risky behaviors prior to the crisis. There is anecdotal evidence that policy-makers at the Federal Reserve Board and the Treasury ignored any moral hazard concerns at the time of the policy design in 2008. As Timothy Geithner stated, “Trying to mete out punishment to perpetrators during a genuinely systemic crisis—by letting major forms fail or forcing senior creditors to accept haircuts—can pour gasoline on the fire. It can signal that more failures and haircuts are coming, encouraging creditors to take their money and run. Old Testament vengeance appeals to the populist fury of the moment, but the truly moral thing to do during a raging financial inferno is to put it out.”

Second, I ignore the externalities that foreclosures generate on the price of neighboring houses. The literature has documented this externality, but its effects are very local and do not seem to be very large (see Campbell et al. (2011)). In any case, including a price externality would make foreclosures even costlier, which would reinforce the results obtained in this paper. Including these two frictions is left for future research.

This paper is organized as follows. Section 2.2 reviews the literature to which this paper is related. Section 1.3 illustrates the main mechanism driving the welfare-maximizing policy, through a simple example. Section 2.3 presents the heterogeneous agents’ model used throughout the rest of the paper. Section 1.5 states the definition of a recursive general equilibrium in this environment. Section 2.5.2 describes the calibration of the model to the pre-crisis U.S. economy and the main results in the initial steady state. Section 1.7 presents the main results of the paper and Section 2.6 concludes.

---

1.2 Related Literature

This paper models the mortgage market, so it is inherently related to the extensive literature on housing and mortgage decisions. My model builds upon Gervais (2002), who models an economy with heterogeneous agents that optimally choose between homeownership and renting. Jeske et al. (2013) add to the model the presence of short-term mortgages, where households have the possibility of defaulting on their debt obligations and the pricing of mortgages is consistent with the probability of default. This paper also models the mortgage market and allows for the possibility of default. However, I model long-term mortgage markets, rather than short-term contracts. In this respect, this paper is close to Chatterjee and Eyigungor (2015), Corbae and Quintin (2015) and Hedlund (2016), who study foreclosures, homeownership, consumption and other macroeconomic aggregates during the recent financial crisis. I depart from all of these works in that I study the welfare consequences of a subsidy-bailout policy aimed at preserving the solvency of mortgage investors.

This model is also related to models that study the welfare effects of fiscal policies within financial crises, where the economy is modeled in the steady state and counterfactual fiscal policy experiments are performed after an unexpected aggregate shock. A good example of such a paper is Kaplan and Violante (2014).

As in Jeske et al. (2013), this paper assumes that the idiosyncratic component of house prices is driven by an idiosyncratic depreciation that is partially observable by the government. This assumption is supported by the empirical evidence found by Case and Shiller (1989), Flavin and Yamashita (2002) and Piazzesi and Landvoigt (2015), who estimate large idiosyncratic components in the variance of house prices. The partial observability of these idiosyncratic shocks generates strategic default after the announcement of any mortgage refinancing subsidy policy (Foote et al., 2008; Mayer et al., 2014).
This paper is also related to the literature that studies bailout policies. In particular, this paper studies the ex-post costs of bailouts (Acharya et al., 2014), and abstracts from ex-ante moral hazard concerns arising from different policy combinations. For this, I assume that the economy is in the steady state and a one-time, completely unexpected shock hits the economy, so financial institutions and homeowners do not foresee any government policy in the steady state. Fahri and Tirole (2012), Jeanne and Korinek (2013), Chari and Kehoe (2016) and Faria-e-Castro (2016) do take into consideration moral hazard.

Given that I am assuming a completely unexpected aggregate shock to the economy, my model is not appropriate to study any amplification effects of the financial sector during crisis periods. Two such papers that study the effect of household default decisions during the Great Recession on the cost of the funding of banks through a financial accelerator channel are Faria-e-Castro (2016) and Greenwald (2016). Other papers that study fiscal multipliers in New Keynesian settings are Drautzburg and Uhlig (2015) and Auerbach and Gorodnichenko (2012).

1.3 The Mechanism

This section illustrates the underlying trade-off between bailouts and mortgage refinancing subsidies through a simplified example, emphasizing the role of the dead-weight loss of foreclosures and the information friction on house prices. To isolate the mechanism behind the trade-off, this example abstracts from many of the determinants of mortgage default, which are later considered in the quantitative model in Section 2.3.

Assume there are two households, denoted H1 and H2, and one bank in the economy. Each household has a house worth $100 and an outstanding mortgage of $100 on the house. Mortgage contracts are short-term and last only for one period, default implies losing the collateral and there is no recourse in the economy. That is, households will default on current mortgages if and only if they are underwater -they owe more than the
value of their house-, in which case they lose their house to the bank. Assume the bank, which is the mortgage holder, has liabilities of $200 and its only assets are the outstanding mortgages of the households. If there are no shocks to house prices, households will repay their debt and the bank will break even. Finally, assume that the government knows the distribution of house prices, but does not observe the price of each individual house.

Suppose there is a completely unexpected one-time shock to house prices that lowers the price of the house owned by $H_2$ to $80, while the price of $H_1$ remains at $100. After the shock, the house owned by $H_2$ is worth less than the mortgage debt, so $H_2$ optimally defaults and its house is foreclosed upon. Given the 20% price discount on foreclosed housing, the bank only recovers $64 out of the $80 worth the house of $H_2$. Household $H_1$ will repay its debt, as its house price is still worth $100. The bank is now insolvent, as its $164 revenues are not enough to cover the $200 worth of liabilities.

To preserve the solvency of the bank, the government can choose either of two policies. On one hand, the government can offer subsidies to households to prevent default. Given that the government knows the distribution of house prices, it knows that the household that is underwater would need $20 to find it optimal to repay its mortgage. However, the government does not observe individual house prices so, if it offers subsidies to mortgage repayments, both households will claim to be underwater to receive the $20. Therefore, the subsidy policy to prevent the default of $H_2$ would cost $40 and would preserve the solvency of the bank, but the government will have to levy $20 with distortionary taxation to pay to strategic defaulters -household $H_1$-. On the other hand, the government could let $H_2$ default and offer a bailout to the bank. In this case, the economy would assume the cost of the dead-weight loss generated by the foreclosure, equal to $16. The cost of the dead-weight loss is smaller than the subsidies given to strategic defaulters, so a bailout policy would be welfare maximizing. Figure 2 illustrates the effects of either policy.

Now, assume there are, instead, two households of the type $H_2$ and one household of the type $H_1$ described above. A subsidy policy that prevents default would have to
offer $20 to each household in the economy. The cost of such a policy would be $60, out of which $20 correspond to subsidies to strategic defaulters. Otherwise, the government could offer a bailout for each of the two foreclosed houses, and the economy would pay the dead-weight loss, equal to $32. In this case, the dead-weight loss is larger than the amount necessary to subsidize strategic defaulters, so a subsidy policy is preferable.

![Diagram showing subsidy and bailout policies](image)

(a) Subsidy Policy  
(b) Bailout Policy

Figure 2: Subsidy vs. Bailout policy

The welfare maximizing policy depends on the size of strategic default, which corresponds to the proportion of households of type H₁ versus H₂ in the example. If the economy is populated with a large number of households of type H₂, who would default absent any subsidy program, the cost of the information friction is low and a subsidy policy is optimal. If, in contrast, the proportion of H₁ households is large, a subsidy policy will spend a large amount on strategic defaulters. In this case, it is optimal to bail out the banks.
1.4 Quantitative Model

In this section, I introduce a heterogeneous agents’, life-cycle model with housing and mortgage markets. Section 1.4.1 describes the economy and defines the problem of the households, Section 1.4.2 describes the problem of the mortgage holders, Section 1.4.3 states the problem of production firms, and Section 1.4.4 defines the problem of the government. Later sections describe a baseline calibration of this economy to the pre-crisis U.S. economy and perform counterfactual experiments to assess the optimal policy.

1.4.1 Household’s Problem

Demographics

The economy is composed of a continuum of households of constant size. Households live for at most \( T \) periods, after which they die with probability equal to 1. At age \( t \in \{1, \ldots, T\} \), the individual faces an exogenous probability \( \pi_t \) of surviving to the next period. By assumption, \( \pi_T = 0 \). By the law of large numbers, \( \pi_t \) not only is the probability of surviving to the next period, but also the effective amount of people that survive to age \( t + 1 \). I assume there is no population growth and every period a cohort of newborns of size 1 enters the economy, so at every point in time the total population size remains constant at \( \left( 1 + \sum_{t=1}^{T} \prod_{j=1}^{t} \pi_j \right) \).

Preferences

Households derive utility according to a period utility function \( u(c, s, l) \) from non-durable consumption \( c \), housing services \( s \), and labor \( l \). Non-durable consumption is the numeraire good in the economy, and housing can be owned or rented at market-determined prices \( P_h \) and \( q \), respectively. Labor is supplied on a labor market with a competitively determined
wage $w$ per effective unit of labor supplied. Individuals discount the future according to a
discount factor $\beta$ and maximize their expected lifetime utility $E \sum_{t=1}^{T} \beta^t u(c, h, l)$.

**Income Dynamics**

Every period, the labor productivity of an individual of age $t$ is the product of two
components: an idiosyncratic shock $e_t \in \mathcal{E}$, and a deterministic age-dependent productivity
$\bar{e}_t$, so that, given an amount of labor supplied $l$, the effective units of labor supplied in the
market are: $e_t \bar{e}_t l$. As in Storesletten et al. (2004), the idiosyncratic productivity shock $e_t$
follows an autoregressive process, given by:

$$\log(e_{t+1}) = \rho_e \log(e_t) + \sigma_e \epsilon_{t+1}, \quad \epsilon_t \sim N(0, 1)$$

where $\rho_e < 1$ is the persistence of the productivity process, and $\sigma_e$ is the standard deviation
of the innovations. The AR(1) process is approximated by a 3-state Markov process with
transition probability matrix denoted as $F_e(e_{t+1} | e_t)$, using the procedure described in
Tauchen (1986). The age-dependent productivity $\bar{e}_t$ follows an increasing path over the first
periods of the cycle, followed by a decreasing path up to age 65 after which individuals
retire, as documented by Hansen (1993).

Labor income is subject to a marginal income tax $\tau_l$ levied by the government to pay
for government expenditures. In addition, the government levies social security taxes
$\tau_{ss}$, so total after-tax income is equal to $(1 - \tau_l - \tau_{ss})e_t \bar{e}_t w l$. Taxation is distortionary, so
higher government expenditures will generate larger distortions in the economy. After age
65, individuals retire and receive social security benefits $b$ set in equilibrium through a
pay-as-you-go social security system.
Housing

There is a perfectly inelastic supply of housing $H$ in the economy available for renting and owning at market-determined prices $q$ and $P_h$, respectively. As in Jeske et al. (2013) and Elenev et al. (2016), I separate the investment and consumption motives of housing and assume that only rental housing can be used for consumption as housing services $s$, so there is no owner-occupied housing. Households decide to own housing $h$ only for investment purposes and receive benefits through the returns of renting out their property.

The set of available houses for owning and renting is given by $H := [0, \bar{h}]$. I assume that home-owners can choose to rent any amount $s \in H$, potentially different from the amount owned $h \in H$. If households own more than what they rent, $h > s$, I will denote them net owners, whereas those that rent more than they own, $s > h$, are net renters.

Every period, owned housing is subject to an idiosyncratic depreciation shock $\delta \in \Delta := [\delta, \bar{\delta}]$, which is distributed according to a cumulative distribution function $F_\delta(\delta)$ so home-owners face an idiosyncratic shock to their house value. I assume $\delta < 0 < \bar{\delta}$ so there is idiosyncratic risk of having an increase or fall in the house value. This assumption embodies the uninsurable idiosyncratic house price risk that home-owners face (Case and Shiller, 1989; Jeske et al., 2013).

At the end of every period, in order to have a constant housing supply equal to $H$, I assume that the homeowner must pay for the house depreciation $\delta P_h h$ whenever $\delta > 0$, and receives an additional income equal to $|\delta P_h h|$ whenever $\delta < 0$.

The idiosyncratic depreciation $\delta$ is private information for the household and is only partially observed by the government, motivated by Mayer et al. (2014) and Foote et al. (2008). Assuming that the household draws $\delta_0 \in \Delta$, the government observes $\delta_0$ with probability $1 - p$, and $\delta \in \Delta \setminus \{\delta_0\}$ with probability $p$. Given that the decision to default depends on $\delta$, the government does not perfectly observe which households find it optimal to default. In particular, given that all the other state variables of the household are
observable by the government, with probability $1 - p$ the government correctly observes the decision to default of an individual with state variables $(t, e, a, h, m)$, while with probability $p$ the government potentially observes a different default decision.

**Financial Assets and Mortgage Markets**

Households can buy one-period risk-free bonds $a_t$ on financial markets at a price $P_a = 1/(1 + r_f)$, where $r_f$ is the risk-free rate of return set in equilibrium. In addition to risk-free bonds, individuals have access to collateralized long-term mortgages to buy $h$ units of housing. A mortgage of size $m$ that pledges $h$ units of housing as collateral, given to a household with productivity $e$, age $t$ and risk-free bonds $a$ is a contract that delivers $P_m(t, e, a, h, m)m$ in the current period and requires a payment equal to $m$ every period in the future. I assume that every period the mortgage debt disappears with probability $1 - \rho$, so expected mortgage payments follow a decreasing path over time and last for a finite number of periods on average (Chatterjee and Eyigungor, 2015). The loan-by-loan pricing function $P_m(\cdot)$ is described in section 1.4.2, where the problem of the mortgage originators is fully characterized. Issuing a mortgage of size $m$ has a one-time cost $F_{\text{issue}} \cdot \left( \sum_{j=t}^{T} \left[ \prod_{i=t}^{j} \pi_i \right] \left[ \frac{\rho}{1 + \rho} \right]^{j-t} \right) m$, equal to a fraction $F_{\text{issue}}$ of the total outstanding debt given the existence of search costs in mortgage origination (Hurst and Stafford, 2004). Finally, individuals face a Loan-To-Value constraint at origination, such that $\left( \sum_{j=t}^{T} \left[ \prod_{i=t}^{j} \pi_i \right] \left[ \frac{\rho}{1 + \rho} \right]^{j-t} \right) m/P_h h < LTV$. Equivalently, the minimum downpayment required to issue a mortgage is set to $1 - LTV$.

Individuals have the option to default on their mortgage debt after the idiosyncratic depreciation shock $\delta$ is realized, subject to losing the property pledged as collateral. In addition to losing the collateral, individuals cannot access the mortgage market in the period of default but can re-enter the market in any future period. Although in the U.S. there is recourse in most States (Ghent and Kudlyak, 2011), in which case households lose also part of their risk-free assets to cover the difference between the outstanding debt and
the house value, I assume there is no recourse in the economy and leave this for future work. Households also have the option to refinance their mortgage debt. Refinancing a mortgage is equivalent to issuing a new mortgage and using its proceeds to repay the outstanding debt of the original mortgage. This means that households that choose to refinance their mortgage have to pay the proportional fixed cost \( F_{\text{ref}} \cdot \left( \sum_{j=t}^{T} \left[ \prod_{i=t}^{j} \tau_i \right] \left[ \frac{r}{1+r} \right]^{j-t} \right) m' \) over the newly issued mortgage \( m' \). Note that the cost of issuing a new mortgage \( F_{\text{issue}} \) is potentially different from the cost of refinancing an existing mortgage \( F_{\text{ref}} \).

**Government Policies**

In steady state, the government plays a passive role in the economy and only manages the social security system: it levies social security taxes \( \tau_{ss} \) and distributes a pension \( b \) to retired households. Whenever there is an unexpected aggregate shock, or a “crisis period”, I assume that two policy instruments become available to the government: 1) distribute subsidies to mortgage refinancing and 2) offer bailouts to mortgage holders. The motivation for this assumption is that historically these instruments have become available to the governments after Congress approval during severe crisis periods like the Great Depression and the Great Recession. During the Great Recession, these instruments became available through TARP and HAMP. In contrast, during “normal” times - that is, in the steady state - governments do not offer bailouts to banks nor implement long-term policies to favor mortgage refinancing.

This section describes the policy instruments available to the government during crisis periods, while the optimization problem the government solves is postponed to Section 1.4.4.

First, to prevent foreclosures the government subsidizes mortgage refinancing at a rate \( \tau \), which means that households that choose to refinance have to pay only \((1 - \tau)F_{\text{ref}} \cdot \left( \sum_{j=t}^{T} \left[ \prod_{i=t}^{j} \tau_i \right] \left[ \frac{r}{1+r} \right]^{j-t} \right) m' \) on the new mortgage issued \( m' \). As in the Emergency Economic
Stabilization Act, the subsidy is contingent on the mortgage refinancing being executed, to avoid households claiming the subsidy and defaulting on the same period. The purpose of this subsidy is to reduce the number of mortgage defaults by reducing the relative cost of refinancing.

Subsidies are given according to an eligibility rule chosen by the government every period. The eligibility rule potentially depends on all the state variables observed by the government, including the idiosyncratic house price shock $\delta$. That is, an eligibility rule is a function $\Gamma: \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \rightarrow \{0, 1\}$ that determines which households receive subsidies, where $\Delta$ is the idiosyncratic depreciation observed by the government, which need not be equal to the realized depreciation of the household (recall the information friction described in section 1.4.1). This means that if the government implements an eligibility rule that explicitly targets particular values of observed $\delta$ to receive the subsidy, it will potentially make mistakes with probability $p$: households whose realized $\delta$ makes them eligible might not receive the subsidy, while households whose realized $\delta$ makes them non-eligible might receive it. As with the eligibility rule, the mortgage refinancing subsidy can also differ across the distribution of households so $\tau: \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \rightarrow [0, 1]$.

Second, the government offers bailouts $B$ to the mortgage holders to cover for their losses whenever there is an unexpected rise in foreclosures. Bailouts are pure lump-sum transfers. Henceforth, I will refer to the set of government policies as $\Theta: \{\tau(\cdot), \Gamma(\cdot), B, \tau_l\}$.

**Household’s Problem**

Every period, given the individual state variables $(t, e, a, h, m, \delta)$, aggregate state variables denoted by $\Omega$, government policies $\Theta$, a pricing function for mortgages $P_m(t, e, a, h, m; \Omega, \Theta)$, and prices $P_h, q, r_f$ and $w$, households choose the amount of savings to take on to the next period $a'$, housing to own $h'$, housing services to rent $s$, consumption $c$, mortgage units to
acquire $m'$ and whether to default, keep or refinance their current mortgage. The recursive formulation of the problem of the household is defined by the following equation:

$$V(t,e,a,h,m,\delta;\Omega,\Theta) = \max\{V^{keep}(t,e,a,h,m,\delta;\Omega,\Theta), V^{def}(t,e,a,h,m,\delta;\Omega,\Theta), E_p V^{ref}(t,e,a,h,m,\delta;\Omega,\Theta)\}$$

(2)

where $V^{keep}$, $V^{default}$ and $V^{refinance}$ are the value functions associated with keeping, defaulting and refinancing the current mortgage, respectively. The expectation $E_p$ on the decision to refinance depends on the partial observability of $\delta$ by the government and, therefore, the probability of mistakes in subsidy assignment $p$. A household that is eligible for the refinance subsidy and chooses to refinance might not end up receiving it, whereas a household that is not eligible might end up receiving it. This will become clear below. The value function associated with keeping the current mortgage is described by the following problem:

$$V^{keep}(t,e,a,h,m,\delta;\Omega,\Theta) = \max_{c,s,h',a' \geq 0, l \in [0,1]} u(c,s) + \pi_l \beta E_{e',\delta',m'} V(t+1,e',a',h,m',\delta';\Omega',\Theta'), \text{ s.t.}$$

(3)

$$c + m + qs + P_a a' + \frac{\delta h w}{A} = (1 - \tau_l - \tau_{ss}) e\bar{e} w + a + qh$$

$$m' = \begin{cases} m & \text{w.p. } \rho \\ 0 & \text{w.p. } 1 - \rho \end{cases}$$

$$\delta' \sim F_\delta(\delta') , \quad e' \sim F_e(e'|e), \quad \Omega' = G(\Omega), \quad V^{keep}(T+1,\cdot) = 0$$
where $G$ is the law of motion of the aggregate state variables. If the household decides to keep the current mortgage, the payments are equal to $m$ and the mortgage amount will remain outstanding in $t + 1$ with probability $\rho$. With probability $1 - \rho$ the mortgage debt is reset to zero. The last term on the left-hand side of the budget constraint is the cost of the house depreciation $\delta P_h h$, which must be covered by the home-owner at the end of every period. Note that when $\delta < 0$, I am assuming that the household receives an additional income. Given the assumption that households live for at most $T$ periods, I normalize the value at age $T + 1$ to be equal to zero so the final period problem is static. The value function for households that default is described by:

$$
V_{\text{def}}(t, e, a, h, m, \delta; \Omega, \Theta) = \max_{c, s, h', a', m' \geq 0, \ell \in [0, 1]} u(c, s) + \tau l \beta E_{e', \delta', m', \rho} V(t + 1, e', a', 0, 0, \delta'; \Omega', \Theta') \quad \text{s.t.}
$$

\begin{align*}
\quad & c + qs + P_a a' = (1 - \tau l - \tau ss) e\bar{\delta} w + a \\
\quad & \delta' \sim F_{\delta}(\delta'), \quad e' \sim F_{e}(e'|e), \quad \Omega' = G(\Omega), \quad V_{\text{def}}(T + 1, \cdot) = 0
\end{align*}

(4)

Households that choose to default do not have to make mortgage payments and lose the house pledged as collateral before paying for the idiosyncratic depreciation, so larger values of $\delta$ will make more attractive the option to default. Finally, a household that chooses to refinance faces uncertainty on whether the government will make mistakes in mortgage assignation, so long as $p > 0$. The expected value for a household that chooses to refinance is given by:

$$
\mathbb{E}_p V_{\text{ref}}(t, e, a, h, m, \delta; \Omega, \Theta) = (1 - p)V_{\text{ref}, 1-p}(t, e, a, h, m, \delta; \Omega, \Theta) + pV_{\text{ref}, p}(t, e, a, h, m, \delta; \Omega, \Theta)
$$
The value function $V^{ref,1-p}$ for households that decide to refinance their mortgage and receive the correct mortgage subsidy is:

$$V^{ref,1-p}(t, e, a, h, m, \delta; \Omega, \Theta) = \max_{c,s,h',a',m',\delta'; \Omega', \Theta'} u(c, s) + \pi_t \hat{E}_{e',a',h'} V(t+1, e', a', h', m', \delta'; \Omega', \Theta'),$$

(5)

$$c + \left( \sum_{j=t}^{T} \left[ \Pi_{i=j}^{T} \tau_i \right] \left( \frac{\rho}{1+r} \right)^{j-t} \right) \left( m + (1 - \Gamma(\cdot)\tau(\cdot))Fm' \right) + qs + P_h h' + P_a a' =$$

$$(1 - \tau_l - \tau_{ss})e^lwl + a + P_h h - \frac{\delta P_h h'w}{A} + qh' + P_m(t, e, a, h', m'; \Omega, \Theta)m'$$

$$\hat{m}' = \begin{cases} m' & \text{w.p. } \rho \\ 0 & \text{w.p. } 1 - \rho \end{cases}$$

$$F = \begin{cases} F^{issue} & \text{if } m = 0, m' > 0 \\ F^{ref} & \text{if } m > 0, m' > 0 \\ 0 & \text{if } m' = 0 \end{cases}$$

$$\delta' \sim F(\delta), \quad e' \sim F(e|e), \quad \Omega' = G(\Omega), \quad V^{ref}(T+1, \cdot) = 0$$

$$\left( \sum_{j=t}^{T} \left[ \Pi_{i=j}^{T} \tau_i \right] \left( \frac{\rho}{1+r} \right)^{j-t} \right) m'/P_h h' < \text{LTV}$$

Households that decide to refinance have to repay the net present value of the outstanding mortgage debt, given by the second term on the budget constraint. Moreover, the household has to pay an after-subsidy fixed cost $\left( \sum_{j=t}^{T} \left[ \Pi_{i=j}^{T} \tau_i \right] \left( \frac{\rho}{1+r} \right)^{j-t} \right) (1 - \Gamma(\cdot)\tau(\cdot))Fm'$ on the new mortgage $m'$: if the household is eligible, $\Gamma(t, e, a, h, m, \delta) = 1$, there is a subsidy equal to $\tau(t, e, a, h, m, \delta)$ on the cost of mortgage refinancing. Otherwise, the household has to pay the full amount. Moreover, the fraction $F$ depends on whether the household is issuing a new mortgage ($F^{issue}$) or refinancing a current mortgage ($F^{ref}$). Households that choose to prepay the outstanding debt ($m' = 0$) have to pay no fixed cost (so $F = 0$). The last term on the budget constraint, $P_m(t, e, a', h', m'; \Omega, \Theta)m'$, is the amount received
for the new mortgage $m'$. The last constraint is a Loan-To-Value constraint. The value function $V_{ref,p}$ for households that decide to refinance their mortgage and do not receive the correct mortgage subsidy is:

$$V_{ref,1-p}(t,e,a,h,m,\delta;\Omega,\Theta) = \max_{c,s}\beta \mathbb{E}_{e'|c,e',m'} V(t+1,e',a',h',\tilde{m'},\delta';\Omega',\Theta'),$$

(6)

$$c + \left( \sum_{j=t}^{T} \left[ \prod_{i=t}^{j-1} \tau_i \right] \left( \frac{\rho}{1+r} \right)^{j-t} \right) (m + (1 - (1 - \Gamma(\cdot))\tau(\cdot))Fm') + qs + P_ah' + P_ad' =$$

$$(1 - \tau_l - \tau_ss)\tilde{e}lwl + a + P_nh - \frac{\delta P_hw}{A} + qh' + P_m(t,e',a',h',m';\Omega,\Theta)m'$$

$$\tilde{m'} = \begin{cases} 
  m' & \text{w.p. } \rho \\
  0 & \text{w.p. } 1 - \rho
\end{cases}$$

$$F = \begin{cases} 
  \text{issue} & \text{if } m = 0, m' > 0 \\
  \text{ref} & \text{if } m > 0, m' > 0 \\
  0 & \text{if } m' = 0
\end{cases}$$

$$\delta' \sim F_\delta(\delta'), \quad e' \sim F_e(e'|e), \quad \Omega' = G(\Omega), \quad V_{ref}(T+1,\cdot) = 0$$

$$\left( \sum_{j=t}^{T} \left[ \prod_{i=t}^{j-1} \tau_i \right] \left[ \frac{\rho}{1+r} \right]^{j-t} \right) m'/P_nh' < \text{LTV}$$

Note that the only difference with equation (5) is that, in the case of a mistake, households that receive the subsidy are those for which $\Gamma(t,e,a,h,m,\delta) = 0$. In case of a mistake, eligible households do not receive any subsidy, and ineligible households do receive it. In an economy with perfect information $p = 0$, the household faces no uncertainty on mortgage refinancing subsidy assignment.

The following proposition proves some results on the value function for households in this economy:
**Proposition 1.** Assume \( u(c, s) \) is continuous, strictly increasing and strictly concave. Then \( V \) is:

1. Continuous
2. Increasing on \( a \) and \( h \)
3. Decreasing on \( m \) and \( \delta \)

If the productivity persistence parameter \( \rho_e > 0 \) then \( V \) is also increasing on \( e \).

*Proof.* The proof can be found in the appendix. \qed

Because of the discrete choice between keeping, refinancing and defaulting on the current mortgage, the value function \( V \) need not be concave nor differentiable.

### 1.4.2 Mortgage Originators

I assume that in this economy there is no secondary mortgage market, so mortgages are held and serviced by originators. In this way, I assume that mortgage originators are also the owners of each mortgage, which will let me abstract from any possible mismatch of incentives for mortgage refinancing between mortgage investors and servicers. I will refer to them as mortgage investors, mortgage holders or mortgage originators interchangeably throughout the rest of the paper.

There is a continuum of size 1 of mortgage originators, so mortgages are supplied in a competitive market. Mortgage origination firms are owned by all households, so every period each household receives a lump-sum transfer equal to the profits of the firms. Competition is present loan-by-loan, so the origination of each loan is subject to a zero expected profit condition. The pricing function satisfies the following equality:
\[ P_m(t, e', a', h', m'; \Omega, \Theta) m' = \left( \frac{\pi_t \rho}{1 + r_f} \right) E_{\delta'} \left[ d(s'; \Omega, \Theta) \left[ (1 - \Psi) P_h(1 - \delta')h' - \delta' P_h h' \right] + \right. \\
\left. \left( 1 - d(s'; \Omega, \Theta) \right) \left( 1 - s(s'; \Omega, \Theta) \right) \left( m' + P_m(t + 1, e', a'', h', m'; \Omega', \Theta') m' \right) \right] \right) 
\]

where \( s' = (t + 1, e', a', h', m') \) denotes individual state variables, \( d(\cdot) \) is the policy function for default, that takes the value of one if the household chooses to default and zero otherwise, and \( s(\cdot) \) is the policy function for mortgage refinancing. The pricing function \( P_m(t, e, a', h', m'; \Omega, \Theta) \) is such that the outlays of the bank at \( t \) are equal to the expected value of the assets of the bank in \( t + 1 \) in case the household chooses to: a) default, b) refinance, or c) keep the current mortgage. If the household defaults, the bank receives the after-depreciation value of the house at a price discount \( \Psi \). I assume that, in order to keep the economy’s housing supply constant, in case of default the bank has to pay for the depreciation of the house, represented by the second term on the value of default to the bank. If the household chooses to refinance, the bank receives the total outstanding debt, which is the net present value of future mortgage payments. If the household chooses to keep the current mortgage, the bank receives the payment \( m' \) and holds an asset worth \( P_m(t + 1, e', a'', h', m'; \Omega', \Theta') m' \) at the end of the period. Note that period \( t + 1 \) is discounted taking into account the probability \( \rho \) that the mortgage is outstanding at \( t + 1 \) as well as the survival probability of the household \( \pi_t \).

In steady state, equation (7) holds with equality ex-post by the law of large numbers. The outlays of the mortgage holder on every loan are exactly equal to the realized net
present value of payments/assets whenever there are no aggregate unexpected shocks. Note, however, that equation (7) need not hold with equality ex-post whenever there is an aggregate unexpected shock to $\Omega$. An unexpected shock to $\Omega$ will potentially change the default and refinance decisions of households, as well as house prices, so the realized net present value of assets to mortgage investors need not equal the outlays of the previous period. Whenever this happens, the government will ensure that the equation holds ex-post.

1.4.3 Production Firms’ Problem

Final good production firms produce according to a simple linear technology $f(L) = AL$, where only labor is used. Firms produce on a perfectly competitive market, so market wages per efficiency unit of labor are directly pinned down by the technology factor $A$, such that in equilibrium $w = A$.

1.4.4 Government’s Problem

In steady state or, equivalently, whenever there are no aggregate unexpected shocks to the economy ($\Omega$), the government only manages the social security system: levies social security taxes $\tau_{ss}$ to working-age households and pays social security proceedings $b$ to retired households.

Whenever an aggregate unexpected shock hits the economy, two instruments become available to the government: subsidies to mortgage refinancing $\tau$ and bailouts to mortgage holders $B$. These instruments are available only during the period of the unexpected shock and disappear thereafter. The government chooses optimally these instruments to maximize a utilitarian welfare function, subject to keeping the ex-post profits of mortgage originators equal to zero.
In order to finance the expenditures of these policy instruments, the government issues long-term debt worth $A$ after the unexpected shock, in the form of a consol bond that pays a coupon $A \cdot r$ in every period in the future. To pay the coupon every period, the government levies labor income taxes $\tau_i$. At period $j$, the problem of the government is:

$$\max_{\tau(\cdot), \Gamma(\cdot), B, \tau_i} \int_{s'} V(s'; \Omega_j, \tau(\cdot), \Gamma(\cdot), B, \tau_i) d\Phi_j(s'), \quad \text{s.t.}$$

$$\int_s P_m(s_{-\delta}, \Omega_{j-1}, \Theta_{j-1}) m' d\Phi_{j-1}(s) =$$

$$\int_s \left( \frac{\pi_1 \rho}{1 + r_f} \right) \left[ d(s'; \Omega_j, \tau(\cdot), \Gamma(\cdot), B, \tau_i) \left[ (1 - \Psi) P_h (1 - \delta') h' - \delta' P_h h' \right] + s(s'; \Omega_j, \tau(\cdot), \Gamma(\cdot), B, \tau_i) \left( \sum_{j=t+1}^{T} \left[ \Pi_{i=t+1}^{j} \tau_i \right] \left( \frac{\rho}{1 + r} \right)^{j-t} \right) m' + \right.$$  

$$(1 - d(\cdot)) (1 - s(\cdot)) (m' + P_m(s''_{-\delta}, \Omega_j, \tau(\cdot), \Gamma(\cdot), B, \tau_i) m') d\Phi_j(s') + B$$  

(Ex-Post Solvency)

$$(1 - p) \int_{s'} \Gamma(s') \tau(s') \left[ F \cdot \left( \sum_{j=t}^{T} \left[ \Pi_{i=j}^{j} \tau_i \right] \left[ \frac{\rho}{1 + r} \right]^{j-t} \right) m' \right] s(\cdot) d\Phi_j(s') +$$

$$p \int_{s'} (1 - \Gamma(s')) \tau(s') \left[ F \cdot \left( \sum_{j=t}^{T} \left[ \Pi_{i=t}^{j} \tau_i \right] \left[ \frac{\rho}{1 + r} \right]^{j-t} \right) m' \right] s(\cdot) d\Phi_j(s') + B = A$$  

(Bud. Bal. $j$)

$$A \cdot r_i = \int_s \tau_i e_{i, j} w d\Phi_j(s'), \quad \forall i > j$$  

(Bud. Bal. $i > j$)

$$b \cdot \int_{s'} 1 \{ t \geq t^{red} \} d\Phi_j(s') = \int_{s'} 1 \{ t < t^{red} \} \tau_{s} e_{i, j} w l \ d\Phi_j(s')$$  

(Social Security)

where $j$ stands for the period of the aggregate unexpected shock, $j - 1$ for the previous period, and individual state variables are summarized by $s = (t, e, a', h', m', \delta)$, $s' = (t + 1, e', a', h', m', \delta')$, $s''_{-\delta} = (t + 1, e', a', h', m')$ and $s'_{-\delta} = (t + 1, e', a', h', m')$. Note that I explicitly denoted the dependence of the value and policy functions on the instruments of the government $\tau(\cdot), \Gamma(\cdot), B, \tau_i$, rather than with $\Theta$. The ex-post solvency condition is the aggregation of the zero-profit condition for the mortgage pricing function (equation 7)
overall mortgage contracts in the economy at period $j$. The left-hand side of this equation is the value of the liabilities of mortgage originators, which equal the outlays in period $j - 1$, whereas the right-hand side is the value of the bank’s assets in period $j$ plus the bailout transferred to the bank. The budget balance condition at period $j$ states that the total amount of government mortgage refinancing subsidies plus bailouts to mortgage holders must equal the amount of consol bond issued. Note that if $p > 0$, the government might distribute subsidies to households that are not eligible ($1 - \Gamma(\cdot) = 1$). The budget balance condition at $i > j$ states that the government must finance the coupon payments of the consol bonds with labor income taxes. Lastly, the social security restriction describes a balanced budget constraint in the social security system.

Note that the government can achieve the ex-post efficiency constraint through two channels: 1) by choosing $B$, such that the first restriction holds (Ex-Post Solvency), or 2) by choosing $\tau(\cdot), \Gamma(\cdot)$ and $\tau_l$ such that households change their optimal choices and the equality holds.

### 1.5 Recursive Competitive Equilibrium

In this section, I state the definition of an equilibrium in this environment. I allow for the possibility of aggregate unexpected shocks from happening, so all functions and equilibrium prices depend on the aggregate states of the economy $\Omega$, as well as on government policies $\Theta$ which need not be trivial when an aggregate shocks occur. Index $j$ denotes the period.

**Definition 1** (Recursive Competitive Equilibrium). A recursive competitive equilibrium are a value function $V : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \times \{\Omega_j, \Theta\} \rightarrow \mathbb{R}$, policy functions for default and refinancing $d, s : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \times \{\Omega_j, \Theta\} \rightarrow \{0, 1\}$, and all other policy functions $g_{cr}, g_{mr}, g_{hr}, g_{rr}, g_{a} : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \times \{\Omega_j, \Theta\} \rightarrow \mathbb{R}$, for the household, a pricing function $P_m : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \times \{\Omega_j, \Theta\} \rightarrow \mathbb{R}$.
\(\mathcal{M} \times \{\Omega_j, \Theta\} \longrightarrow \mathbb{R}_+\) for mortgages, government policies \(\Theta(\Omega_{j-1}, \Omega_j)\), a price for rental housing \(q \in \mathbb{R}_+\), a price for new housing \(P_h \in \mathbb{R}_+\), a wage \(w \in \mathbb{R}_+\), a risk-free interest rate \(r \in \mathbb{R}_+\), an aggregate law of motion \(G(\Omega)\) and distributions \(\Phi : \{1, \ldots, T\} \times \mathcal{E} \times \mathcal{A} \times \mathcal{H} \times \mathcal{M} \times \Delta \times \{\Omega_j, \Theta\} \longrightarrow [0, 1]\) such that:

1. **[Households Optimize]** Given prices \(P_m(\cdot), P_h, q, w\) and \(r\), aggregate states \(\Omega\), government policy functions \(\Theta\) and the aggregate law of motion \(G\) the value function \(V\) solves the household’s problem (2), with \(d, s, g\_\text{keep}, g\_c, g\_m, g\_h, g\_r\) and \(g\_a\) the respective policy functions.

2. **[Mortgage Originators Optimize]** Given \(P_h, q, w\) and \(r\), aggregate states \(\Omega\), government policy functions \(\Theta\) and the aggregate law of motion \(G\), the pricing function \(P_m(\cdot)\) for mortgages satisfies an expected zero-profit condition loan by loan, given by equation (7). Any profits are transferred in a lump-sum fashion to households, who own these firms.

3. **[Rental Market Clears]** The rental housing price \(q\) is such that demand for rental housing is equal to the total supply of housing in the economy:

\[
\int_s g_r(s) d\Phi(s) = \bar{H}
\]

4. **[Housing Market Clears]** The housing price \(P_h\) is such that home-ownership is equal to total housing supply:

\[
\int_s g_h(s) d\Phi(s) = \bar{H}
\]

5. **[Financial Assets Market Clears]** The risk-free rate \(r\) is such that savings in the economy equal total mortgage outlays by mortgage investors:

\[
\int_s g_a(s) d\Phi(s) = \int_s P_m(s) g_m(s) d\Phi(s)
\]
6. **[Production Firms Optimize]** The wage per efficiency unit of labor \( w \) is pinned down by the technology parameter:

\[
    w = A
\]

7. **[Government Optimizes]** If \( \Omega' = G(\Omega) \), the government only manages social security and the pension \( b \) given to retired households is such that there is budget balance on the social security system:

\[
    b \cdot \int_s \mathbb{1}\{t \geq t_{ret}\} d\Phi(s) = \int_s \mathbb{1}\{t < t_{ret}\} \tau_{ss} e_i w g_i(s) d\Phi(s)
\]

If \( \Omega' \neq G(\Omega) \) - that is, after an aggregate shock - the government solves problem 8.

8. **[Resource Constraint]** The resource constraint holds:

\[
    A \cdot \int_s e_i g_i(s) d\Phi(s) = \int_s \left[ g_c(s) + \delta P_h h + F \left( \sum_{j=1}^{T} \left[ \Pi_{i=1}^{j} \tau_i \right] \left( \frac{\rho}{1 + r} \right)^{j-t} \right) \right] g_m(s) d\Phi(s)
\]

9. **[Distribution Consistency]** The distribution \( \Phi(t, \cdot) \), for \( t \in \{2, \ldots, T\} \) is consistent with the policy functions for the households, the Markov transition probabilities for the idiosyncratic shocks, the aggregate law of motion \( G \) and the initial distribution \( \Phi(1, \cdot) \).

10. **[Aggregate Law of Motion Consistency]** The aggregate law of motion is consistent with the distribution \( \Phi \), policy functions for the households and the Markov transition probabilities for the idiosyncratic shocks.

    Agents in this economy assign zero probability to any aggregate shock and, therefore, do not have expectations with respect to the policies the government might implement during mortgage crises. For this reason, agents do not take overly risky behaviors in steady state, and the model completely abstracts from moral hazard.
Note that this definition of equilibrium allows for the possibility of aggregate shocks from happening. Definition 2 in the appendix defines the steady state equilibrium in this economy, where no aggregate shocks occur, so Definition 2 is a particular case of Definition 1.

1.6 Calibration

This section describes the functional forms and parameter values used to calibrate the model to the pre-crisis steady state U.S. economy. Details on the computation of the model are available in Section 1.2 in the appendix.

Demographics

I set $T = 30$, such that one period in my model corresponds to 2 years of life. Individuals enter the economy at the age of 20 ($t = 1$) and live up to age 80 ($t = 30$). This timing captures the life-cycle patterns of consumption and housing investments while keeping the computation of the model tractable. Every period, an individual of age $t$ survives to period $t + 1$ according to an exogenous probability $\pi_t$, which corresponds to the empirical survival probabilities from the Actuarial Life Tables of the Social Security Administration. I assume that individuals of age $T = 30$ (80 years old) die with probability exactly equal to one. I assume that households retire at age 65, which means that $t^{ret} = 22$ in my model.

Preferences

I use the following parameterization for the utility function:

$$
\sum_{t=1}^{T} \beta^t \pi_t \left[ \left( \psi e^x + (1 - \psi) h^x \right) \frac{1}{1 - \sigma} - \theta \frac{1^{1+\frac{1}{\eta}}}{1 + \frac{1}{\eta}} \right]
$$

8. The life table and survival probabilities can be found at this link.
Where the elasticity of substitution between consumption and housing is equal to \(1/(1 - \kappa)\). Fernández-Villaverde and Krueger (2011) make a review of the findings in the literature and argue that this elasticity varies according to the model specification and samples used for estimation. I set the value of \(\kappa\) to -0.1, which is close to the estimates presented in their review. The Frisch elasticity is given by \(\eta\). Following the macro estimates of the parameter \(\eta\) on life cycle models, I set \(1/\eta = 0.5\) (for a review on the macro and micro estimates, see Keane and Rogerson (2015)). Finally, I set \(\sigma = 2\) and \(\beta = 0.96\) which are standard in the literature, and the weight parameter of the disutility of labor \(\theta_l = 5\) to match an average labor supply equal to 0.34.

**Labor Productivity**

The productivity of households has two components: 1) an idiosyncratic shock, and 2) a life-cycle component. The idiosyncratic component follows the AR(1) process described by equation (1), where I set \(\rho_y = 0.95\) and \(\sigma_\epsilon = 0.22\), which correspond to 2-year values close to the estimates of Storesletten et al. (2004). In order to compute the model, I discretize the idiosyncratic productivity process with a 3-state Markov Chain approximation using the method by Tauchen (1986). I set \(m_y = 1\), so the points in the Markov Chain are at ±1 standard deviations from the mean, resulting in a grid for the stochastic productivity given by \{0.602, 1.0, 1.661\}, and a transition probability matrix:

\[
\begin{bmatrix}
0.9507 & 0.0492 & 0.0 \\
0.036 & 0.9272 & 0.036 \\
0.0 & 0.0492 & 0.9507
\end{bmatrix}
\]

The life-cycle component of productivity is set to match the estimates of Hansen (1993), so average productivity is hump-shaped over the life cycle.

---

9. \(\rho_y = 0.98^2\) and \(\sigma_\epsilon = 0.1 \times \sqrt{2}\).
Financial Markets

The parameter $\rho$, which corresponds to the rate at which mortgage debt survives over time, is chosen so the average duration of mortgage payments lasts 25 years, which in the current calibration corresponds to 12 periods. That is, I choose $\rho = 0.92$. I set the costs of issuing new debt and refinancing existing debt to $F_{\text{issue}} = 0.015$ and $F_{\text{ref}} = 0.025$, respectively, which fall inside the range of estimates by Hurst and Stafford (2004). This means that to issue new debt, the household must pay 1.5% of the value of the total outstanding debt, while to refinance it has to pay 2.5%. Lastly, as in Garriga and Hedlund (2016), a Loan-To-Value limit of $LTV = 1.25$ is assumed. In equilibrium, this limit will not be binding.

Table 1 summarizes the values of the parameters chosen for the initial steady state calibration.

Endogenously Calibrated Parameters

To make the computation feasible I set the idiosyncratic depreciation shock to take values on a three-point grid distributed uniformly over the interval $\delta \in [\underline{\delta}, \bar{\delta}]$. The values of $\underline{\delta}$ and $\bar{\delta}$ are endogenously calibrated so a) the foreclosure rate matches the two-year average of 3.0% during the pre-crisis period, 1991-2007, and the average homeownership rate matches 66.5% for the same period, corresponding to the estimates by the Federal Reserve Bank of St. Louis\textsuperscript{10}. The parameter $\psi$ is the weight of non-durable consumption with respect to overall consumption. According to the NIPA tables and the Bureau of Economic Analysis’ Personal Consumption Expenditure data, the expenditures on housing services account for around 14.1% – 15% of overall expenditures (Jeske et al., 2013; Corbae and Quintin, 2015). I set $\psi$

\textsuperscript{10} Estimates can be found on this link.
such that the ratio of nominal expenditures on housing services with respect to nondurable consumption is equal to 0.141. Table 2 shows the values of $\tilde{\delta}, \delta$ and $\psi$.

Given that the information problem arises when the government offers subsidies to mortgage refinancing, the calibration of the parameter $p$ is postponed to Section 1.7, where I calibrate the model to match the Great Recession and HAMP and TARP are implemented.
1.6.1 Steady State

Table 2 summarizes the moments of the baseline calibration in the initial steady state and the data.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Variable</th>
<th>Data</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Targeted Moments</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \delta )</td>
<td>0.119</td>
<td>Default rate</td>
<td>3.0%</td>
<td>2.4%</td>
</tr>
<tr>
<td>( \tilde{\delta} )</td>
<td>-0.345</td>
<td>Ownership rate</td>
<td>66.5%</td>
<td>65.5%</td>
</tr>
<tr>
<td>( \psi )</td>
<td>0.84</td>
<td>Rent/Cons expenditures</td>
<td>14.1%</td>
<td>14.2%</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Untargeted Moments</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Std. Dev. of ( \delta )</td>
<td>10 – 14.5%</td>
<td>13.4%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average home equity</td>
<td>62%</td>
<td>64.5%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Median size (sq ft.): own/rent</td>
<td>1.51</td>
<td>1.52</td>
</tr>
</tbody>
</table>

Table 2: Model and Data Moments

In addition to matching the targeted moments described above, the model matches untargeted moments related to the housing and mortgage markets. First, the model’s standard deviation of the idiosyncratic house price shock lies within the range estimated by the literature. As estimated by Giacoletti (2016), using the CoreLogic repeat-sales data from San Francisco, Los Angeles and San Diego, the standard deviation of the idiosyncratic component of house price risk over a period of up to two years is between 10% and 14.5%. The standard deviation that results from the endogenously chosen grid for idiosyncratic house price risk, which corresponds to a uniform distribution in the model, is equal to 13.4 %. Second, the model’s average home equity, equal to 64.5 %, is close to the 62% observed in the data. Furthermore, the home equity cumulative distribution function generated by the model, illustrated in Figure 3, closely matches the distribution obtained from the Survey of Consumer Finances 2007, as in Chatterjee and Eyigungor (2015). In the model, the lower tail of the distribution is heavier, but the rest of the distribution is close to the data. A potential explanation why the model does not closely match the left tail of the
distribution is that, in the model, I assume that households start their life with no assets or housing. In the data, there are households at age 20 that hold financial assets and housing mainly because they receive inter-vivos transfers and bequests, or because they start working before age 20. Figure 22 in the appendix illustrates the life-cycle averages of home equity in the model and the data and shows this mismatch at the beginning of the life-cycle. Finally, the model closely matches the ratio of the median size of owner-occupied versus rental housing, obtained by Chatterjee and Eyigungor (2015) from the American Community Survey of 2007.

![Graph showing home equity distribution](image)

**Figure 3: Home Equity Cumulative Distribution - Model vs. Data (SCF 2007)**

Figure 4a shows the steady state distribution of home-owners, according to their age and home equity. There is a positive correlation between age and equity, with a correlation coefficient equal to 0.56 in the model. As households grow over their life cycle and make their mortgage payments, they increase the percentage owned of their house and, therefore, their home equity. There is a mass of 36.8% of home-owners that have no mortgage debt, so their home equity is equal to 100%. These households have made all of their mortgage payments. Figure 4b illustrates the distribution of home-owners according to their savings and home equity. The correlation coefficient between both variables is only slightly positive (equal to 0.06), as both financial assets and housing are substitutes of each other. Some
households accumulate savings until they have enough assets to repay their mortgage debt, which increases their equity but reduces their financial assets.

Figure 4: State space distributions.

Figure 5 illustrates the behavior of net renters in my model, which are households that rent more than what they own. The percentage of net renters follows a decreasing path at the beginning of the life cycle, close to what happens with renters on the Survey of Consumer Finances of 2007. However, given that households know they are going to die with probability one on their last period of life, they sell all of their owned housing at the end of the cycle. This is opposite to what is observed in the data, as in my model there are no bequests and the probability of death is equal to one at age 80. This is not true in the data, because households have bequest motives, and individuals do not die with certainty at age 80. Figure 22 in the appendix illustrates other life-cycle averages of the model.

Default in Steady State

In the model, there is default in steady state. Households that choose to default lose their collateral, are left out of the mortgage market for the period of default, equal to two years, and have to pay the fixed cost if they want to issue a new mortgage in the future. Table 3
characterizes households that choose to default, according to their level of savings $a$ and the income and house value shocks during the period of default.\textsuperscript{11} Of the individuals that default, 100% do so after drawing a negative shock on their house value, corresponding to the largest depreciation $\delta = \bar{\delta}$. Within defaulters, there are individuals with and without savings. Of those households without savings, 79.61% choose to default after a negative income shock. These households find it impossible to continue repaying their mortgage, as this would imply negative levels of non-durable consumption during the period of the shock. In contrast, of the defaulters with positive savings, only 21.17% choose to default after a negative income shock. These households decide to default on their mortgage, even though their income and assets would allow them to continue making mortgage payments.

\begin{table}[h]
\centering
\begin{tabular}{lcc}
\hline
 & Without savings: $a = 0$ & With savings: $a > 0$ \\
\hline
House value shock & 100 % & 100 % \\
Income shock & 79.61 % & 21.17 % \\
Total default & 0.78 % & 1.7 % \\
\hline
\end{tabular}
\caption{\% of defaulters with income and price shocks on the period of default.}
\end{table}

\textsuperscript{11} For this exercise, an income shock occurs when the household draws the lowest possible productivity shock. A price shock occurs when the household draws the largest possible depreciation shock.
Panel (a) in Figure 6 illustrates the average default rate by age, as a percentage of home-owners that hold a mortgage in each age group. Default is largest among young households and decreases over the life-cycle. This happens because households have low levels of savings at the beginning of their life cycle. After a negative income shock these households cannot continue making mortgage payments, either because in the present their income is too low, or because, given the persistence of the shock, their future income will not be sufficient to repay the loan. In contrast, older households have savings that serve as a buffer in case of a negative income shock.

![Graph of default rate by age and home equity](image)

**Figure 6**: Model average default rate (%) (a) by age, and (b) home equity (%)

Panel (b) in Figure 6 illustrates the default rate by home equity. Default occurs only for households with negative equity, or underwater, and reaches 46% for households with sufficiently negative equity. Only 31.5% of households underwater choose to default. As the literature has pointed out, being underwater is a necessary, but not sufficient, condition for default. It is necessary because if the household is not underwater it can always choose to sell the house, repay the mortgage and obtain positive profit. This is not true when the household is underwater and the outstanding debt is larger than the value of the house. Being underwater is not a sufficient condition to default for the following reasons. First, there are fixed costs of mortgage origination, so a household that defaults today will have to pay a fixed cost to originate a new mortgage in the future. This might prevent if from
defaulting on the mortgage. Second, in my model mortgages are long-term contracts, whose terms are fixed at the moment of origination. A household that is underwater might prefer to keep the current contract whenever the terms of the mortgage, that were set at the origination period, are better than the expected terms of a new mortgage in the future. For example, before the crisis, a wealthy and productive household would originate a mortgage with prime interest rates. If in the future this household receives a low-probability, negative productivity shock that reduces its future credit-worthiness, such as a long unemployment spell, it might prefer to stick to its original contract, as any future mortgage will be underwritten at higher interest rates.

**Mortgage Refinancing in Steady State**

Figure 7 illustrates the percentage of homeowners that decide to refinance their mortgage on the age-equity state space. In contrast to default, mortgage refinancing occurs at a later stage in the life cycle and is made by households that have positive home equity. For these households with positive equity, it is never optimal to default, as they can always sell their house, repay their total outstanding debt and keep the profits. However, they might prefer to refinance to get a mortgage with better terms. Mortgages are refinanced for two reasons. They can be refinanced to decrease the size of the mortgage, such that future mortgage payments are lower, or can be refinanced to borrow funds, which increases the size of the mortgage and the future mortgage payments. Figure 8 illustrates the average mortgage refinancing rate for households that choose to refinance upwards, or increase their debt, and for those that choose to refinance downwards, or decrease their debt.

Refinancing a mortgage upwards implies increasing present consumption at the expense of increasing mortgage payments and, thus, lower future consumption. In contrast, refinancing a mortgage downwards implies increasing future consumption at the expense of present consumption. Given that productivity follows a hump-shaped path over the life
cycle and households smooth consumption, the mortgage refinancing rate to increase debt is larger early in the life cycle, whereas the refinancing rate to decrease debt is larger later in the cycle.
In particular, households that have the highest income at the beginning of their life cycles are the ones that choose to refinance their mortgages upwards. Given that the idiosyncratic component of productivity is persistent and the age component is hump-shaped, these households expect a high income path in the future, so choose to refinance upwards early in their lives to smooth consumption. Similarly, households that have high productivity shocks in the peak of the productivity cycle, between ages 45-55, choose to refinance downwards, to substitute present consumption with future consumption.

1.7 The Great Recession

In order to generate a shock that induces a large drop in house prices and an increase in the foreclosure rate, I assume that while the economy is in steady state agents face two completely unexpected shocks that last for 3 periods: 1) a drop in the age component of productivity, so the drop in earnings by age group is close to the findings of Glover et al. (2014) during the Great Recession, summarized in Table 4, and 2) an increase in the minimum downpayment requirement to 25% of the house price, such that the maximum Loan-To-Value at origination falls from its initial steady state value $LTV = 1.25$ to 0.75. The second shock is justified by the Federal Reserve’s Willingness to Lend Survey, according to which the number of banks that declared tightening of credit standards rose from almost zero to above 50% in 2007, and the median downpayment requirement more than doubled, from a pre-crisis level of 5% to 13% (Boz and Mendoza, 2014).

For the baseline calibration, I assume that the government implements a subsidy-bailout policy analogous to TARP and HAMP, described below.
<table>
<thead>
<tr>
<th>Age group</th>
<th>Per capita earnings</th>
</tr>
</thead>
<tbody>
<tr>
<td>20−29</td>
<td>−12.8%</td>
</tr>
<tr>
<td>30−39</td>
<td>−11.1%</td>
</tr>
<tr>
<td>40−49</td>
<td>−8.8%</td>
</tr>
<tr>
<td>50−59</td>
<td>−9.6%</td>
</tr>
<tr>
<td>60−69</td>
<td>−4.4%</td>
</tr>
<tr>
<td>70+</td>
<td>+0.3%</td>
</tr>
</tbody>
</table>

Table 4: Productivity Shock

1.7.1 HAMP

Through the Home Affordable Modification Program (HAMP), the government offered subsidies to mortgage refinancing for households in financial distress. Households with mortgages that satisfied the following conditions were eligible: 1. payments-to-income had to be greater than or equal to 31% of monthly gross income, 2. the household must have been delinquent or in danger of falling behind on mortgage payments, 3. the property pledged as collateral had to be owner-occupied and the primary residence of the household, 4. the property had to be a single-family property with 1-4 units and an unmodified first-lien mortgage balance of up to $729,750, 5. the mortgage had to be originated before January 1st, 2009, and 6. the mortgage modification had to pass a net present value test that would make the mortgage holder better off after the modification (Agarwal et al., 2017). The program aimed at reducing mortgage payments to exactly 31% of monthly income. When designed, the government allocated $75 billion to HAMP.

In the baseline calibration of the model, households that satisfy conditions analogous to 1. and 2. are eligible for a mortgage refinancing subsidy. That is, access to HAMP is granted to households that satisfy two conditions. First, they must have payments-to-income above some threshold described below. Second, given that I don’t model strategic complementarities that lead some households to “strategically default”, while others don’t,
I assume that once the subsidy program is announced, all households claim to be “in
danger of falling behind on mortgage payments”. The government then observes the
households’ state variables $\tilde{s}$ and grants subsidies to those for whom default is optimal
according to $\tilde{s}$. Given that the government does not perfectly observe the individual
state variables, it will subsidize households for which default is optimal according to the
observed $\tilde{s}$, but is not according to the realized $s$, which are the strategic defaulters in my
model. In equilibrium, the probability of observing an incorrect state, $p$, will be calibrated
such that strategic default accounts for 10% of total default, as described in Section 1.7.2.

### 1.7.2 The Information Parameter $p$

Mayer et al. (2014) estimate the magnitude of strategic default when mortgage refinancing
programs are implemented. The authors use an exogenous settlement between the U.S.
Federal Government and Countrywide Financial Corporation in 2008 - before HAMP -, in which Countrywide agreed to “offer modifications to seriously delinquent borrowers” with
subprime, first-lien mortgages. Immediately after the announcement, delinquency rates
of eligible households increased 10% yearly, compared to non-eligible mortgages and
mortgages of the same class from other financial institutions. Furthermore, the increase
was larger among households with large available credit on credit cards and lower current
LTV, who were least likely to default otherwise.

Even though 10% is a point estimate of strategic default for the specific case of the
settlement between Countrywide and the Federal Government, given the lack of other
estimates for strategic default during the Great Recession, I assume that in the case of
HAMP strategic default accounted for exactly 10% of total default. This assumption might
overstate total strategic default, as HAMP only allowed refinancing that reduced payments-
to-income exactly to 31%, while the Countrywide settlement allowed for unconstrained
refinancing. In this sense, households might have had more incentives to strategically
default during the Countrywide settlement.
To obtain a 10% strategic default, I set the parameter $p = 0.016$ such that, in equilibrium, in 1.6% of the cases the government observes states $\tilde{s}$ that are associated with default for households that would not choose to default according to their realized $s$. These households receive the benefits of HAMP, even though they would not have defaulted in the absence of the program.

### 1.7.3 TARP

The Troubled Assets Relief Program was implemented to preserve the solvency of systemically important financial institutions. Although, initially $700$ billion were allocated to the purchase of troubled assets, by late 2008 TARP had only invested $243$ billion through the Capital Purchase Program. Under Section 202 of the EESA, the Office of Management Budget (OMB) and the Congressional Budget Office (CBO) were required to perform a semiannual risk-adjusted net present value of TARP. The first of such reports, published by the CBO in January, 2009, estimated a risk-adjusted net present value of the Capital Purchase Program (CPP) equal to $-61$ billion (Calomiris and Khan, 2015). In the baseline calibration, I assume that TARP is a one-time transfer from the government to the mortgage investors to cover for the losses after the aggregate shock.

### 1.7.4 Results and Counter-factual Experiments

I model HAMP and TARP as one-time programs that are implemented in the same period of the aggregate shock and last only for one period, which corresponds to two years. The reason for a one-period duration is that, once the aggregate shock takes place, there is no uncertainty in the economy, as agents perfectly foresee the transition path of the economy towards the steady state. Therefore, any policy to preserve the solvency of mortgage originators can be implemented in the first period of the shock. The duration I am assuming is not far from what happened with TARP, where most of the investments
undertaken by late 2008 were reversed in less than a year, as the financial institutions recovered. In contrast, HAMP lasted through December 2016, and was modified over the years.

In the baseline calibration, the government implements TARP and HAMP so as to preserve ex-post solvency of mortgage originators, where 45% of the resources correspond to direct transfers to the mortgage investors (TARP), and 55% as subsidies to mortgage refinancing (HAMP). These numbers match the proportion of expenditures during the Great Recession, where the government expected to spend $60 billion on implicit bailouts through TARP, and allocated $75 billion to HAMP. For these proportions to hold in the model, the payments-to-income eligibility threshold for HAMP is set at 27.5%, which is close to the 31% threshold observed in the data. The subsidy is contingent on mortgage refinancing, which means that households cannot obtain the subsidy and default in the same period. The size of the subsidy is such that, in the period of the transfer, payments to income would be exactly 27.5% if the household continued under the same debt contract. In my model, however, households receive this transfer and have to refinance their mortgage without constraint.

The solid lines in Figure 9 illustrate the behavior of housing prices and foreclosures in the baseline calibration, when the government implements HAMP and TARP. The aggregate shock generates a decrease in house prices of 21%, while foreclosure rates in the model almost tripled from 2.4% to 6.8%. This behavior is close to what happened during the Great Recession, illustrated in Figure 1. As an external check, the model does well at replicating the percentage of households underwater during the recession, which is a non-targeted moment, equal to 16.2% in the model and 15% in the data (Melzer, 2017).

It is worth explaining the behavior of the foreclosure rate during the transition path. Initially, there is a sharp increase in the foreclosure rate, which rapidly reverts and falls below the pre-crisis level, finally returning back to the initial steady state after 20 years.
This happens because during the peak of the crisis a large fraction of households is left with negative home equity and lower incomes due to the productivity shock. Some of them choose to default and others choose to refinance. Those who receive HAMP benefits are able to modify their mortgages to better terms and move out of the lower part of the equity distribution, having a lower risk of default in the future. For this reason, default is below the pre-crisis level for some periods after the crisis. As the economy recovers and the distribution of households returns back to steady state, the default rate rises to the initial levels. Figure 10 illustrates the behavior of the equity distribution over the transition. In the first period of the shock, the distribution shifts to the left. Given that default and refinancing rates increase during the crisis, after some periods the equity distribution shifts to the right and slowly converges to the pre-crisis distribution.

Figure 9 illustrates two counter-factual experiments. In the first one, denoted “Subsidy-only Policy” in the figure, the government completely eliminates TARP and lowers the payments-to-income eligibility threshold of HAMP to 22.5%, so as to reduce foreclosures up to the level where mortgage originators have no losses. Note that, since the aggregate

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure9.png}
\caption{Transitional dynamics of (a) house prices and (b) foreclosure rate.}
\end{figure}
shock lasts for three periods and households default and refinance while the shock lasts, foreclosures are below the pre-crisis level for some years after the shock disappears. Given this, in order to preserve the solvency in the first period, default need not be reduced to the pre-crisis level. Using a utilitarian welfare function, welfare increases 0.2% in consumption equivalent terms, as compared to the baseline calibration.

There are two reasons for this. First, the welfare cost of the dead-weight loss associated with foreclosures is larger than the welfare cost caused by the distortion generated by the taxation levied to subsidize strategic defaulters. This happens because the Frisch elasticity of labor supply is not sufficiently large, so increasing taxes to subsidize an additional 10% of mortgage holders - strategic defaulters - does not generate a large welfare cost in the economy. Moreover, redistributing resources to strategic defaulters is not bad per se, so the welfare gain of strategic defaulters outweighs part of the distortion generated by taxation. In contrast, a 20% dead-weight loss of foreclosed houses generates unambiguously negative welfare consequences that outweigh the cost of strategic default. In this sense, it is welfare preferable to implement a subsidy policy that prevents foreclosures from happening, than to allow households to default and offer bailouts to mortgage holders to cover the ex-post losses.
The second reason is more subtle. As discussed at the end of Section 2.5.4, being underwater is a necessary, but not a sufficient condition for mortgage default, and only households that are sufficiently underwater choose the default. For this reason, after a negative house price shock, the government has to spend fewer resources to prevent foreclosures through mortgage refinancing subsidies, than to cover for the losses of that default, given that the default could be prevented with a transfer that is smaller than the negative balance. For this reason, the government can preserve the solvency of the banks by spending fewer resources on mortgage refinancing subsidies, than by covering its ex-post losses.

In the second experiment, denoted “Bailout-only Policy” on the figure, the government completely eliminates HAMP and only makes a bailout to mortgage originators to preserve ex-post solvency. In this case, default increases, as illustrated on the right panel of Figure 9, and welfare decreases 0.8% in consumption equivalent terms.

Given that subsidies redistribute resources within the economy, there are some groups that benefit more from a subsidy-only policy. Figure 11 illustrates the average welfare gains of implementing the subsidy-only policy instead of the baseline HAMP-TARP, according
to home equity. Home-owners with $-40\%$ to $-20\%$ of equity have the largest gains, as they are the most indebted ones and receive the largest subsidies. This group would be willing to reduce its lifetime consumption up to $2.5\%$ in the HAMP-TARP baseline, to be in the subsidy-only scenario. Households with equity between $-20\%$ and $0\%$ also receive a large portion of the subsidies and would be willing to reduce their consumption up to $0.9\%$ in order to be in the subsidy-only policy. The welfare gains of households with equity between $0\%$ and $60\%$ are small, as only a small fraction of them receive subsidies. Households that have equity above $60\%$ on their home experience negative welfare gains, as under the subsidy-only case house prices do not fall as much, when compared to the baseline scenario (see Figure 9). Given that these households are the largest house investors, higher prices during the crisis reduce the future returns on their investments. Moreover, they do not receive subsidies so, on average, the subsidy-only program reduces their welfare. Finally, the group of households that do not own a house experience a welfare gain of $0.11\%$ in consumption terms, because on the subsidy-only scenario the government levies fewer taxes.

Figure 12: Default rate by age, before and after the Great Recession.

Figure 12 illustrates default by age, before and during the Great Recession. Given that young households suffered the largest drop in earnings and were the most indebted
ones, the default rate increased more among them, when compared to the pre-crisis level. Although the HAMP policy targeted households with high Payments-to-Income, which is a variable that is correlated with age, a better eligibility rule should have an age component that targets the youngest households. In that way, the government could achieve a larger reduction in foreclosures, at a lower cost.

A final counter-factual experiment does precisely that, by eliminating TARP and implementing a subsidy-only policy, where the eligibility rule is modified. Under the new policy, only households that were not defaulting in the initial steady state and choose to default after the shock are eligible. That is, the subsidies are targeted mostly at the youngest homeowners. Moreover, the size of the subsidy is the same as is HAMP but has an additional component which decreases with age. Under this scenario, using a utilitarian welfare function yields welfare gains of 0.4% in consumption equivalent terms. This means that a better-designed subsidy policy based on observable characteristics can yield welfare improvements during mortgage crises.

1.8 Concluding Remarks

Housing crises are events in which a decrease in aggregate house prices leads to higher-than-expected foreclosure rates, generating potentially large losses to mortgage investors. To preserve the solvency of financial institutions, which are the largest investors in the mortgage market, governments have historically implemented two policies: a) offer bailouts to institutions to cover for their losses, and b) subsidize the mortgage refinancing of households to prevent additional foreclosures. During the Great Recession, foreclosures tripled after a house price drop of over 20%. Through the Emergency Economic Stabilization Act of 2008, the government implemented the Troubled Assets Relief Program (TARP), through which it implicitly bailed out troubled financial institutions through asset purchases, and
the Home Affordable Modification Program (HAMP), which offered subsidies for the mortgage refinancing of households in risk of default.

This paper studies the welfare-maximizing policy to preserve the solvency of financial institutions, assuming that the planner can offer bailouts and mortgage refinancing subsidies. The use of these instruments involves a trade-off, determined by two frictions. On one hand, foreclosed houses lose 20% of their value during the foreclosure process, as houses get damaged during the process and depreciation is larger because of vacancy. Therefore, if the government offers bailouts to investors rather than subsidies to households to prevent foreclosures, the economy has to bear the welfare cost of the dead-weight loss.

On the other hand, the literature has identified an idiosyncratic component in house prices, which is not perfectly observable by the government. Given that mortgage default depends on house prices, the government does not have perfect information on individual households’ decision to default. When offered subsidies, households have incentives to engage in strategic default to qualify for the benefits of the subsidy program. A subsidy policy avoids the dead-weight loss of foreclosed houses, since it prevents mortgage default, but subsidizes households that would not default in the absence of the program. If taxation is distortionary, the taxes levied to subsidize strategic defaulters generate a welfare loss in the economy, through a distortion in the labor decision of households.

This paper quantitatively assesses the welfare maximizing-policy during mortgage crises. Specifically, a heterogeneous agents’ model is calibrated to the pre-crisis U.S. economy, where the trade-off between bailouts and subsidies is disciplined using empirical micro estimates on the dead-weight loss of foreclosed houses and the size of strategic default. In equilibrium, the welfare cost generated by the dead-weight loss is larger than the distortion generated by the taxation levied to subsidize strategic defaulters, given the Frisch elasticity of labor estimated in the macro literature. For this reason, a subsidy-only policy would have generated welfare gains of up to 0.4%, measured as the consumption
equivalent variation, when compared to the baseline calibration that matches the TARP and HAMP programs. Given that a subsidy policy implies a redistribution of resources within the economy, the welfare gains are heterogeneous. Households on the left tail of the equity distribution, which benefit the most from a subsidy program, obtain the largest welfare gains. In contrast, a bailout-only policy would have generated a welfare loss of 0.8%.

The set-up I use has three main shortcomings. First, I assume that the Great Recession was generated by exogenous shocks to fundamentals. In my model, any bailout offered to financial institutions has the sole effect of transferring funds to cover the losses, without any effect on house prices. Different results would be obtained under the assumption that the Great Recession was generated by changes in expectations. In that alternative setting, a bailout policy could be a mechanism to align expectations toward an equilibrium in which housing prices do not fall as much. This means that, to the extent that the Great Recession was expectations-driven, my paper understates the potential welfare gains of using bailouts over subsidies.

Second, I explicitly ignore the moral hazard concerns of bailing out banks or subsidizing households. Clearly, both policies can generate incentives for banks and households to engage in overly-risky behaviors, affecting the likelihood of future crises and, therefore, having welfare consequences. To avoid moral hazard issues, I assume that the crisis was a zero-probability event, so neither households nor banks had expectations about the policies the government would implement during a crisis. The moral hazard consequences of government policies during mortgage crises are a topic left for future research.

Finally, a recent strand of the literature evaluates the impact of government bailouts during crises, through the financial accelerator channel. Bailouts may reduce the risk of banks running out of equity, which reduces the financing costs of the economy and increases economic activity. For computational reasons, this paper does not model the amplification effects of government policies through their effect on the balance sheets of
banks. I abstract from this effect by assuming that aggregate shocks are zero-probability events in the economy, so in the steady state banks, finance their activities at the risk-free rate. In this sense, my paper under-estimates the benefits of a bailout policy.
Chapter 2

General Equilibrium Effects of Student Loans on the Provision and Demand for Higher Education

We characterize the outcomes of the tertiary education market in a context where borrowing constraints bind, there is a two-tier college system operating under monopolistic competition in which colleges differ by the quality offered, and returns to education depend on the quality of the school. Our main finding shows that subsidized student loans can lead to a widening gap in the quality of services provided by higher education institutions. This happens because the demand for elite institutions unambiguously increases when individuals can borrow. This does not happen in non-elite institutions, since relaxing borrowing constraints makes some individuals move from non-elite to elite institutions. The higher increase in demand for elite institutions allows them to increase prices and investment per student. The motivation of this paper is the case of Colombia, which implemented massive student loan policies during the last decade and experienced a widening in the gap of quality supplied by elite and non-elite universities.
Such results show that, when analyzed in general equilibrium, subsidized loan policies can have regressive effects on the income distribution.

### 2.1 Introduction

The market for higher education has received significant attention in the economics literature. In particular, the effects that subsidized loan policies have on the demand side of the market have been widely studied, given the dramatic increase in student debt during the last two decades in the U.S. Overall, there is a consensus in the literature on the fact that credit constraints explain only a small fraction of enrollment decisions in higher education in the U.S. However, this is not the case in developing countries, where student financial aid systems are weak and evidence suggests that college enrollment is highly determined by family wealth (Bank, 2003, 2012). Within this context, the implementation of subsidized student loan policies increases the demand for education, which may have additional equilibrium effects, such as increases in tuition prices and changes in the quality of services offered by colleges.

Understanding the effects of subsidized student loan policies is of central importance, given the massive investments that have been made in student credit programs during the last two decades in the developing world, Latin America and some African countries. The demand side effects of these policies in a context where borrowing constraints determine enrollment decisions have been studied by the literature and the conclusions are certainly appealing: an expansion in student loans increases the demand for higher education among the most able students, which reduces the inefficiency that exists when very high-ability individuals with low initial wealth cannot access tertiary education (Canton and Blom, 2004). This partial equilibrium analysis unambiguously suggests that such policies

---

12. In this paper we use the terms “college” and “universities” indistinctly
have welfare improving effects on its beneficiaries. As a consequence, these programs have often received the support of international organizations, such as the World Bank and the Inter-American Development Bank\textsuperscript{13}.

However, the implementation of subsidized student loan programs has general equilibrium effects that have not been studied by the literature. The increase in demand for education generated by student loan policies can potentially affect tuition prices and quality offered by certain colleges. This general equilibrium effects can affect the welfare of individuals that do not have access to the loan programs, which might offset the overall benefits of the policies.\textsuperscript{14}

This paper contributes to the literature by analyzing the general equilibrium effects that subsidized student loan policies have on the quality of education provided by different tiers of colleges. We assume there are two tiers of colleges, which we denote low and high-quality, or elite and non-elite colleges, that operate under monopolistic competition (Dennis Epple and Sieg, 2006). Colleges choose the skills thresholds for admission, tuition rates, and investments per student to maximize the quality of the education offered, which is a function of the skills of the student body and on total investments per student. In equilibrium, students who attend elite colleges have higher expected returns in the labor market, when compared to students who attend to the non-elite system. Individuals choose whether and which college to attend, given the expected returns of each option. We characterize the demand for higher education, the incentives of each tier to invest and admit students, and the equilibrium consequences of subsidized-loan policies.

\textsuperscript{13} These institutions have contributed to different student loan projects in the developing world. For example, the World Bank has been financing the Colombian ACCES program since 2002 and committed in 2014 to lend $200 million during the period 2014-2019. Recently, the IDB provided a $10 million dollar loan to the Higher Education Finance Fund in 2012, to finance student loan programs in 4 Latin American countries.

\textsuperscript{14} ? argues that in an incomplete markets setting, although increasing borrowing limits increases the welfare of borrowing constrained individuals, in equilibrium this also leads to an increase in the interest rate paid by the borrowers. The two effects oppose each other, so the effect of loosening borrowing limits on welfare is ambiguous and follows a U-shape. Although we do not take into account the effect of borrowing constraints on the interest rate and assume government student loans are subject to an exogenous interest rate, his findings strengthen our theory that student loan policies might have negative effects on welfare, in equilibrium.
We find a set of equilibria in which a subsidized student loan policy widens the gap of the quality supplied by elite and non-elite institutions. When the loan program is implemented, the demand for elite colleges unambiguously increases, as the loans loosen the borrowing constraints of high-skilled individuals with low wealth. Higher demand allows elite universities to increase the skill acceptance threshold and, as a consequence, the average skills of the student body, while maintaining budget balancedness. As long as the skills of the student body and investments per student are complements in the production function for education quality, colleges have incentives to increase tuition and investments per student, generating a further increase in quality. In contrast, the demand for non-elite colleges does not necessarily increase with the student loan policy. Even though some individuals that would not study in the absence of the program will enroll in non-elite schools, some others that would attend non-elite institutions in the absence of the program will decide to attend elite schools when they have access to loans. The effect of the subsidy program on the average skills of the student body of non-elite colleges does not necessarily increase, nor do the investments per student. This generates a widening of the gap of the education quality provided.

In addition to studying the response of education quality to subsidized student loan policies, our analysis is novel given that we focus on developing countries, as opposed to the structural literature that has only explored the U.S. context, as far as our knowledge goes. The educational sector in developing economies is particularly different from that of developed economies, for three main reasons. First, there is evidence that credit constraints play a role in determining college enrollment decisions among households of developing countries (Melguizo et al., 2015), as opposed to the case of developed countries. Second, in many developing countries, private institutions own a larger share of the market for higher education, as compared to European countries or even the U.S. (see Figure 1). This is important because public institutions may not be as responsive to market incentives, but rather follow the social planner’s objectives. In contrast, private
institutions are potentially more responsive to market signals, so any change in demand will generate stronger equilibrium effects in developing economies. Third, enrollment rates in developing countries are very low, when compared to enrollment in developed countries. As documented by Mestieri (2016), there is an existing positive correlation between enrollment rates and income per capita at a cross-country level.

Figure 1: % of enrollment in private institutions by country.

The main purpose of this paper is to rationalize how a widening of the education quality gap can arise as a consequence of a subsidized student loan policy. This would imply that student loan programs also have downsides when studied in general equilibrium, and all of this should be taken into account for future policy design. We focus in the case of Colombia, a developing country that undertook a massive expansion of publicly supplied student loan availability during the last decade. We use a novel dataset that allows us to analyze the evolution of various measures of quality of education before and after the policy was implemented. After the introduction of the policy, the number of students enrolled increased. However, there is evidence consistent with a widening gap in the quality of elite and non-elite universities, measured as average test scores in entry and
exit examination tests, the number of professors per students and various measures of academic production such as articles published per faculty.

Finally, our analysis gives us tools to discuss the design of the optimal student loan policy in a context where the government has outside funds that has to allocate within the existing population. From a partial equilibrium perspective, we find that the student loan policy that maximizes utilitarian welfare and enrollment is one that gives priority to the lowest-ability individuals that are borrowing constrained and would like to study. The reasoning behind this is the following: high-ability individuals will receive higher incomes over their lifetime, regardless of their education level. Since marginal utility is decreasing, the benefits of studying to have additional income are relatively small. In addition, those that are borrowing constrained and study will not be able to smooth consumption. In this sense, they face a higher opportunity cost of education. Therefore, relaxing the constraint for high ability individuals will change the study decision of fewer individuals, than if the constraint were relaxed among lower ability individuals.

However, from a general equilibrium perspective, there is an opposing force in action, which suggests that the optimal policy should offer subsidized loans to the most able. Individuals choose whether and where to study according to the quality offered by each college, since their future earnings depend on it. The quality offered by colleges, in turn, is a composite of the average ability of the student body and investments per student made by the college. As assumed by the literature, these two inputs are complements in the quality production function. In this regard, the best quality-enhancing student loan policy would maximize the average student ability, which would increase returns to investments per student, and would lead to higher quality offered.

In this regard, there are two opposing forces shaping the optimal student loan policy. Relaxing the borrowing constraints of lower ability households will have the highest impact on school enrollment, but will reduce the returns to investments per student done by schools and the quality of education. In contrast, relaxing the borrowing limits of high
ability households maximizes the education quality offered, but enrollment is not as large. The issue becomes even more complex once we incorporate a two-tier education system, in which colleges might respond differently in their pricing, admissions and investment policies when faced to a demand shock of this nature.

The rest of the paper is organized as follows. In Section 2.2 we describe the relevant literature. In Section 2.3 we describe a model of the market for higher education, characterize the demand for a two-tiered education system and explains the mechanism through which borrowing constraints affect equilibrium quality supplied. In this section, we illustrate the main theoretical results of the paper. That is, a policy leading to subsidized loans can increase the gap of quality of education. We describe the case and Colombia and illustrate how this case is consistent with what we predict in the theoretical model in Section 2.4. In Section 2.5 we describe the numerical analysis and calibration exercise. We conclude in Section 2.6 concludes.

### 2.2 Related Literature

This paper is related to various lines of the literature on the economics of education. First, our paper is related to the literature studying the relevance of borrowing constraints in the access to higher education. Given that we are studying the welfare effects of government loan policies in developing countries, knowing whether borrowing constraints matter is of central importance. Although there is evidence suggesting that borrowing constraints do not determine school attendance of students in advanced economies, the opposite is the case for developing economies. Second, our paper is related to the literature that studies general equilibria in the market for education. This literature has mostly studied what is known as the “Bennett Hypothesis”\(^{15}\). Our paper adds to this literature in two dimensions. First, we study equilibrium effects that go beyond prices, as we analyze the effects on the

---

\(^{15}\) The “Bennett Hypothesis” states that an expansion in the number of grants provided to students are almost totally appropriated by colleges through increases in tuition prices.
quality offered by colleges. Second, this literature has been focused extensively in the United States. In this paper, we provide a new context of analysis for general equilibrium effects in the market for higher education: a developing economy where credit constraints bind. Finally, our paper is related to the industrial organization literature analyzing the behavior of colleges in non-competitive markets. We implement the framework proposed by Dennis Epple and Sieg (2006) and extend it to assess the general equilibrium effects of tuition prices, college quality, and admittance rules, in the context of a developing economy.

2.2.1 The Role of Borrowing Constraints

There seems to be a consensus in the literature suggesting that the effects of borrowing constraints on the post-secondary decisions of youngsters are negligible. Using the 1979 of the National Longitudinal Survey of Youth (NLSY79), Carneiro and Heckman (2002) find evidence that borrowing constraints and family income account for a very small fraction of post-secondary school attendance decisions, while early childhood differences are determinant. According to their estimates, only between 0% and 8% of high school graduates are actually borrowing constrained. Similarly, Keane and Wolpin (2001) find that, although borrowing constraints are tight in the U.S. and individuals cannot even borrow the amount to cover one year of schooling, their existence does not determine the decision to study. In counterfactual experiments, when the authors remove the borrowing constraints, the educational attainment does not change significantly. Borrowing constraints only affect labor supply and savings decisions of students.

Dinarsky (2003) measures the impact of the exogenous elimination of the Social Security Student Benefit Program in 1982 on school attendance in the U.S. This program provided monthly payments to college students who had a family member who was deceased, disabled, or a retired social security beneficiary. The paper finds that the exogenous reduction in aid led to a decrease in the probability of being enrolled by students at
the margin. However, the author argues that this cannot be interpreted as existence of borrowing constraints, since grants do not only relax the borrowing constraints of households, but change also the relative price of education.

More recent studies argue that, although credit constraints did not seem to affect the schooling decision some decades ago, during the last two decades they might be playing an important role in post-secondary schooling in the United States. Using data from the 1979 and 1997 National Longitudinal Survey of Youth (NLSY79 and NLSY97), Belley and Lochner (2007) find a dramatic increase in the importance of family income on school attainment, after controlling for family background and ability as in the previous studies. Similarly, Lochner and Monge-Naranjo (2011) estimate a structural model that suggests that, although American households were not borrowing constrained during the 1980s, during the last decade family income has been determinant in schooling decisions. They argue that in the last two decades there have been rising costs and returns to education, while government student loan programs have not grown at the same pace, so people have become borrowing constrained.

Although there is not much research that studies the role of credit constraints in educational choices in developing countries, the existing evidence seems to unambiguously point towards the importance of borrowing constraints in the educational decisions. As Attanasio and Kaufmann (2009) state, “one important difference between Mexico and the U.S., for instance, might be the wider availability of scholarships and student loans in the U.S., cannot be found in Mexico for higher education.” Attanasio and Kaufmann (2009) and Kaufmann (2014) provide evidence suggesting that liquidity constraints do determine the post-secondary schooling decision in Mexico. They use data that characterizes the expected returns of education for every household in their sample. If credit constraints were not binding, there should exist a positive gradient between subjective expected returns from schooling and school attendance. Their results show that this gradient breaks for the lowest income households in their sample. Under their interpretation, this is evidence of
existing borrowing constraints. Solis (2013) studies the existence of borrowing constraints in Chile. Using administrative data on the entire sample of individuals that participate in the college admissions’ process, he uses a regression discontinuity approach to study the impact of providing educational loans. After controlling for socio-economic covariates, individuals right above the eligibility threshold for receiving educational loans have a significantly higher probability of enrolling in college than those right below the threshold. The author finds evidence suggesting a positive gradient between income and enrolment among those households that have no access to the government loans. This gradient disappears for individuals that access the program. Also using a regression discontinuity approach, Marc Gurgand and Melonio (2011) find evidence that the enrolment to college of households without access to student loans is 20 percentage points lower in South Africa. Regarding the Colombian case, Melguizo et al. (2015) find evidence that the implementation of a massive government loan program in the past decade, which is the topic of this paper, did increase student enrolment.

2.2.2 General Equilibrium Effects and the Bennett Hypothesis

During the last decade, the literature that has tried to explain what has become to be known as the Bennett Hypothesis: expansions of government-supplied student aid for education have been almost entirely appropriated by colleges through an increase in tuition prices. As the former U.S. Secretary of Education stated in 1987, "If anything, increases in financial aid in recent years have enabled colleges and universities blithely to raise their tuitions, confident that Federal loan subsidies would help cushion the increase.”16 Singell and Stone (2007) study the effect that Pell Grants have had on tuition prices of public and private schools. They study the Pell Program, which has been the biggest post-secondary educational loan program in the United States. In 1999, the Pell Grants were awarded to 3 million students across more than 6000 colleges, out of a total of 9 million students. The

authors estimate the impact of Pell Grants per student on tuition charged by universities, using a panel of 1554 colleges from 1989 to 1996. They find that the increase in Pell Grants caused an almost one-to-one increase in the price of tuition charged by private and public out-of-state colleges. However, they find no such a causality on the in-state tuition charged by public schools. In contrast, Rizzo and Ehrenberg (2002) find evidence that private and public out-of-state tuition prices were not affected by government loans, while in-state tuition by public colleges were. Finally, David O. Lucca and Shen (2016) use exogenous variation in the legislation that rules Pell Grants, to study the relationship between student aid and tuition.

Gordon and Hedlund (2015) study the increase in tuition prices by estimating a structural model in which universities provide human capital and households decide their investments in education. They study the rise in college tuition over the last decades, as a reaction to cuts in state appropriations, an increase in the costs of skilled labor in other industries, and an increase in government supplied loans. The authors find out that the increase in government loans explains 102% of the tuition increase, as opposed to only 16% of the other two hypotheses. This result provides evidence in favor of the Bennett Hypothesis. Our paper differs from theirs in the sense that we want to study the equilibrium effects on quality provided and welfare effects of relaxing borrowing constraints in a context in which they matter. The authors study increases in the borrowing limits in the context of the U.S. As has been already argued, there is evidence that these constraints are of secondary importance on the decision to attend school. Therefore, relaxing these limits does not improve efficiency. In contrast, in countries in which the borrowing constraints are binding, relaxing them does generate efficiency improvements.

2.2.3 The Education firms

Our paper makes part of the literature that models universities as firms in the educational sector. Universities produce human capital and use households both as inputs and
costumers. This approach has been used to study different questions regarding post-secondary education. For instance, Hector Chade and Smith (2014) model the universities as an oligopoly with a fixed number of universities (firms), in which the goods produced by universities (education) are ranked exogenously by all households in the same way. Universities only choose admission standards, so as to fill a fixed capacity of students and maximize the ability of the student body. The paper studies the role of frictions in the application process on the student sorting between universities. Namely, the model has information frictions and fixed costs of application. The authors, as Caucutt (2001), treat the utility that households receive of attending each of the universities as exogenous and independent from the product offered by each university. We endogenize the valuation of households as a function of the equilibrium quality offered. The authors do not include tuition prices as a policy of universities, assume an exogenous valuation for the universities and take the size of universities as fixed. We depart from all of these assumptions, but assume there are no frictions in the application process. The reason is that our purpose is not to study the outcome of the application process but, instead, model the strategic interactions in the post-secondary education sector between universities and households.

The educational sector in our model closely mimics Dennis Epple and Sieg (2006). In their paper, the authors model the supply side of the educational sector as an oligopoly sector in which a fixed amount of colleges interact to attract students and maximize the quality of the education they offer, subject to a balanced budget constraint. Quality by universities is a composite of average student ability, to resemble peer effects in schooling, and the average investments per student. This treatment of quality has been standard in the literature that models schools (Caucutt, 2001). Households value quality as an input on their utility function. In their model, households play a passive role in their model, since their purpose is not to estimate equilibrium interactions between households and firms. Rather, they concentrate in studying thoroughly the supply side. Furthermore, they estimate their model by using a “club goods” approach, instead of explicitly solving
the Nash equilibrium of the monopolistically competitive market. We depart from this approach, since we consider that the strategic interactions between colleges are of first order importance to explain the different reaction of elite and non-elite institutions to subsidized student loan policies.

Finally, we treat wages of college graduates as a function of the quality supplied by the school attended. To the best of our knowledge, this approach has not been used in structural estimations in the past, but there is empirical evidence that relates future wages to the quality of the education. Black and Smith (2006) estimate a latent model in which quality is a latent variable, and there are “signals” of quality. They find out that SAT scores, faculty-student ratio, rejection rate, freshmen retention rate, and faculty salaries are significant signals of quality. Furthermore, the latent variable of quality significantly affects post-college wages of individuals. Similarly, Dan Black and Daniel (2005) find evidence that quality increases post college earnings, driven by higher wages. Leaving quality aside, there is extensive evidence that estimates positive returns to college attendance in terms of higher future wages (Zimmerman, 2014; Harry Anthony Patrinos and Sakellariou, 2006). OECD and Bank (2012) estimates that average starting earnings for individuals with a bachelor’s degree were 4 times higher than those of individuals with high-school degree. Although these estimates do not control for unobservable household characteristics, other estimates find that people with post-secondary degrees earn significantly higher wages in Latin America (L. Gasparini and Acosta, 2011).

2.3 A Model of the Market for Higher Education with Credit Constraints

There are two types of agents in the economy: households and universities. There is a government that offers educational credits to high-ability individuals that decide to attend college, at an exogenous interest rate \( R \geq r \), where \( r \) is the risk free interest rate.
In addition, the government subsidizes the interest paid by the poorest households that access the credit, at a subsidy rate $s$. In order to finance these subsidies, the government levies a marginal tax, $\tau$, to every household in the economy. The government policies are exogenous, fixed before the economy starts and satisfy budget balance. Given these policies, the market of higher education operates under monopolistic competition. Universities supply human capital in the market for education, by choosing a tuition price, a minimum ability level for admission and a level of investment per student. Given government and university policies, the households decide if they want to study in any university at the prevailing market prices.

2.3.1 Households

Households are born with innate ability and wealth $(\theta, b)$, according to a bivariate distribution $F(\theta, b)$ over the space $[0, 1] \times [\underline{b}, \bar{b}]$. Individuals live for two periods, after which they die with probability equal to one. In period 1, individuals choose either to study at the university or work in the non-skilled labor market at a wage $w$ per efficiency unit of labor. Individuals that do not study receive a wage $\theta w$, do not have access to credit markets and can save at the risk-free rate $r$. There are two universities in the economy denoted by $h$ and $l$. Each university sets a threshold $\theta^j$ for $j = h, l$ such that only students that have ability $\theta \geq \theta^j$ are admitted to university $j$, and we assume that this information is public. Therefore, individuals with $\theta < \min\{\theta^h, \theta^l\}$ cannot study and have to work. Individuals who decide to study at university $j$ cannot work, and have to pay a tuition, $P^j$, set by the university.

In order to finance education, the government offers student loans of up to the price of the tuition, $P^j$, at the interest rate $R$ to people that decide to study and have an ability level $\theta \geq \theta_{\min}$. In addition, students with low wealth, $b \leq b_{\max}$, that decide to study and

---

17. We assume that $\theta^j, j = h, l$ is a public threshold, since our purpose is not to study the frictions in the college application process, as opposed to some papers in the literature that model explicitly these information frictions (Hector Chade and Smith, 2014; Fu, 2014).
have access to the loan will receive a subsidy on the interest rate, $s$. Loans are given conditional on studying, and individuals that study and are eligible for the loan choose whether to borrow from the government or not. In order to finance these subsidies, the government levies a proportional tax, $\tau$, to every individual in the economy. Individuals for which $\theta < \theta_{\text{min}}$ are borrowing constrained and can only finance education with their initial wealth. Therefore, in the first period the household decides its level of consumption, $c$, whether to study or not in any university, $h, l$, and the level of savings, $a$, which can be potentially negative for households that study and satisfy the government conditions for the educational loans.

In the second period, the households are either non-, low- or high-skilled, depending on whether they decided to study in the first period and which college they attended. Those who decided to study in period $1$, will enter the $j$-skilled labor market in period $2$, and receive a wage equal to $w_\theta(1 + z^j)$, where $z^j$ is a skill premium that is university specific. This quality is an equilibrium object that depends on the quality of the student body and investments per student, and is fully characterized in the next section. We assume that individuals have perfect foresight of the value of $z^j$ for $j = h, l$ when they optimize. Individuals who do not study will become part of the non-skilled labor force at a wage $w_\theta$. We exclude the possibility of default in the model by assuming that repayment is fully enforced, so in the second period individuals that have government debt will repay their student loan. Given prices $R, r, w$, government policies $\tau, s$, university policies
\( \{ \theta^j, P^j \}_{j=h,l} \), and perfect foresight about education quality \( \{ z^j, z^h \} \), a household that is eligible for studying at the university \( j \), \( \theta \geq \theta^j \), and decides to study gets a utility equal to:

\[
V^j(\theta, b) = \max_{c,a} \quad u(c) + \beta u(c') \quad \text{s.t.} \\
\quad c + a + P^j = b \cdot (1 - \tau) \\
\quad c' = a(1 + r) \cdot 1_{\{a \geq 0\}} + a(1 + \bar{R}) \cdot 1_{\{a < 0\}} + w\theta(1 + z^j) \\
\bar{R} = \begin{cases} 
R(1 - s) & \text{if } b \leq b_{\text{max}} \\
R & \text{if } b > b_{\text{max}} 
\end{cases} \\
a \geq -1_{\{\theta \geq \theta_{\text{min}}\}} \cdot P^j, \quad c \geq 0, \quad c' \geq 0
\]

Individuals that decide not to study, get the following utility:

\[
V^N(\theta, b) = \max_{c,a} \quad u(c) + \beta u(c') \quad \text{s.t.} \\
\quad c + a = b \cdot (1 - \tau) + w\theta \\
\quad c' = a(1 + r) + \theta \\
\quad a \geq 0, \quad c \geq 0, \quad c' \geq 0
\]

The individual with ability and wealth \((\theta, b)\) decides to study at university \( j \) whenever \( \theta \geq \theta^j \) and \( V^j(\theta, b) \geq V^N(\theta, b) \), and \( V^j(\theta, b) \geq V^{-j}(\theta, b) \) if they can attend to the other university \(-j\), i.e. \( \theta \geq \theta^{-j} \). Otherwise, the individual decides not to study. Therefore, the household’s value function is given by:

\[
V(\theta, b) = \begin{cases} 
\max\{V^h(\theta, b), V^l(\theta, b), V^N(\theta, b)\} & \text{if } \theta \geq \max\{\theta^h, \theta^l\} \\
\max\{V^j(\theta, b), V^{-j}(\theta, b)\} & \text{if } \theta^{-j} > \theta \geq \theta^j \\
V^N(\theta, b) & \text{if } \theta < \min\{\theta^h, \theta^l\}
\end{cases}
\]
Figure 2: Representation of the education decisions on the state space.

The following section gives a detailed characterization of the demand for both tiers of schools in the state space. This characterization will allow us to give insights on the optimal student loan policy on a monopolistically competitive market.

Characterization of the Demand

For a given set of initial parameters, the shaded region in Figure 2 illustrates the individuals that choose to study in the state space when both universities set their acceptance threshold to 0 and there are no government-supplied student loans. The following sequence of theorems characterize the demand for college education on the state space, and its close relationship with borrowing constraints. This will let us derive some results about the socially optimal student loan policy. First, we describe the college decision for households that are unconstrained.
Theorem 1. Among the unconstrained households, the decision of whether and where to study is independent of initial wealth, $b$, and follows a cut-off rule on $\theta$. That is, there exist $\theta$ and $\bar{\theta}$ such that:

- If $\theta \leq \theta$, the individual will not study.
- If $\theta \leq \theta \leq \bar{\theta}$, the individual will attend the low-quality college.
- If $\bar{\theta} \leq \theta$, the individual will attend the high-quality college.

where:

$$\bar{\theta}_l = 1 + \frac{r}{w} \left( \frac{P_l}{z_l - (1 + r)} \right), \quad \bar{\theta}_h = 1 + \frac{r}{w} \left( \frac{P_h - P_l}{z_h - z_l} \right)$$

Proof. See Proof 2.1

Theorem 1 is a result of the fact that ability $\theta$, unskilled labor $w$ and quality of the school attended $z_j$ are complements. In particular, this complementarity implies: a) among the unconstrained individuals, those with higher ability face higher marginal returns of education, so will choose, ceteris paribus, a higher quality school for a given wealth, b) as the wages of unskilled labor $w$ increase, the marginal returns to education rise for every $\theta$, so marginal individuals will shift to higher levels of education, c) if college $j$, for $j \in \{l, h\}$, increases its price $P_j$ or reduces its quality $z_j$, marginal individuals will change their schooling decision in the expected direction. That is, if $P_j$ increases or $z_j$ decreases, marginal individuals will change their decision of attending school $j$. Finally, d) if the interest rate $r$ increases, present consumption becomes more valuable than future consumption, so marginal individuals will reduce their present expenditures in education. Theorem 2 characterizes the individuals that, given their decision to attend college $j$, are borrowing constrained.
**Theorem 2.** Given an ability $\theta$, there exist cut-offs, $\bar{b}^j_u(\theta)$, $j \in \{N, I, H\}$, on the initial wealth, such that individuals with $b \geq \bar{b}^j_u(\theta)$ that attend college $j$ will not be borrowing constrained. Individuals that attend college $j$ and have $b < \bar{b}^j(\theta)$ will be borrowing constrained and will not be able to smooth consumption over time. The cut-offs are linear, increasing in $\theta$ and take the form:

$$
\bar{b}^N_u(\theta) = -\bar{A}(1 + (\beta(1 + r))^{-1/\sigma}(1 + r)) - w\theta(1 - (\beta(1 + r))^{-1/\sigma})
$$

$$
\bar{b}^I_u(\theta) = p_l + (\beta(1 + r))^{-1/\sigma}w\theta(1 + z_l) - \bar{A}(1 + (\beta(1 + r))^{-1/\sigma}(1 + r))
$$

$$
\bar{b}^h_u(\theta) = p_h + (\beta(1 + r))^{-1/\sigma}w\theta(1 + z_h) - \bar{A}(1 + (\beta(1 + r))^{-1/\sigma}(1 + r))
$$

**Proof.** See Proof .2.1

Given a level of education and initial wealth, individuals with a higher $\theta$ have higher lifetime income and in an unconstrained world would consume more in every period of their lives. Given the existence of a borrowing limit $\bar{A}$, for a sufficiently high $\theta$ individuals will be borrowing constrained. As a consequence, the initial wealth that individuals must have not to be borrowing constrained is increasing in ability. Figure 2 illustrates the cut-off functions $\bar{b}^j_u(\theta)$ on the state space. As illustrated, individuals above the $\bar{b}^j_u(\theta)$ function, will decide to study in college $j$ whenever her $\theta$ falls inside the corresponding interval in the cut-offs defined in Theorem 1. Note also that individuals that are borrowing constrained when studying at college $I$ will also be borrowing constrained when studying in $H$, assuming a higher price of education in the high-quality college (which, of course, is an equilibrium object). Moreover, the functions $\bar{b}^j_u(\theta)$ are steeper when the quality $z_j$ increases, since quality of schooling and ability are complements. Finally, we do not consider the case in which individuals are borrowing constrained when they do not study.
Since in our context, individuals that do not study earn the same wage in every period, they will only be borrowing constrained when the interest rate $\beta(1 + r) << 1$. However, for a reasonable calibration, individuals will be able to smooth consumption. The following two theorems illustrate the study decision of individuals that are borrowing constrained.

**Theorem 3.** Given ability $\theta$, the decision to study in the low-quality college, $l$, or not study at all, follows a cut-off strategy on $b$, such that individuals with $b \geq \bar{b}_l(\theta)$ will attend college $l$, and those with $b < \bar{b}_l(\theta)$ will not study. The cut-off is characterized implicitly by equation (1) in the proof. Moreover, if the intertemporal elasticity of substitution is lower than 1 the cutoff is U-shaped and there exists a $\theta^*$ such that $\bar{b}_l(\theta)$ is:

- decreasing in $\theta$ for $\theta \leq \theta^*$
- increasing in $\theta$ for $\theta \geq \theta^*$

where $\theta^*$ solves:

$$
\left(\frac{1}{1 - \sigma}\right) (b(\theta^*)) - P_l + \bar{A})^{1-\sigma} + \left(\frac{\beta}{1 - \sigma}\right) (w\theta^*(1 + z_l) - (1 + r)\bar{A})^{1-\sigma} - \\
\Phi(w\theta^*(2 + r) + b(\theta^*)(1 + r))^{1-\sigma} = 0
$$

$$
b(\theta) = \theta \left[ \frac{wX(1+z_l) - w(2 + r)}{1 + r} \right] - X\bar{A}
$$

$$
X = \left[ \frac{\Phi(1 + r)(2 + r)}{\beta(1 + z_l)} \right]^{1/\sigma}
$$

*Proof.* See Proof 2.1. □

The cut-off $\bar{b}_l(\theta)$ is illustrated in Figure 2, where we assume that the utility function is CRRA with $\sigma = 2$, as is common in the literature, so the intertemporal elasticity is lower.
than 1. The individuals who are constrained (below \( \bar{b}_l(\theta) \)) will choose either to study at \( l \) or not, if their initial wealth exceeds \( \bar{b}_c(\theta) \). The cut-off is U-shaped because two effects are in action. First, the “complementarity” effect means that, given a \( b \), individuals with higher \( \theta \) will have higher marginal returns from studying, so are willing to study even though they will not be able to smooth consumption. Therefore, the cut-off is initially decreasing. However, the “constrainedness” effect dominates after some point: given an initial wealth \( b \), individuals with higher \( \theta \) will face a larger wedge in their Euler equation, meaning that they will be able to smooth consumption to a lower extent. When the wedge is large enough, individuals will prefer not to study and smooth consumption by deciding not to study. Of course, this results strongly depends on the value of \( \sigma \) chosen, and continues to hold for any \( \sigma > 1 \). For the sake of exposition, in Appendix 2.1 we characterize the demand for education with a linear utility function (that is, when \( \sigma = 0 \) and there is an infinite elasticity of substitution). Figure 23 illustrates the decision of individuals in the state space. As can be expected, in the linear case individuals derive no utility from consumption smoothing, so there does not exist such a “constrainedness” effect. In this case, the threshold is never increasing. The next theorem characterizes the cut-off for individuals that are constrained when studying at \( h \). The results are parallel to Theorem 3.

**Theorem 4.** Given ability \( \theta \), the decision to study in \( h \) or \( l \), follows a cut-off strategy on \( b \), such that individuals with \( b \geq \bar{b}_c(\theta) \) will attend college \( h \), and those with \( b < \bar{b}_c(\theta) \) will attend \( l \). The cut-off is characterized implicitly by equation (2) in the proof. Moreover, if the intertemporal elasticity of substitution is lower than 1 the cutoff is U-shaped and there exists a \( \theta^{**} \) such that \( \bar{b}_c(\theta) \) is:

- decreasing in \( \theta \) for \( \theta \leq \theta^{**} \)
- increasing in \( \theta \) for \( \theta \geq \theta^{**} \)
where \( \theta^{**} \) solves:

\[
\frac{1}{1 - \sigma} \left( b^*(\theta^{**}) - P_h + \bar{A} \right)^{(1 - \sigma)} + \frac{\beta}{1 - \sigma} \left( w\theta^{**}(1 + z^h) - (1 + r)\bar{A} \right) - \\
\Phi \times \left( w\theta^{**}(1 + z_l) + b(1 + r) - P_l(1 + r) \right) = 0
\]

\[
b^*(\theta) = \theta w \left( X^*(1 + z^h) - (1 + z_l) \right) - X^* + P_l
\]

\[
X^* = \left( \frac{\Phi \times (1 - \sigma)(1 + z_l)}{\beta(1 + z_h)} \right)^{1/\sigma}
\]

Proof. See Proof 2.1.

Having characterized the demand for education in the state space, we can say a couple of things about the relationship between borrowing constraints and the demand. The following result describes the differential effect of relaxing the borrowing limits to households, \( \bar{A} \).

**Theorem 5** (Borrowing constraints). If the intertemporal elasticity of substitution is lower than 1, for any given \( \theta \) the cut-offs \( \bar{b}_l^c(\theta) \) and \( \bar{b}_h^c(\theta) \) are decreasing on \( \bar{A} \). Moreover, the elasticities of \( \bar{b}_l^c(\theta) \) and \( \bar{b}_h^c(\theta) \) with respect to the borrowing limit \( \bar{A} \) are decreasing on \( \theta \), meaning that a relaxation of the borrowing constraint has a higher impact on enrollment among the marginal individuals that have lower \( \theta \).

Proof. See Proof 2.1.

Theorem 5 states that among the constrained individuals, those with lower \( \theta \) are more sensitive to relaxing the borrowing constraints. That is, if the borrowing constraints were relaxed by the same amount to all the individuals, more low-\( \theta \) individuals would
change their study decision. This result is a consequence of the decreasing marginal utility. Individuals with high $\theta$ and sufficiently low initial wealth have a trade-off between earning relatively high wages in every period and smoothing consumption if they do not study, or studying to earn large wages in the second period at the expense of very low consumption in the first period. However, because of decreasing marginal utility, the utility of a very large wage in the second period is not as large as for lower $\theta$ individuals, so individuals will optimally decide to study only when there is a large increase in the borrowing limits of the first period.

This result has very important implications on the design of an optimal student loan policy in a partial equilibrium setting. If the objective of the government is to maximize enrollment, the policy should target the lower ability individuals. As a matter of illustration of Theorem 5, Figure 3 illustrates the number of individuals of ability $\theta$ that change their study decision as the borrowing constraint is relaxed from $\bar{A} = 0$. As stated in Theorem 5, the individuals in the state space with low ability that would study in the unconstrained world (those with $\theta \in [\bar{\theta}_l, \bar{\theta}_h]$) are more sensitive to relaxing the borrowing constraints. Therefore, increasing the borrowing capacity increases enrollment more among the low ability individuals.

2.3.2 Universities

Universities act as firms that maximize an objective function. Given that university systems in most countries are non-profit firms, we follow the literature on education and industrial organization and assume that universities maximize a composite of the quality they offer to students, denoted by $z$, and the economic diversity of their student body, subject to a budget constraint. Quality offered by universities is an abstract concept. The literature has argued that the quality offered by a school is determined both by the quality of the student body and the investments per student done by the school. Dennis Epple and Sieg
Figure 3: Number of students that change their study decision when borrowing constraints change from $\bar{A} = 0$ to $\bar{A}$, by ability $\theta$.

(2006), for instance, model the objective function of the university as a composite of the average ability of the student body, the investment per student and the inverse of the mean income. They argue that there is empirical and anecdotal evidence that shows that colleges engage in policies to attract low income students. Universities take as given the values of $\tau, s, R, w$ and the distribution $F$. Additionally, we assume that universities set their policies simultaneously and so, the pricing and admission policies set in equilibrium should satisfy the no profitable one shot deviation principle.

University $j$ takes as given $(\tau, r, s, R, w, P^{-j}, \theta^{-j})$ and will set the pricing and admission policies $(P^j, \theta^j)$ in order to solve the following problem:

$$\max_{\{P^j, \theta^j, P^{-j}\}} \left( z^j \right)^{\alpha} \left( \sigma_i^j \right)^{1-\alpha}$$

(10)
subject to:

\[ z^j = \tilde{\theta}^{a_1} (I^j)^{a_2} \]  
(11)

\[ \tilde{\theta}^j = \int_{\Theta \times B} \theta \cdot e^j(\theta, b)dF(\theta, b) \]  
(12)

\[ I^j \cdot N^j + V^j(N^j) + C^j = P^j \cdot N^j + E^j \]  
(13)

\[ N^j = \int_{\Theta \times B} e^j(\theta, b)dF(\theta, b) \]  
(14)

where \( \tilde{\theta}^j \) is the average ability of the individuals that attend school \( j \). \( \sigma^j \) is the inverse of the average income of the student body and reflects the fact that universities care about the diversity in their student body. \( e^j(b, \theta) \) indicates with values zero or one if a student with ability \( \theta \) and wealth \( b \) decides to study or not. \( I^j \) is the monetary amount that the university invests per student, \( V^j \) is a convex cost function, \( N^j \) is the size of the student body, \( C^j \) is a fixed cost and \( E^j \) the university’s endowment. Note that the policy \( P^j \) does not depend on student’s characteristics such as wealth or skills. This is not only a simplifying assumption but also follows closely the case of Colombia where private universities do not price-discriminate students based on ability or wealth. As will be discussed in the relevant section, the extent of financial aid provided by such institutions is very limited in the period of analysis.

2.3.3 Discussion

Although in principle the solution to the problem of the university might seem simple given that there are only two variables of choice, there are several elements of the model that increase the complexity of such decision. First of all, both policies are interdependent. When a university changes one decision variable -either the price or the admission threshold- this will distort the incentives faced when setting the other policy. For instance, a change in tuition price will not only change the revenue of the university but will change
the demand in a way that we expect to see a change in the average ability of the student body. Such a change in the average ability of the student body will affect the marginal productivity of investments made by the university, which in turn will affect its pricing decisions.

Moreover, we need to deal with the fact that in equilibrium no university should have incentives to deviate. Given that both universities make the decision simultaneously and that there are no elements of incomplete information in the model, the relevant equilibrium concept is Nash Equilibrium: no university will have incentives to deviate given the decisions made by the other university. Note that given the nature of the problem we cannot be sure of the existence of such equilibrium -university payoffs are not continuous-and moreover, uniqueness cannot be guaranteed.

The aforementioned elements make it clear why analyzing the consequences that subsidized loan policies will have in the market of higher education is a complex problem. Let’s suppose that the government imposes such policy by subsidizing the interest rate of student loans. The first effect such policy will have is an increase in the number of students going to college. Note, however, that it is also not unreasonable to assume that the quality of the student body will change. This is because people who changed their decision to go to college are either those who were credit constrained or those having low ability levels that now decide to go to college given the decrease in the opportunity cost.

We can expect that after imposing such a policy, households will react by changing their decision of studying and universities should expect a change not only in the size of their student body but also in their quality. Given such changes, universities might want to change the prices charged to their students. This is due to the fact that as the quality of the student body changes, the productivity of investment will also be affected. Additionally, the willingness to pay for educational services is affected by such policy and universities will react to that. Moreover, universities might want to change the admission threshold either to improve the quality of their student body or to attract less able students.
that are willing to pay more for education. The overall effect depends on how sensitive is the demand for education with respect to the quality of services being provided.

Finally, note that -as said previously- the decisions of universities need to be analyzed in equilibrium. When deciding what is optimal, each college needs to take into account what their competitor is doing in the market and there should be no room for profitable deviations. After imposing a subsidized loan policy we might end up in an equilibrium where one college serves a specific part of the population. For instance one college serves a large demand for students with relatively low levels of ability whereas the other one specializes in providing high quality education for a reduced number of high ability students. Additionally, we can have a symmetric equilibrium where both firms are indistinguishable from one another or one in which only one firm operates in the market.

2.3.4 Government

We do not model the government as a welfare maximizing agent in the economy. We abstract from this fact and simply analyse the impact of the change in the government policies on the higher education market. However, we do interpret the student loan policy implementation as a way of the government to reduce the existent inefficiency in the educational market.

In a social planner’s solution, the efficient outcome would be one in which the high ability individuals decide to study, independent of their wealth. Thus, the role of the student loan policy can be interpreted as a way to reduce the existing inefficiency in the educational sector, although we do not model it as an optimal decision. We assume that the government has a borrowing constraint in the international borrowing markets, so is only able to finance the education of some fraction of the individuals in the economy. For now, we assume that the government finances individuals that have $\theta \geq \theta_{min}$, and of those
that can access the loans, subsidizes the interest rate on the loan for those individuals that have \( b \leq b_{\text{max}} \). The government sets thresholds \( \bar{b} \) and \( \theta_0 \), such that

\[
\begin{align*}
s \cdot (R - 1 - r) \cdot \int_{\Theta_2 \times B_2} \left( e^l(b, \theta) + e^h(b, \theta) \right) \times dF(\theta, b) = \tau \int_{\Theta \times B} b dF(\theta, b)
\end{align*}
\]

where \( \Theta_2 \times B_2 = (\Theta_1 \times B_1) \cap ([\theta_0, 1] \times [0, \bar{b}]) \) is the set of households who study and decide to take the subsidy.

**Definition 1** (Competitive Equilibrium). Given a set of government policies, \( \tau, s, b_{\text{max}}, \theta_{\text{min}}, \) and prices \( R, r, w \), a competitive equilibrium is a set of university policies \( (P_j, \theta_j^l, I_j^l) \) and household’s value function \( V(\theta, b) \) and policy functions \( c(\theta, b), a(\theta, b), e^h(\theta, b), e^l(\theta, b) \), such that:

1. Given \( \tau, s, b_{\text{max}}, \theta_{\text{min}}, R, r, w \), and university policies \( \{P_j^l, \theta_j^l, I_j^l\}_{j=h,l} \), the value function \( V(\theta, b) \) solves the household’s problem, with \( c(\theta, b), a(\theta, b), e^h(\theta, b) \) and \( e^l(\theta, b) \) being the corresponding policy functions.

2. For each university \( j = h, l \), it should hold that given \( \tau, s, b_{\text{max}}, \theta_{\text{min}} \); prices, \( R, r, w \); policy functions \( c(\theta, b), a(\theta, b), e^h(\theta, b), e^l(\theta, b) \); and policies from university \(-j\), \( (P^{-j}, \theta^{-j}, I^{-j}) \), university \( j \) chooses policies \( (P_j^l, \theta_j^l, I_j^l) \) that solve the university’s problem described in 10-14.

3. The government’s budget is balanced (equation 15 holds).

The nature of the problem makes it hard not only to compute the competitive equilibrium but also to show its existence. Note that, by only analyzing the supply side of the market, we cannot be sure that such an equilibrium will exist in this economy. In order to
compute the Nash equilibrium of the supply side of the market, we need to find pricing
and admission policies that are profit-maximizing given what the policies of the other
university.

The computation of such equilibrium is more involved when we note that there is an
additional fixed-point problem in the computation of the equilibrium. Universities offer
their students a given level of quality that needs to be self-fulfilled: the quality offered by
universities will attract certain students to the market but the quality of students going
to universities determines the quality offered by universities. It is not possible to use
any fixed-point theorem to show existence of a fixed point in this quality self-fulfilling
problem given that the necessary assumptions are not satisfied. In particular, note that
the fixed-point quality problem is not continuous as whenever the low-quality university
offers the same quality as the high one, all students who are beyond the ability threshold
will go to the cheapest one, generating a massive exit from one university to the other one,
generating a discontinuous jump in the quality being offered.

In order to illustrate this point extensively, we show in appendix .2.1 the failure to
prove existence of the equilibrium in the case of a linear utility function.

2.4 The “Revolución Educativa” of Colombia

In the present research, we will use Colombia as a natural experiment of a country that
implemented a rapid credit expansion program to alleviate credit constraints. Colombia is
a developing country which by the beginning of last decade had low enrollment rates in
post-secondary education, and significant differences in enrollment by quintiles of income.
As will be argued, the majority of students came from high-income families, and the
existence of financial constraints kept high-ability individuals from the lowest quintiles out
of the education market. During the last decade, the government engaged into the strategy
Revolución Educativa, aimed at increasing the education coverage at all levels. During the
decade, there were substantial increases in enrollment and educational credit access (see Figure 4).

(a) Enrollment and % of students with financial aid. (b) Average income and % of students with financial aid.

Figure 4: Enrollment, income and financial aid.

2.4.1 Enrollment and inequality

At the beginning of last decade, college enrollment in Colombia was among the lowest in Latin America and a student financial aid system was almost non-existent. In 2000, 23.2% of the people between 18 and 23 years old enrolled in tertiary education, below the enrollment rates of Bolivia, Peru, Brazil, Chile and Venezuela, and very close to the enrollment rates of Mexico. Because of a lack of a well-functioning financial aid system, less than 5%\(^{18}\) of the entering cohorts had any kind of public or private financial support (Bank, 2003, 2012). By the end of the decade, the enrollment rates grew to 37%, and reached 50% in 2015. The fraction of students with some type of credit increased to almost 25% of the entering cohorts (see Figure 4(a)).

Access to education has always been unequal and, despite the fast growth of enrollment, many disparities persist. In 2013, only 45% of the low-income students graduated from high school, and only 25% of them enrolled in tertiary education. Of the high-income households, 60% graduated from high school and 54% of them enrolled in a post-secondary

84

\(^{18}\) Extracted from the dataset of indicators for tertiary education, SPADIES, from the Ministry of Education.
institution (Melguizo et al., 2015). According to Bank (2003, 2012), the enrollment gap between the lowest and the highest quintiles of wealth widened throughout the decade: in 2001, the enrollement rates were of 8% in the lowest quintile and 41% in the highest, while in 2010 these numbers grew to 10% and 52%, respectively. If quality is taken into account, disparities are even larger as a larger proportion of the low-income students attend non-professional institutions, which have less resources and offer lower expected income in the future. Many theories have been used to explain the low enrollment of low-income students, such as disparities in the quality of public and private high school education, the high costs of tertiary education and the lack of a well-functioning financial aid system (Melguizo et al., 2015).

2.4.2 Higher Education institutions

The university system in Colombia functions as a monopolistically competitive market in which there are significant institutional barriers to entry, and universities do not have fixed “production capacities”, as assumed by Hector Chade and Smith (2014) (Figure 5). There are approximately 300 tertiary education institutions, of which around 190 are universities, and the rest offer non-professional degrees (mainly technical and technological). Despite the growing size of the entering cohorts throughout the decade, the number of institutions remained almost constant, while the average size of each institution doubled, on average. It is important to note that around 45 – 50% of the total student body is enrolled in private tertiary education institutions (OECD and Bank, 2012). Private institutions do not have any regulations regarding the price or investment per student they offer, although they have to satisfy a minimum quality requirement in terms of the programs and degrees offered. Therefore, the education market in Colombia can be studied as a monopolistically competitive market with barriers to entry and not subject to much government regulation.
In Colombia, every student that wants to graduate from high school has to present an exam called SABER11 set by the Colombian Institution for Education Evaluation (ICFES), similar to the SAT test in the U.S. Although not every tertiary education institution takes into account the results of the SABER11 in their admission decision, 78% use it as a criterion for admission (OECD and Bank, 2012). As SABER11 has no pass-mark, each institution sets its own minimum threshold for admission. In contrast to what happens in Chile and some European countries, in Colombia there is not any institution that clears the market for admissions, so individuals apply to as many institutions as they like and universities choose their admission standards independently (Melguizo et al., 2015). Although not perfect, the results in the SABER11 exam can be used as a proxy for the quality of the student body at universities. Figure 6 illustrates the average decile of the SABER11 scores of the entering cohorts to tertiary education institutions. Throughout the decade, universities seem to have adjusted their admissions standards in such a way that led to a reduction in the ability of the student body, as measured by relative position in the test scores.
2.4.3 The ACCES Program

To alleviate the low access, in 2002 the government implemented the credit program *Access with Quality to Higher Education*, ACCES, with the support of the World Bank, that massively increased the available credit to students. The credit is awarded to students that have test scores above a threshold set by the government, and covers up to 75% of the tuition for the lowest income students, and up to 50% for the rest. The credit has a subsidized zero-real interest rate for the poorest households, and a real interest rate of 8% for the high-income students. Students that graduate from their programs have twice the time of their study period to repay the loan. The ACCES program has full coverage, in the sense that any student that has test scores on the highest deciles of their region can access this credit line. The test score cut-offs vary by region, to account for disparities in the quality of secondary education across regions with different infrastructure and economic development. Given that the credit is awarded according to regional cutoffs, the disparities in the ability of people accessing the credit are large. The best students from the poorest regions might not have high ability and preparation when compared to the best students of the principal cities, so the credit is not awarded to the highest ability individuals in absolute terms.
Using a regression discontinuity approach, Melguizo et al. (2015) find evidence that the ACCES program had a positive impact on the enrollment rates, especially for individuals that come from poor households. Although the growth in the number of students enrolled in college may have been a consequence of other factors, such as better economic activity, the massive increase in financing seems to be a driving factor of such a trend.

2.4.4 Product Differentiation in the Market for Higher Education

In this subsection we introduce the dataset constructed to analyze the behavior of colleges before and after the introduction of the subsidized loan program. We use administrative data from the Ministry of Education including the SABER-PRO examination scores of each college. These are major-specific examinations that are mandatory in Colombia in order to receive the equivalent of a Bachelor’s degree. Additionally, we use publicly available information scrapped from the internet regarding the academic production of professors, as well as the academic credentials of the professorial body for each university. Moreover, we build information regarding the major-specific tuition charged by each college in order to track its behavior during the last ten years.

The analysis suggests that, after the introduction of the subsidized loan policy, elite institutions engaged in significant efforts to improve the quality of services provided. All the evidence suggest that once the subsidized loan policy was introduced, the gap in quality between elite and non-elite institutions increased. Figure 9 shows how the composition of the entering student body in elite institutions changed during the period of reference when compared to non-elite institutions. When ranked according to the decile in the distribution they are located by the SABER-11 examination score, we find that in 2007, students entering to elite institutions where, on average, located 1.5 deciles above the average student entering to non-elite universities. After five years we see that such gap increased to 1.7 and has remained constant until the last period of data available.
Figure 7: Differences in quality of student body

Note: Differences in the average decile of entering cohort in SABER-11 examination scores. This dataset is constructed using publicly available information provided by the Colombian Ministry of Education on its official website: http://www.mineducacion.gov.co/1759/w3-channel.html

Differences in the quality of student body are also observed when analyzing exit-level examination scores. Figure 8 shows the evolution of average test scores in written comprehension and reading comprehension, for students attending elite and non-elite colleges. The test scores are standardized to be mean zero and standard deviation one for every year in the dataset. Although in 2009 there were negligible differences between test scores of elite and non-elite colleges, we observe that in 2014 the average student graduating from an elite institution would score 60% of a standard deviation above the mean whereas students in non-elite institutions would score slightly below the mean. Taking into account that the average length of a bachelor’s degree program lasts 4.5 years, the score for 2014 corresponds to students who were entering in 2008, approximately. This fact is consistent with the scores for reading comprehension presented in panel B of the corresponding figure. Moreover, reading comprehension exams were being done since 2006 and thus we have a longer panel allowing us to infer that no significant changes were observed until cohorts graduating after 2010.
So far we have provided evidence suggesting that the gap between elite and non-elite institutions, when it comes to the quality of entering and exiting cohorts, increased after the introduction of the subsidized loan policy. However, the evidence suggest that the behavior of universities changed beyond the quality of their student body. Figure 9 shows that during the same period, elite universities engaged in significant efforts to increase the ratio of professors per students when compared to non-elite institutions. In 2007, the difference in the ratio of professors per student between elite and non-elite institutions, was under 0.02. However, for 2013 the difference more than doubled beyond 0.05.
Figure 9: Difference in professors per student

Note: This data is publicly available at the National System for Information on Higher Education website: [http://www.mineducacion.gov.co/sistemasdeinformacion/1735/w3-propertynamem2672.html](http://www.mineducacion.gov.co/sistemasdeinformacion/1735/w3-propertynamem2672.html)

We can go beyond the gross statistics of professors per student and analyze the academic credentials of the faculty composition of elite and non-elite colleges. In Colombia, it is not uncommon to see new hired faculty whose highest academic credential corresponds to a Bachelor’s or a Master’s degree. Taking into account this fact, the trend observed in Figure 9 would not imply by itself that elite institutions are making significant efforts to improve the quality of their faculty body. They might be substituting PhD professors by faculty whose highest academic credential is a Bachelor’s degree. However, In Figure 10 we find that the professors-student ratio of elite institutions increased when compared to non-elite institutions for every category of professors: those with a PhD, with a Master’s degree, and with a Bachelor’s degree.
Finally, the dataset also allows us to analyze the academic production of faculty from every college in Colombia. We construct a dataset of articles published in refereed journals by authors’ affiliation as well as total number of books by faculty. The results are presented in Figure 11. When we analyze the academic production per students, as measured by articles and books published, we also find evidence suggesting that the gap in academic production between elite and non-elite universities increased dramatically after the introduction of the subsidized loan policy.
(a) Writing score. (b) Reading comprehension score.

Figure 11: Gap in academic production.

Note: We constructed this dataset by scraping information available online at the Administrative Department of Science, Technology and Innovation (Colciencias) http://www.colciencias.gov.co/. A more detailed description of how this dataset was constructed is available in Spanish at http://laramaciudadana.com/universidades.html

Finally, we analyze the evolution of tuition being charged by higher education institutions. Figure 12 illustrates the behavior of the average real price of tuition during the decade, in terms of 2004 pesos. As can be observed, there has been a steady increase in the real price of education throughout the decade for all universities in Colombia. Additionally, the price of the high-quality colleges seems to have peaked at a higher pace for Law, Engineering and Medicine schools. This increasing trend suggests that the Bennett Hypothesis might also be taking place in the Colombian context, given the fast increase in the government provided loans to education.
Finally, we conclude this subsection by summarizing the main findings we observe from the data. We find that after the subsidized loan policy program was implemented in Colombia, the degree of product differentiation between elite and non-elite universities in Colombia increased. We conclude this after analyzing the trend of four key characteristics of universities in Colombia. First, the gap in the quality of student body increased dramatically when analyzing it via entering (SABER-11) or exiting (SABER-PRO) test
scores. Second, we observe a gap in the professors per student ratio for every possible category of professorship (PhD, Master’s and Bachelor’s degree). Third, the gap in academic production, measured as number of peer-reviewed articles and books published, per student, increased during the same period. Finally, we observe that in both, elite and non-elite institutions, there was a significant increase in the tuition being charged for some of the most popular degrees of study. All this evidence is consistent with the fact that after the introduction of subsidized loan policies, the gap in quality between elite and non-elite institutions increased significantly.

2.5 Numerical Analysis

In this section we perform a numerical analysis of the economic presented in Section 2.3 illustrating that the increased gap in quality between elite and non-elite institutions observed in the data, can be rationalized as a consequence of the introduction of subsidized loans for higher education.

2.5.1 Evolution of Quality

According to the specifications assumed in the model, we are able to identify the parameters of the wage equation. For this, we will use data on the average wages of graduates from each university in Colombia through 2007-2012, and the minimum wage, as a measure of \( w \), to estimate the parameters of the quality production function of universities. Per-efficiency unit wages are given by:

\[
w_h = w \cdot (1 + z_h), \quad w_l = w \cdot (1 + z_l)
\]
Where \( w_h \) and \( w_l \) are the wages of high- and low-quality college graduates, given equilibrium qualities of education \( z_h \) and \( z_l \), respectively, and \( w \) is the wage of non-skilled labor per-efficiency unit. The quality of education, \( z \), is given by equation (11) in the universities’ problem. We have a panel of data for 50 universities in Colombia from 2007 to 2012. We have the average ability of students in the entering cohorts, number of professors per student and average wages during the first year after graduation. For every university \( i \) in our sample, the following equation holds:

\[
w_i = w \cdot (1 + \kappa \bar{\theta}_i \bar{I}_i^{23})
\]

Rearranging and taking logarithms:

\[
\log \left( \frac{w_i}{w} - 1 \right) = \log \kappa + \alpha_1 \bar{\theta}_i + \alpha_2 I_i
\]

Assuming that there is measurement error in the wages of each of the universities, and assuming an exclusion restriction that the measurement error is uncorrelated with the explanatory variables, we can estimate the following equation:

\[
\log \left( \frac{w_{i,t}}{w_t} - 1 \right) = \log \kappa + \alpha_1 \bar{\theta}_{i,t} + \alpha_2 I_{i,t} + \eta T_{i,t} + \phi_t + \psi_i + \epsilon_{i,t}
\]

where \( T_{i,t} \) is an indicator function that takes the value of one when the university \( i \) is a low-quality institution, and zero otherwise. Under this specification, we can estimate possible differences in the technology parameter, \( \kappa \), between top and second tier schools. In order to isolate possible omitted variable bias, we estimate the above model under three different specifications, with and without time and geographic fixed effects, \( \phi_t \) and \( \psi_i \), respectively.
For the estimation, we constructed a panel of the top 50 universities in Colombia, according to a quality ranking published by the Ministry of Education in 2014\textsuperscript{19}. This panel includes data on average wages during the first year after graduation for graduates of every school, as a measure of $w_{i,t}$, the average test scores for the entering cohorts, as a measure of $\theta_{i,t}$, and the number of professors per student, as a measure for $I_{i,t}$. We also have data on total operational expenditures by each school for 2014. However, with only one year we are not able to construct the evolution of quality of universities over time. Since the number of professors per student are a good indicator of the total expenditures per student, we will use that variable, instead. For the non-skilled labor wages, $w_t$, we will use the values of the real minimum wage (in 2007 pesos). The average wages of college graduates are strictly above the minimum wage during the period, so the dependent variable is well defined for every college in every period. In addition, we have information about the municipality of the school, to control for regional differences. The results of the estimation are displayed in Table 1.

The estimates show that the elasticities $\alpha_1$ and $\alpha_2$ are fairly robust to different specifications and do not change dramatically when including control variables. Moreover, the parameter $\eta$ is negative in two of the specifications, although non statistically significative. This means that, on average, tier 2 universities have a lower technology parameter, $\kappa$, on their quality production function. This will be one of the main differences between tier 1 and tier 2 universities in our calibration of the model.

\subsection*{2.5.2 Calibration of the Model}

In order to draw conclusions about the relevance of our model, we calibrate the parameters to values that are relevant to the Colombian context. To achieve this, we will map a life-cycle model to a two period model, so the conclusions of Section 2.3 hold. We follow

\textsuperscript{19} The ranking is published in the website of the Ministry of Education, and can be found in the following link: http://www.mineducacion.gov.co/cvn/1665/w3-article-351855.html
<table>
<thead>
<tr>
<th>Parameter</th>
<th>OLS</th>
<th>OLS</th>
<th>OLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{\alpha}_1 )</td>
<td>0.211</td>
<td>0.228</td>
<td>0.168</td>
</tr>
<tr>
<td></td>
<td>(0.026)</td>
<td>(0.026)</td>
<td>(0.026)</td>
</tr>
<tr>
<td>( \hat{\alpha}_2 )</td>
<td>0.358</td>
<td>0.478</td>
<td>0.414</td>
</tr>
<tr>
<td></td>
<td>(0.361)</td>
<td>(0.357)</td>
<td>(0.403)</td>
</tr>
<tr>
<td>( \hat{\eta} )</td>
<td>-0.029</td>
<td>0.008</td>
<td>-0.046</td>
</tr>
<tr>
<td></td>
<td>(0.047)</td>
<td>(0.043)</td>
<td>(0.046)</td>
</tr>
<tr>
<td>( \log(\kappa) )</td>
<td>-0.84</td>
<td>-0.957</td>
<td>-0.163</td>
</tr>
<tr>
<td></td>
<td>(0.232)</td>
<td>(0.228)</td>
<td>(0.198)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Effect</th>
<th>Yes</th>
<th>Yes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Time fixed effects</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>City fixed effects</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( N )</td>
<td>382</td>
<td>382</td>
<td>382</td>
</tr>
<tr>
<td>R-squared</td>
<td>0.353</td>
<td>0.444</td>
<td>0.567</td>
</tr>
<tr>
<td>Prob &gt; F</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Robust standard errors in parenthesis

\*\*\* \( p < 0.01 \), \*\* \( p < 0.05 \), \* \( p < 0.1 \)

Table 1: Estimates for the quality production function.

an approach similar to the one used by Lochner and Monge-Naranjo (2011), but in a discrete time economy. The environment is as follows.

Individuals live for \( T \) periods, after which they die with certainty. Individuals start their adult life at \( t = 0 \), when they must choose whether to attend the low- or high-quality school, or not study at all. Studying lasts for \( S \) periods, so those individuals that attend college will not receive any income during \( t \in \{0, \ldots, S - 1\} \) and have to pay a per-period price of \( P_j \) for attending school \( j \). Moreover, during the first \( S \) periods individuals are borrowing constrained. Those that decide not to study cannot borrow at all. Those that decide to study, can borrow up to the exogenous limit set by the government student loan policy, \( \bar{A} \). After period \( S - 1 \), the individual enters the labor market and earns a per-period wage \( \theta w(1 + z_j) \), that depends on the quality of the school attended. During periods \( S, \ldots, T \) individuals only consume and save. We assume that from period \( S \) onwards, individuals enter into perfect financial markets where debt repayments are fully enforced. In this context, individuals can borrow any amount they want.
Clearly, individuals that are not borrowing constrained during their study period will perfectly smooth consumption along the life-cycle. However, those individuals that are constrained during the first $S$ years of life will exhibit a jump in their consumption once they graduate from college. This setting can be easily embedded into the two-period model described in last section, by setting the discount factors and budget constraints appropriately. Namely, the problem for the household becomes:

$$\max_{c,c'} \frac{c^{1-\sigma}}{1-\sigma} + \tilde{\beta} \frac{(c')^{1-\sigma}}{1-\sigma}, \quad \text{s.t.}$$

$$c + c' \left( \frac{\Phi_S}{\Phi_0 (1 + r)^S} \right) + (P_H h + P_l l) \left( \frac{\Phi_y^l}{\Phi_0} \right) =$$

$$w(1-h)(1-l) \left( \frac{\Phi_y^l}{\Phi_0} \right) + w\theta(1+z_j) \left( \frac{\Phi_y^0}{\Phi_0 (1 + r)^S} \right) j + \frac{b}{\Phi_0}$$

$$a \geq \bar{A}$$

The derivation of the parameters $\tilde{\beta}, \Phi_0, \Phi_S, \Phi_y^l, \Phi_y^0$ is explained in detail in the Appendix 2.1. In this environment, all the results from Section 2.3 hold.

### 2.5.3 Parameterization

In our calibration, we set one period to be exactly one year. We will set some parameter values to match the Colombian educational market. All parameter values are reported in Table 2.

We set $S = 5$, so that the individuals that choose to attend a college study during 5 periods, since most professional degrees in Colombia take exactly 5 years. In Colombia, life expectancy at birth is 73.95 years of life\[^{20}\]. Although the National Statistics Department

\[^{20}\] See life expectancy tables here.
of Colombia (DANE) does not publish the life expectancy by age, we estimate the life expectancy at 18 years to be 55 more years of life\textsuperscript{21}. That is, we set $T = 55$ to match the life expectancy in Colombia for high-school graduates.

We set $\sigma = 2$, which a standard parameter in the literature (Fernández-Villaverde and Krueger, 2011; Lochner and Monge-Naranjo, 2011). For the real interest rate, we choose $r = 8.9\%$, which is the value for Colombia in 2014 published by the World Bank\textsuperscript{22}. We do not claim that this value is representative of developing countries, since the real interest rate for most Latin American countries has a huge variation, ranging from negative values in Argentina ($-4.1\%$) and Venezuela ($-14.5\%$ in 2013), to very high values like Brazil ($23.5\%$). We choose $\beta = 0.92$ such that $\beta = 1/(1 + r)$. With these parameter values, the discount factor in our two-period model becomes $\tilde{\beta} = 1.89$. This reflects the fact that the post-college period is much longer than the study period, even though individuals discount time at a high rate.

As for the university parameters, we use the estimations of Section 2.5. In particular, we choose $\alpha_1 = 0.211$, $\alpha_2 = 0.358$, $\kappa_l = 0.8$ and $\kappa_h = 0.85$, obtained from the wage regressions displayed in Table 1.

\textsuperscript{21} For instance, in the U.S. life expectancy at 18 is only 0.79 more years than life expectancy at birth. Therefore, we will set life expectancy at 18 in Colombia to be 1.05 years above life expectancy at birth, as a conservative estimate.

\textsuperscript{22} See the real interest rates for all the countries in this link.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utility and discount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.97</td>
<td>Literature</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>2</td>
<td>Literature</td>
</tr>
<tr>
<td>$r$</td>
<td>2%</td>
<td>Colombia</td>
</tr>
<tr>
<td>$w$</td>
<td>2</td>
<td>Normalization</td>
</tr>
<tr>
<td>Time parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$T$</td>
<td>78</td>
<td>Colombia</td>
</tr>
<tr>
<td>$S$</td>
<td>5</td>
<td>Colombia</td>
</tr>
<tr>
<td>University parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\alpha_1$</td>
<td>0.211</td>
<td>Estimation</td>
</tr>
<tr>
<td>$\alpha_2$</td>
<td>0.358</td>
<td>Estimation</td>
</tr>
<tr>
<td>$\kappa_l$</td>
<td>1.4</td>
<td>Estimation</td>
</tr>
<tr>
<td>$\kappa_h$</td>
<td>1.2</td>
<td>Estimation</td>
</tr>
<tr>
<td>$E^h - C^h$</td>
<td>-12</td>
<td>Estimation</td>
</tr>
<tr>
<td>$E^l - C^l$</td>
<td>-7</td>
<td>Estimation</td>
</tr>
</tbody>
</table>

Table 2: Parameter values

2.5.4 Results

In this section we show the results of the numerical computation of the equilibria without the subsidized-loan policy being implemented and once it was implemented. In order to mimic as closely as possible the post-reform equilibrium, we set up a tax rate of 10%
used to fund a subsidized loan policy offering credits for higher education for people whose income is below the median income in the economy. The policy implemented in Colombia is designed as a subsidy to the interest rate paid by students. In the model we set up the subsidy in such a way that students that have access to it only have to pay 50% of the interests accumulated in students debts. In addition to having an income below the median, a student who wants to qualify for the policy must have an ability level in the top 30%.

Table 3 illustrates the results before and after the implementation of the student loan policies. As can be observed, after the reform there is a widening gap in the quality offered by each university. Elite universities offer a higher quality, while non-elite universities reduce it. There is also a market segmentation, where better students attend the elite institution, and the ability of the students attending the low-quality institutions falls.

<table>
<thead>
<tr>
<th></th>
<th>Pre-reform</th>
<th>Post-reform</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Elite institutions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Students attending</td>
<td>5,863</td>
<td>9,431</td>
</tr>
<tr>
<td>Average ability of student body</td>
<td>0.48</td>
<td>0.64</td>
</tr>
<tr>
<td>Quality offered</td>
<td>1.01</td>
<td>1.19</td>
</tr>
<tr>
<td><strong>Non-elite institutions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Students attending</td>
<td>6,971</td>
<td>6,753</td>
</tr>
<tr>
<td>Average ability of student body</td>
<td>0.41</td>
<td>0.38</td>
</tr>
<tr>
<td>Quality offered</td>
<td>0.53</td>
<td>0.42</td>
</tr>
</tbody>
</table>

23. The institutional details of the policy implemented in Colombia are fully described in Melguizo et al. (2015)
2.6 Conclusion

Subsidized loan policies have been widely implemented in both developing and developed economies, as a policy tool to increase college attendance. Such policies are particularly relevant in a context where credit constraints explain college attendance decisions and individuals are borrowing constrained. However, when implementing such policies it is important to take into account, not only their effect on the demand side of the market, but also the way they affect the supply side of the market by changing the incentives of the providers of higher education.

We show that subsidized loan policies can distort the incentives of colleges providing services of higher education in a way that can be harmful for a group of households in the economy. Taking into account that the market for higher education operates under a monopolistic competition setting, granting subsidized loans does not generate an expansion in the number of providers of higher education, but rather on the same colleges facing a new set of incentives. As elite institutions unambiguously observe an increase in their demand, they can use their pricing and admission policies to be more selective in their admission process and to spend more per student, which translates into providing better services for their student body. On the contrary, the universities in the low-quality tier will observe a migration to the high-quality group when such policies are implemented. The result is a new equilibrium in the market for higher education where the quality gap between elite and non-elite institutions is widened as a result of the implementation of subsidized loan policies.

Our model is consistent with what we observe in the market for higher education in Colombia: an expansion of the gap in the quality offered by different institutions. In such scenario, subsidized policy loans can make some households worse off as, although the attendance to higher education institutions becomes easier, the gains from attending
low-quality universities is not offset by the amount households have to pay in taxes in order to pay for the policy implemented.
Chapter 3

A Practical Guide to Parallelization in Economics

3.1 Introduction

Economists, more than ever, need high-performance computing. In macroeconomics, we want to solve models with complex constraints and heterogeneous agents to simultaneously match micro and aggregate observations. In industrial organization, we aim to characterize the equilibrium dynamics of industries with several players and multiple state variables. In asset pricing, we seek to price complex assets in rich environments with numerous state variables. In corporate finance, we like to track the behavior of firms with rich balance sheets and intertemporal choices of funding. In international economics, we are interested in analyzing multisectoral models of trade. In econometrics, we have to evaluate and simulate from moment conditions and likelihood functions that often fail to have closed-form solutions. And machine learning and big data are becoming ubiquitous in the field.
One of the most important strategies to achieve the required performance to solve and estimate the models cited above is to take advantage of parallel computing. Nowadays, even basic laptops come from the factory with multiple cores (physical or virtual), either in one central processing unit (CPU) or several CPUs. And nearly all of them come with a graphics processing unit (GPU) that can be employed for numerical computations. Many departments of economics and most universities have large servers that researchers can operate (a few even have supercomputer centers that welcome economists). Furthermore, cloud computing services, such as Amazon Elastic Compute Cloud or Google Cloud Compute Engine, offer, for economical prices, access to large servers with dozens of CPUs and GPUs, effectively making massively parallel programming available to all academic economists.

Unfortunately, there are barriers to engaging in parallel computing in economics. Most books that deal with parallelization are aimed at natural scientists and engineers. The examples presented and the recommendations outlined, valuable as they are in those areas, are sometimes hard to translate into applications in economics. And graduate students in economics have taken rarely many courses in programming and computer science.

Over the years, we have taught parallel programming to many cohorts of young economists. Thus, it has become clear to us that an introductory guide to parallelization in economics would be useful both as a basis for our lectures and as a resource for students and junior researchers at other institutions.

In this guide, we discuss, first, why much of modern scientific computing is done in parallel (Section 3.2). Then, we move on to explain what parallel programming is with two simple examples (Section 3.3). And while parallel programming is a fantastic way to improve the performance of solving many problems, it is not suitable for all applications. We explain this point in Section 3.4 with two fundamental problems in economics: a value function iteration and a Markov chain Monte Carlo. While the former is perfectly gathered for parallelization, the latter is much more challenging to parallelize. That is why, in
Section 3.5, we introduce a life-cycle model as our testbed for the rest of the paper. The life-cycle model is sufficiently rich as to be an approximation to problems that a researcher would like to solve in “real life.” At the same time, parallelization of the model is not trivial. While we can parallelize its solution over the assets and productivity variables, we cannot parallelize over the age of the individual. Section 3.6, then, briefly introduces the different types of computers that a typical economist might encounter in her parallel computations. We spend some time explaining what a cloud service is and how to take advantage of it.

Sections 3.7 and 3.8 are the core of the paper. In Section 3.7, we outline the main approaches to parallelization in CPUs, with Julia, Matlab, R, Python, C++--OpenMP, Rcpp--OpenMP, and C++--MPI. In Section 3.8, we present CUDA and OpenACC. In these sections, we discuss the strengths and weaknesses of each approach while we compute the same life-cycle model that we presented in Section 3.5. We do not envision that a reader will study through each part of these two sections in full detail. A reader can, for instance, pick those approaches that she finds more attractive and study them thoroughly while only browsing the other ones (although we strongly recommend reading Subsection 3.7.1, as it presents several key ideas that we will repeat for all programming languages).

Regardless of the situation, the reader should follow our explanations while checking the code we have posted at the Github repository of this paper and forking it with her improvements. Learning computation without trying it oneself is nearly impossible, and in the case of parallelization, even more so. We cannot stress this point enough.

We conclude the guide with a mention in Section 3.9 of existing alternatives for parallel computation beyond CPUs and GPUs (such as manycore processors, field-programmable gate arrays, and tensor processor units) and with some brief remarks summarizing our computational results in Section 3.10.

Before we get into the main body of the guide, we must highlight three points. First, this guide’s goal is pedagogical. Our codes were written putting clarity ahead of performance
as a basic criterion. We did not optimize them to each language or employ advanced coding styles. Instead, we compute the same model in the same way across languages and compare the performance of parallel computing, using as the benchmark the serial computation for each language. Most students in economics do not come to the field from computer science or engineering, and they prefer clearer codes to sophisticated ones. This means that, for example, we will employ too few anonymous functions, too many loops, and too much old-style imperative programming (also vectorization and related techniques work less well in the problems we deal with in economics than in other fields; see Aruoba and Fernández-Villaverde, 2015). While there is much to be said about improving coding style among economists, that is a battle for another day. You teach one lesson at a time.

Second, we do not cover all possible combinations of languages, parallelization techniques, and computers. Doing so would be next to impossible and, in any case, not particularly useful. We are, instead, highly selective. There is no point, for instance, in covering parallelization in Java or Swift. Neither of these languages is designed for scientific computation, nor are they widely used by economists. Among programming languages, perhaps the only absence that some readers might miss is Fortran. In previous related exercises (Aruoba and Fernández-Villaverde, 2015), we kept Fortran more as a nod to old times than out of conviction. But, by now, we feel justified in dropping it. Not only does Fortran not deliver any speed advantage with respect to C++, but it also perpetuates approaches to numerical computation that are increasingly obsolete in a world of massive parallelization. To program well in parallel, for instance, one needs to understand MapReduce and employ rich data structures. Fortran fail to meet those requirements. While keeping legacy code and libraries may require knowledge of Fortran for some researchers, there is no point in young students learning what is by now an outdated language despite all the efforts to modernize it in the most recent standards (such as the soon-to-be-released Fortran 2018). Among programming techniques, we
skip POSIX and Boost threads and threading building blocks (too low level for most users) and OpenCL (as we believe that OpenACC is a better way to go).

Perhaps the only topic of importance that we miss is the use of Spark (possibly with Scala), a general-purpose cluster computing system particularly well-adapted for the manipulation of large data sets.\textsuperscript{24} Future editions of this guide might correct that shortcoming, but the current version is sufficiently long, and we are afraid of losing focus by introducing a whole new set of issues related to data parallelization.

Third, we will not explain how to write algorithms in parallel since this would require a book in itself (although we will cite several relevant books as we move along as references for the interested reader). This lack of explanation implies that we will not enter into a taxonomy of computer architectures and parallel programming models (such as the celebrated Flynn’s classification) beyond passing references to the differences between parallelism in tasks and parallelism in data or the comparison between the shared memory of OpenMP and the message passing structure of MPI.

With these three considerations in mind, we can enter into the main body of the guide.

\section*{3.2 Why Parallel?}

In 1965, Gordon Moore, the co-founder of Intel, enunciated the most famous law of the computer world: the number of components (i.e., transistors) per integrated circuit (i.e., per “chip”) will double each year (Moore, 1965). Although this number was later downgraded by Moore himself to a mere doubling every two years (Moore, 1975), this path of exponential growth in the quantity of transistor chips incorporated is nothing short of astonishing. Moore’s law predicts that the computational capability of human-made machines will advance as much during the next 24 months as it has done from the dawn of mechanical devices until today.

\textsuperscript{24} See \url{https://spark.apache.org/}.
Figure 1: Number of transistors

But Moore’s law has been nearly as breathtaking in terms of its forecasting success. Figure 1 plots how the number of transistors has evolved from 1971 to 2017. Moore’s law predicted that the representative chip would have 8,388,608 times more transistors by 2017 than in 1971. Indeed, the 2017 32-core AMD Epyc processor has 8,347,800 times more transistors than the 1971 Intel 4004 processor. If you look, however, at the end of the sample, there are some indications that the law may be slowing down to a doubling of transistor count every 2.5 years: the 32-core AMD Epyc processor has many more transistors than other processors of its generation and it achieves that density because it suffers from higher cache latency.

The recent hints of a slowdown in the rate of transistor growth might be telltale signs of future developments. Although the demise of Moore’s law has been announced in the past and technology has kept on surprising skeptics, there are reasons to believe that we are approaching the maximum number of transistors in a chip, perhaps in a decade or two. Not only does the electricity consumption of a chip go up by $x^4$ when the transistor

---

25. Data from Figure 1 come from [https://en.wikipedia.org/wiki/Transistor_count](https://en.wikipedia.org/wiki/Transistor_count).
size falls by a factor $x$, but we must endure considerable increases in the heat generated by the chip and in its manufacturing costs.

At a more fundamental level, there are inherent limits on serial chips imposed by the speed of light (30 cm/ns) and the transmission limit of copper wire (9 cm/ns). This means that there are hard constraints on the speed at which a processor can compute an operation and that it is virtually impossible to build a serial Teraflop machine without resorting to parallelization.\footnote{A Teraflop is $10^{12}$ floating point operations per second (i.e., the number of calculations involving real numbers). As a point of comparison, the Microsoft Xbox One X has a top theoretical performance of 6 Teraflops thanks to its aggressive reliance on GPUs.} Even more relevant is the observation that the real bottleneck for scientific computing is often memory access. Random Access Memory (RAM) latency has been only improving around 10 percent a year.

![Figure 2: Cray-1, 1975](image)

That is why the industry has moved from building machines such as the serial Cray-1, in 1975 (Figure 2), to the massively parallel Sunway TaihuLight (Figure 3), the fastest
supercomputer in the world as of April 2018.\textsuperscript{27} The Cray-1, built around the idea of vector processing, aimed to deliver high performance through the simultaneous operations on one-dimensional arrays (vectors, in more traditional mathematical language).\textsuperscript{28} Although the Cray-1 represented a leap over existing machines, it required a specific arranging of the memory and other components (the circular benches around the main machine that can be seen in Figure 2 were essential for the power supply to be delivered at the right place) as well as quite specialized software. These specificities meant that the cost curve could not be bent sufficiently quickly and that the limitations of single processor machines could not be overcome.

![Figure 3: Sunway TaihuLight, 2016](link to Wikipedia image of Figure 2 and link to Flickr image of Figure 3)

The solution to these limits is employing more processors. This is the route followed by the Sunway TaihuLight. The computer is built around 40,960 manycore processors, each with 256 cores, for a total of 10,649,600 cores. Instead of a single piece of machinery like the Cray-1, Figure 3 shows rows of cabinets with racks of processors, each handling a

\textsuperscript{27} Link to Wikipedia image of Figure 2 and link to Flickr image of Figure 3.  
\textsuperscript{28} This computer was named after Seymour Cray (1925-1996), whose pioneering vision created modern high-performance computing (see Murray, 1997). The company he founded to manufacture this machine, Cray Research, led the industry for decades. Modern-day Cray Inc. descends from the original Cray Research after several turbulent changes in ownership. An early example of a parallel computer was the pathbreaking CDC 6600, released in 1964.
part of some computation. Thanks to this parallelization, the 160 Megaflops ($10^6$) Cray-1 has been replaced by the 125.4 Petaflop ($10^{15}$) Sunway TaihuLight, an increase in speed of around $0.78 \times 10^9$. Without these speed gains, everyday computational jobs on which we have grown to depend –such as high-definition video rendering, machine learning, search engines, or the tasks behind Spotify, Facebook, and blockchain– would not be feasible.

3.3 What Is Parallel Programming?

The main idea behind parallel programming is deceptively simple: if we have access to several processors (either in the same computer or in networked computers), we can divide a complex problem into easier pieces and send each of these components to a different processor. Within the field of high-performance computing, this division can be done either for numerical analysis—for example, to multiply two matrices— or for the handling of large amounts of data—for instance, the computation of a sample average through MapReduce.

Let us develop these two examples to understand the basic idea. In our first example, we have two matrices:

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

and

$$B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

and we want to find:

$$C = A \times B = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}.$$
Imagine, as well, that we have access to a machine with five processors. We can select one of these processors as a “master” and the other four as “workers” (sometimes, and less delicately, also called “slaves”). The master processor will send \((a_{11}, a_{12}, b_{11}, b_{21})\) to the first worker and ask it to compute \(a_{11}b_{11} + a_{12}b_{21}\). Similarly, it will send \((a_{11}, a_{12}, b_{12}, b_{22})\) to the second worker and ask it to compute \(a_{11}b_{12} + a_{12}b_{22}\) and so on with the third and fourth workers. Then, the results of the four computations will be returned to the master, which will put all of them together to generate \(C\). The advantage of this approach is that while the first worker is computing \(a_{11}b_{11} + a_{12}b_{21}\), the second worker can, at the same time, find \(a_{11}b_{12} + a_{12}b_{22}\). In comparison, a serial code in one processor will need to finish the computation of \(a_{11}b_{11} + a_{12}b_{21}\) before moving into computing \(a_{11}b_{12} + a_{12}b_{22}\).

Doing a parallelization to multiply two matrices \(2 \times 2\) is not efficient: we will lose more time transferring information between master and workers than we will gain from parallelizing the computation of each of the four entries of \(C\). But a simple extension of this algorithm will parallelize the multiplication of matrices with hundreds of rows and columns, a costly task that appears in all sorts of scientific computations (for instance, when we compute a standard OLS estimator with many regressors and large data sets), and generate significant time gains.

In our second example, we want to find the average of a sample of observations:

\[
y = \{1, 2, 3, 4, 5, 6, 7, 8\}
\]

and, again, we have access to a machine with five processors divided between a master and four workers. The master can send \(\{1, 2\}\) to the first worker to compute the average of the first two observations, \(\{3, 4\}\) to the second worker to compute the average of the next two observations, and so on with the third and fourth workers. Then, the four averages are returned to the master, which computes the average of the four sub-averages \(\{1.5, 3.5, 5.5, 7.5\}\). Thanks to the linearity of the average operator, the average of sub-
averages, 4.5, is indeed the average of \( y \). As was the case for our matrix multiplication example, computing the average of 8 observations by parallelization is inefficient because of the overheads in transmitting information between master and workers.

But this approach is a transparent example of the MapReduce programming model, a successful paradigm for the handling of millions of observations (among other tasks, later we will show how it can be applied as well to the solution of a life-cycle model). MapReduce is composed of two steps: a Map operator that takes an input and generates a set of intermediate values (in this case the input is \( y \) and the intermediate inputs are the sub-sample averages) and a Reduce operator that transforms the intermediate values into a useful final value, which, in our example, computes the average of the four sub-sample averages. In addition, MapReduce can be applied recursively, and therefore, the final value can be used as an intermediate input in another computation.\(^{29}\)

The previous two examples highlight two separate issues: first, the need to develop algorithms that divide problems of interest into components that can be parallelized; second, the need to have coding techniques that can “tell” the computer how to distribute the work between a master and workers.

In the next pages, we can offer only a very parsimonious introduction for economists to these issues, and the interested reader should consult more complete references. A few books we have found particularly useful to extend our knowledge include:


\(^{29}\) MapReduce’s name derives from two combinators in Lisp, a deeply influential functional programming language created in 1958, map and reduce. Google developed the modern-day MapReduce approach to handle the immense amounts of data its search engine requires. Hadoop is a popular implementation of MapReduce. Currently, a more efficient alternative, Spark, based on the resilient distributed data set (RDD), is quickly overtaking Hadoop as a weapon of choice in handling large data problems with massive parallelization.


Below, we will cite some additional books as we deal with the concrete aspects of parallelization.

### 3.4 When Do We Parallelize?

The two examples above show the promise of parallelization for high-performance computing, but also its potential drawbacks regarding algorithmic design and communication among processors.

![Granularity Diagram](image)

Figure 4: Granularity
Problems in high-performance computing can be classified according to their scalability (i.e., how effectively we can divide the original problem of interest into smaller tasks) and granularity (i.e., the measure of the amount of work involved in each computational task in which the original problem of interest is subdivided). Depending on their scalability, problems are either strongly scalable –i.e., inherently easy to parallelize– and weakly scalable –i.e., inherently difficult to parallelize. Depending on their granularity, problems can be coarse –i.e., requiring more computation than communication between a master and workers –or fine –i.e., requiring more communication than computation. In Figure 4, we plot two columns representing the tasks involved in the computation of two problems. The left column is a computation requiring mostly computation and little communication among processors. The right column, in comparison, is a problem with much more communication. While the former problem is a good candidate for parallelization, the latter is not.

However, whether the problem is easy to parallelize may depend on the way you set it up. You need to take advantage of the architecture of your computer and exploit it to the maximum. This will help you to control for overheads (for instance, in the example of sample average computation, the final sum of sub-sample averages) and achieve optimal load balancing (we need to assign to each different processor chunks of the problem that require roughly the same amount of time).

These considerations dictate relying on productivity tools such as a state-of-the-art IDE (integrated development environment), a debugger, and a profiler that can handle parallel code. Without these tools, coding, debugging, and profiling your program for accuracy and performance can become most difficult, as there is much more potential for mistakes and inefficiencies in a parallel code than in a serial one. Also, these productivity tools may help you achieve a better tradeoff between speed-ups delivered by parallelization and coding time. There is no point in parallelizing a code if doing so takes more time than the
savings in running time, even after considering the numerous times a code might need to be run during a whole research project.

To illustrate some of these ideas, we introduce two common applications in economics: one where parallelization works wonderfully well and one where it struggles to achieve speed improvements.

### 3.4.1 An application where parallelization works: Value function iteration

Our first application is a computational problem where parallelization works with ease: the canonical value function iteration problem.

Let us assume that we have a social planner that is maximizing the discounted utility of a representative household in a standard deterministic neoclassical growth model. The recursive representation of the problem of this social planner can be written in terms of a value function $V(\cdot)$ as:

$$
V(k) = \max_{k'} \left\{ u(c) + \beta V(k') \right\}
$$

s.t. $c = k^\alpha + (1 - \delta) k - k'$

where $k$ is aggregate capital in the current period, $k'$ the aggregate capital in the next period, $c$ the consumption of the representative household, $u(\cdot)$ the period utility function, $\beta$ the discount factor, $\alpha$ the elasticity of output to capital, and $\delta$ the depreciation rate.

The solution of the social planner’s problem can be found by applying value function iteration. Imagine that we have access to $j$ processors in a parallel pool. Then, we can implement the following algorithm:
Data:
Grid of capital with \( j \) points, \( k \in [k_1, k_2, \ldots, k_j] \).
Initial guess of value function over the capital grid \( V^0(k) \).
Tolerance \( \epsilon \).

Result:
Converged value function \( V^\infty(k) \)
Optimal decision rule \( k' = g^\infty(k) \)

while \( \sup |V^n(k) - V^{n-1}(k)| < \epsilon \) do
    Send the problem:
    \[
    \max_{k'} \{ u(c) + \beta V^n(k') \} \\
    \text{s.t. } c = k_1^i + (1 - \delta) k_1 - k'
    \]
    to processor 1 to get \( V^{n+1}(k_1) \). Store optimal decision rule \( k' = g^{n+1}(k_1) \).
    Send the problem associated with \( k_i \) to worker \( i \).
    When all processors are done, gather \( V^{n+1}(k_i) \) for \( k \in [k_1, k_2, \ldots, k_j] \) back and construct \( V^n(k) \).
end

Algorithm 1: Parallelized value function iteration

The intuition of the algorithm is straightforward. Given a current guess of the value function \( V^n(k) \), processor \( m \leq j \) can solve for the optimal choice of capital next period given capital \( k_m \) without worrying about what other processors are doing for other points in the grid of capital. Once every processor has solved for the optimal choice of capital, we can gather all \( V^{n+1}(k_m) \) and move to the next iteration. This scheme is often called the “fork-join” paradigm.

Figure 5 illustrates the execution of the fork-join parallel code that would compute the model presented above. At every iteration \( n \), the master node splits the computation
across the parallel workers (fork). When all processors are done with the computations, the information is passed back to the master node (join) and the process is repeated.

The algorithm is easily generalized to the case where we have more grid points $j$ than workers $w$ by sending to each processor $\lfloor \frac{j}{w} \rfloor$ grid points (and the remaining points $j - w \times \lfloor \frac{j}{w} \rfloor$ to any worker). Furthermore, this parallelization algorithm is efficient, as it allows an increase in speed that can be nearly linear in the number of processors, and it is easy to code and debug.$^{30}$

3.4.2 An application where parallelization struggles: A random walk

Metropolis-Hastings

Our second application is a computational task where parallelization faces a difficult time: a random walk Metropolis-Hastings, the archetypical Markov chain Monte Carlo method.

---

$^{30}$ The speed-up will depend on the quantity of information moved and the quality of the hardware performing the communication among the processors. Also, the gain in efficiency is diminished by the difficulty in applying properties of the solution such as the monotonicity of the decision rule of the social planner (i.e., $g^w(k_j) < g^w(k_{j+1})$) that simplify the computation involved in the max operator in the Bellman equation. However, when the number of grid points is sufficiently larger than the number of workers, this difficulty is only a minor inconvenience. Furthermore, there are several “tricks” to incorporate additional information in the iteration to capture most of the gains from monotonicity.
Many problems in econometrics require drawing from a distribution. More concretely, we want to draw a sequence of $\theta_1, \theta_2, \theta_3, \ldots$, such that:

$$\theta_i \sim P(\cdot).$$

This draw may allow us to simulate some stochastic process where the $\theta$’s are the innovations or to evaluate functions of interest of the distribution –such as its moments– appealing to a law of large numbers (i.e., the theoretical mean of the distribution will be well approximated by the mean of the drawn $\theta$’s).

In principle, computers face a difficult time drawing a random number: they are, after all, deterministic machines that follow a predetermined set of instructions. Computer scientists and applied mathematicians have developed, nevertheless, deterministic algorithms that generate a sequence of pseudo-random numbers and whose properties are, up to some cycle, very close to those of a truly random sequence.\(^31\)

These pseudo-random number generators typically produce a sequence of draws that mimics a draw from a uniform distribution in $[0, 1]$. Through different change-of-distribution transformations, such a sequence from a uniform distribution can yield a sequence from a wide family of known parametric distributions (Devroye, 1986, provides an encyclopedic treatment of such transformations).

The challenge appears when the distribution $P(\cdot)$ does not belong to any known family. In fact, it is often the case that we cannot even write down $P(\cdot)$, although we can numerically evaluate it at an arbitrary point $\theta_i$. This happens, for example, with the likelihood function of a dynamic equilibrium model: a filter (either linear, such as the Kalman filter, or non-linear, such as the particle filter in Fernández-Villaverde and Rubio-Ramírez, 2007) provides the researcher with a procedure to evaluate the likelihood

\(^{31}\) When true randomness is absolutely required, such as in cryptography, one can rely on different physical phenomena built into hardware with true random number generators. This is, however, rarely necessary in economics.
of the model given some observations for arbitrary parameter values, but not how to write a closed-form expression for it. We want to draw from this likelihood function to implement a Bayesian approach (after the likelihood has been multiplied by a prior) or to maximize it, from a frequentist perspective, through some stochastic optimization search algorithm.

A random walk Metropolis-Hastings is an algorithm that allows us to draw from an arbitrary distribution a Markov chain whose ergodic properties replicate those of the distribution of interest. The arrival of the random walk Metropolis-Hastings and other Markov chain Monte Carlo methods in the 1990s revolutionized econometrics and led to an explosion of research in Bayesian methods (Robert, 2007) and estimation by simulation (Gourieroux and Monfort, 1997).

The algorithm is as follows:
Data:
Distribution of interest \( P(\cdot) \).
Initial value of the chain \( \theta_1 \).
Scaling of the innovation \( \lambda \).

Result:
Sequence \( \{\theta_1, \theta_2, ..., \theta_m\} \).

while \( n \leq m \) do

<table>
<thead>
<tr>
<th>Given a state of the chain ( \theta_{n-1} ), generate a proposal:</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \theta^* = \theta_{n-1} + \lambda \varepsilon, \ \varepsilon \sim \mathcal{N}(0, 1) )</td>
</tr>
</tbody>
</table>

Compute:
\[ \alpha = \min \left\{ 1, \frac{P(\theta^*)}{P(\theta_{n-1})} \right\} \]

Set:
\[ \theta_n = \theta^* \] with probability \( \alpha \)
\[ \theta_n = \theta_{n-1} \] with probability \( 1 - \alpha \)

end

Algorithm 2: Random walk Metropolis-Hastings

The key of the algorithm is to generate a proposal \( \theta^* \) for the new state of the chain through a random walk specification

\[ \theta^* = \theta_{n-1} + \lambda \varepsilon, \ \varepsilon \sim \mathcal{N}(0, 1) \]
(the proposal is the new state plus an innovation) that is accepted with some probability $\alpha$

$$\alpha = \min \left\{ 1, \frac{P(\theta^*)}{P(\theta_{n-1})} \right\}$$

construed to induce the desired ergodic properties of the drawn sequence. By appropriately tuning $\lambda$, we can get an optimal acceptance rate and achieve faster convergence (Roberts et al., 1997).

From the perspective of parallelization, the problem of the algorithm is that to generate $\theta^*$, we need $\theta_{n-1}$, so we cannot instruct processor $i$ to compute $\theta_{n-1}$ and, at the same time, instruct processor $j$ to compute $\theta_n$. The Markov property of the chain makes it an inherently serial algorithm. Thus, we cannot easily subdivide the problem. The “obvious” solution of running parallel chains fails because it violates the asymptotic properties of the chain: we need a long sequence of size $m$, not many short sequences that add up to $m$, as the latter might not have traveled to the ergodic regions of the distribution.

Researchers have attempted to come up with proposals to parallelize the random walk Metropolis-Hastings algorithm (see, among others, Ingvar, 2010, and Calderhead, 2014). Also, in economics, other parts of the problem (such as the evaluation of the distribution $P(\cdot)$) can be parallelized. For example, if $P(\cdot)$ comes from a particle filter, such a filter is easy to parallelize. But despite these partial remedies, at the end of the day, the random walk Metropolis-Hastings algorithm is not a good candidate for parallelization. Unfortunately, many other problems in economics (such as those involving long simulations) suffer from the same drawback. Parallelization is a great tool for many computational challenges in economics, but not for all.
3.4.3 The road ahead

We will now move on to show how to parallelize in a realistic application in economics: a life-cycle model of consumption-saving. Slightly more complex versions of this model are at the core of much research in economics regarding consumption and savings across ages (Fernández-Villaverde and Krueger, 2011), portfolio choice (Cocco, 2005), household formation (Chang, 2017), human capital accumulation (Ben-Porath, 1967), housing decisions (Zarruk-Valencia, 2017), taxation policies (Nishiyama and Smetters, 2014), social security reform (Conesa and Krueger, 1999), etc. Not only is the application simple enough to fulfill its pedagogical goals, but it also presents sufficient sophistication as to illustrate the most relevant features of parallelization.

3.5 A Life-Cycle Model

We consider the following model for our experiments. We have an economy populated by a continuum of individuals of measure one who live for $T$ periods. Individuals supply inelastically one unit of labor each period and receive labor income according to a market-determined wage, $w$, and an idiosyncratic and uninsurable productivity shock $e$. This shock follows an age-independent Markov chain, with the probability of moving from shock $e_j$ to $e_k$ given by $P(e_k|e_j)$. Individuals have access to one-period risk-free bonds, $x$, with return given by a market-determined net interest rate, $r$.

Given the age, $t$, an exogenous productivity shock, $e$, and savings from last period, $x$, the individual chooses the optimal amount of consumption, $c$, and savings to carry for
next period, $x'$. The problem of the household during periods $t \in \{1, \ldots, T - 1\}$ is described by the following Bellman equation:

$$V(t, x, e) = \max_{\{c \geq 0, x'\}} \{u(c) + \beta \mathbb{E}V(t + 1, x', e')\} \text{ s.t. } c + x' = (1 + r)x + ew$$

$$e' \sim P(e'|e),$$

for some initial savings and productivity shock. The problem of the household in period $T$ is just to consume all resources, yielding a terminal condition for the value function:

$$V(T, x, e) = u((1 + r)x + ew).$$

A terminal condition, such as a value of bequests or altruism for future generations, could be easily added.

As referred to earlier, even this simple life-cycle model has important features related to parallelization. First, it is a problem with three state variables. This means that if the number of grid points on each variable is sufficiently large, the model might take some time to compute. If another state variable is included, the model might become infeasible to estimate. For instance, any model that studies health, housing, durable goods, or the consequences of having multiple financial assets implies keeping track of additional state variables and estimation requires the repeated solution of the model for many different parameter values. Second, although there are three state variables, the problem is not parallelizable along the three dimensions, as will become clear later.

---

32. For the purpose of this paper, we do not need to worry too much about how we specify those initial conditions, and we would compute the solution of the model for all possible initial conditions within a grid. In real research applications, the initial conditions are usually given by the desire to match observed data.
The following pseudo-code summarizes how to compute the model in serial using a standard backward-induction scheme:

**Data:**

- Grid for assets $X = \{x_1, \ldots, x_{n_x}\}$.
- Grid for shocks $E = \{e_1, \ldots, e_{n_e}\}$.
- Transition matrix $P(e'|e)$.

**Result:**

Value function $V(t, x_i, e_j)$ for $t = 1, ..., T$, $i = 1, ..., n_x$, and $j = 1, ..., n_e$.

For $\forall x_i \in X$ and $e_j \in E$ do

$$V(T, x_i, e_j) = u((1 + r)x_i + e_jw)$$

End

For $t = T - 1, \ldots, 1$, do

Use $V(t + 1, x_i, e_j)$ to solve:

$$V(t, x_i, e_j) = \max_{c, x' \in X} \underbrace{u(c) + \beta \sum_{k=1}^{n_e} P(e_k|e_j)V(t + 1, x', e_k)}_{\text{s.t. } c + x' = (1 + r)x_i + e_jw}$$

End

**Algorithm 3:** Life-cycle value function

Note that the expectation in the right-hand side of the Bellman operator has been substituted by a sum over future value functions by taking advantage of the Markov chain structure of the productivity shock.

The **Julia** code in Box 1 computes the value function in serial:

In the code, the function `Value` computes the maximum value attainable given state variables `age`, `ix`, `ie`. We loop over all possible combinations of the state variables to
for(age = T:-1:1)
  for(ix = 1:nx)
    for(ie = 1:ne)
      V[age, ix, ie] = Value(age, ix, ie);
  end
end
end

Box 1: Julia code to compute the value function.

get our value function. The function Value is described in Box 2. The function searches, among all the possible values of the grid for assets, for the optimal level of savings for the next period. For the last period of life, the function picks, trivially, zero savings (if we had a terminal condition such as a valuation for bequests, such a choice could be different). We could have more sophisticated search methods but, for our argument, a simple grid search is most transparent.

This life-cycle problem can be parallelized along the saving and productivity shock dimensions (x and e), but not along the age dimension. Given the age of the household, t, the computation of $V(t, x_i, e_j)$ is independent of the computation of $V(t, x_{i'}, e_{j'})$, for $i \neq i'$, $j \neq j'$. While one of the processors computes $V(t, x_i, e_j)$, another processor can compute $V(t, x_{i'}, e_{j'})$ without them having to communicate. For a given $t$, a parallel algorithm could split the grid points along the x and e dimensions, and assign different computations to each of the processors, so that total computation speed can be significantly reduced. In other words, the problem can be parallelized along the x and e dimensions in the same way that we parallelized the basic value function iteration in Subsection 3.4.1. This is not the case, however, for the age dimension. For the computation of any $V(t, x_i, e_j)$, the computer has to first have computed $V(t+1, x_i, e_j)$. The computation is, by its very nature, recursive along t.
function Value(age, ix, ie)
    VV = -10^3;
    for(ixp = 1:nx)
        expected = 0.0;
        if(age < T)
            for(iep = 1:ne)
                expected = expected + P[ie, iep]*V[age+1, ixp, iep];
            end
        end
        cons = (1 + r)*xgrid[ix] + egrid[ie]*w - xgrid[ixp];
        utility = (cons^(1-ssigma))/((1-ssigma) + bbeta*expected;
        if(cons <= 0)
            utility = -10^5;
        end
        if(utility >= VV)
            VV = utility;
        end
    end
    return(VV);
end

Box 2: Julia code to compute the value function, given state variables age, ix, ie.

The following pseudo-code computes the model in parallel:
begin

1. Set $t = T$.

2. Given $t$, one processor can compute $V(t, x_i, e_j)$, while another processor computes $V(t, x'_i, e'_j)$.

3. When the different processors are done computing $V(t, x_i, e_j)$, $\forall x_i \in X$ and $\forall e_j \in E$, set $t = t - 1$. Go to 1.

end

**Algorithm 4:** Parallelizing the value function

Figures 6 and 7 illustrate the performance of each of the processors of an 8-thread CPU when the process is executed in serial (Figure 6) and in parallel with the maximum number of threads (Figure 7). While in the first figure only one processor works (CPU2, represented by the line that jumps in the middle), in the second figure all 8 CPUs are occupied.

Figure 6: 1 Core used for computation
Before we show how to implement Algorithm 4 in practice, we must spend some time discussing how the choice of a particular computer matters for parallelization and how we implement it.

Amdahl’s law (name after the computer architect Gene Amdahl) states that the speed-up of a program using multiple processors in parallel is limited by the time needed for the sequential fraction of the program. That is, if 25 percent of the computation is inherently serial, even a machine with thousands of processors cannot lower computation time below 25 percent of its duration when running serially. Thus, the speed of each processor is still of paramount relevance.

Similarly (and as we will describe below in more detail for each programming language), we face the fixed cost of starting a thread or a process/worker when we want to distribute computations among workers, transfer shared data to them, and synchronize results. These tasks are time intensive and, often, they depend more on the hardware allowing the communication among processors than on the quality of the processors themselves. Indeed, much of the cost of state-of-the-art supercomputers is in top-of-the-line equipment that maximizes network bandwidth and minimizes latency, not in the
processors. Furthermore, the primary barrier facing extremely large machines is load imbalance among processors. In actual applications, it is difficult to use more than 10 percent of the theoretically available computing power.

These issues are somewhat less relevant for users of workstations and desktops with just a few processors, but cannot be completely ignored. For instance, an Intel Core™ i9-7980XE Extreme Edition has 18 cores and can run up to 36 threads. A performance workstation equipped with such a processor is priced at less than $5,000 and, thus, affordable to many researchers. This unprecedented level of sophistication for a personal computer requires an investment of time and effort in getting to know the limits a particular computer imposes on parallelization. A thorough discussion of such issues is well beyond the scope of this chapter (and would be, in any case, contingent on rapidly evolving hardware).

We would be remiss, however, if we did not mention other hardware possibilities. For example, many institutions provide affiliated researchers with large servers, some of them with dozens of cores, and even with mighty supercomputers. Similarly, several companies offer computing services on demand over the internet. The main advantage of these services is the replacement of a large initial capital cost (purchasing a workstation) for a variable cost (use-as-needed), the supply of constant updates in the hardware, and the avoidance of maintenance considerations (at times acute in a department of economics where IT support can be spotty).

A popular example of those services is Amazon Elastic Compute Cloud (EC2), which allows requesting nearly as much computational power as one would desire in economics. As of April 2018, you can rent on-demand 8 processors with 32 GiB of memory, running on Linux and gathered for general purposes for $0.3712 per hour and 96 processors with 384 GiB of memory for $4.608 per hour. These prices mean that one can use 96 processors.

33. [https://aws.amazon.com/ec2/](https://aws.amazon.com/ec2/), where in addition, the reader can find complete documentation of how to create and operate an instance online.
for an entire month, non-stop, for $3,318. Prices can be substantially cheaper if the user is willing to pay for spot pricing at lower-demand times.

The access to EC2 is straightforward. After setting up an account, one can go to Amazon EC2, click on “Launch a virtual machine” and follow the webpage links (for example, one can select Ubuntu Server 16.04 LTS, an instance that runs on the well-known Ubuntu distribution of Linux).

The only difficulty lies in the need to use a public key. The required steps are i) create a new key pair; ii) download the key; iii) store it in a secure place (usually ~/.ssh/); and iv) run an instance.

On an Ubuntu terminal this requires, first, transferring the folder from local to instance with scp:

```
$ scp -i "/path/PUBLICKEY.pem" -r "/pathfrom/FOLDER/"
    ubuntu@52.3.251.249:~
```

Second, to make sure the key is not publicly available:

```
$ chmod 400 "/path/PUBLICKEY.pem"
```

And, third, to connect to instance with ssh:

```
$ ssh -i "/path/PUBLICKEY.pem" ubuntu@52.3.251.249
```

Once the instance is running, the terminal where it operates works like any other Linux terminal and the user can execute its parallel codes.

---

34. [https://aws.amazon.com/ec2/pricing/on-demand/](https://aws.amazon.com/ec2/pricing/on-demand/). If one considers the expected life of a server before it becomes obsolete plus the burden of maintenance in terms of time and effort, the user cost of the server per month might be quite higher than $3,318.
3.7 Parallelization Schemes on CPUs

We are now ready to review several of the most popular parallelization schemes that take advantage of having access to multiple cores (physical or virtual), either in one or several CPUs. This is the most traditional form of parallelism. In Section 3.8, we will review parallelization schemes for GPUs and mixed approaches that can use both CPUs and GPUs. This second form of parallelism has gained much attention during the last decade and it deserves its own section.

Except for the case of MPI in Subsection 3.7.8 with its rich set of operations, the schemes in this section will be, in some form or another, simple variations of two different ideas: the for loop and Map and Reduce. In the for loop approach, we add a statement before the for loop that we want to parallelize. For example, in the code displayed in Box 2, if we want to parallelize the computation of the value function along the productivity (x) grid, we add a statement on the code specifying that the for (ix = 1:nx) loop should be computed in parallel. This approach is simple to implement, transparent in its working, and efficient for many computations. However, it faces scalability constraints.

In comparison, a Map function receives as inputs the function Value described in Box 1 and a data structure containing every possible combination of the state variables, and returns the value associated which each of them. This computation is done in parallel, so there is no need to loop over all state variables as in Box 2. Although our particular case does not require it, a Reduce function takes the values computed by the Map and reduces them in the desired way, such as taking the average or the sum of them.

In most programming languages, there is at least one package/extension to perform parallel computing, and most follow one of the two procedures described above. Some of them require the programmer to add a statement before the loop that should be computed in parallel, while some others require the use of a Map function. In the remaining sections of this paper, we show how to implement these schemes in some of the most popular
programming languages in economics. More concretely, we will deal with Julia, Matlab, R, Python, C++ with OpenMP, and Rcpp with OpenMP. We will close with explanations of how to apply MPI, which can interact with different programming languages.\footnote{As we indicated in the introduction, we do not try to be exhaustive in our coverage of programming languages, as such an exercise would be too cumbersome. Furthermore, the set of languages we select spans other alternatives with minimal changes. For example, once one understands how to implement parallelization in C++ with OpenMP, doing the same in Scala or Fortran is straightforward. For a comparison of programming languages in economics that motivates our choice of languages for this exercise, see \cite{aruoba2015}.}{35}

### 3.7.1 Julia - Parallel for

Julia is a modern, expressive, open-source programming language with excellent abstraction, functional, and metaprogramming capabilities. Designed from the start for high-performance numerical computing, Julia is particularly well-suited for economics.

A design principle of Julia was to provide tools for easy and efficient parallelization of numerical code. One of those tools is the `@parallel for`, most convenient for “small tasks,” that is, for situations where there are few computations on each parallel iteration. This might be because we have few control variables, or few grid points on control variables, or the computation requires relatively few steps.\footnote{See Julia Parallel Computing. These paragraphs follow the syntax Julia v0.6.2, the current release as of April 2018. Some details may change before the release of the first “stable” 1.0 version, announced for some time in late 2018. Our computations are done with Julia v0.6.0, which has minimal changes with respect to Julia v0.6.2.}{36}

To implement `@parallel for`, we start by setting up the number of workers according to the architecture of our machine with the instruction `addprocs()`:

```julia
addprocs(6) # Here we have 6; check the number of processors you have available
```

\footnote{As we indicated in the introduction, we do not try to be exhaustive in our coverage of programming languages, as such an exercise would be too cumbersome. Furthermore, the set of languages we select spans other alternatives with minimal changes. For example, once one understands how to implement parallelization in C++ with OpenMP, doing the same in Scala or Fortran is straightforward. For a comparison of programming languages in economics that motivates our choice of languages for this exercise, see \cite{aruoba2015}.}{35}

\footnote{See Julia Parallel Computing. These paragraphs follow the syntax Julia v0.6.2, the current release as of April 2018. Some details may change before the release of the first “stable” 1.0 version, announced for some time in late 2018. Our computations are done with Julia v0.6.0, which has minimal changes with respect to Julia v0.6.2.}{36}
In Julia, as in most languages described below, initializing the pool of workers can take some time. However, the programmer has to consider that this is a one-time fixed cost that does not depend on the size of the problem. If the parallel pool is initialized to perform a time-consuming task or to repeat the same task multiple times, the fixed cost vanishes as a proportion of total computation time. However, if the user only uses the parallel pool to perform a one-time small task, initializing the pool of workers might make parallel computing more time expensive than serial computing.

When variables are declared in Julia, they are not part of the shared memory by default. That is, variables are not generally observable/accessible by workers. To grant the workers access to the variables they need for their computations, we must explicitly declare those variables to be accessible/modifiable by workers. If we only want a variable to be observed by workers, but not modified on parallel iterations, we should declare it as global with `@everywhere`:

```julia
@everywhere T = 10;
#...
@everywhere gridx = zeros(nx);
```

However, there are some variables that we want the parallel workers to modify simultaneously in each iteration. In our example, we want the workers to modify the entries of the value function matrix \( V \) that are assigned to them. To accomplish this, we declare these variables as `SharedArray`:

```julia
V = SharedArray(Float64, (T, nx, ne),
    init = V -> V[Base.localindexes(V)] = myid());
```
Once the required variables have been declared to be observable and modifiable by the workers, we should explicitly state the for loop we want to be executed in parallel. For this, it suffices to add the directive @parallel before the for statement. For example, if we want to compute in parallel the values assigned to each point in the savings grid, we add @parallel before the for loop associated with the x grid:

```plaintext
@parallel for(ix = 1:1:nx)
  # ...
end
```

As explained above, our life-cycle problem can be parallelized across the savings and productivity dimensions, \( x \) and \( e \), but not across age, \( t \). This means that, for every \( t \), we should assign grid points in the \( E \times X \) space to each worker to compute the value assigned to those points, gather back the computations, stack them on the matrix \( V(t, \cdot, \cdot) \), iterate to \( t - 1 \), and repeat. Before iterating to \( t - 1 \), we need all the workers to finish their job so that we can collect all the information in \( V(t, \cdot, \cdot) \). To make sure that every worker finishes its job before iterating to \( t - 1 \), we must add the directive @sync before the @parallel for statement:

```plaintext
@sync @parallel for(ix = 1:1:nx)
  # ...
end
```

These three elements (i) declaring observable and modifiable variables, ii) a directive @parallel for loop, and iii) a directive @sync) are all one needs to run the code in parallel.
in Julia. The execution of the code in the REPL terminal or the batch model is the same as for any other Julia file.

There is, however, an important further point to discuss. In our example, we stated that the problem could be computed in parallel across two dimensions: $x$ and $e$. It turns out that, in Julia, it is not possible to embed a `@parallel` for loop inside another `@parallel` for loop. The programmer must choose whether to perform parallel computing on one grid but not on the other, or to convert the two nested loops into a single loop that iterates over all the points in both grids.

Let us explore the first alternative (parallelize along one dimension, but not the other) first. The code below presents the two possible choices of the loop to parallelize in two columns:

```
nx = 350;
ne = 9;
for(ie = 1:ne)
  @sync @parallel for(ix = 1:nx)
    # ...
  end
end
```

```
for(ix = 1:nx)
  @sync @parallel for(ie = 1:ne)
    # ...
  end
end
```

Although the choice of computing in parallel the $x$ or the $e$ grid might seem innocuous, the implied performance differences are large. Computing in parallel the loop corresponding to the $x$ grid is much faster. The reason is mainly the time spent in communication vs. actual computation.

Note there are many more grid points in the $x$ grid, 350, than in the $e$ grid, 9 (this is typical in life-cycle models; in other models, the larger grid can be another dimension).
In the left column above, for every point in the $e$ grid, the master node has to split the $x$ grid among the workers. That is, on every iteration on the $e$ grid, the master node has to communicate and exchange information with the workers. In the above example, with 9 points on the $e$ grid, the master node has to move information 9 times. On each of these 9 iterations, the master node divides a grid of 350 points among 8 workers. If the tasks are equally split, each worker will have around 43 computations on every iteration.

In the right column, when we choose to parallelize across the $e$ grid, the opposite happens. Since $nx=350$, the master node has to move information 350 times, one after every iteration. Moreover, the computing time is small, as the 9 grid points in the $e$ grid are divided among 8 workers; that is, at most, two computations per worker! In this case, the communication time is very large when compared to computation time, so parallel computing is not only slower than in the first case, but it can be even slower than any parallelization at all.

The second alternative collapses the two nested loops into a single loop that iterates over all the grid points. In this case, the parallelization is performed over all the grid points, which minimizes the communication time and maximizes the computations per worker:

```plaintext
@sync @parallel for(ind = 1:(ne*nx))
    ix = convert(Int, ceil(ind/ne));
    ie = convert(Int, floor(mod(ind-0.05, ne))+1);
    # ...
end
```

In the Github repository of this paper, the reader can find an example in which we compute the baseline model described with parallel loops across both $x$ and $e$ dimensions.
We repeat the computation of the model with a different number of cores for parallelization, up to 8 cores. The experiments are performed on a laptop computer with 4 physical cores and 8 virtual cores. In each case, we execute the code 20 times, to account for small differences in performance between runs.

Figure 8a illustrates the average total computing time, with the number of cores employed on the x-axis and the time in seconds on the y-axis. The performance gains are substantial up to the third core, going from around 16 seconds to less than 8, but they decrease after that. Since total time spent in communication is large, adding workers does not necessarily increase speed. This point is stressed in Figure 8b, which plots the number of cores against the performance gains in terms of the normalized performance of one processor. If the parallelization was perfect and no time was spent in communication, the performance gains should be linear. That is, doubling the number of processors should double the speed of computation, as represented by the 45° line (the inherently serial part of the code affected by Amdahl’s law is small). In reality, the performance gains are always significantly below the optimal gains and start decreasing after the third core. To make this argument clearer, we plot in both figures the mean of the runs and the band between the min and max performance. Of course, in larger problems (for instance, with more points in the state variable grids), performance might continue increasing with additional processors. Similarly, better machines (for example, with more physical processors) might deliver improved performance for a longer range of processors used.

When implementing parallel computing in Julia, the programmer must take into account that the total computation speed is particularly sensitive to the use of global (@everywhere) variables and large SharedArray objects. Many elements in shared memory can significantly hamper the potential time gains that parallel computing brings. In applications where shared memory is sizable, it might even be preferable to perform serial computing.

---

37 More concretely, we use a Lenovo Y50-70 with Intel Core™ i7-4710HQ CPU @2.50GHzx8, 15.6 GiB, and NVIDIA GeForce GTX 860M/PCIe/SSE2 GPU. The operating system is Ubuntu 14.04, 64-bit.
Figure 8: Results in Julia with different number of processors. Number of experiments: 20.

3.7.2 Julia - MapReduce

Recall that the @parallel for is designed for applications, such as our life-cycle model, when the problems assigned to each worker are small. In cases where there are many more control variables or where we work with fine-mesh grids, or in models with many discrete choices, @parallel for might slow down the computation. In these situations, a MapReduce-type of function built into Julia is a superior approach.

To do so, we must look at our life-cycle problem from a different perspective. To solve the problem serially, we compute the maximum attainable value –given the restrictions– for every point –t, x, and e– in the grids (see Algorithm 4). A MapReduce function, instead, takes as inputs a function and vector of values where we want to evaluate that function (recall our discussion in Section 3.3). In our case, the inputs would be the function Value that computes the optimal choice given state variables, and the vector of values would be a data structure containing all possible combinations of the points in the grids for the state variables (or, perhaps, some relevant subset of those combinations).
We show now how to implement this idea in practice. As before, the first step is to initialize the number of parallel workers:

```
addprocs(6)
```

Next, we define a data structure `modelState` composed of all parameters and variables needed for the computation in `Value`, and declare this structure as global:

```
@everywhere type modelState
  ix::Int64
  age::Int64
  # ...
end
```

Note that the idea of creating a rich data structure to store all the state variables can be applied to a wide class of dynamic models that goes well beyond our simple life-cycle economy. More generally, using rich data structures allows for cleaner and less bug-prone code, since we can define operators on the whole structure or overload already existing ones, instead of operating on each element of the structure one at a time.\(^{38}\)

Given that the function `Value` will now depend on the data structure: `modelState`, instead of on each of the parameters, the function must be modified accordingly to take the structure as an input:

---

38. Overloading an operator is to define a different effect of the operation when the inputs are different, i.e, the operator “+” works differently when applied to two integers than when we apply it to two matrices of complex numbers. Also, we can be flexible in the definition of structures. For example, we can incorporate on them just the states of the model, but not the parameters as well, as we do in the main text.
Since the workers in the parallel pool must be able to observe the function Value, this must also be declared as global with the directive @everywhere.

The last step is to use the Map-type of function to compute Value in parallel. In Julia, this is done with the pmap function (pmap generalizes map to the parallel case), which receives as inputs the function Value and a vector with all possible combinations of state variables as stored in our data structure modelState:
Figures 9a and 9b illustrate the total computing time and the time gains, respectively, with a different number of processors when using the `pmap` function. Absolute computing times using this procedure are very large when compared to the results of the last section, and the performance gains from adding more workers to the parallel pool are very poor. One possible explanation is that the `pmap` function is designed to be applied to tasks where each worker is assigned a large task. In our example, this is not the case. The computation speed, in this case, could also be reduced by applying Julia-specific tricks, such as wrapping the code inside a function. However, since the purpose of this paper is to make the comparison as transparent as possible, we avoid applying language-specific procedures different from the parallelization itself.

![Graphs showing computing time and performance gains with different number of processors.](image)

Figure 9: Results with `pmap` function in Julia with different number of processors. Number of experiments: 20.
3.7.3 Matlab

Matlab performs parallel computing through the parallel toolbox. While this tool is probably the simplest of all the alternatives covered in this paper to implement, it also suffers from limited performance and it is expensive to purchase.

As opposed to Julia, in which variables are neither observable nor modifiable by default by all the workers in the parallel pool, in Matlab they are. Every time you declare a variable, it can be accessed by any worker in the parallel pool. In this way, the programmer only has to take care of initializing the parallel pool of workers and explicitly stating which for loop she wants to be parallelized.

To initialize the parallel pool of workers, we use the function `parpool()`:

```
parpool(6)
```

In Matlab, as in Julia, initializing the workers in the parallel pool is also time consuming. It can take some seconds, even a considerable fraction of a minute. Again, if the parallel pool is initialized to perform a time-consuming task or to repeat the same task multiple times, the fixed cost of initializing the parallel pool vanishes as a proportion of total computation time, and parallel computing might increase performance. If the code only performs a small task, the initialization cost of parallel computing might offset its benefits.

To state which loop is to be computed in parallel, instead of declaring the for loop, the programmer must declare it as `parfor`:

```
for age = T:1:1
    parfor ie = 1:1:ne
```

39. We employ Matlab 2017b, with its corresponding parallel toolbox.
Beyond these two steps (i.e., initializing the parallel pool and employing `parfor`), the developer does not need to exert further effort and `Matlab` takes care of all the rest.

![Graph showing computing time and performance gains](image)

(a) Computing time (s)  
(b) Performance gains.

Figure 10: Results in `Matlab` with different number of processors. Number of experiments: 20.

Total computing time in `Matlab` is considerably higher than in `Julia` with the `@parallel` `for` procedure, as illustrated by Figure 10a. While `Julia` takes around 16 seconds to compute the model with one core, `Matlab` takes over 50 seconds. Similar to `Julia`, the performance gains of using parallel computing in `Matlab` are not very large, as illustrated in Figure 10b. The speed-ups are well below the perfect linear gains represented by the 45° line. On average, using 4 workers in the parallel pool decreases computing time by less than 2.5 times. Computing in `Matlab` is very simple to implement, but does not bring substantial benefits when compared to other languages. Indeed, after the computer reaches the limit of physical cores, computing time deteriorates with additional virtual cores.
Parallelization in R is important because the language’s speed in serial computations is quite often disappointing in comparison with other alternatives (see Aruoba and Fernández-Villaverde, 2015), and yet a researcher might want to code on it to take advantage of its superior statistical and graphics capabilities.

R, as an open source language, has multiple parallel computing packages created by different developers. Here we will cover only the widely used parallel package, although there are other alternatives, such as the foreach and doParallel (we use R 3.4.1, which includes parallel by default). In later sections, we will describe how to perform parallel computing with C++ and how to embed C++ code in R through the Rcpp package, allowing the user to expand the use of parallel computing in R.

The main functions in the parallel package for parallel computing are the `parLapply` and the `parSapply` function, which are Map-type functions that perform the same task as the `lapply` and `sapply` functions, respectively. `lapply` applies a function to a list of variables. `sapply` performs the same task in a more user-friendly manner. R has strong functional programming capabilities to manipulate lists that are often overlooked (for a crisp explanation of them, see Wickham, 2014).

To access the parallel package in R, the user must first install it and open the library:

```r
install.packages("parallel")
library("parallel")
```

This package has the `parLapply` function. This is a parallel Map kind of function. As we did with Julia, when the parallel procedure employs a MapReduce scheme, the user must define the function `Value`:

```r
```
In R there is no need to define a data structure, as the vector of parameter and state variables can be set as a list. Also, as opposed to Julia, in R all variables are observable and modifiable by the workers of the parallel pool. Therefore, it suffices to generate a list that contains all parameters and every possible combination of state variables. In our example, we perform this using the `lapply` function, which is a `Map` that returns a list with parameters and all possible combinations of state variables:

```R
states = lapply(1:(nx*ne), function(x) list(age=age, ix=x, ..., r=r))
```

Here we could have used the `parLapply` function to create the list of inputs to solve our main problem. Since we want to emphasize the use of parallel computing to compute the value function, we omit this step.

The function `makeCluster` initializes the desired number of workers:
Finally, the function `parLapply(cl, states, Value)` computes the function `Value` at every element of the list `states`, which contains all combinations of state variables. The package uses the cluster `cl` initialized above:

```r
for(age in T:1){
  states = lapply(1:(nx*ne), function(x) list(age=age,ix=x, ...))
  s = parLapply(cl, states, Value)
}
```

The output of the `parLapply` function is a list with the elements of the evaluation of `Value` at each of the elements in `states`. To store them in a three-dimensional matrix `V`, the user must finally loop over all the elements of `s` and store them at `V`.

The results of our computation are reported in Figures 11a and 11b, where we see that R is extremely slow at computing the model. It takes over 1600 seconds with one core, and over 800 seconds with 8 cores (that is why we only run 5 experiments; with long run times, small differences in performance between one run and the next are negligible). Moreover, among all the computing languages presented in this paper, the performance gains of `parLapply` in R are the smallest. Using 4 workers for the parallel computation increases performance by just over 1.5, and using all available 8 workers reduces total computing time almost by half. These results imply that, among the languages included in this paper, R is the least desirable one to use for the computation of this class of models (for a similar result, see Aruoba and Fernández-Villaverde, 2015).
3.7.5 Python

Python is an open-source, interpreted, general purpose language that has become popular among many scientists due to its elegance, readability, and flexibility. It is also a very easy language to learn for those without previous experience in coding in general domain languages.

As was the case with R, there are multiple modules to perform parallel computing in Python. In this section, we will review the `Parallel` function from the `joblib` module, which is a `Map`-like function.\(^\text{40}\) Thus, the reader will notice the similarities to previous sections that used `Map`-type functions.

First, we import the `joblib` and `multiprocessing` modules:

```python
from joblib import Parallel, delayed
import multiprocessing
```

---

\(^\text{40}\) We use Python 2.7.6. `joblib` provides lightweight pipelining in Python and, thus, is a good choice for our exercise. For a more general description of parallelization in Python, see Gorelick and Ozsvárd (2014).
Next, we define a data structure that contains the parameters and state variables, which is going to be one of the inputs of the Map-type function:

```python
class modelState(object):
    def __init__(self, age, ix, ...):
        self.age = age
        self.ix = ix
        # ...
```

In Python, the variables declared are globally observable and modifiable by the workers in the parallel pool, so there is no need to specify their scope when defining them.

Next, we define the function `Value` that computes the maximum given some values of the states:

```python
def Value(states):
    nx = states.nx
    age = states.age
    # ...
    VV = math.pow(-10, 3)
    for ixp in range(0, nx):
        # ...
    return [VV];
```
The function `Parallel`, from the `joblib` package, receives as inputs the number of workers of the parallel pool, the function `Value`, and a list of elements of the type `modelState`, one for each combination of values of the state variables. This function returns the function `Value` evaluated at each of the state variables. This parallelization is done at every age \( t \in \{1, \ldots, T\} \):

```python
def value_func(modelState, age, w, r):
    # Function definition
```

Finally, we construct the value function by storing the values of `results` in a matrix `V`:

```python
for age in reversed(range(0,T)):
    results = Parallel(n_jobs=cores)(delayed(value_func)(modelState(ind,ne,...)) for ind in range(0,nx*ne))
    for ind in range(0,nx*ne):
        ix = int(math.floor(ind/ne));
        ie = int(math.floor(ind%ne));
        V[age, ix, ie] = results[ind][0];
```

The results of the computations with Python are displayed in Figures 12a and 12b. As in R, total computing time with one core is extremely slow. It takes almost 2,000 seconds, on average, to compute the life-cycle model. However, parallel gains turn out to be very good, as increasing the number of cores up to 4 decreases computing time to almost a third. Increasing the number of cores beyond 4 does not generate any time improvements.
When compared to Julia, Matlab, and R, the performance gains of using 4 computing cores with the `Parallel` function in Python are the largest. However, given its absolute performance, it is still not a desirable language to compute models such as the one presented in this paper.

![Graphs showing computing time and relative performance](image)

(a) Computing time (s)  
(b) Performance gains.

Figure 12: Results in Python with different number of processors. Number of experiments: 20.

Python can be accelerated with tools such as Numba. Dealing with them here, however, requires mixing both parallelization and acceleration in Python, complicating our expository goals. See, for more information, Gorelick and Oszvald (2014).

### 3.7.6 C++ and OpenMP

So far, we have dealt with scripting languages such as Julia, Matlab, R, and Python that allow for fast parallelization using either built-in directives or additional packages. This approach had the advantage of simplicity: minor changes to the serial code deliver programs that can be run in parallel. Thus, these scripting languages are great ways to start working on parallel or to solve mid-size problems. On the other hand, our results also show the limitations of the approach: scaling up the problem in terms of processors is
difficult. And even with parallel processing, a researcher might still need extra speed from each processor.

A more powerful, but also more cumbersome, approach is to switch to a compiled language and take advantage of the tools available for parallel programming in that language. C++ is a natural choice of a compiled language because of its speed, robustness, flexibility, industry-strength capabilities, the existence of outstanding open-source compilers such as GCC (indeed, for all the experiments with C++ in this paper, we will use GCC 4.8.5), excellent teaching material, and broad user base.41

One of the easiest tools to implement parallel computing in C++ is OpenMP (Open Multi-Processing). OpenMP is an open-source application programming interface (API) specification for parallel programming in C, C++, and Fortran in multiprocessor/core, shared-memory machines (wrappers for OpenMP exist for other languages, including Python and Mex files in Matlab). OpenMP was released in 1997 and its current version, as of April 2018, is 4.5. OpenMP is managed by a review board composed of major IT firms and it is implemented by the common compilers such as GCC and supported by standard integrated development environments (IDEs) such as Eclipse.

OpenMP can be learned quickly as it is easy to code, and yet rather powerful. For example, most implementations of OpenMP (although not the standard!) allow for nested parallelization and dynamic thread changes and advanced developers can take advantage of sophisticated heap and stack allocations. At the same time, most problems in economics only require a reduced set of instructions from the API. However, OpenMP is limited to

---

41 Future standards of C++, such as C++20, plan to include executors as a building block for asynchronous, concurrent, and parallel work. Parallel programming in C++, then, might look closer to the code of the scripting languages. In comparison, explicitly parallel languages such as Unified Parallel C, µC++, Charm++, Chapel, and X10 have never left the stage of small experimental languages, and we will not cover them here. High Performance Fortran and Coarray Fortran have suffered from the demise of Fortran outside a small niche of specialized users, dealing more often than not with legacy code from previous decades.
shared-memory machines, such as personal computers or certain computer clusters, and the coder needs to worry about contention and cache coherence.⁴²

OpenMP follows a multithreading “fork-join” paradigm: it executes the code serially until a directive for parallel computing is found in the code. At this point, the part of the code that can be computed in parallel is executed using multiple threads, and communication happens between the master thread and the rest of the workers in the parallel pool. After the parallel region is executed, the rest of the code is computed serially. Figure 13 illustrates the execution of an OpenMP code. This suggests that a good rule of thumb for OpenMP is to employ one thread per core.

![OpenMP computing](image)

Figure 13: OpenMP computing

The “fork-join” scheme, however, means that it is the job of the user to remove possible dependencies across threads (for instance, variable values in thread \( n \) required in thread \( n - j \)) and synchronize data. To avoid race conditions (some variables being computed too early or too late), the developer can impose fence conditions and/or make some data private to the thread, but those choices may cause performance to deteriorate. Synchronization is expensive and loops suffer from time overheads.

We implement OpenMP as follows. First, before compilation, we must set the environmental variable `OMP_NUM_THREADS` equal to the number of cores used in the parallel

computation of the code. In a Linux/Unix computer, this can be done with a Bash shell by opening the terminal window and typing:

```bash
export OMP_NUM_THREADS=32
```

Second, at compilation time, the user must add the specific flag, depending on the compiler. These flags tell the compiler to generate explicitly threaded code when it encounters OpenMP directives. With the GCC compiler, the compilation flag is `-fopenmp`. With the PGI compiler, the required flag is `-ta=multicore`. Our particular example compiles as:

```bash
$ g++ Cpp_main_OpenMP.cpp -fopenmp -o -O3 Cpp_main
```

In the absence of this flag, the code will be compiled as regular C++ code, always a good first step to debug and optimize the code before running it in parallel (perhaps with a less challenging computing task, such as smaller grids or more lenient convergence tolerances). Note that we also use the `-O3` flag to generate optimized code. This flag can be omitted in a first exercise while we debug and test the code, but, in “production” code, it accelerates the performance of the code considerably.

Inside the code, we include the header `omp.h` to load the relevant library:

```c
#include <omp.h>
```

Then, we add the directive `#pragma omp parallel for` right before each `for` loop we want to parallelize. This directive has multiple options, and has to be accompanied by the variables that are private to each worker and those that are shared. This is done using the `shared(...)` and `private(...)` directives:
The programmer can specify additional options for the parallelization, such as explicitly setting barriers that synchronize every thread, setting access to shared-memory objects by workers, dividing tasks between the master thread and the workers, setting private threads for specific workers, and so on.

Given the simplicity of implementing parallel computing in OpenMP, it is worth using it even for small tasks. First, as opposed to Julia and Matlab, the time taken to initialize the parallel pool is negligible, and the code is executed almost immediately. Second, the coding time needed to compute the code in parallel is relatively low, as only a couple of lines of code are needed. Third, as mentioned above, you can always compile your code without the -fopenmp flag to run the code serially to compare the results obtained with and without parallelization and debug them.

Total computing times in OpenMP with a different number of cores are presented in Figure 14a. As can be observed, the total time with only one core is around 4 seconds, considerably better than Julia in Section 3.7.1 and orders of magnitude better than Matlab, R, and Python. This is not a surprise given the advantages of compiled languages over scripting ones. See, for similar results, Aruoba and Fernández-Villaverde (2015). It is worth recalling that the C++ code for these computations was compiled with the -O3 optimization flag. In its absence, C++’s performance deteriorates to nearly 20 seconds (see the Appendix for details about the consequences of different optimization options).43

43. Compilers for C++ such as GCC and Intel have options for automatic parallelization (see https://gcc.gnu.org/wiki/AutoParInGCC and https://software.intel.com/en-us/node/522532) that try to
Figure 14: Results in C++ with OpenMP with different number of processors. Number of experiments: 20.

Figure 14b illustrates the performance gains with a different number of cores. The performance grows almost linearly up to the third core; using 3 cores improves computation speed by a factor of more than 2.5. Although there seems to be no performance gain in adding the fourth core on average, the best case of the experiments made shows that it is possible to achieve an almost-linear gain. In the best experiment, using 4 cores improves computation speed by a factor of more than 3.5. Also, in contrast to the scripting languages, there are gains from adding virtual cores beyond the number of physical cores. When computing the problem with 8 cores, the average computing time falls by more than a factor of 3.

---

44 The relatively high variance of the experiments is due to the short run time of the solution of the model in C++. A fix to avoid this variance would be to increase the number of grid points. But then, the computation of R and Python would take too long. Since the primary goal of this paper is to illustrate how each parallelization method works—not to measure their time performance with utmost accuracy—we keep the experiments the way they are.
3.7.7 Rcpp and OpenMP

Having described the relatively poor performance of parallel computing in R and the great gains in C++ at a relatively low cost, it is worth asking whether it is possible to perform the computations in R by creating a C++ function that uses OpenMP. Fortunately, the Rcpp package in R allows the user to create functions in C++. These functions can use OpenMP to perform the computations in parallel.45

All the documentation of the Rcpp package is available on the Rcpp for Seamless R and C++ Integration website. To use Rcpp, the user must first install the package and import the library from within R:

```r
install.packages("Rcpp")
library("Rcpp")
```

The user must write the C++ function with the appropriate C++ directives, as described in Section 3.7.6:

```cpp
#include <omp.h>

// ...

#pragma omp parallel for shared(...) private(...) 
for(int ix = 0; ix<nx; ix++){
    //...
}
```

45. Although other packages allow parallel computing with Rcpp, such as the RcppParallel, their implementation costs are higher. As described in the last section, with OpenMP, parallel computing can be achieved by adding only a couple of lines of code. Also, a similar mixed-programming approach can be implemented with Mex files in Matlab. Our experience is that Rcpp is much easier and efficient than Mex files. See, for instance, the code and results in Aruoba and Fernández-Villaverde (2015).
Before importing the function with Rcpp, we must explicitly state which functions in the C++ code are to be imported to R, by adding the following code on the line before the function C++ code:

```cpp
// [[Rcpp::export]]
```

There are three options to set the number of workers on the parallel pool. The first one is to set the environmental variable `OMP_NUM_THREADS` to the desired number of workers directly from the terminal. In a Linux/Unix machine, this can be done in a `.bash_profile` file or from the terminal:

```bash
export OMP_NUM_THREADS=32
```

The second option is to include the following code right before the first parallel region of the C++ code:

```cpp
omp_set_num_threads(n)
```

The third option is to include the `num_threads(n)` option on the `#pragma` statement, where `n` is the number of desired workers in the parallel pool:

```cpp
#pragma omp parallel for shared(...) private(...) num_threads(n)
for(int ix=0; ix<nx; ix++){
   // ...
}
```
Given that the C++ is compiled from within R, we must be sure that the OpenMP flag is added at compilation. For this, we should add the `−fopenmp` flag using the `Sys.setenv()` function from within R:

```
Sys.setenv("PKG_CXXFLAGS"=" -fopenmp")
```

Finally, the C++ code can be compiled and imported from R, so the function that computes the value function can be called from R. The function `sourceCpp` is used to compile the C++ code and create such a function:

```
sourceCpp("my_file.cpp")
```

The sample codes in R and C++ can be found on the Github repository of this paper. The user can compare how the C++ code has to be modified in order to import it with Rcpp, and how it can be executed from R.

The results, presented in Figures 15a and 15b, are outstanding and very similar to those from C++ with OpenMP in Section 3.7.6. This is not surprising, as Rcpp is nothing different than compiling and executing the C++ from within R. Again, the performance improvements are almost linear up to the third core. The best-case scenario when adding the fourth core is almost linear. Adding workers beyond the fourth core generates small improvements in average performance. One difference with the C++ code is that the Rcpp code runs much faster. This is because the C++ code in Section 3.7.6 was compiled without any optimization flag. In contrast, Rcpp compiles the code with the `-O3` optimization flag by default, which significantly improves absolute computation speeds. Section 3.10 presents the results when the C++ code is compiled with different optimization flags.
Despite its simplicity, OpenMP requires, as we have repeated several times, the use of shared memory. Such a requirement limits the range of machines that we can use for parallelization. Most large computers use distributed memory as shared memory becomes unfeasible after some size limit. Also, the use of shared memory by OpenMP might complicate the portability of the code among machines with different memory structures. Beyond the requirement of the reproducibility of research, portability matters in parallel computing more than in serial applications because a researcher may want to develop, debug, and test the code on a computer different from the one on which the final code will be run. For example, the former tasks can be done on a regular workstation (cheap and convenient), while the latter is done on a large server (expensive and more cumbersome).

The natural alternative is to use MPI in C++ (or other appropriate language; MPI has bindings for other programming languages, such as Fortran, Python, and R). MPI stands for message passing interface, and is a library and a set of function calls that allow communication between parallel processes executed on distributed systems, such as

![Figure 15: Results in Rcpp with OpenMP with different number of processors. Number of experiments: 20.](image)
computer clusters and multicore computers. MPI ensures that the code will be portable from one platform to another with minimal changes and it allows for outstanding optimization of performance.

More concretely, we will use a popular open source implementation of MPI, called Open MPI, prepared by an extensive group of contributors and supported by many members and partners (including AMD, Amazon, Intel, and NVIDIA). This implementation conforms with the MPI-3.1 standards and is fully documented on Open MPI’s website.\footnote{A basic tutorial of MPI can be found at Linux Magazine. More advanced tutorials are at LLNL tutorials and An MPI Tutorial. Gropp et al. (2014) is an excellent textbook on MPI.}

In MPI, the whole code is executed simultaneously by every thread, and communication directions are executed, which means that the programmer must explicitly give instructions for the communication that should take place between workers in the parallel pool. Figure 16 illustrates the behavior of the threads in an MPI program; all workers perform tasks simultaneously, and communication happens between them at different moments. For instance, thread 1 sends message $m_\alpha$ to thread 2, while it receives message $m_\gamma$ from thread 3. You can compare Figure 16 with the “fork-join” paradigm of OpenMP shown in Figure 13, where some parts of the code are executed in parallel and some others serially.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{mpi_diagram.png}
\caption{MPI computing}
\end{figure}
In particular, MPI can broadcast (transfers an array from the master to every thread), scatter (transfers different parts of an array from the master node to every thread; note the difference with broadcasting), and gather information (transfers an array from each of the threads to a single array in the master node). Figure 17 illustrates the ideas of broadcasting, scattering, and gathering of information among workers.

This richness in capabilities means that, in comparison with the simplicity of OpenMP, MPI is much more complex. For example, MPI 3.1 has more than 440 functions (subroutines). This richness is required to provide the multiple directives for communication between threads to ensure that the library can handle even the most challenging numerical problems. While a description of all the MPI functions and their inputs would require a whole book, some of the main ones are:

- MPI_Init: initializes the threads.
• MPI_Comm_size: returns the total number of MPI processes in the specified communicator.

• MPI_Comm_rank: returns processor rank within a communicator.

• MPI_Bcast: broadcasts an array.

• MPI_Scatter: scatters an array.

• MPI_Gather: gathers an array.

• MPI_Barrier: stops the execution until every thread reaches that point.

• MPI_Send: sends messages between threads.

• MPI_Recv: receives messages from another thread.

• MPI_Finalize: finalizes the threads.

Other functions, such as MPI_Barrier, are used to synchronize the work of each thread in the parallel pool, as illustrated in Figure 18.

The MPI programmer must explicitly divide the state space between the workers in the parallel pool and assign specific tasks to each worker. This strategy is, therefore, very different from the parallel computing routines in previous subsections, in which it sufficed to state the loop that had to be parallelized or to use a Map-type function.

The first step to start the parallel pool is to initialize the threads with the MPI_Init() function:

\[
\text{MPI_Init(NULL, NULL)}
\]

Then, we store the total number of threads and thread id to assign jobs:
Figure 18: MPI synchronization with barrier function

```c
int tid, nthreads;
MPI_Comm_rank(MPI_COMM_WORLD, &tid)
MPI_Comm_size(MPI_COMM_WORLD, &nthreads)
```

Next, we define the portion of state space that each thread must execute, according to the thread id, `tid`:

```c
int loop_min = (int)(tid * ceil((float) nx*ne/nthreads))
int loop_max = (int)((tid+1) * ceil((float) nx*ne/nthreads))
```

The `MPI_Bcast` function instructs the master thread to send copies of the same information to every worker in the parallel pool. In our model, at every iteration `t`, we need
the master thread to send a copy of the value function at $t + 1$. For this reason, we must employ the MPI_Bcast function at the beginning of the loop:

```c
MPI_Bcast(Value, T*ne*nx, MPI_FLOAT, 0, MPI_COMM_WORLD);
```

Given that every thread will execute the same task, each one for a different portion of the state space, we must make this explicit on the for:

```c
for(int ind = loop_min; ind < loop_max; ind++){
    ix = floor(ind/ne);
    ie = ind % ne;
    // ...
}
```

The MPI_Gatherv function collects information from each worker in the pool and stacks it in a matrix. At the end of the for loop, the master thread needs to collect the respective computations of the value function made by each worker, and stack them in a matrix that stores the value function. This value function is then broadcast in the next iteration to workers, and so on:

```c
MPI_Gatherv(Valp, ..., MPI_COMM_WORLD);
```

The last step is finalizing the multithreading with the function MPI_Finalize.

```c
MPI_Finalize();
```
The complete code can be found on the Github repository of this paper and deserves some attention to fully understand our paragraphs above.

When applied to our life-cycle model, the performance of MPI with C++ is excellent. MPI takes advantage of all 8 processors and computes the problem nearly four times faster than in serial mode (see Figures 19a and 19b). At the same time, the performance is not outstandingly better than OpenMP’s. Up to 4 cores, the performance is almost equivalent, being linear on average up to the third core, and the variance of performance gains growing with the fourth core. The only significant difference is that with MPI, the variance of performance across experiments is smaller and the average larger when using more than four threads.

![Figure 19](image)

(a) Computing time (s)  
(b) Performance gains.

Figure 19: Results in C++ with MPI with different number of processors. Number of experiments: 20.

But our example was not ideal for MPI: we run the code in a machine with shared memory and only 8 cores. If we wanted to run the code with dozens of cores, MPI would shine and show all its potential, while OpenMP would reach its limits.

Our results highlight the trade-off between OpenMP and MPI: OpenMP is much simpler to code and its performance is nearly as good as MPI, but it requires shared memory. MPI can
handle even the most difficult problems (state-of-the-art numerical problems in natural sciences and engineering are computed with MPI), but it is much harder to employ.

Our educated guess is that, for most of the computations performed in economics (and the machines many researchers will have access to), economists will find OpenMP more attractive than MPI, but with a sufficient number of exceptions as to justify some basic familiarity with the latter.

3.8 Parallelization Schemes on GPUs

Over the last ten years, parallel computation on GPUs has gained much popularity among researchers and in industry. Although initially developed for graphics rendering (thus, their name), GPUs are particularly convenient for operations on matrices, data analysis, and machine learning. The reason is that tasks involved in these algorithms require operations with array manipulation similar to the ones for which GPUs are optimized. In these problems, the serial part of the code is run in the CPU, and the GPU is used only when the user explicitly states the part of the code to be computed in parallel. Furthermore, GPUs are comparatively cheap and come pre-installed on nearly all computers for sale nowadays. See Aldrich et al. (2011) and Aldrich (2014) for detailed explanations of how GPUs work and for the application of their computational power to economics.

The advantage of GPU computing is that, unlike CPUs, which have at most a couple of dozen processing cores, GPUs have thousands of computing cores. Table 1 presents the number of computing cores and memory of some of the available desktop NVIDIA GPUs (NVIDIA is the leading vendor of GPUs). And just as you can stack several CPUs in one machine, it is also possible to install several GPUs in the same computer, which increases performance even more (although it also complicates programming). Thus, through GPU computing, the user can achieve much larger speed gains than with CPU parallel computing. Depending on the type of computations performed and on the
<table>
<thead>
<tr>
<th>GPU</th>
<th>CUDA Cores</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeForce GTX TITAN Z</td>
<td>5760</td>
<td>12 GB</td>
</tr>
<tr>
<td>NVIDIA TITAN Xp</td>
<td>3840</td>
<td>12 GB</td>
</tr>
<tr>
<td>GeForce GTX 1080 Ti</td>
<td>3584</td>
<td>11 GB</td>
</tr>
<tr>
<td>GeForce GTX 690</td>
<td>3072</td>
<td>4 GB</td>
</tr>
<tr>
<td>GeForce GTX 780 Ti</td>
<td>2880</td>
<td>3 GB</td>
</tr>
<tr>
<td>GeForce GTX 1080</td>
<td>2560</td>
<td>8 GB</td>
</tr>
<tr>
<td>GeForce GTX 780</td>
<td>2304</td>
<td>3 GB</td>
</tr>
<tr>
<td>GeForce GTX 980</td>
<td>2048</td>
<td>4 GB</td>
</tr>
<tr>
<td>GeForce GTX 1070</td>
<td>1920</td>
<td>8 GB</td>
</tr>
<tr>
<td>GeForce GTX 970</td>
<td>1664</td>
<td>4 GB</td>
</tr>
<tr>
<td>GeForce GTX 770</td>
<td>1536</td>
<td>2 GB</td>
</tr>
</tbody>
</table>

Table 1: Number of CUDA cores and memory in NVIDIA GPUs.

memory handling and communication management between the CPU and the GPU, the performance gains range from 10 times as fast as the serial code, up to hundreds of times faster. Examples of applications in economics well-suited for GPUs include dense linear algebra, value function iteration, Monte Carlos, and sequential Monte Carlos.

But the thousands of cores of a GPU come at a cost: GPUs devote more transistors to arithmetic operations rather than flow control (i.e., for branch prediction, as when you execute an “if” conditional) and data caching. Thus, not all applications are well-suited for computation on a GPUs (or the corresponding algorithms may require some considerable rewriting). For instance, applications involving searching or sorting of vectors typically require too much branching for effective GPU computing. Also, memory management can be difficult.

Figure 20 illustrates the architecture of an NVIDIA GPU. The CPU is referred to as the “host,” and the GPU is referred to as the “device.” The device is composed of grids,
where each grid is composed of two-dimensional blocks. The dimensions of the grids and blocks are defined by the user, with the only restriction being that each block can handle up to 512 threads. This limitation means that the programmer must carefully divide the tasks such that each of these blocks does not have more than 512 jobs. The user must also take care in regard to communication and memory handling. More concretely, at every parallel region of the code, the user must explicitly transfer variables from the CPU to the GPU by appropriately managing the memory available in the GPU.

In this section, we will present two approaches to parallelization in GPUs: CUDA, in Subsection 3.8.1, and OpenACC, in Subsection 3.8.2. CUDA is the traditional environment for coding in GPUs, with plenty of examples and a wide user base. The drawback is that CUDA requires the codes to include many low-level instructions to communicate between
the CPU and GPU and for memory management.\textsuperscript{48} \texttt{OpenACC} is a more recent, but rather promising, avenue to simplify computation in GPUs. \texttt{OpenACC} requires much less effort by the coder and can work across different platforms and mixed processing environments, but at the potential cost of slower code. We will skip a description of \texttt{OpenCL}, an open standard for computing in heterogeneous platforms. While \texttt{OpenCL} has the advantage of not being tied to NVIDIA GPUs, its user base is thinner and less dynamic, and its performance is generally inferior to \texttt{CUDA}'s. Similarly, one can program in the GPU with the \texttt{Matlab parallel toolbox} and in \texttt{R} with the \texttt{gpuR} package. However, if a researcher needs enough speed as to require the use of a GPU, perhaps moving first to a language more efficient than \texttt{Matlab} or \texttt{R} before diving into GPUs seems a wiser course of action.\textsuperscript{49}

3.8.1 CUDA

\texttt{CUDA}, which stands for Computer Unified Device Architecture, is a parallel computing model with extensions for \texttt{C}, \texttt{C++}, \texttt{Fortran}, and \texttt{Python}, among others, that allows the implementation of parallel computing on the GPU of a computer. The \texttt{CUDA} model was designed by NVIDIA. It was first released in 2007 and the current stable version, as of April 2018, is 9.2, with an extensive range of applications and a deep user base.\textsuperscript{50}

The first step for parallel computing in \texttt{CUDA} is to explicitly define the functions that are going to be executed from within the GPU, as opposed to those that are executed from

\textsuperscript{48} This disadvantage can partially be addressed by the use of libraries such as \texttt{ArrayFire} (https://arrayfire.com/), which allow high-level instructions and function calls. In our assessment, however, going with \texttt{OpenACC} seems at this moment a better idea for the long run.

\textsuperscript{49} The repository https://github.com/JuliaGPU lists several alternatives for GPU computing in Julia, but the packages are still far from easy to employ by less experienced users.

the CPU. The former should be preceded by the directive __device__, while the latter should be preceded by __global__:

```c
// Functions to be executed only from GPU
__device__ float utility(float consumption, float ssigma){
    float utility = pow(consumption, 1-ssigma) / (1-ssigma);
    // ...
    return(utility);
}

// Functions to be executed from CPU and GPU
__global__ float value(parameters params, float* V, ...){
    // ...
}
```

Next, the user must allocate memory in the device (GPU) for the variables that are going to be transferred from the host (CPU) at the beginning of the parallel computation. Analogous to the malloc function in C++, the cudaMalloc allocates the required memory in the GPU. This function has as input the name of the variable to be copied and its size, which is defined as an element of type size_t. In our case, we are going to initialize the value function in the GPU with the name V:

```c
float *V;
size_t sizeV = T*ne*nx*sizeof(float);
```

51 The reader will benefit much from following the next pages while checking the code posted on the Github repository.
Since we now have to handle memory in the CPU and memory in the GPU, we must assign different names to objects that live in both. For example, the value function is going to be kept in the host memory, but will be transferred to the device and updated at each iteration. Hence, we must assign a name to the value function in the host, and a different name for the value function in the device. With the `cudaMalloc` function above, we named the value function living in the device as `V`.

Now we have to initialize the value function that lives in the host. For ease of reference, we will precede the name of variables in the host with `h`, so the value function in the host is called `hV`:

```c
float *hV;
hV = (float *)malloc(sizeV);
```

Once memory is allocated in the GPU, the user must transfer from host to device the variables that need to be accessed and/or modified in the parallel computations. The function used to transfer information between host and device is `cudaMemcpy`, which uses as input the object in the device memory that was already initialized (`V`), the object in the host memory that is going to be copied to the device (`hV`), the size of the object (`sizeV`) and the reserved word `cudaMemcpyHostToDevice` that specifies the direction of the transfer:

```c
cudaMemcpy(V, hV, sizeV, cudaMemcpyHostToDevice);
```

The user should initialize and transfer any other variables that will be used in the device for every computation in the same way.
Next, we must take care of defining the dimensions of the blocks and grids for parallelization in the GPU. Each thread in the GPU will have a 5-dimensional “coordinate” given by the number of its grid, the 2-dimensional coordinate of its block, and its 2-dimensional coordinate within the block. This 5-dimensional “coordinate” facilitates parallelization. For example, if the user only wants to parallelize along one dimension—such as the capital $x$ dimension—she can define only one grid and one block of dimensions $1 \times n_x$. In this way, the thread $i$ will be assigned the task of computing the value function at the $i$-th point in the grid for capital. Here, we must be careful not to violate the restriction that any given block can have at most 512 threads, which means that $n_x \leq 512$ must hold.

Similarly, if the user wants to parallelize across 2 dimensions—say, the productivity $e$ and capital $x$ dimensions—she can define one grid with one block of dimensions $n_e \times n_x$. Here, the thread with coordinates $(i, j)$ inside the block will be assigned the task of computing the value function at the $i$-th entry of the productivity grid and the $j$-th entry of the capital grid. Again, we must ensure that $n_e \cdot n_x \leq 512$.

The dimensions of blocks and grids are defined as objects of type `dim3`, using the CUDA constructors `dimBlock` and `dimGrid`. Remember not to assign more than 512 threads per grid!

```cpp
int block_height = 20;
int block_width = ne;
dim3 dimBlock(block_height, block_width);
dim3 dimGrid(nx/block_size, 1);
```

Given that it is very likely that the restriction $n_x \cdot n_e \leq 512$ will be violated, we define only one grid with multiple blocks. Each block has a width of size $n_e$ and a height of
size 20. We define as many blocks as necessary to cover all points in the productivity and capital grids.

As already explained, the job assigned to each thread should be a function of its coordinates inside the 5-dimensional structure defined by the user within the device. The variables blockIdx.x, blockIdx.y, blockIdx.z, blockDim.x, ..., are used to access the coordinates of each thread. Given these coordinates, the user should define the tasks assigned to every thread. For example, in the case where we want to parallelize across the productivity and capital dimensions, the function that computes the value function will start as follows:

```c
__global__ float value(parameters params, float* V, ...){
  int ix = blockIdx.x * blockDim.x + threadIdx.x;
  int ie = threadIdx.y;
  // ...
}
```

This function will compute the value function at the ix point in the capital grid and ie point in the productivity grid. Here, ie is simply given by the y-coordinate inside the block, threadIdx.y, but ix is given both by the coordinate of the block, blockIdx.x, and the x-coordinate within the block, threadIdx.x.

Finally, to compute the value function in parallel, the user can call the function value, explicitly stating the dimensions defined above within <<<...>>>. After calling the function, we must check that all workers finish their job before iterating to t − 1. For this, we use the function cudaDeviceSynchronize right after calling the function value:

```c
value<<<dimGrid, dimBlock>>>(params, V, ...);
```
cudaDeviceSynchronize();

The function value is executed in the device, but is called from the host, so it is defined as _global_. Only functions that are called by the device, within parallel executions, should be preceded by _device_.

The function value is called at every iteration $t \in \{1, \ldots, T\}$. Once the value function is computed for a given $t$, the value function must be copied back to the CPU, again using the cudaMemcpy function. However, the last input must be set as cudaMemcpyDeviceToHost, to specify that $V$ must be copied from device to host:

cudaMemcpy(hV, V, sizeV, cudaMemcpyDeviceToHost);

Finally, the function cudaFree is used to free the memory in the device at the end of the computation:

cudaFree(V);

We run our CUDA code on the same machine as all the previous experiments. The computer is equipped with a GeForce GTX 860M GPU with 640 CUDA cores and 4GiB of memory. The average run time in 20 experiments is 1.86 seconds, slightly better, for instance, than C++-OpenMP in 8 cores. As in the case with MPI, our problem is not complex enough to show all the power of CUDA, but the results give, at least, a flavor of how to use it and its potentialities.
3.8.2 OpenACC

OpenACC is a model designed to allow parallel programming across different computer architectures with minimum effort by the developer. It is built around a very simple set of directives, very similar in design to OpenMP. To implement parallel computing, it suffices to add a couple of lines of code, or directives, before a section of the code the researcher wants to parallelize. Moreover, it is explicitly designed to be portable across different computer architectures. This means that the same program can be compiled to be executed in parallel using the CPU or the GPU (or mixing them), depending on the hardware available. In this way, OpenACC can be used to perform the same task as OpenMP and MPI, with a similar performance, but can also be used to perform GPU computing when available. Given the similarities to its implementation in OpenMP, but the clear advantage of allowing the user to execute the code in the GPU in addition to the CPU, OpenACC represents a more powerful tool, although one still in relatively early stages of development. There are only a couple of books written about it, for example, Farber (2017) and Chandrasekaran and Juckeland (2017), neither of which is fully satisfactory.

OpenACC, like OpenMP, executes the code serially using the fork-join paradigm, until a directive for parallel computing is found. The section of the code that is parallelizable is then computed with the use of multiple CPU (or GPU) threads. Communication between the master and worker threads in the parallel pool is automatically handled, although the user can state directives to grant explicit access to variables, and to transfer objects from the CPU to the GPU when required. After the parallel region is executed, the code is serially executed until a new directive is found. As was the case with OpenMP, Figure 13 illustrates code execution in OpenACC.

The OpenACC website describes multiple compilers, profilers, and debuggers for OpenACC. We use the PGI Community Edition compiler, which is a free release of PGI. The reason for using PGI rather than GCC, as in Section 3.7.6 with OpenMP, is that the PGI
compiler can be used with the CPUs and with NVIDIA Tesla GPUs. In this way, it suffices to select different flags at compilation time to execute the code in the CPU or the GPU, as will become clear in this section.

To implement the parallelization, we must include the directive \texttt{#pragma acc parallel loop} right before the \texttt{for} loop:

\begin{verbatim}
#pragma acc parallel loop
for(int ix=0; ix<nx; ix++){
    // ...
}
\end{verbatim}

This directive has multiple options, among which the \texttt{private(...)} option allows the user to specify which variables are private to each worker. The programmer can also explicitly set barriers that synchronize every thread, synchronize the memory in the CPU and the GPU, and so on. If we want to perform parallelization only with the CPU, in which the memory is shared by workers, this directive and any options included are all that is required. We can go on to compile the code, as explained below, and execute the program. If we want to execute the code with the GPU, a data movement must be specified between the host memory (CPU) and the device memory (GPU), given that the CPU and GPU do not share memory.

Once compiled, the \texttt{OpenACC} code works as follows. The program is executed in serial by the CPU until the first parallelization directive is encountered in the code. At this point, the variables and objects required for each computation must be copied from the host memory (CPU) to the device memory (GPU), before executing the code in parallel. Once the parallel region is executed, for the program to work correctly, the objects that were stored in the device memory must be copied back to the host memory. After that, the
program is executed by the CPU in serial, until another parallel directive is encountered. For this reason, the user must specify the points at which data movements should occur between the CPU and the GPU.

The main directive for data movements is `#pragma acc data copy(...)`, where the variables that should be copied back and forth from CPU to GPU are specified inside the `copy(...)` option. There are different options for data movements, such as `copyin(...)`, which only moves data from CPU to GPU, `copyout(...)`, which only moves data from GPU to CPU, and so on. This directive should be specified right before the `#pragma acc parallel loop` that declares the beginning of a parallel region:

```c
#pragma acc data copy(...)
#pragma acc parallel loop
for(int ix = 0; ix<nx; ix++){
    //...
}
```

If we include the directive `#pragma acc data copy(...)`, but compile the code to be executed only with the CPU, this data movement directive is ignored, a convenient feature for debugging and testing.

At this point, by choosing the appropriate compilation flag, the user can choose to compile the code to be executed in parallel only in the CPU or in the GPU. To compile the code to be executed by the CPU, we must include the `-ta=multicore` flag at compilation. In addition, the `-acc` flag must be added to tell the compiler this is OpenACC code:

```
pgcc Cpp_main_OpenACC.cpp -o Cpp_main -acc -ta=multicore
```
If, instead, the user wants to execute the program with an NVIDIA GPU, we rely on the -ta=nvidia flag:

```
pgc++ Cpp_main_OpenACC.cpp -o Cpp_main -acc -ta=nvidia
```

Importantly, we do not need to modify the code to parallelize with a multicore CPU or with a GPU. This makes OpenACC a very portable parallel computing extension, and reduces significantly the barriers to GPU computing.

When we run the code 20 times on the same computer and GPU described in previous sections, we measure an average run time of 1.42 seconds, a record among all the options presented so far. This result is impressive: given how easy it is to code in OpenACC and the portability of the code among machines and processing units, OpenACC seems to be one of the best available avenues for parallelization.

### 3.9 Beyond CPUs and GPUs

There are several alternatives for parallel computation beyond using CPUs and GPUs. First, there is an intermediate option, the bootable host processors Intel Xeon Phi™, which can pack, in the current generation, up to 72 cores. In this guide, it is not necessary to cover these processors, since MPI and OpenACC can easily handle them with straightforward adaptations of the material presented in previous pages.

Similarly, we will not discuss how to use tensor processing units (TPUs) and field-programmable gate arrays (FPGA). The former have just come on the market and are still in an early stage of development, and we are not aware of any application in economics that uses them. The latter are probably not efficient for most problems in economics due to the hardware development cost. Nevertheless, for examples of this approach in finance, where the cost considerations might be different, see the papers in Schryver (2015).
3.10 Conclusion

As a summary of our discussion, Figure 21 illustrates the absolute performance of the computation with each of the programming languages, according to the number of threads used in the parallel pool (for codes in the GPU, we report a constant line, since, by construction, we do not change the number of cores). The computation times for Julia are calculated with the \texttt{pmap} function. Matlab, Python, and R are not included, because they take much longer than the rest of the languages.

![Figure 21: Computing time (s) by language with different number of processors.]

The best performance is attained when computing the model with CUDA and OpenACC (when the latter performs the computations with the GPU). When looking only at parallel CPU computing, the best performance is obtained with C++--OpenMP and C++--MPI. Rcpp does an excellent job, nearly matching the speed of C++--OpenMP and C++--MPI. Julia falls somewhat behind, pointing perhaps to the need to reconfigure our code to take advantage
of the peculiarities of the language. Our results show also that employing MATLAB, Python, and R for this class of problems is not recommended and that parallelization does not fix the speed problem of these languages.

We conclude our guide by reiterating that parallelization offers a fantastic avenue to improve the performance of computations in economics. Furthermore, parallelization is relatively easy to implement once one understands the basic concepts. Even in Julia, where parallelization is trivial, and with a standard laptop, one can cut in half the run time of a realistic model. If the researcher is willing to jump to OpenMP and OpenACC, the gains are potentially much larger, especially if she has access to a more powerful computer or cloud services.
Appendices
.1 Wall Street or Main Street: Who to Bail Out?

.1.1 Steady State Equilibrium

This definition is a particular case of Definition 1. In the initial and final steady states, there is perfect foresight of house prices and foreclosures, so banks need no bailouts and the government does not have access to the emergency policies described above. This means that $\Omega$ and $\Theta$ are constant, so I will omit the dependency of the problem on the aggregate variables. The following is the definition of a steady state equilibrium of this economy.

Definition 2 (Stationary Recursive Competitive Equilibrium). A stationary recursive competitive equilibrium are a value function $V : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \rightarrow \mathbb{R}$, policy functions for default and refinancing $d, s : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \rightarrow \{0, 1\}$, and all other policy functions $g_{ce}, g_{mr}, g_{hr}, g_{ra} : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \rightarrow \mathbb{R}^+$ for the household, a pricing function $P_m : \{1, \ldots, T\} \times E \times A \times H \times M \rightarrow \mathbb{R}^+$ for mortgages, a price for rental housing $q \in \mathbb{R}^+$, a price for new housing $P_h \in \mathbb{R}^+$, a wage $w \in \mathbb{R}^+$, a risk-free interest rate $r \in \mathbb{R}^+$ and distributions $\Phi : \{1, \ldots, T\} \times E \times A \times H \times M \times \Delta \rightarrow [0, 1]$ such that:

1. [Households Optimize] Given prices $P_m(\cdot), P_h, q, w$ and $r$ the value function $V$ solves the household’s problem (2), with $d, s, g_{keep}, g_{ce}, g_{mr}, g_{hr}, g_{ra}$ and $g_{a}$ the respective policy functions.
2. **[Mortgage Originators Optimize]** Given $P_h$, $q$, $w$, and $r$, the pricing function $P_m(\cdot)$ for mortgages satisfies an expected zero-profit condition loan by loan, given by equation (7).

3. **[Rental Market Clears]** The rental housing price $q$ is such that demand for rental housing is equal to the total supply of housing in the economy:

$$\int_s g_r(s) d\Phi(s) = \bar{H}$$

4. **[Housing Market Clears]** The housing price $P_h$ is such that home-ownership is equal to total housing supply:

$$\int_s g_h(s) d\Phi(s) = \bar{H}$$

5. **[Financial Assets Market Clears]** The risk-free rate $r$ is such that savings in the economy equal total mortgage outlays by mortgage investors:

$$\int_s g_a(s) d\Phi(s) = \int_s P_m(s) g_m(s) d\Phi(s)$$

6. **[Production Firms Optimize]** The wage per efficiency unit of labor $w$ is pinned down by the technology parameter:

$$w = A$$
7. **[Social Security Budget Balance]** The pension \( b \) given to retired households is such that there is budget balance on the social security system:

\[
b \cdot \int_s \mathbb{1}\{t \geq t_{ret}\} d\Phi(s) = \int_s \mathbb{1}\{t < t_{ret}\} \tau_{st} e_{it} \omega_{it}(s) d\Phi(s)
\]

8. **[Resource Constraint]** The resource constraint holds:

\[
A \cdot \int_s e_{it} g_{it}(s) d\Phi(s) = \int_s \left[ g_{c}(s) + \delta P_{ih} h + F \left( \sum_{j=i}^T [\Pi_{i=1}^j r_i] \left[ \frac{\rho}{1+r} \right]^{j-t} \right) g_{m}(s) \right] d\Phi(s)
\]

9. **[Distribution Consistency]** The distribution \( \Phi(t, \cdot) \), for \( t \in \{2, \ldots, T\} \) is consistent with the policy functions for the households, the Markov transition probabilities for the idiosyncratic shocks and the initial distribution \( \Phi(1, \cdot) \).

This is the definition in steady state, where no aggregate shocks occur, so banks perfectly forecast the foreclosure rate in the economy and price the mortgages accordingly. In equilibrium, by the law of large numbers, profits are exactly equal to zero, loan by loan, and there is no government intervention.
.1.2 Computation of the Model

As described in the Definition 2, in equilibrium the household’s problem must be solved and the zero-profit condition for mortgages must hold. Note that the policy and value functions for the household’s problem depend on the mortgage pricing function, and vice versa. This means that both the household’s functions and mortgage pricing function must be solved as a fixed point: a) given \( P_m(t, e, a', h', m'; \Omega) \), the household’s problem must be solved and, b) given the household’s policy functions, the zero-profit condition for mortgages must hold.

To solve the fixed point, the computation of this model is composed by two sub-routines: a) given a pricing function \( P_m(t, e, a', h', m'; \Omega) \), solve the household’s problem described in the recursive formulation in equation (2), by starting at the last period of life and iterating backwards; and b) given the policy functions that solve the household’s problem, solve the fixed point of the pricing function \( P_m(t, e, a', h', m'; \Omega) \) given by equation (7), by doing pricing function iteration.

Given the large number of state variables, I perform the computation using massive parallel GPU computing, as described in Aldrich et al. (2010). Given a set of parameters, the following algorithm is used to compute the equilibrium of the economy:

**Stationary Recursive Competitive Equilibrium**

1. Given parameters \( \beta, \sigma, \kappa, \psi, \eta, \theta_t, \pi_t, T, t^{ret}, \lambda_e, \sigma_e, m, \delta, \rho, \ldots \), start with an initial guess for prices \( P^0_h, q^0, r^0 \).
1. Set $n = 0$ and an initial guess for:

Mortgage pricing function: \( P^0_m(t, e, a', h', m'; \Omega) \)

Policy functions: \( D^0(t, e, a, h, \delta; \Omega), C^0(t, e, a, h, m, \delta; \Omega), \ldots \)

Value function: \( V^0(t, e, a, h, m, \delta; \Omega) \)

2. Iteration $n$:

**Subroutine 1 - Pricing Function Computation:**

Given $D^n(t, e, a, h, m, \delta; \Omega), C^n(t, e, a, h, m, \delta; \Omega), \ldots$

(a) Set $j = 0$ and $P^{n,j}_m(t, e, a', h', m'; \Omega) = P^0_m(t, e, a', h', m'; \Omega)$.

(b) Given $j$ and the policy functions $D^n(t, e, a, h, m, \delta; \Omega), C^n(\cdot), \ldots$

compute $P^{n,j+1}_m(t, e, a', h', m'; \Omega)$, according to equation (7).

(c) If \[ \| P^{n,j+1}_m(t, e, a', h', m'; \Omega) - P^{n,j}_m(t, e, a', h', m'; \Omega) \| < \epsilon, \]

stop and set $P^{n+1}_m(t, e, a', h', m'; \Omega) = P^{n,j+1}_m(t, e, a', h', m'; \Omega)$.

Otherwise, set $j = j + 1$ and repeat (b).

**Subroutine 2 - Value and Policy Functions Computation:**

Given $P^{n+1}_m(t, e, a', h', m'; \Omega)$:

(a) For $j = T$ compute $V^{n+1}(j, e, a, h, m, \delta; \Omega), D^{n+1}(j, e, a, h, m, \delta; \Omega), C^{n+1}(\cdot), \ldots$, by solving the static problem in equation (2)

(b) For $j = T - 1, \ldots, 1$, given $V^{n+1}(j + 1, e, a, h, m, \delta; \Omega)$ compute $V^{n+1}(j, e, a, h, m, \delta; \Omega), \ldots$, by solving the equation (2)

3. If \[ \| P^{n+1}(\cdot) - P^{n,j}_m(\cdot) \| + \| V^{n+1}(\cdot) - V^n(\cdot) \| < \epsilon, \]

continue.

Otherwise, set $n = n + 1$ and repeat 1.
4. If $P_{m}^{n+1}(t,e,a',h',m';\Omega)$ and $V^{n+1}(j,e,a,h,m,\delta;\Omega), D^{n+1}(\cdot), C^{n+1}(\cdot), \ldots$ are such that markets clear (Definition 2), stop. Otherwise, set a new guess for $P^{0}_{m}, q^{0}$ and $r^{0}$ and go to 0.

**Recursive Competitive Equilibrium**

To compute the transitional dynamics of the model, assume that after the unexpected shock the economy takes $S$ periods before getting to the new stationary steady state, after which it stays with certain probability.

0. Given parameters $\beta, \sigma, \kappa, \psi, \eta, \theta_{l}, \pi, T, t^{ret}, \lambda, \sigma, m, \delta, \rho, \ldots$, start with an initial guess for prices along the transition $\{P_{s}, q_{s}, r_{s}\}_{s=1}^{S}$

1. Compute the pre-shock steady state and the corresponding distribution of households $\Phi_{0}(t,e,a',h',m';\Omega)$

2. Iterate Backwards:

   (a) $s = S$: Given $P_{s}, q_{s}, r_{s}$, compute $P_{m}^{s}(t,e,a',h',m';\Omega)$ and $V^{s}(\cdot), \ldots$, using the algorithm for the Stationary Recursive Competitive Equilibrium (Section 4.1.2)

   (b) Compute $P_{m}^{s-1}(t,e,a',h',m';\Omega)$ given $P_{m}^{s}(t,e,a',h',m';\Omega)$ and $V^{s}(t,e,a',h',m';\Omega), \ldots$

   (c) Compute $V^{s-1}(t,e,a',h',m';\Omega), \ldots$, given $P_{m}^{s-1}(t,e,a',h',m';\Omega)$ and $V^{s}(t,e,a',h',m';\Omega), \ldots$

   (d) If $s > 1$, set $s = s - 1$ and go to (b). Otherwise, continue.

3. Iterate Forward:

   (a) $s = 1$: Set $\Phi_{1}(t,e,a',h',m';\Omega) = \Phi_{0}(t,e,a',h',m';\Omega)$

   (b) Compute $\Phi_{s+1}(t,e,a',h',m';\Omega)$, given $\Phi_{s}(t,e,a',h',m';\Omega)$ and the policy functions $A^{s}(t,e,a',h',m';\Omega), \ldots$
(c) Compute aggregate quantities at \( s + 1 \), given prices \( P_{s+1}, q_{s+1}, r_{s+1} \), distribution of individuals \( \Phi_{s+1}(t, e, a', h', m'; \Omega) \) and policy functions \( A^{s+1}(t, e, a', h', m'; \Omega), \ldots \), as described in Definition 1.

(d) If \( s < S - 1 \), set \( s = s + 1 \) and go to (b). Otherwise, continue.

4. If markets clear at \( s = 1, \ldots, S \) (Definition 1), stop. Otherwise, set a new guess for \( \{P_s, q_s, r_s\}_{s=1}^S \) and go to 1.

The tolerance level is given by \( \epsilon > 0 \), and the number of periods the economy takes to arrive to the new steady state is assumed to be \( S = 10 \), that correspond to 20 years in the data.
.1.3 Proves of Propositions

Proof. The proof is done by induction on $t$. Define the following budget correspondence, based on the household’s problem:

$$
\Gamma^{\text{keep}}(t, e, a, h, m, \delta; \Omega) = \{ (l, c, a', s, h') \in [0, 1] \times \mathbb{R}^2 \times \mathcal{H}^2 \mid c + (1 - \tau_m)m + qs + P_a a' + P_h \delta h \leq (1 - \tau_l)e\bar{e}wl + a + qh \}$$

In period $t = T + 1$, $V^{\text{keep}}(T + 1, \cdot) = 0$, so the value function $V^{\text{keep}}(T, \cdot)$ only depends on $a$ through the budget correspondence $\Gamma^{\text{keep}}(t, e, a, h, m, \delta; \Omega)$. Let $a_1 \leq a_2$. Given that $\Gamma^{\text{keep}}(T, e, a_1, h, m, \delta; \Omega) \subseteq \Gamma^{\text{keep}}(T, e, a_2, h, m, \delta; \Omega)$, it is straightforward that $V^{\text{keep}}(T, e, a_1, h, m, \delta; \Omega) \leq V^{\text{keep}}(T, e, a_2, h, m, \delta; \Omega)$, as the maximization is done over a larger set. Analogously, $V^{\text{def}}(T, \cdot)$ and $V^{\text{ref}}(T, \cdot)$ are increasing on $a$. Given that the maximum of three functions that are increasing on $a$ is itself increasing on $a$, then $V(T, \cdot)$ is increasing on $a$. By an analogous argument, $V(T, \cdot)$ is increasing on $h$ and decreasing on $m$ and $\delta$. Continuity of $V(T, \cdot)$ follows from Berge’s Maximum Theorem, given that $u(c, s)$ is continuous, the correspondence $\Gamma^{\text{keep}}(t, e, a, h, m, \delta; \Omega)$ is non-empty, compact, upper- and lower-semicontinuous.

Now, assume $V(t + 1, \cdot)$ is increasing on $a$ and $h$ and decreasing on $m$ and $\delta$. Note that $V(t, \cdot)$ only depends on $a, h, m$ and $\delta$ through the budget set. By the same argument as above, $V(t, \cdot)$ is increasing on $a$ and $h$ and decreasing on $m$ and $\delta$.

To prove that $V(t, \cdot)$ is increasing on $e$, assume $V(t + 1, \cdot)$ is increasing on $e'$. Note that for positive values of $\rho_e$, and $e_1 \leq e_2$ the conditional distribution $F_e(e'|e_2)$ first-order
stochastically dominates \( F_e(e' | e_1) \), given that \( F_e(e' | e_2) \leq F_e(e' | e_1), \forall e' \). Given that \( V(t + 1, \cdot) \) is increasing on \( e \), this implies that 
\[
E_{e' | e_2, \delta', m'} V(t + 1, e', a', h', m', \delta'; \Omega') \geq E_{e' | e_1, \delta', m'} V(t + 1, e', a', h', m', \delta'; \Omega') \quad \text{for all } e', a', h', m', \delta' \text{ and } \Omega'.
\]
1.4 Other Figures

Figure 22: Life cycle averages of (a) Owner-occupied housing, (b) Mortgage size, (c) Home equity, and (d) Savings


.2 General Equilibrium Effects of Student Loans on the Provision and Demand for Higher Education

.2.1 Appendix A

The problem of the households is:

$$\max_{c, l, h, a} c^{1-\sigma} + \beta c'^{1-\sigma} \quad \text{s.t.}$$

$$a + c + hP_h + lP_l = w\theta(1 - h)(1 - l) + b$$

$$c' = w\theta + w\theta z_h h + w\theta z_l l + (1 + r)a$$

Solution of the unconstrained households:

Proof of Theorem 1. The unconstrained consumptions are:

$$c^N = \frac{(\beta(1 + r))^{-1/\sigma} (w\theta(2 + r) + (1 + r)b)}{1 + (\beta(1 + r))^{-1/\sigma} (1 + r)}$$

$$c'^N = \frac{(w\theta(2 + r) + (1 + r)b)}{1 + (\beta(1 + r))^{-1/\sigma} (1 + r)}$$

$$c^l = \frac{(\beta(1 + r))^{-1/\sigma} (w\theta(1 + z_l) + (1 + r)b - P_l(1 + r))}{1 + (\beta(1 + r))^{-1/\sigma} (1 + r)}$$

$$c'^l = \frac{(w\theta(1 + z_l) + (1 + r)b - P_l(1 + r))}{1 + (\beta(1 + r))^{-1/\sigma} (1 + r)}$$

$$c^h = \frac{(\beta(1 + r))^{-1/\sigma} (w\theta(1 + z_h) + (1 + r)b - P_h(1 + r))}{1 + (\beta(1 + r))^{-1/\sigma} (1 + r)}$$
The utilities of each of the options are:

\[ u_N = \Phi \times \left( w \theta (2 + r) + b(1 + r) \right)^{1-\sigma} \]

\[ u_l = \Phi \times \left( w \theta (1 + z_l) + b(1 + r) - P_l (1 + r) \right)^{1-\sigma} \]

\[ u_h = \Phi \times \left( w \theta (1 + z_h) + b(1 + r) - P_h (1 + r) \right)^{1-\sigma} \]

where

\[ \Phi = \left( \frac{1}{1 - \sigma} \right) \left( \frac{1}{1 + (\beta(1 + r))^{-1/\sigma} (1 + r)} \right)^{1-\sigma} \left( (\beta(1 + r))^{(\sigma-1)/\sigma} + \beta \right) \]

The household’s decision of whether and where to study follows a cut-off rule on \( \theta \), and the decision is independent of initial wealth, \( b \). The cut-offs are:

\[ \bar{\theta}_l = \frac{1 + r}{w} \left( \frac{P_l}{z_l - (1 + r)} \right), \quad \bar{\theta}_h = \frac{1 + r}{w} \left( \frac{P_h - P_l}{z_h - z_l} \right) \]

Wealth cutoff rules for households:

Proof of Theorem 2. The debt levels of the unconstrained households are:
Given the exogenous borrowing constraint \( \bar{A} \), for a given \( \theta \) we can construct a cut-off \( \bar{b}(\theta) \) on the initial wealth such that individuals with \( b < \bar{b}(\theta) \) are constrained and \( b \geq \bar{b}(\theta) \) are unconstrained. These are given by:

\[
a^N = \frac{\theta(1 - (\beta(1 + r))^{-1/\sigma}) + b}{1 + \beta(1 + r))^{-1/\sigma} (1 + r)}
\]

\[
a^l = \frac{b - P_l - (\beta(1 + r))^{-1/\sigma} \theta(1 + z_l)}{1 + \beta(1 + r))^{-1/\sigma} (1 + r)}
\]

\[
a^h = \frac{b - P_h - (\beta(1 + r))^{-1/\sigma} \theta(1 + z_h)}{1 + \beta(1 + r))^{-1/\sigma} (1 + r)}
\]

That is, the cut-offs are:

\[
\bar{b}_N(\theta) = -\bar{A}(1 + (\beta(1 + r))^{-1/\sigma} (1 + r)) - \theta(1 - (\beta(1 + r))^{-1/\sigma})
\]

\[
\bar{b}_l(\theta) = P_l + (\beta(1 + r))^{-1/\sigma} \theta(1 + z_l) - \bar{A}(1 + (\beta(1 + r))^{-1/\sigma} (1 + r))
\]

\[
\bar{b}_h(\theta) = P_h + (\beta(1 + r))^{-1/\sigma} \theta(1 + z_h) - \bar{A}(1 + (\beta(1 + r))^{-1/\sigma} (1 + r))
\]
This subdivides the state space in three subregions, as shown in the following Figure 2.

Solution of the constrained households:

Next, we have to consider the decision of studying of those households that are constrained. Note that, although if an individual is borrowing constrained when he decides to study, he might prefer to study and not smooth consumption, than not studying and being able to smooth consumption. Therefore, we must compare the utility of studying while being constrained, with the utility of not studying and being unconstrained. The constrained consumptions are given by:

\[ c^N_c = w\theta + b + \bar{A}, \quad c'^N_c = w\theta - (1 + r)\bar{A} \]
\[ c^l_c = b - P_l + \bar{A}, \quad c'^l_c = w\theta(1 + z_l) - (1 + r)\bar{A} \]
\[ c^h_c = b - P_h + \bar{A}, \quad c'^h_c = w\theta(1 + z_h) - (1 + r)\bar{A} \]

There are three decisions to characterize:

1. Whether to study in \( l \) or not study, for individuals that are constrained when studying in \( l \). These individuals will study in \( l \) whenever:

\[
\left( \frac{1}{1 - \sigma} \right) (b - P_l + \bar{A})^{1-\sigma} + \left( \frac{\beta}{1 - \sigma} \right) (w\theta(1 + z_l) - (1 + r)\bar{A})^{1-\sigma} - \\
\Phi \times (w\theta(2 + r) + b(1 + r))^{1-\sigma} \geq 0
\]
2. Whether to study in \( l \) or in \( h \), for individuals that are constrained when studying in \( h \) but not constrained when studying in \( l \). These individuals will study in \( h \) whenever:

\[
\left( \frac{1}{1 - \sigma} \right) (b - P_l + \bar{A})^{1 - \sigma} + \left( \frac{\beta}{1 - \sigma} \right) (w\theta(1 + z_h) - (1 + r)\bar{A})^{1 - \sigma} - \\
\Phi \times \left( w\theta(1 + z_l) + b(1 + r) - P_l(1 + r) \right)^{1 - \sigma} \geq 0
\]

3. Whether to study in \( l \) or in \( h \), for individuals that are constrained when they decide to study in \( h \) or \( l \). These individuals will study in \( h \) whenever:

\[
\left( \frac{1}{1 - \sigma} \right) (b - P_h + \bar{A})^{1 - \sigma} + \left( \frac{\beta}{1 - \sigma} \right) (w\theta(1 + z_h) - (1 + r)\bar{A})^{1 - \sigma} \\
\left( \frac{1}{1 - \sigma} \right) (b - P_l + \bar{A})^{1 - \sigma} - \left( \frac{\beta}{1 - \sigma} \right) (w\theta(1 + z_l) - (1 + r)\bar{A})^{1 - \sigma} \geq 0
\]

The cut-offs that define the college decision for constrained individuals are defined in the following theorem proofs:

**Proof of Theorem 3.** Define the following function:

\[
G(\theta, b) = \left( \frac{1}{1 - \sigma} \right) (b - P_l + \bar{A})^{1 - \sigma} + \left( \frac{\beta}{1 - \sigma} \right) (w\theta(1 + z_l) - (1 + r)\bar{A})^{1 - \sigma} - \\
\Phi \times \left( w\theta(2 + r) + b(1 + r) \right)^{1 - \sigma}
\]

Let the function \( \bar{b}_c^l(\theta) \) be implicitly defined by the equality \( G(\theta, \bar{b}_c^l(\theta)) = 0 \). By the implicit function theorem,

\[
\frac{\partial \bar{b}_c^l(\theta)}{\partial \theta} = -\frac{\partial G/\partial \theta}{\partial G/\partial b}
\]
Setting \( \partial G/\partial \theta = 0 \) gives the result in Theorem 3.

**Proof of Theorem 4**

**Proof of Theorem 4.** The proof is similar to Proof 2.1. Define:

\[
G^*(\theta, b) = \left( \frac{1}{1-\sigma} \right) (b - P_h + \bar{A})^{1-\sigma} + \left( \frac{\beta}{1-\sigma} \right) (w\theta(1 + z_h) - (1 + r)\bar{A})^{1-\sigma} - \\
\Phi(w\theta^*(1 + z_l) + b(1 + r) - P_l(1 + r))^{1-\sigma}
\]

and setting \( \partial G/\partial \theta = 0 \) gives the result in Theorem 4.

**Proof of Theorem 5**

**Proof of Theorem 5.** By implicit function theorem, \( \partial b/\partial \bar{A} = -\frac{\partial G/\partial \bar{A}}{\partial G/\partial b} \).

\[
\frac{\partial G}{\partial \bar{A}} = (b - P_l + \bar{A})^{-\sigma} + \beta(1 + r)(w\theta(1 + z_l) - (1 + r)\bar{A})^{-\sigma} \\
\geq 0
\]

Since \( \partial G/\partial b > 0 \), the first result follows.

For the second result, note that:
\[
\frac{\partial b}{\partial A \partial \theta} = \frac{1}{(\cdot)^2} \left[ \left( \sigma \beta (1 + r) w (1 + z_i)(w \theta (1 + z_i) - (1 + r) \bar{A})^{-(1+\sigma)} \right) \cdot ((1 + r)((b - P_l + \bar{A})^{-\sigma} - \Phi(1 - \sigma)(1 + r)(w \theta (2 + r) + b(1 + r))^{-\sigma})) \right] \\
+ \frac{1}{(\cdot)^2} \left[ ((b - P_l + \bar{A})^{-\sigma} + \beta (1 + r)(w \theta (1 + z_i) - \bar{A}(1 + r))^{-\sigma}) \cdot (\sigma \Phi(1 - \sigma)w(1 + r)(2 + r)(w \theta (2 + r) + b(1 + r))^{-\sigma}) \right] \\
\geq 0
\]

This proves Theorem 5.

\[ \square \]

**Computation of Nash Equilibrium**

In this section we will describe the algorithm used to compute the Nash Equilibrium between elite and non-elite universities. The Nash Equilibrium is composed by a tuple \((P^*_h, \theta^*_h, P^*_l, \theta^*_l)\) such that:

\[
(P^*_i, \theta^*_i) \in \arg \max_{(P_i, \theta_i) \in \mathbb{R}^+ \times [0,1]} \left( z_i(P_i, \theta_i, P^*_l, \theta^*_l) \right)^{\alpha} \left( \sigma b_i(P_i, \theta_i, P^*_l, \theta^*_l) \right)^{1-\alpha}
\]

Note that the problem defined in 4 involves solving for a fixed point nested within another fixed point problem. In particular, the universities will offer a given level of \(z_i, z_h\) to the households and, conditional on such offer households will demand education services that need to fulfill the promised levels of \(z_i, z_h\). This implies that when solving for the optimal of the universities we need to take into account that the offered level of productivities need to be satisfied by the demand of educational services. The full procedure to find the Equilibrium is described below:

**Computation of the Nash Equilibrium**
1. Start algorithm with some initial guess \( \langle P_h^0, \theta_h^0, P_l^0, \theta_l^0 \rangle \). Set \( E = 10 \).

2. Find \( \langle P_h^T, \theta_h^T \rangle \in \arg \max_{\langle P_h, \theta_h, P_l, \theta_l \rangle \in \mathbb{R}^4 \times [0,1]} \left( z_h(P_h, \theta_h, P_l, \theta_l) \right) \) \( a^{(1-a)} \left( c_h(P_h, \theta_h, P_l, \theta_l) \right) \)

   (a) Set \( \langle P_h^0, \theta_h^0 \rangle = \langle P_h^T, \theta_h^T \rangle \)

   (b) Given \( \langle P_h^0, \theta_h^0, P_l^0, \theta_l^0 \rangle \), go to 5. to compute \( (z_h, z_l) \)

   (c) Given \( S = \langle P_h^0, \theta_h^0, P_l^0, \theta_l^0, z_h, z_l \rangle \) compute the objective function of the university \( H(S) \).

   (d) Update for a new guess of the optimal \( \langle P_h^T, \theta_h^T \rangle = \langle P_h^0, \theta_h^0 \rangle \) according to some rule.

   (e) Repeat (b) – (d) until optimal \( \langle P_h^T, \theta_h^T \rangle \) is found

3. Find \( \langle P_l^T, \theta_l^T \rangle \in \arg \max_{\langle P_h, \theta_h, P_l, \theta_l \rangle \in \mathbb{R}^4 \times [0,1]} \left( z_l(P_h, \theta_h, P_l, \theta_l) \right) \) \( a^{(1-a)} \left( c_l(P_h, \theta_h, P_l, \theta_l) \right) \)

   (a) Set \( \langle P_l^0, \theta_l^0 \rangle = \langle P_l^T, \theta_l^T \rangle \)

   (b) Given \( \langle P_h^0, \theta_h^0, P_l^0, \theta_l^0 \rangle \), go to 5. to compute \( (z_h, z_l) \)

   (c) Given \( S = \langle P_h^0, \theta_h^0, P_l^0, \theta_l^0, z_h, z_l \rangle \) compute the objective function of the university \( L(S) \).

   (d) Update for a new guess of the optimal \( \langle P_l^T, \theta_l^T \rangle = \langle P_l^0, \theta_l^0 \rangle \)

   (e) Repeat (b) – (d) until optimal \( \langle P_l^T, \theta_l^T \rangle \) is found

4. Set \( E = ||\langle P_h^0, \theta_h^0, P_l^0, \theta_l^0 \rangle - \langle P_h^T, \theta_h^T, P_l^T, \theta_l^T \rangle|| \). If \( E \) is smaller than a tolerance level, stop the algorithm, the NE is given by the tuple \( \langle P_h^T, \theta_h^T, P_l^T, \theta_l^T \rangle \). Otherwise, set \( \langle P_h^0, \theta_h^0, P_l^0, \theta_l^0 \rangle = \langle P_h^T, \theta_h^T, P_l^T, \theta_l^T \rangle \) and go to 2.

5. Computation of \( (z_h, z_l) \) given \( \langle P_h, \theta_h, P_l, \theta_l \rangle \)

   (a) Start algorithm with some initial guess \( \langle z_h^0, z_l^0 \rangle \) and set \( \epsilon = 10 \)

   (b) Given \( \langle P_h, \theta_h, P_l, \theta_l \rangle \), the guess \( \langle z_h^0, z_l^0 \rangle \) and the policy functions of the households, compute the realized values of \( \langle z_h^0, z_l^0 \rangle \)

   (c) set \( \epsilon = (z_h^0 - z_h^0)^2 + (z_l^0 - z_l^0)^2 \).

   (d) If \( \epsilon \) is smaller to a tolerance level, the algorithm is complete. Otherwise, set \( \langle z_h^0, z_l^0 \rangle = \langle z_h^0, z_l^0 \rangle \) and go to (b).
Analysis in the linear case

In order to get a clear idea of how credit constraints affect the market for higher education, we illustrate the linear case where $\sigma = 1$. Furthermore, we need to distinguish scenarios where households would like to substitute future for current consumption and the other way around. This is given by the inequality $\beta(1 + r) < 1$. Whenever this inequality is satisfied, households would prefer to get as much debt during the first period. The opposite case, when $\beta(1 + r) \geq 1$ will motivate households to save as much as possible given that the returns to savings, in terms of utility, are more than one to one.

**Case 1. $\beta(1 + r) \geq 1$**

In this case, households will prefer to save as much as they want and then the value functions for each case (not study, study in low quality university or study in high quality university) are given by:

$$V^N(b, \theta) = \beta [b(1 - \tau)(1 + r) + w\theta(2 + r)]$$

The value function for households going to the low quality university is only defined whenever they can afford it. That is, whenever $P_l - b(1 - \tau) \leq \min\{A, \frac{w\theta(1 + \gamma)}{1 + r}\}$. In particular, consider the case where $P_l - b(1 - \tau) \leq 0$. If this holds, then households are able to afford the price of education with their income after taxes and thus we have no concerns about they not getting enough debt to fund their education.
However, when students should get positive debt in order to attend the low quality university, the amount of debt should satisfy two constraints:

\[ P_l - b(1 - \tau) \leq \bar{A} \]  
\[ P_l - b(1 - \tau) \leq \frac{w\theta(1 + z_l)}{1 + r} \]

The constraint given in 7 states that the amount of debt students get should not exceed the upper limit given exogenously in the economy. The inequality given in 8 guarantees that students have enough funds to get the necessary debt to attend college. The two aforementioned inequalities give bounds in \( b \) and \( \theta \) for students to being able to pay the tuition in the low quality college:

\[ b \geq b_{pl} = \frac{\bar{A} - P_l}{1 - \tau} \]  
\[ b \geq L(\theta) = \frac{P_l}{1 - \tau} - \frac{w\theta(1 + z_l)}{(1 - \tau)(1 + r)} \]

Now, for households with state variables \((b, \theta)\) such that low quality education is affordable, we can define the value of going to the low university as:

\[ V^L(b, \theta) = \beta \left[ (b(1 - \tau) - P_l)(1 + r) + w\theta(1 + z_l) \right] \]
Similarly, in order to be able to go to the high quality institutions, it should be the case that:

\[ b \geq b_{ph} = \frac{A - P_h}{1 - \tau} \]  

\[ b \geq H(\theta) = \frac{P_h}{1 - \tau} - \frac{w\theta(1 + z^h)}{(1 - \tau)(1 + r)} \]  

For those households, we can define the value of going to the high quality college as:

\[ V^H(b, \theta) = \beta \left[ (b(1 - \tau) - P_h)(1 + r) + w\theta(1 + z^h) \right] \]  

Consider the case of a person who is deciding whether to go to the low quality college or not study. In such case, granted that he could afford to pay tuition, he will decide to attend whenever \( V^L(b, \theta) \geq V^N(b, \theta) \). This implies that the decision will be to go to the low quality college whenever:

\[ \theta_l \geq \theta_L = \frac{P_l(1 + r)}{w(z^l - r - 1)} \]  

Similarly, when a person is deciding whether to go to the high quality college or to the low quality one, and granted he could afford both, the relevant decision rule will be to go to the high quality college whenever \( V^H(b, \theta) \geq V^L(b, \theta) \). This inequality generates the decision rule of going to college whenever:

\[ \theta \geq \theta_H = \frac{(P_h - P_l)(1 + r)}{w(z^h - z^l)} \]  

The decision rules can be represented in the state space according to the following graph:
Figure 23: Representation of the education decisions on the state space.

Note that we can express $N^H$ in terms of elements that we have found previously:

$$N^H = \int_{\theta^H}^{\theta^b} \int_{H(\theta)}^{b_H} dF(b, \theta) + \int_{0}^{1} \int_{b_{pl}}^{b} dF(b, \theta)$$

(17)

where $\theta^H$ is the maximum level of bequests in the state space and

$$\theta^H = \frac{(1 + r)A}{(1 + z^H)w}$$

(18)

For the sake of simplicity, we will assume a uniform distribution for $(b, \theta)$. As long as $P^h > P^l$ and $z^h > z^l$ we can express the measure of people going to the high quality university as:

$$N^H = \frac{1}{b} \left[ \left( b - \frac{P_h}{1 - \tau} \right) \left( \frac{(1 + r)A}{(1 + z^h)w} - \frac{(P_h - P_l)(1 + r)}{w(z^h - z^l)} \right) \right]$$

(19)
Similarly, the average level of skills of people attending such college is given by:

\[ \bar{\theta}^H = \frac{1}{b} \left[ \left( \frac{(1+r)\bar{A}}{(1+z^h)w} \right)^2 - \left( \frac{(P_h - P_l)(1+r)}{w(z^h - z^l)} \right)^2 \right] \left( \frac{\bar{b}}{2} - \frac{P_h}{2(1-\tau)} \right) + \]

\[ \frac{w(1+z^h)}{3(1-\tau)(1+r)} \left[ \left( \frac{(1+r)\bar{A}}{w(1+z^h)} \right)^3 - \left( \frac{(P_h - P_l)(1+r)}{w(z^h - z^l)} \right)^3 \right] + \]

\[ \frac{1}{2} \left[ \bar{b} - \frac{\bar{A}}{1-\tau} + \frac{P_h}{1-\tau} \left( 1 - \left( \frac{(1+r)\bar{A}}{1(1+z^h)} \right)^2 \right) \right] \]

We can express the relevant variables for low quality college, granted \( P_h > P_l \) and \( z_h > z_l \), as:

\[ N^L = \int_{L(\theta)}^{\bar{\theta}^L} dF(b, \theta) + \int_{\bar{\theta}^L}^{\bar{\theta}^H} f^{H(\theta)} L(\theta) dF(b, \theta) + \]

\[ \int_{\tilde{b}^{P_l}}^{\bar{b}^{P_l}} f^{H(\theta)} dF(b, \theta) + \int_{b^{P_l}}^{1} f^{b^{P_l}} dF(b, \theta) \]

\[ \bar{\theta}^L = \int_{\tilde{\theta}_L}^{\bar{\theta}^L} 1 0 dF(b, \theta) + \int_{\bar{\theta}^L}^{\bar{\theta}^H} 1 0 dF(b, \theta) + \]

\[ \int_{\tilde{b}^{P_l}}^{b^{P_l}} 1 0 dF(b, \theta) + \int_{b^{P_l}}^{1} 1 0 dF(b, \theta) \]
\[ \mu_{b_L} = \int_{\theta_L}^{\theta_{L_1}} \int_{L(\theta)}^{1} b dF(b, \theta) + \int_{\theta_{L_1}}^{H(\theta)} \int_{L(\theta)}^{1} b dF(b, \theta) + \int_{\theta_{L_1}}^{\theta_{H_1}} \int_{b_{L_1}}^{1} b dF(b, \theta) + \int_{\theta_{H_1}}^{b_{H_1}} b dF(b, \theta) \] (23)

It is important to note that throughout this analysis we have not implemented the fact that both colleges are able to set a threshold rule such that people with a level of skills below such threshold will not be admitted. In such a case, we will simply modify the regions of integration to consider that only people with ability beyond the threshold will be able to attend.

**Existence of equilibrium**

The expressions found in 19, 20, 21 and 22 can be used to express the necessary conditions that the offered qualities need to satisfy in equilibrium. In particular, we need to find \( z^h, z^l \) such that:

\[
\begin{bmatrix}
  z^h \\
  z^l
\end{bmatrix} = \begin{bmatrix}
  k^h \left( \tilde{\theta}^h(\bar{\theta}^h, \bar{\theta}^l, P_h, P_l, z^h, z^l) \right)^{a_1} \left( I(\bar{\theta}^h, \bar{\theta}^l, P_h, P_l, z^h, z^l) \right)^{a_2} \\
  k^l \left( \tilde{\theta}^l(\bar{\theta}^h, \bar{\theta}^l, P_h, P_l, z^h, z^l) \right)^{a_1} \left( I(\bar{\theta}^h, \bar{\theta}^l, P_h, P_l, z^h, z^l) \right)^{a_2}
\end{bmatrix}
\] (24)

We need to prove existence of a fixed point in the qualities offered by universities before proving the existence of the Nash Equilibrium. Note, however, that difficulty arises in this point given the fact that there is no natural way to bound the set of qualities offered by the universities. Additionally, note that equations 19, 20, 21 are not continuous in \( z^h = z^l \). The inability of proving the existence of a fixed point in the qualities offered by universities shows that it is not possible to prove existence of the Nash Equilibrium. We rely purely on the computational analysis to find a Nash Equilibrium in this case that
might not be unique.

**Case 2.** $\beta(1 + r) < 1$

This case is more involved as households value more current consumption than future and will try to get as much debt as possible. The difficulty arises as even when students can afford to pay college, they might be constrained given that they want to substitute future by current consumption. Additionally, we need to establish which is the relevant constraint that households face when getting the desired level of debt, either the exogenously given level of credit constraint or they reach a point where they can’t fund the debt with their resources.

We start analyzing the case of a person who is not going to university. In this case, the person will get as much debt as possible and he will be constrained whenever $\frac{w\theta}{1+r} > \tilde{A}$. If this is the case, the person will get the maximum level of debt $\tilde{A}$. Taking into account this case when computing the value of not going to college, we see that:

\[
V^N(b, \theta) = \begin{cases} 
  b(1 - \tau) + w\theta \frac{2r}{1+r} & \text{if } \theta \leq \frac{\tilde{A}(1+r)}{w} \\
  b(1 - \tau) + w(\theta)(1 + \beta) + \tilde{A}[1 - \beta(1+r)] & \text{if } \theta > \frac{\tilde{A}(1+r)}{w} 
\end{cases}
\]  

(25)

Now, let's consider a household that goes to the low-quality university. Evidently, the value function will only be defined for the case when it is possible to pay tuition price via endowment or debt. For people whose income is below the tuition price ($b(1 - \tau) < P_l$) and who are constrained either by the exogenous level $\tilde{A}$ or by their earning capacity $\frac{w\theta(1+z_l)}{1+r}$, the value of going to the low quality college will not be defined.

An individual who is not constrained and takes as much debt as he can, will derive utility given by $b(1 - \tau) - P_l + \frac{w\theta(1+z_l)}{1+r}$. The first term, $b(1 - \tau) - P_l$ corresponds to net income after tuition and the remaining part $\frac{w\theta(1+z_l)}{1+r}$ is simply the amount they will make.
in the second period taken to the present value of the first period.

If the net income after tuition is negative, an individual will not be credit constrained so long as:

\[ P_l - b(1 - \tau) \leq \min\{ \bar{A}, \frac{w(1 + z^l)}{1 + r} \} \]  

(26)

However, it is possible to have individuals who are borrowing constrained even if the net income after tuition is positive. These individuals are those who would like to borrow against their future income, given that current consumption is more valuable than future consumption, but they are not able to borrow as much as they want given the exogenous limit \( \bar{A} \). Those are individuals such that:

\[ \frac{w(1 + z^l)}{(1 + r)} < \bar{A} \]  

(27)

and they are forced to borrow no more than \( \bar{A} \). This implies that we can define the value of going to low-quality college as:

\[ V^L(b, \theta) = \begin{cases} 
    b(1 - \tau) - P_l + \frac{w(1 + z^l)}{1 + r} & \text{if } b(1 - \tau) - P_l \geq 0 \quad \theta \leq \frac{\bar{A}(1+r)}{w(1+z^l)} \\
    b(1 - \tau) - P_l < 0 & \text{or } b(1 - \tau) - P_l - b(1 - \tau) \leq \min\{ \bar{A}, \frac{w(1+z^l)}{1+r} \} \\
    b(1 - \tau) - P_l + \bar{A}[1 - \beta(1+r)] + w\beta(1+z^l) & \text{if } b(1 - \tau) - P_l > 0 \text{ and } \theta > \frac{\bar{A}(1+r)}{w(1+z^l)}
\end{cases} \]  

(28)
Finally, doing the same analysis but with $P_h$ and $z^h$ we can find the value of going to the high quality college:

$$V^{H}(b, \theta) = \begin{cases} 
  b(1 - \tau) - P_h + \frac{w(1 + z^h)}{1 + r} & \text{if } b(1 - \tau) - P_h \geq 0 \quad \theta \leq \frac{\bar{A}(1 + \rho)}{\bar{A}(1 + \rho)} \\
  \text{or} \\
  b(1 - \tau) < 0 \quad P_h - b(1 - \tau) \leq \min\{\bar{A}, \frac{w(1 + z^h)}{1 + r}\} \\
  b(1 - \tau) - P_h + A[1 - \beta(1 + r)] + w\beta(1 + z^h) & \text{if } b(1 - \tau) - P_h > 0 \text{ and } \theta > \frac{\bar{A}(1 + \rho)}{\bar{A}(1 + \rho)} 
\end{cases}$$

(29)

**Life-cycle Model**

In this section we embed a life-cycle model into a two-period model, so our calibration of Section 2.5.2 is realistic. We solve the household’s problem in two parts: 1) during the study periods, $t = 0, \ldots, S - 1$, and 2) after college age, $S, \ldots, T$, and leave the problem expressed as a two-period maximization problem in which households decide how much to save for post-college periods. First, we start by solving the post-college optimization problem. We assume that after college graduation, individuals enter perfect financial markets, so there is perfect consumption smoothing. The problem of the households is:

$$\max_{c_t} \sum_{t=S}^{T} \beta^{t-S} \frac{c_t^{1-\sigma}}{1 - \sigma}, \quad s.t.$$

$$c_S = b + a_{S+1} + w(1 + z_j)\theta$$

$$c_t + a_t(1 + r) = a_{t+1} + w(1 + z_j)\theta, \quad t \in \{S, \ldots, T\}$$

where $a_{t+1}$ is the debt at period $t$ to be repaid next period, and $b$ are the savings that the individual carries from the college years. In here, we assume that there are no borrowing
constraints, since households enter perfect financial markets. Solving this problem, yields
the present value budget constraint in period $S$:

$$\sum_{t=S}^{T} \frac{c_t}{(1 + r)^{t-S}} = b + \sum_{t=S}^{T} \frac{w\theta(1 + z_j)}{(1 + r)^{t-S}}$$

Combining this with the Euler equation, the optimal consumption path is given by:

$$c_S = \frac{1}{\Phi_S} (b + w(1 + z_j)\theta\Phi_r)$$

$$c_t = ((1 + r)\beta)^{t-S} c_S, \quad t \in \{S, \ldots, T\}$$

where $\Phi_S$ and $\Phi_r$ are given by the following expressions:

$$\Phi_S = \frac{1 - \left(\frac{\beta}{(1+r)^{T-S}}\right)^{T-S+1}}{1 - \left(\frac{\beta}{(1+r)^{T-S}}\right)^{S}}$$

$$\Phi_r^o = \frac{1 - \left(\frac{1}{(1+r)^{T-S}}\right)^{T-S+1}}{1 - \left(\frac{1}{(1+r)^{T-S}}\right)^{S}}$$

The present value utility at time $S$ of this consumption path is given by:

$$\sum_{t=S}^{T} \beta^{t-S} u(c_t) = \Phi_S u(c_S)$$

Note that $c_S$ is determined for every given savings $b$ carried from the college period, so
without solving the problem for periods $\{0, \ldots, S-1\}$, it will not be completely pinned
down. Now, we solve for the households’ problem during periods $0, \ldots, S-1$. Given
that during college, there exist exogenous borrowing constraints given by $\bar{A}$, there are
two cases: \( a \) individuals are unconstrained, and \( b \) individuals are constrained. The unconstrained solution of the problem in periods \( \{0, \ldots, S - 1\} \) yields:

\[
c_0 \Phi_0 + (P_h h + P_l l) \Phi_y^y + \frac{a}{1 + r^S} = w \Phi_y^y (1 - l)(1 - h) + b
\]

\[
c_t = ((1 + r) \beta)^t c_0, \quad t \in \{1, \ldots, S - 1\}
\]

where \( b \) is the initial wealth of individuals, and \( \Phi_0, \Phi_y^y \) are given by:

\[
\Phi_0 = \frac{1 - \left( \frac{\beta}{(1 + r)^{S-1}} \right)^{\frac{S}{\beta}}}{1 - \left( \frac{\beta}{(1 + r)^{S-1}} \right)^{\frac{S}{\beta}}}
\]

\[
\Phi_y^y = \frac{1 - \left( \frac{1}{1 + r} \right)^{S}}{1 - \left( \frac{1}{1 + r} \right)^{S}}
\]

Utility in period \( o \) is given by

\[
\sum_{t=0}^{S} \beta^t u(c_t) = \Phi_0 u(c_0)
\]

Note that now, the problem can be perfectly embedded in the two-period model described in Section 2.3. Households solve the following two-period problem:

\[
\max_{c_0, c_S} \quad u(c_0) + \tilde{\beta} u(c_S), \quad s.t.
\]

\[
c_S \Phi_S = a + w \theta (1 + z_j) \Phi_y^o
\]

\[
c_0 \Phi_0 + (P_h h + P_l l) \Phi_y^y + \frac{a}{1 + r^S} = w \Phi_y^y (1 - l)(1 - h) + b
\]

\[
a \geq -\bar{A}
\]
where:

\[
\hat{\beta} = \frac{\beta S}{\Phi_0}
\]

These two budget constraints can be rewritten as a single lifetime budget constraint:

\[
c_0 \Phi_0 + (P_h h + P_l l) \Phi_y r + \frac{c_s \Phi_s}{(1 + r)^S} = w \theta \Phi_y r (1 - l)(1 - h) + \frac{w \theta (1 + z_j) \Phi_y}{(1 + r)^S} + b
\]

The unconstrained consumptions are given by:

\[
c_n = \frac{(\beta(1 + r))^{(-S/\sigma)} \left[ w \theta \left( \frac{\Phi_y + (1+r)^S \Phi_y^0}{\Phi_y} \right) + \frac{b(1+r)^S}{\Phi_y} \right]}{1 + (\beta(1+r))^{(-S/\sigma)} \Phi_y (1+r)^S} \]

\[
c_h = \frac{(\beta(1 + r))^{(-S/\sigma)} \left[ w \theta (1 + z_h) \frac{\Phi_y}{\Phi_y} + \frac{b(1+r)^S}{\Phi_y} - \frac{P_h \Phi_y (1+r)^S}{\Phi_y} \right]}{1 + (\beta(1+r))^{(-S/\sigma)} \Phi_y (1+r)^S} \]

\[
c_l = \frac{(\beta(1 + r))^{(-S/\sigma)} \left[ w \theta (1 + z_l) \frac{\Phi_y}{\Phi_y} + \frac{b(1+r)^S}{\Phi_y} - \frac{P_l \Phi_y (1+r)^S}{\Phi_y} \right]}{1 + (\beta(1+r))^{(-S/\sigma)} \Phi_y (1+r)^S} \]
.3 A Practical Guide to Parallelization in Economics

To understand the role that optimization flags play in the performance of the C++ codes, we plot in Figure 24 the run times for a different number of processors with no optimization flags and with three flags (from less aggressive, -O1, to most aggressive, -O3) in the case of C++--OpenMP.

![Graph showing computing times in C++ with different numbers of processors and optimization flags.](image)

Figure 24: Computing time(s) in C++ with different number of processors and optimization flags.
Bibliography
Bibliography


Hurst, E. and Stafford, F. (2004). Home is where equity is: Mortgage refinancing and household consumption. *Journal of Money, Credit and Banking.*


