Rationing & Bayesian Expectations with Application to the Labour Market

DISSERTATION

zur Erlangung des akademischen Grades
Doktor der Wirtschafts- und Sozialwissenschaften
(Dr. rer. pol.)
in der Wissenschaftsdisziplin
Volkswirtschaftslehre

Erstgutachter: Prof. Dr. Carlo Jaeger
Zweitgutachter: Prof. Dr. Wilfried Fuhrmann

eingereicht an der
Fakultät für Wirtschafts- und Sozialwissenschaften
der Universität Potsdam
von

Hannah Förster

Potsdam, 18. Dezember 2006

Universitätsverlag Potsdam, 2007

Universitätsverlag Potsdam, Am Neuen Palais 10, 14469 Potsdam
Fon +49 (0) 331 977 4517 / Fax 4625
e-mail: ubopus@uni-potsdam.de
http://info.ub.uni-potsdam.de/verlag.htm

Elektronisch veröffentlicht auf dem
Publikationsserver der Universität Potsdam:
http://pub.ub.uni-potsdam.de/volltexte/2007/1521/
urn:nbn:de:kobv:517-opus-15217
[http://nbn-resolving.de/urn:nbn:de:kobv:517-opus-15217]

thank you

Contents

List of Figures

List of Tables

List of Abbreviations

ACE	Agent-Based Computational Economics
FOP	First-order Probability
PDF	Probability Density Function
PIK	Potsdam Institute for Climate Impact Research
PPE	Perceived Probability of Obtaining Employment Conditional on Application
PPERand	Perceived Probability of Obtaining Employment Conditional on Application under Random Rationing
PPERank	Perceived Probability of Obtaining Employment Conditional on Application under Ranking Rationing
PPEPigeon	Perceived Probability of Obtaining Employment Conditional to Application under Pigeonhole Rationing
SOP	Second-order Probability

List of Variables

History of Thought

c	consumption demand
C	aggregated consumption demand
$\tilde{C}$	aggregated effective consumption demand
d	demand
d^*	purchases
D	aggregated demand
D^*	aggregated purchases
$\bar{d}$	demand constraint
G	Government demand
i, j, h, k	indices
l^s	labour supply
$\bar{l}$	labour market constraint
L^s	aggregated labour supply
L^d	aggregated labour demand
L	aggregated actual labour
$\tilde{L}^d$	aggregated effective labour demand
$\tilde{L}^s$	aggregated effective labour supply
$\bar{L}$	bound on labour
m	money holdings
M	aggregated money holdings
$\tilde{M}$	aggregated effective money demand
$\mathbf{p}$	price vector
p	price
P	price in macro model
q	output quantity
r	costs
s	supply

$\bar{s}$	supply constraint
s^*	sales
S	aggregated supply
S^*	aggregated sales
T	tax
u	utility
w	nominal wage
x	consumption demand
$\mathbf{x}$	vector of preferences for commodities
$\bar{x}$	bound on consumption demand
y	income
Y	aggregated output
$\bar{Y}$	bound on aggregated output
z	effective demand
π	profits
ω	commodity holdings

Methods

a	amount of a commodity to be allocated
H	hypothesis
m	random variable
n	number of agents
$\mathbf{N}$	finite set of demanders
$P(.)$	probability
R	preference preordering
$\mathbf{S}$	set of all continuous preference preorderings
x, y, z	units of a commodity
x^*	peak of preferences
θ	event
μ, λ	share
π	permutations of population
ϕ	rationing method
$\mathbf{\Omega}$	sample space
ω	elementary event

A Rationing Toolbox in Mathematica

$< \ldots >$	partially ordered set
$\lvert < \ldots > \rvert$	cardinality of partially ordered set
a	amount of commodity to be allocated
i, j, k, l, t	indices
$\mathbf{d}$	ordered set with demands
D	sum of demands
$\mathbf{f}$	ordered set of realised plans
I	set of indices
$\mathbf{s}$	ordered set with supplies
S	sum of supplies
$\mathbf{e}$	totally ordered set with plans from excess market side
b	number of elements from a set that are unequal to zero
$\mathbf{o}$	ordered set with original plans from excess market side
$\mathbf{p}$	ordered set with plans that are going to be served (random uniform rationing)
$\mathbf{q}$	ordered set with transactions from short market side
$\mathbf{v}$	ordered set with (actual) productivities
$\mathbf{w}$	ordered set of productivity indicators as employer perceives
x	set of state variables
$\mathbf{X}$	state space
$\mathbf{z}$	ordered set with elements that are position indicators
$\mathbf{y}$	ordered set with transactions on excess side of market
ϕ	mapping
ω	random variable

Applications and Simulations

a	productivity
$\mathbf{A}$	set of productivities
$\mathbf{AP}$	set of applicants' productivities
$\bar{a}$	upper bound of pigeonhole
$\underline{a}$	lower bound of pigeonhole
b	being more productive than the observed agent
c_i	characteristic of an agent i relative to the observed agent; i can be l, b, s
cp	critical probability
$F(a)$	probability density function of productivities
$\mathbf{H}$	set of hypotheses
I	specific index during rationing process
k_θ	number of competitors in a pool a
l	being less productive than the observed agent
$\tilde{m}$	random variable
n	number of agents in population $n > 1$ (= number of pools a)
n_l	number of agents with less productivity than the observed agent
n_s	number of agents with the same productivity as the observed agent
n_b	number of agents with higher productivity than the observed agent
$P(.)$	probability
$P(d\|b)$	conditional probability of d given b is true
$p(.)$	probability density
ppe	perceived probability of obtaining employment
r	number of relative characteristics, $r = 3$
R	rationing method
s	being the same productive as the observed agent
s_j	second order probability of hypothesis j
st_θ	number of states in pool θ
v	number of vacancies
$\mathbf{W}$	set of employees
X	state space
z_θ	a state in pool θ
δ	pigeonhole size

θ	pool size
Θ	sample space of pool sizes θ
$\#(z_\theta)$	number of instances of state z in pool size θ
ω	random event
Ω	sample space of random events

Abstract

The first goal of the present work focuses on the need for different rationing methods of the The Global Change and Financial Transition (GFT) working group at the Potsdam Institute for Climate Impact Research (PIK): I provide a toolbox which contains a variety of rationing methods to be applied to micro-economic disequilibrium models of the *lagom* model family. This toolbox consists of well known rationing methods, and of rationing methods provided specifically for *lagom*. To ensure an easy application the toolbox is constructed in modular fashion.

The second goal of the present work is to present a micro-economic labour market where heterogenous labour suppliers experience consecutive job opportunities and need to decide whether to apply for employment. The labour suppliers are heterogenous with respect to their qualifications and their beliefs about the application behaviour of their competitors. They learn simultaneously – in Bayesian fashion – about their individual perceived probability to obtain employment conditional on application (PPE) by observing each others' application behaviour over a cycle of job opportunities.

Zusammenfassung

In vorliegender Arbeit beschäftige ich mich mit zwei Dingen. Zum einen entwickle ich eine Modellierungstoolbox, die verschiedene Rationierungsmethoden enthält. Diese Rationierungsmethoden sind entweder aus der Literatur bekannt, oder wurden speziell für die *lagom* Modellfamilie entwickelt.

Zum anderen zeige ich, dass man mit Hilfe von Rationierungsmethoden aus der Modellierungstoolbox einen fiktiven Arbeitsmarkt modellieren kann. Auf diesem agieren arbeitssuchende Agenten, die heterogen im Bezug auf ihre Qualifikation und ihre Vorstellungen über das Bewerbungsverhalten ihrer Konkurrenten sind. Sie erfahren aufeinanderfolgende Jobangebote und beobachten das Bewerbungsverhalten ihrer Konkurrenten, um in Bayesianischer Weise über ihre individuelle Wahrscheinlichkeit eine Stelle zu erhalten zu lernen.

Chapter 1

Introduction

> What are the basic ingredients that we require from an alternative view? (...) First, we would like to model the economy as a system in which there is *direct interaction* among individuals. We would like to specify agents who, in a sense, have local as opposed to global knowledge. It may well be the case that they have a limited, even wrong, view of the world. Second, we should require that agents behave in a "reasonable" but not "optimal" way; for example, they may use simple rules and they should not act against their own interest. Moreover, these reasonable agents should evolve in the sense that they learn from previous experience. Third, the system should function over time but without necessarily converging to any particular state.
>
> Kirman (2006, p. xiv)

1.1 Setting the Stage

The Global Change and Financial Transition (GFT) working group at the Potsdam Institute for Climate Impact Research (PIK) constructs *lagom*[1], a model family which addresses macro-economic and micro-economic issues. Some of the micro-economic branches of the *lagom* model family are disequilibrium models. Trading at disequilibrium prices is allowed and can lead to mismatched demands and supplies. To generate transactions from inconsistent demands and supplies, a market mechanism is needed, such as a rationing method. Inconsistent demands and supplies may occur on any market, and markets may differ greatly from each other. Therefore, providing a single rationing method is not sufficient.

The first goal of the present work focuses on this need for different rationing methods and provides a toolbox which contains a variety of such methods. These methods are suitable for micro-economic models as mentioned above, but also applicable to ACE (Agent-based Computational Economics) models with many interacting agents. Tesfatsion (2006, pp. 192) describes such a model and where rationing takes place. The rationing toolbox will be provided specifically for the *lagom* context, but is general enough to be applied in different contexts. It consists of well known rationing methods, and of rationing methods programmed specifically for *lagom*. To ensure an easy application the toolbox is constructed in modular fashion.

The second goal of the present work is embedded into the framework of modelling micro-economic labour markets. I will present a labour market model which is on the one hand of a structure that allows for an analysis of the underlying mathematics, and, on the other hand, is also a multi-agent model fitting the characteristics suggested by Kirman (2006, p. xiv)[2] and

[1] Lagom is a Swedish word denoting a sense of balance and harmony, with a flavour of equilibrium but richer, perhaps akin to the chinese "Tao".

Haas and Jaeger (2005, p. 4)

[2] see quote on previous page

further specified by Axtell (2006, pp. 203) in that it allows for the modelling of very large populations of agents, presumed the computational power is available.

Usually, the literature dealing with micro-economic labour markets assumes that (a) an agent who seeks employment applies for a job if the offered wage exceeds the agent's reference wage, and that (b) an employer who faces a pool of applicants can not distinguish between them according to their qualifications. The reference wage is assumed to be correlated positively with an agent's qualification. Mookherjee (1987), and Weiss (1976, 1980) model such labour markets. By increasing the wage, an employer can increase the number of qualified applicants, but still faces an anonymous pool of applicants.

On many labour markets the wage tariffs are negotiated between trade unions and federations of employers, and many employers screen their applicants regarding their qualifications and choose their workforce accordingly. Such a context of non-negotiable wages and recruitment according to the perception of qualifications is the framework to be investigated. On a micro-economic labour market on which wages are non-negotiable, a heterogenous population of potential applicants is faced with consecutive job opportunities. Each job opportunity consists of a certain number of vacancies, and each agent needs to decide at each job opportunity whether to apply. While in Mortensen and Pissarides (1994), heterogenous productivity is modelled by introducing productivity shocks, in the current model heterogenous productivity is depicted by letting agents exhibit heterogenous qualifications. The employer is able to perceive the qualifications of agents more or less accurately; and conducts recruitment with respect to one of three rationing methods, each characterised by a different level of accuracy in perception of applicants' qualifications. The method that the employer chooses to recruit the workforce is common knowledge. An agent's incentive to apply for employment is not the wage but the perceived probability to obtain employment conditional on application (PPE). Only if an agent's PPE exceeds a critical probability will he apply. The population of agents is heterogenous with respect to their qualifications and with respect to their opinions about their competitors' application behaviour. Furthermore, each agent is capable of learning by observing how the other agents act.

The questions to be tackled within this framework are the following:

1. How can agents determine their individual PPE under different rationing methods?

2. How can one model heterogenous labour suppliers who must make application decisions, and who learn about the application behaviour of others by observing them?

3. Is it possible that an employer in the given framework can, though inaccurate in the perception of applicants' qualifications, obtain an equally productive workforce as if he were accurate?

1.2 Structure

Chapter 2 consists of two parts. The first provides a brief history of thought of disequilibrium theory and quantity rationing. A few authors whose contributions play a direct or indirect role for the modelling part of this work will be mentioned in this context. The second part introduces important concepts, namely: A Walrasian model, the dual decision hypothesis, an economy with price and quantity signals, and a neo-Keynesian model. It will be pointed out how and where the concepts introduced previously are of relevance during the presentation of a neo-Keynesian model.

Chapter 3 gives an insight into the methods used in subsequent parts of this work. Rationing axioms are introduced, followed by several relevant rationing methods. Bayesian inference is introduced, a method that is applied to a learning process of agents on a labour market in Chapter 5. A short introduction to SimEnv follows, a simulation environment focusing on evaluation and usage of models with large and multi-dimensional output (cf. Flechsig, Böhm, Nocke, and Rachimov (2005)), as this software is used to obtain information about the stability of the simulations in Chapter

5.

In Chapter 4, the rationing toolbox for the *lagom* model family is presented and documented economically and mathematically. Computational documentation is included in Appendix A. The toolbox contains the rationing methods which have been introduced in Chapter 3, and additional methods that have been programmed specifically for the *lagom* context.

In Chapter 5, it is derived how an agent can determine his PPE under different rationing methods. Then a labour market is modelled where heterogenous Bayesian agents learn about their competitors application behaviour over a cycle of job opportunities. Finally, a complete sequence of recruitment processes is modelled. Over the cycle of job opportunities, the Bayesian labour suppliers determine their PPE, decide whether to apply, and are recruited. For varying degrees of accuracy in an employers perception of applicants' qualifications the average productivity of the workforce is determined. To obtain information about the stability of the results, a sensitivity analysis with the help of SimEnv is carried out.

In Chapter 6 the key points and the results are summarised. The thesis is concluded with proposals for further extensions of the work.

Appendix A contains the computational documentation of the rationing toolbox from Chapter 4. Appendix B contains additional figures to those included in Chapter 5, demonstrating the stability of the results. It also contains the documented code of the programs that have been created for Chapter 5.

Chapter 2
History of Thought

The following history of thought regarding quantity rationing shows how the necessity for rationing methods emerged from a rich history of general equilibrium modelling. A short summary of important contributions to the emergence of a theory of disequilibrium is given. Then a Walrasian model is described, standing for general equilibrium theory, followed by concepts that define the basic elements for a theory that allows for equilibrium under quantity rationing. These concepts are the dual decision hypothesis and an economy that allows for price and quantity signals. Finally, a neo-Keynesian model that combines the elements of these concepts will be introduced and I point out where rationing takes place and what the underlying concepts that allow for such a model framework are.

2.1 History of Thought on Quantity Rationing

2.1.1 On Instantaneous Revision & Rationing

The models and concepts that will be described in this chapter all share one crucial characteristic: decisions of agents are revised instantaneously, which in this context means within the current period. In a Walrasian model, prices are announced, demands and supplies are expressed, and, if necessary, prices are adjusted by the Walrasian auctioneer until the equilibrium price vector has been found. It is assumed that all this happens instantaneously. In many neo-Keynesian models, the microeconomic foundations of these models, and their predecessors, prices and wages are assumed to be fixed for one period, during which no price adjustment can take place. The adjustment must now happen through an alternative channel, which in this case is the quantities traded. At the fixed prices and wages, agents express their wishes on all the markets and receive quantity signals on each market where they face rationing. Rationing will influence an agent's behaviour on the other markets so that he will revise his decisions there, and this might lead agents on the opposite market side to revise some of their plans, too. This quantity tâtonnement takes place until a situation is obtained where transactions can occur. In contrast to a Walrasian model (see below and Section 2.2.2) the agents do not realise their original demands and supplies, but revised versions of these. All this happens in an instant, too.

A rationing scheme is the quantity signal that rationed agents receive and that informs them about the maximum quantity they can buy or sell. These quantity signals are functions of demands and supplies of the other agents (cf. Benassy (2002, p. 12)).

2.1.2 On General Equilibrium Theory

Walras

Walras (1877) formulated the first general equilibrium model, the basic assumptions and functionality of which will be explored in Section 2.2.2. In the Walrasian world, the economy consists of commodities, a price system and agents. The price system consists of a price for each commodity. The agents demand or supply commodities and may be a demander of one commodity and a supplier of another. The price system is exogenously given and no agent can exert influence on it. Walras (1877) described the state of an economy by means of a price system which can give rise to three different situations for each market: aggregate excess demand, aggregate excess supply, or equilibrium. As, generally, only few of the infinite possible price systems will produce an equilibrium, some price adjustment will need to take place to reach equilibrium. This adjustment process is coordinated by an invisible institution often called the *Walrasian auctioneer*.

The agents of the economy, taking the prices as given, express demand and supply at these prices. Whatever quantities of the available goods they plan to buy or sell, they assume to be able to realise. The auctioneer announces a price system, compares aggregate demands and supplies at these prices and adjusts the system until aggregate demand and aggregate supply match on all markets. The price system that gives rise to equated aggregate demands and supplies for all commodities is called an equilibrium price system. Only with this price system will trade occur.

An important implication that follows directly from this outline is the following: The ex ante demands and supplies of agents are always fulfilled because all agents are price takers, assume that they can realise their demand and supply, and since trade only occurs at equilibrium prices the original demands and supplies are always satisfied. Walrasian demand and supply are also referred to as notional demand and supply (see Benassy (2002), Muellbauer and Portes (1978)).

Arrow & Debreu

Walras (1877) described an competitive economy by mathematical means, by stating the conditions for a general equilibrium, but never gave a proof that the system of equations that he constructed had a solution (cf. Arrow and Debreu (1954, p. 254), Debreu (1959, p. ix)). Rather he thought that proof of existence of a general equilibrium was to ensure that there were as many equations as unknowns (cf. Blaug (1996, p. 552)).

Since, to that date, the proofs for the existence of an equilibrium had not been demonstrated for an integrated model of exchange and production[1] Arrow and Debreu (1954) studied the assumptions needed so that an integrated model of a competitive economy would establish equilibrium.

Their model suggests that when assumptions are made that fulfill certain conditions (such as perfect competition, convexity, demand independence), there would be a system of prices so that aggregate demands and supplies for all commodities equilibrate simultaneously. Furthermore, such a system of prices would then exist for any future time period. Arrow and Debreu (1954) derive two theorems which state the conditions under which a comptetive equilibrium would exist. These two theorems can be stated basically as follows (cf. Arrow and Debreu (1954, p. 266)):

1. If each individual in the economy is initially endowed with a positive amount of each commodity that is available for sale, then a competitive equilibrium exists, and
2. the existence of the competitive equilibrium can be asserted if there are several types of labour that exhibit the following properties:
   - ⋆ Each agent is able to supply a positive amount of at least one type of labour, and
   - ⋆ Each type of labour is useful in the production of a desired good.

Debreu (1959) followed up on the ideas stated in this model and gave an axiomatic analysis of economic equilibrium. In his general equilibrium model, the agents are again price-takers. They are, furthermore, consumers and producers. A number of commodities exist, differentiated by physical characteristics, the time and the location of trade, and for that reason each supplier of a commodity acts as a local monopolist. Each agent in the economy has an admissible set of actions, determined by his budget set if he

[1] Wald (1936) gave separate proof of the existence of general equilibrium for a model of production and a model of exchange.

acts as a consumer or determined by his technology set if he acts as a producer. Following his objectives, he chooses the optimum action within this set.

For a consumer, this action is the choice of a consumption bundle which consists of non-negative quantities of all commodities. The agent is restricted by limitations such as his wealth. Furthermore, consumers have preferences regarding commodity bundles. These are defined via a preference pre-ordering for any two commodity bundles, which specifies either the agent's preference for one, his indifference between the two, or the impossibility of comparison. The *goal* of the consumer is to maximise his utility through the consumption of commodity bundles. He chooses the commodity bundle which – according to his preference pre-ordering – yields the greatest utility. Each of the consumers is equipped with an initial endowment of commodities and with a share of the profits from production. In interaction with the price system, this constitutes the income (determined by the share of profits and the value of endowment) of a consumer.

A producer maximises his profit by choosing a commodity bundle from his production possibility set. It is important that he should know the market value of such a commodity bundle determined by the price system.

It is assumed that there is a market for each commodity where the supplies and demands meet, and, under an equilibrium price system, quantities are exchanged. The economic system consists of as many markets as there are commodities and is assumed to be a system of complete markets. With the price system given and the agents being price-takers, it is possible to determine the excess demand for the economic system. By subtracting, from the aggregate supply of each commodity the aggregate demand, the excess demand for each commodity is determined. The sum of excess demands of the commodities is the aggregate excess demand of the system. The state of the economy can be described by the price system and the optimum actions of each agent. If the optimum actions of the agents are not unique under the given price system, then the relation of the price system to all possible resulting total excess demands is called *excess demand relation* (cf. Herings (1996)). If the actions are unique, then the excess demand relation is called *total excess demand function*. It might be that the optimum actions of the agents are not compatible for a given price system and therefore total excess demand is not equal to zero. Because trade happens only under an equilibrium price system, as in Walras (1877), some price adjustment is needed.

A price adjustment mechanism, like the Walrasian acutioneer, generates equilibrium prices and brings the system into a Walrasian equilibrium, where all agents realise their notional plans.

Gaps in the Walrasian General Equilibrium Theory

According to Benassy (1982), the Walrasian theory is a good depiction of real markets if these are actually organised by an auctioneer, but not all markets are organised in this way. This gap was one of the starting points for a theory that deals with potentially non-clearing markets. Benassy (1982) identified two shortcomings that open paths for a new theory and these are:

1. All the agents receive the same price signals but none of the agents sends price signals to the markets. Prices are exclusively influenced by the Walrasian auctioneer.
2. All the agents send quantity signals to the markets but none of the agents makes use of such quantity signals.

Various authors have identified these issues and modified them (allowing for price setters and the use of quantity signals) for the modelling of an economy with incomplete markets and quantity rationing.

2.1.3 On Disequilibrium Theory

Keynes, Clower & Leijonhufvud

Keynes (1936) is central to the development of a theory that allows for the treatment of persistent disequilibrium situations. One of the first authors who interpreted Keynes (1936) in terms of a theory of disequilibrium was Clower (1965). He agreed (cf. Backhouse and Boianovsky (2003, p. 7)) with Axel Leijonhufvud, who suggested that the general difference between Keynes (1936) and the Classics was in the speeds of adjustments of prices and quantities (cf. Felderer and Homburg (2005, p. 278/9)):

> *In the Keynesian macrosystem the Marshallian ranking of price- and quantity-adjustment speeds is reversed*: In the shortest period flow quantities are freely variable, but one or more prices are given, and the admissible range of variation for the rest of the prices is thereby limited.
>
> Leijonhufvud (1968, p. 52)

In 1974, Leijonhuvfud refracted his interpretation and wrote that

> (...) it is not correct to attribute to Keynes a general reversal of the Marshallian ranking of relative price and quantity adjustment velocities.
>
> Leijonhufvud (1974, p. 169)

Nevertheless, the ideas of Clower and Leijonhuvfud remained of strong interest from an analytical point of view and many publications implemented these ideas (cf. Felderer and Homburg (2005, p. 279)). Clower (1965) also had a strong impact on subsequent developments with his formulation of the dual decision hypothesis, which stated that economic agents decide step-wise by taking into account possible constraints on various markets (e.g. excess labour supply, excess commodity demand). Given a constraint, the economic agent would behave differently than if there had been no constraint. Clower (1965) contended that Keynes (1936) also had such a dual decision hypothesis in mind, if only implicitly:

> It is another question whether Keynes can reasonably be considered to have had a dual decision theory of household behaviour at the back of his mind when he wrote the General Theory. For my part, I do not think there can be any serious doubt that he

> did, although I can find no direct evidence in any of his writings to show that he ever thought explicitly in these terms. But indirect evidence is available in almost unlimited quantity (...).
>
> Clower (1965, p. 120)

Clower and Leijonhufvud proposed a Walrasian-Keynesian synthesis by suggesting that instead of pursuing *unemployment equilibrium* with imperfect markets, one should analyse a *prolonged disequilibrium* without ad hoc rigidities (cf. cepa (2006b))[2].

Patinkin

Before Clower and Leijonhufvud, the synthesis of Walrasian and Keynesian theory had already been proposed by Patinkin (1956) whose central concern was the integration of monetary theory and Walrasian value theory (cf. cepa (2006a)). He modified the classical Walrasian approach in such a way as to be able to describe the transmission process of money quantity adjustments in unchanged equilibrium situations; this approach became the basis for a number of disequilibrium models. He created a general equilibrium style model that, under flexible prices and wages, had a tendency to arrive at equilibrium (cf. Rothschild (1981, p. 33)).

This approach was used as the basis to analyse the phenomenon of persistent involuntary unemployment that was central to Keynes (1936). He introduced the notions *voluntary* and *involuntary unemployment*. An agent was considered *involuntarily* unemployed if he offered labour on the market but could not find a job, whereas an agent was considered *voluntarily* unemployed if he withdrew labour from the market due to a wage decrease. Even if this withdrawal was painful, this kind of unemployment was categorised as voluntary. The basic model would, with flexible prices and wages that adjusted immediately, converge towards equilibrium and agents would realise their ex ante plans voluntarily, as no bounds would

[2] Neo-Keynesian and neo-Keynesianism are used as in Felderer and Homburg (2005) referring to the works by Patinkin (1956), Clower (1965), Leijonhufvud (1968), Barro and Grossman (1976), Malinvaud (1977), and others, which are characterised by temporary equilibria with quantity rationing.

be experienced. With rigid prices and wages, underemployment resulted which, if these rigidities were not eliminated, became persistent. Then, the short side of the labour market determined the employment level and rationing occurred. Patinkin (1956) assumed that, even with flexible prices and wages, price/wage adjustments would not happen immediately, but rather quantity adjustment would take place. If the demand for commodities decreased, firms that found they could not sell all they wished would decrease their demand for labour (cf. Backhouse and Boianovsky (2003, p. 4)), resulting in unemployment. Only thereafter would prices and wages adjust, but not sufficiently to eliminate unemployment. A detailed exposition can be found in Patinkin (1956, pp. 313) and in Rothschild (1981, pp. 32).

From Clower Onwards

Clower (1965) was one of the first authors who reinterpreted Keynes (1936) in a way that provided a basis for a theory which was not characterised by the equilibrium as in Walras (1877) and Arrow and Debreu (1954), but rather by an equilibrium which might leave some agents with unsatisfied wishes. This equilibrium with quantity rationing was the result of an imperfect price adjustment process and was in contrast to what had been assumed in Walras (1877) and Arrow and Debreu (1954):

> In a general economic equilibrium the price system communicates sufficient information to allow producers and consumers to coordinate their separate production and consumption decisions. Prices adjust to bring supply and demand into balance.
>
> Starr (2001, p. 31)

Rather, the assumption that prices adjusted fast enough to let the economic system arrive at equilibrium was dismissed, and, in exchange, quantities were adjusted. Nearly all models that were introduced by neo-Keynesian authors in the 1970s such as (Bénassy(1975), (1977)), Barro and Grossman (1976), Malinvaud (1977), Hahn (1978), and Muellbauer and Portes (1978) discussed general equilibrium with quantity rationing. The once common assumption that agents could trade whatever quantity they wished was abandoned and substituted by the distinction between effective demands and effective supplies on the one hand and purchases and sales (transactions) on the other. This allowed the modelling of imperfect markets where aggregate transactions must be equal, but effective demands and supplies need not be equal.

Malinvaud

Malinvaud (1977) created a macroeconomic model in which he viewed general equilibrium as a temporary situation that itself was a special case. His model resembled those created by Patinkin (1956), Clower (1965), and Barro and Grossman (1976) but focused specifically on the investigation of the labour market and unemployment. In contrast to a Walrasian model, no auctioneer adjusted prices, and prices were fixed for one period. In order for trades to take place, an alternative adjustment had to be implemented through a quantity tâtonnement. One of the purposes of the model was to combine the two distinct views of classical models and Keynesian models regarding unemployment. While unemployment in classical models was due to too high wages and could only be reduced by lowering the wages, Keynesian unemployment was characterised by firms facing too little demand. These firms would hire more employers at the current wage, but they would not be able to sell the additional output. The policy measure to reduce unemployment would be a rise in effective demand through the government (cf. Rothschild (1981, p. 95)). Classical unemployment was, in view of Malinvaud (1977), just a partial analysis, but, from a general equilibrium point of view, the labour market was connected with other markets in such a way that rationing on the labour market impacted the goods market and vice versa. Therefore, it could not be the wage alone that determined the level of unemployment: it was necessary to take the interdependence with other markets into account. Since his model was not a traditional Walrasian model, demands (supplies) did not have to be equal to purchases (sales). The demands and supplies of agents were an attempt to realise their plans. Since these plans were not independent from what happened on other markets, agents formulating them took into account whether they faced rationing on other markets. If they faced rationing on the labour market, they would revise demand for goods. Analogous revisions took place when agents experienced rationing on the goods market. A process of quantity revision took place until the wishes of the individuals coincided and trade could take place. The nature of this revision process depended on the initial situation, which could be one of three different regimes that were characterised by different combinations of rationing as will be shown in Section 2.2.5.

2.1.4 On Rationing

The models that have been introduced in the preceding sections have in common that trade can take place at non market-clearing prices. If trading at non market-clearing prices takes place, rationing necessarily also takes place, and one of the market sides will not be able to realise their plans. Clower (1965) was one of the first to state general ideas about a rationing method and what characteristics it should have (on the macro level). He formulated the *short-side rule* that was later specified in more detail by Benassy (1982) and others. On the macro level, the short-side rule identifies the market side that realises its planned transactions.

> Under this rule, agents on the short side of the market realize their desired transactions, whereas agents on the long side are rationed (how this is done at the microeconomic level is usually not specified).
>
> Benassy (1986, p. 13)

Since I will be aiming to provide a toolbox that contains rationing methods to be applied to microeconomic modelling, it is necessary to investigate how rationing at the micro level can be specified. While at the macro level the short-side rule determines which market side will realise its desired transactions, at the micro level the question is to find a rule that specifies how the given quantities are to be distributed to the agents. A rationing method is such a rule and in the 1990s, several authors introduced different rationing methods. Sprumont (1991) proposed general axioms that a rationing method should ideally fulfill, deriving from them the uniform allocation rule as the only method that does so. By dismissing one or more of these axioms, a variety of other rationing rules can be constructed. Moulin (1999) stated ideas about a rationing method when preferences of agents are single-peaked[3]. Barbera and Jackson (1995) considered strategy-proof[4] allocation methods, Tasnádi (2002) introduced stochastic rationing methods that are based on the rationing methods derived by Moulin (1999, 2000). Since rationing methods as such play a crucial role in subsequent chapters, before implementing them into an economic context, an axiomatic approach to rationing methods and basic rationing methods will be discussed in Chapter 3.

In Section 2.2 I will first introduce a general equilibrium model and then present building blocks for a neo-Keynesian disequilibrium model.

[3]With a single peak there is only one allocation that an agent finds optimal, even if there are allocations associated with a larger value.

[4]Strategy-proofness implies that an agent facing rationing is best off by stating his true preferences. He is not able to receive more by stating incorrect preferences than he would by stating his true preferences.

2.2 A General Equilibrium Model, and Building Blocks of a Disequilibrium Model

2.2.1 Introduction

Drawing from Dixon (2000a) and Gravelle and Rees (1992), a Walrasian model will be specified in order to point out its weaknesses with respect to price and quantity signals. When rationing does occur, it will be demonstrated how the dual decision hypothesis enables the modelling of step-wise decision on the consumer side. Subsequently, an economy with price and quantity signals will be introduced, where step-wise decision making takes place. Finally, a neo-Keynesian model will be described and the different arrangements of rationing that can emerge will be pointed out.

2.2.2 The Walrasian Economy

Walras (1877) was the first to express a general equilibrium model mathematically. His central vision was an economy that consists of inter-connected markets where events on one market influence what happens on others (cf. Staley (1989, p. 170)). All prices are assumed to be variable and equilibrium requires all markets to clear simultaneously. This was the first model to consider market interdependencies. The multi-market model that Walras (1877) laid out is a pure exchange model where all agents are equipped with an initial commodity bundle which they can use for exchange. Each agent takes the market prices as independent from his actions (cf. Varian (1984, p. 191)) and is thus a price taker. The tastes of each agent regarding commodities $\mathbf{x}$ is described as a utility function u

$$u = u(\mathbf{x}) \tag{2.1}$$

The Walrasian economy consists of m agents, who trade n commodities on markets. An exogenously given price system defines a price p_i for each of the $i = 1, ..., n$ commodities, such that the price system can be expressed as a vector of prices $\mathbf{p} = (p_1, p_2, ..., p_n)$. Demands and supplies of a commodity x_i are functions of its price p_i. An agent can be demander and supplier of various commodities at the same time. He demands commodities that he wishes to own and offers commodities that he owns but wants to dispose of. His decision of how many units of which commodity to offer for sale or to demand is determined before trade takes place by maximising his individual utility subject to the budget constraint. The planned demand (supply) of agent j for commodity i is expressed as d_{ij} (s_{ij}). The maximisation problem then is:

$$\begin{aligned} &\max_x u_j(\mathbf{x}) \\ &\quad \text{s.t. } \textstyle\sum_{i=1}^{n} p_i d_{ij} = \sum_{i=1}^{n} p_i s_{ij} \end{aligned} \tag{2.2}$$

The second line of 2.2 expresses the budget-constraint. Walras (1877) established a system with n equations (one equation for each commodity) and n unknowns. The prices are the unknowns, and the solution of the system is a price vector $\mathbf{p}^*$ that equilibrates demand and supply for each commodity. The monetary value of a commodity is given by its price p_i. Before trade takes place, a price vector is stated by an auctioneer and for these prices, each agent conducts his utility maximisation and presents the resulting demands and supplies to the markets. In any situation – even if the currently stated market prices are not the equilibrium prices – the monetary value of all commodities an individual wants to buy must be equal to the monetary value of all commodities he offers for sale, i.e. the budget constraint from 2.2 must be met.

In equilibrium, the value of the quantities demanded by all individuals must be equal to the value of the quantities offered for sale by all individuals. Considering the planned demands and supplies of all m agents leads to:

$$\sum_{j=1}^{m}\sum_{i=1}^{n} p_i d_{ij} = \sum_{j=1}^{m}\sum_{i=1}^{n} p_i s_{ij} \tag{2.3}$$

which can be rearranged to give

$$\sum_{i=1}^{n} p_i (\sum_{j=1}^{m} d_{ij}) = \sum_{i=1}^{n} p_i (\sum_{j=1}^{m} s_{ij}) \tag{2.4}$$

The expressions in parentheses expresses the total market demands and supplies for the ith commodity and can be rewritten as d_i and s_i since they are the sum of the individual demands and supplies of that commodity.

$$\sum_{i=1}^{n} p_i d_i = \sum_{i=1}^{n} p_i s_i \tag{2.5}$$

Equation 2.5 is Walras's identity (cf. Dixon (2000a)) and states that, in equilibrium, the value of the aggregated planned market purchases must be equal to the aggregate value of all planned market sales. General equilibrium is then defined as a situation of the economy where no excess demand is present on any market (cf. Morishima (1981, p. 16)).

Two implications can be derived from Walras's identity.

1. *Generality of equilibrium:* It is assumed that prices have been found such that demand and supply are equal in all but one market. Then for $n-1$ markets:

$$\sum_{i=1}^{n-1} p_i d_i = \sum_{n=1}^{n-1} p_i s_i \tag{2.6}$$

   If this expression is subtracted from Walras's identity (2.5)

$$p_n d_n = p_n s_n \tag{2.7}$$

   or $d_n = s_n$, which implies that the nth market is also in equilibrium. If all but one market are in equilibrium, then so is the remaining market.

2. *Walras's law:* If one looks at the whole economy, there can be neither aggregate excess supply nor aggregate excess demand. Rearranging (2.5) gives

$$\sum_{i=1}^{n} (p_i d_i - p_i s_i) = 0 \tag{2.8}$$

   Walras's law stems from the budget constraint and expresses the interdependence of excess demand equations of the general equilibrium system (cf. Patinkin (1987, p. 863)). Equation 2.8 expresses that the aggregate excess demand is equal to zero.

   > This implies that, should there ever be an excess of demand over supply for any one commodity, there must be a corresponding excess supply over demand (an excess of supply over demand is also called "negative excess demand")

> for at least one other commodity, otherwise the aggregate value of amounts agents wish to supply could not be equal to the aggregate value of amounts agents wish to demand. Another way to put this, is to say that the sum of excess demands over all the markets in the economy must equal zero and that this applies whether or not all markets are in (general) equilibrium.
>
> Dixon (2000b)

How do prices adjust to equilibrium values and how do transactions occur in such a framework?

The commodities are exchanged among the agents. All agents receive the same price signal which can be expressed as a vector $\mathbf{p}$ with n elements. The agents assume ex ante that they can exchange whatever quantities they wish. As said earlier, demands and supplies are functions of the price, so after having received the price signal, each agent sends his demand and supply quantities, which have been obtained by individual utility maximisation as in 2.2, to the market. If demands and supplies of a commodity do not *match*, adjustments are needed. In a Walrasian economy, price adjustment is accomplished exclusively by the Walrasian auctioneer. This institution changes the price vector $\mathbf{p}$ until an equilibrium price vector $\mathbf{p}^*$ has been found. At these prices, the corresponding demands and supplies of all commodities and all agents coincide and only at these prices do transactions take place. The agents face no quantity constraints since they realise the planned demands and supplies which they had stated at these prices before trade took place. Their ex ante demands and supplies have been, and always will be, fulfilled in such a framework.

> The Walrasian story is a good description for the few real world markets, such as the stock market which inspired Walras, where the equality between demand and supply is ensured institutionally by an actual auctioneer. For all other markets with no auctioneer in attendance, the Walrasian story is clearly incomplete, something pointed out by Arrow(1959) himself.

Benassy (2002, p. 4)

Benassy (2002) stresses two points as being very important characteristics of the Walrasian model. These are:

1. All the agents receive the same price signals, but prices are influenced exclusively by the Walrasian auctioneer.

2. All the agents send quantity signals to the markets but none of the agents makes any use of these quantity signals. E.g. no demander who receives the number of produced items as a quantity signal uses this information to reconsider his demand.

These are gaps to be filled. Benassy (2002) established a consistent theory of decentralised markets where no auctioneer is present and where market clearing is not axiomatically assumed. He also implemented a system in which quantity signals (at least from rationed agents) have to be considered in addition to price signals.

2.2.3 The Dual Decision Hypothesis, Drèze and Clower Demands

Introduction

Clower (1965) formulated the dual decision hypothesis. It demonstrates that the Keynesian consumption function can be validated from a neoclassical point of view and shows that decisions of agents can be modelled to happen step-wise.

The Hypothesis

In the neoclassical framework, consumption demand depends only on the real wage $\frac{w}{p}$. A consumer does not choose consumption with his income in mind but rather with the real wage, and, hence, his income is determined by his choice of labour supply l^s. This can be pointed out by following an example from Felderer and Homburg (2005) which investigates a household that maximises utility. Utility u depends on consumption c and leisure $1 - l^s$ and is expressed by the following utility function:

$$u = u(c, (1 - l^s)) \tag{2.9}$$

Furthermore, the household faces a budget constraint that requires it to finance its consumption from his labour income:

$$p \cdot c - w \cdot l^s = 0 \tag{2.10}$$

where p denotes the price and w the nominal wage. By maximising equation (2.9) subject to (2.10) the neoclassical consumption function and the labour supply function are obtained:

$$c = c(\frac{w}{p}) \tag{2.11}$$

$$l^s = l^s(\frac{w}{p}) \tag{2.12}$$

The income y of the consumer is the product of labour supply and real wage

$$y = l^s \cdot \frac{w}{p} \tag{2.13}$$

Consumption demand expressed by 2.11 depends uniquely on the real wage. This is in contrast with the Keynesian view where the consumption demand of a consumer is determined by his given real income.

Clower (1965) showed that, from a neoclassical point of view, the Keynesian consumption function could be validated. There is one central aspect for this proof. It is the fact that, in the neoclassical view, it is always assumed that the household presumes that it can realise its plans and it is validated ex post that this is true. Clower (1965) modified this and introduced the possibility that a consumer faces a constraint on one of the markets, meaning he will not be able to realise (all of) his plans. This provides a starting point for validating the Keynesian consumption function within a neoclassical framework. If a consumer faces rationing on the labour market, he will not be able work as much as he would like to and will earn a smaller income, leaving less scope for consumption. It is natural to assume that as a reaction, he will revise his consumption demand, the revision being larger the stronger the constraint on the labour market is. This is the central statement of the dual decision hypothesis: Decisions by the consumer are made step-wise.

In a first step, the consumer formulates his notional demands on the basis of price signals and assumes that he will realise these demands. Then, if he finds himself rationed on one market, in a second step he will reformulate his plans for other markets. Additional to the price signals he now takes quantity signals into account by decreasing his consumption demand in consideration of the lower income. With the revised consumption demand he has then optimised his utility by being aware of the lower income due to rationing on the labour market. His demand will be lower than in a classical context and is called *effective demand* or *Clower demand*. The modified problem of a consumer who faces a constraint on the labour market can be expressed as

$$\text{choose } \min(c(\frac{w}{p}), \frac{w}{p} \cdot \bar{l}), \tag{2.14}$$

where $\bar{l}$ is the constraint faced on the labour market. Firstly, the consumer states his hypothetical consumption demand $c(\frac{w}{p})$ and his labour supply

l^s. If he does not face a constraint on the labour market, the process will be complete and he has realised his ex ante plans. If, however, he faces a constraint $\bar{l}$ on the labour market, he will adjust his consumption demand to his given real income ($\frac{w}{p} \cdot \bar{l}$) so that both match.

Clower kept the neoclassical core, the utility maxmimising behaviour of the agent; the aspect he modified was the circumstances influencing the maximising behaviour. While the neoclassical view implicitly assumed the realisation of notional demands, Clower introduced an environment where plans may need to be be revised. If this is the case, not only the real wage, but also the real income will be part of the optimisation problem of a household.

Drèze and Clower Demand

This modified environment for agents makes it necessary to investigate how to describe the behaviour of agents who face rationing on any market. Two different types of behaviour have been identified by the literature. An agent who faces rationing (for example) on the labour market might react in one of the following ways:

1. He decreases labour supply so as to match the rationing level and revises his consumption demand, or
2. He keeps his labour supply constant but revises his consumption demand.

These are called *Drèze demand* and *Clower demand*[5] respectively. The general assumption is that decisions happen step-wise. An agent has formulated his notional demands by maximising utility subject to the usual budget constraint, but if he finds himself rationed on a market,

> (...) then a second round of decision making will take place, namely maximising utility subject to a modified budget constraint,
>
> Clower (1965, p. 119)

[5] I use the term demand here, but if in some cases it actually refers to a supply (e.g. labour).

or subject to the budget constraint and additional constraints (depending on the choice of representation). Clower implemented this by postulating the dual decision hypothesis. Drèze demand includes a modified optimisation problem, too. Both differ from each other in the way they describe the behaviour of an agent who faces rationing on a market. The Clower demand is determined by an optimisation problem which looks like the following[6]:

$$
\begin{aligned}
&\max_{x_i} Z && = f(x) && \\
&\text{s.t.} && && \\
&p \cdot x && \geq 0 && \\
&x_j && \leq \bar{x}_j, && \forall x_j \geq 0, \quad j \neq i \\
&x_j && \geq \bar{x}_j, && \forall x_j \leq 0, \quad j \neq i
\end{aligned}
$$

with:

x_j	consumption demand on market j,
$\bar{x}_j$	bound on consumption on market j,
i	index for the market in that agent faces rationing,
x_i	demand on market i.

When determining demand x_i on market i, the agent only considers bounds experienced by rationing **on the other** markets. He keeps his demand on the market where he faces rationing constant, while he revises his plans on other markets.

If, on the other hand, an agent faces rationing on a market and revises demand **on that** market (and on others) then he states a Drèze demand. The optimisation problem of an agent stating Drèze demand can be expressed

[6] $x_j \leq \bar{x}_j$ refers to a market where the agent supplies a commodity. Originally, his supply is larger than the bound $\bar{x}_j$, so in the optimisation problem the constraint $x_j \leq \bar{x}_j$ expresses that his supply must be equal or less than the bound given.

as:

$$
\begin{aligned}
\max_{x_i} Z \quad & = f(x) \\
\text{s.t.} \quad & \\
p \cdot x \quad & \geq 0 \\
x_j \quad & \leq \bar{x}_j, \qquad && \forall x_j \geq 0 \\
x_j \quad & \geq \bar{x}_j, \qquad && \forall x_j \leq 0
\end{aligned}
$$

The first constraint is a budget constraint. The other two constraints express the fact that the agent considers the bounds experienced through rationing **on all** markets while determining his demand on market i.

Application

Clower and Drèze demands differ in the way agents consider information from markets. While the Drèze demand is generated on the basis of information from all markets, Clower demand is generated by considering information from all *but* the present market. Both types of behaviour are rational in different contexts. The Drèze demand is considered to be rational by Felderer and Homburg (2005) in a model where

- ⋆ the agent knows the degree of rationing without doubts,
- ⋆ the rationing scheme is not manipulable, and
- ⋆ maintaining a demand larger than the bound given by rationing is costly.

Within such a setting, it is plausible that an agent facing rationing will revise his demand to match the bound given by rationing on that market. Clower demand, on the other hand, is considered to be rational in a world where

- ⋆ the agent does not know the degree of rationing for certain,
- ⋆ the agent assumes that the rationing scheme is manipulable, and

⋆ maintaining an excess demand over the quantity that the agent would obtain through rationing is not costly.

In such a setting, the agent maintains his notional demand on the market where he faces rationing. It is rational for him, since he sees the possibility to realise his notional demand or a demand that lies between his notional and his Drèze demand. Since it is not costly to state his notional demand in the market where he is rationed, he does not see the necessity to revise it.

This outline indicates that the choice of a specific rationing method will be a crucial determinant for the agents' actual behaviour. If agents assume that the rationing method contains a stochastic element and is manipulable, then it seems natural to let them send demands to the market that violate their constraints (they want to experiment to find out the actual value of the constraint). Depending on the rationing method, not only can agents' behaviour be modelled in different ways, but also the equilibrium obtained depends on the choice of the rationing method (cf. Drazen (1980, pp. 287)).

2.2.4 An Economy with Price and Quantity Signals

Introduction

In Section 2.2.2 I introduced some facts about the Walrasian economy in a framework of general equilibrium and discussed markets that clear at any point in time and agents who realise in full their ex ante demands and supplies. The goal of this present section is to describe how Bénassy (1982, 1986, 2002) constructs an economy that allows for non-clearing markets and imperfect competition. In such an economy, some demands and/or supplies[7] cannot be realised. There is a need for some kind of rationing method which allocates a given quantity of a commodity to individuals. The goal of this section is to introduce

- ⋆ non-clearing markets,
- ⋆ the formation of quantity signals,
- ⋆ effective demands and supplies, and
- ⋆ price formation.

These concepts will be introduced by restricting attention to a single market, but they are constructed in such a way as to be applicable to the general equilibrium framework.

Non-Clearing Markets

In contrast to a Walrasian economy, the agents of an economy with non-clearing markets may not be able to realise their ex ante demands and supplies. Therefore, demands and supplies need to be distinguished from trades (purchases and sales). Purchases and sales are the quantities that the agents realise. In the following, the focus will be on the agents' behaviour. The demand of agent i on market h is denoted d_{ih} and s_{ih} denotes the supply of agent i on market h. These demands and supplies are signals agents send to the market before an exchange takes place. They are the values representing the quantities that the agents plan to trade, and they are

[7] Now demands and supplies are again distinguished, in contrast to Section 2.2.3

determined by individual utility maximisation. In a non-clearing market framework with quantity signals, this maximisation differs from the one in Walrasian markets, as agents now need to consider possible rationing bounds on the markets (cf. Section 2.2.3). Purchases of agent i on market h are denoted d^*_{ih} and sales of the same agent on the same market as s^*_{ih}. A market process must result in coinciding transactions:

$$D^*_h = \sum_{i=1}^{n} d^*_{ih} = \sum_{i=1}^{n} s^*_{ih} = S^*_h \tag{2.15}$$

Market h is further analysed by assuming that the price p_h for the commodity traded on that market has been stated already. If the price p_h is not the market clearing price then there is a situation of inconsistent demands and supplies

$$D_h = \sum_{i=1}^{n} d_{ih} \neq \sum_{i=1}^{n} s_{ih} = S_h \tag{2.16}$$

If the economy happens to be in such a situation, there has to be some kind of mechanism that generates consistent transactions that fulfill equation (2.15). Since there is a shortage of either supplies or demands, some agents will remain with unsatisfied plans; these agents will face rationing. A framework of non-clearing markets is a framework where demands and supplies must not necessarily be the same as purchases and sales. This is in stark contrast to the Walrasian theory.

Formation of Quantity Signals

At least the rationed agents need to receive quantity signals in order to be able to revise their plans. Consider a simple market with one demander stating demand d_1 and one supplier, stating supply s_2. The *rule of minimum* applies and the transactions are simply the minimum of demand and supply

$$d^*_1 = s^*_2 = min(d_1, s_2) \tag{2.17}$$

As a part of the transaction process, the agents receive from one another a quantity signal which tells them the maximum quantities they can buy/sell. Here, demander 1 is faced with the supply s_2 and knows that he will not be able to purchase more than this. The supplier on the other hand knows that he will not be able to sell more than d_1. The agents can implement this expectation into their maximisation problems as upper bounds on purchases/sales denoted $\overline{d}_1$ and $\overline{s}_2$ respectively. In the present example the bounds are

$$\overline{d}_1 = s_2, \qquad \overline{s}_2 = d_1 \tag{2.18}$$

The rationing scheme can be expressed by determining, for the demander, the minimum of his demand and the bound on demand, and for the seller, the minimum of his supply and the bound on supply

$$d_1^* = \min(d_1, \overline{d}_1) \tag{2.19}$$

$$s_2^* = \min(s_2, \overline{s}_2) \tag{2.20}$$

Under voluntary exchange where no agent can be forced to sell or buy, for a number of individuals acting on one market it is true that

$$d_i^* = min(d_i, \overline{d}_i) \tag{2.21}$$

$$s_i^* = min(s_i, \overline{s}_i) \tag{2.22}$$

Quantity signals are introduced to the market in such a way that agents who face rationing use these signals as information when determining their effective demands and supplies.

Effective Demand and Effective Supply

In order to formulate demands and supplies that consider these quantity signals, it is necessary to implement the information from above. Demands and supplies based on quantity signals differ significantly from those of a Walrasian economy. In contrast to the Walrasian economy, the agents implement the quantity signals they have received into their maximisation problem. In the simple case of deterministic constraints, the effective demands and supplies are the trades maximising the agents' utility/profit subject to the usual Walrasian budget-constraint and to the quantity constraints on the other markets. Regarding the one-market problem from above, the Walrasian optimisation problem for demander i acting on market h can be expressed as:

$$\begin{aligned} &\max_{z_i} u_i(\omega_i + z_i, m_i) \\ &\quad \text{s.t.} \\ &\quad pz_i + m_i = \overline{m}_i \end{aligned} \tag{2.23}$$

with:

$\bar{m}_i$	quantity of money of agent i at the beginning of the period,
m_i	quantity of money of agent i (income),
	(budget constraint for a monetary economy).
z_i	effective demand of agent i,
ω_i	holdings of commodities of agent i,
p	price

Money enters the utility function because it can be used as a store of value and for exchange purposes (cf. Benassy (2002, p. 44)). Implementing the quantity constraints of the other markets yields

$$\begin{aligned} &\max_{z_i} u_i(\omega_i + z_i, m_i) \\ &\quad \text{s.t.} \\ &\quad pz_{ih} + m_i = \overline{m}_i \\ &\quad -\overline{s}_{ik} \leq z_{ik} \leq \overline{d}_{ik}, \qquad k \neq h \end{aligned} \tag{2.24}$$

with:

$\overline{s}_{ik}$	perceived supply constraint of agent i on market k,
$\overline{d}_{ik}$	perceived demand constraint of agent i on market k.

In both cases, an agent solves this problem with respect to z_i. Repeating 2.24 for all markets, one obtains a vector of effective demands of agent i. The maximisation problems 2.23 and 2.24 clearly differ. The effective demand obtained through the optimisation problem of 2.24 is the demand that maximises the transactions for the agent (see Benassy (2002, p. 66/7)). While the Walrasian problem only takes the budget constraint into account, the effective demand problem also considers the perceived supply/demand constraints of the market. These constraints can be results of objective or the results of subjective perception. The latter leaves space for perception mistakes.

Price Formation and Perceived Demand Curves

Traditionally, agents of an economy take prices as given and treat them as parameters in their optimisation problems. They assume that they can exchange whatever they want at the given prices. This assumption is validated ex post since the price is determined by the equality of demand and supply. But if the price setting process is modelled internally then the quantity signals that have been introduced above can be of use to the agents. In the following it is assumed a seller sets the price and a demander takes the price as given, and, thus, trade has to occur at the price set by the seller.

Before trade takes place, the seller makes assumptions (probably based on previous observations) about the maximum demand he will encounter. This quantity influences the price he sets. The relation between the demand he assumes, and the price he sets is called *perceived demand curve* and constitutes a constraint on his sales (supply constraint). After he has stated his price, transactions will occur and he will face the actual constraint $\bar{s}_i$ which is equal to the sum of demands of the other agents

$$\bar{s}_i = \sum_{j \neq i} d_j = D \tag{2.25}$$

Before transactions occur, the seller needs to decide about the price. In contrast to a Walrasian seller, the present seller does explicitly consider his quantity constraint $\bar{s}_i$. He does not treat it parametrically. Rather, he sets the price so as to manipulate this constraint. He tries to steer the demand which will be expressed after he has stated the price. It is assumed that his expectations are deterministic, implying that he thinks he is certain about the demand he will face. His perceived demand $D(p_i)$ can then be expressed as

$$D(p_i) = \bar{s}_i(p_i) \tag{2.26}$$

Depending on the actual knowledge of the seller, this perceived demand will either be perceived correctly or not. If it is anticipated correctly this perceived demand curve will be called *objective demand curve* and looks like this:

$$\bar{s}_i = \bar{s}_i(p_i) = D(p_i) \tag{2.27}$$

On the other hand, if the seller is not fully aware of the demand that he faces, his perceived demand will only to some extent approximate the real demand curve, resulting in a subjective demand curve. Once the seller has generated a perceived demand curve, he decides which price to set. He knows that his sales should not be greater than the value given by the perceived demand function (he is unaware of possible errors in his perception). Similar to the maximisation problem of the demander, the supplier now maximises his objective function with respect to the perceived demand curve. The problem of a seller could then look like this:

$$\begin{aligned} &\max_{q_i, p_i} \quad p_i s_i - r_i(q_i) \\ &\text{s.t.} \\ &s_i \leq q_i \\ &s_i \leq \bar{s}_i(p_i) \end{aligned} \tag{2.28}$$

with:

r_i costs depending on the output, and
q_i output of good i.

The seller will always choose a point that lies on his perceived demand curve. Figure 2.1 illustrates the price choice of a seller.

The seller has the perceived demand curve $D(p_i)$ that depicts his constraint on sales. He chooses a price p_i^*, and produces the quantity q_i^*, on the perceived demand curve at the intersection of supply curve s_i and perceived demand curve. Other prices do not make sense: if he chose a lower price than p_i^*, e.g. $\hat{p}_i$ then he could, by increasing the price up to $\hat{p}_i'$, keep his sales constant and at the same time increase profit.

The solution to his maximisation problem will always draw him to choose a point that lies on the constraint. The seller has expectations about the demand he will be faced with after having stated a price. He makes use of this perceived demand curve by implementing it into his optimisation problem. In this way, he is able to generate a price that optimises his profit under the given supply constraint.

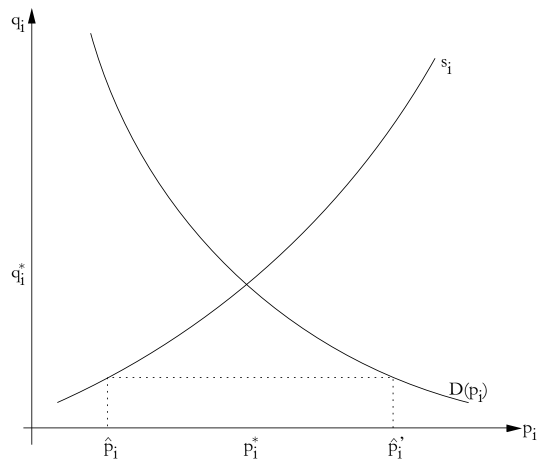

Figure 2.1: Price Choice of a Seller in a Framework with Non-Clearing Markets and Perceived Demand Curves

2.2.5 A Neo-Keynesian Model

Introduction

This section introduces a basic neo-Keynesian model as in Malinvaud (1977), Barro and Grossman (1976), and summarised by Felderer and Homburg (2005). With this model it is possible to identify three regimes, distinguished by the combination of rationing. Since this is a macroeconomic model, it requires some kind of aggregation. For example, an aggregate household is obtained by summing up the individual results regarding labour and consumption. This is done in the fashion of Malinvaud (1977). Then it is possible to account for the possibility that one or more individuals might experience rationing.

> One can (...) study the state of the market in the current period either by postulating that adjustments are made only by price movements (temporary competitive equilibrium), or by assuming that prices are temporarily fixed during the period and that adjustments are made by quantity rationing (temporary equilibrium with quantity rationing).
>
> Grandmont (1977, p. 536)

Here, the latter is the case; prices and wages remain fixed during a period and agents treat them parametrically. Whether the prices are set by the supplier at the beginning of the period by considering a perceived demand curve or whether the prices are given is not discussed. However, prices and wages remain fixed during the period and quantity adjustments are triggered by the quantity signals sent across the market. The model has the following characteristics:

- ⋆ equilibrium loci are formed by effective demands and supplies,
- ⋆ equilibria with quantity rationing,
- ⋆ quantity tâtonnement rather than price tâtonnement,
- ⋆ the circumstances for market clearing can only be defined by taking the other market(s) into account, too.

The Aggregate Household

The aggregate household (subsequently household) is composed of several households that exhibit identical preferences and identical initial money holdings at the micro level. A consumer buys a quantity of the commodity, offers labour on the labour market and has a terminal money holding (cf. (Malinvaud 1977, p. 41)). The household demands consumption goods C, offers labour L^s, saves a fraction of the income $\triangle M_H$, receives income from profits π_0, and pays tax T on its income.

Utility is assumed to depend upon consumption, leisure $(1-L)$[8] and real savings ($\frac{\triangle M_H}{P}$), so that it can be expressed as:

$$U = U\left(C, (1-L), \frac{\triangle M_H}{P}\right) \tag{2.29}$$

The usual assumptions apply: positive first order derivatives and decreasing marginal rate of substitution. The household faces a budget constraint

$$P \cdot C + \triangle M_H = w \cdot L + \pi_0 - T \tag{2.30}$$

where π_0 denotes the profits from the previous period, received at the beginning of the current period. $\triangle M_H$ is a flow figure since it depends on the desired amount of cash M_H and the money held at the beginning of the period M_0:

$$\triangle M_H = M_H - M_0 \tag{2.31}$$

Several assumptions are made:

- ⋆ Consumption, leisure and savings are absolutely superior. If the household receives a larger income from profits, none of these values will decrease absolutely,
- ⋆ Consumption, leisure and savings are net substitutes,
- ⋆ The substitution effect of a change in the real wage is larger than the income effect, and
- ⋆ Nominal savings react positively to increases in the price level.

[8] L denotes the realised labour which need not necessarily coincide with labour supply L^s.

In a first step, the household formulates its notional plans. They are the result of utility maximisation subject to *only* the budget constraint. These plans might not be fulfilled if the household faces rationing on one of the markets, and it might need to revise them. The notional plans depend on the price variables and on variables related to the income. In general, these plans and their dependencies can be expressed as:

$$C(\alpha) = C(\underset{(-)}{P}, \underset{(+)}{w}, \underset{(+)}{\pi_0}\underset{(+)}{M_0}, \underset{(-)}{T}),$$

$$L^s(\alpha) = L^s(\underset{(-)}{P}, \underset{(+)}{w}, \underset{(-)}{\pi_0}\underset{(-)}{M_0}, \underset{(+)}{T}),$$

$$M_H^d(\alpha) = M_H^d(\underset{(+)}{P}, \underset{(+)}{w}, \underset{(+)}{\pi_0}\underset{(+)}{M_0}, \underset{(-)}{T}),$$

with $\alpha := (P, w, \pi_0, M_0, T)$.

The household might face rationing on the labour market, or the goods market, or both. If the household expects that it will be able to withdraw from rationing, it will formulate its effective plans in a second step, and the dependencies are as follows:

$$\tilde{C} = \tilde{C}(P, w, \pi_0, M_0, T, \underset{(+)}{\bar{L}}), \qquad \tilde{M}_H^d = \tilde{M}_H^d(P, w, \pi_0, M_0, T, \underset{(+)}{\bar{L}})$$

$$\tilde{L}^s = \tilde{L}^s(P, w, \pi_0, M_0, T, \underset{(+)}{\bar{C}}), \qquad \tilde{M}_H^d = \tilde{M}_H^d(P, w, \pi_0, M_0, T, \underset{(-)}{\bar{C}})$$

$$\tilde{M}_H^d = \tilde{M}_H^d(P, w, \pi_0, M_0, T, \underset{(-)}{\bar{C}}, \underset{(+)}{\bar{L}})$$

These equations express the following:

- ⋆ Less rationing on the labour market (the rationing bound on the labour market moves up, giving more space for consumption) results in a higher consumption demand and a higher demand for money.

- ⋆ Less rationing on the goods market results in an increased labour supply and decreased money demand.

- ⋆ If an agent is rationed on both markets, the money demand depends positively on decreased rationing on the goods market and negatively on decreased rationing on the labour market.

If, in Walrasian fashion, the money market is assumed to be in equilibrium if the other two markets are also in equilibrium, it is possible to define the nominal marginal propensity to consume and the nominal marginal propensity to labour supply as

$$c' := \frac{\partial P\tilde{C}}{\partial w\bar{L}} \tag{2.32}$$

$$l^{s'} := \frac{\partial w\tilde{L}^s}{\partial P\bar{C}} \tag{2.33}$$

Equation (2.32) expresses how strongly consumption expenditures react to a change in labour income. Equation (2.33) expresses how strongly the value of labour supply reacts to a change in supply on the goods market. With these definitions, the household is completely depicted.

Aggregate Firm

The aggregate firm (subsequently firm) is not able to stockpile commodities, adjusts production quantities, and maximises its profits. The firm has the profit function

$$\pi = P \cdot Y - w \cdot L \tag{2.34}$$

It faces a production function with positive but decreasing marginal revenues,

$$Y = f(L) \tag{2.35}$$

With the Lagrange technique, profit is maximised subject to the budget constraint and the following notional functions are obtained:

$$Y(\alpha) = Y(\underset{(+)}{P}, \underset{(-)}{w}) \tag{2.36}$$

and

$$L^d(\alpha) = L^d(\underset{(+)}{P}, \underset{(-)}{w}) \tag{2.37}$$

If the firm faces rationing on a market, it will revise its plans and state its effective Clower plans instead. These are:

$$\tilde{Y}(\bar{L}) = f(\bar{L}) \tag{2.38}$$

and

$$\tilde{L}^d(\bar{Y}) = f^{-1}(\bar{Y}) \tag{2.39}$$

In contrast to the household, it is assumed that the firm can face binding rationing only on one market because it is assumed that no stockpiling of goods is possible. If stockpiling were allowed, then binding rationing on both markets would be possible on the firm's side. Stockpiling is the equivalent to savings of the household sector, which occur involuntarily if the aggregate household faces rationing on both markets.

The exposition given here allows for modelling underemployment. It is possible to account for a situation in which the firm demands labour independently from the current wage rate as shown in Figure 2.2:

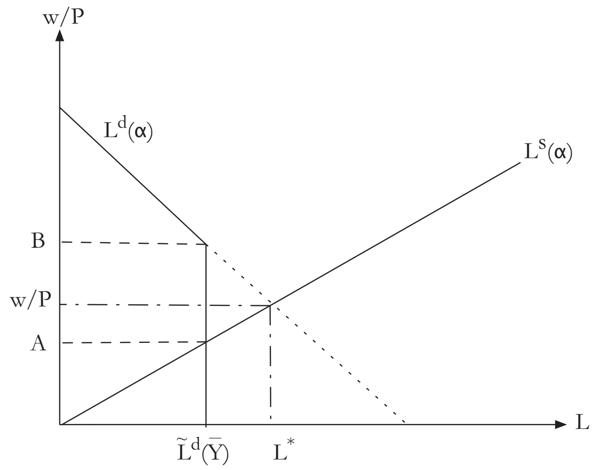

Figure 2.2: Underemployment Through Rationing of Sales in a Neo-Keynesian Model (cf. Felderer and Homburg (2005, p. 310))

If the firm faces a constraint on its sales ($\tilde{L}^d(\bar{Y})$), it needs less labour than it would need to produce its planned output. Barro and Grossman (1976, p. 42) show that a firm will choose its labour demand independently of the current wage rate so as to generate the output specified by the constraint. Changes in the demand for the firm's commodity influence effective labour demand independently of changes in W/P (cf. Barro and Grossman (1976, p. 43)). Even if the real wage decreased, the firm would only demand as much labour as it needed for the production of the quantity resulting from the constraint. This demonstrates how, within the context of the dual decision hypothesis, too little demand for commodities might result in underemployment. Furthermore, the firm demands money

$$\triangle M_F^d := \pi - \pi_0 \tag{2.40}$$

Money holdings of the firm are the difference between actual profits (π) and distributed profits (π_0). If actual profits π exceed the distributed profit π_0 the difference will be kept by the firm and distributed in the next period.

Government

The government is designed as an homogeneous construct of fiscal and monetary authorities. It faces a budget constraint with price P, government demand G, tax income T and household savings $\triangle M^s$;

$$P \cdot G = T + \triangle M^s \tag{2.41}$$

The government demands consumption goods and has to finance this demand by creation of money $\triangle M^s$ and by levying a lump-sum tax T. Government demand is assumed to have priority, i.e the government always enforces its demand while the household may face rationing.

Markets

The three agents interact on three markets: the goods market, the labour market and the money market. In a first step the notional plans of the agents are investigated. The equilibrium conditions for the notional plans are:

$$C(\alpha) + G = Y(\alpha) \tag{2.42}$$

$$L^d(\alpha) = L^s(\alpha) \tag{2.43}$$

and

$$\triangle M_H^d(\alpha) + \triangle M_F^d(\alpha) = \triangle M^s \tag{2.44}$$

Viewing the system as Walrasian:
If one views the system as Walrasian, i.e. the money market is neglected

and money is the numéraire, the following price-adjustment functions are specified:

$$\frac{dP}{dt} = H_1[C(\alpha) + G - Y(\alpha)]; \quad H_1(0) = 0, H_1^{'} > 0 \tag{2.45}$$

$$\frac{dw}{dt} = H_2[L^d(\alpha) - L^s(\alpha)]; \quad H_2(0) = 0, H_2^{'} > 0 \tag{2.46}$$

On the goods market, prices increase (decrease) in a situation of excess demand (excess supply) and analogous adjustments happen with the wage rate. When the agents have stated their notional plans it is possible to visualise equilibrium loci for the goods and labour market and Figure 2.3 represents these.

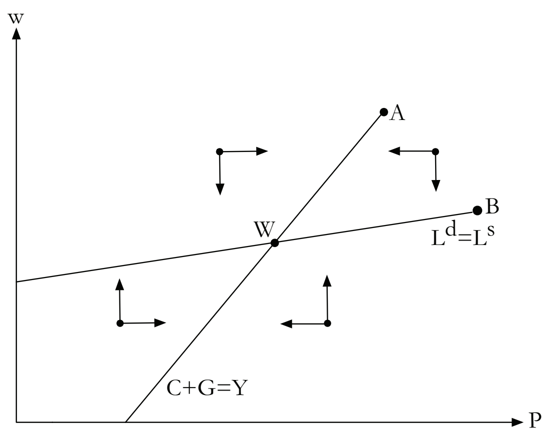

Figure 2.3: Walrasian Equilibrium Loci in a Neo-Keynesian Model (cf. Felderer and Homburg (2005, p.312))

Each of the curves represents equilibrium loci for a market. Displayed are the locus for the goods ($C + G = Y$) and the locus for the labour market ($L^d = L^s$). Points on a locus identify quantity equilibria on the market that is depicted by it. At point W both markets are in Walrasian equilibrium. Here, the given prices are the equilibrium prices. All points except W represent some excess supply and/or excess demand, and the following regions can be identified:

- ⋆ *upper left* : excess demand on the goods market, excess supply on the labour market,

⋆ *upper right* : excess supply on both markets,

⋆ *lower right* : excess supply on the goods market, excess demand on the labour market,

⋆ *lower left* : excess demand on both markets.

The adjustment process of the prices will lead the system into Walrasian equilibrium W. The paired arrows show the directions in which the prices adjust starting from the various regions of the graph. So far, this is still general equilibrium theory, and, without modifications, the system will converge towards equilibrium where no agent faces rationing. Equilibrium prices will be established and the system will have arrived at point W.

Viewing the System as Neo-Keynesian:
At this point, a neo-Keynesian model makes a difference. By application of the dual decision hypothesis, a world can be described where trade takes place at disequilibium prices. One knows from the dual decision hypothesis that an agent revises his demand (supply) once he faces rationing in a market. By applying this assumption to the present model, a different representation of equilibrium loci is obtained. With the modified version of the model it is possible to construct quantity adjustment processes to establish equilibria with quantity rationing at disequilibrium prices. The application of the dual decision hypothesis modifies the graph shown in Figure 2.3. For example, part WB (not including W itself) of the equilibrium loci of the labour market is characterised by the labour market being in equilibrium and the goods market being in disequilibrium. There is excess supply of goods: the firm is being rationed on the goods market and, according to the dual decision hypothesis, revises its supply of goods by demanding less labour to produce fewer goods. The equilibrium locus WB becomes obsolete. Similar arguments hold for the other regions. New neo-Keynesian equilibrium loci will be established and these are depicted by Figure 2.4[9]:

[9] Details about the slopes of the curves can be found in Felderer and Homburg (2005, p. 421)

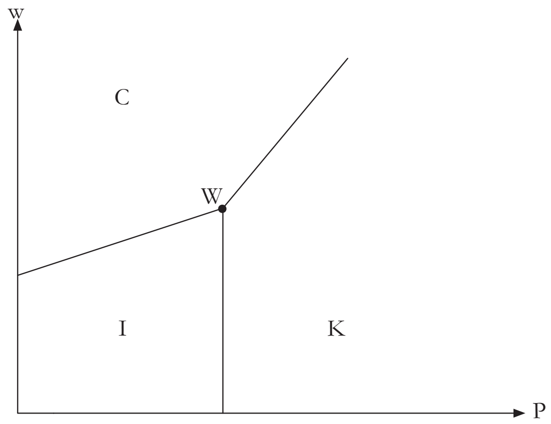

Figure 2.4: Equilibrium Loci in a Neo-Keynesian Model (cf. Felderer and Homburg (2005, p. 314))

Regimes

Figure 2.4 represents the neo-Keynesian equilibrium loci which can be derived by application of the dual decision hypothesis. The loci are characterised by the combinations of price and wage so that the effective plans of the agents match, while points that do not lie on one of the loci represent a situation where the effective plans do not match and rationing occurs. Three such rationing regimes can be identified.

- ⋆ *C*: classical unemployment, excess supply for labour and excess demand for good: the household is rationed on the goods and the labour market,

- ⋆ *K*: Keynesian unemployment, excess supply in goods and labour market: the household is rationed on the labour market and the firm faces rationing on the goods market,

- ⋆ *I*: supressed inflation, excess demand for labour and for goods: the household faces rationing on the goods market and the firm is rationed on the labour market.

So how does the model adjust from one of the regimes to an equilibrium with rationing (notional plans not being met)?

Classical Unemployment
In C, the household is rationed on both markets. It revises labour supply and consumption demand. In this situation its optimisation problem includes the bounds faced on all markets. The household formulates its effective demands (supplies). Because of the rationing on the labour market the household reduces its demand for consumption, and because of the rationing faced on the goods market, it revises its labour supply, too. A situation on the equilibrium locus is achieved through a double revision of the plans by the household. The firm is not affected in this situation and, as the short-side rule implies, is able to realise its notional plans.

Keynesian Unemployment
At Keynesian unemployment, both firm and household face rationing. The adjustment process is not as simple as in the case of classical unemployment, as a quantity tâtonnement takes place before equilibrium is established. In the initial situation, the household requests a quantity of the good less than the quantity offered by the firm. The effective plans of consumer and producer do not match. As a result, the firm reduces its supply of goods because it faces rationing on the goods market. Since it now needs to produce less quantity, it will also demand less labour. The plans on the goods market match, but because of the decrease in labour demand there is an excess supply of labour and the household now faces rationing. It decreases labour supply and consumption demand to the rationing level. The firm is rationed once again and the process continues until the demands of the household and the firm coincide. Note that, since this is a fixed-price fixed-wage model, only a quantity adjustment takes place, because during any given period prices and wages remain fixed.

Repressed Inflation

As under Keynesian unemployment, the firm and the household are rationed. A quantity tâtonnement takes place, analogous to the process described under *Keynesian Unemployment*.

The processes towards an equilibrium in a neo-Keynesian model can be summarised visually by this flow chart:

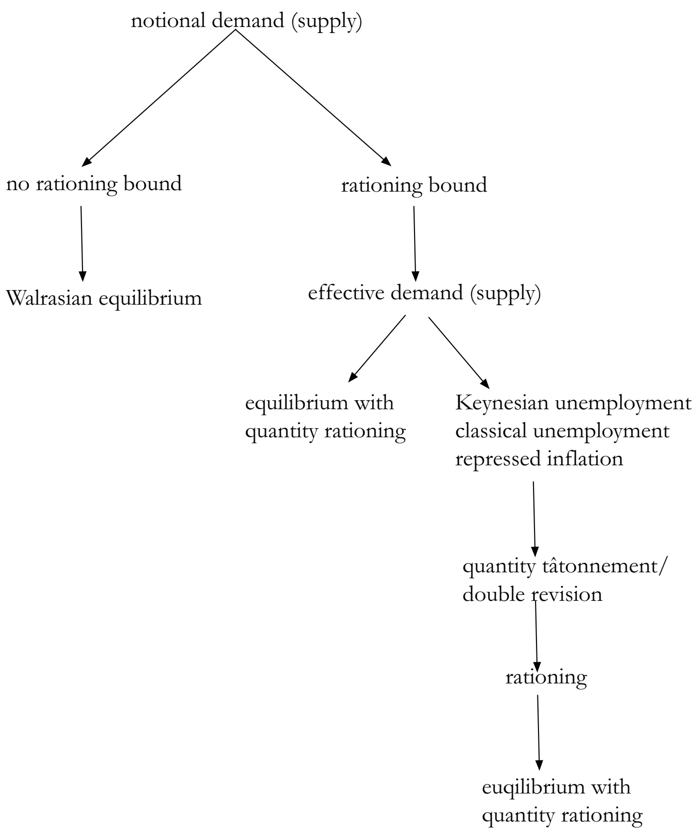

Figure 2.5: Processes Towards Equilibrium in a Neo-Keynesian Model

Chapter 3
Methods

Rationing and the consideration of quantity signals are crucial when it comes to modelling an economy that allows for trade at disequilibrium prices. While in general equilibrium models, all agents realise their notional plans, this assumption is dismissed in models with quantity rationing. Here, decisions can be taken step-wise and with respect to whether an individual faces rationing or not. In a neo-Keynesian model where households and firms can face rationing, different regimes can emerge, depending on who faces rationing and the adjustment process towards a stable arrangement. Having outlined such a macroeconomic model, how does rationing actually take place on the micro-level? How are commodities allocated to individuals? In the following, I introduce basic axioms regarding rationing after Sprumont (1991) and demonstrate the uniform allocation rule, which is the unique rationing method that fulfills all of the axioms, and other rationing methods. The other methods that will be used subsequently in this work, such as Bayesian inference and computational simulation with SimEnv, are then also introduced.

3.1 Rationing

3.1.1 Introduction

According to Wordsmyth, rationing is:

> 1. to restrict use of (scarce) goods to authorized people in authorized amounts.
> 2. to distribute (supplies, food, or the like) in allotted amounts.
>
> Wordsmyth (2002)

The development from general equilibrium theory through the reinterpretations of Keynes (1936) to models of the neo-Keynesian type, made it necessary to include rationing into such models. As laid out in Chapter 2, and remarked by Polterovich (1993, p. 1) in the 1970s, several authors developed models with quantity rationing, such as Barro and Grossman (1976), Benassy (1977) and Malinvaud (1977). The question was now how to implement the possibility of unsatisfied wishes of some agents. In both Walrasian models and neo-Keynesian models, tâtonnements take place. While in the former this is a price adjustment process, in the latter it is a quantity adjustment process which takes place during a period, and prices are usually assumed to stay fixed. The agents in a neo-Keynesian framework revise their planned transactions in view of quantity signals that they only receive when facing rationing. If they receive no signals there is no need to restate plans. Those agents who face rationing and those who do not are determined by the underlying rationing method which sends quantity signals to those agents who are affected by it. On the macro level rationing simply defines the side of the market that remains with unsatisfied wishes, while on the micro level a rationing method determines which agents are affected by it and how. In the present chapter, a basic axiomatic introduction to rationing on the micro level will be presented in order to define several rationing methods.

Let there be an economy in the sense of Benassy (2002) with price and quantity signals being exchanged by the agents and with the sum of individual effective demands out of alignment with the sum of individual

effective supplies. Prices are in disequilibrium as they have not had time to adjust to equilibrium values, but, nevertheless, a distribution needs to take place and for that purpose a rationing method needs to be defined (cf. Thomson (1994, p. 220)). The situation can be summarised as follows: a given amount a of a commodity with a set price per unit is offered to a given number of consumers. The consumers request a certain amount of the commodity, but the available quantity of this commodity does not match the sum of individual demands. A rationing problem can thus be defined following along the lines of Moulin (2000):
A rationing problem is given by a population of agents, a profile of individual preferences for each agent (i.e. plans regarding transactions), and a quantity of a commodity to be divided among these agents. A *rationing method* can be defined as a method that with every rationing problem associates a profile of individual shares for each agent.

3.1.2 Axioms

The following outline is based on Sprumont (1991), who identifies four axioms that a *perfect* rationing method should fulfill. These axioms are fulfilled uniquely by the uniform allocation rule, but by relaxing one or more of them, other rationing methods can be constructed. In the following, the framework within which Sprumont (1991) embeds the axioms will be introduced.

Let there be an amount a of some commodity available for a population of demanders which is expressed as a finite set $\mathbf{N} = \{1, ..., n\}$. Each demander $i \in \mathbf{N}$ has a complete preference pre-ordering of $[0, a]$ which is depicted as R_i. x, y are quantities of the commodity. The following assumptions regarding the preference pre-ordering are made:
For all $x, y \in [0, a]$:

- $\star$ xR_iy means that agent i thinks that consuming x units is at least as good as consuming y units,
- $\star$ xP_iy means that agent i strictly prefers x units over y units, and
- $\star$ xI_iy means that agent i is indifferent between x and y units.

For each $x \in [0,a], \{y \in [0,a] | yR_ix\}$ and $\{y \in [0,a] | xR_iy\}$ are closed sets. Sprumont (1991) further assumes that the preferences are single peaked[1] and strictly decreasing around that peak. The preference pre-ordering R_i for demander i fulfills the condition below:

There is an allocation $x^* \in [0,a]^2$ such that:

$$\forall y,z \in [0,a] : \begin{cases} x^* < y < z \Rightarrow x^* P_i y P_i z \\ x^* > y > z \Rightarrow x^* P_i y P_i z \end{cases} \tag{3.1}$$

where x^* is the optimum allocation for agent i. Allocations y and z constitute larger values than x^*, but x^* is the preferred allocation. Since x^* depends on the preference pre-ordering it can be written as $x^*(R_i)$. Let $\mathbf{S}$ denote the set of all continuous pre-orderings of $[0,a]$ that satisfy (3.1). For any $x^* \in [0,a]$, $\mathbf{S}(x^*)$ depicts the subset of those preferences in $\mathbf{S}$ which have as a peak x^*.

Demanders announce their preferences by stating a demand. $R = (R_i)_{i \in N}$ denotes the vector that depicts the announced preferences of all agents, i.e a list that contains one element for each announced preference. A *perfect* rationing method is said to be a function $\phi : \mathbf{S}^\mathbf{N} \to [0,a]^\mathbf{N}$ that satisfies the following axioms:

1. Feasibility

$$\forall R \in \mathbf{S}^\mathbf{N} : \sum_{i \in \mathbf{N}} \phi_i(R) = a$$

2. Efficiency

$$\forall R \in \mathbf{S}^\mathbf{N} : \{\sum_{i \in \mathbf{N}} x^*(R_i) \geq a\} \Rightarrow \{\phi_i(R) \leq x^*(R_i), \forall i \in \mathbf{N}\}$$

$$\forall R \in \mathbf{S}^\mathbf{N} : \{\sum_{i \in \mathbf{N}} x^*(R_i) \leq a\} \Rightarrow \{\phi_i(R) \geq x^*(R_i) \forall i \in \mathbf{N}\}$$

[1] At single-peak preferences agents become satiated once an allocation has reached a certain level. Allocations that exceed the satiation level decrease utility (cf. Thomson (1994, p. 220)).

[2] The index of x^* is omitted. It refers to the preference pre-ordering of agent i, R_i, so one could also write $x^*(R_i)$.

3. Anonymity
   For all permutations π of the population $\mathbf{N}$, all $R \in \mathbf{S}^{\mathbf{N}}$:

$$\phi_i(R^\pi) = \phi_{\pi(i)}(R)$$

   where $R^\pi = (R_{\pi(i)})_{i\in\mathbf{N}}$

4. Strategy-Proofness

$$\forall i \in \mathbf{N}, R \in \mathbf{S}^{\mathbf{N}}, R_i^{'} \in \mathbf{S} : \phi_i(R_i, R_{-i}) R_i \phi_i(R_i^{'}, R_{-i})$$

   where $R_i^{'}$ denotes misreported preferences.

An allocation is said to be *feasible* if the sum of the allocated units is equal to the units available (a). An allocation is *efficient*, if under excess demand nobody is forced to receive more than he wishes, and under excess supply everybody receives as much or more than he wishes. *Anonymity* postulates that the allocation for each demander is the same no matter which order the demanders are served in. Finally, *strategy-proofness* requires that if a demander announces incorrect preferences in order to obtain more than he would by stating his correct preferences, he will not be successful. The axioms above are only fulfilled by the uniform allocation rule, but by relaxing one or more of the axioms it is possible to construct other rationing methods[3]. The uniform allocation rule and some other rationing methods are now introduced.

3.1.3 Rationing Methods

Uniform Allocation Rule

The uniform allocation rule is the unique rule fulfilling all the axioms from above. A proof can be found in Sprumont (1991, p. 511), where anonymity

[3]Benassy (2002) also discusses properties of rationing methods but uses different notions and considers only three properties, namely *voluntary exchange, market efficiency* and *non-manipulability*. Voluntary exchange in Benassy (2002) refers to the same as efficiency in Sprumont (1991), while efficiency in Benassy (2002) refers to frictionless markets where rationed demanders and rationed suppliers can not both be found at once. Strategy-proofness as in Sprumont (1991) is called non-manipulability in Benassy (2002). By considering only these three properties the rationing methods that can be derived are not restricted to the uniform allocation rule.

is posed, while Ching (1992) demonstrates that the uniform allocation rule also fulfills all axioms when anonymity is substituted by the stronger assumption of *envy-freeness*[4] and preferences are single-plateaued[5]. Another proof, independent from the one in Sprumont (1991) can be found in Ching (1994). Uniform rationing can be expressed as:

$\forall i \in \mathbf{N}$,

$$\phi_i(R) \begin{cases} = min\{x^*(R_i), \mu(R)\} \; if \; \sum_{i \in \mathbf{N}} x^*(R_i) \geq a \\ = max\{x^*(R_i), \lambda(R)\} \; if \; \sum_{i \in \mathbf{N}} x^*(R_i) \leq a \end{cases} \tag{3.2}$$

with:
$\mu(R)$ solving $\sum_{i \in \mathbf{N}} min\{x^*(R_i), \mu(R)\} = a$
$\lambda(R)$ solving $\sum_{i \in \mathbf{N}} min\{x^*(R_i), \lambda(R)\} = a$

The uniform allocation computes, in case of excess demand, an equal share $\mu(R)$ by dividing the available amount a by the number of demanders. Each of the demanders obtains the minimum of his demand $x^*(R_i)$ (which is the peak of his preferences) and the share $\mu(R)$. This process continues until all units have been allocated.

If there is a situation of excess supply this rule can also be applied. Sprumont (1991) views excess supply as a *bad* which has to be completely allocated to the population of demanders. Therefore the uniform allocation rule then computes an equal share of supply which is then distributed to the demanders as the maximum of this share and their demand.

An alternative approach to deal with excess supply is the following: the suppliers supply more units than demanded, and strategy-proofness/voluntary exchange are met. The suppliers sell the minimum of their supply and the equal share of demand until there is no demand left, the expression given by 3.2 is simply applied analogously. In the remainder of this work this view is applied unless noted differently.

[4] Envy-freeness postulates that no agent prefers any other agent's allocation.
[5] At single-plateaued preferences there is a segment of preferred allocations, while at single-peakedness there is only a single preferred allocation.

Queuing Rationing

Queues are common in reality, as demanders are often served in the order indicated by the waiting line (e.g. ticket counter). This rationing method can also be referred to as priority rule, since the queue could be interpreted as an ordering that assigns highest priority to the first agent in the line and descending priority to the following agents. The first demander obtains what he wishes:
$\phi_1(R) = x^*(R_i)$.[6]
The following demanders obtain shares according to the following rule:

$$\begin{aligned} &\phi_i(R) = min\{x^*(R_i), 1 - \textstyle\sum_{j<i} \phi_j(R) \\ &\text{for } 1 < i \leq n \end{aligned} \tag{3.3}$$

A demander obtains his demand, or if his demand exceeds what is left, he obtains what the demanders before him ($j < i$) have not taken. This method is not anonymous as the order in which the agents appear determines the share they receive.

Proportional Rationing

Rationing on the basis of a proportional rule is not strategy-proof as it can result in overbidding, as shown by Benassy (2002, p. 22-3). Agents can increase their share by exaggerating their preferences. Proportional rationing can be expressed as

$$\phi_i(R) = \frac{x^*(R_i)}{\sum_{j \in \mathbf{N}} x^*(R_j)}. \tag{3.4}$$

The amount a is divided among the agents proportionally to their demand.

Egalitarian Rationing

At egalitarian rationing each demander obtains a fixed amount of the commodity and, hence, egalitarian rationing is not efficient.

$$\phi_i(R) = \frac{a}{n}. \tag{3.5}$$

[6] Sprumont (1991) does not mention the case in which the first agent demands more than there is available a. This could be the case, and then the expression would be this: $\phi_1(R) = \min(x^*(R_i)$, a).

3.2 Bayesian Inference

3.2.1 Introduction

> (...) the degree of our belief is capable of modification and may need it. But in accordance with what is the belief to be modified? Obviously in accordance with experience; it cannot be trusted by itself (...)
>
> Venn (1888, p. 25)

Human agents often need to act on the basis of expectations that they can not, from the onset, have great confidence in and which they need to improve as new information becomes available. When modelling such an expectation formation, some reasonable way of improving expectations is needed. In many situations it is necessary to consider not only frequentist probabilities but also guesses about them (cf. Haas and Jaeger (2005, p. 6)). Haas and Jaeger (2005) exemplify that there is a difference between relative frequencies that one can observe and make forecasts about (e.g. the rolling of a dice), and unknown relative frequencies that one can not forecast, but only make guesses about (e.g. the application behaviour of competitors). Furthermore, by referring to de Finetti's theorem, which shows how an unknown probability distribution can be approximated by considering information from additional samples that are obtained step by step, they show that probability measures can be used for analysing guesses about frequentist probabilities. The approximation of the relative frequentist probability may start from very diverse initial guesses, as different individuals may have very good reasons for distinct guesses. In Chapter 5, Bayesian reasoning, the approximation of relative frequentist probability, is used to depict labour suppliers and their decision to apply for employment. The labour suppliers do not have knowledge about their competitors' application behaviour but need to consider it when assessing their chances of making a successful application.

3.2.2 On Probabilities, Events and Densities

The following outline draws specifically on Lee (2004). Let $\mathbf{\Omega}$ be a sample space consistent with the total data available, e.g. the set of possible applicant pools. The probability that some event from $\mathbf{\Omega}$ happens shall be $P(\mathbf{\Omega}) = 1$, and the probability that no event from $\mathbf{\Omega}$ occurs shall be denoted $P(\emptyset) = 0$. ω denotes the elementary event. An elementary event is a singleton[7] containing an element from the sample space $\mathbf{\Omega}$. For example when rolling a dice then $\mathbf{\Omega} = \{1,2,3,4,5,6\}$, and an elementary event is any singleton from $\mathbf{\Omega}$, e.g $A = \{2\}$. If each elementary event has the same probability, it is reasonable to call the cardinality quotient of A and $\mathbf{\Omega}$ the probability of event A (cf. Reinhardt and Soeder (1998, p. 467)).

$\tilde{m}(\omega)$ shall denote a random variable which is a function of the elementary event. $P_{\tilde{m}}(m)$ denotes the probability that the random variable takes the value m, that is $P(\tilde{m} = m) = P(\{\omega : \tilde{m}(\omega) = m\})$. A random variable has a probability distribution which can be expressed as a cumulative probability distribution function (CDF). The CDF for a discrete random variable is defined as:

$$\begin{aligned} F(m) &= F_{\tilde{m}}(m) = P(\tilde{m} \leq m) = P(\{\omega : \tilde{m}(\omega) \leq m\}) \\ &= \textstyle\sum_{k \leq m} P_{\tilde{m}}(k) \end{aligned} \tag{3.6}$$

while for a continuous case it is

$$F(m) = P(\tilde{m} \leq m) = P(\{\omega : \tilde{m}(\omega) \leq m\}) \tag{3.7}$$

with

$$\begin{array}{ll} F(m) \leq F(m'), & \text{if } m \leq m', \\ \lim_{m\to\infty}(F(m)) = 1, & \lim_{m\to-\infty} F(m) = 0. \end{array}$$

In the case that the random variable is continuous, a probability density function $p(m)$ (more strictly $p_{\tilde{m}}(m)$) exists, so that

$$F(m) = \int_{-\infty}^{m} p_{\tilde{m}}(m)dm \tag{3.8}$$

where $p(m)$ cannot itself be interpreted as a probability, but if σ is some parameter, then for sufficiently small values of σm a probability can be ob-

[7] A singleton is a set containing a single element.

tained:

$$p(m)\sigma m \cong P(m < \tilde{m} \leq m + \sigma m) = P(\{\omega : m < \tilde{m}(\omega) \leq m + \sigma m\}). \quad (3.9)$$

$p(m)\sigma m$ is called a probability element. However, letting σm converge towards zero ($\sigma m \to 0$), results in a probability of zero: $P(\tilde{m} = m) = 0$, which means that the probability of a continuous random variable $\tilde{m}$ exhibiting the exact value m is equal to zero.

3.2.3 Bayes' Theorem

Let θ denote an event, H a hypothesis regarding the probability of an event, and let $P(\theta|H)$ stand for the conditional probability of θ given that H is true. It has often been argued that $P(\theta|H)$, is the long-run frequency with which θ happens under the assumption that H holds (cf. Lee (2004, p.3)) A problem is then that in order to obtain a good approximation of $P(\theta|H)$, a large number of (identical) experiments must be made, and one can never be sure about the availability of such experiments. Sometimes one wants to find out probabilities of events but can not rely on a large number of trials. An alternative approach to a series of experiments is to express $P(\theta|H)$ as a measure of belief in θ, given that one knows that H is true. Lee (2004) lists the following axioms that such a probability measure should fulfill:

1. $P(\theta|H) \geq 0, \quad \forall \theta, H,$
2. $P(H|H) = 1, \quad \forall H,$
3. $P(\theta \cup T|H) = P(\theta|H) + P(T|H)$ if $\theta \cap T \cap H = \emptyset$, and
4. $P(\theta|T \cap H) \cdot P(T|H) = P(\theta T|H)$

By assuming that $T = H \backslash \theta$ in axiom 3 and by applying the first two axioms, it follows that the degree of belief in θ given H should be less than or equal to one and larger than zero:

$$0 < P(\theta|H) \leq 1, \quad \forall \theta, H$$

Bayes' theorem can be derived from these axioms (see Lee (2004, pp. 7)) and can be expressed as follows for the case of several hypotheses:

$$P(H_j|\theta) = \frac{P(H_j)P(\theta|H_j)}{P(\theta)} = \frac{P(H_j)P(\theta|H_j)}{\sum_l P(\theta|H_l)P(H_l)} \quad (3.10)$$

The posterior probability of hypothesis H_j, $P(H_j|\theta)$, is the updated degree of belief in H_j after θ has occured. It is the product of the conditional probability $P(\theta|H_j)$ that θ occurs when hypothesis H_j is true and the prior probability $P(H_j)$ of hypothesis H_j. This product is divided by the total probability $P(\theta)$ of the event θ. The total probability of θ is the probability of θ given all information that is available, i.e all hypotheses that are considered. $P(\theta)$ is obtained by summing up the weighted probabilities of θ at each hypothesis.

Within a dynamic context, Bayes theorem enables the modelling of learning processes where several hypotheses are considered and the prior probabilities of the hypotheses are revised as soon as new information becomes available. Such a learning process will be modelled in Chapter 5.

3.2.4 First and Second-Order Probabilities, Priors and Posteriors

In Chapter 5, a labour market will be modelled where agents consider several hypotheses, and learn in the course of action about the probability of these hypotheses. Each hypothesis in itself is a probability distribution for some events on the labour market. For simplicity, an invariant number of static distributions $H_1, ..., H_m$ will be considered. On a basic level, a specific distribution expresses the uncertainty regarding the actual occurrence of a specific event. These distributions will be called *first-order probabilities*. On a meta-level the *second-order probabilities* express the uncertainty about the first-order probabilities. The second-order probabilities convey an agent's degree of belief in his alternative hypotheses.

> An example of a second order probability is furnished by a cartoon in "The New Yorker" showing a forecaster making following statement: *There is now a 60% chance of rain tomorrow, but there is 70% chance that later this evening the chance of rain tomorrow will be 80%.*
>
> Gaifman (1986, p. 277)

A *prior* or *prior second-order probability* expresses the probability that is assigned to a first-order probability before observation of an event, while a *posterior second-order probability* is obtained by taking into account new information. When considering a learning process where events happen consecutively, the posterior second-order probability of a hypothesis obtained while learning from the previous event serves as the new prior second-order probability of that hypothesis before observing the next event. At the very beginning of such a learning process the *initial prior second-order probability* of a hypothesis is assigned on the basis of subjective reasoning.

3.3 SimEnv

3.3.1 About SimEnv

> SimEnv is a multi-run simulation environment that focuses on evaluation and usage of models with large and multi-dimensional output mainly for quality assurance matters and scenario analyses using sampling techniques. Interfacing models to the simulation environment is supported for a number of model programming languages by minimal source code modifications and in general at the shell script level. Pre-defined experiment types are the backbone of SimEnv, applying standardised numerical sampling schemes for model parameters, initial or boundary values, or driving force spaces. The resulting multi-run experiment can be performed sequentially or in parallel.
>
> Flechsig, Böhm, Nocke, and Rachimov (2005, p. 11)

This software is provided by the IT department of PIK. Chapter 5 contains a labour market model with several parameters that influence the results of a model run. Accomplishing a single evaluation of the model at one parameter calibration is not sufficient to allow for general statements about the behaviour of the model. Alternative evaluations of the model at different combinations of the parameters might produce very different results. SimEnv helps a modeller to gain information about the model behaviour at different parameter combinations and provides a variety of experiment types, including parameter screening. Before an experiment takes place, the modeller selects the appropriate experiment type and defines a set of factors representing parameters, boundaries, values, or drivers, according to which the model should be analysed. This set is a numerical sample and defines the multi-run experiment, which, once started, modifies the selected parameters numerically. A single simulation corresponds to one of the possible combinations of the parameters from the parameter set. In order to gain insights into how the model behaves at various parameter calibrations, a parameter screening with SimEnv will be carried out in Chapter 5. For each parameter, SimEnv is supplied with a set containing the values which this parameter should take, and SimEnv then generates as many model calibrations as there are combinations of the given parameter values. If there are three parameter sets X_1, X_2, X_3, SimEnv generates as many calibrations as given by the cardinality of the Cartesian product of the parameter sets $|X_1 \times X_2 \times X_3|$. Each calibration corresponds to a single run of the whole model. SimEnv lets the model run through all of these calibrations and keeps the output for post-processing. This includes the visualisation of the results for different parameter calibrations. The software generates a very large number of parameter combinations, and, thus, the results will be visualised for a limited range of parameter combinations within the subsequent text.

Chapter 4

A Rationing Toolbox in Mathematica

The uniform allocation rule is the only method which meets all the axioms previously introduced for rationing methods. By relaxing one or several axioms a variety of rationing methods can be constructed. The Mathematica toolbox provides modules that implement most of the rationing methods presented in the previous chapter. It also provides rationing modules that are variations of the methods introduced previously. These variations are specifically provided for the *lagom* framework. It enables the user, with a modular modelling approach, to adopt the rationing methods that he finds useful for his model. Economic, mathematical and computational documentation of all methods is available. Economic and mathematical documentation will be presented in this chapter, the documented code can be found in Appendix A.

4.1 Introduction to the Rationing Toolbox

4.1.1 Purpose

The aim of the rationing toolbox is to provide a variety of rationing methods that can be used and exchanged easily during the modelling work with *lagom*[1], and that could be readily extended to other modelling environments. As shown in Chapter 2, the need for models of general equilibrium with quantity rationing arises from a rich history regarding general equilibrium modelling. Rtioning methods as such, are analysed for example by Sprumont (1991), A general framework of models of general equilibrium with quantity rationing provided by Benassy (2002) enables the treatment of quantity rationing, and Clower (1965) describes agents' behaviour when they face rationing on some of the markets. However, it is necessary to find out which rationing method is best in which kind of setting and for which market. The need for the implementation of a specific rationing method arises, and the present rationing toolbox provides the modeller with a variety of such methods. Since the toolbox has been programmed to be applied within the *lagom* context, it contains not only some of the rationing methods from the literature and that were introduced in the previous section, but also some extended versions which fit the *lagom* context. Furthermore, two rationing methods have been provided specifically for the *lagom* context. These methods are not included in the literature and were therefore not introduced in Chapter 3.

Each of the subsequent methods has been programmed in the same fashion, so that the modeller can exchange methods with only minimal modifications to the program code. In the following, each method is referred to as a module[2], so the rationing toolbox contains as many modules as rationing methods.

Each rationing module generates – on arbitrary markets – from incon-

[1] *lagom* was introduced in Chapter 1.

[2] A module can be loaded within a program, and the methods contained in it can be called by the program by a specific command.

sistent demands and supplies consistent transactions. Transactions are purchases and sales that actually take place in the markets, even if overall demands and supplies do not match. Characteristics of markets differ (e.g. number of agents, efficiency of the market, behaviour of the agents). A rationing method that generates consistent transactions in one market might not do so in another market because the conditions are different. It may also be the case that the rationing method that works in one market works in another one, too, but provides undesired outcomes. Equilibria depend crucially on the choice of the rationing method, as Drazen (1980) notes with respect to unemployment equilibria.

The rationing methods are programmed so that they can be applied to nearly any model that requires rationing devices. The remainder of this chapter introduces the methods and provides specific documentations. Each rationing module will be documented economically and mathematically. A computational documentation has been included within the program code and can be found in Appendix A.

4.1.2 Prerequisites

In the *lagom* framework, one is confronted with markets with many different characteristics, depending not only on the number of agents on each side of the market but also on the behaviour of these agents and on the organisation of the market. On a labour market, there might be additional information available, such as productivities of agents, while on the goods market there might not be. Therefore, the rationing methods of the rationing toolbox must fulfill a number of prerequisites:

- ⋆ Each method must be able to distribute a given amount of an arbitrary commodity from the short side of the market to the long side of the market,
- ⋆ The rationing modules must be general enough to distribute from an arbitrary number of agents from the short side of the market to an arbitrary number of agents on the long side, and

⋆ The output of each rationing module must be transactions for both market sides.

4.1.3 On Mathematical Documentation

The main routine of each rationing module will be expressed as a *dynamical system*, while computations outside the main routine are described in a more relaxed fashion. Only a brief note on dynamical systems is given here in order to provide a basic understanding of what constitutes such a system. For a detailed but very compact introduction to dynamical systems see Denker (2005).

A dynamical system is an ordered pair $(\mathbf{X}, \phi)$, where $\mathbf{X}$ denotes the state space and ϕ denotes the evolution rule of the system. The state space $\mathbf{X}$ describes all possible states that the system can be in at any instance/iteration t (in case it is a discrete dynamical system) and consists of the state variables $(x_1, ..., x_n)$. A state is a specific combination of values of state variables. The evolution rule ϕ is a mapping from the state space $\mathbf{X}$ onto itself: $\phi : \mathbf{X} \rightarrow \mathbf{X}$ and has the following properties:

1. $\phi(0, x) = x, \forall x$

2. $\phi(t, \phi(s, x)) = \phi(s + t, x) \; \forall s, t, x$

The first property says that the state variables of the dynamical system do not change when not iterated. The second property expresses that the system evolves in s iterations from x to $\phi(s, x)$, and thereafter in t iterations from $\phi(s, x)$ to $\phi(s + t, x)$ (cf. Wikipedia (2006a)).

For mathematical documentation of the rationing methods I will proceeded as follows: Firstly, the *scenario* and the state space of the relevant rationing method will be introduced. During this introduction, iteration indices will be omitted for simplicity. The evolution rule will then be introduced to show how the system evolves from one state to the next, and here iteration indices will be applied. When stochastics are involved it may be the case that an element is randomly chosen from a set, and this set may a singleton. Iterations start with $t = 1$, while the initial conditions are given by $t = 0$.

4.1.4 Preparation of User Input

The user input for the rationing modules is processed identically by each module in order to prepare for the rationing process.

Initially, the plans of each market side are summed up in order to determine which side of the market is in excess and which side is in shortage. Since all agents on the short side of the market will realise their plans, the short side of the market can be summarised by the sum of the plans from that side.

Lists are used throughout the rationing modules. Here, a list is defined as an ordered set with an arbitrary but finite number of elements from $\mathbb{R}_+$. Initially, in each module, there are (at least) two lists, one depicting supplies **s** and another depicting demands **d**:

$$\begin{aligned} &\mathbf{s} = < s_1, ..., s_n > \\ &\text{and} \\ &\mathbf{d} = < d_1, ..., d_m > \end{aligned} \tag{4.1}$$

Each element of $s(d)$ depicts the supply (demand) of an agent. The sums of the elements in s and d are computed:

$$S = \sum_{i=1}^{n} s_i \tag{4.2}$$

$$D = \sum_{j=1}^{m} d_j \tag{4.3}$$

If the aggregate supplies and demands are equal, $S = D$, no rationing need take place. The rationing module sends a message to the user, and returns the demands and supplies, as these are realised as transactions. If this was the case for all markets of a model, each agent would have realised his notional or effective[3] plans, and a Walrasian equilibrium as introduced in Chapter 2 would have been established. However, if, $S \neq D$ the smaller sum is determined:

$$a = \min(S, D) \tag{4.4}$$

and a corresponds to the sum of the plans of the agents from the short side of the market. The plans of these agents will be fulfilled, giving as a first result the list q containing the elements whose sum is associated with a. q denotes the list of which each element depicts individual transactions of an agent on the short side of the market.

It remains to distribute the amount a of the commodity that is available to the agents on the long side of the market; these are depicted by the list the sum of which was larger, which shall be denoted **e**. The list **e** will be modified during the rationing process. In order to generate transactions, however, the original data from the long side will be needed. Therefore, **o** denotes the original data from the long market side.

[3] Whether these are notional or effective plans depends on how the agents are modelled. If the agents assume that they will realise their plans, no matter what, they are said to state notional plans. But if they anticipate some rationing and adapt their plans so as to account for it, they are said to state efficient Clower or Drèze plans. Whether they state Clower or Drèze plans depends on whether they assume to be able to manipulate the rationing level they face (see Section 2.2.3).

4.2 Random Uniform Rationing

4.2.1 Economic Documentation

This rationing module contains a two-stage rationing process which combines a random rationing process with uniform allocation. The uniform allocation rule is viewed as a relatively fair allocation rule when compared to other rules, as neither the strategic misreporting of preferences, nor the order in which the agents appear has an impact on the share they receive. Sprumont (1991), Ching (1992), Cachon (1999), and Klaus, Peters, and Storcken (1997) discuss properties of the uniform allocation rule within different contexts. If one applied the uniform rule exclusively, each agent would obtain a share of what is available for allocation. This does not seem very realistic (e.g. on a labour market this implies *full employment* in a broad sense). In order to gain from the advantages (i.e fairness) of the uniform allocation rule, but to avoid that all agents on the long side of the market will receive a share, a two-stage-process has been implemented:

Before applying the uniform allocation rule, a random process generates a sub-population of agents for which the uniform allocation rule is applied. The plan of each agent is known, and agents are drawn randomly from the population until the sum of their plans is equal to or greater than the sum of the plans from the short side. This sub-population of agents will fulfill at least part of their plans. The agents who are not chosen during the random process do not realise their plans even partially.

The amount of the scarce commodity is allocated to the sub-population under the uniform allocation rule. For that purpose, the available amount of the commodity is divided by the number of agents in the sub-population. A share is obtained and each agent from the sub-popuation now obtains the minimum of this share and his actual plan. If, after each agent obtained a share, there are agents left with unfulfilled plans and (!) there are still units of the commodity left, the remaining units of the commodity are divided by the number of agents whose plans have not been fulfilled. This process is continued until there are no more units to allocate. At the end of the process, the agents from the short side of the market will have realised all their plans, while only some agents from the long side of the market will have realised their plans completely or partially.

4.2.2 Mathematical Documentation

Random Rationing

There is a list $\mathbf{e} =< e_1, ..., e_m >$[4] depicting the preferences (plans) of the agents from the long side of the market. Elements are chosen randomly and are appended to a list $\mathbf{p} =< p_1, ..., p_n >$, which at first is empty. At the same time the rationer keeps a separate list **z** that keeps track of the positions of the elements drawn from **e**, avoiding the danger of choosing the same element repeatedly. This list **z** is needed later to generate the transactions on the excess side. The rationer keeps a copy **pr** of the original plans from the sub-population **p** to determine the transactions once the rationing process is over. Random elements from **e** are chosen until the sum of elements in **p** is equal to or greater than the amount a to be distributed. **p** now contains fewer or as many elements as **e**.

Uniform Allocation

Initially there are n agents who have been chosen randomly from the long side of the market, each with a plan $p_i \in \mathbb{R}_+$, so there is an ordered list of plans $\mathbf{p} =< p_1, ..., p_n >$. In this context, the agents from the sub-population are indexed i in contrast to those from the whole excess side which are indexed l. How many elements from **p** are unequal to zero is determined. This is denoted as $b \in \mathbb{N}$ and depicts the number of plans that have not yet been satisfied. At the beginning of the process no elements will be equal to zero, after iterations have taken place elements may be zero. Then an equal share is computed. To each agent i whose plan is captured in **p**, either the share or the plan is allocated, whichever is lower. The uniform allocation rule can be expressed as a dynamical system that converges to its fixed point, which is reached when all units a have been allocated. The

[4] Indices m and n are now used within this context, and may differ from those used in the previous section.

state space capturing all state variables is

$$\mathbf{X} = \mathbb{R}_+^n \times \mathbb{R}_+ \times \mathbb{N} \tag{4.5}$$

and contains $\mathbf{p} \in \mathbb{R}_+^n$, the amount $a \in \mathbb{R}_+$ to be allocated, and the number of elements from $\mathbf{p}$ that are non-zero $b \in \mathbb{N}$, so the tuple of state variables is given by $x = (\mathbf{p}, a, b)$. The dynamics of the system $\phi : \mathbf{X} \to \mathbf{X},\ x_{t-1} \mapsto x_t = \phi(x_{t-1})$ are defined as:

$$\begin{aligned} a_t &:= a_{t-1} - \textstyle\sum_{i \in \{1,\ldots,,n\}} \min(p_{i,t-1, \frac{a_{t-1}}{b_{t-1}}}) \\ b_t &:= |\{p_{i,t} \in p | p_{i,t-1} > 0\}| \\ \forall i &\in \{1,...,n\} : p_{i,t} := p_{i,t-1} - \min(p_{i,t-1}, a_t) \end{aligned} \tag{4.6}$$

When $a_t = 0$, the complete amount a has been allocated. $\mathbf{p}$ now represents the remaining plans of the agents from the sub-population at the end of the rationing process. The iteration index will be omitted now, as the rationing process is over. It remains to compute the realised plans of the agents from the sub-population and then to construct a list of the original population where all elements are set to zero, except those from the sub-population. The realised plans of the agents whose remaining plans are captured by $\mathbf{p}$ are determined by subtracting from each element in $\mathbf{pr}$ (see Section 4.2.2) the value of its corresponding element in $\mathbf{p}$. The values are stored in a new list $\mathbf{f}$:

$$\mathbf{f} = < f_i \mid f_i = pr_i - p_i > \tag{4.7}$$

The list that depicts the transactions of the whole population must now be generated. This is achieved by setting all elements of $\mathbf{e}$ that were not in $\mathbf{pr}$ to zero and substituting all elements from $\mathbf{e}$ that were in $\mathbf{pr}$ by their corresponding values from $\mathbf{f}$. The list with the positions $\mathbf{z}$ can be used and $\mathbf{e}$ is modified, so that

$$\forall i \in \{1,...,m\} : e_i = \begin{cases} 0 & if\ e_i \notin \mathbf{z} \\ f_i & if\ e_i \in \mathbf{z} \end{cases} \tag{4.8}$$

Finally, there are two lists, $\mathbf{q}$ and $\mathbf{e}$, that contain elements whose values depict the realised plans on each side of the market.

4.3 Queuing Rationing

4.3.1 Economic Documentation

This rationing method conducts rationing with regard to the order in which agents appear on the market. The agents on the long side of the market state their plans. The agent who is first in line, obtains the lesser of his demand and the units that are available (analogous argumentation in case of a line of suppliers can be given). The agents lower down the queue receive whichever is less: either their planned trade or what the agents before them have not taken.

Queuing rationing on a labour market seems unrealistic as no qualifications would be taken into account, while it might be a suitable method to distribute goods (e.g. queuing in a bakery).

4.3.2 Mathematical Documentation

There are m agents on the long side of the market whose plans are captured by the ordered list $\mathbf{e}$. There is an amount a to be allocated to the $l = 1, ..., m$ agents. $I := \min\{l \,|\, e_l > 0)\}$ shall denote the index l of the first element of $\mathbf{e}$ that is larger than zero.

The first agent in the queue whose plan is larger than zero is determined. This agent receives the lesser of his plan e_l and what is to be allocated. This process continues until the system has converged to its fixed point when all available units a have been allocated. The state space of this system is the following:

$$\mathbf{X} = \mathbb{R}_+^m \times \mathbb{R}_+ \times \mathbb{N}, \tag{4.9}$$

where $\mathbf{e} \in \mathbb{R}_+^m$, $a \in \mathbb{R}_+$, and $I \in \mathbb{N}$. The state variables of the system are given by the tuple $x = (\mathbf{e}, a, I)$. The function determining the dynamics of the system $\phi : \mathbf{X} \to \mathbf{X}$, $x_{t-1} \mapsto x_t = \phi(x_{t-1})$ is

$$
\begin{aligned}
&I_t := \min\{l \,|\, e_{l,t} > 0)\} \\
&a_t := a_{t-1} - \min(e_{l,t-1}, a_{t-1}) \\
&\forall l \in \{1,...,m\} : e_{l,t} := \begin{cases} e_{l,t-1} - \min(e_{l,t-1}, a_t) & \text{for } l = I_t \\ e_{l,t-1} & \text{else} \end{cases}
\end{aligned}
\tag{4.10}
$$

Generating Transactions

Once all units have been allocated, we are left with a list **e** that contains the excess demands/supplies of the agents from the long side of the market. The output of each rationing method is required (by the modellers) to be a list of transactions; one list for each market side. The transactions of the short market side **q** are already known. For the long side of the market an element from **e** is subtracted from the corresponding element in the original list **o**, resulting in a new list **y** that holds the transactions for the long side:

$$\mathbf{y} =< y_l \,|\, y_l = o_l - e_l > \tag{4.11}$$

Finally, the rationer returns the list with transactions for the short side of the market **q**, and the list with transactions realised by the long side of the market **y**.

4.4 Egalitarian Rationing

4.4.1 Economic Documentation

The egalitarian rationing method allocates a fixed amount of the scarce commodity to each agent, no matter whether states a demand/supply or not. The order in which the agents appear does not matter. Each agent simply receives the ahre that is obtained by dividing the total amount that is available by the number of agents. No agent can influence the result by strategically misreporting preferences. This rule is not efficient, as agents may be forced to obtain more units than they planned to trade. In a free economy this seems unrealistic, both on the labour market and on a typical goods market. An example where this might be a realistic concept is a market of insurance policies where the government forces each agent to insure against certain damage (e.g. health insurance, car insurance), whether the agent wants this or not.

4.4.2 Mathematical Documentation

Under egalitarian rationing all m agents obtain the same share that is determined by dividing the available amount a by the number of agents m. The state space is therefore:

$$\mathbf{X} = \mathbb{R}_+^m \times \mathbb{R}_+ \tag{4.12}$$

containing $\mathbf{e} \in \mathbb{R}_+^m$, and $a \in \mathbb{R}_+$, so the tuple of state variables is $x = (\mathbf{e}, a)$. The dynamics $\phi : \mathbf{X} \to \mathbf{X}$, $x_{t-1} \mapsto x_t = (\phi(x_{t-1}))$ are defined by

$$\begin{aligned} a_t &= 0 \; \forall t > 0 \\ \forall l \in \{1, ..., m\} : e_{l,t} &= e_{l,t-1} - \tfrac{a_{t-1}}{m} \end{aligned} \tag{4.13}$$

After one iteration the system has converged to its fixed point.

Generating Transactions

Transactions are determined as outlined in Section 4.3.2.

4.5 Proportional Rationing

4.5.1 Economic Documentation

When conducting proportional rationing, the agents on the long side of the market are served according to the relative size of their plans. An agent that demands (supplies) five percent of the aggregate demand (supply) obtains five percent of the available supply (demand). Such a rationing scheme can be manipulated, as shown by Benassy (2002), resulting in overbidding. Agents can increase the share they receive by strategically overstating their preferences.

4.5.2 Mathematical Documentation

Each agent obtains a share proportional to his plans. Only one iteration needs to be conducted until the system converges to the fixed point. The proportion that each agent obtains is determined by dividing each agent's plan e_l by the sum of the plans of all agents. The state space of this system is given by

$$\mathbf{X} : \mathbb{R}_+^m \times \mathbb{R}_+, \tag{4.14}$$

where $\mathbf{e} \in \mathbb{R}_+^m$, and $a \in \mathbb{R}_+$. The state variables of the system are given by $x = (\mathbf{e}, a)$, and the function of the dynamics $\phi : \mathbf{X} \to \mathbf{X}$, $x_{t-1} \mapsto x_t = \phi(x_{t-1})$ is defined by:

$$\begin{aligned} &a_t := a_{t-1} - \textstyle\sum_{l \in \{1,...,m\}} e_{l,t-1}, \\ &\forall l \in \{1, ..., m\} : e_{l,t} := e_{l,t-1} - a_{t-1} \frac{e_{l,t-1}}{\sum_{l \in \{1,...,m\}} e_{l,t-1}}. \end{aligned} \tag{4.15}$$

Generating Transactions

Transactions are determined as outlined in Section 4.3.2.

4.6 Ranking Rationing

4.6.1 Economic Documentation

The ranking rationing method depicts how labour units could be allocated to the agents on the labour market. Usually, e.g. in Germany and the EU over recent decades, unemployment is widespread but variable (see Eurostat (2006)). There may be exceptions on sub-markets, such as the IT market in the 1990s, but for the *lagom* context it will be assumed that there is usually an excess supply of labour.

The following outline refers to a labour market but can be applied to any market where some kind of quality/priority indicator plays an important role. The quality indicator assumed for labour supplies is qualification, but it could be any kind of indicator that allows for a ranking procedure. It is assumed that there is a population of agents who wish to work and offer some units of labour. On the other side of the market, one or several employers need to assemble a workforce, ideally, the best qualified one.

The ranking method allocates the available units of labour to the most qualified labour suppliers. The labour suppliers obtain – in the order of their qualification – the lesser of the remaining labour units and their labour supply. Once all units of labour have been allocated the rationing process is complete. The agents remaining with unsatisfied labour supplies do not have adequate qualification to be hired at the current level of labour demand. In order to conduct this method it is assumed that the qualification of the agents can be inferred correctly by the employer(s). However, on some labour markets, labour supply could be the short side of the market, and then the offered labour units are allocated to the employers by means of a queuing method (as in Section 4.3).

4.6.2 Mathematical Documentation

Ranking

There is an ordered list $\mathbf{w} =< w_1, ..., w_m >\in \mathbb{R}_+{}^m$, where each element indicates the qualification of the corresponding labour supply from $\mathbf{e} =< e_1, ..., e_m >\in \mathbb{R}_+^m$. There is an amount a to be distributed among the m agents who each state a labour supply e_l. The agents are ranked according to their qualification w_l, and the most qualified agent receives the lesser of what he supplies and what is demanded. The process continues until all units a have been distributed and the system has converged to its fixed point. I denotes that index of the unsatisfied labour supply that is associated with the highest qualification. It is assumed that no two agents have exactly the same qualification, so it is guaranteed that there is always only a single I. The state space is defined by:

$$\mathbf{X} = \mathbb{R}_+^m \times \mathbb{R}_+ \times \mathbb{N} \tag{4.16}$$

where $\mathbf{e} \in \mathbb{R}_+^m$, $a \in \mathbb{R}_+$, and $I \in \mathbb{N}$. The tuple of state variables is $x = (\mathbf{e}, a, I)$, and the function of the dynamics of the system $\phi : \mathbf{X} \to \mathbf{X}$, $x_{t-1} \mapsto x_t = \phi(x_{t-1})$ is

$$\begin{aligned}
&I_t = \min\{l \,|\, e_{l,t-1} > 0, \wedge \forall j = \{1, ..., m\} : w_j \leq w_l\} \\
&a_t := a_{t-1} - \min(a_{t-1}, e_{I_t,t-1}) \\
&\forall l \in \{1, ..., m\} : e_{l,t} := \begin{cases} e_{l,t-1} - \min(e_{l,t-1}, a_t) & \text{for } l = I_t \\ e_{l,t-1} & \text{else} \end{cases}
\end{aligned} \tag{4.17}$$

At each iteration t, the supplier from $\mathbf{e}$ associated with the largest productivity given by $\mathbf{w}$ is determined. This supplier obtains the lesser of his supply $e_{l,t-1}$ and the remaining units a_t. This process continues until $a_t = 0$

Generating Transactions

Transactions are determined as outlined in Section 4.3.2

Queuing

If, however, the demand side of the market is the short side, then queuing rationing takes place. The employers are served in the order indicated by the list **e**. Note that this procedure is specific for the application to labour markets within the *lagom* context. It may be modelled differently. The queuing process is identical to the one outlined in Section 4.3.

4.7 Pigeonhole Rationing

4.7.1 Economic Documentation

This method has been developed for labour markets within the *lagom* model family. It is able to depict an employer's inaccurate perception of the abilities of applicants. Usually, labour suppliers (i.e applicants) are screened for their abilities, and the most qualified applicants obtain employment. Such a procedure can easily be depicted by application of a ranking rationing.

However, if one wants to depict employers who inaccurately perceive the qualification of their applicants, one needs to modify the classical ranking method. Similar to ranking rationing, under pigeonhole rationing, each applicant has a specific productivity. But the employers are not able to perceive these productivities correctly. There can be many possible reasons for this: it may be too expensive to obtain the necessary information, they are simply not able to assess the applicants correctly, or do not care too much and apply a rule of thumb. Under pigeonhole rationing, the employers pigeonhole the applicants, and applicants that have been put into the same pigeonhole are perceived as being equally qualified. The larger the pigeonholes, the larger the inaccuracy. After pigeonholing the applicants, an employer ranks the pigeonholes in descending order. He chooses his workforce by hiring applicants from the best pigeonhole first. If there are applicants with more supply in a pigeonhole than units of labour are to be distributed, he will choose randomly among them until he will have allocated all units that have been available by allocating to each chosen agent the lesser of this agent's plan and the units available. Consequentially, pigeonhole rationing is a combination of ranking rationing and random rationing. The smaller the pigeonholes, the more this method resembles ranking rationing, and the larger the pigeonholes the more random the allocation process is.

Of course this method is not exclusively applicable to the labour market. It can be applied to any market when signs of priority are applicable. Agents within the same pigeonhole exhibit the same priority and, if their priority is sufficient, are being served in random order. In short, when applied to any market, the method works like this: The agents are pigeonholed, and the pigeonholes are sorted in descending order. At each iteration, the pigeonhole with the maximum priority is determined. An agent in this category, exhibiting a positive plan, will be picked randomly and receives the lesser of his plan and the remaining amount to be allocated. Another is then chosen and so on. If something is left after having allocated to all agents from one pigeonhole, it is then the turn of the next pigeonhole.

4.7.2 Mathematical Documentation

There is an amount a to be distributed among m agents, where each of these agents offers labour e_l and a qualification $v_l \in \mathbb{R}_+$. The list of agents supplying labour is given by $\mathbf{e} =< e_1, ..., e_m >\in R_+$, the set of qualifications is given by $\mathbf{v} =< v_1, ..., v_m >$, but the employer incorrectly perceives the qualifications as $\mathbf{w} =< w_1, ..., w_m >\in \mathbb{R}_+$. At each iteration the highest perceived qualification is determined. $\text{argmax}_w := \{l \,|\, \forall_j \in \{1, ..., m\} : w_j \leq w_l\}$ denotes the set of indices corresponding to the maximum qualifications as perceived by the employer. There can be more than one maximal qualification, depending on the employer's perception. If there is more than one candidate exhibits this highest perceived qualification, the index of the element that will be served is chosen using a random variable ω. The lesser of that supply and the amount currently available it is allocated to the corresponding labour supply. This is repeated until all units have been allocated. The state space of this system is

$$\mathbf{X} = \mathbb{R}_+ \times \mathbb{R}_+^m \times \mathbb{R}_+^m \times \mathbb{N} \tag{4.18}$$

with $a \in \mathbb{R}_+$, $\mathbf{e} \in \mathbb{R}_+^m$, the qualifications $\mathbf{w} \in \mathbb{R}_+^m$, and $\omega \in \mathbb{N}$. The tuple of state variables is given by $x = (\mathbf{e}, \mathbf{w}, \omega)$ The function of the dynamics' of the system $\phi : \mathbf{X} \to \mathbf{X}$, $x_{t-1} \mapsto x_t = \phi(x_{t-1})$ is defined by

$$\text{choose } \omega_t \in \text{argmax}_{w_{t-1}} \text{ randomly,}$$

$$a_t = a_{t-1} - \min(a_{t-1} - e_{\omega_{t-1},t-1})$$

$$\forall l \in \{1,...,m\} : e_{l,t} := \begin{cases} e_{\omega_t,t-1} - \min(e_{\omega_t,t-1}, a_t) & \text{if } l = \omega_t \\ e_{l,t-1} & \text{else} \end{cases} \tag{4.19}$$

$$\forall l \in \{1,...,m\} : w_t := \begin{cases} w_t & \text{if } l \neq \omega_t \\ 0 & \text{else}^5 \end{cases}$$

Generating Transactions

Transactions are determined as outlined in Section 4.3.2.

[5]Note, that the qualification of the agent being served is not set to zero, but the perception of the employer regarding this agent (reflecting that this agent does not play a role in the allocation process any longer.). It is then possible to avoid allocating to the same agent repeatedly.

Chapter 5

Applications and Simulations

The rationing modules introduced in the previous chapter enable the modelling of rationing on arbitrary markets in a modular fashion. This chapter demonstrates that rationing methods are not only applicable for rationing as such, but that they can help to create a framework for modelling heterogenous agents that act on a labour market. Subsequently it will be shown (a) how heterogenous job seeking agents determine their *probability of obtaining employment conditional on application* (PPE) under different recruitment methods, (b) how these agents learn simultaneously about the application behaviour of their competitors by observing each other, and (c) how average productivities of workforces differ when an employer recruits at different accuracies of perception of the applicants' qualifications.

5.1 Introduction

5.1.1 Outline

This chapter investigates a fictitious labour market on which a number of vacancies are available and a population of potential employees needs to decide whether to apply for employment or not. Three issues are to be investigated:

1. How can agents determine their perceived probability of obtaining employment conditional to application (PPE)?
2. How can a learning process be implemented in which heterogeneous agents observe each others' application behaviour and learn from it in the course of action?
3. How do the productivities of workforces that have been recruited according to different levels of accurate perception differ? (Does it pay off for an employer to perceive applicants' qualifications correctly?)

Firstly, I will derive methods to determine the PPE for three different rationing methods which are conceivable on a labour market. This will prepare for the investigation conducted in the second and third part.

Following this, a learning process on the labour market is modelled. During the learning process, agents apply the methods that have been derived in part one to to determine their individual PPE. The agents need to decide whether to apply. They are heterogenous with respect to their productivity and the initial prior second-order probabilities[1] which they assign to an invariant set of static probabilistic hypotheses regarding their competitors' application behaviour. These initial prior second-order probabilities are revised by observing the application behaviour of competitors over a sequence of job opportunities, i.e. the agents learn simultaneously by observing each other. The information gained is used to refine their individual PPE, since the PPE depends on the second-order probabilities of

[1] In Section 3.2.4 second-order probabilities were presented.

the hypotheses. The learning process is implemented via Bayes rule.

Part three presents a model that simulates, over a sequence of job opportunities, the decision making process of potential applicants, the recruitment process, the updating of prior second-order probabilities, and the determination of the average productivity of the recruited workforce will be presented. This model will be executed at different levels of accuracy of perception of applicants' qualifications in order to gain information about how the average productivity of a workforce depends on this accuracy.

5.1.2 Terminology & Assumptions

Several concepts will be used throughout this chapter and are the following:

agent	a labour supplier that seeks employment
applicant	an agent who applies for employment
employer[2]	an institution that allocates a given number of vacancies to applicants
job / job opportunity	a number of vacancies
pool size / pool	a number of applicants for a job opportunity
productivity	measure that depicts an agent's ability/qualification
competitors	all applicants except the observed agent
state	a combination of productivities in a pool (excluding the agent in question)
pigeonhole	a range of qualifications within which agents are perceived to be equally productive
characteristic	the productivity of an agent relative to the observed agent; can be more, less, or equal

Assumptions are (unless noted differently):

⋆ wages are fixed and given

[2]This employer can alternatively be interpreted as a central employment agency, as in Pissarides (1979), which conducts recruiting regarding a specific rationing method and distributes employers to different firms with vacancies. In contrast to Pissarides (1979), the agents do not perform random search but rely solely on the central agency.

- ⋆ agents only apply when their PPE exceeds some critical probability
- ⋆ the recruitment method of the employer is implemented as a rationing method
- ⋆ the recruitment method is common knowledge
- ⋆ an agent knows his own productivity
- ⋆ the distribution of productivities over the population is common knowledge
- ⋆ the number of agents is common knowledge
- ⋆ job opportunities occur consecutively
- ⋆ employment lasts for one job
- ⋆ an agent holds several hypotheses regarding his competitors' application behaviour and assigns an initial second-order probability (belief) to each of his hypotheses (parts 2 and 3 of this chapter)
- ⋆ an agent can, ex post, observe how many other agents applied for employment (parts 2 and 3 of this chapter).

5.2 Perceived Probabilities of Obtaining Employment Conditional on Application

5.2.1 Outline

Vacancies on the labour market are usually assumed to be given to persons who qualify for the position and have the ability to fulfill the tasks associated with it. An employer will be interested in choosing the most qualified candidates from a number of applicants. Based on his observation ability/willingness, he perceives the qualifications of applicants more or less correctly. In the present context, the recruitment behaviour of an employer is expressed as a rationing method, and the focus will be on such methods that can account for differing levels in the accuracy of perception. The rationing methods can be interpreted as a specification of a matching function, as known from search theory (see Merz (2002) for an overview over recent developments in search theory) where rationing is, in contrast to Walrasian models, at the very heart of the model (cf. Burdett and Wright (1998)). The rationing methods applied here were introduced in Chapter 4, and are *ranking rationing, random rationing* and *pigeonhole rationing*.

If the employer were able to perceive agents' qualifications correctly, he would allocate positions with a ranking method. At the other extreme, where the employer can not differentiate at all between qualifications, he allocates positions randomly. Pigeonhole rationing can account for both the extreme cases and those in between and is the third method to be investigated. The figure below illustrates the relation between the three rationing methods.

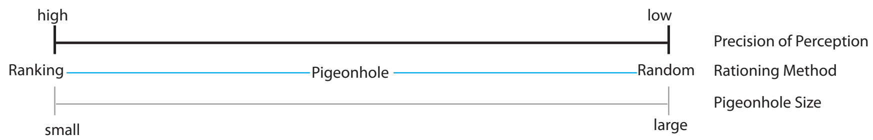

Figure 5.1: Relation between Ranking Rationing, Random Rationing and Pigeonhole Rationing with Respect to Precision of Perception and Pigeonhole Size

These three rationing methods were introduced in Chapter 4.

As in Pissarides (1979) it is assumed that wages are non-negotiable, i.e. agents and firms are wage-takers. Usually, as for example in Weiss (1976, 1980) and Mookherjee (1987), wages are assumed to be the incentive for agents to apply for employment. But since in the present context wages are assumed to be non-negotiable, another incentive for application needs to be considered. Thus, it will be assumed that agents apply for employment only if their perceived probability of obtaining employment conditional on application (PPE) exceeds some critical value. An employer will be assumed to be able to recruit his workforce according to the three rationing methods illustrated in Figure 5.1. This part shows how an agent can determine his PPE under these three recruitment methods.

Several axioms should be fulfilled by the results gained with each of the methods, for example the PPE should always be positive and never exceed the value one. The axioms in question are introduced with the methods to compute the PPE. In order to illustrate that the axioms are fulfilled, it will be investigated (exemplarily) how the PPE varies with the number of agents in the population, the number of vacancies available, and the agent's productivity.

Consider a population of agents $i = 1, ..., n$, and each agent i has an individual productivity a_i. Productivity does not necessarily and uniquely depict output per unit time, but can also reflect other factors, as a synonym for qualification. It is assumed that the productivity distribution $F(a)$ over the population is common knowledge. Since an agent i knows his own productivity, he knows which area of the productivity distribution his abilities lie in. There is no evidence of how productivities/abilities are actually distributed in. a real population. However, it can be observed that there are many moderately educated people, some poorly educated and some highly educated ones in a country like Germany (see Sozialpolitik-Aktuell (2005)). Such stylised facts may be depicted by a variety of continuous probability density functions that are bounded below by zero.

An agent i who wants to determine his PPE needs information about his own productivity a_i, the productivity density of the population $F(a)$, the number of vacancies v, the number of agents in the population n, and the rationing method R used by the employer. Where some information is not available he must make guesses.

Under ranking rationing, an agent needs to compute how likely it is that a sufficient number of agents has a productivity less than his own. Under pigeonhole rationing, the productivities of two agents are put in the same class if they both lie within a certain interval. The probability of obtaining employment then depends on the size of these intervals. Finally, under random rationing, the probability of obtaining employment is the same for each agent, and it does not depend on his productivity.

5.2.2 Productivity Distribution

The productivity of the population is assumed to be distributed regarding a density function that is bounded below by zero because productivities less than zero should not exist in the present context, i.e there are no destructive employees. The Weibull density function ($\tilde{W} = ab^{-a}e^{-\left(\frac{x}{b}\right)^a}x^{a-1}$, (cf. Mathworld (1999))) is a probability density function that exhibits this characteristic (see Wikipedia (2006b)), and in this section, productivities are assumed to follow this probability density function with the parameters set to $a = 3$ and $b = 2$. Figure 5.2 illustrates the shape of this density function.

5.2.3 On Pool Sizes and Possible Pool Sizes

At a specific job offer, a pool of applicants will emerge and be characterised by the number of agents in it (denoted *pool*, *pool size*). However, when an agent calculates his PPE, he can not know in advance which pool size will emerge, but he needs to consider all pools, since for each possible pool his probability to obtain employment will be different. The sample space of pools is $\Theta = \{\theta_1,...,\theta_n\}$, a specific pool θ_i is a singleton from the sample space, and the probability that any pool from Θ emerges is $P(\Theta) = 1$, while $P(\emptyset) = 0$.

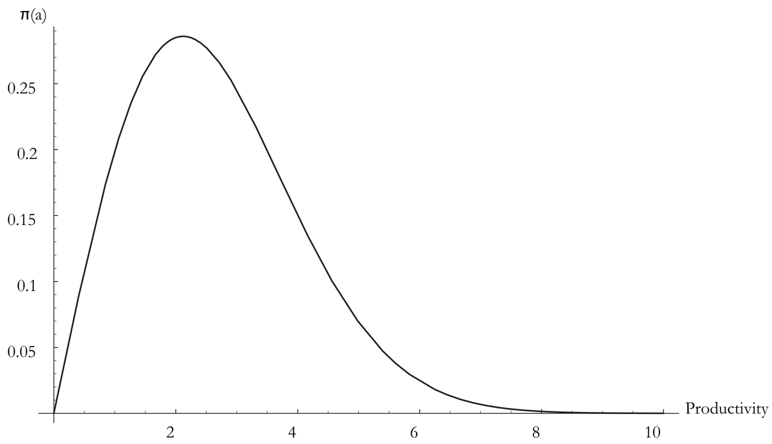

Figure 5.2: Weibull Probability Density Function ($\tilde{W} = ab^{-a}e^{-\left(\frac{x}{b}\right)^a}x^{a-1}$; $a = 3$, $b = 2$)

For the remainder of this section, pool sizes of applicants are assumed to be uniformly distributed. This is done for simplicity reasons. Each of the three methods to compute PPEs that are derived below is capable of working with arbitrary distributions, but calibrating them becomes more complicated and is deferred until later in this chapter.

5.2.4 Perceived Probability of Obtaining Employment Conditional on Application under Ranking Rationing

An agent who determines his PPE under ranking rationing needs to determine the probability that he fulfills the productivity requirements in each of the possible pools of applicants, and, thus, he needs to determine

- ⋆ the probability of each possible pool size
- ⋆ in each possible pool size:
    - ○ the probability of obtaining employment

The probability of obtaining employment in a specific pool is the probability with which a sufficient number of agents has a productivity lower than that of the agent in question.

If a pool consists of 5 applicants (including the agent determining his PPE) and 3 vacancies are available, 2 applicants would need to be less productive than the agent who determines his PPE so that he obtains employment. This implies that 2 agents can be more productive than him, as they would obtain employment, leaving one vacancy which he obtains (if and only if 2 agents are less productive than him).

The PPE for any agent is the cumulated weighted probability of obtaining employment in each possible pool size:

$$\sum_{\theta=1}^{n} P(\theta)\left(P(a_j < a_i)^{\theta - v}\right) \tag{5.1}$$

with

$$\theta - v = \begin{cases} \theta - v, & \text{if } v < \theta \\ 0, & \text{if } v \geq \theta \end{cases}$$

$$\text{and} \quad P(a_j < a_i) \quad = \int_0^{a_i} F(a)da$$

where:

n	number of possible pools,
$P(\theta)$	probability of pool size θ,
v	number of vacancies,
$P(a_j < a_i)$	probability of exhibiting a productivity less than a_i.

The number of possible pools n is equal to the number of agents in the population. The probability of pool size θ, $P(\theta)$, needs to be substituted into Formula 5.1. It is either the value of the distribution function of pool sizes at the appropriate value, or, with a continuous density function, the integral depicting the corresponding pool size (see Section 5.3.2 for the latter case). At the moment the pools are still assumed to be distributed uniformely (an assumption that will be dropped later in this chapter), so at the moment $P(\theta) = \frac{1}{n}$.

The formula can be justified by the following thoughts. From the Kolmogorov axioms it is possible to deduce, inter alia, the addition rule for computing the joint probability of disjunct events A and B, $P(A \cup B) = P(A) + P(B)$ (compare Reinhardt and Soeder (1998, p. 467)). The probability that events A and B happen is, $P(A \cap B) = P(A) \cdot P(B)$, for independent events. It is assumed that the events, observing any agent from a pool of applicants, are independent. Formula 5.1 takes into account that there are possible states (combination of competitors' productivities in a specific pool) that have a larger probability because the order in which the competitors in this pool appear does not matter, so that all permutations of these states need to be considered, too. The following example illustrates the permutation issue: A dice is rolled $q = 2$ times, giving results x_1 and x_2. What is the probability that, both times, a number smaller than three is the outcome? The possibilities are

$$P(x_1, x_1 < 3) = P(1,1) + P(1,2) + P(2,1) + P(2,2) \tag{5.2}$$

The requirement is fulfilled if one of the above events of $(x1, x2)$ happens. The probability that one of these events occurs is computed by the above formula. It can be rearranged to yield

$$P(x_1, x_2 < 3) = P(1)P(1) + P(1)P(2) + P(2)P(1) + P(2)P(2) \tag{5.3}$$

The probability of rolling a one and a two is larger than rolling two ones or two times two, since it is possible to roll a one first and then a two, or first the two and then the one. These are specified as two disjunct events (with the same outcome). The order in which they happen, either 1, 2 or 2, 1 does not matter, and, thus, the probability is larger than rolling two times the same number. 5.3 can be simplified to:

$$P(x_1, x_2 < 3) = (P(1) + P(2))(P(1) + P(2)). \tag{5.4}$$

This can be rewritten to yield:

$$P(x_1, x_2 < 3) = P(x_1 < 3) \cdot P(x_2 < 3) = P(x_\phi < 3)^q. \tag{5.5}$$

The same argumentation holds for computing the probability that a sufficient number of applicants has a productivity less than the agent who is determining his PPE, if one assumes these events to be independent. Additionally, since the agent in questions needs to know how many agents have to be less qualified than he is, the number of vacancies must be subtracted from the number of applicants in the current pool ($\theta - v$). If $\theta - v$ applicants are less able than the agent in question, he will obtain employment for this specific pool.

The results obtained by application of Formula (5.1) should fulfill the following three axioms:

⋆ The PPE decreases with increasing number of agents and converges towards zero

⋆ The PPE increases with increasing number of vacancies and converges towards one when there are as many or more vacancies than agents

⋆ An increase in the productivity of an agent results in a larger PPE, converging towards one

The results gained with Formula 5.1 have been analysed for a large variety of settings and are found to satisfy all axioms. Figures 5.3-5.5 illustrate this for specific settings.

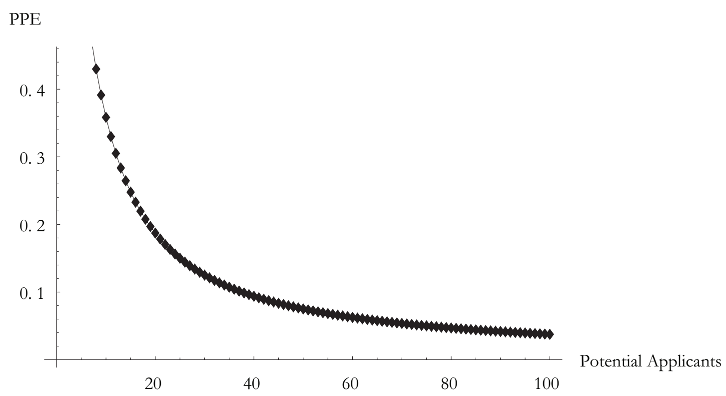

Figure 5.3: PPE under Ranking Rationing Depending on the Number of Competitors; Productivity of Agent: 3.45, Potential Applicants: 1-10, Vacancies: 1

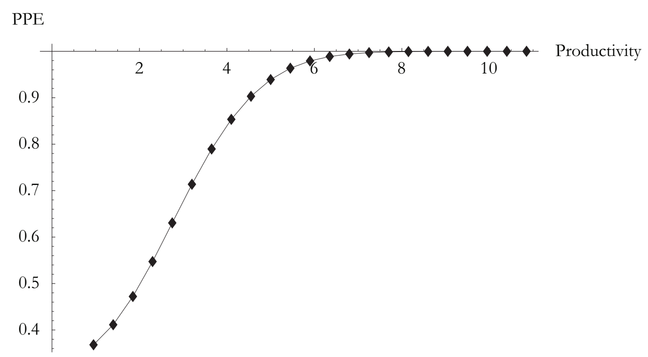

Figure 5.4: PPE under Ranking Rationing Depending on an Agent's Productivity; Productivities: 1-11, Potential Applicants: 3, Vacancies: 1

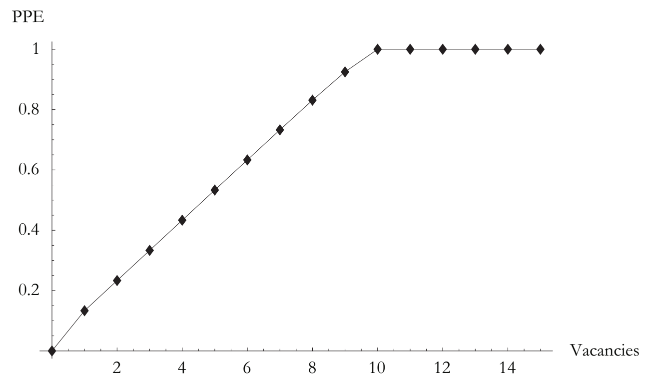

Figure 5.5: PPE under Ranking Rationing Depending on the Number of Vacancies; Productivity of 3.45, Potential Applicants: 10, Vacancies: 0-13

5.2.5 Perceived Probability of Obtaining Employment Conditional on Application under Pigeonhole Rationing

The inaccurate perception of an employer can be depicted by the size of pigeonholes. The smaller a pigeonhole, the more precise the employers' perception of applicants' productivities. In the present context it is assumed that an agent knows the size of the pigeonholes and which pigeonhole his productivity belongs to. With additional information about the productivity distribution in the population and the size of the population, he is able to determine his PPE. The pigeonholes from an potential applicant's point of view[3] can be illustrated like this:

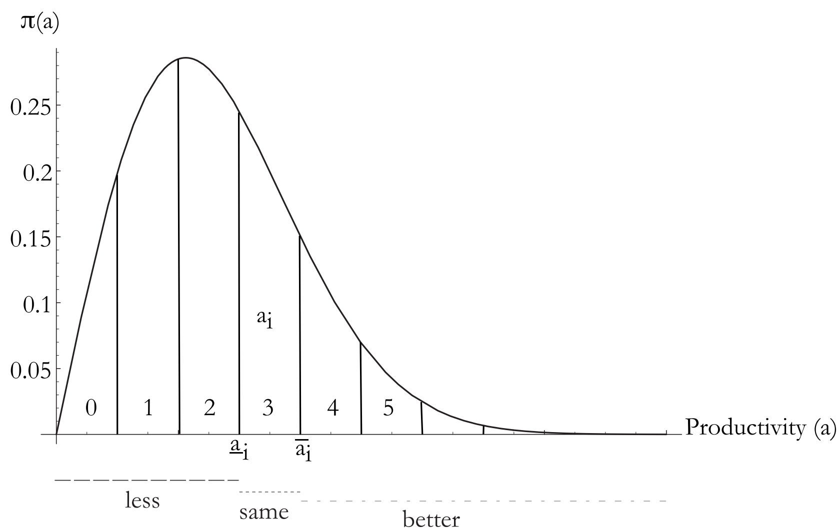

Figure 5.6: Productivity Density Function of Population ($\tilde{W} = ab^{-a}e^{-\left(\frac{x}{b}\right)^a}x^{a-1}$; $a = 3, b = 2$) with Pigeonholes (with their indicators 0-5) from a Potential Applicant's Point Of View

An agent who needs to determine his individual PPE needs to find out

- ⋆ the probability of each possible pool size
- ⋆ for each possible pool size:
  - ∘ the probability of each possible state
  - ∘ for each possible state:
    - ∗ the probability of obtaining employment

He needs to consider all possible sizes of applicants pools and all possible states in each possible pool size. A state is a specific combination of the competitors' characteristics in a certain pool. The characteristics are expressed relatively to the observed agent and can be: less productive (l), same productive (s) or better productive (b) (see Figure 5.6 for the specification of the characteristics). The number of states in a specific pool θ is

[3]From an employer's point of view the same pigeonholes apply, but for the distribution of the applicants' productivities. This may very well be a different (depending on how many agents apply) than for the whole population.

given by

$$st_\theta = \frac{(r + k_\theta - 1)!}{k_\theta!(r-1)!} \tag{5.6}$$

with:

r number of characteristics (i.e 3), and

k_θ number of competitors in a specific pool θ.

Some states have a larger probability than others because the order in which the competitors appear is not relevant, therefore the number of instances $\#(z_\theta)$ of a specific state must be determined and is given by the number of permutations of the characteristics in that state. A state contains $\theta - 1$ agents; it does not include the observed agent. The number of instances of a specific state is

$$\#(z_\theta) = \frac{k_\theta!}{n_l!n_s!n_b!} \tag{5.7}$$

where

z_θ a state at pool size θ,

θ pool size (includes observed agent),

n_b number of competitors in the pool with better productivity b,

n_s number of competitors in the pool with same productivity s, and

n_l number of competitors in the pool with less productivity l.

For a total of v vacancies the probability of obtaining employment e, $P_{z_\theta}(e)$, in a specific state is determined by dividing the remaining vacancies $v - n_b$, by the number of agents that are perceived as equally productive as the agent in question.

$$P_{z_\theta}(e) = \begin{cases} \frac{v-n_b}{\theta-n_b-n_l} & \text{for } v \geq n_b \\ 0 & \text{for } v < n_b \end{cases} \tag{5.8}$$

An agent determines his overall chances of obtaining employment by computing the probability of obtaining employment in each possible state of each possible pool, where he needs to consider all instances of each state. The sum of the probabilities of obtaining a position in each pool is the overall perceived probability of a successful application and is expressed by formula (5.9);

$$\sum_{\theta=1}^{n} P(\theta) \left(\sum_{z=1}^{st_\theta} \left[\left(\prod_{\rho=1}^{k_\theta} P(c_\rho) \right) \#(z_\theta) \right] P_{z_\theta}(e) \right). \tag{5.9}$$

with:

n	number of pools,
$P(\theta)$	probability of pool size θ,
z	state number z,
ρ	competitor ρ,
k_θ	number of competitors in pool θ,
$P(c_\rho)$	probability of characteristic of competitor ρ.

In each pool size, for each state, the product of the probabilities of the competitors' characteristics $P(c_\rho)$ is determined ($\prod_{\rho=1}^{k_\theta} P(c_\rho)$) and multiplied by the number of instances of that state $\#(z_\theta)$. The result is then multiplied by the probability of obtaining employment in that state $P_{z_\theta}(e)$. The sum of the probabilities of obtaining employment in each of the pools gives the overall probability of obtaining employment with a given number of competitors and vacancies.

In order to substitute the probability of a pool size $P(\theta)$ into the formula, assumptions about the probability distribution of pool sizes need to be made, as there is no common knowledge about the true probability distribution.

The characteristic of the current competitor is substituted for $P(c_\rho)$, where c can be either l, s or b, and

$$\begin{aligned} P(l) = &\quad \int_0^{\underline{a_i}} F(a)da \\ P(s) = &\quad \int_{\underline{a_i}}^{\bar{a}_i} F(a)da \\ P(b) = &\quad \int_{\bar{a}_i}^{\infty} F(a)da \end{aligned} \tag{5.10}$$

where:

$F(a)$	probability density function of productivities,
$\underline{a_i}$	the lower productivity-limit of agent i's pigeonhole,
$\bar{a}_i$	the upper productivity-limit of agent i's pigeonole.

It has to be determined whether the results are consistent with the following axioms:

- ⋆ The PPE decreases with increasing number of agents and converges towards zero
- ⋆ The PPE increases with increasing number of vacancies and converges towards one when there are at least as many vacancies as agents
- ⋆ As the productivity of an agent increases, his PPE increases or stays the same
- ⋆ PPEs are the same for productivities within a pigeonhole

The results gained with Formula 5.9 have been analysed for a large variety of settings and are found to satisfy all axioms. Figures 5.7-5.9 illustrate this for specific settings. Figure 5.8 illustrates clearly how, within a pigeonhole, the PPE remains the same.

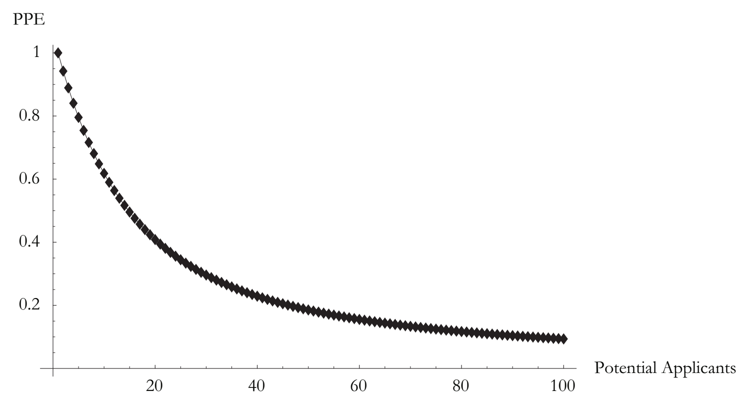

Figure 5.7: PPE under Pigeonhole Rationing Depending on the Number of Competitors; Productivity of Agent: 4.35 Number of Agents; 0-100, Vacancies: 1, Pigeonhole Size: 1.0

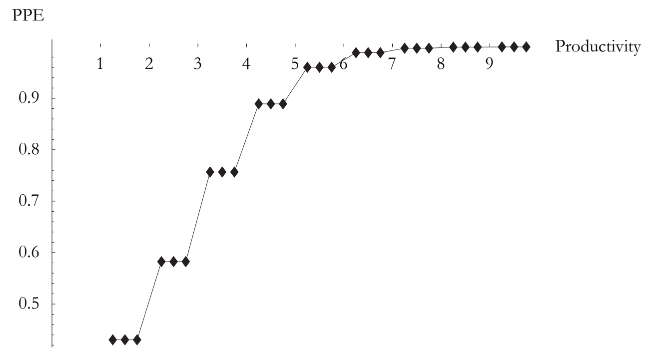

Figure 5.8: PPE under Pigeonhole Rationing and a Varying Productivity; Productivity of Agent: 0-10, Number of Agents: 3, Vacancies: 1, Pigeonhole Size: 1.0

5.2.6 Perceived Probability of Acceptance Conditional on Application under Random Rationing

An agent does not need much information to determine his PPE when vacancies are filled randomly. The PPE is the same throughout the whole population of potential applicants, no matter what the productivity of an agent is. An agent needs to

- ⋆ determine the probability of each possible pool size
- ⋆ in each possible pool size:

- ◦ the probability of obtaining employment

Since the productivities of agents do not play a role under random rationing, the probability of an agent in a pool obtaining employment is simply the probability for the pool $P(\theta)$ multiplied by the number of vacancies v (or the size of the pool θ, if there are as many or more vacancies as agents in the pool) divided by the size of the pool θ, so that for n pools the sum probability of obtaining employment (which is the overall probability of obtaining employment) is:

$$\sum_{\theta=1}^{n} P(\theta)\frac{q}{\theta}, \tag{5.11}$$

with:

$$q = \begin{cases} v, & \text{if } v < \theta, \\ \theta, & \text{if } v \geq \theta. \end{cases}$$

The following axioms must be fulfilled for the formula to be validated:

- ⋆ The PPE decreases with increasing number of agents, tending to zero
- ⋆ The PPE increases with increasing number of vacancies, tending to one when there are at least as many ,vacancies as agents

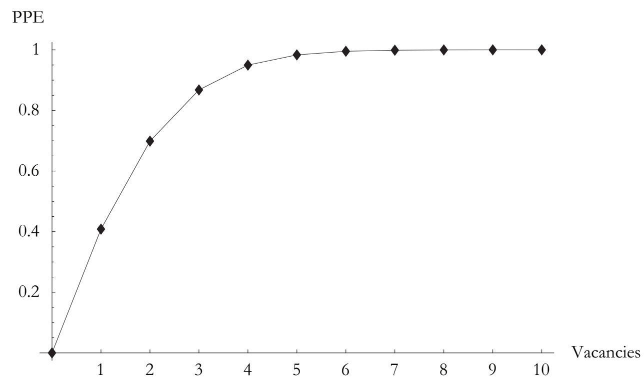

Figure 5.9: PPE under Pigeonhole Rationing Depending on the Number of Vacancies; Productivity of Agent: 4.35, Number of Agents: 20, Vacancies: 0-10, Pigeonhole Size: 1.0

The results gained with Formula 5.11 have been analysed for a large variety of settings and are found to satisfy all axioms. Figures 5.10 and 5.11illustrate this for specific settings.

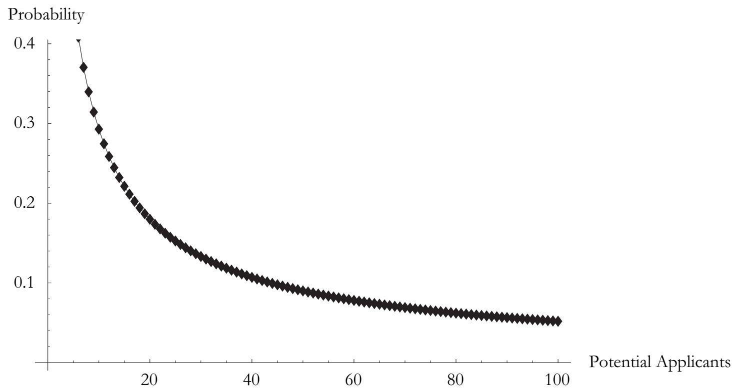

Figure 5.10: PPE under Random Rationing Depending on the Number of Competitors; Number of Agents: 0-100, Vacancies: 1

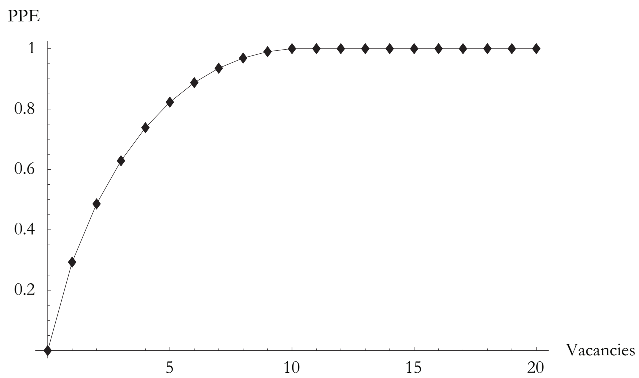

Figure 5.11: PPE under Random Rationing Depending on the Number of Vacancies; Number of Agents: 10, Vacancies: 0-20

5.2.7 Comparison of the PPE under Random Rationing, Ranking Rationing and under Pigeonhole Rationing

How does an agent perceive the probabilities of acceptance when an employer uses different recruitment methods? The remainder of this section compares, for a specific scenario, the PPEs under the three different recruitment methods: PPE under ranking rationing (PPERank), PPE at pigeonhole rationing (PPEPigeon) and PPE under random rationing (PPERand). All assumptions previously made are still assumed to hold. Figure 5.12 shows, for all three methods of recruitment, how the PPE of an agent with productivity 3.25 evolves as the number of agents in the population varies from one to forty, one vacancy is available, and pigeonholes reach from one to the next positive integer.

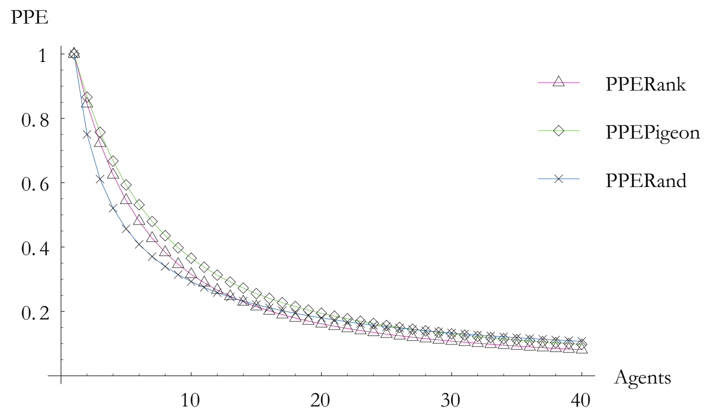

Figure 5.12: PPE under Ranking, Pigeonhole and Random Rationing at a Variation in the Number of Agents; Setting: Productivity: 3.25, Vacancies: 1, Number of Agents: 1-40, Pigeonhole Size: 1.0

All graphs confirm what individual validation has shown before: the PPE decreases with increasing number of agents. PPEPigeon is largest, followed by PPERand. The lowest PPE is obtained under ranking rationing. Interestingly, PPERand and PPEPigeon converge towards each other faster than PPEPigeon and PPERank. For all numbers of competitors, the agent with productivity 3.25 perceives the best chances of obtaining employment when the employer pigeonholes the applicants.

Figure 5.13 illustrates the PPEs at an increasing productivity of the agent. PPERand remains the same throughout all productivities because in its computation productivities simply do not matter. It has been assumed that ten agents face one vacancy, pigeonholes range from one integer to the next, and the productivity of the agent determining his PPE is varied from one to fourteen.

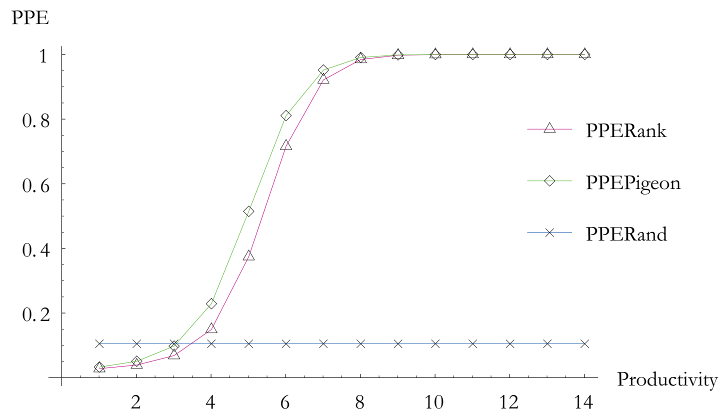

Figure 5.13: PPE under Ranking, Pigeonhole and Random Rationing, and a Variation of the Productivity of Agents; Setting: Agents: 10, Vacancies: 1, Productivity: 1-14, Pigeonhole Size: 1.0

In Figure 5.13, the PPEPigeon curve is offset to that of PPERanking. This is due to the size of the pigeonholes. As explained above, by a variation in the size of the pigeonholes, pigeonhole rationing can account for ranking and random rationing, too. In the limit, when pigeonholes tend to zero (infinity) and each agent is inferred correctly (as the same), pigeonhole rationing accounts for ranking rationing (random rationing). Figure 5.14 shows how the PPEs under pigeonhole rationing will converge to those under ranking rationing when the pigeonholes are very small. If pigeonhole sizes tend to zero, these two graphs will not be distinguishable any longer.

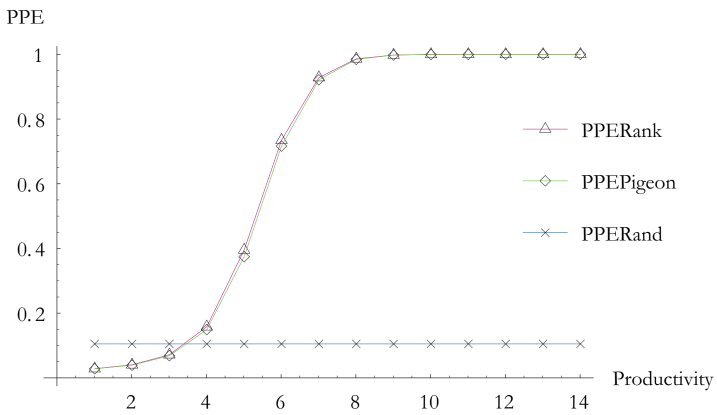

Figure 5.14: PPE under Ranking, Pigeonhole and Random Rationing, and a Variation of the Productivity of Agents; Setting: Agents: 10, Vacancies: 1, Productivity: 1-10, Pigeonhole Size: 0.15

Finally, figure 5.15 demonstrates how the PPEs behave as the number of vacancies varies.

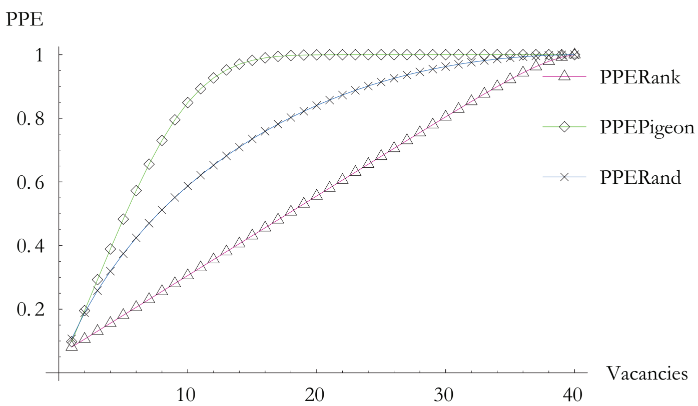

Figure 5.15: PPE under Ranking, Pigeonhole and Random Rationing, and a Variation of the Number of Vacancies; Setting: 30 Agents, Productivity: 3.25, Number of Vacancies: 1-30.

With an increasing number of vacancies, the PPEs under all methods increase. Again, the values of PPERandom lie between those of PPERanking and PPEPigeon, except at the origin of the series. The values of PPEPigeon are the largest throughout.

That the PPE under pigeonhole rationing is the largest depends on the choice of the size of the pigeonholes. The larger the pigeonholes, the more the PPE corresponds to those under random rationing, and the smaller the pigeonholes, the more will they correspond to those under ranking rationing.

Under certain circumstances, when the inaccuracy of perception has a certain level, agents perceive more chances of success than under more accurate perception of their qualifications. This indicates that, if the PPE was a criterion which agents use to decide whether to apply for employment, a certain inaccuracy might attract more applicants. The implications of this are considered in the next sections.

5.3 Agents, Bayes, and the Labour Market I: Learning by Observing

5.3.1 Introduction

Formulas 5.1, 5.9 and 5.11 derived in the previous section demonstrate how to determine the PPE for different recruitment methods. If one wants to model a whole population of agents who simultaneously determine their individual PPE in order to make application decisions, it would be unrealistic to let all of them adopt perfect foresight regarding their competitors' application behaviour. If each agent had this foresight, most probably only those agents would apply who receive employment. Human agents usually lack perfect foresight and this lack can be used to heterogenise agents. Each agent will then adopt personal assumptions regarding his competitors' application behaviour. The agents modelled below will be heterogenised in this way, and then also regarding their productivities.

More specifically, each agent will be modelled to have his own representation of the circumstances by holding three hypotheses about his competitors' application behaviour. Previous experiences on the labour market and reports from friends or media may have led agents to distinct beliefs about the probability of these hypotheses. An agent might consider all three hypotheses as relevant but not as equally likely. This can be accounted for by letting each agent initially assign a weighting-factor (initial prior second-order probability) to each hypothesis. The larger the weighting, the greater the agent's belief in the corresponding hypothesis. Each agent can determine his total individual PPE by determining his PPE at each hypothesis and weighting it with the corresponding prior second-order probability. The sum of these weighted PPEs is an agent's individual PPE.

In the following, the agents are not only heterogenised in this way, but also observe each others' application behaviour at consecutive job opportunities. The number of applicants at each job opportunity serves as new information which an agent uses to rephrase his current representation of his competitors' application behaviour. Using Bayesian methods, each agent updates his prior second-order probabilities for all his hypotheses with this new information. As job opportunities go by, representations change and agents refine the determination of their PPE by step-wise adaptation of the prior second-order probabilities of all hypotheses in the light of new information.

Each agent is characterised by his individual productivity. In Weiss (1980) and Mookherjee (1987), agents are characterised by their reservation wages and it is known that the reservation wage is positively correlated with the agents' abilities. Only if the offered wage exceeds their reference wage will agents apply for employment, so an employer can increase his chances of obtaining productive employees by increasing the wage rate, as then the number of (qualified) applicants increases. He then faces a pool of applicants, but cannot distinguish between more and less qualified applicants and needs to choose randomly from them (cf. Weiss (1980, p. 527)). Weiss (1980) extends the model and distinguishes between those characteristics of agents that can be observed at expense and those that can be observed without expense. A firm would then only observe the latter, and group the agents with same characteristics into classes. Workers from the same class would obtain a uniform wage. In the present context, a similar framework will be built, but the wage that is being paid does not play a role. Rather, it is assumed that a number of vacancies is announced at a fixed wage and, as in sectors where tariff wages are being paid[4], the wage is not subject to discussion. When this is the case, the wage rate is not a (unique) incentive for application, but the PPE becomes one.[5]

Similar to Weiss (1980), the present employer is not able to perceive the productivities of the agents correctly. In Cornell and Welch (1996) the level of incorrect perception depends on the number of signals an employer ob-

[4]In 2003, 62% of the employees in western Germany and 43% of the employees in eastern Germany were paid according to tariff wages (cf. IAB (2004)).

[5]Note that it is possible to translate the PPE into an expected wage by multiplying it with the wage rate. Additionally, the probability of not obtaining a position multiplied with the social benefits in case of unemployment could be added. Then each agent would compare his expected wage with his reference wage. But for the current exposition this does not make a difference and PPEs are assumed to be the incentive.

serves. The more signals he takes into account, the better he infers the qualities of individual abilities of the applicants. In the present exposition the employer arranges agents (together with their productivities) with similar productivities into the same category (pigeonhole), and a number of pigeonholes emerges. All agents with productivities within a pigeonhole are perceived to have the same qualification, and the better the perception of the employer, the more pigeonholes emerge.

In contrast to Weiss (1980) and Mookherjee (1987), the present employer ranks the applicants in descending pigeonholes. Positions are firstly allocated to applicants in the *best* pigeonhole. Only when there are no applicants left in this pigeonhole, but open positions remain, applicants from the next pigeonhole obtain employment. When there are more applicants with sufficient qualifications within a category than needed, vacancies are allocated randomly among them. With the categories (i.e pigeonholes), it is possible to model different levels of precision on the employer's side. If the employer was able to perceive agents abilities correctly, each agent together with his productivity would form a category, and the employer would allocate positions according to a ranking method. In the other extreme, where the employer can not differentiate at all between abilities, only one category emerges and he allocates positions randomly. Pigeonhole rationing is assumed to be applied by the employer in the subsequent model.

The labour suppliers of the present model learn in Bayesian manner by experiencing job opportunities that are assumed to happen consecutively. Bayesian learning has been an issue in literature dealing with labour markets, but mainly for different applications. Two broad categories can be sketched to exist: The first focuses on learning on the employer's side of the labour market, and the second focuses on learning on the employee/applicant side of the labour market.

Farmer and Terrell (1996), Lewis and Terrell (2001), and Feltovich and Papageorgiou (2004) model employers that learn about the abilities of their employees, and about the abilities of specific groups of employees. In Farmer and Terrell (1996) and Feltovich and Papageorgiou (2004), the employer entertains certain prior assumptions about the abilities of certain groups of workers (e.g. ethnic groups). At the beginning of the learning process individuals from a specific group are perceived regarding the initial priors that the employer entertains about that group. While Feltovich and Papageorgiou (2004) demonstrate how a Bayesian employer learns about group specific abilities, Farmer and Terrell (1996) let their employer learn additionally about the abilities of an individual employee by letting him observe individual output to update his prior assumptions accordingly. Lewis and Terrell (2001) focus on the learning process about individual abilities and how underestimated individuals can gain from such a process. If the employer finds that he underestimated (overestimated) an individual employee, he will increase (decrease) the wage of that worker.

Less literature seems to deal with Bayesian learning of employees. Breen and García-Peñalosa (2002) apply Bayesian methods to model male and female workers who need to make decisions about types of careers and therefore need to determine their chances of success at different occupations. Once success/failure is experienced, the agents update their priors and the resulting posteriors are bequested from father to son and from mother to daughter, thus leading to gender specific learning processes. Breen and García-Peñalosa (2002) find that, even when preferences of male and female workers become relatively similar, gender seggregation may persist. Murphy and Tam (2004) investigate decisions that applicants must make, but focus on their learning from new information about the organisation and the job.

It seems that there is not much literature applying Bayesian learning to the labour market in general and to the applicants' side in specific. The literature mentioned has in common that it focuses on the investigation of a single individual who learns about specific issues by observing. Here a different approach is adopted, inasmuch as there will be a number of heterogenous Bayesian agents (potential labour suppliers), each of whom observes his environment, which is in this case the other agents' behaviour, and uses the information to (a) update his individual representation of the environment, and (b) to use his updated representation of the environment to determine his chances of obtaining employment. The difference to the usual literature is not only that there are many heterogenous agents simultaneously conducting Bayesian methods, but that they observe each others' behaviour to update their priors. By doing so, each agent actively has an impact on the development of the environment he acts upon.

5.3.2 Building Blocks of the Model

Pools, Productivities and Probabilities

A pool θ is a number of actual applicants. Each pool θ has a probability $P(\theta)$ that is unknown, but in order to determine his individual PPE ppe_i, an agent i must take information about probabilities of pools into consideration. Therefore, he makes individual assumptions. Since an agent i does not know whether his assumptions are correct, he holds several of them. His assumptions are about the distributions of pool sizes. Each assumption is expressed as a hypothesis H_j regarding the probability density function (PDF) of pool sizes. Additionally, an agent i has some information at hand. This includes the size of the population n, the number of vacancies v available, the qualification distribution of the population (which follows a PDF $F(a)$), the rationing method R that the employer conducts, and his own qualification a_i. With this information the agent is able to determine his chance of obtaining employment:

$$ppe_i = \sum_{\theta=1}^{n} P(\theta) \left(\sum_{z=1}^{st_\theta} \left[\left(\prod_{\rho=1}^{k_\theta} P(c_\rho) \right) \#(z_\theta) \right] P_{z_\theta}(e) \right) \tag{5.12}$$

Since an agent i does not know the probability P of each pool size θ he forms hypotheses about how the pools might be distributed. And since he finds different assumptions reasonable, he considers several hypotheses $\mathbf{H} = \{H_1, ..., H_3\}$ about PDFs of pool sizes. By assigning individual initial prior second-order probabilities to each hypothesis H_j, an agent i expresses his beliefs in the different hypotheses. The larger an initial prior second-order probability $s_{j,i}$, the greater his belief in the corresponding hypothesis H_j. His PPE ppe_i now becomes the sum of the weighted PPEs, each gained with one of the hypotheses H_j regarding the underlying PDF:

$$ppe_i = \sum_{j=1}^{k} s_{j,i} \left(\sum_{\theta=1}^{n} P(\theta) \left(\sum_{z=1}^{st_\theta} \left[\left(\prod_{\rho=1}^{k_\theta} P(c_\rho) \right) \#(z_\theta) \right] P_{z_\theta}(e) \right) \right) \tag{5.13}$$

with:

$s_{j,i}$ prior second-order probability agent i assigns to hypothesis H_j.

Throughout the course of action, the agents observe each other and see what pool sizes actually come to be. Agents sequentially revise their prior second-order probabilities of their hypotheses with the help of new data – the actual pool size. Based on that information they refine their future computations. It will be observed how these individual weights develop over a sequence of job opportunities.

The productivities of the population are assumed to be Weibull distributed, since the PDF of the Weibull distribution can be calibrated to roughly depict the stylised fact that in Germany there are many moderately educated persons, some poorly educated and some highly educated persons (cf. Sozialpolitik-Aktuell (2005)).

Hypotheses

Each agent i possesses three hypotheses H_1, H_2, H_3 about how the pool sizes may truly be distributed. These hypotheses need not necessarily include the true PDF. At the beginning of the learning process, each agent i assigns individual initial prior second-order probabilities $s_{j,i}$ to each of the hypotheses from $\mathbf{H}$. These weights sum up to one and the weight of a hypothesis H_j expresses an agent's belief in this hypothesis. He may have assembled these beliefs through past experiences and/or reports from others that have experience on the labour market – essentially they are based on his best knowledge. The larger the weight of a hypothesis H_j, the higher the agent's belief in it. All agents possess the same hypotheses, but assign different individual weights $s_{j,i}$ to them in the beginning of the updating process. Figure 5.16 illustrates three hypotheses regarding pool sizes for a population with 5 agents.

Naturally, a pool of applicants θ consists of whole agents only. However, it is in fact easier to calibrate a pool size distributions with continu-

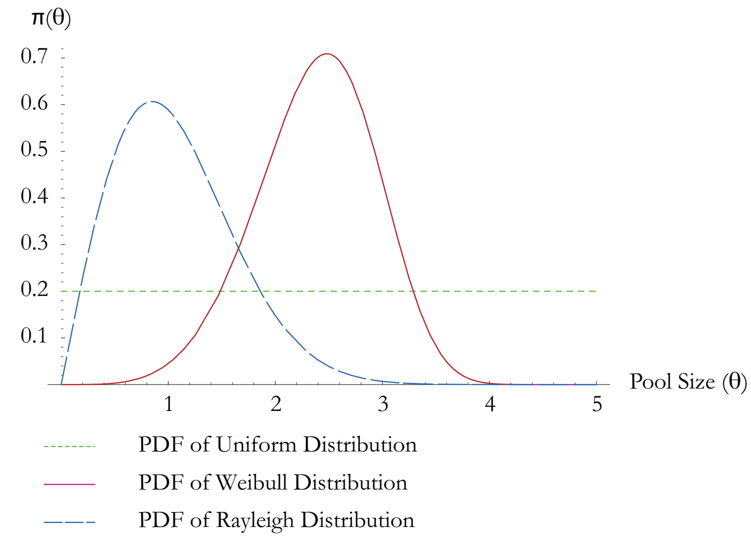

Figure 5.16: Possible Hypotheses Regarding Pool Size Probabilities for a Population with 5 Agents

ous probability density functions than with discrete distribution functions[6]. The probability of a pool size θ is then obtained by integrating over the interval that represents a specific pool size. As a consequence, the PDFs are discretised by the use of non-overlapping intervals, each depicting a specific pool size (e.g. the interval $(1,2]$ depicts a pool size with one applicant and its probability is $\int_1^2 F(\theta)d\theta$). As job opportunities (and with them pools of applicants θ,) follow one on the other, the prior second-order probabilities will shift as new data makes the one or the other hypothesis more likely. After observing the first actual pool size θ_1, an agent i updates his initial prior second-order probabilities $s_{0,i}$. These updated values serve as new prior second-order probabilities for the expected pool size θ_2 at the next job opportunity.

Updating Beliefs about Competitors' Application Behaviour

Some basics about Bayesian learning have been introduced in Section 3.2. In the present context, agents learn by observing the other agents' application behaviour. In the beginning, when no previous data sets are avail-

[6]This is of how Mathematica handles software packages.

able on which agents could base their predictions, the agents assign initial prior second-order probabilities, $s_{j,i}$, to each of the hypotheses H_j regarding the probability density functions of pool sizes they possess. As introduced above, initial prior second-order probabilities are assigned subjectively on the basis of beliefs that agents have assembled through different channels. Thus, initial prior second-order probabilities can differ across the population of agents. Each of the sets of initial prior second-order probabilities is a measure of subjective degree of belief in the different hypotheses, and these beliefs are modified on the basis of Bayesian methods acquiring new information. The conditional probability that pool size θ comes about at hypothesis H_j, $P(\theta|H_j)$, is called first-order probability of θ given hypothesis H_j. The following figure depicts hypotheses, a limited sample of events, prior second-order probabilities and posterior second-order probabilities.

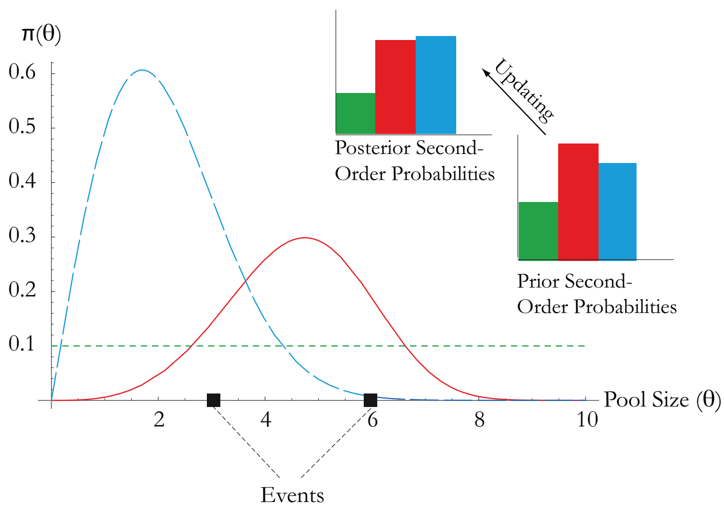

Figure 5.17: Bayesian Updating of Prior Second-Order Probabilities

Each agent updates his prior second-order probabilities of the hypotheses $\mathbf{H} = \{H_1, ..., H_m\}$ when new data θ becomes available. For a sequence of job opportunities he observes the pool size θ_t for each job opportunity. The posterior prior second-order probability, for hypothesis H_j, given the new data θ_t is

$$s_{j,i,t+1} = \frac{P(\theta_t|H_j)s_{j,i,t}}{\sum_{H_l \in \mathbf{H}} P(\theta_t|H_l)s_{l,i,t}} \tag{5.14}$$

with:

$s_{j,i,t}$	prior second-order probability of hypothesis H_j at job opportunity t assigned by of agent i,
$P(\theta_t\|H_j)$	first-order probability of pool size θ given hypothesis H_j.

Note that the first-order probability of a pool size does not depend on time, as an invariant set of static hypotheses is assumed. Each agent i repeats the updating procedure during the sequence of job opportunities that he experiences. For the next updating step $s_{j,i,t+2}$ of a hypothesis H_j at the next job opportunity $t+2$, its previous posterior second-order probability $s_{j,i,t+1}$ becomes its new prior second-order probability $s_{j,i}$. As job opportunities go by, hypotheses that perform well will be weighted more, while hypotheses that perform poorly will be weighted less. The agents constantly build and rebuild representations of the labour market by continuously shifting the weights in the light of new information.

For such a Bayesian learning process to be well-defined, no observed data shall produce a probability of zero given any hypothesis, so it shall hold that:

$$P(\theta|H_j) > 0, \forall H_j \in \mathbf{H}, \forall \theta \in \mathbf{\Theta}.$$

The hypotheses that have been chosen for the subsequent model all fulfill these requirements.

5.3.3 The Complete Model

A number of n potential applicants faces a sequence of job opportunities, where each job opportunity is associated with a number of vacancies v[7]. Each potential applicant needs to decide at each job opportunity whether

[7] Here, each job opportunity is associated with the same number of vacancies, but it could also be a varying number of vacancies.

he wants to apply. It does not matter whether he is currently employed or not, as employment happens on a job-to-job basis. Furthermore each agent holds three probabilistic hypotheses, H_1, H_2, H_3, about his competitors' application behaviour, each expressed as a probability density function regarding pool sizes. To each hypothesis H_j an agent i initially assigns a prior second-order probability, $s_{j,i,t=0}$. The prior second-order probabilities of the three hypotheses add up to one. The application decision of an agent i depends on the individual chances of success he perceives ppe_i , and only if ppe_i exceeds the critical probability[8], cp, will he apply. The individual chance of success depends on several factors: the productivity of the agent a_i, the weights $s_{1,i,t=0}, s_{2,i,t=0}, s_{3,i,t=0}$, the number of vacancies v, the productivity distribution of the population $F(a)$, and the rationing method R, according to which vacancies are allocated to the applicants. At each job opportunity, each agent decides whether to apply, the workforce is determined via the rationing method, each agent observes how many other agents have applied. With the observed information, each agent revises his prior second-order probabilities and uses them as information for determining ppe_i at the next job opportunity. This process will be expressed as a dynamical system in the following.

Define $\mathbf{A} = \{a_1, ..., a_n\}$ to be the set of productivities of the potential applicants. The productivities are assumed to be drawn from a Weibull density function. It is assumed that no two agents can exhibit exactly the same productivity. There are n such potential applicants. Each agent i has a critical probability, $cp \in I := [0,1]$, which must be exceeded before he decides to apply for a position. The set of productivities of the agents who decide to apply is defined as $\mathbf{AP} = \{a_1, ..., a_m\} \subseteq \mathbf{A}$. Furthermore, each agent holds three probabilistic hypotheses, H_1, H_2, H_3, regarding the competitors application behaviour. These are expressed as probability density functions regarding the the number of applicants for a given job opportunity $\theta := |\mathbf{AP}|$[9]. Initially, each agent i assigns an initial second-order probability $s_{j,i,t=0}$ to each hypothesis H_j that depicts the probability the agent assigns

[8]At the moment, each agent has the same critical probability cp, and the index i will be omitted below.

[9]$|SET|$ denotes the cardinality of SET.

to that hypothesis. Job opportunities are assumed to occur consecutively. For each time step t there are a number of vacancies $v_t \in \mathbb{N}$ available to be allocated among applicants who are characterised by their productivities **AP**. A rationing method R generates from the set of applicants' productivities **AP** a set of workers' productivities $\mathbf{W} \subseteq \mathbf{AP}$, thus the rationing method can be expressed as a function $R : \mathscr{P}(\mathbf{A}) \times \Omega \to \mathscr{P}(\mathbf{A})$, where Ω is a sample space if R has a stochastic component. If it does not contain a stochastic component $\Omega = \emptyset$. For a job opportunity with v vacancies each agent i determines his probability to make a successful application, ppe_i. When each agent i has decided whether to apply, so that **AP** is determined, and the workforce has been chosen, the agents observe the actual pool size, $\theta := |\mathbf{AP}|$. This information is used to revise the beliefs. The state space can now be described as:

$$\mathbf{X} : I^{3n} \times I^n \times \mathscr{P}(\mathbf{A}) \times \mathscr{P}(\mathbf{A}) \times \mathbb{N}. \tag{5.15}$$

It consists of the three second-order probabilities for each agent $\{s_{1,i}, ..., s_{l,n}\} \in I^{3n}$, the individual chances of success for each agent $\{ppe_1, ..., ppe_n\} \in I^n$, the set of applicants' productivities $\mathbf{AP} \in \mathscr{P}(\mathbf{A})$, the set of workers' productivities $\mathbf{W} \in \mathscr{P}(\mathbf{A})$, and the pool of applicants $\theta \in \mathbb{N}$. The function of the dynamics $\phi : \mathbf{X} \to \mathbf{X}, x_t \mapsto x_{t+1} = \phi(x_t)$ is defined by:

$$\begin{aligned}
&\forall i = 1, ..., n : ppe_{i,t+1} = ppe(s_{1,i,t}, s_{2,i,t}, s_{3,i,t}; a_i), \\
&\mathbf{AP}_{t+1} = \{a_i \in \mathbf{A} \,|\, ppe_{i,t+1} \geq cp\}, \\
&\mathbf{W}_{t+1} = R(\mathbf{AP}_{t+1}, \omega_t), \\
&\theta_{t+1} = |\mathbf{AP}_{t+1}|, \\
&\forall i, j : s_{j,i,t+1} = \frac{P(\theta_{t+1}|H_j)s_{j,i,t}}{\sum_{H_l \in \mathbf{H}} P(\theta_t|H_l)s_{l,i,t}},
\end{aligned} \tag{5.16}$$

where $\omega_t \in \Omega$ is a random event.

5.3.4 Configuration & Simulation

> Although a simulation does not constitute a mathematical proof, it does provide the tool to study mass agent behaviour where every agent is somewhat different from all others.
>
> Shubik (1999, p. 188)

In the following the configuration of the model for the subsequent simulation will be introduced. 10 agents update their beliefs regarding the three hypotheses on pool sizes as illustrated in Figure 5.18 over a sequence of 30 job opportunities. The hypotheses regarding pool sizes depicted by this figure are the PDF of the uniform distribution

$$\tilde{U} = \begin{cases} 0 & \text{for } x < c \\ \frac{1}{d-c} & \text{for } c \leq x \leq d \\ 0 & \text{for } x > d \end{cases}$$

with $c = 0.0$ and $d = 10.0$, the PDF of the Weibull distribution

$$\tilde{W} = ab^{-a}e^{-\left(\frac{x}{b}\right)^a}x^{a-1}$$

with $a = 4.0$ and $b = 5.1$, and the PDF of the Rayleigh distribution

$$\tilde{R} = \frac{\frac{xe^{-x^2}}{2\beta^2}}{\beta^2}$$

with $\beta = 1.7$.

Table 5.1 summarises the complete initial calibration of the model.

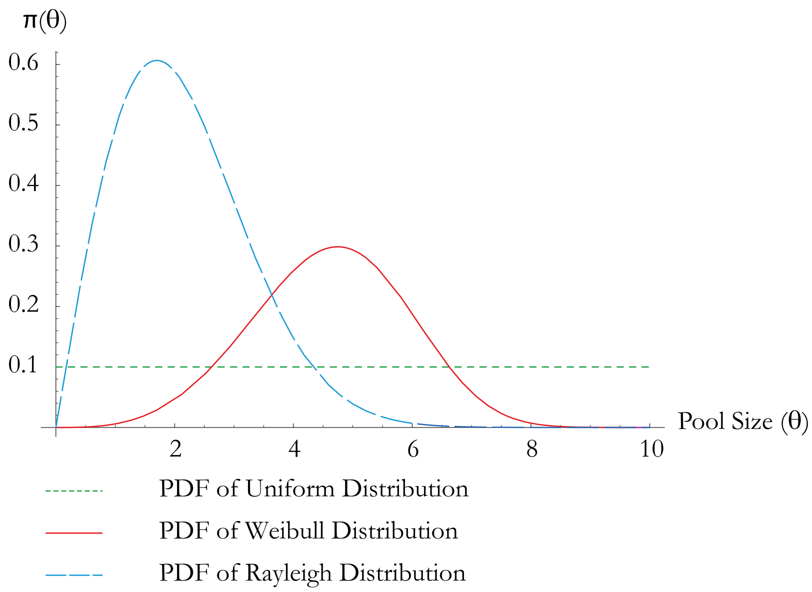

Figure 5.18: Hypotheses Regarding Pool Size Probabilities in the Current Model

Number of potential applicants n	10
Productivity Distribution $F(a)$	Weibull Distribution a=2.0,b=12.0
Critical Probability cp	0.35
Number of Job Offers y	30
Vacancies per Job Offer v	1
Hypotheses on Pool Distributions	H_1: $\tilde{U}$, $c = 0.0$, $d = 10.0$
	H_2: $\tilde{W}$, $a = 4.0$, $b = 5.1$
	H_3: $\tilde{R}$, $\beta = 1.7$
Pigeonhole Size δ	1.0
Productivities of Population A	{6.86,9.95,19.98,5.72,16.48,0.82,3.53,4.09,11.58,5.51}

Table 5.1: Initial Model Calibration with 10 Agents and 30 Job Opportunities

For determining ppe_i, each agent needs to know the upper and lower productivity limits of the pigeonhole that he / his productivity belongs to. For the current scenario, the lower limit $\underline{a_i}$ is the greatest integer less or equal to the agents productivity a_i, and the upper limit $\bar{a}_i$ is the smallest integer greater or equal to a_i. By modifying the upper and lower productivity limits, pigeonholes of different sizes can be created and different levels of precision in the perception can be modelled.

5.3.5 Results

Table 5.2 illustrates, for all 10 agents, the productivities, initial prior second-order probabilities (initial SOP), and final posterior second-order probabilities (final SOP).

Productivity	**Initial SOP (H_1,H_2,H_3)**	**Final SOP (H_1,H_2,H_3)**
6.68	(0.9,0.05,0.05)	($4.708096 \cdot ^{-7}$, 0.003434, 0.996565)
9.95	(0.05,0.9,0.05)	($2.471322 \cdot ^{-8}$, 0.058409, 0.941591)
10.98	(0.7,0.1,0.2)	($9.170382 \cdot ^{-8}$, 0.001720, 0.998280)
5.72	(0.1,0.1,0.8)	($3.279367 \cdot ^{-9}$, 0.000431, 0.999569)
16.48	(0.1,0.5,0.4)	($6.533416 \cdot ^{-9}$, 0.004289, 0.995711)
0.82	(0.4,0.4,0.2)	($5.213315 \cdot ^{-8}$, 0.006845, 0.993155)
3.53	(0.1,0.1,0.8)	($3.279367 \cdot ^{-9}$, 0.000431, 0.999569)
4.09	(0.33,0.33,0.33)	($2.615610 \cdot ^{-8}$, 0.003434, 0.996566)
11.58	(0.25,0.25,0.5)	($1.310055 \cdot ^{-8}$, 0.001720, 0.998280)
5.51	(0.4,0.2,0.4)	($2.620109 \cdot ^{-8}$,0.001720, 0.998280)

Table 5.2: Productivities, Initial Prior Second-Order Probabilities and Final Posterior Second-Order Probabilities with Initial Model Calibration

All agents (except agents 4 and 7) started with different beliefs into the hypotheses. At the end of the sequence, all the agents have shifted their second-order probabilities towards a high belief in the Rayleigh PDF (H_3), little belief in the Weibull PDF (H_2) and nearly no belief in the uniform PDF (H_1). The ensemble of hypotheses and weights assigned to them is an approximation of the true density of pool sizes that emerges endogenously.[10] The beliefs of agent 4 and agent 7 take identical values throughout the learning process, as they started with identical initial prior second-order-probabilities. While this table reveals that the agents' beliefs develop towards the same combination of beliefs, it does not reveal how this development process evolves over the sequence of job opportunities. Figures 5.19-5.21 depict the development of the individual second-order probabilities regarding the three hypotheses. Each figure illustrates how the weight each agent assigns to the depicted hypothesis develops over the sequence of 30 job opportunities. The weight assigned to the hypothesis is indicated by the colour.

[10]This true distribution may be a point distribution.

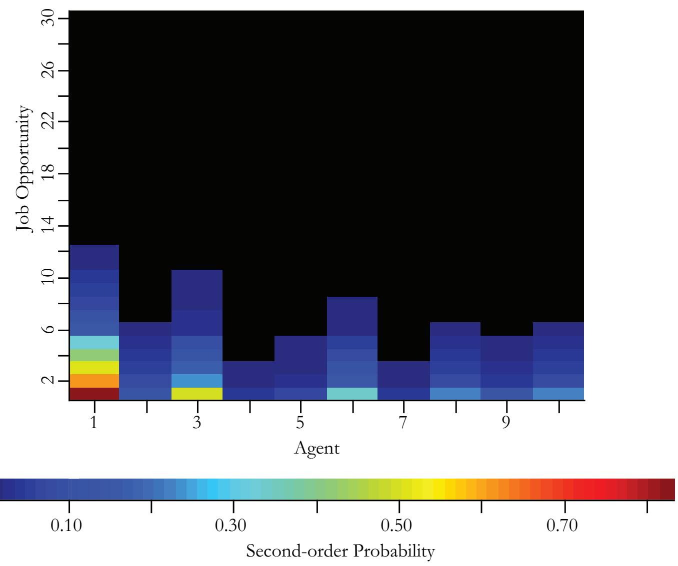

Figure 5.19: Development of Individual Prior Second-Order Probabilities Regarding H_1 with Initial Model Calibration

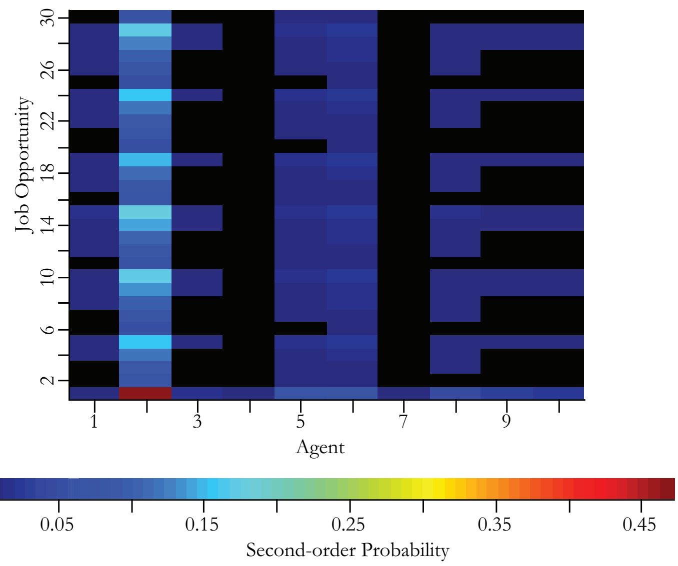

Figure 5.20: Development of Individual Prior Second-Order Probabilities Regarding H_2 with Initial Model Calibration

After having experienced some job opportunities, all agents assign most weight to the Rayleigh PDF and the second-order probabilities fluctuate slightly around it, while the beliefs of the second agent fluctuate more, as can be seen in Figure 5.21. It can also been seen that the development of the prior second-order probabilities of agent 4 and agent 7 follow the same path due to the same initial prior second-order probabilities of these agents.

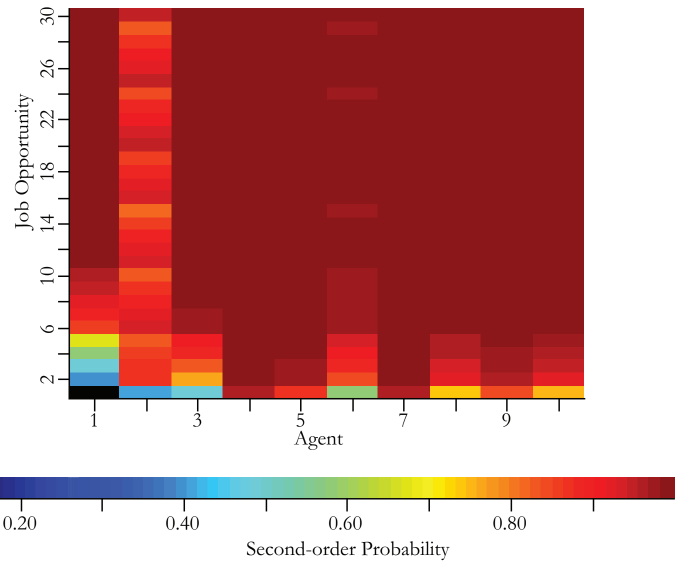

Figure 5.21: Development of Individual Prior Second-Order Probabilities Regarding H_3 with Initial Model Calibration

The development of the second-order probabilities depends crucially on the configuration of the model. Different parameter values for critical probability cp and the number of vacancies v may result in different processes. To gain insights into the model's outcome for different parameter combinations, a sensitivity analysis was conducted with help of the SimEnv software which has been introduced in Section 3.3. The software systematically changes parameter values and runs the model with each setting, saving the output for post-processing. In the present case, the parameters for critical probability cp and number of vacancies v have been modified. Table 5.3 summarises the ranges of values that have been set for each parameter.

Parameter	**Values**
Vacancies per Job Opportunity v	1-10 $\in \mathbb{N}$
Critcial Probability cp	0.1-0.9 in steps of 0.1

Table 5.3: Value Ranges of Parameters cp and v for Behavioural Analysis of the Model with SimEnv.

A total number of 90 model configurations are processed. For each agent, and each model configuration, the sequence of individual prior second-order probabilities regarding each hypothesis is included in the output. This enables the investigation, for each parameter combination, of the development of the individual prior second-order probabilities. Figures 5.22-5.24 illustrate that there must not necessarily be such a clear process as shown by the Figures 5.19- 5.21. Figures 5.22-5.24 depict the convergence process at cp=0.6 and v=2.

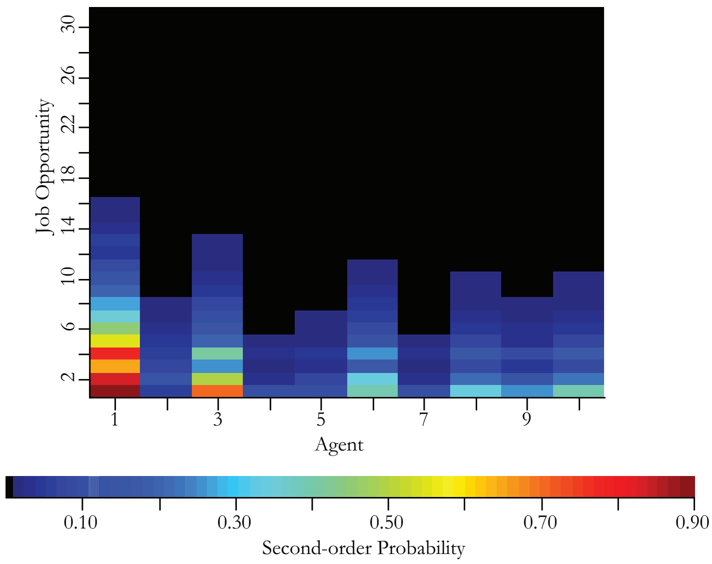

Figure 5.22: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 1 (Uniform PDF) with Calibration: 2 Vacancies, Critical Probability 0.6

After 16 job opportunities, all agents have found that the uniform PDF is not depicting their competitors' behaviour well. Rather they find a combination of H_2 and H_3 to be likely, but the weights of H_2 and H_3 fluctuate: When H_2 is relative likely, H_3 is not and vice versa. Apparently, the application behaviour of their competitors leads the agents to revise the second-order probabilities of these two hypotheses constantly. The true density function regarding that pool sizes are distributed, if there is one, is approximated by the combination of the weights assigned to the three hypotheses.

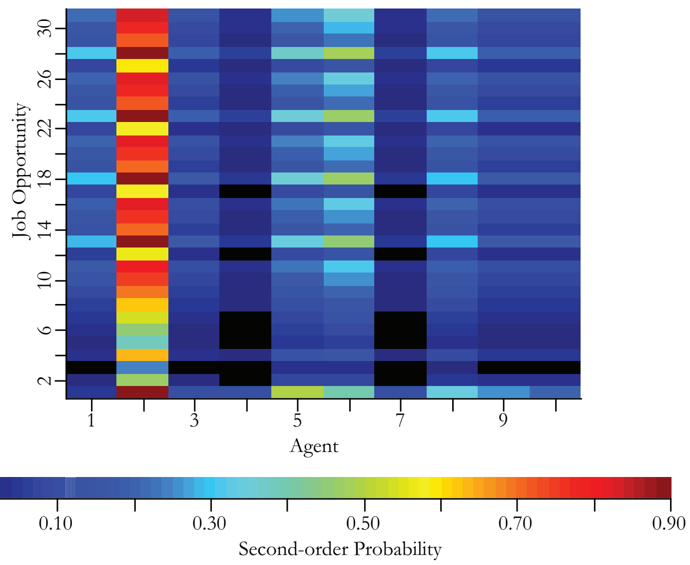

Figure 5.23: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 2 (Weibull PDF) with Calibration: 2 Vacancies, Critical Probability 0.6

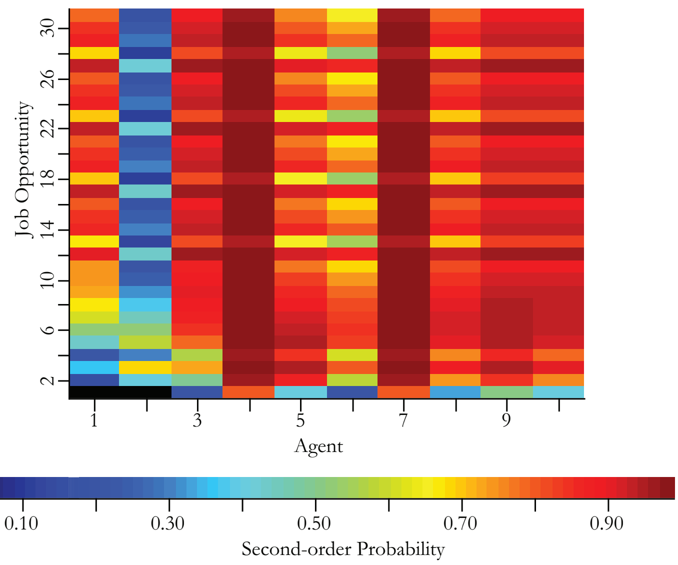

Figure 5.24: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 3 (Rayleigh PDF) with Calibration: 2 Vacancies, Critical Probability 0.6

Below, the number of applicants per job opportunity is depicted.

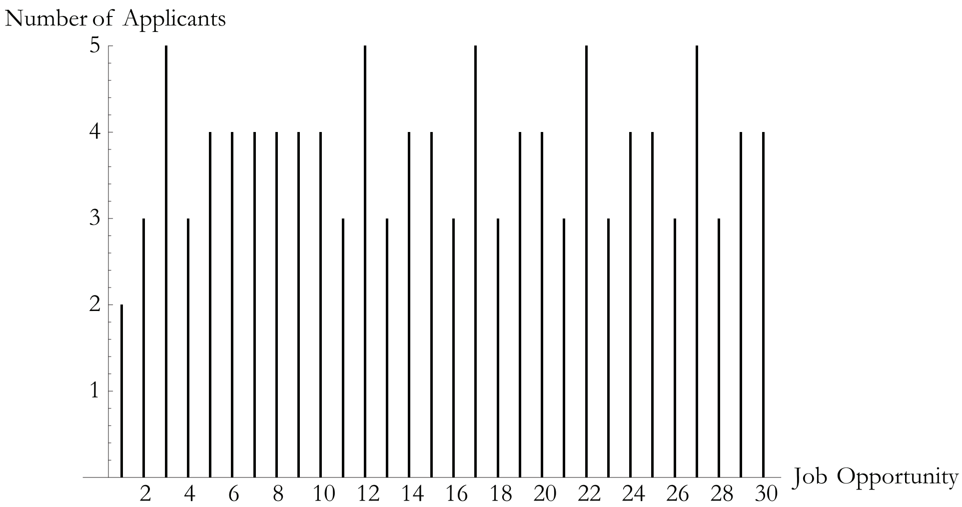

Figure 5.25: Number of Applicants with Calibration: 2 Vacancies, Critical Probability 0.6

Changes in the number of applicants lead the agents to shift the weights $s_{j,i,t}$ of their hypotheses, which results in higher or lower individual PPE. Some agents might experience transitions from a PPE below their critical probability cp to values above it. If this is the case, their application decision changes. Figure 5.25 shows that this is the case for some agents. The number of applicants is not distributed in accordance with a pure hypothesis. Rather, the true distribution of pool sizes emerges from the scenario itself, and is an ensemble of the three weighed hypotheses that each agent possesses.

5.4 Agents, Bayes, and the Labour Market II: Average Productivities of Workforces Recruited Under Different Rationing Methods

5.4.1 Outline

In the previous section, Bayesian reasoning has been used to model heterogenous and learning agents on a labour market. In order to refine the perception of their chances to obtain one job, the agents were equipped with different hypotheses regarding their competitors application behaviour. Depending on their previous experience, they assigned individual initial prior second-order probabilities to each of their hypotheses, observed their competitors' behaviour over a sequence of job opportunities, and updated their prior second-order probabilities accordingly. The question of the productivity of a workforce recruited according to different levels of inaccuracy of perception on the employer's side is investigated in the remainder of this section.

The settings for the present investigation are similar to the ones from the previous section: In the basic scenario, 10 agents act on the labour market, seek a job and experience 30 job offers. They base their application decision on whether their ppe_i exceeds their individual (but at the moment identical for all agents) critical probability cp; only then do they apply. They observe each others' application behaviour over the sequence of job opportunities and a revision of the prior second-order probability lets them not only learn about their competitors' application behaviour but also refine the approximation of their ppe_i.

Additionally, the recruitment takes place and employees are being determined from the pool of applicants θ. In the current model, recruitment always takes place over a whole sequence of job opportunities. Adopting different rationing methods for these sequences enables the observation of how the average productivity q of workforces depend on different accuracies of perception regarding the applicants' productivities.

5.4.2 Recruitment Methods

Methods with which an agent i can determine his ppe_i at different recruitment methods R (ranking, pigeonhole and random), were developed in Section 5.2 and applied in Section 5.3. It has also been noted that pigeonhole rationing can account for both ranking and random rationing by choosing pigeonholes with appropriate sizes. An applicant's actual productivity a_i is not revealed to the employer, but the employer perceives him as belonging to a pigeonhole, so that if a pigeonhole spans from productivity 1.0 to productivity 2.0 (including 1.0, excluding 2.0), productivities 1.2 and 1.99 both *belong* in this pigeonhole.

In the present section, the sizes of the pigeonholes δ are modified systematically to obtain information about how average productivities q of workforces $\mathbf{W}$ depend on the size of the pigeonholes, i.e. the accuracy of the employer's perception of the applicants' qualifications. Ranking rationing is the most sophisticated method to recruit employees, as each of them is inferred correctly and, therefore, the most productive workforce will be obtained with this method. It seems plausible that with decreasing accuracy of perception of the applicants' qualifications, the average productivity of the assembled workforce will decrease monotonously.

5.4.3 The Complete Model

The model remains the same as introduced in 5.3.3, but additionally, at each job opportunity the average productivity $q \in \mathbb{R}_+$ of the workforce $\mathbf{W}$ will be calculated. $I := [0,1]$ shall denote the interval of values which probabilities can take. The state space differs only slightly from the previous one and is as follows:

$$X : I^{3n} \times I^n \times \mathscr{P}(\mathbf{A}) \times \mathscr{P}(\mathbf{A}) \times \mathbb{N} \times \mathbb{R}_+. \tag{5.17}$$

It consists of the three second-order probabilities for each agent $\{s_{1i}, ..., s_{ln}\} \in I^{3n}$, the individual chances of success for each agent $\{ppe_1, ..., ppe_n\} \in I^n$, the set of applicants $\mathbf{AP} \in \mathscr{P}(\mathbf{A})$, the set of workers $\mathbf{W} \in \mathscr{P}(\mathbf{A})$, the pool of applicants $\theta \in \mathbb{N}$, and the average productivity of the workers $q \in \mathbb{R}_+$. The function of the dynamics $\phi : \mathbf{X} \rightarrow \mathbf{X}, x_t \mapsto x_{t+1} = \phi(x_t)$ is defined by:

$$
\begin{aligned}
&\forall i = 1,...,n : ppe_{i,t+1} = ppe(s_{1,i,t}, s_{2,i,t}, s_{3,i,t}; a_i),\\
&\mathbf{AP}_{t+1} = \{a_i \in \mathbf{A} \,|\, ppe_{i,t+1} \geq cp\},\\
&\mathbf{W}_{t+1} = R(\mathbf{AP}_{t+1}, \omega_t),\\
&\theta_{t+1} = |AP_{t+1}|,\\
&\forall i,j : s_{j,i,t+1} = \frac{P(\theta_{t+1}|H_j)s_{j,i,t}}{\sum_{H_l \in \mathbf{H}} P(\theta_t|H_l)s_{l,i,t}},\\
&q_{t+1} = \frac{\sum_{a_i \in \mathbf{W}_{t+1}} a_i}{|\mathbf{W}_{t+1}|},
\end{aligned}
\tag{5.18}
$$

where $\omega_t \in \Omega$ is a random event., and $|...|$ denotes the cardinality of a set.

5.4.4 Experimental Settings

The initial model calibration that has been introduced in Table 5.1 is used here, too. It has been extended so as to include the actual recruitment procedure and the determination of the average productivity of the resulting workforce. For that purpose, an additional parameter has been introduced to the program: δ represents the size of the pigeonholes. The larger δ, the larger are the pigeonholes and the more inaccurate is the employer's perception of applicants' qualifications. A sensitivity analysis of the model is conducted using a SimEnv experiment. For several combinations of critical probability cp and number of vacancies v, the parameter δ is increased systematically, simulating different levels of inaccuracy including both ranking rationing and random rationing as the extreme cases. For each value of δ, the model simulates, for a given combination of cp and v, the application process, the learning process and the actual recruitment over the sequence of 30 job opportunities. At each job opportunity, the average productivity of the workforce is determined, and at the end of the sequence the overall average productivity is calculated. Consequentially, it is possible to determine how the average productivity of a workforce at a specific calibration of cp and v depends on the inaccurate perception δ of the employer regarding the applicants' productivities. The following table summarises the experiment settings for the basic scenario.

Parameter	**Values**
Number of potential applicants n	10
Job Offers	30
Productivity Distribution $F(a)$	Weibull Distribution a=2.0,b=12.0
Hypotheses on Pool Distributions[11]	H_1: $\tilde{U}$, $c = 0.0$, $d = 10.0$
	H_2: $\tilde{W}$, $a = 4.0$, $b = 5.1$
	H_3: $\tilde{R}$, $\beta = 1.7$
Critical Probability cp	0.1,0.2,0.3,0.35,0.5,0.6,...,0.9
Vacancies v	1-10
Pigeonhole sizes δ	0,0.5,1.0,1.5,2.0,...,17.5,1000
Productivities **A**	{6.68173, 9.95951, 10.9797, 5.72044,
	16.4826, 0.82039, 3.53244, 4.09626,
	11.5802, 5.51566}

Table 5.4: Model Calibrations for a Simulation of Average Productivities of Workforces in Dependence on the Level of Perception Error Regarding Applicants' Productivities.

This experiment comprises a total of 3700 single runs of the underlying model (executed for each combination of cp, v and δ) [12].

5.4.5 Results

One would expect that an increasing inaccuracy on the employers side leads to a monotonic decrease of the average productivity of the workforce. The results are surprising and do not confirm these expectations completely. Figures 5.26 and 5.27[13] illustrate the average productivities of workforces over the series of pigeonhole sizes given by Table 5.4. On the x-axis the inaccuracy of the employer's perception represented by the

[11]The functional forms of the hypotheses were introduced in Section 5.3.

[12]In this model calibration, pigeonhole Sizes 0 and 1000 in Table 5.4 are keys for indicating either ranking rationing or random rationing. When one of these keys is set, the program executes the corresponding parts of the code, which differ from one another. Note that when pigeonholes exhibit size 17 or 17.5, pigeonhole rationing corresponds to random rationing but pigeonhole rationing is executed.

[13]These two calibrations have been visualised regarding the development of prior second-order probabilities in the previous section,too.

size of the pigeonhole is depicted. On the y-axis the average productivity q of the workforce recruited at the corresponding pigeonhole size δ is shown. Each bar illustrates the average productivity of the workforce taking into account all thirty job opportunities during which agents determine whether to apply or not, are recruited, and update their hypotheses regarding their competitors' behaviour.

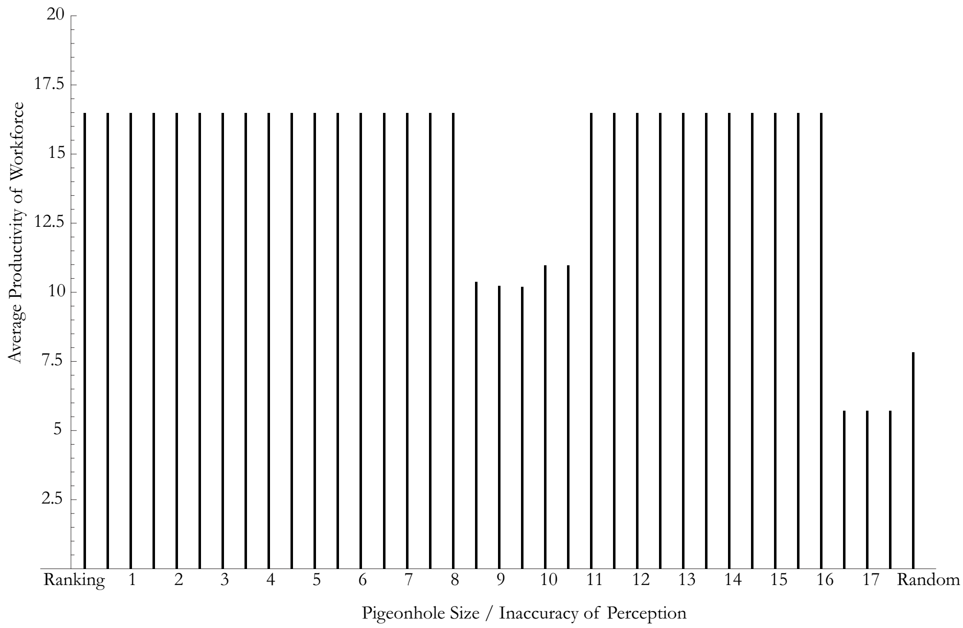

Figure 5.26: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy with Initial Model Calibration (Agents: 10, Vacancies: 1)

Under the initial model calibration with ten agents and thirty job opportunities with one vacancy, the average productivity of the workforce does not decrease monotonically with increasing inaccuracy of inference. Rather, average productivity remains constant until it drops at a certain level of inaccuracy, and then recovers. Only at very large inaccuracies does the average productivity differ much from workforces recruited by ranking rationing. At random rationing the bar displays a larger average productivity than at the two inaccuracies before. This is because at random rationing there is the chance of obtaining productive employees, too. Repeating the whole experiment many times should produce an average productivity at random rationing that corresponds to the average productivity of the population of potential applicants. The explanation for the drop and recovery of average productivity at some pigeonhole sizes follows from the setting of the scenario: At pigeonhole sizes between 8.5 and 10.5, more than one agent perceive sufficient probabilities to apply for the position, but all these agents' productivities (11.5802,10.9797,16.4826) are perceived to belong to the same pigeonhole. At pigeonhole sizes smaller than 8.5 and larger than 10.5 they do not. When they belong to the same pigeonhole, the employer cannot distinguish between them and randomly chooses one of the applicants; the actual productivity a_i being revealed only after the choice has been made. At pigeonholes between 11.0 and 16.0, the employer is able to distinguish the applicants with productivity 16.4826 and 11.5802, and, hence, the employer is able to choose the more productive applicant, even if his overall perception is relatively inaccurate. This seems to imply the following: There is no perfect choice of pigeonhole sizes that will always produce the best workforce. The average productivity of the workforce will depend on the relation between the productivities of the applicants and the pigeonhole sizes. Since each applicants' productivity is being categorised into the next smaller pigeonhole, even at very large pigeonholes, presumed the productivities lie accordingly, applicants may be perceived correctly. Figure 5.27 illustrates a similar pattern for a model calibration where two vacancies are available and the critical probability is set to 0.6.

The sensitivity analysis regarding the basic scenario revealed that the results illustrated by Figure 5.26 are stable, even at diverse combinations of v and cp. Several other figures can be found in Appendix B.2.

The average productivity of a workforce seems to depend on the relation of productivities and inaccuracies and in certain situations an employer may not need to make the effort of categorising his applicants precisely.[14] Since such a categorisation may require additional time and money – a cost factor which has not been modelled at the current stage – it may not be worth investing in. Appendix B shows other figures regarding the average productivity at a different scenario (twenty agents, twenty job op-

[14]How an employer anticipates when such an situation is given is not part of this work.

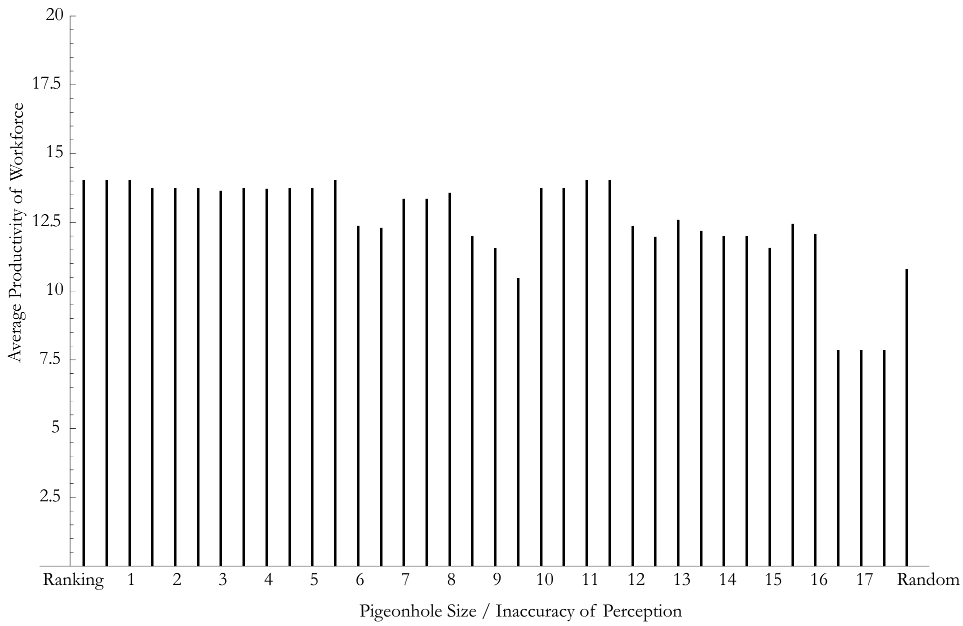

Figure 5.27: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy. Calibration: 2 Vacancies, Critical Probability 0.6

portunities) and at different calibrations of this scenario. Summarising the results:

⋆ Average productivity does not necessarily decrease monotonically with increasing perception error.

⋆ At relatively large degree of randomness in the recruiting process, average productivity is significantly less than at precise perception.

⋆ The average productivity of the workforce seems to depend on the relation between productivities of applicants and the choice of the pigeonhole sizes. Is the relation appropriate, even at larger pigeonholes applicants can be perceived precisely.

Chapter 6

Summary and Conclusion

6.1 Summary & Conclusion

The aim of the present work has been twofold. First to provide a rationing toolbox which contains various rationing methods, and second to apply rationing methods from the toolbox to investigate a specific micro-economic topic in detail. A fictitious labour market was modelled in order to answer several questions.

The *lagom* model family is constructed by the working group *Global Change and Financial Transition* (GFT) at the Potsdam Institute for Climate Impact Research (PIK), and conducts micro- and macro-economic modelling. Some of the micro-economic branches of *lagom* are disequilibrium models. Trading at disequilibrium prices is allowed and may lead to mismatched demands and supplies. A market mechanism is needed to generate transactions from such mismatched demands and supplies. Such a mechanism is a rationing method. Since inconsistent demands and supplies may occur on markets that differ greatly from one another, providing a single rationing methods is not sufficient.

The first goal arose from this need for different rationing methods. A rationing toolbox was developed specifically for use within *lagom*. The toolbox contains several well known rationing methods and some methods that were specifically programmed for the *lagom* context. Each of the rationing methods was documented economically and mathematically. The code containing computational documentation, was provided in Appendix A. The applicability of the methods contained in the toolbox was tested whilst working on the second aim of the present work.

The second goal was to apply rationing methods to model a micro-economic labour market. The literature dealing with micro-economic labour markets usually assumes that (a) a labour supplier who seeks employment applies for a job if the offered wage exceeds his reference wage, and that (b) an employer who faces a pool of applicants is not able to distinguish them according to their qualifications. Such an employer, thus, chooses his workforce blindly from the pool of applicants. By increasing the wage, he can increase the number of qualified applicants, but he still faces an anonymous pool of applicants.

The present work has provided a modified framework to bring together ways of dealing with rationing methods with approaches dealing with micro-economic labour markets.

On many labour markets, wages are set through negotiations between trade unions and federations of employers, and are thus not negotiable between employers and applicants. Many employers screen their applicants regarding their qualifications and choose the workforce accordingly. The framework for investigating a micro-economic labour market was, thus, set to depict this fact.

On a labour market on which wages are non-negotiable, a heterogenous population of potential applicants faces consecutively occurring job opportunities, each exhibiting a certain number of vacancies. Each agent needs to decide whether to apply in each case. Since the wage is non-negotiable it is assumed that the incentive to apply is an agent's perceived probability of obtaining employment conditional on application (PPE). Only if an agent's PPE exceeds some critical probability will he apply. To determine his PPE, an agent needs to anticipate his competitors' application behaviour. The assumptions he makes about this are expressed as an invariant set of static probabilistic hypotheses. Each agent possesses the same set of hypotheses but initially assigns subjective degrees of beliefs (initial prior second-order probabilities) to each of the hypotheses. The agents are heterogenous with respect to their productivities and their initial prior second-order probabilities. During a cycle of job opportunities the agents observe each other and learn from the information.

The employer is able to perceive the qualifications of his applicants more or less accurately. He conducts recruitment with respect to one of three rationing methods, each of them exhibiting a different level of accuracy in the perception of applicants' qualifications. The rationing method which the employer uses is common knowledge.

The questions that were tackled within this framework are presented below, followed by the results.

1. *How can agents determine their individual PPE under different rationing methods?* To answer this question, the kind of information an agent needs to make statements about the probability of a successful application was determined. This information is: the productivity distribution of the population of potential applicants, the agent's own productivity, the probability of each possible pool size of applicants (i.e their competitors' application behaviour), the rationing method used by an employer, and the number of vacancies. With this information, an agent is able to determine his individual PPE. For ranking rationing, pigeonhole rationing, and random rationing the analytical determination of the PPE was presented. Axioms have been postulated for each method and the results gained are consistent with the axioms. This has been illustrated for each method by providing one figure for each axiom.

2. *How can one model heterogenous labour suppliers who must make application decisions and simultaneously learn about their competitors application behaviour by observing them?* By answering question one it was found that in order to determine their individual PPE, agents need to have information about their competitors' application behaviour. It is, however, likely that they do not possess this information. In this case what they can reasonably do is make hypotheses about their competitors' application behaviour, and, before any experience has been gained, furnish each hypothesis with a initial degree of belief (i.e. prior second-order probability). Such initial prior second-order probabilities are based on subjective reasoning and differ across the population. As the agents experience consecutive job opportunities, they observe the actual application behaviour of their competitors. They update their second-order probabilities so as to account for this newly gained information. This kind of learning process was implemented via Bayesian reasoning.

The distinctive feature of this present implementation is the following: Many heterogenous agents observe each other and learn simultaneously about each other's application behaviour which emerges endogenously by their own actions.

This differs from contributions such as Feltovich and Papageorgiou (2004), Lewis and Terrell (2001), Farmer and Terrell (1996), and others, where the learning process of only a single agent is modelled.

Under several model calibrations, the second-order probabilities of all agents develop towards representing the same beliefs, while in other calibrations they do not develop in such a clear way. The application behaviour emerges endogenously and, thus, the agents' observations may lead to diverse updated second-order probabilities.

3. *In the given framework, can an employer, though inaccurate in the perception of applicants' qualifications, obtain an equally productive workforce as if he were accurate?* This third question was tackled by combining methods developed for answering question one and question two. A model was built simulating a whole sequence of consecutive job opportunities. At each time step it included: the determination of each agent's PPE, each agent's decision whether to apply, the observation of how many agents applied, the updating of each agent's degrees of belief (prior second-order probabilities) regarding his competitors' application behaviour, the actual recruitment procedure, and the determination of the average productivity of the workforce. The actual recruitment procedure was implemented by applying the corresponding rationing method from the rationing toolbox.

   In order to find out whether an employer can, by a specific choice of inaccurate perception of applicants qualifications, obtain the most productive workforce, the sequence of job opportunities was executed repeatedly and the accuracy of the employers perception of applicants' qualifications has been varied systematically. Many different levels of accuracy could thus be accounted for, with the extremes corresponding to either very precise or random allocation of jobs to applicants.

Only with very large perception inaccuracies regarding applicants' qualifications, i.e a great degree of randomness, did the average productivity of the workforce decrease significantly. With a limited sample of agents' productivities, at nearly all levels of inaccuracy the average productivity corresponded to the best that can be obtained. There were, however, some minor decreases at some levels of inaccuracy, but these were always followed by increases.

Looking at the model data, it becomes clear how such a pattern could emerge. The average productivity of the workforce depends on the relation between productivities of applicants and inaccuracy of perception. If a certain relation occurs, then even when being relatively inaccurate the employer perceives applicants' qualifications correctly.

The result is: If the productivities of applicants exhibit a suitable relation to an employer's inaccurate perception of these productivities, he can obtain the most qualified workforce even when he does not invest in a precise perception.

6.2 Concluding Remarks

Each agent in the model which has been presented in this work has in itself the computational machinery to figure out how to behave in particular situations (cf. Axtell (2006, p. 208)). However, the current model is still highly idealised and to develop the model further into the directions pointed out by Axtell (2006, pp.203) the following shortcomings should be overcome:

- ⋆ The agents are assumed to possess an invariant set of static hypotheses regarding their competitors' application behaviour. The true application behaviour may not be covered by any of these hypotheses.
- ⋆ The model was not tested to see how it reacts to:
  - ○ the productivity distribution of potential applicants, and

- ○ the sets of static probabilistic hypotheses.

⋆ The qualification of the agents does not change in the course of action.

At the present state of the model, more sensitivity analyses may yield further insights. Such analyses include the systematic variation of the productivity distribution of the population and the systematic variation of the set of hypotheses.

If agents possess an invariant and static set of hypotheses, the second-order probabilities will develop so as to represent the true application behaviour, which emerges endogenously, as closely as possible. If one were to equip each agent with a variant set of hypotheses, agents could be modelled such that they could add a more realistic hypothesis to their set once they anticipate that the true behaviour is not depicted by one of the hypotheses.

Furthermore, each agent may be modelled to possess one hypothesis for the probability of each possible pool size of applicants. If this were the case, there would be as many hypotheses as pool sizes (i.e potential applicants). During the learning process, the second-order probabilities would indicate which pool size seems most probable. However, conducting this approach for large populations of potential applicants would necessitate an increase in computational power and would require each agent to be very sophisticated in observing his environment. In that sense, the invariant set of static probabilistic hypotheses that had been criticised above may very well serve as a better approximation of reality than a more sophisticated approach.

The agents' productivities in this model are assumed to be fix. Persistent unemployment – because of decreasing qualification caused by long-term unemployment – and learning-by doing during employment, as suggested by Fuhrmann (1988, pp. 348), are not yet implemented. Thus, a reasonable extension towards a more realistic model includes the modeling of human capital accumulation through employment and the decrease of human capital through unemployment. The resulting model would then account for the fact that unemployment histories matters in the recruiting process.

Another fruitful extension may be to model the agents to be insecure about what kind of rationing methods the employer conducts. They could then be modelled to learn about the rationing methods in Bayesian fashion.

6.2.1 Further Research

The framework provided by Chapter 5 provides a fruitful basis for further research that combines two strands of literature. These strands are models that deal with application processes and incentives for application, such as Mookherjee (1987) and Weiss (1976, 1980), and literature that deals specifically with rationing. Literature that assumes wages to be the incentive for labour suppliers to apply for a job usually does not specify a rationing method that the employer conducts to recruit his workforce. Either, as in Mortensen and Pissarides (1994)[1] a matching functions brings together jobs and employees, or an employer picks randomly from a pool of applicants, or a revelation method is used (see Mookherjee (1987)). In the two latter models, an employer attracts more qualified agents if he were to set a larger wage. Then, more qualified agents find that their expected wage exceeds their reference wage (which is positively correlated with their qualifications) and apply for a job. An employer then picks randomly from them. He has increased his chance to obtain more qualified applicants but he cannot identify who they are.

Within the present framework it is possible to construct a model where employers set wages and conduct a specific rationing method, and labour suppliers decide whether to apply during a two-stage decision process. In such a model, a labour supplier would first determine whether the offered wage exceeded his reference wage, and if it did he would determine his individual PPE. Only if his PPE exceeded his critical probability would he apply. In such a framework an employer could, by increasing the wage

[1] Here the wage is not an incentive in the usual sense, since it is assumed that the highest wage is being paid and no agent will decline a job.

rate, attract more qualified agents, and, by conducting a certain rationing method he could also pick the best workforce from the applicants. The potential applicants, on the other hand, would only apply for jobs where the wage offered exceeded their reference wage, and where they had enough PPE to apply. The critical probability that needs to be exceeded by the PPE so an application will be made could also be modelled to be correlated with qualifications (e.g. more qualified agents need to perceive less chances of success to apply, since they know that they could relatively easily find another job). In general, such a model could be equipped with all features and extensions that have been mentioned in this present work. A further combination of the two strands of literature, the basics of which have been accomplished by the present work, could thus be achieved.

On a more general level there are two possible paths to pursue. These are not necessarily disjunct. One path to pursue is the analysis of the dynamics of the model by mathematical means. The other path is the extension of the model so that it can account for very large populations of heterogenous agents in ACE (Agent-based Computational Economics) style.

> Exploiting the growing capabilities of computers, ACE is the computational study of economic processes modeled as dynamic systems of interacting agents.
>
> Tesfatsion (2006, p. 177)

The agents in the current model interact, learn from past experiences, are reasonable (using simple decision rules) and act in a system with emergent properties; emergent in the sense that there are properties that arise from the interactions of the agents and that are not properties of the individual agents themselves (cf. Tesfatsion (2006, p. 178)).

6.3 Deutsche Zusammenfassung

In der vorliegende Arbeit habe ich mich mit zwei Dingen beschäftigt. Zum einen entwickelte ich eine Modellierungstoolbox, die verschiedene Rationierungsmethoden enthält. Diese Rationierungsmethoden waren entweder aus der Literatur bekannt, oder wurden speziell für die *lagom* Modellfamilie entwickelt.

Zum anderen habe ich gezeigt, wie man mit Hilfe einiger der Rationierungsmethoden aus der Modellierungstoolbox einen fiktiven Arbeitsmarkt modellieren kann, auf dem heterogene arbeitssuchende Agenten einander beobachten und daraus in Bayesianischer Weise über ihre Chancen einen Job zu bekommen lernen.

Die *lagom* Modellfamilie wird von der Arbeitsgruppe *Global Change and Financial Transition* (GFT) am Potsdam Institut für Klimafolgenforschung (PIK) erstellt. Diese Arbeitsgruppe modelliert sowohl auf der mikroökonomischen als auch auf der makroökonomischen Ebene. Einige der Modelle auf der mikroökonomischen Ebene sind Ungleichgewichtsmodelle, und Agenten können auch bei Nichtgleichgewichtspreisen handeln. Dies hat zur Folge, dass Nachfragen und Angebote sehr wahrscheinlich nicht übereinstimmen und es einen Mechanismus geben muss, der aus dieser Situation Transaktionen generiert. Solch ein Mechanismus ist eine Rationierungsmethode. Nicht übereinstimmende Nachfragen und Angebote können auf verschiedenen Märkten auftreten, ewelche sehr unterschiedlich charakterisiert sein können. Daher ist es nicht ausreichend, nur eine Rationierungsmethode für ein Modell, welches viele Märkte umfasst, zur Verfügung zu stellen.

Das erste Ziel der Arbeit hat sich genau auf diesen Sachverhalt bezogen. Ich entwickelte eine Modellierungstoolbox, die eine Anzahl von Rationierungsmethoden beinhaltet. Neben einigen aus der Literatur bekannten Rationierungsmethoden beinhaltet diese Toolbox auch Methoden, die speziell für *lagom* konzipiert wurden. Inm Rahmen dieser Arbeit habe ich jede der Rationierungsmethoden ökonomisch und mathematisch dokumentiert. Eine informatische Dokumentation habe ich innerhalb des Programmcodes erstellt und in Appendix A bereitgestellt. Die Anwendbarkeit der Rationierungsmethoden wurde durch das Bearbeiten der zweiten Zielstellung dieser Arbeit bewiesen.

Das zweite Ziel der vorliegenden Arbeit war es, Rationierungsmethoden zu verwenden um einen mikroökonomischen Arbeitsmarkt zu modellieren. Die Literatur, die sich üblicherweise mit der Modellierung von mikroökonomischen Arbeitsmärkten beschäftigt, trifft in der Regel folgende Annahmen: (a) ein Arbeitssuchender bewirbt dich dann auf eine Stelle, wenn der angebotene Lohn den eigenen Referenzlohn – welcher positiv mit der Qualifikation korreliert ist – überschreitet, und (b) Arbeitgeber, die sich einem Pool von Bewerbern gegenübersehen sind nicht in der Lage, diese hinsichtlich ihrer Qualifikationen zu unterscheiden. Ein so modellierter Arbeitgeber kann durch Erhöhen des Lohnes mehr qualifizierte Bewerber anziehen, muss aber dennoch immer blind aus dem tatsächlichen Pool von Bewerbern eine Belegschaft wählen.

Die vorliegende Arbeit hat nun – um Literatur die sich mit Rationierung beschäftigt mit der zu verbinden, die sich mit Arbeitsmärkten beschäftigt – einen veränderten Rahmen geschaffen, in dem Arbeitgeber Belegschaften hinsichtlich ihrer Qualifikation wählen, und in dem Rationierungsmethoden auf den Arbeitsmarkt angewendet werden. Auf vielen Arbeitsmärkten werden die Löhne durch Verhandlungen zwischen Gewerkschaften und Arbeitgeberverbänden festgelegt. Das bedeutet, das weder ein typischer Arbeitgeber, noch ein typischer Arbeitnehmer in solch einem Markt in der Lage ist, den Lohn zu beeinflussen. Ausserdem wählen wohl die meisten Arbeitgeber ihre Belegschaft hinsichtlich der Qualifizierung aus. Der Rahmen, der in der vorliegenden Arbeit zur Modellierung eines Arbeitsmarktes erstellt wurde ist daher der folgende:

Auf einem Arbeitsmarkt, auf dem die Löhne exogen gegeben sind, gibt es eine heterogene Population von Arbeitssuchenden, welche aufeinanderfolgende Arbeitsmöglichkeiten erleben. Jede Arbeitsmöglichkeit besteht aus einer positiven Anzahl von Stellen. Jeder Arbeitssuchende muss sich bei jeder Arbeitsmöglichkeit entscheiden, ob er sich bewirbt oder nicht. Da der Lohn nicht verhandelbar ist, ist der Anreiz zur Bewerbung die von jedem Arbeitssuchenden für sich hergeleitete Wahrscheinlichkeit eine Stelle zu erhalten (perceived probability of obtaining employment conditional on application (im Folgenden PPE)). Nur wenn die PPE eines Arbeitssuchenden eine kritische Wahrscheinlichkeit überschreitet wird er sich bewerben. Um aber seine individuelle PPE zu ermitteln, muss ein Arbeitssuchender auch das Bewerbungsverhalten aller Konkurrenten mit einbeziehen. Da er keine Information darüber hat, wie andere Arbeitssuchende sich verhalten, muss er darüber Hypothesen aufstellen. Jeder Arbeitssuchende besitzt das gleiche Set an Hypothesen, findet aber seine Hypothesen unterschiedlich wahrscheinlich. Dies wird durch das Zuweisen von *initial second-order probabilities* dargestellt. Die Arbeitssuchenden sind heterogen im Bezug auf Ihre Qualifikation und den initial second-order probabilities, die sie ihren Hypothesen am Anfang eines Lernprozesses zuordnen. Über einen Zeitraum, in dem Arbeitsmöglichkeiten aufeinanderfolgen, beobachten sie sich gegenseitig und lernen aus dem Beobachteten für ihre zukünftige Chancenermittlung.

Der Arbeitgeber kann die Qualifizierungen der Bewerber mehr oder weniger gut einschätzen. Dies wurde dadurch modelliert, dass er seine Belegschaft nach der einen oder der anderen Rationierungsmethode auswählt. Die Rationierungsmethode, die der Arbeitgeber verwendet ist allgemein bekannt. Die Fragen die mit Hilfe dieses Rahmens beantwortet wurden sind die folgenden:

1. *Wie können Arbeitssuchende ihre individuelle PPE bei verschiedenen Rationierungsmethoden ermitteln?* Um diese Frage zu beantworten wurde ermittelt, welche Information notwendig ist, um Aussagen über die Wahrscheinlichkeit einer erfolgreichen Bewerbung zu machen. Identifiziert worden sind dabei folgende Informationen: die Qualifikationsverteilung der Population, die Qualifikation des Arbeitssuchenden der seine PPE ermittelt, die Wahrscheinlichkeit jeder möglichen Poolgrösse von Bewerbern, die Rationierungsmethode die der Arbeitgeber verwendet sowie die Anzahl der offenen Stellen. Mit Hil-

fe dieser Information kann ein Arbeitssuchender seine individuelle PPE ermitteln. Für drei Rationierungsmethoden wurde gezeigt, wie die Ermittlung der PPE eines Arbeitssuchenden analytisch aussehen kann. Für jede dieser Methoden wurden Axiome postuliert, welche zu erfüllen sind, und die Methoden wurden bezüglich der Erfüllung der Axiome überprüft. In der Arbeit wurde beispielhaft, anhand einiger Grafiken gezeigt, dass die Axiome von allen Methoden zur Ermittlung der PPE erfüllt werden.

2. *Wie kann man heterogene Agenten modellieren, die Bewerbungsentscheidungen treffen müssen und gleichzeitig das Bewerbungsverhalten ihre Konkurrenten beobachten um aus dieser Information über ihre eigene PPE zu lernen?* Bei der Beantwortung der ersten Frage wurde herausgefunden, dass, um Aussagen über die PPE eines Arbeitssuchenden zu treffen, Informationen über das Bewerbungsverhalten der Konkurrenten des Arbeitssuchenden notwendig sind. Es ist aber wahrscheinlich, dass ein Arbeitssuchender diese Information nicht besitzt. In diesem Fall kann man annehmen, dass ein Arbeitssuchender Hypothesen über das Bewerbunsgverhalten der Konkurrenten aufstellt und jede Hypothese mit einer individuellen Gewichtung versieht. Da Arbeitssuchende unterschiedliche Erfahrungen haben, wird die Gewichtung am Anfang von Indivduum zu Individuum unterschiedlich ausfallen. Um einen Lernprozess abzubilden, wurde in der vorliegenden Arbeit erfolgreich ein Bayesianisches Updating mit einem entscheidenden Merkmal implementiert: Während in der Literatur wie Farmer and Terrell (1996), Lewis and Terrell (2001) und Feltovich and Papageorgiou (2004) üblicherweise nur der Lernprozess von einem Indivduum modelliert, wurde in dieser Arbeit eine große Anzahl von Individuen modelliert, die simultan und durch gegenseitiges Beobachten lernen. Damit trägt diese Arbeit einen Schritt dazu bei, die Implementierung von Lernverhalten in Multiagentenmodelle mit heterogenen Agenten voranzutreiben.

3. *Kann ein Arbeitgeber bei relativ geringer Genauigkeit in der Einschätzung der Qualifikation von Bewerbern eine so qualifizierte Belegschaft einstellen, wie er es täte, wenn er in eine genaue Einschätzung investieren würde?* Die

dritte Frage wurde beantwortet, in dem die Methoden und Konzepte zur Beantwortung der ersten beiden Fragen miteinander kombiniert wurden. Es wurde ein Modell erstellt, in dem eine Population von heterogenen Agenten eine ganze Folge von Arbeitsmöglichkeitenn erfährt und zu jedem Jobangebot passierte das folgende: Jeder Arbeitssuchende ermittelt seine PPE, entscheidet sich für oder gegen eine Bewerbung, aktualisiert seine individuellen second-order probabilities der Hypothesen, das Rekruitment der Belegschaft wird durchgeführt und die Durchschnittsqualifikation der Belegschaft wird ermittelt. Dieses Modell wurde wiederholt ausgeführt und entscheidende Paremeter wurden verändert. Unter Ihnen die Genauigkeit des Arbeitgebers hinsichtlich der Einschätzung der Qualifikation von Bewerbern, die in kleinen Schritten variiert wurde, so dass für eine große Anzahl von Genauigkeiten Aussagen getroffen werden konnten.

Erst bei sehr großer Ungenauigkeit des Arbeitsgebers hinsichtlich der Qualifikation von Bewerbern konnte eine sinkende Durchschnittsqualifikation der Belegschaft festgestellt werden. Durch Untersuchung der Modellstruktur liess sich ermitteln, dass das Ergebnis, wie natürlich alle Ergebnisse, von der Kalibrierung des Modells abhängt: Das Verhältnis von Qualifikationen der Bewerber zu der Ungenauigkeit des Arbeitgebers bestimmt, ob auch bei relativ hoher Ungenauigkeit Bewerber korrekt eingeschätzt werden oder nicht. Das Gesamtergebnis ist daher das folgende: Ein Arbeitgeber kann, wenn die Qualifikation von Bewerbern ein bestimmtes Verhältnis zu seiner Ungenauigkeit hat, auch bei sehr großer Ungenauigkeit eine sehr qualifizierte Belegschaft rekruitieren. In diesem Fall muss er nicht in eine (eventuell zeitintensive, kostenintensive) genaue Kategorisierung seiner Bewerber investieren.

Appendix A

Documented Code: Rationing Toolbox

RANDOM UNIFORM RATIONING

INTRODUCTION
This program contains the code for conducting a two-stage rationing process.
There is a number of suppliers and one a number of demanders. There is either an excess supply or an excess demand. The task is to allocate units of the short side to the long side.

PREPARATION
At the beginning there is an array *lists* that contains two lists, *supply* and *demand*. As a preparation it is determined which of the two lists constituted the short market side, and which the long market side. Then, in order to keep the information until the end of the program, indicators are generated. The long market side is expressed as a new list *excess*. The short side is summarised by the sum of its elements. This sum is kept in a list *shortage*.

RANDOM RATIONING
In a first stage elements from the list *excess are* chosen randomly until the sum of the taken elements is larger or equal to the number found in list *shortage*. Once this is accomplished, there is a number of chosen "agents" that obtain some of the quantity given by the sum of *shortage* .

UNIFORM ALLOCATION
In a second step the uniform allocation rule is conducted. Agents get assigned the minimum between their wish (element in *excess)*, and a share *sh*, determined by dividing the available amount from *shortage*, by the number of agents that have been chosen in the first step.

The program can be applied to any case where there is an excess on one market side and a shortage on the other. It does not matter how many agents there are on each side of the market.
The user needs to fill in correctly two lists into the array *lists*. The first list depicts the supply list and the second the demand list. The program knows this and can determine from this information on which market side there is an excess and on which there is a shortage. It then arranges the lists in the correct way in order to allocate the units to the side which is in excess. The output of the program are again two lists. These lists depitc the realised sales and purchases of the whole population.

```
"Clear all variables";

"Function to call the rationier."
rat[list_]:=Module[{b},

ClearAll[a,b,s,v,rn];

"Definition of a function that computes the sum of the elements of an arbitrary list";
sum[a_]:=Sum[a[[j]],{j,1,Length[a]}];

"lists is an array that contains two lists that depict supply (first list) and demand (second list)";

"lists={{7.88},{7.88}}";

"The sum of each list from array lists is computed";

sumsupply=sum[lists[[1]]];
sumdemand=sum[lists[[2]]];
```

" The short side is determined
An empty list *shortage* is generated. This list will be filled with one element. This element depicts the short side of the market. The shortage is determined: It is the lesser value of *sumsupply* and *sumdemand*. This number is appended to *shortage*. We then create an empty list *remembershortage*. This list is filled with a number that is either *1* or *2*. *1* indicates, that the shortage is supply and *2*

indicates that the shortage is demand. The number corresponds to the position of the supply and demand lists in *lists*. ";

```
shortage={};
AppendTo[shortage,Min[sumdemand,sumsupply]];
remembershortage={};
If[sumdemand>sumsupply,AppendTo[remembershortage,1],AppendTo[remembershortage,2]];
```

"The excess side is determined. A new list *excess* is generated. This list corresponds to the list from *lists* that has the larger sum. The sum had been determined via *sumsupply* and *sumdemand*. The result is a list called *excess* that contains the elements from the excess side of the market.";

```
If[sumsupply>sumdemand,excess=lists[[1]],excess=lists[[2]]];
excess[[1]];
```

"A new empty list s is generated. The random number generator is initialised. Then we define a function *rn* that takes a random element of type integer from the range *1* to the length of list *excess*. Then, while the sum (*sum[s]*) of the elements from list *s* is less than the value of the number in *shortage*, we take a random element from *excess* and append it to *s*. Once the sum of the elements in *s* is equal or larger than the number in *shortage* we quit this process. `positions` is a list where we keep the positions in the original list from the taken elements that are now stored in *excess*. By doing so we can avoid to have elements being picked twice (this is checked when the random number *randomnumber* is drawn), and we have a list that tells us where to put back the elements once we determine sales and purchases of the whole population.";

```
s={};
SeedRandom[1];
rn:= Random[Integer,{1,Length[excess]}];
positions={};

While[
        sum[s]<shortage[[1]],
        randomnumber=rn;

        If[MemberQ[positions,randomnumber],randomnumber=rn,
        AppendTo[positions,randomnumber];
        AppendTo[s,excess[[randomnumber]]]]
        ];

Print["--------------"];
Print ["These were taken from the population             ",s];
Print["These are their positions in the original location ",positions];
```

"*newexcess* is assigned to be s. This should be skipped lateron.";

```
newexcess=s;
```

"A function *nn* is defined. This function subtracts from the length of *newexcess* (which corresponds to the number of elements in *newexcess*) the number of elements in *newexcess* that are equal to zero. The zero elements correspond to demands/supplies that have already been satisfied.";

```
nn:=Length[newexcess]-(Count[newexcess,0]);
```

"*share* is a function that determines a number. This number will be distributed to the elements of *newexcess* later. It is the current value of the element in *shortage* divided by the number of elements in *newexcess* that are not equal to zero.";

```
share:=shortage[[1]]/nn;
sh=share;
```

"Here, a process is started that takes place until the element in *shortage* is equal to zero. Here, in the beginning, it is determined how large the value of the element in *shortage* is. Then we compute *sh*, the share to be allocated. ";

```
While [shortage[[1]]>0,j=1;sh=share;
```

"For each element in *newexcess* it is determined whether it is larger or smaller/equal to *sh*. If an element is larger than *sh*, then from this value it is subtracted *sh*. If it is smaller than *sh*, then this element is set to *zero* (economically this corresponds to the fact that its demand/supply is satisfied). At the same time we subtract the determined value from the element in *sh* too.";

```
    For[j=1,j<Length[newexcess]+1,
        Which [
            newexcess[[j]]>sh,
            newexcess[[j]]=newexcess[[j]]-sh;
            shortage[[1]]=shortage[[1]]-sh,
            newexcess[[j]]<=sh,
            shortage[[1]]=shortage[[1]]-newexcess[[j]];newexcess[[j]]=0];
            ;j++]];
```

"Here we determine the actual sales and purchases of the persons that had been randomly drawn from our population. Therefore we use, as an indicator, the number that we find in *remembershortage*. This number tells us, which side of the market is the short one. If its value is 2, then the short side is the demand side. Then the original list (from *lists*) can be taken in order to depict the purchases of the whole population. However, for the long side of the market we have to make a new list that depicts the sales of the whole population. Therefore we first determine the actual sales of the taken agents. Then we create a new list that we call *sales* and fill it with zeros. It contains as many zeros as there were elements in the original supply list.
The analogous is done when the short side is the other side. ";

```
If[remembershortage[[1]]==2,

purchases=lists[[2]];
For[j=1,j<Length[s]+1,
s[[j]]=s[[j]]-newexcess[[j]];j++];
sales=Table[0,{i,0,Length[excess]-1}],


sales=lists[[1]];
For[j=1,j<Length[s]+1,
s[[j]]=s[[j]]-newexcess[[j]];j++];
purchases=Table[0,{i,0,Length[excess]-1}]];
```

"Now that we have determined the short and the long side of the market we need to create a list for the long side that depicts the sales/purchases of the whole population. Depending on which side of the market is the short side, we take the list that contains all zeros and replace, at the correct positions (determined with the use of *positions*), the zeros by the values from *s*. This is done until all elements from *s* have been put in place. The program is now finished and prints the results. Purchases and sales for the whole population have been determined.";

```
If [ remembershortage[[1]]==2,
For[

i=1,i<Length[positions]+1,
sales=a,
a=ReplacePart[sales,s[[i]],positions[[i]]];
i++]
,
For[

i=1,i<Length[positions]+1,
purchases=b,
b=ReplacePart[purchases,s[[i]],positions[[i]]];
i++]
];


"RESULT"
"-----------------------------------------------------------"
Print["Sales of population      ", sales];
Print["Purchases of population  ", purchases];
Print ["Excess demand/supply     ", newexcess];
```

```
];

"Function to call the rationier.";
lists = {{5., 2., 3.}, {12.}};
rat[lists]

--------------

These were taken from the population              {12.}

These are their positions in the original location {1}

Sales of population      {5., 2., 3.}

Purchases of population  {10.}

Excess demand/supply     {2.}
```

queuing.nb 1

QUEUING RATIONING

Variables and parameters:

demand	ordered list with positive real numbers, depicting demands
supply	ordered list with positive real numbers, depicting supplies

The other variables and parameters appear below and are introduced in the documentation of the program code.

This program conducts rationing of available units regarding a waiting line. The agents on the long side of the market are served in the order in that they appear in the waiting line. The waiting line is depicted by a list of wishes. The first wish gets sevred first, then the second and so one. This is done until no more units to be allocated are left.

```
RatQueue[lists_] := Module[{demand, supply, excess, shortage, remembershortage},

  (*---------DEFINITION OF FUNCTIONS-----------*)
  "This function computes the sum of elements from a given list a.";
     sum[a_] := Return[Sum[a[[j]], {j, 1, Length[a]}]];


  "The function that conducts the rationing
     process after the preparations have been made above. ";
  RationingQueue :=

   For[i = 0, i <= Length[excess],
    Which[
     excess[[i]] >= shortage,
     excess[[i]] = excess[[i]] - shortage;
     shortage = 0
     ,

     excess[[i]] <= shortage,
     shortage = shortage - excess[[i]];
     excess[[i]] = 0


    ] i++
   ];
  (*---------END DEFINITION OF FUNCTIONS--------*)



  (*---------START EXECUTION-------------------*)

  "Below here, the execution of the program starts by preparing the lists,
    determining short side and long side and then calling the function
    RationingQueue that determines remaining excesses. Thereafter the
    purchases and sales made are determined and returned as a list.";

  "The relevant lists, demand depicts market demands and supply
    depicts  market supplies.  Furthermore two lists, pruchases and
    sales, are generated. These will be filled with the results below. ";
  supply = lists[[1]];
  demand = lists[[2]];
```

Printed by Mathematica for Students

```
Print["demand", demand];
purchases = {};
sales = {};


"Here, the program determines the sums of the elements in  demand and supply.";
sumsupply  = sum[supply];
sumdemand = sum[demand];


"It is determined which of the sums is smaller.";
 shortage = Min[sumdemand, sumsupply];
Print["shortage", shortage];


"remembershortage is an indicator to remember the short
  side,  it is equal to 1 if the short side is the supply side
  and it is equal to 2 if the short side is the demand side.";

If[sumdemand > sumsupply, remembershortage = 1, remembershortage = 2];
Print["remembershortage", remembershortage];


"Here it is determined whether there is the need to ration. If the sums of
  demand and supply are equal, then no rationing needs to take place and the
  purchases an sales are equal  to  demand and supply. If this is the case
  the program terminates, returns purchases and sales and prints a message
  to inform  the user. If the sums are not equal, the program continues. ";

If[sumsupply == sumdemand, StylePrint[" NOTE: there was nothing to
    be rationed because  it is an equilibrium of demand and supply.
    Sales and purchases are therefore equal to supply and demand"];
 Return[{supply, demand}],


 "Here, the excess side is determined. As the elements of  the list that depicts
   the excess side needs to be manipulated during the program execution we need
   to keep the list  intact. Therefore we create the list called excess.";

 ClearAll[excess]
  If[sumsupply > sumdemand, excess = lists[[1]], excess = lists[[2]]];
   excess[[1]];
 Print["excess", excess];


 "After the short and long side have been determined tha rationing can
   take place. Therefore the function RationingQueue is called. ";
 RationingQueue;

 "Determination of the sales and purchases lists with the help of
   the indicator remembershortage. If this is equal to 1 then
   the supply side was the short side. sales then correspond to
   the original supply list given at the beginning. Purchases
   are then the difference between the corresponding elements in
   the lists demand and excess. At the end there are two lists,
   sales and purchases, which display the sales and purchases. If
   remembershortage is equal to 2 the process works vice versa.";

 If[
  remembershortage == 1,
```

```
        sales = lists[[1]];

        For[j = 1, j < Length[excess] + 1,
         AppendTo[purchases, demand[[j]] - excess[[j]]];
         j ++]
        ,

        purchases = lists[[2]];
        For[j = 1, j < Length[excess] + 1,
         AppendTo[sales, supply[[j]] - excess[[j]]];
         j ++],

       ];

       Return[{sales, purchases}];
      ]];

   (*---------END  EXECUTION-------------------*)
   (*--------- BEGIN FUNCTION CALL--------------*)
   list = {{1, 2}, {3., 2.}}
   queue = RatQueue[list];
   sales = queue[[1]];
   purchases = queue[[2]];
   Print["sales", sales];
   Print["purchases", purchases];
   (*--------- END FUNCTION CALL---------------*)

Out[10]= {{1, 2}, {3., 2.}}

   demand{3., 2.}

   shortage3

   remembershortage1

   excess{3., 2.}

   sales{1, 2}

   purchases{3., 0.}
```

EGALITARIAN RATIONING

The egalitarian rationing rule. Regarding this rule agents obtain a fixed amount of an available commodity. For that purpose the available amount is divided by the number of agents on that obtain units of it. This rule is not efficient, as agents can be forced to obtain more units than they initially wish.

The following program conducts such an egalitarian rule by allocating units from the short side of the market to the agents on the long side of the market.

```
In[23]:= RatEgalitarian[lists_] :=
   Module[{demand, supply, sumsupply, sumdemand, remembershortage, shortage,
     excess, share},
    "Function to determine the sum of elements in a list.";
     sum[a_] := Return[Sum[a[[j]], {j, 1, Length[a]}]];

    "Determination of the supply and demand lists.";
    supply = lists[[1]];
    demand = lists[[2]];

    "Determination of  the aggregate supplies and demands.";
    sumsupply = sum[supply];
    sumdemand = sum[demand];


    "Determines the side that is in
      excess and produces a list that is called excess that
  holds the corresponding wishes. ";
    If[sumsupply > sumdemand, excess = lists[[1]], excess = lists[[2]]];

  "Determines the short side. shortage is the
      list with the wishes from the short side. It corresponds
      to the transactions that can be realised on that side. ";
    shortage = Min[sumsupply, sumdemand];

    "remembershortage is a parameter that reminds
      which side has been in shortage. It is needed for determining
  the transactions.";
    If[sumdemand > sumsupply, remembershortage = 1, remembershortage = 2];


    "The fixed share that each agent on the short side receives";
    share = shortage / Length[excess];

    "For each agent on the short side share is substracted from his wish. Note that
      here negative values are possible, since agents might be forced to buy
      more than they actually wish.  This computation is not necessary, but
      could be used for finding out which agents had wishes over fulfilled.";
    For[
     i = 0, i < Length[excess],
     excess[[i]] = excess[[i]] - share,
     i++


    ];

    "Transactions of both sides are being determined.
      For the short side this equals the wishes while on the long
      side it equals, for each agent, the share, share. Two lists,
      purchases and sales are being determined and returned.";
```

```
If [
 remembershortage == 1,
 sales = lists[[1]];
 purchases = {};

 For
  [j = 0, j < Length[excess],
  AppendTo[purchases, share],
  j ++],
 purchases = lists[[2]];
 sales = {};

 For
  [k = 0, k < Length[excess],
  AppendTo[sales, share],
  k++];
 Return[{sales, purchases}]
]]
```

In[34]:=

```
(*------------------------Begin Function call "-----------------------*)
liste = {{0., 2.5, 5}, {1., 1.}};
a = RatEgalitarian[liste];
sales = a[[1]];
purchases = a[[2]];
Print["sales ", sales, ", purchases ", purchases]
```

```
sales {0.666667, 0.666667, 0.666667}, purchases {1., 1.}
```

PROPORTIONAL RATIONING

Variables and parameters:

supply	ordered list with real numbers that indicate supplies
demand	ordered list with real numbers indicating demands
lists	a list containing the above lists in the same order

This is the proportional rationing rule. Agents on the short side of the market face rationing. An agent that wishes x% of the aggregated wishes obtains x% of the total amount that is available.

```
RatProportional[lists_] := Module[{supply, demand, sumdemand, sumsupply, shortage},
  "lists={{1.,20.},{3.,1.,2.}}";

  "Function to determine the sum of elements in a list.";
  sum[a_] := Return[Sum[a[[j]], {j, 1, Length[a]}]];

  "Determination of the supply and demand lists.";
  supply = lists[[1]];
  demand = lists[[2]];

  "Determination of  the aggregate supplies and demands.";
  sumsupply = sum[supply];
  sumdemand = sum[demand];

  "Determines the side that is in
    excess and produces a list that is called excess that
holds the corresponding wishes. ";
  If[sumsupply > sumdemand, excess = lists[[1]], excess = lists[[2]]];

"Determines the short side. shortage is the list with the wishes from the short
    side. It corresponds to the transactions that can be realised on that side. ";
  shortage = Min[sumsupply, sumdemand];

  "remembershortage is a parameter that reminds
    which side has been in shortage. It is needed for determining
the transactions.";
  If[sumdemand > sumsupply, remembershortage = 1, remembershortage = 2];


  "The share of the total amount for each agent. This is obtained by
    computing the share of each agent of the aggregate wishes. Therefore
    the sum of the excess wishes is divided by the sum of this list.";

  shareFactors = {};
  For [i = 0, i < Length[excess],
   AppendTo[shareFactors, excess[[i]] / sum[excess]],
   i++];

  "From the elements of the excess list, the transaction that each of
    them is able to realise is subtracted. Therefore each shareFactor
    is multiplied by the total amount available (shortage). The
    resulting amount is the actual transaction for the current agent.";

  transactions = {};
  For[j = 0, j < Length[excess],
    AppendTo[transactions, shareFactors[[j]] * shortage],
    j ++]
```

```
"Here the final transaction are determined. With the help of the
  variable remembershortage it is found out which type the list excess
  is of (demand or supply). Then the lists with transactions are built
  subsequently. At the end purchases and sales are being returned. ";

If [
 (*If short side is demand side*)
 remembershortage == 1,
 sales = lists[[1]];
 purchases = {};
 For
 [k = 0, k < Length[excess],
  AppendTo[purchases, transactions[[k]]],
  k++],
 (*If short side is supply side*)
 purchases = lists[[2]];
 sales = {};
 For
 [l = 0, l < Length[excess],
   AppendTo[sales, transactions[[l]]],
   l++];
];
(*Return list with transactions*)
Return[{sales, purchases}]];

(*------------------Begin Function Call -------------------------*)
(*supply=Input["List of supplies"];
 demand=Input["List of demands"];*)
supply = {1, 2, 3};
demand = {12};

xx = RatProportional[{supply, demand}];
sales = xx[[1]];
purchases = xx[[2]];
Print["sales ", sales, ", purchases ", purchases]
(*------------------End Function Call -------------------------*)

sales {1, 2, 3}, purchases {6}
```

RANKING RATIONING

Variables and parameters:

supply	ordered list with real numbers that indicate supplies
tags	ordered list with real numbers that indicate productivities
demand	ordered list with real numbers indicating demands
lists	a list containing the above lists in the same order

```
"The overall function that needs to be called in order to execute
   the whole rationing process. In it all necessary functions for
   the process are defined and executed in the correct order";
RankingRationing[lists_] := Module[{supply, demand,
   sumdemand, sumsupply, remembershortage, shortage},

  ClearAll[];

  " Function that determines the sum of elements in a list";
  sum[a_] := Return[Sum[a[[j]], {j, 1, Length[a]}]];

  "Initialisation. Extraction of the
    information from the function call and generation of two
    lists that will hold the realised purchases and sales";
  Print["lists", lists];
  supply = lists[[1]];
  tags = lists[[2]];
  demand = lists[[3]];
  purchases = {};
  sales = {};

  "determination of the sums of demand and supply. ";
  sumsupply = sum[lists[[1]]];
  sumdemand = sum[lists[[3]]];

  "Testing whether the sums of demand and supply are equal or not.
    If they are equal, then the program terminates and tells the
    user that there is nothing to be rationed as demands and supplies
    coincide. Purchases and sales are then equal to supplies
    and demands. If the sums of demand and supply are
    not equal to zero, the program continues execution.";

  If[sumsupply == sumdemand,
   StylePrint[" NOTE: there was nothing to be rationed because
      it is an equilibrium of demand and supply. Sales and
      purchases are therefore equal to supply and demand"];
   Return[{supply, demand}],

   "determination of shortage. Because the wished of the short side
    will be fulfilled the shortage it is sufficient to summarise
    it as a sum. In the end the transactions of the short
    side correspond to the original of the short side.";
   shortage = {};
    AppendTo[shortage, Min[sumdemand, sumsupply]];
```

```
 "Here, an indicator is generated that keeps in mind
 which side is the short one. It is set to 1 if the short
 side is the supply side and to 1 if is the other.";
 If[sumdemand > sumsupply, remembershortage = 1, remembershortage = 2];


"Determination of the excess side. This side needs to remain a list
 as we manipulate its elements in the rationing process.";
ClearAll[excess]
 If[sumsupply > sumdemand, excess = lists[[1]], excess = lists[[3]]];
   excess[[1]];


"Function that computes the position
 of the element in supply that corresponds
 to the largest productivity indicated by tags.";
position[a_] := Return[Flatten[Position[a, Max[a]]][[1]]];


"Function to allocate labour
 to the suppliers in case of excess supply.
 Subtracts, from the element in excess that
 has the highest productivity regarding to tags,
 the minimum of the excess/shortage.  If the element
 in excess is been smaller than shortage it sets
 the element and the tag for this element equal
 to zero, so that it won't be allocated to it again. It
 does so through the whole list
 of supplies until shortage is equal to zero. ";
RatioExcessSupply :=
 For [i = 0, i <= Length[excess],

  Which[
    excess[[position[tags]]] >= shortage[[1]],
    excess[[position[tags]]] = excess[[position[tags]]] - shortage[[1]];
    shortage[[1]] = 0,


    excess[[position[tags]]] < shortage[[1]],
    shortage[[1]] = shortage[[1]] - excess[[position[tags]]];
    excess[[position[tags]]] = 0;
    tags[[position[tags]]] = 0;

   ] i++];

"Function to allocate labour in case of excess demand. Works
 similar to the above function. Since the units have to be allocated
 to the demanders, it makes no use of the tags. Rather the
 demanders are served in the order in that they appear
 in the list demand. Then, from the current element,
 it is subtracted the minimum of this element and
 shortage. This is done until shortage is equal to zero.";
RatioExcessDemand :=

 For[i = 0, i <= Length[excess],
```

```
 "Here, an indicator is generated that keeps in mind
 which side is the short one. It is set to 1 if the short
 side is the supply side and to 1 if is the other.";
 If[sumdemand > sumsupply, remembershortage = 1, remembershortage = 2];


"Determination of the excess side. This side needs to remain a list
 as we manipulate its elements in the rationing process.";
ClearAll[excess]
 If[sumsupply > sumdemand, excess = lists[[1]], excess = lists[[3]]];
   excess[[1]];


"Function that computes the position
 of the element in supply that corresponds
 to the largest productivity indicated by tags.";
position[a_] := Return[Flatten[Position[a, Max[a]]][[1]]];


"Function to allocate labour
 to the suppliers in case of excess supply.
 Subtracts, from the element in excess that
 has the highest productivity regarding to tags,
 the minimum of the excess/shortage.  If the element
 in excess is been smaller than shortage it sets
 the element and the tag for this element equal
 to zero, so that it won't be allocated to it again. It
 does so through the whole list
 of supplies until shortage is equal to zero. ";
RatioExcessSupply :=
 For [i = 0, i <= Length[excess],

  Which[
    excess[[position[tags]]] >= shortage[[1]],
    excess[[position[tags]]] = excess[[position[tags]]] - shortage[[1]];
    shortage[[1]] = 0,


    excess[[position[tags]]] < shortage[[1]],
    shortage[[1]] = shortage[[1]] - excess[[position[tags]]];
    excess[[position[tags]]] = 0;
    tags[[position[tags]]] = 0;

   ] i++];

"Function to allocate labour in case of excess demand. Works
 similar to the above function. Since the units have to be allocated
 to the demanders, it makes no use of the tags. Rather the
 demanders are served in the order in that they appear
 in the list demand. Then, from the current element,
 it is subtracted the minimum of this element and
 shortage. This is done until shortage is equal to zero.";
RatioExcessDemand :=

 For[i = 0, i <= Length[excess],
```

```
  the list containing the supplies. A tag at position x in the tags
  list refers to the supply at position x in the supply list!*)
Null

(*-----------------------------------*)
ClearAll[rankinginput];
 rankinginput = {{3, 4., 12.}, {3, 1, 2}, {3, 4}};
 (*-----------------------------------*)
 result = RankingRationing[rankinginput];
 sales = result[[1]];
 purchases = result[[2]];
 Print["purchases", purchases,  "sales", sales];
 (*-----------------------------------*)
```

PIGEONHOLE RATIONING

This program conducts pigeonhole rationing. At pigeonhole rationing, each demand and each supply is characterised by an indicator, a tag. This tag expresses some kind of priority of that agent's wishes (on the labour market it might be interpreted as an indicator for the abilities of a worker). The higher this tag, the higher is the priority of this agents' wishes. But if agents have tags that fall in a certain interval, then their priorities cannot be perceived correctly. Rather, these agents are being perceived as having the same priority. This is accomplished by introducing intervals and modifying each tag into the corresponding intervals' tag. At the moment a tag is a number and the interval it belongs to is found by finding the greatest smaller integer than the tags value. If there are more than one agent in one interval, the short side of the market cannot distingiush between those and agents are picked randomly to allocate the remaining amount of commodity. It is assumed that the agents are assembled randomly from the beginning on. Therefore it is not necessary to randomise the order in which they appear. Rather the algorithm chooses the first maximal element it finds to allocate the remaining amount.

```
In[39]:= RatPigeonhole[lists_] := Module[{supply, demand, indicatorsSupply,
            indicatorsDemand, shortage, sumdemand, sumsupply, drawerIndicators, p},

           "Function to determine the sum of elements in a list.";
           sum[a_] := Return[Sum[a[[j]], {j, 1, Length[a]}]];

           "Determination of the supply and demand lists.";
           supply = lists[[1]][[1]];
           demand = lists[[2]][[1]];

           "The tags that belong to supply and demand.";
           indicatorsSupply = lists[[1]][[2]];
           indicatorsDemand = lists[[2]][[2]];

           "Determination of  the aggregate supplies and demands.";
           sumsupply = sum[supply];
           sumdemand = sum[demand];

           "Abort if aggregate demand equals aggregate supply, continue otherwise.";
           If[sumsupply == sumdemand,
            StylePrint["There is no need for rationing."];
            Return[{supply, demand}],

            "Determines the side that is in
              excess and produces a list that is called excess that
            holds the corresponding wishes. ";
            If[sumsupply > sumdemand,
             excess = lists[[1]][[1]]; excessIndicators = lists[[1]][[2]],
             excess = lists[[2]][[1]]; excessIndicators = lists[[2]][[2]]];

            "Determines the short side. shortage is the
              list with the wishes from the short side. It corresponds to
             the transactions that can be realised on that side. ";
            shortage = Min[sumsupply, sumdemand];

            "remembershortage is a parameter that reminds
              which side has been in shortage. It is needed for determining
             the transactions.";
            If[sumdemand > sumsupply, remembershortage = 1, remembershortage = 2];


            "this is a list of indicators that belongs
              to the excess side (excess). It determines the number of the
              drawer that an excess belongs to. At the moment it is set
```

```
  to be the greatest integer less than the elements' value.";
drawerIndicators = {};
For[
 i = 0, i < Length[excess],
 AppendTo[drawerIndicators, Floor[excessIndicators[[i]]]],
 i++
];


"The postion of the first maximum element from the list drawerIndicators is
  being determined. At that position the corresponding element in excess
  gets assigned the  minimum of its value and the remaining amount
  to be distributed (shortage). This is subtracted from the element
  and from shortage. Then the element in the list drawerIndicators
  is set to zero, so as not to be chosen repeatedly. The process
  begins again until no more units are to be allocated, since shortage
  equals zero.  Since we assume that agents are appearing randomly
  right from the start we need not mix them up at this point.";

While [shortage > 0,

 (*Determination of the position of the larges element from drawerIndicators.
    Needs to be element [[1]][[1]], as otherwise it is a list. *)
 p = Position[drawerIndicators, Max[drawerIndicators]][[1]][[1]];

 (*The current indicator is set to zero, so as to not be chosen repeatedly*)
 drawerIndicators[[p]] = 0;

 If[excess[[p]] > shortage,

  excess[[p]] = excess[[p]] - shortage;
  Print["excess ", excess];
  shortage = 0,

  shortage = shortage - excess[[p]];
  excess[[p]] = 0;
  Print["excess ", excess];

 ]];

"Determination of the transaction for both
  sides. For the short side it is already known and equal to
the original data. With the help of the indicator
  remembershortage we determine which side has been
in excess and subsequently determine this sides transactions
  by subtracting the remaining excess from
the original data.";
If [
 (*Short side is supply side*)
 remembershortage == 1,

 sales = lists[[1]][[1]];
 purchases = {};

 For[j = 0, j < Length[excess],
  AppendTo[purchases, lists[[2]][[1]][[j]] - excess[[j]]],
  j++],
 (*Short side is demand side*)
 purchases = lists[[2]][[1]];
 sales = {};
 For
  [k = 0, k < Length[excess],
```

```
            AppendTo[sales, lists[[1]][[1]][[j]] - excess[[j]]],
            k++];];

          (*Returns the result, lists with sales and purchases being realised.*)
          Return[{sales, purchases}]]];

In[40]:=
     (*---------------------Begin Function Call ---------------------------*)

     lists = {{{0., 3., 1.}, {2.5, 1.2, 1.5}}, {{1., 2., 2.}, {20., 1., 3.5}}};
      xx = RatPigeonhole[lists];
      sales = xx[[1]];
      purchases = xx[[2]];
      Print["sales ", sales, ", purchases ", purchases]

      (*---------------------End Function Call ----------------------------*)

   excess {0, 2., 2.}

   excess {0, 2., 0}

   excess {0, 1., 0}

   sales {0., 3., 1.}, purchases {1., 1., 2.}

Out[40]= Null^5
```

Appendix B
Supplement to Chapter 5

B.1 Development of Individual Prior Second-Order Probabilities Under Different Models and Calibrations

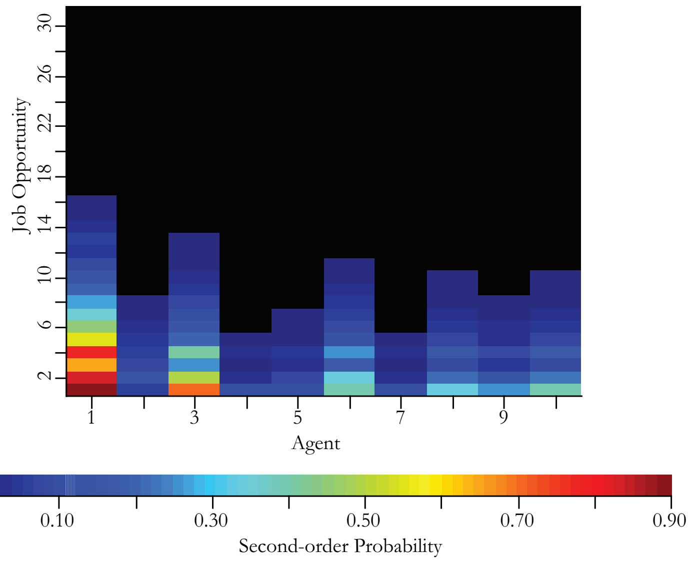

Figure B.1: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 1 (Uniform PDF) under Basic Model with Calibration: 4 Vacancies and Critical Probability 0.7

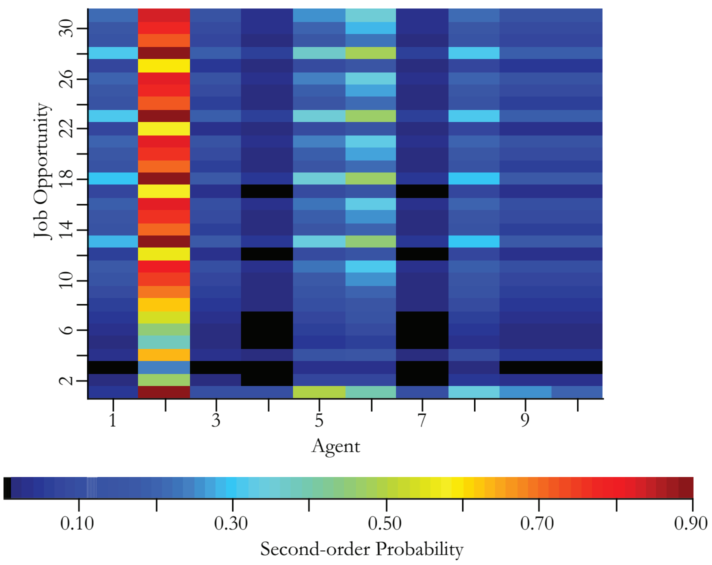

Figure B.2: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 2 (Weibull PDF) under Basic Model with Calibration:4 Vacancies and Critical Probability 0.7

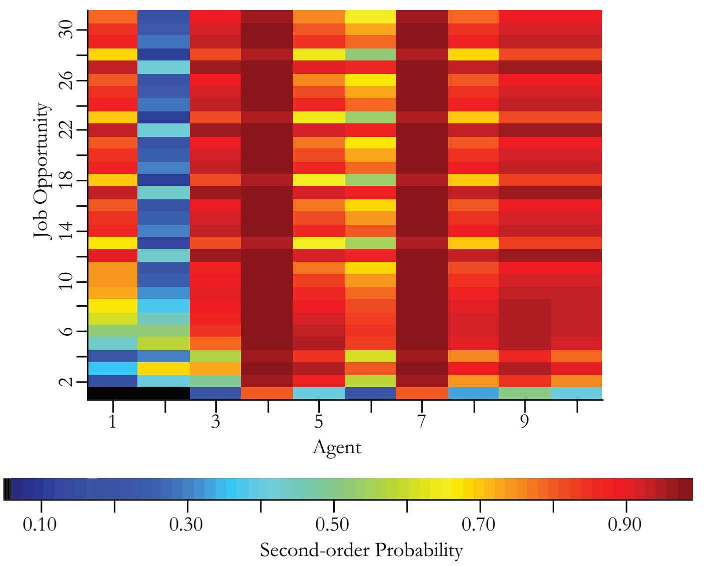

Figure B.3: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 3 (Rayleigh PDF) under Basic Model with Calibration:4 Vacancies and Critical Probability 0.7

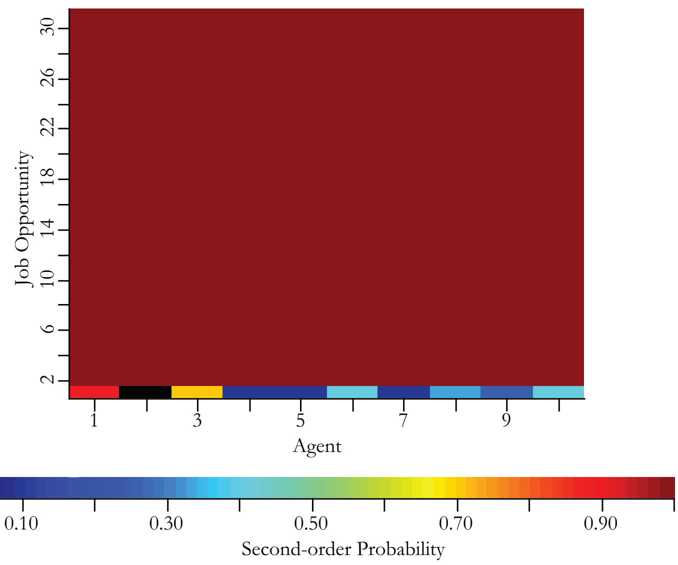

Figure B.4: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 1 (Uniform PDF) under Basic Model with Calibration: 8 Vacancies and Critical Probability 0.9

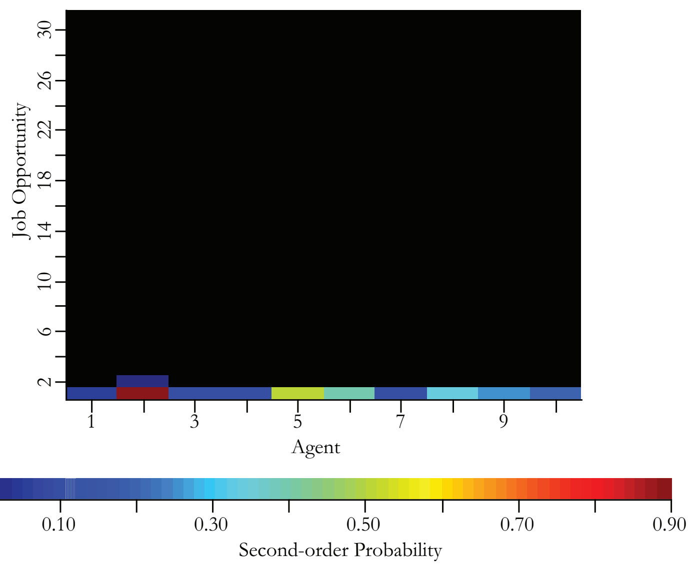

Figure B.5: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 2 (Weibull PDF) under Basic Model with Calibration: 8 Vacancies and Critical Probability 0.9

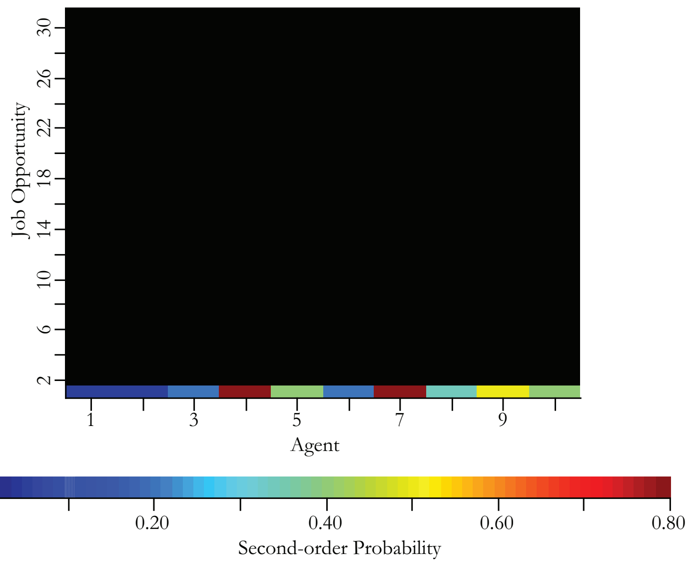

Figure B.6: Development of Individual Prior Second-Order Probabilities Regarding Hypothesis 3 (Rayleigh PDF) at 30 Job offers, 8 Vacancies and Critical Probability 0.9

B.2 Average Productivities of Workforces under Basic and Alternative Model and Different Calibrations

The following figures demonstrate how the average productivity of workforces recruited at different levels of accuracy in the perception of the applicants' productivities depends on that accuracy. These graphs stem from results obtained through a sensitivity analysis of the underlying model. Figures B.7-B.8 correspond to a scenario where 10 agents revise their PPE over a sequence of 30 job opportunities and are recruited at different levels of accuracy; this model has been the basic scenario in Chapter 5. To gain insights regarding the stability of the results, another calibration with 20 agents that revise their PPE over a sequence of 20 job opportunities was subjected to the same sensitivity analysis. Some of the results are demonstrated by Figures B.9-B.12.

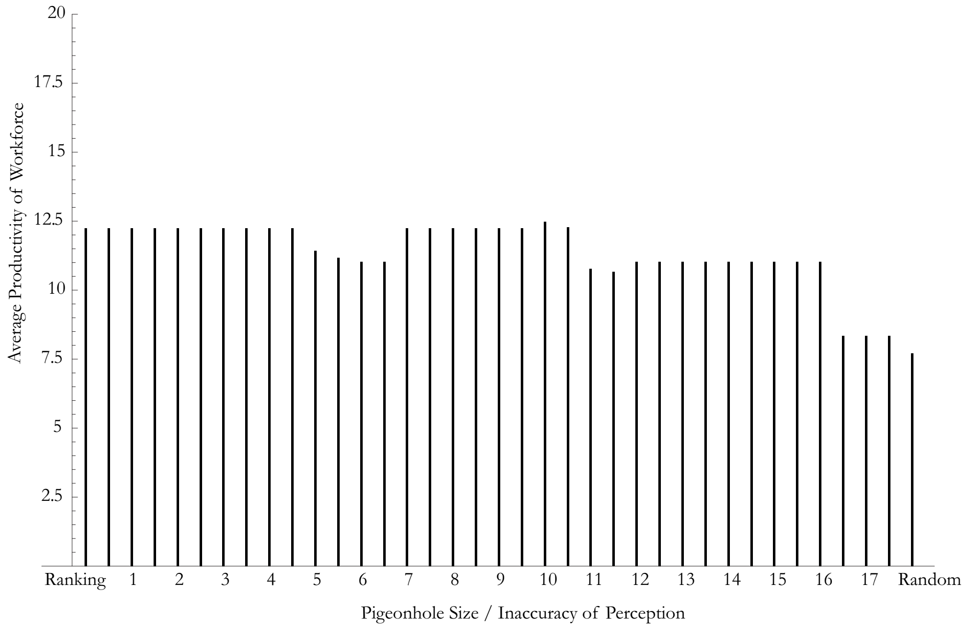

Figure B.7: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy. Basic Model with Calibration: 4 Vacancies, Critical Probability 0.7

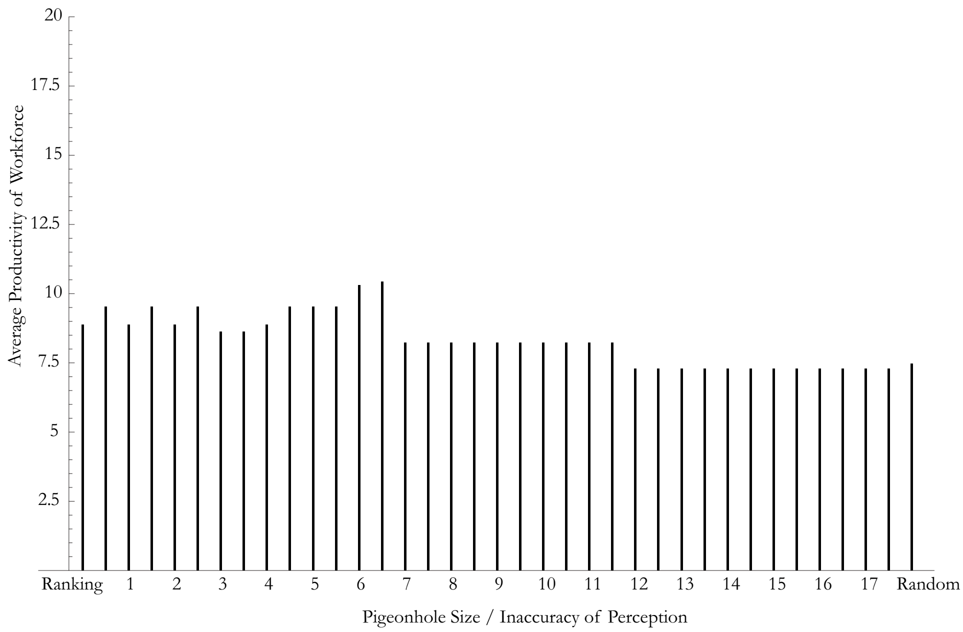

Figure B.8: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy. Basic Model with Calibration: 8 Vacancies, Critical Probability 0.9

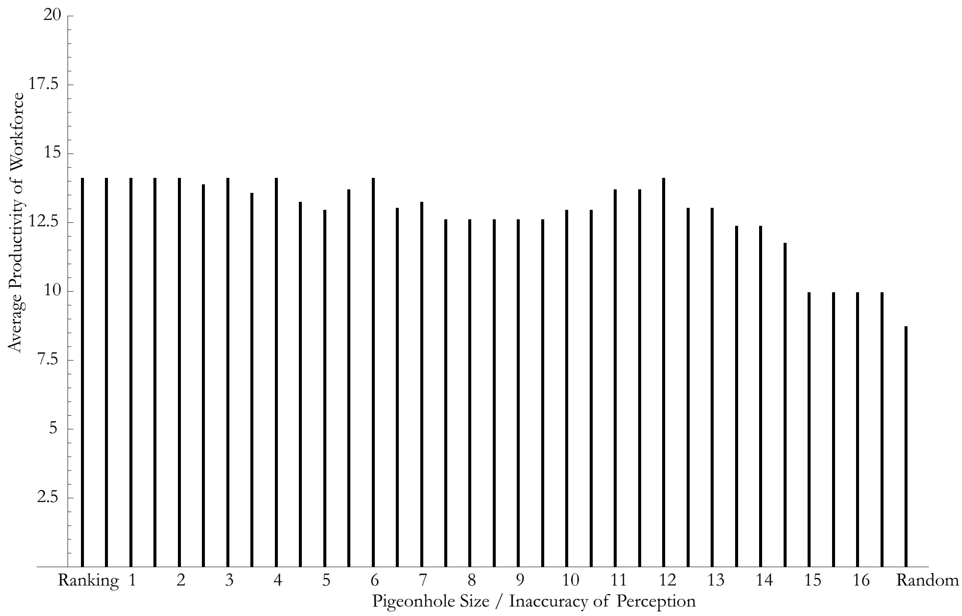

Figure B.9: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy. Alternative Model with 20 Agents, 20 Job Offers, 5 Vacancies, Critical Probability 0.2

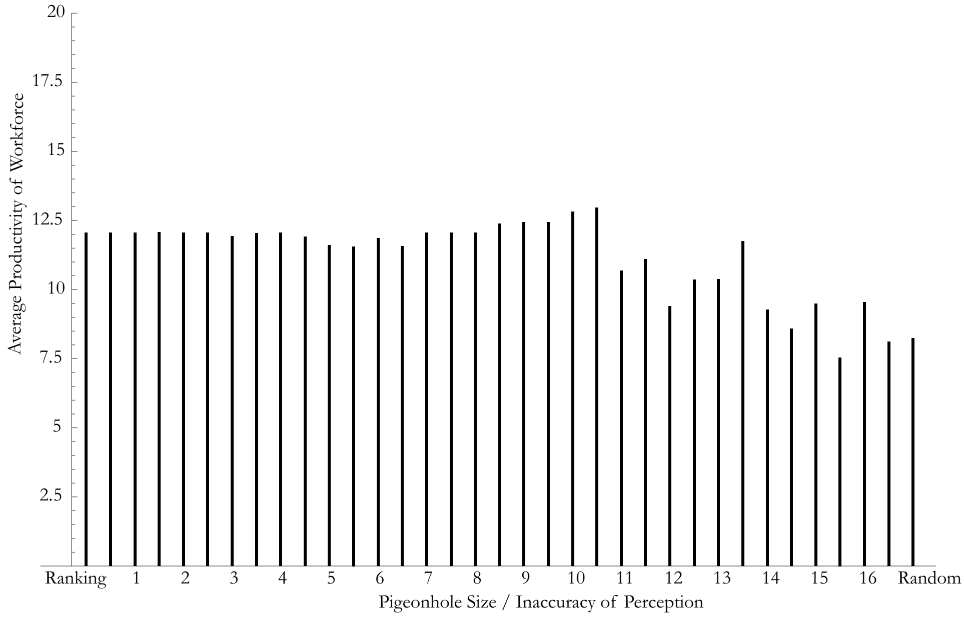

Figure B.10: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy. Alternative Model with 20 Agents, 20 Job Offers, 13 Vacancies, Critical Probability 0.4

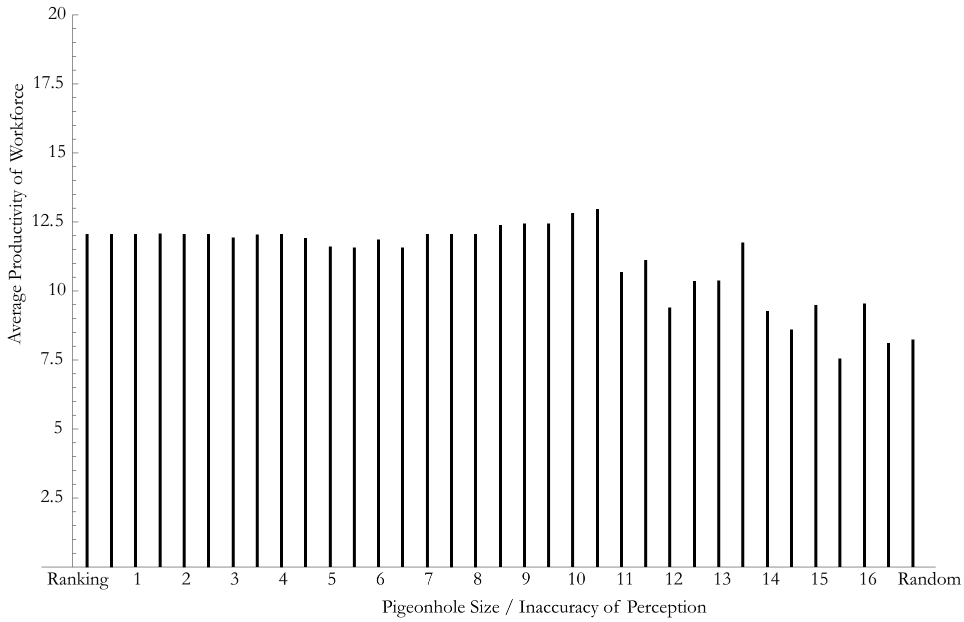

Figure B.11: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy. Alternative Model with 20 Agents, 20 Job Offers, 17 Vacancies, Critical Probability 0.1

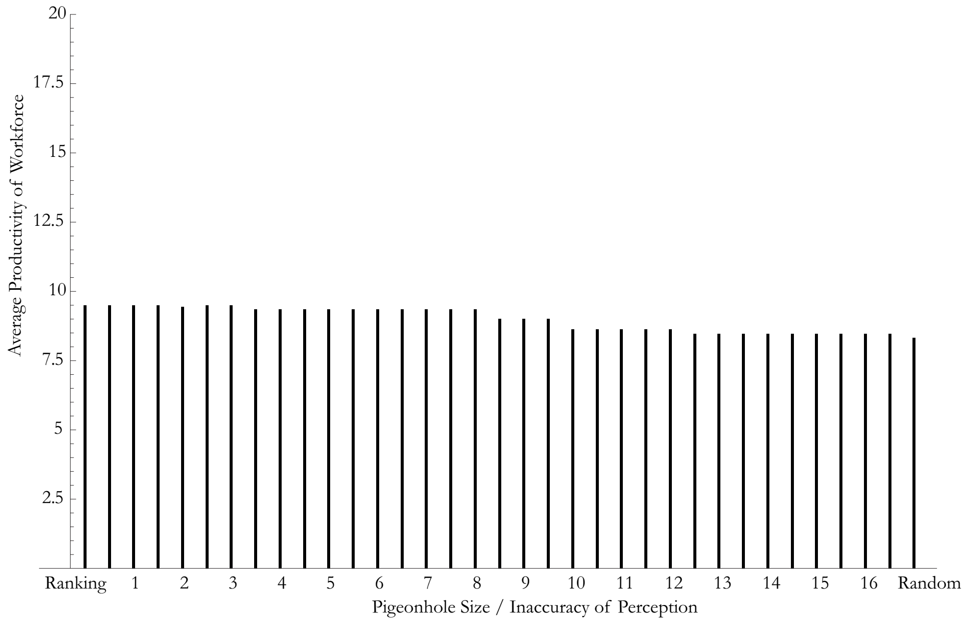

Figure B.12: Average Productivity of the Workforce Recruited under different Levels of Inaccuracy. Alternative Model with 20 Agents, 20 Job Offers, 17 Vacancies, Critical Probability 0.5

B.3 Perceived Probabilities of Obtaining Employment Conditional on Application (PPE): Code

Perceived Probability of Acceptance Conditional to Application at Random Rationing

Variables:

pop number of agents in population
vacancies number of vacancies
results list that holds the probabilities of acceptance for each of the states that are possible for the given scenario
states combination of the agents productivities relative to the observed one
characteristics number of characteristics relative to observed agents (can be less,same,more),value is 3
distrProd density function regarding which the productivities are distributed
distrPool density function regarding which the probabilities of pool sizes are distributed

This program computes perceived probabilities of acceptance conditional to application when the employer conducts random rationing.The productivities of the agents and the probabilities of pool sizes of applicants may be distrbuted regarding arbitrary density functions.The density functions are supplied by the user at function call. Note, that especially for the density function of the pool sizes one must keep in mind how many agents there are in the population in order to calibrate the function correctly, so that the probabilities of the pool sizes add up to one. If this is not done correctly,the algorithm cannot compute correct values.

```
(*Import of statistics packages*)
<< Statistics`ContinuousDistributions`


"The function computes, for all possible sample sizes, the probability of
   being accepted. This is the probability of this sample size multiplied
   by the probability of obtaining a job at this sample size. The
   probability of obtaining a job is simply the number of vacancies
   divided by the number of agents in the current sample. If there are as
   many vacancies as agents in the sample then this probability is one. ";

pprDistr::usage =
  "pprDistr computes the perceived probability of acceptance conditional
    to application if an employer conducts random rationing.
    The function is called like this: ppr[agents_,vacancies_,
    distrPool_]. Substitute for agents_ the number of agents and for
    vacancies_ the number of vacancies. distrPool_ is the density
    function regarding which the poolsizes are distributed.";


pprDistr[pop_, vacancies_, distrPool_] :=
```

$$\sum_{n=1}^{pop}\left((\mathrm{CDF}[\mathrm{distrPool},\ n] - \mathrm{CDF}[\mathrm{distrPool},\ n-1.]) * \frac{\mathrm{If}[\mathrm{vacancies} < n,\ \mathrm{vacancies},\ n]}{n}\right);$$

```
Context[
 CDF]

Statistics`Common`DistributionsCommon`



(*---------------FUNCTION CALL---------------------------------------*)
distrPool = UniformDistribution[0, 3];
pprDistr[3., 3., distrPool]
(*---------------/ FUNCTION CALL---------------------------------------*)
```

Printed by Mathematica for Students

Perceived Probability of Acceptance Conditional to Application at Pigeonhole Rationing

Variables:

```
pop             number of agents in population
agentsprod      productivity of the observed agent
lowerbound      the  lower bound of the agents' drawer
upperbound      the upper bound of the agents' drawer
vacancies       number of vacancies
competitors     number of competitors (agents-1)
pless           probability of agents being less productive than the observed
agent
psame           probability of agents being the same productive as observed agent
pmore           probability of agents being more productive than observed agent
less            being less productive than observed agent
same            being the same productive as observed agent
results         list that holds the probabilities of acceptance for each of the
states that are possible for the given scenario
states          combination of the agents productivities relative to the
observed one
characteristics number of characteristics relative to observed agents (can be
less,same,more), value is 3
distrProd       density function regarding which the productivities are
distributed
distrPool       density function regarding which the probabilities of pool sizes
are distributed
```

This program computes perceived probabilities of acceptance conditional to application when the employer conducts pigeonhole rationing. The productivities of the agents and the probabilities of pool sizes of applicants can be distrbuted regarding arbitrary density functions. The density functions are supplied by the user at function call. Note that especially for the density function of the pool sizes one must keep in mind how many agents there are in the population in order to calibrate the function correctly, so that the probabilities of the pool sizes add up to one. If this is not done correctly, the algorithm cannot compute correct values.

```
(*Here we define several functions that will be used
  below. sum[] computes the sum of elements from a list.*)
sum[a_] := Sum[a[[j]], {j, 1, Length[a]}];
```

```
(*states computes the number of states regarding
  a certain pool size. (= all possible combinations
      of characteristics in a certain pool size)*)
```

$$\text{states} := \frac{(\text{characteristics} + \text{competitors} - 1)\,!}{\text{competitors}\,!\;(\text{characteristics} - 1)\,!};$$

Printed by Mathematica for Students

```
(*allocator is a function that determines the probability of
  acceptance for the observed agent in a certain state. It is the
  number of vacancies minus the number of agents that are more
  productive relative to the observed one divided by the number of
  agents that are the same productive (and that is the total number
   of agents minus the less and more productive agents.) *)

              vacancies - more
allocator := ------------------ ;
             agents - less - more


(*ProbSingle is the function to determine the overall
  probability of being accepted at a given number of competitors
  (=a given pool size). Since we need to consider all possible
  states (combination of characteristics) and in these states all
  possible permutations the formula firstly creates all states. For
  each state it then computes the probability of this state,
 multiplies this by the number of permutations,
 and then multiplies it by the probability for
  the agent to be accepted in this state. The result
  (SumSingle) is is the acceptance probability for the
  observed agent in the current sample (=pool size). *)

results = {};
 (*--------------------------*)


ProbSingleDistr[agents_, vacancies_, pless_, psame_, pmore_] :=
  For[ i = 0, i ≤ competitors,
   If[i == 0, less = 0, less = less + 1]; (*Print["i",i];Print["......."];*)
   For[j = 0, j <= competitors - less, (*Print["j",j]*)
     If[j == 0, same = 0., same = same + 1]; more = competitors - less - same;
     AppendTo[results, {(pless^less * psame^same * pmore^more) *
          competitors !
        ------------------- * Which[allocator >= 1., 1.,
        less ! same ! more !
         allocator <= 0, 0., allocator < 1, allocator]}];
     If[i == competitors, Flatten[results]; SumSingle = sum[results];
      (*Print[SumSingle]*); Break];

     (*Print["less",less];Print["same",same];Print["more",more];*)

     j ++]
    i ++];

(*AcceptanceDistr is the function that computes the overall acceptance
  probability considering all possible sample sizes .It  computes the
  probabilities of acceptance for each possible sample size by calling
  ProbSingleDistr. Then it weighs the result with the probability of
  this sample by considering the probability density function regarding
  which the sample sizes are distributed (distrPool). Finally,
 the sum of all weighed probabilities is computed as the overall
  probability of acceptance conditional to application. *)
```

```
AcceptanceDistr[pop_, vacancies_,
   pless_, psame_, pmore_, distrPool_] := Module[{},
   poolTest = {};
   For [agents = 1., agents <= pop, results = {};
    competitors = agents - 1;
    poolProb = CDF[distrPool, agents] - CDF[distrPool, agents - 1.];
    Print["Pool Probability ", poolProb];
    AppendTo[poolTest, poolProb];
    ProbSingleDistr[agents, vacancies, pless, psame, pmore];
    AppendTo[FinalProbs, poolProb * SumSingle];
    Print["sumsingle", SumSingle];
    If[agents == pop, FinalProbs = Flatten[FinalProbs];
     For [i = 1, i <= Length[FinalProbs],
      AppendTo[FinalCumulatedProb, FinalProbs[[i]]];

      i++];
     finalnew = sum[FinalCumulatedProb];
     poolProof = sum[poolTest];
    ];
    agents ++];
   Return[{finalnew, poolProof}];
  ];
```

$Null^6$

```
(*---------------FUNCTION
  CALL------------------------------------------------------------*)
(* All variables and parameters that are used below are cleared here*)

ClearAll[agentsprod, agents, vacancies, pless, psame, pmore]

 (*Here, the parameters for the computation are set. We need to set the
   agents productivity the number of agents in the population,
  the number of competitors, the number of  vacancies and
   the probabilities of the three characteristics. *)

 (*--------------- PARAMETERS---------
   ------------------------------------------------------*)
 << Statistics`ContinuousDistributions`
 distrProd = UniformDistribution[0, 50.];
 distrPool = UniformDistribution[0, 3.];
 (*distrPool=WeibullDistribution[1.,.2];
  (* calibrated to work for 3 agents in the population *)*)
 agentsprod = 0.000000003;
 pop = 3;
 vacancies = 0;
 lowerbound = Floor[agentsprod];
 upperbound = Ceiling[agentsprod];
 (*NOTE: pless is not allowed to be zero,
  as then indeterminate results are possible. Therefore
   set pless to a very small value in case it was zero.*)
 pless = If[CDF[distrProd, lowerbound] > 0,
    CDF[distrProd, lowerbound], 0.0000000000000001];
 psame = CDF[distrProd, upperbound] - CDF[distrProd, lowerbound];
 pmore = CDF[distrProd, ∞] - CDF[distrProd, upperbound];
 Print["pless ", pless];
 Print["psame ", psame];
 Print["pmore ", pmore];
 (*Lists for results*)
 FinalProbs = {};
 FinalCumulatedProb = {};

 (*--------------- FUNCTION
   CALL------------------------------------------------------------*)
 result = AcceptanceDistr[pop, vacancies, pless, psame, pmore, distrPool];
 ppa = result[[1]]
 poolproof = result[[2]]

 Print["the overall acceptance probability is ........"  , finalnew];
 ClearAll[agents, vacancies, pless, psame, pmore, results];


 (*---------------/ FUNCTION
   CALL------------------------------------------------------------*)
```

Printed by Mathematica for Students

Perceived Probability of Acceptance Conditional to Application at Ranking Rationing

Program that computes the perceived probability of acceptance conditional to application when the employer conducts ranking rationing.

```
    Variables:

    pop         number of agents in population
    vac         number of vacancies
    agentsprod  productivity of observed agent
     pless       probability of being less productive than the observed agent
(can be given or                    determined by a density function.)
     distrPool   density function regarding which the probabilities of sample
sizes are distributed
```

```
(*Here we define several functions that will be used
  below. sum[] computes the sum of elements from a list.*)
sum[a_] := Sum[a[[j]], {j, 1, Length[a]}];


 ppraDistr::usage =
   "ppraDistr  computes the perceived probability of acceptance
     conditional to application. \n \n In order to run this function you
     need to call it like this: \n \n ppraDistr[pop_,vac_,agentsprod_,
     pless_,distrPool_], \n \n where you fill in the corresponding values,
     pop_ for the number of agents, vac_ for the number of vacancies,
     agentsprod_ for the productivity of the observed agent, and pless_
     as the probability of an agent being less able than the observed
     one.  pless must be determined in the code and  can be defined by
     the user (from discrete or continuous density function etc....) .
     distrPool_ must be substituted by a density function regarding which
     the probabilities of the sample sizes are distributed. Note that the
     user needs to import the corresponding statistics package like this:
<<Statistics`ContinuousDistributions`
A function call might look like this:

pop=10.;
vac = 2.;
agentsprod= .5;
<<Statistics`ContinuousDistributions`
distrProd=WeibullDistribution[2,3]
pless=CDF[distrProd,.5];
distrPool=UniformDistribution[0,2.]
ppraDistr[pop, vac, agentsprod, pless,distrPool]
 .";


 "ppraDistr comptes the probability of being accepted
    conditional to application by computing the probability
    that a sufficient number of agents is less productive than
    the observed agent. It does so for all possible sample
    sizes. It weighs the obtained value with the probability
    of that sample size (CD[distrPool,n]-CDF[distrPool,n-1])."; 

 ppraDistr[pop_, vac_, agentsprod_, pless_, distrPool_] := Module[{},
   resultppra = {};
   For[n = 1, n <= agents,
    AppendTo[resultppra,
     If[vac > 0, (CDF[distrPool, n] - CDF[distrPool, n - 1.]) *
       ((pless)^If[n>vac, n-vac, 0]), 0]];
    n++];
   finalresult = sum[resultppra];
   Return[{finalresult}]];


(*---------------FUNCTION CALL---------------------------------------*)
<< Statistics`ContinuousDistributions`
```

```
distrPool = UniformDistribution[0, 3.];
agents = 3.;
vac = 2.;
agentsprod = 3.5;
pless = .25;
ppraDistr[pop, vac, agentsprod, pless, distrPool]
(*---------------/ FUNCTION CALL---------------------------------------*)
```

B.4 Agents, Bayes, and the Labour Market I: Learning by Observing: Code

Agents, Bayes, and the Labour Market I: Learning in the Course of Action

```
<< Statistics`ContinuousDistributions`;
"The three methods to compute the
   perceived probabilities condistional to application";
<< pprae_distr.m
<< SOP_generator.m

(*-----------------Parameters & Preparation -------------------------*)


"Delete any previous files";
DeleteFile["./variables"];
DeleteFile["./output"];

"Critical value for application";


If[ValueQ[criticalValue] == False,
  criticalValue = 0.35,
  criticalValue = criticalValue
 ];


"Number of agents in population.";
pop = 10;


"Number of periods to update";
periods = 30;


"Number of vacancies ";

If[ValueQ[vac] == False,
  vac = 1,
  vac = vac
 ];

"Hypotheses regarding the pool sizes. ";
Pools = {UniformDistribution[0., 10.],
   WeibullDistribution[4., 5.1], RayleighDistribution[1.7]};


"With the help of the module SOP_generaor, as many lists with
   second order probabilities as there are automatically created. For
   each agent, listSOP is called and generates as many second order
   probabilities as hypotheses are assumed in (Pools). Then it appends
   this list (res[[1]]) to (sop).  At the end, (sop) holds lists (
   for each agent one) that contain all second order probabilities. ";

(*sop={};
 For[agent=1,agent≤pop,
  res=listSOP[Length[Pools],0.001];
  AppendTo[sop,res[[1]]],
  agent++];*)
```

```
sop = {{0.9, 0.05, 0.05}, {0.05, 0.9, 0.05},
   {0.7, 0.1, 0.2}, {0.1, 0.1, 0.8}, {0.1, 0.5, 0.4}, {0.4, 0.4, 0.2},
   {0.1, 0.1, 0.8}, {0.333333, 0.333333, 0.333333}, {0.25, 0.25, 0.5},
   {0.4, 0.2, 0.4}};


"Check whether there is as many second order probability
   lists as agents. If not, program is aborted and user is promted
   to correct the error. If there is a correct number of second
   order probability lists, the program continues regularly.";
If [Length[sop] ≠ pop, Print[Length[sop]];
  StylePrint["There are less or more second order probability
     lists than agents. Please correct and run again."]; Abort[],


  "Initialisation of random numbers";
  SeedRandom[3];

  "Productivity density function.";
  distrProd = WeibullDistribution[2, 12.];



  "List with productivities of agents in population. Generates random numbers (as
    many as agents in population (pop)) from given distribution (distrProd)";
  productivities = RandomArray[distrProd, pop];
  Print[productivities];

  (*-----------------/ Parameters & Preparation -------------------------*)

  (*-----------------Main Program ------------------------*)
  "Here, the whole process of determining the individual
     probabilities of being hired, followed by the observation of the
     actual pool size and the updating process starts. The whole program
     runs over the periods given by the parameter (periods) above.;"
   For[period = 1, period <= periods,
    Print["PERIOD        ", period];
    (*sop=sop;*)
    resultsPigeon = {};


    "Generating empty lists for the results,
      as many empty lists as agents in the population (pop)";
    For[r = 1, r ≤ pop,
     AppendTo[resultsPigeon, {}],
     r++];


    "Going through all agents to compute their
      perceived probability of acceptance for each hypothesis";
    For[person = 1, person <= Length[productivities],

     Print["person ", person];

     "determination of productivity of current agent";
     agentsprod = productivities[[person]];
     Print["agentsprod ", agentsprod];
     "probability of being less able than current agent";
     pless = CDF[distrProd, agentsprod];

     "Going through all hypotheses and
```

```
  computing the weighed perceied probability of acceptance";
For [hypo = 1, hypo <= Length[Pools],

 "The current second order probability";
 sopCurrent = sop[[person]][[hypo]];
 Print ["current sop ", sopCurrent];

 "The current density function";
 distrPool = Pools[[hypo]];
 Print["distr Pool ", distrPool];

 (* PIGEONHOLE RATIONING*)
 "Determining the perceived probability at current hypothesis
   and agent at pigeonhole rationing. At first, determination
   of the lowerbound and upperbound for the current agent. This
   determines the drawer in which the agents productivity lies.";
 lowerbound = Floor[agentsprod];
 upperbound = Ceiling[agentsprod];

 "Determination of the probabilities
   of being ess, same or more productive than current agent";
 (*NOTE: pless is not allowed to be zero, as then indeterminate results are
   possible. Therefore set pless to a very small value in case it was zero.*)
 pless = If[CDF[distrProd, lowerbound] > 0, CDF[distrProd, lowerbound],
   0.0000000000000001];
 psame = CDF[distrProd, upperbound] - CDF[distrProd, lowerbound];
 pmore = CDF[distrProd, ∞] - CDF[distrProd, upperbound];

 "Lists to be filled with results from the computation";
 FinalProbs = {};
 FinalCumulatedProb = {};

 "Conducting the algorithm,
   determination of perceived probability, unweighed";
 vacancies = vac;
 Print["current vacancies ", vacancies];
 pigeon = AcceptanceDistr[pop, vacancies, pless, psame, pmore, distrPool];
 resultPigeon = pigeon[[1]];
 Print["unweighed result at pigeon ", resultPigeon];

 "Multiplying the result with the current
   second order probability to obtain the weighed probability";
 finalResultPigeon = sopCurrent * resultPigeon;
 Print["sopCurrent ", sopCurrent];
 Print["weighed result at pigeon ", finalResultPigeon];


 "Appending the final result to a list ith
   perceived probabilities that belongs to the current agent";
 AppendTo[resultsPigeon[[person]], finalResultPigeon];


 "Going to the next hypothesis for current agent";
 hypo++];


"Computing the sum of each persons weighed perceived probabilities. This is
  the overall probability of being accepted, when considering all hypotheses.
  Each list in those resultsRationinName lists corresponds to one agent. ";
resultsPigeon[[person]] = sum[resultsPigeon[[person]]];

"Proceeding to next agent";
```

```
 person++];

(*Print["perceived probs at random, each weighed with corresponding sop ",
  resultsRandom] *)
Print["perceived probs at pigeonhole, each weighed with corresponding so ",
 resultsPigeon];



(*APPLICATION DECISION*)

"Now that probabilities have been determined, each agent checks whether the
  probability is higher than his critical value. If this is the case,
  then he applies, if not then he doesn't. This is at the moment only
  done for pigeonhole rationing, the others will be implemented later. ";

Applicants = {};
"List with critical values of applicants (these have been exceeded)";
criticalsApplicants = {};
"List with actual probabilities of applicants";
probsApplicants = {};


"Going through all agents (=
  decisionmakers) and computing their decision to apply or not. ";
For[decisionmaker = 1, decisionmaker ≤ Length[productivities],

 "If actual probability is larger than critical
   value then agents productivity is appended to Applicants";
 If[ resultsPigeon[[decisionmaker]] ≥ criticalValue,
  AppendTo[Applicants, productivities[[decisionmaker]]];
  AppendTo[criticalsApplicants, criticalValue];
  AppendTo[probsApplicants, resultsPigeon[[decisionmaker]]]
 ];
 decisionmaker++
];
Print["Applicants ", Applicants];



(*OBSERVATION ACTUAL POOL SIZE AND UPDATING PROCESS*)
Print[];
Print ["STARTING UPDATING PROCESS"];
Print[];
"Updating process for the second order probabilities. Each
  agent now updates his second order probabilities by applying the
  bayesian formula. Each agent does this for each hypothesis.";

"The list that will hold the updated second order probabilities";
newsops = {};

For[n = 1, n ≤ Length[sop],
 AppendTo[newsops, {}],
 n++];



"For each agent (=updater), the second order
  probabilities regarding each hypothesis are being updated.";
For [updater = 1, updater ≤ Length[productivities],
```

```
"Going through each hypothesis for the current agent.";
For [sops = 1, sops ≤ Length[sop[[1]]],

 Print["updating agent ", updater];
 Print["updating this SOP ", sops];
 Print["old sop ", sop[[sops]]];

 "Formula for updating the current second order probability.  Is the
   probability of the current pool size mutiplied by the second order
   probability of the current hypothesis, divided by the sum (over the
   different hypotheses) of the probability of the current pool size
   multiplied by the probability of the corresponding hypothesis.";

 updatedsop = ((CDF[Pools[[sops]], Length[Applicants]] -
       CDF[Pools[[sops]], Length[Applicants] - 1]) * sop[[updater]][[sops]]) /
   ( Sum[hyps=1, Length[sop[[1]]]]   (CDF[Pools[[hyps]], Length[Applicants]] -
        CDF[Pools[[hyps]], Length[Applicants] - 1]) * sop[[updater]][[hyps]]);

 sops ++

  "Appending the current updated SOP to (newsops)";
 AppendTo[newsops[[updater]], updatedsop]];


"Going to the next agent that updates his SOPs.";
updater ++];


"Printing old and new second order probabilities";
Print["old sops ", sop ];
Print["new sop", updatedsop];
Print["new sops ", newsops];


"lists for output later";
sop1 = {};
sop2 = {};
sop3 = {};
probs = {};
probs = resultsPigeon;


"Generation of lists for output to SimEnV. For each agent,
  the variables sop1, sop2, sop3 are appended to a corresponding
  list. At the end there is a list for sop1, sop2, sop3 which lists
  the variable values for each agent. This enables an output file
  of the form sop1 for agents 1-x, sop 2 for agents 1-x, ....";


"Add the initia´l second order
  probabilities to the list in the first time step";

If[period == 1,

 For[x = 0, x < pop,
  AppendTo[sop1, sop[[x]][[1]]];
  AppendTo[sop2, sop[[x]][[2]]];
```

Printed by Mathematica for Students

```
    AppendTo[sop3, sop[[x]][[3]]],
    x++];
   variables = Join[sop1, sop2, sop3, probs];
   variables >>> "./variables",

   Print["nicht Periode eins"]];



  "The new initial sops (sop) are the updated sops!";
  sop = newsops;


  For[t = 0, t < pop,
   AppendTo[sop1, sop[[t]][[1]]];
   AppendTo[sop2, sop[[t]][[2]]];
   AppendTo[sop3, sop[[t]][[3]]],
   t++];


  "The lists for the output file are joined and exported to a file called
    vars. The order of the variables in this file  are: sop1, sop2,sop3,
    probs, depicting: the SOP for the first hypothesis, the SOP for the
    second hypothesis, the SOP for the third SOP and the probabilities
    of acceptance. This information spreads over one line. At the
    end of a model run this file has as many lines as time steps.";


  variables = Join[sop1, sop2, sop3, probs];
  Print["vars ", variables];
  variables >>> "./variables";


  period++]];


"Preparation of the output file for SimEnV. Reads the variable
   file and exports to CSV format. One line for one time step";

a = ReadList["./variables"];
Export["./output", a, "CSV"];

(*-----------------/ Main Program -------------------------*)
```

Printed by Mathematica for Students

B.5 Agents, Bayes, and the Labour Market II: Average Productivities of Workforces

© Hannah Förster, foerster@pik-potsdam.de

Agents, Bayes, and the Labour Market II: Average Productivity of Workforces

```
<< Statistics`ContinuousDistributions`;
<< pprae_distr.m
<< SOP_generator.m
<< pigeonhole_labour.m
<< ppr_distr.m

"Function to determine the sum of elements in a list.";
 sum[a_] := Return[Sum[a[[j]], {j, 1, Length[a]}]];


(*----------------------------------------------------------*)
              (*PARAMETERS & PREPARATION *)
(*----------------------------------------------------------*)


avProdPig[] := Module[{},


  "Delete previous files";
  DeleteFile["./variables"];
  DeleteFile["./output"];
  DeleteFile["./avProd"];
  DeleteFile["./output2"];


  "Critical value for application. Only if it is exceeded, an agent applies.";

  If[ValueQ[criticalValue] == False,
   criticalValue = 0.35,
   criticalValue = criticalValue
  ];


"Number of agents.";

pop = 10;


"Number of job offers that are observed.";

timeSteps = 30.;


"Number of vacancies at each job offer.";

If[ValueQ[vacancies] == False,
   vacancies = 1,
   vacancies = vacancies
  ];

Print["vacancies ", vacancies];
"List  to memorize productivities of employees";
```

```
prodWorkMem = {};

"Hypotheses regarding the pool sizes";

(*For 10 Agents*)
Pools = {UniformDistribution[0, 10.],
  WeibullDistribution[4, 5.1], RayleighDistribution[1.7]};

(* For 20 Agents *)
(*Pools={UniformDistribution[0,20.],
    WeibullDistribution[4.5,10.9],RayleighDistribution[3.7]};*)

(*For 40 Agents*)
(*Pools={UniformDistribution[0,40.],
    WeibullDistribution[5.9,24.9],RayleighDistribution[7.4]};*)


"The prior SOP for each agent and each hypothesis. These are either determined
  manually, as here, or via the method SOP_generator as in the code below. ";

(*For 10 Agents*)
sop = {{0.9, 0.05, 0.05}, {0.05, 0.9, 0.05}, {0.7, 0.1, 0.2},
   {0.1, 0.1, 0.8}, {0.1, 0.5, 0.4}, {0.4, 0.4, 0.2}, {0.1, 0.1, 0.8},
   {0.333333, 0.333333, 0.333333}, {0.25, 0.25, 0.5}, {0.4, 0.2, 0.4}};


"With the help of the module SOP_generaor, as many lists with
  second order probabilities as there are automatically created. For
  each agent, listSOP is called and generates as many second order
  probabilities as hypotheses are assumed in (Pools). Then it appends
  this list (res[[1]]) to (sop). At the end, (sop) holds lists (
  for each agent one) that contain all second order probabilities.";

(*For 20/40 agents*)
(*sop={};
 For[agent=0,agent<pop,
  res=listSOP[Length[Pools],0.001];
  AppendTo[sop,res[[1]]],
  agent++];*)


"Test whether there are as many lists with prior SOPs
  as agents. If not, program is aborted and user is prompted to
  correct the error, otherwise the program continues regularly.";

If [Length[sop] ≠ pop, Print[Length[sop]];
  StylePrint["There are less or more second order probability
     lists than agents. Please correct and run again."];


  Abort[],


  "The random number generator is initialised.";
  SeedRandom[3];


  "The density function regarding
    that the productivities are assumed to be distributed.";
```

```
distrProd = WeibullDistribution[1.08, 5.];

"List with productivities of agents in pop. Generates as many random
  numbers  as agents (pop) from given distribution (distrProd)";

productivities = RandomArray[distrProd, pop];
Print["productivities ", productivities];


"Increment for the list with bounds. If input
  is 0.0, then it is going to set the ranking indicator to 1,
  if there comes a 1000, sets the random indicator to 1.";


If[
 ValueQ[increment] == False,
 increment = 1.0;
 bounds = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18},
 increment = increment
];


Which[increment == 0.0, rank = 1;
 Print["increment ", increment];
 random = 0,
 increment == 1000.0,
 random = 1;
 Print["increment ", increment];
 rank = 0,

 (*This is the lowest bound possible. Since it can not be set to zero,
  as this is used by the pigeonhole rationing method as an indicator,
  it is set to a very small value.*)

 bounds = {0.00000000000001};

 increment ≠ 0 && increment ≠ 1000,
 rank = 0;
 random = 0;
 i = 0;
 While[
  i < Max[productivities],
  AppendTo[bounds, i + increment];
  i += increment]
];

Print["bounds ", bounds];


"A list with lower productivity bounds (lowerBoundings) is constructed. For
  each agent it is determined which element from (bounds) is the one closest
  to his productivity. This is done by adding his productivity to the list (
  boundsDet), sorting this list and then finding out the previous element.
  If however, the pigeonholes are so small that the recruitment corresponds
  to ranking rationing, then the lowerBound is the agent's productivity
  minus a small factor. If the recruitment follows along the lines of
  random rationing, then the lowerBounds are set to zero for each agent.";

lowerBoundings = {};
For[g = 1, g <= Length[productivities],
```

```
If[

 (*If pigeonhole rationing happens*)
 rank == 0 ,

 If[

  random == 0,
  Print["pigeonholing"];
  boundsDet = bounds;
  prod = productivities[[g]];
  AppendTo[boundsDet, prod];
   boundsDet = Sort[boundsDet];
  pos = Position[boundsDet, prod][[1]][[1]];
  lowerBound = If[pos == 1, boundsDet[[pos]], boundsDet[[pos - 1]]];
  AppendTo[lowerBoundings, lowerBound],

   (*If random rationing happens*)
  Print ["random"];
  For [ i = 1, i ≤ pop,
   AppendTo[lowerBoundings, 0.],
   i++];

 ],

 (*If ranking happens*)

 Print["ranking "];
 AppendTo[lowerBoundings, productivities[[g]] - 0.0000000000001];
 (*For upper and lower bounds below.*)
 bounds = lowerBoundings;

];
g++];


(*----------------------------------------------------------*)
              (*START OF MAIN PROGRAM*)
(*----------------------------------------------------------*)
"Thole process of determining the individual PPAs, the application
  decision, the choice of employees,  the observation of the actual
  pool size and the updating process starts. The program runs over the
  number of job offers given by the parameter (timeSteps) above.";

For[period = 1, period <= timeSteps,
 Print["PERIOD        ", period];

 resultsPigeon = {};


 "Generating empty lists for the results,
   as many empty lists as agents in the population (pop).";

 For[r = 1, r ≤ pop,
  AppendTo[resultsPigeon, {}],
  r++];


 "Going through all agents (=
   person) to compute their PPA at each hypothesis.";
 For[
```

```
person = 1, person <= Length[productivities],

(*Print["person ", person];*)

"Determination of productivity of current agent.";

agentsProd = productivities[[person]];
(*Print["agentsProd ", agentsProd];*)


"Determining the PPA at current hypothesis and current agent. At first,
  determination of the lowerBound and upperbound for the current agent.
  This determines the pigeonhole in which the agent's productivity lies.";

boundsDet = bounds;
AppendTo[boundsDet, agentsProd];
   boundsDet = Sort[boundsDet];
   pos = Position[boundsDet, agentsProd][[1]][[1]];
   lowerBound =
 If[pos == 1, boundsDet[[pos]], boundsDet[[pos - 1]]]; upperbound =
 If[pos + 1 > Length[boundsDet], boundsDet[[pos]], boundsDet[[pos + 1]]];


"Going through all hypotheses (=hypo) and computing the weighed PPA";

For [hypo = 1, hypo <= Length[Pools],


 "The current prior SOP";
 sopCurrent = sop[[person]][[hypo]];


 "The current hypothesis";
 distrPool = Pools[[hypo]];



 (*---------------------------------------------------------*)
      (* DETERMINATION OF PERCEIVED PROBABILITIES*)
 (*---------------------------------------------------------*)

 "Determination of the probabilities of being less, same
   or more productive than current agent. This is obtained by
   integrating the underlying density function appropriately.";
 (*NOTE: pless is not allowed to be zero,
  as then indeterminate results are possible. Therefore
   set pless to a very small value in case it was zero.*)

 pless =
  If[CDF[distrProd, lowerBound] > 0, CDF[distrProd, lowerBound], .00000000001];
 psame = CDF[distrProd, upperbound] - CDF[distrProd, lowerBound];
 pmore = CDF[distrProd, ∞] - CDF[distrProd, upperbound];


 "Lists to be filled with results from the computation";

 FinalProbs = {};
 FinalCumulatedProb = {};


 "Conducting the algorithm, determination of unweighed PPAs";
 (*Print["current vacancies ", vacancies];*)
```

```
 If[random == 0,

  (*If the indicator for random is zero, do pigeonhole rationing*)

  pigeon = AcceptanceDistr[pop, vacancies, pless, psame, pmore, distrPool];
  resultPigeon = pigeon[[1]],

  (*else do random rationing*)
  resultPigeon = pprDistr[pop, vacancies, distrPool];

 ];


 "Multiplying the result
   with the current prior SOP to obtain the weighed PPA";

 finalResultPigeon = sopCurrent * resultPigeon;
 (*Print["weighed result at pigeon ", finalResultPigeon];*)


 "Appending the final result to
   a list with PPAs that belongs to the current agent";

 AppendTo[resultsPigeon[[person]], finalResultPigeon];


 "Going to the next hypothesis for current agent";

 hypo++];

"Computing the sum of each agent's weighed PPA. This is the overall
  probability of being accepted, when considering all hypotheses.
  Each list in those (resultsPigeon) corresponds to one agent. ";

resultsPigeon[[person]] = sum[resultsPigeon[[person]]];


"Proceeding to next agent";

person++];


Print["perceived probs at pigeonhole, each weighed with corresponding sops ",
 resultsPigeon];



(*---------------------------------------------------------*)
             (*APPLICATION DECISION OF AGENTS*)
(*---------------------------------------------------------*)

"Now that the PPAs have been determined, each
  agent checks whether his PPA exceeds his critical value (
  criticalValue). If this is the case he applies, else he doesn't.";

"List to be filled with productivities of applicants.";
```

```
Applicants = {};

"List with critical values of applicants (these have been exceeded)";

criticalsApplicants = {};

"List with actual probabilities of applicants";

probsApplicants = {};

"Going through all agents (=
  decisionmakers) and computing their application decision. ";

For[
 decisionmaker = 1, decisionmaker ≤ Length[productivities],

 "If the PPA is larger than the agent's critical value, the
   agent's productivity is appended to the list (Applicants).";

 If[ resultsPigeon[[decisionmaker]] ≥ criticalValue,
  AppendTo[Applicants, productivities[[decisionmaker]]];
  AppendTo[criticalsApplicants, criticalValue];
  AppendTo[probsApplicants, resultsPigeon[[decisionmaker]]]
 ];
 decisionmaker++
];

Print["Applicants ", Applicants];


(*----------------------------------------------------------*)
        (* RATIONING AND DETERMINATION OF WORKFORCE *)
(*----------------------------------------------------------*)
"Determination of the lower productivity bounds of the applicants.  For that
  purpose, the position (pos) of each applicant in the list of  productivities
  is determined. Then the element at that position in  lowerBoundings
  is taken and appended to the new list (lowerBoundsApplicants). ";


lowerBoundsApplicants = {};
laboursupply = {};
For[
 i = 0, i < Length[Applicants],
 pos = Position[productivities, Applicants[[i]]][[1]][[1]];
 AppendTo[lowerBoundsApplicants, lowerBoundings[[pos]]];
 AppendTo[laboursupply, 1.0],
 i++
];

Print["lower bound appl ", lowerBoundsApplicants];


If[random == 0,
 (*Pigeonhole*)

 "Choice of the workforce by conducting pigeonhole rationing. Therefore, the
```

```
    method RatPigeonhole is called with the argument lists that contains
    information about the labour supplies of the applicants, the number
    of vacancies and the lower productivity bounds of the applicants.";

  lists = {laboursupply, {vacancies}, lowerBoundsApplicants};
  rat = RatPigeonhole[lists];
  sales = rat[[1]];
  Print["sales ", sales],

  (*Random*)

  Print["choice via random"];

  "Choice of workforce by random rationing. Random Applicants are being
    chosen until the number of vacancies is filled. The labour supply
    of applicants that obtain a job is set to zero. At the end, to
    sales is appended a one is an agent has no more labour supply
    or a zero, if an agent has not realised his labour supply.";

  remainingApplicants = Applicants;

  For[v = 1, v ≤ vacancies,

   pos = Random[Integer, {1, Length[remainingApplicants]}];
   laboursupply[[pos]] = 0;
   Delete[remainingApplicants, pos];

   v++];

  sales = {};
  For[x = 1, x <= Length[laboursupply],


   If [laboursupply[[x]] == 0,
    AppendTo[sales, 1],
    AppendTo[sales, 0]
   ];

   x++]

 ];

 Print["sales finally ", sales]



  "Productivities of workforce are being determined by appending
    to the list (productivitiesWorkfoce) the productivities of the
    agents that have been employed. This is done by finding out in
    the list of sales the elements that are equal to one. Then at the
    corresponding position in (Applicants) the productivity of the
    agent is extracted and appended to (productivitiesWorkforce).";

 productivitiesWorkforce = {};
 For[
  a = 0, a < Length[Applicants],
  If[
   sales[[a]] == 1,
   AppendTo[productivitiesWorkforce, Applicants[[a]]];
   AppendTo[prodWorkMem, Applicants[[a]]],
   Print[]
```

```
  ],
  a++
 ];

Print[productivitiesWorkforce];
Print[prodWorkMem];

(*--------------------------------------------------------*)
   (*OBSERVATION ACTUAL POOL SIZE AND UPDATING PROCESS*)
(*--------------------------------------------------------*)
"Updating process for the prior SOPs. Each
  agent now updates his prior SOPs by applying the Bayesian
  theorem. Each agent does this for each of his hypotheses.";

"The list that will hold the updated prior SOPs";

newsops = {};

For[
 n = 1, n ≤ Length[sop],
 AppendTo[newsops, {}],
 n++
];


"For each agent (=updater), the prior
      SOPs  regarding each hypothesis are being updated.";

For[
 updater = 1, updater ≤ Length[productivities],


 "Going through each hypothesis for the current agent.";

 For[
  sops = 1, sops ≤ Length[sop[[1]]],


  "Formula for updating the current prior SOP.  It is the probability of the
    current pool size (FOP) mutiplied by the prior SOP of the current
    hypothesis, divided by the sum (over the different hypotheses)
    of the probability of the current pool size multiplied by the
    probability of the corresponding hypothesis (marginal probability).";

  updatedsop = ((CDF[Pools[[sops]], Length[Applicants]] -
        CDF[Pools[[sops]], Length[Applicants] - 1]) * sop[[updater]][[sops]]) /
    (Sum[(CDF[Pools[[hyps]], Length[Applicants]] -
        CDF[Pools[[hyps]], Length[Applicants] - 1]) * sop[[updater]][[hyps]],
      {hyps, 1, Length[sop[[1]]]}]);

  sops++

   "Appending the current updated prior SOP to (newsops)";

  AppendTo[newsops[[updater]], updatedsop]];
```

$$\text{updatedsop} = \frac{(\text{CDF}[\text{Pools}[[\text{sops}]], \text{Length}[\text{Applicants}]] - \text{CDF}[\text{Pools}[[\text{sops}]], \text{Length}[\text{Applicants}] - 1]) * \text{sop}[[\text{updater}]][[\text{sops}]]}{\sum_{\text{hyps}=1}^{\text{Length}[\text{sop}[[1]]]} (\text{CDF}[\text{Pools}[[\text{hyps}]], \text{Length}[\text{Applicants}]] - \text{CDF}[\text{Pools}[[\text{hyps}]], \text{Length}[\text{Applicants}] - 1]) * \text{sop}[[\text{updater}]][[\text{hyps}]]}$$

```
 "Going to the next agent that updates his SOPs.";

 updater++];


"Lists for output regarding the updated probabilities of the hypotheses";
sop1 = {};
sop2 = {};
sop3 = {};



"Generation of lists for output to SimEnv. For each agent,
  the variables sop1, sop2, sop3 are appended to a corresponding
  list. At the end there is a list for sop1, sop2, sop3 which lists
  the variable values for each agent. This enables an output file
  of the form sop1 for agents 1-x, sop 2 for agents 1-x, ....";


"Adds the initial prior SOPs to the list in the first time step. If
  it is not the first time step then it adds the new prior SOPs.";

If[period == 1,

 For[x = 0, x < pop,
  AppendTo[sop1, sop[[x]][[1]]];
  AppendTo[sop2, sop[[x]][[2]]];
  AppendTo[sop3, sop[[x]][[3]]],
  x++];
 variables = Join[sop1, sop2, sop3, {0.}];
 variables >>> "./variables",
 Print[]
];


sop1 = {};
sop2 = {};
sop3 = {};


For[t = 0, t < pop,
 AppendTo[sop1, newsops[[t]][[1]]];
 AppendTo[sop2, newsops[[t]][[2]]];
 AppendTo[sop3, newsops[[t]][[3]]],
 t++];



"The lists for the output file are joined and exported to a file called
  variables. The order of the variables in this file  are: sop1,
  sop2,sop3,probs, depicting: the SOP for the first hypothesis, the
  SOP for the second hypothesis, the SOP for the third SOP and the
  probabilities of acceptance. This information spreads over one line.
  At the end of a model run this file has as many lines as time steps.";


noApplicants = {Length[Applicants]};

variables2 = Join[sop1, sop2, sop3, noApplicants];
variables2 >>> "./variables";
```

```
    "The new initial sops (sop) are the updated sops!";
    Print["new sops ", newsops];
    sop = newsops;
    Print["new sops ", sop];


    period++]];

"Determination of average productivity by dividing the sum of the
    elements from (prodWorkMem) by the number of elements of this list.";

  avProd = sum[prodWorkMem] / Length[prodWorkMem];
Print["The average productivity of the workforce is ", avProd];
  avProd >> "./avProd";

  a = ReadList["./variables"];
  Export["./output", a, "CSV"];
  b = ReadList["./avProd"];
  Export["./output2", b, "CSV"];
 ]

(*--------------PROGRAM START-----------*)
avProdPig[ ];
(*-------------------------------------*)
```

Bibliography

ARROW, K. J., AND G. DEBREU (1954): "Existence of an Equilibrium for a Competitive Economy," *Econometrica*, 22(3), 265–90.

AXTELL, R. (2006): "Multi-Agent Systems Macro: A Prospectus," in *Post Walrasian Macroeconomics: Beyond the Dynamic Stochastic General Equilibrium Model*, ed. by D. Colander, pp. 203–20. Cambridge University Press, Cambridge.

BACKHOUSE, R., AND M. BOIANOVSKY (2003): "Disequilibrium Macroeconomics: An Episode in the Transformation of Modern Macroeconomics," Working paper, University of Birmingham.

BARBERA, S., AND M. O. JACKSON (1995): "Strategy-Proof Exchange," *Econometrica*, 63(1), 51–97.

BARRO, R. J., AND H. I. GROSSMAN (1976): *Money, Employment and Inflation*. Cambridge University Press, Cambridge, 1st edn.

BENASSY, J.-P. (1975): "Disequilibrium Exhange in Barter and Monetary Economies," *Economic Inquiry*, 12, 131–56.

——— (1977): "On Quantity Signals and the Foundations of Effective Demand Theory," *Scandinavian Journal of Economics*, pp. 147–68.

——— (1982): *The Economics of Market Disequilibrium*, Economic Theory, Econometrics, and Mathematical Economics. Academic Press Inc., London, 1st edn.

——— (1986): *Macroeconomics: An Introduction to the Non-Walrasian Approach*, Economic Theory, Econometrics, and Mathematical Economics. Academic Press, Orlando.

——— (2002): *The Macroeconomics of Imperfect Competition and Nonclearing Markets*. MIT Press, Cambridge.

BLAUG, M. (1996): *Economic Theory in Retrospect*. Cambridge University Press, Cambridge, 5th edn.

BREEN, R., AND C. GARCÍA-PEÑALOSA (2002): "Bayesian Learning and Gender Segregation," *Journal of Labor Economics*, 20(4), 899–922.

BURDETT, K., AND R. WRIGHT (1998): "Two-Sided Search with Nontransferable Utility," *Review of Economic Studies*, 1, 220–45.

CACHON, G. P., L. M. A. (1999): "An Equilibrium Analysis of Linear, Proportional and Uniform Allocation of a Scarce Good," *IIE Transactions*, 31, 835–49.

CEPA (2006a): "Don Patinkin," `http://cepa.newschool.edu/het/profiles/patinkin.htm`.

——— (2006b): "The Neoclassical-Keynesian Synthesis," `http://cepa.newschool.edu/het/schools/synthesis.htm`.

CHING, S. (1992): "A Simple Characterization of the Uniform Allocation Rule," *Economic Letters*, 40, 57–60.

——— (1994): "An Alternative Characterization of the Uniform Allocation Rule," *Social Choice and Welfare*, 11, 131–6.

CLOWER, R. (1965): "The Keynesian Counterrevolution: A Theoretical Approach," in *The Theory of Interest Rate*, ed. by F. Hahn, and F. Brechling, pp. 103–25. Macmillan and Co Ltd., London.

CORNELL, B., AND I. WELCH (1996): "Culture, Information, and Screening Discrimination," *Journal of Political Economy*, 104(3), 542–71.

DEBREU, G. (1959): *Theory of Value. An Axiomatic Analysis of Economic Equilibrium*. Cowles Foundation, Yale.

DENKER, M. (2005): *Einführung in die Analysis dynamischer Systeme*. Springer, Berlin.

DIXON, R. (2000a): "Walras Law: A Proof," http://www.economics.unimelb.edu.au/rdixon/walproof.html.

——— (2000b): "Walras Law and Macroeconomics," http://www.economics.unimelb.edu.au/rdixon/wlaw.html.

DRAZEN, A. (1980): "Recent Developments in Macroeconomic Disequilibrium Theory," *Econometrica*, 48(2), 283–306.

EUROSTAT (2006): "Total Unemployment Rate," http://epp.eurostat.ec.europa.eu/portal/page?_pageid=1996,39140985&_dad=portal&_schema=PORTAL&screen=detailref&language=en&product=STRIND_EMPLOI&root=STRIND_EMPLOI/emploi/em071.

FARMER, A., AND D. TERRELL (1996): "Discrimination, Bayesian Updating of Employer Beliefs, and Human Capital Accumulation," *Economic Inquiry*, 34, 204–19.

FELDERER, AND HOMBURG (2005): *Makroökonomik und neue Makroökonomik*. Springer Verlag, 9th edn.

FELTOVICH, N., AND C. PAPAGEORGIOU (2004): "An Experimental Study of Statistical Discrimination by Employers," *Southern Economic Journal*, 70(4), 837–49.

FLECHSIG, M., U. BÖHM, T. NOCKE, AND C. RACHIMOV (2005): "The Multi-Run Simulation Environment SimEnv - User Guide for Version 1.23," Intruction manual, Potsdam Institute for Climate Impact Research.

FUHRMANN, W. (1988): "Stabilisierungspolitik bei 2 Mio. rationalen Arbeitslosen?," *Jahrbuch für Sozialwissenschaft*, 39, 329–50.

GAIFMAN, H. (1986): "A Theory of Higher Order Probabilities," *Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning about Knowledge*, pp. 275–92.

GRANDMONT, J. M. (1977): "Temporary General Equilibrium Theory," *Econometrica*, 45(3), 535–72.

GRAVELLE, H., AND R. REES (1992): *Microeconomics*. Addison Wesley Longman Ltd., London, 2 edn.

HAAS, A., AND C. J. JAEGER (2005): “Agents, Bayes, and Climatic Risks- a modular modelling approach,” *Advances in Geosciences*, 4, 3–7.

HAHN, F. (1978): “On Non-Walrasian Equilibria,” *The Review of Economic Studies*, 1, 1–17.

HERINGS, J.-J. (1996): *Static and Dynamic Aspects of General Disequilibrium Theory*, Theory and Decision Library. Kluwer Academic Publishers.

IAB (2004): “Tarifbindung 2003,” Website, Institut für Arbeitsmarkt- und Berufsforschung.

KEYNES, J. M. (1936): *The General Theory of Employment, Interest and Money*, Great Mind Series. Prometheus Books.

KIRMAN, A. (2006): “Foreword,” in *Post Walrasian Macroeconomics: Beyond the Dynamic Stochastic General Equilibrium Model*, ed. by D. Colander, pp. xiii–xxi. Cambridge University Press, Cambridge.

KLAUS, B., H. PETERS, AND T. STORCKEN (1997): “Reallocation of an Infinitely Divisible Good,” *Economic Theory*, 10(2), 305–33.

LEE, P. M. (2004): *Bayesian Statistics: An Introduction*. Oxford University Press, Oxford, 3rd edn.

LEIJONHUFVUD, A. (1968): *On Keynesian Economics and the Economics of Keynes*. Oxford University Press, New York.

——— (1974): “Keynes’ Employment Function. Comment by Axel Leijonhufvud,” *History of Political Economy*, 6(2), 164–70.

LEWIS, D., AND D. TERRELL (2001): “Experience, Tenure, and the Perceptions of Employers,” *Southern Economic Journal*, 67(3), 578–97.

MALINVAUD, E. (1977): *Theory of Unemployment Reconsidered*. John Wiley and Sons.

MATHWORLD (1999): “Weibull Distribution,” `http://mathworld.wolfram.com/WeibullDistribution.html`.

MERZ, M. (2002): "Search Theory Rediscovered: Recent Developments in the Macroeconomics of the Labor Market," in *Search Theory and Unemployment*, ed. by S. Woodbury, and C. Davidson, pp. 17–44. Kluwer Academic Publishers, Boston.

MOOKHERJEE, D. (1987): "Involuntary Unemployment and Worker Self-Selection," *Journal of Economic Theory*, 45, 171–88.

MORISHIMA, M. (1981): *Walras' Economics. A Pure Theory of Capital and Money*. Cambridge University Press, Cambridge, 1st edn.

MORTENSEN, D. T., AND C. A. PISSARIDES (1994): "Job Creation and Job Destruction in the Theory of Unemployment," *The Review of Economic Studies*, 61(3), 397–415.

MOULIN, H. (1999): "Rationing a Commodity along Fixed Paths," *Journal of Economic Theory*, 84, 41–72.

——— (2000): "Priority rules and other asymmetric rationing methods," *Econometrica*, 68(3), 643–84.

MUELLBAUER, J., AND R. PORTES (1978): "Macroeconomic Models with Quantity Rationing," *The Economic Journal*, 88, 788–821.

MURPHY, K. R., AND A. P. TAM (2004): "The Decisions Job Applicants Must Make: Insights From a Bayesian Perspective," *International Journal of Selection and Assessment*, 12(1), 66–74.

PATINKIN, D. (1956): *Money, Interest and Prices: An Integration of Monetary and Value Theory*.

——— (1987): "Walras Law," in *The New Palgrave, A Dictionary of Economics*, ed. by J. Eatwell, M. Milgate, and P. Newman, vol. 4, p. 1025. Macmillan, London.

PISSARIDES, C. A. (1979): "Job Matching with State Employment Agencies and Random Search," *The Economic Journal*, 89(356), 818–33.

POLTEROVICH, V. (1993): "Rationing, Queues and Black Markets," *Econometrica*, 61(1), 1–28.

REINHARDT, F., AND H. SOEDER (1998): *dtv-Atlas Mathematik Band 2. Analysis und angewandte Mathematik*, vol. 2 of *dtv-Atlas*. dtv, Bielefeld, 2nd edn.

ROTHSCHILD, K. W. (1981): *Einführung in die Ungleichgewichtstheorie*. Springer, Heidelberg.

SHUBIK, M. (1999): *The Theory of Money and Financial Institutions 2*, vol. 2. MIT Press, Chicago.

SOZIALPOLITIK-AKTUELL (2005): "Erwerbstätige nach höchstem beruflichen Ausbildungs- bzw. Fachhochschul- / Hochschulabschluss 2004," `http://www.sozialpolitik-aktuell.de/datensammlung/4/ab/abbIV12.pdf`.

SPRUMONT, Y. (1991): "The Division Problem with Single-Peaked Preferences. A Characterization of the Uniform Allocation Rule," *Econometrica*, 59(2), 509–19.

STALEY, C. E. (1989): *A History of Economic Thought*. Basil Blackwell Inc., Cambridge.

STARR, R. M. (2001): *General Equilibrium Theory*. Cambridge University Press, Cambridge.

TASNÁDI, A. (2002): "On Probabilistic Rationing Methods," *Mathematical Social Sciences*, 44, 211–21.

TESFATSION, L. (2006): "Agent-based Computationa Modeling and Macroeconomics," in *Post Walrasian Macroeconomics: Beyond the Dynamic Stochastic General Equilibrium Model*, ed. by D. Colander, pp. 175–202. Cambridge University Press, Cambridge.

THOMSON, W. (1994): "Consistent Solutions to the Problem of Fair Division When Preferences Are Single-Peaked," *Journal of Economic Theory*, 63, 219–45.

VARIAN, H. R. (1984): *Microeconomic Analysis*. W.W. Norton and Company, New York, 2nd edn.

VENN, J. (1888): "The Subjective Side of Probability," in *Studies in Subjective Probability*, ed. by H. Kyburg Jr., and H. Smokler, p. 203. John Wiley and Sons, New York.

WALD, A. (1936): "Über einige Gleichungssysteme der mathematischen Ökonomie," *Zeitschrift für Nationalökonomie*, pp. 637–70.

WALRAS, L. (1877): *Elements of Pure Economics, or the Theory of Social Wealth*.

WEISS, A. (1976): "A Theory of Limited Labor Markets," Dissertation, University of Stanford.

——— (1980): "Job Queues and Layoffs in Labor Markets with Flexible Wages," *Journal of Political Economy*, 88(3), 526–38.

WIKIPEDIA (2006a): "Dynamisches System," http://de.wikipedia.org/wiki/Dynamisches_System.

——— (2006b): "Weibull Distribution," http://en.wikipedia.org/wiki/Weibull_distribution.

WORDSMYTH (2002): "Rationing," http://www.wordsmyth.net/live/home.php?content=wotw/110199/wotw_college#word4.

Erklärung

Ich versichere, dass ich die Dissertation selbständig und ohne fremde Hilfe verfasst habe, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die aus den benutzten Werken wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.