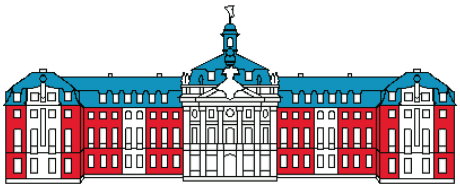

WESTFÄLISCHE
WILHELMS-UNIVERSITÄT
MÜNSTER

Diplomarbeit

Topic:	Numerical Methods for Equilibrium Models in General Markets
submitted by:	Anna Weisweiler [1]
submitted at:	May 8, 2008
Supervising Tutor:	Professor Dr. Martin Burger

[1] email: annaweisweiler@gmx.de

Contents

Chapter 1

Introduction

The intention of this diploma thesis is to solve the iterative problem of two algebraic equations and a few linear partial differential equations (PDE's). Solving this problem means economically generating the market equilibrium. In this case every market participant acts optimally and the market is cleared.

This problem appears in a paper by Vito D. Gala [1] and in a paper by Nicolae Gârleanu and Stavros Panageas [2]. Hence, one can consider this paper as a type of extention to the papers [1] and [2].
Both papers conctruct a general equilibrium. Paper [1] considers all firms in one block and divides agents into two groups of different risk aversions. Furthermore, paper [2] considers every individual firm and one representative investor.

The second chapter contains the theory of the main problem, that consists of a system of differential equations and two algebraic equations in two variables. Moreover, the proof of the existence of a solution, the discretization of the problem and the corresponding program that is constructed with MATLAB are also presented in chapter 2.
The third chapter is about the application of paper [1]. There the problem of paper [1] and its numerical solution in MATLAB is described.
The fourth chapter includes the application of paper [2] and the numerical solution of the problem given by paper [2].
The results are given in chapter five.
Moreover, a list of parameters used in this paper and all implementations constructed with MATLAB are provided in the appendix.

[1] cf. [2]
[2] cf. [1]

Chapter 2

Theory

In this chapter I present the main problem, its discretization, and the corresponding implementation constructed with the programming language MATLAB. The proof of the existence of a solution of the stationary equation, whose idea is used in the iteration of the implementation, is given in this chapter, too.

2.1 The Main Problem

Usually only one kind of agent and a single property is considered, which leads to only one differential equation. We consider heterogeneous agents (i.e. several groups). In order to describe this problem, we have to examine several differential equations. Hence, we consider a system of two or three differential equations and a system of two algebraic equations, which are coupled in a nonlinear way.

The typical structure of such problems is the following system of differential equations for Φ, Ψ and Θ:

$$\begin{aligned} -a_0(x)\Phi''(x) + a_1(F(x))\Phi'(x) + a_2(G(x))\Phi(x) &= a_3(x), \quad x \in (0,L) && (2.1)\\ -b_0(x)\Psi''(x) + b_1(F(x))\Psi'(x) + b_2(F(x),G(x))\Psi(x) &= b_3(x), \quad x \in (0,L) && (2.2)\\ -c_0(x)\Theta''(x) + c_1(F(x))\Theta'(x) + c_2(F(x),G(x))\Theta(x) &= c_3(x), \quad x \in (0,L) && (2.3) \end{aligned}$$

where the functions F and G satisfy the following equations:

$$\begin{aligned} d_1(x)F(x) + d_2(x)G(x) &= d_3(\Phi(x),\Psi(x),\Theta(x)), \quad x \in (0,L) && (2.4)\\ e_1(x)F(x) + e_2(x)G(x) &= e_3(\Phi(x),\Psi(x),\Theta(x)), \quad x \in (0,L) && (2.5) \end{aligned}$$

In typical economic applications the diffusion coefficients a_0, b_0, and c_0 disappear at the boundary, i.e. the boundary conditions are included implicitly in equations (2.1), (2.2), and (2.3). Hence, equations (2.1) - (2.3) are called degenerate elliptic equations and we cannot apply standard results. Thus, the analysis is difficult and we have to pay attention to the appropriate numerical solution of the problem.

We discretize the differential equations (2.1), (2.2), and (2.3) by using upwind techniques[1]. That means using backward differencing for $a_1 > 0$ ($b_1 > 0$, $c_1 > 0$) and forward differencing for the opposite case (coefficients < 0).

Moreover, the damping is important to achieve convergence of the system. Instead of solving the equation

$$-a(x)u''(x) + b(x)u'(x) + c(x)u(x) \quad = \quad d(x)$$

we solve for the following equation with the damping parameter $\tau > 0$ (similarly for $\tau < 0$)

$$-a(x)u''_{new}(x) + b(x)u'_{new}(x) + c(x)u_{new}(x) + \tau u_{new}(x) \quad = \quad d(x) + \tau u_{old}(x)$$

in order to perform one iteration step.
The last equation can be derived from the parabolic equation

$$\frac{\partial u}{\partial t} - a(x)\frac{\partial^2 u}{\partial x^2}(x) + b(x)\frac{\partial u}{\partial x}(x) + c(x)u(x) \quad = \quad d(x),$$

where $\frac{\partial u}{\partial t}$ can be approximated by $\frac{u-u_{old}}{\overline{\tau}}$ with $\overline{\tau} = \frac{1}{\tau}$ and u_{old} from the previous iteration step. (See also the next section.)

[1] cf. [3, p. 13]

2.2 Existence of a Solution

In the following section we verify the existence of a solution by a fixed point approach, which is later used for the numerical solution of the problem, too.

The fixed point form is given by

$$(u_1, u_2, u_3) = A(v_1, v_2, v_3) \tag{2.6}$$

using the fixed point mapping

$$\begin{aligned} A : C([0, L])^3 &\to C([0, L])^3 \\ (v_1, v_2, v_3) &\mapsto (u_1, u_2, u_3) \end{aligned}$$

where (u_1, u_2, u_3) are the solutions of the following equations:

$$\begin{aligned} -a_0(x)u_1''(x) + a_1(F(x))u_1'(x) + a_2(G(x))u_1(x) + \tau u_1(x) &= a_3(x) + \tau v_1(x) && (2.7) \\ -b_0(x)u_2''(x) + b_1(F(x))u_2'(x) + b_2(F(x), G(x))u_2(x)\tau u_2(x) &= b_3(x) + \tau v_2(x) && (2.8) \\ -c_0(x)u_3''(x) + c_1(F(x))u_3'(x) + c_2(F(x), G(x))u_3(x)\tau u_3(x) &= c_3(x) + \tau v_3(x) && (2.9) \end{aligned}$$

And the functions F and G are computed by:

$$\begin{aligned} d_1(x)F(x) + d_2(x)G(x) &= d_3(v_1(x), v_2(x), v_3(x)) && (2.10) \\ e_1(x)F(x) + e_2(x)G(x) &= e_3(v_1(x), v_2(x), v_3(x)) && (2.11) \end{aligned}$$

For the following proofs we define a reasonable assumption:

Assumption 1:

All coefficients have to be continuous.

$$a_i, b_i, c_i, d_i, e_i \in C([0, L]), \tag{2.12}$$

and the coefficients a_0, b_0, c_0 satisfy:

$$\begin{aligned} a_0(x) &\geq 0 \quad x \in [0, L] && (2.13)\\ b_0(x) &\geq 0 \quad x \in [0, L] && (2.14)\\ c_0(x) &\geq 0 \quad x \in [0, L] && (2.15) \end{aligned}$$

Furthermore, the coefficients a_0, b_0, c_0 and a_1, b_1, c_1 satisfy the following boundary conditions:

$$\begin{aligned} a_0(0) &= a_0(L) = 0 && (2.16)\\ b_0(0) &= b_0(L) = 0 && (2.17)\\ c_0(0) &= c_0(L) = 0 && (2.18)\\ a_1(F(0)) &< 0 && (2.19)\\ a_1(F(L)) &> 0 && (2.20)\\ b_1(F(0)) &< 0 && (2.21)\\ b_1(F(L)) &> 0 && (2.22)\\ c_1(F(0)) &< 0 && (2.23)\\ c_1(F(L)) &> 0 && (2.24) \end{aligned}$$

Moreover, the matrix B has to be regular for all $x \in (0, L)$

$$B = \begin{pmatrix} d_1(x) & d_2(x) \\ e_1(x) & e_2(x) \end{pmatrix}$$

and the condition number of B has to be uniformly bounded in x.

We want to prove the folowing theorem:

Theorem 1:

Let Assumption 1 hold. Then there exists a solution of equations (2.1) - (2.5).

Proof of Theorem 1:

The proof is done by **Schauder's Fixed Point Theorem**[2] using the fixed point mapping A that is described above.
The fixed point operator A is compact and maps a ball into itself, which follows from Proposition 1, Proposition 2, and Proposition 3 in the next section. Thus, we only have to show that the fixed point mapping is continuous.

We know from Assumption 1 that all coefficients are continuous in $[0, L]$. Hence, the following mappings are continuous as well:

$$\begin{aligned}
F(x) &\mapsto a_1(F(x)) && (2.25)\\
F(x) &\mapsto b_1(F(x)) && (2.26)\\
F(x) &\mapsto c_1(F(x)) && (2.27)\\
G(x) &\mapsto a_2(G(x)) && (2.28)\\
F(x) &\mapsto b_2(F(x), G(x)) && (2.29)\\
G(x) &\mapsto b_2(G(x), G(x)) && (2.30)\\
F(x) &\mapsto c_2(F(x), G(x)) && (2.31)\\
G(x) &\mapsto c_2(G(x), G(x)) && (2.32)
\end{aligned}$$

Thus, the mapping described by equations (2.7) - (2.9) is continuous:

$$(F, G) \mapsto (u_1, u_2, u_3) \tag{2.33}$$

Moreover, we know from the assumptions on matrix B that the inverse B^{-1} exists and is continuous. Hence, the mapping described by equations (2.10) - (2.11) is continuous as well:

$$(v_1, v_2, v_3) \mapsto (F, G) \tag{2.34}$$

Thus, we have shown that the fixed point mapping A is continuous. □

Hence, we show the following main proposition, related to the stationary equation in $\Omega = (0, L)$:

$$-a_i(x)u_i''(x) + b_i(x)u_i'(x) + c_i(x)u_i(x) + \tau u_i(x) = d_i(x) + \tau v_i(x) \tag{2.35}$$

[2] cf. [4, p. 502, Theorem 3] and Theorem 2 next section

Proposition 1:

Suppose u_i satisfies equation (2.35) in $\Omega = (0, L)$, $a_i, b_i, c_i, d_i \in C([0, L])$, $a_i(x) \geq 0$, $c_i(x) \geq 0$, $\tau > 0$, and the following boundary values:

$$\begin{aligned} a_i(0) &= a_i(L) = 0 \\ b_i(0) &< 0 \\ b_i(L) &> 0 \end{aligned}$$

Then there exists a unique solution $u_i \in C^1([0, L])$ of the stationary equation (2.35).

2.2.1 Proof of Proposition 1

For the proof we need **Schauder's Fixed Point Theorem**[3] that is given by:

Theorem 2:

> Suppose X is a real Banach space, M $\subset$ X is compact and convex and
> $A : M \to M$ is continuous.
> Then A has a fixed point in M.

Proof of Proposition 1:

We define $M = \{u \in X \mid \|u_i\|_\infty \leq K_i,\ \|u_i'\|_\infty \leq \widetilde{K}_i\}$ with positive constants K_i and $\widetilde{K}_i$.
M is compact and convex in L^∞, because the first derivative of u is also bounded in L^∞.
Note that the space of all continuous functions defined on a closed interval is a Banach space with the supremum norm.

Hence, we will show the next two propositions in the following subsections:

Proposition 2:

> Suppose u_i satisfies equation (2.35). Then there exists $K_i > 0$ such that for $\|v_i\|_\infty \leq K_i$ the following estimate holds:
>
> $$\|u_i\|_\infty \leq K_i$$

Proposition 3:

> Suppose u_i satisfies equation (2.35). Then there exists $\widetilde{K}_i > 0$ such that for $\|v_i\|_\infty \leq \widetilde{K}_i$ the following estimate holds:
>
> $$\|u_i'\|_\infty \leq \widetilde{K}_i$$

After showing these requirements for Schauder's Fixed Point Theorem, we can define the mapping A in the following way:

$$A : M \to M \text{ with } A(v) = u \text{ and } u \text{ is a solution of (2.35)}$$

With Schauder's Fixed Point Theorem it follows that A has a fixed point in M.

From this follows the existence of a solution $u_i(x)$ of (2.35) with $\tau = 0$.

In order to show uniqueness we apply the Maximum Principle to the difference of two solutions of equation (2.35). □

[3] cf. [4, p. 502, Theorem 3]

2.2.2 Proof of Proposition 2

First of all we have to show $\|u_i\|_\infty \leq K_i$. For this sake we need the **Maximum Principle**[4].

Theorem 3:

If $u(x)$ satisfies the differential inequality

$$Lu \;=\; u'' + g(x)u' + h(x)u \;\geq\; 0$$

in an interval (a, b) with $h(x) \leq 0$, if g and h are bounded on every closed subinterval, and if u assumes a nonnegative maximum value K at an interior point, then $u(x) \equiv K$.

Proof of Proposition 2:

We define

$$\widetilde{L}u_i(x) \;=\; -a_i(x)u_i''(x) + b_i(x)u_i'(x) + (c_i(x) + \tau)u_i(x) \tag{2.36}$$

$$f_i(x) \;=\; d_i(x) + \tau v_i(x) \tag{2.37}$$

with $a_i(x) \geq 0$ and $a_i = 0$ at the boundary values, $c_i(x) \geq 0$, $\tau \geq 0$, $b_i < 0$ at the left boundary point and $b_i > 0$ at the right boundary point, which is a suitable assumption satisfied in practice.

Fist we concider the case $\widetilde{L}u_i(x) \leq 0$:

If we apply the Maximum Principle, three cases can occur:

- In the first case u_i is a positive constant. We can apply this to equation (2.36). From the fact that the first and the second derivative of u_i is zero and from $(c_i(x) + \tau) \geq 0$ follows that $\widetilde{L}u_i(x) \geq 0$, which is inconsistent to $\widetilde{L}u_i(x) \leq 0$.

- The second case denotes $u_i(x) \leq 0$.

[4] cf. [5, p. 64, Theorem 6]

- In the last case u_i has no local maximum at an interior point of the interval. This can also be disproved:
  Assume that $u_i(x) > 0$ and consider equation (2.35) at the boundary. Because of $a_i = 0$ at the boundary, equation $\widetilde{L}u_i(x) \leq 0$ reduces to the following inequality:

$$b_i(x)u_i'(x) + (c_i(x) + \tau)u_i(x) \quad \leq \quad 0$$

  From $(c_i(x) + \tau) \geq 0$ and $u_i(x) > 0$ we know that $-(c_i(x) + \tau)u_i(x) \leq 0$ holds. So we obtain the next inequality:

$$b_i(x)u_i'(x) \quad \leq \quad 0$$

  At the left boundary value $b_i < 0$ holds. Thus, we have $u_i'(x) \geq 0$ at the left part of the interval. Hence, $u_i'(x) \leq 0$ holds analogically for $b_i > 0$ at the right part of the interval. From this it follows for an interior point x:

$$u_i(x) \geq \text{the boundary values of } u_i$$

  We have assumed that $u_i(x) > 0$, so u_i has a positive maximum at an interior point. This is inconsistent to the case we consider. Thus, this case is also disproved.

Hence, $\widetilde{L}u_i(x) \leq 0$ and the Maximum Principle leads to $u_i(x) \leq 0$.

Now we consider the case $\widetilde{L}u_i(x) \geq 0$:

If we apply the Maximum Principle, there can also occur three cases:

- In the first case u_i is a negative constant. We can apply this to equation (2.36) and analogically obtain $\widetilde{L}u_i(x) \leq 0$, which is inconsistent to $\widetilde{L}u_i(x) \geq 0$.

- The second case denotes $u_i(x) \geq 0$.

- In the last case u_i has no local minimum at an interior point of the interval. This can also be disproved as in the case $\widetilde{L}u_i(x) \leq 0$:
  Assume that $u_i(x) < 0$ and consider equation (2.35) at the boundary. Because of $a_i = 0$ at the boundary, $(c_i(x) + \tau) \geq 0$ and $u_i(x) < 0$ equation $\widetilde{L}u_i(x) \geq 0$ reduces to the following inequality:

  $$b_i(x)u_i'(x) \quad \geq \quad 0$$

  Due to the assumptions of b at the boundary values, we obtain $u_i'(x) \leq 0$ at the left part of the interval and $u_i'(x) \geq 0$ at the right part of the interval. From this it follows for an interior point x:

  $$u_i(x) \leq \text{the boundary values of } u_i$$

  We have assumed that $u_i(x) < 0$, so u_i has a negative minimum at an interior point. That is inconsistent to this case, so it is disproved.

Thus, $\widetilde{L}u_i(x) \geq 0$ leads to $u_i(x) \geq 0$.

Hence, we can now derive the estimate $\|u_i\|_\infty \leq K_i$.

In the case $\widetilde{L}u_i(x) \leq 0$ we define

$$\widetilde{u}_i(x) \quad = \quad u_i(x) - K_i \tag{2.38}$$

where K_i is a positive constant. Moreover, we have shown that $u_i(x) \leq 0$ follows from $\widetilde{L}u_i(x) \leq 0$. If we apply this to $\widetilde{u}_i(x)$, we obtain:

$$u_i(x) \quad \leq \quad K_i \tag{2.39}$$

We can show that K_i depends on $d_i(x)$ and $c_i(x)$ by calculating the following equation with (2.37):

$$\widetilde{L}\widetilde{u}_i(x) \quad = \quad f_i(x) - \widetilde{L}K_i \tag{2.40}$$

Rearranging leads to

$$\widetilde{L}\widetilde{u}_i(x) \quad = \quad c_i(x)\left(\frac{d_i(x)}{c_i(x)} - K_i\right) + \tau(v_i(x) - K_i) \tag{2.41}$$

Because of $c_i(x) \geq 0$, $\tau \geq 0$ and $\widetilde{L}\widetilde{u}_i(x) \leq 0$ the following inequalities have to hold:

$$\begin{aligned} \max\left(\frac{d_i(x)}{c_i(x)}\right) &\leq K_i & (2.42)\\ v_i(x) &\leq K_i & (2.43) \end{aligned}$$

Moreover, in the case $\widetilde{L}u_i(x) \geq 0$ we define

$$\widetilde{u}_i(x) = u_i(x) + K_i \tag{2.44}$$

where K_i is a positive constant. Furthermore, we have shown that $u_i(x) \geq 0$ follows from $\widetilde{L}u_i(x) \geq 0$. If we apply this to $\widetilde{u}_i(x)$, we obtain:

$$-K_i \leq u_i(x) \tag{2.45}$$

We can show in an analogous way to the case $\widetilde{L}u_i(x) \leq 0$ that K_i depends on $d_i(x)$ and $c_i(x)$. Hence, we obtain the next inequalities:

$$\begin{aligned} -K_i &\leq \min\left(\frac{di(x)}{ci(x)}\right) & (2.46)\\ -K_i &\leq v_i(x) & (2.47) \end{aligned}$$

Combining (2.39) and (2.45) leads to the desired estimate $\|u_i\|_\infty \leq K_i$. □

2.2.3 Proof of Proposition 3

In the following we show the bound on u_i' in the supremum norm.

For this sake we divide the considered interval into three parts.

Proof of Proposition 3:

First of all we want to verify the estimate for the left part of the interval:

We define $w_i(x) = u_i'(x)$. Hence, equation (2.35) leads to the following equation for w_i:

$$a_i(x)w_i'(x) \quad = \quad b_i(x)w_i(x) + (c_i(x) + \tau)u_i(x) - d_i(x) - \tau v_i(x) \tag{2.48}$$

In the case $w_i(x) > 0$, we define $w_i(x) = \widetilde{w}_i(x) + \gamma$ with $\gamma > 0$ sufficiently large such that

$$b_i(x)\gamma + (c_i(x) + \tau)F_i(x) - d_i(x) - \tau v_i(x) \quad < \quad 0 \tag{2.49}$$

holds with the antiderivative $F_i(x)$ of $\widetilde{w}_i(x)$. It is possible to choose γ like that, because $b_i(0) < 0$.

For $\widetilde{w}_i(x)$ equation (2.48) reads as follows:

$$a_i(x)\widetilde{w}_i'(x) \quad = \quad b_i(x)\widetilde{w}_i(x) + b_i(x)\gamma + (c_i(x) + \tau)F_i(x) - d_i(x) - \tau v_i(x) \tag{2.50}$$

From (2.49) we obtain the next inequality:

$$a_i(x)\widetilde{w}_i'(x) \quad < \quad b_i(x)\widetilde{w}_i(x) \tag{2.51}$$

In the case $\widetilde{w}_i(x) > 0$ rearranging leads to

$$\frac{\widetilde{w}_i'(x)}{\widetilde{w}_i(x)} \quad < \quad \frac{b_i(x)}{a_i(x)} \tag{2.52}$$

We obtain the following inequality, because $b_i < 0$ at the left boundary value.

$$\frac{\widetilde{w}_i'(x)}{\widetilde{w}_i(x)} \quad < \quad 0 \tag{2.53}$$

Hence, it follows for the derivative of the logarithm that

$$[log(\widetilde{w}_i(x))]' \quad < \quad 0 \tag{2.54}$$

From the monotonicity of the logarithm we obtain

$$\widetilde{w}_i(x) \quad < \quad \widetilde{w}_i(0) \tag{2.55}$$

where x is an interior point of the interval.
This leads to the following:

$$w_i(x) \quad < \quad w_i(0) \tag{2.56}$$

In the case $\widetilde{w}_i(x) \leq 0$ we obtain:

$$w_i(x) \quad \leq \quad \gamma \tag{2.57}$$

Combining (2.56) and (2.57) leads to the following estimate:

$$0 \ \leq \ w_i(x) \ \leq \ \max\left(\gamma, w_i(0)\right) \tag{2.58}$$

In the case $w_i(x) < 0$, we define $w_i(x) = \widetilde{w}_i(x) - \gamma$ with $\gamma > 0$ sufficiently large such that

$$-b_i(x)\gamma + (c_i(x) + \tau)F_i(x) - d_i(x) - \tau v_i(x) \quad > \quad 0 \tag{2.59}$$

holds with the antiderivative $F_i(x)$ of $\widetilde{w}_i(x)$. It is possible to choose γ like that, because $b_i(0) < 0$.
For $\widetilde{w}_i(x)$ equation (2.48) reads as follows:

$$a_i(x)\widetilde{w}_i'(x) \quad = \quad b_i(x)\widetilde{w}_i(x) - b_i(x)\gamma + (c_i(x) + \tau)F_i(x) - d_i(x) - \tau v_i(x) \tag{2.60}$$

From (2.59) we obtain the next inequality:

$$a_i(x)\widetilde{w}_i'(x) \quad > \quad b_i(x)\widetilde{w}_i(x) \tag{2.61}$$

In the case $\widetilde{w}_i(x) < 0$ rearranging leads to

$$\frac{\widetilde{w}_i'(x)}{\widetilde{w}_i(x)} \quad < \quad \frac{b_i(x)}{a_i(x)} \tag{2.62}$$

We obtain the following inequality, because $b_i < 0$ at the left boundary value.

$$\frac{\widetilde{w}_i'(x)}{\widetilde{w}_i(x)} \quad < \quad 0 \tag{2.63}$$

Moreover, it follows for the derivative of the logarithm that

$$[log(-\widetilde{w}_i(x))]' \quad < \quad 0 \tag{2.64}$$

From the monotonicity of the logarithm we obtain

$$-\widetilde{w}_i(x) \quad < \quad -\widetilde{w}_i(0) \tag{2.65}$$

where x is an interior point of the interval. This is equivalent to the following inequality:

$$\widetilde{w}_i(x) \quad > \quad \widetilde{w}_i(0) \tag{2.66}$$

In the case $\widetilde{w}_i(x) \geq 0$ we obtain:

$$w_i(x) \quad \geq \quad -\gamma \tag{2.67}$$

Combining (2.66) and (2.67) leads to the following estimate:

$$0 \geq w_i(x) \geq \max(-\gamma, w_i(0)) \tag{2.68}$$

Combining (2.58) and (2.68) we obtain:

$$|w_i(x)| \quad \leq \quad \max(\gamma, |w_i(0)|) \tag{2.69}$$

Furthermore, we have to verify the estimate for the right part of the interval:

This part is analogous to the one above. There are just a few differences:

- $b_i > 0$ at the right boundary value
- In the case $w_i(x) > 0$ we choose $\gamma > 0$ sufficiently large such that

$$b_i(x)\gamma + (c_i(x) + \tau)F_i(x) - d_i(x) - \tau v_i(x) \quad > \quad 0$$

- In the case $w_i(x) < 0$ we choose $\gamma > 0$ sufficiently large such that

$$-b_i(x)\gamma + (c_i(x) + \tau)F_i(x) - d_i(x) - \tau v_i(x) \quad < \quad 0$$

After this we obtain the following estimate

$$|w_i(x)| \quad \leq \quad \max(\gamma, |w_i(L)|) \tag{2.70}$$

where L is the right boundary value of the interval.

Finally, we have to show the estimate for the central part of the interval:

For this proof we need the **Gronwall Lemma**[5]:

Lemma 1:

> Let $u(t)$ be a continuous function defined on a closed interval $J = [0, T]$. If $u(t)$ satisfies in J the inequality
>
> $$u(t) \quad \leq \quad \alpha + \beta \int_0^t u(s)\, ds$$
>
> with $\beta > 0$ and an arbitrary constant α. Then the following inequality holds in J:
>
> $$u(t) \quad \leq \quad \alpha e^{\beta t}$$

[5] cf. [6, p. 361, Lemma 12.27]

Consider equation (2.35), define $w_i(x) = u_i'(x)$ and set $d_i(x) + \tau v_i(x) - (c_i(x) + \tau)u_i(x) = C$ with a constant C. Hence,

$$a_i(x)w_i'(x) \quad = \quad b_i(x)w_i(x) - C \tag{2.71}$$

where for interior points of the interval it holds $a_i(x) \geq \delta > 0$ with a positive constant δ. We conclude

$$w_i'(x) \quad \leq \quad \frac{b_i(x)}{a_i(x)}w_i - \frac{C}{a_i(x)} \tag{2.72}$$

This inequality also holds as equality, but we only need the inequality. We can write this inequality in the following way:

$$w_i(x) \quad \leq \quad \alpha_{1i} + \beta_{1i}\int_0^x w_i(s)\,ds \tag{2.73}$$

With Gronwall's Lemma we obtain:

$$u_i'(x) \;=\; w_i(x) \;\leq\; \alpha_{1i}\, e^{\beta_{1i}x} \tag{2.74}$$

Now we define $w_i(x) = -u_i'(x)$. Hence,

$$a_i(x)w_i'(x) \quad = \quad b_i(x)w_i(x) + C \tag{2.75}$$

where for interior points of the interval it holds $a_i(x) \geq \delta > 0$ with a positive constant δ. It follows that

$$w_i'(x) \quad \leq \quad \frac{b_i(x)}{a_i(x)}w_i + \frac{C}{a_i(x)} \tag{2.76}$$

This inequality also holds as equality, but we only need the inequality. We can write this inequality in the following way:

$$w_i(x) \leq \quad \alpha_{2i} + \beta_{2i}\textstyle\int_0^x w_i(s)\,ds \tag{2.77}$$

With Gronwall's Lemma we obtain:

$$w_i(x) \quad \leq \quad \alpha_{2i}\, e^{\beta_{2i}x} \tag{2.78}$$

Moreover, this inequality is equivalent to the following one:

$$-u_i'(x) \quad \leq \quad \alpha_{2i}\, e^{\beta_{2i} x} \tag{2.79}$$

i.e.

$$u_i'(x) \quad \geq \quad -\alpha_{2i}\, e^{\beta_{2i} x} \tag{2.80}$$

With (2.74) and (2.78) we obtain the following estimate:

$$|w_i(x)| \quad \leq \quad C_i \tag{2.81}$$

$$C_i \quad = \quad \max\left(\alpha_1\, e^{\beta_1 x}, \alpha_2\, e^{\beta_2 x}\right) \tag{2.82}$$

Combining (2.69), (2.70), and (2.81) we obtain the desired estimate

$$\|u_i'\|_\infty \leq \widetilde{K}_i \tag{2.83}$$

where $\widetilde{K}_i = \max\left(|w_i(0)|, |w_i(L)|, C_i, \gamma\right)$. □

2.3 Motivation and Discretization

We can solve the problem of equations (2.1) - (2.5) by a numerical algorithm. That means we first have to compute F and G from equations (2.4), (2.5) and from initial values of Φ, Ψ, and Θ. Afterwards we can evaluate new values of Φ, Ψ, and Θ from equations (2.1) - (2.3). This iteration continues until convergence is achieved.
For calculating Φ, Ψ, and Θ we need the finite difference method. This discretisation is now explained by equation (2.1).

We want to apply the discretization e.g. to equation (2.1). Hence, at each point j with $j = 1, 2, ..., n$ we obtain the following equation:

$$a_0(x)[\partial^2_{xx}\Phi]_j + a_1(F(x))[\partial_x\Phi]_j + a_2(G(x))\Phi_j \quad = \quad a_3(x) \tag{2.84}$$

Thus, we need the discretization of $\Phi''(x) = \partial^2_{xx}\Phi(x)$ and $\Phi'(x) = \partial_x\Phi(x)$. For the discretization of the first derivatives we use upwind techniques[6] as described in chapter 2.1.
Thus, we obtain a M-matrix[7] and the discrete Maximum Principle is fulfilled. This can also be proven by Schauder's Fixed Point Theorem.

The derivative of second order with respect to x is approximated by:

$$[\partial^2_{xx}\Phi]_j = \frac{\Phi_{j+1} - 2\Phi_j + \Phi_{j-1}}{h} \tag{2.85}$$

The derivative of first order with respect to x is discretized for $a_1(F(x)) < 0$ by forward differencing:

$$[\partial_x\Phi]_j = \frac{\Phi_{j+1} - \Phi_j}{h} \tag{2.86}$$

Moreover, the derivative of first order with respect to x for $a_1(F(x)) > 0$ is approximated by backward differencing:

$$[\partial_x\Phi]_j = \frac{\Phi_j - \Phi_{j-1}}{h} \tag{2.87}$$

[6] cf. [3, p. 13]
[7] cf. [3, Section 2.3.3, p. 25]

This leads to the following discretization of equation (2.1):

$$\begin{aligned} a_3(x) &= \left[\frac{a_0(x)}{h^2} + \frac{\min\left(a_1(F(x)),0\right)}{h}\right]\Phi_{j+1}(x) \\ &+ \left[a_2(G(x)) - \frac{2a_0(x)}{h^2} + \frac{\max\left(a_1(F(x)),0\right)}{h} - \frac{\min\left(a_1(F(x)),0\right)}{h}\right]\Phi_j(x) \\ &+ \left[\frac{a_0(x)}{h} - \frac{\max\left(a_1(F(x)),0\right)}{h}\right]\Phi_{j-1}(x) \end{aligned} \tag{2.88}$$

We can rewrite this equation in matrix form

$$A\Phi(x) = a_3(x) \tag{2.89}$$

where the first sub diagonal of matrix A is denoted by r, the main diagonal of matrix A is denoted by s and the first super diagonal of matrix A is denoted by t.

$$r = \frac{a_0(x)}{h} - \frac{\max\left(a_1(F(x)),0\right)}{h} \tag{2.90}$$

$$s = a_2(G(x)) - \frac{2a_0(x)}{h^2} + \frac{\max\left(a_1(F(x)),0\right)}{h} - \frac{\min\left(a_1(F(x)),0\right)}{h} \tag{2.91}$$

$$t = \frac{a_0(x)}{h^2} + \frac{\min\left(a_1(F(x)),0\right)}{h} \tag{2.92}$$

Hence, the described problem can be computed by the implementation that is given in the next section.

2.4 The Main Implementation

In the following we describe the implementation **Iteration** that solves two equations for F and G and a system of three linear differential equations for U,V, and W.

First we need inital values of U, V, and W to solve equations (2.7) - (2.9). With these values we are able to compute F and G from equations (2.10) and (2.11).

That means we compute the vector o which consists of F and G one below the other

$$o(x) = \begin{pmatrix} F(x) \\ G(x) \end{pmatrix} = B^{-1} \begin{pmatrix} d_3(x) \\ e_3(x) \end{pmatrix}$$

with matrix B of Assumption 1.

Thereafter, we have to calculate U, V, W and F, G in turn until convergence is achieved.

For the calculation of U, V, and W we have to define the damped coefficients as follows:

$$\begin{aligned} \widetilde{a}_2 &= a_2 + \tau && (2.93) \\ \widetilde{a}_3 &= a_3 + \tau U && (2.94) \\ \widetilde{b}_2 &= b_2 + \tau && (2.95) \\ \widetilde{b}_3 &= b_3 + \tau V && (2.96) \\ \widetilde{c}_2 &= c_2 + \tau && (2.97) \\ \widetilde{c}_3 &= c_3 + \tau W && (2.98) \end{aligned}$$

where τ denotes the damping factor.

Moreover the computation of U, V, and W result from the function **DGLfunction**, where U, V, and W are calculated by the following equations:

$$\begin{aligned} U &= A^{-1}\,\widetilde{a}_3 && (2.99) \\ V &= A^{-1}\,\widetilde{b}_3 && (2.100) \\ W &= A^{-1}\,\widetilde{c}_3 && (2.101) \end{aligned}$$

where matrix A is described by equations (2.90), (2.91) and (2.92) with the dumping coefficients $\widetilde{a}_2$, $\widetilde{b}_2$, and $\widetilde{c}_2$ instead of a_2, b_2, and c_2.

The complete implementation is included in the appendix.

Chapter 3

Application I

The paper [1] by Gârleanu and Panageas constructs a general equilibrium. In the model all firms are concentrated in one block and agents are divided into less risk-averse and more risk-averse agents. Therefore the authors investigate one representative firm and two types of agents denoted by type A and type B agents.
There exist many relations to other papers. For example, the authors denote their paper as an extension of the perpetual youth model of Blanchard[1]. In contrast to Blanchard the model of Gârleanu and Panageas is stochastic. Besides this model is similar to Campbell and Cochrane[2], but with different economic mechanisms and justification.
This paper includes four main differences to other papers: First of all, they integrate agents with finite lifespans. Furthermore, the agent's ability to work declines with age. Moreover, agents may have different preferences, and finally, consumption and dividends are not equal.

3.1 Derivation of the Problem

In the following we describe the problem of paper [1] by the required equations.

The consumption share X_t is given by the stochastic process

$$dX_t = \mu_X dt + \sigma_X dB_t \tag{3.1}$$

where B_t is a Brownian motion and μ_X, σ_X are the drift and diffusion coefficients of the process.

[1] cf. [7]
[2] cf. [8]

Moreover, the process of the stochastic discount factor ξ_t is defined as:

$$d\xi_t = -r_t\xi_t dt - \kappa_t\xi_t dB_t \tag{3.2}$$

where r_t is the interest rate at time t and κ_t denotes the Sharpe ratio at time t.

Furthermore, the stochastic output Y_t is given by the following equations:

$$Y_t = Z_t f(H_t) \tag{3.3}$$
$$\frac{dZ_t}{Z_t} = \mu_Z dt + \sigma_Z dB_t, \quad \text{with positive constants } \mu_Z \text{ and } \sigma_Z \tag{3.4}$$

where H_t denotes the agent's aggregate hours worked at time t, Z_t stands for the exogenous productivity process, and $f(H_t)$ is an increasing and concave function on H_t.

The functions $g_t = g(X_t)$, ω_t and the prevailing wage (equilibrium wage) $w_t = w(X_t)$ are defined as follows:

$$g(X_t) = \frac{Y_t}{Z_t} \tag{3.5}$$
$$w(X_t) = Z_t\omega(X_t) \tag{3.6}$$
$$\omega_t = \frac{w_t}{Z_t} \tag{3.7}$$

Moreover, the following equation is important to compute the diffusion coefficient σ_X, the Sharpe ratio κ_t, the drift coefficient μ_X and the interest rate r_t. This equation follows from equation (34)[3] in [1].

$$X_t Y_t = v\int_{-\infty}^{t} \pi e^{-\left(\pi+\frac{\rho_A}{\gamma_A}\right)(t-s)} \beta_s^A Z_s g_s \left(\frac{Z_t\omega_t}{Z_s\omega_s}\right)^{\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}} \left(\frac{\xi_t}{\xi_s}\right)^{-\frac{1}{\gamma_A}} ds \tag{3.8}$$

[3] cf. [1, p. 14, Equation (34)]

Hence, π is the constant hazard rate of death. Beyond it v refers the mass of type A agents, ρ_A stands for the subjective discount rate of type A agents, γ_A denotes the relative risk aversion of type A agents and ψ_A controls the relative importance of leisure and consumption of type A agents. Furthermore, $\beta_t^A = \beta^A(X_t)$ will be determined later.

For the following calculations we need **Ito's Lemma**[4]:

Lemma 2:

Suppose $u(t, s_1, ..., s_d)$ is a continuous function on $[t_0, T] \times R^d$ with continuous partial derivatives u_t, u_{s_i} and u_{s_i,s_j}, $i, j \leq d$.
Furthermore, d one-dimensional stochastic processes $S_i(t)$ in $[t_0, T]$ are given by:

$$dS_i(t) = f_i(t)dt + g_i(t)dW_t, \quad i = 1, 2, ..., d$$

Thus, the one-dimensional process $V_t = u(t, S_1(t), ..., S_d(t))$ is also described in $[t_0, T]$ by a stochastic differential:

$$dV_t = \left(u_t + \sum_{i=1}^{d} u_{s_i} f_i + \frac{1}{2} \sum_{i=1}^{d} \sum_{j=1}^{d} u_{s_i s_j} g_i g_j \right) dt + \left(\sum_{i=1}^{d} u_{s_i} g_i \right) dW_t$$

Moreover, the product rule for $u = s_1 s_2$ and $d = 2$ is given by:

$$\begin{aligned} d(S_1(t)S_2(t)) &= S_1(t)dS_2(t) + S_2(t)dS_1(t) + dS_1(t)dS_2(t) \\ &= S_1(t)dS_2(t) + S_2(t)dS_1(t) + g_1(t)g_2(t)dt \end{aligned}$$

Computing $\mathrm{d}w_t$ and $\mathrm{d}\big(w^a \xi^b\big)$ leads to the equation for $\mathrm{D}\big(w^a \xi^b\big)$. We compute $\mathrm{d}\big(w^a \xi^b\big)$ by Ito's Lemma and the product rule of Ito's Lemma:

$$\begin{aligned} d(w^a \xi^b) &= d(w^a)\xi^b + w^a d(\xi^b) + d(w^a)d(\xi^b) && (3.9) \\ &= ... \\ &= w^a \xi^b \left(\widetilde{\mu} dt + \widetilde{\sigma} dB\right) \quad \text{with} && (3.10) \\ \widetilde{\mu} &= a\left(\mu_Z + \frac{\sigma_Z^2}{2}(a-1) + \frac{\omega'}{\omega}(\mu_X + a\sigma_Z\sigma_X - \kappa b \sigma_X) - b\kappa\sigma_Z\right) \\ &\quad + \frac{\sigma_X^2}{2}\left(a(a-1)\left(\frac{\omega'}{\omega}\right)^2 + a\frac{\omega''}{\omega}\right) - b\left(r + \frac{\kappa^2}{2}(1-b)\right) && (3.11) \\ \widetilde{\sigma} &= a\sigma_Z + a\frac{\omega'}{\omega}\sigma_X - b\kappa && (3.12) \end{aligned}$$

[4] cf. [9, p. 104,105, (5.3.11) and (5.4.1)]

Hence, $\mathrm{D}\left(w^a\xi^b\right)$ is the quotient of the drift term of $\mathrm{d}\left(w^a\xi^b\right)$ and $w^a\xi^b$:

$$\begin{aligned} D\left(w^a\xi^b\right) &= a\left(\mu_Z + \frac{\sigma_Z^2}{2}(a-1) + \frac{\omega'}{\omega}(\mu_X + a\sigma_Z\sigma_X - \kappa b\sigma_X) - b\kappa\sigma_Z\right) \\ &\quad + \frac{\sigma_X^2}{2}\left(a(a-1)\left(\frac{\omega'}{\omega}\right)^2 + a\frac{\omega''}{\omega}\right) - b\left(r + \frac{\kappa^2}{2}(1-b)\right) \end{aligned} \tag{3.13}$$

Now we can apply the d-operator on both sides of (3.8). Equation (3.14) contains the diffusion term of the application of the d-operator to equation (3.8) divided by X_tY_t.

$$\frac{\sigma_X}{X_t} + \sigma_Z + \frac{g'}{g}\sigma_X = \frac{(1-\psi_A)(\gamma_A - 1)}{\gamma_A}\left(\frac{\omega'}{\omega}\sigma_X + \sigma_Z\right)\frac{\kappa_t}{\gamma_A} \tag{3.14}$$

Afterwards equation (3.15) is the quotient of the drift term of the application of the d-operator to equation (3.8) and Y_t.

$$\begin{aligned} &\mu_X + \left(\mu_Z + \frac{g'}{g}(\mu_X + \sigma_X\sigma_Z) + \frac{\sigma_X^2}{2}\frac{g''}{g}\right)X_t + \left(\frac{g'}{g}\sigma_X + \sigma_Z\right)\sigma_X \\ &\qquad = X_t\left[D\left(w_t^{\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}}\xi_t^{-\frac{1}{\gamma_A}}\right) - \left(\pi + \frac{\rho_A}{\gamma_A}\right)\right] + v\pi\beta_t^A \end{aligned} \tag{3.15}$$

In the following we derive the other equations that we need for the calculation of the diffusion coefficient σ_X, the Sharpe ratio κ_t, the drift coefficient μ_X and the interest rate r_t. This equations will follow from equation (38)[5] in [1], which is given by:

$$\begin{aligned} Z_tg(X_t) &= \int_{-\infty}^{t} \pi e^{-\left(\pi + \frac{\rho_A}{\gamma_A}\right)(t-s)} v\beta_s^A Z_s g_s \left(\frac{Z_t\omega_t}{Z_s\omega_s}\right)^{\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}} \left(\frac{\xi_t}{\xi_s}\right)^{-\frac{1}{\gamma_A}} ds \\ &\quad + \int_{-\infty}^{t} \pi e^{-\left(\pi + \frac{\rho_B}{\gamma_B}\right)(t-s)} (1-v)\beta_s^B Z_s g_s \left(\frac{Z_t\omega_t}{Z_s\omega_s}\right)^{\frac{(1-\psi_B)(\gamma_B-1)}{\gamma_B}} \left(\frac{\xi_t}{\xi_s}\right)^{-\frac{1}{\gamma_B}} ds \end{aligned} \tag{3.16}$$

[5] cf. [1, p. 15, Equation (38)]

Equation (3.17) implies the diffusion term of the application of the d-operator to equation (3.16) divided by Y_t

$$\begin{aligned}\sigma_Z + \frac{g'}{g}\sigma_X &= X_t\left[\frac{\kappa_t}{\gamma_A} + \frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}\left(\frac{\omega'}{\omega}\sigma_X + \sigma_Z\right)\right] \\ &\quad +(1-X_t)\left[\frac{\kappa_t}{\gamma_B} + \frac{(1-\psi_B)(\gamma_B-1)}{\gamma_B}\left(\frac{\omega'}{\omega}\sigma_X + \sigma_Z\right)\right] \qquad (3.17)\end{aligned}$$

and equation (3.18) implies the drift term of the application of the d-operator to equation (3.16) divided by Y_t.

$$\begin{aligned}\mu_Z + \frac{g'}{g}(\mu_X + \sigma_X\sigma_Z) + \frac{1}{2}\frac{g''}{g}\sigma_X^2 = v\pi\beta_t^A + X_t\left[D\left(w^{\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}}\xi^{-\frac{1}{\gamma_A}}\right) - \left(\pi + \frac{\rho_A}{\gamma_A}\right)\right] \\ +(1-v)\pi\beta_t^B + (1-X_t)\left[D\left(w^{\frac{(1-\psi_B)(\gamma_B-1)}{\gamma_B}}\xi^{-\frac{1}{\gamma_B}}\right) - \left(\pi + \frac{\rho_B}{\gamma_B}\right)\right] \qquad (3.18)\end{aligned}$$

From Lemma 1 in [1] it follows that ϕ^A solves the differential equation (3.19).

$$\begin{aligned}0 &= \frac{\sigma_X^2}{2}\left(\phi^A\right)'' + \left(\phi^A\right)'(\mu_X + \sigma_X(\sigma_Y - \kappa)) + \phi^A(\mu_Y - r - \sigma_Y\kappa - \pi - \chi) \\ &\quad +\psi_A\frac{\pi+\chi}{\pi}\frac{\omega}{g} \qquad (3.19)\end{aligned}$$

The function $\zeta^i(X_t)$ solves the differential equation (3.20), where the following notations are used: $\phi^B(X_t) = \frac{\psi_B}{\psi_A}\phi^A(X_t)$, $\beta^i(X_t) = \frac{\phi^i(X_t)}{\zeta^i(X_t)}$, $i \in \{A, B\}$

$$\begin{aligned}-1 &= \frac{\sigma_X^2}{2}(\zeta^i)'' + \left(\mu_X + \sigma_X\frac{(1-\psi_i)(\gamma_i-1)}{\gamma_i}\left(\sigma_Z + \frac{\omega'}{\omega}\sigma_X\right) - \sigma_X\frac{\gamma_i-1}{\gamma_i}\kappa\right)(\zeta^i)' \\ &\quad + \left[D\left(w_t^{\frac{(1-\psi_i)(\gamma_i-1)}{\gamma_i}}\zeta_t^{1-\frac{1}{\gamma_i}}\right) - \left(\pi + \frac{\rho_i}{\gamma_i}\right)\right]\zeta^i \qquad (3.20)\end{aligned}$$

Moreover, parameter μ_Y and σ_Y are given by:

$$\mu_Y = \mu_Z + \frac{g'}{g}(\mu_X + \sigma_X\sigma_Z) + \frac{\sigma_X^2}{2}\frac{g''}{g} \qquad (3.21)$$

$$\sigma_Y = \sigma_Z + \frac{g'}{g}\sigma_X \qquad (3.22)$$

We can identify equations (3.15) and (3.18) as a special case of equations (2.4) and (2.5) by setting $\mu_X = F(X)$ and $r = G(X)$. Similarly, we can identify equation (3.19) and (3.20) with the three differential equations (2.1), (2.2), and (2.3) by setting additionally $\phi^A = \Phi$, $\zeta^A = \Psi$ and $\zeta^B = \Theta$. Rearranging (3.15), (3.18), (3.19), and (3.20) and omitting the index t leads to:

$$
\begin{aligned}
d_1 &= 1 + X\frac{g'}{g} - \frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}X\frac{\omega'}{\omega} && (3.23)\\
d_2 &= -\frac{X}{\gamma_A} && (3.24)\\
d_3 &= X\left[\widehat{D}\left(\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}, -\frac{1}{\gamma_A}\right) - \left(\pi + \frac{\rho_A}{\gamma_A}\right) - \mu_Z - \frac{g'}{g}\sigma_Z\sigma_X - \frac{g''}{g}\frac{\sigma_X^2}{2}\right] \\
&\quad -\frac{g'}{g}\sigma_X^2 - \sigma_Z\sigma_X + v\pi\frac{\Phi}{\Psi} && (3.25)\\
e_1 &= \frac{g'}{g} - X\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}\frac{\omega'}{\omega} - (1-X)\frac{(1-\psi_B)(\gamma_B-1)}{\gamma_B}\frac{\omega'}{\omega} && (3.26)\\
e_2 &= -\frac{X}{\gamma_A} - \frac{1-X}{\gamma_B} && (3.27)\\
e_3 &= v\pi\frac{\Phi}{\Psi} + (1-v)\pi\frac{\psi_B}{\psi_A}\frac{\Phi}{\Theta} + X\left[\widehat{D}\left(\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}, -\frac{1}{\gamma_A}\right) - \left(\pi + \frac{\rho_A}{\gamma_A}\right)\right] && (3.28)\\
&\quad +(1-X)\left[\widehat{D}\left(\frac{(1-\psi_B)(\gamma_B-1)}{\gamma_B}, -\frac{1}{\gamma_B}\right) - \left(\pi + \frac{\rho_B}{\gamma_B}\right)\right] - \mu_Z - \frac{g'}{g}\sigma_X\sigma_Z - \frac{\sigma_X^2}{2}\frac{g''}{g} \\
a_0 &= \frac{\sigma_X^2}{2} && (3.29)\\
a_1 &= F + \sigma_X(\sigma_Y - \kappa) && (3.30)\\
a_2 &= \mu_Y - g - \sigma_Y\kappa - \pi - \chi && (3.31)\\
a_3 &= -\psi_A\frac{\pi+\chi}{\pi}\frac{\omega}{g} && (3.32)\\
b_0 &= \frac{\sigma_X^2}{2} && (3.33)\\
b_1 &= F + \sigma_X\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}\left(\sigma_Z + \frac{\omega'}{\omega}\sigma_X\right) - \sigma_X\frac{\gamma_A-1}{\gamma_A}\kappa && (3.34)\\
b_2 &= D\left(w^{\frac{(1-\psi_A)(\gamma_A-1)}{\gamma_A}}\xi^{1-\frac{1}{\gamma_A}}\right) - \left(\pi + \frac{\rho_A}{\gamma_A}\right) && (3.35)\\
b_3 &= -1 && (3.36)\\
c_0 &= \frac{\sigma_X^2}{2} && (3.37)\\
c_1 &= F + \sigma_X\frac{(1-\psi_B)(\gamma_B-1)}{\gamma_B}\left(\sigma_Z + \frac{\omega'}{\omega}\sigma_X\right) - \sigma_X\frac{\gamma_B-1}{\gamma_B}\kappa && (3.38)\\
c_2 &= D\left(w^{\frac{(1-\psi_B)(\gamma_B-1)}{\gamma_B}}\xi^{1-\frac{1}{\gamma_B}}\right) - \left(\pi + \frac{\rho_B}{\gamma_B}\right) && (3.39)\\
c_3 &= -1 && (3.40)
\end{aligned}
$$

In the equations above, the differential operator $\widehat{D}(a,b)$ is defined in the following way:

$$\widehat{D}(a,b) \quad = \quad D\left(w^a \xi^b\right) - a\frac{\omega'}{\omega}F(X) + bG(X) \tag{3.41}$$

3.2 Numerical Solution in MATLAB

In this section we describe the implementation **IterationGarleanu**.
The problem in the previous section is analogous to the problem in Chapter 2. Thus, we use the same discretization as in Chapter 2.3.

The required parameters of paper [1] are given as follows:

$\mu_Z = 0.018$	$\psi_B = 0.95$
$\sigma_Z = 0.041$	$\rho_A = 0.001$
$v = 0.1$	$\rho_B = 0.3$
$\pi = 0.01$	$\beta_1 = 0.99$
$\chi = 0.018$	$\beta_2 = 0.81$
$\gamma_A = 3$	$\beta_3 = -0.5$
$\gamma_B = 18$	$\beta_4 = 0.3$
$\psi_A = 1$	

For computing Φ, Ψ and Θ, we still need to estimate f, g, g', g'', ω, ω', ω'', σ_X, κ, μ_Y, and σ_Y.

The function g is defined by $f(H)$ and the function ω by $f'(H)$. Hence, we have to compute H and f first. For both H and f, we need to define $\alpha(H)$ as a function of H. The following equations from paper [1] help to compute α, H, and f, where $\mathcal{N}$ denotes the cumulative distribution function of the standard normal distribution.

$$\alpha(H(X)) = (\beta_1 - \beta_2)\mathcal{N}(\beta_3(X - \beta_4)) + \beta_2 \tag{3.42}$$

$$H_t = 1 - \frac{H_t}{\alpha(H_t)}\left(\frac{(1-\psi_A)}{\psi_A}X_t + \frac{(1-\psi_B)}{\psi_B}(1 - X_t)\right) \tag{3.43}$$

$$f'(H) = \frac{\alpha(H)f(H)}{H}, f(0) = 0 \tag{3.44}$$

We have to solve equation (3.44) numerically by the function **ffunction** which computes f and f' with the calculation of α and H.
Hence, we solve the differential equation (3.44) in the interval [0.001,1] with $f(0.001) = 0.001$ as an approximation for [0,1] and $f(0) = 0$. Furthermore, we compute the derivative f' from equation (3.44) which is done by the function **dffunction**. Moreover, we need to evaluate f and f' at the value of H.

The following component of the program **IterationGarleanu** evaluates α, f, f' and H by the function **ffunction** and furthermore defines ω and g.

$$
\begin{aligned}
[f, f'] &= ffunction(\psi_A, \psi_B, X, \beta_1, \beta_2, \beta_3, \beta_4); \\
\omega &= f'; \\
g &= f;
\end{aligned}
$$

We have evaluated f, g, and ω, and can now define g', g'', ω', and ω'' where g', ω' use backward differencing and g'', ω'' use the difference quotient for the second derivative.

Thus, we still need σ_X, κ, μ_Y, and σ_Y for the numerical solution. The parameters σ_X and κ both satisfy the equations (3.14) and (3.17). Hence, we can transform this equations into the following form:

$$
\begin{aligned}
a_{11}\sigma_X + a_{12}\kappa &= f_1 && (3.45) \\
a_{21}\sigma_X + a_{22}\kappa &= f_2 && (3.46)
\end{aligned}
$$

where the coefficients are given by:

$$
\begin{aligned}
a_{11} &= \frac{1}{X} + \frac{g'}{g} - \frac{(1-\psi_A)(\gamma_A - 1)}{\gamma_A}\frac{\omega'}{\omega} && (3.47) \\
a_{12} &= -\frac{1}{\gamma_A} && (3.48) \\
a_{21} &= \frac{g'}{g} - X\frac{(1-\psi_A)(\gamma_A - 1)}{\gamma_A}\frac{\omega'}{\omega} - (1-X)\frac{(1-\psi_B)(\gamma_B - 1)}{\gamma_B}\frac{\omega'}{\omega} && (3.49) \\
a_{22} &= -\frac{X}{\gamma_A} - \frac{1-X}{\gamma_B} && (3.50) \\
f_1 &= \left(\frac{(1-\psi_A)(\gamma_A - 1)}{\gamma_A} - 1\right)\sigma_Z && (3.51) \\
f_2 &= \left(X\frac{(1-\psi_A)(\gamma_A - 1)}{\gamma_A} - 1 + (1-X)\frac{(1-\psi_B)(\gamma_B - 1)}{\gamma_B}\right)\sigma_Z && (3.52)
\end{aligned}
$$

It is not permitted to divide by zero. Hence, we have to modify equation (3.47) as follows:

$$
a_{11} = \frac{1}{\max(X, \epsilon)} + \frac{g'}{g} - \frac{(1-\psi_A)(\gamma_A - 1)}{\gamma_A}\frac{\omega'}{\omega} \qquad (3.53)
$$

In the numerical implementation we use $\epsilon = 10^{-10}$.
Hence, we can solve the linear system of equations which follows from equations (3.45) and (3.46).

This is realized in the function **skfunction**. We access this function in the main implementation in the following way:

$$[\sigma_X, \kappa] = skfunction(n, X, g, g', \omega, \omega', \gamma_A, \gamma_B, \psi_A, \psi_B, \sigma_Z);$$

The parameters σ_Y and μ_Y can now easily be computed from equations (3.21) and (3.22). But we have to take into consideration that μ_Y also depends on F. Hence, we have to compute μ_Y in every iteration step.

Thus, all requirements for the general approach are fulfilled. Moreover, the damping coefficients are defined anologous to equations (2.93) - (2.98) in Chapter 2.4.

$$\widetilde{a}_2 = a_2 + \tau \tag{3.54}$$
$$\widetilde{a}_3 = a_3 + \tau U \tag{3.55}$$
$$\widetilde{b}_2 = b_2 + \tau \tag{3.56}$$
$$\widetilde{b}_3 = b_3 + \tau V \tag{3.57}$$
$$\widetilde{c}_2 = c_2 + \tau \tag{3.58}$$
$$\widetilde{c}_3 = c_3 + \tau W \tag{3.59}$$

where τ denotes the damping coefficient and U, V, and W stand for the variables of the three differential equations.

Note that the notations of the damping coefficients in the implementation are of opposite signs.

The discretization has an analogous structure as the original PDE (cf. (2.1) - (2.5)). Hence, one could show by analogous methods that a discrete solution exists and the iteration converges. Thus, the problem in this chapter can be implemented with MATLAB using the implementation discussed in the previous chapter with the modifications outlined in this section.
The complete implementations are included in the appendix.

Chapter 4

Application II

The paper [2] by Vito D. Gala also presents a general equilibrium model. The author examines a set of heterogeneous firms that are infinitely-lived. Thus, every individual firm is considered in this set. Furthermore, all investors are combined into one representative investor.
This model is developed of the work of Gomes, Kogan, and Zhang[1], who have constructed a multiple-firm general equilibrium model. In contrast to that paper the author of [2] models firms not projects.

4.1 Derivation of the Problem

In the following we describe the problem of paper [2] by the required equations.
In this paper $\overline{q}$, the component of the firm marginal q that is common to all firms, and $\widehat{q}$, which denotes an extra contribution to the market value of firm capital, are the two variables that can be identified with the functions F and G in Chapter 2.

Furthermore, we need the following equations of paper [2] to describe the problem of this paper:

$$q_i = \overline{x}\overline{q}(a,\omega) + [x_i - \overline{x}]\widehat{q}(a,\omega) \tag{4.1}$$

$$\begin{aligned} c^*(a,\omega) &= e^a\omega + f - \widehat{i} \\ &\quad - \left(\frac{n-1}{\alpha n}\right)^{n-1}\left(g(a,\omega,n-1,1) + \left(\frac{n-1}{n}\right)g(a,\omega,n,1)\right) \end{aligned} \tag{4.2}$$

$$g(a,\omega,m_1,m_2) = \widehat{q}^{m_1}\sum_{k=0}^{m_1}\frac{\Gamma(m_1+1)\Gamma_U(k+v,\theta\widetilde{x})}{\Gamma(m_1+1-k)\Gamma(k+m_2)\Gamma(v)}\left(-\widetilde{x}\right)^{m_1-k}\theta^{-k} \tag{4.3}$$

$$\widetilde{x} = \frac{1-\overline{x}(\overline{q}-\widehat{q})}{\widehat{q}} \tag{4.4}$$

[1] cf. [10]

where $q_i = q(a, \omega, x_i)$ denotes the firm marginal and $c^*(a, \omega)$ stands for the consumption policy rate. Moreover, x denotes the firm specific productivity, a stands for the economy wide productivity and ω refers the capital-weighted average of the firm specific productivities. Furthermore, $\widehat{i}$ is the minimum investment rate, γ denotes the risk aversion coefficient, f stands for the time-invariant component of the productivity, α refers the adjustment cost parameter, n denotes the degree of curvature of the adjustment cost function, $\widetilde{x}$ denotes the investment threshold and $\overline{x}$ is the long-run mean of the idiosyncratic productivity.

Moreover, the functions $\Gamma(a)$ and $\Gamma_U(a, z)$ are defined as

$$\Gamma(a) = \int_0^\infty x^{a-1} e^{-x} dx \tag{4.5}$$

$$\begin{aligned} \Gamma_U(a, z) &= \int_z^\infty x^{a-1} e^{-x} dx \\ &= \int_0^\infty x^{a-1} e^{-x} dx - \int_0^z x^{a-1} e^{-x} dx \\ &= \Gamma(a) - \int_0^z x^{a-1} e^{-x} dx \\ &= \Gamma(a) \left(1 - \frac{1}{\Gamma(a)} \int_0^z x^{a-1} e^{-x} dx\right) \end{aligned} \tag{4.6}$$

where

$$\Gamma_{inc} = \frac{1}{\Gamma(a)} \int_0^z x^{a-1} e^{-x} dx \tag{4.7}$$

is called incomplete gamma function.

Hence, we can define the system of algebraic equations that is given by equations (75) and (76)[2] in paper [2]:

$$\overline{q}_t = c^*(a_t, \omega_t)^\gamma \overline{\Phi}(a_t, \omega_t) \tag{4.8}$$

$$\widehat{q}_t = c^*(a_t, \omega_t)^\gamma \widehat{\Phi}(a_t, \omega_t) \tag{4.9}$$

[2] cf. [2, p. 38, Equations (75), (76)]

Furthermore, the system of partial differential equations is given by equations (79) and (80)[3] in paper [2]:

$$\left(\rho + (1-\gamma)\left(\delta - \widehat{i}\right) + \gamma\left(\frac{n-1}{\alpha n}\right)^{n-1} g(a,\omega,n-1,1)\right)\overline{\Phi} - D\left[\overline{\Phi}\right] = \frac{e^a + \overline{x}^{-1}\left(f - \widehat{i}\right)}{c^*(a,\omega)^\gamma} \quad (4.10)$$

$$\left(\rho + \kappa_x + (1-\gamma)\left(\delta - \widehat{i}\right) + \gamma\left(\frac{n-1}{\alpha n}\right)^{n-1} g(a,\omega,n-1,1)\right)\widehat{\Phi} - D\left[\widehat{\Phi}\right] = \frac{e^a}{c^*(a,\omega)^\gamma} \quad (4.11)$$

The infinitesimal generator of the stochastic processes a and ω is defined by the next equation:

$$D\left[\Phi\right] = \kappa_a(\overline{a} - a)\partial_a\Phi + \frac{1}{2}\sigma_a^2\partial_{aa}^2\Phi + \mu_\omega(a,\omega)\partial_\omega\Phi \quad (4.12)$$

Although ρ denotes the time preference parameter, δ stands for the economic depreciation rate, κ_x is the rate of mean reversion of the idiosyncratic productivity, κ_a is the rate of mean reversion of the productivity variable, $\overline{a}$ refers the long-run mean of the aggregate productivity and σ_a denotes the volatility of the aggregate productivity. Moreover, we will see later that $\mu_\omega(a,\omega)$ is the drift term of the stochastic process ω.

In order to solve the equilibrium we have to modify these equations. This yields to equations (118a), (118b), (119), and (120)[4] in paper [2]:

At each node $i \times j$ with $i = 1, 2, ..., I$ and $j = 1, 2, ..., J$ we have a discretized system of two algebraic equations.

$$\overline{q}_{i,j} = (c_{i,j})^\gamma \overline{\Phi}_{i,j} \quad (4.13)$$

$$\widehat{q}_{i,j} = (c_{i,j})^\gamma \widehat{\Phi}_{i,j} \quad (4.14)$$

[3] cf. [2, p. 38, Equations (79), (80)]
[4] cf. [2, p. 47, Equations (118a), (118b), (119), (120)]

This equations can be solved iteratively in combination with the next system of differential equations for $\overline{\Phi}_{i,j}$ and $\widehat{\Phi}_{i,j}$.

$$\left(\rho + (1-\gamma)\left(\delta - \widehat{i}\right) + \gamma\left(\frac{n-1}{\alpha n}\right)^{n-1} g_{i,j}\right)\overline{\Phi}_{i,j} - \widehat{D}\left[\overline{\Phi}_{i,j}\right] = \frac{e^{a_i} + \overline{x}^{-1}\left(f - \widehat{i}\right)}{(c_{i,j})^{\gamma}} \tag{4.15}$$

$$\left(\rho + \kappa_x + (1-\gamma)\left(\delta - \widehat{i}\right) + \gamma\left(\frac{n-1}{\alpha n}\right)^{n-1} g_{i,j}\right)\widehat{\Phi}_{i,j} - \widehat{D}\left[\widehat{\Phi}_{i,j}\right] = \frac{e^{a_i}}{(c_{i,j})^{\gamma}} \tag{4.16}$$

The functions $c_{i,j}$, $g_{i,j}$ and the finite-difference approximation $\widehat{D}\left[\Phi_{i,j}\right]$ are given by the following equations:

$$\begin{aligned} c_{i,j} &= c^*(a_i, \omega_j) && (4.17)\\ g_{i,j} &= g(a_i, \omega_j, n-1, 1) && (4.18)\\ \widehat{D}[\Phi_{i,j}] &= \kappa_a(\overline{a} - a_i)[\partial_a \Phi]_{i,j} + \frac{1}{2}\sigma_a^2[\partial_{aa}^2 \Phi]_{i,j} + \mu_\omega(a_i, \omega_j)[\partial_\omega \Phi]_{i,j} && (4.19) \end{aligned}$$

Furthermore, we discretize the derivatives of Φ. For the discretization of the first derivatives we use upwind techniques[5] as described in Chapter 2.1.

The derivative of second order with respect to a is discretized by:

$$[\partial_{aa}^2 \Phi]_{i,j} = \frac{\Phi_{i+1,j} - 2\Phi_{i,j} + \Phi_{i-1,j}}{h_a^2} \tag{4.20}$$

The derivative of first order with respect to a is approximated for $-\kappa_a(\overline{a} - a_i) < 0$ by forward differencing:

$$[\partial_a \Phi]_{i,j} = \frac{\Phi_{i+1,j} - \Phi_{i,j}}{h_a} \tag{4.21}$$

[5] cf. [3, p. 13]

Furthermore, the derivative of first order with respect to a for $-\kappa_a(\overline{a} - a_i) > 0$ is discretized by backward differencing:

$$[\partial_a \Phi]_{i,j} = \frac{\Phi_{i,j} - \Phi_{i-1,j}}{h_a} \tag{4.22}$$

In an analogous way the derivative of first order with respect to ω for $-\mu_\omega(a_i, \omega_j) < 0$ is approximated by forward differencing:

$$[\partial_\omega \Phi]_{i,j} = \frac{\Phi_{i,j+1} - \Phi_{i,j}}{h_\omega} \tag{4.23}$$

Moreover, the derivative of first order with respect to ω is approximated for $-\mu_\omega(a_i, \omega_j) > 0$ by backward differencing:

$$[\partial_\omega \Phi]_{i,j} = \frac{\Phi_{i,j} - \Phi_{i,j-1}}{h_\omega} \tag{4.24}$$

Thus, for fixed j and $i = 1, 2, ..., I$ we can rewrite equations (4.15) and (4.16) in the following way

$$\begin{aligned} \overline{A}_{i,j}\overline{\Phi}_{i,j} + B_i\overline{\Phi}_{i+1,j} + C_i\overline{\Phi}_{i-1,j} &= \overline{F}_{i,j} && (4.25)\\ \widehat{A}_{i,j}\widehat{\Phi}_{i,j} + B_i\widehat{\Phi}_{i+1,j} + C_i\widehat{\Phi}_{i-1,j} &= \widehat{F}_{i,j} && (4.26) \end{aligned}$$

where the coefficients are given by

$$\begin{aligned} \overline{A}_{i,j} &= \rho + (1-\gamma)\left(\delta - \widehat{i}\right) + \gamma\left(\frac{n-1}{\alpha n}\right)^{n-1} g_{i,j} + \frac{\sigma_a^2}{h_a^2} \\ &\quad + \max\left(\frac{-\kappa_a(\overline{a} - a_i)}{h_a}, 0\right) - \min\left(\frac{-\kappa_a(\overline{a} - a_i)}{h_a}, 0\right) && (4.27)\\ \widehat{A}_{i,j} &= \rho + \kappa_x + (1-\gamma)\left(\delta - \widehat{i}\right) + \gamma\left(\frac{n-1}{\alpha n}\right)^{n-1} g_{i,j} + \frac{\sigma_a^2}{h_a^2} \\ &\quad + \max\left(\frac{-\kappa_a(\overline{a} - a_i)}{h_a}, 0\right) - \min\left(\frac{-\kappa_a(\overline{a} - a_i)}{h_a}, 0\right) && (4.28) \end{aligned}$$

$$
\begin{aligned}
B_i &= \min\left(\frac{-\kappa_a(\overline{a}-a_i)}{h_a},0\right)-\frac{\sigma_a^2}{2h_a^2} && (4.29)\\
C_i &= -\max\left(\frac{-\kappa_a(\overline{a}-a_i)}{h_a},0\right)-\frac{\sigma_a^2}{2h_a^2} && (4.30)\\
\overline{F}_{i,j} &= \left(e^{a_i}+\overline{x}^{-1}\left(f-\widehat{i}\right)\right)(c_{i,j})^{-\gamma}+\mu_\omega(a_i,\omega_j)\left[\partial_\omega\overline{\Phi}\right]_{i,j}^{old} && (4.31)\\
\widehat{F}_{i,j} &= e^{a_i}(c_{i,j})^{-\gamma}+\mu_\omega(a_i,\omega_j)\left[\partial_\omega\widehat{\Phi}\right]_{i,j}^{old} && (4.32)
\end{aligned}
$$

and where $\left[\partial_\omega\overline{\Phi}\right]_{i,j}^{old}$ and $\left[\partial_\omega\widehat{\Phi}\right]_{i,j}^{old}$ are given by the suitable difference quotiont (4.23) or (4.24) with $\overline{\Phi}$ and $\widehat{\Phi}$ of the previous iteration step.

4.2 General Form and Discretization

The general form of the system of partial differential equations is given for each unknown Φ in the form:

$$-\frac{1}{2}\sigma_a^2\partial_{aa}^2\Phi + \beta(a,\omega)\partial_a\Phi + \epsilon(a,\omega)\partial_\omega\Phi + \lambda(a,\omega)\Phi \quad = \quad \chi(a,\omega) \tag{4.33}$$

At each node $i \times j$ with $i = 1, 2, ..., I$ and $j = 1, 2, ..., J$ we obtain the following equation:

$$-\frac{1}{2}\sigma_a^2[\partial_{aa}^2\Phi]_{i,j} + \beta(a_i,\omega_j)[\partial_a\Phi]_{i,j} + \epsilon(a_i,\omega_j)[\partial_\omega\Phi]_{i,j} + \lambda(a_i,\omega_j)\Phi_{i,j} \quad = \quad \chi(a_i,\omega_j) \tag{4.34}$$

Moreover, we discretize the derivatives of Φ. For the discretization of the first derivatives we use upwind techniques[6] as described in Chapter 2.1.

The derivative of second order with respect to a is discretized by:

$$[\partial_{aa}^2\Phi]_{i,j} = \frac{\Phi_{i+1,j} - 2\Phi_{i,j} + \Phi_{i-1,j}}{h_a^2} \tag{4.35}$$

The derivative of first order with respect to a is approximated for $\beta(a_i,\omega_j) < 0$ by forward differencing:

$$[\partial_a\Phi]_{i,j} = \frac{\Phi_{i+1,j} - \Phi_{i,j}}{h_a} \tag{4.36}$$

Moreover, the derivative of first order with respect to a for $\beta(a_i,\omega_j) > 0$ is discretized by backward differencing:

$$[\partial_a\Phi]_{i,j} = \frac{\Phi_{i,j} - \Phi_{i-1,j}}{h_a} \tag{4.37}$$

[6] cf. [3, p. 13]

In an analogous way the derivative of first order with respect to ω is approximated for $\epsilon(a_i, \omega_j) < 0$ by forward differencing:

$$[\partial_\omega \Phi]_{i,j} = \frac{\Phi_{i,j+1} - \Phi_{i,j}}{h_\omega} \tag{4.38}$$

Furthermore, the derivative of first order with respect to ω for $\epsilon(a_i, \omega_j) > 0$ is discretized by backward differencing:

$$[\partial_\omega \Phi]_{i,j} = \frac{\Phi_{i,j} - \Phi_{i,j-1}}{h_\omega} \tag{4.39}$$

We use implicit and explicit discretization in the course of the iteration. I.e., we use the value of Φ from the previous iteration step in (4.39).

This leads to the following discretized problem to be solved in each iteration step

$$\widetilde{A}_{i,j}\Phi_{i,j} + \widetilde{B}_{i,j}\Phi_{i+1,j} + \widetilde{C}_{i,j}\Phi_{i-1,j} \quad = \quad \widetilde{F}_{i,j} \tag{4.40}$$

where the coefficients are given by

$$\widetilde{A}_{i,j} = \lambda(a_i, \omega_j) + \frac{\sigma_a^2}{h_a^2} + \max\left(\frac{\beta(a_i, \omega_j)}{h_a}, 0\right) - \min\left(\frac{\beta(a_i, \omega_j)}{h_a}, 0\right) + \tau \tag{4.41}$$

$$\widetilde{B}_{i,j} = \min\left(\frac{\beta(a_i, \omega_j)}{h_a}, 0\right) - \frac{\sigma_a^2}{2h_a^2} \tag{4.42}$$

$$\widetilde{C}_{i,j} = -\max\left(\frac{\beta(a_i, \omega_j)}{h_a}, 0\right) - \frac{\sigma_a^2}{2h_a^2} \tag{4.43}$$

$$\widetilde{F}_{i,j} = \chi(a_i, \omega_j) - \epsilon(a_i, \omega_j)[\partial_\omega \Phi]_{i,j}^{old} + \tau \Phi^o l d_{i,j} \tag{4.44}$$

and where $[\partial_\omega \Phi]_{i,j}^{old}$ is given by the suitable difference quotient (4.38) or (4.39) with Φ of the previous iteration step. Moreover, the damping factor is denoted by τ.

If we choose the parameter as done in the next section, we will get problems with the implementation. Thus, the following changes have to be realized to generate a more general implementation:

$$\begin{aligned}
\gamma &= 1 && (4.45)\\
c_{i,j} &= 0.5 \quad \text{for all i, j} && (4.46)\\
\mu_\omega(a_i, \omega_j) &= 1 \quad \text{for all i, j} && (4.47)\\
\beta(a_i, \omega_j) &= -\kappa_a(\overline{a} - a_i) && (4.48)\\
\epsilon(a_i, \omega_j) &= -\mu_\omega(a_i, \omega_j) && (4.49)\\
\lambda_1(a_i, \omega_j) &= \rho \quad \text{for the first } \Phi && (4.50)\\
\lambda_2(a_i, \omega_j) &= \rho + \kappa_x \quad \text{for the second } \Phi && (4.51)\\
\chi(a_i, \omega_j) &= e^{a_i} && (4.52)
\end{aligned}$$

With this changes the implementation can be carried out as described in the next section. The program with the modifications outlined in this section is called **IterationGala_general**. It is also included in the appendix.

4.3 Numerical Solution in MATLAB

In the following section we describe the implementation **IterationGala**.
The main difference to the previous chapters is that we have now two processes a and ω, and hence partial differential equations with two variables. Thus, we have two increments h_a and h_ω in the discretization of these partial differential equations.

The required parameters of paper [2] are given as follows:

$\gamma = 14$	$\overline{a} = -2.22$
$\rho = 0.01$	$\kappa_a = 0.27$
$\alpha = 2$	$\sigma_a = 0.05$
$n = 2$	$\overline{x} = 1$
$\delta = 0.13$	$\kappa_x = 0.15$
$\widehat{i} = 0.12$	$\sigma_x = 0.27$
$f = 0.12$	

For the computation of $\overline{\Phi}$, $\widehat{\Phi}$, $\overline{q}$, and $\widehat{q}$ we have to evaluate the functions $g(a, \omega, m_1, m_2)$, $\mu_\omega(a, \omega)$, and $c(a, \omega)$. For these functions we need to compute the parameters θ, v, $\widetilde{x}$, and i. The equations of the parameters $\widetilde{x}$, θ, and v derive from page 33 and 34 of paper [2].

$$\widetilde{x} = \frac{1 - \overline{x}(\overline{q} - \widehat{q})}{\widehat{q}} \tag{4.53}$$

$$\theta = \frac{2\kappa_x}{\sigma_x^2} \tag{4.54}$$

$$v = \frac{2\kappa_x \overline{x}}{\sigma_x^2} \tag{4.55}$$

Thus, we can compute $g(a, \omega, m_1, m_2)$ from equation (4.3), which is done by the function **g_{ij}function**. Furthermore, $\Gamma_U(k + v, \theta\widetilde{x})$ is defined by equation (4.6).

Hence, we can define the missing parameters $c(a, \omega)$, i, and $\mu_\omega(a, \omega)$ by using equation (4.2) and equations (20), (23)[7] in paper [2]:

$$i = \widehat{i} + \left(\frac{n-1}{\alpha n}\right)^{n-1} g(a, \omega, n-1, 1) \tag{4.56}$$

$$\mu_\omega = \left(\kappa_x + i - \widehat{i}\right)(\overline{x} - \omega) + (i - \widehat{i})\theta^{-1}\frac{g(a, \omega, n-1, 0)}{g(a, \omega, n-1, 1)} \tag{4.57}$$

[7] cf. [2, p. 12, Equations (20), (23)]

In order to compute $\overline{\Phi}$ and $\widehat{\Phi}$ from equations (4.25) and (4.26), we have to calculate the derivative $[\partial_\omega \Phi]_{i,j}^{old}$ of the previous value of Φ. This is realized by the function **dPhidw_function**:

Hence, we obtain the desired derivative denoted by the variable y:

$$y = A\Phi \tag{4.58}$$

where A is a three-diagonal matrix with the first sub diagonal r, the main diagonal s and the first super diagonal t:

$$r = -\max\left(sign(-\mu), 0\right)\frac{1}{h_\omega} \tag{4.59}$$

$$s = \left[\max\left(sign(-\mu), 0\right) + \min\left(sign(-\mu), 0\right)\right]\frac{1}{h_\omega} \tag{4.60}$$

$$t = -\min\left(sign(-\mu), 0\right)\frac{1}{h_\omega} \tag{4.61}$$

As above, the damped coefficients are defined in the following way:

$$\widetilde{A_1} = \overline{A} + \tau \tag{4.62}$$

$$\widetilde{A_2} = \widehat{A} + \tau \tag{4.63}$$

$$\widetilde{F_1} = \overline{F} + \tau\overline{\Phi} \tag{4.64}$$

$$\widetilde{F_2} = \widehat{F} + \tau\widehat{\Phi} \tag{4.65}$$

Hence, we can compute the sulutions $\overline{\Phi}$ and $\widehat{\Phi}$ from equations (4.25) and (4.26). This is realized by the function **DGLfunction_Gala**:

Thus, we obtain Φ by solving a linear system of equations:

$$\Phi = M^{-1}\,\widetilde{F} \tag{4.66}$$

where Φ is a variable for $\overline{\Phi}$ or $\widehat{\Phi}$ and $\widetilde{F}$ stands for $\widetilde{F_1}$ or $\widetilde{F_2}$.

Moreover, M is a three-diagonal matrix with the first sub diagonal C, the main diagonal $\widetilde{A}$ and the first super diagonal B, where $\widetilde{A}$ denotes $\widetilde{A_1}$ or $\widetilde{A_2}$, B stands for equation (4.29) and C denotes equation (4.30).

Note that B, C are vectors and $\widetilde{A_1}$, $\widetilde{A_2}$, $\widetilde{F_1}$, $\widetilde{F_2}$ are matrices. Hence, $\widetilde{A}$ and $\widetilde{F}$ consists only of the jth column of the matrix and we have to use the function **DGLfunction_Gala** for every j.

Moreover, we compute $\overline{q}$ and $\widehat{q}$ from equations (4.13) and (4.14).
Hence, we can calculate $\overline{\Phi}$, $\widehat{\Phi}$ and $\overline{q}$, $\widehat{q}$ in turn until convergence is achieved.

The complete implementations are included in the appendix.

Chapter 5

Results

This chapter contains the numerical results related to the problems of Chapter 2, 3, and 4. All computations reported in the following were performed in MATLAB.

5.1 General Convergence Test

The following two diagrams illustrate that the program of Chapter 2 is convergent. The diagrams are generated by the program **Iteration** with step size $h = 0.01$ and the damping factor $\tau = 30$. The implementation is also convergent for a smaller τ around 5.
Furthermore, we have realized several tests with different step sizes h and different damping factors τ, which yielded similar results.

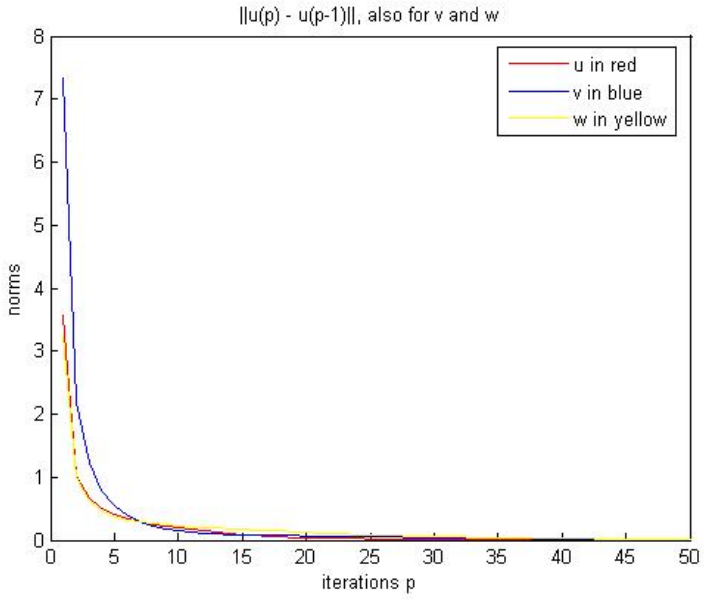

Figure 5.1: These graphics illustrate the Euclidean norms of vectors of U,V, and W.

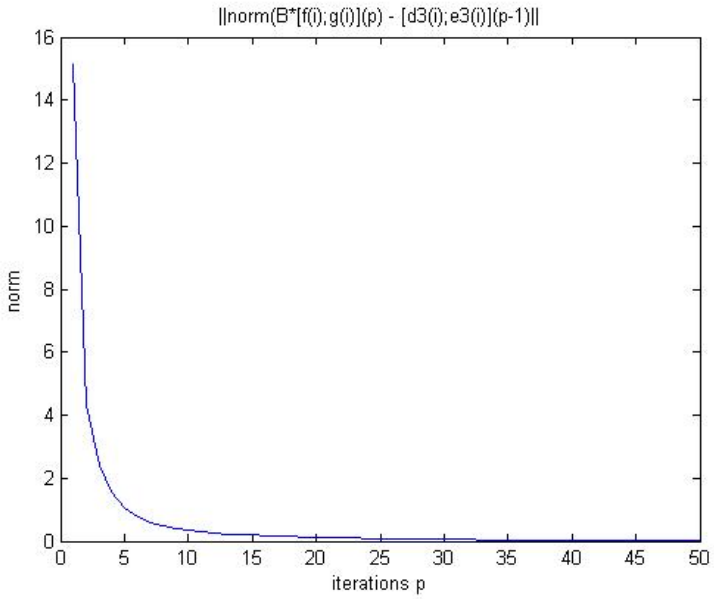

Figure 5.2: A plot of the residual norm of the linear system of equations for F and G versus the number of iterations is illustrated.

Figure 5.1 illustrates the convergence of the solution of the problem in Chapter 2 by plotting the following Euclidean norms for the number of iterations $p = 1, ..., 50$:

$$\|U^p - U^{p-1}\|_2 \tag{5.1}$$
$$\|V^p - V^{p-1}\|_2 \tag{5.2}$$
$$\|W^p - W^{p-1}\|_2 \tag{5.3}$$

Figure 5.2 demonstrates that the solution really converge to the right result. We compute the vector N^p by the following equation with matrix B of Assumption 1 in Chapter 2.2:

$$N^p = \left\| B^p \begin{pmatrix} F \\ G \end{pmatrix}^p - \begin{pmatrix} d_3 \\ e_3 \end{pmatrix}^{p-1} \right\|_2$$

Then we generate the Euklidean norm $\|N^p\|_2$.

5.2 Results for Application I

The next two graphics illustrate that the iteration of paper [1] of Chapter 3 is numerical stable and convergent. The diagrams are generated by the program **IterationGarleanu** with step size $h = 0.01$ and the damping factor $\tau = 10$.
Moreover, we have realized several tests with different step sizes h and different damping factors τ and have yielded similar results.

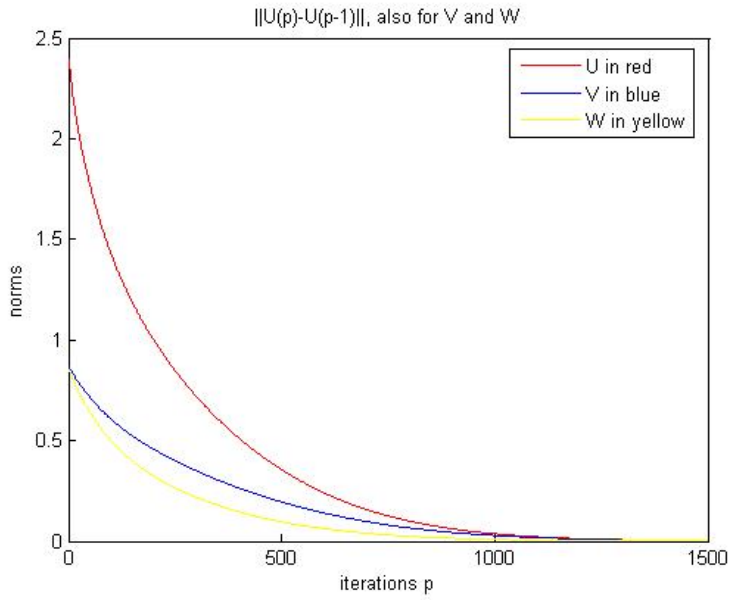

Figure 5.3: This graphics illustrate the Euclidean norms of vectors of U, V, and W.

Figure 5.3 illustrates the convergence of the solution of the problem in Chapter 3 by plotting the following Euclidean norms for $p = 1, ..., 1500$:

$$\|U^p - U^{p-1}\|_2 \tag{5.4}$$
$$\|V^p - V^{p-1}\|_2 \tag{5.5}$$
$$\|W^p - W^{p-1}\|_2 \tag{5.6}$$

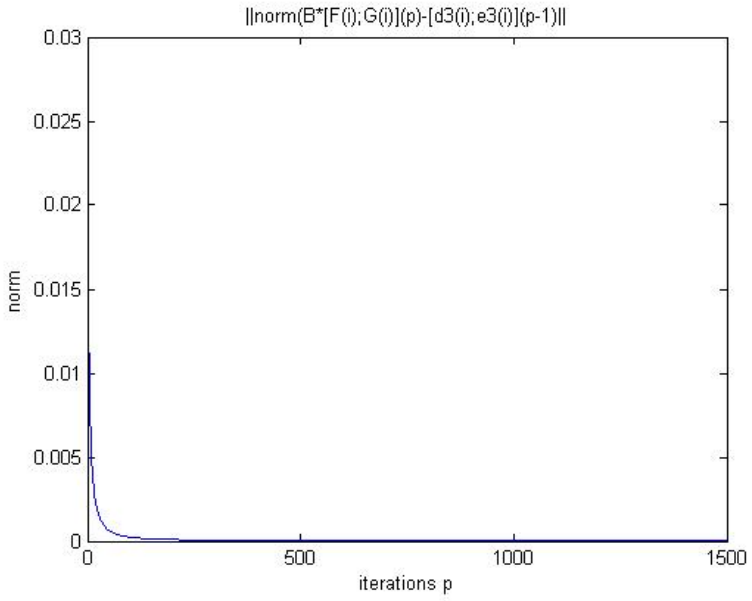

Figure 5.4: This diagram depicts the residual norm of the linear system of equations for F and G versus the number of iterations.

Figure 5.4 illustrates that the solution converge to the right result. We compute the vector N^p by the following equation with matrix B of Assumption 1 in Chapter 2.2 and d_3, e_3, and the entries of B of Chapter 3.1:

$$N^p = \left\| B^p \begin{pmatrix} F \\ G \end{pmatrix}^p - \begin{pmatrix} d_3 \\ e_3 \end{pmatrix}^{p-1} \right\|_2$$

Then we generate the Euklidean norm $\|N^p\|_2$.

Moreover, the relative errors of the results U, V, W, and e in 1500 iterations are given by the next graphics. There e denotes the residual norm of the linear system of equations for F and G as in the previous diagram. The graph is plotted versus the number of calculated errors (index of grid size). The first error is computed by the Euclidean norm of the difference between the results of step size $h_0 = 0.1$ and $h_1 = 0.025$ divided by the Euclidean norm of the result. For the second error we need the results of step size $h_1 = 0.025$ and $h_2 = 0.00625$, and for the third error we use the results of step size $h_2 = 0.00625$ and $h_3 = 0.0015625$.

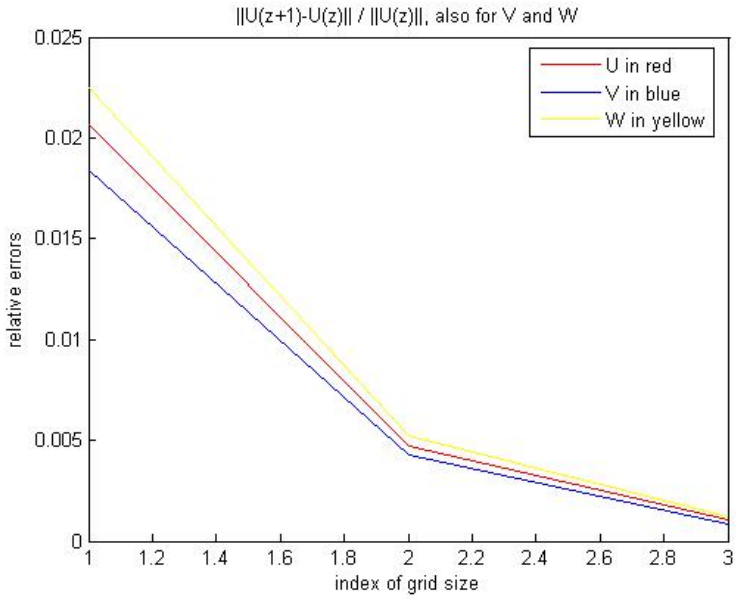

Figure 5.5: This graph depicts the relative errors of U, V, and W versus the index of grid size.

Figure 5.5 illustrates the relative errors of the solutions U, V, and W of Chapter 3 versus the index of grid size. The relative errors are given by:

$$\frac{\|U^{z+1} - U^z\|_2}{\|U^z\|_2} \tag{5.7}$$

$$\frac{\|V^{z+1} - V^z\|_2}{\|V^z\|_2} \tag{5.8}$$

$$\frac{\|W^{z+1} - W^z\|_2}{\|W^z\|_2} \tag{5.9}$$

where z is the index of grid size.

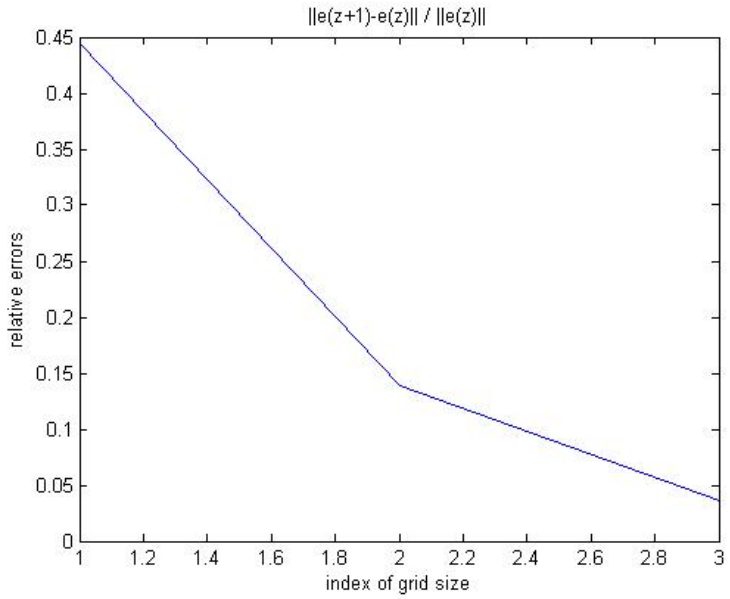

Figure 5.6: This diagram illustrates the relative error of e versus the index of grid size.

Figure 5.6 depicts the relative error of the residual norm of the linear system of equations for F and G versus the index of grid size. The residual norm is denoted by e and the described relative error is given by:

$$\frac{\|e^{z+1} - e^z\|_2}{\|e^z\|_2} \tag{5.10}$$

with

$$e \quad = \quad \|N^{1500}\|_2 \tag{5.11}$$

The following graphics illustrate the Sharpe ratio κ and the volatility σ_X of X after 1500 iterations, which appear exactly as in the paper [1] of Gârleanu and Panageas.

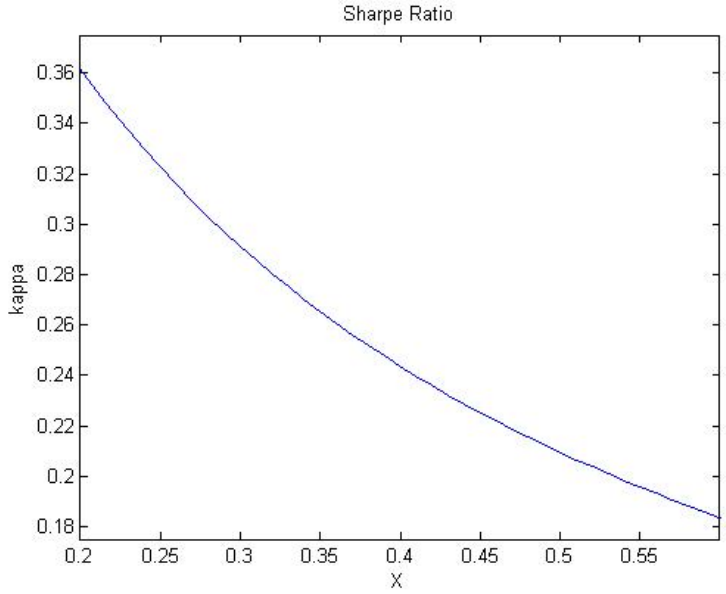

Figure 5.7: The diagram illustrates κ as a function of X.

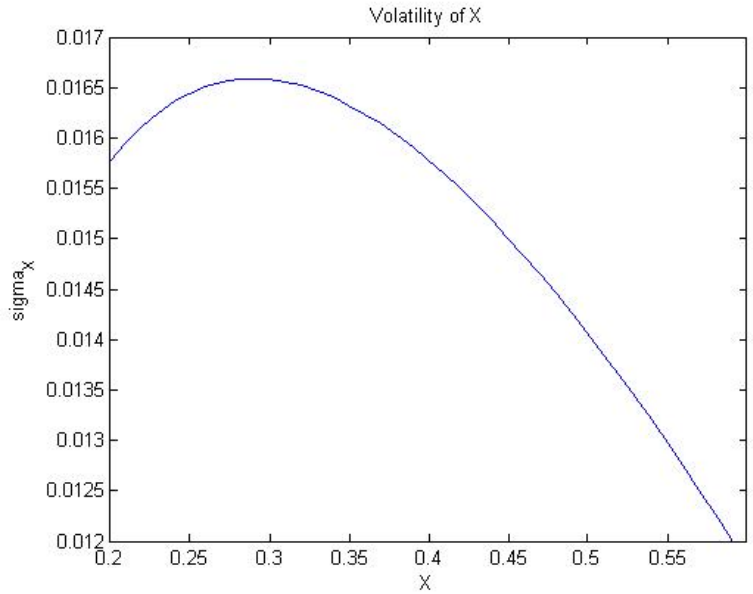

Figure 5.8: The graphic depicts σ_X as a function of X.

The next diagrams illustrate the two variables F and G, which correspond to the drift rate μ_X of X and the interest rate r. These graphics also result from 1500 iterations. The results for μ_X and r are in the same order of magnitude as the results in [1], but their form looks slightly different. In particular the interest rate r in this figure is convex and concave in [1]. Due to our convergence studies we are confident of the correctness of our results.

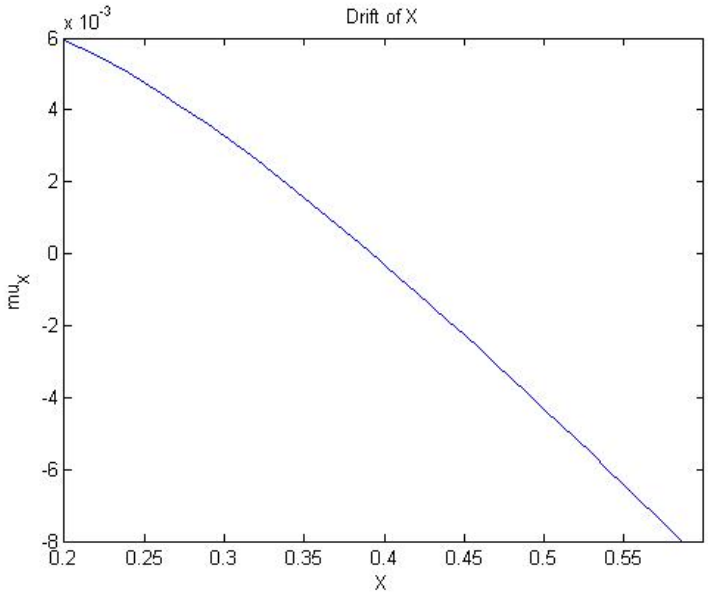

Figure 5.9: The graphic depicts $F(X) = \mu_X$ as a function of X.

Figure 5.9 illustrates the drift rate μ_X of X as a function of X. The drift rate also corresponds to the variable F.

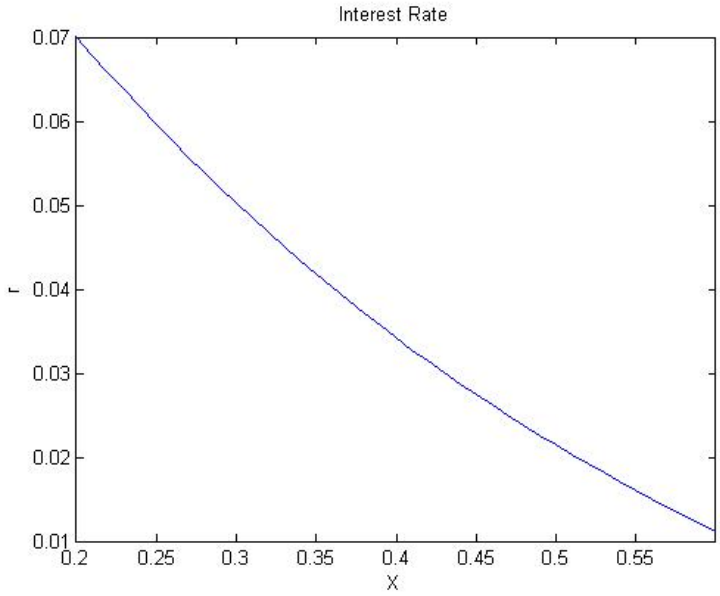

Figure 5.10: This diagram illustrates $G(X) = r$ as a function of X.

Figure 5.10 depicts the interest rate r as a function of X. The interest rate also corresponds to the variable G.
In this diagram the interest rate r is convex and not concave as in paper [1].

5.3 Results for Application II

The following two graphics illustrate the convergence of the solution of the general implementation of paper [2] in Chapter 4. The diagrams are generated by the program **IterationGala_general** with $Q = \Omega = 1$, I = 10, J = 20, and the damping factor $\tau = 10$.
We have also realized several tests with different step sizes h and different damping factors τ and have yielded similar results.

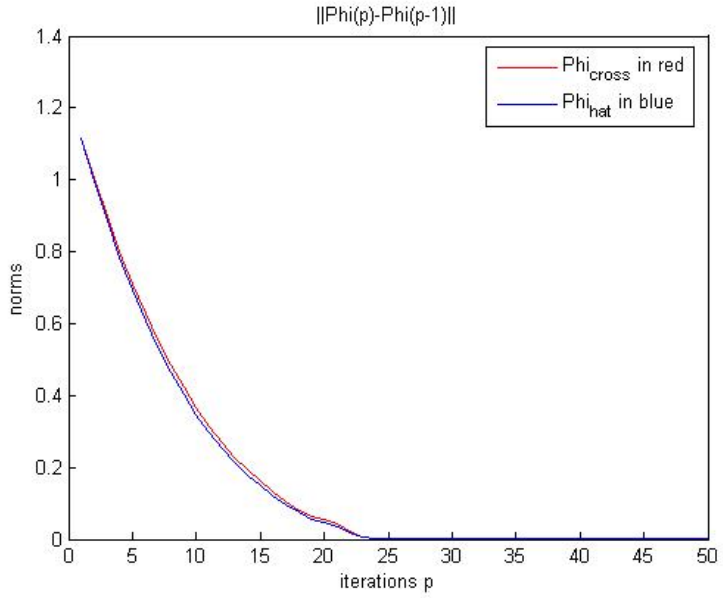

Figure 5.11: This graphics illustrate the Euclidean norms of $\overline{\Phi}$ and $\widehat{\Phi}$ versus the number of iterations.

Figure 5.11 depicts the convergence of the solutions $\overline{\Phi}$ and $\widehat{\Phi}$ of the problem in Chapter 4 by plotting the following Euclidean norms for $p = 1, ..., 50$:

$$\|\overline{\Phi}^p - \overline{\Phi}^{p-1}\|_2 \tag{5.12}$$

$$\|\widehat{\Phi}^p - \widehat{\Phi}^{p-1}\|_2 \tag{5.13}$$

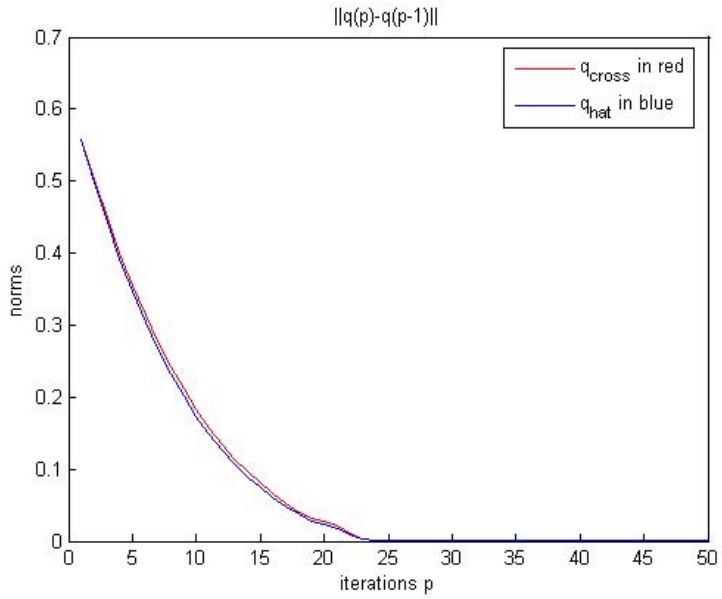

Figure 5.12: This diagram depicts the Euclidean norms of $\overline{q}$ and $\widehat{q}$ versus the number of iterations.

Figure 5.12 illustrates the convergence of the solutions $\overline{q}$ and $\widehat{q}$ of the problem in Chapter 4 by plotting the following Euclidean norms for $p = 1, ..., 50$:

$$\|\overline{q}^p - \overline{q}^{p-1}\|_2 \tag{5.14}$$

$$\|\widehat{q}^p - \widehat{q}^{p-1}\|_2 \tag{5.15}$$

Appendix A

A.1 List of Parameters

This section contains a list of parameters that are used in the previous chapters. The parameter definitions used in Chapter 3 result from paper [1] and the parameter definitions used in Chapter 4 result from paper [2].

Parameter Chapter 2	Definition
h	step size
x	vector with entries of [0,1]
n	dimension of x
tau, τ	damping parameter

Parameter Chapter 3	Definition
X_t	consumption share of type A agents
Y_t	stochastic output
v	mass of agents of type A
π	constant hazard rate of death
ρ_A	subjective discount rate of type A agents
ρ_B	subjective discount rate of type B agents
γ_A	relative risk aversion for type A agents
γ_B	relative risk aversion for type B agents
ψ_A	controls the relative importance of leisure and consumption of type A agents
ψ_B	analogue for type B agents
β_t^A	defined by $\frac{\phi^A(X_t)}{\zeta^A(X_t)}$
β_t^B	defined by $\frac{\phi^B(X_t)}{\zeta^B(X_t)}$
Z_t	exogenous productivity process
g_t	g(X_t) defined by $\frac{Y_t}{Z_t}$
ω_t	defined by $\frac{w_t}{Z_t}$
w_t	prevailing wage
ξ_t	stochastic discount factor
μ_X	drift coefficient of dX_t
σ_X	diffusion coefficient of dX_t
μ_Z	drift coefficient of $\frac{dZ_t}{Z_t}$
σ_Z	diffusion coefficient of $\frac{dZ_t}{Z_t}$
κ	Sharpe ratio
r	interest rate
χ	agent's endowment of hours declines exponentially over the life-cycle at the rate χ
β_1, β_2	constants that control the range of α
β_3, β_4	constants that control the steepness of the function α
H_t	agent's aggregate hours worked at time t

Parameter Chapter 4	Definition
i	index of the value of a
j	index of the value of ω
a	economy wide productivity (exogenous productivity index)
x	firm specific productivity
ω	capital-weighted average of the firm specific productivities (endogenous productivity index)
q	firm marginal
$\overline{q}$	component of the marginal q common to all firms
$\widehat{q}$	extra contribution to the market value of firm capital
c	consumption policy rate, denoted with $c(a, \omega)$
γ	risk aversion coefficient
ρ	time preference parameter
α	adjustment cost parameter
n	degree of curvature of the adjustment cost function
δ	economic depreciation rate
$\widehat{i}$	minimum investment rate
f	the time-invariant component of productivity
$\overline{a}$	long-run mean of the aggregate productivity
κ_a	rate of mean reversion of the productivity variable
σ_a	volatility of aggregate productivity
$\overline{x}$	long-run mean of the idiosyncratic productivity
$\widetilde{x}$	investment threshold defined by $\frac{1-\overline{x}(\overline{q}-\widehat{q})}{\widehat{q}}$
κ_x	rate of mean reversion of the idiosyncratic productivity
σ_x	volatility of the idiosyncratic productivity
$\Gamma_U(a, z)$	defined by $\int_z^\infty x^{a-1}e^{-x}dx$
$\Gamma(a)$	well-known Γ-function $\Gamma(a) = \int_0^\infty x^{a-1}e^{-x}dx$
h_a	increment of a
h_ω	increment of ω

A.2 MATLAB Implementations

This section includes programs that are all constructed with the programming language MATLAB.

A.2.1 The Implementation of Chapter 2

The following program **Iteration** solves the main problem in Chapter 2.

```
clear all;
close all;

h = 0.01;
n = 1/h + 1;
x = 0:h:1;
x = x';
f = zeros (n, 1);
g = zeros (n, 1);
tau = 30;

% initial values for u, v, and w:
u = x;
v = 3 * x;
w = x;

% coefficients of the algebraic equations:
d1 = ones (n, 1);
d2 = zeros (n, 1);
e1 = zeros (n, 1);
e2 = ones (n, 1);

% computation of f and g:
d3 = d3function (u, v, w);
e3 = e3function (u, v, w);
for k = 1:n

    B = [d1(k) d2(k); e1(k) e2(k)];
    o = B\[d3(k); e3(k)];
    f(k) = o(1);
    g(k) = o(2);

end
```

```
% iteration loop:
p = input ('Enter the number of iterations:');
for j = 1:p

    % definition of the coefficients of the differential equations:
    a0 = (x - 0) .* (x - 1);
    a1 = a1function (f);
    a2 = a2function (g);
    a3 = ones (n, 1) * (-1);
    b0 = - (x - 0) .* ((x - 1) .^ 2);
    b1 = b1function (f);
    b2 = b2function (f, g);
    b3 = ones (n, 1);
    c0 = (x - 0) .* ((x - 1) .^ 3);
    c1 = c1function (f);
    c2 = c2function (f, g);
    c3 = ones (n, 1) * 2;

    % definition of the damping coefficients:
    a2 = a2 + tau;
    a3 = a3 + tau * u;
    b2 = b2 + tau;
    b3 = b3 + tau * v;
    c2 = c2 + tau;
    c3 = c3 + tau * w;

    uk = u;
    vk = v;
    wk = w;
    dk = d3;
    ek = e3;

    % computation of u, v, and w:
    a = a0; b = a1; c = a2; d = a3;
    u = DGLfunction (h, n, a, b, c, d);
    a = b0; b = b1; c = b2; d = b3;
    v = DGLfunction (h, n, a, b, c, d);
    a = c0; b = c1; c = c2; d = c3;
    w = DGLfunction (h, n, a, b, c, d);

    % computation of f und g:
    d3 = d3function (u, v, w);
    e3 = e3function (u, v, w);
    for k = 1:n
        B = [d1(k) d2(k); e1(k) e2(k)];
        o = B\[d3(k); e3(k)];
```

```
        f(k) = o(1);
        g(k) = o(2);
        e(k) = norm (B * [f(k); g(k)] − [dk(k); ek(k)]);
    end

    % computation of the norms:
    l(j) = norm (u − uk);
    m(j) = norm (v − vk);
    q(j) = norm (w − wk);
    y(j) = norm (e);

end

% plots of the norms in the number of iterations:
figure (1)
z = 1:p;
hold on
plot (z, l, 'r')
plot (z, m, 'b')
plot (z, q, 'y')
hold off
xlabel ('iterations p')
ylabel ('norms')
title ('‖u(p) − u(p-1)‖, also for v and w')
legend ('u in red', 'v in blue', 'w in yellow', 1)
figure (2)
plot (z, y)
xlabel ('iterations p')
ylabel ('norm')
title ('‖norm (B * [f(i); g(i)](p) − [d3(i); e3(i)](p-1)‖')
```

We need the following functions to use this program. The function **DGLfunction** solves a discretized differential equation.

```
function u = DGLfunction (h, n, a, b, c, d)

A = zeros (n);
u = zeros (n, 1);

r = a/(h^2) − max (b, 0)/h;
s = c − 2 * a/(h^2) − min (b, 0)/h + max (b, 0)/h;
t = a/(h^2) + min (b, 0)/h;
```

```
r = [r(2:n); 0];
t = [0; t(1:n − 1)];

A = spdiags ([r s t], −1:1, n, n);

u = A\d;
```

The other functions needed for the program **Iteration** are given below:

```
function y = a1function (f)
y = f .^ 3 + 2 * f;

function y = a2function (g)
y = g .^ 2;

function y = b1function (f)
y = 3 * f;

function y = b2function (f, g)
y = f .^ 2 + g;

function y = c1function (f)
y = − 2 * f;

function y = c2function (f, g)
y = 2 * f + g;

function y = d3function (u, v, w)
y = u + v + w;

function y = e3function (u, v, w)
y = u − v − w;
```

A.2.2 The Implementation of Chapter 3

The program **IterationGarleanu** solves the problem in paper [1], which is described in Chapter 3.

```
clear all;
close all;

h = 0.01;
n = 1/h + 1;
x = 0:h:1;
x = x';
F = zeros (n, 1);
G = zeros (n, 1);
tau = 10;

% given parameters:
muz = 0.018;
sigmaz = 0.041;
v = 0.1;
pi = 0.01;
chi = 0.018;
gammaA = 3;
gammaB = 18;
psiA = 1;
psiB = 0.95;
rhoA = 0.001;
rhoB = 0.3;
beta1 = 0.99;
beta2 = 0.81;
beta3 = - 0.5;
beta4 = 0.3;
bruchA = (1 - psiA) * (gammaA - 1) / gammaA;
bruchB = (1 - psiB) * (gammaB - 1) / gammaB;

% computation of the other parameters:
[f, df] = ffunction (psiA, psiB, x, beta1, beta2, beta3, beta4);
omega = df;
g = f;
domega = domegafunction (omega, n, h);
ddomega = ddomegafunction (omega, n, h);
dg = dgfunction (g, n, h);
ddg = ddgfunction (g, n, h);
[sigmax, kappa] = skfunction (n, x, g, dg, omega, domega, gammaA, gammaB,
psiA, psiB, sigmaz);
```

```
sigmay = sigmaz + (dg./g) .* sigmax;

% initial values of U, V, and W:
U = x + 0.1;
V = 3 * x + 0.1;
W = 2 * x + 0.05;

% coefficients of the algebraic equations:
d1 = 1 + x .* (dg ./ g) − bruchA * x .* (domega ./ omega);
d2 = − x ./ gammaA;
e1 = (dg ./ g) − x .* bruchA .* (domega ./ omega) − (1 − x) .* bruchB .* (domega
./ omega);
e2 = − x ./ gammaA − (1 − x) ./ gammaB;

% computation of F and G:
D1 = Dfunction (bruchA, −1 / gammaA, muz, sigmaz, omega, domega, ddomega,
sigmax, kappa);
D2 = Dfunction (bruchB, −1 / gammaB, muz, sigmaz, omega, domega, ddomega,
sigmax, kappa);
d3 = x .* (D1 − (pi + rhoA / gammaA) − muz − (dg ./ g) .* sigmaz .* sigmax
− (ddg ./ g) .* ((sigmax .^ 2) / 2)) − (dg ./ g) .* (sigmax .^ 2) − sigmaz * sigmax
+ v * pi * (U ./ V);
e3 = v * pi * (U ./ V) + (1 − v) * pi * (psiB / psiA) * (U ./ W) + x .* (D1 −
(pi + rhoA / gammaA)) + (1 − x) .* (D2 − (pi + rhoB / gammaB)) − muz −
(dg ./ g) .* sigmax .* sigmaz − (sigmax .^ 2) ./ 2 .* (ddg ./ g);

for k = 1:n

    B = [d1(k) d2(k); e1(k) e2(k)];
    o = B\[d3(k); e3(k)];
    F(k) = o(1);
    G(k) = o(2);

end

% computation of muy:
muy = muz + (dg ./ g) .* (F + sigmax .* sigmaz) + (sigmax .^ 2) ./ 2. * ddg ./
g;

% iteration loop:
p = input ('Enter the number of iterations:');
for j = 1:p

    % definitions of the coefficients of the differential equations:
    D3 = Dfunction (bruchA, 1 − 1 / gammaA, muz, sigmaz, omega,
    domega, ddomega, sigmax, kappa) + bruchA * (domega ./ omega) .* F
```

```
− (1 − 1 / gammaA) * G;
D4 = Dfunction (bruchB, 1 − 1 / gammaB, muz, sigmaz, omega,
domega, ddomega, sigmax, kappa) + bruchB * (domega ./ omega) .* F
− (1 − 1 / gammaB) * G;
a0 = (sigmax .^ 2) ./ 2;
a1 = F + sigmax .* (sigmay − kappa);
a2 = muy − G − sigmay .* kappa − pi − chi;
a3 = − psiA * (pi + chi) / pi * (omega ./ g);
b0 = (sigmax .^ 2) ./ 2;
b1 = F + sigmax .* bruchA .* (sigmaz + (domega ./ omega) .* sigmax)
− sigmax .* ((gammaA − 1) / gammaA) .* kappa;
b2 = D3 − (pi + rhoA / gammaA);
b3 = − ones (n, 1);
c0 = (sigmax .^ 2) ./ 2;
c1 = F + sigmax .* bruchB .* (sigmaz + (domega ./ omega) .* sigmax)
− sigmax .* ((gammaB − 1) / gammaB) .* kappa;
c2 = D4 − (pi + rhoB / gammaB);
c3 = − ones (n, 1);

% definition of the damping coefficients:
a0 = − a0;
a1 = − a1;
a2 = − a2 + tau;
a3 = − a3 + tau * U;
b0 = − b0;
b1 = − b1;
b2 = − b2 + tau;
b3 = − b3 + tau * V;
c0 = − c0;
c1 = − c1;
c2 = − c2 + tau;
c3 = − c3 + tau * W;

uk = U;
vk = V;
wk = W;
dk = d3;
ek = e3;

% computation of U, V, and W:
a = a0; b = a1; c = a2; d = a3;
U = DGLfunction (h, n, a, b, c, d);
a = b0; b = b1; c = b2; d = b3;
V = DGLfunction (h, n, a, b, c, d);
a = c0; b = c1; c = c2; d = c3;
W = DGLfunction (h, n, a, b, c, d);
```

```
    % computation of F and G:
    d3 = x .* (D1 − (pi + rhoA / gammaA) − muz − (dg ./ g) .* sigmaz
    .* sigmax − (ddg ./ g) .* ((sigmax .^ 2)./2)) − (dg ./ g) .*
    (sigmax .^ 2) − sigmaz .* sigmax + v * pi * (U ./ V);
    e3 = v * pi * (U ./ V) + (1 − v) * pi * (psiB / psiA) * (U ./ W) +
    x .* (D1 − (pi + rhoA / gammaA)) + (1 − x) .* (D2 − (pi + rhoB /
    gammaB)) − muz − (dg ./ g) .* sigmax .* sigmaz − (sigmax .^ 2) ./ 2
    .* (ddg ./ g);

    for k = 1:n
        B = [d1(k) d2(k); e1(k) e2(k)];
        o = B\[d3(k); e3(k)];
        F(k) = o(1);
        G(k) = o(2);
        e(k) = norm(B * [F(k); G(k)] − [dk(k); ek(k)]);
    end

    % computation of muy:
    muy = muz + (dg ./ g) .* (F + sigmax .* sigmaz) + (sigmax .^ 2) ./ 2
    .* ddg ./ g;

    % computation of the norms:
    l(j) = norm (U − uk);
    m(j) = norm (V − vk);
    q(j) = norm (W − wk);
    y(j) = norm (e);

end

% plots of the norms in the number of iterations:
figure (1)
z = 1:p;
hold on
plot (z, l, 'r')
plot (z, m, 'b')
plot (z, q, 'y')
hold off
xlabel ('iterations p')
ylabel ('norms')
title ('||U(p) − U(p-1)||, also for V and W')
legend ('U in red', 'V in blue', 'W in yellow', 1)
figure (2)
plot (z, y)
xlabel ('iterations p')
ylabel ('norm')
```

```
title ('||norm(B * [F(i); G(i)](p) − [d3(i); e3(i)](p-1)||')
```

We need the following functions to use the program **IterationGarleanu**. The function **DGLfunction** is already known. The first unknown function is called **ffunction**.

```
function [f, df] = ffunction (psiA, psiB, x, beta1, beta2, beta3, beta4)

% computation of H and alpha with equation (3.42) and (3.43):
alpha = (beta1 − beta2) .* normcdf (beta3 .* (x − beta4)) + beta2;
H = 1 ./ (1 + (((1 − psiA) / psiA) * x + ((1 − psiB) / psiB) *
(1 − x)) ./ alpha);

% solution of the ODE with equation (3.44):
[t, y] = ode45 (@dffunction, [0.001,1], 0.001);

% interpolation of y at the point H:
f = interp1 (t, y, H);
df = dffunction (H, f);
```

For the function **ffunction** it is necessary to apply the following function **dffunction**.

```
function dy = dffunction (t,y)

beta1 = 0.99;
beta2 = 0.81;
beta3 = − 0.5;
beta4 = 0.3;

a = (beta1 − beta2) .* normcdf (beta3 .* (t − beta4)) + beta2;
dy = (a ./ t) .* y;
```

The next functions **dgfunction**, **ddgfunction**, **domegafunction** and **ddomegafunction** compute the derivatives of g and ω.

```
function y = dgfunction (g, n, h)
A = (spdiags ([−ones(n, 1) ones(n, 1)], −1:0, n, n)) ./ h;
y = A * g;

function y = ddgfunction (g, n, h)
A = (spdiags ([ones(n, 1) (−2) * ones(n, 1) ones(n, 1)], −1:1, n, n)) ./ (h^2);
```

```
y = A * g;

function y = domegafunction (omega, n, h)
A = (spdiags ([−ones(n, 1) ones(n, 1)], −1:0, n, n)) ./ h;
y = A * omega;

function y = ddomegafunction (omega, n, h)
A = (spdiags ([ones(n, 1) (−2) * ones(n, 1) ones(n, 1)], −1:1, n, n)) ./ (h^2);
y = A * omega;
```

The function **skfunction** evaluates κ and σ_x with equations (3.14), (3.17) and accordingly (3.45), (3.46).

```
function [sigmax, kappa] = skfunction (n, x, g, dg, omega, domega, gammaA,
gammaB, psiA, psiB, sigmaz)

% coefficients of equations (3.45) and (3.46):
a11 = 1 ./ max (x, 1e − 10) + dg ./ g − ((1 − psiA) * (gammaA − 1) / gammaA)
* domega ./ omega;
a12 = ones (n, 1) * (−1 / gammaA);
f1 = ones (n, 1) * (((1 − psiA) * (gammaA − 1) / gammaA − 1) .* sigmaz);
a21 = dg ./ g − x .* ((1 − psiA) * (gammaA − 1) / gammaA) .* domega ./ omega
− (1 − x) .* ((1 − psiB) * (gammaB − 1) / gammaB) .* domega ./ omega;
a22 = − x ./ gammaA − (1 − x) ./ gammaB;
f2 = (x .* ((1 − psiA) * (gammaA − 1) / gammaA) − 1 + (1 − x) .* ((1 − psiB)
* (gammaB − 1) / gammaB)) .* sigmaz;

for k = 1:n

    B = [a11(k) a12(k); a21(k) a22(k)];
    o = B\[f1(k); f2(k)];
    sigmax(k) = o(1);
    kappa(k) = o(2);

end

sigmax = sigmax′;
kappa = kappa′;
```

The last function for the program **IterationGarleanu** is called **Dfunction**.

function y = Dfunction (aa, bb, muz, sigmaz, omega, domega, ddomega, sigmax, kappa)

y = aa .* (muz + (sigmaz .^ 2) ./ 2 .* (aa − 1) + (domega ./ omega) .* (aa .* sigmaz .* sigmax − kappa .* bb .* sigmax) − bb .* kappa .* sigmaz) + (sigmax .^ 2) ./ 2 .* (aa .* (aa − 1) .* (domega ./ omega) .^ 2 + aa .* (ddomega ./ omega)) − bb .* (kappa .^ 2) ./ 2 .* (1 − bb);

A.2.3 The Implementation of Chapter 4

The program **IterationGala_general** solves a more general problem of paper [2], which is described in Chapter 4.2.

```
clear all;
close all;

Q = 1;
Omega = 1;
I = 10;
J = 20;
h_a = 2 * Q / (I − 1);
h_w = Omega / (J − 1);
a = −Q:h_a:Q;
a = a′;
omega = 0:h_w:Omega;
omega = omega′;
tau = 20;

% given parameters:
gamma = 1;
rho = 0.01;
alpha = 2;
n = 2;
delta = 0.13;
i_hat = 0.12;
f = 0.12;
a_cross = −2.22;
kappa_a = 0.27;
sigma_a = 0.05;
x_cross = 1;
kappa_x = 0.15;
sigma_x = 0.27;
theta = 2 * kappa_x / (sigma_x)^2;
v = 2 * kappa_x * x_cross / (sigma_x)^2;

% initial values for q_c and q_h:
q_c = (Q − abs(a)) ./ Q * ((Omega − omega) ./ Omega)′;
q_h = (Q − abs(a)) ./ Q * ((Omega − omega) ./ Omega)′;

% definition of the other parameter mu_w (a_i,w_j) and c_ij:
mu_w = ones (I, J);
c = 0.5 .* ones (I, J);
```

```
% computation of the initial values of Phi_cross and Phi_hat:
Phi_cross = (c .^ (−gamma)) .* q_c;
Phi_hat = (c .^ (−gamma)) .* q_h;


% iteration loop:
p = input ('Enter the number of iterations:');
for k = 1:p

    % computation of the derivation of Phi:
    for i = 1:I

        mu = mu_w(i, :)';
        Phi1 = Phi_cross(i, :)';
        Phi2 = Phi_hat(i, :)';
        dPhidw_cross = dPhidw_function(J, mu, h_w, Phi1);
        dPhidw_hat = dPhidw_function(J, mu, h_w, Phi2);
        dPhidw_c(i, :) = dPhidw_cross;
        dPhidw_h(i, :) = dPhidw_hat;

    end
    % definition of the coefficients of the differential equations:
    % for −kappa_a (a_cross − a) > 0 backward differencing (for ... < 0
    forward differencing)
    for j = 1:J

        Max(:, j) = max (−kappa_a .* (a_cross − a) ./ h_a, 0);
        Min(:, j) = min (−kappa_a .* (a_cross − a) ./ h_a, 0);
        H(:, j) = exp(a);

    end
    A_cross = rho + ((sigma_a)^2) / (h_a)^2 + Max − Min;
    A_hat = rho + kappa_x + ((sigma_a)^2) / (h_a)^2 + Max − Min;
    B = min (−kappa_a .* (a_cross − a) ./ (h_a), 0) − ((sigma_a)^2) / (2 *
    (h_a)^2);
    C = − max (−kappa_a .* (a_cross − a) ./ (h_a), 0) − ((sigma_a)^2) / (2
    * (h_a)^2);
    F_cross = H;
    F_hat = H;
    F_cross = F_cross + mu_w .* dPhidw_c;
    F_hat = F_hat + mu_w .* dPhidw_h;

    % definition of the damping coefficients:
    A_cross = A_cross + tau;
    A_hat = A_hat + tau;
    F_cross = F_cross + tau .* Phi_cross;
    F_hat = F_hat + tau .* Phi_hat;

    Phi_c_old = Phi_cross;
    Phi_h_old = Phi_hat;
```

```
        q_c_old = q_c;
        q_h_old = q_h;

        % computation of the solutions Phi_hat and Phi_cross:
        B = [0; B(1:I-1)];
        C = [C(2:I); 0];
        for j = 1:J
            A = A_cross (:, j);
            F = F_cross (:, j);
            Phi_c = DGLfunction_Gala (A, B, C, F, I);
            Phi_cross (:, j) = Phi_c;
            A = A_hat (:, j);
            F = F_hat (:, j);
            Phi_h = DGLfunction_Gala (A, B, C, F, I);
            Phi_hat (:, j) = Phi_h;
        end

        % computation of q_c and q_h:
        q_c = (c.^gamma) .* Phi_cross;
        q_h = (c.^gamma) .* Phi_hat;

        % computation of the norms:
        l(k) = norm (Phi_cross - Phi_c_old);
        m(k) = norm (Phi_hat - Phi_h_old);
        y(k) = norm (q_c - q_c_old);
        o(k) = norm (q_h - q_h_old);

end

% plots of the norms in the number of iterations:
figure (1)
z = 1:p;
hold on
plot (z, l, 'r')
plot (z, m, 'b')
hold off
xlabel ('iterations p')
ylabel ('norms')
title ('||Phi(p) - Phi(p-1)||')
legend ('Phi_cross in red', 'Phi_hat in blue', 1)
figure (2)
hold on
plot (z, y, 'r')
plot (z, o, 'b')
hold off
xlabel ('iterations p')
```

```
ylabel ('norms')
title ('‖q(p) − q(p-1)‖')
legend ('q_cross in red', 'q_hat in blue', 1)
```

The program **IterationGala** is described in Chapter 4.3.

```
clear all;
close all;

Q = 1;
Omega = 1;
I = 10;
J = 10;
h_a = 2 * Q / (I − 1);
h_w = Omega / (J − 1);
a = −Q:h_a:Q;
a = a';
omega = 0:h_w:Omega;
omega = omega';
tau = 10;

% given parameters:
gamma = 14;
rho = 0.01;
alpha = 2;
n = 2;
delta = 0.13;
i_hat = 0.12;
f = 0.12;
a_cross = −2.22;
kappa_a = 0.27;
sigma_a = 0.05;
x_cross = 1;
kappa_x = 0.15;
sigma_x = 0.27;
theta = 2 * kappa_x / (sigma_x)^2;
v = 2 * kappa_x * x_cross / (sigma_x)^2;

% initial values for q_c and q_h:
q_c = (Q − abs(a)) ./ Q * ((Omega − omega) ./ Omega)';
q_h = (Q − abs(a)) ./ Q * ((Omega − omega) ./ Omega)';

% computation of the other parameter x_tilde, i, g_ij, mu_w (a_i,w_j), and c_ij:
x_tilde = (1 − x_cross .* (q_c − q_h)) ./ max (q_h, 1e − 10);
```

```
g1 = g_ijfunction (a, omega, n − 1, 1, q_c, q_h, v, theta, x_tilde, I, J);
g2 = g_ijfunction (a, omega, n − 1, 0, q_c, q_h, v, theta, x_tilde, I, J);
g3 = g_ijfunction (a, omega, n, 1, q_c, q_h, v, theta, x_tilde, I, J);
i = i_hat + ((n − 1) / (alpha * n))^(n − 1) .* g1;
for i = 1:I

    G(i, :) = (x_cross − omega);

end
mu_w = (kappa_x + i − i_hat) .* G + (i − i_hat) .* theta^(−1) .*
(g2 ./ max (g1, 1e − 10));
c = exp(a) * omega′ + f − i_hat − ((n − 1) / (alpha * n))^(n − 1) .*
(g1 + ((n − 1) / n) .* g3);
c = (c > 0) .* max (c, 1) + (c <= 0) .* min (c, −1);


% computation of the initial values of Phi_cross and Phi_hat:
Phi_cross = (c .^ (−gamma)) .* q_c;
Phi_hat = (c .^ (−gamma)) .* q_h;


% iteration loop:
p = input (′Enter the number of iterations:′);
for k = 1:p

    % computation of the derivation of Phi:
    for i = 1:I

        mu = mu_w(i, :)′;
        Phi1 = Phi_cross(i, :)′;
        Phi2 = Phi_hat(i, :)′;
        dPhidw_cross = dPhidw_function(J, mu, h_w, Phi1);
        dPhidw_hat = dPhidw_function(J, mu, h_w, Phi2);
        dPhidw_c(i, :) = dPhidw_cross;
        dPhidw_h(i, :) = dPhidw_hat;

    end
    % definition of the coefficients of the differential equations:
    % for −kappa_a (a_cross − a) > 0 backward differencing (for ... < 0
    forward differencing)
    for j = 1:J

        Max(:, j) = max (−kappa_a .* (a_cross − a) ./ h_a, 0);
        Min(:, j) = min (−kappa_a .* (a_cross − a) ./ h_a, 0);
        H(:, j) = exp(a);

    end
    A_cross = rho + (1 − gamma) * (delta − i_hat) + gamma * ((n − 1)
    ./ (alpha * n))^(n − 1) .* g1 + ((sigma_a)^2) / (h_a)^2 + Max − Min;
    A_hat = rho + kappa_x + (1 − gamma) * (delta − i_hat) + gamma *
    ((n − 1) ./ (alpha * n))^(n − 1) .* g1 + ((sigma_a)^2) / (h_a)^2 + Max
    − Min;
```

```
B = min (−kappa_a .* (a_cross − a) ./ (h_a), 0) − ((sigma_a)^2) / (2 *
(h_a)^2);
C = − max (−kappa_a .* (a_cross − a) ./ (h_a), 0) − ((sigma_a)^2) / (2
* (h_a)^2);
F_cross = (H + (x_cross)^(−1) .* (f − i_hat)) .* c.^( − gamma);
F_hat = H .* c.^( − gamma);
F_cross = F_cross + mu_w .* dPhidw_c;
F_hat = F_hat + mu_w .* dPhidw_h;

% definition of the damping coefficients:
A_cross = A_cross + tau;
A_hat = A_hat + tau;
F_cross = F_cross + tau .* Phi_cross;
F_hat = F_hat + tau .* Phi_hat;

Phi_c_old = Phi_cross;
Phi_h_old = Phi_hat;
q_c_old = q_c;
q_h_old = q_h;

% computation of the solutions Phi_hat and Phi_cross:
B = [0; B(1:I-1)];
C = [C(2:I); 0];
for j = 1:J
    A = A_cross (:, j);
    F = F_cross (:, j);
    Phi_c = DGLfunction_Gala (A, B, C, F, I);
    Phi_cross (:, j) = Phi_c;
    A = A_hat (:, j);
    F = F_hat (:, j);
    Phi_h = DGLfunction_Gala (A, B, C, F, I);
    Phi_hat (:, j) = Phi_h;
end

% computation of q_c and q_h:
q_c = (c.^gamma) .* Phi_cross;
q_h = (c.^gamma) .* Phi_hat;

% computation of the norms:
l(k) = norm (Phi_cross − Phi_c_old);
m(k) = norm (Phi_hat − Phi_h_old);
y(k) = norm (q_c − q_c_old);
o(k) = norm (q_h − q_h_old);

% computation of the parameters for the next iteration:
x_tilde = (1 − x_cross .* (q_c − q_h)) ./ max (q_h, 1e − 10);
```

```
    g1 = g_ijfunction (a, omega, n − 1, 1, q_c, q_h, v, theta, x_tilde, I, J);
    g2 = g_ijfunction (a, omega, n − 1, 0, q_c, q_h, v, theta, x_tilde, I, J);
    g3 = g_ijfunction (a, omega, n, 1, q_c, q_h, v, theta, x_tilde, I, J);
    i = i_hat + ((n − 1) / (alpha * n))^(n − 1) .* g1;
    for i = 1:I
        G(i, :) = (x_cross − omega);
    end
    mu_w = (kappa_x + i − i_hat) .* G + (i − i_hat) .* theta^(−1) .*
    (g2 ./ max (g1, 1e − 10));
    c = exp(a) * omega′ + f − i_hat − ((n − 1) / (alpha * n))^(n − 1) .*
    (g1 + ((n − 1) / n) .* g3);
    c = (c > 0) .* max (c, 1) + (c <= 0) .* min (c, −1);
end

% plots of the norms in the number of iterations:
figure (1)
z = 1:p;
hold on
plot (z, l, 'r')
plot (z, m, 'b')
hold off
xlabel ('iterations p')
ylabel ('norms')
title ('‖Phi(p) − Phi(p-1)‖')
legend ('Phi_cross in red', 'Phi_hat in blue', 1)
figure (2)
hold on
plot (z, y, 'r')
plot (z, o, 'b')
hold off
xlabel ('iterations p')
ylabel ('norms')
title ('‖q(p) − q(p-1)‖')
legend ('q_cross in red', 'q_hat in blue', 1)
```

We need the function **g_ijfunction** to use the programs **IterationGala_general** and **IterationGala**.

```
function g = g_ijfunction (a, omega, m1, m2, q_c, q_h, v, theta, x_tilde, I, J)

sum = 0;
for k = 0:m1
```

```
        GammaU = gamma(k + v) .* (1 − gammainc (theta .* x_tilde, k + v));
        sum = sum + (gamma (m1 + 1) .* GammaU) ./ (gamma (m1 + 1
        − k) .* gamma (k + m2) .* gamma (v)) .* (−x_tilde).^(m1 − k) .*
        theta^(−k);

    end

    g = (q_h.^m1) .* sum;
```

Moreover, the function **dPhidw_function** is given by:

```
    function y = dPhidw_function (J, mu, h_w, Phi)

    A = zeros (J);
    y = zeros (J, 1);

    r = − max (sign (−mu), 0) .* 1/h_w;
    s = (max (sign (−mu), 0) + min (sign (−mu), 0)) .* 1/h_w;
    t = − min( sign (−mu), 0) .* 1/h_w;

    A = spdiags ([r s t], −1:1, J, J);

    y = A * Phi;
```

Furthermore, we need the function **DGLfunction_Gala** to compute the solutions Phi_cross and Phi_hat:

```
    function u = DGLfunction_Gala (A, B, C, F, I)

    M = zeros (I);
    u = zeros (I, 1);

    M = spdiags ([C A B], −1:1, I, I);

    u = M\F;
```

A.3 Acknowledgment

In this section I want to thank all the people who have supported me during my studies.

Special thanks go to Professor Dr. Martin Burger for the supervision of this diploma thesis. I also thank Professor Dr. Nicole Branger for providing me with the economic background information.

Finally, I want to thank my family because they have always assisted me in my studies.

Bibliography

[1] Gârleanu, N. and Panageas, S., *Young, old, conservative and bold: The implications of heterogeneity and finite lives for asset pricing*, working paper, University of Pennsylvania, June 2007

[2] Gala, Vito D., *Investment and returns*, working paper, University of Chicago, January 28, 2006

[3] Burger, Martin, *Numerik partieller Differentialgleichungen*, Westfälische Wilhelms-Universität Münster, WS 2006/07

[4] Evans, Lawrence C., *Partial Differential Equations*, American Mathematical Society Providence, Rhode Island, 1998

[5] Protter, Murray H. and Weinberger, Hans F., *Maximum Principles in Differential Equations*, Springer-Verlag New York, 1984

[6] Walter, Wolfgang, *Analysis 1*, Springer-Verlag Berlin, 2004

[7] Blanchard, O. J., *Dept, Deficits, and Finite Horizons*, Journal of Political Economy 93(2), 1985

[8] Campbell, J. Y., and Cochrane, J. H., *By Force of Habit: A Consumption-Based Explanation of Aggregate Stock Market Behavior*, Journal of Political Economy 107(2), 1999

[9] Arnold, Ludwig, *Stochastische Differentialgleichungen*, R. Oldenburg Verlag München, 1973

[10] Gomes, Joao F., Kogan, Leonid, and Zhang, Lu, *Equilibrium Cross Section of Returns*, Journal of Political Economy 111, 2003

I affirm with my signature that I have written this diploma thesis independently and that I have not used any other utilities than those declared.

Weisweiler, Anna Münster, May 8, 2008