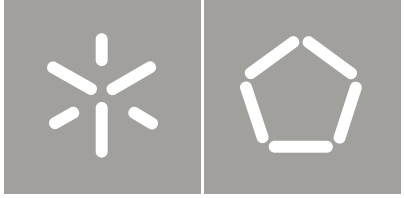
Universidade do Minho
Escola de Engenharia

Marco António da Silva Esteves

Simulação de um Mercado de Agentes Racionais numa perspectiva de Inteligência Colectiva

Outubro de 2010

Simulação de um Mercado de Agentes Racionais
numa perspectiva de Inteligência Colectiva
Marco António da Silva Esteves
UMinho | 2010



Universidade do Minho
Escola de Engenharia

Marco António da Silva Esteves

Simulação de um Mercado de Agentes Racionais numa perspectiva de Inteligência Colectiva

Tese de Mestrado
Ciclo de Estudos Integrados Conducentes ao
Grau de Mestre em Engenharia Electrónica Industrial e Computadores

Trabalho efectuado sob a orientação do
Professor Paulo Garrido

Outubro de 2010

Agradecimentos

Em primeiro lugar agradeço ao Professor Paulo Garrido. Este trabalho não poderia existir sem o suporte teórico e prático, orientações e ensinamentos sobre o que é aprender.

Agradeço ao Professor Renato Morgado, pelo apoio e tutoria ao longo do meu percurso académico. Para mim, será sempre o exemplo máximo dos ensinamentos de Confúcio: "A essência do saber está em, tendo-o, aplicá-lo."

Agradeço a esta instituição, Universidade do Minho, por todas as oportunidades que me foram concedidas. Aos funcionários competentes que humanizam a academia servindo de elo de ligação com os alunos.

Não posso deixar de agradecer aos meus pais, à minha família, à Joana, aos amigos que são família, por todo o suporte, amor e carinho.

Resumo

Neste trabalho foi implementado um modelo utilizando o paradigma de programação multi-agente. O modelo multi-agente que será implementado é o proposto por Garrido (2010c). Estabeleceram-se dois critérios com o objectivo de avaliar o comportamento dos agentes no contexto da sociedade colectiva artificial. O primeiro critério verifica se todos os agentes possuem um nível de riqueza igual ou superior a um valor mínimo de subsistência. O segundo critério verifica se o somatório total da riqueza cresce ao longo do tempo. Neste contexto entende-se por riqueza a quantidade de produtos de consumo na posse dos agentes.

Os objectivos deste trabalho passam por averiguar em que medida são cumpridos os critérios de avaliação comportamental dos agentes no modelo com a funcionalidade racionalidade acrescida.

O algoritmo escolhido para dotar os agentes de racionalidade é o *single Q* do *Reinforcement Learning*. Escolheu-se um algoritmo de *Reinforcement Learning* porque o *Reinforcement Learning* é conhecido por aproximar a aprendizagem humana à computacional.

Estudou-se o modelo desprovido de racionalidade e com racionalidade. No que diz respeito à racionalidade averiguou-se a influência da exploração do ambiente, o modo como os agentes adquirem a informação e o modo como descontam o seu futuro.

Conclui-se que o algoritmo *single Q-learning* não garante o cumprimento dos critérios.

Abstract

In this work one describes the implementation of a model based on a multi-agent system (MAS). The multi-agent model that will be implemented was proposed by Garrido (2010c). The implemented model has two essential criteria of agent's behaviour: — The system has the basic objective of making available to its members a level of wealth exceeding a subsistence minimum. It has also the objective of increasing the collective wealth over the time.

The main objective of this work was endow agents with rationality to the agents and investigate to what extent this influences the satisfaction of behaviour's criteria of the model.

The algorithm chosen to endow the agents with rationality is the single Q of Reinforcement Learning. Reinforcement Learning algorithm was chosen because Reinforcement Learning is known to approximate human learning.

Two platforms were used to implement the model: — LSD and ScicosLab. Both were evaluated and ScicosLab was preferred. The model devoid of rationality and the model with rationality were studied. With regard to rationality, the influence of the exploitation of the environment and the influence of learning rate and discount factor of agents, were tested.

We conclude that the Reinforcement Learning Q-learning algorithm does not guarantee compliance with the criteria.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Acrónimos

DP	-	*Dynamic Programming*
GIMP	-	*GNU Image Manipulation Program*
GPL	-	*Gnu Public License*
GTK	-	*GIMP Toolkit*
INRIA	-	*Institut National de Recherche en Informatique et en Automatique*
LSD	-	*Laboratory for Simulation Development*
LMM	-	*Lsd Model Manager*
MAS	-	*Multi-agent System*
MARL	-	*Multi-agent Reinforcement Learning*
MCM	-	*Monte Carlo Method*
MDP	-	*Markov Decision Process*
RL	-	*Reinforcement Learning*
TD	-	*Temporal Difference*

Capítulo 1

Introdução

Neste projecto será implementado um modelo utilizando o paradigma de programação multi-agente. O modelo multi-agente que será implementado é o proposto por Garrido (2010c) em "Modeling and Simulation of Markets upon a Collective Intelligence Perspective". Este modelo não pressupõe dotar os agentes da capacidade de tomar decisões. Os agentes cumprem as regras estabelecidas seguindo as premissas impostas pelo modelo.

No referido modelo, existem dois tipos de produtos: produtos de consumo e produtos de equipamento. Por conseguinte, existem agentes produtores de bens de consumo e agentes produtores de bens de equipamento. De salientar que apesar dos agentes se subdividirem em duas classes distintas de produtores, todos os agentes são consumidores.

Estabeleceram-se dois critérios com o objectivo de avaliar o comportamento dos agentes no contexto da sociedade colectiva artificial. O primeiro critério verifica se todos os agentes possuem um nível de riqueza igual ou superior a um valor mínimo de subsistência. O segundo critério verifica se o somatório total da riqueza cresce ao longo do tempo. Neste contexto entende-se riqueza pela quantidade de produtos de consumo na posse dos agentes. Se se cumprirem os dois critérios em simultâneo considera-se que o sistema está dotado de Inteligência

Colectiva. Por conseguinte, os critérios mencionados podem ser encarados como objectivos a cumprir pelos agentes para alcançarem Inteligência Colectiva.

Partindo do modelo supramencionado adicionou-se racionalidade aos agentes. Entenda-se racionalidade como a capacidade de tomar decisões/opções. Escolheu-se o algoritmo *single Q-learning* para dotar os agentes de racionalidade.

Os objectivos deste trabalho passam por averiguar em que medida são cumpridos os critérios para a avaliação comportamental dos agentes no modelo com a funcionalidade racionalidade acrescida. Por conseguinte, implica avaliar a existência de Inteligência Colectiva.

Este trabalho está subdivido em 6 capítulos, intitulados respectivamente, "Introdução", "Descrição do modelo", "Estado da arte", "Implementação", "Análise de Resultados", "Conclusões e propostas de trabalho futuro".

Capítulo 2

Descrição do Modelo

Neste capítulo vai-se descrever o modelo proposto por Garrido (2010c) em "Modeling and Simulation of Markets upon a Collective Intelligence Perspective".

No referido modelo, existem dois tipos de produtos: produtos de consumo e produtos de equipamento. Por conseguinte, existem agentes produtores de bens de consumo e agentes produtores de bens de equipamento. De salientar que apesar dos agentes se subdividirem em duas classes distintas de produtores, todos os agentes são consumidores.

2.1 Descrição do modelo em estudo

No modelo supracitado existem dois tipos de produtos: produtos de consumo, pc e produtos de equipamento pe. Por conseguinte, existem agentes produtores de bens de consumo e agentes produtores de bens de equipamento, indexados respectivamente por jc e je. De salientar que apesar dos agentes se subdividirem em duas classes distintas de produtores, todos os agentes são consumidores.

Estabeleceram-se dois critérios com o objectivo de avaliar o comportamento dos agentes no contexto da sociedade colectiva artificial.

O primeiro critério verifica se todos os agentes possuem um nível de riqueza igual ou superior a um valor mínimo de subsistência. Tal pode ser descrito por:

$$\forall j, w_j \geq w_{\min} \tag{2.1}$$

O segundo critério verifica se o somatório total da riqueza cresce ao longo do tempo. O rácio do incremento da riqueza, ΔW, ao correspondente intervalo de tempo, Δt, é positivo:

$$\forall \Delta t, \frac{\Delta W}{\Delta t} > 0 \tag{2.2}$$

A cada instante de tempo discreto, k, de acordo com o seu tipo, é suposto um agente produzir e vender uma determinada quantidade de produto de equipamento ou de consumo : *sprode* ou *sprodc*.

$$\begin{aligned} sprodc(j_c, k) &= K_{c,j_c} \cdot pe(j_c, k-1) \\ sprode(j_e, k) &= K_{e,j_e} \cdot pe(j_e, k-1) \end{aligned} \tag{2.3}$$

Nas equações 2.3 $pe(j_c, k-1)$ corresponde à quantidade de equipamento na posse de um agente produtor de bens de consumo no instante $k-1$ e $pe(j_e, k-1)$ corresponde à quantidade de equipamento na posse de um agente produtor de bens de equipamento no instante $k-1$; K_{c,j_c} e K_{e,j_e} são os parâmetros que expressam a produtividade específica por agente, de acordo com o seu tipo. Os produtos de equipamento são necessários à produção, tanto de bens de equipamento como bens de consumo.

Os valores totais de produtos vendidos no instante k, no caso de consumo $tot\_sprodc(k)$ e equipamento $tot\_sprode(k)$, determinam-se por:

$$\begin{aligned} tot\_sprodc(k) &= \sum_{j_c=1}^{J_c} sprodc(j_c, k) \\ tot\_sprode(k) &= \sum_{j_e=1}^{J_e} sprode(j_e, k) \end{aligned} \tag{2.4}$$

Na equação 2.4, J_c e J_e correspondem respectivamente ao número máximo de agentes de cada tipo. A sua soma é J.

Os preços unitários de produtos de consumo, *upc*, e de produtos de equipamento, *upe*, determinam-se por:

$$\begin{aligned} upc(k) &= \frac{tot\_imppc(k-1)}{tot\_sprodc(k)} = \frac{\sum_{j=1}^{J} imppc(j,k-1)}{\sum_{j_c=1}^{J_c} sprodc(j_c,k)} \\ upe(k) &= \frac{tot\_imppe(k-1)}{tot\_sprode(k)} = \frac{\sum_{j=1}^{J} imppe(j,k-1)}{\sum_{j_e=1}^{J_e} sprode(j_e,k)} \end{aligned} \tag{2.5}$$

Na equação 2.5, *imppc* e *imppe* correspondem às quantidades de dinheiro a gastar por um agente j na compra de produtos de consumo e equipamento, respectivamente, no período de tempo que vai desde o instante $k-1$ até a k. Por sua vez, *tot_imppc* e *tot_imppe* correspondem ao total de dinheiro despendido pela sociedade no período de $k-1$ até k, e expressam-se:

$$\begin{aligned} tot\_imppc(k-1) &= \sum_{j=1}^{J} imppc(j,k-1) \\ tot\_imppe(k-1) &= \sum_{j=1}^{J} imppe(j,k-1) \end{aligned} \tag{2.6}$$

Os valores de *imppc* e *imppe* são determinados pelas preferências específicas dos diferentes agentes, $mfpc$ para bens de consumo, $mfpe$ para bens de equipamento.

$$\begin{aligned} imppc(j,k-1) &= mfpc(j,k-1) \cdot m(j,k-1) \\ imppe(j,k-1) &= mfpe(j,k-1) \cdot m(j,k-1) \end{aligned} \tag{2.7}$$

A variável $m(j,k-1)$ corresponde à quantidade de dinheiro na posse do agente j desde o instante $k-1$ até ao instante k.

Em troca do dinheiro gasto, os agentes recebem produtos de equipamento ou de consumo. Então, os produtos comprados de consumo designam-se por *ppc* e

os de equipamento por *ppe*.

$$\begin{aligned} ppc(j,k) &= imppc(j,k-1)/upc(k) \\ ppe(j,k) &= imppe(j,k-1)/upe(k) \end{aligned} \tag{2.8}$$

A cada instante k a quantidade de produtos de equipamento pe e de produtos de consumo pc na posse de um determinado agente é:

$$\begin{aligned} pc(j,k) &= \lambda_c \cdot pc(j,k-1) + ppc(j,k) \\ pe(j,k) &= \lambda_e \cdot pe(j,k-1) + ppe(j,k) \end{aligned} \tag{2.9}$$

De salientar que ambos os tipos de produto decaem em valor ao longo do tempo; porém, a velocidades diferentes.

Na equação 2.9 as quantidades λ_c e λ_e são definidas por:

$$\begin{aligned} \lambda_c &= 1 - \gamma_c \\ \lambda_e &= 1 - \gamma_e \end{aligned} \tag{2.10}$$

Em que γ_c corresponde à taxa de decaimento para os produtos de consumo e γ_e à taxa de decaimento para produtos de equipamento.

No que se segue é usada a restrição de:

$$mfpc(j,k) + mfpe(j,k) = 1. \tag{2.11}$$

A quantidade de dinheiro na posse de um dado agente j no instante k depende do tipo de produto que o agente produz e pode expressar-se do seguinte modo:

$$\begin{aligned} m(j_c,k) &= m(j_c,k-1) - imppc(j_c,k-1) - imppe(j_c,k-1) + sprodc(j_c,k) \cdot upc(k) \\ m(j_e,k) &= m(j_e,k-1) - imppc(j_e,k-1) - imppe(j_e,k-1) + sprode(j_e,k) \cdot upe(k) \end{aligned} \tag{2.12}$$

2.1.1 Determinação do padrão de qualidade de vida

Nesta sociedade pode estabelecer-se o conceito de qualidade de vida através da imposição de padrões para os agentes. Neste contexto, um produto de consumo é alguma coisa que um agente tem que possuir numa quantidade mínima para se manter vivo e se possível "feliz". Por conseguinte, assume-se uma quantidade mínima de produtos de consumo que um dado agente tem que possuir para se manter vivo, *minpc*.

Assume-se que está disponível para um dado agente adquirir a cada instante a quantidade mínima de produto de consumo para se manter vivo. Aliás, é objectivo de toda a sociedade (equação 2.2) realizar um valor mínimo de riqueza para cada membro pertencente ao sistema. A condição para que isso seja possível, numa distribuição equitativa, é que a soma total de produtos de bens de consumo, *tot_sprodc*, produzidos e vendidos em qualquer intervalo k pelos produtores de bens de consumo (equação 2.4) deve ser maior ou igual ao número total de agentes J a multiplicar por *minpc*:

$$\begin{aligned} &\left(tot\_sprodc(k) = \sum_{j_c=1}^{J_c} sprodc(j_c, k) \right) \geq J \cdot minpc \\ &\Leftrightarrow tot\_sprodc(k) \geq J \cdot minpc \end{aligned} \tag{2.13}$$

É possível definir-se um *standard* de vida colectivo médio para um intervalo k fazendo o quociente entre o valor total de produtos de consumo vendidos pelo valor mínimo para a sobrevivência de todos os agentes:

$$S(k) = \frac{tot\_sprodc(k)}{J \cdot minpc} \tag{2.14}$$

Com base na equação 2.14 pode determinar-se um *standard* de qualidade de vida para um agente:

$$s(k) = \frac{pc(j, k)}{minpc} \tag{2.15}$$

2.1.2 Definição das condições de fronteira para a produção

Proporcionar um montante total de bens de consumo, *tot_sprodc*, obriga a impor valores mínimos totais de equipamentos e valores médios mínimos de produtividade específica para cada tipo de agente produtor.

Estas condições devem ser satisfeitas para qualquer instante de tempo k, pelo que pode omitir-se a utilização do índice k. Da equação 2.4 e 2.3 têm-se que:

$$tot\_sprodc = \sum_{j_c=1}^{J_c} sprodc(j_c) = \sum_{j_c=1}^{J_c} K_{c,j_c} \cdot pe(j_c) \tag{2.16}$$

Definindo-se *tot_pec* como sendo o somatório do equipamento na posse dos produtores de bens de consumo:

$$tot\_pec = \sum_{j_c=1}^{J_c} pe(j_c) \tag{2.17}$$

$\bar{K}_c$ é a produtividade média dos produtores de bens de consumo:

$$\bar{K}_c = \frac{tot\_sprodc}{tot\_pec} = \frac{\sum_{j_c=1}^{J_c} K_{c,j} \cdot pe(j_c)}{\sum_{j_c=1}^{J_c} pe(j_c)} \tag{2.18}$$

Daqui resulta que:

$$tot\_sprodc = \bar{K}_c \cdot tot\_pec \tag{2.19}$$

Utilizando as equações 2.19 e 2.14 obtém-se a condição necessária para sustentar o padrão médio de vida dos agentes:

$$\bar{K}_c \cdot tot\_pec \geq S \cdot J \cdot minpc \tag{2.20}$$

Se assumir que $\bar{K}_c$ é constante e que o valor do equipamento decai ao longo do tempo, a relação 2.19 vai impor restrições no somatório do total de produtos de equipamento na posse de produtores de equipamento, *tot_pee*; bem como em

$\bar{K}_e$, a sua produtividade específica. As equações obtêm-se de forma similar às supramencionadas, equações 2.17 e 2.18, podendo definir *tot_sprode* como:

$$tot\_sprode = \bar{K}_e \cdot tot\_pee \tag{2.21}$$

A seguinte relação deve garantir que o equipamento total, $tot\_pe = tot\_pec + tot\_pee$ cresce ou pelo menos não decresce:

$$tot\_sprode \geq \gamma_e \cdot tot\_pe \tag{2.22}$$

De realçar que esta condição, equação 2.22, está relacionada com um dos objectivos da sociedade (equação 2.2). O que significa:

$$\bar{K}_e \cdot tot\_pee \geq \gamma_e \cdot (tot\_pec + tot\_pee) \tag{2.23}$$

A partir da análise destas equações consegue perceber-se de que forma é que pode ser mantida a sustentabilidade de um determinado nível padronizado de vida S.

Para cada S, *minpc*, λ_e e quantidade inicial de produtos de equipamento, *pe_init*, existe um valor de $\bar{K}_c$ e de $\bar{K}_e$ que garante a produtividade mínima para a sustentabilidade da sociedade.

2.1.3 Análise dinâmica

A análise dinâmica será descrita com base no no *working paper* "Dynamics of a two class multi-agent model" de Garrido (2010b). Tem como objectivo determinar estocasticamente o ponto fixo do modelo.

É importante determinar o sinal da variação de uma variável de estado, x_i, desde o instante k até ao instante $k+1$. A variação de x_i traduz-se na seguinte equação:

$$\Delta x_i(k) = x(k+1) - x(k) \tag{2.24}$$

Se $\Delta x_i = 0$ está-se perante um ponto fixo. Utilizando uma sintaxe pouco ortodoxa escreve-se:

$$\Delta x_i(k) >=< 0 \Leftrightarrow expressao1 >=< expressao2 \tag{2.25}$$

De forma a abreviar uma escrita mais extensa de:

$$\begin{aligned} &\Delta x_i(k) > 0 \Leftrightarrow expressao1 > expressao2 \\ &\Delta x_i(k) = 0 \Leftrightarrow expressao1 = expressao2 \\ &\Delta x_i(k) < 0 \Leftrightarrow expressao1 < expressao2 \end{aligned} \tag{2.26}$$

Para se estabelecer as relações simbolizadas pela equação 2.25 explorar-se-á as variáveis de estado do modelo inicializadas com valores não negativos; bem como os parâmetros e variáveis de decisão restritos a valores não negativos. Sob esta assumpção, as variáveis de estado do modelo terão sempre valores não negativos. Portanto, explora-se o facto de:

$$\forall x_i \forall k, x_i(k) \geq 0 \Rightarrow \left(\Delta x_i(k) >=< 0 \Leftrightarrow \frac{x_i(k+1)}{x_i(k)} >=< 1 \right) \tag{2.27}$$

Os produtos de equipamento desempenham um papel preponderante na dinâmica da sociedade, dado que são essenciais para a produção de bens de equipamento e de consumo. Nessa medida é interessante definir uma quantidade denominada factor de investimento dos agentes produtores de bens de equipamento porque estes impulsionam todo o funcionamento da sociedade.

Define-se um factor de investimento em produtos de equipamento, ι_e:

$$\iota_e(k) = \frac{tot\_imppe(je, k)}{tot\_imppe(jc, k) + tot\_immpe(je, k)} \tag{2.28}$$

É possível simplificar a equação 2.28 do seguinte modo:

$$\iota_e(k) = \frac{tot\_imppe\_e(k)}{tot\_imppe(k)} \tag{2.29}$$

O valor do factor de investimento de produtos de equipamento é crítico para a dinâmica da sociedade:

$$
\begin{aligned}
tot\_pee(k+1) &= \sum_{j_e=1}^{J_e} pe(j_e, k+1) \\
tot\_pee(k+1) &= \sum_{j_e=1}^{J_e} \left[\lambda_e \cdot pe(j_e, k) + ppe(j_e, k)\right] \\
tot\_pee(k+1) &= \lambda_e \cdot tot\_pee(k) + \frac{tot\_imppe\_e(k)}{upe(k)} \\
tot\_pee(k+1) &= \lambda_e \cdot tot\_pee(k) + \frac{tot\_imppe\_e(k)}{\frac{tot\_imppe(k)}{tot\_sprode\_e(k)}} \\
tot\_pee(k+1) &= \lambda_e \cdot tot\_pee(k) + \frac{tot\_imppe\_e(k) \cdot tot\_sprode(k)}{tot\_imppe(k)} \\
tot\_pee(k+1) &= \lambda_e \cdot tot\_pee(k) + \frac{tot\_imppe\_e(k) \cdot \bar{K}_e \cdot tot\_pe\_e(k)}{tot\_imppe(k)} \\
tot\_pee(k+1) &= \lambda_e \cdot tot\_pee(k) + \bar{K}_e \cdot tot\_pe\_e(k) \left(\frac{tot\_imppe\_e(k)}{tot\_imppe(k)}\right) \\
tot\_pee(k+1) &= tot\_pee(k) \cdot \left[\lambda_e + \bar{K}_e \left(\frac{tot\_imppe\_e(k)}{tot\_imppe(k)}\right)\right] \\
\frac{tot\_pee(k+1)}{tot\_pee(k)} &= \lambda_e + \bar{K}_e \cdot \frac{tot\_imppe\_e(k)}{tot\_imppe(k)}
\end{aligned}
\tag{2.30}
$$

Portanto:

$$\Delta tot\_pee(k) >=< 0 \Leftrightarrow \lambda_e + \bar{K}_e \iota_e(k) >=< 1 \tag{2.31}$$

$$\Delta tot\_pee(k) >=< 0 \Leftrightarrow \iota_e(k) >=< \frac{\gamma_e}{\bar{K}_e} \tag{2.32}$$

Utilizando a relação estabelecida na equação 2.32 consegue perceber-se como é que o factor de investimento, ι_e varia em relação ao ponto fixo de pe_e, γ_e/K_e.

Capítulo 3

Estado da Arte

Neste capítulo abordar-se-ão conceitos importantes para a contextualização temática deste trabalho que permitirão uma melhor compreensão do mesmo.

3.1 Do agente ao multi-agente

O conceito de agente não é consensual.

A maioria dos autores não concebem uma definição precisa e/ou suficientemente abrangente que alcance todas as disciplinas que utilizam este paradigma. A constatação desta realidade é compreensível dada a quantidade e diversidade de áreas e tecnologias que utilizam agentes. Alguns exemplos de aplicabilidade são: ciências cognitivas, engenharia, economia.

No caso em estudo pretende-se que os agentes sejam autónomos e inteligentes.

3.1.1 O agente

De uma forma geral, consideram-se seres humanos e restantes animais como agentes. Engenheiros e cientistas constroem robôs, sistemas, programas e simulações que também são considerados agentes. Então, "em que difere um agente

de outro sistema artificial qualquer?" (Florian, 2003)

Steels and Brooks (1995) consideram que a essência da teoria de agentes reside em "o agente poder controlar, numa determinada extensão, o seu próprio destino".

O agente necessita de mecanismos que lhe permitam "sentir" o ambiente em que está inserido para poder agir em conformidade com o que foi projectado no modelo.

3.1.2 Autonomia

Para os agentes autónomos "a base do seu auto-governo tem origem (pelo menos em parte) na própria capacidade do agente em formar e adaptar os seus princípios de comportamento" (Steels and Brooks, 1995). A autonomia não se pode quantificar, uma vez que não se trata de uma propriedade absoluta.

Outros autores consideram que o agente é implicitamente autónomo. Entre esses autores estão Franklin and Graesser (1997), que procuram definir agente autónomo como "um sistema localizado no ambiente e é, em simultâneo, parte do ambiente enquanto entidade que "sente" e que actua no ambiente, ao longo do tempo. E procura alcançar a sua própria agenda para realizar o que "sente", no futuro".

Para Maes (1995), os agentes podem ser vistos como "sistemas computacionais que habitam num ambiente dinâmico (algo complexo), sentem e actuam autonomamente nesse ambiente e assim realizam uma série de tarefas e/ou objectivos para os quais foram projectados".

3.1.3 Inteligência

A questão da inteligência dos agentes não é consensual. Alguns autores mais deterministas, consideram que a inteligência reside na aprendizagem a partir da experiência. Outros relacionam-na com pensamentos abstractos, *cogito ergo sum*

(penso, logo existo)?

Para Rosenschein (1999), na "MIT Encyclopedia of Cognitive Scince", "um agente inteligente é um dispositivo que interage com o ambiente numa forma flexível e direccionada ao objectivo, reconhecendo os estados importantes do ambiente e actuando de forma a alcançar os resultados desejados".

Dada a falta de consenso entre autores, optar-se-á por se assumir inteligência como a capacidade de maximizar um ou mais critérios através da escolha de opções, que originam decisões, sejam estas resultado de aprendizagem ou de condicionantes impostas pelo próprio modelo (Garrido, 2010a).

Ao longo deste trabalho sempre que for mencionado "o agente" estar-se-á a referir, implicitamente, a um agente que é autónomo, posicionado, corporificado e inteligente.

3.1.4 A implementação do agente

Sob o ponto de vista do programador pode considerar-se, de uma forma simplista, que o agente é um excerto de código que adiciona uma ou mais funcionalidades específicas. Todavia, para quem realiza a implementação de um modelo com agentes são necessários cuidados específicos.

Luck et al. (1997) propõem uma divisão hierárquica de entidades num programa.

> "As entidades podem ser utilizadas para agrupar atributos para qualquer propósito sem lhes ser adicionada uma camada de "funcionalidade". Servem como um mecanismo de abstracção útil porque estas funcionalidades são consideradas distintas do restante ambiente e podem organizar a percepção. Um objecto é somente algo com atributos e habilidades, mas não define características. Um agente é um objecto que é útil, geralmente, a outro agente onde a sua utilidade é definida ao nível do alcance dos objectivos do agente." (Luck

et al., 1997)

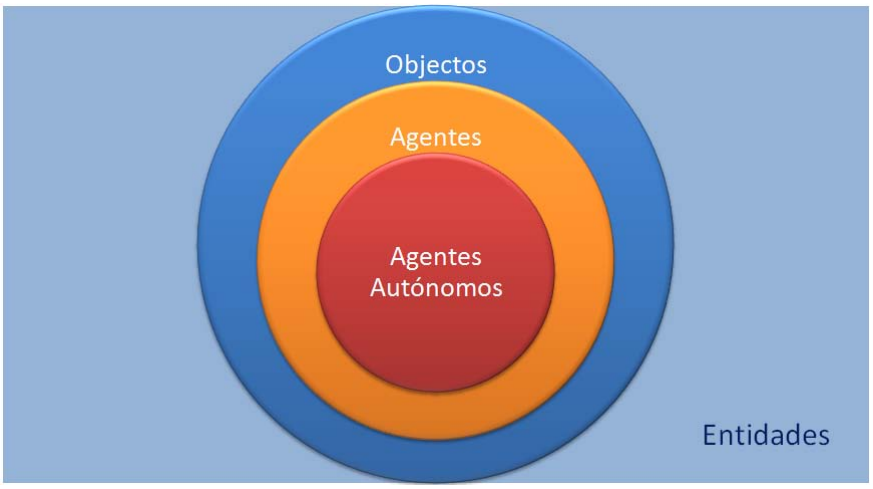


Figura 3.1: Diagrama que ilustra a hierarquia das entidades.

O diagrama 3.1 ilustra uma possível divisão hierárquica que é a defendida por (Luck et al., 1997). Os autores concluem que se existirem atributos agrupados, mas não existirem capacidades, está-se perante uma entidade. Se não existirem objectivos, a entidade é um objecto. Se existirem objectivos, mas não existirem motivações, está-se perante um agente. Se existirem motivações e objectivos, trata-se de um agente autónomo.

3.1.5 O multi-agente

Um sistema multi-agente pode ser definido como uma agregação de múltiplos agentes que partilham o mesmo ambiente, ou partes deste, no qual interagem com o objectivo de atingir objectivos comuns ou individuais.

Dada a sua natureza, um sistema multi-agente proporciona um certo grau de incerteza devido à partilha do ambiente com outros agentes, assim como conhecimento/desconhecimento que possuem relativamente aos restantes agentes.

3.2 Do *Reinforcement Learning* ao *Multi-Agent Reinforcement Learning*

Como foi mencionado anteriormente (secção 3.1.3), assume-se a inteligência do agente como a capacidade de tomar decisões. O problema associado a esta premissa coloca-se em saber como atribuir essa capacidade ao agente.

Escolheu-se o *Reinforcement Learning* (RL) como metodologia de suporte à decisão. Trata-se de um conjunto de métodos e algoritmos que aproximam a aprendizagem computacional à aprendizagem humana.

A abordagem seguida na secção 3.2.1 até 3.2.1.3 é a de Sutton and Barto (1998) na obra "*Reinforcement Learning: An Introduction*".

3.2.1 O *Reinforcement Learning*

Num sistema RL (Figura 3.2) os agentes seleccionam acções às quais o ambiente reage. O ambiente atribui as recompensas, que são valores numéricos cuja soma os agentes procuram maximizar ao longo do tempo.

O sistema é dinâmico. O agente interage com o ambiente numa sequência de tempo discreto $k = 0, 1, 2, 3...$.

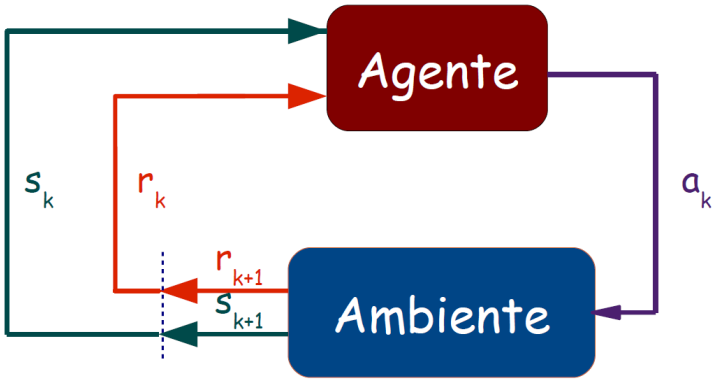


Figura 3.2: Diagrama que representa a dinâmica de um sistema RL .

No instante k (consultar Figura 3.3), os agentes recebem informações do ambiente para estabelecerem em que estado se encontram, $s_k \in S$, em que S é um conjunto de estados possíveis. Baseados no estado, s_k, seleccionam uma acção $a_k \in A(s_k)$, em que $A(s_k)$ é o conjunto de acções possíveis no estado s_k. Em consequência da selecção de uma dada acção recebem uma recompensa r_k.

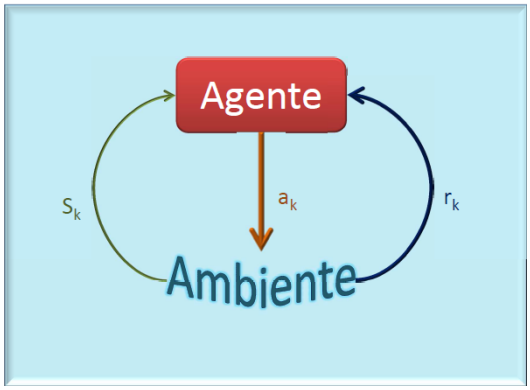


Figura 3.3: Diagrama que ilustra o funcionamento de um sistema RL no instante k.

No instante seguinte (consultar Figura 3.4), $k+1$, em consequência da acção, anteriormente seleccionada, o agente recebe a recompensa r_{k+1}, e entra num dos possíveis estados, s_{k+1}. Note-se que o agente pode manter-se no mesmo estado se essa for uma das acções possíveis.

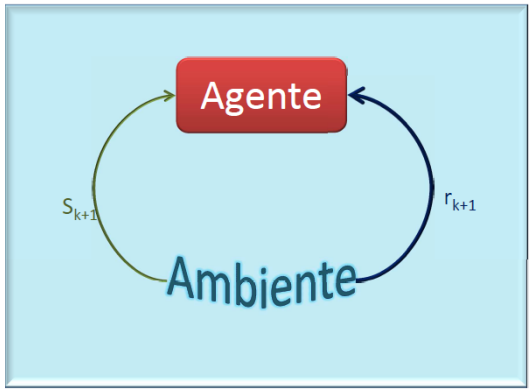


Figura 3.4: Diagrama que ilustra o funcionamento de um sistema RL no instante $k+1$.

Há termos na linguagem corrente do RL que são importantes reter, tais como: recompensa, retorno, desconto e política.

As recompensas são consideradas um número, $r_k \in \Re$. O somatório das recompensas recebidas é designado retorno, R_k, e será este o valor que o agente procurará maximizar. Note-se que maximizar o retorno pode não implicar maximizar a recompensa imediata.

No caso mais simples R_k pode ser definido como soma das recompensas recebidas,

$$R_k = r_{k+1} + r_{k+2} + r_{k+3} + ... + r_K \tag{3.1}$$

em que K é um instante final.

Num caso mais geral, o retorno é definido como a soma descontada das recompensas:

$$R_k = r_{k+1} + \gamma r_{k+2} + \gamma^2 r_{k+3} + ... = \sum_{l=0}^{\infty} \gamma^l r_{k+l+1} \tag{3.2}$$

Em que $\gamma \in [0, 1[$, denomina-se taxa de desconto e é um parâmetro que influencia directamente o valor futuro das recompensas.

A cada instante k, o agente escolhe uma acção $a_k \in A(s_k)$ com probabilidade $\pi_k(s, a)$, sendo esta a probabilidade de $a_k = a$ se $s_k = s$. A distribuição completa de probabilidade $\pi_k(s, a)$ constitui a política do agente. Os métodos de aprendizagem RL especificam como é que os agentes modificam a sua política em função das suas experiências.

O agente implementa um mapeamento dos estados de probabilidades de selecção de cada possível acção. O objectivo do agente é seleccionar acções de modo a que o retorno seja máximo ao longo do tempo.

3.2.1.1 Propriedade de Markov

O ambiente responde no instante $k+1$ a uma acção tomada no instante k. No caso mais geral, a reposta deve depender de tudo que aconteceu anteriormente. Então, neste caso a dinâmica do sistema pode ser definida pela distribuição de probabilidades conjunta:

$$Pr\{s_{k+1} = s', r_{k+1} = r | s_k, a_k, r_k, s_{k-1}, a_{k-1}, ..., r_1, s_0, a_0\} \tag{3.3}$$

$\forall$ s', r, e todos os valores possíveis dos eventos anteriores: s_k, a_k, $r_k..r_1$, s_0, a_0. Em que s' é estado onde o agente se situa após a transição e r a respectiva recompensa.

Por outro lado, se o sistema for *markoviano* a resposta do ambiente em $k+1$ depende apenas do estado e da acção em k. Esta propriedade, chamada de Markov, pode definir-se por:

$$Pr\{s_{k+1} = s', r_{k+1} = r | s_k, a_k\} \tag{3.4}$$

$\forall$ s', r, s_k, e a_k.

Um problema de *Reinforcement Learning* que satisfaz a propriedade de Markov denomina-se processo de decisão de Markov, *Markov Decision Process* (MDP). Se o processo de Markov em estudo for finito, denomina-se processo de decisão de Markov finito.

Um dado processo de decisão finito de Markov pode ser definido pelo seu estado, acções e pela dinâmica associada ao seu ambiente.

A probabilidade, $\mathbf{P}^{\mathbf{a}}_{\mathbf{ss'}}$, associada a uma transição para um qualquer estado seguinte s', seleccionada uma acção a tem a propriedade:

$$\mathbf{P}^{\mathbf{a}}_{\mathbf{ss'}} = Pr\{s_{k+1} = s' | s_k = s, a_k = a\} \tag{3.5}$$

A equação 3.5 define as probabilidades de transição. De uma forma semelhante, com o mesmo estado s e acção a pode determinar-se o valor espectável da

recompensa do estado seguinte. Escreve-se:

$$\mathbf{R}^{\mathbf{a}}_{\mathbf{ss'}} = E\{r_{k+1}|s_k = s, a_k = a, s_{k+1} = s'\} \tag{3.6}$$

De forma simplista, pode afirmar-se que um MDP define um dado sistema RL.

3.2.1.2 Funções de valor e funções de valor óptimas

Funções de valor

Praticamente todos os algoritmos RL baseiam-se na estimativa de funções de valor: — funções de estado ou funções de pares estado-acção. As funções de estado estimam "quão bom" é para um agente estar num dado estado. As funções de pares estado-acção estimam "quão bom" é para um agente escolher uma determinada acção num dado estado. A noção de "quão bom" é definida em termos de retorno obtido.

Relembrando que a política π é o mapeamento para cada estado, $s \in S$, e a acção, $a \in A(s)$. $\pi(s, a)$ corresponde à probabilidade de tomar a acção a num estado s. Para um MDP pode definir-se $V^\pi(s)$:

$$V^\pi(s) = E_\pi\{R_k|s_k = s\} = E_\pi\left\{\sum_{l=0}^{\infty}\gamma^l r_{k+l+1}|s_k = s\right\} \tag{3.7}$$

em que E_π é o valor espectável se for seguida a política π e k representa qualquer instante de tempo. A função V^π denomina-se a função estado-valor para a política π.

De um modo similar, pode definir-se o valor correspondente à escolha de uma acção a num estado s seguindo a política π, denominado-se $Q^\pi(s, a)$, como o retorno espectável começando num estado s, escolhendo a acção a e depois disso seguir a política π:

$$Q^\pi(s, a) = E_\pi\{R_k|s_k = s\} = E_\pi\left\{\sum_{l=0}^{\infty}\gamma^l r_{k+l+1}|s_k = s, a_k = a\right\} \tag{3.8}$$

Q^π denomina-se função acção-valor para a política π.

Funções de valor óptimas

De uma forma simplista pode considerar-se que resolver um problema com RL implica encontrar a política π que permite garantir o maior retorno possível. Para MDPs finitos pode definir-se uma política óptima que garanta o maior retorno possível. Uma política π é definida como sendo melhor ou igual a uma política π' se o retorno espectável for maior ou igual a π' para todos os estados. Isto é:

$$\pi \geq \pi', \forall s' \Leftrightarrow V^{\pi}(s) \geq V^{\pi'}(s) \tag{3.9}$$

Há pelo menos uma política melhor ou igual a qualquer outras políticas: — a política óptima. Havendo mais do que uma política óptima, π^* representa todas as políticas óptimas. As políticas óptimas partilham a mesma função de estado--valor, denominada função óptima estado-valor, representa-se por V^* e define-se como:

$$V^*(s) = \max_{\pi} V^{\pi}(s), \forall s \in S \wedge a \in A(s) \tag{3.10}$$

As políticas óptimas também partilham a mesma função de acção-valor Q^*:

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a), \forall s \in S \wedge a \in A(s) \tag{3.11}$$

Para cada par estado-acção esta função fornece o retorno espectável escolhendo a acção a num dado estado s e seguindo uma política óptima π^*. Por conseguinte, pode escrever-se Q^* em relação a V^*:

$$Q^*(s,a) = E\left\{r_{k+1} + \gamma V^*(s_{k+1}) | s_k = s, a_k = a\right\} \tag{3.12}$$

3.2.1.3 Métodos para implementar RL

Programação Dinâmica, *Dynamic Programming* (DP), é um dos métodos utilizados para implementar sistemas de RL. A DP caracteriza-se por agregar algoritmos de aprendizagem capazes de retornar políticas óptimas, se for perfeitamente conhecido o ambiente envolvente e o seu respectivo MDP.

Os Métodos de Monte Carlo, Monte Carlo Methods (MCMs), ao contrário dos métodos de DP, não necessitam do conhecimento total do ambiente. Acrescentam alguma capacidade de exploração ao agente. Com os MCMs surge o conceito de experiência, o que implica observar uma sequência de estados, acções e recompensas, provenientes de uma experiência real ou de simulação computacional. Contudo, é necessário um modelo para gerir as transições entre acções. Não obstante, este não necessita que seja determinada toda a distribuição de probabilidades de transição, como ocorre nos métodos DP.

Os métodos "*on-policy*" caracterizam-se por tentarem avaliar ou melhorar a política que é utilizada para a tomada de decisões.

Por outro lado, os métodos "*off-policy*" implicam ter duas políticas distintas: a política que gera o comportamento, política comportamental, e a política estimada.

A premissa anterior constitui uma vantagem considerável dado que a política estimada pode ser determinística, enquanto a política comportamental contempla todas as acções possíveis num dado estado.

Um bom algoritmo *off-policy* estabelece uma boa relação de utilização entre política comportamental e estimada.

Os métodos de aprendizagem por diferenças temporais, *Temporal Difference* (TD), são os mais importantes no contexto do RL. Eles resultam de uma fusão de ideias dos MCMs com as dos métodos de DP.

Dos MCMs herdam a capacidade de aprender directamente de um conjunto de experiências, sem necessidade de obter o conhecimento total do ambiente. Tal como os métodos de DP, os TDs actualizam a política sem terem que aguardar pelo fim das experiências.

Dois algoritmos importantes que utilizam os Métodos de TD são o *Sarsa*[1] (do

[1]O algoritmo *Sarsa* é assim denominado porque implica que sejam actualizados os 5 eventos seguintes: $(s_k, a_k, r_{k+1}, s_{k+1}, a_{k+1})$

tipo *on-policy*) e o *Q-learning* (do tipo *off-policy*).
Este último, o *Q-learning*, é dos mais relevantes e amplamente utilizado, havendo adaptações para variados propósitos baseados no original de Watkins (1989).

O algoritmo *Q-learning* pode ser definido por,

$$Q(s_k, a_k) \leftarrow Q(s_k, a_k) + \alpha \left[r_{k+1} + \gamma \max_{a_{k+1}} Q(s_{k+1}, a_{k+1}) - Q(s_k, a_k) \right] \quad (3.13)$$

Neste caso, a função de valor estado-acção, Q, aproxima-se a Q^*, função acção--valor óptima, independentemente da política seguida (comportamental ou estimada). Esta premissa implica uma simplificação na avaliação do algoritmo e garante convergência (quando a aprendizagem é feita de forma solitária).

α é o parâmetro que representa a taxa de aprendizagem. Determina o modo como o agente acumula a informação recebida. Para um valor de α igual a 0 o agente não aprende, ppara um valor unitário de α o agente toma em conta toda a informação actual.

γ é o parâmetro do *Q-learning* responsável por determinar a importância das recompensas futuras. Um γ igual a 0 torna o agente oportunista, em que só dá importância aos valores das recompensas imediatas; por oposição, um γ próximo de 1 proporciona ao agente um desconto muito pequeno do seu futuro.

A expressão do algoritmo Q coloca-se da seguinte forma, Figura 3.5. O valor $Q(s, a)$ a actualizar num dado instante de tempo k vai depender do seu valor anterior, do valor da recompensa que o agente recebe ao chegar ao novo estado e da acção que proporcionou obter o valor máximo de Q para qualquer um dos estados possíveis num dado ambiente.

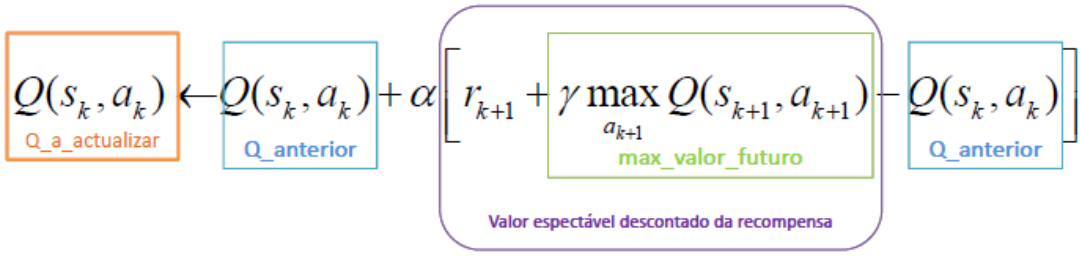


Figura 3.5: Algoritmo *Q-learning*

Por exemplo, se um agente está no estado 2 e optou pela acção 1 para actualizar o valor de $Q(2,1)$ vai precisar do valor de $Q(2,1)$ anterior. Para descontar o valor máximo futuro, precisa de calcular a acção que proporcionou atingir o valor máximo de Q para qualquer um dos estados seguintes possíveis, s_{k+1}.

$$Q(2,1) \leftarrow Q(2,1) + \alpha \left[r_{k+1} + \gamma \max_{a_{k+1}} Q(s_{k+1}, a_{k+1}) - Q(2,1) \right]$$

3.2.2 O *Multi-Agent Reinforcement Learning*

Pode considerar-se o *Multi-Agent Reinforcement Learning* (MARL) como uma extensão da *framework* do RL para sistemas multi-agente.

Uma das principais características diferenciadoras reside na forma em como é definido (e construído) o ambiente em MARL; ao contrário do RL, utiliza-se o *framework* jogo estocástico, substituindo o conceito de MDP como base para a definição do sistema (secção 3.2.1.1).

3.2.2.1 Teoria dos Jogos e MARL

Nesta secção, abordar-se-á a temática "Teoria dos Jogos e MARL" com base no relatório técnico "*An Analysis of Stochastic Game Theory for Multiagent Reinforcement Learning*" de (Bowling and Veloso, 2000).

3.2.2.1.1 Jogos Matriciais

Um jogo matricial ou jogo estratégico, é um tuplo $(n, A_{i...n}, \mathbf{R_{i...n}})$ em que n corresponde ao número de jogadores, A_i é o conjunto da acções disponíveis para o jogador i (A é o espaço de acção conjunta $A_i \times ... \times A_n$), e $\mathbf{R}$ corresponde à função de *payoff*, $A \rightarrow \mathbf{R}$. Os jogadores seleccionam acções do conjunto de acções possíveis com o objectivo de maximizar o *payoff* que depende das suas acções .

Um bom exemplo para um jogo matricial é o jogo "Papel, Tesoura e Pedra". Trata-se de um jogo para 2 jogadores, $n = 2$, as acções disponíveis sendo a escolha de "papel", "tesoura" ou "pedra". O *payoff* pode ser definido, e.g., em relação ao jogador 1, como -1 para o caso de derrota e 1 para o caso de vitória. Na Figura 3.6 pode consultar-se as possíveis combinações das acções e *payoffs* respectivos.

		Acções Jogador1		
		Papel	Tesoura	Pedra
Acções Jogador 2	Papel	0	1	-1
	Tesoura	-1	0	1
	Pedra	1	-1	0

Figura 3.6: Jogo "Papel, Tesoura e Pedra".

Cada iteração resulta na vitória de um dos jogadores (consequentemente derrota do outro) ou empate. Não há nenhuma estratégia determinística que garanta a vitória.

Definição 1

Para um jogo, define-se a função que dá a melhor resposta para o jogador i, *best-response function*, $BR_i(\sigma_{-i})$, como sendo o conjunto de todas as estratégias óptimas, possivelmente aleatórias, assumindo que o(s) outro(s) jogador(es) jogam a estratégia conjunta, σ_{-i}.

Definição 2

O equilíbrio de Nash é uma colecção de estratégias (possivelmente mistas) para todos os jogadores, σ_i, com

$$\sigma_i \in BR_i(\sigma_{-i}). \tag{3.14}$$

Pode concluir-se que nenhum jogador pode fazer melhor se modificar as suas estratégias, assumindo que os outros jogadores continuam a seguir a estratégia de equilíbrio.

O que torna a noção de equilíbrio convincente é que todos os jogos matriciais têm pelo menos um equilíbrio de Nash.

No exemplo do jogo "Papel, Tesoura e Pedra" se um jogador optar tendencialmente por uma das opções, e.g. Papel, e o outro jogador mantiver a estratégia de equilíbrio aleatória, o que optou por "papel" garante que perde um terço das vezes. Logo, não consegue fazer melhor do que "jogar sempre à sorte".

Tipos de jogos matriciais

Os jogos matriciais podem ser puramente colaborativos ou puramente competitivos. Nos jogos puramente colaborativos todos os agentes têm o mesmo *payoff*. Ao invés nos jogos puramente competitivos os *payoff* são simétricos.

3.2.2.1.2 Jogos Estocásticos

Um jogo estocástico é um tuplo $(n, S, A_{1\ldots n}, T, \mathbf{R_{i\ldots n}})$, em que n é o número de agentes, S é conjunto de estados, A_i é o conjunto da acções disponíveis para o jogador i (A é o espaço de acção conjunta $A_i \times ... \times A_n$), T é a função de transição $S \times A \times S \rightarrow [0, 1]$ e $\mathbf{R_i}$ é a função de recompensa para o i-ésimo agente $S \times A \rightarrow \mathbf{R}$. Esta definição parece-se com a de um MDP, todavia possui n agentes que seleccionam acções que resultam num estado seguinte. As recompensas dependem das acções conjuntas dos agentes. Desta forma pode aceitar-se o jogo estocástico como uma extensão natural de um MDP para um sistema multi-agente. Também

é uma extensão de jogos matriciais para múltiplos estados. Cada estado de um jogo estocástico pode ser visto como um jogo matricial como os *payoff* definidos por $\mathbf{R_i}(\mathbf{s}, \mathbf{a})$ para acção conjunta . Após jogar o jogo matricial e receber o *payoff* o jogador transita para um novo estado ou jogo matricial dependendo da aplicação.

3.2.2.2 Características do Multi-Agent Reinforcement Learning

Projectar e implementar um sistema multi-agente traz inúmeras vantagens. Quando se está perante um sistema muli-agente com RL, MARL, um maior número de vantagens poderão ser alcançadas. Com um sistema MARL pode estudar-se e/ou simular-se comportamentos ou características humanas, tais como cooperação, competição, aprendizagem solitária e/ou cooperativa.

Nem sempre é simples encontrar um bom compromisso entre as questões que se querem avaliar quando se projecta um modelo com os objectivos que se estabelecem para os agentes atingirem.

Para além de limitações inerentes ao modelo que é implementado, deve ter-se em conta as limitações impostas pela complexidade dos algoritmos RL, bem como factores de escalabilidade, tais como, a dimensão da sociedade computacional em estudo.

> "A maldição da dimensionalidade inclui o crescimento exponencial do espaço estado-acção discreto no número de estados e variáveis (dimensões) de acção. Desde os algoritmos básicos RL, como o *Q-learning*, estimar valores para cada um dos estados discretos possíveis ou pares estado-acção resulta num crescimento exponencial ao nível da complexidade computacional. A complexidade em MARL é exponencial também ao nível do número de agentes porque cada agente adiciona as suas variáveis ao espaço conjunto estado-acção. Especificar um bom algoritmo MARL num jogo estocástico genérico é um desafio complexo, dado que os retornos dos agentes estão correlacionados e não podem ser maximizados independentemente." (Busoniu et al., 2008)

Tal como nos algoritmos RL convencionais existe um compromisso entre exploração e utilização a manter em MARL. Podem utilizar-se métodos que existem, também no RL, tais como a política ε-greedy ou temperatura de Boltzmann.

> "Em MARL surgem novas complicações porque se está na presença de múltiplos agentes. Os agentes exploram para obter informação não somente do ambiente, também de outros agentes. Demasiada exploração, contudo, pode destabilizar as dinâmicas de aprendizagem dos outros agentes, tornando as tarefas de aprendizagem mais difíceis para o agente explorador. A necessidade de coordenação num dado sistema, tem origem no facto de uma acção de um agente no ambiente depender também das acções dos outros agentes no mesmo ambiente." (Busoniu et al., 2008)

Tal facto pode ser constatado através da análise de jogos estocásticos. Em jogos estocásticos totalmente cooperativos, o retorno pode ser maximizado em conjunto porque é o mesmo para todos os agentes. Todavia, noutros tipos de jogos, os retornos dos agentes são diferentes, apesar de estarem correlacionados. Desta feita, conclui-se que "especificar um objectivo bom em MARL é geralmente um problema difícil." (Busoniu et al., 2008)

Os objectivos de aprendizagem geralmente incorporam duas questões essenciais: — a estabilidade da dinâmica de aprendizagem e a adaptação à dinâmica comportamental. A questão da estabilidade é também importante, mas resume-se à existência de convergência para uma política estacionária. (Bowling and Veloso, 2002, 2001)

Bowling and Veloso (2001), consideram que a convergência é necessária para a estabilidade e adicionam o conceito de racionalidade como critério de adaptação.

A "racionalidade é definida como uma exigência para a convergência do agente para uma função *best-response* quando os restantes agentes se mantêm num estado estacionário. Embora a convergência para um equilíbrio de Nash não seja

explicitamente exigida, surge naturalmente se todos os agentes no sistema forem racionais e convergentes."

Um conceito alternativo à racionalidade é o do *no-regret*, que é definido como a exigência que o agente tem em recolher um retorno pelo menos tão bom como o de qualquer política estacionária, que se mantém para qualquer conjunto das estratégias de outros agentes. Podem ser acrescentados alguns requisitos de adaptação em termos de limites de recompensas médias, tais como: a optimalidade alvo, compatibilidade e segurança.

A optimalidade alvo requer uma recompensa média, em vez de um conjunto de algoritmos alvo, o que é pelo menos a recompensa média de uma função *best-response*. A compatibilidade prescreve um nível de recompensa média no próprio jogo. A segurança pressupõe um nível de segurança da recompensa média contra todos os algoritmos. (Busoniu et al., 2008; Powers and Shoham, 2004)

Não é claro de que forma se estendem estas características aos restantes critérios, e nem é garantido que um algoritmo que as possua convirja para uma política estacionária.

Outro tipo de relação que se pode estabelecer é a de "oponente consciente". Um algoritmo "oponente consciente" converge para uma estratégia que é parte de uma solução de equilíbrio independentemente do que os outros agentes estão a fazer. Um algoritmo "oponente consciente" aprende os modelos de outros agentes e reage usando uma forma de função *best-response* (Busoniu et al., 2008)

Baseando-se nas características deste tipo de algoritmo pode introduzir-se mais uma característica do agente: a predição. A predição é a capacidade do agente de aprender de forma aproximada dos modelos de outros agentes (Chalkiadakis, 2003). Chalkiadakis (2003) acrescenta que neste contexto, que "um agente é considerado racional se maximizar o retorno espectável conhecendo os modelos dos outros agentes".

3.3 Inteligência Colectiva

Tadeusz Szuba propõe a seguinte definição para o fenómeno da inteligência colectiva: "um sistema que possui inteligência colectiva deve ser inconsciente, aleatório, paralelo, é um processo computacional distribuído, e é construído em lógica matemática segundo a sua estrutura social." (Szuba, 2001b)

O modelo de sociedade artificial que está implementado, sem racionalidade, utiliza analogias com o intuito de facilitar o entendimento dos processos de interacção em que os agentes estão envolvidos. Algumas analogias recorrentes prendem-se com a utilização de termos como "dinheiro", "compra", "venda", "mercado", etc.

Szuba (2001a) na obra "Computacional Collective Intelligence", a partir de simulações que realizou em *Random Parallel Prolog* (RPP), sugere que o dinheiro é uma descoberta da inteligência colectiva das sociedades em estágios iniciais de desenvolvimento. Realizou também diversas simulações em RPP que sugerem que:

- O dinheiro é um mecanismo geral de cooperação;
- O dinheiro requer um nível mínimo de inteligência dos agentes individuais;
- O dinheiro requer um nível mínimo de complexidade da estrutura social;
- Um sistema financeiro pode ser *quasi*-caótico e não requer controlo centralizado;
- O dinheiro não requer relações interpessoais para quem efectua as trocas.

Em sintonia com as ideias de Szuba, Garrido (2010c) definiu o modelo (secção 2.1) que serve de base para implementação e estudo da inteligência deste trabalho.

3.3.1 O algoritmo escolhido: — *single Q-learning* para MARL

Neste contexto é interessante observar de que maneira um agente independente pode evoluir numa sociedade colectiva, alterando as suas preferências. Para tal, escolheu-se o algoritmo *single agent Q-learning*.

> Considera-se que "um algoritmo MARL é um algoritmo com um agente que aprende de forma independente se os agentes aprendem os valores de Q para as suas acções individuais baseando-se na equação 3.13 . Isto é, realizam as suas acções, obtêm uma recompensa e actualizam os seus valores de Q sem terem conhecimento das acções realizadas pelos outros agentes" (Claus and Boutilier, 1998).

Num modelo económico pode tornar-se uma questão pertinente a reter, dado que nenhum agente pode restringir as suas acções baseando-se nas acções ou tendências de outros.

Sob o ponto de vista sociológico, é interessante observar a evolução de uma sociedade que pode sair beneficiada (ou prejudicada) pela soma das partes, isto é, não há nenhum agente coordenador ou legislador . A "lei" resume-se às regras básicas impostas pelo modelo.

Capítulo 4

Implementação

Neste capítulo abordar-se-ão os assuntos relacionados com a implementação do modelo. Entre eles as plataformas utilizadas, algoritmos, programação vectorial e descrição do jogo.

4.1 Plataformas utilizadas na implementação

4.1.1 *Laboratory for Simulation Development*

Uma das plataformas utilizadas na implementação e desenvolvimento do modelo referido na secção 2.1, foi o *Laboratory for Simulation Development* (LSD). Trata-se de uma plataforma *opensource* criada e desenvolvida por Marco Valente[1], que tem como objectivo romper com os *trade-offs* associados às ferramentas de programação para simulação. (Valente, 2010)

Os modelos implementados em LSD são essencialmente programas em C++. Todavia, utilizam-se as *macros* criadas pelo autor para implementar o modelo com agentes, como pode ser consultado um exemplo em "Bloco de código 4.1" de um "código tipo" em LSD.

[1]Economista, investigador e professor associado na Universidade de Áquila (Itália)

Em alternativa, pode misturar-se código C com os *macros* do LSD, porém não estão acessíveis de uma forma imediata instruções mais complexas.

Bloco de código 4.1: Código tipo LSD

```
EQUATION("Sales")
/*
Level of sales:
Sales[t]=Sales[t-1]*{1+a*(Quality/AverageQuality-1)}
The sales of a firm are adjusted in respect of the ratio between
   firm's own Quality and the
average quality.
*/
v[0]=V("a");
v[1]=VL("Sales",1);
v[2]=V("Quality");
v[3]=VL("AverageQuality",1);
RESULT( v[1]*(1+v[0]*(v[2]/v[3]-1)) )
```

O LSD possui uma janela principal, denominada *Lsd Model Manager* (LMM) (Figura 4.1) em que se faz a edição e compilação do código LSD. Após a compilação (sem erros) pode utilizar-se a janela LSD (Figura 4.2) onde é construído o modelo multi-agente, onde se definem os parâmetros e se inicializam as variáveis. A partir da janela LSD pode executar-se a simulação, ficando acessível um utilitário para a análise de dados (Figura 4.4).

O LSD não suporta de modo nativo o desenho de gráficos, utiliza do *gnuplot* para tal tarefa (Figura 4.5).

As vantagens e desvantagens da utilização dessa plataforma serão mencionadas de uma forma mais extensa na secção 4.5. Pode-se desde já adiantar que o LSD revelou ter problemas de escalabilidade que levaram à procura de alternativas à sua utilização, entre elas o ScicosLab.

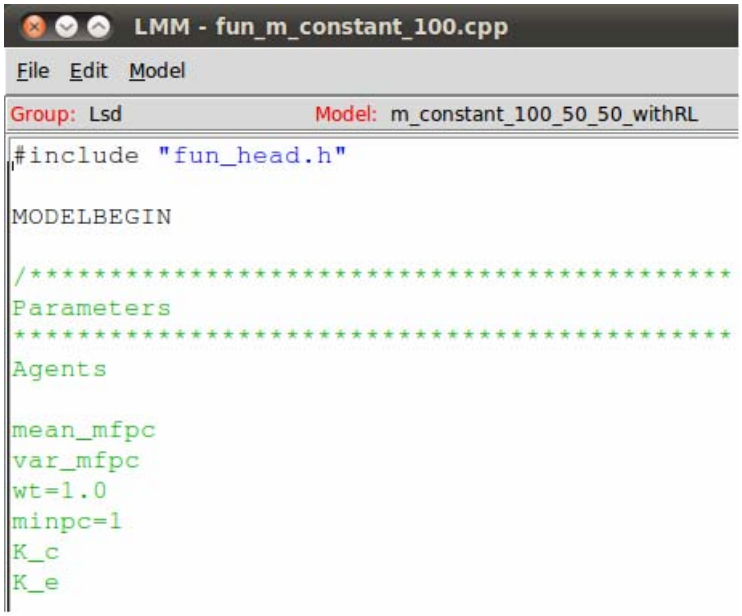


Figura 4.1: Janela LMM.

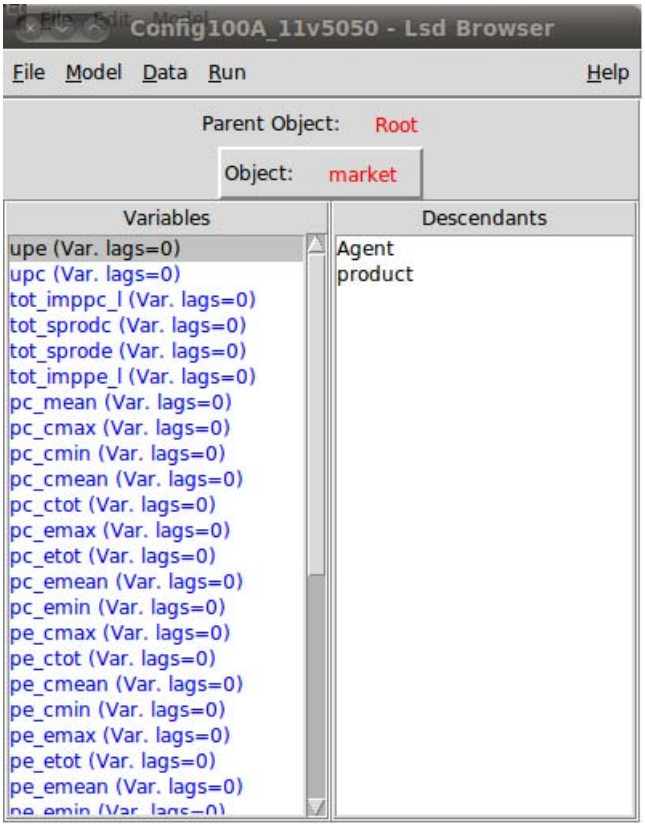


Figura 4.2: Janela LSD.

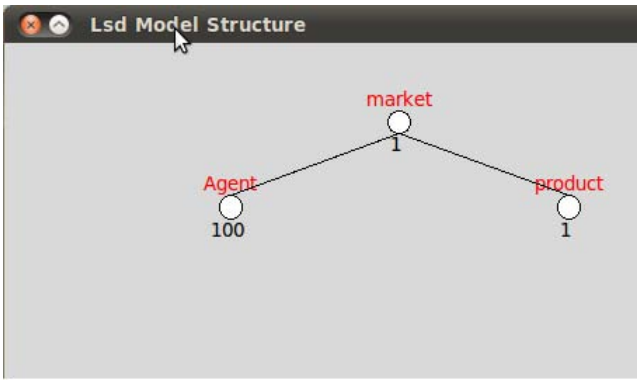


Figura 4.3: Janela que contém a estrutura do modelo.

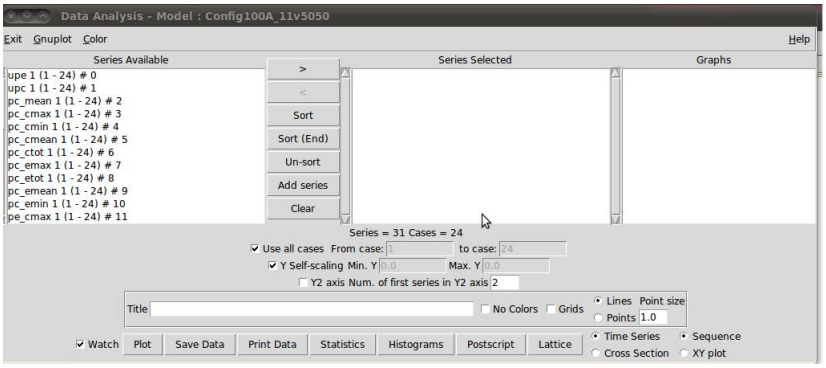


Figura 4.4: Janela em que se faz a análise dos dados.

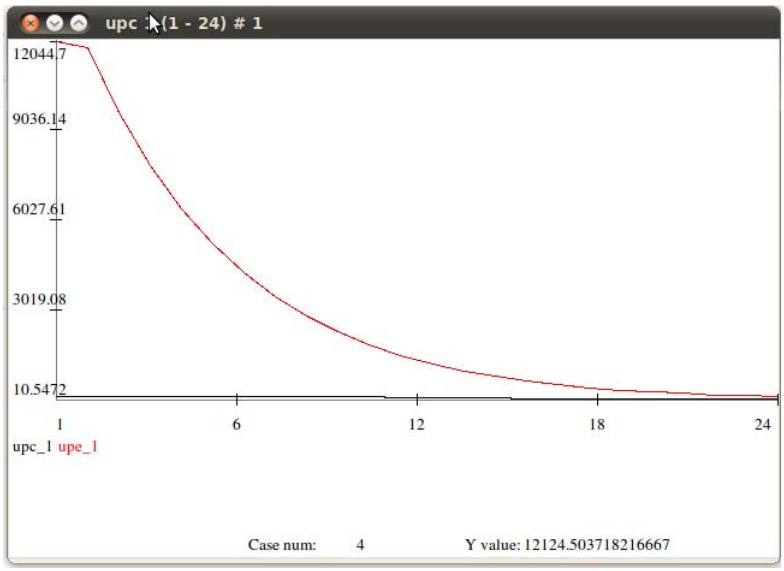


Figura 4.5: Gráfico desenhado em *gnuplot*.

4.1.2 *ScicosLab*

O Scilab é uma plataforma gratuita dedicada à computação numérica.

O ScicosLab (Figura 4.6) é uma versão da plataforma Scilab em ambiente GTK+, que é suportada pelo INRIA (*Institut National de Recherche en Informatique et en Automatique*). Para além de ser um *interface* GTK+ do Scilab agrega também o Scicos que permite modelização e simulação por blocos, à semelhança do Matlab Simulink.

Tanto o ScicosLab como o Scilab possuem uma licença compatível com a *Gnu Public License* (GPL).

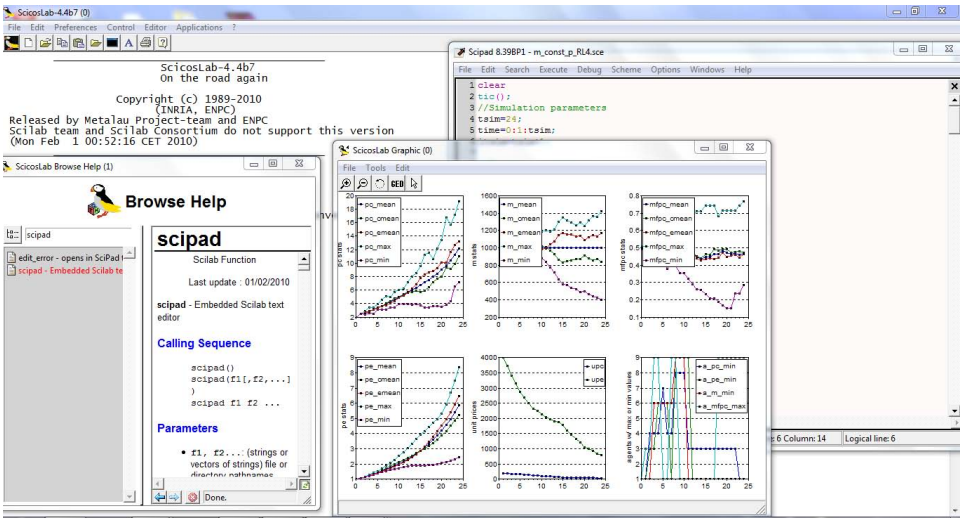


Figura 4.6: Ambiente *ScicosLab*.

4.2 Implementação do algoritmo *Q-learning* em LSD

Não é possível utilizar matrizes em LSD.

O LSD estabelece relações entre objectos num formato hierárquico em árvore. Por exemplo, como pode ser consultado na Figura 4.7, o agente do tipo *market*

contém um objecto do tipo *agent* e o agente do tipo *product*. Desta forma o objecto *market* tem acesso a todas as variáveis do objecto *agent* e do objecto *product*. Porém os objectos *product* e agente não tem acesso directo às variáveis um do outro. O objecto *agent* foi "clonado" cem vezes, originando, deste modo, cem agentes que partilham as mesmas equações, variáveis e parâmetros entre si.

O conjunto de instruções (por exemplo, consultar a lista 4.1) bem como o esquema de herança são as características mais relevantes da utilização do LSD.

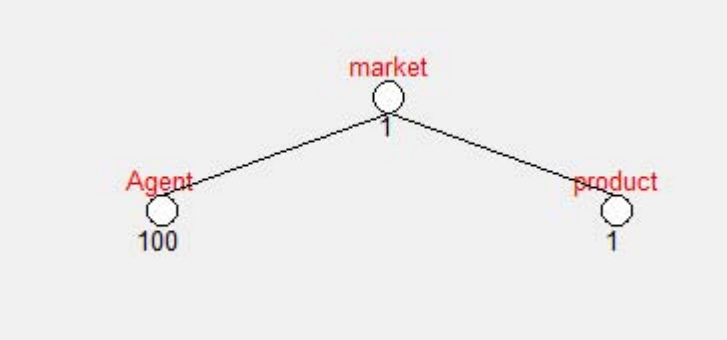


Figura 4.7: Esquema relacional do programa LSD.

O modelo original foi previamente codificado em LSD. Era objectivo deste trabalho acrescentar a "funcionalidade MARL" à plataforma LSD. Todavia esse objectivo acabou por se revelar tecnicamente impossível dentro da calendarização estipulada para este projecto. Porém, sem acrescentar esse módulo à plataforma é possível dotar os agentes com o algoritmo Q utilizando o LSD. Este não suporta a manipulação de matrizes. Não obstante, é possível utilizar as relações de herança para se "construir uma matriz".

Para se perceber o funcionamento das matrizes propôs-se resolver um problema simples para um único agente com *Q-learning* na linguagem LSD. A verificação será realizada através da implementação do mesmo programa noutra plataforma — o ScicosLab.

4.2.1 Exemplo *Gridworld*

O problema escolhido foi um do tipo "*Gridworld*". O mundo em estudo possui nove estados possíveis . A cada estado está associada a respectiva recompensa. Para os estados 1, 2, 3; 7, 8 e 9 a recompensa é de -1000, para os estados 4 e 5 a recompensa é de 0 e para o estado 6 a recompensa é de 1000 (consultar a Figura 4.8)

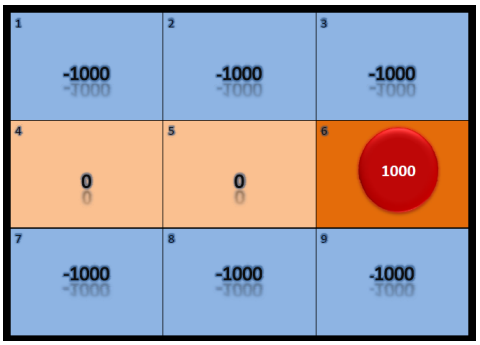


Figura 4.8: Representação do mundo e respectivas recompensas.

Em cada estado o agente pode escolher uma das quatro acções possíveis (cima, direita, baixo, esquerda), numeradas de 1 a 4, como ilustrado na Figura 4.9.

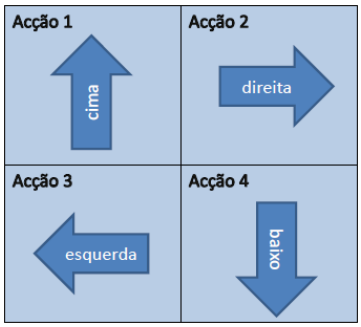


Figura 4.9: Acções que um agente tem acesso em cada estado.

O agente inicia o percurso de forma aleatória em qualquer um dos estados e procurará atingir o objectivo que é alcançar o estado 6. Na Figura 4.10 está ilustrado um possível caminho a seguir.

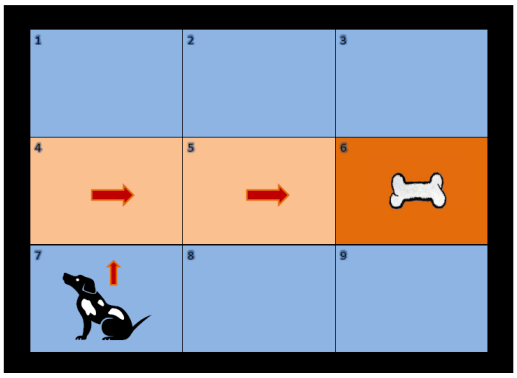


Figura 4.10: Possível caminho para um determinado agente chegar ao seu objectivo.

Como supramencionado, o LSD não suporta matrizes de forma nativa. Quando se quer utilizar matrizes é necessário estabelecer uma relação entre objectos para se conseguir manipular dados que tenham de ser organizados no formato de matriz. Para resolver o problema em particular, que passa por "construir" a matriz $Q(s, a)$ encontrou-se a relação entre objectos ilustrada na Figura 4.11. Os objecto *QsaStates* e *QsaActions* correspondem, respectivamente, ás colunas e linhas da matriz $Q(s, a)$.

O código da linguagem LSD que resolve o problema *Gridworld* encontra-se no anexo A.1.1

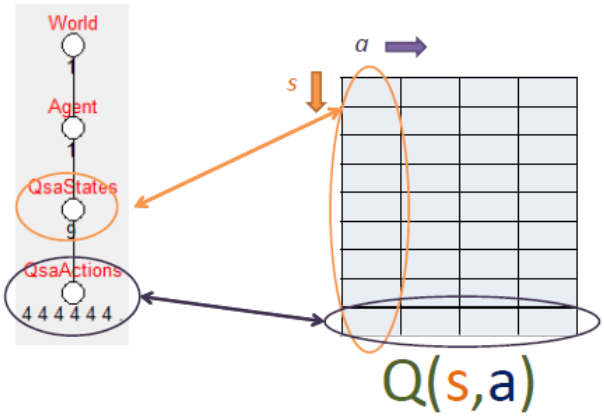


Figura 4.11: Relações de equivalência entre matrizes e o LSD.

4.2.1.1 Verificação do código em LSD

Com o objectivo de se provar a consistência da codificação em LSD implementou-se um programa em ScicosLab com o mesmo algoritmo que resolve o problema *Gridworld*. O código em Scicoslab pode ser consultado no anexo A.1.2.

Para o mesmo número de episódios, 100000, os resultados nas duas plataformas são os mesmos. Na Figura 4.12 podem consultar-se os valores finais de Q para todos os estados e respectivas acções. As setas a verde representam a melhor acção possível para cada estado.

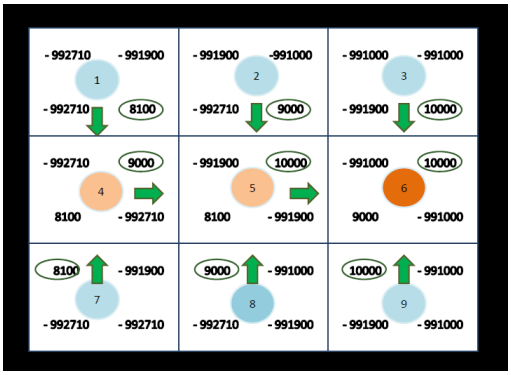


Figura 4.12: Resultados do problema *Gridworld*.

4.3 Descrição do jogo estocástico

É implícita a um problema implementado com um algoritmo MARL a definição do respectivo jogo estocástico (secção 3.2.2.1).

Como anteriormente referido (em 3.2.2.1.2) a definição do jogo estocástico consiste em determinar quais as acções e estados associados a um agente inserido num determinado ambiente. Este ambiente fornece as respectivas recompensas (*rewards*). As regras que regem o ambiente, em que estão inseridos os agentes, são as propostas por Garrido (2010c) em "Modeling and Simulation of Markets upon a Collective Intelligence Perspective".

No contexto deste trabalho utilizaram-se diferentes recompensas :

- Quantidade de produtos de consumo na posse de cada agente (pc);
- Quantidade de produtos de equipamento na posse de cada agente (pe);
- Quantidade de dinheiro na posse de cada agente (m).

Determinou-se que, neste ambiente, o agente pode estar em três estados distintos. Os estados foram determinados, num dos casos, em função da quantidade de produtos de consumo na posse dos agentes, pc, em relação à quantidade média de produtos de consumo na posse de todos os agentes, pc_mean. No outro caso, os estados são determinados em função da quantidade de produtos de equipamento da posse dos agentes, pe, em relação à quantidade média de produtos de equipamento na posse de todos os agentes, pe_mean.

Tomando como exemplo o caso de pc e pc_mean:

- **Estado 1**, se o agente possui uma maior quantidade de produtos de consumo (pc) do que $\pm 5\%$ do valor médio da quantidade de produtos de consumo a ($\pm 5\% * pc\_mean$) da sociedade em que está inserido;
- **Estado 2**, se o agente possui uma quantidade de produtos de consumo (pc) que está num intervalo de $\pm 5\%$ do valor médio da quantidade de produtos de consumo ($\pm 5\% * pc\_mean$) da sociedade em que está inserido;
- **Estado 3**, se o agente possui uma menor quantidade produtos de consumo (pc) do que $\pm 5\%$ do valor médio da quantidade de produtos de consumo ($\pm 5\% * pc\_mean$) da sociedade em que está inserido;

O agente poderá modificar as preferências específicas de compra dos produtos de consumo ($mfpc$) e dos produtos de equipamento ($mfpe$). É requisito do modelo a soma das preferências dos diferentes tipos de equipamento ser igual a 1. Por conseguinte, o agente tem à disposição as seguintes opções:

- Incrementar o valor das preferências por produtos de consumo;
- Decrementar o valor das preferências por produtos de consumo;
- Manter as preferências actuais.

Para o caso do estado ser determinado em função dos produtos de equipamento, *pe*, as comparações são realizadas em relação à média de produtos de equipamento na posse dos agentes, *pe_mean* em vez do valor médio dos produtos de consumo na posse dos agentes, *pc_mean*.

4.4 Implementação do modelo com MARL em ScicosLab

Pode consultar-se o código em ScicosLab referente ao modelo descrito na secção 2.1, no apêndice A.5.

Um dos códigos que proporciona racionalidade aos agentes pode consultar-se no apêndice A.4.

Utilizou-se o *single agent Q-learning* (3.3.1) para a implementação de racionalidade nos agentes.

A abordagem para se implementar um sistema multi-agente em ScicosLab baseou-se em técnicas de programação vectorial.

O algoritmo correspondente à configuração em que o estado é determinado em função da quantidade de produtos de consumo na posse dos agentes, *pc*, em relação à quantidade média de produtos de consumo na posse dos agentes, *pc_mean*, e o critério a maximizar (recompensa) escolhido é a quantidade de produtos de consumo, *pc*. Encontra-se no Apêndice A.2.

4.4.1 Programação vectorial

A abordagem às técnicas de programação vectorial foi realizada com base no *working paper* de Garrido (2010d) "Programming heterogeneous behavioral agent models with vector and matrix data types support".

Vai-se, então, começar por descrever o caso de um agente único

4.4.1.1 Caso de um só agente

Para programar um agente computacional a partir de um modelo comportamental deve-se estabelecer uma regra para processar o valor de cada uma das variáveis no modelo no instante k. Ao fazê-lo, qualquer indeterminação causal que as equações do modelo possam ter presente deve ser removida. Ambas as tarefas podem ser executadas se se expressar cada equação no modelo para definir o cálculo de uma variável.

Vai distinguir-se as variáveis do modelo com agentes, em variáveis de entrada ou variáveis externas $u_1, ..., u_l, ..., u_L$, e variáveis internas $v_1, \ldots, v_m, \ldots, v_M$. Vão assumir-se como parâmetros $g_1, \ldots, g_{mp}, \ldots, g_{MP}$. As variáveis u são processadas de forma externa ao agente. Pode escrever-se:

$$\begin{aligned} & v_1(k) = f_1\left(\mathbf{v}_1(k), \mathbf{u}_1(k), \mathbf{g}_1\right) \\ & ... \\ & v_m(k) = f_m\left(\mathbf{v}_m(k), \mathbf{u}_m(k), \mathbf{g}_m\right) \\ & ... \\ & v_M(k) = f_M\left(\mathbf{v}_M(k), \mathbf{u}_M(k), \mathbf{g}_M\right) \end{aligned} \tag{4.1}$$

Os componentes dos vectores $\mathbf{v}_m(k)$ são variáveis internas, quer no instante k ou em algum instante anterior, utilizadas para calcular o valor de $v_m(k)$. Os vectores $\mathbf{u}_m(k)$ são variáveis externas, quer no instante k ou em algum instante

anterior, utilizadas para calcular o valor de $v_m(k)$. As componentes e parâmetros dos vectores $\mathbf{g}_m(k)$ são utilizados para calcular o valor de $v_m(k)$. As equações 4.1 devem satisfazer as seguintes condições:

- Claramente, nenhum $\mathbf{v}_m(k)$ pode conter $v_m(k)$.
- A ordenação das funções $f_1, ..., f_m, ..., f_M$ deve ser tal que $v_m(k)$ não necessite um valor do conjunto $\{v_{m+1}(k), ..., v_M(k)\}$.

Além disso, a ordenação para a computação dos valores das variáveis por todos os agentes no modelo deve ser tal que: — Todos os valores de $\mathbf{u}_m(k)$ devem ser processados.

Assumindo que o modelo é composto por agentes únicos heterogéneos, o último requisito impõe uma sequência de cálculo ao longo do agentes, ou das variáveis dos agentes, a cada instante k.

Supondo que o objectivo do modelo é gerar uma instância do comportamento do agente, i.e.. os valores das variáveis desde o instante $k = 1$ até $k = K$. As condições iniciais são valores das variáveis no instante 0. Para cada variável $v_m(k)$ no modelo um array uni-dimensional é gerado. Tal está ilustrado na Figura 4.13 .

Figura 4.13: *Array* uni-dimensional com $v(0)$ inicial e os valores calculados das variável v_m do instante 1 até K.

4.4.1.2 Agentes homogéneos

Agentes homogéneos partilham o mesmo conjunto de variáveis $v_1, ..., v_m, ..., v_M$ e as mesmas funções computacionais $f_1, ..., f_m, ..., f_M$. São distinguidas por terem diferentes valores de parâmetros. Vai colocar-se um índice j sobre os agentes ho-

mogéneos de 1 a classe JN. Utilizando os índices j e k, pode indexar-se cada variável ao longo dos agentes j e a instantes de tempo k: $v_1(j,k), ..., v_m(j,k), ..., v_M(j,k)$. Além disso, pode indexar-se os parâmetros como $g_1(j), ..., g_{mp}(j), ..., g_{MP}(j)$.

As equações 4.1 são empilhadas por agente. Se se escrever as equações para a mesma variável ao longo dos agentes homogéneos (empilhar sobre as variáveis) tem-se para a variável v_m:

$$\begin{aligned} & v_m(1,k) = f_m\left(\mathbf{v}_m(1,k), \mathbf{u}_m(1,k), \mathbf{g}_m(1)\right) \\ & ... \\ & v_m(j,k) = f_m\left(\mathbf{v}_m(j,k), \mathbf{u}_m(j,k), \mathbf{g}_m(j)\right) \\ & ... \\ & v_m(NJ,k) = f_m\left(\mathbf{v}_m(NJ,k), \mathbf{u}_m(NJ,k), \mathbf{g}_m(NJ)\right) \end{aligned} \tag{4.2}$$

Supondo que o objectivo do modelo é gerar uma instância do comportamento do agentes, i.e.. os valores das variáveis desde o instante $k = 1$ até $k = K$. As condições iniciais são valores das variáveis no instante 0. Para cada variável $v_m(j,k)$ no modelo um array NJ-dimensional é gerado. Tal está ilustrado na Figura 4.2.

$v_m(1,0)$	$v_m(1,1)$	...	$v_m(1,k)$	...	$v_m(1,K)$
...	...	...	...	...	...
$v_m(j,0)$	$v_m(j,1)$	...	$v_m(j,k)$	...	$v_m(j,K)$
...	...	...	...	...	...
$v_m(NJ,0)$	$v_m(NJ,1)$	...	$v_m(NJ,k)$	...	$v_m(NJ,K)$

Figura 4.14: *Array* bi-dimensional com os valores iniciais de $v(j,0)$ e os valores calculados da variável $v_m(j,k)$ dos agentes 1 até NJ do instante 1 ate K

4.4.1.3 Aplicação ao modelo em estudo

No modelo implementado, os agentes produtores de bens de consumo e de equipamento comportam-se como os agentes homogéneos supra descritos. Assim, as instâncias das suas variáveis e parâmetros foram codificados como ilustrado na Figura 4.2 . Por sua vez as variáveis e parâmetros do mercado que todos agentes partilham comportam-se como no caso de um agente único, como ilustrado na Figura 4.1 .

4.5 Vantagens e desvantagens do ScicosLab e do LSD

4.5.1 Vantagens e desvantagens do ScicosLab

O ScicosLab, em relação ao LSD, tem como principal vantagem uma maior facilidade de aceder a funções matemáticas e suporta directamente *arrays*. Outra vantagem notória é possuir ferramentas que permitem visualizar de uma forma mais eficaz gráficos e resultados.

O ScicosLab é altamente portável, pois é distribuído para MAC OS X, Windows e Linux.

Em relação a outras plataformas do mesmo tipo, por exemplo Matlab, a grande vantagem reside em ter uma licença GPL.

A principal desvantagem que apresenta em relação ao LSD é que o código não é compilado, é interpretado, o que implica que a execução seja potencialmente mais lenta.

A desvantagem que apresenta em relação a outras plataformas dedicadas à computação numérica é a fraca documentação e suporte. Algumas das funções escritas para ScicosLab não são *standard* em relação a outras plataformas deste

tipo.

4.5.2 Vantagens e desvantagens do LSD

A principal vantagem do LSD é o código ser compilado, tornando a execução do programa mais rápida do que o ScicosLab para modelos mais pequenos. Permite facilmente clonar objectos variando o número de agentes num determinado problema sem ter que alterar código. É possível guardar várias configurações diferentes para um mesmo código e carregá-las de forma simples.

Tal como o ScicosLab é uma plataforma *opensource*, com licença GPL.

As principais desvantagens do LSD prendem-se com problemas de escalabilidade. Sempre que um problema se torna mais complexo o LSD apresenta dificuldades na codificação, bem como na inicialização das variáveis. Somando ao facto de não suportar directamente a manipulação de *arrays* pode afirmar-se que não se trata da plataforma indicada para desenvolver modelos mais complexos, que possuam uma grande quantidade de variáveis e agentes, ou que necessitem de manipulação de funções mais complexas.

O sistema de análise de resultados também é deficitário, bem como o *plot* dos gráficos, que nem sempre se revela eficaz na visualização dos dados. Este facto levou a que no decorrer deste trabalho se fizessem *scripts* de importação dos dados do ficheiro de simulação para serem tratados em ScicosLab/Scilab. Para além do programa codificado em Scilab, o ficheiro resultado de simulação do LSD teria que ser copiado para uma folha de cálculo *OpenOffice Calc*. O código respectivo pode ser consultado no apêndice A.3.

Capítulo 5

Análise de Resultados

Neste capítulo abordar-se-á a análise dos resultados obtidos das diferentes simulações realizadas.

5.1 Opções de simulação

Optou-se por apresentar todos os gráficos deste capítulo com 800 passos de simulação. Este número surgiu da experimentação do modelo com RL, tendo-se revelado um número de iterações necessário para observar a evolução dos agentes sob a influência do *Q-learning* no modelo.

No modelo com RL, por vezes, a mesma variável atinge valores com ordens de grandeza muito diferentes, por exemplo 10^{-10} e 10^{6}. Com a utilização de escalas logarítmicas é possível observar, no mesmo gráfico, evoluções de variáveis com ordens de grandeza tão díspares.

Apesar de estar estabelecido um mínimo de subsistência, *minpc*, optou-se por não eliminar os agentes que atingem valores inferiores a este. Conceptualmente esses agentes "morrem", porém o objectivo de os manter em simulação é verificar a dinâmica do modelo com todos os agentes, sob o ponto de vista matemático.

De relembrar que todos os agentes são iguais dentro das suas classes. Nas simulações em que estes são dotados de racionalidade as diferenças entre eles advêm das diferentes opções tomadas.

5.2 Modelo desprovido de decisão

Como anteriormente mencionado na secção 4.4 implementou-se o modelo desprovido de racionalidade em ScicosLab (consultar anexo A.4). O referido modelo foi explorado para números iguais de agentes produtores de bens de equipamento e agentes produtores de bens de consumo. As preferências de compra dos agentes são constantes ao longo do tempo e iguais para todos os agentes.

5.2.1 Cálculo dos valores mínimos de K_c e K_e

Existem diferentes valores de K_c e K_e que garantem a sustentabilidade da sociedade. Como anteriormente referido, em 2.1.2, os valores mínimos estão dependentes da escolha dos valores dos parâmetros nível de vida, S, quantidade mínima de produtos de consumo que garante a subsistência, $minpc$, e da quantidade inicial de produtos de equipamento que os agentes têm à disposição, pe_init.

Assim, para um nível de vida $S = 2$, sendo a quantidade mínima para a subsistência do agente unitária, $minpc = 1$, pode determinar-se a produtividade mínima, K_{c_min}, para garantir essa mesma subsistência recorrendo à equação 2.20. Assim K_{c_min} pode determinar-se como:

$$K_{c\_min} = \frac{S \cdot J \cdot minpc}{tot\_pec} \tag{5.1}$$

Todos os agentes são inicializados com uma quantidade de produtos de equipamento unitária, $pe\_init = 1$, sendo $tot\_pec = J_c \cdot pe\_init$ e $J_c = 1/2 \cdot J$, pode

reescrever-se a equação 5.1 do seguinte modo:

$$K_{c\_min} = \frac{S \cdot J \cdot minpc}{1/2 \cdot J \cdot pe\_init} \tag{5.2}$$

A equação 5.2 é equivalente a,

$$K_{c\_min} = \frac{2S \cdot minpc}{pe\_init} \tag{5.3}$$

Substituindo pelo valor de S, de $minpc$, e pe_init obtém-se $K_{c\_min} = 4$.

Determina-se o valor de produtividade de produtos de equipamento que permite sustentar à partida um *standard* no nível de vida, K_{e_min}, de um modo similar como se determinou K_{c_min}.

A partir da equação 2.23 determina-se K_{e_min}. Neste modelo a taxa de decaimento para os produtos de equipamento é $\gamma_e = 0.05$. Sabe-se que o total de produtos de equipamento na posse de produtores de bens de equipamento traduz-se em $tot\_pe = J_e \cdot pe\_init$, o número de produtores de equipamento corresponde a metade da sociedade, $J_e = 1/2 \cdot J$, e que o total de produtos de equipamento estendido a todos os indivíduos expressa-se por $tot\_pee = J_e \cdot pe\_init$. Logo, pode reescrever-se a equação 5.3 da seguinte forma:

$$K_{e\_min} = \frac{2\gamma_e}{pe\_init} \tag{5.4}$$

Substituindo, obtém-se $K_{e\_min} = 0.1$.

Utilizando os valores mínimos de K_c e K_e que garantem sustentabilidade no modelo inicialmente proposto em que as preferências de compra dos agentes são fixas, obtém-se os resultados ilustrados na Figura 5.1.

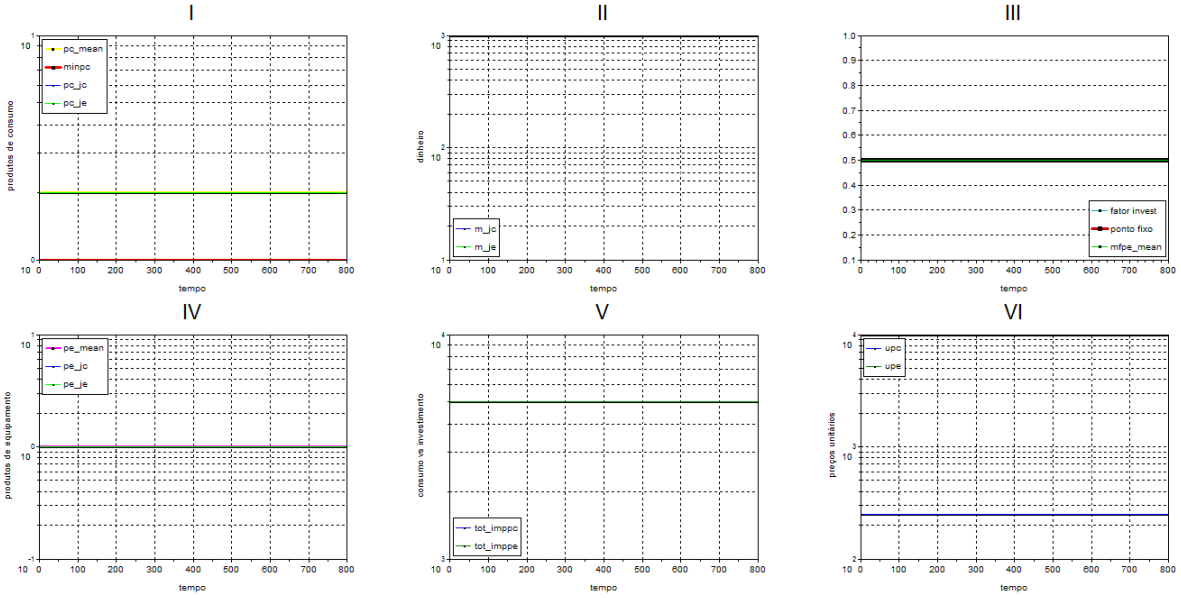


Figura 5.1: Gráficos correspondentes aos valores de K_c e K_e mínimos que garantem e sustentabilidade correspondente às:

Gráfico I - Distribuições de produtos de consumo pelos agentes

Gráfico II - Distribuição de dinheiro pelos agentes

Gráfico III - Distribuição das preferências dos agentes e factor de investimento

Gráfico IV - Distribuição de produtos de equipamento pelos agentes

Gráfico V - Totais de dinheiro investido no consumo vs investimento

Gráfico VI - Preços unitários

A partir da análise da Figura 5.1 pode verificar-se que os valores mantêm-se constantes ao longo do tempo garantindo a sustentabilidade da sociedade. De notar que q quantidade de produtos de consumo é igual ao valor mínimo de subsistência. Os valores mínimos de K_c e K_e conduzem o modelo ao menor valor de um ponto fixo que garante a sustentabilidade dos agentes. O ponto fixo deste modelo define-se pela equação 2.32. Substituindo na equação 2.32 os valores de K_e por 0.1 e γ_e por 0.05, obtém se o factor de investimento $\iota_e = 0.5$ que se pode

visualizar no gráfico III da Figura 5.1.

De salientar que utilizar estes valores de K_c e K_e não garante o cumprimento do critério comportamental que visa verificar que a riqueza aumenta a cada instante de tempo de $k-1$ a k (equação 2.2).

Para ser possível a riqueza aumentar ao longo do tempo (equação 2.2) os valores de K_c e K_e utilizados são 10% superiores em relação aos mínimos. Sendo assim, $K_c = 4.4$ e $K_e = 0.11$.

5.2.2 Cálculo do ponto fixo para os novos valores de K_c e K_e

Substituindo na equação 2.32, $K_e = 0.11$ e $\gamma_e = 0.05$ obtém-se o valor do factor de investimento $\iota_e = 0.45$. Utilizando o valor calculado do factor de investimento ι_e como valor das preferências pela aquisição de produtos de equipamento, $mfpe = \iota_e$, obtêm-se os resultados ilustrados na Figura 5.2

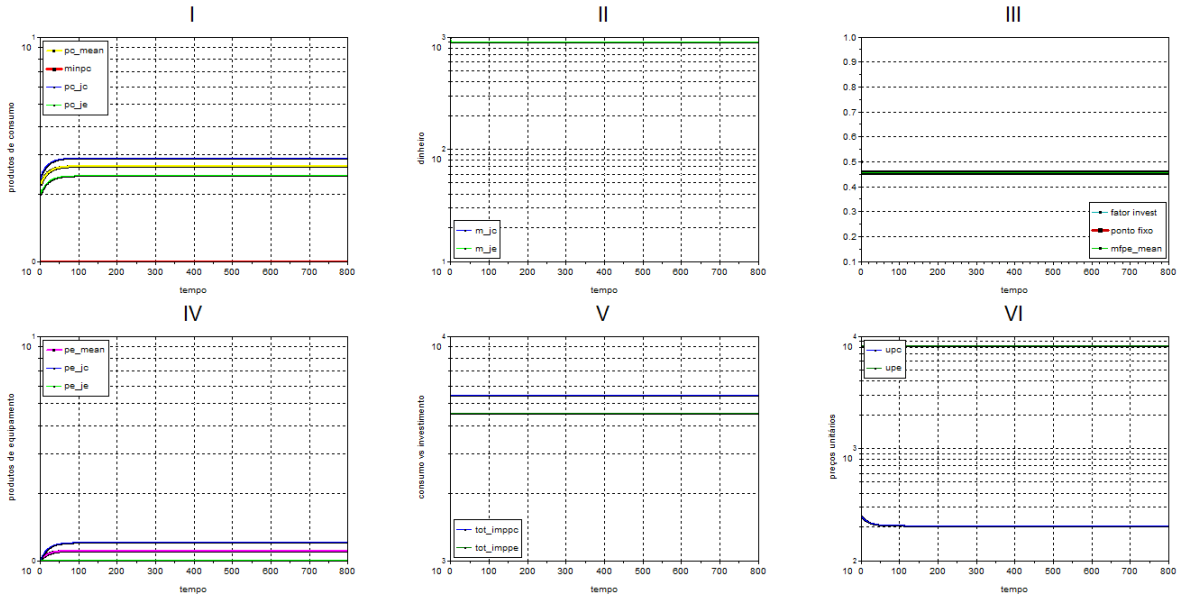


Figura 5.2: Resultados de simulação no ponto fixo, $mfpe = 0.45$.

Como se constata partir da análise da Figura 5.2 os valores das variáveis ficam

constantes a partir das 100 iterações. Desta forma, confirma-se a existência de um ponto fixo para $\iota_e = 0.45$ para os valores de K_c e K_e utilizados no estudo do modelo tanto com preferências fixas como com RL.

5.2.3 Variação dos valores das preferências dos agentes

Estudou-se a variação das preferências de compra dos agentes. O objectivo é observar a evolução da sociedade e averiguar se são cumpridos os critérios das equações 2.1 e 2.2. Os casos estudados foram:

- Maior tendência para adquirir produtos de equipamento, $mfpe = 0.8$, $mfpc = 0.2$;

- Maior tendência para adquirir produtos de consumo, $mfpe = 0.2$, $mfpc = 0.8$;

- Preferências de compra equitativas para os 2 tipos de produto, $mfpe = 0.5$, $mfpc = 0.5$.

5.2.3.1 Tendência para comprar mais produtos de equipamento

A Figura 5.3 agrega os resultados da simulação com preferências fixas de compra de produtos de equipamento com $mfpe = 0.8$.

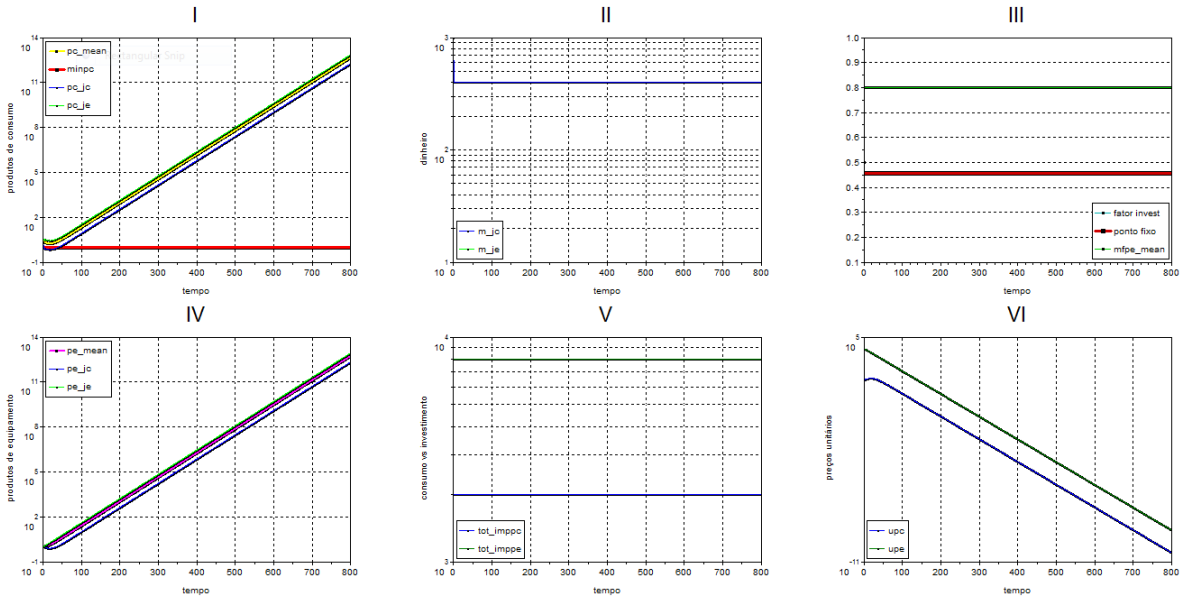


Figura 5.3: Resultados da simulação para $mfpe = 0.8$.

Dos resultados ilustrados na Figura 5.3 pode inferir-se que o valor do factor de investimento é constante e sempre superior ao valor do ponto fixo (Figura 5.3, gráfico III). Consequentemente os preços unitários decrescem de forma constante ao longo do tempo (Figura 5.3, gráfico VI). Desta forma a quantidade de produtos de equipamento e de consumo aumenta para os dois tipos de agente (consultar gráficos I e IV da Figura 5.3).

Porém, é de realçar que os agentes atingem valores de produtos de consumo inferiores ao mínimo de subsistência nos instantes iniciais, infringindo desta forma o critério imposto pela equação 2.1.

Dado que os valores de preferências são fixos a quantidade de dinheiro nos agentes será constante, (Figura 5.3, gráfico II), assim como os valores totais investidos em cada tipo de produto (Figura 5.3, gráfico V).

5.2.3.2 Tendência para comprar mais produtos de consumo

A Figura 5.4 compila os resultados de simulação com $mfpe = 0.2$.

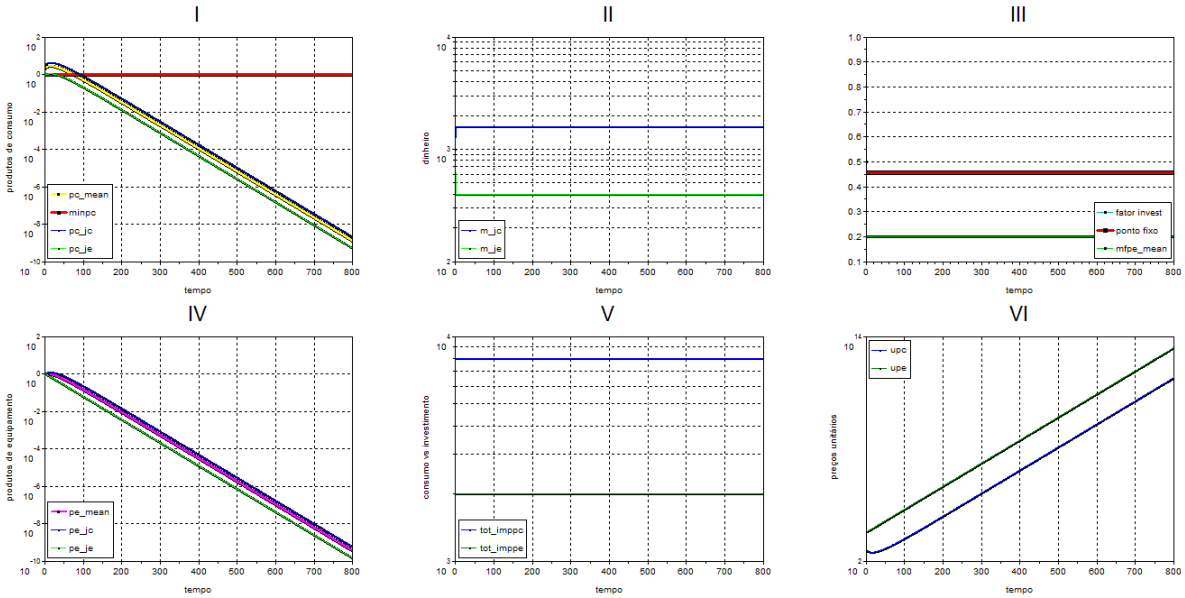


Figura 5.4: Resultados da simulação para $mfpe = 0.2$.

O valor do factor de investimento é sempre inferior ao do ponto fixo para os produtos de equipamento (Figura 5.4, gráfico III). Desta forma os preços unitários aumentam exponencialmente (Figura 5.4, gráfico VI), as quantidades de produtos de consumo e de equipamento decrescem ao longo do tempo para todos os agentes (Figura 5.4, gráficos I e IV). Da observação do gráfico I constata-se que em poucas iterações os agentes atingem valores inferiores aos do seu limite de subsistência.

5.2.3.3 Distribuição equitativa das preferências de compra

A Figura 5.5 contém os resultados da simulação com $mfpe = 0.5$.

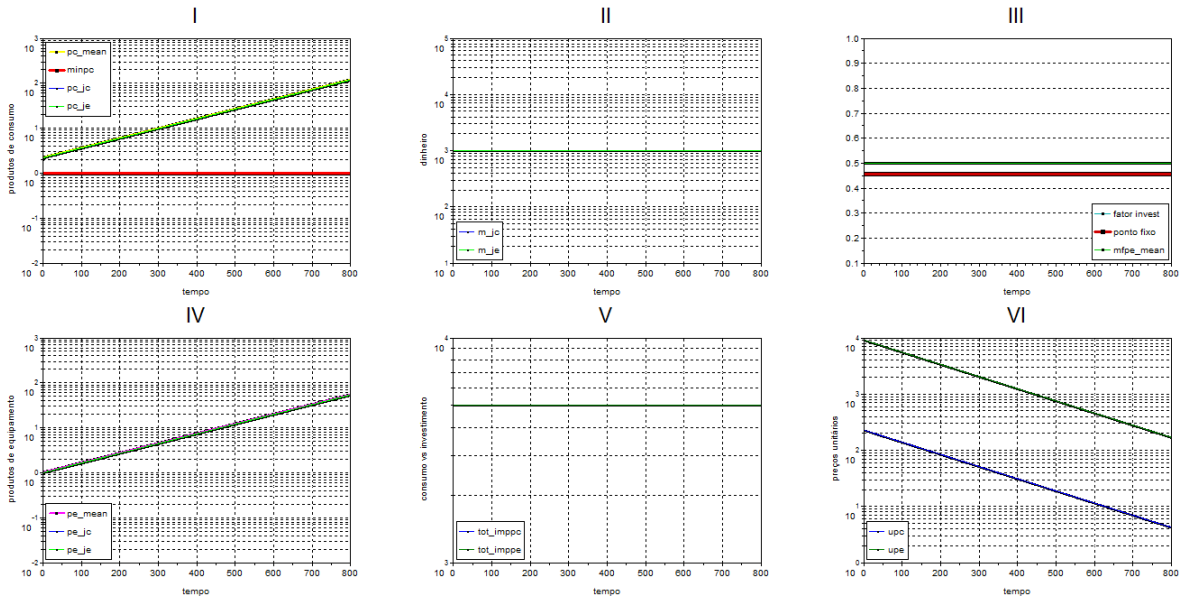


Figura 5.5: Resultados da simulação para $mfpe = 0.5$.

O factor de investimento é ligeiramente superior ao ponto fixo; mantendo uma tendência para os preços unitários decrescerem ao longo do tempo (Figura 5.5, gráfico VI). Verificando-se a tendência de queda dos preços, constata-se que a quantidade de produtos de consumo e equipamento aumentam (Figura 5.5, gráficos I e IV). Todavia, o crescimento não é tão acentuado como o verificado na configuração em que se colocou $mfpe = 0.8$. Não obstante, os agentes ficam sempre acima do mínimo de subsistência (Figura 5.5, gráfico I).

De sublinhar que esta configuração cumpre os dois critérios impostos pelo modelo (2.1 e 2.2).

5.3 Modelo com RL

No modelo provido de RL (consultar o código no apêndice A.5) foram estudadas as consequências na economia da sociedade resultantes da variação dos parâmetros do algoritmo RL implementado: α, γ, ϵ. De relembrar que α é o parâmetro que representa a taxa de aprendizagem. Este determina o modo como o agente

acumula a toda a nova informação recebida. Para um valor de α igual a 0 o agente não aprende, para um valor unitário de α o agente toma em conta toda a informação possível. γ é o parâmetro do *Q-learning* responsável por determinar a importância das recompensas futuras. Um γ igual a 0 torna o agente oportunista, em que só dá importância aos valores das recompensas imediatas; por oposição, um γ próximo de 1 proporciona ao agente um desconto muito pequeno do seu futuro. ϵ, no contexto de uma estratégia ϵ-*greedy*, é a probabilidade de um agente explorar uma acção não óptima. Isto é, $(1 - \epsilon)$ corresponde à probabilidade do agente seguir a acção óptima.

Com a experimentação do modelo dotado de racionalidade, concluiu-se que um dos factores mais importantes e determinantes no sucesso dos agentes reside na escolha do valor de ϵ. Isto é, a forma como a sociedade evolui com agentes mais exploradores ou que seguem a acção que lhes garante (à partida) maior retorno. No estudo da variação de ϵ utilizou-se $\alpha = 0.5$ e $\gamma = 0.9$.

Outro factor determinante é a inicialização da matriz Qsa. Todos os agentes são iguais, isto é, dentro de cada classe de agente todos partilham as mesmas características de mercado, e no global, toda a sociedade é dotada com o mesmos parâmetros RL. A matriz Qsa é inicializada de forma aleatória com valores entre 0 e 1 de forma a garantir variedade na escolha de acções, dado que, à partida os agentes não vão ter todos a mesma acção *greedy* para seguir. Não obstante, os agentes que têm a "sorte" de ter a acção *greedy* "correcta" partirão em vantagem em relação aos restantes. Todo o sucesso ou insucesso dos agentes (nos instantes iniciais) depende da *seed* com que foi inicializada a matriz Qsa. Para todas as simulações, a *seed* foi a mesma.

Com estas premissas presentes considerou-se que se podem estudar comportamentos de agentes mais exploradores ou menos exploradores com maior tendência para dar importância à inovação ou não. Achou-se interessante associar o modo como o agente desconta o seu futuro à forma como dá importância à informação adquirida. Definiu-se que um agente com uma maior tendência para dar im-

portância à informação recebida nos instantes de tempo mais recentes, $\alpha = 0.7$, também iria descontar menos o seu futuro, $\gamma = 0.9$. Por outro lado o agente com menor tendência para dar importância à informação recebida, $\alpha = 0.3$, iria descontar mais o seu futuro, $\gamma = 0.1$.

Em todas as simulações realizadas com RL os agentes partiram de preferências de compra iniciais iguais.

Um aspecto relevante a ter em conta prende-se com a implementação de um filtro nas opções dos agentes.

5.3.1 Filtro implementado

De recordar que na definição do jogo (na secção 4.3) deste modelo determinou-se que o agente teria à sua disposição 3 acções possíveis, que seriam incrementar, decrementar ou manter as preferências de compra por produtos de consumo ($mfpc$). Com o objectivo de suavizar a alteração entre escolha de acções implementou-se um filtro, cujo código se apresenta no bloco de código 5.1:

Bloco de código 5.1: Implementação do filtro nas opções dos agentes

```
if option(j,k)==1 then
mfpc(j,k)=0.95*mfpc(j,k-1)+0.05;          //incrementa
elseif option(j,k)==3 then
mfpc(j,k)=0.95*mfpc(j,k-1);                          //decrementa
else mfpc(j,k)=mfpc(j,k-1);
end
```

O objectivo do filtro (5.1) é de aproximar a tomada de decisão ao comportamento humano. De uma forma genérica, os seres humanos não incrementam as suas preferências de uma forma iterativa. O filtro pretende proporcionar incrementos e decrementos suaves.
Por exemplo, na Figura 5.6 pode observar-se a simulação da resposta do filtro no sentido de incrementar $mfpc$ a partir do valor inicial de 0.5.

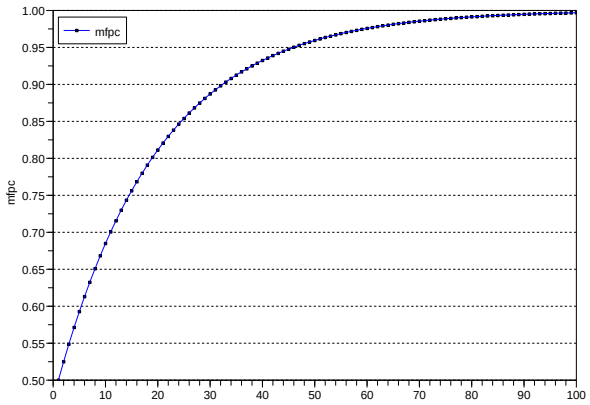


Figura 5.6: Simulação do comportamento do filtro com incrementos.

Na Figura 5.7 pode observar-se a simulação da resposta do filtro no sentido de decrementar $mfpc$ a partir do valor inicial de 0.5.

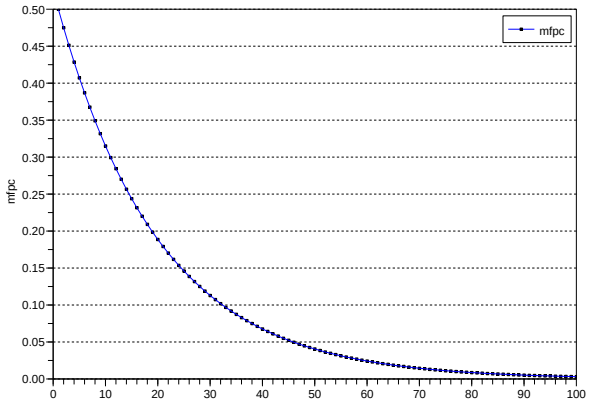


Figura 5.7: Simulação do comportamento do filtro com decrementos.

5.3.2 Estudo da influência da exploração/não-exploração

Como anteriormente mencionado, na secção 4.4, estudaram-se diferentes combinações no funcionamento do algoritmo RL. Utilizou-se a quantidade de produtos de consumo, *pc*, como recompensa, e o dinheiro na posse dos agentes, m. Para determinar o estado utilizou-se o *pc* e o respectivo valor médio, pc_mean e o *pe* e respectivo valor médio, pe_mean. Como está descrito na tabela 5.1, utilizaram-se três configurações: configuração A, em que a quantidade de produtos de consumo na posse dos agentes, *pc*, é utilizada para determinar o estado e é recompensa; a configuração B, em que a quantidade de produtos de consumo na posse dos agentes *pc* é utilizada para determinar o estado e o dinheiro,m é recompensa; e por último, a configuração C, em que a quantidade de produtos de equipamento na posse dos agentes, *pe*, são utilizados na determinação do estado e a recompensa é o dinheiro na posse dos agentes, m.

A escolha das diferentes configurações tem como base a experimentação do modelo com RL. Inicialmente o objectivo seria maximizar a quantidade de produtos de consumo na posse dos agentes, portanto seria lógico utilizar *pc* como recompensa. Todavia, percebeu-se que tal não acontecia e verificou-se melhores resultados com a quantidade de dinheiro, m. A determinação do estado a partir

da quantidade de produtos de equipamento na posse dos agentes, *pe*, serviu como comparação entre configurações.

Tabela 5.1: Tipos de configuração utilizados no estudo do modelo com RL

	estado	recompensa
Configuração A	pc	pc
Configuração B	pc	m
Configuração C	pe	m

O objectivo do estudo da influência da exploração é verificar em medida a quantidade de tempo que exploram influencia as decisões dos agentes, bem como o comportamento de toda a sociedade. De recordar que ϵ é o parâmetro do RL que determina a percentagem de tempo que um dado agente explora; e que o comportamento da sociedade é avaliado em função dos critérios comportamentais.

5.3.2.1 Configuração A

Na Figura 5.8 podem consultar-se os resultados da simulação para $\epsilon = 0.1$, $\alpha = 0.5$ e $\gamma = 0.9$.

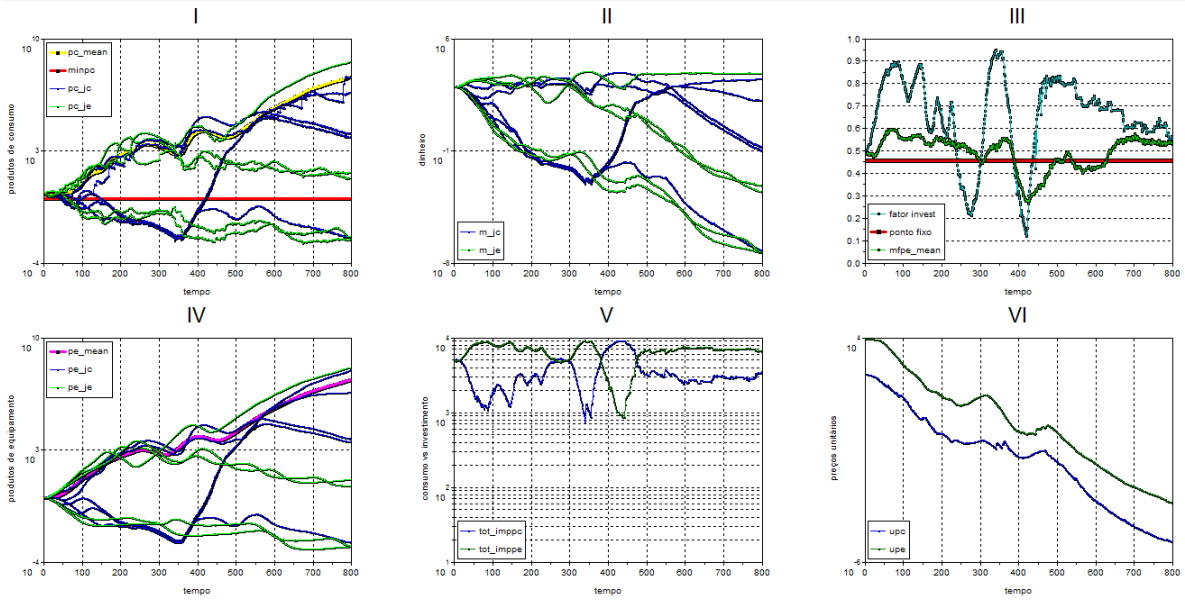


Figura 5.8: Resultados de simulação da configuração A com $\epsilon = 0.1$, $\alpha = 0.5$ e $\gamma = 0.9$.

Nesta configuração vários agentes atingem valores inferiores ao nível de subsistência, como pode ser observado na Figura 5.8, no gráfico I.

O factor de investimento oscila entre valores muito elevados (perto da unidade) e valores próximos de 0 (Figura 5.8, gráfico III). Verifica-se que sempre que o factor de investimento atinge valores inferiores ao ponto fixo há uma subida dos preços unitários (Figura 5.8, gráficos III e VI).

A partir da observação do gráfico IV pode inferir-se que os agentes que nos instantes iniciais investiram mais em equipamento mantêm uma posição de supremacia ao longo do tempo. Por outro lado, os que não investiram em equipamento ficam na "miséria". Com a excepção de um dos agentes produtores de bens de consumo que a partir do instante de tempo 350 inverte as suas preferências de compra, conseguindo recuperar; porém este obteve valores abaixo do limite da sua subsistência. É interessante observar o seu comportamento sob o ponto de vista de funcionamento do modelo, dado que na prática o agente ressuscitou para uma posição de comando. O mesmo agente segue claramente a tendência de toda

a sociedade que é imposta pelo factor de investimento (gráfico III).

Da análise do gráfico II conclui-se que a maioria do dinheiro esta na posse dos agentes que mais produtos têm, enquanto os que estão abaixo do limite de subsistência atingem valores de dinheiro na ordem de 10^{-9}.

Na Figura 5.9 podem consultar-se os resultados da simulação para $\epsilon = 0.3$, $\alpha = 0.5$ e $\gamma = 0.9$.

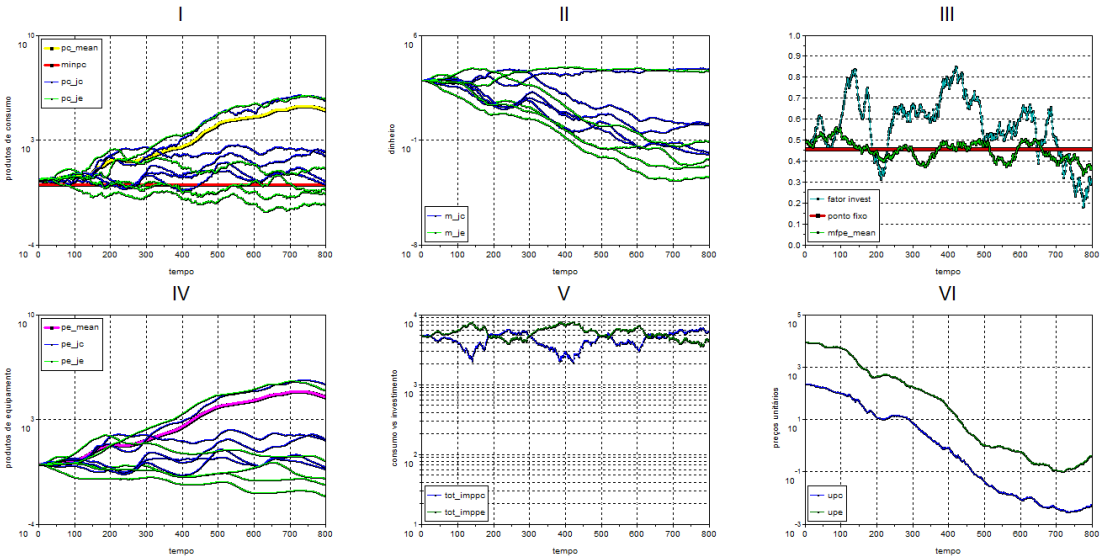


Figura 5.9: Resultados de simulação da configuração A com $\epsilon = 0.3$, $\alpha = 0.5$ e $\gamma = 0.9$.

Nesta configuração constata-se que há um decréscimo da quantidade de produtos de consumo na posse dos agentes. Há menos agentes a atingir valores inferiores ao limite de subsistência, sendo a maioria produtores de bens de equipamento (consultar gráfico I da Figura 5.9). Mais uma vez pode relacionar-se com as opções inicias tomadas, dado que correspondem aos agentes que escolhem ter menos equipamento nos instantes iniciais (consultar Figura 5.9, gráficos I e IV). Da análise em conjunto dos gráficos I e IV da Figura 5.9 conclui-se que dois agentes possuem praticamente todos os produtos da sociedade, um produtor de bens de equipamento e um outro produtor de bens de consumo.

A evolução dos preços unitários acompanha, naturalmente, a tendência imposta pelo factor de investimento (Figura 5.9).

A quantidade de dinheiro na posse dos agentes é menos desequilibrada, porém dois agentes estão na posse de praticamente todo o dinheiro da sociedade (Figura 5.9, gráfico II).

De um modo geral, a sociedade tem uma menor riqueza agregada, mas há maior igualdade entre agentes (com excepção dos dois "milionários"). Por conseguinte, não fica garantido o aumento de riqueza ao longo do tempo.

Na Figura 5.10 podem consultar-se os resultados da simulação para $\epsilon = 0.5$, $\alpha = 0.5$ e $\gamma = 0.9$.

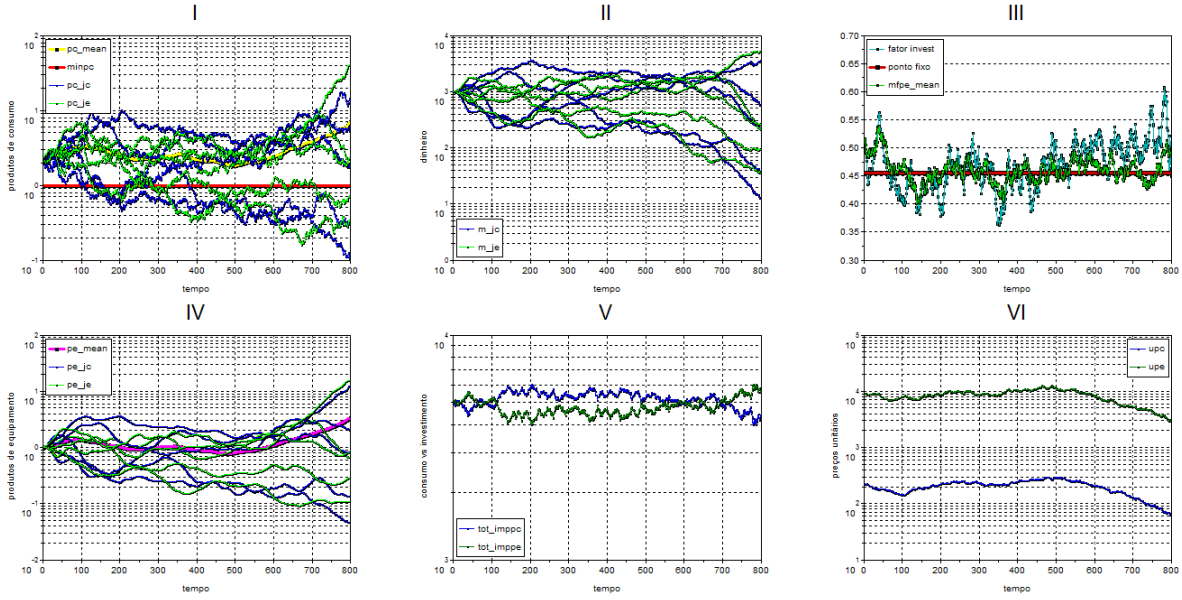


Figura 5.10: Resultados de simulação da configuração A com $\epsilon = 0.5$, $\alpha = 0.5$ e $\gamma = 0.9$.

Comparativamente aos estudos anteriores, verifica-se um maior equilíbrio entre os agentes (Figura 5.10, gráficos I, II, IV e V), mas de realçar que se trata de um equilíbrio na "miséria". Nenhum agente ganha muito mais que os restantes, porém a sociedade possui menos produtos que nas variações de ϵ anteriores (Figura 5.10,

gráficos I e IV). Metade dos agentes não se consegue manter acima do limite de subsistência.
É possível explicar este acontecimento se se analisar o gráfico III da Figura 5.10. Do instante 100 até ao instante 600 o factor de investimento ronda o valor do ponto fixo fazendo com que os preços unitários se mantenham praticamente constantes. Só quando o valor do factor de investimento é superior ao do ponto fixo é que os preços unitários descem (Figura 5.10, gráficos III e VI) tornando possivel a aquisição de uma maior quantidade de produto (Figura 5.10, gráficos I e IV).
Pode concluir-se que com o aumento da exploração/escolha da acção não óptima há uma maior tendência para a igualdade na sociedade, todavia há um decréscimo de riqueza.

Em nenhuma das variações de ϵ estudadas nesta configuração A fica garantido o aumento de riqueza ao longo do tempo, equação 2.2, nem que os agentes fica acima de um limite de subsistência, equação 2.1.

5.3.2.2 Configuração B

Da análise desta configuração inferiu-se que a tendência de evolução em relação à exploração é semelhante à configuração A. Não obstante, vai mostrar-se apenas os resultados para $\epsilon = 0.1$ e $\epsilon = 0.5$.

De relembrar que a grande diferença entre a configuração A e B reside na *reward*. Na configuração B o *reward* é o dinheiro, m.

Na Figura 5.11 podem consultar-se os resultados da simulação para $\epsilon = 0.1$, $\alpha = 0.5$ e $\gamma = 0.9$.

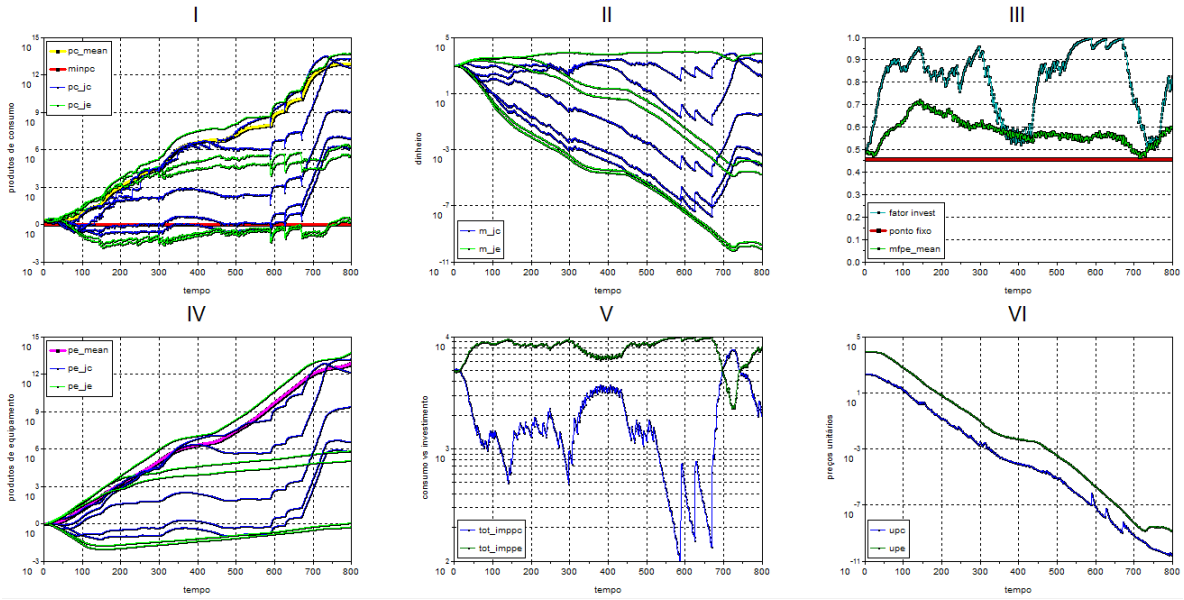


Figura 5.11: Resultados de simulação da configuração B com $\epsilon = 0.1$, $\alpha = 0.5$ e $\gamma = 0.9$.

À semelhança dos resultados ilustrados na Figura 5.8, na Figura 5.11 constata-se que há uma grande quantidade de produtos na posse dos agentes. Todavia, quatro agentes atingem valores bem abaixo do limite de subsistência, dois de cada tipo (Figura 5.11, gráfico I).

Verifica-se um grande desequilíbrio na distribuição de riqueza. Praticamente todos os produtos e dinheiro estão na posse de três agentes (Figura 5.11, gráficos I, II e IV).

A tendência dos preços unitários é decrescer ao longo do tempo, com excepção no instantes de 350 a 450 em que se mantém e de 750 a 800. Nestes instantes o valor do factor de investimento aproxima-se dos valores das preferências de compra de bens de equipamento e ambos do valor do ponto fixo (Figura 5.11, gráficos III e VI).

Na Figura 5.12 podem consultar-se os resultados da simulação para $\epsilon = 0.5$.

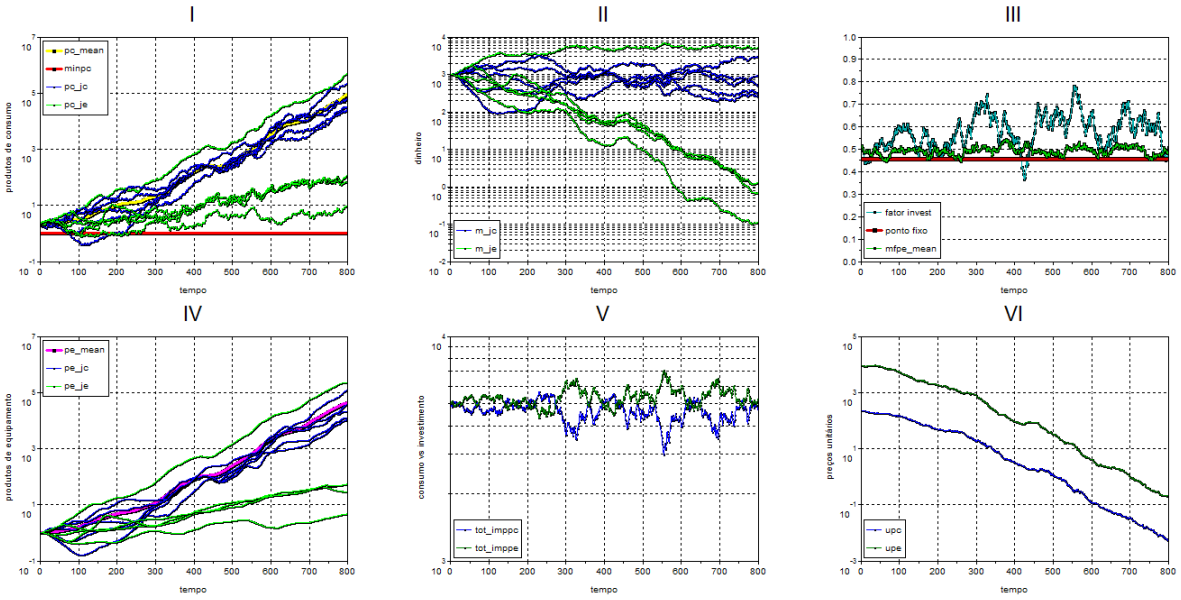


Figura 5.12: Resultados de simulação da configuração B com $\epsilon = 0.5$, $\alpha = 0.5$ e $\gamma = 0.9$.

Nesta simulação apenas um agente desceu abaixo do limite de subsistência. A quantidade de produtos aumentou ao longo do tempo. Uma situação interessante é que o agente que mais sai beneficiado é um produtor de equipamento, todavia toda a classe de agentes produtores de bens de consumo se mantém bastante equilibrada e em crescimento. Por outro lado, os agente que ganham menos são produtores de bens de equipamento (Figura 5.12, gráficos I e IV).

O dinheiro fica concentrado na classe de agentes produtores de bens de consumo e no agente produtor de equipamento que já se tinha destacado na posse de produtos (Figura 5.12, gráfico II).

Os resultados assemelham-se aos do modelo com preferências fixas. De notar que os valores das preferências dos agentes rondam os 50%, isto é, $mfpe = 0.5$ (Figura 5.5).

Como o factor de investimento se mantém próximo, mas acima do valor do ponto fixo os preços unitários decrescem ao longo do tempo (Figura 5.12, gráficos III e VI). Os resultados obtidos com a variação de ϵ na configuração B não satis-

fazem o critério imposto pela equação 2.1, *ipsis verbis*, visto não ser garantido que os agentes se mantenham acima do limite de subsistência. Para $\epsilon = 0.1$ também não é cumprido o critério imposto pela equação 2.2, porém para $\epsilon = 0.5$ já há um aumento da riqueza na sociedade, ténue, mas efectivo.

5.3.2.3 Configuração C

Na Figura 5.13 podem consultar-se os resultados da simulação para $\epsilon = 0.1$, $\alpha = 0.5$ e $\gamma = 0.9$.

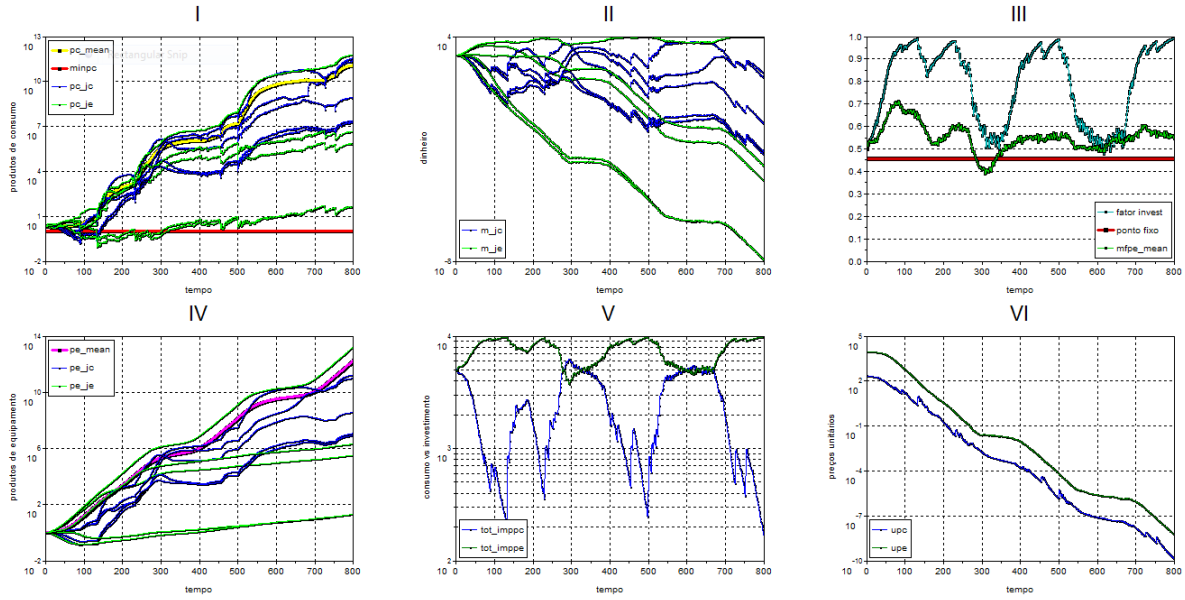


Figura 5.13: Resultados de simulação da configuração C com $\epsilon = 0.1$, $\alpha = 0.5$ e $\gamma = 0.9$.

Na Figura 5.14 podem consultar-se os resultados da simulação para $\epsilon = 0.5$.

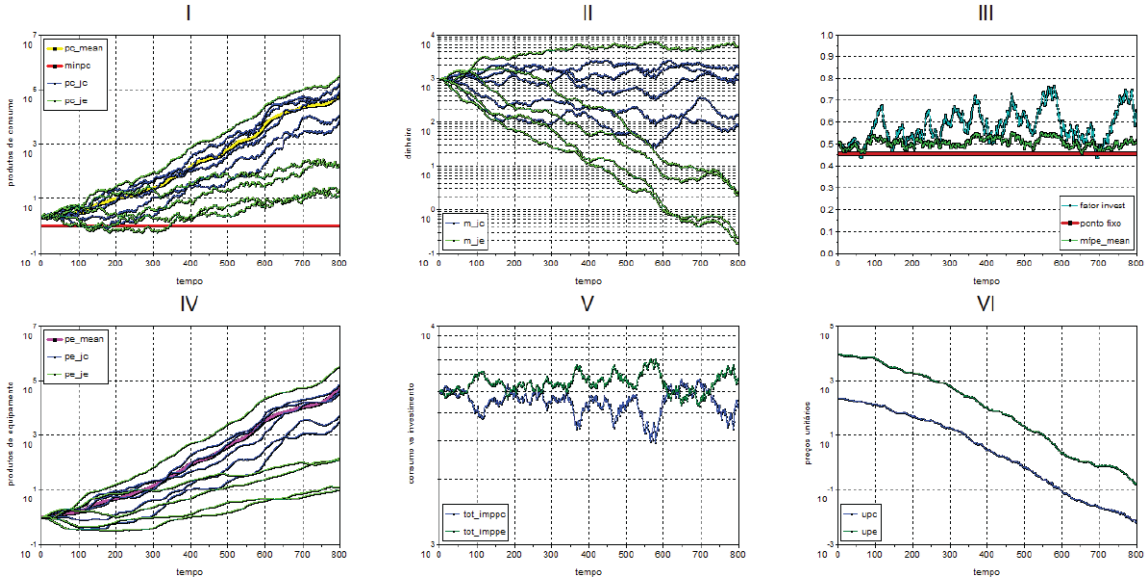


Figura 5.14: Resultados de simulação da configuração C com $\epsilon = 0.5$, $\alpha = 0.5$ e $\gamma = 0.9$.

Os resultados obtidos nas simulações referentes à configuração C (figuras 5.13 e 5.14) são muito semelhantes aos da obtidos na configuração B (figuras 5.11 e 5.12).

Note-se que quando o dinheiro é recompensa é menor o número de agentes que atingem valores inferiores ao do nível de subsistência.

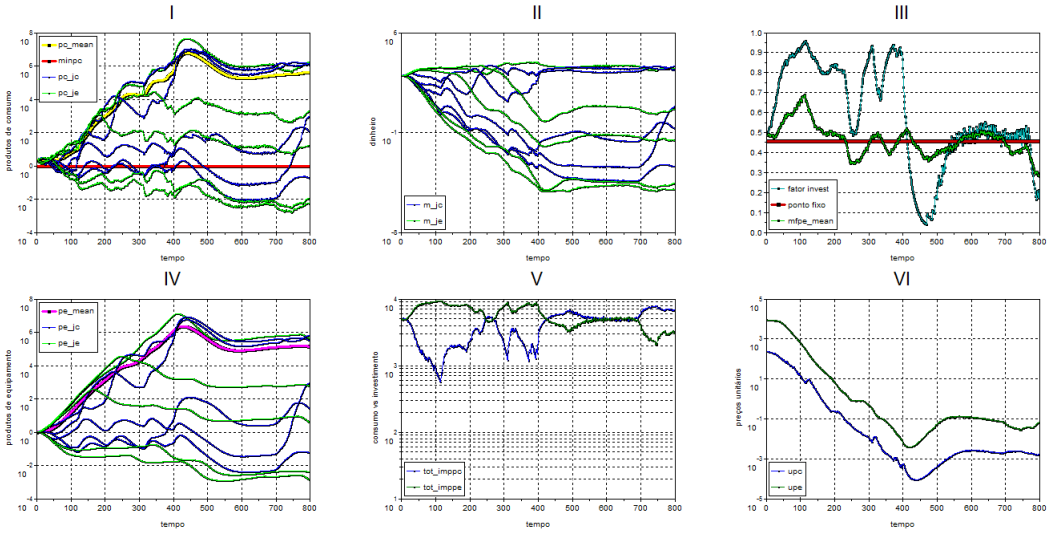
5.3.3 Influência da aquisição de informação e desconto do futuro

O estudo da influência da aquisição de informação e desconto do futuro tem como objectivo verificar a influência dos parâmetros γ e α do RL. Assim, para valores constantes de exploração, ϵ, variam-se os valores de α, que influencia a forma como um agente acumula a informação recebida, e de γ, que determina a importância

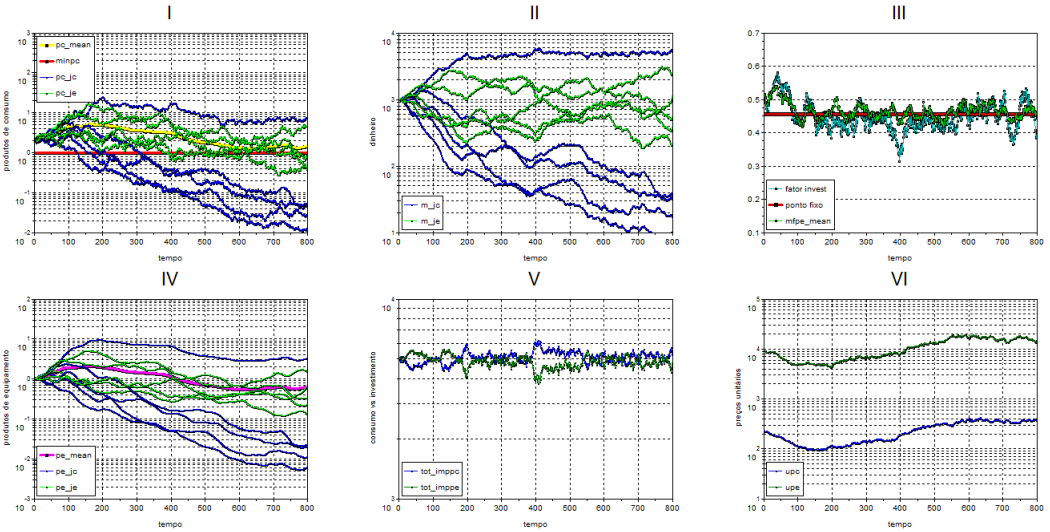
das recompensas futuras obtidas pelo agente.

5.3.3.1 Configuração A

Na Figura 5.15a podem consultar-se os resultados da simulação para um agente com maior tendência para dar importância à informação mais recente e que desconta menos o futuro ($\gamma = 0.9$, $\alpha = 0.7$) com $\epsilon = 0.1$. Na Figura 5.15b podem consultar-se os resultados de simulação para um agente com maior tendência para dar importância à informação mais recente e que desconta menos o ($\gamma = 0.9$, $\alpha = 0.7$) com $\epsilon = 0.5$.



(a)



(b)

Figura 5.15: Resultados de simulação da configuração A com $\alpha = 0.7$ e $\gamma = 0.9$:
(a) Com $\epsilon = 0.1$;
(b) Com $\epsilon = 0.5$.

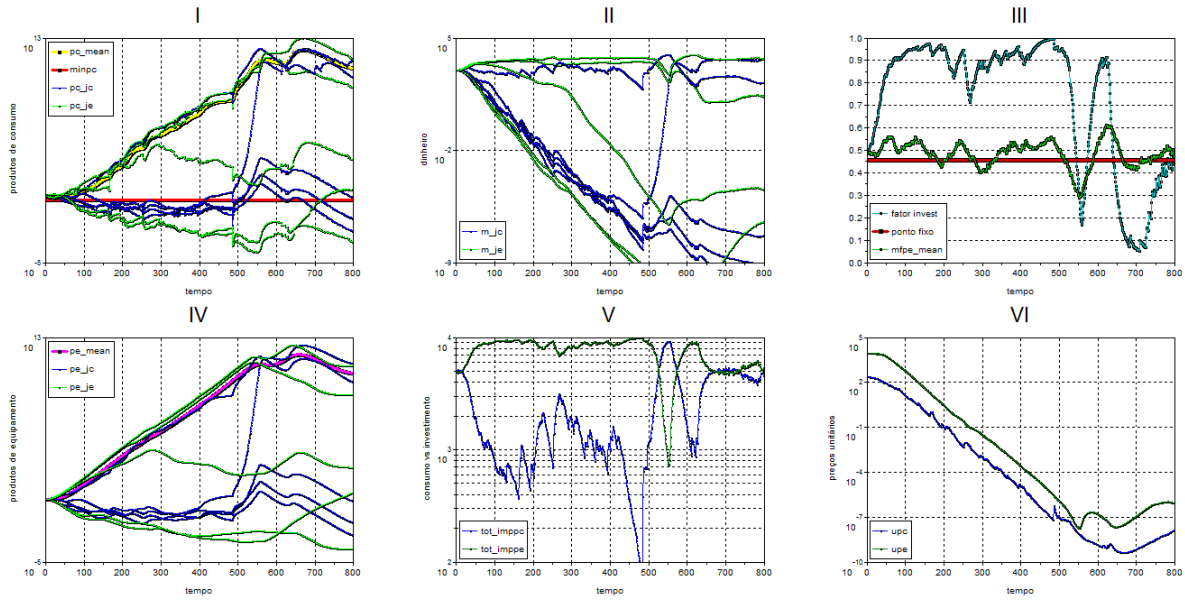
As oscilações no mercado estão directamente relacionadas com o factor de investimento. Sempre que este se mantém acima do valor do ponto fixo os preços descem. Sempre que baixa o valor do factor de investimento os preços sobem. Na fronteira verifica-se a tendência para estes manterem (Figuras 5.15 (a) e (b),

gráficos III e IV).

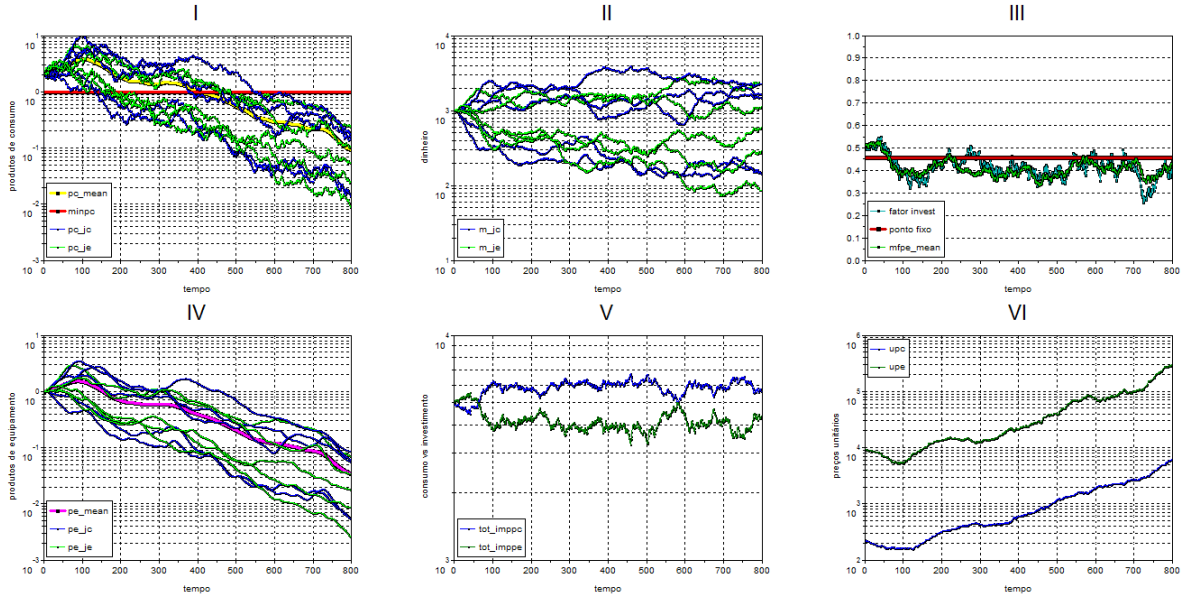
Na configuração em que os agentes exploram menos (Figura 5.15a) há um crescimento da quantidade de produtos na posse dos agentes até ao instante de tempo 450 (gráficos I e IV). A partir daí revela-se uma tendência para manter os valores. Esse fenómeno deve-se ao facto do decréscimo do factor de investimento nesse instante. Apesar dos agentes "cometerem erros", os valores agregados permitem-lhes manter o nível, pelo menos para os agentes com mais produtos na sociedade (Figura 5.15, gráficos I,III e IV). Volta a verificar-se a tendência de agentes atingirem valores inferiores ao dos limite de subsistência, com destaque para os produtores de equipamento. E mais uma vez o dinheiro concentra-se nos "líderes" da sociedade.

Na configuração em que os agentes exploram mais, Figura 5.15b, há um decréscimo na quantidade de produtos de consumo disponíveis, ficando o valor médio dos produtos de consumo muito perto do limite de subsistência. Praticamente todos os produtores de bens de consumo "caem na miséria", com a excepção de um agente que acaba por se tornar o mais rico da sociedade, embora com quantidades de produtos muito menores dos obtidos na simulação com agentes menos exploradores (Figura 5.15a).

Na Figura 5.16a podem consultar-se os resultados da simulação para um agente com uma menor tendência para dar importância à informação adquirida e que desconta menos o seu futuro ($\gamma = 0.1$, $\alpha = 0.3$) com $\epsilon = 0.1$. Na Figura 5.16b) podem consultar-se os resultados da simulação para um agente com uma menor tendência para dar importância à informação adquirida e que desconta mais o seu futuro ($\gamma = 0.1$, $\alpha = 0.3$) com $\epsilon = 0.5$.



(a)



(b)

Figura 5.16: Resultados de simulação da configuração A com $\alpha = 0.3$ e $\gamma = 0.1$:
(a) Com $\epsilon = 0.1$;
(b) Com $\epsilon = 0.5$.

Os resultados obtidos nas configurações correspondentes às Figuras 5.15 e 5.16 a são muito semelhantes. Há um aumento de produtos na posse dos agentes em

ambas (gráficos I e IV, Figuras 5.15 e 5.16a). Os preços unitários decrescem a um ritmo praticamente constante (gráfico VI, Figuras 5.15 e 5.16).

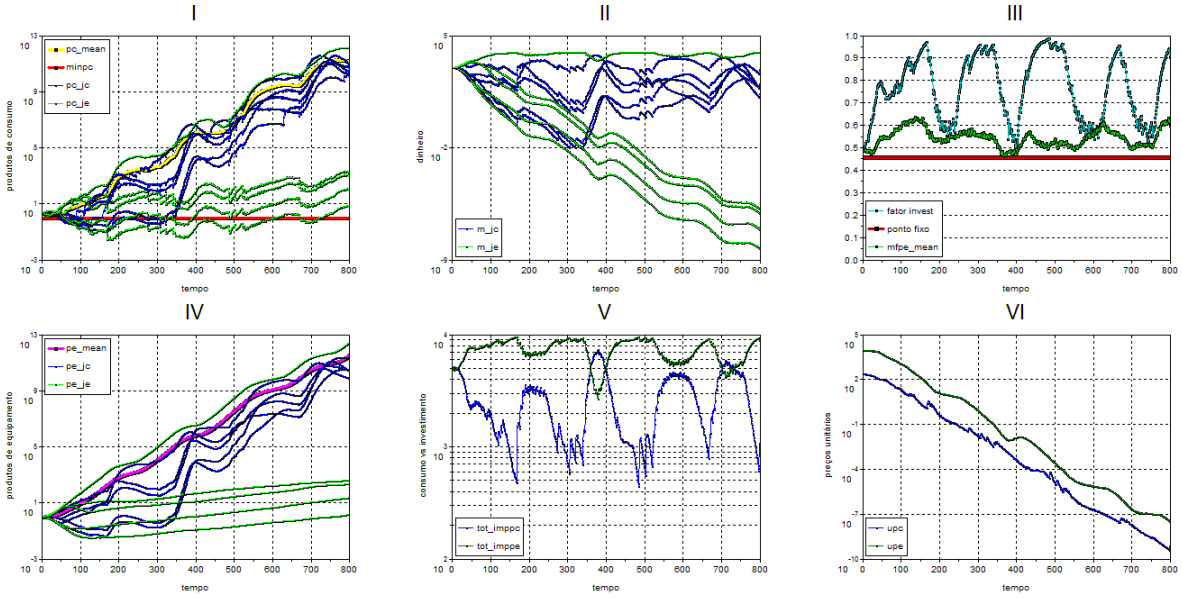
Na Figura 5.15b consta-se que há um crescimento constante da quantidade de produtos na posse dos agentes. Contudo, é mais equilibrado no observado na Figura 5.15a. Há um maior equilíbrio, porém há uma menor quantidade total de produtos na sociedade comparativamente à Figura 5.15a.

Na Figura 5.16b todos os agentes entram em ruptura (Figura 5.16, gráficos I e IV). O factor de investimento atinge valores inferiores aos do ponto fixo constantemente (Figura 5.16b, gráfico III) o que implica que os preços unitários cresçam ao longo do tempo (Figura 5.16, gráfico VI).

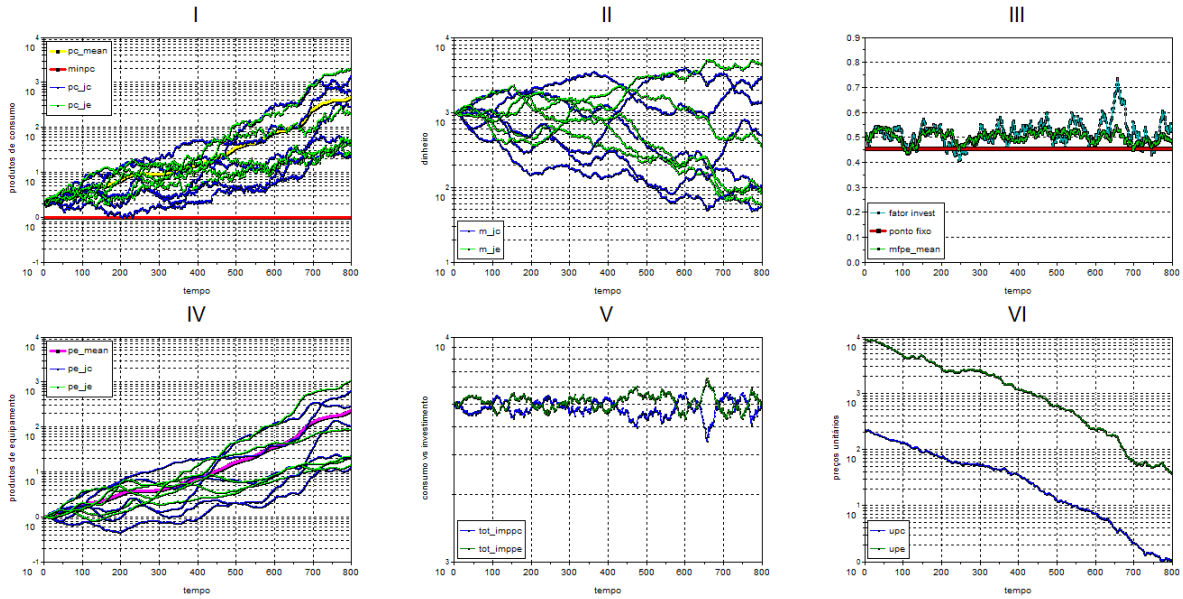
Os agentes que dão menos importância à informação recente e que descontam menos o futuro ficam muito prejudicados nestas configurações.

5.3.3.2 Configuração B

Na Figura 5.17a podem consultar-se os resultados da simulação para um agente com maior tendência para dar importância à informação mais recente e que desconta o futuro de uma forma mais eficaz ($\gamma = 0.9$, $\alpha = 0.7$) com $\epsilon = 0.1$. Na Figura 5.17b podem consultar-se os resultados da simulação para um agente com maior tendência para dar importância à informação mais recente e que desconta o futuro de uma forma mais eficaz ($\gamma = 0.9$, $\alpha = 0.7$) com $\epsilon = 0.5$.



(a)



(b)

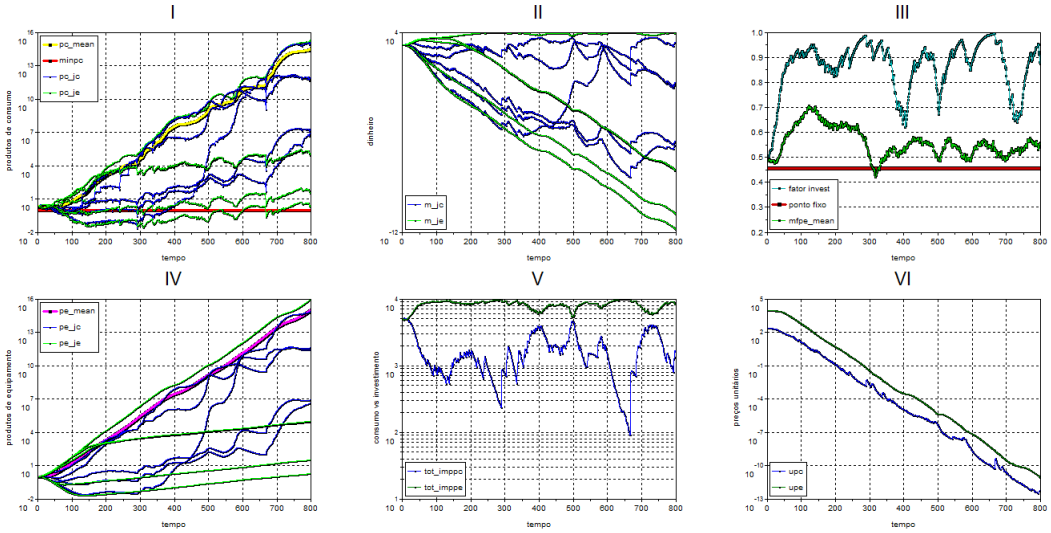
Figura 5.17: Resultados de simulação da configuração B com $\alpha = 0.7$ e $\gamma = 0.9$:
(a) Com $\epsilon = 0.1$;
(b) Com $\epsilon = 0.5$.

Os resultados obtidos nesta configuração são muito semelhantes aos obtidos no estudo da variação do ϵ para esta mesma configuração (consultar Figura 5.11

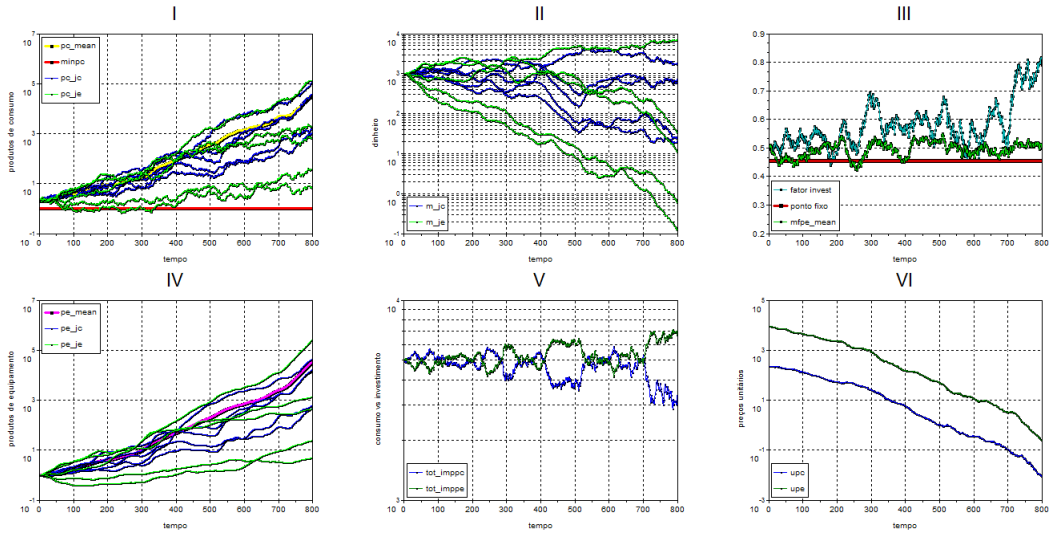
e 5.12).

A variação dos parâmetros γ e α não traz muita novidade, com a excepção de se verificar que todos os agentes conseguem manter quantidades de produto que lhes permitem estar acima do limite de subsistência.

Na Figura 5.18a podem consultar-se os resultados da simulação para agentes com uma menor tendência para dar importância à informação adquirida e que desconta de uma forma pouco eficaz o seu futuro($\gamma = 0.1$, $\alpha = 0.3$) com $\epsilon = 0.1$. Na Figura 5.18b podem consultar-se os resultados da simulação para agentes com uma menor tendência para dar importância à informação adquirida e que desconta de uma forma pouco eficaz o seu futuro ($\gamma = 0.1$, $\alpha = 0.3$) com $\epsilon = 0.5$.



(a)



(b)

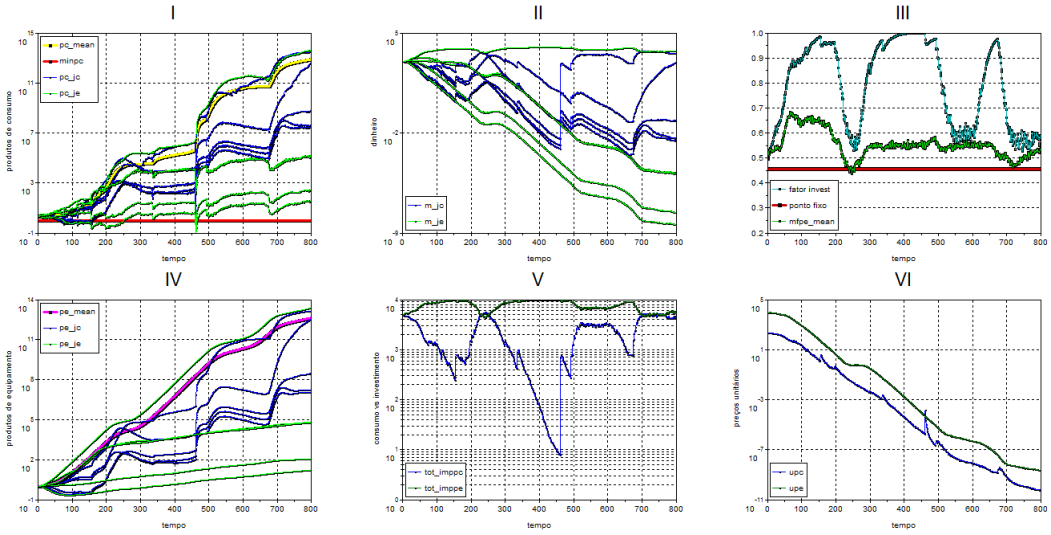
Figura 5.18: Resultados de simulação da configuração B com $\alpha = 0.3$ e $\gamma = 0.1$:
(a) Com $\epsilon = 0.1$;
(b) Com $\epsilon = 0.5$.

Os resultados nesta simulação são muito semelhantes para um agente com uma maior tendência para dar importância à informação mais recente (Figuras 5.17 (a) e 5.17)b. Também são muito semelhantes aos encontrados no estudo da variação de ϵ (Figuras 5.11 e 5.12).

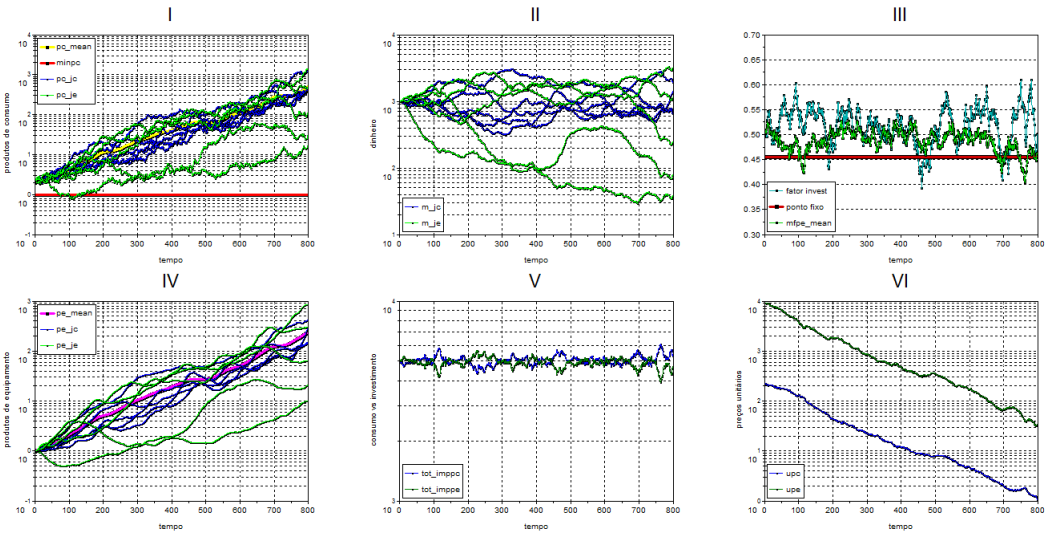
Pode concluir-se que, para este modelo, o factor que mais influencia é sem dúvida a **exploração** do agente. Quando os agentes exploram pouco obtém-se diferenças acentuadas entre eles, mas a riqueza aumenta. Quando os agentes exploram metade do seu tempo, há uma maior igualdade na sociedade e decréscimo da riqueza total.

5.3.3.3 Configuração C

Na Figura 5.19a podem consultar-se os resultados da simulação para um agente maior tendência para dar importância à informação mais recente e que desconta o futuro de uma forma mais eficaz ($\gamma = 0.9$, $\alpha = 0.7$) com $\epsilon = 0.1$. Na Figura 5.19b podem consultar-se os resultados da simulação para um agente com maior tendência para dar importância à informação mais recente e que desconta o futuro de uma forma mais eficaz ($\gamma = 0.9$, $\alpha = 0.7$) com $\epsilon = 0.5$.



(a)

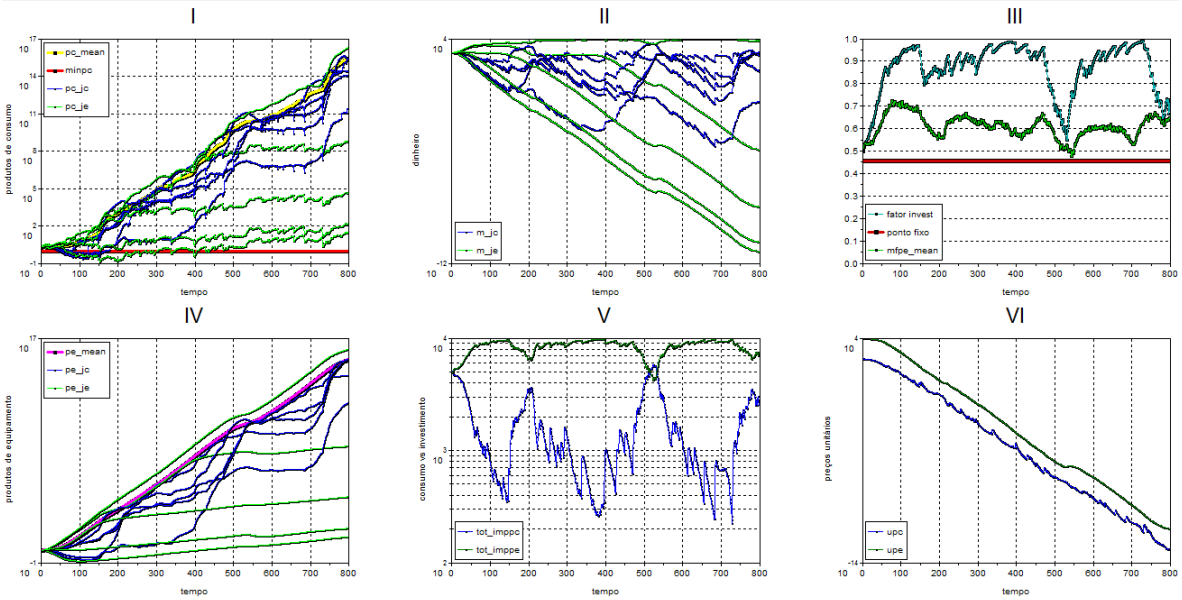


(b)

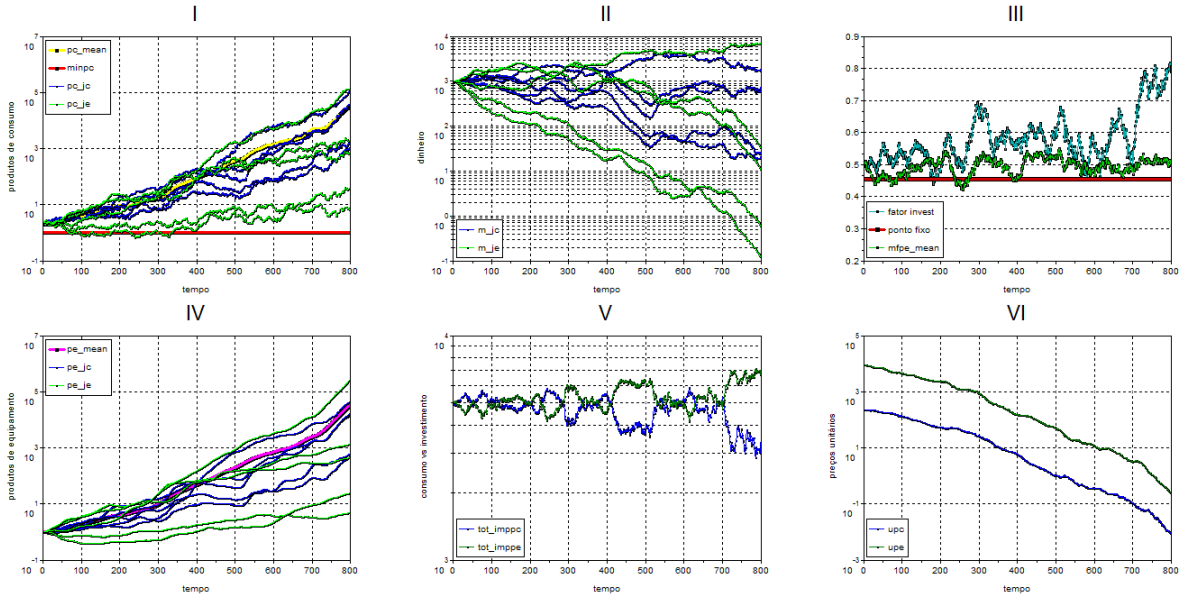
Figura 5.19: Resultados de simulação da configuração C com $\alpha = 0.7$ e $\gamma = 0.9$:
(a) Com $\epsilon = 0.1$;
(b) Com $\epsilon = 0.5$.

Na Figura 5.20 a podem consultar-se os resultados da simulação para um agente com uma menor tendência para dar importância à informação adquirida e que desconta de uma forma pouco eficaz o seu futuro ($\gamma = 0.1$, $\alpha = 0.3$) com $\epsilon = 0.1$. Na Figura 5.20 b podem consultar-se os resultados da simulação para um agente

com uma menor tendência para dar importância à informação adquirida e que desconta de uma forma pouco eficaz o seu futuro ($\gamma = 0.1$, $\alpha = 0.3$) com $\epsilon = 0.5$.



(a)



(b)

Figura 5.20: Resultados de simulação da configuração C com $\alpha = 0.3$ e $\gamma = 0.1$:

(a) Com $\epsilon = 0.1$;

(b) Com $\epsilon = 0.5$.

Mais uma vez, os resultados nesta configuração são muito semelhantes para os agentes (Figuras 5.19 a e 5.19b). Por conseguinte, são muito semelhantes aos encontrados no estudo da variação de ϵ (figuras 5.13 e 5.14).

Pode concluir-se que o factor preponderante do comportamento dos agentes é a **exploração**. Sendo ainda mais relevante, nos casos em que a recompensa é o dinheiro, m.

Na Tabela 5.2 encontra-se a síntese de resultados. As linhas que têm a mesma cor correspondem a configurações com evoluções e/ou resultados semelhantes.

Tabela 5.2: Síntese de resultados

Modelo em Estudo	Análises	Configurações	Parametrizações	Critérios Comportamento		Riqueza média total no instante $tsim$
				$pc \geq minpc$	$\frac{\Delta pc}{\Delta t} > 0$	$pc\_mean(tsim)$
Modelo com preferências fixas	n.a.	n.a.	$mfpe = 0.2$	✗	✗	$\simeq 10^{-8}$
	n.a.	n.a.	$mfpe = 0.5$	✓	✓	10^{2}
	n.a.	n.a.	$mfpe = 0.8$	✓	✓	$\simeq 10^{12}$
Modelo com RL	Análise da exploração	Configuração A	$\epsilon = 0.1$, $\alpha = 0.5$, $\gamma = 0.9$	✗	✗	$\simeq 10^{10}$
			$\epsilon = 0.5$, $\alpha = 0.5$, $\gamma = 0.9$	✗	✗	$\simeq 50$
		Configuração B	$\epsilon = 0.1$, $\alpha = 0.5$, $\gamma = 0.9$	✗	✗	$\simeq 10^{11}$
			$\epsilon = 0.5$, $\alpha = 0.5$, $\gamma = 0.9$	✗	✗	$\simeq 10^{6}$
		Configuração C	$\epsilon = 0.1$, $\alpha = 0.5$, $\gamma = 0.9$	✗	✗	$\simeq 10^{11}$
			$\epsilon = 0.5$, $\alpha = 0.5$, $\gamma = 0.9$	✗	✗	$\simeq 10^{5}$
	Análise da aquisição de informação e taxa de desconto	Configuração A	$\epsilon = 0.1$, $\alpha = 0.7$, $\gamma = 0.9$	✗	✗	$\simeq 10^{6}$
			$\epsilon = 0.5$, $\alpha = 0.7$, $\gamma = 0.9$	✗	✗	$\simeq 1$
			$\epsilon = 0.1$, $\alpha = 0.3$, $\gamma = 0.1$	✗	✗	$\simeq 10^{13}$
			$\epsilon = 0.5$, $\alpha = 0.3$, $\gamma = 0.1$	✗	✗	$\simeq 10^{-1}$
		Configuração B	$\epsilon = 0.1$, $\alpha = 0.7$, $\gamma = 0.9$	✗	✗	$\simeq 10^{11}$
			$\epsilon = 0.5$, $\alpha = 0.7$, $\gamma = 0.9$	✓	✗	$\simeq 10^{3}$
			$\epsilon = 0.1$, $\alpha = 0.3$, $\gamma = 0.1$	✗	✗	$\simeq 10^{13}$
			$\epsilon = 0.5$, $\alpha = 0.3$, $\gamma = 0.1$	✗	✗	$\simeq 10^{4}$
		Configuração C	$\epsilon = 0.1$, $\alpha = 0.7$, $\gamma = 0.9$	✗	✗	$\simeq 10^{14}$
			$\epsilon = 0.5$, $\alpha = 0.7$, $\gamma = 0.9$	✗	✗	$\simeq 10^{3}$
			$\epsilon = 0.1$, $\alpha = 0.3$, $\gamma = 0.1$	✗	✗	$\simeq 10^{15}$
			$\epsilon = 0.5$, $\alpha = 0.3$, $\gamma = 0.1$	✗	✗	$\simeq 10^{5}$

Capítulo 6

Conclusões e propostas de trabalho futuro

6.1 Conclusões

O modelo desprovido de racionalidade (secção 5.2) pode ser encarado como um sistema dinâmico em que se aplicam degraus às variáveis de decisão, $mfpe$ e $mfpc$ e observa-se de que forma evoluem as variáveis de estado, pc, pe e m. Analisando os resultados, pode concluir-se que para parametrizações do modelo que garantam que o factor de investimento de produtos de equipamento seja superior ao valor do ponto fixo, cumprem-se os critérios da inteligência colectiva (equações 2.1 e 2.2). Na prática, usando a parametrização utilizada neste trabalho, valores de preferências de produtos de equipamento iguais ou superiores a 0.5 garantem os referidos critérios. De sublinhar que no caso de $mfpe = 0.8$ obteve-se uma maior produtividade do que para o caso de $mfpe = 0.5$, porém o crescimento foi desigual. Os agentes produtores de bens de equipamento ficam a ganhar em relação aos produtores de bens de consumo.

O modelo em que se dotou os agentes de racionalidade pode ser visto como um sistema em que o algoritmo RL, através de *feedbacks* fornecidos pelas variáveis

de estado, permite que os agentes modifiquem os valores das suas preferências. A utilização do algoritmo *single Q* não resultou numa maximização de qualquer uma das *rewards* utilizadas (produtos de consumo ou dinheiro). A sociedade atingiu enormes desigualdades entre agentes; às vezes entrou em ruptura dado que os preços unitários nem sempre desciam implicando que a produtividade fosse reduzida; muitas das vezes agentes atingiram valores inferiores ao limite de subsistência. Em suma, dotar os agentes de racionalidade através do algoritmo *single Q* não garantiu o cumprimento dos critérios da inteligência colectiva deste modelo.

Sob o ponto de vista de análise social pode inferir-se que apesar de não serem cumpridos os critérios da inteligência colectiva, em algumas simulações com racionalidade implementada, se obtém valores de produtividade muito superiores aos obtidos nos estudos em que as preferências de compra são fixas, porém nesses casos a desigualdade ainda é mais acentuada. Será que é necessária a desigualdade para haver maior riqueza total numa sociedade? Não se tem a reposta, mas estabelecendo uma analogia com a sociedade real, é uma pergunta profundamente pertinente e igualmente sem resposta imediata.

6.2 Propostas de trabalho futuro

A melhoria do trabalho realizado passa pela criação de um algoritmo mais específico para o problema da sociedade colectiva. De salientar que o algoritmo Q é perfeitamente genérico e não permite a troca de informação entre agentes. Fará todo o sentido criar um algoritmo em que se partilhem informações e estratégias entre classes, produtores de equipamento e bens de consumo, bem como objectivos globais da sociedade, como os critérios de inteligência colectiva. Um algoritmo de inteligência artificial colectiva resolveria os problemas não solucionados pelo algoritmo utilizado.

Acrescentar outras classes de agentes complementaria a actual configuração da sociedade. Os tipos de agentes a acrescentar poderão ser:

- agente do tipo "banco central";
- agente do tipo "estado";
- agente do tipo "prestador de conhecimento".

O agente do tipo "banco central" será responsável por controlar a quantidade de dinheiro disponível para a sociedade. Em troca de produtos de equipamento fornece dinheiro. Será possível também fazer empréstimos.

O agente do tipo "estado" será responsável por regulamentar o funcionamento da sociedade. Cobrará impostos e irá aplicar o dinheiro recolhido na aquisição de produtos de interesse colectivo.

O agente do tipo "prestador de conhecimento" será responsável por introduzir um novo tipo de produto na sociedade: — o produto do tipo "conhecimento". O conhecimento acrescenta valor aos produtos. Desta forma poderá haver distinção entre um produto com maior base tecnológica e menor base tecnológica. Procurar-se-á quantificar, de alguma forma, o valor do conhecimento dentro da sociedade. A introdução destes agentes deverá ser progressiva e desenvolvido um algoritmo de inteligência específico para cada um deles.

Sob o ponto de vista de desenvolvimento de ferramentas para simulação e programação de sistemas multi-agente penso que se devem dar passos para a simplificação e optimização dos algoritmos de forma a ser possível a integração com aplicações *web*. O ideal seria estar pronto a tempo de integrar a próxima revolução na forma de abordar e navegar em páginas de *internet* com a actualização do HTML para a versão 5. Uma nova geração de produtos será criada, sendo o sistema multi-agente com simulação interactiva por *browsing*, um deles.

Bibliografia

Bowling, M. and Veloso, M. (2000). An analysis of stochastic game theory for multiagent reinforcement learning. Technical report, Computer Science Department, Carnegie Mellon University.

Bowling, M. and Veloso, M. (2001). Rational and convergent learning in stochastic games. *IN PROCEEDINGS OF THE SEVENTEENTH INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 1021—1026.

Bowling, M. and Veloso, M. (2002). Multiagent learning using a variable learning rate. *ARTIFICIAL INTELLIGENCE*, 136:215—250.

Busoniu, L., Babuska, R., and Schutter, B. D. (2008). A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(2):156–172.

Chalkiadakis, G. (2003). Multiagent reinforcement learning: Stochastic games with multiple learning players.

Claus, C. and Boutilier, C. (1998). The dynamics of reinforcement learning in cooperative multiagent systems. *IN PROCEEDINGS OF THE FIFTEENTH NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 746—752.

Eisenhardt, K. M. (1989). Agency theory: An assessment and review. *The Academy of Management Review*, 14(1):57–74. ArticleType: primary_article /

Full publication date: Jan., 1989 / Copyright © 1989 Academy of Management.

Florian, R. (2003). Autonomous artificial intelligent agents. Technical Coneural-03-01, Center for Cognitive and Neural Studies (Coneural).

Franklin, S. and Graesser, A. (1997). Is it an agent, or just a program?: A taxonomy for autonomous agents. In *ECAI '96: Proceedings of the Workshop on Intelligent Agents III, Agent Theories, Architectures, and Languages*, pages 21–35, London, UK. Springer-Verlag.

Garrido, P. (2010a). Comunicação pessoal.

Garrido, P. (2010b). Dynamics of a two class multi-agent model. Working paper. Algoritmi Center. University of Minho.

Garrido, P. (2010c). Modeling and simulation of markets upon a collective intelligence perspective. Working paper. Algoritmi Center. University of Minho.

Garrido, P. (2010d). Programming heterogeneous behavioral agent models with vector and matrix data types support. Working paper. Algoritmi Center. University of Minho.

INRA (2010). Scicoslab.

Luck, M., Griffiths, N., and Al, C. C. (1997). From agent theory to agent construction: A case study. *INTELLIGENT AGENTS III (LNAI VOLUME 1193*, 1193:49—63.

Maes, P. (1995). Artificial life meets entertainment: lifelike autonomous agents. *Commun. ACM*, 38(11):108–114.

Powers, R. and Shoham, Y. (2004). New criteria and a new algorithm for learning in multi-agent systems. In *NIPS*.

Rosenschein, J. S. (1999). Intelligent agent architecture. In Wilson, F. K., editor, *The MIT encyclopedia of cognitive sciences*. MIT Press, Cambridge:MA.

Scilab, C. . R. T. (2010). Scilab.

Steels, L. and Brooks, R. (1995). *The Artificial Life Route To Artificial Intelligence: Building Embodied, Situated Agents*. Psychology Press, illustrated edition edition.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. The MIT Press.

Szuba, T. M. (2001a). *Computational Collective Intelligence*. Wiley-Interscience, 1 edition.

Szuba, T. M. (2001b). A formal definition of the phenomenon of collective intelligence and its IQ measure. *Future Gener. Comput. Syst.*, 17(4):489–500.

Valente, M. (2010). Laboratory for simulation development.

Watkins, C. (1989). *Learning from Delayed Rewards*. PhD thesis, Cambridge University, England.

Apêndice A

Apêndices

A.1 Códigos referentes ao exemplo RL *GridWorld*

A.1.1 Código LSD do exemplo *GridWorld*

Bloco de código A.1: Código LSD do exemplo *GridWorld*

```
#include "fun_head.h"

MODELBEGIN

//Atribuicao do que e a reward
/*
WORLD
[
        -inf    -inf    -inf
        0               0       1000
        -inf    -inf    -inf
        ];
accoes,
1-cima
```

```
2-direita
3-baixo
4-esquerda
*/
EQUATION("state")
v[0]=VL("option",1);
v[1]=VL("state",1);
v[2]= -1000000;//-inf
v[3]=0;
v[4]=1000;

        if(v[1]==1 && v[0]==2)  //estado anterior= 1 e accao =
           direita => estado 2
        {
        v[5]=2;
        WRITE("r",v[2]);
        }

        if(v[1]==1 && v[0]==3)  //estado anterior= 1 e accao =
           baixo => estado 4
        {
        v[5]=4;
        WRITE("r",v[3]);
        }

        if(v[1]==1 && (v[0]==1 || v[0]==4))     //estado anterior
           = 1 e accao = baixo => estado 4
        {
        v[5]=1;
        WRITE("r",v[2]);
        }

        //fim da definicao do estado 1
```

```
if(v[1]==2 && v[0]==2) //estado anterior=2 e accao =
    direita => estado 3
{
v[5]=3;
WRITE("r",v[2]);
}

if(v[1]==2 && v[0]==3) //estado anterior=2 e accao =
    baixo => estado 5
{
v[5]=5;
WRITE("r",v[3]);
}

if(v[1]==2 && v[0]==4)  //estado anterior=2 e accao =
    esquerda => estado 1
{
v[5]=1;
WRITE("r",v[2]);
}

if(v[1]==2 && v[0]==1)  //estado anterior=2 e accao =
    esquerda => estado 2
{
v[5]=2;
WRITE("r",v[2]);
}
//fim da definicao do estado 2


if(v[1]==3 && v[0]==3)  //estado anterior=3 e accao =
    baixo => estado 6
{
v[5]=6;
WRITE("r",v[4]);
```

```
}

if(v[1]==3 && v[0]==4)  //estado anterior=3 e accao =
    esquerda => estado 2
{
v[5]=2;
WRITE("r",v[2]);
}

if(v[1]==3 && (v[0]==1 || v[0]==2))
{
v[5]=3;
WRITE("r",v[2]);
}
//fim da definicao do estado 3

if(v[1]==4 && v[0]==1)  //estado anterior=4 e accao =
    cima => estado 1
{
v[5]=1;
WRITE("r",v[2]);
}

if(v[1]==4 && v[0]==2)  //estado anterior=4 e accao =
    direita => estado 5
{
v[5]=5;
WRITE("r",v[3]);
}

if(v[1]==4 && v[0]==3)  //estado anterior=4 e accao =
    baixo => estado 7
{
v[5]=7;
WRITE("r",v[2]);
```

```
        }

        if(v[1]==4 && v[0]==4)  //estado anterior=4 e accao =
            baixo => estado 7
        {
        v[5]=4;
        WRITE("r",v[3]);
        }
        //fim da definicao do estado 4


        if(v[1]==5 && v[0]==1)
        {
        v[5]=2;
        WRITE("r",v[2]);
        }

        if(v[1]==5 && v[0]==2)
        {
        v[5]=6;
        WRITE("r",v[4]);
        }

        if(v[1]==5 && v[0]==3)
        {
        v[5]=8;
        WRITE("r",v[2]);
        }

        if(v[1]==5 && v[0]==4)
        {
        v[5]=4;
        WRITE("r",v[3]);
        }
//fim da definicao do estado 5
```

```
        if(v[1]==6 && v[0]==1)
        {
        v[5]=3;
        WRITE("r",v[2]);
        }

        if(v[1]==6 && v[0]==3)
        {
        v[5]=9;
        WRITE("r",v[2]);
        }

        if(v[1]==6 && v[0]==4)
        {
        v[5]=5;
        WRITE("r",v[3]);
        }

        if(v[1]==6 && v[0]==2)
        {
        v[5]=6;
        WRITE("r",v[4]);
        }
//fim da definicao de 6

        if(v[1]==7 && v[0]==1)
        {
        v[5]=4;
        WRITE("r",v[3]);
        }

        if(v[1]==7 && v[0]==2)
        {
        v[5]=8;
        WRITE("r",v[2]);
```

```
        }

        if(v[1]==7 && (v[0]==3 || v[0]==4))
        {
        v[5]=7;
        WRITE("r",v[2]);
        }
//fim da definicao do estado 7

        if(v[1]==8 && v[0]==1)
        {
        v[5]=5;
        WRITE("r",v[3]);
        }

        if(v[1]==8 && v[0]==2)
        {
        v[5]=9;
        WRITE("r",v[2]);
        }

        if(v[1]==8 && v[0]==4)
        {
        v[5]=7;
        WRITE("r",v[2]);
        }

        if(v[1]==8 && v[0]==3)
        {
        v[5]=8;
        WRITE("r",v[2]);
        }
        //fim da definicao do estado 8

        if(v[1]==9 && v[0]==1)
```

```
        {
        v[5]=6;
        WRITE("r",v[4]);
        }

        if(v[1]==9 && v[0]==4)
        {
        v[5]=8;
        WRITE("r",v[2]);
        }

        if(v[1]==9 && (v[0]==2 || v[0]==3))
        {
        v[5]=9;
        WRITE("r",v[2]);
        }
//fim da definicao do estado 9

RESULT(v[5]);

EQUATION("option")

v[0]=VL("state",1);     //estado anterior
v[1]=V("state");                                    //estado actual
v[2]=VL("option",1);//opcao anterior

//parametros e variaveis do RL
v[3]=V("r");
v[4]=V("gamma");
v[5]=V("alpha");

cur=SEARCH_CND("IdS",v[1]);//cur<- copia de QsaState
    correspondente ao estado actual
v[6]=VS(cur,"maxQ");//v[6]<- valor do estado actual do parametro
    maxQ
```

```
cur1=SEARCH_CND("IdS",v[0]); //cur1 <- cópia de QsaStates
    correspondente ao estado anterior

cur2=SEARCH_CNDS(cur1,"IdA",v[2]); //cur2 <- cópia de QsaActions
    correspondente a opcao anterior
v[7]=VS(cur2,"Q");//(Q(s(k-1),a(k-1))

v[8]=v[7]+v[5]*(v[3]+v[4]*v[6]-v[7]);

WRITES(cur2,"Q",v[8]); //faz o update de (Q(s(k-1),a(k-1)) com o
    novo valor


v[10]=0; //aux iterador

CYCLES(cur1,cur3,"QsaActions")
{
        v[11]=VS(cur3,"Q");

        if(v[11]>v[10]) //encontrar o valor maximo de Q
                {
                        v[10]=v[11];
                }
}

WRITES(cur1,"maxQ",v[10]);      //escreve o valor maximo de Q no
    parametro maxQ do QsaState anteior


cur4=SEARCH_CNDS(cur1,"Q",v[10]);       //cur4<- copia de
    QsaAction que tem o valor de Q

v[12]=VS(cur4,"IdA"); // guarda os indices das accao(oes) que
    possui(em) o valor maximo
```

```
WRITES(cur1,"IdMaxA",v[12]); //escreve o valor de IdA no
   parametro IdMaxA do QsaState anterior

//algoritmo de decisao

v[13]=UNIFORM(0,1);
v[14]=V("epsilon");//0.1
v[15]=V("IdMaxA");

if(v[13] < v[14])
        {
                cur5=RNDDRAWFAIRS(cur,"QsaActions"); // cur5<-
                   copia de QsaActions que tem o valor de Q
                v[16]=VS(cur5,"IdA"); //guarda em v[16] o IdA da
                   escolhida.
                }
else
        v[16]=v[15];

RESULT(v[16]);

MODELEND

void close_sim(void)
{

}
```

A.1.2 Código ScicosLab do exemplo *GridWorld*

Bloco de código A.2: Código ScicosLab do exemplo *GridWorld*

```
N_MAX_EPISODIOS=1e5;
bad=-1e6;
medium=0;
goal=1000;

nStates=9;
nActions=4;
//parametros do Q learning
gama=0.9;
alpha=0.4;
epsilon=0.5;

last_state=5;
last_action=grand(nActions,'uin',1,nActions);
maxQaction=last_action;

//definicao dos vectores do RL

Reward=[bad,bad,bad,medium,medium,goal,bad,bad,bad];
Q=zeros(nStates,nActions); //definicao da matriz Q
maxQvalue=0;

//Definicao dos States
//E necessario inicializar o State e a Action
State=0;
Action=grand(nActions,'uin',1,nActions);
for i=1:1:N_MAX_EPISODIOS
  //Depois de verificado este switch passara como uma funcao
        if (last_state == 1 & last_action ==2)
          State=2;end
```

```
if (last_state == 1 & last_action ==3)
  State=4;end
if (last_state == 1 & (last_action ==1 |last_action
   ==4))
  State=1;end
//fim da definicao do estado 1
if (last_state == 2 & last_action ==2)
  State=3;end
if (last_state == 2 & last_action ==3)
  State=5;end
if (last_state == 2 & last_action ==4)
  State=1;end
if (last_state == 2 & last_action ==1)
  State=2;end
//fim da definicao do estado 2
if (last_state == 3 & last_action ==3)
  State=6;end
if (last_state == 3 & last_action ==4)
  State=2;end
if (last_state == 3 & (last_action ==1 | last_action==
   2))
  State=3;end
//fim da definicao do estado 3
if (last_state == 4 & last_action ==1)
  State=1;end
if (last_state == 4 & last_action ==2)
  State=5;end
if (last_state == 4 & last_action ==3)
  State=7;end
if (last_state == 4 & last_action ==4)
  State=4;end
//fim da definicao do estado 4
if (last_state == 5 & last_action ==1)
  State=2;end
if (last_state == 5 & last_action ==2)
```

```
  State=6;end
if (last_state == 5 & last_action ==3)
  State=8;end
if (last_state == 5 & last_action ==4)
  State=4;end
//fim da definicao do estado 5
if (last_state == 6 & last_action ==1)
  State=3;end
if (last_state == 6 & last_action ==3)
  State=9;end
if (last_state == 6 & last_action ==4)
  State=5;end
if (last_state == 6 & last_action ==2)
  State=6;end
//fim da definicao do estado 6
if (last_state == 7 & last_action ==1)
  State=4;end
if (last_state == 7 & last_action ==2)
  State=8;end
if (last_state == 7 & (last_action ==3 | last_action
   ==4))
  State=7;end
 //fim da definicao do estado 7
if (last_state == 8 & last_action ==1)
  State=5;end
if (last_state == 8 & last_action ==2)
  State=9;end
if (last_state == 8 & last_action ==4)
  State=7;end
if (last_state == 8 & last_action ==3)
  State=8;end
  //fim da definicao do estado 8
if (last_state == 9 & last_action ==1)
  State=6;end
if (last_state == 9 & last_action ==4)
```

```
            State=8;end
          if (last_state == 9 & (last_action ==2 | last_action
             ==3))
            State=9;end
          //fim da definicao do estado 9
r=Reward(State); //reward associada ao estado actual

//encontrar o maximo valor de Q e respectiva accao maxima
[maxQvalue,maxQaction]=max(Q(State,:));

Qtemp=Q(last_state,last_action);
Q(last_state,last_action)= Qtemp+ alpha*(r + gama*maxQvalue -
   Qtemp);

//algoritmo de decisao
aux=rand(1);

if aux < epsilon
    Action=grand(nActions,'uin',1,nActions);
    else
    Action=maxQaction;
end
//update dos estados seguintes
 last_state=State;
 last_action=Action;
end //fim do for principal
```

A.2 Algoritmo para a implementação de racionalidade

input : **Parâmetros do modelo Económico**
Parâmetros do RL
gamma;
epsilon;
alpha;
nE ; `// número de episódios`
J ; `// número de agentes`

output: Acção escolhida

Declarar variáveis e vectores; `// do modelo e do RL`

for $z \leftarrow 1$ **to** nE **do**
  for $k \leftarrow 2$ **to** $tsim$ **do**
    `// Nível do mercado`
    Cálculo de variáveis agregadas;
    for $j \leftarrow 1$ **to** J **do**
      `// Nível de decisão do agente`
      `// encontrar o estado`
      if $pc(j,k) > pc_{mean}(k)$ **then**
        $state(j,k) = 1$;
        else if $pc(j,k) < pc_{mean}(k)$ **then** $state(j,k) = 3$;
        else $state(j,k) = 2$;
      end
      `// Actualizar o Q anterior`
      $Q_{toupdate} = Qsa(state(j,k-1), option(j,k-1), j)$;
      Encontrar o valor máximo de Q, maxQ, e a respectiva acção que o originou, option$_{maxQ}$
      `// Substituir no algoritmo Q`
      $Qsa(state(j,k-1), option(j,k-1), j) =$
      $Q_{toupdate} + alpha * (pc(j,k) + gamma * maxQ - Q_{toupdate})$;
      `// Decisão entre o agente explorar ou não`
      if $epsilon > Uniform(0,1)$ **then** ; `// explora`
        $option(j,k) = uinrand(1,3)$;
        while $option(j,k) == option_{maxQ}$ **do** a acção for igual à máxima
          $option(j,k) = uinrand(1,3)$
        end
        ;
        `// Atribui-se uma aleatória`
      end
      else $option(j,k) = option_{maxQ}$; ; `// não explora`
      `// Consequência da acção escolhida`
      if $option(j,k)==1$ **then** o agente incrementa a preferência por bens de consumo
        $mfpc = 0.95 * mpfc(j,k-1) + 0.05)$ **else if** $option(j,k)==3$ **then** o agente incrementa a preferência por bens de consumo
          $mfpc = 0.95 * mpfc(j,k-1))$
        end
        else
          $mfpc = mpfc(j,k-1)$
        end
      end
    end
    Executão-se equações de update;
  end
end

Algoritmo 1: algoritmo RL

A.3 Código que permite a importação de dados de um ficheiro de resultados no LSD

Bloco de código A.3: Código ScicosLab do modelo desprovido de racionalidade

```
clear;
clc;
for z=1:1:10
  close(z)
  end
//Sheets = readxls('D:\Disciplinas\Dissertacao\LSD_models\
   m_constant_100_5050\folhas_calculo\ficheiros_res_convertidos\
   Data100v5050.xls');
Sheets = readxls('\media\winD\Disciplinas\Dissertacao\code\
   analise10_10\analise.xls');

typeof(Sheets)
s1=Sheets(4);//O indice 1 corresponde a descricao, os restantes
   para aceder as respectivas folhas
              //Data10_2b
              //Data10_11
              //Data10_22
              //Data10_2bRL
              //Data10_11RL
              //Data10_22RL



typeof(s1);
s1.value; //get the first sheet value field
s1.text; //get the first sheet text field
//typeof(s1(2,:));
editvar s1

[a,b]=size(s1);
data=s1(3:a,:);
```

```
text=s1(1,:);

[c,d]=size(data);

x1=data(:,1);    //upe
x2=data(:,2);    //upc
x3=data(:,3);    //pc_mean
x4=data(:,4);    //pc_cmax
x5=data(:,5);    //pc_cmin
x6=data(:,6);    //pc_cmean
x7=data(:,7);    //pc_ctot
x8=data(:,8);    //pc_emax
x9=data(:,9);    //pc_etot
x10=data(:,10);  //pc_emean
x11=data(:,11);  //pc_emin
x12=data(:,12);  //pe_cmax
x13=data(:,13);  //pe_ctot
x14=data(:,14);  //pe_cmean
x15=data(:,15);  //pe_cmin
x16=data(:,16);  //pe_emax
x17=data(:,17);  //pe_etot
x18=data(:,18);  //pe_emean
x19=data(:,19);  //pe_emin
x20=data(:,20);  //m_cmax
x21=data(:,21);  //m_ctot
x22=data(:,22);  //m_cmean
x23=data(:,23);  //m_cmin
x24=data(:,24);  //m_emax
x25=data(:,25);  //m_etot
x26=data(:,26);  //m_emean
x27=data(:,27);  //m_emin
x28=data(:,28);  //pc_tot
x29=data(:,29);  //pe_tot
x30=data(:,30);  //pe_mean
x31=data(:,31);  //m_tot
```

```
//acrescentar aqui as series dos parametros e variaveis
   especificos do RL

k=1:1:c;

//Actuais combinacoes de graficos
//grafico do dinheiro
scf(1);
w=get("current_axes");//get the handle of the newly created axes
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x20,'-rs');
plot(k,x22,'-g+');
plot(k,x23,'-k>');
plot(k,x24,'-bx');
plot(k,x26,'-mo');
plot(k,x27,'-y<');
xtitle("GrÃ¡fico do Dinheiro");
xlabel("k");
legend(['m_cmax';'m_cmean';'m_cmin';'m_emax';'m_emean';'m_emin'])
   ;
//
//
////
////grafico das pc de c e de e
scf(2);
w=get("current_axes");//get the handle of the newly created axes
```

```
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x4,'-bs');
plot(k,x6,'-g+');
plot(k,x11,'-r<');
plot(k,x16,'-mo');
plot(k,x18,'-c+');
plot(k,x19,'-b^');
plot(k,x3,'-k>');
xtitle("DistribuiÃ■Ã■o dos Produtos de consumo")
xlabel("k");
legend(['pc_cmax';'pc_cmean';'pc_emin';'pe_emax';'pe_emean';'
   pe_emin';'pc_mean']);

//
////grafico das pc de e de e
scf(3);
w=get("current_axes");//get the handle of the newly created axes
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x12,'-bs');
```

```
plot(k,x14,'-g+');
plot(k,x19,'-r<');
plot(k,x16,'-co');
plot(k,x18,'-m^');
plot(k,x19,'-bv');
plot(k,x30,'-k>');
xtitle("DistribuiÃ■Ã■o dos Produtos de equipamento")
legend(['pe_cmax';'pe_cmean';'pe_emin';'pe_emax';'pe_emean';'
   pe_emin';'pe_mean']);
xlabel("k");


//////grafico dos precos
scf(4);
w=get("current_axes");//get the handle of the newly created axes
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x2,'-rs');
plot(k,x1,'-k>');
xtitle("PreÃ■os")
legend(['upc';'upe']);
xlabel("k");

////grafico do dinheiro na posse dos agentes
scf(5);
w=get("current_axes");//get the handle of the newly created axes
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
```

```
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x21,'-rs');
plot(k,x25,'-k>');
plot(k,x31,'-bo');
xtitle("Dinheiro na posse dos agentes produtores de equipamento e
    produtores de bens consutmo");
xlabel("k");
legend(['m_ctot';'m_etot';'m_tot']);

//

scf(6);
w=get("current_axes");//get the handle of the newly created axes
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x21,'-rs');
plot(k,x25,'-k>');
xlabel("k");
xtitle("Produtos de equipamento na posse de produtores de consumo
   ");
legend(['m_ctot';'m_etot']);
//
```

*Código que permite a importação de dados de um ficheiro de resultados no LSD*115

```
////grafico PCs de e tots
scf(7);
w=get("current_axes");//get the handle of the newly created axes
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x7,'-rs');
plot(k,x9,'-k>');
plot(k,x28,'-bo');
xlabel("k");
xtitle("DistribuiÃ■Ã■o dos Produtos de Consumo");
legend(['pc_ctot';'pc_etot';'pc_tot']);
//
scf(8);
w=get("current_axes");//get the handle of the newly created axes
w.axes_visible="on"; // makes the axes visible
w.font_size=3; //set the tics label font size
//a.x_location="top"; //set the x axis position
//a.data_bounds=[-100,-2,-1;100,2,1]; //set the boundary values
   for the x, y and z coordinates.
w.sub_tics=[5,0];
w.labels_font_color=1;
w.grid=[1,1];
//a.box="off";
plot(k,x13,'-rs');
plot(k,x17,'-k>');
plot(k,x29,'-bo')
xlabel("k");
xtitle("Distribuição dos Produtos de Consumo");
```

```
legend(['pe_ctot';'pe_etot';'pe_tot']);
```

A.4 Código do modelo desprovido de racionalidade

Bloco de código A.4: Código ScicosLab do modelo desprovido de racionalidade

```
clear
clc
tic();
//Simulation parameters
tsim=800;
time=0:1:tsim;
itsim=tsim+1;
//Product parameters
c=1;
e=2;
lambda_c=0;
lambda_e=0.95;
gamma_e=1-lambda_e;

//Market parameters
J_c=5;
J_e=5;
J=J_c+J_e;

//Computing indexes
jc1=1;
jcl=J_c;

je1=jcl+1;
jel=jcl+J_e;

//Market parameters for initializing agents
minpc=1;
S=2;
pc_init=S*minpc;
```

```
pe_init=1;
tot_pec=J_c*pe_init;
tot_pee=J_e*pe_init;
tot_pe=tot_pec+tot_pee;

Kc_min=S*J*minpc/tot_pec;
Kc_mean=4;
Kc_std=0;
mprintf('\nMean value of Kc set to %f times minimum!\n',Kc_mean/
    Kc_min);

Ke_min=gamma_e*tot_pe/tot_pee;
Ke_mean=0.1;
Ke_std=0;
mprintf('\nMean value of Ke set to %f times minimum!\n',Ke_mean/
    Ke_min);

mfpc_mean=0.5;
mfpc_std=0;

mfpe_mean=1-mfpc_mean;
mfpe_std=0;

m_init_plus=1000;
m_init_minus=1000;

//Agents producing consuming parameters and variables and
    initialization

a_type(1:J_c)=c;
a_type(J_c+1:J)=e;

K(1:J_c)=grand(J_c,'nor',Kc_mean,Kc_std);
K(J_c+1:J)=grand(J_e,'nor',Ke_mean,Ke_std);
```

```
sprodc=zeros(J_c,1);
sprode=zeros(J_e,1);
sprod=cat(1,sprodc,sprode);

pe=pe_init*ones(J,1);

pc=pc_init*ones(J,1);

m=grand(J,1,'unf',m_init_minus,m_init_plus);
mfpc=grand(1,'nor',mfpc_mean,mfpe_std);
mfpe=1-mfpc;
imppc=m*mfpc;
imppe=m*mfpe;

//Market variables initialization
tot_sprodc=zeros(1,itsim);
tot_sprodc(1:itsim)=%nan;

tot_sprode=zeros(1,itsim);
tot_sprode(1:itsim)=%nan;

tot_imppe_e=zeros(1,itsim);
tot_imppe_e(1:itsim)=%nan;

tot_imppc=zeros(1,itsim);
tot_imppc(1:itsim)=%nan;

tot_imppe=zeros(1,itsim);
tot_imppe(1:itsim)=%nan;

fator_investimento_e=zeros(1,itsim);
fator_investimento_e(1:itsim)=%nan;

upc=zeros(1,itsim);
```

```
upc(1,1:itsim)=%nan;

upe=zeros(1,itsim);
upe(1,1:itsim)=%nan;

tot_pc=zeros(1,itsim);
tot_pc(1,1:itsim)=%nan;
tot_pc(1)=sum(pc(:,1));

pc_mean=zeros(1,itsim);
pc_mean(1,1:itsim)=%nan;
pc_mean(1)=tot_pc(1)/J;


//Simulation
for k=2:1:tsim+1

  //Agent level
  sprod(:,k)=K.*pe(:,k-1);

  //Market level
  tot_sprodc(k)=sum(sprod(1:J_c,k));
  tot_sprode(k)=sum(sprod(J_c+1:J,k));

  tot_imppe_e(k)=sum(imppe(je1:jel,k-1));
  tot_imppe(k)=sum(imppe(:,k-1));
  tot_imppc(k)=sum(imppc(:,k-1));
  fator_investimento_e(k)=tot_imppe_e(k)/tot_imppe(k);

  upc(k)=tot_imppc(k)./tot_sprodc(k);
  upe(k)=tot_imppe(k)./tot_sprode(k);

  //Agent level
  pc(:,k)=lambda_c*pc(:,k-1)+imppc(:,k-1)/upc(k);
  pe(:,k)=lambda_e*pe(:,k-1)+imppe(:,k-1)/upe(k);
```

```
  for j=1:1:J
    if a_type(j)==c
    m(j,k)=m(j,k-1)+sprod(j,k)*upc(k)-imppc(j,k-1)-imppe(j,k-1);
    else m(j,k)=m(j,k-1)+sprod(j,k)*upe(k)-imppc(j,k-1)-imppe(j,k
        -1);
    end
  end

  imppc(:,k)=m(:,k)*mfpc;
  imppe(:,k)=m(:,k)*mfpe;

  //Market level
  tot_pc(k)=sum(pc(:,k));
  pc_mean(k)=tot_pc(k)/J;
end
exec_time=toc();
mprintf('\nSimulation run in %f seconds!\n',exec_time)
```

A.5 Código do modelo com RL

Bloco de código A.5: Código ScicosLab do modelo com RL

```
clc
clear
rand('seed',1);
//grand set seed
grand('setsd',2);
tic();
//Simulation parameters
tsim=800;
time=0:1:tsim;
itsim=tsim+1;

//Product parameters
c=1;
e=2;
lambda_c=0;
lambda_e=0.95;
gamma_e=1-lambda_e;

//Market parameters
J_c=5;
J_e=5;

J=J_c+J_e;

//Computing indexes
jc1=1;
jcl=J_c;

je1=jcl+1;
jel=jcl+J_e;

//Market parameters for initializing agents
```

```
minpc=1;
S=2;
pc_init=S*minpc;

pe_init=1;
tot_pec=J_c*pe_init;
tot_pee=J_e*pe_init;
tot_pe=tot_pec+tot_pee;

Kc_min=S*J*minpc/tot_pec;
Kc_mean=4.4;
Kc_std=0;
mprintf('\nMean value of Kc set to %f times minimum!\n',Kc_mean/
   Kc_min);

Ke_min=gamma_e*tot_pe/tot_pee;
Ke_mean=0.11;
Ke_std=0;
mprintf('\nMean value of Ke set to %f times minimum!\n',Ke_mean/
   Ke_min);

mfpc_mean_i=0.5;

mfpc_std_i=0;

mfpe_mean_i=1-mfpc_mean_i;

mfpe_std_i=0;

m_init_plus=1000;
m_init_minus=1000;

//Agents producing consuming parameters and variables and
   initialization
```

```
K(jc1:jcl)=grand(J_c,'nor',Kc_mean,Kc_std);
K(je1:jel)=grand(J_e,'nor',Ke_mean,Ke_std);

//RL basic initialization
nS=3;
nO=3;

Qsa=rand(nS,nO,J);

eps=0.5;
alpha=0.3;
gamma_R=0.1;


sprod(1:J,1:itsim)=%nan;

pe=zeros(J,itsim);
pe(:,1)=pe_init;

pc=zeros(J,itsim);
pc(:,1)=pc_init;

m=zeros(J,itsim);
m(:,1)=grand(J,1,'unf',m_init_minus,m_init_plus);

mfpc=zeros(J,itsim);
mfpc(:,1)=grand(J,1,'nor',mfpc_mean_i,mfpc_std_i);

mfpe=zeros(J,itsim);
mfpe(:,1)=grand(J,1,'nor',mfpe_mean_i,mfpe_std_i);

imppc=zeros(J,itsim);
imppc(:,1)=m(:,1).*mfpc(:,1);

imppe=zeros(J,itsim);
```

```
imppe(:,1)=m(:,1).*mfpe(:,1);

//Agents decision parameters and variables (to include with
   changes in initialiation above)

state=zeros(J,itsim);
state(:,1)=2;

option=zeros(J,itsim);
option(:,1)=2;

mfpc_xg_a=0.9;
mfpc_xg_b=1-mfpc_xg_a;

//Market variables initialization

tot_sprodc=zeros(1,itsim);
tot_sprodc(1:itsim)=%nan;

tot_sprode=zeros(1,itsim);
tot_sprode(1:itsim)=%nan;

tot_imppe_e=zeros(1,itsim);
tot_imppe_e(1:itsim)=%nan;

tot_imppc=zeros(1,itsim);
tot_imppc(1:itsim)=%nan;

tot_imppe=zeros(1,itsim);
tot_imppe(1:itsim)=%nan;

fator_investimento_e=zeros(1,itsim);
fator_investimento_e(1:itsim)=%nan;

upc=zeros(1,itsim);
```

```
upc(1,1:itsim)=%nan;

upe=zeros(1,itsim);
upe(1,1:itsim)=%nan;

tot_pc=zeros(1,itsim);
tot_pc(1,1:itsim)=%nan;
tot_pc(1)=sum(pc(:,1));

pc_mean=zeros(1,itsim);
pc_mean(1,1:itsim)=%nan;
pc_mean(1)=tot_pc(1)/J;

//Simulation
for k=2:1:itsim

  //Agent level
  sprod(:,k)=K.*pe(:,k-1);

  //Market level
  tot_sprodc(k)=sum(sprod(jc1:jcl,k));
  tot_sprode(k)=sum(sprod(je1:jel,k));

  tot_imppe_e(k)=sum(imppe(je1:jel,k-1));
  tot_imppe(k)=sum(imppe(:,k-1));
  tot_imppc(k)=sum(imppc(:,k-1));
  fator_investimento_e(k)=tot_imppe_e(k)/tot_imppe(k);

  upc(k)=tot_imppc(k)./tot_sprodc(k);
  upe(k)=tot_imppe(k)./tot_sprode(k);

  //Agent level
  pc(:,k)=lambda_c*pc(:,k-1)+imppc(:,k-1)/upc(k);
  pe(:,k)=lambda_e*pe(:,k-1)+imppe(:,k-1)/upe(k);
```

```
m(jc1:jcl,k)=m(jc1:jcl,k-1)+sprod(jc1:jcl,k)*upc(k)-imppc(jc1:
    jcl,k-1)-imppe(jc1:jcl,k-1);
m(je1:jel,k)=m(J_c+1:J,k-1)+sprod(je1:jel,k)*upe(k)-imppc(je1:
    jel,k-1)-imppe(je1:jel,k-1);

//Market level
tot_pc(k)=sum(pc(:,k));
pc_mean(k)=tot_pc(k)/J;

for j=1:1:J
  if pc(j,k)>pc_mean(k) then
    state(j,k)=1;
  elseif pc(j,k)<pc_mean(k) then
    state(j,k)=3;
  elseif abs(pc(j,k)-pc_mean(k))<= 0.05*pc_mean(k) then
    state(j,k)=2;
  end

  Q_to_update=Qsa(state(j,k-1),option(j,k-1),j);
  [maxQ,option_maxQ]=max(Qsa(state(j,k),:,j));
  Qsa(state(j,k-1),option(j,k-1),j)=Q_to_update+alpha*(m(j,k)+
      gamma_R*maxQ-Q_to_update);

    if grand(1,'def')<eps then //explore
        option(j,k)=grand(1,'uin',1,n0);

          while option(j,k)==option_maxQ
          option(j,k)=grand(1,'uin',1,n0);
          end
    else //exploit
    option(j,k)=option_maxQ;
    end

  if option(j,k)==1 then
  mfpc(j,k)=0.95*mfpc(j,k-1)+0.05; //incrementa
```

```
    elseif option(j,k)==3 then
    mfpc(j,k)=0.95*mfpc(j,k-1);      //decrementa
    else mfpc(j,k)=mfpc(j,k-1);
    end

  end

  mfpe(:,k)=1-mfpc(:,k);
  imppc(:,k)=m(:,k).*mfpc(:,k);
  imppe(:,k)=m(:,k).*mfpe(:,k);

end
exec_time=toc();

mprintf('\nSimulation run in %f seconds!\n',exec_time)
```

A.6 Código que permite o desenho dos gráficos para os modelos codificados em ScicosLab

Bloco de código A.6: Código que permite o desenho dos gráficos para os modelos codificados em ScicosLab

```
r_pc=sum(pc,'c');
pontofixo=gamma_e/Ke_mean;
minpc_array=minpc*ones(time);

mprintf('\nRPC is %f\n',r_pc)
//Results analysis
mfpe_mean=mean(mfpe,'r');
mfpe_cmean=mean(mfpe(jc1:jcl,:),'r');
mfpe_emean=mean(mfpe(je1:jel,:),'r');

[pc_max,a_pc_max]=max(pc,'r');
[pc_min,a_pc_min]=min(pc,'r');
pc_cmean=mean(pc(jc1:jcl,:),'r');
pc_emean=mean(pc(je1:jel,:),'r');

[pe_max,a_pe_max]=max(pe,'r');
[pe_min,a_pe_min]=min(pe,'r');
pe_mean=mean(pe,'r');

pe_cmean=mean(pe(jc1:jcl,:),'r');
pe_emean=mean(pe(je1:jel,:),'r');

mprintf('\nthe miniminum value of pc is: %f', min(pc));
mprintf('\nthe maximum value of pc is: %f ',max(pc));

//Visualization

drawlater();
```

```
scf(0);clf
//xtitle('Parameters')

subplot(2,3,1)
plot(time,pc_mean,'-y',time,minpc_array,'-r',time,pc(1,:),'-b',
   time,pc(6,:),'-g',time,pc(2:5,:),'-b',time,pc(7:10,:),'-g');
a=gca();
  title('I','fontsize',14);
  a.grid=[0,0];
  a.log_flags="nln";
  a.data_bounds=[0,0.1;tsim,1000000];
  a.children.children.mark_mode='on';
  a.children.children.mark_style=11;//
  a.children.children.mark_size=2;
  a.children.children.mark_background=0;
  p=get("hdl");// //get handle on current entity (here the
     polyline entity)

  p.children(12).thickness=3;

  p.children(11).thickness=3;

  a.y_label.text='produtos de consumo';
  a.x_label.text='tempo';
  h1=legend('pc_mean','minpc','pc_jc','pc_je',2);

 subplot(2,3,2)
plot(time,m(1,:),'-b',time,m(6,:),'-g',time,m(2:5,:),'-b',time,m
   (7:10,:),'-g')
  a=gca();
  title('II','fontsize',14);
  a.grid=[0,0];
  a.log_flags="nln";
  a.children.children.mark_mode='on';
  a.children.children.mark_style=11;
```

```
  a.children.children.mark_size=2;
  a.children.children.mark_background=0;
  p=get("hdl"); //get handle on current entity (here the polyline
      entity)

  a.y_label.text='dinheiro';
  a.x_label.text='tempo';
  h1=legend('m_jc','m_je',3);

subplot(2,3,3)
plot(time,fator_investimento_e,time,pontofixo,time,mfpe_mean);
  a=gca();
  title('III','fontsize',14);
  a.grid=[-1,0];
  a.data_bounds=[0,0;tsim,1];
  a.children.children.mark_mode='on';
  a.children.children.mark_style=11;
  a.children.children.mark_size=3;
  a.children.children.mark_background=0;
  a.x_label.text='tempo';
  p=get("hdl"); //get handle on current entity (here the polyline
      entity)
  p.children(1).foreground=15;
  p.children(1).thickness=1;
  p.children(2).foreground=21;
  p.children(2).line_style=1;
  p.children(2).thickness=3;
  p.children(3).foreground=17;
  p.children(3).line_style=1;
  p.children(3).thickness=1;
  h1=legend('fator invest','ponto fixo','mfpe_mean',3);

subplot(2,3,4)
//plot(time,pc_mean,'-y',time,pc(1,:),'-b',time,pc(6,:),'-g',time
   ,pc(2:5,:),'-b',time,pc(7:10,:),'-g');
```

```
plot(time,pe_mean,'-m',time,pe(1,:),'-b',time,pe(6,:),'-g',time,
    pe(2:5,:),'-b',time,pe(7:10,:),'-g');
  a=gca();
  title('IV','fontsize',14);
  a.grid=[0,0];
  a.data_bounds=[0,0.1;tsim,1000000];
  a.log_flags="nln";
  a.children.children.mark_mode='on';
  a.children.children.mark_style=11;
  a.children.children.mark_size=2;
  a.children.children.mark_background=0;
  p=get("hdl"); //get handle on current entity (here the polyline
      entity)

  //limite de subsistencia
 // p.children(1).foreground=6;
  p.children(11).thickness=3;
 // p.children(1).line_style=1;
  a.y_label.text='produtos de equipamento';
  a.x_label.text='tempo';
  h1=legend('pe_mean','pe_jc','pe_je',2);

subplot(2,3,5)
plot(time,tot_imppc,time,tot_imppe);
  a=gca();
  title('V','fontsize',14);
  a.data_bounds=[0,1000;tsim,10000];
  a.grid=[0,0];
  a.log_flags="nln";
  a.children.children.mark_mode='on';
  a.children.children.mark_style=11;
  a.children.children.mark_size=2;
  a.children.children.mark_background=0;
  a.y_label.text='consumo vs investimento';
```

```
  a.x_label.text='tempo';
  h1=legend('tot_imppc','tot_imppe',3);

subplot(2,3,6)
plot(time,upc,time,upe);
  a=gca();
   title('VI','fontsize',14);
  a.grid=[0,0];
  a.log_flags="nln";
  a.children.children.mark_mode='on';
  a.children.children.mark_style=11;
  a.children.children.mark_size=2;
  a.children.children.mark_background=0;
  a.y_label.text='preços unitários';
  a.x_label.text='tempo';
  h1=legend('upc','upe',3);
drawnow();
```