Simulações numéricas em Matlab para modelos de equilíbrio dinâmico e estocástico: teste de eficiência com diferentes funções de utilidade.

(Dr. Ricardo Luis Chaves Feijó)

Resumo

O ensaio fornece um conjunto de técnicas computacionais de simulação numérica que são empregados no teste de convergência para o equilíbrio geral, no contexto dinâmico e estocástico em modelos que não permitem facilmente obter-se soluções algébricas explícitas. A técnica de simulação é empregada em diferentes hipóteses de função de utilidade. Em cada qual, é calculado o tempo de processamento até que a condição de equilíbrio seja alcançada dentro de uma margem de flutuação máxima admissível e, em seguida, a possibilidade de convergência do modelo para o equilíbrio geral dinâmico e estocástico é avaliada em termos de um cálculo de probabilidade que considera o tempo de processamento. O modelo de simulação inclui a maximização da utilidade do agente, observância das restrições orçamentárias e das condições de equilíbrio geral. Emprega-se a hipótese de carteira de ativos eficientes e a de um processo GARCH na técnica de determinação dos retornos e dos preços dos títulos.

Abstract

This paper provides a set of computational techniques of numerical simulation that are employed in the convergence test towards the general equilibrium, on the dynamic and stochastic context in models that do not allow easily to obtain explicit algebraic solutions. The simulation technique is employed in different models of utility function. It is computed the processing times until the equilibrium condition is reached within a permissible maximum fluctuation margin and then the possibility of convergence of the model into the dynamic and stochastic general equilibrium is evaluated in terms of a probability calculation that considers the processing time. The simulation model includes maximizing agent's utility, meeting of the budget constraints and general equilibrium conditions. It employs the assumption of efficient portfolio and a GARCH process on determining the returns and the prices of securities.

Palavras-chave (Keywords)

Consumo agregado, simulação numérica, equilíbrio geral, processo estocástico (*Dynamic general equilibrium, stochastic process, convergence test, numerical simulation*)

Código JEL G1 G14/G17

Introdução

Nesta pesquisa, oferecem-se métodos de estatística computacional visando soluções numéricas ao problema de alcançar-se, por simulação, números atribuído aleatoriamente às variáveis de um modelo que sejam compatíveis com as condições estipuladas de equilíbrio geral dinâmico e estocástico num contexto com agentes maximizadores de utilidade, e com retornos de ativos financeiros que evoluem por estágios perfazendo um caminho estocástico. Nesse sentido, esforça-se em desenvolver modelos de simulação

compatíveis com uma técnica robusta que ofereça boa fundamentação microeconômica tanto no problema de maximização de utilidade quanto no atendimento das condições de equilíbrio geral, bem como na escolha de carteiras de ativos eficientes e na descrição de retorno e de preços dos ativos por um processo estocástico realista e apropriado. O eixo principal da pesquisa preocupa-se em desenvolver rotinas de programação em Matlab correspondentes ao modelo proposto pela investigação teórico-abstrata. Destarte, estima-se o comportamento do modelo em ambiente de simulação numérica no qual se dá a determinação computacional aleatória dos valores das variáveis estocásticas do modelo em tela e o teste de atendimento das condições almejadas, qual sejam, a maximização da utilidade de todos os agentes (ou quase todos) e o atendimento, por parte deles, de um conjunto de equações que traduzam relações lógicas ou restrições orçamentárias. Para tanto, emprega-se a ideia de maximização de utilidade por meio da seleção de pontos candidatos em um processo de simulação que evolui a cada dois períodos adjacentes, e na incorporação explícita de carteiras de ativos eficientes de Markowitz, e ainda no uso de funções de expectativas de retorno, na modelagem do retorno e dos preços dos ativos, usando-se, para tanto, a hipótese de um processo com memória descrito pela técnica GARCH.

Ao contrário do tratamento algébrico convencional, não se preocupa aqui em especificar uma função orientadora do equilíbrio dinâmico. O que se oferece aqui é o desenvolvimento de uma ferramenta para estimativa numérica do equilíbrio dinâmico em simulações com dados imaginários. Para tanto, desenvolvem-se técnicas para simulações numéricas de mercados financeiros que não pagam dividendos ao longo do tempo, cujos retornos são apenas ganhos de capital, nos quais as dotações dos agentes seguem uma trajetória comandada por um processo recursivo discreto. Trabalha-se em cenários com agentes heterogêneos, com diferentes taxas de preferência intertemporal e diferentes expectativas quanto ao retorno almejado. Testamos o modelo com distintas funções de utilidade e avaliamos cuidadosamente o tempo de processamento e a relação disso com a convergência do processo estocástico a um equilíbrio estacionário. No exercício de simulação, foca-se essencialmente no problema da escolha entre consumir e comprar uma carteira de ativos num contexto de programação dinâmica e com variáveis estocásticas. Para tanto, oferecem-se rotinas de programas rodados no *software* Matlab.[1] Os programas são testados com o uso de um computador do tipo *laptop* com 8 GB de memória RAM e 3,4 GHz com CPU Intel i-7.

O modelo de simulação numérica e computacional proposto tem relação com a literatura que discute a existência ou não de equilíbrio geral dinâmico e estocástico em uma classe de modelos. De fato, no problema com renda estocástica gerada por um componente autônomo, bem como gerada pela venda líquida de ativos financeiros ao longo de mudanças temporais nas carteiras de investimentos individuais e pelo retorno dessas carteiras, a prova de existência de um equilíbrio dinâmico nos mercados financeiros em

[1] Os extensos programas em Matlab estão disponíveis a quem se interessar pelo e-mail riccfeij@usp.br.

questão nem sempre pode ser explicitamente demonstrada. Sendo assim, opta-se pela busca de solução por simulação numérica do problema da escolha intertemporal ótima entre consumir e investir na presença de dotações que evoluem por um processo estocástico e considerando-se ainda as restrições no problema da escolha de uma carteira ótima. Cada indivíduo, no modelo, busca resolver o problema da otimização num contesto dinâmico. As dotações caminham por um processo estocástico seguindo um movimento browniano discreto. As restrições orçamentárias para a escolha de carteira levam em conta tal processo estocástico nas dotações. Os modelos teóricos procuram identificar as trajetórias ótimas de consumo e investimento nesse contexto dinâmico e estocástico. O foco consiste em determinar o tempo de processamento em que se obtém tais trajetórias, que além de ótimas são compatíveis com o equilíbrio geral dinâmico. O modelo de simulação numérica se aplica ao teste de convergência de uma gama de modelos do tipo considerado. Acreditamos que tal teste possa sinalizar aos economistas matemáticos puros em quais modelos valeria a pena enveredar esforços no tratamento teórico a fim de se demonstrar a existência de um equilíbrio dinâmico. Nessa ótica, apenas nos modelos que apresentarem boa convergência com base nas simulações numéricas valeriam a pena esforços no sentido da obtenção de uma solução algébrica.

A fim de investigar a temática, o presente ensaio divide-se em seis seções. A primeira examina o problema teórico e oferece a proposta de simulação; a segunda seção explica o caso em que os retornos dos ativos seguem um processo do tipo GARCH. A seção seguinte mostra as hipóteses e os procedimentos básicos adotados no programa. A seção 4 discute como são feitas as estimativas por simulação. O teste de convergência ao equilíbrio geral dinâmico é feito na seção seguinte. Finalmente, a seção 6 conclui o ensaio.

1. O problema teórico e a proposta de simulação numérica.

Sabe-se que, em economias dinâmicas, a trajetória de equilíbrio pode ser teoricamente identificada com base em uma sequência de funções que geram o histórico de realizações em um processo estocástico, tendo-se em conta preços e alocações estimados, de tal modo que as condições de equilíbrio sejam atendidas, ou seja, que todos os agentes na economia maximizem utilidade e que todos os mercados se equilibrem. Quando a trajetória de equilíbrio é estimada utilizando-se, a cada período, informações importadas do período anterior (ou da sequência de períodos passados) fala-se em equilíbrios *recursivos*. Em geral, na literatura teórica, tais equilíbrios são matematicamente caracterizados no formalismo de espaço de estados, *policy function* e funções de transição. Outra estratégia, que não substitui mas complementa o tratamento puramente algébrico e topológico, consiste em desenvolver algoritmos para uso computacional. Nesse âmbito, as simulações consideram ativos financeiros comercializados no início de cada período ao longo de uma trajetória estocástica. Há apenas um único bem de consumo. Em tais condições busca-se encontrar um equilíbrio dinâmico estacionário caracterizado de maneira peculiar. Como em Burnside (1998), Collard e Juillard (2001) e Schmitt-Ghohé e Uribe (2004), supõem-se agentes

representativos que maximizam a função utilidade ao longo dos períodos e para os quais os retornos dos ativos adquiridos seguem um processo estocástico exógeno. O problema de maximização de utilidade pelos agentes ocorre a cada período e pressupõe um tipo específico de função de utilidade, de livre escolha para a modelagem. Para o agente, o ganho envolvido é apenas o ganho de capital associado a variações nos preços dos ativos. Esse tipo de modelo só funciona (isto é, o equilíbrio geral estocástico pode ser obtido) quando se impõe artificialmente que os agentes não entrarão num esquema de jogo Ponzi. Isso pode ser evitado impondo-se de antemão um limite exógeno ao endividamento em operações de *short sale*.

Tem-se um modelo dinâmico recursivo em dois tipos de variáveis exógenas: as dotações dos agentes e o retorno dos ativos financeiros. Oferece-se um tratamento dinâmico discreto no qual as decisões econômicas de otimização são realizadas a cada par de períodos, considerando-se apenas os sorteios vigentes nos períodos em questão e a estrutura do modelo. Os resultados de equilíbrio das variáveis gerados em um par de períodos são utilizados como valores de referência para o processo de otimização no próximo par. Para tanto, supõe-se a existência de um conjunto finito de ativos. Os retornos por unidade de ativo são idênticos entre os agentes. A diferença entre os agentes incide apenas em suas preferências intertemporais, que afetam as escolhas entre consumir hoje ou no período seguinte, nas taxas de retornos almejados dos ativos e em certos parâmetros aleatórios. Não se consideram explicitamente a firma e a maximização do lucro. O enfoque recai apenas na maximização de utilidade dos consumidores num ambiente não determinista. O equilíbrio para essa economia constitui um processo de construção de carteiras e decisões de consumo, além da formação de preços de ativos financeiros nos quais, para todos eles, os mercados se equilibram e cada agente maximiza sua função utilidade num conjunto orçamentário que também leva em conta o fluxo de retornos dos ativos e da renda líquida com a compra deles.

A técnica de simulação numérica auxilia em contornar tais dificuldades, pois ela permite uma avaliação do comportamento do sistema dinâmico em questão sem que se conheça de antemão a existência do equilíbrio dinâmico, e muito menos qual as funções do caso. O método de simulação aqui proposto funciona então como um estágio preparatório antes que se dirijam esforços a fim de se obter funções explícitas. De fato, nas simulações pode-se avaliar qualitativamente o comportamento do sistema dinâmico antes mesmo da determinação delas. Assim, o grau de convergência do sistema dinâmico a um equilíbrio dinâmico é avaliado qualitativamente pelas margens de flutuação alcançadas, pelos níveis de consumo, pelos tempos de processamento etc. A cada período, a simulação gera um vetor das variáveis endógenas relevantes, como consumo, carteira eficiente, retornos dos ativos e seus preços. Tal modelo segue a exposição teórica de Zame (1990) e Dubey, Geanakoplos e Shubik (2005). Considera-se um conjunto contável e finito Ω dos estados da natureza ($\Omega = \{1, 2, \dots\}$). O plano de consumo x especifica o consumo na data 0 e na data 1, $x = (x(0), x(\omega))$. O consumo na data 0 é representado por $x(0) \in \mathbb{R}$ e o plano de consumo na data 1 por $x(\omega): \Omega \to \mathbb{R}$; $x(\omega)$ é o consumo quando ocorre o estado ω. $x(\omega)$ é limitado. $A(\omega)$ é o retorno do ativo A no estado

ω. Para N ativos, $A_1, \dots, A_N$, tem-se uma carteira representada por $\theta = (\theta_1, \dots, \theta_N) \in \mathbb{R}^N$. O retorno da carteira é $\sum_{i=1}^{N} \theta_n A_n(\omega)$. O vetor dos preços dos ativos é representado por $q \in \mathbb{R}^N$. Portanto, o valor da carteira θ aos preços q é então $q.\theta = \sum_{i=1}^{N} q_n \theta_n$. Cada consumidor $h \in \{1, \dots, H\}$ é definido pela dotação d^h e pela função utilidade U^h. O consumidor maximiza a utilidade gerada pelo consumo na data 0 e na data 1, de acordo com alguma distribuição de probabilidade μ^h em Ω. Com um número de estados finito, a utilidade do indivíduo é dada pela função:

(1) $U^h(x) = v_h\big(x(0),\ x(\omega)\big)$

A função v é contínua e pode assumir diferentes formas (oito delas neste ensaio). As derivadas parciais são finitas. Para cada h, as dotações são positivas e $d^h \gg 0$. Sobre o mercado em questão, Zame (1990) define-o como sendo o conjunto $\varepsilon = (\{(d^h, U^h)\}, \{A_n\})$, onde $\{(d^h, U^h): 1 \leq h \leq \mathrm{H}\}$ é um conjunto finito de consumidores e $\{A_n: 1 \leq n \leq N\}$ é um conjunto finito de ativos. O conjunto orçamentário do consumidor, com função de utilidade U^h e dotação d^h é $B^h(q, d^h)$. O equilíbrio maximizador leva à escolha de um conjunto de triplas (x^h, φ^h, ψ^h), onde x^h é um plano de consumo viável; φ^he ψ^h são carteiras de compras e vendas de ativos financeiros. O equilíbrio geral para o mercado com um único bem e um número finito de ativos é a quadra $(q, x^h, \varphi^h, \psi^h)$, onde $q \in \mathbb{R}^N$ é o vetor dos preços dos ativos, e cada x^h trata-se de um plano de consumo viável (plano de consumo de equilíbrio de h) tais que o mercado de bem de consumo está em equilíbrio e o excesso de oferta dos ativos se anula dentro de uma margem de flutuação admitida. Ou então que a soma dos dois tipos de excesso se aproxime de zero. Além disso, os planos são financeiramente viáveis: para cada h, $(x^h, \varphi^h, \psi^h) \in B^h(q, d^h)$ e os consumidores otimizam em seus conjuntos orçamentários. Ou seja, se $(y^h, \varphi^h, \psi^h) \in B^h(q, d^h,)$ então $U^h(x^h) \geq U^h(y^h)$.

Para cada estado da natureza ω, pode-se associar uma correspondente variável endógena $x(\omega)$. A cada dois períodos, com estado futuro já revelado, o agente depara-se com um problema de escolha entre consumir hoje ou no período seguinte análogo ao problema da escolha entre dois bens no mesmo período. O consumo no amanhã poderia ser descontado por um fator β que traduz preferência temporal. Diversos tipos de função de utilidade poderiam ser empregados. Por exemplo, uma Cobb-Douglas:

(2) $U^h(x) = x(0)^{\alpha}\big(\beta x(\omega)\big)^{1-\alpha}$

Onde o consumo em ω foi descontado pelo fator β. Partindo-se desse modelo de dois períodos, constrói-se um modelo de horizonte mais amplo encadeando-se o problema num processo recursivo. Nesse âmbito, a variável de dotação irá se comportar como fonte alimentadora em cada estágio, cujo valor é fornecido pela solução do mesmo problema no par de períodos anterior. Para tanto, devemos explicitar o caminho estocástico das variáveis exógenas recursivas dentro do par de períodos. A dotação individual segue um caminho dado pela expressão:

(3) $d_{n+1} = d_n(1 + z + \tau\varepsilon_i)$

Onde z representa um termo determinista da tendência do crescimento da dotação e τ a variância do termo aleatório do crescimento da dotação. ε_i é um termo aleatório (uma flutuação com média 0 e desvio padrão 1) cujo valor específico, em cada caso, associa-se a um certo estado da natureza. Quanto ao retorno dos ativos, trabalha-se com a hipótese em que tais retornos seguem um processo GARCH. O processo de crescimento da dotação e do retorno de cada ativo *i* depende fundamentalmente do número aleatório ε_i, de distribuição normal. Cada estado da natureza caracteriza-se por uma coleção particular de ε_i, com um valor para cada ativo i e para a dotação. A ideia da simulação é associar um determinado estado da natureza a cada passo sucessivo. Na geração das variáveis e dos parâmetros, o modelo de simulação trabalha com 9 tipos de ativos e dez agentes em dez *experimentos*. τ e z são gerados por sorteio aleatório a cada experimento. Ao longo de períodos defasados, são fornecidos pelo programa um vetor de pesos envolvido no cálculo das médias de retorno e outro vetor de pesos envolvido no cálculo das médias de volatilidade dos ativos. Os desvios-padrão do ativo, em cada estágio, vão sendo sorteados e ponderados com os desvios-padrão do passado. O valor de β é informado pelo usuário. O preço do único bem de consumo é sempre 1 (ele funciona como numerário). Os ativos financeiros têm os preços expressos em termos de quantidades que poderiam ser trocadas entre cada ativo em questão e o único tipo de bem de consumo do modelo.

Veja-se como são feitas as estimativas da utilidade associada ao consumo de certo indivíduo em determinado experimento: no problema de maximização de utilidade no equilíbrio geral, começa-se observando as restrições. Para cada geração de valores aleatórios para as variáveis exógenas e parâmetros, o programa calcula a soma das diferenças entre consumo e dotação autônoma para os dez indivíduos. As condições de equilíbrio de mercado do bem de consumo e a de que o excesso de oferta dos ativos se anule são simultaneamente atendidas quando se encontra, por sorteios repetidos, uma combinação de valores das variáveis aleatórias e dos parâmetros, gerada a cada passo, que atenda à condição (4) abaixo, para um valor de ϵ suficientemente pequeno:

(4) $\sum_{h=1}^{H}(x^h - d^h) + \sum_{h=1}^{H}(\varphi^h - \psi^h) \leq \epsilon$

Atendida a essa condição subsidiária do problema, temos de prosseguir na solução do modelo de equilíbrio geral de dois períodos. Para cada tipo empregado de função de utilidade, interpreta-se certo experimento como sendo uma simulação em particular, para um dado conjunto de variáveis geradas por sorteio aleatório. A cada par de períodos adjacentes, na sequência de estágios estocásticos, o consumidor maximiza uma função utilidade. Os parâmetros da função utilidade são específicos aos indivíduos. Tais indivíduos também atuam no mercado de ativos. A carteira de ativos representa uma oportunidade de escolhas de consumos distribuídos no tempo que não dependem apenas da trajetória da dotação estocástica gerada por processo autônomo. O consumo x^h deve obedecer à restrição orçamentária de dois períodos, um par de $\omega's \in \Omega$. Em geral, o vetor que descreve o estado das variáveis aleatórias ω_t depende do estágio. Em nossos exercícios, estamos trabalhando fazendo-se com que ω_t dependa somente de um sorteio (ou

processo aleatório) independente realizado em cada par de estágios. A independência do passado diz respeito ao sorteio de valores atribuídos às variáveis aleatórias (comandadas por uma função de distribuição) que ocorre em cada estágio, porém há uma certa dependência dos períodos imediatamente anteriores, em relação às dotações e retornos dos ativos. Nota-se, ainda, que o processo de otimização se relaciona apenas com o par de estágios vizinhos considerados e não com a memória do processo estocástico antecedente ou com a projeção sobre o futuro.

Os diversos vetores de consumo individual em cada passo, determinados pelo sorteio das variáveis aleatórias do modelo nesse passo, e por valores de parâmetros aleatórios estabelecidos para o experimento e válidos para todos os passos, são avaliados em diferentes saídas do *loop* que geram, cada qual, um vetor específico com base no atendimento das condições de equilíbrio geral (a menos de uma margem de flutuação assumida). O programa calcula o valor da utilidade associada a cada entrada do vetor e seleciona dentre um conjunto de pontos do espaço orçamentário, chamado aqui de "candidatos", qual deles maximiza o consumo para o maior número de agentes. Esse procedimento de busca da utilidade máxima em cada par de estágios sucessivos na trajetória do processo estocástico é o que comanda a localização do caminho particular dentre os inúmeros caminhos possíveis pelo sistema. Ele funciona como a variável de controle na busca do caminho ótimo.

O modelo parte da ideia de que o controle do caminho estocástico ótimo pode ser feito resolvendo-se sequencialmente um problema de otimização de dois períodos. No eixo horizontal está a quantidade de consumo do único bem em questão no período *t*. No outro eixo, a quantidade do mesmo bem consumida em *t+1*. Nosso problema de otimização consiste em encontrar a combinação (x_t, x_{t+1}) que maximiza uma função de utilidade para cada indivíduo, genericamente descrita como sendo $U = U(x_t, x_{t+1})$. Em nossa simulação, impõe-se que o sistema precisa estimar o valor alcançado pela utilidade para um conjunto de pontos no $\mathbb{R}^2$ (mais propriamente, na fronteira orçamentária) dentre os pontos "candidatos", naturalmente pontos que atendam às restrições orçamentárias do problema. Nota-se que o consumo, estimado por agente, é uma variável aleatória. Vejamos como foram gerados os números para o consumo: o consumo de cada agente é estimado *endogenamente* por meio de uma equação de equilíbrio orçamentário na qual o montante consumido em dois períodos adjacentes equivale à soma das dotações em cada período mais o retorno financeiro da quantidade comprada de ativos, e ainda se soma a renda líquida nas transações de compra e venda de ativos nesta etapa. x_t e x_{t+1} são variáveis aleatórias (e endógenas para $t > 1$). Queremos que o programa simule um ponto que seja uma escolha maximizadora de utilidade. O programa tenta alcançar o ponto em questão comparando entre si os valores das utilidades, $U(x_t, x_{t+1})$, estimadas para todos os pontos candidatos. O programa permite inclui-se até 21 pontos candidatos. Para tanto, é preciso garantir algumas condições: o programa deve permitir mudanças tanto em x_t quanto em x_{t+1}. No exercício, é preciso que ambas as variáveis sejam aleatórias nas estimativas de utilidades associadas a diferentes pontos candidatos

a máximo. Isso significa que o processo de geração do valor testado de x_t deve ocorrer ao mesmo tempo em que se estima o valor testado de x_{t+1}. Temos de gerar x_t e o subsequente x_{t+1} no mesmo processo aleatório, em ambos aplicando-se a condição de que se atenda o equilíbrio geral de mercados no respectivo estágio, e impondo-se uma condição adicional para que o par (x_t, x_{t+1}) seja candidato a ser o ponto de equilíbrio maximizador, e que, portanto, eles estejam relacionados a uma equação orçamentária comum.

Para o exercício de maximização de utilidade de dois períodos, a fronteira orçamentária relevante é obtida da seguinte maneira: chamando-se o retorno financeiro dos ativos no período t $(retorno(\varphi^h(\omega))$ de r_t, e definindo-se o retorno líquido com as transações de ativos em t $(q.(\psi^h(\omega) - \varphi^h(\omega)))$ como sendo rl_t, teríamos, portanto, para o par de períodos considerados, a equação:

(5) $x_t + x_{t+1} = d_t + d_{t+1} + r_t + r_{t+1} + rl_t + rl_{t+1}$

A equação se refere a um agente, omitimos o índice h. O segundo membro da equação pode ser pensado como a restrição orçamentária individual para o problema da escolha intertemporal de consumo. Afinal, estamos pensando nessa escolha como se x_t e x_{t+1} fossem dois bens distintos em uma cesta de consumo. A restrição orçamentária a ser considerada deve ser a soma de duas dotações autônomas, uma para cada período, mais a soma dos retornos da carteira de ativos nos dois períodos, mais o retorno líquido das operações com compra e venda de ativos nos dois períodos. Naturalmente poderíamos considerar taxas de desconto δ que traduzissem preferências temporais ou juros. A valor presente, a equação ficaria assim:

(6) $x_t + x_{t+1} = d_t + (1+\delta)^{-1} d_{t+1} + r_t + (1+\delta)^{-1} r_{t+1} + rl_t + (1+\delta)^{-1} rl_{t+1}$.

A escolha do par de consumo (x_t, x_{t+1}) estaria então sujeita a essa restrição comum. O ponto candidato deve ser pensado como um par (x_t, x_{t+1}). Até aqui podemos conhecer apenas a soma $x_t + x_{t+1}$ que atende às condições orçamentárias imposta para a escolha entre dois períodos. Ora, isso não significa que não possamos definir valores específicos na escolha de um candidato. De fato, os sorteios aleatórios que consideraram a restrição comum apontam como candidato qualquer ponto ao longo da fronteira em questão. Mas devemos, na rodada de seleção do candidato, escolher um desses pontos. Naturalmente, o que maximiza a utilidade para o agente. O mesmo procedimento para os demais agentes considerados no exercício. Sendo assim, o método de escolha sugerido, para certo agente, é o seguinte: chama-se o segundo membro da equação (6) de C_t. Portanto, $x_t + x_{t+1} = C_t$. O primeiro x_t candidato poderia ser a metade do intercepto horizontal: $x_t = C_t/2$. E, portanto, $x_{t+1} = C_t/2$. Para nove pontos candidatos, conhecido o ponto candidato inicial $(x_t, x_{t+1}) = (C_t/2, C_t/2)$, procura-se outros oito candidatos para essa rodada de seleção: quatro à direita $\{(C_t/1{,}8, 4C_t/9), (C_t/1{,}6, 3C_t/8), (C_t/1{,}4, 2C_t/7), (C_t/1{,}2, C_t/6)\}$ e quatro à esquerda do ponto inicial $\{(C_t/2{,}4, 7C_t/12), (C_t/2{,}8, 9C_t/14), (C_t/3{,}4, 12C_t/17), (C_t/5{,}2, 21C_t/26)\}$. De modo análogo para um número maior de pontos candidatos.

Em seguida, o programa busca o valor máximo da utilidade aplicando-se a função utilidade a esses pontos. Para certo agente, quando a função utilidade $U = U(x_t, x_{t+1})$ assume o maior valor no ponto

(x_t^*, x_{t+1}^*), esse ponto configura os valores de x_t e x_{t+1} escolhidos na rodada. Trata-se do par de coordenadas em questão, o primeiro ponto escolhido nessa rodada de seleção. Naturalmente o processo se repede para os demais agentes no mesmo giro no *loop* do programa. A cada rodada, seleciona-se o par de candidatos (x_t, x_{t+1}) que, por construção, satisfaz à condição de máximo de utilidade do agente em questão (dentre os pontos pesquisados), o processo repetindo-se, no mesmo *loop*, para todos os demais agentes. Esses pontos realmente são pontos bem distribuídos ao longo da reta orçamentária e que atendem não apenas à condição de equilíbrio dos mercados (condição atendida quando o programa avalia as margens de flutuação alcançadas em cada giro), mas à condição do *equilíbrio maximizador* em cada agente.

Novamente procura-se comparar os pontos selecionados como candidato para ver qual é o *candidato dominante*. Dentre os pontos selecionados como candidato no processo anterior de escolha, por *candidato dominante* entendemos o ponto do hiperespaço no qual um número mínimo de agentes (informado pelo usuário) obtém nele o máximo de utilidade. Nosso método tenta localizar esse candidato (definido como uma combinação de carteira, preços e retornos dos ativos). O processo repete-se a cada janela de dois períodos. Em cada qual, há uma escolha (potencialmente) maximizadora (x_t^i, x_{t+1}^i) para cada agente *i*. Assim, chega-se à série estocástica de consumo $\{(x_0^i, x_1^i), (x_1^i, x_2^i), \dots, (x_{N-2}^i, x_{N-1}^i)(x_{N-1}^i, x_N^i)\}$. Nota-se que o x_1^i do primeiro par ordenado pode não ser igual ao x_1^i do segundo par, assim como o x_{N-1}^i do penúltimo par não necessariamente equivale ao x_{N-1}^i do último par, o mesmo em outras posições intermediárias de pares. Portanto a sequência de consumo maximizador em cada indivíduo *i* é descontínua em todos os pontos nos extremos dos intervalos discretos (exceto nos extremos do intervalo total envolvendo todos os estágios considerados).

No exercício de simulação, as escolhas de carteiras ocorrem de modo puramente aleatório, mas os agentes escolhem *carteiras eficientes*. O programa percorre um número de tentativas. Em cada qual se geram variáveis aleatórias, até atingir uma condição de equilíbrio geral e uma condição de escolha de carteiras eficientes. Sendo o *mercado eficiente*, vejamos como seria matematicamente possível escolher uma carteira ótima. À luz da conhecida teoria de construção de carteiras, com base na média e na variância, proposta por Markowitz, vejamos então como formalmente se constrói uma carteira eficiente. Tal carteira é a que maximiza o retorno esperado e minimiza o risco. Considere uma carteira de *N* ativos $(N = 1, \dots, N)$. Se a riqueza, ou o dinheiro disponível, é 1 (por simplicidade), a fração de uma unidade de riqueza investida no ativo *i* denomina-se de θ_i $(i = 1, \dots, N)$. Note que $\theta_i < 0$ também tem significado econômico. Traduz a situação de *short sale* (vender sem a entrega) (ou, ainda, toma-se emprestado indexado ao ativo). As taxas de retorno dos *N* ativos são expressas no vetor de componentes aleatórios $\boldsymbol{\rho} = (\rho_1, \dots, \rho_N)^T$. O retorno esperado é $r = E\,\boldsymbol{\rho} = (r_1, \dots, r_N)^T$. A matriz de covariância dos retornos dos ativos é representada por $\boldsymbol{V} = \left(\sigma_{ij}\right)$, na qual $\sigma_{ij} = cov(\rho_i, \rho_j), i, j = 1, \dots, N$. Nota-se que os elementos na diagonal dessa matriz são as

variâncias $\sigma_i^2 = \sigma_{ii}$. O desvio-padrão do ativo é considerado uma medida de risco. Certa combinação de ativos define uma carteira específica. O retorno esperado dessa carteira é dado por $r_p = \boldsymbol{r}^T\boldsymbol{\theta}$. Também se define para a carteira uma variância, entendida como a variância do retorno dela. Tal variância pode ser expressa como sendo $\sigma_p^2 = \boldsymbol{\theta}^T\boldsymbol{V}\boldsymbol{\theta}$. O risco da carteira é o desvio-padrão correspondente a essa variância.

Em vez de gerar-se escolhas puramente aleatórias de carteiras, trabalha-se com *carteiras eficientes*. Assim sendo, as quantidades compradas (φ_i) e vendidas (ψ_i) pelo agente *i*, em cada estágio do processo estocástico, são tais que, ao cabo, o indivíduo terá uma carteira eficiente $\boldsymbol{\theta}^*$. Vejamos como o agente constrói uma carteira eficiente. Olha-se apenas os retornos esperados, pelo agente, da carteira de ativos e para a estrutura de covariância dos retornos individuais de cada ativo uns com os outros. O problema da escolha individual da carteira eficiente pode ser formalizado da seguinte maneira: o agente busca atingir certo nível de retorno almejado μ com o mínimo de risco. A carteira correspondente é chamada de carteira de variância mínima, ou seja, que possui o menor risco. Formalizando-se:

(7) $min\ \frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{V}\boldsymbol{\theta}$ sujeito a $\mathbf{1}^T\boldsymbol{\theta} = 1$ e $\boldsymbol{r}^T\boldsymbol{\theta} = \mu$, para dado retorno esperado μ.

Nota-se agora qual a solução geral para o problema de minimização em tela para o caso em que só existam ativos com risco, e que seja permitida a presença de *short sale*. A ocorrência mais comum é quando $\mathbf{1}^T\boldsymbol{V}^{-1}\boldsymbol{r} \neq 0$. Para essa situação usual, a solução compatível com carteira eficiente é dada pela expressão $\boldsymbol{\theta}^* = \boldsymbol{z}_1 + \mu\boldsymbol{z}_2$, na qual, $\boldsymbol{z}_1 = \frac{1}{\Delta}(C\,\boldsymbol{V}^{-1}\mathbf{1} - B\,\boldsymbol{V}^{-1}\boldsymbol{r})$ e $\boldsymbol{z}_2 = \frac{1}{\Delta}(A\,\boldsymbol{V}^{-1}\boldsymbol{r} - B\,\boldsymbol{V}^{-1}\mathbf{1})$. Nessas expressões, $A = \mathbf{1}^T\boldsymbol{V}^{-1}\mathbf{1}$, $B = \mathbf{1}^T\boldsymbol{V}^{-1}\boldsymbol{r}$, $C = \boldsymbol{r}^T\boldsymbol{V}^{-1}\boldsymbol{r}$ e $\Delta = AC - B^2$. $\boldsymbol{z}_1$ e $\boldsymbol{z}_1$ podem ser pensados como carteiras e, portanto, a carteira eficiente seria uma combinação delas.[2] Impondo-se que as carteiras escolhidas de ativos, por cada agente, sejam sempre carteiras eficientes, a cada estágio do processo, e a cada *loop* sucessivo no teste das condições de equilíbrio geral, o vetor $\boldsymbol{\theta}^*$ das alocações de ativos é gerado com base em um processo aleatório. No entanto, não se gera um vetor aleatório qualquer para a definição da carteira de ativos comprados. O vetor é sempre de alocação eficiente, mas ele pode ser recalculado a cada sorteio de um conjunto de parâmetros que rege a equação que determina o $\boldsymbol{\theta}^*$para cada indivíduo.

A matriz $\boldsymbol{V}$, de covariância dos retornos, depende da sequência de números aleatórios ε_j. A trajetória efetiva do retorno dos ativos depende do sorteio da sucessão de estados da natureza ao longo dos estágios. Ainda mantemos a mesma matriz de flutuações aleatórias para todos os estágios do processo (para cada experimento) e também a cada rodada do *loop* que busca o atendimento das chamadas condições de equilíbrio geral, mas sortearemos a linha escolhida dessa matriz a cada tentativa dentro do *loop for*. O programa gera *n* linhas de estados da natureza que constituem a matriz com esse número de linhas e um número de colunas de entradas aleatórias correspondente ao número de ativos mais uma coluna para o termo aleatório da dotação. Um sorteio *da linha* de flutuações aleatórias ocorre várias vezes em cada estágio

[2] Ver Dupacová *et al.* (2001).

do processo. Ora, a cada sorteio de uma linha da matriz muda-se a trajetória temporal dos retornos efetivos dos ativos e muda-se também as relações de variância e covariância (pois estas são calculadas com base nas trajetórias efetivas, após o processo de sorteio, dos retornos de cada ativo, e, portanto, com os estados da natureza já conhecidos). Pode-se perfeitamente argumentar que o futuro é incerto, mas que o passado deve ser conhecido. Portanto, a cada estágio do processo estocástico, apenas os valores da matriz de entradas aleatórias do passo considerado seriam passíveis de sorteio, ficando congelados os *estados da natureza* relativos a estágios passados, cujas variáveis aleatórias já assumiram valores específicos. De fato, parece mais correto fixar a sequências de flutuações para a *janela* (estágios passados sucessivos e perfazendo, cada qual, um tamanho variado), excetuando-se a última entrada dela. Assim, temos uma janela de observação de tamanho inicial de 10 estágios, no interior do qual os sorteios, que caracterizam a trajetória temporal do retorno dos ativos, já foram estabelecidos. Podemos, a cada estágio do processo, e da análise, avançar com a janela, que cresceria de tamanho. Pode-se começar com a janela de valores aleatórios, que definem a trajetória de retorno dos ativos, já definida e com valores calculados para, por exemplo, dez períodos hipotéticos. Assim, no estágio 1 do processo estocástico sendo examinado, acrescentamos um período (o atual), cujos valores das variáveis aleatórias estão sendo sorteados e, portanto, por sorteio, determinamos a matriz $\boldsymbol{V}$. O procedimento se repetirá nos estágios seguintes, acrescentando-se novos dados, obtidos por sorteio e referentes ao estágio em questão. Nota-se que o sorteio é feito várias vezes no estágio em questão, pois diferentes valores para as entradas de $\boldsymbol{V}$ são testados até que o consequente valor de $\boldsymbol{\theta}^*$, por indivíduo, atenda às condições de equilíbrio geral que estão sendo testadas no *loop*. Ou seja, por esse método tornamos a matriz $\boldsymbol{V}$ aleatória a cada estágio e também a cada rodada nesse estágio. Alterando-se a última entrada na janela temporal dos retornos, também se pode afetar o retorno esperado de cada ativo, as médias dos valores calculados dentro da janela. Assim também a variável $\boldsymbol{r}$ está sujeita a sorteios repetitivos no estágio em questão. Em dado estágio do processo, as quantidades demandas, por cada agente, de cada ativo são as que mantém a carteira eficiente no estado sorteado. Além de se avaliar as condições de equilíbrio geral, os sorteios sucessivos geram diferentes carteiras que são, todas elas, carteiras eficientes. São eficientes dados os valores correspondentes estimados para as entradas da matriz dos coeficientes aleatórios, e, portanto, para a matriz de covariância dos retornos dos ativos e o vetor de retorno esperado.

Observe que ainda não se discriminam os agentes quanto ao vetor $\boldsymbol{\theta}^*$, de carteira eficiente, específico a cada agente. Todas as carteiras devem ser ao mesmo tempo eficientes; e mantida a estrutura comum de $\boldsymbol{V}$ e $\boldsymbol{r}$ parece que não haveria como distinguir os agentes nesse processo de escolha de carteiras eficientes. Na verdade, também aqui se pode distinguir os diferentes agentes, pois o problema de minimização do risco da carteira ($\frac{1}{2}\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{V}\boldsymbol{\theta}$) está colocado para cada agente em função do retorno almejado da carteira, expresso na variável μ. Esse retorno seria, portanto, específico de cada agente (ou de cada grupo representativo dele).

Na fórmula, $\boldsymbol{\theta}^* = \boldsymbol{z}_1 + \mu \boldsymbol{z}_2$, a escolha da carteira ótima ($\boldsymbol{\theta}^*$) depende de μ. Então cada agente *i* se caracteriza por um μ_i próprio, que faz com que a carteira eficiente dependa fundamentalmente de cada qual.

Como se determinam as quantidades compradas e vendidas de ativos em cada estágio do processo estocástico? No primeiro estágio, cada agente usa parte de sua dotação inicial para a compra da carteira. O restante da dotação é usado para o consumo inicial. Nesse passo do processo, as quantidades efetivamente compradas de ativos são determinadas pelo vetor $\boldsymbol{\theta}^*$ de carteira eficiente, específico a cada agente. Há, nesse estágio, apenas a compra de ativos, pois é nele que o agente monta sua carteira de aplicações, é nele que começa o processo de diversificação entre consumo e investimento financeiro. Constrói-se então a carteira inicial de ativos. A partir do segundo estágio, a carteira eficiente, para cada agente, muda de $\boldsymbol{\theta}_t^*$ para $\boldsymbol{\theta}_{t+1}^*$. Então define-se aí as quantidades de ativos compradas (φ_i) e vendidas (ψ_i), nesse estágio, para cada agente *h*, como sendo simplesmente a diferença $\theta_{h\,t+1}^* - \theta_{h\,t}^*$, com entradas positivas representando os valores comprados (φ_i^h) e entradas negativas relativas a valores vendidos (ψ_i^h).

2. Retornos que seguem um processo GARCH.

O retorno de um ativo pode ser expresso simplesmente como um ganho com variações de preços. Ou o contrário: conhecido o vetor de preço inicial, no primeiro estágio, e a série de retornos em cada estágio, pode-se determinar os preços dos ativos com base em seus respectivos retornos. Propor-se aqui uma boa forma de modelar, para efeito de simulação, a série temporal dos retornos estocásticos dos ativos.[3] Um modelo geral para séries de tempo, aplicado ao retorno financeiro, poderia ser o seguinte: $r_t = \mu_t + \varepsilon_t$, onde ε_t pode ser escrito como $\varepsilon_t = \sigma_t \xi_t$. ξ_t é o ruído branco com o comportamento tradicional ($\xi_t \sim i.i.d\ (0,1)$). μ_t é a média dos retornos passados e σ_t o desvio-padrão com base nas observações do passado. Quando a média μ_t é associada ao prêmio de risco, ele tradicionalmente fica vinculada ao desvio-padrão σ_t. Trata-se do conhecido modelo ARCH da média (ARCH-M), no qual, por exemplo, $\mu_t = s + \ln \sigma_t^2$. A variável *s* é interpretada como sendo a taxa de retorno sem risco.[4]

Portanto, ter-se-ia a fórmula $r_t = s + \ln \sigma_t^2 + \sigma_t \xi_t$ que descreve o retorno do ativo no tempo *t* em função da taxa de retorno livre de risco, do desvio-padrão observado e do ruído branco em *t*. O desvio-padrão σ_t (ou a variância σ_t^2) pode ser modelado assumindo-se que ele tenha uma dependência linear, aos pesos α_i, dos erros ε_t passados, até *q* períodos anteriores ao *t* atual. Então assume-se a expressão $\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2$. Onde ω é simplesmente um termo determinista. Com essa expressão, o modelo ARCH-M é convertido no

[3] Veja, a respeito, Härdle e Hafner (2000).
[4] O nome ARCH vem da sigla em inglês "autoregressive conditional heteroskedasticity". O modelo ARCH-M foi proposto por Engle, Lilien e Robens (1987).

modelo ARCH (q).[5] Pode-se generalizar o modelo ARCH (q), imaginando-se a existência de um processo do tipo ARMA para σ_t^2. Chega-se então ao modelo GARCH (p, q).[6] Esse último modelo toma a forma:

(8) $\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2$.

Portanto, a equação completa para a estimativa do retorno do ativo financeiro, com todas assas hipóteses e considerações, fica sendo:

(9) $r_t = s + \ln\left(\omega + \sum_{i=1}^{q} \alpha_i (\sigma_{t-i}\xi_{t-i})^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2\right) + \sqrt{\omega + \sum_{i=1}^{q} \alpha_i (\sigma_{t-i}\xi_{t-i})^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2}\, \xi_t$.

Modelos do tipo GARCH (p, q) oferecem muitas vantagens sobre a formulação tradicional da heterocedasticidade condicionada, contida na família ARCH. A hipótese GARCH modela a volatilidade de um modo mais natural. A ideia original, contida em modelos de volatilidade do tipo ARCH, era a de que o valor de σ_t dependeria dos resíduos passados (ε_{t-i}) do processo. No entanto, esse modelo apresenta a séria deficiência de não considerar adequadamente o impacto de choques assimétricos na volatilidade. De fato, "boas notícias" não possuem necessariamente o mesmo impacto na volatilidade que as "más notícias". Acredita-se que modelos GARCH (p, q) contornariam esse problema.[7] Vicejam algumas dificuldades em implementar-se a fórmula para processos GARCH em modelos para simulação numérica em computador. Na expressão (9), s e ω são termos deterministas que podem ser estimados por simples calibragem do modelo. α_i's e β_j's são pesos, envolvidos no cálculo de médias, que podem ser estipulados por sorteio e pela escolha do programador. Como usual, iremos conferir pesos maiores a períodos mais recentes. ξ_t's são fatores aleatórios que podemos obter gerando-se, no programa Matlab, matrizes de entradas aleatórias.

Os valores para q e p devem ser escolhidos com critério e mudam de acordo com o estágio do processo estocástico considerado na simulação. Para efeito da simulação, q mede o tamanho da série no estágio considerado do processo estocástico. p também pode ser o tamanho da série de tempo, mas ele representa o número de períodos, na série, em que, no interior deles, a volatilidade permanece constante. Cada σ_t será gerado pela fórmula (8), e, portanto, ele dependerá de pesos, da série de valores dele mesmo no passado e da série de ruídos brancos (ξ_t).

Nota-se como seria possível implementar a fórmula de cálculo do retorno do ativo no exercício de programação para efeito de simulação numérica. Uma série de retornos do ativo no passado deve ser criada e oferecida, estando disponível já na simulação do resultado numérico para o primeiro estágio do processo

[5] Sobre o modelo ARCH(q), vide Engle (1982).

[6] Na análise estatística de séries temporais, modelos do tipo ARMA (*autoregressive-moving average*) fornecem uma descrição de um processo estocástico estacionário (e simples) em termos de dois polinômios: um para a autorregressão e outro para a média móvel. Modelos do tipo ARMA foram descritos, em 1951, na tese de Whittle (1951), e popularizados no livro de George Box e Gwilym Jenkins, publicado vinte anos depois. Dada uma série temporal de dados, o modelo ARMA trata-se de uma ferramenta para a previsão aproximada de valores futuros da série. O modelo consiste em duas partes: uma que descreve processos autorregressivos e uma que assume uma média móvel. O modelo é geralmente referido como modelo ARMA (p, q), onde p é a ordem da parte autorregressiva e q é a ordem da parte associada à média móvel.

[7] Um *survey* sobre vários modelos paramétricos que contornam o "problema da simetria" é oferecido por Engle e Ng (1993). Härdle e Hafner (2000) destacam três modelos especialmente eficazes no combate ao "problema da simetria": modelos do tipo EGARCH, modelos ARCH-limiar (*threshold*) (TARCH) e ARCH-limiar qualitativo (QTARCH). Härdle e Tsybakov (1997) propuseram uma interessante versão em que a volatilidade dos retornos do ativo é modelada em função dos próprios retornos passados. Os mesmos autores, com um coautor adicional, estendem o modelo para o caso multivariado, Härdle, Tsybakov e Yang (1996).

estocástico considerado. Iremos supor a construção prévia de uma série de tempo dos retornos dos ativos do tamanho de dez passos. Isso define o tamanho inicial da janela temporal de análise. A partir do primeiro estágio da simulação, a janela, de tamanho crescente, avança no tempo. Portanto, já a partir do primeiro estágio examinado (estágio inicial no qual se testam as condições de equilíbrio do processo estocástico) disporemos de uma série na qual a janela temporal de análise irá caminhar.

Para cada ativo *i*, a criação da séria prévia de retorno, com tamanho de dez passos, pode ser feita assim: fornecem-se *s* e ω como valores conhecidos e fixos (ω é específico ao ativo em questão) Em $t = 1$, a α_1 e β_1 atribui-se valor 1. $q = p = 1$ e o desvio-padrão é calculado pela fórmula. Deve-se escolher arbitrariamente um valor para σ_0 (específico ao ativo) e sortear-se o fator de ruído branco (ξ_0). Sejam valores específicos para *s* e ω. O valor de σ_1 é calculado então. A expressão do retorno do ativo permite obter $r_1 (= s + \ln \sigma_1^2 + \sigma_1 \xi_1)$. O ξ_1 deve ser sorteado. Portanto, calcula-se o primeiro retorno da série para esse ativo. Ainda para o mesmo ativo, inicia-se agora a estimativa do seu retorno para o segundo termo da série de retorno: em $t = 2$, a α_1 e β_1 atribui-se valor 0,6 e a α_2 e β_2, o valor 0,4 (estamos dando o maior peso ao passo mais recente). Agora $q = p = 2$. Conhece-se o valor para σ_1 (específico do ativo), mas deve-se calcular o valor de σ_2. Sejam os mesmos *s* e ω, e um novo termo de flutuação gerado (ξ_1). O novo desvio-padrão (σ_2) é calculado. Obtém-se também o novo retorno do ativo (r_2). O ξ_2 é agora sorteado. Calcula-se o segundo retorno da série para esse ativo. Seguindo esse mesmo procedimento para os demais oito passos, temos, portanto, uma série de taxas de retorno, de desvios-padrão e de resíduos, tal que, nos estágios do processo estocástico em que se estará testando as condições de equilíbrio, pode-se calcular sempre, a cada passo do processo, a fórmula GARCH (*p*, *q*) para r_t. Nota-se que, desde o primeiro estágio dos 10 estágios em que iremos estimar as condições de equilibro do processo estocástico, já temos as estimativas dos parâmetros *s* e ω, bem como dos σ_{t-i}'s e dos ξ_{t-i}'s, exceto os valores destas duas últimas variáveis assumidos para o período em questão. Estes últimos são estimados ao mesmo tempo em se faz a janela crescer. Podemos, assim, obter a série completa dos retornos dos ativos, fazendo-se a janela avançar por 10 passos. Teremos, ao cabo, informações de retorno e volatilidade em 20 passos (10 deles antes mesmo dos estágios considerados na análise de equilíbrio estocástico). Da posse da série de retornos, podemos estimar a série dos preços dos ativos pela expressão $p_t = r_t p_{t-1} + p_{t-1}$. Para tanto, temos de escolher arbitrariamente apenas o vetor inicial p_0.

De fato, modelam-se ativos financeiros com retornos independentes, mas não com volatilidade igual e retornos identicamente distribuídos. Não se trata de um processo i.i.d., até porque os parâmetros *s*'s, ω's, σ_{t-i}'s e dos ξ_{t-i}'s não são iguais entre diferentes ativos. Mas as variações temporais dos retornos são supostas independentes. Isso é um problema, pois não se está considerando a possibilidade de que as diferentes séries de retornos ensejam processos de arbitragem. Uma maneira de garantir que não haja

oportunidades de arbitragem é fazer com que todos os ativos tenham o valor presente descontado igual aos respectivos preços na data de referência para o desconto, que pode ser o passo inicial da série de 20 passos que estamos gerando.[8]

3. Programa em Matlab para a simulação numérica: motivação e procedimentos.

O usuário deve optar pelo número de tentativas, o número de vezes em que o programa gera as variáveis aleatórias do modelo até atingir uma condição. Deve informar também a margem de flutuação máxima admitida. A condição de margem é verificada pelo programa. Considerando-se que foram calculados os consumos e as dotações dos 10 indivíduos a cada passo do experimento, a ideia inicial seria tomar-se as diferenças entre dotação e consumo para cada indivíduo e somar essas diferenças. Far-se-ia o mesmo para as diferenças entre ativos comprados e vendidos. A soma dos valores absolutos dos dois resultados forneceria a flutuação verificada. Assim, se essa flutuação fosse menor que a margem previamente estipulada pelo usuário, o programa forneceria como dado de saída o número de *loops* realizados até se obter tal condição e qual foi a flutuação encontrada. Se fosse maior, o programa continuaria gerando números aleatórios, em novos *loops*, até que a condição seja satisfeita. No entanto, não é necessário somar essas duas margens. A condição de normalidade do vetor de carteira eficiente impõe que a soma dos percentuais da riqueza gasta com a compra e a venda de ativos resulte em 100%. Esse fato muda com a presença de *short sale*, pois as vendas a descoberto financiam as aquisições de outros ativos da carteira. A normalização já *quase* garante a condição de que o total de ativos vendidos seja igual ao total de ativos comprados (no contexto estocástico, a menos de uma condição de margem de flutuação). Quase, mas não totalmente, pois as operações de venda a descoberto (*short sale*) podem fazer com que o total de ativos comprados não corresponda ao total de ativos efetivamente vendidos e entregues. Contudo, impõe-se uma condição para o equilíbrio geral que estabelece limites às vendas a descoberto, evitando-se assim situações explosivas em que os agentes se endividam com *short sale* excessivo. Tal condição assevera que a somatória das quantidades de ativos vendidos no primeiro período para todos os indivíduos (*short sales*) seja aproximadamente igual (a menos de uma margem de tolerância) aos ativos vendidos ou *short sales* no segundo período (o que implica que não há, no período, ativos efetivamente vendidos pelos próprios agentes considerados no modelo quando se considera a totalidade das operações e dos agentes). Com base nessas ideias, o programa impõe a condição de equilíbrio geral, que deve ser atendida dentro da margem de flutuação admitida pelo usuário.[9]

---

[8] Wolfgang Härdle e Christian M. Hafner (2000) argumentam que, sob certas condições, um processo GARCH, do tipo que estamos considerando na simulação, já garantem ausência de oportunidades de arbitragem. Os autores afirmam que "*quando o modelo GARCH (1, 1) é estimado sob a medida empírica P, aplicando-se uma metodologia de precificação com neutralidade a risco (veja Cox e Ross (1976)), a medida tem de ser transformada tal que o processo resultante de preço descontado seja uma martingale. Isso garante que não haja oportunidades de arbitragem (Harrison e Kreps (1979))*". (p. 5).

[9] Em alguns casos, especialmente para estágios mais avançados do processo estocástico em analise, a condição pode não ser atendida dentro da margem estipulada pelo usuário. O programa busca o atendimento dela apenas num número de giros informados pelo usuário.

O programa calculada os consumos e as dotações no passo em questão do experimento pelo seguinte processo: uma série de variáveis aleatórias é gerada e servirá para todos os passos do experimento em questão. Um *contador* assume o valor 1 e pede-se o número de candidatos e o número de agentes que maximizam a utilidade. Por *candidatos*, conforme já explicado, entende-se os pontos do espaço orçamentário que serão examinados (tais pontos dependem do agente em questão). O usuário informa o número mínimo de agentes (dos 10 existentes no modelo) que terão, no candidato, o valor de utilidade acima de todos os outros valores de utilidade encontrados para o mesmo agente quando se percorre todos os pontos candidatos. Muitas vezes, o programa não consegue, em tempo hábil, chegar a um resultado em que a utilidade é máxima simultaneamente para todos os dez agentes. Cabe ao usuário, então, informar quantos agentes atenderão à condição. Naturalmente, quanto mais agentes exigidos maior será o tempo de processamento da máquina. A ideia é a de trabalhar-se com o maior número de candidatos (entre outras vantagens, a fim de se evitar que o máximo de utilidade encontrado seja um máximo local e não um máximo global do conjunto de restrição estocástica). Acontece, porém, que quanto maior o número de candidatos, menor a possibilidade de que um deles maximize a utilidade do mínimo de agentes requerido, dentre os dez. Após atribuir valores gerados aleatoriamente aos parâmetros das equações, começa-se então a sucessão de experimentos. O comando "*tic*" do Matlab demarca o início da contagem de tempo feita pelo programa. Pois, queremos contar o tempo consumido em cada passo até que a condição estipulada seja atendida. Fornecido pelo usuário o número mínimo de agentes que maximiza a utilidade, a condição de *loop while* ordena que o percurso seja repetido até que o contador seja maior ou igual ao número informado de maximizadores. No contexto de cada candidato, ainda no *loop while*, inicia-se um *loop* com o *for*.

A cada rodada, certo contador crescerá unitariamente até alcançar o valor informado pela variável que controla o número de percursos repetidos no *loop for*. A variável em questão informa o número de tentativas, o número de vezes em que o programa gera as variáveis aleatórias do modelo até atingir uma condição. Geram-se números aleatórios que condicionam a matriz de covariância dos retornos. Note que, assim como os retornos dos ativos, suas quantidades compradas e vendidas serão recalculadas a cada rodada do *loop*, de modo a serem compatíveis com carteiras eficientes. Os preços dos ativos são calculados com base nos respectivos retornos. A diferença entre ganhos com venda de ativos e gastos com as compra deles é registrada numa variável. Ainda para esse passo, o programa gera, para cada indivíduo, a dotação e o consumo. O programa, em seguida, entra no processo de se testar a condição de margem de flutuação máxima. Se atendida, interrompe-se o *loop*. Um contador assume o valor 1 e, por meio do comando *break*, sai-se do *loop* iniciado pelo *for* anterior. Caso contrário, o sistema primeiro registra a flutuação encontrada como uma entrada de um dado vetor e repete o giro do *for* até percorrer o número de tentativas anteriormente informado pelo usuário. Em todas as tentativas permitidas, caso o sistema não tenha alcançado a condição de flutuação inferior à margem máxima autorizada, o programa informará que o

número de *loops* efetivos é igual ao número de tentativas e indicará, de todos as flutuações verificadas na sequência de repetições, qual fora a margem efetiva mínima verificada.

Em seguida, no *loop* para *for* associado ao número de candidatos, o programa calcula a utilidade de cada agente no passo em questão. Trabalha-se com simulações para diversos tipos de função utilidade. No problema de maximização de utilidade entre dois períodos consecutivos, inicialmente, emprega-se uma função de utilidade do tipo Cobb-Douglas (CD). A função assume a forma: $U(x_1, x_2) = x_1^{\alpha} x_2^{1-\alpha}$. Além disso, adaptamos o programa para outros tipos de função no intuito de avaliar como os tempos de processamento se alteram quando se muda a especificação do modelo nesse sentido. Assim utilizamos diferentes tipos de função utilidade no problema de maximização. Sete outros tipos foram testados, além da Cobb-Douglas:

i. Uma função de utilidade com *elasticidade de substituição constante* (CES):

   $U(x_1, x_2) = [\alpha \times x_1^{\rho} + (1-\alpha) \times x_2^{\rho}]^{\frac{1}{\rho}}$; $0 < \alpha < 1$; $|\rho| <= 1$

ii. Função de utilidade para *bens substitutos perfeitos* (SP): $U(x_1, x_2) = \alpha \times x_1 + x_2$.

iii. Do tipo *complementos perfeitos* (CP): $U(x_1, x_2) = \min [\alpha \times x_1, x_2]$.

iv. *Preferências quase-lineares* (PQL): $U(x_1, x_2) = v(x_1) + x_2$, ou $U(x_1, x_2) = \sqrt{x_1} + x_2$.

v. *Stone-Geary* (SG): $U(x_1, x_2) = (x_1 - \lambda_1)^{\beta_1} \times (x_2 - \lambda_2)^{\beta_2}$, também conhecida como utilidade associada a funções de demanda hicksianas, quando $\beta_i > 0$ e $(x_i - \lambda_i) > 0$.

vi. Utilidade associada a *funções de demanda marshallianas* (UDM):

   $U(x_1, x_2) = a_1 \log(x_1 - \lambda_1) + a_2 \log(x_2 - \lambda_2)$; $a_i > 0$; $(x_i - \lambda_i) > 0$ e $a_1 + a_2 = 1$.

vii. *Função de utilidade translog* (FUT): definida como $-\ln U(x_1, x_2) = a_0 + a_1 log\ x_1 + a_2 log\ x_2 + \frac{1}{2} a_3 log\ x_1\ log\ x_2$ para uma aproximação de segunda ordem de uma função de utilidade arbitrária. Ou, isolando a utilidade no primeiro membro: $U(x_1, x_2) = e^{-[a_0 + a_1 log\ x_1 + a_2 log x_2 + \frac{1}{2} a_3 log\ x_1\ log x_{12}]}$.

Deve-se observar quais mudanças são necessárias quando se muda a especificação da função utilidade em relação ao programa para o caso de utilidade Cobb-Douglas. Começa-se com o caso da função de utilidade CES. Nesse, além do parâmetro α ($0 < \alpha < 1$), teve-se de gerar, por meio de uma função de distribuição, o parâmetro ρ, com $|\rho| <= 1$. Então temos a sequência específica de programação. Para o caso de função de utilidade para bens *substitutos perfeitos* (SP), é necessário apenas o parâmetro α, gerado da mesma forma. São as únicas mudanças necessárias quando se muda a especificação da função utilidade para SP em relação ao programa para o caso de utilidade Cobb-Douglas. No caso de função de utilidade do tipo *complementos perfeitos* (CP), como parâmetro dessa função, é necessário apenas o parâmetro α, gerado com base no mesmo processo do α da Cobb-Douglas. Idem para o caso de utilidade do tipo SP. Mas naquele tem-se uma fórmula de cálculo específica que utiliza a função *min* do Matlab. No caso de utilidades do tipo PQL, não se necessita gerar o valor aleatório de nenhum parâmetro específico. No caso de função CES, o vetor de dez entradas para o parâmetro *alpha* foi gerado como no caso da função utilidade do tipo CD, mas

agora se tem o vetor para o parâmetro ρ, cuja geração aleatória de valor está programada de tal forma que o módulo de cada entrada, de fato, fica abaixo ou igual a 1, pois estamos subtraindo de um número entre 0 e 1 um número entre -1 e 0. Essas, portanto, são as únicas mudanças necessárias quando se muda a especificação da função utilidade em relação ao programa para o caso de utilidade Cobb-Douglas. No caso da função de utilidade Stone-Geary (SG), a função de utilidade da qual se busca o máximo tem a correspondente expressão. Nota-se que, nesse caso, foram empregados dois vetores de parâmetros gerados aleatoriamente, *l1* e *l2*, (representantes de λ_1 e λ_1), conforme mostra a passagem do programa acima. Os valores de *b1* e *b2* (representantes de β_1 *e* β_2) foram obtidos pela função *rand* do Matlab. Nota-se que se atende à condição $\beta_i > 0$. Na demais funções de utilidade, os parâmetros dessas funções são gerados uma única vez e os valores assim obtidos são empregados em todos os passos do experimento em questão. No entanto, os vetores de parâmetros *l1* e *l2* , da função SG, são gerados de modo repetido a cada giro no *loop*. A geração repetida desses parâmetros aleatórios a cada giro é uma maneira de fazer com que a condição $(x_i - \lambda_i) > 0$ seja sempre satisfeita (pois os x_i são estimados a cada giro). Nota-se, pela fórmula de cálculo no Matlab, que as componentes de l_i (os λ_i) serão necessariamente menores que as respectivas componentes em x_i. Outro comentário específico é feito em relação às utilidades associadas a funções de demanda marshallianas. Neste caso, temos dois parâmetros aleatórios a_1 e a_2, positivos e tais que $a_1 + a_2 = 1$. Há novamente os parâmetros λ_1 e λ_2 e a condição $(x_i - \lambda_i) > 0$. A solução em termos de Matlab para esse caso é a seguinte: gera-se a variável aleatória a uma única vez, como sendo o a_1. Naturalmente $a_2 = 1 - a_1$. No *loop* em questão, gera-se o vetor de valores de λ_1 e λ_2 e a correspondente estimativa da função utilidade. Por construção, temos novamente que $(x_i - \lambda_i) > 0$. Finalmente, tem-se o caso para a função de utilidade *translog*. Aqui precisamos gerar aleatoriamente quatro parâmetros: a_0, a_1, a_2 e a_3. Tais parâmetros são gerados uma única vez e os valores obtidos são empregados em todos os experimentos. O programa gera-os pela função *rand*. Com essas alterações no formato das funções utilidades podemos avaliar o modelo de simulação com diferentes especificações da utilidade e testar se esse aspecto do modelo afeta as probabilidades de convergência à condição de equilíbrio avaliadas (aferidas pelo tempo de processamento).

Ainda dentro do giro associado ao comando *for* para número de candidatos (antes, portanto, de percorrer todos os candidatos informados), o sistema, em cada rodada, cria uma linha matricial para as escolhas dos agentes. O próximo procedimento consiste em comparar-se os diferentes vetores de consumo associados a essas linhas e verificar a condição de máximo de utilidade. Primeiramente, o sistema buscará em qual giro se obtém o máximo de utilidade em comparação à utilidade alcançada em outras rodadas. Isso é feito por indivíduo, comparando-se sempre as utilidades relativas ao mesmo indivíduo (não há comparação interpessoal de utilidade). Em seguida, o programa localiza em quais linhas aparecem os "candidatos vencedores" (máximo de utilidade). As diferentes linhas vencedoras, em cada giro a coluna da

matriz em questão, são arranjadas para formar certo vetor. O sistema mostra os candidatos e calcula a incidência máxima de candidatos pelo comando *mode*. A *moda* em questão fica registrada numa variável. Um contador registra, ao final, quantos agentes maximizam sua utilidade na linha que é considerada a *moda*.

Conclui-se o *loop* iniciado com o *while* quando se atinge o número mínimo de agentes que maximizam a utilidade informado pelo usuário. Em seguida, o sistema calcula o tempo despendido na rodada específica do *while*. Para tanto, ele marca a passagem pelo uso do comando do Matlab *toc* aplicado ao tempo inicial do *tic* registrado numa variável. O tempo final fica registrado em outra variável.[10] O programa anuncia quando o tempo para a totalidade dos loops do *while* ultrapassa um máximo de 240 segundos. Caso o processamento exceda esse tempo, o programa enviará uma mensagem de alerta. Neste caso, ele limpa o conteúdo do contador de tempo e reinicia o programa do passo em questão. Nota-se que o sistema nem sempre consegue atingir o número mínimo de agentes informado que tem a utilidade maximizada dentro do tempo máximo estipulado. O tempo efetivo consumido até que se alcance o número mínimo de agentes que obtém o máximo de utilidade no candidato escolhido é exportado para uma variável que aparece como entrada de um vetor que cobre todos os passos do processo estocástico. O programa gera saídas para as matrizes de utilidades geradas e de consumo. A alocação do candidato vencedor no passo do experimento é registrada num vetor. A *procedure (proc)* do passo termina com uma saída mostrando as dotações de todos os agentes nesse passo, a outra variável *endógena* nesta etapa, junto ao consumo de equilíbrio maximizador e as utilidades.

Na *proc* para o nono passo (cada passo com dois períodos), viceja uma sequência adicional de linhas de programação. Nessas linhas, programa-se a exibição de dois gráficos. No primeiro deles, distribuições de consumo por uma função normal são exibidas para 10 períodos. Outro gráfico é gerado, no qual se mostra a trajetória dos valores de consumo, obtidos pelas *modas* das respectivas distribuições de frequência. Nota-se que um vetor é construído com entradas que correspondem ao consumo médio estimado em cada um dos dez passos do experimento. Após se completar o último passo do experimento, o programa atualiza o contador de experimentos (sequência de sorteios) a fim de se referir apropriadamente ao próximo deles. Ao cabo de cada experimento, a *proc* apresenta um conjunto de resultados obtidos nos dez passos dele: o preço dos ativos em casa passo; as quantidades compradas desses ativos em cada passo; as respectivas quantidades vendidas; os retornos dos ativos nos 10 estágios (e também na janela virtual anterior de mesmo tamanho); bem como os parâmetros fixos τ e z. A *proc* em questão também gera saída para os tempos de processamento em cada passo. A *proc* gera, em seguida, um gráfico ilustrativo representando a evolução dos preços dos ativos no experimento de dez passos e, em outro, a trajetória do retorno dos ativos.

[10] O tempo total alocado no *loop* em questão também pode ser lido na variável *toc*. Mas note que esse tempo é estimado em termos de uma unidade interna da máquina, sem significado nas unidades padrão de tempo. Para o registro do tempo de processamento em *segundos*, utiliza-se a variável *elapsedTime*, montando-se a equação "*elapsedTime=toc*".

À luz dessa exposição, faz-se um programa em Matlab inicialmente para o caso da função de utilidade do tipo Cobb-Douglas (procedimentos inteiramente análogos foram aplicados aos outros tipos de função utilidade considerados). O programa começa perguntando sobre o número de estados da natureza, o número de tentativas, a margem de flutuação admitida e o número de candidatos. Pede-se que o número de candidatos seja ímpar. O número de agentes que efetivamente estarão maximizando a utilidade será informado, pelo usuário, a cada passo. Estima-se a amplitude do termo aleatório da taxa de crescimento da dotação (τ) pela equação $tau = 0{,}015 \times (1 + rand(1))$, e o termo determinista da taxa de crescimento da dotação (z) por $zeta = 0{,}055 \times (1 + rand(1))$. A variável α (o expoente da função Cobb-Douglas) assume valores, igualmente espaçados, entre 0 e 1. Supomos que a componente autônoma da dotação (a que não depende das transações com ativos financeiros) seja, no primeiro estágio, duas vezes a magnitude do gasto com a compra de ativos financeiros. A seguir, o programa estima o retorno da carteira de ativos no período em questão. A aplicação de cada agente alcança um retorno distinto, pois cada qual faz uma combinação diferente de ativos em sua carteira. Pesquisa-se a estimativa do retorno dos ativos no primeiro estágio. Especificam-se depois o retorno da carteira associada a cada agente. Portanto, o vetor de retornos no estágio inicial para os dez agentes pode ser construído. Agora pode-se montar a linha de programação para a restrição orçamentária envolvida no problema de maximização em dois períodos. O programa seleciona os pontos candidatos pelo método já comentado. O usuário do programa já informou o número ímpar de candidatos, ou de pontos da fronteira orçamentária estocástica que serão examinados quanto ao correspondente nível de utilidade. A ideia é selecionar-se pontos bem distribuídos ao longo da fronteira orçamentária para dois períodos a fim de se avaliar em qual ponto a utilidade atinge um máximo. Com essa escolha de pontos, tem-se sempre, para qualquer número de candidatados informados (número ímpar de 3 a 21) um conjunto de pontos bem distribuídos (aproximadamente uniformemente distribuídos) ao longo da fronteira orçamentária estocástica.

Agora segue a parte de programa que testa o atendimento das condições de equilíbrio geral dada pela fórmula (4). Nota-se que cada candidato (um par de valores consumidos) deve ser testado quanto ao atendimento das condições de equilíbrio geral. Caso não se atenda à margem de flutuação estatística, ele é gerado mais vezes até que se fique dentro dessa margem. A condição se aplica à soma $x_1 + x_2$ e não a cada x_i, ou seja, as condições de equilíbrio geral (e de ausência de excesso de oferta do ativo) não são impostas a cada estágio do processo, mas apenas para o problema de dois estágios considerados. Em seguida, impõe-se o atendimento da margem, ou seja, das condições de equilíbrio geral no problema com dois estágios. Nas rodadas sucessivas do *loop* em questão, o sistema gera, para cada estágio, um correspondente vetor com todas as margens estimadas. Caso em nenhuma rodada tenha sido satisfeita a condição de margem de flutuação máxima, o sistema retorna o melhor resultado em cada estágio (percorrida todas as tentativas informadas pelo usuário). Selecionado o *candidato* (ponto com duas coordenadas, o consumo estimado em

cada um dos dois estágios considerados), o programa calcula a utilidade associada a este ponto candidato. Uma sequência de programação mostra o vetor com os valores consumidos de todos os candidatos nos dois estágios. O programa irá identificar o ponto candidato que resulta no máximo de utilidade estimada numericamente. Ele calcula então o tempo de processamento, em segundos, para a escolha de cada candidato nesse passo de dois estágios. A *proc* inicial mostra os valores das quantidades consumidas nos dois estágios que maximizam a utilidade e termina evocando a próxima *proc*, que tratará dos estágios 2 e 3. Esta última começa pedindo novamente o número de agentes que maximizam sua utilidade. Segue a rotina *while*. Na estimativa envolvendo os estágios 1 e 2, a dotação do primeiro período d_1 era definida com base no gasto com a compra inicial de ativos financeiros. No período seguinte, o valor efetivo da dotação (d_2) depende do sorteio particular de um termo de flutuação que está sendo feito na simulação do par inicial de estágios (ε_1), e é fornecido por $d_2 = d_2 \times (1 + z + \tau \times \varepsilon_1)$. Para o próximo par de estágios, estão em questão os estágios 2 e 3. Como o primeiro deles não se trata mais do princípio de toda a sequência de estágios considerados no processo, há uma nova sistemática para a definição da primeira dotação da dupla. Para tanto, precisa-se sortear duas linhas da matriz de estados da natureza (os termos de flutuação). A sequência de programa é análoga à do caso para os estágios 1 e 2. Note que os dados para o estágio dois estão agora sendo reestimados. Em seguida, ocorre a estimativa das dotações nos dois períodos considerados. O valor efetivo de $d_2{}'$ depende de um sorteio particular de um termo de flutuação que está sendo feito na simulação do par de estágios em questão (2 e 3). Portanto o valor de $d_2{}'$, neste último exercício, quase nunca coincide com a estimativa do mesmo feita para o par anterior (d_2).

Em seguida, é feita a estimativa do retorno dos ativos no estágio 2, o retorno da carteira sendo estimado para cada agente. Aparece, na sequência, a equação para a restrição orçamentaria em dois períodos em questão, que se afigura inteiramente análoga à que foi empregada na análise para os estágios 1 e 2. Segue a sequência de comando *if* para a identificação das coordenadas dos candidatos, que é idêntica à empregada no programa para os estágios 1 e 2. Vem, a seguir, o teste do atendimento das condições ditas como de equilíbrio geral a cada volta do *loop*. A escolha do candidato maximizador também é análoga ao do par de estágios anterior. O programa apresenta a saída para a alocação dos agentes relativa ao candidato vencedor (maximizador) no segundo e terceiro estágios (lembrando que, para o segundo deles, seu valor já tinha sido estimado no par de estágios 1 e 2, e o novo valor quase nunca coincide com o da estimativa anterior). O programa, para esta etapa, termina com a estimativa da dotação dos agentes. Cada consumo individual estimado corresponde a uma condição de maximização de utilidade, senão para todos, para um número especificado de agentes, dos dez considerados. Um tipo de gráfico mostra essas dez trajetórias, uma para cada agente. Nota-se que as trajetórias se caracterizam por uma descontinuidade nos pontos intermediários, pois temos dois consumos estimados para esses pontos, um para cada par vizinho de estágios considerados na simulação.

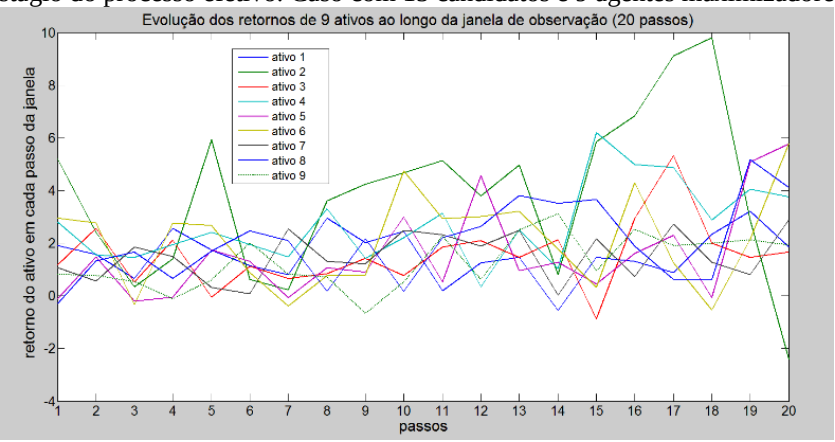
4. Estimativas por simulação.

O Matlab gera uma matriz com os tempos de processamento em suas entradas. A matriz mostra os tempos de cada experimento em uma linha (com as colunas relativas aos passos em questão). Numa figura, representam-se as diferentes trajetórias de consumo estocástico (individual e representativo), uma para cada experimento, obtidas com base na geração de parâmetros aleatórios, na maximização de utilidade (num problema de escolha temporal entre dois períodos subsequentes) e no atendimento de condições de equilíbrio geral dentro de uma margem de erro permitida. Há de se levar em conta que as diferentes trajetórias de consumo do experimento em questão foram obtidas a partir dos mesmos números gerados para α, τ, z etc., bem como da mesma matriz de estados da natureza dos termos aleatórios do ruído branco. As diferenças de trajetória de um experimento a outro devem-se aos diferentes vetores aleatórios; em cada passo, diferentes sequências dos vetores de retorno dos ativos etc. Curvas com as trajetórias do consumo individual são geradas ao cabo de cada experimento, inicialmente para a função de utilidade Cobb-Douglas.

O usuário tem acesso, ao cabo de cada experimento, a dados sobre: o vetor de consumo dos agentes no candidato vencedor do experimento; as dotações de todos os agentes em cada passo do experimento; o vetor de preços dos ativos em cada passo do experimento em questão; a matriz das quantidades compradas e vendidas de cada um dos ativos em cada passo do experimento; a carteira eficiente em cada passo, os retornos dos ativos em cada passo e naturalmente os tempos em cada passo do experimento.

A figura 1 mostra a trajetória dos retornos de nove ativos no experimento de número 1 para a função de utilidade Cobb-Douglas. A janela de observação mostrada é a que se forma no último estágio do processo efetivo, com tamanho final de vinte passos. Trajetórias semelhantes foram obtidas em todos os passos e para todos os demais sete tipos de função utilidade considerados.

Figura 1 – Trajetória dos retornos de nove ativos com processo GARCH. Experimento de número 1 para função de utilidade Cobb-Douglas. Último estágio do processo efetivo. Caso com 15 candidatos e 9 agentes maximizadores.



Outro gráfico mostra a trajetória do consumo *representativo* dos dez agentes em cada um dos dez experimentos para o caso da função de utilidade Cobb-Douglas. No eixo horizontal estão os passos e no eixo vertical estão os valores de consumo representativo (moda da distribuição de frequência ajustada ao

histograma de dispersão dos consumos maximizadores). Como nos passos intermediários há dois valores estimados do consumo representativo, opta-se pela média desses dois consumos. As soluções, em cada etapa do processo de simulação, dependem fundamentalmente dos sorteios iniciais dos parâmetros básicos. Dependendo dos valores gerados aleatoriamente e assumidos por esses parâmetros, o modelo de simulação funciona bem, com a possibilidade, inclusive, de se maximizar a utilidade para os dez agentes considerados, num intervalo temporal, a cada passo, inferior ao máximo de 240 segundo estipulados por passo. Roda-se o programa muitas vezes e testa-se, em cada qual, a adequação dos parâmetros sorteados. Assim obtém-se os consumos maximizadores individuais, e considerando-se uma média para os estágios intermediários, na medida em que há, meses caso, dois consumos estimados envolvidos, chega-se à estimativa da trajetória de consumo individual maximizador de utilidade para o modelo com 10 agentes e função de utilidade do tipo Cobb-Douglas. A trajetória de consumo maximizador de utilidade dependeu, de fato, do conjunto de sorteios aleatórios que compõe a sequência de sorteios. Esses sorteios definiram o estado da natureza e a seleção de uma linha específica da matriz dos estados da natureza em cada passo considerado na simulação. Os sorteios são feitos para o mesmo *experimento*, ou seja, considerando-se os mesmos parâmetros informados para a simulação, e os mesmos valores inicialmente sorteados. A curva de consumo individual maximizador depende, portanto, de uma sequência de ocorrências aleatórias; se fosse outra, mesmo para o mesmo conjunto de parâmetros iniciais que caracterizam o experimento, a curva de consumo gerada seria certamente outra. Há, portanto, para o mesmo experimento, curvas distintas associadas a diferentes sorteios. Faz-se uma sequência completa de sorteios em cada caso.

Uma boa covariância entre os retornos dos ativos surge quando se imagina um processo do tipo GARCH para a formação dos retornos de cada ativo. Cabe uma explicação do processo de construção da janela de observação. Iremos gerar, portanto, inicialmente uma janela de 10 estágios, para os 10 retornos iniciais do ativo em questão. Para o ativo 1, temos $B_{11}, B_{12}, \dots, B_{110}$. Os retornos dos ativos são gerados por um processo GARCH. Assim, define-se o vetor de retorno do ativo 1 ao longo da janela como $B_1 = (B_{11}, B_{12}, \dots, B_{110})$. Para os demais ativos, a situação é totalmente análoga. Temos, portanto, vetor de dez entradas, mas devemos completar a janela inicial. Isso é feito quando se incorpora a análise do processo estocástico efetivo, isto é, considerando-se o primeiro estágio do par de estágios em questão (começa-se com os estágios 1 e 2). Cada análise com um sucessivo par de estágios chamaremos de "passo". Uma série de sorteios definem dois **V**'s para o primeiro passo (estágios 1-2). Constrói-se então o vetor para a janela do tamanho de 12 estágios $A_1 = (B_{11}, B_{12}, \dots, B_{110}, A_{11}, A_{12})$. E assim por diante para os demais ativos.

Além de os ativos fictícios apresentarem boa covariância, é possível obter uma matriz 9×9 onde nenhuma linha seja igual a outra ou possa ser obtida como combinação linear de outras. De fato, a matriz **V**, assim obtida, tem o *posto* igual à dimensão, e, portanto, é não singular, com determinante não nulo. A escolha de se trabalhar com apenas 9 ativos evita a obtenção de matrizes **V** singulares. É sempre possível

averiguar se o programa roda bem para 9 ativos. Isso pode ser feito interagindo-se com o usuário, mostrando-lhe a matriz produto da **V** com sua inversa e pedindo que o usuário se manifeste se o resultado é o esperado, ou seja, se se trata da matriz identidade. Caso o resultado não o seja, o usuário pode retomar a *proc* desde o início, para uma nova tentativa. Caso a simulação *não* encontre uma matriz idêntica à matriz identidade 9×9, o usuário pede que o programa reinicie uma nova estimativa. Para a fórmula da carteira eficiente, entra também os valores de $\boldsymbol{r}$, o retorno esperado. Ora, como se está, pelo método proposto, alterando as últimas entradas na janela temporal dos retornos (até aqui apenas acrescentam-se as últimas entradas para completar a janela inicial de 12 estágios), também se está afetando o retorno esperado de cada ativo. Esse retorno esperado obtém-se pela média dos valores calculados dentro da janela (incluindo-se o último vetor de retorno dos ativos A_{ji} para o estágio *i* em questão). Assim, também o vetor $\boldsymbol{r}$ está sujeito a sorteios repetitivos no estágio em questão. Ele é simplesmente a média dos vetores A_{ji} para certo ativo *j* e *i* no intervalo $(i - n, \dots, i)$, para cada uma das *n* entradas do vetor $\boldsymbol{r}$.

Na fórmula da carteira eficiente, precisa-se, por fim, estimar um valor para o μ. O valor de μ é específico a cada um dos 10 agentes considerados. Como se viu, atribuir-se um valor para essa variável, associado a cada agente, afigura uma forma de discriminá-los, associando-se, portanto, a cada qual, em dado estágio, um vetor $\boldsymbol{x}^*$, uma carteira *eficiente* específica. Nota-se que, a cada estágio, tais carteiras foram obtidas com a estrutura comum de **V** e $\boldsymbol{r}$, mas os diferentes valores atribuídos de μ garantem certa individualidade à carteira eficiente. Seja a carteira de ativos A_i, para o estágio *i*. Não é factível que o sistema encontre uma solução para um μ_i maior que o valor máximo dentre todos os A_{ji} , *j = 1,..., 9*. Sendo assim, para evitar desperdício de tempo em problemas já sabidos insolúveis, o sistema gera os valores do retorno esperado almejado pelos agentes de modo factível. Seja o máximo de A_{ji}, dado o estágio *i*. Os agentes poderiam ser tipificados da seguinte maneira: um deles almeja a *metade* do retorno do máximo, os demais se distribuem em torno desse - quatro abaixo e cinco acima, ou seja, quatro deles almejam retornos menores na carteira e cinco miram retornos acima desse (mas menores que o do *máximo*). Para certo A_{ji} máximo (A_{ji}^m), o vetor de retornos esperados ($\boldsymbol{\mu}$) ficaria assim, por exemplo: $\boldsymbol{\mu} = \left(\frac{A_{ji}^m}{1,1}, \frac{A_{ji}^m}{1,2}, \frac{A_{ji}^m}{1,4}, \frac{A_{ji}^m}{1,6}, \frac{A_{ji}^m}{1,8}, \frac{A_{ji}^m}{2}, \frac{A_{ji}^m}{3}, \frac{A_{ji}^m}{4}, \frac{A_{ji}^m}{5}, \frac{A_{ji}^m}{6}\right)$. Então o programa poderia, em cada estágio *i*, encontrar o A_{ji} máximo e propor esse vetor $\boldsymbol{\mu}$ específico dos retornos de carteiras almejados pelos dez agentes considerados na simulação. O programa busca encontrar a carteira eficiente em cada estágio. Assim, pode-se calcular as quantidades compradas e vendidas pelos agentes no passo inicial. Os agentes não possuem um estoque de ativos inicial, antes do estágio 1 efetivo, sendo assim entendemos por quantidades vendidas nesse estágio as operações de venda a descoberto (*short sale*). O segmento de programação lê a matriz de carteira eficiente como sendo a compra (entradas com valores positivos) ou a venda (*short sale*) de ativos no primeiro estágio. Nota-se que algumas entradas são negativas, pois o modelo admite a possibilidade de *short sale*. Veja, ainda, que a soma dos pesos dos ativos

comprados por cada indivíduo é sempre 1, pois as carteiras estão todas normalizadas. A quantidade comprada são as entradas positivas do vetor de carteira eficiente. Já a quantidade vendida de ativos, no primeiro estágio, são as entradas negativas. No segundo estágio, compra-se o que falta, partindo-se da carteira eficiente anterior, para se chegar à nova carteira eficiente. Vende-se o que se tem a mais, de cada ativo, em relação à nova carteira que se pretende alcançar.

Define-se o retorno dos ativos e da carteira, no estágio. Com isso, constrói-se a restrição orçamentária. Segue-se o programa para a identificação dos candidatos (até 21 candidatos) e os testes da condição de equilíbrio geral. O programa prossegue com os testes para a maximização de utilidade. Em seguida, após a conclusão do *loop* de seleção de candidatos, o usuário pode averiguar, por inspeção visual, se as matrizes de covariâncias obtidas na rodada final, que atenderam às condições de equilíbrio impostas, de fato, são matrizes *não-singulares*. Neste caso, o produto dela pelo inverso deve resultar na matriz unitária. O sistema mostra ao usuário as matrizes encontradas nos dois estágios considerados no passo em questão e pergunta se ele está satisfeito, se as matrizes correspondem efetivamente, cada qual, à matriz identidade. Ou seja, caso não se tenha obtido um resultado satisfatório, reinicia-se a simulação para o passo. A mesma simulação é feita para distintas sequências de sorteios dentro do mesmo experimento.

Nota-se agora o problema na relação entre preço e retorno dos ativos financeiros, bem como o impacto desses dois problemas na geração das séries estocásticas. Para tanto, analisemos a relação entre a trajetória estocástica desses preços e a correspondente trajetória dos retornos dos ativos financeiros. Modelaremos, em nossa simulação, a série temporal dos retornos estocásticos dos ativos considerando que os retornos das séries temporais de ativos financeiros exibem um padrão inconstante de volatilidade. Para a descrição desses retornos propõe-se um modelo GARCH (p, q). Incorporando-se essas ideias, desenvolve-se novo fragmento de programa que estima os retornos dos ativos supondo um processo desse tipo. Os pesos envolvidos no cálculo de médias são extraídos de um conjunto de entradas do vetor alfas e betas, ambos sendo [0,005; 0,025; 0,05; 0,075; 0,095; 0,1; 0,125; 0,15; 0,175; 0,2; 0,225; 0,25; 0,275; 0,295; 0,3; 0,325; 0,35; 0,375; 0,4; 0,425; 0,45; 0,475; 0,5; 0,525; 0,55, 0,575, 0,6]. Nos estágios 11º e 12º (primeiro e segundo estágios efetivos, fora da janela virtual) atribuímos ao estágio corrente o peso de 50%, tanto em α quanto em β, e 50% para a soma dos pesos de todos os estágios anteriores.

Um fragmento de programa gera os retornos nos dois primeiros estágios do processo efetivo, fora da janela virtual. O programa continua com a sequência de programação para a identificação das coordenadas dos candidatos. Em seguida, o programa testa as condições de equilíbrio dinâmico. Começa-se com função de utilidade do tipo Cobb-Douglas e depois aplica-se o modelo a sete outros tipos dessa função. Finalmente a *proc* para os estágios efetivos 1 e 2 gera os relatórios de saída com estimativas e gráficos. Apenas lembrando que a janela dos retornos dos ativos é de tamanho variável, cada vez maior a cada passo subsequente. Outro fragmento da *proc* mostra gráficos sobre a evolução dos retornos dos nove ativos ao

longo da janela de tamanho variável. Então a *proc* para os estágios 1 e 2 com processo GARCH fora testada rodando-se o programa. Os resultados foram bastantes positivos. Com 15 pontos da restrição orçamentaria sendo testados (15 candidatos), em um tempo inferior a 2 minutos o programa alcança, na grande maioria dos casos, a maximização da utilidade para 9 dos 10 agentes considerados.

Um experimento foi feito com esse modelo de formação dos retornos dos ativos. Para mil linhas na matriz de estados da natureza, cem tentativas de alcançar a margem de 2 unidades na condição de equilíbrio geral, e com 15 pontos do espaço orçamentário sendo testados, o programa estima valores para τ, z etc. e resolve o problema de maximização para, no mínimo, nove agentes maximizando simultaneamente no ponto vencedor (com poucas exceções nas quais esse mínimo cai para oito ou sete). Na maioria das tentativas, o sistema alcança valores abaixo da margem máxima permitida para a solução do problema de equilíbrio geral, pelo menos até o quinto estágio efetivo, podendo violar essa condição nos estágios finais. Como valores de saída, ele apresenta a renda líquida nas transações financeiras em dois estágios, bem como os preços dos ativos estimados com base nos retornos. Uma matriz de retornos é gerada. Observa-se que alguns retornos são negativos. A matriz mostra a evolução do retorno de cada ativo ao longo de 12 passos, no caso em que é usada para estimativas no primeiro par de estágios efetivos.

A *proc* para os estágios 1 e 2 do processo estocástico com os retornos seguindo um padrão GARCH funcionou muito bem. Em seguida, desenvolveram-se *procs* semelhantes a essa a fim de cobrir-se, na simulação, o processo estocástico para os passos seguintes. Começando-se pelo programa Matlab referente ao processo GARCH, para os estágios 2 e 3. Nessa *proc*, tem-se a sequência de programação que identifica as coordenadas dos candidatos a cada tentativa dentro do *loop for*. Tal sequência envolve uma série de *if*'s. Quando o sistema encontra um candidato que atenda às condições de equilíbrio geral, isto é, quando se chega a uma *margem* dentro do limite de flutuação máxima admitido, o comando *break* permite a saída do *loop for*. Mas quando, após o número de tentativas (estipulada em cem pelo usuário do programa), o sistema não alcança a margem máxima estipulada pelo usuário, o *loop* continua. Após identificar a tentativa que resultou na margem mínima, o programa pede que se encontre aquela em que se obteve essa margem. Aos consumos em cada estágio são atribuídos os valores estimados na tentativa mais bem-sucedida. Em seguida, segue a escolha do ponto candidato maximizador de utilidade. Está-se descartando as tentativas que não resultem em consumo *representativo* não negativo (alguns consumos individuais podem ser negativos, apenas o centro da distribuição, ajustada por uma função normal, está num ponto de coordenada positiva). Tal filtro seletivo torna a simulação mais adequada e eficiente. A *proc* em tela prossegue com a apresentação das estimativas e gráficos de saída. *Procs* inteiramente semelhantes, com as devidas adaptações, fazendo-se avançar e expandir a janela de retorno dos ativos, foram construídos para os passos 2 (estágios 2-3), 3 (estágios 3-4), 4 (estágios 4-5), 5 (estágios 5-6), 6 (estágios 6-7), 7 (estágios 7-8), 8 (estágios 8-9) e 9 (estágios 9-10). A última *proc* considerada mostra, ao final, a distribuição do consumo,

ajustada por uma função gaussiana, para todos os pares de estágios em questão. A *proc* em questão termina gerando o gráfico com a trajetória dos preços dos nove ativos considerados, preços esses que são gerados com base nos retornos estimados por um processo GARCH. Finalmente a mesma *proc* remete à outra, que gera gráficos adicionais, bem como o relatório com o número de agentes maximizadores informado pelo usuário a cada estágio considerado (idealmente ao menos nove dos dez agentes em questão). Ao final da última *proc*, encontra-se a sequência de programação na qual o usuário tem a opção de fazer novo experimento. Fazem-se novas estimativas da matriz dos estados da natureza, dos termos deterministas e da amplitude do termo aleatório do crescimento da dotação, do termo determinista desse crescimento etc.

5. O teste de convergência ao equilíbrio geral dinâmico e estocástico.

Inicia-se a simulação informando os parâmetros de mil linhas na matriz de estados da natureza, cem tentativas a cada *loop for*, 15 candidatos, margem de tolerância de flutuação de 2 e taxa de desconto temporal de 0,2. O sistema estima então z, $\sigma's$ iniciais e os retornos dos ativos no 11º e 12º estágios da janela. Informa-se um número de 9 agentes efetivamente maximizadores, de dez possíveis.[11] Nesse caso, o sistema atende com eficiência às condições de equilíbrio geral na maioria dos candidatos.

O programa gera os gráficos das trajetórias dos retornos de nove ativos na janela estendida até 20 passos que se obtém ao final do último estágio efetivo (janela virtual de dez passos mais dez passos adicionais da simulação na rodada até o último estágio efetivo) (Figura 1). Nota-se o padrão em que os retornos oscilam, segundo um processo GARCH. Na figura em questão, estão desenhadas as trajetórias finais dos retornos dos nove ativos considerados na simulação quando a janela de observação alcança sua extensão máxima. No modelo robusto, faz-se com que os preços dos ativos reflitam o retorno estimado deles. Ou seja, os retornos são estimados por um processo GARCH e os preços calculado como variável dependente dos retornos pela fórmula simples $p_n = \ retorno \times p_{n-1} + \ p_{n-1}$. Isso faz com que o preço do ativo cresça sempre que o seu retorno também cresça.

O sistema gera as distribuições de consumo individual entre os dez agentes, nesses dez estágios considerados, para cada experimento na simulação com dada função de utilidade. Nota-se que alguns agentes, nos últimos estágios, ficam com um consumo estimado negativo. Como se viu, o programa proíbe que as estimativas cheguem a um consumo médio negativo, mas é possível que um ou poucos indivíduos apresentem consumo negativo em algum estágio avançado; muito embora a ideia de consumo negativo não tenha um significado econômico claro em nosso modelo.[12] Nota-se ainda que os consumos individuais

[11] Poderia ser dez agentes. Trabalha-se com nove apenas por economia de tempo de processamento. No tempo de somente dois minutos por *loop*, o sistema, em geral, consegue obter até nove maximizadores.

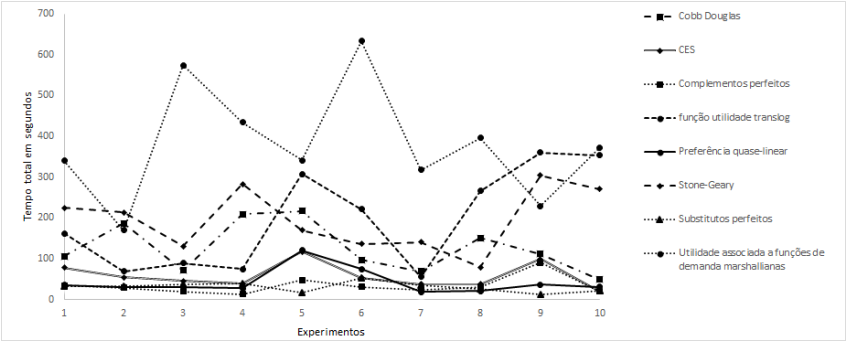
[12] Podemos facilmente introduzir, no programa, uma restrição de não negatividade sobre todo e qualquer consumo individual em todos os estágios considerados na simulação. Isso, no entanto, demandaria bem mais tempo de processamento e a melhoria na qualidade da simulação não seria significativa, pois interessa aqui o consumo representativo e não os consumos individuais em si mesmos.

crescem sobremaneira nos dois últimos estágios. Após os exercícios para o programa Matlab com função de utilidade Cobb-Douglas, efetuaram-se diversas estimativas para os modelos com as sete outras funções de utilidade aqui consideradas. Os programas, na verdade, são inteiramente análogos ao caso para a função de utilidade Cobb-Douglas, com as mesmas ideias, mesmo método de atribuição de valores a variáveis aleatórias, as mesmas saídas e gráficos. Para as simulações com 15 pontos candidatos, os relatórios de saída do programa apresentam os valores gerados para os parâmetros aleatórios α, ρ e τ, na CES; α e τ, na SP; α e τ, na CP; τ, na PQL; α e τ, na SP; λ_1, λ_2, β_1, β_2, e τ, na SG e λ_1, λ_2, α_1e τ, na UDM e a_0, a_1, a_2, a_3, e τ, na translog. Lembrando-se de que, para cada função utilidade em questão, esses valores serão gerados por passo, ou por giro do *loop* em questão no caso dos valores de λ_1 e λ_2, que são novamente estimados a cada movimento desse tipo.

As funções de distribuição do consumo para os dez passos considerados, em experimentos para os modelos com as oito funções de utilidades em tela, são bastante distintas entre si. Essas diferenças se devem às diferentes matrizes de estados da natureza das variáveis aleatórias para o ruído branco, às diferenças nas trajetórias entre esses estados da natureza ao longo dos 10 passos, às diferenças entre as variáveis aleatórias geradas para a classe de experimentos (α, ρ, τ, λ_1, λ_2, β_1, β_2, α_1, a_0, a_1, a_2, a_3), e naturalmente ao diferente tipo de função de utilidade empregada. Depende também da aleatoriedade inerente à cada experimento, onde os valores são obtidos por sorteio. Outro conjunto de gráficos que vale a pena considerar, nessa comparação de diferentes funções de utilidade para 15 pontos candidatos, é a trajetória de consumo representativo, ao longo dos passos, para os dez experimentos, vendo os resultados em todos os tipos de função de utilidade considerados. Em cada classe de função de utilidade, nos respectivos 10 experimentos, há um padrão próprio de evolução do feixe de consumos representativos gerados pelo programa. O resultado depende, em parte, do tipo de função utilidade utilizada e seu efeito sobre o processo que busca atender à condição de equilíbrio geral. Procurou-se rodar todos os experimentos com um número de, no mínimo, 9 agentes maximizadores de utilidade (dos 10 considerados no modelo). Apenas em alguns passos específicos não se alcançou essa meta no tempo imposto de 240 segundos de processamento por passo.

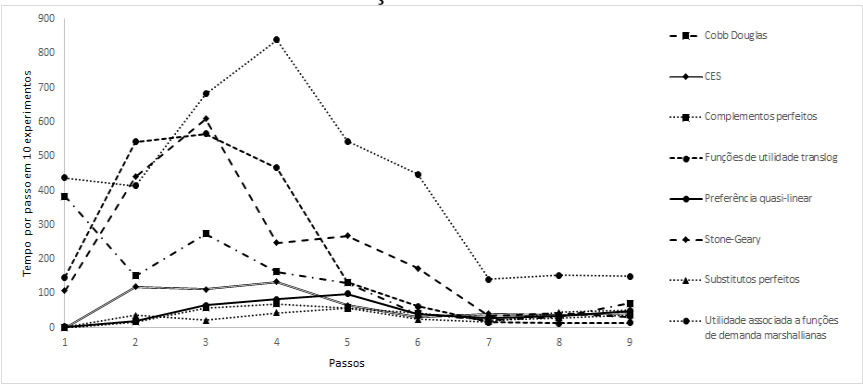
É da maior importância para o teste de convergência dos modelos de simulação o tempo de processamento dispendido pela máquina até que se alcance a condição de flutuação máxima estipulada pelo programa. A Figura 2 apresenta os tempos gastos em todos os experimentos para todos os passos considerados. Nela, consta o tempo total consumido por experimento. Trata-se do caso com 15 pontos candidatos e 9 agentes maximizadores de utilidade. Então a figura mostra o tempo *total* de processamento em cada um dos modelos de simulação com as oito funções de utilidade consideradas. Nota-se que, em uns poucos passos, o programa ultrapassou o tempo máximo de processamento de 240 segundos por par de estágios, o que levou a se trabalhar com um número menor de maximizadores (sete ou oito).

Figura 2 – Tempos de processamento para os dez passos nos dez experimentos considerados. Função de utilidade do tipo Cobb-Douglas, CES, complementares perfeitos, translog, preferência quase-linear, Stone Geary, substitutos perfeitos e utilidade associada a funções de demanda marshallianas.



A análise do tempo total por passo, considerando-se todos os experimentos para o passo em questão, aparece na Figura 3. A análise de tempo pode ser empregada com o fito de se testar o problema teórico em modelos de equilíbrio geral dinâmico e estocástico, em sua expressão algébrica. Nesse sentido, o tempo de convergência obtido nas simulações pode ser interpretado como a probabilidade de o modelo convergir, dependendo de suas características estruturais, com o tipo de função utilidade utilizado. Sendo assim, considerando-se as informações das figuras 2 e 3, pode-se lançar uma hipótese quanto à convergência dos modelos ao equilíbrio dinâmico com base nos tempos de processamento observado.

Figura 3 – Tempos de processamento para cada passo, dentre dez, na soma de todos os dez experimentos realizados. Função de utilidade do tipo Cobb-Douglas, CES, complementares perfeitos, translog, preferência quase-linear, Stone Geary, substitutos perfeitos e utilidade associada a funções de demanda marshallianas.



A tabela 1 mostra a média dos tempos totais consumidos, em dez passos e dez experimentos, para as distintas funções de utilidade empregadas. Considera-se o menor desses tempos como sendo o *tempo de convergência certa do modelo*, o tempo em que o modelo certamente convergiria (mesmo com outro conjunto de simulações). Estabelecido este tempo como 100% de probabilidade de convergir, supõe-se que, para tempos menores, se mantenham os mesmos 100% e para tempos maiores ocorra um decaimento da probabilidade por uma função exponencial de expoente negativo. Seria, por exemplo, a função exponencial $prob = e^{(-0{,}01(t-30{,}05572))}$ a que determina as probabilidades de convergência nessa classe de experimento.

Tabela 1 - Tempo médio de convergência nas simulações e probabilidade de convergir. Funções de utilidade de oito tipos: Cobb-Douglas, CES, complementares perfeitos, translog, preferência quase-linear, Stone Geary, substitutos perfeitos e utilidade associada a funções de demanda marshallianas. 15 candidatos e de 7 a 9 maximizadores.

Função de Utilidade	*Tempo médio de convergência*	*Probabilidade de convergir (%)*
Cobb-Douglas	126,63	38
CES	58,22	75
Complementares perfeitos	33,87	96
Translog	196,04	19
Preferência quase-linear	42,52	88
Stone Geary	195,16	19
Substitutos perfeitos	30,06	**100**
Associada a funções de demanda marshallianas	380,95	3

Então a referida tabela fornece as probabilidades de convergência. Portanto, esse tipo de análise de convergência de modelos teóricos trata-se de um possível emprego desses exercícios de simulação numérica computacional.

6. Considerações Finais.

O avanço mais importante, e que demandou mais estudos e custo de implementação no programa Matlab, foi desenvolver o modelo de programação em que a formação dos retornos dos ativos se dá por um processo GARCH (p, q), aplicado a uma janela de observação de tamanho crescente, começando com dez períodos e evoluindo para vinte estágios do processo. A técnica aqui desenvolvida, de simulação de mercados financeiros, deve ser concebida e interpretada como uma ferramenta de análise numérica através de simulações computacionais. O modelo funciona como uma caixa de ferramenta para diversas aplicações. Por exemplo, em outro ensaio testamos o poder preditivo de nosso modelo de simulação em descrever adequadamente o padrão de volatilidade heterocedástico da função de consumo agregado das famílias dos EUA. O modelo de simulação procurou aperfeiçoar hipóteses utilizadas na programação computacional. Buscou-se não apenas maior clareza e nitidez nas hipóteses lançadas, mas trabalhar ostensivamente com a premissa de agentes que se comportam de modo racional. Assim é que a carteira de escolha individual de ativos tem de ser eficiente no sentido de Markowitz; a escolha de consumo entre dois períodos vizinhos é a que maximiza a utilidade para um maior número de agentes; os agentes são "conscientes" (observam o mercado) antes de determinar o retorno *almejado* da certeira, e outras hipóteses de racionalidade empregadas no modelo. O modelo, além de impor racionalidade, também buscou realismo, por exemplo ao modelar a trajetória dos retornos dos ativos, na simulação, por um processo GARCH que se aproxima bem do comportamento real dos ativos financeiros.

O tipo de aplicação do modelo desenvolvido diz respeito ao uso dele como ferramenta auxiliar aos trabalhos em teoria pura de economistas matemáticos que buscam, em seus sofisticados modelos, demostrar as condições para a existência de um equilíbrio dinâmico com variáveis estocásticas. De fato, examina-se

a possibilidade de processos de convergências ao equilíbrio dinâmico com base ao tempo de processamento da máquina para se atingir uma condição de equilíbrio geral dentro de uma margem de flutuação aceitável. Há situações em que o modelo matemático se torna tão complexo que não se consegue uma solução analítica explícita para ele. Nesse ponto, as simulações computacionais podem avaliar a possibilidade de equilíbrio no modelo em questão sem a tal solução analítica. Nesse sentido, a simulação serve para investigar a possibilidade de existência de equilíbrios em determinados *espaços de estados*, ou então examina sob quais condições as trajetórias de equilíbrio obtidas nesses espaços podem se tornar estacionárias. Em algumas outras situações, o economista matemático já conhece a solução teórica, o chamado *equilíbrio exato*, e então esse tipo de exercícios de aproximação numérica em simulação funciona para avaliar a possibilidade de se obter, nas simulações, resultados que se aproximam dele.

Referências Bibliográficas.

Burnside, C. (1988) Solving asset pricing models with Gaussian shocks. *Journal of Economic Dynamics and Control*, Vol. 22, 329-340.

Collard, F.; Juillard, M. (2001) Accuracy of stochastic pertubation methods: the case of asset pricing models. *Journal of Economics Dynamics and Control*, Vol 25, 979-999.

Cox, J. C., Ross, S.A. (1976) The Valuation of Options for Alternative Stochastic Process, *Journal of Financial Economics*. Vol. 3, 145-166.

Dubey, D.; Geanakoplos, J. ; Shubik, M. (2005) Default and punishment in general equilibrium. *Econometrica*, Vol. 73, No. 1.

Dupacová, Jitka; Hurt, J.; Stepan, J. (2001) Stochastic Modeling in Economics and Finance. Springer Verlag, 2001.

Engle, R. (1982) Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of U.K. inflation, *Econometrica*, Vol. 50, 987-1008.

Engle, R.; Lilien, D.; Robens, R. (1987) Estimating Time Varying Risk Premia in the Term Structure: The ARCH-M Model, *Econometrica*, Vol. 55, 391-407.

Engle, R.; Ng, V. (1993) Measuring and Testing the Impact of News on Volatility. *Journal of Finance*, Vol. 48, 1749-1778.

Härdle, Wolfgang; Hafner, Christian M. (2000) Discrete time option pricing with flexible volatility estimation.

Härdle, Wolfgang; Tsybakov, A. (1997) Local Polynomial Estimation of the Volatility Function, *Journal of Econometrics.*

Härdle, W.; Tsybakov, A.; Yang, L. (1996) Nonparametric Vector Autoregression, *Journal of Statistical Planning and Inference.*

Harrison, J. M.; Kreps, D. (1979) Martingales and Arbitrage in Multiperiod Securities Markets, *Journal of Economic Theory*, Vol. 20, 381-408.

Schmitt-Grohé, S.; Uribe, M. (2004) Solving dynamic deneral equilibrium models using a second-order approximation to the policy function, *Journal of Economics Dynamics & Control*, Vol. 28, 755-775.

Whittle, Peter (1951), *Hypothesis Testing in Time Series Analysis*

Zame, W. R. (1990) Efficiency and the Role of Default when Security Markets are Incomplete. UCLA Working Paper, No. 585, University of California.