UNIVERSIDADE FEDERAL DE PERNAMBUCO
DEPARTAMENTO DE FÍSICA – CCEN
PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA

DISSERTAÇÃO DE MESTRADO

Simulações Numéricas em Armadilha Magneto-Ótica Através de Algoritmo Hierárquico

por

Rubens Soares de Oliveira

Dissertação apresentada ao Programa de Pós-Graduação em Física do Departamento de Física da Universidade Federal de Pernambuco como parte dos requisitos para obtenção do título de Mestre em Física.

Banca Examinadora:

Profa. Sandra Sampaio Vianna (Orientador-UFPE)
Prof. Ernesto Carneiro Pessoa Raposo (UFPE)
Prof. José Wellington Rocha Tabosa (UFPE)
Profa. Solange Bessa Cavalcanti (UFAL

Recife - PE, Brasil
Agosto – 2003

À minha mãe, Maria Lourenço, uma fonte de inspiração, ao meu pai, Antônio Soares (in memoriam), um bravo, e aos meus irmãos e irmãs, dedico essa dissertação, porque muito propriamente diz um antigo ditado popular: "Quem sai aos seus, não degenera".

Agradecimentos

Primeiramente quero agradecer à Professora Sandra pela orientação e por todo o aprendizado construído nesses últimos dois anos.

Ao Professor Ernesto Raposo pela co-orientação, pela prontidão e pelo incentivo à minha vinda para a cidade do Recife.

A Daniel Barbosa pelas discussões teóricas em Física e em programação.

A Josione (Jô) e Sandro pela amizade e pela paz.

A Janete e às suas idéias praticamente rodriguianas (relativo à Nelson Rodrigues) no que concerne à alma feminina. Coisas como: “O homem mais desejado para uma mulher é o namorado da outra”.

A Robson e seus maus exemplos que me levaram para o caminho das drogas como o cigarro e as bebidas alcoólicas, exceto a cachaça que embora alcoólica, é docinha.

A Eveline por estar por perto para me lembrar como é chato ser “aborrecente” (adolescente).

A Alexandre Rosas por receitar filmes de terror sanguinários para curar estresse elevado. Não compreendo o fundo científico, mas funciona!

A Zé Maria pelas conversas mais insólitas, desde política no Recife da década de quarenta até poemas materialistas egípcios antigos – viagens na maionese.

A Ana Carolina por ser uma “Fátima”.

A Ana Maria mesmo que ela tenha tentado pegar no meu pixito e somente depois tenha descoberto que eu não tenho um desses.

A Nina (Maria Selma) e sua segurança em opinar a respeito de qualquer assunto tendo sempre uma postura bem definida e inabalável, além das suas várias estórias.

A Marluce e seus bolos. A Batista e sua maneira profética de aconselhar. A Fátima, Fernando, Farla e Fenanda.

A “mundiça” do dfte.ufrn: Chico, o iluminado; João Maria, o póby; Neemias, o bebê; Pedro, o corajoso; Rose, a sonhadora entre outros.

A “catrevage” do df.ufpe: Carlos e Sandra (ES), Dona Célia de Aracajú e o patrão, Delza e Raquel, pela paciência.

Aos companheiros de caminhadas: Kaênia (a sereia) e Flávio, mesmo que ele tenha feito uma operação para corrigir o septo, ou sexo. Bem, ele nunca falou a verdade! Pelo menos nunca foi convincente.

Por último e não menos importante: ao grupo: “Os Cabra”, pela sua inenarrável criação: a baratinha, e seu indiscutível caráter psico-calmante-terapêutico.

Resumo

Neste trabalho, estudamos os modos orbitais que surgem na armadilha magneto-ótica quando os feixes de laser, responsáveis pelo aprisionamento dos átomos, apresentam um desalinhamento no plano *xy* de forma a induzir um movimento orbital em torno do eixo *z*. Para explicar as diferentes estruturas espaciais observadas experimentalmente, consideramos, além da interação com os lasers, também a interação entre pares de átomos devido ao múltiplo espalhamento de fótons. Concentramos nossos estudos nos efeitos da interação sobre a estrutura na forma de um anel e seu desenvolvimento para dois anéis concêntricos. Implementamos um algoritmo numérico com estrutura hierárquica, o qual permite controlar a aproximação feita no cálculo numérico da interação. Com este algoritmo, pudemos simular a dinâmica de até 10^6 átomos na armadilha. Esse é o limite para simulações usando programação linear. Os efeitos da interação com o aumento do número de átomos na armadilha são observados no alargamento da estrutura em forma de anel, bem como no aumento do raio de equilíbrio desta estrutura. Para 10^6 átomos esta estrutura espacial apresenta um pequeno pico lateral indicando um possível início da estrutura de dois anéis concêntricos.

Abstract

In this work, we study the orbital modes that appear in the Magneto-Optical Traps from the misalignment of trap's beams in the *xy*-plane. To elucidate the different spatial distributions observed experimentally, we consider the action of the lasers over the individual atoms, as well as the interaction between these atoms that came from the multiple scattering of photons. We concentrate our studies on the interaction effects over the single ring cloud and the development to the double concentric ring. A numerical algorithm with hierarchical structure was elaborated to carry out the numerical calculation of the interaction with a control of the approximation level. This procedure can simulate the temporal dynamics up to 10^6 atoms in the trap. This is the limit to perform simulations using a linear computer program. The interaction effects are observed in the single ring enlargement and on the increase of the average radius of this structure. For 10^6 atoms in the trap, the single ring shows a small side peak which might indicate the onset of the double concentric ring structure.

Índice

Capítulo I

Introdução

A idéia central para a construção de armadilhas para átomos está no fato da luz transportar momentum. Com base nessa constatação, em 1970 A. Ashkin [1] consegue levitar pequenas partículas esféricas de látex suspensas em gases e líquidos usando um laser de argônio. Cinco anos mais tarde, com T. W. Hänsch e A. L. Schawlow, surgem as primeiras idéias de usar laser para esfriar e aprisionar átomos [2], mas somente em 1987 E. L. Raab e colaboradores [3] conseguem construir uma armadilha para átomos usando uma configuração de seis feixes de laser contrapropagantes dois a dois em conjunto com um gradiente de campo magnético. Nesse caso, a armadilha era carregada pelo resfriamento de um feixe de átomos. Mais tarde C. Monroe e colaboradores [4] mostram ser possível carregar a armadilha a partir de uma célula de vapor à temperatura ambiente, o que facilitou sua montagem. Melhoradas as técnicas de construção das armadilhas, partia-se agora para a obtenção de armadilhas cada vez mais densas, com o objetivo de chegar à condensação

de Bose-Einstein. Em estudos teóricos [5], chegou-se a prever a implosão de uma nuvem de átomos de sódio (Na) numa armadilha com seis feixes de laser contrapropagantes ainda sem gradiente de campo magnético, o que se mostrou um equívoco. No início da década de noventa, D. Sesko e colaboradores [22] mostram que, devido à interação entre os átomos na armadilha, existe uma densidade limite para os átomos aprisionados, o que significava uma barreira para se atingir um estado de condensação. Mais tarde, em 1995, M. H. Anderson e colaboradores [6], usando técnicas de evaporação, conseguem obter um condensado de Bose-Einstein.

Desde o início dos estudos sobre armadilhas magneto-óticas, diferentes estruturas têm sido observadas (figura 1.1). Numa armadilha onde os três pares de feixes contra-propagantes encontram-se alinhados, surge somente uma estrutura atômica na forma aproximada de uma bola no centro, comumente referida na literatura como modo estático. Estruturas diferentes de uma bola surgem numa armadilha desse tipo quando os feixes de luz laser são desalinhados num dos planos.

O desalinhamento dos feixes dá origem a uma estrutura na forma de um anel (fig. 1.1-b) [7] que pode ser observada a baixas ou altas densidades. Para altas densidades, outras estruturas espaciais também têm sido observadas, como por exemplo: um anel com uma bola no centro (fig. 1.1-c) [22] e dois anéis concêntricos (fig. 1.1-d) [4]. Essa última estrutura tem sido observada para densidades da ordem de 10^7

átomos/cm^3. Devido ao deslocamento dos feixes, os átomos fora da estrutura central apresentam-se girando em torno do centro da armadilha, daí o nome de modos orbitais a estas diferentes estruturas.

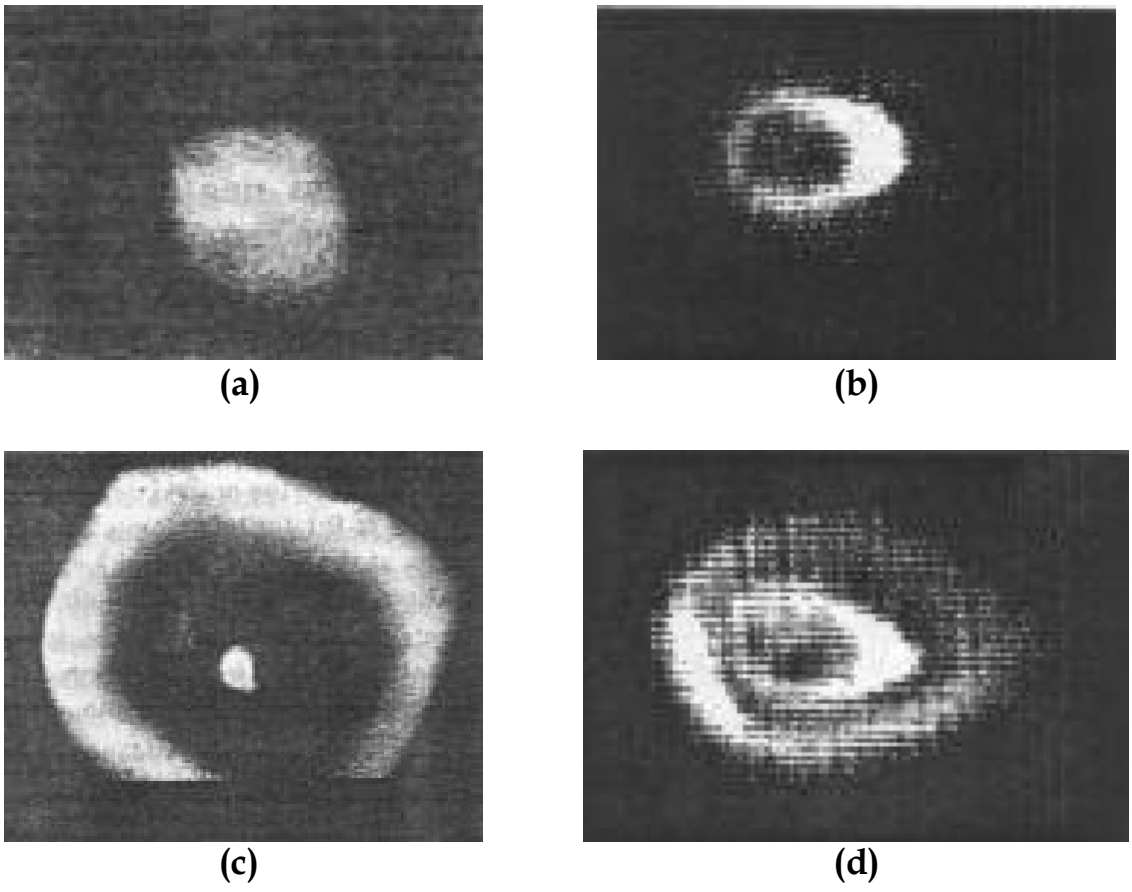


Figura 1.1 – Diferentes formas das nuvens atômicas observadas em armadilhas magneto-óticas. (a) com seis feixes contra-propagrantes, (b), (c) e (d) com desalinhamento dos feixes em um dos planos.

Um primeiro estudo, a respeito das formas das nuvens atômicas que surgem do desalinhamento dos feixes de luz laser, é apresentado no artigo de Sesko e colaboradores [22]. O sucesso deste estudo está na determinação do limite na densidade da nuvem de átomos aprisionados. Esta determinação foi feita experimentalmente pela observação da variação do tamanho da estrutura na forma de bola à medida que o número de átomos era aumentado na armadilha. Neste trabalho, Sesko e colaboradores apresentam uma teoria de interação entre os átomos baseada no múltiplo espalhamento de fótons. Essa teoria de interação mostrou muito boa concordância com resultados experimentais à cerca da densidade limite dos átomos aprisionados. Eles tentaram explicar a observação de diferentes formas de nuvens atômicas como conseqüência dessa interação sem apresentar qualquer teoria detalhada relativa às estruturas observadas.

Mais tarde foi mostrado que o modo orbital de um anel sozinho pode ocorrer mesmo em regime de baixas densidades atômicas. Na referência [8], a estrutura na forma de um anel sozinho foi explicada considerando apenas a ação dos feixes de laser desalinhados sobre cada átomo, sem o uso da interação entre eles. O desalinhamento dos lasers dá origem a uma força de vórtice [9] que surge como a componente radial das forças dos lasers, sendo responsável pelo movimento orbital e pela estrutura na forma de um anel. Algumas tentativas foram feitas [10] para explicar a

formação do modo constituído por dois anéis, a partir da força de vórtice, entretanto, os resultados obtidos estão fora do limite de validade das aproximações efetuadas.

Um estudo destas diferentes estruturas em armadilhas a baixas densidades, isto é, sem interação entre os átomos [11], mostrou que o desalinhamento dos feixes só consegue explicar o aparecimento do anel sozinho.

Motivados a entender melhor o comportamento dos átomos numa armadilha, nas condições que os levam a assumir uma estrutura de duplo anel, consideramos a interação entre os átomos e abordamos o problema usando simulação numérica computacional. Para isso adaptamos e implementamos um programa baseado num algoritmo hierárquico proposto por J. Barnes e P. Hut [38] em 1986 para simulações de sistemas auto-gravitantes.

Apresentamos no Capítulo II as forças que atuam sobre um átomo na região de uma armadilha magneto-ótica. Levamos em conta a velocidade de cada átomo para cada instante de tempo, bem como sua posição na armadilha. Essas considerações são importantes, pois a expressão da força que atua sobre os átomos sofre alterações devido ao efeito Doppler, referente à velocidade do átomo, e ao efeito Zeeman, referente à sua posição na armadilha.

Numa armadilha a altas densidades, devemos considerar a força de interação de cada átomo com os demais. Fazemos então uma dedução dessa força de interação com base no trabalho de Steane e colaboradores [23], considerando uma contribuição devido à luz duplamente espalhada e um efeito de sombreamento que ocorre devido à presença dos demais átomos na armadilha. Encontramos uma força de interação do tipo $1/r^2$, uma força de longo alcance da mesma forma da força gravitacional. Aqui nos deparamos com um problema bem conhecido na física, o problema de N corpos interagentes. A dificuldade de resolver, mesmo numericamente, as equações de movimento para N corpos interagentes, com N grande, tem por muito tempo sido objetivo de estudo de muitos pesquisadores. Sem um procedimento padrão exato e, ao mesmo tempo numericamente factível, do ponto de vista dos longos tempos computacionais exigidos para $N \sim 10^5$ partículas ou superior, partimos para o uso de um método aproximativo, para resolver essa situação. Por razões históricas evidentes, o problema de N corpos interagentes vem sendo atacado com mais afinco nos ramos da Cosmologia e Astrofísica. Lá, impossibilitados de experimentar com objetos celestes e com o cosmo como um todo, os físicos dessas áreas costumam utilizar modelos computacionais para suas experiências. Vários métodos aproximativos [39] têm sido propostos por eles.

A escolha do método aproximativo que utilizamos [38] teve como parâmetros: a viabilidade de implementação e a proposta no ganho do

tempo gasto para sua execução. A implementação foi feita em Linguagem C de programação para compilação e execução em sistema operacional Linux. O tempo de execução para o método direto, sem aproximação, cresce com o quadrado do número de átomos, N, enquanto o método aproximativo escolhido implica num crescimento com $N \log N$. O Capítulo III é dedicado à descrição detalhada do método aproximativo utilizado, com ênfase na aplicação ao nosso problema de armadilha magneto-ótica.

Também desenvolvemos um programa para a simulação via método direto ou exato, ou seja, sem aproximação. Isto nos permitiu fazer comparações que foram importantes para nos certificarmos da eficiência e confiabilidade do método aproximativo.

As simulações foram feitas usando parâmetros que correspondem ao átomo de sódio (Na) e próximos das condições experimentais onde a estrutura de dois anéis é observada [10]. As análises e os resultados obtidos são apresentados no Capítulo IV. Em seguida apresentamos nossas conclusões a cerca dos resultados obtidos e perspectivas.

Como apêndice, no final dessa dissertação, anexamos o código fonte base das simulações realizadas. Para compreensão do código fonte, é necessário ter noções de elementos de linguagem C de programação tais como: estruturas, ponteiros e alocação dinâmica de memória.

CAPÍTULO II

Forças Numa Armadilha Magneto-Ótica

Nesse capítulo vamos descrever o princípio de funcionamento de uma armadilha magneto-ótica com seis feixes de luz laser contra-propagantes dois a dois. Deduziremos as expressões para a força que atua sobre um átomo, as quais serão utilizadas para o desenvolvimento das simulações.

Na primeira seção fazemos uma dedução da força de radiação, ou força de pressão de radiação, exercida por um conjunto de seis feixes de luz laser sobre um átomo individual. Essa força, em conjunto com um gradiente de campo magnético, é usada para gerar uma armadilha magneto-ótica. Uma dedução detalhada das forças para esse tipo de armadilha pode ser encontrada nas referências [24] e [25].

Na seção seguinte discutimos como levar em conta a interação entre os átomos considerando efeitos de múltiplo espalhamento de fótons e efeitos de sombreamento da luz do laser devido à nuvem de átomos armadilhados (regime de altas densidades).

Na seção 2.3, obtemos as expressões para a força considerando o desalinhamento dos lasers. Na última seção deste capítulo usamos o fato de que, para um desalinhamento dos feixes de laser no plano *xy* de uma quantidade *s*, as trajetórias dos átomos aprisionados são aproximadamente circulares. Fazemos a aproximação de que o movimento é circular e com isto construímos um gráfico da forma dessa força de aprisionamento.

2.1 - Armadilha Magneto-Ótica com Seis Feixes

Começaremos por considerar o caso em que a luz laser está aproximadamente ressonante com uma determinada transição atômica, de modo que podemos tratar o átomo, em primeira aproximação, como um sistema de dois níveis. Vamos designar o nível inferior, ou fundamental por |1>, com momento angular total J = 0, e o nível superior, ou estado excitado, por |2>, com momento angular total J = 1. Isto será suficiente para explicarmos o processo que leva à criação dos termos de esfriamento e de restauração na expressão da força que atua sobre o átomo em questão.

O que calculamos, na verdade, é a força média que o átomo sofre quando se encontra interagindo com um feixe de laser, sendo esta devido aos processos de emissão e absorção de fótons. A força de armadilhamento pode ser divida em dois termos, um de esfriamento e um termo restaurador. Calcularemos estes dois termos da força separadamente.

2.1.1 - Termo de Esfriamento

Os átomos de um vapor podem possuir diferentes velocidades, as quais obedecem a uma distribuição de velocidades de Maxwell-Boltzmann. Então façamos incidir nestes um feixe de luz laser com uma freqüência ω_L ligeiramente abaixo da freqüência da transição atômica ω_A. Veja figura abaixo.

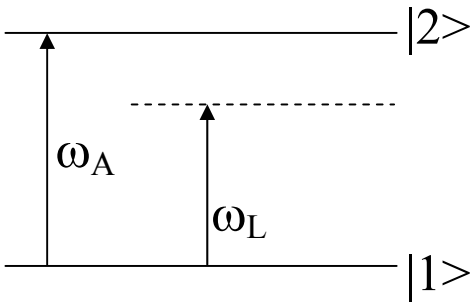


Figura 2.1 – Representação de um sistema de dois níveis.

Chamaremos a dessintonização entre a freqüência do laser e a freqüência da transição atômica para o átomo parado de $\delta = \omega_A - \omega_L$.

Por causa do movimento dos átomos, a freqüência da luz laser será sentida de forma diferente para cada um deles. Identificaremos essa freqüência deslocada por ***ω'***, a qual é dada pela expressão do efeito Doppler:

$$\omega' = \delta - \vec{k} \cdot \vec{\upsilon}\,, \tag{2.1}$$

onde $\vec{k}$ é o vetor de onda da luz laser e $\vec{\upsilon}$ é a velocidade do átomo em questão. Assim, de acordo com a sua velocidade o átomo poderá estar mais ressonante para absorver os fótons do feixe de luz, se a freqüência do laser estiver abaixo da freqüência de transição de um átomo parado (figura 2.1) e a velocidade do átomo for oposta ao vetor de onda do laser.

Após a absorção, o átomo retornará para o estado fundamental emitindo um fóton. Após vários ciclos de absorção e emissão, a velocidade do átomo diminui. O efeito coletivo, numa configuração de seis feixes contra-propagantes dois a dois, é a diminuição da temperatura da nuvem de átomos armadilhados. A quantidade de ciclos de absorção e emissão, necessária para esfriar a nuvem atômica, depende da velocidade de cada átomo. O tempo médio de permanência no estado excitado é dado pelo inverso da largura de linha da transição (Γ^{-1}) sendo a direção das emissões espontâneas, aleatória.

A força média sofrida pelo átomo é devida aos processos de absorção dos fótons provenientes dos feixes de luz laser, ou seja, a troca de momentum entre os fótons e o átomo é que ocasionará o surgimento da força de resfriamento. Sabemos que a força média é a variação temporal média do momentum total do átomo. Na forma de equação, temos:

$$< \vec{F} >=< \frac{\Delta \vec{P}}{\Delta t} >= \frac{< \Delta \vec{P} >}{\Delta t}, \qquad (2.2)$$

onde $\vec{P}$ é o momentum após alguns ciclos de emissão e absorção.

Levando-se em conta todos os processos de transição (absorção, emissão estimulada e emissão espontânea), temos que o momentum num certo intervalo de tempo será [27]:

$$\vec{P} = \vec{P}_0 + \hbar \cdot \vec{k}(N_+ - N_-) + \sum_s \hbar \vec{k}_s \qquad , \qquad (2.3)$$

onde $\vec{P}_0$ é o momentum inicial do átomo, $\hbar$ é a constante de Planck, N_+ é o número de fótons *absorvidos* de forma estimulada, N_- é o número de fótons *emitidos* de forma estimulada, Σ é o somatório sobre os fótons emitidos de forma espontânea e $\vec{k}_S$ é o vetor de onda de um fóton emitido espontaneamente.

Tomando a média temporal do momentum transferido ao átomo, e sabendo que no processo de emissão espontânea os fótons são emitidos de forma aleatória, isto é,

$$< \sum_s \hbar . \vec{k}_s > = 0 \quad , \tag{2.4}$$

temos então que a força média é dada por:

$$< \vec{F} > = \hbar \vec{k} \left[\frac{< N_+ > - < N_- >}{\Delta t} \right] . \tag{2.5}$$

Consideramos que o sistema de dois níveis junto com todos os processos de transição é descrito como indicado na figura 2.2, onde n_1 é a população normalizada dos átomos no estado fundamental (|1>) e n_2 a população normalizada dos átomos no estado excitado (|2>). W_{esp} é a taxa de emissão espontânea, W_{em} a taxa de emissão estimulada e W_{abs} a taxa de absorção estimulada. Com isto, temos que o número médio de fótons absorvidos e emitidos, de forma estimulada, pode ser escrito como:

$$\begin{aligned} < N_+ > &= n_1 W_{abs} \Delta t \quad , \\ < N_- > &= n_2 W_{em} \Delta t \quad . \end{aligned} \tag{2.6}$$

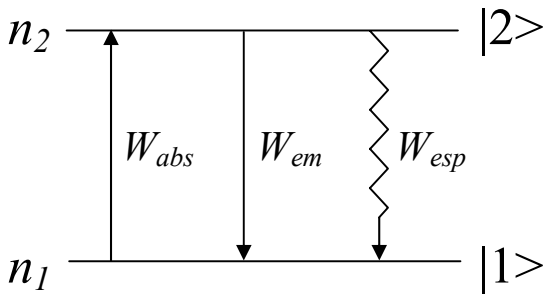


Figura 2.2 - Esquema de um sistema de dois níveis com as respectivas taxas de absorção, emissão estimulada e emissão espontânea.

No regime estacionário, e valendo-se da condição de normalização (n_2 + n_1 = 1), podemos escrever:

$$n_1 W_{abs} = n_2 (W_{em} + W_{esp}) . \tag{2.7}$$

Fazendo W_{esp} = Γ, e usando as equações 2.5 e 2.7, temos que a força média pode ser escrita como:

$$< \vec{F} >= \hbar \vec{k} \Gamma n_2 . \tag{2.8}$$

Para encontrarmos a densidade de átomos no estado excitado, n_2, é necessário resolver a equação de Schrödinger para o sistema, o que equivale a encontrar as amplitudes de probabilidade do átomo estar nos estados excitado e fundamental. No formalismo da matriz densidade, isto equivale a encontrar os elementos que compõem esta matriz.

Comecemos por considerar o hamiltoniano para esse sistema como:

$$\hat{H} = \hat{H}_0 + \hat{H}_{\text{int}} , \tag{2.9}$$

onde H_0 é o hamiltoniano do átomo isolado.

Já o hamiltoniano de interação entre o átomo e o campo elétrico, na aproximação de dipolo elétrico é dado por:

$$\hat{H}_{\text{int}} = -\vec{p}.\vec{E}(\vec{r},t) , \tag{2.10}$$

onde $\vec{p}$ é o momento de dipolo elétrico do átomo e $\vec{E}(\vec{r},t)$ é o vetor campo elétrico do laser, que depende da posição e do tempo.

A matriz densidade para um sistema de dois níveis é dada por:

$$\hat{\rho} = \begin{pmatrix} \rho_{ee} & \rho_{ef} \\ \rho_{fe} & \rho_{ff} \end{pmatrix}, \tag{2.11}$$

onde ρ_{ee} e ρ_{ff} são as populações nos estados: excitado e fundamental, respectivamente, e os elementos fora da diagonal principal representam a coerência entre estes estados.

A evolução da matriz densidade na representação de Schrödinger é dada pela equação de Liouville [29]:

$$\frac{d\hat{\rho}}{dt} = \frac{i}{\hbar}\left[\hat{\rho}, \hat{H}\right] + \textit{termos de relaxação.} \tag{2.12}$$

Os termos de relaxação são inseridos com base em considerações físicas. Ao desligarmos o campo elétrico, as colisões destroem a coerência entre o estado excitado e o estado fundamental. O tempo de relaxação da coerência usado foi $\Gamma/2$.

Resolvendo a equação de Liouville, obtemos:

$$n_2 = \rho_{ee} = \frac{S}{2(1+S)}, \tag{2.13}$$

sendo S o parâmetro de saturação para um feixe dado por:

$$S = \frac{\Omega^2}{2(\omega')^2 + \frac{\Gamma^2}{2}}, \tag{2.14}$$

onde $\Omega(\vec{r}) = \Omega_0 \exp[r_\perp{}^2 / w^2]$, $\vec{r}_\perp$ é o vetor posição perpendicular a $\vec{k}$, $\Omega_0 = 2pE_0/\hbar$ é a freqüência de Rabi, w é a cintura do feixe e a exponencial deve-se ao fato de estarmos considerando feixes gaussianos.

O cálculo detalhado dos elementos da matriz densidade pode ser encontrado no livro de Yariv [28].

Podemos escrever então a força exercida por um dos feixes laser usando as equações 2.8, 2.13 e 2.14 como:

$$\vec{F} = \frac{\hbar\Gamma\Omega^2\vec{k}}{\Gamma^2 + 2\Omega^2 + 4(\delta - \vec{k}\cdot\vec{\upsilon})^2}. \tag{2.15}$$

2.1.2 - Termo Restaurador

Ainda não é possível aprisionar os átomos somente com esta força, pois à medida que os átomos interagem com a luz laser vão perdendo gradativamente velocidade. Os fótons do laser exercem sobre os átomos uma força resistiva comportando-se como um fluido altamente viscoso, o que se conhece como "melaço ótico". Essa força resistiva somente

desacelera os átomos sem conduzi-los ao centro da armadilha. É necessário, portanto, inserir no sistema um campo magnético que remova a simetria de translação.

O campo magnético, $\vec{B}$, é produzido experimentalmente por duas bobinas coaxiais percorridas por correntes em direções opostas uma da outra (bobinas anti-Helmholtz) e é dado por:

$$\vec{B} = b\left(z\hat{k} - \frac{x}{2}\hat{i} - \frac{y}{2}\hat{j} \right), \tag{2.16}$$

onde *b* é o gradiente de campo magnético.

A presença desse campo levanta a degenerescência dos níveis atômicos associada às componentes hiperfinas do átomo (efeito Zeeman), e o sistema representado na figura 2.1 toma uma nova configuração representada na figura 2.3.

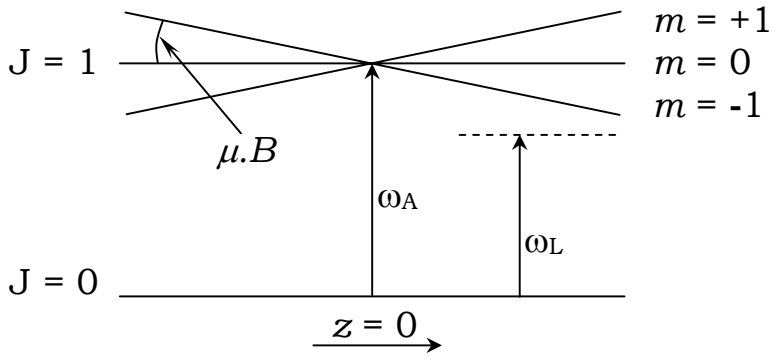


Figura 2.3 – Diagrama de dois níveis na presença de um campo magnético. O nível fundamental |1> tem degenerescência zero enquanto que o nível excitado |2> tem tripla degenerescência. Os sub-níveis Zeeman são representados por *m*.

Para que os átomos sejam conduzidos a uma mesma posição de equilíbrio, os feixes deverão ter polarizações opostas, σ^+ e σ^-. Assim, os átomos tenderão a absorver mais fótons do feixe que se propaga na direção oposta àquela para a qual o átomo está deslocado, ou seja, na direção *z*, por exemplo, o átomo deslocado para a esquerda ($z < 0$ – figura 2.3) tende a absorver mais fótons do feixe que se propaga na direção +z, enquanto o átomo deslocado para a direita ($z > 0$) tende a absorver mais fótons do feixe que se propaga na direção -z. O feixe que se propaga no sentido +*z* tem polarização σ^+ e excita o átomo para o sub-nível com número quântico magnético $m = 1$, enquanto que o feixe que se propaga no sentido –*z* tem polarização σ^- e excita o átomo para o sub-nível com $m = -1$ preferencialmente, criando uma posição de equilíbrio em torno de $z = 0$.

Considerando agora o efeito Zeeman, temos que a freqüência sentida pelo átomo será:

$$\omega' = \delta - \vec{k} \cdot \vec{\upsilon} - \frac{\vec{\mu} \cdot \vec{B}}{\hbar}, \qquad (2.17)$$

onde $\vec{\mu}$ é o momento de dipolo magnético.

A forma final da força exercida por um único feixe de luz laser pode ser escrita como:

$$\vec{F} = \frac{\hbar\Gamma\Omega^2\vec{k}}{\Gamma^2 + 2\Omega^2 + 4\left(\delta - \vec{k}\cdot\vec{\upsilon} - \frac{\vec{\mu}\cdot\vec{B}}{\hbar}\right)^2}\,. \tag{2.18}$$

Para obtermos a forma final da força devido aos seis feixes, faremos aqui uma aproximação. Esta consiste em considerar a força total que atua sobre cada átomo sendo dada pela soma da força exercida por cada feixe individualmente, a saber:

$$\vec{F} = F_{+x} + F_{-x} + F_{+y} + F_{-y} + F_{+z} + F_{-z}\,, \tag{2.19}$$

com cada componente dada por:

$$F_{\pm x} = \frac{\mp\hbar k\Gamma{\Omega_0}^2\exp\left(-\frac{y^2+z^2}{w^2}\right)}{\Gamma^2 + 2{\Omega_0}^2\exp\left(-\frac{y^2+z^2}{w^2}\right) + 4\left(\delta \pm k\upsilon_{\pm x} \pm \frac{\gamma bx}{2}\right)^2}\,, \tag{2.20}$$

$$F_{\pm y} = \frac{\mp\hbar k\Gamma{\Omega_0}^2\exp\left(-\frac{x^2+z^2}{w^2}\right)}{\Gamma^2 + 2{\Omega_0}^2\exp\left(-\frac{x^2+z^2}{w^2}\right) + 4\left(\delta \pm k\upsilon_{\pm y} \pm \frac{\gamma by}{2}\right)^2}\,, \tag{2.21}$$

$$F_{\pm z} = \frac{\mp\hbar k\Gamma{\Omega_0}^2\exp\left(-\frac{x^2+y^2}{w^2}\right)}{\Gamma^2 + 2{\Omega_0}^2\exp\left(-\frac{x^2+y^2}{w^2}\right) + 4\left(\delta \pm k\upsilon_{\pm z} \mp \gamma bz\right)^2}\,, \tag{2.22}$$

onde $\gamma = \mu/\hbar$ é a constante Zeeman do átomo.

Esta é a forma final da força devido à luz laser, junto com a contribuição do campo magnético, que atua sobre cada átomo na armadilha.

Com esta aproximação, estamos desprezando boa parte dos efeitos de saturação que ocorrem no sistema devido à competição entre os diversos feixes nos processos de absorção e emissão estimulada. A aproximação feita é válida para o regime de baixa saturação ($S<<1$) [5], onde os efeitos de competição não são importantes. Apesar de preferencialmente restrita à utilização em baixa saturação, esta aproximação também pode ser usada no regime de alta saturação. Quando aplicada nesta região, esta aproximação é muitas vezes utilizada em conjunto com modificações empíricas nos termos de saturação das expressões das forças [22].

2.2 – Forças de Interação Entre Átomos Aprisionados

Acreditava-se inicialmente [5] que fosse possível acrescentar cada vez mais átomos na armadilha até que, a partir de um certo limite, a nuvem atômica implodiria. Em estudos posteriores [23] foi mostrado que o acréscimo de átomos no sistema provoca o aumento da densidade da

nuvem atômica até que essa atinja uma densidade limite, a partir da qual a nuvem somente cresce sem aumentar sua densidade, uma vez que, numa armadilha, além da ação das forças usadas para o aprisionamento, cada átomo também sofre a ação da presença dos demais átomos.

Nessa seção, faremos uma dedução da forma da força de interação entre os átomos aprisionados e de sua dependência com a posição e a velocidade dos átomos em cada ponto. Seguiremos a dedução feita por Steane e colaboradores [23].

A força de interação entre os átomos na armadilha pode ser descrita como a composição de dois termos, um deles devido ao múltiplo espalhamento de fótons vindos do laser e o outro, devido ao efeito de sombreamento causado pela presença dos demais átomos.

Começaremos considerando que a razão na qual um átomo espalha fótons provenientes do laser é dada por $n_2\Gamma$, sendo n_2 a população de átomos no estado excitado e Γ, a taxa de relaxação do estado excitado para o estado fundamental.

Considere a situação representada na figura abaixo:

Figura 2.4 – Laser incidindo sobre átomo 1 que por sua vez sombreia o átomo 2.

Como discutimos anteriormente, a força média exercida sobre o átomo 1 é dada pelo produto entre a taxa na qual esse espalha a luz laser e o momentum transportado por cada fóton. A luz espalhada não tem direção preferencial de forma que sua intensidade cai com o quadrado da distância ao átomo espalhador. A força produzida pela incidência dessa luz espalhada pelo primeiro átomo num outro situado a uma distância *r* será:

$$\vec{F}_R = n_2 \Gamma \hbar k \frac{\sigma_R}{4\pi r^2} \hat{r} = \frac{\hbar k \Gamma}{2} \frac{S}{(S+1)} \frac{\sigma_R}{4\pi r^2} \hat{r} \,, \qquad (2.23)$$

onde $k\hbar$ é o momento transferido ao segundo átomo devido à luz espalhada pelo primeiro, Γn_2 é a taxa na qual o primeiro espalha luz laser, σ_R é a secção transversal de espalhamento do átomo 2 para a luz espalhada pelo átomo 1, $\hat{r}$ é o versor na direção do átomo 1 para o átomo 2 e 4π o ângulo sólido.

Podemos expressar a componente repulsiva da força de interação de outra forma dizendo que a intensidade da radiação que incide no segundo átomo proveniente do primeiro é dada por:

$$I_{rad} = \frac{\sigma_L I_T}{4\pi r^2} \,, \qquad (2.24)$$

onde I_T é a intensidade da luz de um feixe laser e σ_L é a secção transversal de espalhamento para a luz laser.

A taxa na qual o segundo átomo absorve radiação proveniente do primeiro é $I_{rad}.\sigma_R$. Essa luz absorvida pelo segundo átomo vai transferir momento para este de modo que surge uma força repulsiva dada por:

$$\vec{F}_R = \frac{\sigma_R \sigma_L I_T}{4\pi c r^2}\hat{r}\,, \tag{2.25}$$

onde c é a velocidade da luz.

As duas formas descritas para a força devido a re-emissão (força repulsiva, equações 2.23 e 2.25) devem ser equivalentes. Delas podemos obter uma expressão para a secção transversal de espalhamento devido à luz laser:

$$\sigma_L = \frac{\hbar\omega_L\Gamma}{2}\frac{S}{I_T(S+1)}\,. \tag{2.26}$$

A luz absorvida pelo segundo átomo proveniente do primeiro tende a afastar os dois, ao mesmo tempo em que o átomo 1 impede que o átomo 2 sofra a ação da luz laser, lançando uma sombra sobre este. Este efeito de sombreamento tende a aproximar os dois átomos na medida em que o átomo 1 ao absorver um fóton é empurrado na direção do átomo 2. Em analogia com a força devido ao múltiplo espalhamento, podemos escrever a força de sombreamento como:

$$\vec{F}_S = -\frac{I_T \sigma_L \sigma_L}{4\pi c r^2}\hat{r}\,. \tag{2.27}$$

A força total de interação entre cada par de átomos aprisionados é então obtida pela soma dos termos de re-emissão e sombreamento:

$$\vec{F}_I = \vec{F}_R + \vec{F}_S = \frac{\alpha}{r^2}\hat{r}. \tag{2.28}$$

O parâmetro α é o que chamaremos daqui por diante de parâmetro de interação. Sua forma explícita é:

$$\alpha = \frac{I_T \sigma_L^2 \left(\frac{\sigma_R}{\sigma_L} - 1 \right)}{4\pi c}. \tag{2.29}$$

Note que a força de interação é do tipo $1/r^2$, como a força gravitacional.

A força de interação exercida sobre um átomo *i* devido à presença dos N-1 átomos *j* será então:

$$\vec{F}_i = \sum_{j \neq i}^{N} \alpha \frac{\left(\vec{r}_i - \vec{r}_j \right)}{\left| \vec{r}_i - \vec{r}_j \right|^3}, \tag{2.30}$$

onde $\vec{r}_n$ é o vetor posição do n-ésimo átomo.

É importante ressaltar aqui que no processo de múltiplo espalhamento foram considerados os termos até o segundo espalhamento, o que nos leva a ter a interação apenas entre dois átomos.

O parâmetro α pode ser estimado a partir de dados experimentais. Consideramos os feixes alinhados e fazemos o cálculo para o centro da armadilha. Nessas condições, a razão $\langle\sigma_R\rangle/\langle\sigma_L\rangle = 1{,}2$ e o parâmetro de saturação S = 1 (referências [22] e [23]). Para o átomo de sódio (Na), temos que, a intensidade total no centro da armadilha I_T = 3 x 10^{-4} W/mm^2, a largura de linha da transição $\Gamma/2\pi$ = 10^7 Hz, o vetor de onda dos fótons do laser k = 1,086 x 10^7m^{-1}. Usando esses valores nas equações 2.26 e 2.29, obtemos α = 5,2 x 10^{-30} N mm^2.

2.3 - Modos Orbitais

Conforme comentamos no capítulo I, quando a armadilha está construída com os seis feixes contra-propagantes, a nuvem de átomos aprisionados toma uma forma semelhante a de uma bola, achatada na direção *z*, devido ao campo magnético ser mais intenso nessa direção. À medida que impomos um desalinhamento no plano *xy*, os feixes que se propagavam em direções opostas deixam de se sobrepor (figura 2.5), de forma que a força resultante, no plano *xy*, que conduz os átomos para o centro da armadilha diminui. Como na direção *z* não é imposto nenhum desalinhamento, a força nessa direção tem a função de confinar a nuvem

atômica numa região próxima do plano *xy*. Assim, conforme o desalinhamento vai aumentando, podemos observar uma mudança da forma da nuvem de uma bola para um disco e em seguida para um anel. Este comportamento sugere uma aproximação para o estudo dos modos orbitais: tratar o problema só no plano *xy*, isto é, em duas dimensões, impondo que a coordenada *z* da posição dos átomos seja nula. Implementamos e executamos um programa para simulações em três dimensões e comparamos com o resultado obtido para duas dimensões. Os resultados dessas simulações numéricas, apresentados no capítulo IV, indicam que tratar o problema apenas em duas dimensões consiste numa boa aproximação.

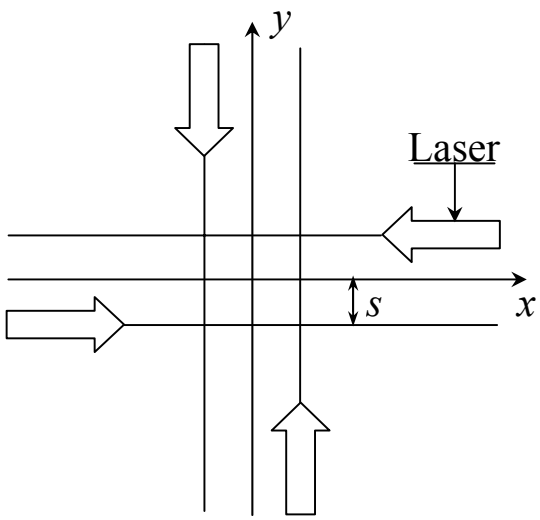


Figura 2.5 – Eixos coordenados com feixes de luz laser da armadilha deslocados no sentido anti-horário de uma quantidade *s*.

Para as simulações em duas dimensões, fazemos $z = 0$ nas equações 2.20, 2.21 e 2.22, $\upsilon_z = 0$ em 2.22, e acrescentamos o parâmetro de desalinhamento, s. Obtemos então:

$$F_{\pm x} = \frac{\mp \hbar k \Gamma {\Omega_0}^2 \exp\left(-\frac{(y \mp s)^2}{w^2}\right)}{\Gamma^2 + 2{\Omega_0}^2 \exp\left(-\frac{(y \mp s)^2}{w^2}\right) + 4\left(\delta \pm k\upsilon_{\pm x} \pm \frac{\gamma' b x}{2}\right)^2}, \qquad (2.31)$$

$$F_{\pm y} = \frac{\mp \hbar k \Gamma {\Omega_0}^2 \exp\left(-\frac{(x \pm s)^2}{w^2}\right)}{\Gamma^2 + 2{\Omega_0}^2 \exp\left(-\frac{(x \pm s)^2}{w^2}\right) + 4\left(\delta \pm k\upsilon_{\pm y} \pm \frac{\gamma' b y}{2}\right)^2}, \qquad (2.32)$$

$$\vec{F}_i = \sum_{j \neq i}^{N-1} \alpha \left\{ \frac{\left(x_i - x_j\right)}{\left|\vec{r}_i - \vec{r}_j\right|^3} + \frac{\left(y_i - y_j\right)}{\left|\vec{r}_i - \vec{r}_j\right|^3} \right\}. \qquad (2.33)$$

Note que nas equações (2.31) e (2.32) o desalinhamento s é acrescentado nas exponenciais pois elas é que descrevem a forma dos feixes gaussianos. O deslocamento dos feixes de laser, para um valor de s positivo, produzirá um movimento orbital no sentido anti-horário. Veja que a força de interação (2.33) sofre apenas a supressão da componente z relativa à posição de cada átomo nesse eixo.

Essas últimas equações (2.31-2.33) serão usadas na construção do programa que fará a simulação da evolução temporal de um conjunto de N átomos numa armadilha magneto-ótica. Os outros parâmetros usados em todas as simulações são apresentados na tabela 2.1 e correspondem ao átomo de sódio (Na).

Átomo de sódio (Na)	
$\Gamma/2\pi$	10^7 Hz
m	$3{,}819 \times 10^{-26}$ kg
k	$1{,}086 \times 10^7$ m^{-1}
$\gamma/2\pi$	1.4×10^6 Hz/G
I_s	5×10^{-5} W/mm^2

Tabela 2.1 – Parâmetros correspondentes ao átomo de sódio, usados nas simulações.

Na referência [11] pode ser encontrado um estudo da dinâmica dos átomos aprisionados num limite de baixas densidades, bem como essa dinâmica se relaciona com as propriedades mensuráveis da armadilha como: gradiente de campo, *detuning*, desalinhamento dos feixes, freqüência de Rabi, etc. Esse estudo é feito usando, basicamente, as equações 2.31 e 2.32.

2.4 – Forma da Força na Aproximação de Movimento Circular Uniforme

Os modos orbitais observados experimentalmente descrevem trajetórias quase circulares. A dependência das forças de aprisionamento com a posição e velocidade dos átomos, dificulta a construção de um gráfico que ilustre o seu comportamento. Para construirmos o gráfico que dá a forma dessa força, faremos aqui uma aproximação que consiste em considerar trajetórias circulares, para o movimento dos átomos na armadilha, onde os feixes de laser no plano *xy* estão desalinhados de uma quantidade *s*.

Como estamos aproximando o movimento como sendo circular e uniforme, podemos analisar o problema sobre apenas um eixo. Tomaremos o eixo *x*. Sobre esse eixo temos $y = 0$, $\upsilon_x = 0$ e

$$F_x = \frac{m\upsilon_y^2}{x}. \tag{2.34}$$

Com essas modificações, as equações para a força de aprisionamento 2.31 e 2.32 tomam a seguinte forma:

$$F_x = \frac{\hbar k \Gamma A}{\Gamma^2 + 2A + 4\left(\delta - \frac{\gamma' bx}{2}\right)^2} - \frac{\hbar k \Gamma A}{\Gamma^2 + 2A + 4\left(\delta + \frac{\gamma' bx}{2}\right)^2}, \tag{2.35}$$

$$F_y = \frac{\hbar k \Gamma A E_-}{\Gamma^2 + 2AE_- + 4\left(\delta - k\upsilon_y\right)^2} - \frac{\hbar k \Gamma A E_+}{\Gamma^2 + 2AE_+ + 4\left(\delta + k\upsilon_y\right)^2}, \qquad (2.36)$$

com

$$A = \Omega_0{}^2 \exp\left(-\frac{s^2}{w^2}\right),\ E_+ = \exp\left(-\frac{x^2 + 2sx}{w^2}\right)$$

$$\text{e}\ E_- = \exp\left(-\frac{x^2 - 2sx}{w^2}\right). \qquad (2.37)$$

Para a construção da figura 2.6, usamos as equações 2.34-2.37 obtemos a forma da força na direção *y*, para pontos sobre o eixo *x*. Os parâmetros usados foram: cintura do feixe laser, ω = 3mm; desalinhamento dos feixes no plano *xy*, $s = \omega$; freqüência de Rabi, $\Omega_0 = 4\Gamma$ e gradiente de campo magnético b = 0,5 G/mm.

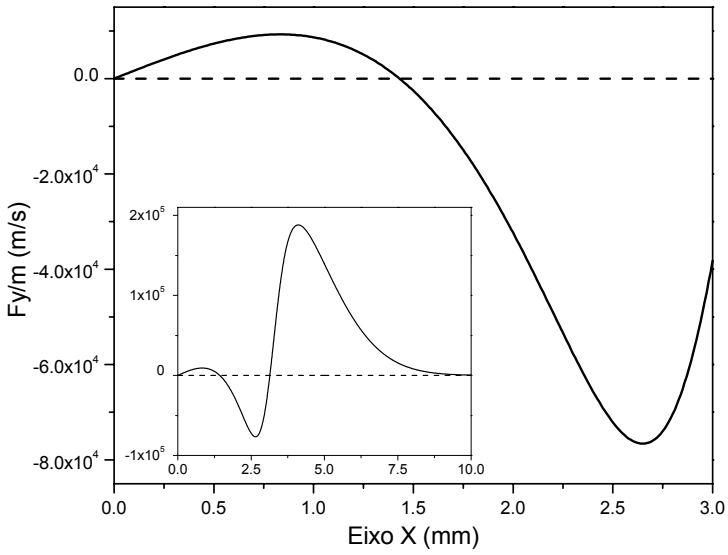


Figura 2.6 – Força na direção y sobre o eixo x usando a aproximação de movimento circular uniforme.

Com base no gráfico mostrado na figura 2.6, podemos dizer que existe uma órbita estável para um raio de 1,43 milímetros o valor obtido através das simulações realizadas é de 1,34 milímetro. Também podemos afirmar que átomos em raios acima de 3,1 milímetros serão expulsos. As simulações mostram que para raios maiores que 2,4 milímetros os átomos são de fato jogados para fora da armadilha.

Capítulo III

Métodos Computacionais.

O problema de N corpos interagentes consiste em solucionar o conjunto de N-1 equações acopladas, correspondentes à interação de cada um dos N corpos com os N-1 demais. É impraticável solucionar analiticamente essas equações, que para o nosso problema, consiste em resolver o somatório na equação 2.30. É necessário então resolver numericamente esse conjunto de equações. Escrever um programa de computador para resolver esse problema exatamente é relativamente fácil. Ocorre que um programa para resolver diretamente tem um custo computacional muito elevado, ou seja, o tempo de execução, ou tempo de máquina cresce proporcionalmente ao quadrado do número de átomos envolvidos, pois cada átomo interage com os outros, N-1, átomos. Esse crescimento nos limita a um valor de $N \sim 10^4$ átomos, menor que o valor que consideramos ser necessário manipular para observar os efeitos da interação numa armadilha ($N \sim 10^7$ átomos). Valores numéricos desse crescimento serão apresentados no capítulo seguinte.

Procedemos à busca e implementação de um método aproximativo, com o objetivo de obter ganho no tempo de máquina em relação ao método direto. Com um menor custo computacional, nós podemos aumentar o valor de N nas simulações.

Estudos quantitativos na linha de iteração numérica, para a solução do problema de muitos corpos interagentes, são muito comuns nos ramos da Cosmologia, Astronomia e Astrofísica. Um apanhado dos métodos usados atualmente com suas vantagens e seus limites pode ser encontrado na referência [39]. Nas referências [43] e [44], encontramos aplicações a problemas específicos usando técnicas de simulação de N corpos interagentes. É desses ramos da Física que trazemos o método aproximativo usando estrutura hierárquica, proposto na referência [38], pois a forma da força de interação (equação 2.30), existente no caso de átomos presos em armadilhas, é a mesma existente entre corpos celestes, ou seja, ambas decrescem com o inverso do quadrado da distância que separa dois corpos sujeitos a essas forças.

Esse capítulo está dividido em quatro seções: na primeira fazemos o cálculo do tempo gasto na aproximação de campo médio tradicional e comparamos este com o tempo proposto pelo método da Árvore Hierárquica ou Estrutura Hierárquica. Esse segundo método aproximativo está descrito na segunda seção e foi proposto por Barnes & Hut [38]. Na terceira seção fazemos o cálculo do tempo computacional gasto no método

da árvore hierárquica; na última seção são fornecidas algumas especificações técnicas da programação para o desenvolvimento dos softwares usados nas simulações. O código fonte do software, usado como base para a construção dos demais, pode ser encontrado num apêndice no final dessa dissertação.

3.1 – Campo Médio versus Árvore Hierárquica

A escolha do método aproximativo a ser implementado e utilizado foi feita levando em conta o ganho no tempo de execução. Para o método árvore, o tempo de execução deve crescer com $N \log N$ (seção 3.3). Faremos aqui, o cálculo para o comportamento do tempo de máquina para a aproximação de campo médio tradicional em duas dimensões.

Na aproximação de campo médio, o espaço é dividido numa grade de células do mesmo tamanho. A grade deve ser grande o suficiente para englobar todo o sistema a ser estudado. Consideraremos que a interação deve ser calculada exatamente para átomos na mesma célula e nas células imediatamente vizinhas, enquanto que para os átomos em cada uma das demais células, a aproximação consiste em considerar todos esses átomos como um único átomo equivalente situado no centro de massa dessa

célula. Na figura 3.1 destacamos uma célula em cinza escuro com suas primeiras vizinhas em cinza mais claro.

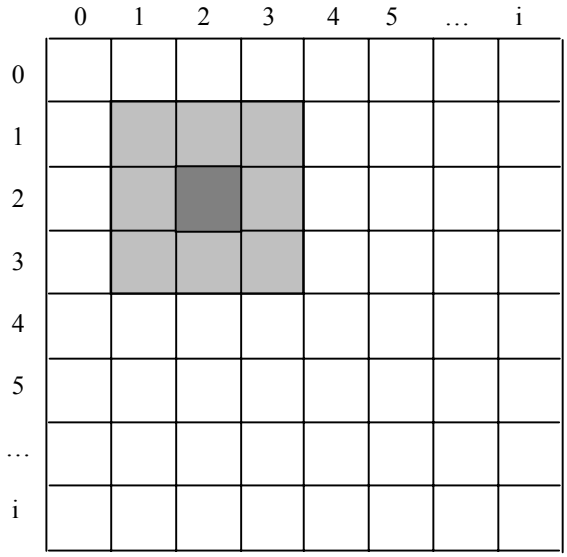


Figura 3.1 – Grade para a interação segundo a aproximação de campo médio.

Seja N o número de átomos no sistema. No caso exato, o número de iterações entre pares de átomos é $N(N-1)/2 \sim N^2$. Façamos o cálculo para uma distribuição espacial uniforme de átomos na aproximação de campo médio. Seja n_c o número total de células no qual o sistema está dividido.

O número de iterações entre os pares de átomos, o qual é proporcional ao tempo de execução de máquina, é dado por:

$$TM \propto \frac{n(n-1)}{2} n_C + \frac{n^2}{2} 8 n_C + n(n_C - 9) n_C \,, \qquad (3.1)$$

onde $n = N/n_C$ é o número de átomos em cada célula no caso de distribuição uniforme.

Para $N >> n_C$ e $n_C >> 1$, obtemos:

$$TM \propto \frac{9}{2}\frac{N^2}{n_C} + Nn_C , \tag{3.2}$$

Da equação anterior temos que o tempo de máquina é mínimo quando $n_C = 3(N/2)^{1/2}$. Substituindo esse valor na equação anterior, temos:

$$TM \propto N^{3/2} . \tag{3.3}$$

Esse é o menor fator com o qual deve crescer o tempo de máquina para uma simulação de um sistema com N átomos interagentes segundo a aproximação de campo médio. É importante notificar que quando consideramos primeiros e segundos vizinhos, o crescimento muda apenas por um fator constante.

Na figura 3.2 comparamos o crescimento teórico da aproximação de campo médio com o método árvore. Os pontos quadrados indicam os tempos obtidos para a aproximação de campo médio, enquanto os pontos circulares, indicam o crescimento para o método árvore.

Para efeito de comparação, o tempo previsto para a simulação de um sistema com N = 10^5 átomos é duas ordens de grandeza maior para o

campo médio, o que justifica o investimento na implementação do método aproximativo da árvore hierárquica.

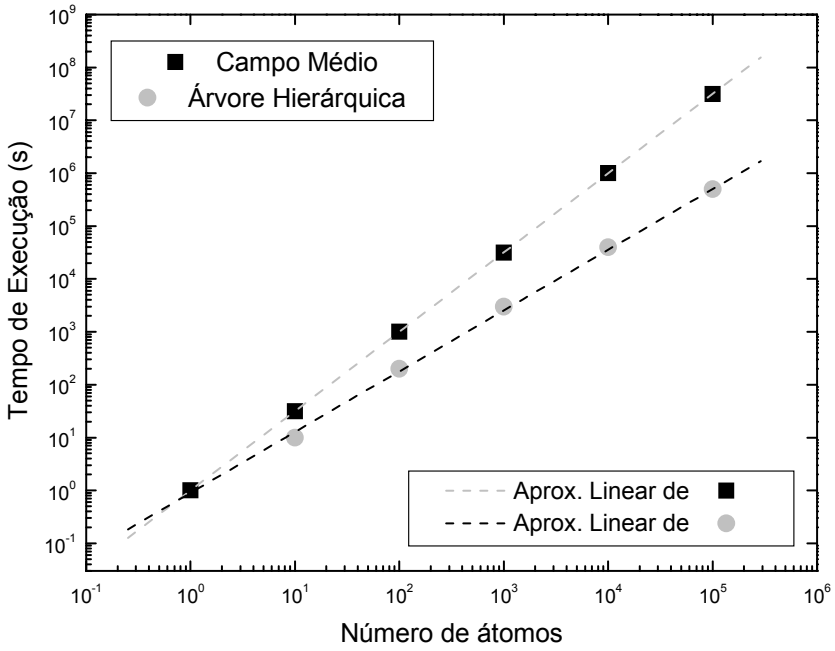


Figura 3.2 – Comparação do crescimento do tempo de máquina para dois métodos aproximativos distintos.

3.2 - Descrição do Método da Árvore Hierárquica

A estrutura hierárquica apresentada aqui foi desenvolvida a partir do artigo de Barnes & Hut [38] no contexto astronômico destinado a melhorar o desempenho na solução do problema de N corpos auto-gravitantes. A escolha desse método ocorreu devido à sua ambiciosa proposta de reduzir o crescimento do tempo computacional na interação

de N corpos para $N \log N$ (seção seguinte), em comparação com o crescimento que se obtém no cálculo exato, que é da ordem N^2. A grade construída na árvore hierárquica amolda-se à distribuição existente cobrindo toda uma área previamente definida. Essa área é escolhida como sendo suficientemente grande para englobar a armadilha (figura 3.3).

O método hierárquico também permite controlar o nível de aproximação feito no cálculo da interação entre os átomos. Esse controle é feito através do parâmetro de precisão θ, a ser definido mais adiante, que serve como uma espécie de botão de ajuste para aumentar ou diminuir o nível de aproximação feita no cálculo das interações.

A hierarquia na qual se baseia o método se apresenta na forma como é construída a grade de controle da interação. Os seus pormenores ficarão mais claros na sub-seção seguinte quando descrevermos em maior detalhe o funcionamento do programa desenvolvido com o intuito de simular a evolução temporal de um conjunto de N átomos sob a ação das forças de uma armadilha magneto-ótica acrescida da força de interação entre os próprios átomos.

Os ingredientes essenciais para a implementação de um programa usando esse método são:

1) Divisão virtual do espaço em células e sub-células quadradas, com as sub-células tendo seu tamanho como sendo exatamente igual à metade do lado da célula mãe;

2) A construção de uma estrutura semelhante a uma árvore virtual de células com as seguintes condições:

i) descartando células vazias;

ii) dividindo recursivamente células com mais de um átomo em sub-células;

iii) aceitando células com apenas um átomo;

3) Refazendo essa construção a cada passo.

Como estamos tratando de um problema muito específico: o comportamento de átomos confinados numa região do espaço bem localizada, nós vamos tomar uma porção fixa do espaço. Por simplicidade tomaremos um quadrado de lado a como a região inicial, que permanecerá inalterada durante todo o processo e a qual chamaremos de célula mãe (figura 3.3). Fora dessa região os átomos são expulsos ou escapam da armadilha e, portanto, não participam mais da simulação.

Para melhor nos situarmos, estamos considerando que a origem do nosso sistema de eixos coordenados coincide com o centro da armadilha e também coincide com o centro da célula mãe.

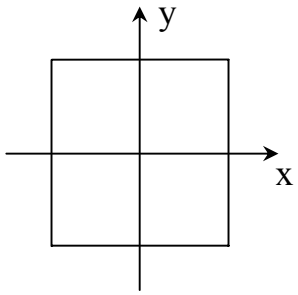


Figura 3.3 - Célula mãe com os eixos coordenados.

Como discutimos na seção 2.3, vamos tratar o problema em duas dimensões, então faremos a dedução do método hierárquico somente para duas dimensões, embora também tenhamos implementado este em três dimensões.

3.2.1 – Divisão Virtual do Espaço

Começamos com a célula mãe na qual distribuímos aleatoriamente um conjunto de N átomos. A construção da árvore hierárquica ocorre concomitantemente à divisão virtual do espaço.

A figura 3.4 mostra de forma esquemática o processo de divisão do espaço. Dividimos inicialmente a célula mãe em quatro sub-células ou células filhas, de modo que cada uma das sub-células tenha exatamente metade do lado da célula mãe (fig. 3.4b). Deslocamos nossa atenção para

cada uma das células filhas e seguimos dividindo recursivamente estas em sub-células até que reste somente um átomo em cada célula (fig. 3.4c e 3.4d). Descartamos as células vazias, pois estas não contribuem para a iteração.

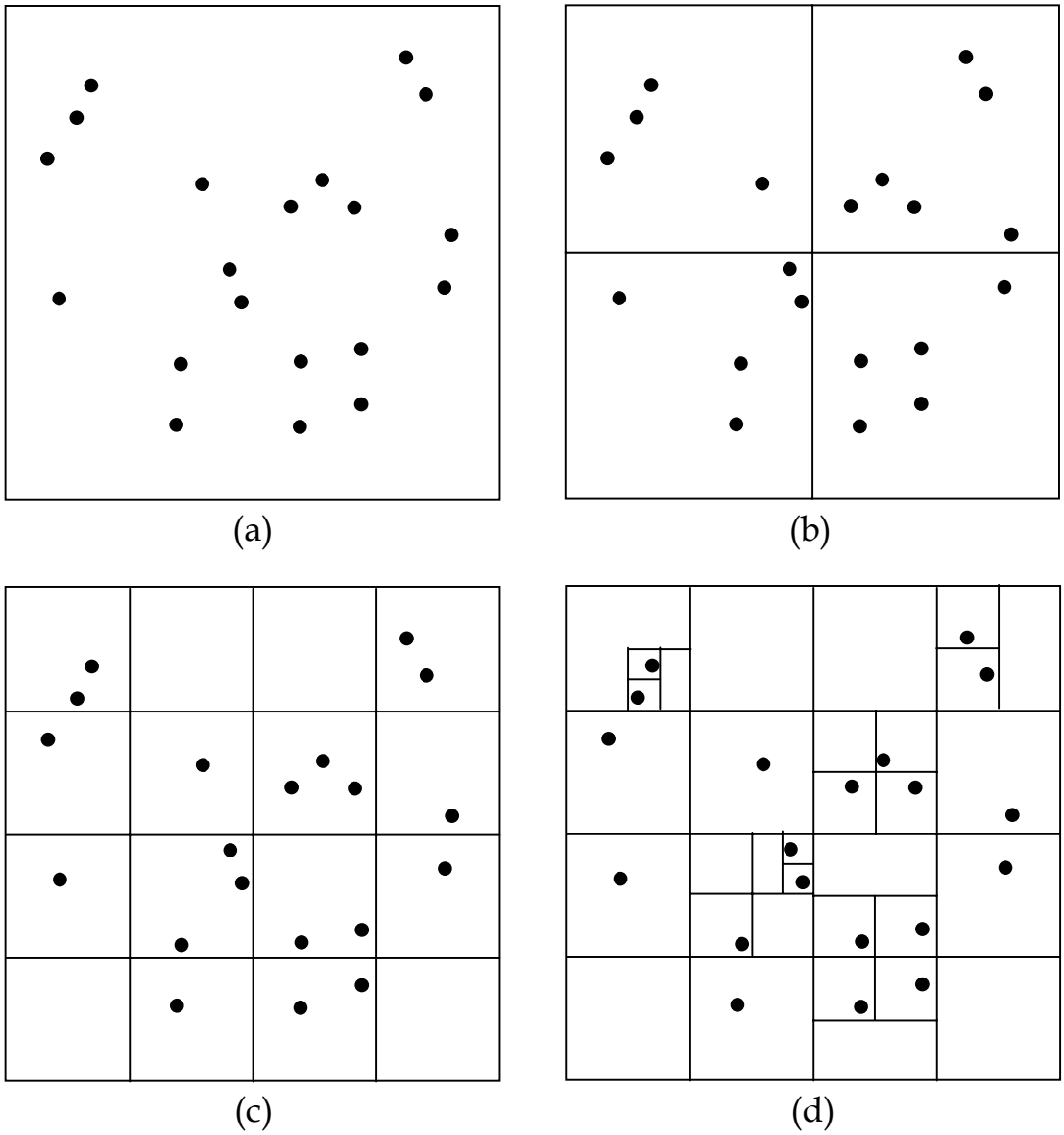


Figura 3.4 – Alguns passos no processo de divisão virtual do espaço.

A forma final do processo de divisão virtual do espaço é mostrada na figura 3.4d para a distribuição tomada como exemplo. A árvore é construída usando todos os passos intermediários, desde a célula mãe, contendo todos os átomos, até a forma final da divisão com um átomo em cada célula. Uma forma ilustrativa da árvore, associada ao exemplo apresentado na figura 3.4, é mostrada na figura 3.5.

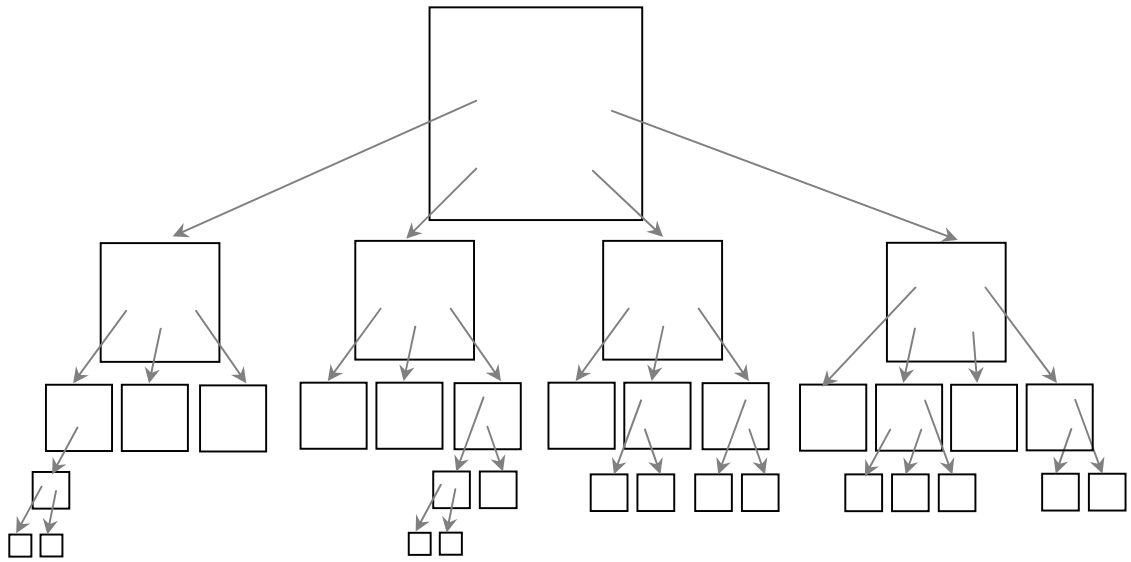


Figura 3.5 – Árvore construída juntando todos os processos intermediários da divisão virtual do espaço, mostrado na figura 3.4.

Temos que cada conjunto de células filhas é diretamente subordinado às células de onde estas se originam. Esta condição de

subordinação estabelece uma hierarquia e essa hierarquia está na base das aproximações feitas no cálculo da interação. Chamaremos de nível cada conjunto de células do mesmo tamanho.

3.2.2 - Cálculo da Interação

Para facilitar o entendimento da forma como é calculada a interação no método da estrutura hierárquica, faremos, sempre que possível, um paralelo com o método exato e a aproximação de campo médio.

Calcular a interação no método exato, onde todos os átomos interagem com os demais um a um, consiste em calcular os elementos da matriz de interação. Cada elemento M_{ij} dessa matriz corresponde à interação entre o átomo A_i com o átomo A_j. Na aproximação usando a estrutura hierárquica, a matriz de interação é diferente. Cada átomo A_i interage com um conjunto de átomos situado numa célula C_j. A aproximação feita consiste em considerar todos os átomos presentes na célula C_j como estando situados no centro de massa da célula, como é feito na aproximação de campo médio.

Para a escolha das células que irão interagir com o átomo A_i, utilizamos uma condição de controle a qual é aplicada à estrutura criada

em forma de árvore. Esta condição de controle é definida do seguinte modo: seja d_{ij} a distância do átomo A_i até o centro de massa da célula C_j e a_j o lado dessa célula. Caso a razão entre, a_j e d_{ij} seja menor que um dado valor θ, então a interação entre o átomo A_i e a célula C_j é calculada, considerando que a posição dos átomos contidos nessa célula pode ser aproximada pela posição do centro de massa da célula. Essa condição é expressa pela desigualdade:

$$\frac{a_j}{d_{ij}} < \theta \,. \tag{3.4}$$

Assim, o cálculo das forças de interação sobre cada átomo, devido aos outros N-1 átomos da armadilha, é efetuado percorrendo a árvore de cima a baixo e executando a condição de controle em cada nível. Isto é, caso a condição (3.4) não seja satisfeita e a célula C_j tenha mais que um átomo, então deslocamos nossa atenção para as células filhas que estão subordinadas a essa e fazemos o teste novamente com cada uma delas, repetindo recursivamente até a última célula no nível mais inferior.

Como o próprio nome indica, a condição (3.4) nos permite controlar o grau de aproximação dos nossos cálculos. Uma forma de visualizar isso é analisando os limites da desigualdade.

i) No limite em que θ tende a infinito, a condição de controle sempre será satisfeita, uma vez que começamos tomando inicialmente a célula

mãe e a interação se reduz a um problema de dois corpos, pois cada átomo interage somente com o centro de massa do sistema de N-1 átomos. Nessas condições, é lógico afirmar que o erro é máximo, embora o tempo de execução seja mínimo, pois cresce com N. Não há necessidade de acessar os níveis inferiores da estrutura hierárquica.

ii) Para θ igual a zero, a condição de controle só é satisfeita quando a célula contiver apenas um átomo. Dessa forma percorremos a árvore hierárquica desde a célula mãe até os níveis mais inferiores de modo que cada átomo interage diretamente com todas as células que contém apenas um átomo. Nesse caso não estamos fazendo nenhuma aproximação no cálculo da interação, em contrapartida o tempo de execução cresce com N^2.

À medida que aumentamos o valor de θ, perdemos precisão e ganhamos em tempo de máquina, da mesma forma, quando diminuímos o valor de θ, ganhamos em precisão e aumentamos o tempo de máquina até o limite em que θ é igual a zero e temos o cálculo exato na interação. No capítulo seguinte, apresentamos resultados para diferentes valores de θ.

Para θ igual a 1, a condição (3.4) é satisfeita quando a distância entre o átomo e o centro de massa da célula considerada for maior que o lado dessa mesma célula. Adotamos θ igual 1 na maior parte dos nossos resultados. Este valor de θ nos dá um tempo de máquina que cresce com

$N \log N$, que para as máquinas utilizadas, nos permite chegar até N ~ 10^6 átomos.

Para visualizar essa situação, mostramos na figura 3.6 um resultado obtido para θ *igual a* 1 pela execução da parte do programa, que faz a divisão do espaço e seleciona quais células interagem com cada átomo. Apresentamos inicialmente uma distribuição aleatória dos átomos (fig. 3.6a). Em seguida trazemos uma representação da distribuição com o espaço dividido em células (fig. 3.6b). Usando a condição (3.4), com θ = 1, tomamos os átomos um a um e calculamos as células que interagem com cada átomo.

Nas figuras 3.6c e 3.6d mostramos dois exemplos que correspondem ao cálculo para dois átomos distintos. Vemos que o número de células que interagem com cada átomo (em destaque nas figuras 3.6c e 3.6d), para uma mesma configuração muda consideravelmente dependendo da posição de cada átomo em relação ao sistema como um todo. No exemplo mostrado na figura 3.6c, o número de células para interação é mais que o dobro que no exemplo seguinte (fig. 3.6d). Embora haja essa diferença, a aproximação feita, em cada caso é a mesma.

Quando usamos a condição (3.4) estamos impondo um limite na aproximação feita. Como cada célula passa pelo teste, então cada uma delas respeita o limite imposto, de forma que para manter o mesmo nível

de aproximação, à medida que as células se distanciam do átomo com quem interagem, elas ficam maiores.

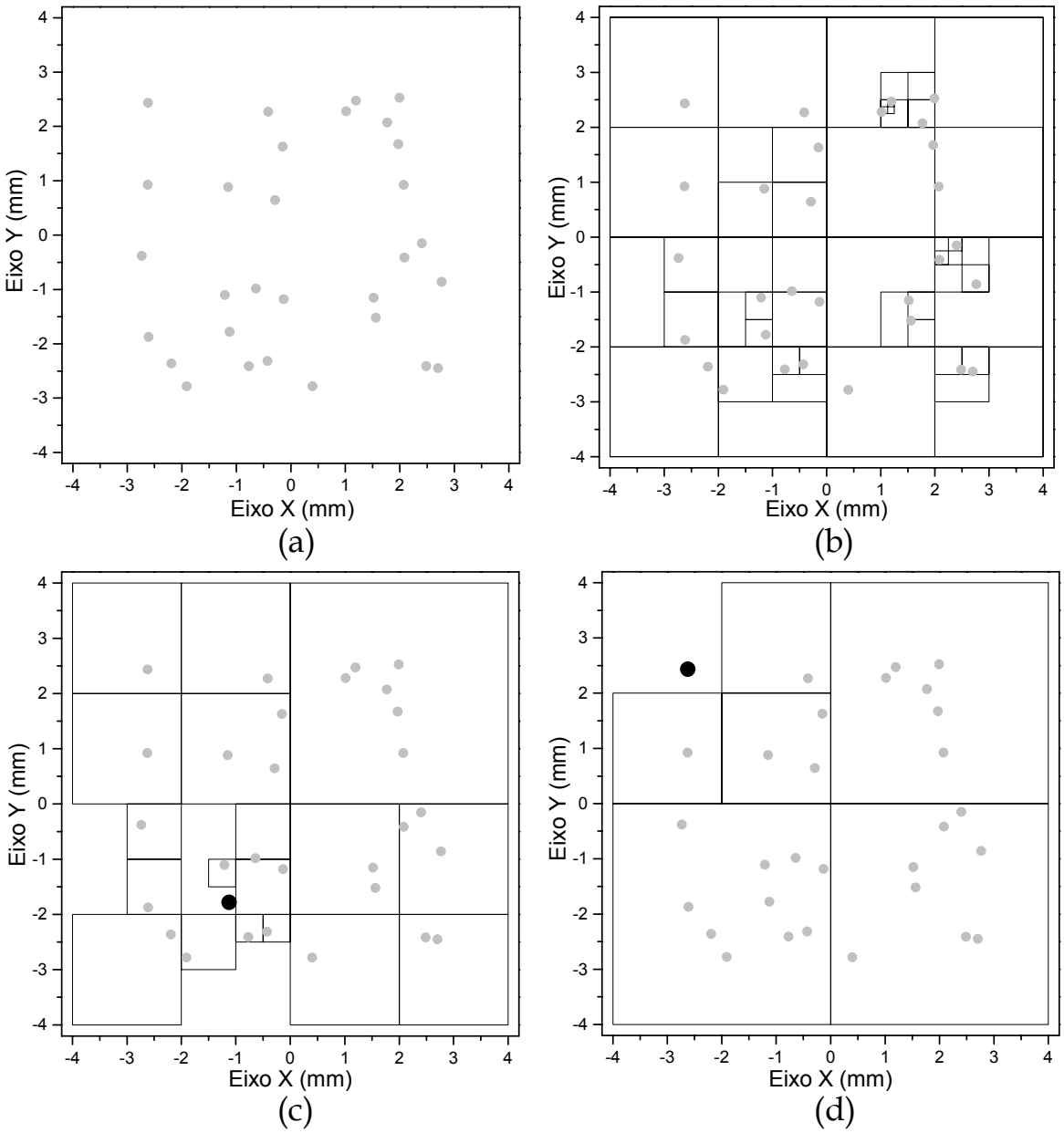


Figura 3.6 – (a) distribuição inicial, (b) árvore construída, (c) e (d) átomo destacado e as células com quem este interage, para $\theta = 1$.

Em contraste, no método de campo médio tradicional, define-se uma grade para a interação com todas as células do mesmo tamanho. O cálculo da interação é feito de forma exata para átomos na mesma célula e em células imediatamente vizinhas, enquanto nas demais a aproximação consiste em considerar todas os átomos contidos nessa célula como um único átomo equivalente. Nesse caso a aproximação é melhor à medida que a célula não vizinha está cada vez mais distante, enquanto para segundos vizinhos a aproximação é mais grosseira.

3.2.3 – Evolução Dinâmica do Sistema

Usamos a equação (2.30) para calcular a força de interação e acrescentamos a esta a força de aprisionamento para obter a força total que atua sobre cada átomo. Consideramos que esta seja constante durante o deslocamento dos átomos num intervalo de tempo *dt*. Dessa forma, podemos calcular a nova velocidade e posição do átomo, após cada intervalo.

Um ponto importante em qualquer trabalho que utilize iteração numérica para solucionar Equações Diferenciais Ordinárias é o tipo de

passo utilizado para a evolução do sistema. Utilizaremos o passo tipo Runge-Kutta. Este consiste em calcular o valor de uma função *F(x)* (derivável até a ordem p no ponto x_0) no ponto x_{n+1} através da expansão em série de Taylor da função *F(x)* em torno do ponto anterior, x_n.

$$F(x) = F(x_0) + \frac{(x - x_0)}{1!} F'(x_0) + \ldots + \frac{(x - x_0)^n}{n!} F^{(n)}(x_0) \qquad (3.5)$$

Tomaremos a expansão até quarta ordem com base nas referências [35] e [36]. Um passo mais simples, conhecido como passo Euler, consiste na aproximação em primeira ordem da equação 3.5. A referência [36] nos fornece uma função lógica para efetuar o passo de Runge-Kutta de quarta ordem na mesma linguagem de programação que usamos por todo esse trabalho (linguagem C), podendo ser anexada diretamente ao nosso código fonte.

O tamanho do passo, *dt*, foi escolhido após vários testes onde executamos o programa com poucos átomos (uma dezena) na armadilha ainda sem considerar interação, sujeitos somente às forças de armadilhamento - equações 2.31 e 2.32. Para isso, executamos o programa com um conjunto de parâmetros (*b*-gradiente de campo, Γ-largura de linha, *s*-desalinhamento, *w*-cintura do feixe) obtidos da referência [10] que, para baixas densidades, nos dá uma configuração do tipo anéis simples, e comparamos o raio obtido para o anel conforme diminuímos o tamanho do passo *dt*, até que o raio fique constante. Sabemos que o valor de *dt* precisa

ser suficientemente grande para permitir o menor número de iterações de modo a viabilizar a simulação computacional, mas ao mesmo tempo, deve ser suficientemente pequeno para não interferir nos resultados. Na figura 3.7 comparamos a convergência dos dois métodos de passo: Runge-Kutta e Euler.

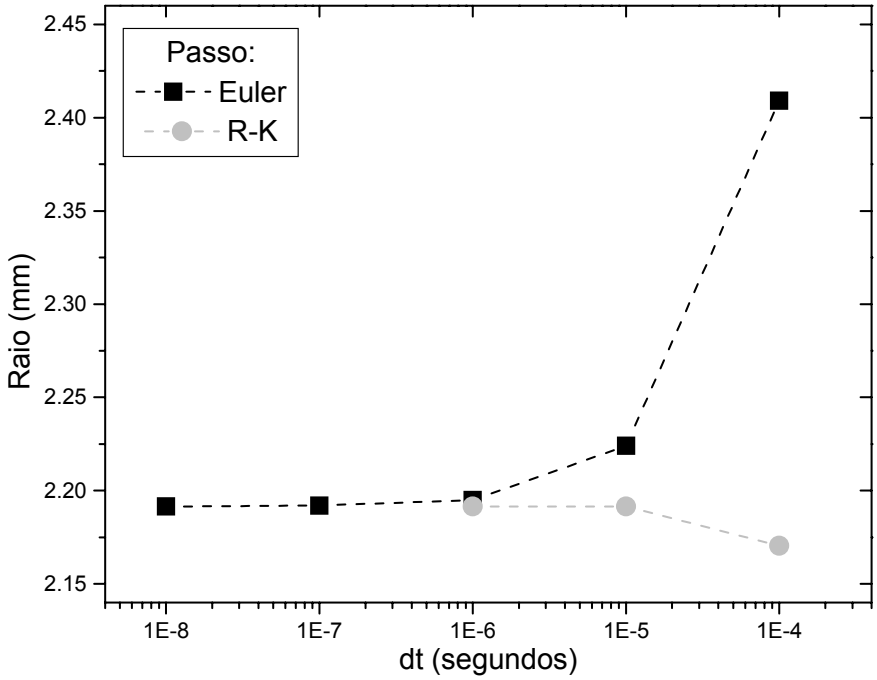


Figura 3.7 – Valores do raio da trajetória dos átomos em função do intervalo de tempo escolhido para a iteração numérica com Passo Runge-Kutta e Euler. Parâmetros usados: b = 0,23G/mm, Ω_0 = 13Γ, δ = -1Γ, ω = 5mm, s = 0,8*ω.

Note que com o passo Runge-Kutta os átomos convergem mais rapidamente para um raio, a partir do qual, mesmo que diminuamos o

intervalo de tempo *dt*, este não muda mais. O maior valor de *dt* obtido para o qual o raio se mantém dentro do erro de leitura (~0,001mm) foi *dt* = $3{,}2 \times 10^{-5}$ segundos.

Interessa-nos acompanhar a evolução do sistema até que este se estabilize. Definimos esta estabilidade como a situação em que a distribuição radial dos átomos permanece constante no tempo. Assim, iniciamos o programa com os átomos lançados aleatoriamente numa região do espaço e permitimos que eles evoluam no tempo sob a ação das forças presentes no meio, usando o *dt* acima. A cada passo medimos a distribuição radial até observarmos que esta praticamente não mais se altera com o passar do tempo. Verificamos que isso acontece após um tempo físico de aproximadamente 200 *milissegundos*. Esse tempo físico é definido pelo produto do tamanho do passo, *dt*, e o número de passos dados. Nas simulações realizadas, esperamos mais 200 *milissegundos* para garantirmos que o sistema está estabilizado.

3.3 – Estimativa do Tempo de Máquina no Algoritmo da Árvore Hierárquica

A dedução do tempo computacional gasto pelo algoritmo usando estrutura hierárquica foi feita com base na referência [42] em cima de

considerações de tempo máximo para esse método, devidamente apresentadas no decorrer da dedução.

Comecemos por considerar a figura 3.8 que mostra a subdivisão virtual do espaço para uma coleção de três partículas.

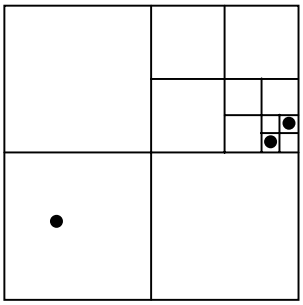


Figura 3.8 – Subdivisão do espaço para uma coleção de três partículas em duas dimensões.

Para esse exemplo, fica claro que um grande número de subdivisões pode ser necessário para separar partículas que estão muito próximas uma da outra. Sendo N o número de partículas no sistema e sendo d a menor distância entre duas partículas, requeremos que $d > 0$ para evitar força de interação infinita. Considere D como sendo o comprimento do lado de uma célula que contém todas as partículas. Claramente, o caso de maior número de níveis para a árvore, dá o pior caminho necessário para separar duas partículas que estão muito próximas uma da outra. O tamanho da menor célula que pode conter duas partículas separadas de uma distância d, em duas dimensões é $d/2^{1/2}$ (partículas nos vértices

opostos de uma célula quadrada). O caminho para separar essas duas partículas pode conter uma subdivisão em células de comprimento menor que $d/2^{1/2}$. Para cada divisão associamos um nó. Desde que a cada subdivisão a célula seja dividida ao meio, o número máximo de nós é dado pelo menor k para o qual

$$\frac{D}{2^k} \leq \frac{d}{2^{1/2}} \Rightarrow k = \log\left(\frac{2^{1/2} D}{d}\right). \tag{3.6}$$

Nesse caso o número de nós para separar duas partículas próximas é da ordem de $\log(D/d)$. Dessa forma, para separar N partículas, o número de nós será da ordem de $N \log(D/d)$.

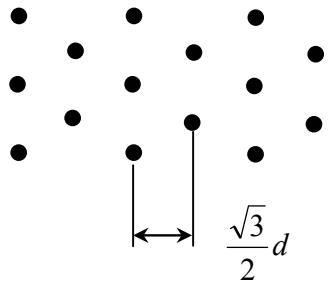


Figura 3.9 – A configuração que minimiza a razão D/d que ocorre quando todas as partículas estão separadas de uma distância d.

Para minimizar a razão D/d, para um número N fixo, todas as partículas deverão estar a uma distância d das suas primeiras vizinhas. A figura 3.9 mostra que a configuração que minimiza a razão D/d, para um

N fixo em duas dimensões, é tal que cada partícula tem seis vizinhos situados a uma distância d.

Observando a configuração mostrada na figura 3.9, grosseiramente temos que o número de partículas que podem ser acomodadas na direção vertical é D/d e na direção horizontal é $2D/3^{1/2}d$, logo

$$N = \frac{D}{d}\frac{2D}{\sqrt{3}d} \Rightarrow \frac{D}{d} = cN^{1/2}, \tag{3.7}$$

onde $c \approx 1$.

No pior dos casos, o número de nós percorridos para compor a interação entre duas partículas é da ordem de $\log N$, assim, para N partículas, o número de nós será da ordem de $N \log N$. Como o tempo computacional cresce com o número de operações necessárias para compor a interação, temos então que o tempo de máquina cresce com o número de nós, ou seja

$$TM \propto N \log N. \tag{3.8}$$

3.4 - Especificações Técnicas da Programação

Todos os programas cujos resultados são apresentados nessa tese, foram executados num computador com sistema operacional Linux. A velocidade de processamento é de 1.2 GHz.

Para a construção do programa de computador que simula a evolução temporal de um conjunto de N átomos, usamos linguagem de programação C. A escolha dessa linguagem ocorreu devido à facilidade para gerenciar quase que diretamente a memória da máquina. A cada passo é necessário construir uma nova árvore e a antiga precisa ser apagada da memória para dar espaço à nova. Definimos células (as folhas da árvore) como sendo variáveis do tipo estrutura (struct). As ligações entre estruturas, representadas na figura (3.5) por setas, são ponteiros: um tipo de variável que guarda um endereço previamente definido (inteiro, ponto flutuante, estrutura, etc.). Os ponteiros foram usados, na construção da árvore, para ligar células localizadas em níveis imediatamente inferiores.

Da mesma forma que as células, os átomos também são definidos como variáveis do tipo estrutura, e estão ligados entre si por ponteiros, sendo que para os átomos, não há hierarquia entre eles. A cadeia de átomos está inicialmente ligada à célula mãe.

A seqüência dos processos para cada passo é a seguinte:

1) Dividir virtualmente o espaço;

2) Construir uma cadeia hierárquica usando todos os passos para a divisão virtual do espaço;

3) Calcular a força sobre cada átomo devido à armadilha e à presença dos demais átomos, levando em conta a precisão θ;

4) Calcular a nova velocidade para um acréscimo tempo dt e a nova posição usando essa nova velocidade;

5) Organizar o sistema limpando espaço de memória para reutilização.

6) Retorna para o passo 1.

Usando os passos descritos acima, construímos um fluxograma da simulação:

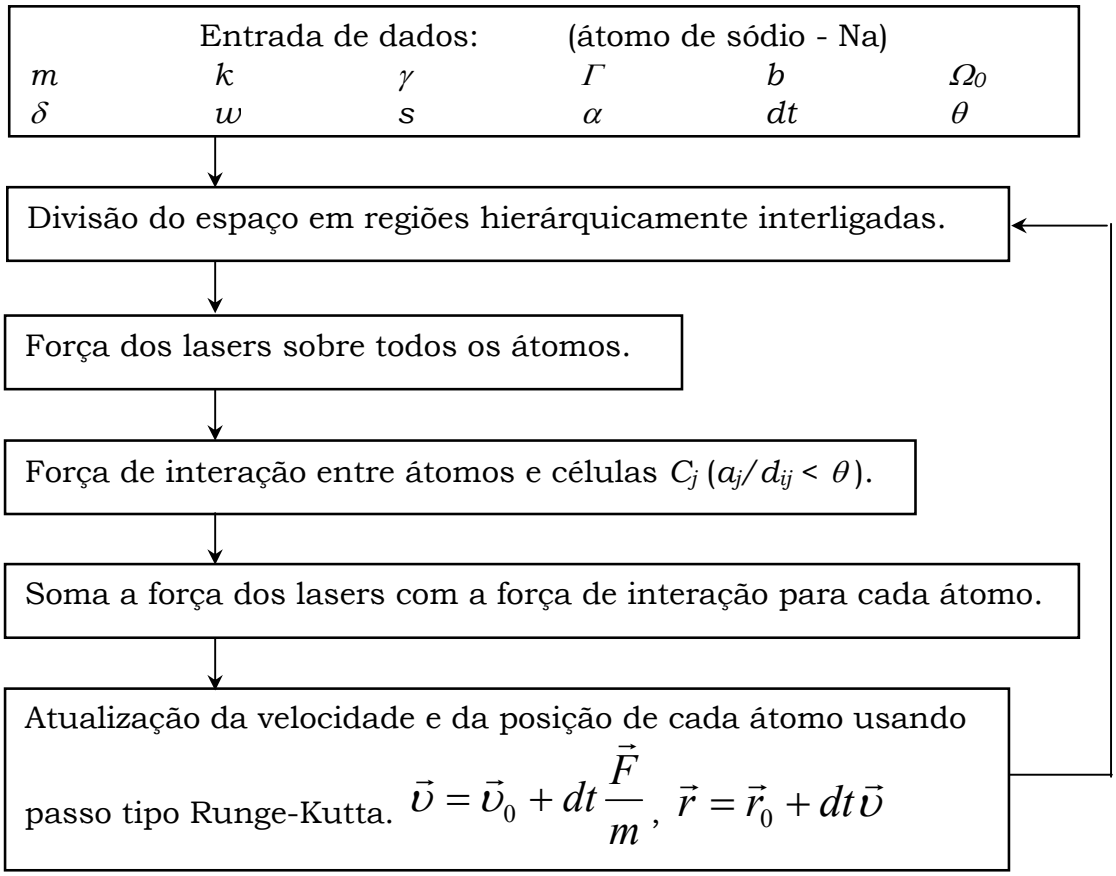


Figura 3.10 – Fluxograma simplificado da simulação numérica segundo o método aproximativo da árvore hierárquica.

Para efeito de comparação, apresentamos aqui o fluxograma simplificado para o cálculo da interação usando o método direto, que é comumente referenciado na literatura como: soma direta, ou partícula-partícula.

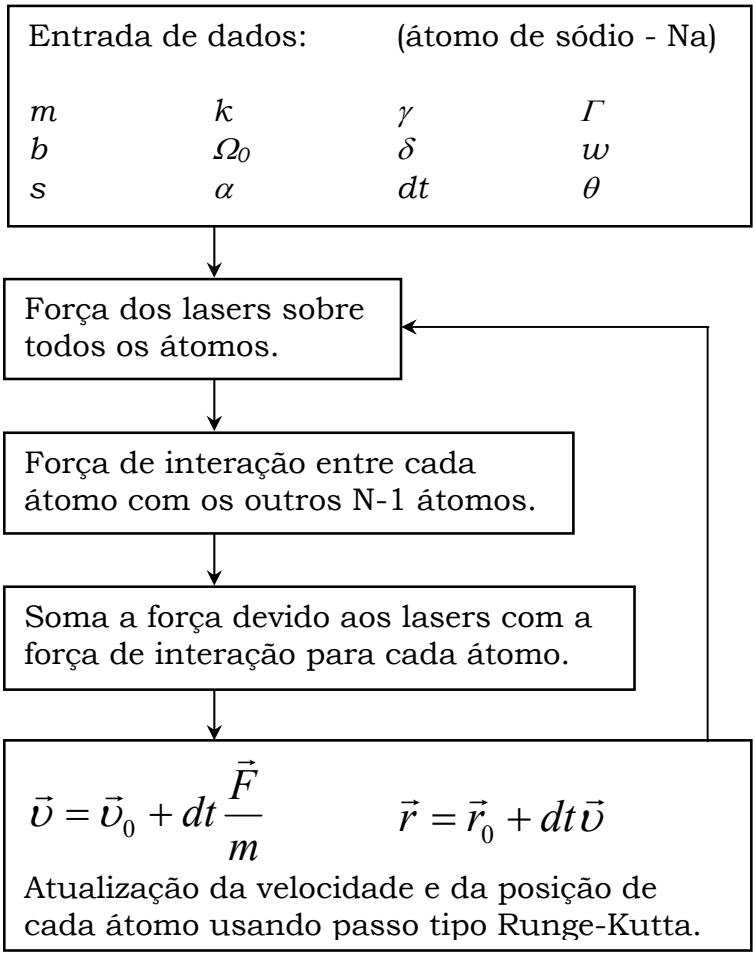


Figura 3.11 – Fluxograma para cálculo da interação segundo o método da soma direta.

Os resultados obtidos usando o método aproximativo descrito aqui serão apresentados no capítulo seguinte.

Capítulo IV

Análise dos Resultados

Neste capítulo apresentamos os resultados obtidos através da execução dos programas construídos para simular a evolução temporal de N átomos numa armadilha magneto-ótica, sujeitos à força dos lasers que a constituem, bem como à interação entre eles provocada pelo múltiplo espalhamento de fótons. Começamos a discussão com o método exato, ou direto, chegando até o seu limite prático ($N = 10^4$ átomos), a partir do qual o tempo de execução excede a escala de meses, o que inviabiliza sua execução. Atingindo o limite do método direto, iniciamos a apresentação dos resultados obtidos através do método árvore hierárquica, com o qual evoluímos até $N = 10^6$ átomos, duas ordens de grandeza acima do limite para o método exato. Esse limite para o cálculo aproximado é justificado na seção 4.2. Nas simulações cujos resultados são apresentados aqui são usados os parâmetros da tabela 2.1, correspondentes ao átomo de sódio. Os valores utilizados para: o gradiente de campo magnético, b; a cintura do laser, w; o desalinhamento dos lasers, s; a freqüência de Rabi, Ωo e o

detuning, δ; são apresentados na tabela 4.1 e correspondem à condição experimental da observação de dois anéis.

Parâmetros da armadilha	
B	0,5 G/mm
w	3 mm
s	3 mm
Ω_0	4Γ
δ	-2Γ

Tabela 4.1 – Parâmetros que caracterizam a armadilha, usados nas simulações.

4.1 – Método Exato ou Soma Direta

Com os feixes de laser da armadilha alinhados, temos somente um ponto de equilíbrio, onde a força resultante é nula, no centro. Com um desalinhamento imposto, mudamos de um único ponto de equilíbrio para uma região aproximadamente circular. Esse é o motivo pelo qual pode-se explicar o surgimento de nuvens atômicas na forma de anel mesmo em regime de baixas densidades.

A força de aprisionamento induz um movimento aproximadamente circular. A região de equilíbrio tem um raio ligeiramente maior sobre os

eixos *x* e *y* que o raio obtido na direção que forma um ângulo de 45º com esses eixos. A diferença é de aproximadamente 0,01 milímetros e não chega a ser percebida para distribuições no plano, como veremos adiante. Como o comportamento no alargamento das distribuições é o mesmo nas duas direções, apresentaremos os resultados obtidos somente sobre o eixo *x*. As distribuições são tomadas varrendo a armadilha a partir do centro contabilizando o número de átomos por centímetro quadrado. Cada átomo cujo raio R esteja entre r e r + Δr será considerado à mesma distância do centro, onde $\Delta r = \delta a / r\pi$. Consideramos que δa é a resolução com a qual varremos a armadilha. Quando não for explicitado, o valor de δa será sempre igual a $4{,}22 \times 10^{-3}$ mm^2.

Como foi explicado no capítulo II, as simulações foram feitas somente em duas dimensões usando as equações 2.31 e 2.32 para as forças da armadilha e a equação 2.33 para a força de interação. Começamos executando o programa para a simulação do método exato com apenas uma centena de átomos e com isso notamos apenas a acomodação destes num raio preferencial, o qual para os parâmetros utilizados, corresponde a $r = 1{,}344 \pm 0{,}001$ mm. Com esse número de átomos a interação não apresenta nenhum efeito que possa ser observado em distribuições radiais, pois toda a distribuição está contida dentro do mesmo elemento δa de resolução.

Evoluímos aumentando o número de átomos de uma ordem de grandeza (para N = 10^3 átomos), obtendo a distribuição mostrada na figura 4.1.

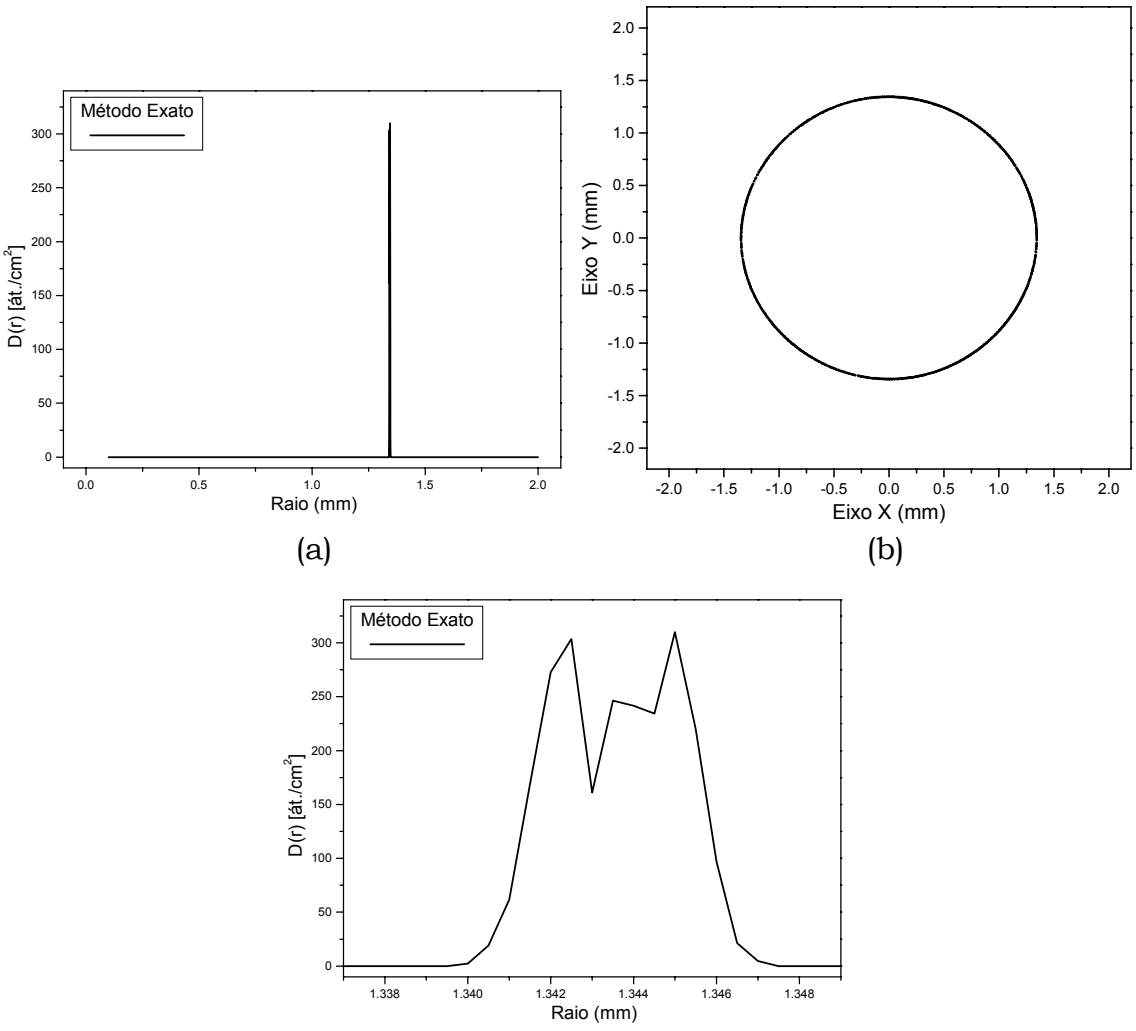


Figura 4.1 – (a) Distribuição radial tomando toda a extensão da armadilha (b) Distribuição no plano, (c) Ampliação em torno da região de equilíbrio. Resultados obtidos com o método exato para N = 10^3 átomos.

Na figura 4.1a, observamos que os átomos se concentram numa região em torno da posição de equilíbrio da força de aprisionamento. Na figura 4.1b temos a distribuição no plano e na figura 4.1c ampliamos a região em torno do equilíbrio e vemos o alargamento da distribuição provocado pela força de interação entre os átomos. Embora os efeitos da interação sejam pequenos, eles já se fazem notar. O eixo das ordenadas está identificado por D(r) que quer dizer distribuição radial onde tomamos o número de átomos por unidade de área.

O tempo gasto para execução do programa com método exato e N = 10^3 átomos foi de 4601 segundos (1h 16' 41") e a largura à altura média da distribuição é 0,0041 milímetros. Como comentamos na seção 3.4, o tempo de execução depende da máquina utilizada e foi medido usando sempre o mesmo computador cuja velocidade é de 1.2 GHz.

Aumentando o valor de N em uma ordem de grandeza (N = 10^4 átomos), obtemos a distribuição apresentada na figura 4.2. Podemos notar basicamente duas diferenças importantes em relação à distribuição para N = 10^3 átomos. Uma delas é o aumento no alargamento da distribuição, o que era esperado, pois quando aumentamos o número de átomos na armadilha, o efeito da força de interação aumenta. Outro ponto a se destacar é uma notória estrutura de picos vista no detalhe à direita (figura 4.3b). Discutiremos essa estrutura de picos na próxima seção.

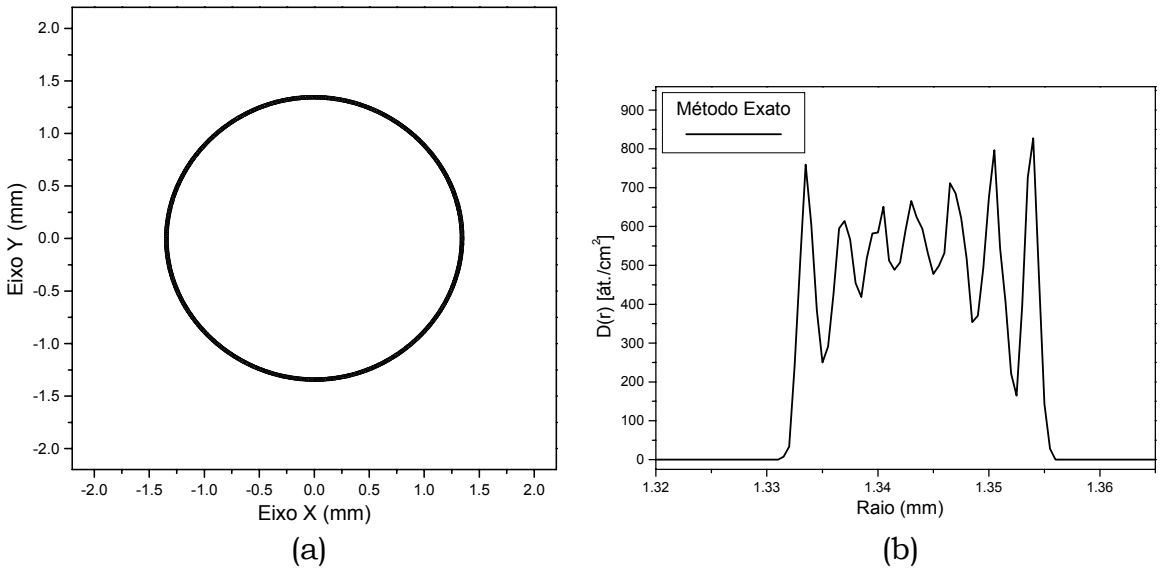


Figura 4.2 – (a) Método Exato com N = 10^4 átomos.
(b) Ampliação da região em torno do equilíbrio.

O tempo gasto para execução do programa com método exato e N = 10^4 átomos foi de 289.827 segundos (80h 30' 27") e a largura à altura média da distribuição é 0,0225 milímetros.

Simulações do método exato estão limitadas a valores de N próximos de 10^4 átomos. Nesse momento o tempo de execução já é suficientemente grande para impedir novas incursões a ordens superiores visto o acentuado crescimento do tempo de máquina

A figura 4.3 ilustra o crescimento do tempo de simulação da armadilha para valores crescentes em ordens de grandeza de N. Esperávamos que o tempo de máquina crescesse com N^2, o que verificamos

é que o tempo de máquina cresce com $N^{1,75}$, um valor ligeiramente inferior ao previsto.

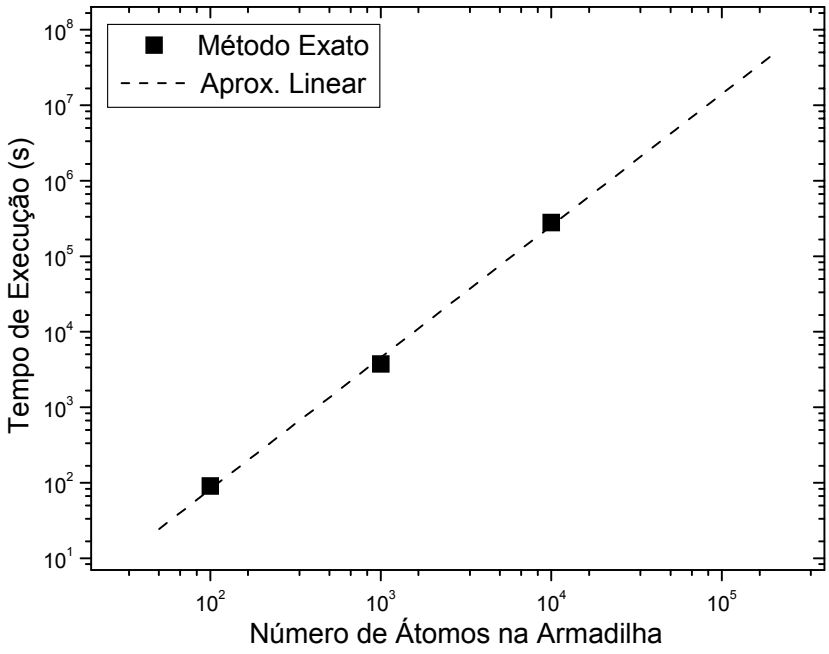


Figura 4.3 – Crescimento do tempo de máquina com o número de átomos na armadilha.

Quando calculamos o tempo de máquina para o método da soma direta, consideramos a força de interação sobre cada par de átomos *i* e *j* como sendo dada por:

$$\vec{F}_{ij} = \alpha \frac{\left(\vec{r}_i - \vec{r}_j\right)}{\left|\vec{r}_i - \vec{r}_j\right|^3} . \qquad (4.1)$$

Considerando que $\vec{F}_{ij} = -\vec{F}_{ji}$ e $\vec{F}_{ii} = 0$, temos para o tempo de máquina:

$$TM_{SD} \propto N(\frac{N}{2} - 1), \tag{4.2}$$

que para N grande fica: $TM_{SD} \propto N^2$.

O tempo previsto para a simulação com N = 10^5 átomos é de aproximadamente 163 dias o que inviabiliza a sua execução. Interessados em aumentar o valor de N, mas impossibilitados pelo acentuado crescimento do tempo de máquina, implementamos e passamos a usar o método aproximativo da árvore hierárquica, descrito no capítulo anterior.

4.2 – Método da Árvore Hierárquica

Começamos executando o programa para a simulação com esse método aproximativo também com apenas uma centena de átomos. Novamente, notamos apenas a acomodação destes num raio preferencial. Evoluímos então aumentando o número de átomos de uma ordem de grandeza (para N = 10^3 átomos) e comparamos as distribuições obtidas para o método exato e aproximado, como mostra a figura 4.4. Com esse número de átomos, faremos a comparação apenas na região em torno do equilíbrio.

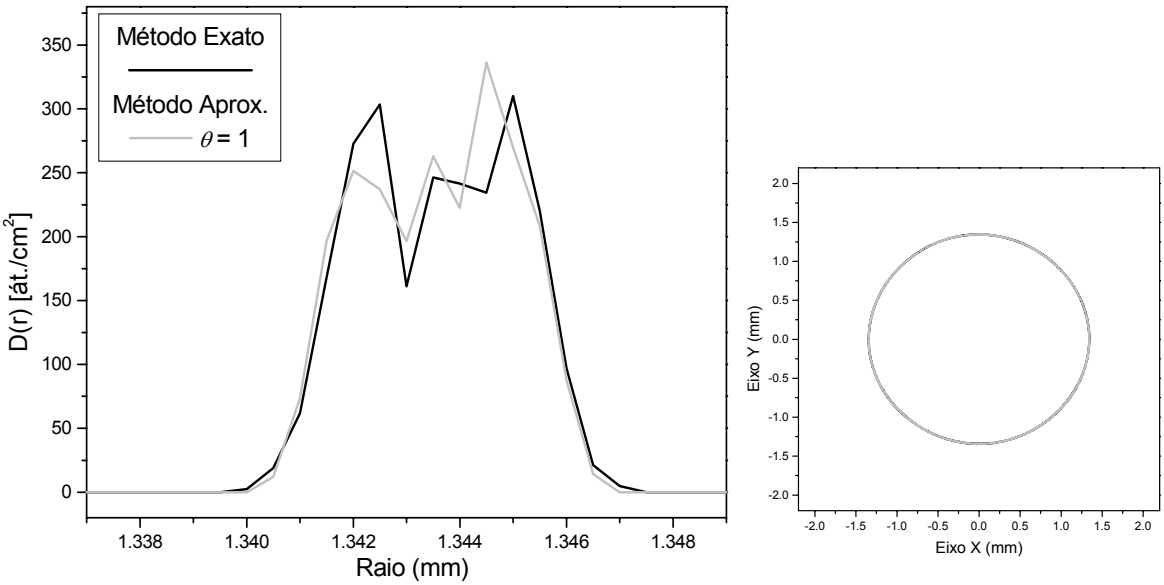


Figura 4.4 – Comparação para N = 10^3 átomos entre método exato e aproximado com $\theta = 1$. (a) Distribuições em torno do equilíbrio e (b) sobrepostas no plano *xy*.

O valor do parâmetro de precisão utilizado foi $\theta = 1$. Para esse valor de θ temos que o tempo de máquina obtido para o método aproximado foi de 2313 segundos (38' 33") e a largura obtida foi a mesma para os dois casos (0,0041 milímetros), de forma que, no plano (figura 4.4b), as distribuições se sobrepõem.

No capítulo anterior, quando descrevíamos o processo de cálculo da interação usando a estrutura hierárquica, definimos o parâmetro de precisão, θ, discutimos seus limites e ilustramos uma solução para um

valor intermediário que seria θ igual à unidade. Na referência [38], de onde veio o método aproximativo da árvore hierárquica, os autores alegam que o valor de θ =1 permite um erro de apenas 1% na conservação de energia. O sistema estudado por eles é constituído de partículas interagindo entre si segundo forças puramente gravitacionais. Esse sistema permite avaliar o quanto a energia se conserva, porque não há troca de energia com o meio externo. No caso em que estamos investigando, não podemos usar o mesmo critério para avaliar o erro que podemos vir a obter devido ao parâmetro de precisão escolhido, uma vez que no nosso sistema há uma troca constante de energia entre a luz dos lasers e os átomos presentes na armadilha. Avaliaremos o erro na aproximação através das diferenças na largura à altura média das distribuições obtidas pela execução usando o método exato.

Na figura 4.5 apresentamos, para comparação, as distribuições obtidas para N = 10^4 átomos, pelo método exato e aproximado, com θ = 1. A diferença na largura das distribuições acima ocorre devido ao grau de precisão adotado no cálculo da interação de cada átomo com os demais presentes na armadilha. A largura da distribuição para o método aproximado é de 0,022 milímetros, o que dá uma diferença de 2,3% e o tempo de execução foi de 21.738 segundos o que equivale a 6 horas, 2 minutos e 18 segundos.

A diferença na largura à altura média, obtida entre as distribuições na figura 4.5, é menor que duas vezes o valor de $\Delta r = \delta a / r\pi$ para raios próximos do raio de equilíbrio e está dentro do erro inserido quando dizemos que um valor de raio entre r e r + Δr é considerado o mesmo. Notamos nas duas últimas comparações o bom acordo entre as distribuições obtidas para N = 10^3 e 10^4 átomos com $\theta = 1$.

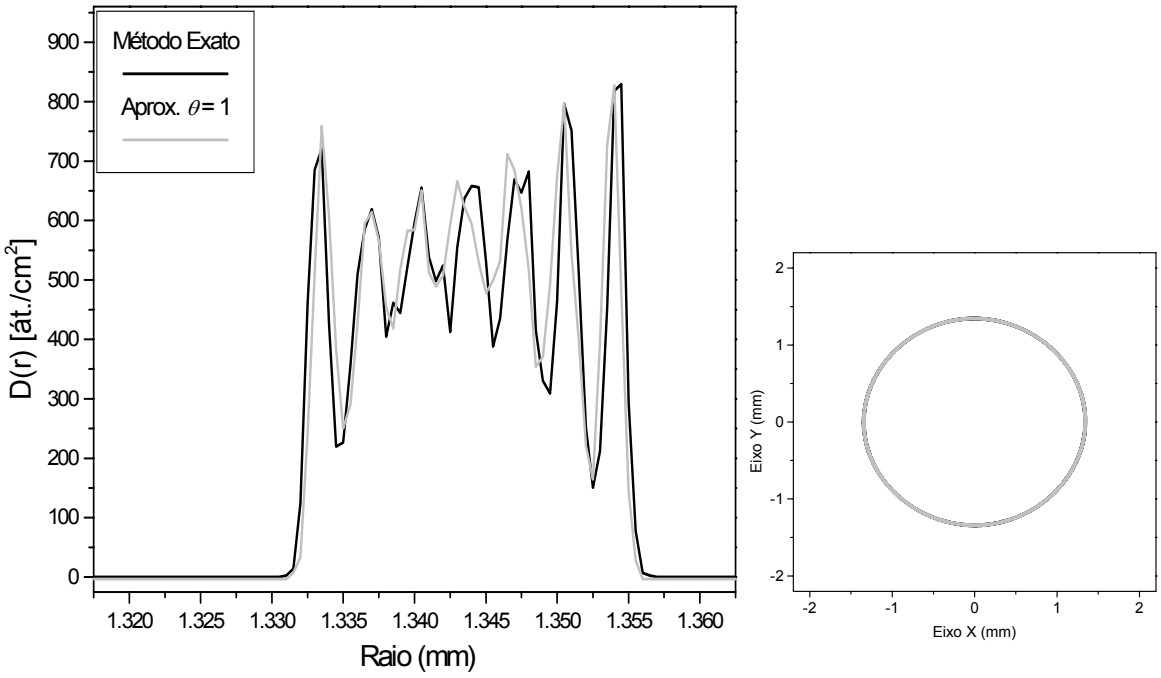


Figura 4.5 – Método Exato e Aproximado com $\theta = 1$ e N = 10^4 átomos. (a) Ampliação na região de equilíbrio (b) distribuição no plano.

Para efeito de comparação, apresentamos na figura 4.6 duas outras situações para outros valores do parâmetro de precisão de θ. Sabemos que

quanto menor for o valor de θ, maior é a precisão no cálculo da interação e a figura 4.6 comprova esse comportamento. Para θ = 0,5 e N = 10^4 átomos as distribuições têm a mesma largura e o tempo de execução no método aproximado é de 41.347 segundos (11h, 29' e 7"). Para θ = 2 N = 10^4 átomos as distribuições diferem em 20% e o tempo de execução cai para 11.085 segundos (3h, 4' e 45"). Comparando os tempos de execução e o erro obtido para variados valores do parâmetro de precisão, optamos por usar θ = 1.

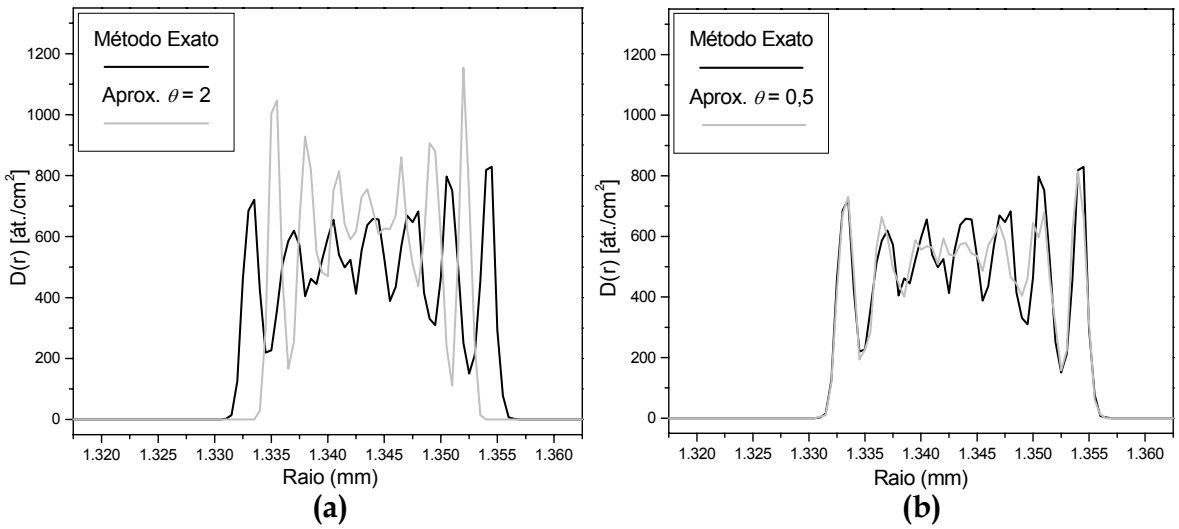


Figura 4.6 – Método exato e aproximado para N = 10^4 átomos, com parâmetros de precisão distintos (a) θ = 2 e (b) θ = 0,5.

Também implementamos e executamos o programa em três dimensões. Para isso foi preciso alterar o código fonte do programa usado

nas simulações em duas dimensões. As modificações feitas foram basicamente: a) mudança das estruturas de quadrados para cubos; b) o número máximo de células filhas passa de quatro para oito; c) as equações da força de aprisionamento usadas são dadas por (2.20)-(2.22). Na figura 4.7 comparamos as distribuições dos átomos obtidas com N = 10^4 átomos e $\theta = 1$ para duas e três dimensões.

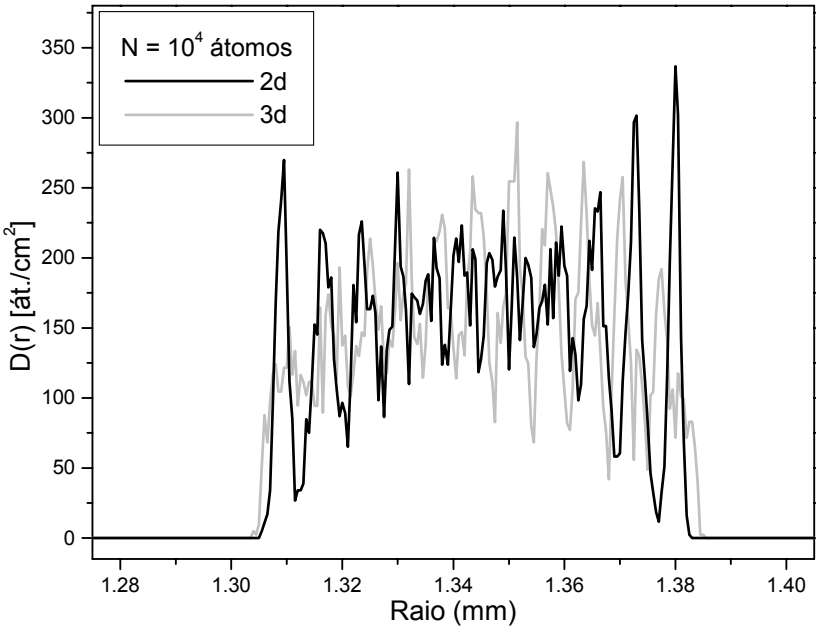


Figura 4.7 – Distribuições com N = 10^4 átomos obtidas para duas e três dimensões com $\theta = 1$.

Note que largura de ocupação é praticamente a mesma, diferindo de aproximadamente 4%. Essa diferença não chega a ser notada em distribuições no plano como nos gráficos 4.2a e 4.1b. O tempo de execução

foi de 43.521 segundos (12 h, 5’ e 21”) o que equivale a aproximadamente o dobro do tempo para a simulação em duas dimensões.

Antes de obtermos as primeiras distribuições, esperávamos que o alargamento provocado pelo aumento no número de átomos na armadilha resultasse em distribuições na forma gaussiana, em vez disso, obtivemos distribuições com crescimento brusco e uma estrutura interna de picos na forma de serra.

A estrutura de picos pode ser entendida considerando um preenchimento gradativo da armadilha. Dessa forma temos que à medida que os átomos vão sendo capturados, eles são conduzidos para o raio de equilíbrio até que a partir de um certo valor de N (superior a 100 átomos), uma vez que a força de interação não permite mais a acomodação dos próximos átomos aprisionados no mesmo raio, esses últimos átomos buscam uma nova posição de equilíbrio. A partir daí temos que se inicia o alargamento da parede da estrutura na forma de anel. A força de aprisionamento atua comprimindo essa parede enquanto a força de interação atua impondo uma distância mínima entre os átomos. A competição entre essas forças cria um arranjo interno aproximadamente regular. A figura 4.8 mostra um detalhe de uma seção do anel para N = 10^5 átomos onde se pode notar um certo alinhamento dos átomos com mesmo raio (horizontal).

Mesmo apresentando uma certa ordem quando no mesmo raio, os átomos não apresentam ordem alguma para raios distintos (vertical), devido ao fato de a velocidade angular dos átomos variar com o raio, originando um movimento relativo entre camadas de átomos em raios distintos. Esse movimento impossibilita a acomodação em uma estrutura regular na direção radial.

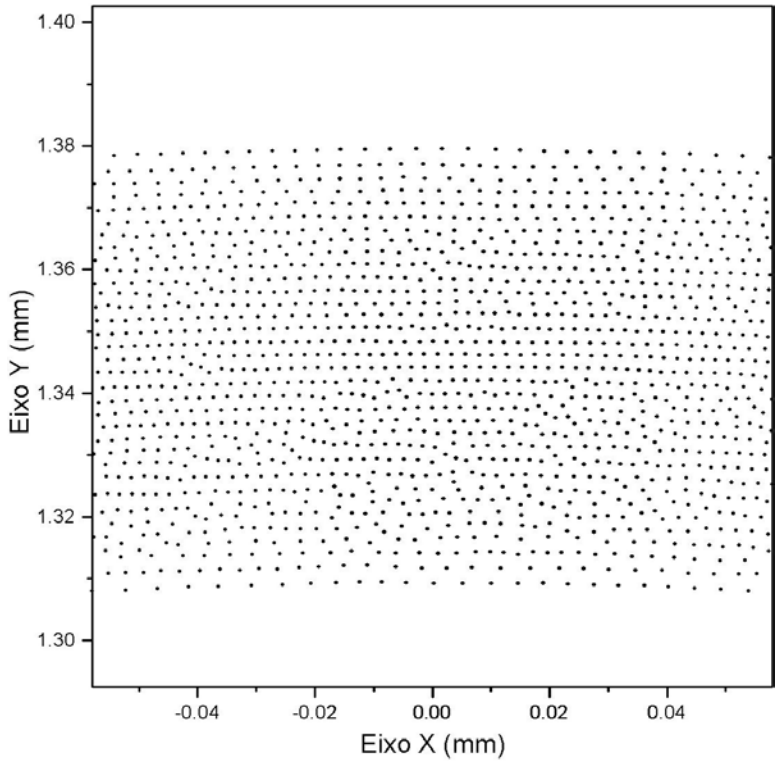


Figura 4.8 – Modo orbital na forma de anel (esquerda) e detalhe da estrutura interna (direita). Com N = 10^5 átomos e $\theta = 1$.

Cabe salientar que a estrutura de picos, que surge nas distribuições apresentadas anteriormente, ocorre devido ao fato de desprezarmos o

termo de emissão espontânea. Como a emissão espontânea não apresenta uma direção preferencial, tomamos a média do recuo num intervalo de tempo Δt como sendo nula, obtendo, então, um comportamento médio dos átomos.

As comparações entre o método exato e o método aproximado foram necessárias para que pudéssemos escolher o melhor parâmetro θ para o programa com o método aproximativo, pois o cálculo exato, como dissemos anteriormente, está limitado à execução com um conjunto de 10^4 átomos. Na figura 4.9, comparamos o tempo de máquina para os dois métodos de cálculo.

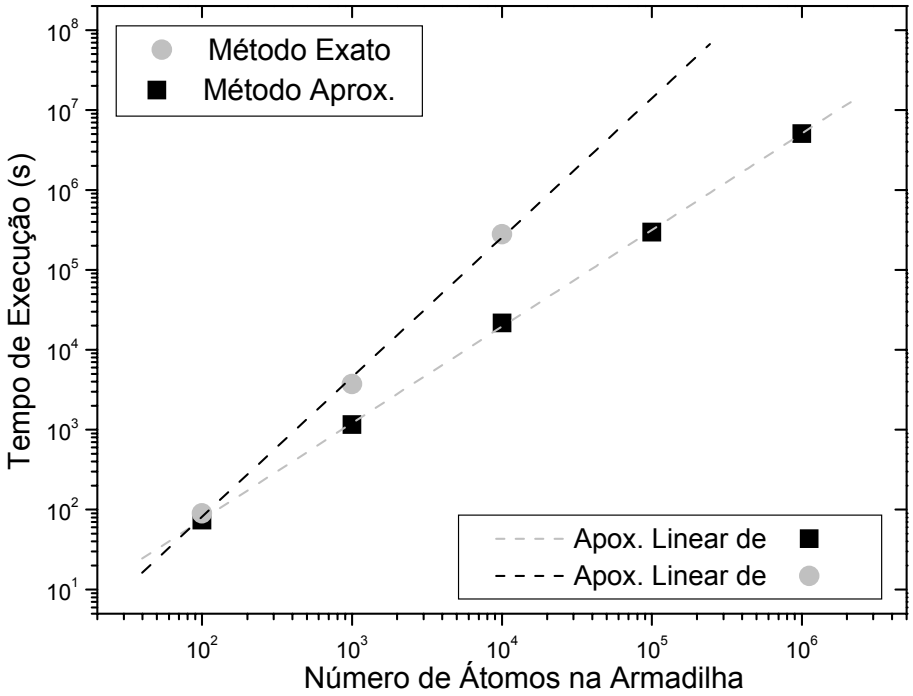


Figura 4.9 – Crescimento do Tempo de Máquina com N para os métodos exato e aproximado.

Vemos que enquanto no método exato o tempo de máquina cresce com $N^{1,75}$, como já discutido, no método aproximativo, o tempo cresce com $N^{1,2}$. Dessa forma, à medida que aumentamos o valor de N a diferença entre o tempo de execução para cada método se acentua, justificando a utilização do método aproximativo.

Devido ao crescimento do tempo de máquina com N, execuções com $N = 10^5$ átomos somente puderam ser realizadas usando o método aproximado. Seguindo a tendência do crescimento do tempo de máquina com o número de átomos na armadilha para o método exato, esperamos, por extrapolação da curva referente ao método exato na figura acima, que o tempo gasto para a execução com $N = 10^5$ átomos seja de aproximadamente $1,4x10^7$ segundos ou 163 dias como citamos anteriormente. No método aproximado para o mesmo valor de N o tempo é de $3,1 \times 10^5$ segundos ou 3,6 dias. Esses resultados numéricos dão uma excelente imagem do ganho obtido no tempo de simulação através do uso da aproximação.

O crescimento do tempo de execução dos programas com o número de átomos independe da máquina utilizada. Já o tempo para cada execução depende da velocidade do processador da máquina. Todos os resultados utilizados para o cálculo do tempo de execução foram obtidos usando um computador com velocidade de processamento de 1,2 GHz.

O crescimento do tempo computacional calculado na seção 3.4 para o método da árvore hierárquica era da ordem de $N \log N$. Na figura 4.10, comparamos o crescimento previsto com aquele obtido pelas simulações. O coeficiente de proporcionalidade usado na relação $TM = kN \log N$ para a obtenção dos valores do crescimento teórico foi imposto k = 1 segundo. Verificamos, portanto, que o modelo teórico prevê um crescimento menos acentuado que o resultado obtido pela execução do programa. Essa diferença entre os tempos para cada caso é pequena e não chega a uma ordem de grandeza quando N = 10^6 átomos.

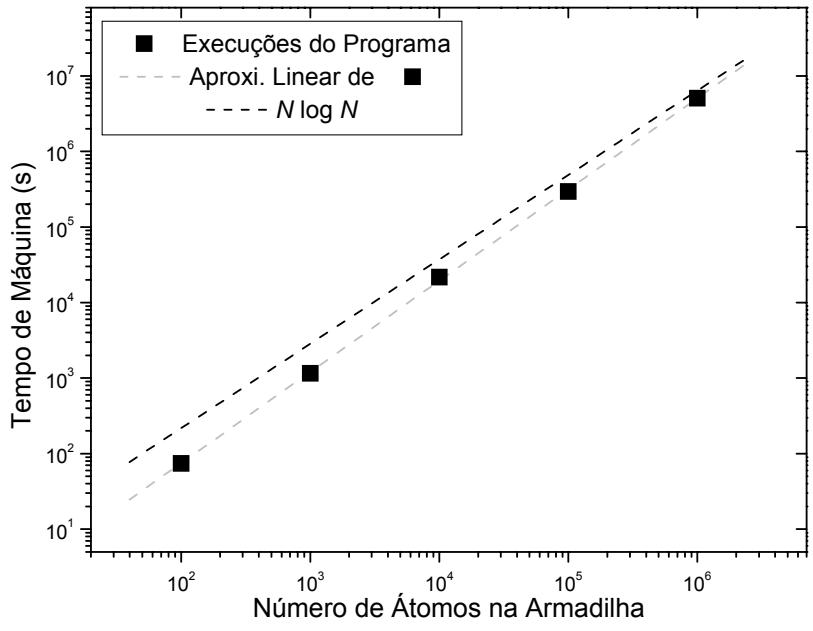


Figura 4.10 – Comparação entre o crescimento previsto pelo método da estrutura hierárquica com θ = 1 e os resultados obtidos através das simulações.

4.3 – Efeitos da Interação

Fizemos simulações da armadilha com sucessivas ordens de grandeza para o número de átomos aprisionados, desde algumas dezenas até N = 10^6 átomos. Até esse valor de N, observamos um aumento na largura de ocupação do anel, provocada pelo acréscimo de átomos na armadilha. Notamos que esse aumento na largura do anel é bem comportado para esses valores de N. Na figura 4.11 mostramos como aumenta a largura de ocupação do anel à medida que acrescentamos átomos na armadilha.

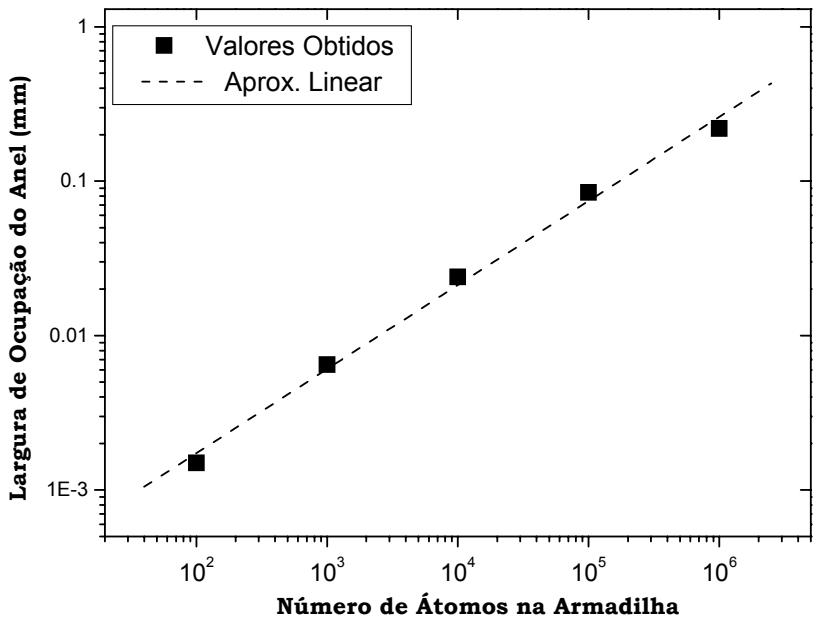


Figura 4.11 – Largura do anel em função do número de átomos na armadilha. $\theta = 1$.

Da figura 4.11 temos que, numa escala logarítmica, a largura de ocupação do anel (distribuição), para valores de N menores ou da ordem de 10^5 átomos, cresce linearmente com N, ou seja, a relação entre a largura de ocupação do anel e o número de átomos aprisionados segue a seguinte lei de potência:

$$L \propto N^{0,58} . \tag{4.3}$$

O aumento na largura de ocupação da distribuição radial ocorre devido a uma maior contribuição da força repulsiva, causada pelo acréscimo do número de átomos, ou seja, a interação torna-se mais importante com o aumento de N.

Interessados em compreender melhor os efeitos da interação, aumentamos arbitrariamente o parâmetro de interação, α, da equação 4.1 em algumas ordens de grandeza. Dando maior peso à interação, esperamos que o comportamento obtido com um certo valor de N possa ser reproduzido com um número menor de átomos na armadilha, porém com um valor alterado do parâmetro de interação. A figura 4.12 apresenta a largura do anel em função do parâmetro α para N = 10^4 átomos e θ = 1.

Notamos que aumentando o valor do parâmetro de interação, ocorre um aumento na largura de ocupação do anel. Esse aumento é linear numa escala logarítmica, o que significa dizer que segue uma lei de potência, a saber:

$$L \propto \alpha^{0,49} . \tag{4.4}$$

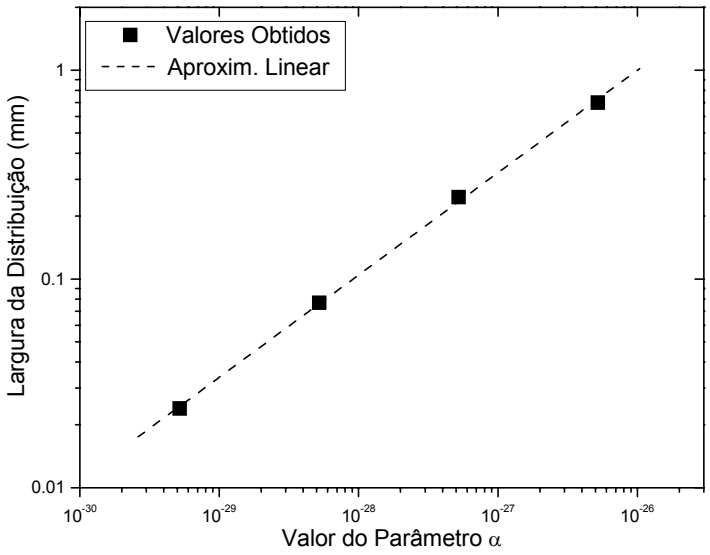


Figura 4.12 – Largura de ocupação do anel como função do valor de do parâmetro de interação α, com N = 10^4 átomos.

Devemos, entretanto, ser cautelosos com respeito ao aumento arbitrário de α com o objetivo de intensificar os efeitos das forças repulsivas entre os átomos. De fato, comparando as equações 4.3 e 4.4, vemos que aumentar o parâmetro de interação ou o número de átomos na armadilha em uma ordem de grandeza não é equivalente. Podemos, contudo, usar o resultado com α aumentando de uma ordem de grandeza para obter uma distribuição semelhante àquela obtida para N também aumentado de uma ordem de grandeza. Aproximações envolvendo valores maiores de α geram erros que comprometem os resultados e devem ser evitados.

Como exemplo, na figura 4.13 comparamos um resultado, obtido com N = 10^4 átomos e α vezes 10 com N = 10^5 átomos e α vezes 1. Estão mostradas apenas as ampliações das distribuições em torno da região de equilíbrio, pois no plano não é possível diferenciá-las. A diferença na largura é de aproximadamente 8,6%.

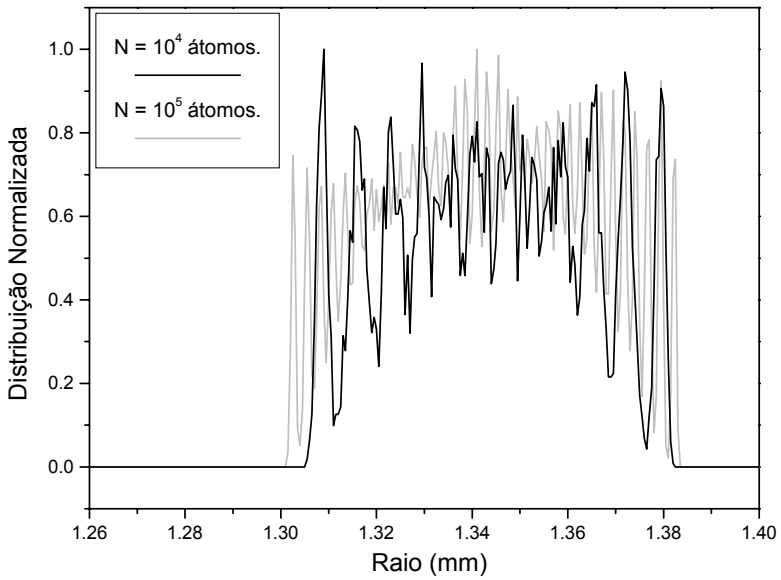
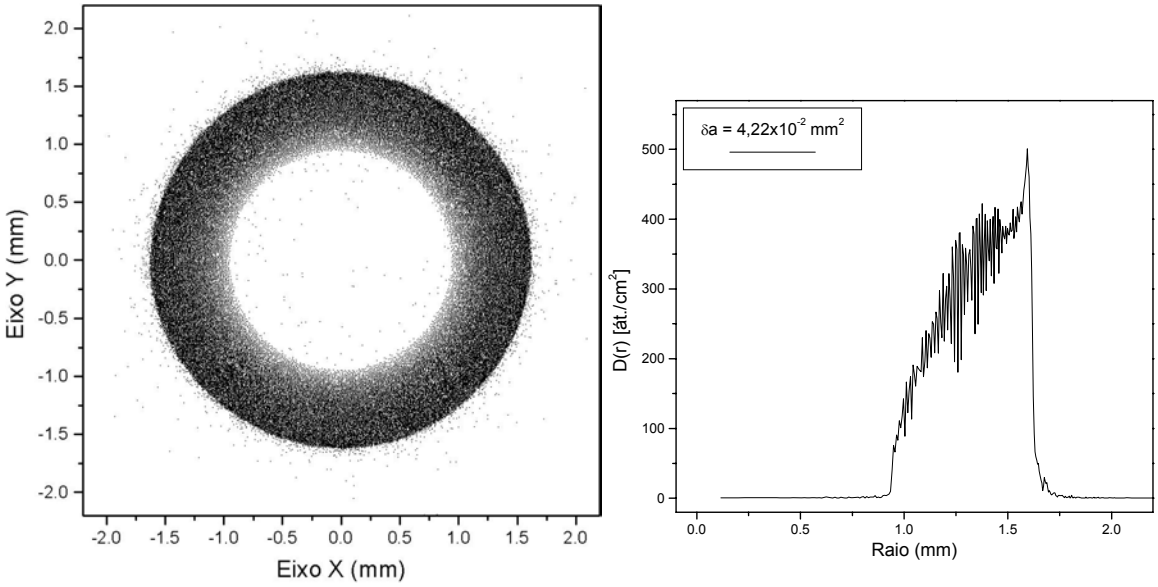


Figura 4.13 – comparando N = 10^4 átomos (α*10) com N = 10^5 átomos e α*1 e θ = 1.

Embora o método aproximativo nos dê um ganho considerável, simular um número de átomos na armadilha da ordem de um milhão requer um tempo de execução de quase dois meses. Quando mudamos em uma ordem de grandeza o valor de α para o mesmo valor de N não mudamos o tempo de execução enquanto obtemos uma distribuição semelhante àquela obtida para N uma ordem acima.

Seguindo essa estratégia de aumentar o parâmetro de interação, mostramos na figura 4.14 o resultado de uma simulação com N = 10^5 átomos e α*100, sujeita, portanto a um maior erro na comparação com possíveis resultados obtidos, para N = 10^7 átomos e α*1. Na figura 4.14b a resolução usada foi de $\delta a = 4{,}22 \times 10^{-2}$ mm^2.



(a) (b)

Figura 4.14 – (a) distribuição para N = 10^5 átomos e α*100 no plano e (b) distribuição radial para os mesmos dados.

Em contraste, na figura 4.15 a distribuição radial para N = 10^6 átomos e α*10 apresenta uma estrutura mais à direita que pode indicar o início do desdobramento da estrutura de um anel em dois anéis concêntricos. Esse mesmo efeito não pode ser observado na distribuição para N = 10^5 átomos com α*100 devido ao erro inserido nessa

aproximação, como discutido. Mostramos na figura 4.16 a distribuição no plano associada à figura 4.15.

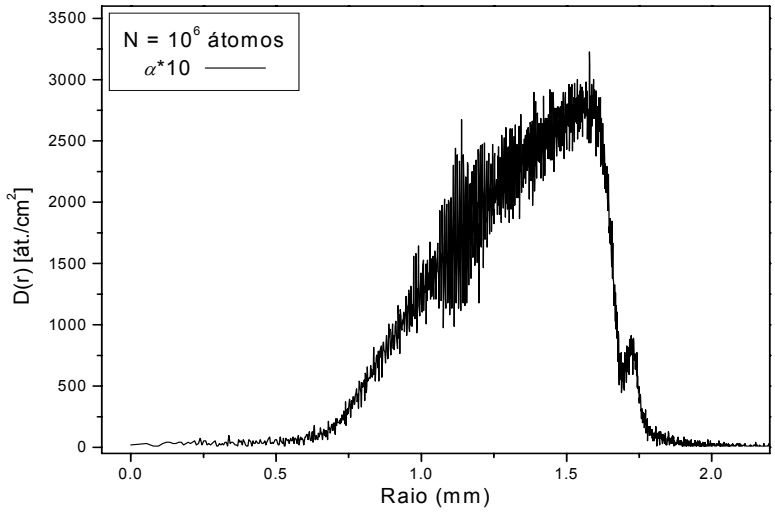


Figura 4.15 – Distribuição radial obtida com N = 10^6 átomos com α*10.

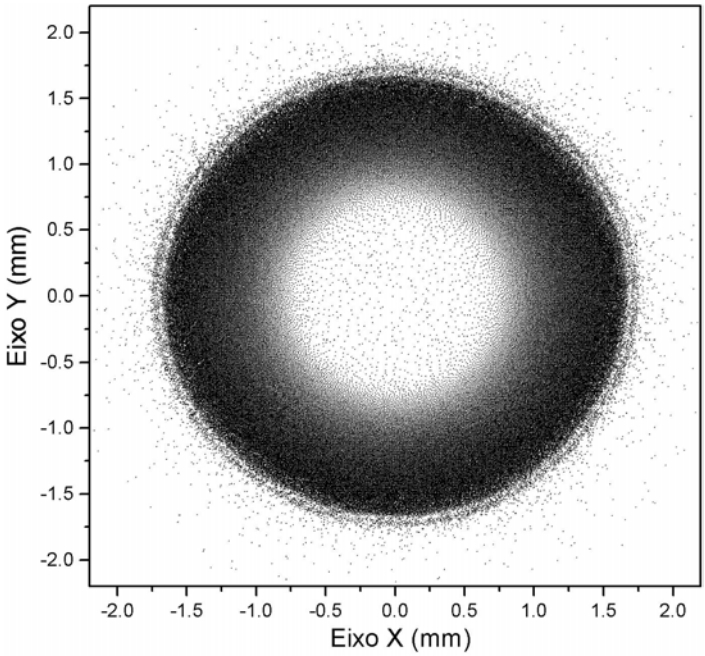


Figura 4.16 – Distribuição no plano para N = 10^6 átomos, α*10.

Num regime de baixas densidades, ou seja, para algumas dezenas de átomos, observamos que estes se acomodam num raio r = 1,344 mm. Nesse caso, a contribuição da interação é desprezível e podemos dizer que esse é o raio de equilíbrio devido somente às forças de aprisionamento dos átomos. É razoável que com o aumento do número de átomos aprisionados e, conseqüentemente, com o aumento da contribuição das forças de interação, a nuvem atômica encontre um novo raio de equilíbrio. Definiremos como raio médio para as distribuições radiais a quantidade:

$$RM = \frac{\sum_i m_i r_i}{\sum_i m_i}, \qquad (4.5)$$

onde m_i é a massa dos átomos contidos no intervalo aberto à direita $[r_i, r_i + \Delta r[$. A equação 4.5 assemelha-se à definição de centro de massa, diferindo no fato de que aqui tomamos as distâncias r_i em módulo, caso contrário, o raio médio seria sempre nulo devido à simetria da nuvem atômica.

Na figura 4.17 mostramos como varia o raio médio da nuvem atômica à medida que aumentamos o número de átomos na armadilha. É importante salientar que os valores da posição do raio médio, obtidos para N = 10^3 e 10^4 átomos são os mesmos, tanto para as simulações usando o método exato quanto usando o método aproximado. A figura 4.17 mostra

ainda que a nuvem atômica acomoda-se em novos raios de estabilidade, maiores que aquele previsto para o regime de baixas densidades, à medida que a armadilha é preenchida. Esperamos que para simulações com N da ordem de 10^7 átomos a instabilidade provocada por estes, apresente uma separação da estrutura de dois anéis, como sugerido pelas figuras 4.15 e 4.16. Contudo, devido a questões estruturais, como por exemplo: a capacidade das máquinas (computadores) utilizadas, nessa dissertação nós estamos limitados a simulações com 10^6 átomos.

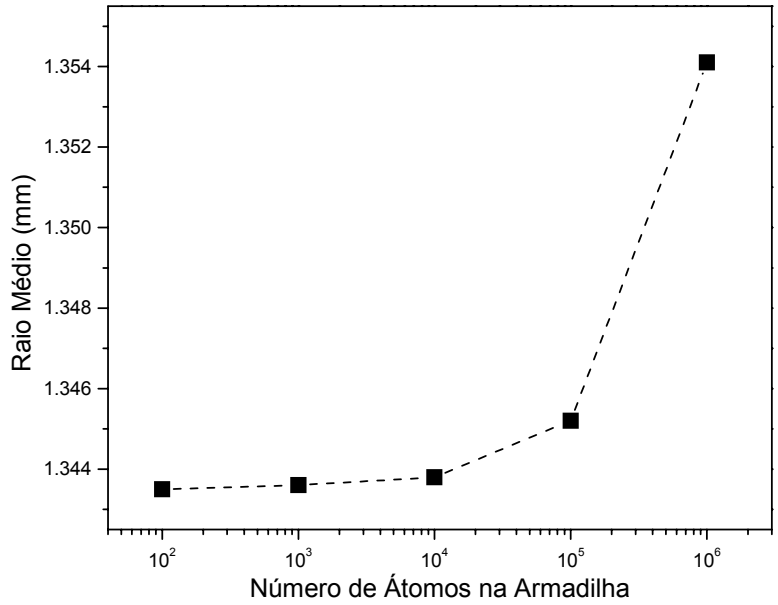


Figura 4.17 – Aumento do diâmetro da nuvem atômica com o aumento dos átomos na armadilha.

Encerramos aqui as simulações cujos resultados foram apresentados nessa dissertação. Estamos confiantes de ter otimizado ao

máximo o tempo disponível, desde a construção e testes dos programas usados nas simulações até a escolha dos parâmetros a serem usados nestas. Da construção dos programas, podemos dizer que constituiu um desafio que se mostrou amplamente enriquecedor no que concerne ao domínio das técnicas de simulação, sendo esta uma ferramenta que certamente ainda tem muito a oferecer a pesquisadores, em todas as áreas da Física.

Sabemos que muito ainda pode ser feito para enriquecer este trabalho, como por exemplo uma análise mais criteriosa do erro envolvido nas simulações. Na verdade, é preciso criar uma forma mais metódica de avaliar o erro, visto que da forma como este é calculado em sistemas auto-gravitantes, de onde trouxemos o método aproximativo usado nas simulações, o método de avaliação do erro não pode ser aplicado para o sistema atômico estudado aqui. Além disso, pretendemos obter resultados com simulações feitas para valores intermediários entre uma e outra ordem de grandeza do número de átomos na armadilha com o objetivo de investir com mais segurança numa incursão em simulações com 10^7 átomos na armadilha, onde esperamos observar uma nuvem com dois anéis concêntricos.

Conclusões

O objetivo principal dessa tese foi investigar os efeitos da interação entre os átomos aprisionados numa armadilha magneto-ótica visando explicar as diferentes distribuições espaciais observadas experimentalmente quando dois pares de feixes são desalinhados. Para isso adaptamos e implementamos um programa baseado num algoritmo hierárquico que tem controle na aproximação feita para o cálculo da interação. Devido ao movimento orbital dos átomos, as configurações das nuvens atômicas são chamadas de modos orbitais e apresentam uma forte dependência com o conjunto de parâmetros que caracterizam a armadilha tais como: *detuning*, gradiente de campo magnético, desalinhamento, intensidade do laser, entre outros. Em nosso estudo, consideramos uma armadilha onde tomamos como base os parâmetros do átomo de sódio, os quais correspondem às condições onde são observados dois anéis praticamente concêntricos. De fato, cada um dos parâmetros da armadilha poderia ser variado na busca dos diversos arranjos atômicos observados experimentalmente, mas devido ao alto custo computacional, essa busca torna-se impraticável. Experimentalmente observa-se que o modo dois anéis concêntricos ocorre como uma evolução do modo de um anel simples. Primeiro surge um anel simples na armadilha e somente com o

aumento da densidade de átomos na armadilha surge o modo de dois anéis concêntricos. Escolhemos manter os outros parâmetros fixos e variar o número de átomos com a finalidade de verificar os efeitos da interação. O aumento do número de átomos na armadilha implica diretamente em aumentar a importância da interação. Uma forma de simular um maior número de átomos na armadilha para aumentar o efeito da interação sem aumentar em nada o custo computacional foi aumentar o parâmetro de interação α. Mostramos que aumentar em até uma ordem de grandeza é uma boa aproximação.

A interação, da maneira como foi utilizada, foi proposta por Carl Wieman e colaboradores [22] e tem como base efeitos de múltiplo espalhamento de fótons. Embora seja uma interação de longo alcance, pois cai com o quadrado da distância, sua intensidade é baixa de modo que se faz necessário trabalhar num regime de altas densidades, ou seja, um número da ordem de 10^4 - 10^6 átomos.

Deparamos-nos com um conhecido problema físico que é solução de equações diferenciais ordinárias para N corpos interagentes. A solução do problema de N corpos, por meio da soma direta da contribuição de cada corpo sobre todos os demais, impõe um custo computacional muito alto, o qual depende das máquinas disponíveis e do tipo de programação utilizada. Buscamos na literatura um método aproximativo que nos propicie um menor custo computacional. Adaptamos o algoritmo

hierárquico para o contexto da ótica a partir do estudo de N corpos auto-gravitantes, usado na Cosmologia e Astrofísica.

Simulações com $N > 10^4$ átomos somente foram possíveis usando o método árvore, com o qual chegamos a $N = 10^6$ átomos. Esse é o limite ao qual é possível chegar usando esse método aproximativo sem o auxílio de supercomputadores nem programação paralela. Atingimos esse limite usando um computador pessoal PC com velocidade de 1,2 GHz. Também fizemos simulações no Laboratório de Física Teórica e Computacional (LFTC) com máquinas de 1,4 GHz.

Para $N = 10^6$ átomos e α^*10, observamos uma configuração, contendo uma segunda região onde os átomos se concentram, que parece indicar o início da formação do modo com dois anéis concêntricos. Quando se mede experimentalmente a distribuição, como visto nas imagens das nuvens atômicas nas armadilhas (figuras 1.1), o que se observa não são os átomos diretamente, mas a fluorescência deles. Zonas de maior aglomeração, ou concentração serão observadas mais facilmente que zonas de baixa concentração.

Perspectivas

Na maior parte do nosso trabalho, buscamos aumentar consideravelmente o número de átomos na armadilha, até atingirmos densidades nas quais pudéssemos comparar nossos resultados com outros obtidos experimentalmente, portanto julgamos necessário aumentar ainda mais uma ordem de grandeza no número de átomos na armadilha. Com esse fim percorremos a literatura na busca de métodos aproximativos mais robustos, encontramos basicamente três métodos aproximativos distintos aplicados na solução do problema de N corpos interagentes, são eles: o método da árvore hierárquica, comumente referido por TREE; o método partícula-malha, referido por PM e os métodos híbridos. O método PM, consiste basicamente em calcular a densidade, $\rho(\vec{r})$, das partículas numa malha fixa cobrindo todo o espaço e calcular a interação nos nós da malha. Os métodos híbridos são formados basicamente pela junção de dois métodos existentes. Com métodos de cálculo da interação mais robustos, como os citados aqui, têm sido feitas simulações com 10^8 – 10^9 partículas. Naturalmente a arquitetura e o tipo de programação, linear ou paralela, influi nesses limites.

Apêndice – Código Fonte

Abaixo disponibilizamos o código fonte do programa utilizado para as simulações, cujos resultados são apresentados no capítulo IV desse trabalho.

Na forma como está posto no código abaixo, a cada intervalo $\Delta t = 10$ milissegundos de tempo físico, o programa coleta uma distribuição e guarda em arquivo, permitindo o acompanhamento da evolução do sistema de átomos na armadilha. O sistema evolui durante 420 milissegundos de tempo físico, a escolha desse tempo foi discutida no capítulo IV.

Além de guardar sucessivas distribuições, também são arquivadas posições e velocidades, dessa forma pudemos usar esse código como base para pequenas modificações, como por exemplo, continuar a simulação a partir da última saída de dados, em caso de queda de energia.

Código fonte:

```
#include <stdlib.h>    // Inclusão de bibliotecas.
#include <stdio.h>
#include <math.h>
#include <time.h>
#define fc 1000.0      // Fator de conversão de milímetro para metro.

double n = 1e5;     // Número de átomos manipulados.
int ndc = 8050;
int *rmai, *rmen;
```

```
static int rel, acmdo_g = 0, acmdo_p = 0;
char livre;
char minucias[25] = "Detalhes.dat";
char distribui[25] = "Distribuições.dat";
char eixo_x[25] = "Posição_x.dat";
char eixo_y[25] = "Posição_y.dat";
double PI = 3.1415926535;
double m = 3.819e-26;
double hc = 1.054589e-34;
double ka = 1.086e7;
double g1 = 1.4e6;
double b = 0.5;
double Wo = 4.;
double del= -2.;
double w1 = 3.;
double alfa = 1.36685e-4;
double teta = 1.;
double dt = 3.2e-5;
double pm = 312.5;      // 0,01 = dt*pm
double NE = 1.32e4;     // 0,42 = dt*NE
double ni, dtt, dt6;
double G, k, g, dg;
double s, w, C, dis, r, fi;
FILE *erro;      // Notifica relocações.

struct atomo {   double x, y;
                 double xt, yt;
                 double vx, vy;
                 double v2x, v2y;
                 double v3x, v3y;
                 double ax, ay;
                 double a2x, a2y;
                 double a3x, a3y;
                 char rp, rg;
                 struct atomo *prox, *adia; };
struct atomo *atual, *novo;

struct cell {double npart;
             double x, y, a;
             double xcm, ycm;
             struct cell *f1, *f2, *f3, *f4;
             struct atomo *anc, *cost;  }eva;

struct arma {struct cell *cela;
             struct arma *seg;   }prim;

int main(void)
{
   double i, nc;
   int l, pc;          // Contadores.
   void rk4();         // Função do Numerical Recipies.
   void divide1();     // Divide o espaço em células usando x, y.
   void interage1();   // Calcula F_I usando x, y, vx, vy e guarda em ax, ay.
```

```
void acumula();     // Constrói as distribuições.
void escreve();     // Guarda em arquivo as distribuições.
void organiza();    // Libera espaço de memória e prepara pare recomeçar.
time_t t_ini, t_end;
FILE *partes;

srand48(time(NULL));
teta = 1./teta;
G = 2e7*PI;
k = ka/G;
g = (g1/G)*PI;
dg = Wo*Wo;
s = w1;
w = w1*w1;
C = dg*ka*G*(hc/m);
dtt = 0.5*dt;
dt6 = dt/6.0;

eva.x = -4.0;
eva.y = -4.0;
eva.a = 8.0;
eva.anc = (struct atomo*)NULL;
eva.cost = (struct atomo*)NULL;
eva.npart = n;

rmai = (int*)malloc((unsigned)ndc*sizeof(int));
if(!rmai) printf("falha na alocação de rmai\n");
rmen = (int*)malloc((unsigned)ndc*sizeof(int));
if(!rmen) printf("falha na alocação de rmen\n");

for(i=0; i<ndc; i++)
{
   rmai[(int)i] = 0;
   rmen[(int)i] = 0;
}

// Alocando todas os átomos na célula eva.
for(i=0; i<n; i++)
{
   novo = (struct atomo*)malloc(sizeof(struct atomo));
   if(!novo) printf("falhou aloc. de atomos na cel. eva.\n");
   if(eva.anc==(struct atomo*)NULL)
   {
      eva.anc = atual = novo;
      eva.cost = eva.anc;
   }
   else
   {
      atual->prox = atual->adia = novo;
      atual = novo;
   }
   atual->x = (2*drand48()-1.)*2.12;
   atual->y = (2*drand48()-1.)*2.12;
```

```
    atual->vx = 0.0; atual->vy = 0.0;
    atual->ax = 0.0; atual->ay = 0.0;
    atual->rp = 'n'; atual->rg = 'n';
    atual->adia = atual->prox = (struct atomo*)NULL;
}

printf("\nEXECUTANDO.....");
partes = fopen(minucias,"a");
fprintf(partes,"\nEXECUÇÃO: %g ÁTOMOS.\n", n);
fprintf(partes,"ARQUIVO FONTE: Cell_ac.c\n");
fprintf(partes,"PARÂMETROS:\n dt=%g | alfa=%.3e | ", dt, alfa);
fprintf(partes,"teta=%.3g.\n", 1./teta);
fprintf(partes," b=%g | Wo=%g | del=%g.\n", b, Wo, del);
t_ini = time(NULL);
fprintf(partes," INICIO: %ld", t_ini);
fclose(partes);

l = (int)pm - 1;
pc = 0;
for( ni=0.; ni<NE; ni++)
{
    rel = 0;
    divide1();
    interage1();
    rk4();
    if( (l%25)==0 )  // Sinal de continuidade.
    {
          partes = fopen(minucias,"a");
          fprintf(partes,"+");
          if( l-300 >= 1 )
             fprintf(partes,"|");
          fclose(partes);
    }
    l++;
    if( l==(int)pm )
    {
       l--;
       pc++;
       acumula();
       if( livre=='s' ) // livre == 's' quando contagem terminada.
       {
          escreve();
          l = pc;
          t_end = time(NULL);
          partes = fopen(minucias,"a");
          fprintf(partes,"\nT_fis: %8g s | ", ni*dt, n, pc);
          fprintf(partes,"n: %6g | pc: %2d | ", n, pc);
          fprintf(partes,"TM:%9.2fs",difftime(t_end,t_ini));
          fprintf(partes," | ni = %5g\t", ni);
          fclose(partes);
          pc = 0;
          acmdo_p = 0;
          acmdo_g = 0;
```

```
            }
        }
        organiza();
        if( rel > 0 )
        {
            erro = fopen("Erro.txt","a");
            fprintf(erro,"%-4g - relocações = %5d\n", ni, rel);
            fclose(erro);
        }
    }
    t_end = time(NULL);
    partes = fopen(minucias,"a");
    fprintf(partes,"\nTERMINO: %ld\nT_fis: %g seg.\n", t_end, ni*dt);
    fprintf(partes,"T_MÁQ.: %.2f seg.\n", difftime(t_end, t_ini));
    fclose(partes);

    printf("\n--> FINAL DE EXECUÇÃO!\n\n");
    return(0);
}


void rk4()
{
    void divide2();     // Divide o espaço usando xt, yt.
    void interage2();   // Calcula F_I usando xt, yt, v3, guarda em a3.
    void interage3();   // Calcula F_I usando xt, yt, v2, guarda em a2.
    void organiza();

    atual = eva.cost;
    while( atual!=(struct atomo*)NULL )
    {
        atual->xt = atual->x + atual->vx*dtt*fc;
        atual->yt = atual->y + atual->vy*dtt*fc;
        atual->v3x = atual->vx + atual->ax*dtt;
        atual->v3y = atual->vy + atual->ay*dtt;
        atual = atual->adia;
    }
    organiza();

    divide2();
    interage2();
    atual = eva.cost;
    while( atual!=(struct atomo*)NULL )
    {
        atual->xt = atual->x + atual->v3x*dtt*fc;
        atual->yt = atual->y + atual->v3y*dtt*fc;
        atual->v2x = atual->vx + atual->a3x*dtt;
        atual->v2y = atual->vy + atual->a3y*dtt;
        atual = atual->adia;
    }
    organiza();

    divide2();
```

```
    interage3();
    atual = eva.cost;
    while( atual!=(struct atomo*)NULL )
    {
        atual->xt = atual->x + atual->v2x*dt*fc;
        atual->yt = atual->y + atual->v2y*dt*fc;
        atual->v2x += atual->v3x;
        atual->v2y += atual->v3y;

        atual->v3x = atual->vx + atual->a2x*dt;
        atual->v3y = atual->vy + atual->a2y*dt;
        atual->a2x += atual->a3x;
        atual->a2y += atual->a3y;
        atual = atual->adia;
    }
    organiza();

    divide2();
    interage2();

    atual = eva.cost;
    while( atual!=(struct atomo*)NULL )
    {
        atual->x = atual->x + dt6*(atual->vx + atual->v3x + 2.0*atual->v2x)*fc;
        atual->y = atual->y + dt6*(atual->vy + atual->v3y + 2.0*atual->v2y)*fc;
        atual->vx = atual->vx + dt6*(atual->ax + atual->a3x + 2.0*atual->a2x);
        atual->vy = atual->vy + dt6*(atual->ay + atual->a3y + 2.0*atual->a2y);
        atual = atual->adia;
    }
}


void divide1()
{
    int i;
    double af;    // Aresta da célula filha.
    double p1x, p1y, p2x, p2y, p3x, p3y, p4x, p4y;
    struct atomo *pramof1, *pramof2, *pramof3, *pramof4;
    struct cell *c_sup, *f1, *f2, *f3, *f4;
    struct arma *davez, *final;

    c_sup = &eva;
    if(!eva.anc)
    {
        printf("eva vazia");
        exit(0);
    }

    prim.cela = &eva;
    prim.seg = (struct arma*)NULL;
    davez = final = &prim;

    atual = eva.cost;
```

```
p1x = p1y = 0.0;
while( atual!=(struct atomo*)NULL )
{
   dis = sqrt( atual->x*atual->x + atual->y*atual->y );
   // Relocação de átomos fujões.
   if( dis > 3.9 ) // Menor que a aresta da célula maior.
   {
      atual->x = (2*drand48()-1.)*2.12;
      atual->y = (2*drand48()-1.)*2.12;
      atual->xt = atual->x;
      atual->yt = atual->y;
      atual->vx = 0.0; atual->vy = 0.0;
      atual->v2x = 0.0;   atual->v2y = 0.0;
      atual->v3x = 0.0;   atual->v3y = 0.0;
      atual->rp = 'n'; atual->rg = 'n';
      rel++;
   }
   p1x += atual->x; p1y += atual->y;
   atual = atual->adia;
}
if( eva.npart!=n )  eva.npart = n;
eva.xcm = p1x/n; eva.ycm = p1y/n;


do{
   af = (c_sup->a)*0.5;

   f1 = (struct cell*)malloc(sizeof(struct cell));
   if(!f1){ printf("falha na alocacao de f1\n"); exit(0); }
   f1->npart = 0;
   f1->x = c_sup->x;
   f1->y = c_sup->y;
   f1->a = af;
   f1->f1 = (struct cell*)NULL;  f1->f2 = (struct cell*)NULL;
   f1->f3 = (struct cell*)NULL;  f1->f4 = (struct cell*)NULL;
   f1->anc = (struct atomo*)NULL;

   f2 = (struct cell*)malloc(sizeof(struct cell));
   if(!f2){ printf("falha na alocacao de f2\n"); exit(0); }
   f2->npart = 0;
   f2->x = c_sup->x + af;
   f2->y = c_sup->y;
   f2->a = af;
   f2->f1 = (struct cell*)NULL;  f2->f2 = (struct cell*)NULL;
   f2->f3 = (struct cell*)NULL;  f2->f4 = (struct cell*)NULL;
   f2->anc = (struct atomo*)NULL;

   f3 = (struct cell*)malloc(sizeof(struct cell));
   if(!f3){ printf("falha na alocacao de f3\n"); exit(0); }
   f3->npart = 0;
   f3->x = c_sup->x;
   f3->y = c_sup->y + af;
   f3->a = af;
```

```
f3->f1 = (struct cell*)NULL;  f3->f2 = (struct cell*)NULL;
f3->f3 = (struct cell*)NULL;  f3->f4 = (struct cell*)NULL;
f3->anc = (struct atomo*)NULL;

f4 = (struct cell*)malloc(sizeof(struct cell));
if(!f4){ printf("falha na alocacao de f4\n"); exit(0); }
f4->npart = 0;
f4->x = c_sup->x + af;
f4->y = c_sup->y + af;
f4->a = af;
f4->f1 = (struct cell*)NULL;  f4->f2 = (struct cell*)NULL;
f4->f3 = (struct cell*)NULL;  f4->f4 = (struct cell*)NULL;
f4->anc = (struct atomo*)NULL;

c_sup->f1 = f1;  c_sup->f2 = f2;
c_sup->f3 = f3;  c_sup->f4 = f4;

p1x = p1y = 0.0; p2x = p2y = 0.0;
p3x = p3y = 0.0; p4x = p4y = 0.0;

// Transferindo os átomos entre células:
atual = c_sup->anc;
while( c_sup->anc!=(struct atomo*)NULL )
{
   if(atual->x > f1->x && atual->x <= (f1->x+f1->a))
   if(atual->y > f1->y && atual->y <= (f1->y+f1->a))
   {
      p1x += atual->x; p1y += atual->y;
      c_sup->anc = atual->prox;
      if(f1->anc==(struct atomo*)NULL)
         f1->anc = atual;
      else   pramof1->prox = atual;
      pramof1 = atual;
      pramof1->prox = (struct atomo*)NULL;
      atual = c_sup->anc;
      f1->npart += 1;
   }
   if(c_sup->anc!=(struct atomo*)NULL)
   if(atual->x > f2->x && atual->x <= (f2->x+f2->a))
   if(atual->y > f2->y && atual->y <= (f2->y+f2->a))
   {
      p2x += atual->x; p2y += atual->y;
      c_sup->anc = atual->prox;
      if(f2->anc==(struct atomo*)NULL)
         f2->anc = atual;
      else
         pramof2->prox = atual;
      pramof2 = atual;
      pramof2->prox = (struct atomo*)NULL;
      atual = c_sup->anc;
      f2->npart += 1;
   }
   if(c_sup->anc!=(struct atomo*)NULL)
```

```
    if(atual->x > f3->x && atual->x <= (f3->x+f3->a))
    if(atual->y > f3->y && atual->y <= (f3->y+f3->a))
    {
       p3x += atual->x; p3y += atual->y;
       c_sup->anc = atual->prox;
       if(f3->anc==(struct atomo*)NULL)
          f3->anc = atual;
       else   pramof3->prox = atual;
       pramof3 = atual;
       pramof3->prox = (struct atomo*)NULL;
       atual = c_sup->anc;
       f3->npart += 1;
    }
    if(c_sup->anc!=(struct atomo*)NULL)
    if(atual->x > f4->x && atual->x <= (f4->x+f4->a))
    if(atual->y > f4->y && atual->y <= (f4->y+f4->a))
    {
       p4x += atual->x; p4y += atual->y;
       c_sup->anc = atual->prox;
       if(f4->anc==(struct atomo*)NULL)
          f4->anc = atual;
       else   pramof4->prox = atual;
       pramof4 = atual;
       pramof4->prox = (struct atomo*)NULL;
       atual = c_sup->anc;
       f4->npart += 1;
    }
 }// Final da alocação dos átomos nas células filhas.

 if( f1->npart >= 1 )
 {
    if( final == &prim )
    {
       prim.seg = (struct arma*)malloc(sizeof(struct arma));
       final = prim.seg;
    }
    else
    {
       final->seg = (struct arma*)malloc(sizeof(struct arma));
       final = final->seg;
    }  if(!final) printf("falha na aloc. de est. arma\n");
    final->cela = f1;
    f1->xcm = p1x/f1->npart;
    f1->ycm = p1y/f1->npart;
 }
 else
 {
    c_sup->f1 = (struct cell*)NULL;
    free(f1);
 }

 if( f2->npart >= 1 )
 {
```

```
      if( final == &prim )
      {
         prim.seg = (struct arma*)malloc(sizeof(struct arma));
         final = prim.seg;
      }
      else
      {
         final->seg = (struct arma*)malloc(sizeof(struct arma));
         final = final->seg;
      }  if(!final) printf("falha na aloc. de est. arma\n");
      final->cela = f2;
      f2->xcm = p2x/f2->npart;
      f2->ycm = p2y/f2->npart;
   }
   else
   {
      c_sup->f2 = (struct cell*)NULL;
      free(f2);
   }

   if( f3->npart >= 1 )
   {
      if( final == &prim )
      {
         prim.seg = (struct arma*)malloc(sizeof(struct arma));
         final = prim.seg;
      }
      else
      {
         final->seg = (struct arma*)malloc(sizeof(struct arma));
         final = final->seg;
      }  if(!final) printf("falha na aloc. de est. arma\n");
      final->cela = f3;
      f3->xcm = p3x/f3->npart;
      f3->ycm = p3y/f3->npart;
   }
   else
   {
      c_sup->f3 = (struct cell*)NULL;
      free(f3);
   }

   if( f4->npart >= 1 )
   {
      if( final == &prim )
      {
         prim.seg = (struct arma*)malloc(sizeof(struct arma));
         final = prim.seg;
      }
      else
      {
         final->seg = (struct arma*)malloc(sizeof(struct arma));
         final = final->seg;
```

```
            }  if(!final) printf("falha na aloc. de est. arma\n");
            final->cela = f4;
            f4->xcm = p4x/f4->npart;
            f4->ycm = p4y/f4->npart;
        }
        else
        {
            c_sup->f4 = (struct cell*)NULL;
            free(f4);
        }

        final->seg = (struct arma*)NULL;
        davez = davez->seg;
        c_sup = davez->cela;
        while( c_sup->npart==1 && davez!=(struct arma*)NULL )
        {
            davez = davez->seg;
            if(davez!=(struct arma*)NULL)
                c_sup = davez->cela;
        }
    }while( davez!=(struct arma*)NULL );
}


void divide2()
{
    int i;
    double af;   // Aresta da célula filha.
    double p1x, p1y, p2x, p2y, p3x, p3y, p4x, p4y;
    struct atomo *pramof1, *pramof2, *pramof3, *pramof4;
    struct cell *c_sup, *f1, *f2, *f3, *f4;
    struct arma *davez, *final;

    c_sup = &eva;
    if(!eva.anc){ printf("eva vazia"); exit(0); }

    prim.cela = &eva;
    prim.seg = (struct arma*)NULL;
    davez = final = &prim;

    atual = eva.cost;
    p1x = p1y = 0.0;
    while( atual!=(struct atomo*)NULL )
    {
        dis = sqrt( atual->xt*atual->xt + atual->yt*atual->yt );
        // Relocação de átomos fujões.
        if( dis > 3.9 ) // Menor que a aresta da célula maior.
        {
            atual->x = (2*drand48()-1.)*2.12;
            atual->y = (2*drand48()-1.)*2.12;
            atual->xt = atual->x;
            atual->yt = atual->y;
            atual->vx = 0.0; atual->vy = 0.0;
```

```
            atual->v2x = 0.0;   atual->v2y = 0.0;
            atual->v3x = 0.0;   atual->v3y = 0.0;
            atual->rp = 'n'; atual->rg = 'n';
            rel++;
        }
        p1x += atual->xt;   p1y += atual->yt;
        atual = atual->adia;
    }
    if( eva.npart!=n )  eva.npart = n;
    eva.xcm = p1x/n;
    eva.ycm = p1y/n;

    do{
        af = (c_sup->a)*0.5;

        f1 = (struct cell*)malloc(sizeof(struct cell));
        if(!f1){ printf("falha na alocacao de f1\n"); exit(0); }
        f1->npart = 0;
        f1->x = c_sup->x;
        f1->y = c_sup->y;
        f1->a = af;
        f1->f1 = (struct cell*)NULL;  f1->f2 = (struct cell*)NULL;
        f1->f3 = (struct cell*)NULL;  f1->f4 = (struct cell*)NULL;
        f1->anc = (struct atomo*)NULL;

        f2 = (struct cell*)malloc(sizeof(struct cell));
        if(!f2){ printf("falha na alocacao de f2\n"); exit(0); }
        f2->npart = 0;
        f2->x = c_sup->x + af;
        f2->y = c_sup->y;
        f2->a = af;
        f2->f1 = (struct cell*)NULL;  f2->f2 = (struct cell*)NULL;
        f2->f3 = (struct cell*)NULL;  f2->f4 = (struct cell*)NULL;
        f2->anc = (struct atomo*)NULL;

        f3 = (struct cell*)malloc(sizeof(struct cell));
        if(!f3){ printf("falha na alocacao de f3\n"); exit(0); }
        f3->npart = 0;
        f3->x = c_sup->x;
        f3->y = c_sup->y + af;
        f3->a = af;
        f3->f1 = (struct cell*)NULL;  f3->f2 = (struct cell*)NULL;
        f3->f3 = (struct cell*)NULL;  f3->f4 = (struct cell*)NULL;
        f3->anc = (struct atomo*)NULL;

        f4 = (struct cell*)malloc(sizeof(struct cell));
        if(!f4){ printf("falha na alocacao de f4\n"); exit(0); }
        f4->npart = 0;
        f4->x = c_sup->x + af;
        f4->y = c_sup->y + af;
        f4->a = af;
        f4->f1 = (struct cell*)NULL;  f4->f2 = (struct cell*)NULL;
        f4->f3 = (struct cell*)NULL;  f4->f4 = (struct cell*)NULL;
```

```
f4->anc = (struct atomo*)NULL;

c_sup->f1 = f1;  c_sup->f2 = f2;
c_sup->f3 = f3;  c_sup->f4 = f4;

p1x = p1y = 0.0; p2x = p2y = 0.0;
p3x = p3y = 0.0; p4x = p4y = 0.0;

// Transferindo os átomos entre células;
atual = c_sup->anc;
while( c_sup->anc!=(struct atomo*)NULL )
{
   if(atual->xt > f1->x && atual->xt <= (f1->x + f1->a))
   if(atual->yt > f1->y && atual->yt <= (f1->y + f1->a))
   {
      p1x += atual->xt;      p1y += atual->yt;
      c_sup->anc = atual->prox;
      if(f1->anc==(struct atomo*)NULL)
         f1->anc = atual;
      else   pramof1->prox = atual;
      pramof1 = atual;
      pramof1->prox = (struct atomo*)NULL;
      atual = c_sup->anc;
      f1->npart += 1;
   }
   if(c_sup->anc!=(struct atomo*)NULL)
   if(atual->xt > f2->x && atual->xt <= (f2->x + f2->a))
   if(atual->yt > f2->y && atual->yt <= (f2->y + f2->a))
   {
      p2x += atual->xt;      p2y += atual->yt;
      c_sup->anc = atual->prox;
      if(f2->anc==(struct atomo*)NULL)
         f2->anc = atual;
      else   pramof2->prox = atual;
      pramof2 = atual;
      pramof2->prox = (struct atomo*)NULL;
      atual = c_sup->anc;
      f2->npart += 1;
      }
   if(c_sup->anc!=(struct atomo*)NULL)
   if(atual->xt > f3->x && atual->xt <= (f3->x + f3->a))
   if(atual->yt > f3->y && atual->yt <= (f3->y + f3->a))
   {
      p3x += atual->xt;      p3y += atual->yt;
      c_sup->anc = atual->prox;
      if(f3->anc==(struct atomo*)NULL)
         f3->anc = atual;
      else   pramof3->prox = atual;
      pramof3 = atual;
      pramof3->prox = (struct atomo*)NULL;
      atual = c_sup->anc;
      f3->npart += 1;
   }
```

```
    if(c_sup->anc!=(struct atomo*)NULL)
    if(atual->xt > f4->x && atual->xt <= (f4->x + f4->a))
    if(atual->yt > f4->y && atual->yt <= (f4->y + f4->a))
    {
       p4x += atual->xt;      p4y += atual->yt;
       c_sup->anc = atual->prox;
       if(f4->anc==(struct atomo*)NULL)
          f4->anc = atual;
       else   pramof4->prox = atual;
       pramof4 = atual;
       pramof4->prox = (struct atomo*)NULL;
       atual = c_sup->anc;
       f4->npart += 1;
    }
} // Final da alocação dos átomos nas células filhas.

if( f1->npart >= 1 )
{
    if( final == &prim )
    {
       prim.seg = (struct arma*)malloc(sizeof(struct arma));
       final = prim.seg;
    }
    else
    {
       final->seg = (struct arma*)malloc(sizeof(struct arma));
       final = final->seg;
    }
    if(!final) printf("falha na aloc. de est. arma\n");
    final->cela = f1;
    f1->xcm = p1x/f1->npart;
    f1->ycm = p1y/f1->npart;
}
else
{
    c_sup->f1 = (struct cell*)NULL;
    free(f1);
}

if( f2->npart >= 1 )
{
    if( final == &prim )
    {
       prim.seg = (struct arma*)malloc(sizeof(struct arma));
       final = prim.seg;
    }
    else
    {
       final->seg = (struct arma*)malloc(sizeof(struct arma));
       final = final->seg;
    }
    if(!final) printf("falha na aloc. de est. arma\n");
    final->cela = f2;
```

```
        f2->xcm = p2x/f2->npart;
        f2->ycm = p2y/f2->npart;
    }
    else
    {
        c_sup->f2 = (struct cell*)NULL;
        free(f2);
    }

    if( f3->npart >= 1 )
    {
        if( final == &prim )
        {
            prim.seg = (struct arma*)malloc(sizeof(struct arma));
            final = prim.seg;
        }
        else
        {
            final->seg = (struct arma*)malloc(sizeof(struct arma));
            final = final->seg;
        }
        if(!final) printf("falha na aloc. de est. arma\n");
        final->cela = f3;
        f3->xcm = p3x/f3->npart;
        f3->ycm = p3y/f3->npart;
    }
    else
    {
        c_sup->f3 = (struct cell*)NULL;
        free(f3);
    }

    if( f4->npart >= 1 )
    {
        if( final == &prim )
        {
            prim.seg = (struct arma*)malloc(sizeof(struct arma));
            final = prim.seg;
        }
        else
        {
            final->seg = (struct arma*)malloc(sizeof(struct arma));
            final = final->seg;
        }
        if(!final) printf("falha na aloc. de est. arma\n");
        final->cela = f4;
        f4->xcm = p4x/f4->npart;
        f4->ycm = p4y/f4->npart;
    }
    else
    {
        c_sup->f4 = (struct cell*)NULL;
        free(f4);
```

```
       }

       final->seg = (struct arma*)NULL;
       davez = davez->seg;
       c_sup = davez->cela;
       while( c_sup->npart==1 && davez!=(struct arma*)NULL )
       {
          davez = davez->seg;
          if(davez!=(struct arma*)NULL)
             c_sup = davez->cela;
       }
    }while( davez!=(struct arma*)NULL );
}


void interage1()
{
    double dcm, dcm3;
    double e1, e2, m1, m2;
    struct cell *cela;
    struct fila {struct cell *celpi;
                 struct fila *post;  };
    struct fila *ini, *ult, *kdv;

    atual = eva.cost;
    while( atual!=(struct atomo*)NULL)
    {
       e1 = exp(-(atual->y-s)*(atual->y-s)/w);
       m1 = del + atual->vx*k + g*b*atual->x;
       e2 = exp(-(atual->y+s)*(atual->y+s)/w);
       m2 = del - atual->vx*k - g*b*atual->x;
       atual->ax = -C*e1/(1.+2.*dg*e1+4.*m1*m1)+C*e2/(1.+2.*dg*e2+4.*m2*m2);

       e1 = exp(-(atual->x+s)*(atual->x+s)/w);
       m1 = del + atual->vy*k + g*b*atual->y;
       e2 = exp(-(atual->x-s)*(atual->x-s)/w);
       m2 = del - atual->vy*k - g*b*atual->y;
       atual->ay = -C*e1/(1.+2.*dg*e1+4.*m1*m1)+C*e2/(1.+2.*dg*e2+4.*m2*m2);

       ini = kdv = ult = (struct fila*)malloc(sizeof(struct fila));
       kdv->post = (struct fila*)NULL;
       cela = kdv->celpi = &eva;

       while( kdv!=(struct fila*)NULL )
       {
          dcm = sqrt((atual->x-cela->xcm)^2+(atual->y-cela->ycm)^2);
          dcm3 = dcm^3;
          if( dcm==0.0 )
          {
             kdv = kdv->post;
             if( kdv!=(struct fila*)NULL )
                cela = kdv->celpi;
             free(ini);
```

```
            ini = kdv;
         }
         else
         {
            if( (dcm/cela->a > teta) || cela->npart==1 )
            {
               atual->ax += alfa*cela->npart*(atual->x-cela->xcm)/dcm3;
               atual->ay += alfa*cela->npart*(atual->y-cela->ycm)/dcm3;

               kdv = kdv->post;
               if( kdv!=(struct fila*)NULL )
                  cela = kdv->celpi;
               free(ini);
               ini = kdv;
            }
            else
            {
               if( cela->npart > 1 )
               {
                  if( cela->f1!=(struct cell*)NULL )
                  {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                     ult = ult->post;
                     ult->celpi = cela->f1; }
                  if( cela->f2!=(struct cell*)NULL )
                  {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                     ult = ult->post;
                     ult->celpi = cela->f2; }
                  if( cela->f3!=(struct cell*)NULL )
                  {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                     ult = ult->post;
                     ult->celpi = cela->f3; }
                  if( cela->f4!=(struct cell*)NULL )
                  {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                     ult = ult->post;
                     ult->celpi = cela->f4; }
                  ult->post = (struct fila*)NULL;
               }
               kdv = kdv->post;
               if( kdv!=(struct fila*)NULL )
                  cela = kdv->celpi;
               free(ini);
               ini = kdv;
            }
         }
      }
      atual = atual->adia;
   }
}


void interage2()
{
   double dcm, dcm3;
```

```
double e1, e2, m1, m2;
struct cell *cela;
struct fila {struct cell *celpi;
          struct fila *post;  };
struct fila *ini, *ult, *kdv;

atual = eva.cost;
while( atual!=(struct atomo*)NULL)
{
   e1 = exp(-(atual->yt-s)*(atual->yt-s)/w);
   m1 = del + atual->v3x*k + g*b*atual->xt;
   e2 = exp(-(atual->yt+s)*(atual->yt+s)/w);
   m2 = del - atual->v3x*k - g*b*atual->xt;
   atual->a3x = -C*e1/(1.+2.*dg*e1+4.*m1*m1)+C*e2/(1.+2.*dg*e2+4.*m2*m2);

   e1 = exp(-(atual->xt+s)*(atual->xt+s)/w);
   m1 = del + atual->v3y*k + g*b*atual->yt;
   e2 = exp(-(atual->xt-s)*(atual->xt-s)/w);
   m2 = del - atual->v3y*k - g*b*atual->yt;
   atual->a3y = -C*e1/(1.+2.*dg*e1+4.*m1*m1)+C*e2/(1.+2.*dg*e2+4.*m2*m2);

   ini = kdv = ult = (struct fila*)malloc(sizeof(struct fila));
   kdv->post = (struct fila*)NULL;
   cela = kdv->celpi = &eva;

   while( kdv!=(struct fila*)NULL )
   {
      dcm = sqrt((atual->xt-cela->xcm)^2+(atual->yt-cela->ycm)^2);
      dcm3 = dcm^3;
      if( dcm==0.0 )
      {
         kdv = kdv->post;
         if( kdv!=(struct fila*)NULL )
            cela = kdv->celpi;
         free(ini);
         ini = kdv;
      }
      else
      {
         if( (dcm/cela->a > teta) || cela->npart==1 )
         {
            atual->a3x += alfa*cela->npart*(atual->xt-cela->xcm)/dcm3;
            atual->a3y += alfa*cela->npart*(atual->yt-cela->ycm)/dcm3;

            kdv = kdv->post;
            if( kdv!=(struct fila*)NULL )
               cela = kdv->celpi;
            free(ini);
            ini = kdv;
         }
         else
         {
            if( cela->npart > 1 )
```

```
                {
                   if( cela->f1!=(struct cell*)NULL )
                   {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                      ult = ult->post;
                      ult->celpi = cela->f1; }
                   if( cela->f2!=(struct cell*)NULL )
                   {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                      ult = ult->post;
                      ult->celpi = cela->f2; }
                   if( cela->f3!=(struct cell*)NULL )
                   {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                      ult = ult->post;
                      ult->celpi = cela->f3; }
                   if( cela->f4!=(struct cell*)NULL )
                   {  ult->post = (struct fila*)malloc(sizeof(struct fila));
                      ult = ult->post;
                      ult->celpi = cela->f4; }
                   ult->post = (struct fila*)NULL;
                }
                kdv = kdv->post;
                if( kdv!=(struct fila*)NULL )
                   cela = kdv->celpi;
                free(ini);
                ini = kdv;
             }
          }
       }
       atual = atual->adia;
    }
}


void interage3()
{
   double dcm, dcm3;
   double e1, e2, m1, m2;
   struct cell *cela;
   struct fila {struct cell *celpi;
                struct fila *post;  };
   struct fila *ini, *ult, *kdv;

   atual = eva.cost;
   while( atual!=(struct atomo*)NULL)
   {
      e1 = exp(-(atual->yt-s)*(atual->yt-s)/w);
      m1 = del + atual->v2x*k + g*b*atual->xt;
      e2 = exp(-(atual->yt+s)*(atual->yt+s)/w);
      m2 = del - atual->v2x*k - g*b*atual->xt;
      atual->a2x = -C*e1/(1.+2.*dg*e1+4.*m1*m1)+C*e2/(1.+2.*dg*e2+4.*m2*m2);

      e1 = exp(-(atual->xt+s)*(atual->xt+s)/w);
      m1 = del + atual->v2y*k + g*b*atual->yt;
      e2 = exp(-(atual->xt-s)*(atual->xt-s)/w);
```

```
m2 = del - atual->v2y*k - g*b*atual->yt;
atual->a2y = -C*e1/(1.+2.*dg*e1+4.*m1*m1)+C*e2/(1.+2.*dg*e2+4.*m2*m2);

ini = kdv = ult = (struct fila*)malloc(sizeof(struct fila));
kdv->post = (struct fila*)NULL;
cela = kdv->celpi = &eva;

while( kdv!=(struct fila*)NULL )
{
   dcm = sqrt((atual->xt-cela->xcm)^2+(atual->yt-cela->ycm)^2);
   dcm3 = dcm^3;
   if( dcm==0.0 )
   {
      kdv = kdv->post;
      if( kdv!=(struct fila*)NULL )
         cela = kdv->celpi;
      free(ini);
      ini = kdv;
   }
   else
   {
      if( (dcm/cela->a > teta) || cela->npart==1 )
      {
         atual->a2x += alfa*cela->npart*(atual->xt-cela->xcm)/dcm3;
         atual->a2y += alfa*cela->npart*(atual->yt-cela->ycm)/dcm3;
         kdv = kdv->post;
         if( kdv!=(struct fila*)NULL )
            cela = kdv->celpi;
         free(ini);
         ini = kdv;
      }
      else
      {
         if( cela->npart > 1 )
         {
            if( cela->f1!=(struct cell*)NULL )
            {  ult->post = (struct fila*)malloc(sizeof(struct fila));
               ult = ult->post;
               ult->celpi = cela->f1; }
            if( cela->f2!=(struct cell*)NULL )
            {  ult->post = (struct fila*)malloc(sizeof(struct fila));
               ult = ult->post;
               ult->celpi = cela->f2; }
            if( cela->f3!=(struct cell*)NULL )
            {  ult->post = (struct fila*)malloc(sizeof(struct fila));
               ult = ult->post;
               ult->celpi = cela->f3; }
            if( cela->f4!=(struct cell*)NULL )
            {  ult->post = (struct fila*)malloc(sizeof(struct fila));
               ult = ult->post;
               ult->celpi = cela->f4; }
            ult->post = (struct fila*)NULL;
         }
```

```
                kdv = kdv->post;
                if( kdv!=(struct fila*)NULL )
                   cela = kdv->celpi;
                free(ini);
                ini = kdv;
             }
          }
       }
       atual = atual->adia;
    }
}


void acumula()
{
   int i;
   double nc, dis;
   double my, ang;

   livre = 's';
   atual = eva.cost;
   while( atual!=(struct atomo*)NULL )
   {
      dis = sqrt( atual->x*atual->x + atual->y*atual->y );
      // Relocação de átomos fujões.
      if( dis > (ndc*5e-4) )
      {
         atual->x = (2*drand48()-1.)*2.12;
         atual->y = (2*drand48()-1.)*2.12;
         atual->xt = atual->x;
         atual->yt = atual->y;
         atual->vx = 0.0; atual->vy = 0.0;
         atual->v2x = 0.0;   atual->v2y = 0.0;
         atual->v3x = 0.0;   atual->v3y = 0.0;
         atual->rp = 'n'; atual->rg = 'n';
         rel++;
         dis = sqrt( atual->x*atual->x + atual->y*atual->y );
      }
      my = sqrt( atual->y*atual->y );
      ang = my/dis;
      // Para o raio maior.
      if( (ang < 0.0871557 || ang > 0.9961947) && atual->rg=='n' )
      {
         nc = dis/5e-4;
         i = (int)nc;
         rmai[i] += 1;
         atual->rg = 'c';
         acmdo_g += 1;
      }
      // Para o raio menor.
      if( ang > 0.6427876 && ang < 0.7660444 && atual->rp=='n')
      {
         nc = dis/5e-4;
```

```
            i = (int)nc;
            rmen[i] += 1;
            atual->rp = 'c';
            acmdo_p += 1;
        }
        if( acmdo_p > n*0.95 && acmdo_g > n*0.95 )
            livre = 's';
        else   livre = 'n';

        atual = atual->adia;
    }
}


void escreve()
{
    int i;
    FILE *grafico;
    FILE *posx, *posy, *velx, *vely;

    grafico = fopen(distribui, "a");
    fprintf(grafico,"SAIDA: %.2g T_fis:%g ", ni/pm+1., ni*dt);
    fprintf(grafico," Npart: %g. RAIO MAIOR:\n", n);
    for(i=0; i<ndc; i++)
        if(rmai[i]>0)
            fprintf(grafico,"%g\t%d\n", (i+1)*dm, rmai[i]);

    fprintf(grafico,"raio menor:\n");
    for(i=0; i<ndc; i++)
        if(rmen[i]>0)
            fprintf(grafico,"%g\t%d\n", (i+1)*dm, rmen[i]);
    fprintf(grafico,"\n");
    fclose(grafico);

    for(i=0; i<ndc; i++)
    {
        rmai[i] = 0;
        rmen[i] = 0;
    }

    posx = fopen("Posix.dat","w");
    posy = fopen("Posiy.dat","w");
    posx = fopen(eixo_x,"w");
    posy = fopen(eixo_y,"w");
    atual = eva.cost;
    while( atual!=(struct atomo*)NULL )
    {
        atual->rp = 'n';
        atual->rg = 'n';

        fprintf(posx,"%g\n", atual->x );
        fprintf(posy,"%g\n", atual->y );
        fprintf(velx,"%g\n", atual->vx );
```

```
        fprintf(vely,"%g\n", atual->vy );

        atual = atual->adia;
    }
    fclose(posx);
    fclose(posy);
    fclose(velx);
    fclose(vely);

    return(0);
}


void organiza()
{
    struct arma *davez, *final;

    atual = eva.cost;
    eva.anc = atual;
    while( atual!=(struct atomo*)NULL )
    {
        atual->prox = atual->adia;
        atual = atual->adia;
    }
    davez = &prim;
    final = prim.seg;
    davez->seg = (struct arma*)NULL;
    while ( final!=(struct arma*)NULL )
    {
        davez = final;
        final = final->seg;
        free(davez->cela);
        free(davez);
    }
}   // Final do código fonte.
```

Referências Bibliográficas

[1] – A. Ashkin, Phys. Rev. Lett., **24**, 156, (1970)

[2] – T. W. Hänsch and A. L. Schawlow, Opt. Commun, **13**, 68, (1975).

[3] – E. L. Raab, M. Prentiss, Alex Cable, Steven Chu and D. E. Pritchard, Phys. Rev. Lett., **59**, 2631, (1987)

[4] – C. Monroe, W. Swann, H. Robinson and C. Wieman, Phys. Rev. Lett., **65**, 1571, (1990).

[5] – J. Dalibard, Opt. Commun. **68**, 203 (1988).

[6] – M. H. Anderson, J. R. Ensher, M. R. Matthews, C. E. Wieman and E. A. Cornell, Science, **269**, 198 (1995).

[7] – D. Felinto, L. G. Marcassa, V. X. Bagnato, and S. S. Vianna, Phys. Rev. A **60**, 2591 (1999).

[8] –L. G. Marcassa, D. Milori, M. Oriá, G. I. Surdutovich, S. C. Zílio and V. S. Bagnato, Braz. J. Phys. **22**, 3 (1992).

[9] –T. Walker, D. Hoffmann, P. Fend and R. S. Williamson III, Phys. Lett. A **163**, 309 (1992).

[10] - M. T. Araújo L. B. Marcassa, S. C. Zílio and V. S. Bagnato, Phys. Rev. A **51**, 4286 (1995).

[11] – D. Felinto, S. S. Vianna, J. Opt. Soc. Am. B, **17**, 681 (2000).

[12] – A. Ashkin, Phys. Rev. Lett., **40**, 729, (1978)

[13] – Y. Castin, H. Wallis, and J. Dalibard, J. Opt. Soc. Am., 6, 2046 (1989).

[14] – A. Ashkin, Phys. Rev. Lett., **25**, 1321, (1970)

[15] – D. E. Pritchard, E. L. Raab, V. Bagnato, C. E. Wieman and R. N. Watts, Phys. Rev. Lett., **57**, 310 (1986).

[16] – V. S. Bagnato, G. P. Lafyatis, A. G. Martins, E. L. Raab, R. N. Ahmad-Bitar and D. E. Pritchard, Phys. Rev. Lett. **58**, 2194 (1987).

[17] – S. L. Gilbert, J. J. Bolinger and D. J. Wineland, Phys. Rev. Lett., **60**, 2022 (1988).

[18] – P. Verkerk, B. Lounis, C. Salomon, C. Cohen-Tannoudji, J.-Y. Courtois and G. Grynberg, Phys. Rev. Lett. **68**, 3861 (1992).

[19] – P. A. Willerns, R. A. Boyd, J. L. Bliss and K. Glibbrecht, **78**, 1660 (1997).

[20] – D. Felinto, Tese de Mestrado, UFPE, 1999.

[21] – V. S. Bagnato, L. B. Marcassa, M. Oriá, G. I. Surdutovich, and S. C. Zílio, Laser Phys. **2**, 172 (1992);

[22] – D. W. Sesko, T. G. Walker e C. E. Wieman, J. Opt. Soc. Am. B **8**, 946 (1991); T. G. Walker, D. W. Sesko e C. E. Wieman, Phys. Rev. Lett. **64**, 408 (1990).

[23] – A. M. Steane, M. Chowdhury and C. J. Foot, J. Opt. Soc. Am. B **9**, 2142, (1992)

[24] – R. J. Cook, Phys. Rev. A, **20**, 224, (1979).

[25] – V. G. Minogin and V. S. Letokhov, Phys. Rep., **73**, 1-65 (1981).

[26] – C. C. A. Feitosa, Tese de Mestrado, UFPE, 1994.

[27] – V. G. Minogin and V. S. Letokhov, New York, Gordon and Breach Science Publishers, 1986.

[28] – A. Yariv, Quantum Eletronics, New York, Jonh Wiley (1989).

[29] – J. J. Sakurai, Modern Quantum Mechanics, Addison Wesley (1995).

[30] – D. J. Wineland, J. J. Bollinger, Wayne M. Itano, and J. D. Prestage, Opt. Soc. Am. B **2**, 1721 (1985).

[31] – J. Dalibard and C. Cohen-Tannoudji, J. Opt. Soc. Am., **6**, 2023 (1989).

[32] – J. Dalibard and C. Cohen-Tannoudji, J. Opt. Soc. Am., **2**, 1707 (1985).

[33] – A. Aspect, J. Dalibard, A Heidmann, C. Salomon and C. Cohen-Tannoudji, Phys. Rev. Lett., **57**, 1688 (1986).

[34] – Y. Castin, H. Wallis, and J. Dalibard, J. Opt. Soc. Am. B, **6**, 2046 (1989).

[35] – D. M. Cláudio, J. M. Martins, Cálculo Numérico Computacional, São Paulo, Atlas S. A., 2º Ed (1994).

[36] – W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Veterling, Numerical Recipes in C, Cambridge, (1990).

[37] – V. V. Mizrahi, Treinamento em Linguagem C, São Paulo, McGraw-Hill, (1990).

[38] – J. Barnes and P. Hut, Nature **324,** 446 (1986).

[39] – S. Colombi, New Astronomy Rev. **45**, 373 (2001).

[40] – A. Beaudoin, S. Huberson, E. Rivoalen, J. Comp. Phys. **186**, 122 (2003).

[41] – J. H. Walther, J. Comp. Phys. **184**, 670 (2003).

[42] – S. Aluru, G. M. Prabhu, J. Gustafson and F. E. Sevilgen, J. of Supercomputing, **12**, 303 (1998); S. Aluru, Ph. D. thesis, Iowa State University (1994).

[43] – Daiichiro Sugimoto, Vistas in Astronomy, **37**, 375 (1993).

[44] – J. O. Burns, C. Loken, K, Roettiger, E. Rizza, G. Bryan, M. L. Norman et all, New Astronomy Rev. **46**, 135 (2002).

[45] – R. Valdarnini, New Astronomy – Article in Press, (2003).

[46] – J. K. Salmon and M. S. Warren, J. Comp. Phys. **111**, 136 (1994).

[47] – J. K. Salmon, Tese de doutorado, Califórnia Institute of Technology, (1990).

[48] – J. Ambrosiano, L. Greengard and V. Rokhlin, Comp. Phys. Commun. **48**, 117 (1988).

[49] – J. S. Bagla, New Astronomy – Article in Press, (2003).

[50] – M. H. Anderson, J. R. Ensher, M. R. Matthews, C. E. Wieman and E. A. Cornell, Science, **268**, 198 (1995).

[51] – L. Greengard and V. Rokhlin, J. Comp. Phys. **135**, 280 (1997).