Universidade Federal de Campina Grande

Centro de Educação e Saúde

Unidade Acadêmica de Física e Matemática

Everaldo Ferreira da Silva

Átomos de Césio em um Melaço Óptico: Uma Possibilidade de Resfriar Átomos

Cuité - PB

2016

Everaldo Ferreira da Silva

Átomos de Césio em um Melaço Óptico: Uma Possibilidade de Resfriar Átomos

Monografia apresentada ao Curso de Licenciatura em Física da UFCG/CES, como requisito para a obtenção parcial do grau de LICENCIADO em Física.

Orientador: Prof. Dr. Pedro Chaves de Souza Segundo

Cuité - PB

2016

FICHA CATALOGRÁFICA ELABORADA NA FONTE
Responsabilidade Msc. Jesiel Ferreira Gomes – CRB 15 – 256

S586a Silva, Everaldo Ferreira da.

Átomos de césio em um melaço óptico uma possibilidade de resfriar átomos.. / Everaldo Ferreira da Silva. – Cuité: CES, 2017.

56 fl.

Monografia (Curso de Licenciatura em Física) – Centro de Educação e Saúde / UFCG, 2017.

Orientador: Pedro Chaves de Souza Segundo.

1. Melaço óptico. 2. Resfriamento atômico. 3. Simulação computacional. 4. Bibliotecas GSl.. I. Título.

Biblioteca do CES - UFCG CDU 539.18

Everaldo Ferreira da Silva

Átomos de Césio em um Melaço Óptico: Uma Possibilidade de Resfriar Átomos

Monografia apresentada ao Curso de Licenciatura em Física da UFCG/CES, como requisito para a obtenção parcial do grau de LICENCIADO em Física.

Trabalho aprovado. Cuité - PB, 20 de outubro de 2016:

Prof. Dr. Pedro Chaves de Souza Segundo
Orientador

Prof. Dr. Heron Neves de Freitas
Convidado 1

Prof. Dr. Fábio Ferreira de Medeiros
Convidado 2

Cuité - PB
2016

Dedico este trabalho, exclusivamente,
aos meus amigos leais e verdadeiros,
e à minha família.

Agradecimentos

A todos que direta ou indiretamente fizeram parte da minha formação acadêmica, e aos que torceram por mim, o meu muito obrigado.

"Tente. E não diga
que a vitória está perdida.
Se é de batalhas
que se vive a vida."
(Raul Seixas)

Resumo

Neste trabalho abordamos teoricamente o Melaço Óptico, e utilizando átomos de Césio, demonstramos que é possível resfriá-los. O resfriamento atômico é um conteudo de extrema importância em física, pois engloba os fundamentos que antecedem as técnicas de aprisionamento de átomos, onde esta tem importantes aplicações físicas. Utilizamos como base, a ideia que a interação do átomo com um feixe de radiação eletromagnética modifica o momento. Esse efeito é dado por uma força média que altera a velocidade do átomo. A força é contrária ao sentido do feixe. Utilizamos a configuração de feixes contrapropagantes para reduzir a velocidade dos átomos. O estudo da dinâmica que fizemos é sobre a velocidade dos átomos na interação desses feixes via cálculo numérico para resolver EDO da relação $F = m\frac{d^2x}{dt^2}$. Fizemos o uso da linguagem de programação C++ aliada as potentes bibliotecas GSL, para fazermos uma simulação computacional. O entendimento dos métodos numéricos foram implementados junto com aplicações nos osciladores harmônicos. Utilizamos a expressão da força da pressão de radiação no melaço óptico, calculamos a velocidade e a aceleração em função do tempo. Estudamos a programação numérica para resolução de EDOs usando o GSL e verificamos o seu funcionamento para o caso de oscilador harmônico em que podemos comparar com os resultados analíticos. Fizemos comparações para diversas condições e sempre observando na dinâmica a redução da velocidade.

Palavras-chave: Melaço Óptico. Resfriamento atômico. Simulação computacional. Bibliotecas GSL.

Sumário

Lista de ilustrações

1 Introdução

Desde épocas remotas, o homem vem tentando desvendar detalhes sobre o meio em que vive, o comportamento da natureza, muitas vezes intrigantes, faz com que alguns homens dediquem suas curtas vidas à uma busca implacável na tentativa de explicar alguns fenômenos da natureza, tarefa muitas vezes impedida pelo senso comum. Podemos destacar, como exemplo: Copérnico. De acordo comNussenzveig(1998), Copérnico defendeu o sistema heliocêntrico, onde a terra descrevia órbitas circulares em torno do sol, e não o oposto, como afirmava a igreja, o ente mais poderoso daquela época.

Neste trabalho, vamos abordar um dos aspectos da natureza, a interação matéria com a radiação. O foco do trabalho é o efeito mecânico dessa interação: variação de momento. Comprovaremos através de uma simulação numérica, que é possível resfriar átomos, será desenvolvido apenas, o caso para uma dimensão.

Nessa introdução do trabalho, vamos mostrar uma abordagem sucinta, sobre a evolução histórica dos conceitos relativos ao entendimento desse trabalho, buscando sempre tratar os conteúdos de forma simplificada. Também faremos uma breve descrição dos capítulos.

Agora, iremos abordar, como evoluiu com tempo, os conceitos de pressão da radiação, desde o contexto clássico de Kepler, passando por Einstein, até composição teórica e experimental dos ganhadores do prêmio Nobel de Física de 1997, quando esses descreveram de forma coesa o entendimento das técnicas de resfriamento e aprisionamento atômico.

De acordo comSEGUNDO(2000), o estudo da interação da luz com a matéria tem seus primórdios na física clássica, onde Kepler afirmou que a deflecção das caudas dos cometas próximos do Sol seria causada pela pressão dos raios do Sol, e as partículas que cercam os cometas seriam empurradas pela radiação para longe dele. De acordo com essa mesma referência, evidências experimentais obtidas por Lebedev comprovaram a afirmação sugerida por Maxwell, que não só afirmava a existência da pressão da luz, como quantificava o seu valor. Já em 1917, em seu artigo sobre a radiação de corpo negro, Einstein previu que o *momentum* é transferido na absorção e emissão da luz. Dessa forma, podemos evidenciar que, enquanto Maxwell mostrou classicamente, que o campo eletromagnético exerce uma pressão igual a sua energia por unidade de volume, Einstein destacou a essência dessa força quanticamente.

De acordo com Einstein, um *quantum* de luz, também denominado de *fóton*, tem energia $h\nu$. Caso um átomo absorva um *fóton* receberá um momento $P = h\nu/c = h/\lambda$ na direção do movimento da luz, onde h é a constante de Planck, ν a frequência, c a

velocidade da luz e λ o comprimento de onda da luz. Sendo assim, levando em consideração a conservação do momento, temos que, se um átomo emitir um *fóton* este recuará na direção oposta a esse *fóton*, alterando o momento, e assim provocando mudanças no movimento atômico (HE,2009).

Outro avanço importante, no contexto da pressão da radiação, deu-se quando em 1933, O. Frisch evidenciou experimentalmente mais uma prova da existência da pressão da luz. Ele verificou um desvio de 0,01 mm em um feixe de átomos de sódio iluminado com uma lâmpada de mesmo elemento (SEGUNDO,2000).

A partir da década de 60 surgiram os *lasers*, e com eles a possibilidade da manipulação e controle dos átomos. A ideia de utilizar *lasers* para defletir feixes de átomos foi proposta inicialmente por A. Ashkin e seus colaboradores, em 1970. Utilizando o *laser* para estudos da pressão de radiação, esse grupo conseguiu obter sucesso fazendo pequenas esferas dielétricas levitarem (SEGUNDO,2000). O laser, pode então ser tratado, como uma ferramenta diferenciada e de grande eficácia no estudo da pressão de radiação, seu uso é destacado pelo fato da sua capacidade de emitir luz do tipo colimada[1] e de alta potência, com feixes coerentes podendo ser sintonizado com frequência e linhas bem estreitas (HE, 2009).

Posteriormente surgiu a proposta de utilizar a luz *laser* para desacelerar e resfriar átomos, foi inicialmente sugerida de forma independente por T.W. Hänsch e A.L. Schawlow e de D.J. Wineland e H. Dehmelt (SEGUNDO,2000). Apesar da independência dos dois grupos eles traduziam a mesma situação, ou seja, a ação mecânica da radiação *laser* com uma frequência muito próxima à correspondente transição atômica, provocando uma diminuição na energia cinética dos átomos. Para Hänsch e Schawlow, o resfriamento é consequência do efeito Doppler, onde sobre a pressão de radiação, o átomo torna-se dependente da velocidade[2], esse caso pode ser visto com mais naturalidade para o esfriamento de átomos livres. Já para Wineland e Dehmelt o fato da energia ser reduzida é consequência de um processo Raman[3], essas conclusões foram obtidas por esse grupo em 1975. Este caso tem mais utilidade para átomos fortemente ligados em um poço de potencial (SEGUNDO,2000).

O primeiro desfecho positivo obtido experimentalmente, ocorreu por volta de 1978, quando Wineland, Drullinger e Walls, utilizando uma armadilha eletromagnética de *Penning* conseguiram resfriar uma nuvem de íons de magnésio Mg (HE,2009).

O período de 1982 a 1989 foi o que mais se destacou no avanço das técnicas de resfriamento de átomos neutros. Em 1982 vieram os primeiros resultados satisfatórios,

---

1 Ocorre quando os raios de luz são quase paralelas, o espalhamento ocorre de maneira lenta no decorrer de sua propagação.

2 Trataremos essa afirmação com mais detalhes no Capítulo3.

3 Transições do nível de energia cinética superior pra inferior são mais prováveis que o inverso.

W. Phillips e H. Metcalf conseguiram desacelerar átomos de sódio fazendo o uso de um feixe de laser contrapropagante em um campo eletromagnético não homogêneo. Em 1985, utilizando apenas feixes de luz *laser*, um grupo liderado por S. Chu, conseguiram resfriar átomos de sódio. Esse processo consistia em utilizar três *lasers* contrapropagantes, um em cada eixo. Esse processo foi batizado com o nome melaço óptico (*optical molasses*), nome utilizado pela razão que a força encontrada no cruzamento dos *lasers* onde se encontra uma nuvem atômica resfriada é altamente viscosa. A força média para cada um dos pares de feixes contrapropagantes é

$$\mathbf{F} = \frac{\Gamma\Omega^2\bar{h}\mathbf{k}}{4(\Delta - \mathbf{k}.\mathbf{v})^2 + \Gamma^2 + 2\Omega^2} \tag{1.1}$$

e

$$\mathbf{F} = -\frac{\Gamma\Omega^2\bar{h}\mathbf{k}}{4(\Delta + \mathbf{k}.\mathbf{v})^2 + \Gamma^2 + 2\Omega^2}. \tag{1.2}$$

Onde Γ é a taxa de decaimento para o estado fundamental, $\Omega = \frac{\mu E(\mathbf{r},t)}{\bar{h}}$ é a frequência de Rabi, $\bar{h}$ é constante reduzida de Planck, $\mathbf{k}$ valor absoluto do vetor de onda, $\mathbf{v}$ o valor da velocidade do átomo, Δ é a diferença $\Delta = \omega - \omega_0$ onde ω_0 representa a frequência de ressonância do átomo (GOMES N. D.; BAGNATO,2014).

Dois anos mais tarde esse mesmo grupo criou a primeira armadilha magneto-ótica (*MOT-Magneto-optical trap*). Essa armadilha não só era capaz de resfriar átomos como também podia aprisioná-los; o aprisionamento consiste em combinar o arranjo do melaço com o efeito de campo eletromagnético externo.

Finalmente a formulação teórica feita por J. Dalibard and C. Cohen-Tannoudji em 1989 possibilitou o entendimento dessa técnica de resfriamento e aprisionamento. Esses trabalhos renderam a seus precursores, Steven Chu, Cohen-Tannoudji e William, o prêmio Nobel de Física de 1997. Vale destacar que, a produção de átomos frios devido as técnicas de resfriamento e aprisionamento, tem como efeito a perda de energia cinética dos átomos sendo estes confinados em um ambiente com poucas pertubações (HE,2009).

Esses conceitos descritos anteriormente, tem caratér introdutório, e é um pré-requisito fundamental para o entendimento geral desse trabalho, e em particular do Capítulo4, onde serão abordado as noções básicas da teoria do melaço óptico e serão desenvolvidas algumas simulações numéricas sobre o referido tema.

Neste trabalho, simulamos numericamente o melaço óptico em uma dimensão, isto é a interação de dois feixes contrapropagantes com um gás de átomos. Para realizar o trabalho, dividimos ele em dois núcleos principais: a teoria da pressão da radiação, que detalha os efeitos da luz sobre os átomos, e a simulação numérica, que engloba a teoria dos métodos numéricos junto com a preparação dos software's para a realização da simulação.

Para entender os conceitos referentes a interação dos átomos com a luz, será feita uma abordagem teórica no Capítulo3, mostrando, desde os princípios da teoria atômica de

Demócrito até os argumentos de Bohr, concluindo com os conceitos da emissão e absorção de luz pelos átomos.

Como vimos anteriormente, o efeito da interação dos feixes de radiação sobre o átomo em um melaço pode ser dado por uma força, assim, a dinâmica do átomo é dada por uma EDO:

$$m\frac{d^2x}{dt^2} = F.$$

E para resolvermos essa dinâmica, utilizaremos o método numérico Runge-Kutta. Estudaremos esse método no Capítulo2, onde mostraremos a aplicação em um modelo que temos o resultado algébrico: oscilador harmônico simples e o amortecido.

No Capítulo4, estudaremos através de resultados de simulações numéricas a dinâmica do movimento atômico do átomo de césio interagindo com dois feixes contrapropagantes. A configuração estudada é a da formação de um melaço ótico unidimensional.

No capítulo5, abordamos o desenvolvimento e os objetivos alcançados por esse trabalho. Destacamos formas de aproveitamento desse, para um possível processo de continuidade, finalizando-o.

2 RESOLVENDO EDO's NUMERICAMENTE

2.1 Introdução

Neste capítulo, vamos entender alguns conceitos relativos às equações diferenciais. Iremos aplicar métodos numéricos para solucionar algumas EDO's.

Para o estudo dos métodos numéricos, primeiramente, iremos introduzir os métodos de Euler, para posteriormente entendermos os métodos de Runge-Kutta. Na seção2.3, faremos aplicações dos métodos de Euler, esta será uma abordagem introdutória para o entendimento das seções subsequentes.

Utilizaremos as ferramentas computacionais expostas emA, aliada à linguagem de programação C++.

Esse capítulo é crucial para um êxito positivo no desenvolvimento desse trabalho, pois aqui será feito testes de verificação e de estabilidade do sistema. Esses testes consistem em resolver numericamente algumas equações dos osciladores harmônicos comparando-as com os resultados algébricos. Contudo, é fundamental alguns conceitos introdutórios, relativos aos osciladores harmônicos, uma abordagem sobre esse tópico é feita na Seção 2.5.

Faremos aplicações dos métodos de Runge-Kutta de quarta ordem, no caso dos osciladores harmônicos, e finalizamos utilizando a poderosa biblioteca GSL, que será utilizada nas simulações numéricas do Capítulo4.

2.2 Equações diferenciais

O objetivo desta seção é apenas dar uma contrapartida para o entendimento de alguns tipos de equações diferenciais, sendo abordados aqui, conceitos mais superficiais e de fácil entendimento sobre esse assunto. Esse tema tem um nível de importância consideravelmente gigantesco, para o tratamento dos enunciados subsequentes, onde serão introduzidos os princípios dos métodos numéricos, acompanhados de algumas simples aplicações.

Para um entendimento minucioso dos conceitos referentes às equações diferenciais, é necessário um trabalho mais intenso e detalhado, no entanto aqui será feito apenas uma introdução simples, detalhando os tópicos mais importantes que serão utilizados nos conteúdos que podem ser associados à esse tema. Seguindo os conceitos definidos

porBoyce e DiPrima Richard C(1969), temos que as ED's podem ser classificadas em Equações diferencias ordinárias (EDO's) e equações diferenciais parciais (EDP's). Para o estudo das EDO's existe a necessidade de saber se a função depende de uma única variável independente, caso a função dependa de várias variáveis independentes a equação é dita equação diferencial parcial, e uma análise desse tipo de equações pode ser considerado irrelevante no desenvolvimento deste trabalho, aqui será dado ênfase apenas para o estudo das EDO's. Como exemplos, temos que, a equação que rege o movimento de um pêndulo simples é dita uma EDO,

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin\theta = 0, \tag{2.1}$$

Já a *equação de onda* é denominada uma EDP,

$$a^2 \frac{\partial^2 u(x,\, t)}{\partial x^2} = \frac{\partial^2 u(x,\, t)}{\partial t^2}. \tag{2.2}$$

Também para esta seção, é indispensável o entendimento inerente a ordem de uma equação diferencial,Boyce e DiPrima Richard C(1969) trata que a ordem de uma equação diferencial é a ordem da derivada de maior ordem que aparece na equação. Como um exemplo geral de uma equação de ordem n, (BOYCE; DIPRIMA RICHARD C,1969) apresenta

$$F(t, y, y^t, ..., y^{(n)}) = 0. \tag{2.3}$$

Desta forma, para fixar a compreensão da equação anterior, pode-se usar o seguinte exemplo:

$$y^{ttt} + 2e^t y^{tt} + yy^t = t^4, \tag{2.4}$$

de onde podemos concluir que esta é uma equação diferencial de terceira ordem para um determinado $y = u(t)$.

Ainda se tratando da classificação das equações diferenciais, podemos evidenciar mais um tópico fundamental das ED's, que é o fato delas serem lineares ou não-lineares. Para uma compreensão coesa, pode-se utilizar duas equações já abordadas nesta seção, Boyce e DiPrima Richard C(1969) afirma que a equação2.3é considerada linear, já a equação2.4não se encaixa nos quesitos da definição, logo ela é não-linear. O fato da equação2.4ser não-linear decorre do termo yy^t. Também como exemplo, pode-se observar que, o problema do pêndulo simples também é uma equação não-linear, equação2.1. Esse argumento é sustentado pelos parâmetros dependentes na equação, no termo $\frac{g}{l}\sin\theta$.

Existem diversos tipos de Equações Diferenciais, e no que diz respeito as soluções, temos que, um método de solução pode ser eficaz para um certo tipo de *ED's*, mas para

outro pode ser desprezível,Boyce e DiPrima Richard C(1969) apresenta diversos métodos de soluções, dentre eles: método do Fator Integrante, Equações Separáveis, Equação diferencial exata, Redução de Ordem, Método da variação de parâmetros etc.

Para uma equação diferencial de primeira ordem,Zill(2001) afirma que se

$$\frac{dy}{dx} = f(x, y), \tag{2.5}$$

podemos submeter uma condição inicial $y(x_0) = y_0$ a essa equação, onde x_0 é um número no intervalo I e y_0 é um número real arbitrário. Então, esse problema onde temos que resolver a equação2.5sujeito a uma condição inicial $y(x_0) = y_0$, é denominado Problema de valor inicial, e se a solução não depender de parâmetros arbitrários temos a denominada solução particular.

Como já citado existem vários métodos para soluções de equações diferenciais, e descrever todos esses procedimentos aqui tornaria esse trabalho exorbitante e enfadonho, e certamente fugiria do verdadeiro objetivo deste capítulo, mas as referências (ZILL,2001), (BOYCE; DIPRIMA RICHARD C,1969) e (NAGLE EDWARD B. SAFF,2012) podem suprir de forma suficientemente considerável qualquer dúvida relacionada as ED's, que consequentemente podem surgir no ato da leitura desse trabalho. No entanto, é de imensa importância esclarecer que a compreensão da temática relacionada às equações ordinárias de primeira ordem será de grande valor nas seções2.3e2.4. Sua importância também merece destaque pelo fato de servir como mecanismo de partida para o entendimento dos movimentos oscilatórios.

Nessa seção, podemos "transitar"pelo mundo das EDO's, entendendo alguns conceitos relativos quanto à sua classificação: tipo, ordem e linearidade. Esses conceitos serão importantes na aplicação dos métodos numéricos.

2.3 Método de Euler e aplicações

A partir de agora, vamos começar a entender alguns conceitos importantes sobre cálculo numérico, partido do método considerado como o mais simples: método de Euler. Após entendermos esse método, iremos utiliza-lo em algumas aplicações simples.

Os métodos de Euler não são usados com muita frequência, porém sua facilidade de manipulação é de grande valia quando o objetivo é ilustrar e dar uma base didática para métodos mais avançados, como o método de Runge-Kutta, por exemplo. Assim, de início podemos começar a entender esse método tão simples. É importante esclarecer que o propósito desta seção não é fazer cálculos complicados, e de nível avançado, mas apenas introduzir e dar uma base teórica para alguns conceitos do *cálculo numérico*.

2.3.1 Método de Euler

O método de aproximação de Euler também chamado método a linha tangente (NAGLE EDWARD B. SAFF,2012) consiste em obter soluções aproximadas para um problema de valor inicial

$$y^t = f(x, y), \qquad y(x_0) = y_0. \tag{2.6}$$

É sabido que esse método pode ser falho em alguns pontos, porém as interações do procedimento converge para soluções verdadeiras, como afirmaNagle Edward B. Saff (2012). O método de Euler é dado por:

$$\begin{aligned} x_{n+1} &= x_n + h, \\ y_{n+1} &= y_n + hf(x_n, y_n), \qquad n = 0, 1, 2, \ldots \end{aligned} \tag{2.7}$$

Em resumo esse processo utiliza como ponto de partida o valor y_0 para resolver $y_1 = y_0 + hf(x_0, y_0)$, depois y_1 para determinar $y_2 = y_1 + hf(x_1, y_1)$, e desta forma essa técnica se repete, o h é definido como um número positivo fixo, denominado tamanho de passo.

2.3.2 Aplicações do método de Euler

Para demonstrar que não é necessário ser um especialista na área de programação para utilizar o método de Euler, vamos utilizá-lo valendo-se de alguns problemas simples de mecânica. Será utilizado um conhecimento rudimentar de matemática básica, e um pouco de domínio de planilhas, o ***LibreOffice Calc*** em particular[1].

A princípio, temos que o método numérico de Euler resumi-se a equação2.7, e partindo dessa equação, podemos aplicá-la para o *movimento uniforme "MU"*. Podemos encontrar os conceitos referentes ao *"MU"* em (NUSSENZVEIG,1998), onde é afirmado que a velocidade pode ser expressa por:

$$v = \frac{dx}{dt}, \tag{2.8}$$

essa equação pode ser representada por uma equação diferencial do tipo:

$$f(t, x) = \frac{dx}{dt}. \tag{2.9}$$

Dessa forma, iremos utilizar o *método de Euler*, empregando *LibreOffice Calc*, para obtermos o resultado do cálculo. Primeiramente será necessário definir a velocidade e a posição inicial. Dessa forma, vamos utilizar 50 m/s para o valor da velocidade, e $x_0 = 0,0$

[1] O ***LibreOffice Calc*** pode ser substituído pelo *excel* ou qualquer outro similar.

m para posição inicial. Os demais parâmetro são: tempo inicial $t_0 = 0,0$ s e tamanho de $h = 0,1$. Agora utilizando a lei horária do movimento retilíneo uniforme, que é dada por Nussenzveig(1998), como $x(t) = x_0 + v(t - t_0)$ [2].

Após inserir os dados definidos anteriormente, junto com as expressões designadas no *LibreOffice Calc*, finalmente podemos fazer os cálculos e comparar os resultados, numérico e analítico. Na Figura1podemos verificar o resultado obtido pelo *LibreOffice Calc*. Sendo que as linhas em cor cinza, correspondem aos resultados numéricos e analíticos: a linha **s_n** representa os **resultados numéricos** e a linha **S_n=s_0+v*t_n** os **resultados analíticos**.

Dados								
t_0	0,000							
s_0	0,000							
h	0,100							
f(t_n, s_n)	50,00							
n	1,00	2,00	3,00	4,00	5,00	6,00	...	25,00
t_n	0,00	0,10	0,20	0,30	0,40	0,50	...	2,40
s_n	0,00	5,00	10,00	15,00	20,00	25,00	...	120,00
f(t_n, s_n)	50,00	50,00	50,00	50,00	50,00	50,00	...	50,00
S_n=s_0+v*t_n	0,00	5,00	10,00	15,00	20,00	25,00	...	120,00

Figura 1 – *Resultados numérico e analítico do movimento retilíneo uniforme, obtido através dos métodos de Euler* [3].

Observamos que a aproximação oriunda do *método de Euler* para problema do *"MU"*, surtiu efeitos positivos, pois dentro do intervalo que pode ser visualizado na Figura 1, os resultados numéricos são muito parecidos com os resultados analíticos.

Como segundo exemplo, podemos utilizar o *MUV*. Como podemos ver em (NUSSENZVEIG,1998), para esse caso, a velocidade deixa de ser constante, e um novo termo deve ser empregado, a *aceleração*. Para esse tipo de movimento a aceleração deve ser constante e essas variáveis são definidas como:

$$v = \frac{dx}{dt}, \qquad a = \frac{dv}{dt}. \tag{2.10}$$

Dessa forma, podemos concluir que, se

$$a = \frac{d}{dt}\left(\frac{dx}{dt}\right) \Rightarrow a = \frac{d^2x}{dt^2}, \tag{2.11}$$

temos que essa equação é uma equação linear de segunda ordem, e dessa forma podemos utilizar o método de *redução de ordem* para solucionarmos2.2. Simplificando, esse método consiste em quebrar uma EDO de segunda ordem em duas EDO's de primeira ordem. O

[2] No *LibreOffice Calc* essa expressão será dada por $x_n = x_0 + vt_n$.
[3] Essa figura corresponde a uma captura de tela obtida de uma tabela do *LibreOffice Calc*.

procedimento de quebra de equações pode ser utilizado em:

$$a = \frac{d^2x}{dt^2} \Rightarrow x^{tt} = a. \tag{2.12}$$

Podemos fazer $x^t = v \Rightarrow dx/dt = v$. Logo $v^t = a \Rightarrow dv/dt = a$. Essas variáveis devem ser reorganizadas para que possam ser inseridas no *LibreOffice Calc*. Em resumo, para o método de Euler temos que as equações *MUV* podem ser definidas como:

$$\frac{dv}{dt} f(t_n, v_n), \qquad \frac{dx}{dt} g(t_n, x_n),$$

$$t_{n+1} = t_n + h,\ v_{n+1}$$

$$= v_n + h.f(t_n, v_n),$$

$$x_{n+1} = x_n + h.g(t_n, x_n).$$

Observemos que existe uma dependência entre essas funções. Sendo assim, o t_{n+1} pode ser utilizado tanto para $f(t_n, v_n)$ como para $g(t_n, x_n)$. Para esse caso, aceleração é dada como 2 m/s^2, e para os outros parâmetros temos: t_0=0,0 s, x_0=0,0 m, v_0=0,0 m/s e h=0,001. A resposta numérica dada pelo *método de Euler* é expressado na Figura2. Sendo que as linhas em cor cinza, correspondem aos resultados numéricos e analíticos: a linha **s_n** representa os **resultados numéricos** e a linha **S_n=s_0+v*t_n** os **resultados analíticos**.

Dados								
t_0	0,000							
s_0	0,000							
v_0	0,000							
h	0,001							
f(t_n, s_n) = a	2,00							
g(t_n, s_n) =v_n	v_n							
a_0	0,000							
n	0,000	1,000	2,000	3,000	4,000	5,000	...	24,000
t_n	0,000	0,001	0,002	0,003	0,004	0,005	...	0,024
g(t_n, s_n) =v_n	0,000	0,002	0,004	0,006	0,008	0,010	...	0,048
s_n	0,00E+000	0,00E+000	2,00E-006	6,00E-006	1,20E-005	2,00E-005	...	5,52E-004
f(t_n, s_n) = a	2,000	2,000	2,000	2,000	2,000	2,000	...	2,000
S=S_0+v_0*t+(a*t^2)/2	0,00E+000	1,00E-006	4,00E-006	9,00E-006	1,60E-005	2,50E-005	...	5,76E-004

Figura 2 – *Resultados numérico e analítico do movimento uniforme variado, obtido através dos métodos de Euler* [4].

Na Figura2, podemos observar que os resultados para a posição não são exatamente iguais, e a diferença pode aumentar ou diminuir, isso depende do número de passos h escolhido. Foi verificado que $h = 0,001$ é o valor que pode produzir uma menor diferença entre os resultados numéricos e analíticos.

[4] Essa figura corresponde a uma captura de tela obtida de uma tabela do *LibreOffice Calc*.

Então, concluímos que, quando o nível de complexidade das equações aumenta, a comparação dos resultados dos cálculos numérico com analítico, utilizando o método de Euler começa a falhar. Assim, esses exemplos servem para esclarecer que, para alguns problemas os *métodos de Euler* não funcionam adequadamente. Daí então, devemos utilizar outros métodos, na próxima seção introduziremos os *métodos de Runge-Kutta*.

2.4 Métodos de Runge-Kutta

Entre muitos métodos numéricos, um que merece destaque são os métodos de Runge-Kutta, a notoriedade desse argumento, reside do fato da sua simplicidade e do extenso meio em que esse método é utilizado. Vamos destacar nesta seção uns dos vários procedimentos utilizados pelo Runge-Kutta.

Vale esclarecer que nesta seção não será feita nenhuma aplicação desse método, visto que isso será feito no capítulo referente aos testes do sistema, onde será utilizado conhecimentos de programação C++. O desenvolvimento desse trabalho até aqui não oferece aparato necessário para utilização de programas C++.

A seguir será expressado de maneira simples, e em ordem sequencial, os métodos de *Runge-Kutta* de primeira à quarta ordem.

De acordo comRuggiero M.A.G.(1988) o princípio desse método consiste em aproveitar as qualidades dos *métodos da série de Taylor* [5], e eliminar um defeito no cálculo das derivadas de $f(x, y)$, sendo que esse defeito torna os métodos de séries de Taylor computacionalmente inaceitáveis.

Inúmeras referências bibliográficas traduz os métodos de Runge-Kutta, segundo Nayara(2012) para obter-se o ***método de Runge-Kutta de primeira ordem***, é necessário o uso do polinômio de Taylor com resto considerando-se que a função $y(x)$ tenha $k+1$ derivadas contínuas em um intervalo aberto com a até x, logo:

$$y(x) = y(a) + y'(a)\frac{x-a}{1!} + \ldots + y^{(k)}(a)\frac{(x-a)^k}{k!} + y^{(k+1)}(c)\frac{(x-a)^{(k+1)}}{(k+1)!}, \qquad (2.13)$$

onde c é um número entre a e x. De acordo comNayara(2012), substituindo a por x_n e x por $x_{n+1} = x_n + h$ na equação2.13, temos

$$y(x_{n+1}) = y(x_n + h) = y(x_n) + hy'(x_n) + \frac{h^2}{2!}y''(x_n) + \ldots + \frac{h^{k+1}}{(k+1)!}y^{(k+1)}(c). \qquad (2.14)$$

Fazendo $k = 1$ e considerado o resto $\frac{h^2}{2!}y''(c)$ muito pequeno, concluímos que

$$y_{n+1} = y_n + hy'_n = y_{n+1} = y_n + hf(x_n, y_n) \qquad (2.15)$$

[5] Uma conferida em (RUGGIERO M.A.G.,1988) *(Capítulo 8)* pode sanar dúvidas referente aos *métodos de série de Taylor*.

Esse é o ***método de Runge-Kutta de primeira ordem*** também chamado de ***método de Euler básico***.

O procedimento para obtermos o ***método de Runge-Kutta de segunda ordem*** segue-se, primeiro fazendo $k = 2$ na equação2.14,

$$y(x_{n+1}) = y(x_n + h) = y(x_n) + hy^{t}(x_n) + \frac{h^2}{2!}y^{tt}(x_n) + \frac{h^3}{3!}y^{ttt}(c). \qquad (2.16)$$

Como podemos ver emNayara(2012), utilizando-se de algumas manipulações algébricas, com algumas substituições e algumas aproximações adequadas, conclui-se que o método de Runge-Kutta de segunda ordem corresponde ao método de Euler melhorado, dessa forma para o método de Runge-Kutta de segunda ordem temos,

$$y_{n+1} - y_n = \frac{h}{2}[f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n))]. \qquad (2.17)$$

O Procedimento para obter o ***método de Runge-Kutta de terceira ordem*** é semelhante ao usado para o de segunda ordem, dessa forma, como pode ser observado em Nayara(2012), basta fazer $k = 3$ na equação2.14, e como exemplo, podemos citar um método de terceira ordem muito conhecido;

$$\begin{aligned}
&y_{n+1} = y_n + \tfrac{1}{6}(k_1 + 4k_2 + k_3),\ n = 0, 1, \ldots, m-1\\
&k_1 = f(x_n, y_n)\\
&k_2 = f\left(x_n + \tfrac{h}{2}, y_n + \tfrac{h}{2}k_1\right)\\
&k_3 = f(x_n + h, y_n + 2hk_2 - hk_1).
\end{aligned}$$

Para o ***método de Runge-Kutta de quarta ordem***, pode-se fazer $k = 4$ na equação2.14, dessa forma encontra-se a fórmula geral para um dos *métodos de Runge Kutta de quarta ordem* (NAYARA,2012),

$$y_{n+1} = y_n + ak_1 + bk_2 + ck_3 + dk_4, \qquad (2.18)$$

Com

$$y(x_{n+1}) = y_n + h\phi(x_n, y_n; h)$$

e

$k_1 = hf(x_n, y_n)$

$k_2 = hf(x_n + \alpha_1 h, y_n + \beta_1 k_1)$

$k_3 = hf(x_n + \alpha_2 h, y_n + \beta_2 k_1 + \beta_3 k_2)$

$k_4 = hf(x_n + \alpha_3 h, y_n + \beta_4 k_1 + \beta_5 k_2 + \beta_6 k_3)$.

Temos concordância com um polinômio de Taylor de quarto grau

$$p_4(x) = y(a) + y^{t}(a)(x-a) + \frac{y^{tt}(a)}{2!}(x-a)^2 + \frac{y^{ttt}(a)}{3!}(x-a)^3 + \frac{y_{(4)}(a)}{4!}(x-a)^4. \qquad (2.19)$$

Como resultado temos onze equações em treze incógnitas, levando a conclusão de infinitas soluções. Podemos citar como exemplo o *métodos de Runge Kutta de quarta ordem* mais usado popularmente;

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$

$$\begin{aligned} k_1 &= hf(x_n, y_n) \\ k_2 &= hf\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1\right) \\ k_3 &= hf\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2\right) \\ k_4 &= hf(x_n + h, y_n + k_3). \end{aligned} \tag{2.20}$$

Logo k_2 depende de k_1, k_3 de k_2 e k_4 de k_3, temos ainda que k_2 e k_3 englobam aproximações às inclinações no ponto médio do intervalo entre x_n e x_{n+1}. Sendo assim uma atenção extra é necessário.

Podemos observar que os métodos de *Runge-Kutta* não precisam utilizar derivadas de ordem superior a um, isso facilita seu uso.

2.5 Introdução teórica dos osciladores harmônicos simples e amortecidos

Diante da necessidade de entendermos exemplos de aplicações dos métodos de Runge-Kutta e testarmos o sistema, podemos unir essas duas situações em uma só. Antes de tudo, existe a necessidade de entendermos alguns modelos teóricos dos osciladores harmônicos, nesta seção abordaremos de forma simplificada os osciladores harmônicos simples e amortecidos.

2.5.1 Oscilador harmônico simples (OHS)

No dia-a-dia, podemos observar frequentemente algumas situações curiosas, por exemplo: O movimento de uma corda, onde esta é presa em uma haste fixa, quando um operador faz um rápido movimento para cima e para baixo, podemos perceber uma ondulação deslocando-se sobre a corda. Uma pedra solta em direção perpendicular a superfície de um lago, observamos também outra pertubação no meio. Assim como em outras inúmeras situações análogas, temos movimentos em dois sentidos, alternadamente em torno de uma posição de equilíbrio. Esses fatos simples representam situações características das ondas mecânicas. Outra classe de ondas, são as ondas eletromagnéticas, onde podemos citar como exemplo: ondas de rádio, raio X, entre outras.

Existem uma gama de exemplos relacionados aos movimentos oscilatórios dos mais simples aos mais complexos. No entanto,Symon(1996) trata o oscilador harmônico simples como oscilador harmônico ou linear, e segundo essa referência, esse é o mais importante movimento unidimensional e mais fácil de resolver. O exemplo mais elementar, se trata de uma massa m presa a uma mola, inicialmente em repouso, onde essa massa se limita a deslocar-se no eixo x. A força que rege esse movimento e dita força restauradora, definida como:

$$F = -kx, \tag{2.21}$$

onde k é denominado como constante elástica da mola. Podemos ainda encontrar em Nussenzveig(1998) a afirmação que a dinâmica de uma partícula pode ser determinada pela segunda lei de Newton, que pode ser expressa por:

$$\mathbf{F} = m\mathbf{a}, \tag{2.22}$$

sendo $\mathbf{F}$ a força atuante na partícula, m é massa e $\mathbf{a}$ a aceleração. Como a velocidade e aceleração podem se dada, respectivamente por:

$$\mathbf{v} = \frac{d\mathbf{x}}{dt}, \qquad \mathbf{a} = \frac{d^2\mathbf{x}}{dt^2}, \tag{2.23}$$

podemos concluir que

$$m\frac{d^2\mathbf{x}}{dt^2} = -k\mathbf{x}. \tag{2.24}$$

Observamos algumas variáveis dependentes nesta última expressão, a posição da partícula x, a velocidade $\vec{v}$ e o tempo t. Ainda podemos reformular essa última expressão, dividindo-a pela massa m, dessa forma temos,

$$\frac{d^2x}{dt^2} = -\frac{kx}{m} \quad \Rightarrow \quad \frac{d^2x}{dt^2} + \frac{k}{m}x = 0, \tag{2.25}$$

onde $k/m = \omega^2$ é definida como frequência angular. Finalmente podemos enunciar que a expressão que descreve um movimento harmônico simples, também chamado de movimento simples sem amortecimento, pode ser dado por:

$$\frac{d^2x}{dt^2} + \omega^2 x = 0. \tag{2.26}$$

Após processos algébricos simples, envolvendo equações diferenciais, Nussenzveig (1998) conclui que a equação do oscilador harmônico simples pode ser resolvida analiticamente com

$$x(t) = A\cos(\omega t + \varphi), \tag{2.27}$$

onde existem duas constantes arbitrárias, a amplitude A e a constante de fase φ.

Dessa forma, vemos que é possível utilizar tanto a equação2.26, como a equação 2.27para obtermos soluções para um OHS (oscilador harmônico simples). A equação2.27 será muito útil para as simulações deste capítulo e na comparação de resultados numéricos e analíticos.

2.5.2 Oscilador harmônico amortecido (OHA)

Da seção anterior podemos concluir que, as oscilações harmônicas simples são típicas de sistemas conservativos, informação que está em conformidade com (NUSSENZVEIG, 1998). Dessa mesma referência vem também a implicação que em sistemas reais sempre existem uma dissipação de energia. De fato, observando a natureza, percebe-se que para os movimentos harmônicos sempre existem pelo menos uma força que resiste.

Fisicamente existem alguns exemplos clássicos, como pode ser visto em (NUSSENZVEIG,1998), onde são destacados alguns acontecimentos que merecem interesse, como é o caso de um pêndulo onde as oscilações são amortecidas em virtude da resistência do ar e do atrito com o suporte, o comportamento da viscosidade em um líquido em um tubo em forma de *U*, onde este líquido amortece as oscilações, fato semelhante também ocorre em um diapasão.

Para esses casos citados anteriormente a força de amortecimento é proporcional a velocidade. O procedimento para obter uma expressão para o caso das ***oscilações amortecidas***, consiste em adicionar o termo $b\dot{x}$ que representa a resistência dissipativa. Esse termo opera em sentido contrário a velocidade, dessa forma, temos:

$$m\ddot{x} = -kx - b\dot{x}, \qquad \geq 0. \tag{2.28}$$

Se dividirmos ambos os membros por m,

$$\ddot{x} + \gamma + \omega_0^2 x = 0, \tag{2.29}$$

onde

$$\omega_0^2 = k/m, \qquad \gamma = b/m > 0. \tag{2.30}$$

Observamos que esta expressão, trata-se uma equação diferencial linear homogênea de segunda ordem com coeficientes constantes. Portanto, como pode ser visto em (NUSSENZVEIG,1998), após alguns passos simples é possível encontrar resultados para a expressão citada. É notável frisar que existem mais de uma solução, dessa forma podemos destacar três casos diferentes: *amortecimento subcrítico*, quando $\gamma/2 < \omega_0$; *amortecimento supercrítico*, quando $\gamma/2 > \omega_0$; *amortecimento crítico*, quando $\gamma/2 = \omega_0$. Nesta subseção será dado ênfase para o *amortecimento subcrítico* e a solução analítica para este caso, pode ser dada por

$$x(t) = Ae^{-\frac{\gamma}{2}t}\cos(\omega t + \varphi), \tag{2.31}$$

onde A e φ são as constantes reais e arbitrárias, e ω é

$$\omega = \sqrt{\omega_0^2 - \frac{\gamma^2}{4}}. \tag{2.32}$$

Podemos ainda substituir o resultado de γ dado em2.30em2.32, dessa forma concluímos que o valor de ω pode ser dado por:

$$\omega^t = \sqrt{\frac{k}{m} - \frac{b^2}{4m^2}},$$

A expressão para a posição em função do tempo pode ser dada, tanto pela a expressão2.31, como pela expressão

$$x(t) = Ae^{-bt/2m}\cos(w't + \varphi). \tag{2.33}$$

Outra solução analítica que pode ser obtida para o oscilador harmônico amortecido é o caso *amortecimento supercrítico*, esse caso tem como principal marca, o fato de $\gamma/2 > \omega_0$, a posição em função do tempo para este caso é dada por

$$x(t) = e^{-\frac{\gamma}{2}t}(ae^{\beta t} + be^{-\beta t}). \tag{2.34}$$

E por último, temos a solução geral do *amortecimento crítico*,

$$x(t) = e^{-\frac{\gamma}{2}t}(a + bt). \tag{2.35}$$

A equação2.33será utilizada para os cálculos numéricos e comparações de resultados que serão feitas nas próximas seções. Ela foi escolhida por ser de solução mais simples e de fácil manipulação.

2.6 Solução numérica para OHS

Vamos partir do conteúdo explicitado na seção referente aos osciladores, onde esclarecemos como obter a solução analítica do OHS. Nesta seção vamos mostrar como resolve-la numericamente.

A equação diferencial do movimento harmônico simples pode ser resolvida numericamente usando Runge-Kutta, para isso, primeiramente um algoritmo deve ser desenvolvido, a linguagem de programação utilizada neste processo é o C++[6].

Partindo da equação do movimento harmônico simples, vemos que esta é uma equação de segunda ordem. Na seção2.2foi especificado que existem alguns tipos de soluções para uma EDO, e um desses métodos é o de redução de ordem, onde é possível reescrever uma EDO de segunda ordem como ED's de primeira ordem.

Para a equação do OHS, vale observar que existe o termo cos, essa função é periódica com valor de período igual a 2π. Dessa forma a solução $x(t)$ deve ser necessariamente periódica. ConsultandoNussenzveig(1998), temos que o período pode ser definido como

$$T = \frac{2\pi}{\omega_0} = \frac{2\pi}{\sqrt{k/m}},$$

onde T depende apenas de k/m.

[6] A linguagem de programação utilizada para todos os programas deste trabalho é a linguagem C++.

Antes de implementar um algoritmo para a solução numérica da equação do OHS, é necessário definir algumas convenções. Para êxito na criação do algoritmo, uma sequência lógica deve ser seguida. É recomendado escolher um valor fixo para velocidade inicial, dessa forma $v(t = 0) = 0$. É aconselhável também, definir $k/m = 1$. O tempo inicial é definido como $t_i = 0$. Sendo o método utilizado o Runge-Kutta de quarta ordem, o algoritmo terá subdivisões do intervalo de tempo equivalente a $[t_i, t_f]$ com tamanho do passo

$$h = \frac{t_f - t_i}{N},$$

onde N é o número de pontos de malhas. O método de Runge-Kutta é usado para obter x_{i+1} e v_{i+1} a partir de valores iniciais x_i e v_i. Após calcular $x(v)_{i+1}$ promovemos $t_{i+1} = t_i + h$.

No apêndiceB, podemos analisar um código C, onde foi utilizado os métodos de Runge-Kutta de quarta ordem para resolver o tipo de oscilador mencionado nesta seção.

Para a compilação deste programa foi utilizado o *Codelite* [7]. Para esse caso a resposta obtida foi registrada no arquivo *OHS.dat*, que pode ser encontrado no diretório que foi escolhido pelo programador na hora da criação do projeto. Esse arquivo *OHS.dat*, expõe dados suficiente para a plotagem de um gráfico.

O programa que será utilizado para a criação dos gráficos será o *QtiPlot* [8]. Esta é uma ferramenta satisfatória, pois além de oferecer uma boa qualidade de gráficos, dispõe de aparatos de ótima eficácia. A ferramenta disponível para a inserção de funções, é sem dúvida, um apetrecho fundamental para a comparação entre os resultados analíticos e numéricos.

2.6.1 Comparando os resultados numéricos e analíticos para o caso do oscilador harmônico simples

A seguir será mostrado passo a passo como obter um gráfico utilizando os dados adquirido através do código *C++* exposto no apêndiceB, também será comparado os resultados numéricos do programa, com os que podem ser obtidos analiticamente. Como o sistema utilizado no desenvolvimento desse trabalho foi *Ubuntu*, os procedimentos expostos aqui são referentes a esse sistema. O *QtiPlot* será utilizado para fazer a comparação dos resultados, este programa pode ser obtido sem dificuldade através da *Central de programas do Ubuntu*, e os procedimentos para a sua utilização foram expostos na seçãoA.8.

Após a compilação do programa [9], será necessário importar os dados do *.dat* e *plotar* o gráfico [10]. Obtido êxito nesta operação, é necessário inserir a função correspondente a solução analítica para o caso do oscilador harmônico simples no programa [11], dessa forma

[7] Todos os programas utilizados neste trabalho foram compilados pelo *Codelite*, verA.3.
[8] Para mais detalhes sobre este programa ver seçãoA.8.
[9] Como compilar um programa passo a passo, seçãoA.3.
[10] Como utilizar o *QtiPlot* para este procedimento, passo a passo, ver seçãoA.8.
[11] A Figura39, exemplifica com clareza esse processo.

podemos comparar o resultado numérico calculado através do código *C* citado no apêndice Be o resultado analítico referenciado em (NUSSENZVEIG,1998). Na Figura3, temos a comparação para a posição em função do tempo, $x(t)$.

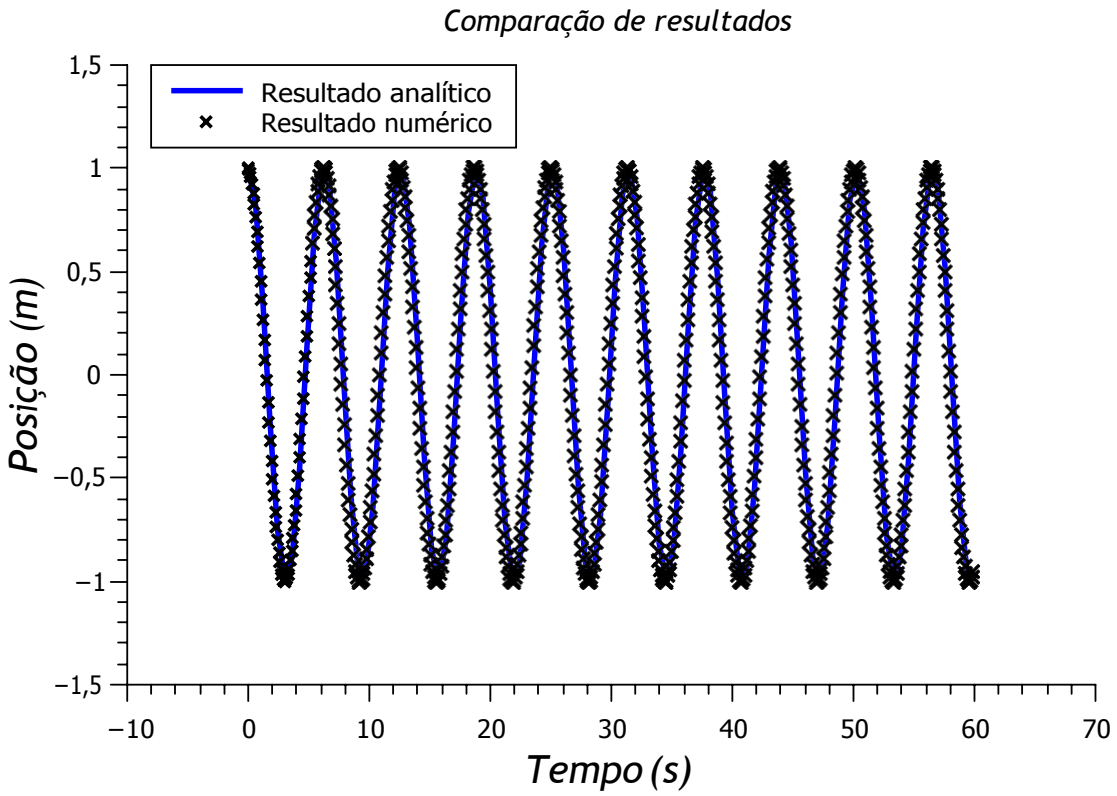


Figura 3 – *Comparação dos resultado analítico com o numérico, para a posição em função do tempo, no caso do oscilador harmônico simples*.

Um exemplo mais detalhado desse gráfico pode ser visto na Figura41a, encontrada no apêndiceB, explicitando com mais detalhes a inserção dos parâmetros no *QtiPlot*.

Também é possível obter um gráfico da velocidade em função do tempo, utilizando as mesmas ferramentas mostradas anteriormente. A Figura4mostra a comparação entre os resultados numéricos e analítico, para $v(t)$.

Um exemplo mais detalhado desse gráfico pode ser visto na Figura42a, encontrada no apêndiceB, explicitando com mais detalhes a inserção dos parâmetros no *QtiPlot*.

Observamos que o resultado numérico condiz com o analítico, essa conclusão pode ser afirmada pelo fato da curva obtida utilizando os parâmetros do resultado analítico, se ajustarem perfeitamente com o gráfico alcançado através dos método numérico, utilizando Runge-Kutta. Isso também comprova que os métodos de Runge-Kutta satisfazem a solução do problema, sendo assim esse é um método numérico adequado para o cálculo computacional de problemas referente aos osciladores harmônico simples ou que tenham características semelhantes aos utilizados nessa seção.

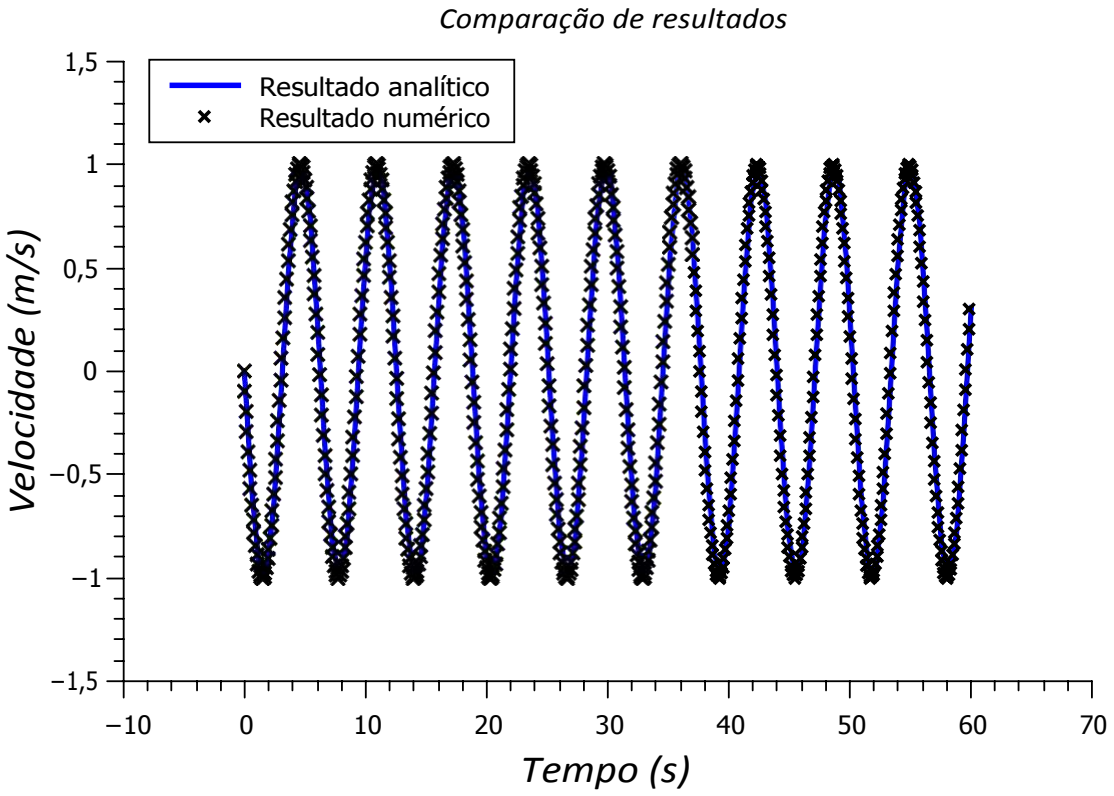


Figura 4 – *Comparação dos resultado analítico com o numérico, para a velocidade em função do tempo, no caso do oscilador harmônico simples.*

2.7 Solução numérica para OHA

Os procedimentos para resolver numericamente a equação do oscilador harmônico amortecido é semelhante ao utilizado na seção anterior. Nesta seção, também vamos comparar os resultados da posição em função do tempo, obtidos numericamente com a solução obtida algebricamente, como na seção anterior iremos usar o QtiPlot.

Para construir um algoritmo que possa resolver a equação do OHA, vamos partir da eq.2.29. Essa expressão também pode ser quebrada em duas expressões, analogamente ao caso do OHS,

$$\frac{d^2x}{dt^2} + \frac{b}{m}\frac{dx}{dt} + \frac{k}{m}x = 0, \tag{2.36}$$

$$u^{tt}(t) + \frac{b}{m}u^{t}(t) + \frac{k}{m}u(t) = 0, \tag{2.37}$$

$$u^{t} = v,$$

$$v^{t} = -\frac{k}{m}u - \frac{b}{m}v.$$

De posse dessas informações, podemos aproveitar o programa utilizado para OHS e adaptá-lo para o OHA, trata-se apenas de inserir a eq.2.37e seus respectivos derivados

no programa, implementando mudanças adequadas. Algumas definições indicam que tipo de oscilador vamos ter: *critico*, *subcrítico* ou *supercrítico*. Para o exemplo utilizado nessa seção, foi definido $k = 7$, $b = 8$ e $m = 1$. Observemos que para este caso, teremos um *oscilador amortecido subcrítico*. O código *C++* para este programa pode ser visualizado no apêndiceC.

A partir da expressão2.31e do arquivo *.dat* fornecido pela compilação do código *C++* [12], torna-se fácil a construção de um gráfico que mostre a posição em função do tempo para um oscilador harmônico amortecido, e a comparação do resultado numérico com o analítico para essa espécie de oscilador. Será utilizado um procedimento análogo ao caso do OHS, a única diferença é que vamos utilizar a equação referente ao OHA. Após inserir a equação no QtiPlot e fazermos os ajustes adequados, colocando os valores correspondente no do código *C++*, obtemos como resposta o gráfico mostrado na Figura 5.

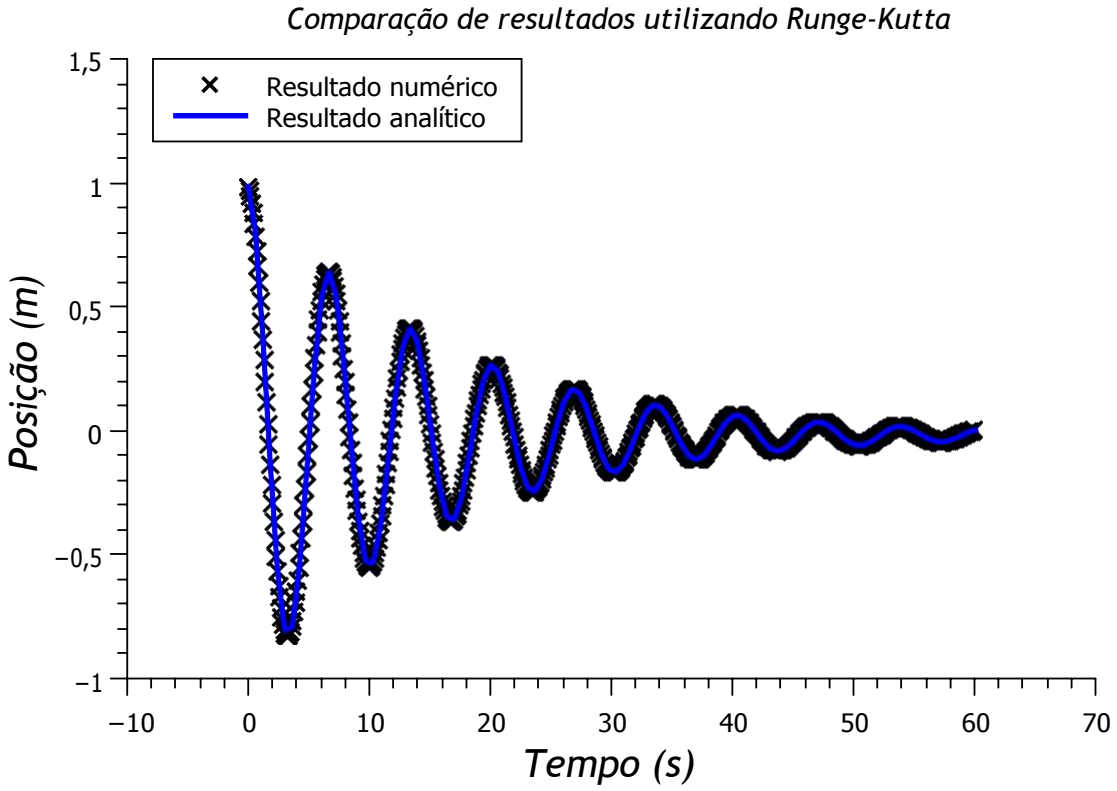


Figura 5 – *Comparação do resultado analítico com numérico, para a posição em função do tempo, no caso do oscilador harmônico amortecido.*

No apêndiceC, é mostrada a Figura43a, que especifica com detalhes como colocar função no QtiPlot para obter esse gráfico, esclarecendo de forma clara como inserir cada parâmetro.

[12] O código do pode ser visualizado no apêndiceC, com todos os detalhes.

Mais uma vez podemos constatar, a eficácia do método de Rugge-Kutta de quarta ordem, análogo ao caso do OHS podemos observar que a curva obtida através dos parâmetros inseridos na equação analítica se assemelha com a curva obtida utilizando o método numérico de Runge-Kutta.

2.8 GSL e os osciladores harmônicos

Nesta seção será apresentado um método alternativo para o cálculo de osciladores harmônicos. Podemos ver na seçãoA.4dos apêndices, uma introdução referente ao *GNU Scientific Library* (GSL). Essa biblioteca numérica será de extrema utilidade se por algum motivo os métodos citados anteriormente não satisfaçam as necessidades de alguns cálculos numéricos. A biblioteca GSL será utilizada no desenvolvimento do capítulo referente as simulações numéricas.

A idéia aqui é semelhante aos processos utilizados anteriormente na seção2.4, a única distinção é a necessidade de um sistema mais apurado, mas se todo o procedimento apresentado no capítuloAfoi reproduzido sem ausência de detalhes, os mecanismo propostos a seguir terão desfecho positivo.

2.8.1 Oscilador harmônico simples

Já foi visto que, a equação que rege os osciladores harmônicos simples é dada pela equação2.25, como já foi especificado também, esta é uma equação diferencial de segunda ordem, e pode ser quebrada em duas ED's de primeira ordem [13].

Notamos que no exemplo dado pela apostila GSL, o código utilizado para a solução da equação de van der pol[14] utiliza o conceito de jacobiano, mas para o caso dos osciladores harmônicos estudados aqui não é necessário. Contudo, por razão de completeza vamos explicar passo a passo a utilização do conceito de *matriz jacobiana*. A *matriz jacobiana* é dada por

$$\begin{bmatrix} \frac{dF_1}{du} & \frac{dF_1}{dv} \\ \frac{dF_2}{du} & \frac{dF_2}{dv} \end{bmatrix}. \tag{2.38}$$

Para o caso do OHS $F_1 = u^t$ e $F_2 = v^t$. Substituindo o resultado desses parâmetros

[13] Esse método de quebra de EDO's de segunda ordem, pode ser visto nas seções2.3e2.4.

[14] A **equação de Van der Pol** representa um oscilador com um termo de amortecimento não linear, sua solução numérica pode ser visto em (FOUNDATION,2016).

na matriz obtemos como resposta,

$$\begin{vmatrix} 0 & 1 \\ -k/m & 0 \end{vmatrix}. \tag{2.39}$$

Um fator importante na adaptação desse algoritmo citado é a implementação de linhas que objetivem a criação de um arquivo após a compilação do programa [15].

O entendimento destes conceitos, citados nesta subseção, são satisfatórios para podermos entender a solução numérica, para o caso do OHS mostrado a seguir.

Vamos agora adaptar o código C++ da Apostila GSL[16]. No apêndiceDpodemos ver o código equivalente ao utilizado para obter os resultados dessa seção.

Para o nosso caso, devemos entender que na parte correspondente ao jacobiano vamos ter: $0,0$ equivalente dF_1/du, e para $0,1$ o valor atribuído será o de dF_1/dv, para $1,0$ será dF_2/du e por fim $1,1$ será o resultado dF_2/dv. No código C++ visualizado no apêndice D, essas alterações serão observadas nas linhas 24 a 27, aproximadamente. Também é definido que $u^t = v$ será $f[0] = [1]$ e $v^t = -\mu u$ será $f[1] = -\mu y[0]$.

Após essas modificações, no código C++ já poderá gerar um arquivo.*dat*, com informações suficientes para a *plotagem* de um gráfico para comparação de resultados semelhante aos casos anteriores.

Utilizando essas informações, podemos fazer uma comparação dos resultados de modo análogo a seção anterior. As figuras a seguir mostram a comparação entre resultados analítico e numérico. Para $x(t)$, Figura6, e para $v(t)$, Figura7. No programa foi inserido $mu = 1$, onde $mu = k/m$. Em2.25foi definido $k/m = \omega^2 \Rightarrow \omega = \sqrt{k/m}$, logo na solução analítica deve-se obrigatoriamente utilizar $\omega = 1$.

No apêndiceDpode ser visualizado as figuras44ae45a, respectivamente elas demonstram com mais clareza os valores utilizados para cada um dos parâmetros da função utilizada.

Podemos observar que as curvas, tanto do resultado numérico como analítico são semelhantes, assim podemos concluir que esse processo de cálculo utilizando o *GSL* é semelhante aos resultados obtidos utilizando Runge-Kutta.

2.8.2 Oscilador harmônico amortecido

O procedimento deve ser semelhante ao anterior. Dessa forma, vamos utilizar os parâmetros de forma semelhante. Para esse caso vamos ter: *massa*, *constante elástica* e

[15] A seçãoA.7especifica com clareza como inserir linhas em código C++ que resulte na criação de um arquivo.

[16] Esse código pode ser visto no exemplo da apostila GSL, na pag. 325,326

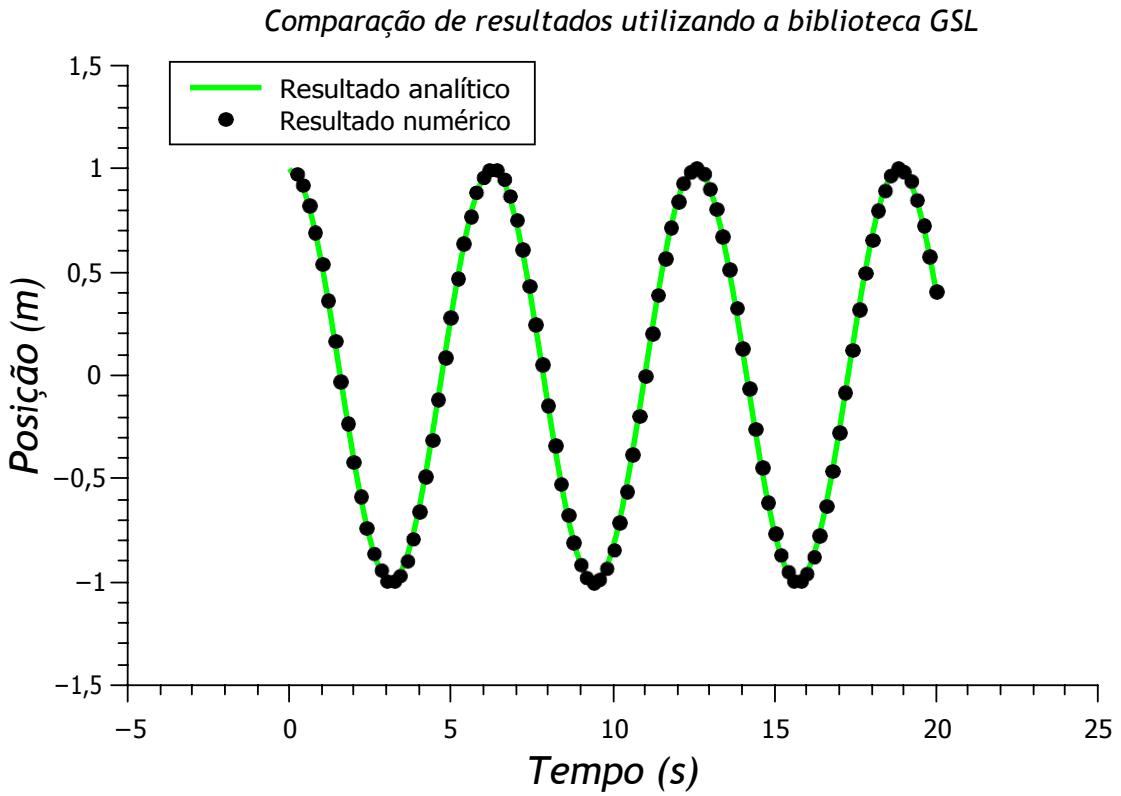


Figura 6 – *Comparação dos resultado analítico com o numérico, para a posição em função do tempo, no caso do oscilador harmônico simples.*

a *constante da viscosidade*. Esses valores serão definidos respectivamente como: $m = 8$, $k = 7$ e $b = 1$.

Como já esclarecido anteriormente, a utilização do jacobiano não é necessária, mas por razão de completeza utilizaremos também neste caso.

Para a quebra da *EDO* de segunda ordem, teremos $u^t = v$ e $v^t = -(k/m)u-(b/m)v$, onde $F_1 = u^t$ e $F_2 = v^t$, logo a matriz jacobiana para o caso do oscilador harmônico amortecido será dada por

$$\begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix}. \tag{2.40}$$

Com esses dados já é possível obter um gráfico para o caso do oscilador harmônico amortecido.

Fazendo o uso do programa adaptado na seção anterior, podemos novamente ajustá-lo para o oscilador harmônico simples, inserindo os dados nos locais adequados, o código C++ para este programa pode ser visualizado no apêndiceE.

A compilação do programa, como já visto em vários casos anteriores, oferece como resposta um arquivo de extensão *.dat*, onde podemos obter informações suficientes para a *plotagem* de um gráfico, para esse caso, um gráfico de um OHA.

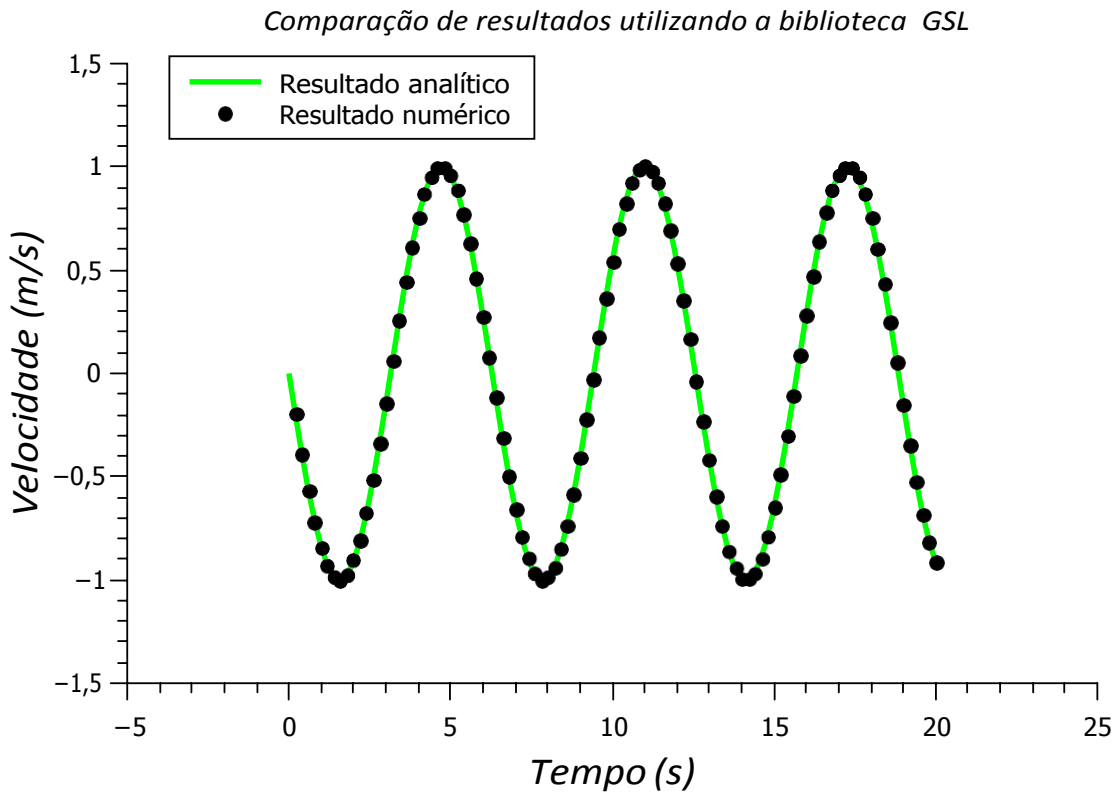


Figura 7 – *Comparação dos resultado analítico com o numérico, para a velocidade em função do tempo, no caso do oscilador harmônico simples.*

A Figura8expressa a resposta obtida pela compilação do programa

Para um melhor entendimento, pode-se analisar a Figura46a, ela demonstra alguns detalhes extras, tanto na inserção da fórmula, como na inserção dos parâmetros referentes a este caso.

Observamos então a comparação de resultados numérico e analítico, para a posição em função do tempo no caso de um oscilador harmônico amortecido subcrítico. Assim como para oscilador harmônico simples a solução para o oscilador harmônico amortecido também pode ser observado um ajuste de curvas entre os resultados.

2.9 Conclusão

Nesse capítulo, abordamos conceitos relativo ao cálculo numérico, vimos que os métodos de Euler são de simples utilização, e aplicamos na solução de equações elementares da mecânica. Concluímos que, se o nível de complexidade de uma equação for mais elevado, os métodos de Euler não são apropriados. Por fim, introduzimos os métodos de Runge-Kutta e fizemos várias aplicações, utilizando esses métodos para o caso do oscilador harmônico simples e amortecido e calculamos posição em ambos os casos. Fizemos o cálculo da velocidade apenas para o oscilador harmônico simples, para o oscilador amortecido não foi

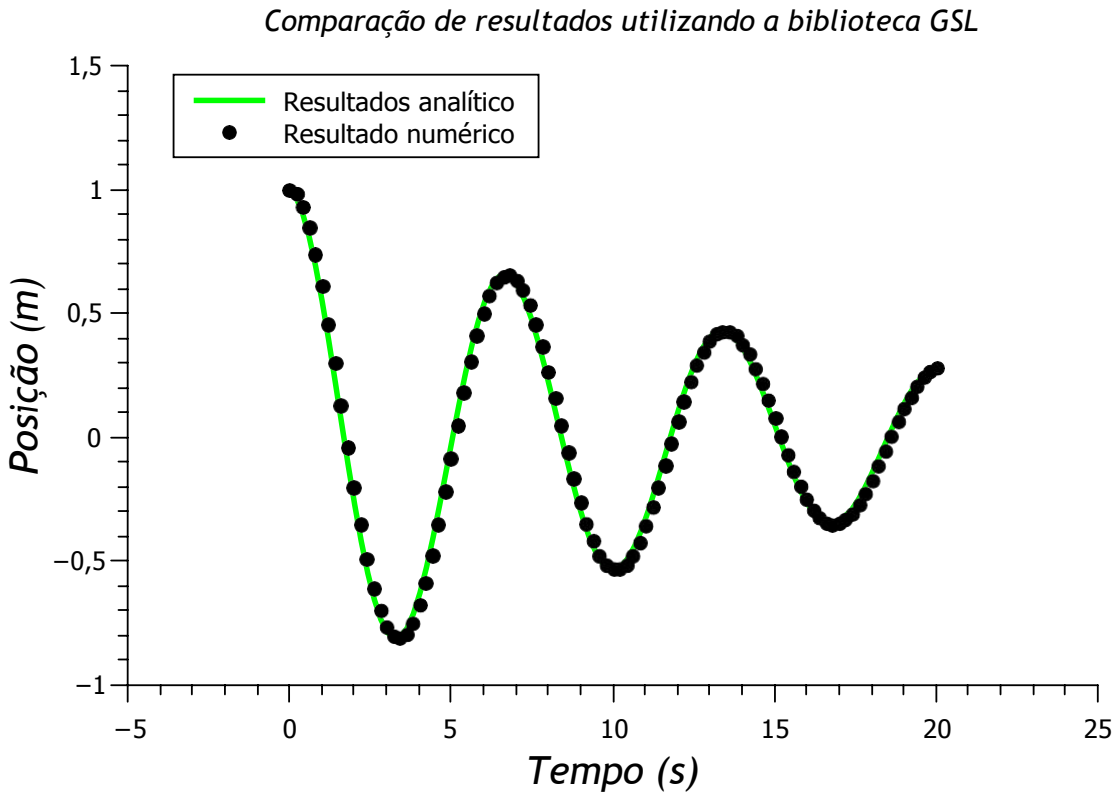


Figura 8 – *Comparação do resultado analítico com numérico, para a posição em função do tempo, no caso do oscilador harmônico amortecido.*

necessário.

Um ênfase especial deve ser dada na aplicação da biblioteca GSL, sendo que essa será de extrema importância no capítulo referente à simulação do melaço.

3 Emissão e Absorção da Luz Pelos Átomos

3.1 Introdução

Neste Capítulo, vamos estudar a dinâmica da evolução da velocidade dos átomos em um melaço ótico. O melaço, como vimos brevemente no Capítulo 1 e veremos com mais detalhes a seguir, ocorre devido as absorções e emissões de fótons pelos átomos.

Iniciaremos o estudo pela evolução dos modelos atômicos para chegarmos na interação de fótons com átomos e entendermos o modo quantizado da energia. Em seguida, na Seção3.2, complementando o entendimento da quantização, é mostrado um pequeno histórico da natureza da luz. Também são mostrados os espectros eletromagnéticos. Desta forma, poderemos compreender os princípios da emissão e absorção da luz pelo átomos.

3.2 Modelos atômicos

Nesta seção abordaremos conceitos pertinentes ao caráter evolucional dos modelos atômicos, desde as primeiras propostas dadas pelos filósofos antigos, até o modelo do átomo Bohr. Além de ajudar a entender o comportamento da luz, vale também esclarecer que esta seção tem como propósito motivacionar as seções seguintes, fornecendo uma base para a compreensão dos fenômenos relacionados ao comportamento da luz.

Uma característica clássica da ciência é a constante mudança no decorrer do tempo, para um reforço válido desse argumento, basta procurarmos entender a teoria dos modelos atômicos. A percepção desse tópico serve para entendermos o comportamento da luz, e consequentemente muitos dos fenômenos relacionados ao comportamento da radiação [1].

Como podemos observar em uma infinidade de referências bibliográficas, o modelo atômico atual é consequência de um intenso processo evolutivo. Como sabemos, a curiosidade dos antigos filósofos deram o impulso necessário para toda a teoria científica atual. A teoria atômica da antiguidade que mais se adapta a constituição da matéria que compõe o Universo atual, foi a hipótese atomista proposta por Leucipo e Demócrito, como pode ser visto emMartins(2002). Uma das razões fundamentais dessa proximidade, é o fato desses filósofos não recorrer a "entidades divinas"ou "misteriosas"para explicar a estrutura da matéria.

A hipótese de Leucipo e Demócrito consistia em dividir a matéria em pedaços cada vez menores, em um processo repetitivo até chegar em uma partícula muito pequena que

[1] Processo de emissão de energia que acontece por meio de ondas ou partículas.

seria indivisível e invisível a "olho nu"[2]. Quase um século mais tarde, Epicuro, também filósofo grego, denominou de *átomo* estas partículas indivisíveis (MARTINS,2002). Podemos notar também, que as ideias de Leucipo, Demócrito e Epicuro apoiavam-se na existência do vazio, como indicado emCaruso(2006). Leucipo evidenciou que os átomos seriam inúmeros elementos, em movimento perpétuo.

Figura 9 – *O atomismo grego*.
Fonte: *Física moderna, Origens Clássicas e Fundamentos Quânticos p.08* [3]

Depois de um imenso recesso na linha evolutiva, no que diz respeito ao avanço do conhecimento da teoria atômica, apenas no século XVI, Pierre Gassend defendeu os precursores da teoria atômica da antiguidade: Leucipo, Demócrito e Epicuro (MARTINS, 2002). De acordo comCaruso(2006), vários outros cientistas fizeram especulações sobre a teoria atômica, mas foi Dalton o primeiro a formular uma teoria científica mais realista para a época.

Retomando a hipótese atômica de Leucipo e Demócrito, John Dalton propôs uma teoria para o modelo atômico em 1808, em que, o átomo seria uma minúscula esfera maciça, impenetrável, indestrutível, indivisível e sem carga. Esse modelo ficou conhecido como *modelo atômico da bola de bilhar*.

Após o modelo atômico proposto por Dalton, alguns outros cientistas se destacaram na tentativa de buscar conceitos mais precisos para teoria atômica. Dentre alguns, temos Pieter Zeeman, que em 1896 obteve as primeiras provas da existência de partículas atômicas. Observando *linhas espectrais* emitidas por átomos na presença de um campo

[2] Quando pode-se visualizar sem o uso de instrumentos ópticos.
[3] Francisco Caruso. Fisica Moderna. Elsevier, Rio de Janeiro, 2006.

magnético, Zeeman confirmou uma relação bem clara entre carga e massa (TIPLER PAUL A.; LLEWELLYN,2010).

Voltando aos modelos atômicos, notamos que até então, os modelos propostos eram imprecisos e confusos, necessitando assim de aperfeiçoamento. Contudo, J.J Thomson propôs um outro modelo atômico, baseado em evidências experimentais que constatavam a existência de elétrons nos átomos (EISBERG,1979). Para Thomson os elétrons estariam "impregnados"em uma forma de distribuição contínua de carga positiva esférica, com raio 10^{-10}. E uma repulsão mútua entre os elétrons, fazia com que eles ficassem uniformemente distribuídos no interior da esfera (EISBERG,1979).

Dentre outras tentativas, Thomson buscou manter o sistema atômico em equilíbrio estável, procurando uma conexão entre os modos normais de vibrações e as frequências que eram observadas nos espectros de emissão, mas as forças eletrostáticas eram insuficientes para tal equilíbrio (TIPLER PAUL A.; LLEWELLYN,2010). Esse modelo ficou conhecido como *modelo pudim de passas*.

Após sucessos obtidos, através de uma série de experimentos de emissão de partículas alfa sobre folhas de ouro, realizados por Ernest Rutherford e seus alunos H. W. Geiger e E. Marsden, o modelo atômico proposto por Thomson começou a perder a relevância para os cientistas da época. Um novo modelo surgiu, o modelo atômico de Rutherford. Para Rutherford, todas as cargas positivas e a massa do átomo, estariam concentradas em uma pequena região no centro, chamada núcleo do átomo (EISBERG,1979).

Uma desvantagem facilmente perceptível do modelo atômico de Rutherford era o conflito com os princípios do eletromagnetismo, onde toda partícula que tenha carga elétrica e se submeta a uma aceleração será condicionado a emissão de energia em forma de ondas eletromagnéticas. Sendo assim, como no modelo atômico de Rutherford o elétron descrevia um movimento orbital, esse seria governado por uma aceleração centrípeta, consequentemente emitiria energia em forma de onda eletromagnética, e considerando o princípio da conservação de energia, o elétron perderia tanto energia cinética como potencial, colapsando contra o núcleo (EISBERG,1979). Essa descrição atômica ficou conhecida como o *modelo de um sistema planetário*.

Dessa forma, a mecânica clássica não explicava a estabilidade do átomo, pois no átomo clássico o elétron descrevia uma órbita em espiral em direção ao núcleo (TIPLER PAUL A.; LLEWELLYN,2010), fato justificado pela emissão constante de radiação.

Niels Bohr utilizou as idéias de Planck e Einstein na tentativa de melhorar o modelo de Rutherford, elaborando um modelo de átomo simples [4] (NUSSENZVEIG,1998). De acordo com o modelo de Bohr, o átomo só deverá irradiar energia se executasse uma transição de uma órbita de raio maior para uma órbita de raio menor.

[4] Bohr utilizou como modelo o átomo de Hidrogênio.

O modelo de Bohr baseiava-se nos seguintes postulados: ***i)*** Obedecia as leis da mecânica clássica, o elétron em um átomo se movia em uma órbita circular em torno do núcleo sob influência da atração coulombiana, interação elétron-núcleo. ***ii)*** O elétron não tinha uma infinidade de órbitas possíveis como previa a mecânica clássica, agora o elétron só se moveria em uma órbita na qual seu momento angular orbital L fosse um múltiplo inteiro de $\overline{h}$. ***iii)*** A energia total E permanece constante, pois apesar de está sempre acelerado, o elétron se move em uma única órbita não emitindo radiação eletromagnética. ***iv)*** Só haverá emissão de radiação eletromagnética, se um elétron sair de sua órbita de energia total E_i para uma órbita de energia total E_f. Dessa forma, a frequência da radiação emitida v, é igual à quantidade $(E_i - E_f)$ dividida pela constante de Planck h (EISBERG, 1979).

Então, concluímos que o primeiro postulados de Bohr trata da existência do núcleo atômico, o segundo introduz a ideia de quantização, o terceiro postula a eliminação do problema da estabilidade de um elétron que se move em órbita circular. E finalmente o quarto, também chamado postulado de Einstein, determina que a frequência de um fóton de radiação eletromagnética é igual a energia carregada pelo fóton dividida pela constante de Planck, dada por: $v = (E_i - E_f)/h$ (EISBERG,1979).

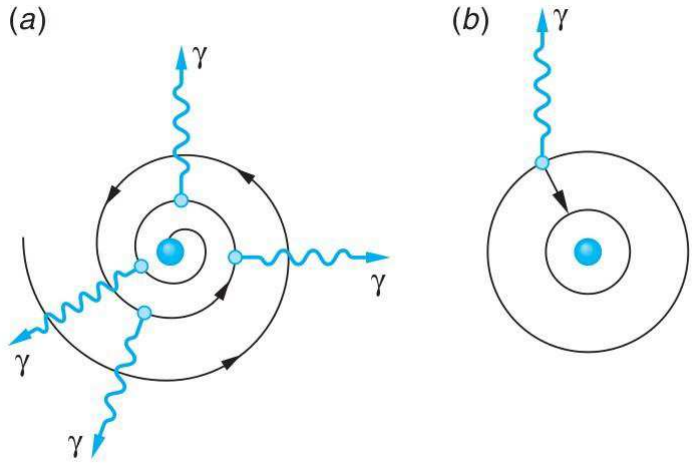


Figura 10 – *Representação de dois modelos para o átomo: modelo clássico e modelo de Bohr.*

Fonte: *Modern Physics, Quinta Edição, p.160* [5]

Na Figura10, podemos comparar dois modelos atômicos: em (a), observamos o modelo clássico do átomo, onde uma trajetória elíptica é descrita pelo elétron em direção ao núcleo, como já discutido, esse fato é decorrente da irradiação de energia constante. Já em (b), podemos visualizar o modelo atômico de Bohr, onde só haverá irradiação de energia pelo elétron, se houver uma transição para uma órbita de raio menor.

[5] Ralph A TIPLER, Paul A.; LLEWELLYN. Física Moderna. Quinta Edição. Editora LTC, Rio de Janeiro, 2010.

Apesar dos conceitos relativos à teoria dos modelos atômicos se estender após o modelo de Bohr com Heisenberg, De Broglie, Schrodinger e outros, os contextos expostos nessa seção já são suficientes para a fundamentação necessária neste trabalho, pois nesse modelo trata da emissão e absorção de átomos. Vale destacar que a teoria de Bohr, trata da emissão de luz por um átomo quando um elétron sofre transição do nível de energia superior para o inferior.

3.3 A natureza da luz

Mencionar tópicos referentes à natureza da luz pode render um grande número de páginas, no entanto, esta seção realizará uma abordagem resumida deste assunto, considerando apenas os aspectos mais relevantes. Será feito um resumo seguindo uma linha temporal, destacando conceitos e acontecimentos físicos relacionados com a luz, desde à visão de Newton até à descrição feita por Einstein, isso deve ser suficiente para uma entendimento simplificado dos conceitos pertinentes a este tópico, que serão apresentados posteriormente.

Lidamos diariamente com observações interessantes, utilizando nossos olhos podemos contemplar uma infinidade de fenômenos relativos ao comportamento da luz, desde o verde das plantas ao agrupamento das cores dos arco-íris. A explicação desses fenômenos encontra-se em um ramo da Física designado de Ótica. Nesse ramo, podemos encontrar uma parte referente as ondas eletromagnéticas[6]. O avanço da Ótica resultou em invenções de grande utilidade como o laser, a fibra ótica, os hologramas, os computadores óticos etc (ZEMANSKY,2009).

Dentre muitos fenômenos, a percepção de conceitos referentes ao comportamento da luz nos permite entender: porque o céu é azul, como funciona o olho humano. E dispositivos como telescópios, micróscopios, câmeras, óculos, dentre outros (ZEMANSKY,2009).

É interessante notarmos que existem duas teorias distintas relacionadas aos fenômenos ópticos, *teoria corpuscular da luz* e *teoria ondulatória da luz* (NUSSENZVEIG, 1998), a seguir faremos uma descrição simplificada desses dois fenômenos, abordando a evolução histórica de cada um.

Pela história da física, sabemos que desde a antiguidade os filósofos antigos buscavam entender fenômenos relacionados com a natureza da luz, em princípio a *ótica geométrica*. Desta forma antes de Newton, defendia-se que a luz era formada por corpúsculos[7], emitidos por fontes de luz (ZEMANSKY,2009).

Por outro lado, evidências sobre as propriedades *ondulatória* da luz surgiram à partir de 1665 (ZEMANSKY,2009). Christian Huygens quem deu a primeira grande

[6] A luz pode ser tratada como uma onda eletromagnética, como veremos mais adiante.
[7] Feixes de minúsculas partículas.

contribuição para essa teoria, formulando o *príncipio de Huyngens* (NUSSENZVEIG, 1998).

Mas foi somente a partir dos trabalhos de Thomas Young e Augustin Fresnel sobre interferência e difração, que a teoria ondulatória começou a ganhar força. Maxwell, formulou as equações básicas do campo eletromagnético (NUSSENZVEIG,1998), e em 1873 propôs o cálculo da velocidade de propagação das ondas eletromagnéticas, que em conjunto com evidências experimentais observada por Heinrich Hertz, iniciadas em 1887, demonstraram de forma clara que a luz é uma onda eletromagnética (ZEMANSKY,2009). Outros fenômenos também comprovaram a natureza ondulatória da luz: interferência, difração e polarização.

Apesar de tudo levar a crença que a luz é uma onda eletromagnética, outros experimentos tiveram efeitos não explicáveis por essa teoria ondulatória, como o efeito fotoelétrico, radiação de corpo negro e a emissão e absorção de luz que demostrava que essa tinha natureza corpuscular, tendo em vista que a luz transportava energia em formas de pacotes discretos, denominados *fótons* ou *quanta*. A partir de 1930, com o progresso da eletrodinâmica quântica, determinou-se que a luz pode se comportar, ora como uma onda, ora como uma partícula. Desta forma, quando se trata de propagação, essa se adequa melhor ao modelo ondulatório, mas para o entendimento de conceitos referentes a emissão e absorção a luz, se adapta melhor a natureza corpuscular (ZEMANSKY,2009).

Nessa seção, vimos uma abordagem resumida de conceitos relativos ao comportamento da luz, debatemos como as teorias de alguns cientistas evoluíram no decorrer do tempo, com relação a natureza da luz. Explicitamos também o contexto polêmico da dualidade da luz. Na próxima seção, trataremos dos espectros eletromagnéticos, que pode ser entendido como uma extensão dos conteúdos relacionados a natureza da luz.

3.3.1 Espectros eletromagnéticos

Uma especulação mais coesa sobre o comportamento da luz pode ser obtida através da observação do espectro eletromagnético. Nesta seção, explicitaremos a compreensão das ondas eletromagnéticas desde a sua descoberta, como também algumas aplicações. Destacaremos o conceito simplificado de luz monocromática e concluiremos mostrando algumas características das ondas eletromagnéticas.

A aplicação da ondas eletromagnéticas são de extrema importância na atualidade, como exemplo podemos citar: a televisão, a telefonia e a internet. Um avanço importante foi feito por James Clerk Maxwell, quando demonstrou que um raio luminoso é uma onda progressiva de campos elétricos e magnéticos, e que o estudo da luz visível, antes pertencente a ótica, é um ramo do eletromagnetismo. Dentro desse contexto, outro marco importante se deu quando, Heinrich Hertz, descobriu que além da luz visível e dos raios

infravermelhos e ultravioletas, existiam também as ondas de rádio, e que estas ondas se propagam com mesma velocidade da luz visível (RESNICK; HALLIDAY; WALKER, 2009).

Atualmente conhecemos um vasto espectro de ondas eletromagnéticas, e nós (seres humanos) estamos expostos a essas inúmeras ondas: radiação do sol, microondas de radares e telefonia celular, ondas eletromagnéticas proveniente de lâmpadas, de metrôs elétricos, máquinas de raio X, entre outras (RESNICK; HALLIDAY; WALKER,2009).

As ondas eletromagnéticas podem ser diferenciadas através do comprimento de onda e da freqüência, que pela equação $f\lambda = c$ são relacionadas a velocidade c da luz.

Das ondas eletromagnéticas, o mais comum é o termo *luz visível* que, está relacionado com o fato dos olhos dos seres humanos serem sensíveis às radiações eletromagnéticas com comprimentos de ondas aproximados de 400 nm até 700 nm (TIPLER,2006). Observando a Figura11, notamos que a faixa da luz visível é apenas um minúsculo segmento do espectro, essa pequena parte é dectada pela visão dos seres humanos, evocando sensações que evidenciam cores diferentes (ZEMANSKY,2008).

De acordo com Zemansky (ZEMANSKY,2008), foram detectadas ondas eletromagnéticas com frequência de no mínimo 1 até 10^{24}Hz, e que independente dessa frequência possuiam a mesma velocidade de propagação. Se o meio denotado for o vácuo, o valor de propagação das ondas eletromagnéticas equivale $c = 299, 792, 459$m/s, mantendo assim a relação $f\lambda = c$, como citado anteriormente.

Podem existir ondas eletromagnéticas com qualquer valor de λ entre 0 e ∞ (VILLATE,2012). As formas invisíveis de radiação eletromagnética merecem destaque pelo fato da sua grande aplicabilidade nas tecnologias (ZEMANSKY,2008) .

Outro termo bastante utilizado é o de *luz monocromática*, para entendermos essa expressão podemos utilizar a luz branca, como sabemos esse tipo de luz inclui todos os comprimentos de luz visível. Então selecionamos um pequeno estreito de comprimento de onda dentro da faixa de alguns *nm*, dessa forma podemos obter aproximadamente uma luz monocromática. Uma luz monocromática absoluta com um único comprimento de onda é uma idealização inatingível (ZEMANSKY,2008).

A Figura11mostra o espectro eletromagnético, desde ondas com comprimento muito pequeno, que é o caso dos *raio-x*, até ondas com comprimento muito longo. Nessa figura é possível notar também fenômenos para os quais as ondas eletromagnéticas são aplicadas, como destaque: rádio, a TV e uso na aeronáutica. Por fim, o espectro visível com seus respectivos comprimentos de onda, dados em *nm*.

Desta forma, notamos a importância do espectro eletromagnético, e podemos obter informações sobre uma variada quantidade de ondas eletromagnéticas e seus respectivos comprimentos. Nesse sentido, é interessante observar o que significa uma onda monocromá-

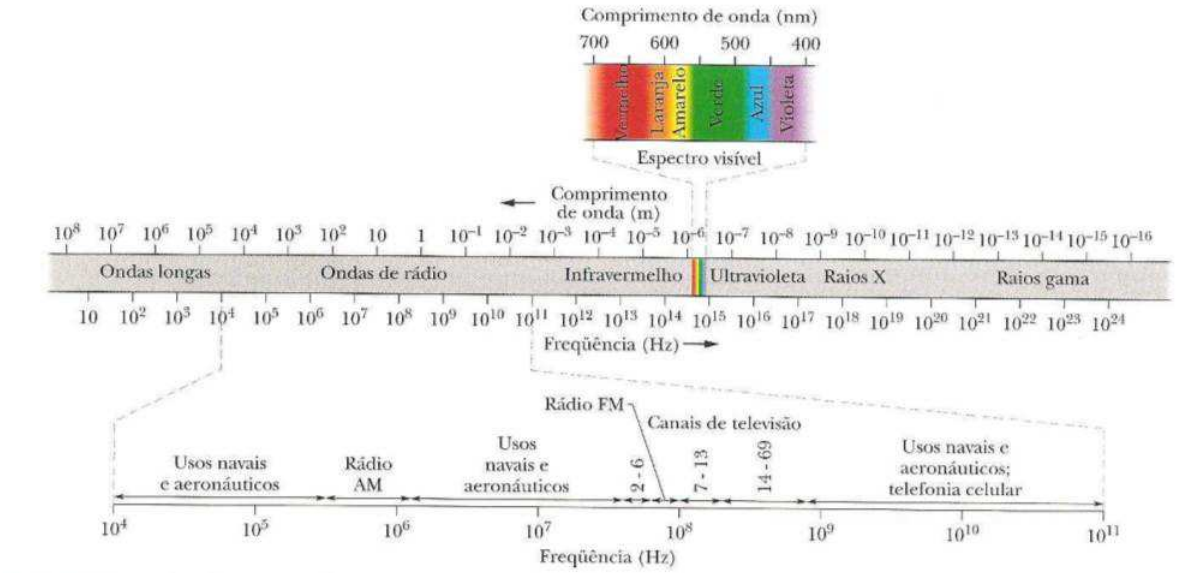


Figura 11 – *O espectro eletromagnético*.
Fonte: *Halliday & Resnick, p.973* [8]

tica dentro de todo o espectro eletromagnético. Na próxima seção, veremos os princípios de absorção e de emissão atômica.

3.4 Princípio de absorção e de emissão atômica

Após termos visto nas seções anteriores, os modelos atômicos e a descrição do átomo, nessa seção, vamos estudar os conceitos referentes a emissão e absorção da luz que estão interligados com a compreensão da força da pressão de radiação, um dos tópicos chave desse trabalho.

O caratér corpuscular da luz, pode ser comprovado através dos fenômenos de emissão, absorção e espalhamento [9] da radiação eletromagnética. Por meio desses fenômenos, é observado que a energia de uma onda eletromagnética é quantizada[10]. Essa energia pode ser absorvida ou emitida em forma de pacotes parecidos com partículas de energias definidas (ZEMANSKY,2009).

Utilizando características do efeito fotoelétrico[11] e a hipótese de Planck, Einstein postulou que um determinado feixe de luz era constituído por pequenos pacotes de luz de energia, denominados *fótons* ou *quanta*. Ainda segundo Einstein, a energia de um fóton

[8] Fundamentals of Physics [10th Edition] - Halliday & Resnick, Versão extendida.

[9] Ocorre quando fótons interagem com outras partículas ou campos sofrendo alterações na sua trajetória ou na sua energia.

[10] De acordo com HallidayRESNICK, HALLIDAY e WALKER(2009), uma grandeza é quantizada quando para ela encontra-se apenas múltiplos inteiros de uma quantidade elementar denominada quantum.

[11] Pode ser observado quando um determinado material é exposto a uma radiação eletromagnética e este emiteelétrons.

é dada por $E = hf = hc/\lambda$, onde h é a constante de Planck, e f é a frequência, e para ondas eletromagnéticas no vácuo $f = c/\lambda$ (ZEMANSKY,2009).

A combinação do conceito de fótons e de níveis de energia resulta na compreensão do espectro de linha[12]. Niels Bohr, em 1913, combinou esses dois efeitos (ZEMANSKY, 2009).

A energia do átomo é quantizada, e um único átomo só poderá assumir alguns valores definidos, chamados níveis de energia (ZEMANSKY,2009).

Um átomo pode fazer uma transição de um nível de energia para outro mais baixo, emitindo um fóton com energia igual a diferença de energia entre o nível inicial e o nível final. Sabemos que a energia do fóton é dada por $hf = hc/\lambda$, e de acordo com a lei de conservação de energia $hf = hc/\lambda = E_i - E_f$, onde E_i é a energia inicial do átomo e E_f é a energia final. Quando ocorre a emissão de um fóton de uma linha particular do espectro, acontece uma transição de nível excitado para um nível fundamental.

A emissão pode ocorrer de duas formas: emissão espontânea e emissão estimulada. Na Figura12, é mostrado o processo de emissão espontânea, no qual o átomo encontra-se inicialmente em um estado de energia superior E_2 e após um determinado tempo, o átomo decai para um nível de energia inferior E_1, emitindo um fóton de frequência $v = (E_2 - E_1)/h$ (ZEMANSKY,2009).

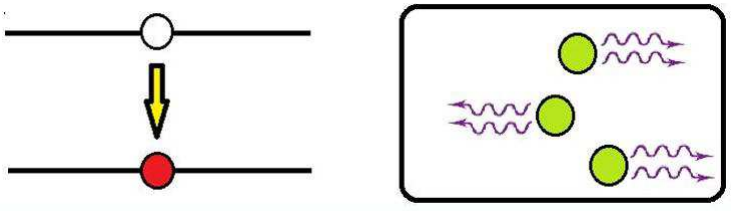


Figura 12 – *Esquema que representa a emissão espontânea, onde o átomo está inicialmente em estado excitado e depois de um tempo passa para um nível de energia inferior*.

Para a emissão estimulada, imaginemos que um átomo inicialmente excitado, seja perturbado por um fóton, haverá então, a emissão de outro fóton na mesma direção, sentido e fase. O processo de emissão estimulada pode ser visualizado na Figura13(ZEMANSKY, 2009).

Um exemplo de um instrumento que faz uso da emissão estimulada para produzir um feixe composto por um grande número de fótons coerentes é o laser (*light amplification by stimulated emission of radiation*), ou seja, amplificação da luz por emissão estimulada de radiação.

[12] Para Bohr os espectros de linha de um elemento eram fótons de energias bem definidas emitidas por átomos de um determinado elemento

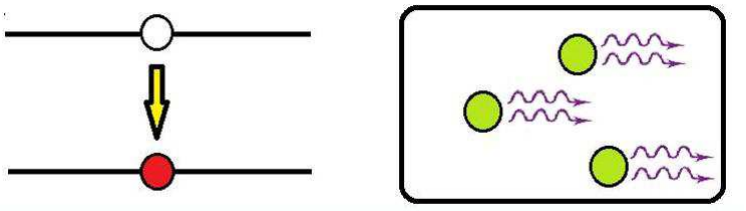


Figura 13 – *Esquema que representa a emissão estimulada, onde o átomo está inicialmente em estado excitado é perturbado por um ente exterior, e passa para um nível de energia inferior.*

Um processo complementar a emissão é a absorção. Esse processo de absorção ocorre quando é aplicado a um átomo um fóton incidente, que estimula o átomo a fazer uma transição de estado de energia inferior para o superior, e um fóton é absorvido pelo átomo nesse processo. A Figura14idealiza o processo de absorção com mais detalhes (ZEMANSKY,2009).

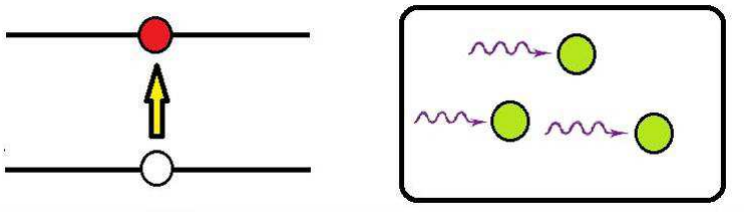


Figura 14 – *Esquema que representa o processo de absorção, ocorre quando o átomo é forçado a fazer uma transição de nível inferior para superior.*

Um exemplo coeso do processo de absorção pode ser idealizado com o seguinte exemplo: fazendo passar através de um gás, um feixe de luz branca, com espectro contínuo, observando com um espectrômetro a luz transmitida, observa-se uma série de linhas negras que correspondem ao comprimento de onda que foram absorvida. Essas linhas caracterizam o espectro de absorção (ZEMANSKY,2009).

Verifica-se que os átomos interagem com uma radiação incidente de frequência v, as populações dos dois níveis de energia são governados, simultaneamente, pelos três processos indicados anteriormente, a dinâmica de tal sistema foi primeiramente analisado por Einstein em 1917. Esses três processos são resultados da interação da radiação com o átomo (EISBERG,1979).

Como há nos fótons, além de energia bem definida, há momento $P = \bar{h}k$. Assim, os átomos que absorvem ou emitem fótons tem seus momentos alterados. Essa modificação do momento é o nosso foco nesse trabalho.

3.5 Conclusão

Partindo dos modelos atômicos, é possível entender a natureza da luz, e assim verificar os conceitos da interação dos átomos com a matéria, os princípios de absorção e de emissão atômica que levam a alteração de momento dos átomos e nos ajuda ao estudo do melaço ótico que é descrito pela força pressão de radiação.

4 Simulação do Melaço Óptico para Átomos de Césio

4.1 Introdução

No capítulo3, vimos que as interações dos átomos com os feixes de radiação são responsáveis pela modificação. Além da energia pela mudança de estado, o momento pela absorção de fótons que carregam. Essa variação de momento representa uma força média.

Neste capítulo, vamos simular a variação do momento para reduzir a velocidade dos átomos de césio em uma configuração chamada melaço ótico. Inicialmente, vamos mostrar na seção4.2as forças de um feixe de radiação sobre o átomo: reativa e dissipativa. Na seção4.3, vamos mostrar a configuração de Steven Chu do uso de 3 pares de feixes contrapropagantes que servem como um melaço ótico.

Após uma breve descrição dessas forças, iremos utilizar dados obtidos numericamente através de um programa C++ para observarmos o comportamento da velocidade e da aceleração e a partir de uma análise rápida entender o comportamento de um átomo de césio quando interage com feixes lasers.

Na seção4.4, estudamos a dinâmica do átomo na presença de um único feixe. Esse passo é necessário para entendimento da ferramenta numérica.

Na seção4.5, simulamos a dinâmica para dois feixes contrapropagantes na configuração de melaço ótico. Estudamos os resultados e verificamos a redução da velocidade atômica para diversos valores de velocidade.

Na seção4.6, comparamos os resultados numéricos com a relação a dependência teórica da aceleração com a velocidade.

Por fim, vamos comparar os resultados numéricos com os analíticos, através de um gráfico da aceleração em função da velocidade, com o objetivo de verificar a veracidade dos resultados numéricos obtidos neste capítulo.

4.2 Expressão da força no melaço óptico

Vimos no capítulo3que a interação da luz com átomos leva a uma modificação de momento e energia. Nessa seção, vamos abordar o conceito de força da pressão de radiação e, em especial, para um melaço. Iremos descrever os dois tipos de força de pressão da radiação para o caso de uma abordagem semiclássica, e por fim destacaremos uma expressão adequada para uma simulação numérica no melaço.

Quando se trata de um modelo semicássico[1] para a pressão da radiação podemos destacar dois tipos de forças: uma do tipo conservativa, sendo que essa está relacionada ao processo de aprisionamento dos átomos, e a outra que representa a força da pressão da radiação. Essa força conservativa tem a capacidade de gerar um potencial do tipo força de dipolo elétrico, sendo esse proporcional ao gradiente do campo da radiação $\Delta\Omega$. Essa força torna-se dominante quando se trata da força exercida pelo campo clássico de um feixe laser sobre um átomo quantizado, isso acontece se os valores da dissitonia forem grandes e a intensidade do laser variar de forma considerável, em uma distância da ordem do comprimento de onda (HE,2009). Como podemos verificar emSEGUNDO(2000), essa força pode ser denominada como reativa ou dipolar, sendo expressa por

$$\mathbf{F}_{reativa} = -\frac{\hbar\Delta}{4}\frac{\nabla\Omega^2}{4\Delta^2 + \Gamma^2 + 2\Omega^2}. \tag{4.1}$$

Os termos dessa expressão, equação4.1, serão explicados nesta seção, logo mais a frente. EmMiguel(2013), podemos ver que a segunda força, como destacado anteriormente, representa a pressão da radiação devido à absorção seguida pela emissão espontânea de fótons, podendo ser considerada como dissipativa ou espontânea. Essa será dominante para pequenos valores de dissitonia,Miguel(2013) também destaca que este termo está relacionado ao processo de resfriamento. De acordo comSEGUNDO(2000) essa força pode ser representada pela expressão

$$\mathbf{F}_{dissip} = +\hbar\mathbf{k}\frac{\Gamma\frac{\Omega^2}{2}}{2\,\Delta^2 + \frac{\Gamma^2}{4} + \frac{\Omega^2}{2}}. \tag{4.2}$$

Em resumo quando se trata dessas forças, temos que, de acordo comMiguel(2013) a força4.1é conservativa e esse tipo força apresenta dependência com a posição, nesse caso a dependência encontra-se no gradiente do campo $\nabla\Omega$. Dessa forma, como precisamos de uma força que dependa da velocidade, a força que melhor se adapta as nossas necessidades é a força4.2,SEGUNDO(2000) afirma que essa expressão só deve ser considerada se o átomo estiver imóvel e fixo. No entanto, no caso de átomos livres, com prováveis velocidades, a frequência laser observada no referêncial do átomo pode ser modificada com a inserção da velocidade, fenômeno denominado efeito Doppler.

Se considerarmos o movimento do átomo na mesma direção de propagação da luz, e introduzindo o efeito Doppler, a frequência laser irá variar em $\mathbf{kv}$. Dessa forma a dissitonia Δ será $\Delta + \mathbf{kv}$, e a expressão4.2torna-se

$$\mathbf{F}_{dissip} = +\hbar\mathbf{k}\frac{\Gamma\frac{\Omega^2}{2}}{2\,(\Delta + \mathbf{kv})^2 + \frac{\Gamma^2}{4} + \frac{\Omega^2}{2}}, \tag{4.3}$$

onde a denominação dos termos dessa equação é a seguinte: Γ é a taxa com que a população do estado excitado do átomo decai para o estado fundamental, $\Omega = \frac{\mu E(\mathbf{r},t)}{\hbar}$ é a frequência

[1] Para (HE,2009) uma descrição é considerada semiclássica, quando o átomo é quantizado e o campo é clássico.

de Rabi, $\overline{h}$ é constante reduzida de Planck, k valor absoluto do vetor de onda, v o valor da velocidade do átomo, Δ é a diferença $\Delta = \omega - \omega_0$ onde ω_0 representa a frequência de ressonância do átomo (GOMES N. D.; BAGNATO,2014).

Notamos que a solução analítica da equação4.3pode ser de difícil solução (se existir) e os métodos de soluções de EDO's que conhecemos podem não ser suficientes para alcançarmos um resultado. Dessa forma, na próxima seção, sugerimos a solução numérica dessa equação, utilizando os métodos numéricos já explicitados em capítulos anteriores.

A força dissipativa é a força que utilizaremos em nossas simulações. Essa força é válida para cada um dos feixes do melaço. Na seção4.4, estudaremos, via método numérico, a evolução do átomo sobre uma força dissipativa dada pela equação4.3.

4.3 Melaço óptico

Esta seção tem como objetivo explicitar, dentro de um contexto físico, o melaço óptico e o comportamento de átomos quando submetido a feixes lasers.

Para descrevermos teoricamente conceitos relativos ao melaço, vamos fazer uso do ponto de vista defendido por Hänsch e Schawlow, como vimos na introdução desse trabalho, que é a descrição que melhor se adapta ao esfriamento de átomos livres. Para resfriar-mos átomos temos que obter uma força do tipo viscosa, proporcional e contrária a velocidade do átomo. Uma força que melhor se encaixa à esse perfil é uma força do tipo dissipativa (SEGUNDO,2000).

Quando um átomo recebe um *fóton* ele passa para um estado excitado, e no processo de decaimento para o estado fundamental ocorre uma emissão de *fóton*. O processo de emissão de um *fóton* pode ocorrer por emissão estimula ou emissão espontânea. Na emissão estimula não há desaceleração, a argumentação é bastante simples, para esse caso o *fóton* emitido tem momento oposto ao obsorvido, destruindo assim a desaceleração. Mas, já no caso da emissão espontânea, o *fóton* é emitido em direções aleatórias, e dessa forma, o momento líquido das várias emissões é nulo, e o *momento* é transferido para o átomo através da absorção do fóton, como consequência, podemos obter átomos resfriados (HE, 2009).

No processo onde um *fóton* incide em um átomo, há uma troca de momento $\mathbf{P} = \overline{h}\mathbf{k}$, interação desse tipo é governada pela dessintonia $\Delta = \omega - \omega_0$, entre a frequência da luz ω, e a frequência de ressonância dos átomos ω_0 (GOMES N. D.; BAGNATO,2014). Considerando o efeito Doppler, pode-se observar que a mudança de velocidade dos átomos resulta na alteração da frequência, e dessa forma concluimos que, a força que é transmitida pela luz dependerá da velocidade.

É mostrado na Figura15, mostra de forma representativa um grupo de átomos

frios em um melaço. Nesta figura podemos notar que existem vários feixes de luz cruzando em uma determinada região do espaço. É nessa região onde se encontra o melaço.

Essa configuração de Steven Chu é um melaço nas 3 dimensões e usa 3 pares de feixes contrapropagantes. Neste trabalho, vamos estudar apenas um par de feixes, ou seja, um melaço de uma dimensão.

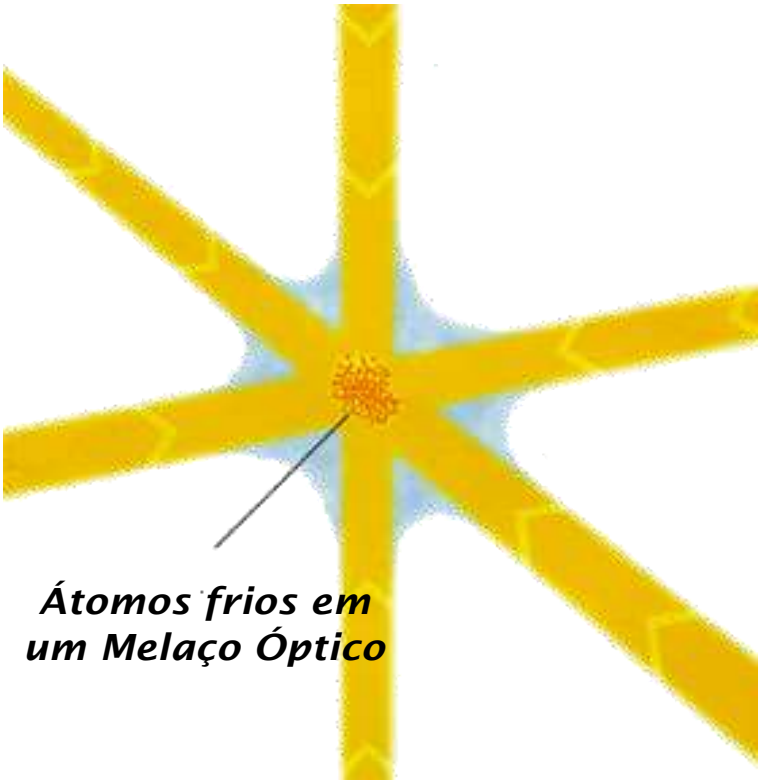


Figura 15 – *Átomos frios em um melaço óptico*.
Fonte: *<http:*
// www.nobelprize.org/ nobel_prizes/ physics/ laureates/ 1997/ illpres/ trapping.html> [2]

4.4 Interação de um único feixe

Anteriormente vimos uma expressão de força para a pressão da radiação que pode ser utilizada para o melaço, notamos que é uma EDO. Nesta seção, vamos fazer o uso do cálculo numérico para obtermos dados suficientes para construção e visualização de gráficos, com o objetivo de analisarmos o comportamento da velocidade e aceleração, ambos em função do tempo. E finalmente a aceleração em função da velocidade.

Iremos utilizar como objeto de estudo o átomo de Césio, para isso precisamos de valores específicos para alguns dos termos da equação4.3. Esses valores podem ser obtidos a partir deSteck(1998). Nesse artigo encontramos algumas das propriedades físicas e ópticas do césio que serão relevantes para o cálculo da equação4.3, pois trataremos dos efeitos

[2] *<http:// www.nobelprize.org/ nobel_prizes/ physics/ laureates/ 1997/ illpres/trapping.html>*

mecânicos da luz em átomos de césio. O valor de cada um dos parâmetros obtidos a partir deSteck(1998) são os seguintes: $\Gamma = 32,87.10^6$Hz, $\Omega = 32,87.10^6$Hz, $\hbar = 1,054.10^{-34}\text{m}^2\frac{\text{kg}}{\text{s}}$ $k = 7,37.10^6$m. É importante esclarecer que vamos começar utilizando como valor para a dissitonia $\Delta = \frac{\Gamma}{2}$.

Notamos ainda que para o caso da expressão4.4, iremos precisar de um valor para m, valor esse que será encontrada no artigoSteck(1998), onde, $m = 2,206.10^{-25}$kg denominado massa atômica do Césio.

Finalmente, temos que ajustar a expressão da força para o melaço, para utilizarmos no código para a simulação. EmNussenzveig(1998), vemos que a $\mathbf{F} = m\mathbf{a} \Rightarrow \frac{d^2\mathbf{x}}{dt^2} = \frac{\mathbf{F}}{m}$, dessa forma para o melaço, no caso desse capítulo, a força é dada por4.3, logo para a nossa simulação iremos utilizar a seguinte expressão:

$$\frac{d^2x}{dt^2} = \frac{\frac{\Gamma\Omega^2\hbar\mathbf{k}}{4(\Delta-\mathbf{k}.\mathbf{v})^2+\Gamma^2+2\Omega^2}}{m}. \tag{4.4}$$

Sabemos que a aceleração é dada por $\mathbf{a} = \frac{d^2\mathbf{x}}{dt^2}$ e a velocidade é $\mathbf{v} = \frac{d\mathbf{x}}{dt}$. Partindo dessas informações podemos utilizar os métodos numéricos para calcular aceleração e velocidade, para este capítulo utilizaremos as bibliotecas GSL ao invés dos métodos clássicos de Runge-Kutta.

Como já destacamos algumas vezes, a resolução analítica da equação4.3exige um embasamento matemático muito bom, no quesito soluções de EDO's. No entanto a dificuldade de resolver essa equação analiticamente pode ser superado através dos métodos numéricos, ver Capítulo2. Dessa forma, iremos fazer uma simulação numérica utilizando a linguagem de programação C++, e de posse dos dados*plotaremos* gráficos utilizando o QtiPlot, para analisarmos o comportamento das entidades físicas em questão.

A simulação consiste em utilizar os valores dos parâmetros de4.3, (STECK,1998), como visto anteriormente. Para obtermos os valores numéricos dessa equação, iremos utilizar os procedimentos citados no Capítulo2, onde obtemos soluções para os osciladores harmônicos. Será necessário apenas, fazer pequenos ajustes nos códigos utilizados, tendo em vista que a equação em questão também é uma EDO. Para a situação de um único feixe laser agindo sobre um átomo de césio, vindo da esquerda (t negativo), podemos obter um gráfico para a velocidade em função do tempo, que conseguimos visualizá-lo a partir figura16.

Para o gráfico exposto na Figura16, foi feito variações para o valor de Δ. Notamos que o átomo se move inicialmente em sentido contrário ao feixe, por isso, ele diminui a velocidade. Depois, ele se move no mesmo sentido, sendo acelerado. Utilizamos como velocidade inicial $17\frac{\text{m}}{\text{s}}$ com uma variação temporal de 0 s à 0,01 s. Podemos notar que para todos os casos haverá uma diminuição da velocidade, a única diferença é que para valores positivos de Δ a velocidade cai quase que no mesmo instante da interação com o feixe

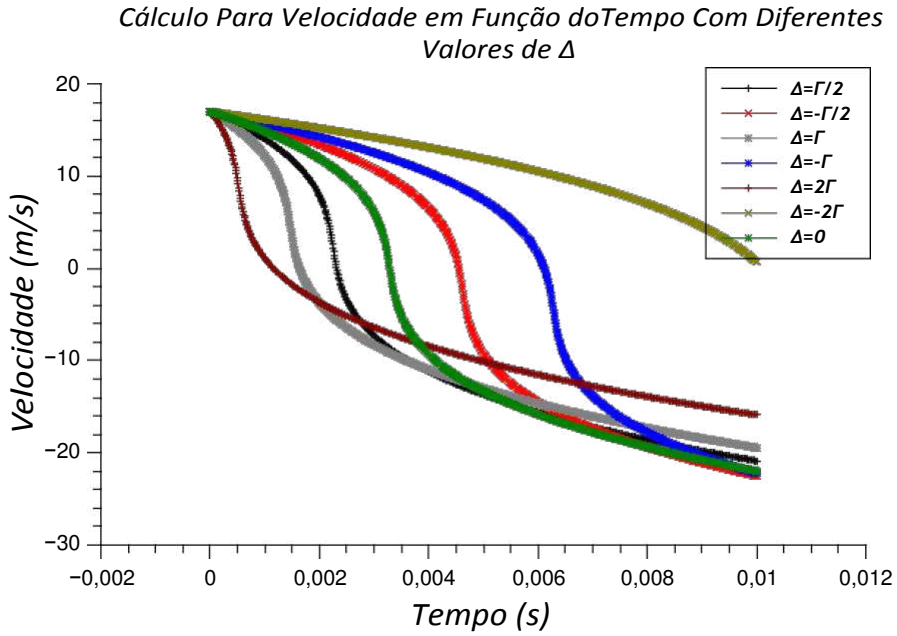


Figura 16 – *Gráfico da velocidade em função do tempo de um átomo com velocidade inicial de 17 m/s contrária ao laser em diversas configurações de dessintonia.*

laser, já para valores negativos, haverá uma demora na queda dessa velocidade. Podemos notar esse fato com mais clareza observando a diferença entre as linhas que representam $\Gamma = 2\Delta$ e $\Gamma = -2\Delta$.

Refazendo os passos anteriores, utilizando os mesmos dados, após a compilação do programa podemos obter um gráfico para a aceleração em função do tempo, a Figura17 mostra esse gráfico.

Notamos pela Figura17uma desaceleração considerável até um ponto mínimo, que são iguais para todos os valores de Δ utilizados, a partir desse ponto acontece um retorno. Semelhante ao que acontece com a velocidade vemos que para valores negativos de Δ haverá uma demora para uma diminuição considerável da aceleração.

Podemos observar tanto através da Figura16como pela17, uma queda notável na velocidade, ou seja, uma desaceleração dos átomos de césio quando interagem com um feixe laser. Na próxima seção vamos inserir outra expressão para força que representará um outro feixe laser.

4.5 Dois feixes contrapropagantes: Melaço ótico 1D

Vimos anteriormente como se comporta átomos de césio quando expostos a um feixe laser, agora iremos observar o caso onde átomos de césio são expostos a dois feixes lasers contrapropagantes, seguindo um procedimento semelhante ao da seção anterior.

Se para o caso de um feixe, a resolução analítica da equação4.3já apresenta

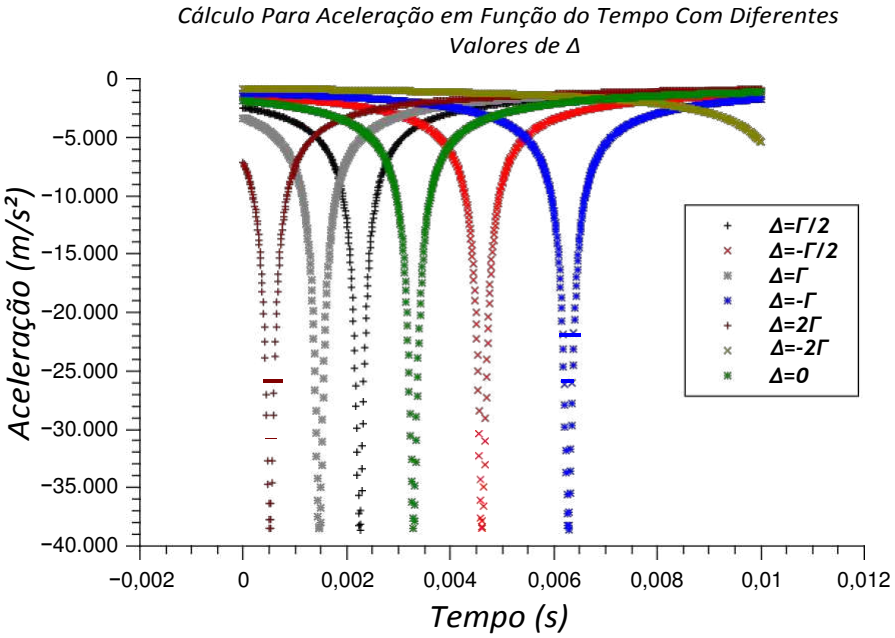


Figura 17 – *Gráfico da aceleração em função do tempo, para o caso de um único feixe laser, deslocando-se em direção contrária ao átomo.*

dificuldades, podemos imaginar que será muito mais difícil para vários feixes. Dessa forma, utilizando como base a situação exemplificada na seção4.4, vamos adicionar um feixe contrapropagante e resolver a expressão obtida, numericamente. O processo será semelhante ao utilizado para o caso de um feixe. Mas agora, a equação da força para o melaço no caso de dois feixes contrapropagantes será expressa por4.5. Contudo é necessário haver igualdade de intensidade e frequência, entre a força de um feixe e a força de dois feixes contrapropagantes.

$$\mathbf{F}_{mel} = \mathbf{F}_{+} + \mathbf{F}_{-} = \frac{\Gamma\Omega^2\hbar\mathbf{k}}{4(\Delta - \mathbf{k}.\mathbf{v})^2 + \Gamma^2 + 2\Omega^2} - \frac{\Gamma\Omega^2\hbar\mathbf{k}}{4(\Delta + \mathbf{k}.\mathbf{v})^2 + \Gamma^2 + 2\Omega^2} \tag{4.5}$$

Notar que, no segundo termo dessa expressão é utilizado uma velocidade positiva, ou seja, na direção da força. Para os parâmetros serão utilizados os mesmos valores da seção anterior. Então, compilando o programa e plotando os gráficos, para o caso de dois feixes lasers contrapropagantes, obtemos a Figura18, que representa a velocidade, para esse caso.

É importante notar que a Figura18, mostra o comportamento da velocidade em função do tempo. As configurações utilizadas são semelhantes as que foram utilizadas para o caso de um feixe, velocidade inicial $17\frac{m}{s}$ e variação temporal de 0 s a 0,01 s. Para esse caso, uma interpretação física importante dá-se para os valores positivos de Δ. Vemos claramente uma queda na velocidade parando próximo ao zero, isso significa que o átomo foi resfriado, ou seja diminuiu sua velocidade. Concluímos também que os valores negativos de Δ não são consideráveis, caso o objetivo seja resfriar átomos.

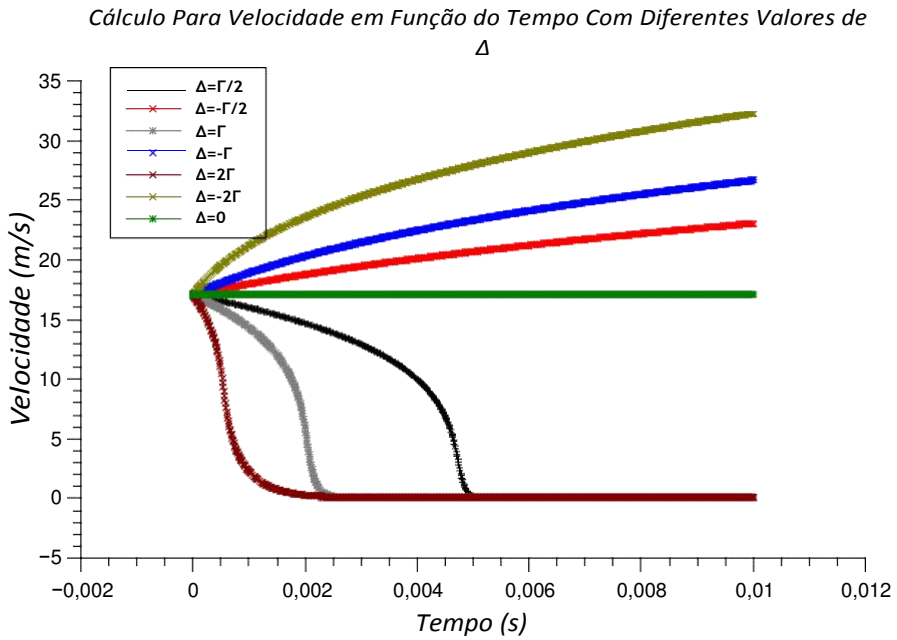


Figura 18 – *Gráfico da velocidade em função do tempo, para o caso de dois feixes lasers contrapropagantes.*

Seguindo a mesma linha de raciocínio, podemos obter um gráfico da aceleração em função do tempo, a Figura19mostra com clareza o comportamnto da aceleração para o caso de dois feixes.

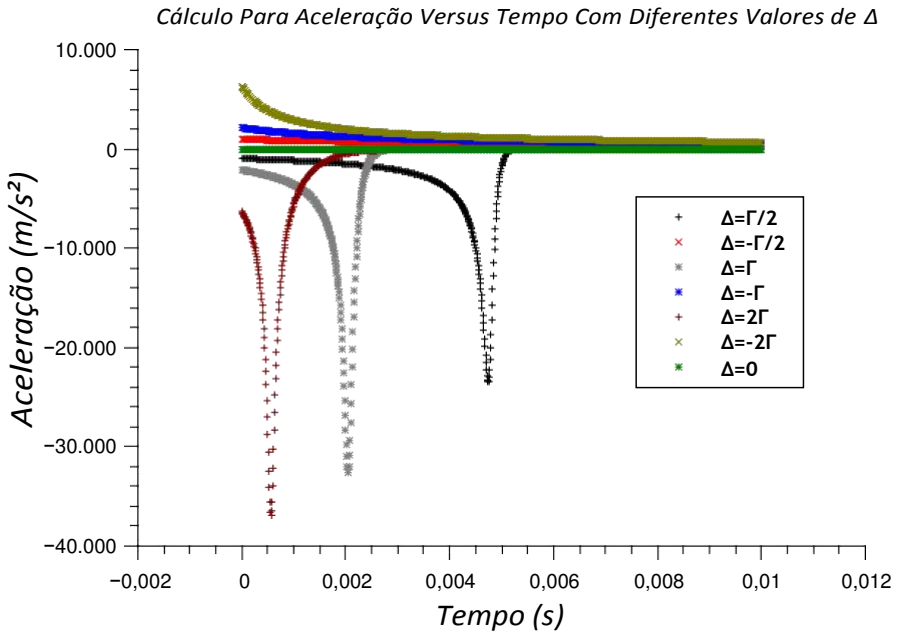


Figura 19 – *Gráfico da aceleração em função do tempo, para o caso de dois feixes lasers contrapropagantes.*

Observando o gráfico representado pela a Figura19, podemos notar uma queda brusca na aceleração e depois de um pequeno intervalo de tempo ela retorna, tendendo

para zero.

Dessa forma, tanto a partir do gráfico da Figura18como da19, podemos ver claramente a diminuição da velocidade do átomo quando exposto a uma situação onde existem dois feixes lasers contrapropagantes, e podemos então comprovar, graficamente, que é possível resfriar átomos, ou seja, obter o melaço óptico através da força da pressão da radiação.

4.6 Teste do método numérico

O objetivo dessa seção é comparar se os resultados obtidos numericamente nas seções anteriores são semelhantes com os resultados analíticos, caso seja semelhantes podemos concluir que são verdadeiras as conclusões obtidas anteriormente.

Quando o objetivo é comparar resultados numérico *versus* analítico, para o caso do melaço, podemos citar como exemplo duas formas: diretamente através da expressão da força, utilizando como base a expressão4.3, ou podemos simplesmente calcular a aceleração em função da velocidade através de um programa C++. Como sabemos, o cálculo analítico da expressão4.3, pode apresentar uma grande dificuldade, como já mencionado anteriormente.

Para obtermos o gráfico da aceleração em função velocidade, nessa seção, utilizamos um bloco de linhas no mesmo programa C++ utilizado para o cálculo dos valores numéricos nas seções anteriores.

Primeiro vamos analisar a situação para o caso de um feixe laser contrapropagante. De início após a compilação do programa, utilizamos os dados para valores variando de $\Delta = \Gamma$ à $\Delta = -\Gamma$, o resultados de um gráfico da aceleração em função da velocidade, pode ser visualizado no gráfico da Figura20.

Agora vamos observar um gráfico do cálculo analítico da aceleração em função da velocidade, Figura21, obtido através do programa C++ citado anteriormente. Para este caso, foi utilizado $\Delta = -\frac{\Gamma}{2}$. Notamos uma enorme semelhança entre o gráfico da Figura 21e a linha vermelha do da Figura20, que representa justamente os dados numéricos de $\Delta = -\frac{\Gamma}{2}$.

A seguir vamos fazer uma comparação entre as curvas para observar com mais clareza essa semelhança entre os resultados.

A partir da comparação visualizada na Figura22, podemos ver com nitidez o acoplamento das curvas, e partir dessa semelhança comprovamos que os resultados numéricos obtidos nas seções anteriores podem ser utilizados sem nenhum problema, pois os resultados numéricos são semelhantes aos analíticos.

Por fim, vamos analisar o caso de dois feixes contrapropagantes, e utilizando o

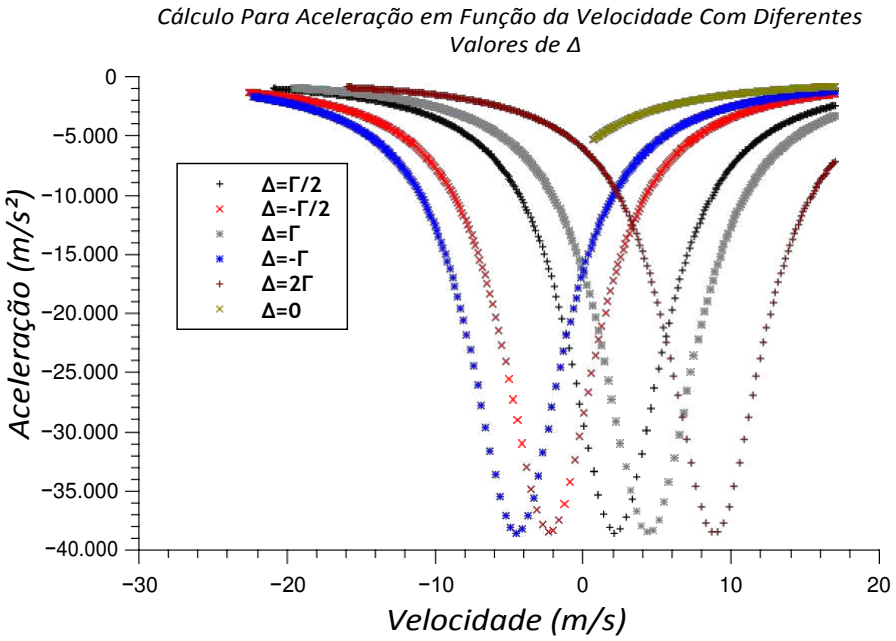


Figura 20 – *Gráfico da aceleração em função da velocidade, para o caso de um único feixe laser, deslocando-se em direção contrária ao átomo.*

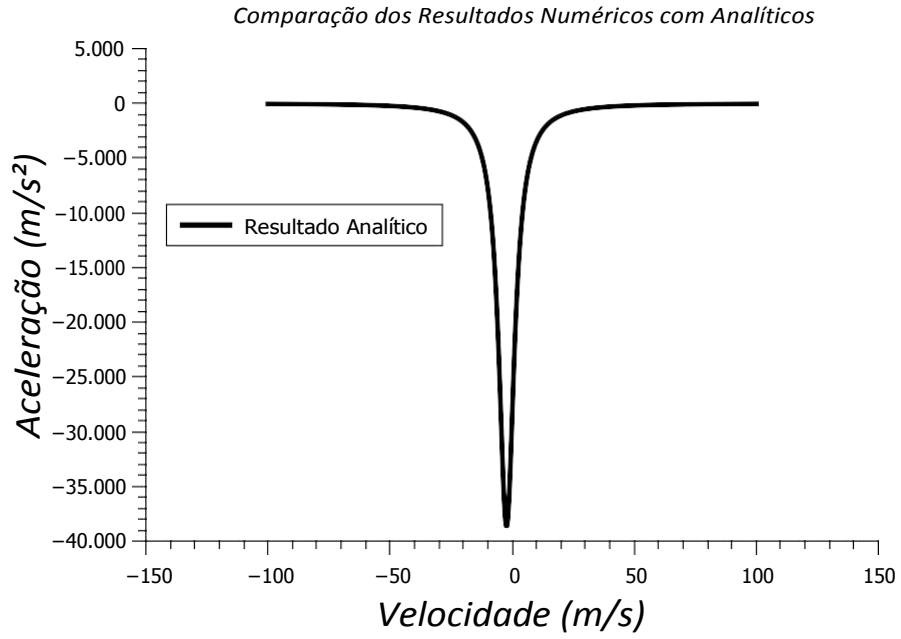


Figura 21 – *Gráfico que representa a solução analítica da aceleração em função da velocidade, para um caso de um único feixe laser, deslocando-se em direção contrária ao átomo.*

raciocínio semelhante ao utilizado para o caso de um feixe, vamos primeiro observar o gráfico numérico da aceleração em função da velocidade, fazendo uma variação de valores de $\Delta = \Gamma$ à $\Delta = -\Gamma$.

Dentro da variação de Γ mostrada na Figura23, podemos ver que apenas os valores

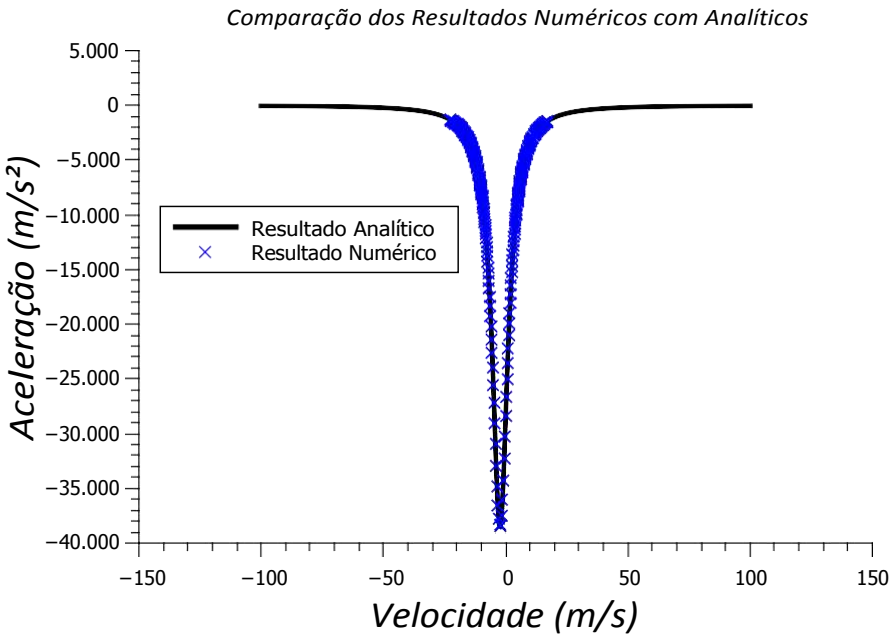


Figura 22 – *Comparação entre as curvas que representam os resultados numérico e analítico, para um caso de um único feixe laser, deslocando-se em direção contrária ao átomo.*

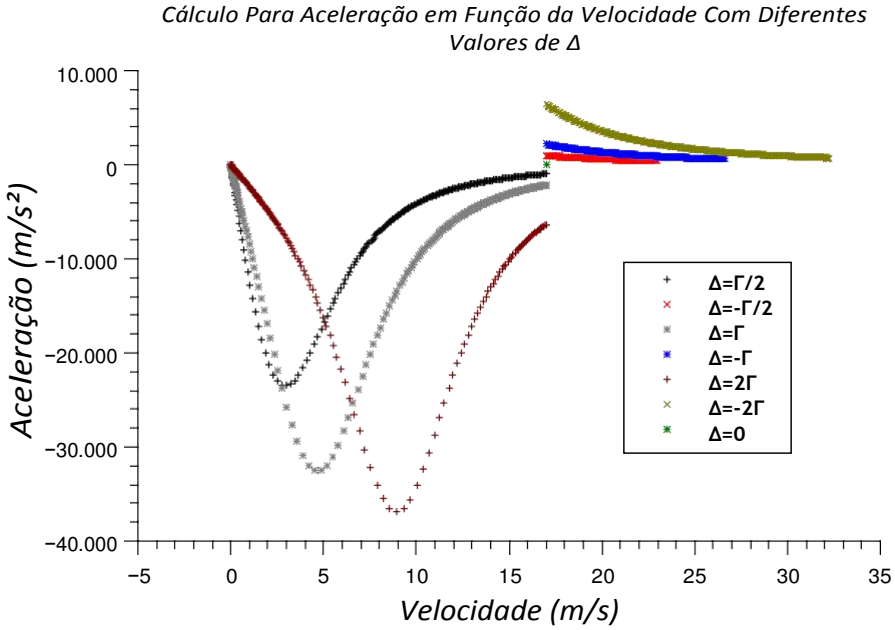


Figura 23 – *Gráfico da aceleração em função da velocidade, para o caso de dois feixes lasers contrapropagantes.*

positivos de Δ são mais adequados para fazermos a comparação de resultados.

Agora vamos verificar como se comporta o gráfico do resultado analítico da aceleração em função da velocidade para o caso de dois feixes contrapropagantes, como vimos na Figura23. Nem todos os valores de Δ são interessantes, dessa forma, vamos utilizar

$\Delta = \Gamma$.

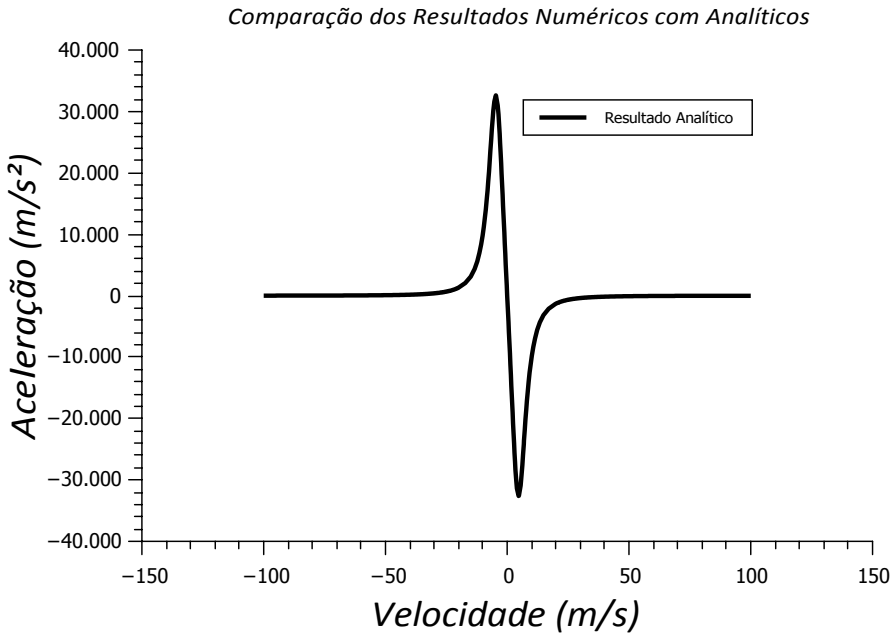


Figura 24 – *Gráfico que representa a solução analítica da aceleração em função da velocidade, para o caso de dois feixes lasers contrapropagantes.*

Por fim, vamos utilizar a curva de $\Delta = \Gamma$ do gráfico da Figura23e comparar com a curva da Figura24, para observamos finalmente se para o caso de dois feixes contrapropagantes os resultados numéricos são semelhantes aos resultados analíticos.

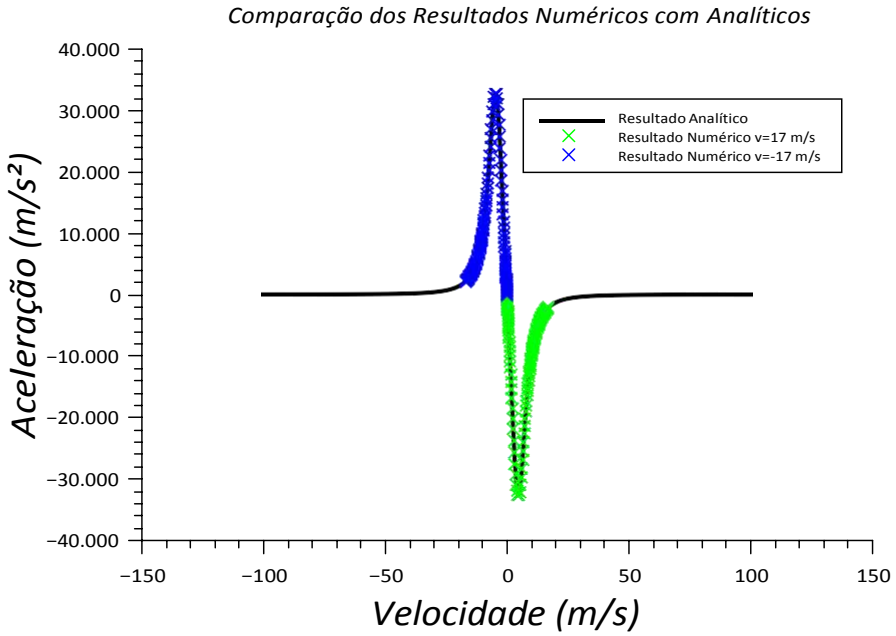


Figura 25 – *Comparação entre as curvas que representam os resultados numérico e analítico, para o caso de dois feixes lasers contrapropagantes.*

Observamos claramente, a partir da Figura25, que as curvas se sobrepuseram, desta forma os resultados numéricos e analíticos são semelhantes.

4.7 Conclusão

Neste capítulo, vimos as forças sobre os átomos devido a interação com feixe de radiação. Simulamos a interação com um único feixe e verificamos a mudança de velocidade de forma viscosa, sempre reduzindo a velocidade quando contrária a velocidade e aumenta quando favorável. A simulação do melaço em uma dimensão, com dois feixes, mostra a redução da velocidade independente do sentido. Essa redução de velocidade é a redução da temperatura.

Fizemos, por questão de controle, comparação entre a simulação e o teórico e observamos que segue a relação aceleração pela velocidade.

Notamos claramente que a velocidade dos átomos diminuiriam com a utilização da força para o caso de dois feixes contrapropagantes. E finalmente, concluímos através dos gráficos da aceleração em função da velocidade, que os resultados numéricos coincidem com os analíticos.

5 Conclusões e Perspectivas

Nesse trabalho, partimos dos conceitos da teoria atômica para entendermos os princípios da pressão da radiação. Utilizamos uma expressão da força para a pressão de radiação para realização de uma simulação numérica utilizando um e dois feixes lasers.

Preparamos um ambiente de trabalho propício para a compilação de programas C++, explicamos passo a passo como efetuar a instalação do compilador bem como iniciar os primeiros programas.

Detalhamos os fundamentos básicos dos métodos numéricos, e posteriormente utilizamos como ferramentas para simulações numéricas.

Por fim, utilizamos todas as ferramentas estudadas durante o trabalho para aplicar na teoria do melaço óptico, onde mostramos através de dados obtidos através de simulações numéricas que é possível resfriar átomos. Obtemos nossos resultados através de analises de gráficos das velocidade em função do tempo e da aceleração em função do tempo.

No final do trabalho, partimos de um gráfico analítico da aceleração em função da velocidade para provar que os dados numéricos são compatíveis com os resultados teóricos, fizemos isso através de uma comparação gráfica teórico versus analítico.

Esse trabalho se resume apenas ao resfriamento de átomos, sendo esse o passo inicial para o processo de aprisionamento, onde passa a ser considerado a inserção de um campo magnético. Logo esse trabalho pode ser aproveitado para uma possível continuação.

Referências

SEGUNDO, P. C. de S. *Efeitos mecânicos da interação laser-átomo: garrafa ótica e armadilha magneto-ótica*. Dissertação (Mestrado) — Universidade Federal da Paraíba, Centro de Ciêcias Exatas e da Natureza, Coordenação de Pós-Graduação em Física, João Pessoa - Paraiba, 2000. Citado 4 vezes nas páginas19,20,58e59.

HE, H. Z. *Construção de Uma Armadilha Magneto-Ótica para Aplicações em Informação Quâantica e Física Atômica*. Dissertação (Mestrado) — Universidade de São Paulo, Instituto de Física, São Paulo, 2009. Citado 4 vezes nas páginas20,21,58e59.

GOMES N. D., M. A. C.; BAGNATO, V. S. Modelo clássico para resfriamento atômico: Uma forma pedagógica de entender o problema. São Paulo - SP, 2014. Citado 2 vezes nas páginas21e59.

BOYCE, W. E.; DIPRIMA RICHARD C, N. Y. *Elementary differential equations and boundary value problems*. New York: [s.n.], 1969. Citado 2 vezes nas páginas24e25.

ZILL, D. G. *Equações diferenciais com aplicações em modelagem*. São Paulo: Peason Makron Books, 2001. Citado na página25.

NAGLE EDWARD B. SAFF, A. D. S. R. K. *Equações diferenciais. 8ª Edição; vol. Único.* Sao Paulo: Pearson Education do Brasil Ltda, 2012. Citado 2 vezes nas páginas25e26.

NUSSENZVEIG, H. M. *Curso de Fısica Básica, vol. 1* : Ótica, relatividade e física quântica. Sao Paulo: Editora Edgard Blucher Ltda, 1998. Citado 3 vezes nas páginas19, 26e27.

RUGGIERO M.A.G., L. V. S. P. *Cálculo Numéricos. 2 Edição*: Aspectos teóricos e computacionais. Sao Paulo: Ed. MacGraw-HiII, 1988. Citado na página29.

NAYARA, K. *Métodos Numéricos de Euler e Runge-Kutta*. Dissertação (Mestrado) — UFMG - Universidade Federal de Minas Gerais, Belo Horizonte, 2012. Citado 2 vezes nas páginas29e30.

SYMON, K. R. *Mecânica, vol. Único*. Rio de Janeiro: Editora Campus Ltda, 1996. Citado na página32.

NUSSENZVEIG, H. M. *Curso de Fısica Básica, vol. 2* : Ótica, relatividade e física quântica. Sao Paulo: Editora Edgard Blucher Ltda, 1998. Citado 6 vezes nas páginas32, 33,34,36,61e91.

MARTINS, J. B. *A história do átomo: de Demócrito aos quarks*. [S.l.]: Editora Ciência Moderna, 2002. Citado 2 vezes nas páginas45e46.

CARUSO, F. *Fisica Moderna.*: Origens clássicas e fundamentos quãnticos. Rio de Janeiro: Elsevier, 2006. Citado na página46.

TIPLER PAUL A.; LLEWELLYN, R. A. *Fisica Moderna. 3ª Edição*. Rio de Janeiro: Editora LTC, 2010. Citado na página47.

EISBERG, R. *Fısica quântica*. [S.l.]: Editora Campus, 1979. Citado 3 vezes nas páginas 47,48e54.

NUSSENZVEIG, H. M. *Curso de Fısica Básica, vol. 4* : Ótica, relatividade e física quântica. Sao Paulo: Editora Edgard Blucher Ltda, 1998. Citado 3 vezes nas páginas47, 49e50.

ZEMANSKY, S. *Física IV. 12ª Edição; vol.4* : Ótica e física moderna. Sao Paulo: Prentice Hall (Grupo Pearson), 2009. Citado 5 vezes nas páginas49,50,52,53e54.

RESNICK, R.; HALLIDAY, D.; WALKER, J. *Fundamentos de física. 8ª edição, vol.4* : Óptica e física moderna. Rio de Janeiro: LTC - Livros técnicos e científicos S.A, 2009. Citado 2 vezes nas páginas51e52.

TIPLER, P. A. G. M. *Fisica Para cientistas e engenheiros, vol.2* : Eletricidade e magnetismo, óptica. Rio de Janeiro: LTC - Livros técnicos e científicos S.A, 2006. Citado na página51.

ZEMANSKY, S. *Física III. 12ª Edição, vol. 3* : Eletromagnetismo. Sao Paulo: Prentice Hall (Grupo Pearson), 2008. Citado na página51.

VILLATE, J. E. *Física 2. Eletricidade e Magnetismo*: Eletricidade e magnetismo. [S.l.: s.n.], 2012. Citado na página51.

MIGUEL, M. L. *Técnicas de resfriamento e aprisionamento de átomos aplicadas a átomos de estrôncio*. Dissertação (Mestrado) — Universidade de São Paulo, Instituto de Física de São Carlos, São Paulo, 2013. Citado na página58.

STECK, D. A. Cesium d line data. Los Alamos, 1998. Citado 2 vezes nas páginas60e61.

MARKET. *Desktop Operating System Market Share*. 2016. [Online; accessed 20-agosto-2016]. Disponível em:<http://netmarketshare.com/>. Citado na página78.

AHO RAVI SETHI, J. D. U. R. d. J. A. V. *Compiladores*: princípios, técnicas e ferramentas. Sao Paulo: LTC - Livros Técnicos e científicos Editora S.A, 1995. Citado na página79.

CODELITE. *O que é codelite?* 2016. [Online; accessed 20-agosto-2016]. Disponível em: <http://codelite.org/>. Citado na página79.

FOUNDATION, F. S. *GSL - GNU Scientific Library*. 2016. [Online; accessed 20-agosto-2016]. Disponível em:<http://www.gnu.org/software/gsl/>. Citado 3 vezes nas páginas39,82e83.

MATHGL. *MathGL*. 2016. [Online; accessed 20-agosto-2016]. Disponível em: <http://mathgl.sourceforge.net/&prev=search>. Citado na página83.

Apêndices

APÊNDICE A – PREPARAÇÃO DOS SOFTWARES

A.1 Introdução

Este Apêndice consiste na inserção de ferramentas fundamentais e essenciais para os passos da metodologia numérica que será utilizada na realização deste trabalho, sendo os métodos numéricos primordiais para o sucesso. Este Apêndice também oferece suporte para outros tópicos de grande importância. Serão sanadas dúvidas referentes ao tipo de sistema operacional adequado, para o desenvolvimento de algum tipo de programa.

Mas, além de um sistema em si, é necessário um compilador de boa qualidade, e veremos que o Codelite pode suprir todas as necessidades, em princípio. Considerado por alguns conhecedores da área por sua poderosa eficácia, apresenta a priori, apenas um fator negativo, esse compilador ainda não apresenta suporte de idioma em português. A seção A.3, explica de forma clara e concisa os passos para a instalação correta desse compilador.

As seçõesA.4eA.5, trata do *GSL* e do *MathGL*, respectivamente. Essas duas bibliotecas são poderosíssimas. O *GSL* é de extremo valor para a realização de cálculos numéricos e científicos e o *MathGL* pode ser utilizado como ferramenta opcional por oferecer um amontoado de aplicações para a criação de gráficos científicos. A instalação de 7 adicionais extras "*cmake-gui, zlib1g-dev, libpng12-dev, libhpdf, libgsl0-dev, libgifdev, libgl1-mesa-dev*"adequa o sistema para inúmeras funções na área de programação, o processo de instalação destes adicionais são explicitados com clareza na seçãoA.6. Após a preparação completa do sistema, ainda pode ser relevante verificar a eficácia do mesmo, essa é a proposta da seçãoA.7. Por fim a seçãoA.8demonstra informações valiosíssimas sobre a instalação e utilização do *QtiPlot*.

A.2 A escolha do sistema operacional

Para quem quer dar os primeiros passos na área de programação científica, a escolha de um sistema operacional pode ser uma dúvida demasiadamente inquietante. Dessa forma, nesta seção vamos fazer uma breve síntese explicativa, sobre motivos do porquê escolher o linux como sistema operacional para a elaboração de bons programas. Veremos quais os sistemas mais utilizados, e meios diferentes de utilizarmos um ou mais sistemas operacionais em um microcomputador.

Um dos fatores primordiais para construção tanto de programas simples como

de simulações computacionais mais completas, é o aparato adequado para este processo, logo a escolha do sistema operacional é de extrema importância. Quando se trata de sistemas operacionais, as opções, em algumas circunstâncias, podem torna-se limitadas. Uma pesquisa rápida pela internet mostra quais são os tipos de sistemas operacionais mais utilizado em desktop, segundomarket(2016) o Microsoft Windows (***Windows***) lidera o ranking com mais 90%. Em segundo, vem o Macintosh Operating System (***Mac***) ultrapassando 7% e em terceiro o GNU/Linux (***Linux***) com menos de 2%. Apesar de não ser o sistema operacional mais utilizado, o *Linux* é o sistema mais adequado para construir programas e tarefas semelhante a essa. Isso pode ser observado em uma infinidade de experiências partilhada por conhecedores da área, também é válido esclarecer que não existe um melhor sistema que outro, mas sim um que se adapta melhor a cada utilizador ou que é mais produtivo em determinada função. Sem sombra de dúvidas a particularidade essencial do *Linux* é *facilidade de instalação e de manipulação de ambientes de desenvolvimento*.

Dentro das várias distribuições *Linux* disponibilizadas, o ***Ubuntu*** torna-se uma escolha considerável pois além de ser uma das mais utilizadas pelos usuários do *Linux*, existem outros motivos que fortalece essa escolha, como por exemplo: facilidade de manuseio e semelhança gráfica com outros sistemas. A versão *Ubuntu* utilizada para a realização desse trabalho foi a versão 16.04.

É importante reforçar que existe mais de uma maneira adequada para utilizar o *Ubuntu* em um computador, algumas maneiras mais importantes merecem destaque: a instalação de uma máquina virtual dentro do *Windows* para emular o *Linux* a partir de um programa [1], utilizar *dual-boot* [2], ou simplesmente pode-se utilizar o *Ubuntu* como sistema único. A segunda e a terceira opção, respectivamente, são consideradas as mais adequadas por apresentar mais instabilidade e segurança.

Infinidades de referências podem ser encontradas, armazenadas na internet, indicando passo a passo como dever ser feita a instalação de um sistema operacional (em particular o *Ubuntu*), como também os passos corretos para executar o *dual boot* em um *desktop*.

Como visto, a escolha de um sistema pode facilitar ou dificultar na criação de programas, e o sistema de mais versatilidade para programação é o linux.

A.3 Instalação correta do compilador

Existem alguns poderosos aliados que facilitam uma efetivação eminente de um trabalho na área de programação, são os ditos ***compiladores***. Em síntese, um *compilador*,

[1] Uma das máquinas virtuais mais utilizadas é a *virtual box*.

[2] Multi boot é um método que permite a escolha de um entre vários sistemas operacionais instalados em um mesmo microcomputador, na hora da inicialização.

é primordial para um desenvolvimento considerável dessa tarefa. Nesta seção será dada uma introdução sutil do conceito de compiladores. Será também explicado, passo a passo, como instalar um *compilador*, em particular o ***CodeLite***[3], seguido de instruções básicas de como elaborar um primeiro programa.

Um *compilador*, segundoAho Ravi Sethi(1995), é um programa que lê um programa escrito em uma linguagem fonte, e o traduz em um outro programa equivalente, em uma outra linguagem alvo. Onde essa linguagem alvo é igualmente variada, podendo ser outra linguagem de programação ou uma linguagem de máquina de qualquer coisa, entre um microprocessador e um supercomputador. Ainda utilizando a referência (AHO RAVI SETHI,1995), podemos comentar que, existe uma variedade de compiladores que pode parecer assustadora, quando se trata de linguagens fonte. Além das linguagens de programação tradicionais, *Fortran*, *pascal*, ***C/C++***, existem muitas outras.

Uma harmonia considerável pode ser obtida, de uma combinação sublime entre o sistema operacional ***Ubuntu*** e o compilador ***Codelite***. Em conformidade comCodelite (2016), *Codelite* é um compilador de uma fonte aberta, livre, IDE multi-plataforma especializada em *C*, *C++*, *PHP*, *JavaScript* e linguagens de programação que funciona em todas as principais plataformas (*Mac OS*, *Windows* e ***Linux***).

A seguir, vamos indicar passo a passo como instalar o Codelite no Ubuntu. Para a instalação desse ambiente de programação uma atenção extra é necessária. Para se obter êxito, se faz necessário a utilização dos procedimentos indicado no site "<http: //codelite.org/LiteEditor/Repositories#toc1>", ou pode-se basicamente utilizar o terminal, inserindo os seguintes códigos na linha de comando [4]

> *sudo apt-key adv –fetch-keys http://repos.codelite.org/CodeLite.asc*
> *sudo apt-add-repository 'deb http://repos.codelite.org/ubuntu/ xenial universe'*
> *sudo apt-get update*
> *sudo apt-get install codelite wxcrafter*

Cada um desses comandos corresponde a uma determinada função. Respectivamente, o primeiro adiciona ao *Codelite* uma chave pública para evitar avisos de *apt*/*aptitude*, o segundo adiciona os repositórios, nessa linha a versão do *Ubuntu* merece atenção [5], o terceiro comando, atualiza os repositórios e finalmente o quarto instala a versão completa do programa.

Após o procedimento citado, o compilador já deve torna-se útil e compilar programas simples. O exemplo clássico é o *hello world*, e deve demonstrar o sucesso da instalação do programa. O mecanismo para a construção desse *"primeiro programa"*, em alguns casos, pode parecer difícil e causar desconforto para usuários iniciantes. Sendo assim, será

[3] Os passos demonstrados nesta seção são referentes ao ***codilite 9.2.0***.
[4] É importante deixar claro que, após atualização de sistema, haverá mudanças nessas linhas.
[5] O *Ubuntu* lança novas versões semestralmente com nome de código e número de versão diferente.

demonstrado a seguir, de forma clara, como proceder.

Após abrir o programa, pode ser observado algo similar à Figura26, observamos em destaque o ícone ***New Workspace Creat a new workspace***, clicando nele vamos obter a Figura27. Caso queiramos, podemos simplesmente percorrer o seguinte caminho: barra de menu **Workspace > Create New Workspace**, Figura27. Logo após vamos obter outra aba, Figura28. A finalidade dessa janela é fornecer lugar para nomearmos o espaço de trabalho. Obtendo êxito nesse processo descrito anteriormente podemos seguir para o próximo passo.

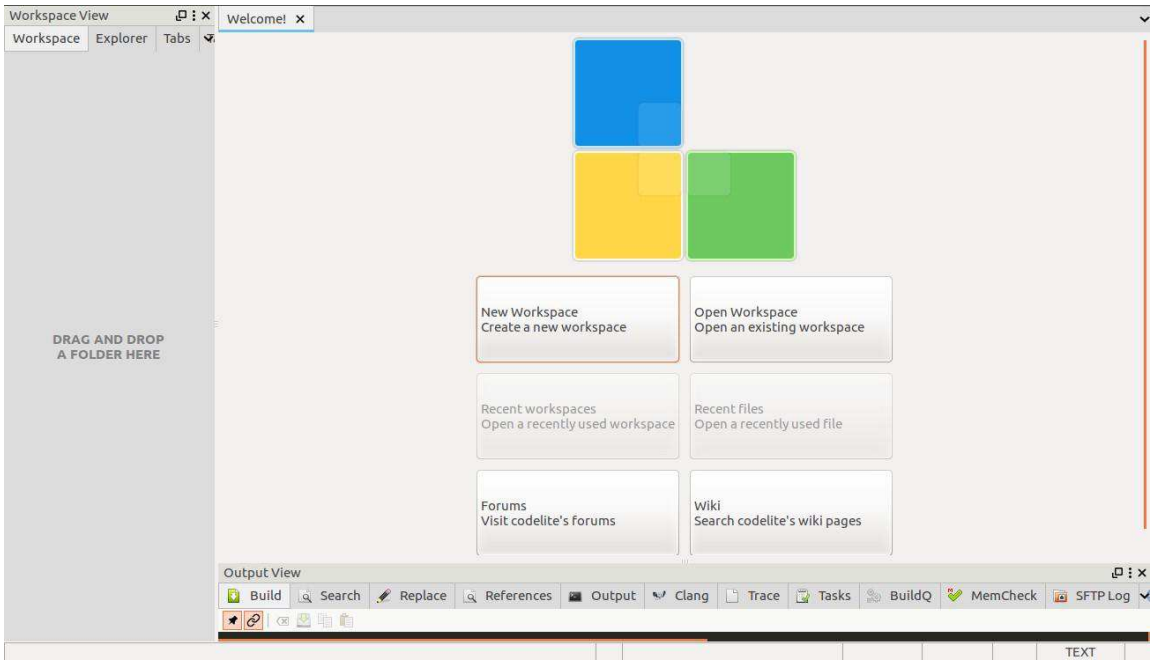


Figura 26 – *Tela inicial do CodeLite*.

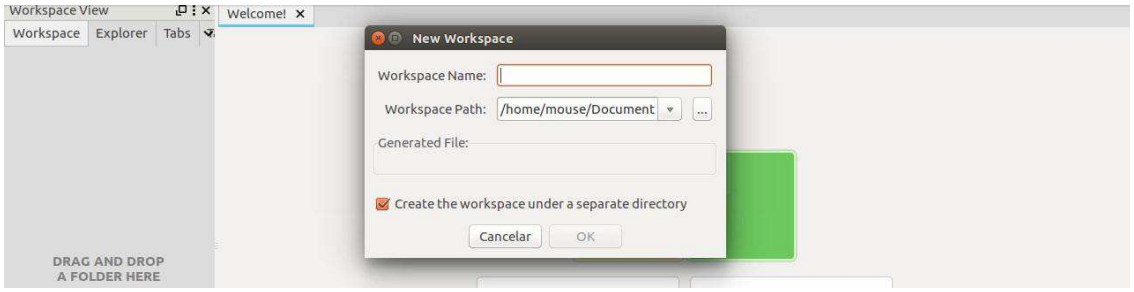


Figura 27 – *Janela preparada para nomear o workspace*.

Após criado o espaço de trabalho, devemos criar um novo projeto. Analogamente a criação do *workspace*, temos mais de uma opção para criação de um *New Project*. Para o primeiro método, basta apenas acessarmos a barra de ferramentas e clicar em **Workspace > New Project**, Figura29. Opcionalmente podemos clicar com o botão direito do *mouse*, no ícone correspondente ao *New Workspace* criado anteriormente, Figura30. Por fim basta nomearmos o novo projeto.

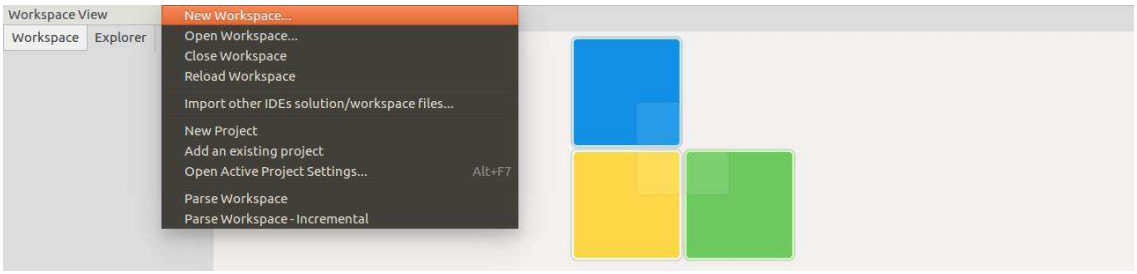


Figura 28 – *Criando um "New Workspace"através de um método alternativo.*

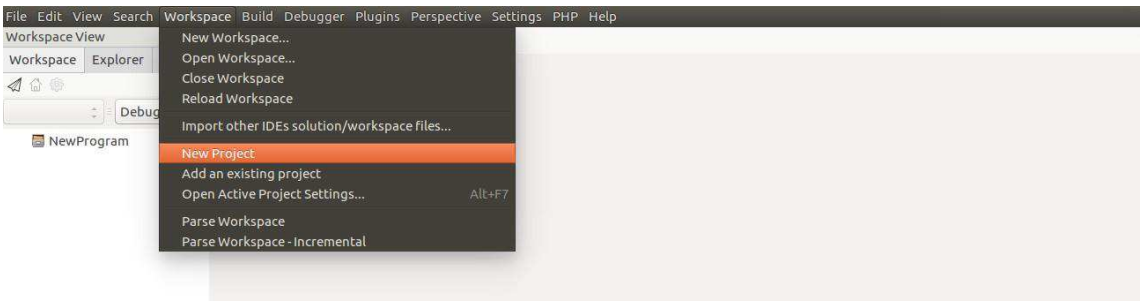


Figura 29 – *Criando um novo projeto "New Project".*

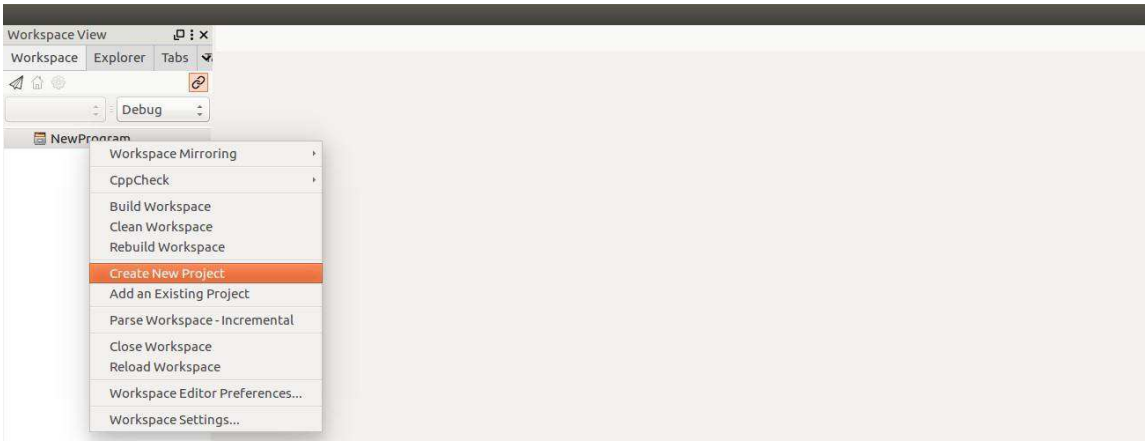


Figura 30 – *Método alternativo para a criação de um new project.*

Finalmente o compilador está pronto para compilar o programa, por padrão o CodeLite já cria automaticamente um programa que imprime no terminal a mensagem *hello world*, basta apenas compilá-lo. Utilizando novamente a barra de ferramentas, selecionamos ***Build > Run***, Figura31. A consequência desse processo anterior, será à aquisição da fabulosa mensagem *hello world* no terminal.

Se o procedimento anterior foi seguido gradativamente, como sugerido, e tudo ocorreu dentro do esperado, é provável que o compilador esteja suficientemente preparado para exercer sua função na elaboração de programas simples. Essa seção pode ser associada com aA.6, para um melhor aperfeiçoamento do compilador instalado aqui. Caso haja a necessidade, algumas reformulações devem ser feita pelo utilizador, mas para o desenvolvimento desse trabalho, os fundamentos apresentados nesta seção são mais que suficientes.

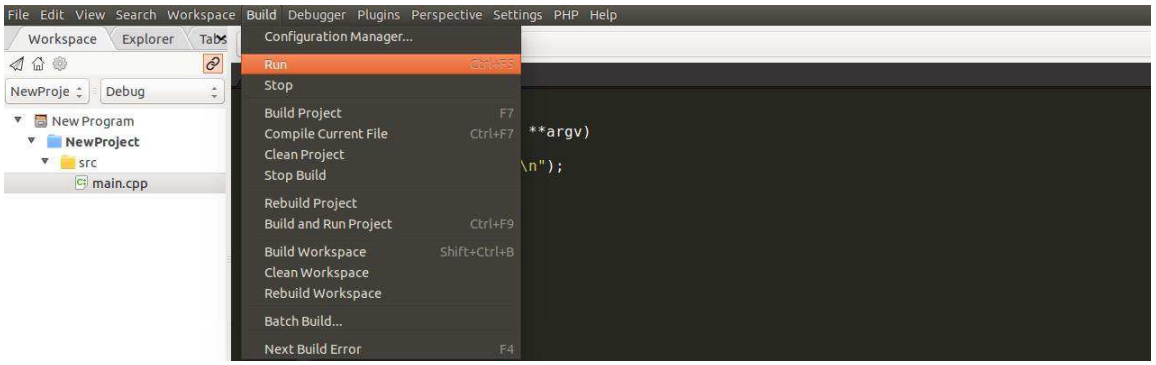


Figura 31 – *Caminho indicado para a compilação de um projeto*.

Dessa forma, podemos utilizar esses argumentos, para compreender superficialmente o conceito de *compilador*, e aprender nitidamente como instalar o *codelite* em um computador que tenha o *Ubuntu* instalado e atualizado.

A.4 Introduzindo o GSL

Na seção anterior foi visto como instalar um compilador, em principio isso já é suficiente para criação de alguns programas simples, mas podemos fortalecer esse compilador, tornando-o mais eficaz para a criação de programas mais elaborados, isso pode ser feito implementando as bibliotecas [6]. Veremos agora uma introdução simples sobre um ***GSL***.

Existem vários tipos de bibliotecas, estas fornecem ferramentas essenciais na aquisição de programas mais complexos. Quando refere-se à cálculos numéricos e científicos a *GSL* merece destaque. Esta biblioteca fornece centenas de funções matemáticas fundamentais para trabalhar com números complexos, álgebra linear, vetores, matrizes, números aleatórios, entre outros. Conforme especificado em (FOUNDATION,2016), a *Biblioteca Científica GNU (GSL)* é uma coleção de rotinas para computação numérica. Essas rotinas foram escritas a partir do zero, na linguagem de programação C, e mostram uma moderna *API (Applications Programming Interface)* para programadores em C, permitindo que invólucros sejam escritos para linguagens de nível mais alto. O código fonte é distribuído sob a Licença Pública Geral (*GNU*).

Para facilitar a manipulação desta Biblioteca, pode-se utilizar um manual disponibilizado em "<http://www.gnu.org/software/gsl/>", nesse endereço, será descrito como usar as rotinas desse manual. Em cada capítulo pode ser encontrado definições detalhadas de várias funções, seguido de programas exemplo e referências para artigos nos quais os algoritmos são baseados.

Alguns diferenciais sobre o *GSL* são claramente especificados em (FOUNDATION,

[6] Coleção de códigos que resolve problemas mais específico, na elaboração de um programa.

2016), segundo essa referência, além das subrotinas na Biblioteca Científica GNU serem *software livre* [7], a biblioteca utiliza um projeto orientado a objetos. Diferentes algoritmos podem ser utilizados com facilidade, e até mesmo alterados em tempo de execução sem a necessidade de recompilar o programa. O *GSL* é destinado a usuários científicos comuns, e qualquer um que conheça um pouco de programação *C* será capaz de começar a usar essa biblioteca de imediato. A interface foi projetado para ser simples, e conectar-se em muitas linguagens de alto nível. Além de fácil de compilar, essa biblioteca não tem nenhuma dependência com outros pacotes. Finalmente pode-se afirmar que, a melhor plataforma para utilização dessa biblioteca é o sistema *GNU*, este permite que a biblioteca aproveite as vantagens e recursos adicionais em um compilador *C/GNU* e em uma biblioteca *C/GNU*. Todavia, a biblioteca é completamente portável e pode compilar sobre a maioria dos sistemas com um compilador C, como por exemplo no Windows.

Vimos então que o *GSL* é uma biblioteca ideal para programadores que utilizam as linguagens *C* e *C++*, onde esse é um software de código aberto, sob licença da *GNU* ; também foi destacado nesta seção como obter o manual do *GSL* e como entender alguns conceitos básicos sobre a importância dessa biblioteca.

A.5 O MathGL - Uma ferramenta opcional para a criação de gráficos

Vamos agora, entender um pouco sobre o ***MathGL***, essa é uma ferramenta de grande utilidade quando o objetivo é obter uma visualização rápida de algum gráfico, produzido por um determinado programa. Esta seção mostra o procedimento eficaz para instalação dessa biblioteca. Uma vantagem excepcional do *MathGL* é que sem a necessidade de um programa extra, essa ferramenta imprime gráficos diretamente como arquivo em um diretório especificado. É de suma importância entender que esta biblioteca é essencial para a compilação da maioria dos programas apresentados neste trabalho.

Sabemos que a visualização de gráficos é um dos principais aliados para o estudo do comportamento de alguns sistemas importantes nas áreas das ciências exatas, portanto, se faz necessário o uso de uma biblioteca de boa qualidade que seja adequada para essa função. A *MathGL* traz inúmeras qualidades, de acordo commathgl(2016) essa é uma biblioteca para fazer gráficos científicos de alta qualidade tanto no *Linux* como no *Windows*. É de grande utilidade na *plotagem* rápida e processamento de grande conjunto de dados, podendo trabalhar em modos de janela e de console. Também é de fácil incorporação em outros programas e disponibiliza uma grande variedade de modelos gráficos.

A MathGL tem disponível mais de 35000 linhas de código, mais de 55 tipos gerais

[7] Isso significa que todo mundo é livre para usá-lo, e para redistribuí-lo em outros programas igualmente livres.

de gráficos para 1d, 2d e 3d, incluindo os gráficos especiais para estatística e química. Pode exportar gráficos nos formatos *EPS* ou *SVG*, tem interfaces *Qt*, *FLTK*, *OpenGL* e pode ser usado até mesmo em programas de console. Além de ter funções para processamento de dados e linguagem de *script MGL*, também tem vários tipos de transparência e formas suavizada, fontes vetoriais e *TeX*, como também símbolo de análise curvilínea arbitrária, e sistema de coordenadas, além de inúmeras outras coisas úteis. Ela pode ser usado a partir do código escrito em *C++, C, Fortran, Python, Octave* e muitas outras linguagens, e o mais satisfatório é que esta plataforma é independente e livre, sob licença *GPL v.2.0*.

A utilização dessa biblioteca consiste primeiramente na instalação adequada da mesma. O arquivo em formato *.tar.gz* pode ser encontrado no site "<http://sourceforge.net/projects/mathgl/>". Depois do *download*, basta descompactar. Existem duas formas mais utilizadas para o processo de descompactação de arquivos no Ubuntu, a primeira implica em clicar com o botão direito do *mouse* e escolher a opção **Extrair aqui**, a segunda consiste em utilizar o terminal executando o seguinte comando, **> tar -xzvf mathgl-2.3.3.tar.gz**. Para que a instalação obtenha o sucesso desejado, primeiramente devemos observar que dentro do diretório descompactado encontramos um arquivo de nome *"INSTALL"* e nele as seguintes instruções:

> *cmake .*
> *cmake .*
> *make*
> *sudo make install*
> *sudo ldconfig*

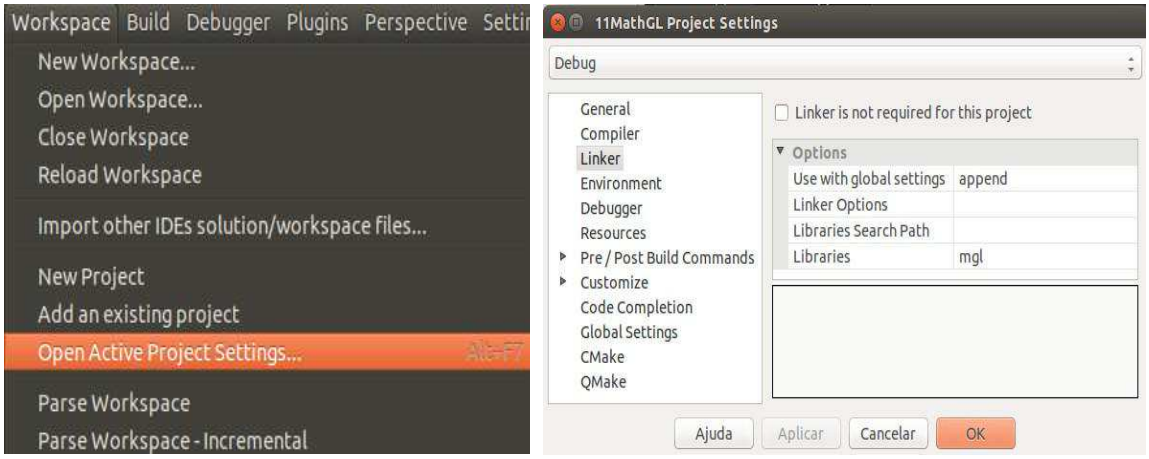
Esses comandos devem ser executados obedecendo respectivamente essa sequência. Se esse processo foi seguido com exatidão, o sistema estará preparado para a plotagem de gráficos de boa qualidade.

Após o desfecho positivo da instalação, o compilador indicado na seçãoA.4 será capaz de executar com sucesso um dos exemplos sugeridos no manual online do *MathGL*, que pode ser obtido em "<http://mathgl.sourceforge.net/doc_en/Examples.html#Examples>", é um código muito pequeno, constituído apenas de oito linhas, como pode ser visto a seguir.

```
1 #include<mgl2/mgl_cf.h>
2 intmain()
3 {
4   HMGL gr = mgl_create_graph(600,400);
5   mgl_fplot(gr,"sin(pi*x)","","");
6   mgl_write_frame(gr,"test.png","");
7   mgl_delete_graph(gr);
8 }
```

Mas para a execução desse exemplo, um pequeno ajuste deve ser feito, no Codelite. Dessa forma, na barra de ferramentas selecionamos *Workspace* e clicamos em *Open Active Project Settings*, ver Figura32a. O passo seguinte consiste em simplesmente inserir o

código "**mgl**"em *CommonSetting –> Linker –> Libraries* 32b.



(a) Open Active Project Settings... (b) ***Linker - Libraries***

Figura 32 – *Inserindo a Biblioteca "mgl"no CodeLite.*

Após a compilação desse programa um arquivo de formato *.png* é criado, ele pode ser localizado no diretório onde foi criado o Workspace [8], o arquivo *.png* obtido após a compilação do programa pode ser visualizado na Figura33

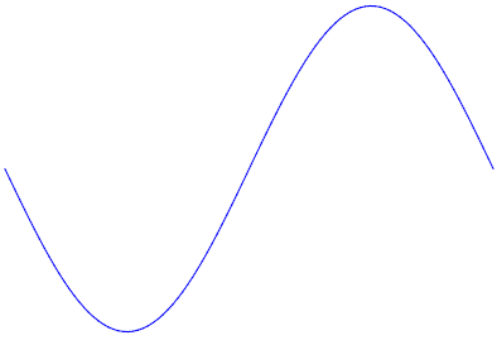


Figura 33 – *Exemplo do gráfico da função* $\sin(\pi x)$ *obtido através da biblioteca MathGL.*

Podemos observar que o gráfico obtido como resposta desse programa é uma oscilação, isso é óbvio visto que a função inserida no programa é uma função *seno*, logo a solução só poderia ser uma curva como essa que foi adquirida após a execução do código C++ citado.

Esta seção evidencia a compreensão conceitual do *MathGL*, como uma ferramenta opcional na visualização de gráficos. Este procedimento será de grande utilidade no

[8] Ambiente de trabalho que é definido no inicio da criação do programa.

desenvolvimento e compreensão dos métodos numéricos, citados no Capítulo2. Esta seção também esclarece como obter e instalar o *MathGL*, assim como executar um exemplo simples, demonstrando assim a eficácia dessa biblioteca.

A.6 Ajustes finais

Esta etapa consiste em fortalecer o sistema, melhorando a capacidade do *compilador*, também terá como objetivo, dar fundamentos para a instalação de algumas bibliotecas e introduzir posteriormente alguns conceitos sobre as mesmas.

Para a melhoria do sistema, alguns adicionais extras são necessários e fazem a diferença na criação de programas notáveis. Eles são fáceis de serem encontrados na *Central de programas Ubuntu*, ou podemos digitar "> *sudo apt-get install* ***nome do programa***"no terminal. A seguir serão feitas apresentações de cada um, e uma breve descrição dos mesmos.

O ***cmake-gui*** (**cmake**), de acordo com o desenvolvedor "<https://cmake.org/>", este programa é usado para controlar o processo de compilação de softwares, utiliza arquivos de configurações simples que são independentes da plataforma e do compilador. O *CMaker* gera Makefile's nativos, e workspaces que podem ser utilizados com vários compiladores, além de ser um software bastante sofisticado, pois suporta ambientes complexos que precisam da configuração do sistema, pré-processamento, geração de código e instanciação de templates.

O ***zlib1g-dev***, este programa, de acordo com "<http://zlib.net/>", além de ser livre e portátil entre plataformas, é uma biblioteca que implementa o método de compressão encontrado no *gzip* e no *PKZIP*, incluindo também arquivos de suporte e desenvolvimento.

O ***libpng12-dev***, "<http://libpng.org/pub/png/libpng.html>"afirma que o *libpng* é a biblioteca de referência oficial *PNG*[9], ela suporta quase todos os recursos *PNG*, é extensível, e foi amplamente testado por mais de 20 anos.

O ***libhpdf-dev***, como pode ser observado em "<http://libharu.org>", essa é uma biblioteca também com licença de código aberto, utilizado para geração de arquivos *.pdf*.

O ***libgsl0-dev***, de "<https://packages.debian.org/wheezy/libgsl0-dev>"vem a afirmação que esse e um pacote desenvolvimento *GNU*[10] sendo um conjunto de rotinas para a análise numérica.

O ***libgif-dev***, em resumo do que é afirmado em "<http://giflib.sourceforge.net/>", este é um pacote de ferramentas portáteis e rotinas de biblioteca para trabalhar com imagens *GIF*.

[9] Portable Network Graphics - Gráficos Portáveis de Rede.
[10] Scientific Library (GSL).

Por fim, o ***libgl1-mesa-dev*** é uma implementação *open-source* do *OpenGL*, conforme "<http://mesa3d.sourceforge.net/>", esse é um sistema de renderização de gráficos *3D* interativos, com uma variedade de controladores e dispositivo que permite a utilização em muitos ambientes diferentes, usufruindo da emulação do software, completa a aceleração do hardware para GPUs modernas.

Apesar de instalado todos os programas essenciais para o funcionamento das bibliotecas, ainda existe a necessidade da inclusão das mesmas no compilador, a não execução deste processo terá como consequência o não funcionamento de algumas tarefas na hora de compilar alguns programas que necessitem destas bibliotecas, causando assim constrangimentos. O procedimento adequado consiste simplesmente em abrir o compilador *(o codelite)*, e em "**Workspace > Open Active Project Settings > CommonSetting > Linker > Libraries**"incluir as seguintes bibliotecas "**gsl;m;gslcblas;mgl**". Um processo semelhante foi demonstrado na seção anterior[11] para uma única biblioteca, para mais de uma, o procedimento é análogo.

Concluída toda a orientação proposta nesta seção, o sistema operacional junto com o compilador sugerido, já deve ser capaz de pelo menos, teoricamente, realizar muitas tarefas importantes no âmbito da programação, e em particular, será apto a desenvolver todos os programas sugeridos no desenvolvimento deste trabalho.

A.7 Verificando a eficácia do sistema

A princípio o sistema está preparado para os objetivos que pressupõe o tema deste trabalho, mas ainda é necessário verificarmos a eficácia do mesmo, dessa forma vamos testa-lo. Isso irá prevenir problemas futuros.

O teste consiste em utilizar um exemplo já feito, que pode encontrado na apostila *"GSL, Exemplo da pag. 325,326 (EDO)"*, esta apostila pode ser baixada no seguinte endereço: "<https://www.gnu.org/software/gsl/manual/>". O código *C++* referente a esse exemplo pode ser visualizado no anexoA. Simplesmente pode-se transferir todo desse exemplo para um projeto já criado no *CodeLite* [12]. Porém existe a necessidade de uma alteração no código *C++* dado no anexoA, essa modificação consiste em alterar o código citado, para obtermos um arquivo com os dados dado pela compilação do programa. Dessa forma, basta inserirmos no programa as seguintes linhas,

```
1 main (void)
2 {//Essasduasprimeiraslinhasabreedefineoarquivo.
3     //Devemserinseridasaproximadamentenalinha33. FILE
4     * arq;
5     arq=fopen("teste.dat","wt");
6     ...    ...    ...    ...    ...
7     //Defineoqueseraimpressonoarquivo.
```

[11] A Figura32bmostra mais detalhes desse processo.

[12] Ver seçãoA.3para entender sobre criação de projetos.

```
8       //Deveserinseridoaproximadamentenalinha50.
9       fprintf (arq,"%.5e%.5e%.5e\n", t, y[0], y[1]);
10      }
11      //Fechaarquivo.
12      //Deveserinseridoaproximadamentenalinha53.
13      fclose (arq);
14      return0;
15  }
```

Observamos que, este fragmento de código, explicita com clareza como fazer a modificação correta, esclarecendo onde é necessário inserir os adicionais. Após compilação do projeto completo, basta verificarmos que dentro do diretório *Codelite*, existirá uma pasta com o nome *Debug* e no interior da mesma, haverá um arquivo com a extensão *.dat* [13].

De posse desses dados, já podemos criar um gráfico e verificar se o mesmo é semelhante ao obtido na apostila *GSL*. O próximo passo, implica na criação do gráfico. Para esta ação, será utilizado o *LibreOffice Calc*. Após abrir esse programa será necessário três passos simples: ***i)*** **Abrir...** e escolher o *.dat*, ***ii)*** Depois de aberto o arquivo, **Editar > Localizar e substituir...**, substituir os pontos por vírgulas, ***iii)*** Por último basta selecionar as colunas desejadas e ***clicar no ícone referente a criação de gráficos***. Se tudo ocorreu sem erro, o gráfico está pronto. Para uma melhor visualização é necessário fazer alguns ajustes, modificando escala, números..., de forma que este se assemelhe com o gráfico obtido no exemplo da apostila *(GSL)*.

Caso todas as etapas propostas nesta seção tenham sido concluídas com êxito, sem dúvida, o sistema terá uma considerada utilidade para criação dos programas propostos em todo trabalho.

A.8 Instalação e utilização do QtiPlot

Na seçãoA.5, foi visto uma forma de visualizar rapidamente um algum gráfico obtido através de um determinado programa, mas algumas vezes não precisamos apenas ter uma noção do gráfico que foi emitido, e sim poder entende-lo e analisa-lo com mais profundidade, dessa forma precisamos de um bom programa para isso. Então, vamos ver passo a passo como efetuar a instalação, assim como iniciar a utilização de um programa denominado ***QtiPlot***, que pode ser utilizado para análise de dados em multiplataforma e visualização de gráficos científicos.

Os passos apresentados aqui, são exclusivos para usuários do *Ubuntu*, acredita-se que não exista nenhuma particularidade extra para usuários de outros sistemas. Uma rápida pesquisa na *Central de programas do Ubuntu* evidencia sem nenhum excesso de esforço, a existência do citado programa. O *QtPlot* pode ser comparado ao famoso ***Originlab***,

[13] Esta extensão *.dat* pode ser modificada por outras na criação do arquivo, basta alterar no código C o *.dat* por *.txt* ou *.odt* ou outras.

sendo que esse considerado um dos melhores programas existente, para obtenção gráficos científico e análise dados.

Existem várias vantagens para os usuários que optem pela utilização do *Qtiplot*, algumas merecem destaque, e sem dúvidas as vantagens mais importantes que merecem ênfase são: Utilizar pouco espaço em disco, ser código aberto e poder ser utilizado por usuários do *Ubuntu*, além de ter a linguagem da interface em português, para usuários do Linux. Algumas informações extras podem ser vistas no site do desenvolvedor<http: //www.qtiplot.com/>.

Como já mencionado, a instalação desse programa não exige nenhuma particularidade extra, dessa forma os detalhes dessa instalação serão ocultadas aqui. Após instalado, a Figura34mostra a aparência inicial do programa.

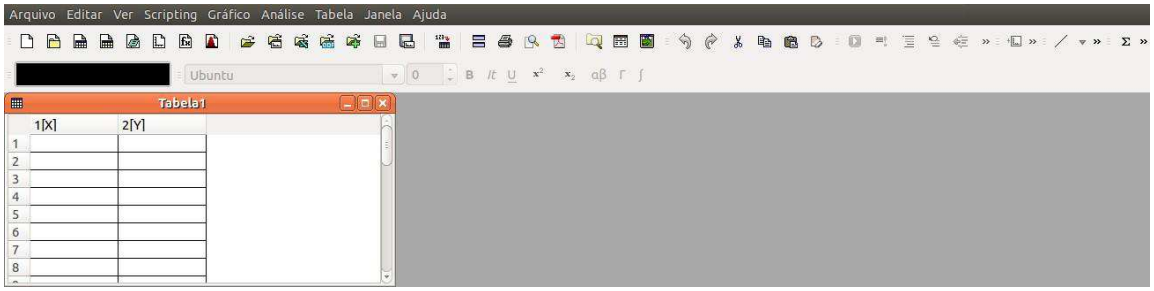


Figura 34 – *Tela inicial do "QtiPlot".*

De inicio é notável a semelhança desse aplicativo com *Originlab*, podemos também observar que a linguagem é apresentada em português e que existem muitas funções onde essas podem ser exploradas de várias formas por utilizadores com objetivos diversos.

A.8.1 Visualizando gráficos com o QtiPLot

Caso o objetivo seja visualisar o gráfico, essa subseção dará suporte para esse procedimento, a princípio existe a necessidade de dados para a plotagem de um gráfico [14], dessa forma vamos utilizar um arquivo *.dat* emitido por um programa que será apresentado no capítulo2. A função utilizada no desenvolvimento do programa citado, é referente a um oscilador harmônico simples [15]. O procedimento de importação dos dados de algum diretório para o QtiPlot é dado da seguinte forma:***Arquivo > Importar > Importar arquivo ASCII...***, ver na Figura35.

A partir de agora os procedimentos são apenas exemplos de uma situação particular, mais pode ser extendida a outras situações. Após encontrarmos o diretório onde se encontra o arquivo *.dat* a ser utilizado, e abrirmos o mesmo através do programa, vamos obter algo semelhante à Figura36.

[14] Podemos simplesmente inserir os dados manualmente ou importar um arquivo.
[15] Mais detalhes sobre o *.dat* no capítulo2.

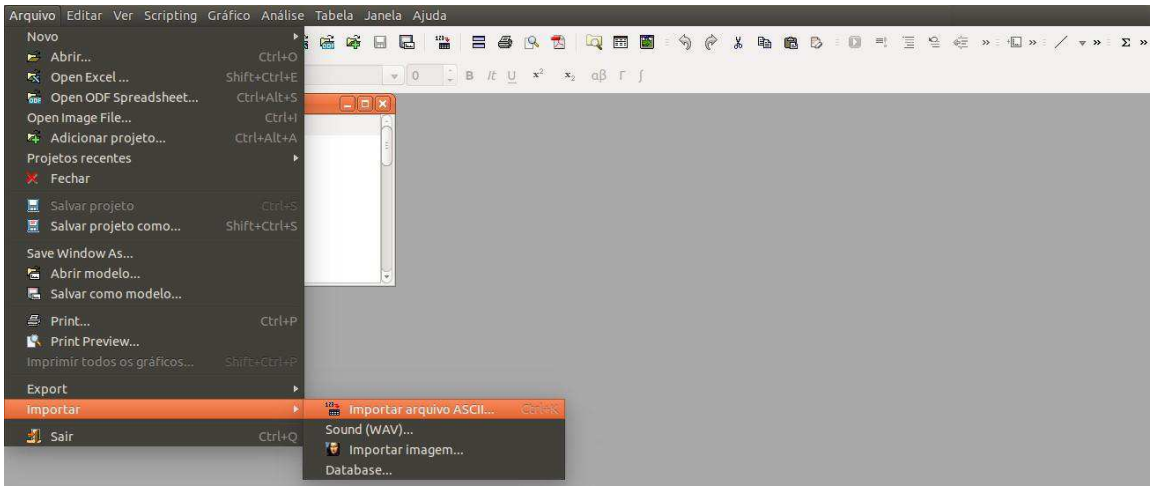


Figura 35 – *Procedimento para importar arquivos para o QtiPlot.*

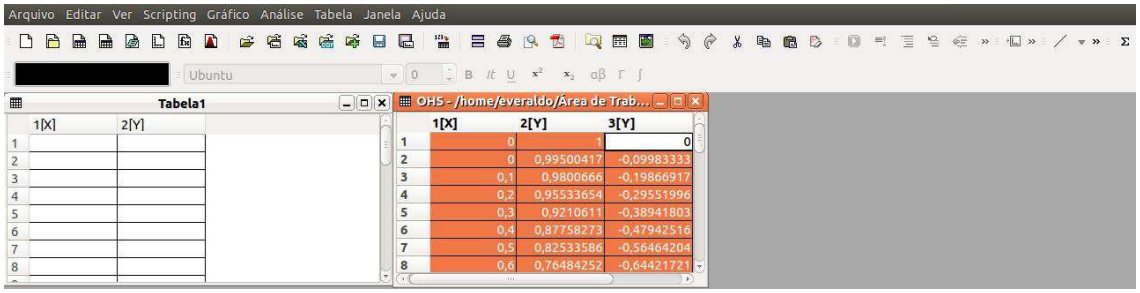


Figura 36 – *Selecionando colunas no QtiPlot.*

Notamos que existem várias colunas na tabela mostrada na Figura36, esse fato advém dos dados contidos no arquivo *.dat* importado para o programa, esse número de colunas pode variar. Um número maior de colunas corresponde a um número maior de gráficos, pois é equivalente a um número maior de funções. Se quisermos plotar um gráfico de uma função cada vez, basta selecionar-mos apenas duas colunas. Após escolhermos o número de colunas desejado, selecionamos a barra de ferramentas e no ícone gráfico temos:

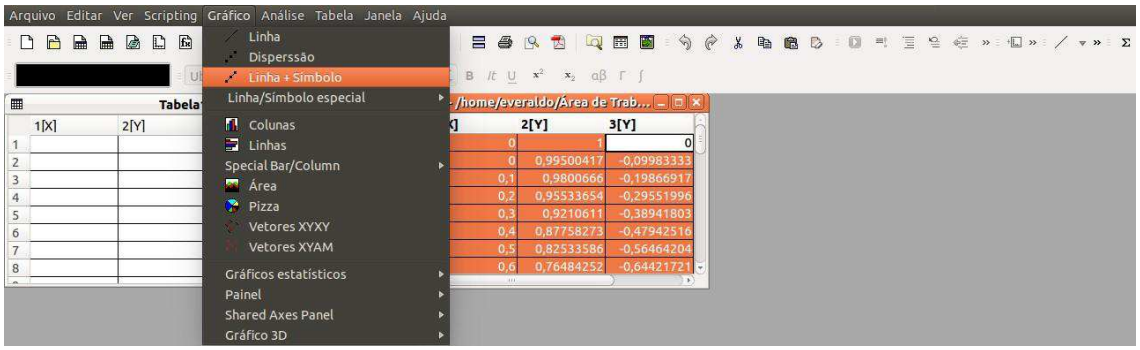


Figura 37 – *Escolhendo vários tipos de gráficos no QtiPlot.*

Podemos ver pela Figura37, que existem mais de uma opção para a "plotagem"de um gráfico, para esse caso particular será escolhido *Linha + Símbolo*. A resposta obtida pelo o programa pode ser visualizado na figura seguinte.

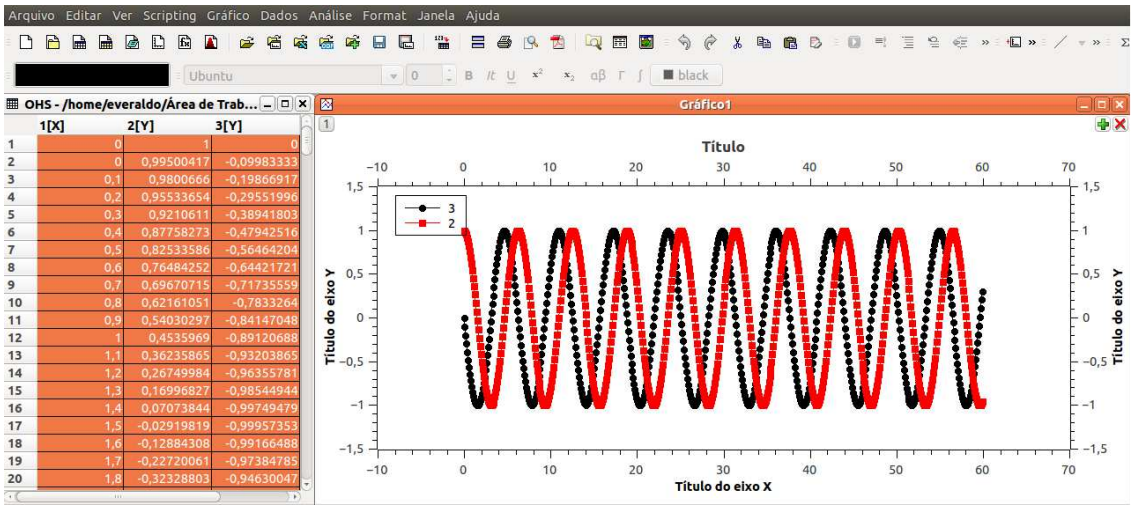


Figura 38 – *"Plotando" um gráfico.*

Observamos que a plotagem se refere a duas funções [16], caso queiramos apenas o gráfico de uma função basta selecionar duas colunas, para o gráfico de várias funções, é necessário selecionar várias colunas equivalentes.

A.8.2 Analisando gráficos com o QtiPLot

Muitas vezes não precisamos apenas visualizar o gráfico, e sim analisa-lo e compara-lo com alguma função, o entendimento do processo de comparação de resultados *numérico versus analítico* é de extrema importância e será uma ferramenta crucial na execução dos mecanismos utilizados no Capítulo2.

Após obtermos êxito no desenvolvimento dos processos citados na subseção A.8.1, podemos aproveitar os passos já reproduzidos e exposto na Figura38e apenas dar continuidade, onde agora selecionamos a barra de ferramentas e no menu de gráficos: *adicionar função...*. A Figura39, destaca os detalhes desse processo.

As funções analíticas correspondentes a esses gráficos são a posição $x(t)$ e a velocidade $v(t)$ para o caso do oscilador harmônico simples [17]. DeNussenzveig(1998) temos que $x(t) = A\cos(\omega t + \varphi)$ e $v(t) = -Aw\sin(\omega t + \varphi)$. Na janela aberta no *QtiPlot* basta inserir as funções, uma de cada vez, dessa forma vamos obter uma linha no gráfico que corresponde a solução analítica da função, Figura40.

Observamos a partir da Figura40, que curvas foram sobreposta as outras que tinham sido obtidas através do arquivo *.dat* ou seja, da solução numérica. De posse dessas informações podemos concluir que os resultados são semelhantes, pois as curvas se sobrepõe quase que de modo a se confundir. Os parâmetros escolhidos devem ser obrigatoriamente iguais aos definido na compilação do programa, para esse gráfico os

[16] Vimos no Capítulo2, que uma curva se refere a x(t) e a outra a v(t).
[17] ver Capítulo2.

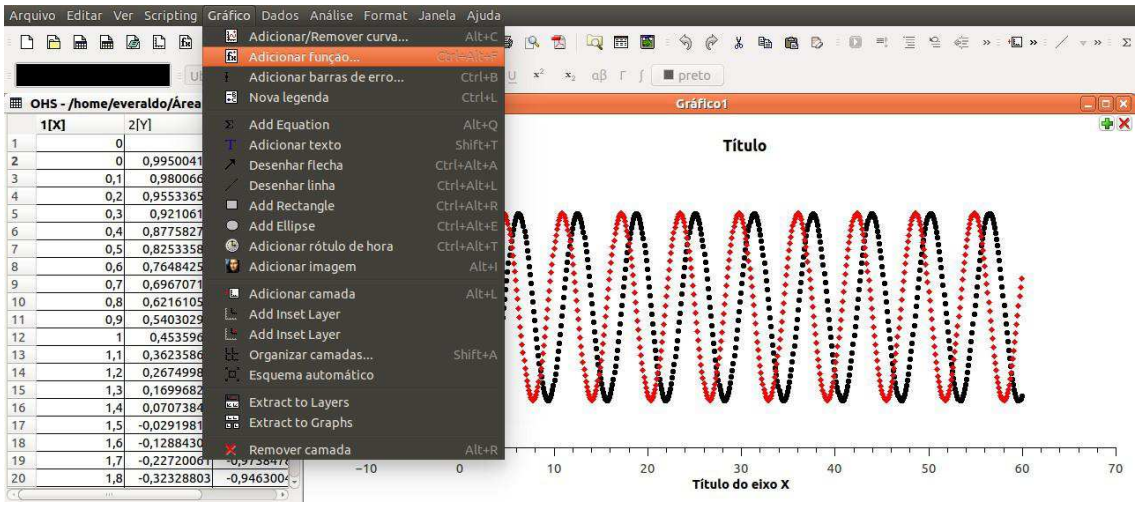


Figura 39 – *Método para adicionar uma função no QtiPlot.*

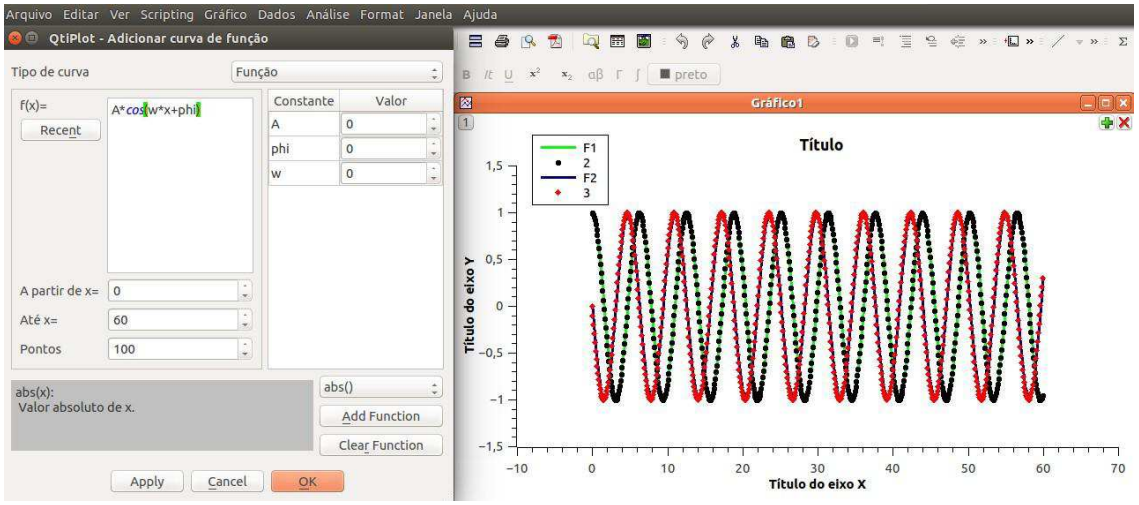


Figura 40 – *Comparando os resultados no QtiPLot.*

parâmetros foram dados como *A*, *phi* e *w*, com valores iguais a 1, 0 e 1, respectivamente, esses dados estão de acordo com os que seram apresentados no capítulo2. As funções utilizadas foram $f(x) = A * cos(w * x + phi)$ que corresponde a ***F1*** *(curva de cor verde)* e $f(x) = -A * w * sin(w * x + phi)$ que corresponde ***F2*** *(curva de cor azul)*. Os gráficos esboçados pelo *QtiPlot* podem ser personalizados, uma dica é utilizar a barra barra de ferramentas e no ícone *format* pode ser notado que é disponibilizadas ferramentas suficiente para a personalização de gráficos, como opção, pode-se ainda, dar apenas dois clics na área que se deseja personalizar.

A.9 Conclusão

A finalização dos processos citados neste Capítulo, torna o sistema preparado para a conclusão dos objetivos relacionados a computação numérica, proposto por este trabalho. Como também, para outras finalidades semelhantes, visto que a combinação do sistema

aliado com o compilador e as bibliotecas instaladas trazem uma boa estabilidade ao sistema. Vale esclarecer a importância dos testes citados nas seções que antecedem essa conclusão, a execução desses, sem erros, demonstra a eficácia do conjunto instalado.

APÊNDICE B – Código C++ para a solução de um OHS

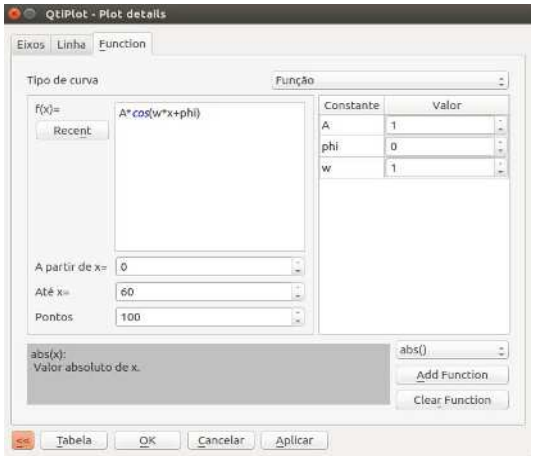
A seguir será mostrado o código *C++* utilizado na seção2.6. Para fins didáticos é de extrema importância que o programa esteja comentado. Para este caso, é observado que a acentuação gráfica nos comentários foram ocultados, o fato dessa omissão, decorre de um erro de compilação emitido pelo compilador do texto *.tex* utilizado para a digitação deste trabalho.

```
1  /*
2       Aquiserautilizandoosmetodosde RungeKuttadeQuartaordemcomintuito deobter
        asolucaonumericaparaooscilador harmonicosimples
3                       PARAESTEPROGRAMA
4       Aequacaoed^2x/dt^2+(k/m)x=0--->d^2x/dt^2=-(omega^2)x
5       ondekeaconstanteelasticaema,massa.Afrequenciaangularedada
6       poromega^2=k/m
7       Vamosdefiniromega^2=1
8       Vamosentao,quebraressaequacaodiferencialdesegundaordememduas
9           equacoes
        diferenciaisdeprimeira,umaparaaposicaooutraparaavelocidade,
10      dx/dt=v
11      dv/dt=-(omega^2)*x
12      Vamosrepresentarxevcomofuncaesdet,daseguinteforma
13      y[0]==x
14      y[1]==v
15      dy[0]/dt=v
16      dy[1]/dt=-(omega^2)x
17 */
18
19 #include<stdio.h>
20 //Aquiseramdefinidasalgumasvariaveis
21 #defineN 2//Numerodeequacoes
22 #definedt 0.1//Passodeintegracao
23 #defineMIN 0.0//tempo(x)minimo
24 #defineMAX 60.0//tempo(x)maximo
25 #defineomega2 1//T=2*pi=6.28s
26 //Esseblocoespecificaafuncaoaserresolvida
27 /*-------------------------------------------------------------------------
28     */
    doublef(doublex,doubley[],inti)
29 {
30      if(i==0)
31           return( y [1]);
32      elseif( i  == 1)
33           return(- omega2 * y [0]);
34      else{
35           printf("Numerodeeq.incorreto!\n");
36           return-999999.;
37           }
38 }
39 /*-------------------------------------------------------------------------
40      */
    //Funcaocorrespondenteaoalgoritmode Runge-Kutta
41 /*-------------------------------------------------------------------------
42      */ voidrunge4
    (doublex,doubley[],doublepasso )
43
```

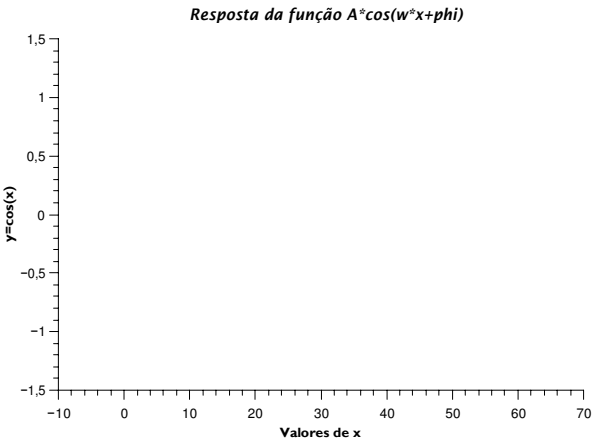
```
44 {
45     doubleh=passo/2.0, t1[N], t2[N], t3[N], k1[N], k2[N], k3[N],k4[N];
46     inti;
47     for( i=0; i< N ; i++){ k1 [ i]=
48         passo * f(x, y , i);
49         t1[i] = y[i]+0.5*k1[i];
50         }
51     for(i=0; i<N; i++){
52         k2[i]=passo*f(x+h, t1, i);
53         t2[i] = y[i]+0.5*k2[i];
54         }
55     for(i=0; i<N; i++) {
56         k3[i]=passo*f(x+h, t2, i);
57         t3[i] = y[i]+ k3[i];
58         }
59     for(i=0; i<N; i++) k4[i] = passo*f(x + passo, t3, i);
60     for(i=0; i<N ; i++) y[i] += (k1 [i]+2*k2 [i]+2*k3 [i]+k4 [i])/6.0;
61 }
62 /*--------------------------------------------------------------------------
       */
63
64 /*--------------------------------------------------------------------------
       */
65
66 intmain()
67 {
68     FILE * arq ; arq = fopen
69     ("OHS.dat","wt");//Criaumarquivodeextensao.datcomonome
           OHS
70
71     doublex, y[N];
72     y[0] = 1.0;//posicaoinicial
73     y[1] = 0.0;//velocidadeinicial
74     x = MIN;//tempoinicial
75
76     fprintf(arq,"%8.6f%10.8f%10.8f%10.8f\n", x, y[0], y[1], 0.5*(y[1]*y[1]+y
           [0]*y[0]));
77     for(x = MIN; x <= MAX ; x += dt){
78         runge4(x, y, dt);
79         fprintf(arq,"%8.6f%10.8f%10.8f%10.8f\n", x, y[0], y[1], 0.5*(y[1]*y
               [1]+y[0]*y[0]));
80     }
81    printf("Osdadosseraoarquivadosem OHS.dat\n");
82    fclose (arq);
83    return0;
84 }
85 /*--------------------------------------------------------------------------
       */
```

A Figura41a, exposta logo a seguir, mostra claramente o procedimento para inserção de uma função no CodeLite com o objetivo de obter o resultado analítica da expressão que representa o oscilador harmônico simples.

Agora iremos ver um exemplo de como inserir a função do oscilador harmônico simples no QtiPlot, para o cálculo analítico da velocidade em função do tempo.

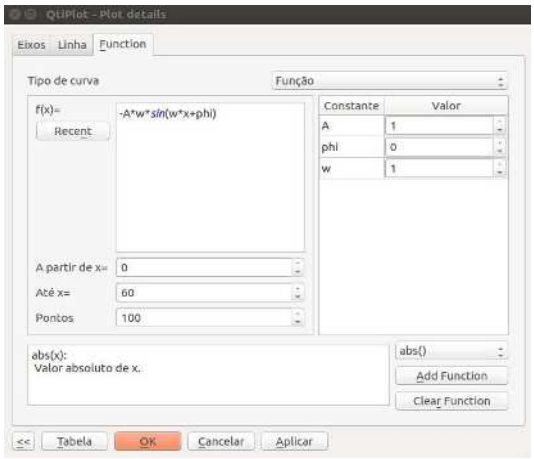


(a) Inserindo a função

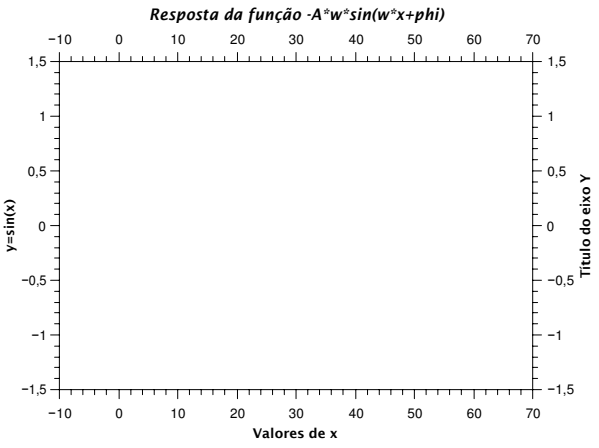


(b) Gráfico da função

Figura 41 – *Exemplo de como inserir a função do oscilador harmônico simples no QtiPlot, para fazer comparação dos resultados da posição em função do tempo.*



(a) Inserindo a função



(b) Gráfico da função

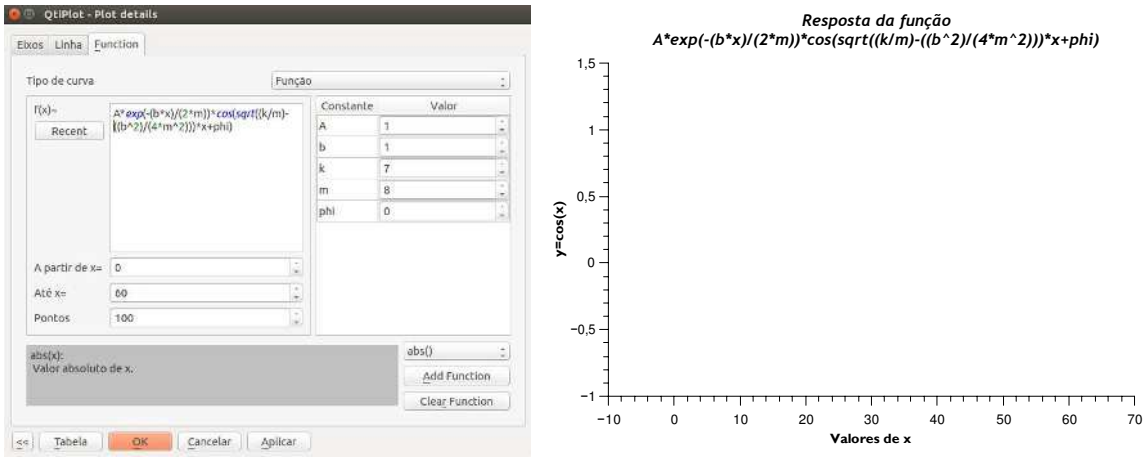
Figura 42 – *Exemplo de como inserir a função do oscilador harmônico simples no QtiPlot, para fazer comparação dos resultados da velocidade em função do tempo.*

APÊNDICE C – Código C++ para a solução de um OHA

```
/*
    Utilizandoosmetodosde RungeKuttade4comintuitodeobter
    asolucaonumericaparaooscilador harmonicoamortecido
                PARAESTEPROGRAMA
    Aequacaoed^2x/dt^2+(b/m)dx/dt+(k/m)x=0--->d^2x/dt^2=-(b/m)dx/dt-(k/m
        )x
    ondekeaconstanteelastica,beaconstantedeviscosidadeemeamassa.
         A
    frequenciaangularadadaporomega^2=k/m
    Vamosdefiniromega^2=1
    Vamosentao,quebraressaequacaodiferencialdesegundaordememduas
        equacoes
    diferenciaisdeprimeira,umaparaaposicaooutraparaavelocidade,
    dx/dt=v
    dv/dt=-(b/m)v-(k/m)x
    Vamosrepresentarxevcomofuncoesdet,daseguinteforma
    y[0]==x
    y[1]==v
    dy[0]/dt=v
    dy[1]/dt=-(b/m)y[1]-(k/m)y[0]
*/

#include<stdio.h>
//Aquiseramdefinidasalgumasvariaveis
#defineN 2//Numerodeequacoes
#definedt 0.1//passodeintegracao
#defineMIN 0.0//tempo(x)minimo
#defineMAX 20.0//tempo(x)maximo
#definek 7.//Constanteelasticadamola
#definem 8//massa
#defineb 1.//constantedeviscosidade

//Esseblocoespecificaafuncaoaserresolvida
/*-------------------------------------------------------------------------
    */
doublef(doublex,doubley[],inti)
{
    if( i == 0)
        return(y[1]);
        elseif( i == 1)
            return(-b*y[1]/m - k*y[0]/m);
        else{
            printf("Numerodeeq.incorreto!\n");
            return-999999.;
            }
}
/*-------------------------------------------------------------------------
    */
//Funcaocorrespondenteaoalgoritmode Runge-Kutta
/*-------------------------------------------------------------------------
    */
voidrunge4(doublex,doubley[],doublepasso)
{
    doubleh=passo/2.0, t1[N], t2[N], t3[N], k1[N], k2[N], k3[N],k4[N];
    inti;
    for(i=0; i<N; i++){
        k1[i]=passo*f(x, y, i);
        t1[i] = y[i]+0.5*k1[i];
```

```
53          }
54      for(  i=0;   i<  N  ;   i++){
55          k2[i]=passo*f(x+h, t1, i);
56          t2[i] = y[i]+0.5*k2[i];
57          }
58      for(i=0; i<N; i++){
59          k3[i]=passo*f(x+h, t2, i);
60          t3[i] = y[i]+ k3[i];
61          }
62      for(i=0; i<N; i++) k4[i] = passo*f(x + passo, t3, i);
63      for(i=0; i<N; i++) y[i] += (k1 [i]+2*k2 [i]+2*k3 [i]+k4 [i])/6.0;
64  }
65  /*-------------------------------------------------------------------------
        */
66
67  /*-------------------------------------------------------------------------
        */
68
69  intmain()
70  {
71      FILE * arq ; arq = fopen
72      ("OHA.dat","wt");//Criaumarquivodeextensao.datcomo
            nomeOHS
73
74      doublex, y[N];
75      y[0] = 1.0;//posicaoinicial
76      y[1] = 0.0;//velocidadeinicial
77      x = MIN;//tempoinicial
78
79
80     fprintf(arq,"%8.6f%10.8f%10.8f%10.8f\n", x, y[0], y[1], 0.5*(y[1]*y[1]+y
           [0]*y[0]));
81     for(x = MIN; x <= MAX ; x += dt){
82          runge4(x, y, dt);
83          fprintf(arq,"%8.6f%10.8f%10.8f%10.8f\n", x, y[0], y[1], 0.5*(y[1]*y
               [1]+y[0]*y[0]));
84          }
85          printf("Osdadosseraoarquivadosem0HA.dat\n");
86      fclose(arq );
87      return0;
88  }
89  /*-------------------------------------------------------------------------
        */
```

A Figura43a, mostra claramente como inserir a expressão oscilador harmônico amortecido no QtiPlot, para podermos comparar a curva do resultado analítico com a curva obtida através do resultado numérico.



(a) Inserindo a função ***(b) Gráfico da função***

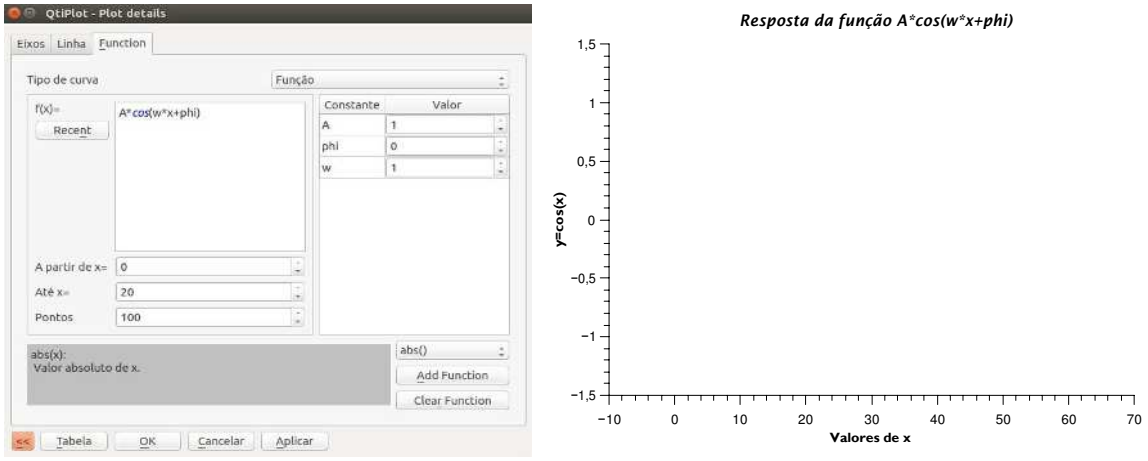
Figura 43 – *Exemplo de como inserir a função do oscilador harmônico amortecido no QtiPlot, para fazer comparação dos resultados da posição em função do tempo.*

APÊNDICE D – Código C++ para a solução de um OHS utilizando o *GSL*

```
/*Estecodigoeadaptadodeummodelodescritonaapostila"GSLExemplo
 dapag.325,326(EDO)"*/

#include<stdio.h>
#include<gsl/gsl_errno.h>
#include<gsl/gsl_matrix.h>
#include<gsl/gsl_odeiv2.h>

int
func (doublet,constdoubley[],doublef[],void*params)
{
    doublemu = *(double*)params;
    f[0] = y[1];
    f[1] = - mu*y[0];
    returnGSL_SUCCESS;
}

int
jac (doublet,constdoubley[],double*dfdy,doubledfdt[],void*params)
{
    doublemu = *(double*)params;
    gsl_matrix_view dfdy_mat = gsl_matrix_view_array (dfdy, 2, 2);
    gsl_matrix * m = &dfdy_mat.matrix;
    gsl_matrix_set (m, 0, 0, 0.0);
    gsl_matrix_set (m, 0, 1, 1.0);
    gsl_matrix_set (m, 1, 0, -mu);
    gsl_matrix_set (m, 1, 1, 0.0);
    dfdt[0] = 0.0;
    dfdt[1] = 0.0;
    returnGSL_SUCCESS;
}

int
main (void)
{
    FILE*arq;
    arq=fopen("teste.dat","wt");
    doublemu = 10;
    gsl_odeiv2_system sys = {func, jac, 2, &mu};
    gsl_odeiv2_driver * d = gsl_odeiv2_driver_alloc_y_new (&sys,
        gsl_odeiv2_step_rk8pd, 1e-6, 1e-6, 0.0);
    inti;
    doublet = 0.0, t1 = 20.0;
    doubley[2] = { 1.0, 0.0 };
        for(i = 1; i <= 100; i++)
            {
            doubleti = i * t1 / 100.0;
            intstatus = gsl_odeiv2_driver_apply (d, &t, ti, y);
                    if(status != GSL_SUCCESS)
                        {
                        printf ("error,returnvalue=%d\n", status);
                        break;
                        }
            fprintf (arq,"%.5e%.5e%.5e\n", t, y[0], y[1]);
            printf ("%.5e%.5e%.5e\n", t, y[0], y[1]);
            }
    fclose (arq);
    gsl_odeiv2_driver_free (d);
```

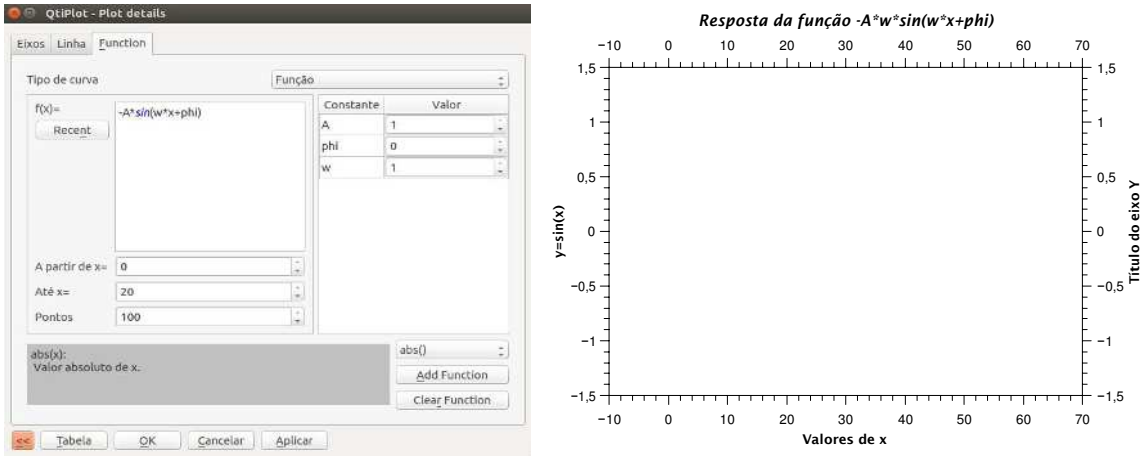
```
58      return0;
59  }
```

As figuras a seguir mostram como inserir a função e os valores para cada parâmetro da equação do oscilador harmônico simples no QtiPLot



(a) Inserindo a função **(b) Gráfico da função**

Figura 44 – *Exemplo de como inserir a função do oscilador harmônico simples no QtiPlot, para fazer comparação dos resultados da posição em função do tempo.*



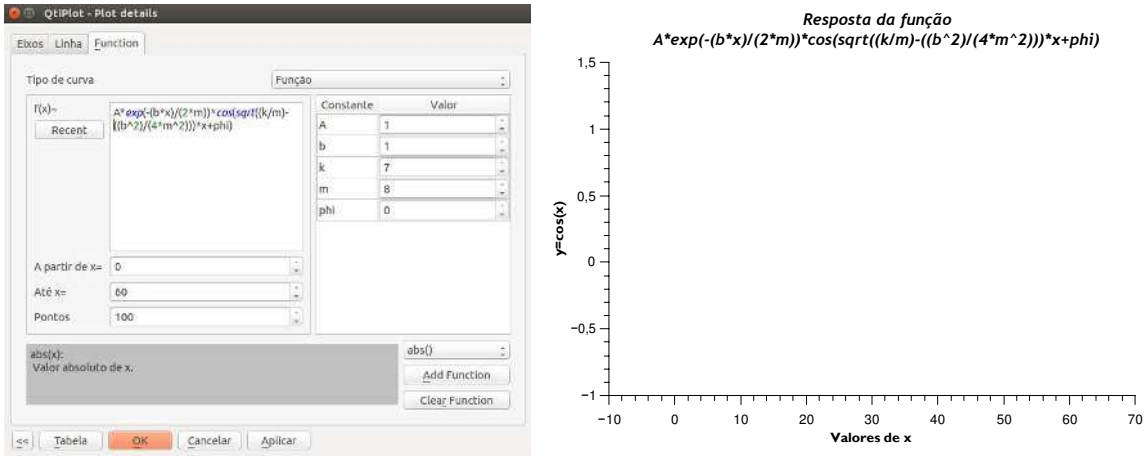
(a) Inserindo a função **(b) Gráfico da função**

Figura 45 – *Exemplo de como inserir a função do oscilador harmônico simples no QtiPlot, para fazer comparação dos resultados da velocidade em função do tempo.*

APÊNDICE E – Código C++ para a solução de um OHA utilizando o *GSL*

```
/*Estecodigoeadaptadodeummodelodescritonaapostila"GSLExemplo
 dapag.325,326(EDO)"*/
#include<stdio.h>
#include<cmath>
#include<gsl/gsl_errno.h>
#include<gsl/gsl_matrix.h>
#include<gsl/gsl_odeiv2.h>

intfunc (doublet,constdoubley[],doublef[],void*params)
{
    double*parametros = (double*)params;
    doublemassa  =parametros[0];
    doublek      =parametros[1];
    doubleb      =parametros[2];

    doubleomega2=k/massa;
    f[0] = y[1];
    f[1] = -b*y[1]/massa - k*y[0]/massa;
returnGSL_SUCCESS;
}

intjac (doublet,constdoubley[],double*dfdy,doubledfdt[],void*params)
{
    double*parametros = (double*)params;
    doublemassa=parametros[0]; doublek=
    parametros[1];     doubleb=
    parametros[2]; doubleomega2=k/
    massa;

    gsl_matrix_view dfdy_mat = gsl_matrix_view_array (dfdy, 2, 2);
    gsl_matrix * m = &dfdy_mat.matrix;
    gsl_matrix_set (m, 0, 0, 0.0);
    gsl_matrix_set (m, 0, 1, 1.0);
    gsl_matrix_set (m, 1, 0, -(k/massa));
    gsl_matrix_set (m, 1, 1, -(b/massa));
    dfdt[0] = 0.0;
    dfdt[1] = 0.0;
returnGSL_SUCCESS;
}

intmain (void)
{
    FILE * arq ; arq = fopen
    ("OHAA.dat","wt");
    doublemassa =8 , k=7 , b =1;
    doublemu [] = { massa ,k , b };
    gsl_odeiv2_system sys = {func, jac, 2, &mu};
    gsl_odeiv2_driver * d =
    gsl_odeiv2_driver_alloc_y_new (&sys, gsl_odeiv2_step_rk8pd,
    1e-6, 1e-6, 0.0);

    inti;
    doublet = 0.0, t1 = 20;
    doubley[2] = { 1, 0.0};
    fprintf (arq,"%.5e%.5e%.5e\n", t, y[0], y[1]);
    printf ("%.5e%.5e%.5e\n", t, y[0], y[1]);
    for(i = 1; i <= 100; i++){
        doubleti = i * t1 / 100.0;
```

```
59          intstatus = gsl_odeiv2_driver_apply (d, &t, ti, y);
60          if(status != GSL_SUCCESS){
61              printf ("error,returnvalue=%d\n", status);
62              break;
63          }
64          fprintf (arq,"%.5e%.5e%.5e\n", t, y[0], y[1]);
65          printf ("%.5e%.5e%.5e\n", t, y[0], y[1]);
66      }
67      fclose ( arq ); gsl_ odeiv 2 _
68      driver_ free  ( d);
69  return0;
70  }
```



(a) Inserindo a função ***(b) Gráfico da função***

Figura 46 – *Exemplo de como inserir a função do oscilador harmônico amortecido no QtiPlot, para fazer comparação dos resultados da posição em função do tempo.*

APÊNDICE F – Código C++ utilizado nas soluções do capítulo 5

```
#include<stdio.h>
#include<gsl/gsl_errno.h>
#include<gsl/gsl_matrix.h>
#include<gsl/gsl_odeiv2.h>

#defineG  32.87e+6//gamma-->TaxadedecaimentoemHz
#defineO  32.87e+6//omega-->FrequenciadeRabiemHz
#defineCP 1.054e-34//constantedeplank-->hcortado=h/2piem(m^2)*kg/s
#defineM  2.206e-25//massaatomicadoCesiumemkg
#defineK  7.37e+6//intensidadedovetordeondaemm
#defineD  G/2//16.44e+6//delta-->dissitoniaemHz//deltas
    =-2,-1,0,1,2gammas

doubleF(doublev)//Forcaparaofeixevindodaesquerda(xnegativo)
{
        return(-G*O*O*CP*K)/(4.0*(D-K*v)*(D-K*v)+G*G+2.0*O*O);
}

doubleA(doublev)//Doisfeixes
{
        return(F(v)-F(-v))/M;
}

int
func (doublet,constdoubley[],doublef[],void*params)
{
    doublemu = *(double*)params;
    f[O] = y[1];
    f[1]=A(y[1]);
    returnGSL_SUCCESS;
}

int
jac (doublet,constdoubley[],double*dfdy,doubledfdt[],void*params)
{
    doublemu = *(double*)params;
    gsl_matrix_view dfdy_mat = gsl_matrix_view_array (dfdy, 2, 2);
    gsl_matrix * m = &dfdy_mat.matrix;
    gsl_matrix_set (m, 0, 0, 0.0);
    gsl_matrix_set (m, 0, 1, 1.0);
    gsl_matrix_set (m, 1, 0, -2.0*mu*y[0]*y[1] - 1.0);
    gsl_matrix_set (m, 1, 1, -mu*(y[0]*y[0] - 1.0));
    dfdt[0] = 0.0;
    dfdt[1] = 0.0;
    returnGSL_SUCCESS;
}

int
main (void)
{
    intN=1000;
    FILE*arq;
    arq=fopen("SimulationZero.dat","wt");
    doublemu = 10;
    gsl_odeiv2_system sys = {func, jac, 2, &mu};
    gsl_odeiv2_driver * d = gsl_odeiv2_driver_alloc_y_new (&sys,
        gsl_odeiv2_step_rk8pd,
    1e-6, 1e-6, 0.0);
```

```
57      inti;
58      doublet = 0.0, t1 = 0.01;//0.10.010.001
59      doubley[2] = { 0.0, 17.0 };//50208.0
60          fprintf (arq,"%.5e%.5e%.5e%.5e\n", t, y[0], y[1], A(y[1]));
61          printf ("%.5e%.5e%.5e%.5e\n", t, y[0], y[1], A(y[1]));
62
63          for(i = 1; i <= N; i++)
64                  {
65                  doubleti = (double) i * t1 / ((double)N);
66                  intstatus = gsl_odeiv2_driver_apply (d, &t, ti, y);
67                                  if(status != GSL_SUCCESS)
68                                          {
69                                          printf ("error,returnvalue=%d\n",
                                                status);
70                                          break;
71                                          }
72                  fprintf (arq,"%.5e%.5e%.5e%.5e\n", t, y[0], y[1], A(y[1]));
73                  printf ("%.5e%.5e%.5e%.5e\n", t, y[0], y[1], A(y[1]));
74                  }
75      fclose ( arq ); gsl_ odeiv 2 _ driver_ free
76      ( d);  system ("qtiplotSimulation
77      Zero.dat");
78
79  //Parafazerograficodaaceleracaopelavelocidade
80          {
81          FILE *velocidade;
82          velocidade=fopen("aceleracao_velocidade_an.dat","wt");
83          for(doublev=-100;v<=100;v+=0.1)
84                  fprintf(velocidade,"%e\t%e\n",v,A(v));
85          fclose(velocidade);
86
87          //system("qtiplotaceleracao_velocidade_an.dat");
88          }
89
90      return0;
91  }
```

Anexos

ANEXO A – Exemplo da Solução Numérica de uma EDO

```
#include<stdio.h>
#include<gsl/gsl_errno.h>
#include<gsl/gsl_matrix.h>
#include<gsl/gsl_odeiv2.h>

int
func (doublet,constdoubley[],doublef[],void*params)
{
    doublemu = *(double*)params;
    f[0] = y[1];
    f[1] = -y[0] - mu*y[1]*(y[0]*y[0] - 1);
    returnGSL_SUCCESS;
}

int
jac (doublet,constdoubley[],double*dfdy,doubledfdt[],void*params)
{
    doublemu = *(double*)params;
    gsl_matrix_view dfdy_mat = gsl_matrix_view_array (dfdy, 2, 2);
    gsl_matrix * m = &dfdy_mat.matrix;
    gsl_matrix_set (m, 0, 0, 0.0);
    gsl_matrix_set (m, 0, 1, 1.0);
    gsl_matrix_set (m, 1, 0, -2.0*mu*y[0]*y[1] - 1.0);
    gsl_matrix_set (m, 1, 1, -mu*(y[0]*y[0] - 1.0));
    dfdt[0] = 0.0;
    dfdt[1] = 0.0;
    returnGSL_SUCCESS;
}

int
main (void)
{
    doublemu = 10;
    gsl_odeiv2_system sys = {func, jac, 2, &mu};
    gsl_odeiv2_driver * d = gsl_odeiv2_driver_alloc_y_new (&sys,
        gsl_odeiv2_step_rk8pd, 1e-6, 1e-6, 0.0);
    inti;
    doublet = 0.0, t1 = 100.0;
    doubley[2] = { 1.0, 0.0 };
        for(i = 1; i <= 100; i++)
            {
            doubleti = i * t1 / 100.0;
            intstatus = gsl_odeiv2_driver_apply (d, &t, ti, y);
                    if(status != GSL_SUCCESS)
                        {
                        printf ("error,returnvalue=%d\n", status);
                        break;
                        }
            printf ("%.5e%.5e%.5e\n", t, y[0], y[1]);
            }
    gsl_odeiv2_driver_free (d);
  return0;
}
```