Universidade Federal do Espírito Santo
Programa Institucional de Iniciação Científica
Jornada de Iniciação Científica 2007/2008
Ciências da Exatas e da Terra

Análise e Simulações em Mecânica Analítica usando Linguagem Funcional

Identificação:

Grande área do CNPq.:Ciências Exatas e da Terra
Área do CNPq: Física Clássica e Física Quântica; Mecânica e Campos
Título do Projeto: Análise e Simulações Computacionais da Mecânica Clássica
Professor Orientador: Ayrton Monteiro Cristo Filho
Estudante PIBIC/PIVIC: Hiezer de Souza da Silva

Resumo: A Mecânica Analítica, através de sua formalização matemática, permite o estudo e a análise de sistemas físicos clássicos. A descrição de sistemas físicos em Mecânica Analítica é baseada no estado energia do sistema, uma abordagem diferente da Mecânica Newtoniana, onde todas as forças devem ser explicitadas. Esta abordagem mais genérica, através das formulações Lagrangiana e Hamiltoniana, permite o estudo de sistemas complexos sem necessariamente conhecer todas as forças de vínculos, mas conhecendo restrições cinemáticas ao movimento. A representação dos métodos da Mecânica Analítica em ambiente computacional permite que estes possam ser formalizados, testados, visualizados e, em alguns casos, generalizados, de modo que o ferramental resultante possa ser aplicado a outros sistemas (Sussman e Wisdom, 2001). Linguagens de Programação Funcionais permitem uma abstração da máquina, dando ênfase à descrição e resolução do problema; toda a computação é feita em termos de definição de funções e são adequadas a problemas de Processamento Simbólico (Abelson e Sussman,1996). Este subprojeto visa fazer uso das potencialidades da programação funcional, através da linguagem de programação Scheme, para representar os métodos da Mecânica Analítica e estruturas físicas do sistema, com o objetivo de simular e, quando possível, resolver numérica e simbolicamente diversos sistemas físicos dássicos.

Palavras chave: Mecânica Clássica, Simulações Computacionais, Análise Simbólica, Programação Funcional.

1 – Introdução

A Mecânica Clássica estuda o movimento dos corpos materiais. A descrição Newtoniana da Mecânica Clássica se concentra na determinação das forças envolvidas no movimento do corpo (ou sistema de partículas) e das equações de movimento do corpo. Uma vez conhecidas todas as forças, é aplicada a equação diferencial (1.0) e as chamadas "Leis de Movimento de Newton" para encontrar a equação do movimento do sistema que deve satisfazer a equação diferencial (1.0) durante todo o movimento.

$$\sum_{i=1,\ldots,n} \vec{F}_i = \frac{d\vec{P}}{dt} \quad (1.0),$$

Onde F é *força* e P é *momento linear*:

$$\vec{p} = m\vec{v} = m\frac{d\vec{r}}{dt} \quad (1.1)$$

Para sistemas simples onde todas as forças são conhecidas *a priori* essa abordagem é aplicável, no entanto, nem sempre é possível determinar todas as forças. Existem, por exemplo, casos de forças de vínculos que são difíceis de serem determinadas *a priori*, podendo mesmo não ser possível expressá-las

explicitamente. Nestas situações a abordagem newtoniana se torna inadequada. Outro ponto importante de se notar é que a equação (1.0) pressupõe um sistema cartesiano de coordenadas para representar as grandezas vetoriais, o que não é adequado para diversos problemas (por exemplo, para um pêndulo simples somente o ângulo entre o pêndulo e a vertical é suficiente para descrever o seu movimento). Em suma, na Mecânica Newtoniana, o conceito de *"Força"*, que é uma grandeza vetorial, é fundamental na descrição do movimento.

A Mecânica Analítica surgiu como uma reformulação da Mecânica Clássica, em que os princípios físicos fundamentais da Mecânica foram expressos formalmente em uma matemática refinada. Se por um lado a Mecânica de Newton se baseia fundamentalmente em aspectos geométricos e vetoriais, tal como a força, na mecânica analítica o movimento do sistema pode ser expresso por grandezas escalares (ângulos, energia, etc). No formalismo Lagrangiano é definida uma grandeza que para muitos sistemas pode ser definida como a diferença entre energia cinética e potencial, chamada de *Lagrangiano,* dependente somente do tempo, da configuração e da taxa de variação da configuração do sistema. O princípio de *"Ação Mínima"* (conhecido também como princípio de *Hamilton)* diz que conhecido o Lagrangiano do sistema, pode-se distinguir a trajetória que é solução do sistema das demais através da integral do Lagrangiano para esta trajetória (integral da Ação ou simplesmente Ação), integral esta que deve ser estacionária para trajetória real do sistema. Resultados de Euler e Lagrange mostram que a trajetória que satisfaz este princípio é solução de uma certa equação diferencial, conhecida por equação de *Euler-Lagrange*(1.2) (Sussman e Wisdom, 2001).

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = 0 \quad (1.2)$$

Na equação (1.2), L é o Lagrangiano do sistema e as derivadas parciais são calculadas em relação às coordenadas genéricas do sistema (e suas derivadas temporais). Coordenadas Genéricas são parâmetros (*n* variáveis) com os quais é possível descrever univocamente a configuração do sistema a cada instante, obedecendo às restrições de natureza cinemática impostas ao movimento (para um sistema com *n* coordenadas genéricas, obtém-se um sistema de *n* equações de Euler-Lagrange) (Lemos, 2004).

Essa formulação traz uma série de conveniências e generalizações sobre a anterior (Newtoniana): não é mais necessário conhecer todas as forças que atuam sobre o sistema, pois seu estado é totalmente descrito por uma função escalar, que pode ser encontrada em termos da energia, e a equação de Euler-Lagrange pode ser utilizada para qualquer sistema de coordenada (ou em coordenadas genéricas). Além das vantagens acima, através desta formulação não é necessário conhecer inicialmente as forças de vínculo, pois podem-se expressar os vínculos através de funções que limitem algumas possíveis trajetórias ou posições do sistema. Estes vínculos podem ser acoplados às equações de Lagrange, de modo que a solução das equações satisfaça automaticamente às restrições impostas ao movimento.

As linguagens de programação funcional permitem ao programador uma abstração da máquina, permitindo assim o foco na descrição e resolução do problema. Em programação funcional os algoritmos e programas são expressos através de definições e composição de funções. Devido às características citadas, as linguagens de programação funcional são adequadas à programação e modelagem computacional de fenômenos físicos. Neste subprojeto o aluno estudou a linguagem Scheme, um dialeto da linguagem Lisp, com uma sintaxe simples, clara e com poucas regras (Abelson e Sussman, 1996). Scheme conta com uma biblioteca, conhecida por *Scmutils*, que implementa operadores genéricos da

linguagem sobre uma variedade de objetos matemáticos e contem ricas ferramentas de processamento simbólico e numérico (Sussman e Wisdom, 2001).

2 – Objetivos

O objetivo principal deste subprojeto é o estudo de diversos sistemas físicos clássicos, utilizando as ferramentas adquiridas no estudo da Mecânica Analítica, descrevendo-os na linguagem de programação funcional Scheme. Pretendeu-se com este projeto avaliar a adequação do uso de linguagens funcionais, especificamente o Scheme e da biblioteca *Scmutils*, para descrever problemas de Mecânica Clássica, além de incentivar uma notação matemática clara, sem ambigüidades e a formalização dos processos e métodos de resolução de sistemas físicos clássicos.

Ao longo da execução das atividades do projeto diversos métodos de modelagem e simulação computacional foram utilizados de modo a avaliar os melhores métodos. Através dos métodos utilizados, simulações e programas implementados, foram desenvolvidas ferramentas que poderão ser utilizadas na análise de outros sistemas físicos.

Para alcançar os objetivos supracitados foram definidas quatro etapas do projeto, cada uma delas possuindo alguns objetivos específicos: estudo de Programação Funcional, estudo das formulações Lagrangiana e Hamiltoniana, implementação das análise e simulações em Mecânica Analítica.

3 – Metodologia

Durante o primeiro estágio do projeto, Estudo de Programação Funcional, o aluno estudou a linguagem Scheme. Foi feito um estudo dirigido do livro *How to Design Programs* (Felleisen et al, 2001), no qual o aluno estudou as seções do livro, implementou alguns exemplo e resolveu os exercícios. O aluno apresentou uma série de seminários internos ao grupo de pesquisa e foram feitas reuniões semanais com o orientador, na medida em que avançava no estudo dirigido. Nos seminários foram expostos o conteúdo estudado, programas desenvolvidos, aplicações e exemplos.

O estudo da linguagem Scheme foi acompanhado da implementação de programas com relevante importância para o curso do projeto, de modo a avaliar as potencialidades do uso da linguagem nas etapas seguintes. Para citar alguns exemplos, foram desenvolvidos programas que implementam o cálculo simbólico e numérico de derivadas, métodos numéricos de quadratura (integração numérica), resolução de equações não-lineares (Newton-Raphson, Ponto Fixo) e álgebra de matrizes.

No final desta etapa do projeto o aluno participou do Ciclo de Seminários CEUNES 2007/2, onde foi apresentada a palestra intitulada *Programação Funcional – Scheme*. Na apresentação foram expostos os conceitos fundamentais de programação funcional e alguns programas desenvolvidos no âmbito do projeto, introduzindo para tal a linguagem Scheme.

Na segunda etapa, o estudo das Formulações Lagrangina e Hamiltoniana foi feito através de reuniões com o grupo de pesquisa e orientador. No grupo de pesquisa foram discutidos os temas relativos à Mecânica Analítica através de palestras e aulas sobre o assunto.

Na etapa de implementação, foi escolhido um sistema físico e aplicando os métodos da Mecânica Analítica foi feita a descrição do movimento. A modelagem computacional foi feita seguindo o seguinte esquema: como passo inicial, descreve-se o Lagrangiano do sistema em linguagem funcional, em seguida utiliza-se o Lagrangiano para encontrar as equações do movimento. Para tal procede-se de duas maneiras

distintas: pode-se gerar uma trajetória que minimiza a integral da ação através de métodos de minimização multidimensional e de interpolação de funções (disponíveis na *Scmutils*); ou então, aplicam-se as equações de Euler-Lagrange para gerar um sistema de equações diferenciais cuja solução é a trajetória do sistema físico (Sussman e Wisdom, 2001). A biblioteca *Scmutils* disponíbiliza funções e procedimentos para resolver sistemas de equações diferenciais. Sempre que possível foram feitas comparações entre os resultados obtidos e dados da literatura.

4 – Resultados e Discussão

No Estudo de Programação Funcional foram implementadas várias funções e programas, explorando as características da linguagem Scheme. Programas usuais em linguagens funcionais foram implementados, tais como ordenação, pesquisa, filtro e mapeamentos (*filter* e *map*, funções comuns em diversas linguagens funcionais, o *filter* seleciona os elementos de uma lista que satisfazem alguma propriedade definida por uma função, e o *map* aplica uma função sobre cada elemento da lista).

Para exemplificar os conceitos estudados é colocado abaixo o código de uma função que faz o cálculo simbólico de derivadas. Para tal são usadas listas como estrutura de dados, e símbolos como representação dos dados e operadores simbólicos, o primeiro elemento da lista é um símbolo para a operação seguido dos operandos, por exemplo, *sin(x)* seria representado por **`(list 'sin 'x)`** e *sen(x)+cos(x)* por **`(list '+ (list 'cos 'x) (list 'sin 'x))`**. Este programa supõe a existência de algumas funções adicionais: funções que determinam qual é a operação (**`soma?`**, **`mult?`**,...), que criam uma estrutura (lista) com operadores e operandos (**`make-soma`**, **`make-mult`**,...) e retornam os operandos de uma dado operação (**`soma-a`**, **`soma-b`**, **`pot-base`**,...). A idéia básica do algoritmo é transformar uma operação em outra aplicando as regras de diferenciação e, usando a regra da cadeia, recursivamente aplicar o algoritmo para cada operando da operação processada. Segue o código em linguagem Scheme do algoritmo.

```
(define (D f x)
  (cond
    ((number? f) 0)
    ((symbol? f) (if (eqv? f x) 1 0))
    ((soma? f) (make-soma (D (soma-a f) x) (D (soma-b f) x)))
    ((mult? f) (make-soma (make-mult (mult-a f) (D (mult-b f) x)) (make-mult (mult-b f)
(D (mult-a f) x))))
    ((seno? f) (make-mult (make-cosseno (seno-a f)) (D (seno-a f) x)))
    ((cosseno? f) (make-mult (make-mult (- 1) (make-seno (cosseno-a f))) (D (cosseno-a
f) x)))
    ((pot? f)
      (cond
        ((eqv? (pot-expoente f) 0) 1)
        ((eqv? (pot-expoente f) 1) (D (pot-base f) x))
        (else
          (make-mult (pot-expoente f) (make-mult (make-pot (pot-base f) (- (pot-expoente
f) 1)) (D (pot-base f) x)))))))))
```

Neste exemplo é importante observar a facilidade de descrição do problema matemático em linguagem funcional e como a recursão aparece naturalmente na descrição destes problemas. Para avaliar a expressão resultante para algum valor numérico deve-se antes definir uma função que avalie a expressão, substituindo cada ocorrência do símbolo que denota a variável pelo valor numérico.

Outras funções do Cálculo Numérico que seriam necessárias posteriormente no curso do projeto foram implementadas, tais como integração numérica, método de Newton-Raphson para estimar raízes de funções, etc.

Após a conclusão do estudo da linguagem Scheme seguiu-se o estudo do formalismo Lagrangiano e Hamiltoniano. O formalismo Lagrangiano foi introduzido nas reuniões do grupo de pesquisa seguindo o livro *Mecânica Analítica* (Lemos, 2004). Foram estudados *vínculos* como restrições cinemáticas ao movimento (definição geral e vínculos holônomos), deslocamentos virtuais e trabalho virtual, onde se definiu vínculo ideal como aquele cujo trabalho virtual é nulo; e em seguida introduziu-se o *Princípio dos Trabalhos Virtuais*, fazendo distinção entre trabalho virtual das forças aplicadas e das forças de vínculo (que para vínculos ideais é nulo).

Estendeu-se então o princípio de Trabalhos Virtuais para a Dinâmica, explorando assim o *Princípio de d'Alembert*. Definiu-se então as *coordenadas generalizadas* e *forças generalizadas* Q_i (aplicando o princípio de d'Alembert), foi encontrada uma equação semelhante à equação de Euler-Lagrange para as forças generalizadas e energia cinética T (Lemos, 2004).

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{q}_i} - \frac{\partial T}{\partial q_i} = Q_i \quad (4.0)$$

Para o caso que as forças derivam de um potencial escalar V, foi mostrado que vale a equação de Euler-Lagrange (1.2), com $L=T-V$.

Devido à extensão dos temas abordados verificou-se a impossibilidade de cumprir o cronograma proposto no projeto. Assim, foi omitido o estudo da formulação Hamiltoniana em detrimento do cumprimento das etapas seguintes.

O primeiro sistema físico que foi analisado foi o oscilador harmônico. O método utilizado consistiu em descrever o Lagrangiano em Scheme e aplicar o princípio de Hamilton para determinar a trajetória do sistema. Para tal é gerado uma lista de pontos lineares, entre as configurações iniciais e finais; a seguir é aplicado um algoritmo de minimização multidimensional (método de Nelder-Mead) disponível na biblioteca *Scmutils* para minimizar os pontos tendo como função objetiva a integral da ação (Sussman e Wisdom, 2001). Uma vez gerado o conjunto de pontos que minimizam a integral da ação, é computada uma função por interpolação polinomial. A importância deste método é que nenhum outro desenvolvimento matemático além da integral da ação é usado na modelagem computacional, consistindo em uma aplicação direta do *princípio de Ação Mínima*. A sua utilidade se encontra na didática, podendo ser usado para demonstrar a aplicação do próprio princípio de Hamilton para encontrar trajetórias do sistema. Não é um método recomendado em situações que sejam exigidos cálculos rápidos e acurados, pois o algoritmo é *extremamente* ineficiente.

A seguir foi escolhido o pêndulo simples não-linear para ser analisado e simulado. A abordagem adotada foi de descrever o Lagrangiano computacionalmente, aplicar as equações de Euler-Lagrange para gerar um sistema de equações e resolvê-lo para encontrar a equação da trajetória. Foi considerado para tal um pêndulo de massa m e comprimento l sob a ação da gravidade g. O sistema tem um grau de liberdade,

sendo que a coordenada generalizada escolhida por conveniência é o ângulo θ entre o pêndulo e direção vertical. Foram gerados gráficos do ângulo em função do tempo para alguns valores iniciais do ângulo θ, a fim de serem feitas comparações com resultados da literatura. O Lagrangiano do sistema pode ser encontrado pela diferença entre a energia cinética e potencial. A energia cinética é dada por um meio do produto da massa pelo quadrado velocidade linear *v* ($T=\frac{m}{2}v^2$). A velocidade escalar pode ser expressa em função do ângulo θ:

$$v = l\dot{\theta} \quad (4.1)$$

Usando (4.1) para calcular a energia cinética tem-se:

$$T = \frac{m}{2}(l\dot{\theta})^2 \quad (4.2)$$

Para a energia potencial obtém-se $V=mgy$. Considerando a projeção da posição do pêndulo sobre o eixo *y*, chega-se a $y=-l\cos(\theta)$. Assim a energia potencial pode ser calculada pela expressão:

$$V = -mgl\cos(\theta) \quad (4.3)$$

Substituindo *T* e *V* na expressão do Lagrangiano tem-se:

$$L = \frac{m}{2}(l\dot{\theta})^2 + mgl\cos(\theta) \quad (4.4)$$

Substituindo o Lagrangiano encontrado na equação de Euler-Lagrange e fazendo os devidos cálculos e simplificações obtém-se uma equação diferencial de segunda ordem não-linear:

$$\frac{d^2\theta}{dt^2} + \omega_0^2\sin(\theta) = 0 \quad (4.5)$$

$$\omega_0 = \sqrt{\frac{g}{l}} \quad (4.6)$$

Belendez *et al.* (2007) resolvem a equação diferencial apresentando uma solução analítica exata em termos da função elíptica de Jacobi *sn(u;m)* e da integral elíptica completa do primeiro tipo *K(m)* (4.8). Na resolução da equação as seguintes condições de contorno (4.7) foram utilizadas:

$$\theta(0) = \theta_0 \qquad \left(\frac{d\theta}{dt}\right)\Big|_{t=0} = 0 \quad (4.7)$$

Segue a solução analítica encontrada:

$$\theta(t) = 2\arcsin\left\{\sin\frac{\theta_0}{2}\,sn\left[K\left(\sin^2\frac{\theta_0}{2}\right) - \omega_0 t\,;\sin^2\frac{\theta_0}{2}\right]\right\} \quad (4.8)$$

A seguir estão as principais funções implementadas na simulação computacional do pêndulo simples.

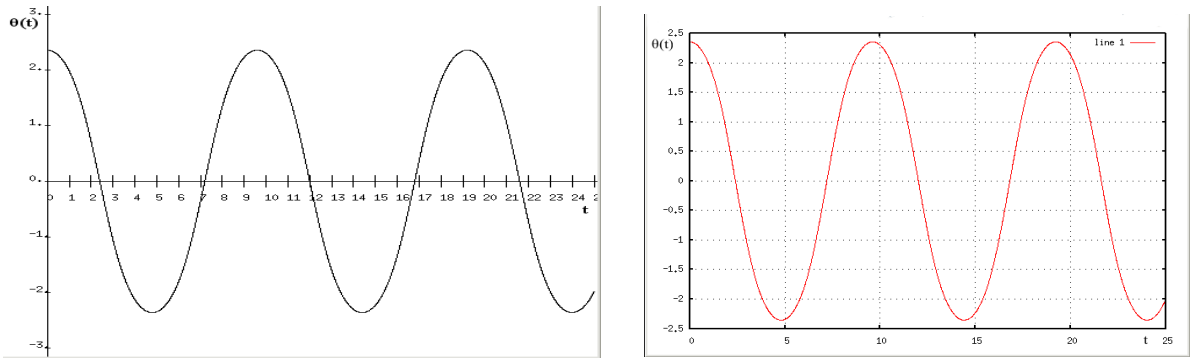
```
(define (L-pendulo m g l);;Lagrangiano do pêndulo
  (lambda (local)
    (let ((q (coordinate local))
          (v (velocity local)))
      (+ (* (/ m 2) (square l) (square v)) (* m g l (cos q))))))

(define (pend-sysder m g l);;retorna a derivada do estado do sistema
  (Lagrangian->state-derivative
   (L-pendulo m g l)))
```

```
((evolve pend-sysder ;;faz a evolução do sistema
        1.0 ;;m
        9.8 ;;g
        9.8 ;;l )
 (up 0.0 (* 0.99 :pi) 0);; to, θ0  e velocidade angular inicial
 (monitor-novo gr);;procedimento que monitora o estado do sistema. gera o gráfico
 0.01 ;; o passo de cada iteração
 25.0 ;; o t final.
 1.0e-13)
```

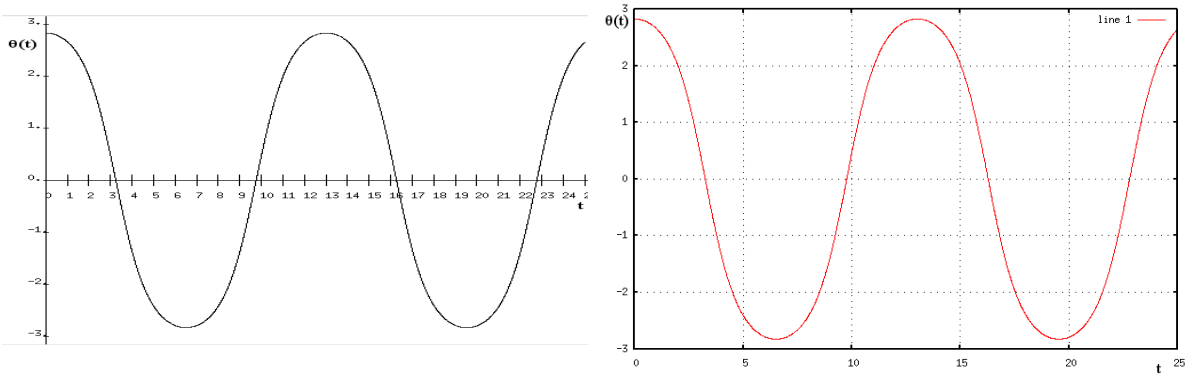
As duas principais funções utilizadas no *Scmutils* para integração de equações diferenciais são o `evolve` e o `state-advancer`. O `state-advancer` avança o estado do sistema a partir de um certo tempo e estado inicial para um tempo qualquer. O `evolve` avança o estado do sistema dentro de um certo intervalo de tempo enquanto monitora o estado do sistema, o que é feito através de chamadas seguidas do `state-advancer`. Estes procedimentos são utilizados na resolução de sistemas de equações diferenciais. Para gerar o conjunto de equações diferenciais é necessário aplicar o Lagrangiano às equações de Euler-Lagrange e fazer algumas manipulações na equação resultante para obter, tendo em mão as condições iniciais, um sistema de equações cuja solução é a trajetória do sistema físico. No *Scmutils* o procedimento `Lagrangian->state-derivative` recebe um Lagrangiano e retorna um procedimento que recebe uma *tupla* do estado (posição, velocidade, aceleração, etc) e retorna a derivada dos elementos desta tupla.

Para fins comparativos foram gerados gráficos do ângulo $\theta(t)$ em função do tempo($t=0$ *a* $t=25s$) para $\theta_0 = 0.75\pi$ $\theta_0 = 0.90\pi$ $\theta_0 = 0.99\pi$ $(g=9.8, l=9.8)$ usando a solução analítica (Belendez *et al.*, 2007) no GNU Octave e também com os resultados das simulações no *Scmutils*.



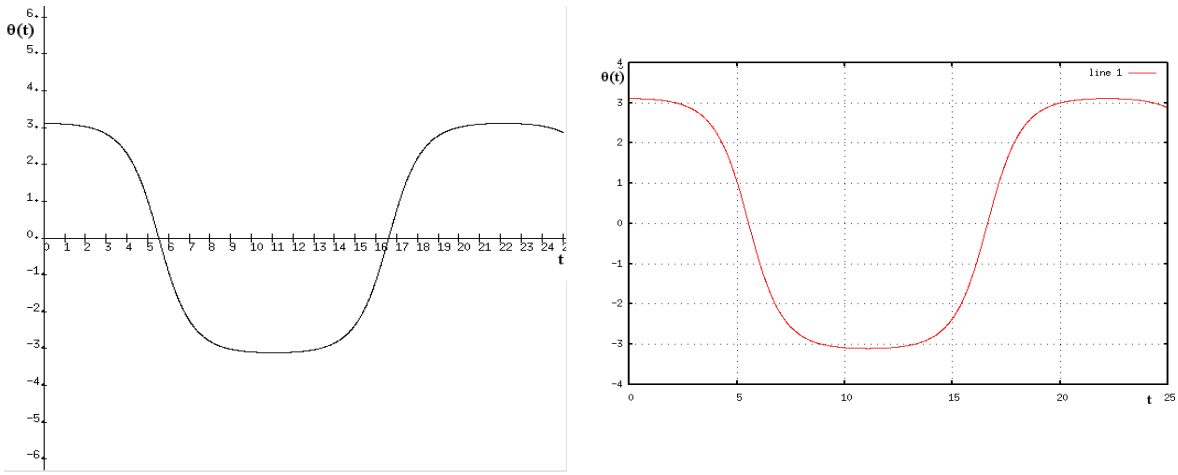
(a) (b)

Figura 1. Gráficos do ângulo $\theta(t)$ em função do tempo e $\theta_0 = 0.75\pi$ (a) gerado na simulação pelo *Scmutils* (b) plotado pelo GNU Octave usando a solução analítica exata.



(a) (b)

Figura 2. Gráficos do ângulo $\theta(t)$ em função do tempo e $\theta_0 = 0.9\pi$ (a) gerado na simulação pelo *Scmutils* (b) plotado pelo GNU Octave usando a solução analítica exata.



(a) (b)

Figura 3. Gráficos do ângulo $\theta(t)$ em função do tempo e $\theta_0 = 0.99\pi$ (a) gerado na simulação pelo *Scmutils* (b) plotado pelo GNU Octave usando a solução analítica exata.

Pela análise dos gráficos é possível verificar que a simulação acompanha a tendência da solução exata relativamente ao crescimento da amplitude da vibração em função do θ_0, o comportamento geral da simulação acompanha ao da solução analítica exata, os valores iniciais são próximos e os zeros das funções aparecem em pontos próximos. Em geral os resultados das simulações são coerentes com o previsto pela solução analítica.

A abordagem utilizada na resolução deste problema é útil na modelagem rápida de sistemas físicos, em situações onde obter uma resposta rápida do comportamento geral do sistema, sem fugir dos formalismos da Mecânica Analítica, é mais importante que se empenhar na resolução analítica do problema.

5 – Conclusões

A Mecânica Analítica através das formulações Lagrangiana e Hamiltoniana permite a modelagem de problemas complexos de Mecânica Clássica, sem a necessidade de expressar todas as forças e usando um formalismo matemática refinado do Cálculo Variacional.

Simulações computacionais são ferramentas importantes para formulação, visualização e teste de modelos propostos. Podem servir como meio de projeto e testes de modelos em situações onde se espera uma resposta rápida do comportamento geral do sistema antes de iniciar as formulações e resoluções com rigor matemático do modelo.

A linguagem de programação funcional Scheme, em conjunto com a biblioteca *Scmutils*, disponibiliza um conjunto de ferramentas poderosas para processamento simbólico e numérico, com funções implementadas que facilitam o processo de modelagem computacional de sistemas físicos. Este conjunto de ferramentas se mostrou adequado à descrição e solução de problemas da Mecânica Clássica, utilizando uma notação sem ambigüidades. A aferição deste resultado foi feita por meio de solução de problemas da Mecânica, além do desenvolvimento de ferramental matemático. Em especial, o problema do pêndulo simples não-linear foi modelado e a solução dada foi comparada com a solução analítica de Belendez *et al* (2007).

Aprofundar os estudos da Mecânica Analítica, através da formulação Hamiltoniana, e aplicar os métodos desenvolvidos neste projeto para problemas mais complexos, tais como estabilidade de órbita, corpos rígidos e osciladores são metas futuras interessantes para este projeto.

6 – Referências Bibliográficas

SUSSMAN, G. J.; WISDOM, J. **Structure and Interpretation of Classical Mechanics**. Cambridge: MIT Press , 2001.

ABELSON, H.; SUSSMAN, G. J. **Structure and Implementation of Computer Programs**. Cambridge: MIT Press, 1996.

MARION, J.B.; THORNTON, S.T. **Classical Dynamics of particles and Systems.** 4a Ed. Philadelphia: Saunders College Publishing, 1995.

FELLEISEN, M. et al. **How to Design Programs – An Introduction to Computing and Programming**. Cambridge: MIT Press, 2001.

LEMOS, N. A. **Mecânica Analítica**. São Paulo: Livraria da Física, 2004.

BELENDEZ, A. *et al.* Exact solution for the nonlinear pendulum. **Rev. Bras. Ens. Fis**, São Paulo, v. 29, n. 4, 2007. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0102-47442007000400024&lng=enDirectory&nrm=iso>. Acesso em: 22 de Julho de 2008. doi:10.1590/S0102-47442007000400024.