DIOGO TENÓRIO CINTRA

UMA ESTRATÉGIA DE ADAPTAÇÃO NO TEMPO BASEADA NA CURVATURA DO HISTÓRICO DE DESLOCAMENTOS

Maceió – AL

Outubro / 2008

DIOGO TENÓRIO CINTRA

UMA ESTRATÉGIA DE ADAPTAÇÃO NO TEMPO BASEADA NA CURVATURA DO HISTÓRICO DE DESLOCAMENTOS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Civil como parte dos requisitos para a obtenção do título de Mestre em Engenharia Civil

Orientador:
Prof. Dr. Eduardo Setton Sampaio da Silveira

Co-orientador:
Prof. Dr. Eduardo Nobre Lages

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA CIVIL
CENTRO DE TECNOLOGIA
UNIVERSIDADE FEDERAL DE ALAGOAS

Maceió – AL

Outubro / 2008

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária Responsável: Helena Cristina Pimentel do Vale

C575u Cintra, Diogo Tenório.
Uma estratégia de adaptação no tempo baseada na curvatura do histórico de deslocamento / Diogo Tenório Cintra. – Maceió, 2008.
xvi, 114 f. : il. grafs.

Orientador: Eduardo Setton Sampaio da Silveira.
Co- Orientador: Eduardo Nobre Lages.
Dissertação (mestrado em Engenharia Civil : Estruturas) – Universidade Federal de Alagoas. Centro de Tecnologia. Maceió, 2008.

Bibliografia: f. 109-112.
Apêndices: f. 113-114.

1. Engenharia – Processamento de dados. 2. Cálculos numéricos. 3. Programa computacional – Framework orientado a objetos. 4. Integração numérica. 5. Curvatura. I. Título.

CDU: 624:004.4’2

Universidade Federal de Alagoas – UFAL
Unidade Acadêmica Centro de Tecnologia – CTEC
Programa de Pós-Graduação de Engenharia Civil – PPGEC
Campus A. C. Simões, Av. Lourival de Melo Mota, S/N
Tabuleiro do Martins – CEP 57072-970 – Maceió – Alagoas
Tel/Fax: (82) 3214-1276
E-mail: ppgec@ctec.ufal.br
Homepage: http:/www.ctec.ufal.br/posgraduacao/ppgec

Membros da Banca de Exame da Dissertação de Mestrado do Engenheiro Civil **DIOGO TENÓRIO CINTRA,** intitulada "UMA ESTRATÉGIA DE ADAPTAÇÃO NO TEMPO BASEADA NA CURVATURA DO HISTÓRICO DE DESLOCAMENTOS", apresentada ao Programa de Pós-Graduação em Engenharia Civil, da Universidade Federal de Alagoas, no dia 03 do mês de outubro do ano de 2008, às 9 horas, na Sala de Aula do PPGEC/CTEC/UFAL.

MEMBROS DA BANCA:

Prof. Dr. Eduardo Setton Sampaio da Silveira
Orientador – CTEC/UFAL

Prof. Dr. Eduardo Nobre Lages
Co-orientador – CTEC/UFAL

Prof. Dr. José Luis Drummond Alves
COPPE/UFRJ

Prof. Dr. Adeildo Soares Ramos Júnior
CTEC/UFAL

Aos meus pais, pela atenção, apoio e amor incondicional.

Agradecimentos

A Deus, por tornar tudo possível.

Aos professores Eduardo Setton Sampaio da Silveira e Eduardo Nobre Lages, pelas orientações, conselhos e incentivos.

Ao professor Alejandro César Frery, pelas sugestões apontadas para este trabalho em seu exame de qualificação.

Aos professores José Luis Drummond Alves e Adeildo Soares Ramos Júnior, pela partipação na banca de defesa deste trabalho.

Aos amigos e colegas de turma, Alexandre, Arnaldo, Camila, Humberto, Márcio e Vitor. Por todos os bons e ótimos momentos vividos ao longo do curso.

Ao Fábio Ferreira e José Adeildo Amorim, pelo apoio nas implementações computacionais e parcerias em trabalhos.

Ao meu irmão Felipe, por toda força.

Aos demais amigos e familiares que contribuiram de forma direta ou indireta neste desafio.

Ao Programa de Pós-Graduação em Engenharia Civil - PPGEC/UFAL, docentes e funcionários, pela qualidade e empenho na formação acadêmica.

Ao Laboratório de Computação Científica e Visualização - LCCV/UFAL, pela infraestrutura de apoio e incentivo à pesquisa e desenvolvimento computacional.

À CAPES, pela concessão da bolsa de mestrado.

“O tempo e o espaço são modos
pelos quais pensamos e não
condições nas quais vivemos.”

Albert Einstein

Resumo

Este trabalho introduz uma estratégia de adaptação no tempo que utiliza um estimador de refinamento baseado no indicador geométrico curvatura. A metodologia desenvolvida é aplicada ao problema de integração numérica temporal, presente, por exemplo, no estudo do comportamento de corpos submetidos a cargas dinâmicas. O estimador de refinamento formulado demanda pouco esforço computacional, sendo facilmente aplicado aos diversos métodos de integração direta existentes. Visando a interação com esses métodos, constrói-se uma biblioteca orientada a objetos que incorpora a técnica de adaptação desenvolvida. A idéia principal desta ferramenta é prover, de forma fácil, a técnica de adaptação concebida a códigos computacionais existentes e que fazem uso dos referidos métodos de integração. Apresentam-se exemplos de dinâmica de corpos sólidos que ilustram o potencial de utilização da técnica e o uso da biblioteca em aplicações existentes.

Palavras-chave: Adaptação no tempo, métodos de integração direta, estimadores de refinamento, curvatura.

Abstract

This work introduces a time adaptive strategy that uses a refinement estimator based on the geometric indicator of curvature. The developed methodology is suitable for problems of numerical time integration present, for example, in the study of bodies subjected to dynamical loads. The refinement estimator demands low computational resources, being easily applied to several direct integration methods. Trying to interact these methods, an object oriented library that uses the developed scheme of adaptation is built. The main idea of this tool is to incorporate this scheme, in an easy way, in existing computational codes that employ direct integration methods. Examples of dynamic solid bodies are presented, ilustrating the technique and library usage in existing applications.

Keywords: Time adaptive integration, direct integration methods, refinement estimators, curvature.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

α_m, α_f, λ , Ω	Parâmetros do algoritmo do MEG-α
Ψ_1, Ψ_2, ϕ_1, ϕ_2	Parâmetros do algoritmo Chung-Lee
β, γ	Parâmetros do algoritmo de Newmark
$\boldsymbol{C}$	Matriz de amortecimento
$\dot{\boldsymbol{d}}$	Deslocamentos nos graus de liberdade
Δt	Incremento de tempo de integração
Δt_{crit}	Incremento de tempo de integração crítico
dk	Tamanho do intervalo de curvatura onde não há troca do incremento de tempo (Curva *k versus* Δt regularizada)
Δt_{new}	Incremento de tempo posterior à mudança (MEG-α adaptativo)
Δt_{old}	Incremento de tempo anterior à mudança (MEG-α adaptativo)
W_{nc}	Trabalho de Forças Não-Conservativas
e	Erro local
E	Módulo de elasticidade
η	Erro normalizado
η_{adm}	Erro admissível
$\boldsymbol{f}^{diss}$	Vetor de forças dissipativas
$\boldsymbol{f}^{ext}$	Vetor de forças externas
f_i	Funções de forças generalizadas
g	Gravidade
$\boldsymbol{\Gamma}(z)$	Vetor tangente à curva $\boldsymbol{\tau}$ no instante z
g_i	Coordenadas generalizadas
h_0	Altura de lançamento
$h(z)$	Função comprimento da curva $\boldsymbol{\tau}$ entre os instantes z_0 e z
$\boldsymbol{K}$	Matriz de rigidez
k_c	Rigidez de contato
$k(t)$	Curvatura do histórico de deslocamentos

$k(z)$	Curvatura da curva $\boldsymbol{\tau}$ no instante z
L	Comprimento
$\boldsymbol{M}$	Matriz de massa
m_p	Massa da partícula (esfera)
μ	Variável que define o limite inferior do intervalo onde o erro deve estar contido (MEG-α adaptativo)
$\|\boldsymbol{e}(t)\|$	Norma euclidiana do resíduo
$\omega, \overline{\boldsymbol{d}}$	Freqüência e modo de vibração associado ao problema de vibração livre
ω_{max}	Freqüência máxima do modelo discreto
Ψ	Parâmetro que define a dissipação numérica do algoritmo Chung-Lee
$r_f(t)$	Resíduo de forças desequilibradas
ρ_∞	Parâmetro do algoritmo do MEG-α
s_i	Deslocamentos modais
T	Período de vibração da estrutura
t	Tempo
$\boldsymbol{\tau}$	Curva parametrizada
θ	Variável que relaciona os incrementos de tempo para o MEG-α adaptativo
T_q	Tempo de queda
T	Energia Cinética
V	Energia Potencial
ξ	Fração do amortecimento crítico correspondente à máxima freqüência do sistema

1 Introdução

À medida que o potencial de transformação humana evolui e que novas tecnologias e materiais são desenvolvidos ou descobertos, surgem novos desafios relativos ao entendimento de uma ciência aplicável a estes novos fatos. Muitas vezes as formulações desenvolvidas com esta finalidade fazem uso de métodos numéricos, a exemplo dos Métodos dos Elementos Finitos, Discretos ou de Contorno. É comum que o emprego destas técnicas faça uso intenso de cálculos numéricos, motivando o uso de computadores. No estudo de modelos mais apurados, o esforço computacional envolvido é bastante elevado, o que se reflete no tempo necessário para a obtenção de uma análise. Um exemplo disso é o estudo de problemas de dinâmica estrutural.

Para a solução deste tipo de problema é comum a utilização de métodos de integração direta. Estes permitem o estudo de um fenômeno temporal a partir de uma discretização da solução em intervalos ou incrementos de tempo (Δt). A magnitude deste intervalo deve atender a critérios de convergência, tem forte impacto sobre o tempo de processamento e está relacionado com a precisão da resposta a ser obtida. Uma das práticas, quando na solução de problemas dinâmicos por este método, é a utilização de intervalos de tempo regulares, ou seja, com valor constante ao longo da análise. A adaptatividade é uma estratégia que permite o controle destes intervalos de tempo ao longo da simulação de forma a aperfeiçoar a relação existente entre qualidade de resposta e o tempo computacional demandado. Para tanto faz-se uso de entidades que de alguma forma quantificam ou estimam a distância existente entre a resposta numérica obtida pelo processo de integração e a resposta exata do problema, os estimadores de refinamento.

A estratégia de adaptação no tempo vem sendo utilizada há algum tempo como ferramenta de otimização de desempenho e qualidade em análises numéricas de problemas dinâmicos. Hibbit & Karlsson (1979), fazendo uso do Método de Newmark (1959), apresentaram um procedimento adaptativo aplicado a um algoritmo de integração implícito. A técnica baseava-se no balanço de forças (cálculo do resíduo) em um tempo intermediário entre as soluções obtidas no início e no final do intervalo. De acordo com esta informação fazia-se o refinamento do incremento de tempo de integração. No entanto, este cálculo representava na prática um alto custo computacional, o que acabava caracterizando um aspecto negativo da técnica. Uma melhoria deste algoritmo pôde ser observada a partir da proposição de cálculo expedito da força

residual para problemas lineares. Ourghouliam & Powel (1982) sugerem que se obtenha esta força residual a partir de um produto envolvendo a matriz de rigidez e o vetor de velocidade incremental, dados disponíveis em alguns tipos de análises dinâmicas.

Até aquele momento, a estratégia de adaptação no tempo foi usada no propósito de melhoria de qualidade de resposta. Por lidarem com algoritmos implícitos, a estratégia de adaptação apenas sugeria o refinamento (diminuição) do intervalo de tempo se o resíduo calculado fosse maior que um dado valor.

Os algoritmos de integração explícitos passaram a ser alvo de estudos a partir do trabalho de Park & Underwood (1980). Naquela ocasião o clássico Método das Diferenças Centrais foi utilizado em uma estratégia de adaptação que se baseava no cálculo das maiores freqüências aparentes do sistema. Esta grandeza era medida a partir dos vetores de deslocamentos e acelerações incrementais para todos os graus de liberdade e a partir do seu valor eram definidos quais os incrementos de tempo que deveriam ser adotados.

Estratégia semelhante também é utilizada por Zhang & Whiten (2001). Um estimador de refinamento é estabelecido a partir de informações incrementais de deslocamentos e velocidades. Com este estimador, obtido em cada passo de integração, faz-se o controle do tamanho do incremento Δt.

Também utilizando as maiores freqüências aparentes da estrutura, Bergan & Mollestad (1985) propuseram um esquema para o cálculo automático do incremento de tempo. Os autores introduziram a utilização de uma função de controle para evitar alterações constantes no incremento de tempo. A principal desvantagem do esquema era a necessidade da montagem da matriz de rigidez da estrutura.

Embora o conceito de freqüência aparente fosse efetivo na análise de problemas de vibração, o mesmo não podia ser aplicado em problemas onde este valor fosse próximo de zero. Com o intuito de analisar problemas desta natureza, Zienkiewicz & Xie (1991) desenvolvem um esquema adaptativo de integração implícita. Um estimador de refinamento simples e de baixo custo computacional, baseado em soluções locais aproximadas, é apresentado. O esquema permite que o incremento de tempo de integração seja adaptativamente ajustado para assegurar que o erro de discretização temporal esteja de acordo com uma precisão dada.

Utilizando também o conceito de solução local aproximada, Hulbert & Jang (1995) propuseram uma estratégia adaptativa desenvolvida especificamente para o Método Implícito Generalizado-α (MIG-α). Este método por sua vez deu origem ao Método Explícito Generalizado-α Adaptativo, que foi utilizado como base no desenvolvimento do trabalho de

Silveira (2001). Isso se fez necessário uma vez que no contexto desse último trabalho os algoritmos de integração implementados não faziam uso da matriz de rigidez da estrutura. Desta forma, a montagem desta matriz não seria justificada apenas com o objetivo da implementação do algoritmo de adaptação no tempo. O Método Explícito Generalizado-α Adaptativo, discutido em maiores detalhes na Seção 2.3.3, é posteriormente apresentado por Cintra & Silveira (2007). Na ocasião, os autores ajustam o mecanismo de adaptação de maneira a considerar múltiplos graus de liberdade e incorporam o método a um sistema computacional responsável pela análise dinâmica de linhas de ancoragem e *risers* (Ferreira, 2005).

Mecanismos de processamento de sinais e de filtros digitais passam a ser aplicáveis ao problema de integração numérica de sistemas dinâmicos. A técnica introduzida por Söderlind (2003) foi concebida com o intuito de produzir seqüências regularizadas de incremento de tempo. O uso destas seqüências tem impacto significativo na estabilidade computacional sem aumentar o custo extra de implementação.

Uma técnica de adaptação aplicada ao Método Explícito Generalizado-α é ainda apresentada por Gobat & Grosenbaugh (2006). Em cada instante de integração o incremento de tempo (Δt) é avaliado e, caso este não seja preciso o suficiente para propagar a solução, seu valor é reduzido por um fator de dez. O processo de redução em Δt pode ser recursivo, com limite prático de até quatro ordens de grandeza para o valor base de Δt.

1.1 Objetivos

Este trabalho tem como objetivo introduzir uma técnica de adaptação onde um estimador de refinamento, baseado no indicador geométrico curvatura, é utilizado. A partir da formulação apresentada para este estimador, alguns aspectos envolvidos na sua utilização em uma metodologia adaptativa são apresentados.

De posse da formulação desenvolvida, propõe-se a criação de uma biblioteca orientada a objetos que incorpore os aspectos relacionados com a técnica adaptativa e que permita o uso deste mecanismo em aplicações que façam uso de soluções iterativas compatíveis. Além disso, casos de uso desta biblioteca são apresentados com o objetivo de validar a implementação computacional realizada.

Alguns exemplos de modelos numéricos analisados fazendo uso desta técnica de adaptação são portanto apresentados com o intuito de avaliar a técnica desenvolvida e motivar trabalhos futuros.

1.2 Organização do Trabalho

Esta dissertação está organizada de maneira a tornar possível ao leitor uma visão geral acerca do procedimento de adaptação no tempo no contexto de integração numérica aplicável à dinâmica estrutural. Uma vez atingido este objetivo, uma nova metodologia de adaptatividade é proposta e exemplos de aplicação são apresentados e discutidos.

No Capítulo 2 são introduzidas noções básicas de dinâmica estrutural que partem do estudo da equação de movimento. Alguns métodos de solução do histórico de deslocamentos são abordados, sendo que o estudo dos métodos de integração direta, mais especificamente os do tipo explícito, recebe maior enfoque. Em seguida são feitas considerações relacionadas com o mecanismo de adaptatividade e de alguns estimadores de refinamento correlacionados. Uma técnica de adaptação existente é ainda apresentada com o objetivo de que comparações possam ser posteriormente realizadas.

O Capítulo 3 aborda a metodologia de adaptação temporal proposta por este trabalho. A formulação envolvida na definição do estimador de refinamento introduzido é apresentada e faz-se uma particularização deste estimador, direcionada ao estudo do histórico de deslocamentos. São abordados ainda aspectos relacionados à utilização da curvatura em um mecanismo de controle do incremento de tempo bem como detalhes referentes à implementação da biblioteca orientada a objetos responsável por esta tarefa.

No Capítulo 4, o projeto de uma biblioteca orientada a objetos que concebe a metodologia desenvolvida anteriormente é alvo de estudo. São apresentadas algumas definições referentes ao paradigma de programação orientada a objetos, bem como algumas representações gráficas para a organização de classes da biblioteca e dos seus principais algoritmos. Neste capítulo ainda são tratadas questões referentes à maneira pela qual a biblioteca se comunica com uma aplicação que faz uso de suas funcionalidades.

Exemplos de aplicações da técnica de adaptação são apresentados no Capítulo 5. Nesta etapa do trabalho, emprega-se a formulação desenvolvida e implementada no estudo de problemas de dinâmica de corpos sólidos.

1.3 Notação Utilizada

Nesta dissertação adota-se a notação matricial, utilizando-se o negrito para representar vetores, tensores e matrizes. As letras maiúsculas simbolizam as matrizes e as minúsculas os vetores e tensores.

Quando se faz necessária uma caracterização mais detalhada para uma variável expressa por uma letra, isto é realizado com o uso de índices sobrescritos ou ainda subscritos dependendo do caso. Em alguns casos as duas simbologias são adotadas no intuito de facilitar a leitura.

2 *Fundamentos: Dinâmica de Corpos Sólidos e Adaptatividade*

2.1 Formulação

As equações governantes da dinâmica estrutural em geral são concebidas de maneira a complementar a formulação da estática. Isso se faz necessário uma vez que as forças inerciais do corpo em estudo passam a ser levadas em consideração.

Algumas metodologias são empregadas quando no estudo de problemas da dinâmica, a exemplo do princípio de Hamilton. Trata-se de um procedimento variacional que considera o movimento de um corpo no espaço entre dois instantes de tempo t_1 e t_2, influenciado por efeitos combinados decorrentes de forças externas aplicadas, modelos constitutivos, amortecimento ou ainda forças inerciais.

A trajetória resultante do movimento do corpo, objetivo principal de qualquer análise dinâmica, pode ser obtida levando em consideração este princípio. Em sua declaração assume-se uma trajetória perturbada desenvolvida entre as posições inicial e final do movimento do corpo, conforme ilustrado na Figura 2.1.

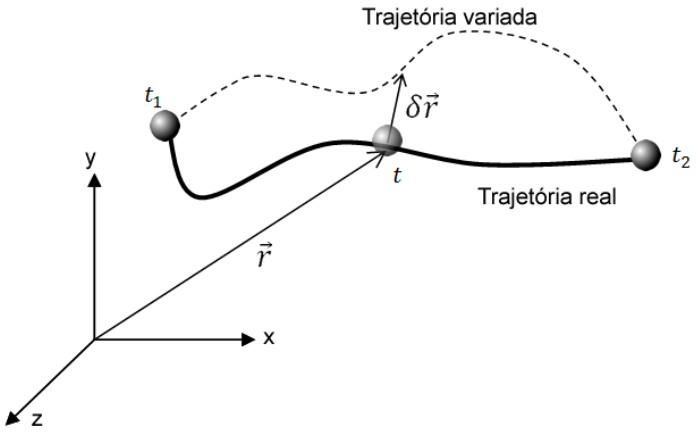


Figura 2.1: Trajetória de movimento.

O princípio de Hamilton indica que a variação temporal das diferenças entre energias

cinéticas, potenciais e o trabalho realizado por forças não-conservativas ao longo do intervalo t_1 a t_2 é nula. Sua expressão integral, descrita ao longo do tempo t, é dada por:

$$\int_{t_1}^{t_2} \delta[T(t) - V(t)]dt - \int_{t_1}^{t_2} \delta W_{nc}(t)dt = 0, \tag{2.1}$$

onde $T(t)$, $V(t)$ representam a energia cinética e a energia potencial total, respectivamente, e $W_{nc}(t)$ corresponde ao trabalho de forças não conservativas.

Note que as forças inerciais e elásticas não estão explicitamente apresentadas nesta declaração, como ocorre no princípio dos trabalhos virtuais. Ao invés disso, utilizam-se grandezas puramente escalares representando dados de energia. Estas informações podem ser obtidas para um dado sistema seja este representado espacialmente de forma contínua ou discreta. No caso discreto, um conjunto de coordenadas generalizadas, o que compõe os graus de liberdade, é utilizado na determinação destas informações. Considerando n graus de liberdade, as coordenadas generalizadas são dadas por $g_1, g_2, \ldots, g_n$. Estas entidades permitem quantificar grandezas em um dado local do espaço. Em dinâmica estrutural estas podem representar dados tais como deslocamentos, velocidades e acelerações.

Para muitos sistemas estruturais ou mecânicos, a energia cinética pode ser expressa em função das coordenadas generalizadas e das suas primeiras derivadas temporais, enquanto que a energia potencial pode ser expressa apenas em termo das coordenadas generalizadas. O trabalho virtual de forças não conservativas pode ser expresso como função das variações arbitrárias de deslocamentos nas coordenadas generalizadas e das correspondentes forças atuantes nestes locais, f_i. Estas três considerações podem ser representadas matematicamente por:

$$T = T(g_1, g_2, \ldots, g_n, \dot{g_1}, \dot{g_2}, \ldots, \dot{g_n}) \tag{2.2}$$

$$V = V(g_1, g_2, \ldots, g_n) \tag{2.3}$$

$$\delta W_{nc} = f_1 \delta g_1 + f_2 \delta g_2 + \ldots + f_n \delta g_n \tag{2.4}$$

Substituindo (2.2), (2.3) e (2.4) na Equação (2.1) e efetuando a primeira variação do termo da integral, tem-se:

$$\int_{t_1}^{t_2} \left(\sum_{i=1}^{n} \frac{\partial T}{\partial g_i} \delta g_i + \sum_{i=1}^{n} \frac{\partial T}{\partial \dot{g}_i} \delta \dot{g}_i - \sum_{i=1}^{n} \frac{\partial V}{\partial g_i} \delta g_i + \sum_{i=1}^{n} f_i \delta g_i \right) dt = 0. \tag{2.5}$$

Integrando por partes o termo dependente da primeira derivada temporal das coordenadas

generalizadas, tem-se:

$$\int_{t_1}^{t_2}\left(\sum_{i=1}^{n}\frac{\partial T}{\partial \dot{g}_i}\delta\dot{g}_i\right)dt=\left[\sum_{i=1}^{n}\frac{\partial T}{\partial \dot{g}_i}\delta g_i\right]_{t_1}^{t_2}-\int_{t_1}^{t_2}\left[\sum_{i=1}^{n}\frac{d}{dt}\Big(\frac{\partial T}{\partial \dot{g}_i}\Big)\delta g_i\right]dt. \tag{2.6}$$

Considerando a manipulação algébrica anterior, e uma vez que as variações δg_i ($i=1,2,\ldots,n$) são nulas nos instantes t_1 e t_2, a Equação (2.5) pode ser escrita como:

$$\int_{t_1}^{t_2}\left[\sum_{i=1}^{n}\frac{\partial T}{\partial g_i}\delta g_i-\sum_{i=1}^{n}\frac{d}{dt}\Big(\frac{\partial T}{\partial \dot{q}_i}\Big)\delta g_i-\sum_{i=1}^{n}\frac{\partial V}{\partial g_i}\delta g_i+\sum_{i=1}^{n}f_i\delta g_i\right]dt=0, \tag{2.7}$$

ou ainda,

$$\int_{t_1}^{t_2}\left\{\sum_{i=1}^{n}\left[\frac{\partial T}{\partial g_i}-\frac{d}{dt}\Big(\frac{\partial T}{\partial \dot{g}_i}\Big)-\frac{\partial V}{\partial g_i}+f_i\right]\delta g_i\right\}dt=0. \tag{2.8}$$

Como as variações δg_i são arbitrárias, então a equação pode ser satisfeita de uma forma geral quando:

$$\frac{d}{dt}\Big(\frac{\partial T}{\partial \dot{g}_i}\Big)-\frac{\partial T}{\partial g_i}+\frac{\partial V}{\partial g_i}=f_i. \tag{2.9}$$

As equações apresentadas em (2.9) representam as equações de movimento de Lagrange. Estas foram obtidas a partir da aplicação direta do princípio de Hamilton e pode ser empregada em sistemas que satisfaçam as hipóteses assumidas neste princípio, sejam esses lineares ou não. Note que a equação de movimento é função de informações de energia definidas com base no conjunto de coordenadas generalizadas.

Em problemas de dinâmica estrutural as coordenadas generalizadas g_i podem ser empregadas na representação discreta da grandeza deslocamento d_i ($d_i \equiv g_i$), cujas primeiras derivadas temporais representam velocidades $\dot{d}_i$ e acelerações $\ddot{d}_i$. Assim sendo, para problemas de natureza linear, é possível expressar as energias cinética e potencial de um dado corpo como:

$$T=\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}M_{ij}\dot{d}_i\dot{d}_j \tag{2.10}$$

e,

$$V=\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}K_{ij}d_id_j. \tag{2.11}$$

onde M_{ij} e K_{ij} representam respectivamente a massa e a rigidez deste corpo em sua forma espacial discreta. Em uma representação matricial, tem-se:

$$T=\frac{1}{2}\dot{\boldsymbol{d}}^T\boldsymbol{M}\dot{\boldsymbol{d}} \tag{2.12}$$

e,

$$V = \frac{1}{2}\boldsymbol{d}^T \boldsymbol{K}\boldsymbol{d}. \tag{2.13}$$

Substituindo as Equações (2.10) e (2.11) na equação de Lagrange, levando em consideração a equivalência $d_i \equiv q_i$, obtem-se a equação de movimento na forma discreta dada por:

$$\boldsymbol{M}\ddot{\boldsymbol{d}} + \boldsymbol{K}\boldsymbol{d} = \boldsymbol{f}. \tag{2.14}$$

Deve-se atentar ao fato de que todas as forças não conservativas, incluindo as de amortecimento, estão representadas pelas forças generalizadas ($\boldsymbol{f}$). A contribuição de trabalho virtual destas ao sistema então está de acordo com a forma com que estas forças manipulam a energia do sistema. Forças externas aplicadas ao corpo ($\boldsymbol{f}^{ext}$) contribuem fornecendo energia ao sistema, enquanto forças dissipativas, a exemplo das forças de amortecimento, acabam por absorver energia e desta forma passam a contribuir com sinal oposto. Assim sendo, considerando forças dissipativas dadas na forma $\boldsymbol{f}^{diss} = \boldsymbol{C}\dot{\boldsymbol{d}}$, onde $\boldsymbol{C}$ representa a matriz de amortecimento discreta, a equação de movimento pode ser re-escrita como:

$$\boldsymbol{M}\ddot{\boldsymbol{d}} + \boldsymbol{K}\boldsymbol{d} = \boldsymbol{f}^{ext} - \boldsymbol{C}\dot{\boldsymbol{d}}, \tag{2.15}$$

ou na forma mais usual:

$$\boldsymbol{M}\ddot{\boldsymbol{d}} + \boldsymbol{C}\dot{\boldsymbol{d}} + \boldsymbol{K}\boldsymbol{d} = \boldsymbol{f}^{ext}. \tag{2.16}$$

De uma forma mais geral pode-se expressar a equação de movimento agrupando suas parcelas em termos que representam forças inerciais, internas e externas, segundo a forma:

$$\boldsymbol{M}\ddot{\boldsymbol{d}} + \boldsymbol{f}^{int} = \boldsymbol{f}^{ext}. \tag{2.17}$$

Trata-se portanto de um sistema de equações diferenciais ordinárias de segunda ordem semi-discretizadas. Enquanto as funções de espaço são discretizadas nos graus de liberdade as funções de tempo são consideradas contínuas. Em um método de integração direta as funções de tempo também são discretizadas. Considerações relacionadas aos métodos de integração direta são tratados na Seção 2.2.1.

O processo de montagem das matrizes de massa $\boldsymbol{M}$ e de rigidez $\boldsymbol{K}$, em uma formulação por elementos finitos, leva em consideração a contribuição de cada elemento que compõe a discretização espacial do corpo cujo comportamento deseja-se obter. Estas matrizes são construídas a partir de um somatório de parcelas correspondentes às contribuições das matrizes correspondentes locais de cada elemento finito da malha (Bathe, 1996).

A determinação da matriz de amortecimento $\boldsymbol{C}$ é de difícil obtenção na prática, em virtude da necessidade de conhecimento de dados viscosos. Por conveniência é comum assumir esta matriz como uma combinação linear das matrizes de rigidez e de massa (amortecimento de *Rayleigh*), na forma:

$$\boldsymbol{C} = a_0\boldsymbol{M} + a_1\boldsymbol{K}, \tag{2.18}$$

onde a_0 e a_1 são constantes determinadas experimentalmente (Zienkiewicz; Taylor, 2000).

2.2 Integração no Tempo

A equação de movimento apresentada anteriormente informa um histórico de resposta para uma dada condição de carregamento aplicado a um corpo. Para problemas de dinâmica estrutural, este histórico corresponde a um campo de funções contínuas que caracterizam o comportamento do campo de deslocamentos do corpo ao longo do tempo. Uma vez que o campo de deslocamentos é discretizado no espaço, em virtude da utilização do método dos elementos finitos, a cada grau de liberdade que compõe a discretização do modelo está associada uma função temporal que revela a sua cinemática. Assim sendo, o campo de deslocamentos de todo o corpo pode ser obtido de forma aproximada a partir das informações de deslocamentos obtidas para os graus de liberdade. As próprias funções de forma dos elementos que compõem a malha do modelo podem ser utilizadas para interpolar o campo de deslocamentos dentro do domínio de cada elemento. Alguns métodos propõem a investigação das funções que descrevem a dinâmica do corpo em consideração, a exemplo do métodos modal, dos vetores de Ritz e de integração direta.

O método modal propõe a solução dos deslocamentos a partir da resolução do problema de vibrações livres dado por:

$$(\boldsymbol{K} - \omega^2\boldsymbol{M})\overline{\boldsymbol{d}} = \boldsymbol{0}. \tag{2.19}$$

Esta solução é válida para problemas lineares. Segundo esta, um vetor de deslocamentos arbitrário pode ser obtido a partir de uma combinação linear de cada de autovetores $\overline{\boldsymbol{d}}_i$, seguindo a forma:

$$\boldsymbol{d} = \sum_{i=1}^{n_m} \overline{\boldsymbol{d}}_i s_i \tag{2.20}$$

onde n_m corresponde ao número de modos de vibração considerados, e s_i são os deslocamentos modais associados. Em virtude desta combinação de modos apresentada o método modal é

tambem chamado de método de superposição modal.

O método modal é indicado quando muitas análises são realizadas para uma dada configuração, quando o tempo de simulação é grande ou quando um número pequeno de modos de vibração predominam a solução. Uma das desvantagem do método modal é a necessidade da resolução de um problema de autovalores e autovetores, que em geral é bastante dispendioso computacionalmente. Isso ocorre uma vez que no método da superposição modal os autovetores constituem a base utilizada para a composição do deslocamento.

O método dos vetores de Ritz propõe uma forma alternativa para a solução deste problema. A necessidade da solução exata dos autovetores é eliminada e uma seqüência de vetores ortogonais, ditos vetores de Ritz, é utilizada no processo de superposição (Wilson *et al.*, 1982 apud Cook *et al.*, 1989). Segundo estudos, o tempo requerido como um todo pelo método dos vetores de Ritz é um terço do gasto pela análise modal (Arnold *et al.*, 1985 apud Cook *et al.*, 1989).

Observe que o método modal e o dos vetores de Ritz resolvem a equação de movimento fazendo uso de um sistema alternativo para a representação dos graus de liberdade do modelo. Os métodos de integração direta não fazem uso deste artifício. O histórico de resposta do corpo é obtido, neste caso, a partir da integração da equação de movimento. Para tanto são utilizados valores de incrementos de tempo.

2.2.1 Métodos de Integração Direta

Os métodos de integração direta, conforme citado anteriormente, permitem a obtenção do histórico de resposta para a equação de movimento sem a necessidade da alteração das equações dinâmicas (2.17). Para tanto, a variável tempo é discretizada e a cinemática do corpo é obtida em um processo de integração passo-a-passo. Desta forma, os deslocamentos da estrutura são computados em instantes de tempo separados por incrementos de tempo de integração. A equação de movimento assume, desta maneira, a sua forma discreta:

$$\boldsymbol{M}\ddot{\boldsymbol{d}}_n + \boldsymbol{f}_n^{int} = \boldsymbol{f}_n^{ext} \tag{2.21}$$

em um dado passo de tempo n.

Para um dado instante de tempo qualquer, e considerando um valor fixo para Δt, é possível encontrar um valor para n de forma simples. Este valor representa o número de vezes que a Equação (2.21) foi avaliada de forma incremental até a obtenção da resposta no instante. Em uma estratégia de adaptação no tempo o valor de Δt não é necessariamente fixo ao longo do

processo de integração. A determinação do valor de n deve neste caso ser feita de maneira apropriada.

De maneira semelhante à consideração feita para a Equação (2.16), a Equação (2.21) pode ser re-escrita de forma a considerar o amortecimento estrutural e uma relação tensão-deformação que ocorra de maneira elástica e linear. Sua forma discretizada, e portanto incremental, é dada por:

$$\boldsymbol{M}\ddot{\boldsymbol{d}}_n + \boldsymbol{C}\dot{\boldsymbol{d}}_n + \boldsymbol{K}\boldsymbol{d}_n = \boldsymbol{f}_n^{ext}. \tag{2.22}$$

O processo de discretização no tempo é ainda acompanhado de aproximações por diferenças finitas para as derivadas temporais. Novamente o incremento de tempo de integração tem influência sobre a magnitude dos erros envolvidos com a utilização desta técnica aproximativa. Na Seção 2.2.1 esta técnica de aproximação é tratada em maiores detalhes.

Os métodos de integração direta são classificados como explícitos ou implícitos dependendo da forma com que a resposta dos passos é obtida. Os algoritmos explícitos utilizam uma expressão na forma geral:

$$\boldsymbol{d}_{n+1} = f(\boldsymbol{d}_n, \dot{\boldsymbol{d}}_n, \ddot{\boldsymbol{d}}_n, \boldsymbol{d}_{n-1}, \dots). \tag{2.23}$$

Observe que para a evolução da resposta para um passo qualquer $n+1$ são necessárias apenas informações de passos anteriores a este. Quando um algoritmo de integração direta faz uso de informações do passo $n+1$ esse é classificado como implícito. Logo, os algoritmos implícitos seguem a forma:

$$\boldsymbol{d}_{n+1} = f(\dot{\boldsymbol{d}}_{n+1}, \ddot{\boldsymbol{d}}_{n+1}, \boldsymbol{d}_n, \dot{\boldsymbol{d}}_n, \ddot{\boldsymbol{d}}_n, \dots). \tag{2.24}$$

No processo incremental envolvido em um algoritmo de integração direta alguns aspectos devem ser observados. Em geral estes aspectos estão associados a fatores tais como estabilidade e economia. Na prática estes dois fatores constituem as principais diferenças entre os algoritmos explícitos e implícitos. A estabilidade pode ser entendida como sendo a capacidade de um algoritmo não amplificar erros provenientes de truncamentos numéricos ou de integrações imprecisas de modos de freqüência alta. O termo economia aqui apresentado está relacionado com os custos das operações realizadas na avaliação de um passo de integração.

Os algoritmos de integração explícitos possuem estabilidade condicional. Isso significa que a estabilidade só é assegurada com o uso de um incremento de tempo de integração menor que um certo valor crítico. Quando essa exigência não é atendida o processo numérico se

torna instável e a convergência para uma solução pode não ocorrer. Uma vez que o valor do incremento de tempo crítico Δt_{crit} é em geral bastante pequeno, muitos passos de integração são necessários no processo de integração numérica. No entanto, as operações efetuadas a cada passo de integração não são dispendiosas permitindo que elas sejam obtidas de forma rápida.

Os métodos explícitos de integração ainda permitem que a resolução da equação de recorrência seja realizada de maneira desacoplada, ou seja, com avaliação independente para os seus termos. Para tanto requer-se que a matriz de massa esteja em sua forma diagonalizada. Esta consideração tem forte impacto positivo nas rotinas computacionais que implementam esta técnica. Mecanismos de paralelização computacional podem ainda ser utilizados de forma adicional no intuito de otimizar o processo de integração, sobretudo em modelos onde a discretização no espaço ocorre de maneira mais refinada (Fahmy; Namini, 1994).

Os métodos de integração implícitos, sob certas condições, possuem estabilidade incondicional. Isso significa que a estabilidade do algoritmo é garantida independentemente do valor do incremento de integração que esteja sendo utilizado. Hughes (1987) apresenta o conjunto de parâmetros para os quais os algoritmos de família de Newmark (Seção 2.2.1) são incondicionalmente estáveis. Isso no entanto não significa que o processo de integração implícita ocorra de maneira mais rápida do que o correspondente processo realizado por um método explícito, uma vez que os cálculos envolvidos em cada passo de integração são mais dispendiosos. Além disso, as matrizes envolvidas no processo de integração não são diagonais, o que acaba por consumir muito mais recursos de armazenamento.

O uso de métodos implícitos é também desfavorecido quando fortes não-linearidades estão presentes, tais como em problemas de propagação de ondas (contato e impacto) ou ainda em análises de estruturas elasto-plásticas submetidas a carregamentos severos. O estudo destes casos exige uma simulação mais refinada que se reflete na utilização de um incremento de tempo de integração pequeno. Desta forma, a eficiência do método de integração implícita acaba sendo comprometida. Para problemas inerciais, freqüentemente chamados de problemas de dinâmica estrutural, o emprego de um método de integração implícito é mais apropriado. Nestes casos as baixas freqüências dominam a resposta e os carregamentos atuantes variam mais lentamente, a exemplo dos carregamentos resultantes de um terremoto. Lowrie (2004) apresenta um estudo comparativo entre alguns algoritmos de integração implícitos, tais como Runge–Kutta e Crank–Nicolson. O trabalho apresenta ainda considerações acerca do incremento de tempo de integração frente a problemas não-lineares.

Belytschko & Hughes (1983) apresentam um estudo comparativo onde os métodos de integração explícitos e implícitos são confrontados frente ao número de multiplicações

envolvidas em um passo de integração. São avaliados três exemplos correspondentes a casos de malhas unidimensionais, planas e tridimensionais. O estudo mostra que a relação de custo computacional, implícito por explícito, tende a crescer com a dimensão da malha. A razão de tal crescimento é em parte associada ao crescimento de forma quadrática dos termos das matrizes cheias utilizadas nos algoritmos implícitos.

No decorrer deste trabalho apenas os algoritmos de integração explícitos são apresentados. Esta opção é baseada no fato de que as aplicações utilizadas na elaboração dos exemplos apresentados neste trabalho optam pelo uso desses algoritmos.

Algoritmos de Integração Explícitos

Conforme apresentado anteriormente, a estabilidade dos algoritmos de integração explícitos é condicionada ao uso de um incremento de integração crítico (Δt_{crit}). Para a definição deste incremento leva-se em consideração o valor da mais alta freqüência natural verificada na malha que discretiza o modelo estudado (ω_{max}). De uma forma geral o valor do incremento de integração Δt a ser utilizado no processo de integração deve atender a:

$$\Delta t \leq \frac{2}{\omega_{max}} = \Delta t_{crit}. \tag{2.25}$$

A forma deste limite superior só é válida para os casos onde o amortecimento não é considerado. Quando este mecanismo dissipativo se faz presente, a Equação (2.25) é corrigida por um fator que leva em conta a fração do amortecimento crítico ξ correspondente à máxima freqüência do sistema. O incremento de tempo deve neste caso deve atender ao critério:

$$\Delta t \leq \frac{2}{\omega_{max}}(\sqrt{1-\xi^2}-\xi). \tag{2.26}$$

Quando as exigências de estabilidade apresentadas anteriormente não são atendidas o processo de integração falha, divergindo ou conduzindo a respostas espúrias. Por outro lado, o uso de um incremento desnecessariamente pequeno torna o custo dos cálculos bastante dispendioso, sendo também indesejável. Logo, o conhecimento preciso de ω_{max} permite que o processo de integração ocorra de forma otimizada. Na prática o conhecimento exato deste valor é evitado, uma vez que requer que o problema de autovalor da Equação (2.19) seja resolvido e que a matriz de rigidez global seja montada. Recorre-se portanto ao conhecimento de um valor aproximado. Uma forma de aproximar o valor de ω_{max} é supor o maior valor de freqüência natural dentre todos os elementos sem suporte que compõem a malha de elementos finitos

(Belytschko; Hughes, 1983). Esta forma de obtenção simplifica bastante o processo uma vez que o cálculo das freqüências de vibração da Equação (2.19) é feito para os elementos de forma individual, e não da malha como um todo. Algumas vezes este cálculo é bastante simplificado, como no caso dos elementos de barra com massa concentrada. Para este tipo de elemento, a freqüência máxima é obtida a partir de informações geométricas e do material, tais como o seu o módulo de elasticidade E, densidade ρ e comprimento L, na forma:

$$\omega_{elem} = \frac{2}{L}\sqrt{\frac{E}{\rho}}. \tag{2.27}$$

Cook *et al.* (1989) apresentam ainda uma forma alternativa de aproximação para o valor de ω_{max} baseado no teorema do círculo de Gerschgorin (Ferreira, 2007). Considerando uma matriz de massa concentrada (diagonal) e sendo n o número de graus de liberdade, tem-se que:

$$\omega_{max}^2 \leq \max_i \left(\frac{1}{\boldsymbol{M}_{ii}} \sum_{j=1}^{n} |\boldsymbol{K}_{ij}| \right) \quad \text{onde} \quad i = 1, 2, \ldots, n. \tag{2.28}$$

Observe que os valores absolutos dos termos de cada linha da matriz de rigidez $\boldsymbol{K}$ são escalados pelo termo correspondente da matriz de massa $\boldsymbol{M}$ e somados. O máximo valor obtido para a soma das linhas caracteriza um limite superior para o valor correto de ω_{max}, podendo ser utilizado como referência de aproximação.

Uma vez que o valor da máxima freqüência esteja definido, seja de forma exata ou aproximada, um valor para o incremento de tempo que atende às exigências de estabilidade é escolhido e um histórico de resposta pode ser então obtido. Tomando como base a equação de movimento apresentada anteriormente, este histórico é caracterizado pelas informações temporais do vetor de deslocamentos nos graus de liberdade do corpo $\boldsymbol{d}$. Em sua obtenção são consideradas aproximações por diferenças finitas para as derivadas temporais, $\dot{\boldsymbol{d}}$ e $\ddot{\boldsymbol{d}}$, que correspondem respectivamente aos vetores de velocidades e acelerações.

A seguir são apresentados detalhes de alguns algoritmos de integração direta explícita. Estes algoritmos serão utilizados nos próximos capítulos quando os exemplos de aplicação da técnica de adaptatividade no tempo forem alvo de estudo.

Método das Diferenças Centrais

Um dos clássicos algoritmos de integração direta explícita largamente utilizado é o das diferenças centrais. Trata-se de um algoritmo de fácil implementação e que possue variantes

apresentadas por diversos autores. O Método das Diferenças Centrais pode ser encarado como uma particularização do algoritmo implícito de Newmark, cujas equações de recorrência são dadas por:

$$\boldsymbol{d}_{n+1} = \boldsymbol{d}_n + \Delta t \dot{\boldsymbol{d}}_n + \Delta t^2 \left[\left(\frac{1}{2} - \beta \right) \ddot{\boldsymbol{d}}_n + \beta \ddot{\boldsymbol{d}}_{n+1} \right], \tag{2.29}$$

e,

$$\dot{\boldsymbol{d}}_{n+1} = \dot{\boldsymbol{d}}_n + \Delta t [(1-\gamma)\ddot{\boldsymbol{d}}_n + \gamma \ddot{\boldsymbol{d}}_{n+1}]. \tag{2.30}$$

Observe que o deslocamento é aproximado por uma expansão em série de Taylor com uma precisão de segunda ordem e que, quando se supõe $\beta = 0$, o algoritmo de Newmark torna-se explícito, em conformidade com a forma geral apresentada na Equação (2.23). A definição de $\gamma = \frac{1}{2}$ é portanto o que justifica a denominação do algoritmo como sendo de diferença central. O algoritmo do método das diferenças centrais, variante do algoritmo de Newmark, é apresentado na Figura 2.2.

Inicialização
a. Definição das condições iniciais do problema ($n = 0$) e montagem da matriz de massa.

Processo Iterativo
1. Cálculo da velocidade no passo n
   $\dot{\boldsymbol{d}}_n = \dot{\boldsymbol{d}}_{n-1} + \frac{\Delta t}{2}\left[\ddot{\boldsymbol{d}}_n + \ddot{\boldsymbol{d}}_{n-1}\right]$
2. Cálculo dos deslocamentos no passo $n+1$
   $\boldsymbol{d}_{n+1} = \boldsymbol{d}_n + \Delta t \dot{\boldsymbol{d}}_n + \frac{\Delta t^2}{2}\ddot{\boldsymbol{d}}_n$
3. Predição de velocidade no passo $n+1$
   $\dot{\boldsymbol{d}}^p_{n+1} = \dot{\boldsymbol{d}}_n + \Delta t \ddot{\boldsymbol{d}}_n$
   ou
   $\dot{\boldsymbol{d}}^p_{n+1} = \left[\boldsymbol{d}_{n+1} - \boldsymbol{d}_n\right]/\Delta t$
4. Cálculo do vetor de forças desequilibradas
   $\boldsymbol{f}^{des}_{n+1} = \boldsymbol{f}^{ext}_{n+1} - \boldsymbol{f}^{int}_{n+1} - \boldsymbol{C}\dot{\boldsymbol{d}}^p_{n+1}$
5. Cálculo da aceleração no passo $n+1$
   $\boldsymbol{M}\ddot{\boldsymbol{d}}_{n+1} = \boldsymbol{f}^{des}_{n+1}$
6. Atualização do passo e tempo corrente
   $n \leftarrow n+1$
   $t \leftarrow t + \Delta t$

Figura 2.2: Algoritmo do Método das Diferenças Centrais (Variante de Newmark).

No trabalho de Krysl & Belytschko (1998) outra variante para o método das diferenças centrais é apresentada. Esta variante toma como base o algoritmo de integração apresentado por Park & Underwood (1980) e faz uma correção iterativa da velocidade presente na equação

de movimento.

Chung-Lee

Um algoritmo de integração com características similares ao método das diferenças centrais é apresentado por Chung & Lee (1994). Trata-se de um esquema de integração explícito aplicável a problemas lineares e não-lineares e que faz uso de técnicas de dissipação numérica.

Rio *et al.* (2005) apresentam um estudo comparativo entre alguns algoritmos de integração explícitos. Segundo os autores, o uso de um esquema de integração explícito em alguns problemas de elementos finitos introduz oscilações espúrias de modos de alta freqüência e que devem ser filtradas. O algoritmo de Chung & Lee é apontado por Rio *et al.* como sendo interessante em problemas deste tipo, uma vez que suaviza mais especificamente as altas freqüências. Isso é possível uma vez que o algoritmo faz uso de amortecedores numéricos. Modelos numéricos para problemas de contato e impacto, por exemplo, costumam fazer uso de tais artifícios.

No trabalho de Ryu *et al.* (2000), por exemplo, o algoritmo de Chung & Lee é empregado no estudo de problemas de contato impulsivo. Este tipo de problema conduz a respostas com sinais de alta freqüência que, no contexto do trabalho citado, estão presentes na análise do movimento de veículos do tipo tanque (*tracked-vehicle*).

O algoritmo de Chung & Lee introduz o uso de um único parâmetro livre Ψ relacionado com a precisão, a estabilidade e a convergência. Segundo os autores, a faixa de utilização deste parâmetro é dada por:

$$1 \leq \Psi \leq 28/27. \tag{2.31}$$

Quando $\Psi = 1$ o algoritmo não apresenta dissipação numérica e o comportamento do algoritmo é semelhante ao das diferenças centrais. Quando Ψ assume o outro valor limite ($\Psi = 28/27$), ocorre a maior dissipação numérica sem perda da estabilidade e sem amortecer significativamente os modos de baixa freqüência. O algoritmo de Chung & Lee é apresentado na Figura 2.3.

Inicialização
a. Definição das condições iniciais do problema ($n = 0$) e montagem da matriz de massa.
b. Definição do parâmetro Ψ - Equação (2.31).
c. Cálculo dos parâmetros Ψ_1, Ψ_2, ϕ_1 e ϕ_2:

$$\Psi_1 = \Delta t^2\left(\tfrac{1}{2} - \beta\right)$$
$$\Psi_2 = \Delta t^2 \Psi$$
$$\phi_1 = -\tfrac{1}{2}\Delta t$$
$$\phi_2 = \tfrac{3}{2}\Delta t$$

Processo Iterativo
1. Cálculo do vetor de forças desequilibradas:

$$\boldsymbol{f}^{des}_{n+1} = \boldsymbol{f}^{ext}_{n+1} - \boldsymbol{f}^{int}_{n+1} - \boldsymbol{C}\dot{\boldsymbol{d}}^{p}_{n+1}$$

2. Cálculo da aceleração no passo $n+1$:

$$\boldsymbol{M}\ddot{\boldsymbol{d}}_{n+1} = \boldsymbol{f}^{des}_{n+1}$$

3. Cálculo do deslocamento no passo $n+1$:

$$\boldsymbol{d}_{n+1} = \boldsymbol{d}_n + \Delta t\dot{\boldsymbol{d}}_n + \Psi_1\ddot{\boldsymbol{d}}_n + \Psi_2\ddot{\boldsymbol{d}}_{n+1}$$

4. Cálculo da velocidade no passo $n+1$:

$$\dot{\boldsymbol{d}}_{n+1} = \dot{\boldsymbol{d}}_n + \phi_1\ddot{\boldsymbol{d}}_n + \phi_2\ddot{\boldsymbol{d}}_{n+1}$$

5. Atualização do passo e tempo corrente:

$$n \leftarrow n+1$$
$$t \leftarrow t+\Delta t$$

Figura 2.3: Algoritmo de Chung-Lee.

Método Explícito Generalizado - α

O Método Explícito Generalizado - α (MEG-α) é apresentado por Hulbert & Chung (1996) como sendo um esquema de integração explícito baseado no Método Implícito Generalizado - α. O método faz uso de um parâmetro que define a dissipação numérica nas altas freqüências e pode ser utilizado efetivamente em problemas de dinâmica em que a dissipação numérica é necessária para minimizar o erro numérico inerente ao processo de integração. Em problemas de impacto, por exemplo, o uso destes dissipadores se faz necessário. PantalÉ (2005), no estudo de problemas com tal característica, utiliza este esquema de integração numérica temporal em um código computacional responsável pela análise dinâmica por elementos finitos.

Assim como no projeto de outros algoritmos dissipativos, o método trata das dissipações em altas-freqüências de vibração com o uso de dissipadores numéricos. Isso, no entanto, é realizado de maneira a minimizar o amortecimento em modos de vibração de baixa-freqüência.

O MEG-α conta com o uso de preditores de deslocamentos e velocidades que acabam por torná-lo mais robusto quando comparado ao algoritmo clássico das diferenças centrais. Três novos parâmetros são introduzidos no algoritmo deste método e suas correspondentes influências sobre a precisão e a estabilidade do processo de integração são discutidos em

maiores detalhes no trabalho de Hulbert & Chung. A Figura 2.4 apresenta o procedimento de integração do algoritmo MEG-α.

Inicialização

a. Definição das condições iniciais do problema ($n = 0$) e montagem da matriz de massa.

b. Definição do parâmetro que define a dissipação numérica - ρ_∞ (Hulbert; Chung, 1996)

c. Cálculo dos Parâmetros α_m, α_f, λ e Ω:

$$\alpha_m = \frac{2\rho_\infty - 1}{\rho_\infty + 1},\ \alpha_f = \frac{\rho_\infty}{\rho_\infty + 1},\ \lambda = \left(\frac{1}{2} - \alpha_m + \alpha_f\right) \text{ e } \Omega = \frac{1}{4}\left(\frac{1}{2} + \lambda\right)^2$$

Processo Iterativo

1. Predição de deslocamentos e velocidades no passo $n+1$

$$\boldsymbol{d}_{n+1}^{p} = \boldsymbol{d}_n + \Delta t \dot{\boldsymbol{d}}_n + \Delta t^2 \left(\frac{1}{2} - \Omega\right) \ddot{\boldsymbol{d}}_n$$

$$\dot{\boldsymbol{d}}_{n+1}^{p} = \dot{\boldsymbol{d}}_n + \Delta t (1 - \lambda) \ddot{\boldsymbol{d}}_n$$

2. Estima os valores de deslocamentos e velocidades em $t_{n+1-\alpha_f}$

$$\boldsymbol{d}_{n+1-\alpha_f} = (1 - \alpha_f)\boldsymbol{d}_{n+1}^{p} + \alpha_f \boldsymbol{d}_n$$

$$\dot{\boldsymbol{d}}_{n+1-\alpha_f} = (1 - \alpha_f)\dot{\boldsymbol{d}}_{n+1}^{p} + \alpha_f \dot{\boldsymbol{d}}_n$$

3. Determina o vetor de forças internas $\boldsymbol{f}_{n+1-\alpha_f}^{int}$ a partir de $\boldsymbol{d}_{n+1-\alpha_f}$ e $\dot{\boldsymbol{d}}_{n+1-\alpha_f}$

4. Aplica a equação de balanço para determinar $\ddot{\boldsymbol{d}}_{n+1-\alpha_f}$

$$\boldsymbol{f}_{n+1-\alpha_f}^{des} = \boldsymbol{f}_{n+1-\alpha_f}^{ext} - \boldsymbol{f}_{n+1-\alpha_f}^{int} - \boldsymbol{C}\dot{\boldsymbol{d}}_{n+1-\alpha_f}$$

$$\boldsymbol{M}\ddot{\boldsymbol{d}}_{n+1-\alpha_m} = \boldsymbol{f}_{n+1-\alpha_f}^{des}$$

5. Determina a aceleração no passo $n+1$ a partir de $\ddot{\boldsymbol{d}}_{n+1-\alpha_m}$:

$$\ddot{\boldsymbol{d}}_{n+1} = \frac{\ddot{\boldsymbol{d}}_{n+1-\alpha_m} - \alpha_m \ddot{\boldsymbol{d}}_n}{1 - \alpha_m}$$

6. Correção para os valores previstos de deslocamentos e velocidades no passo $n+1$:

$$\boldsymbol{d}_{n+1} = \boldsymbol{d}_{n+1}^{p} + \Omega \Delta t^2 \ddot{\boldsymbol{d}}_{n+1}$$

$$\dot{\boldsymbol{d}}_{n+1} = \dot{\boldsymbol{d}}_{n+1}^{p} + \lambda \Delta t \ddot{\boldsymbol{d}}_{n+1}$$

7. Atualização do passo e tempo corrente

$$n \leftarrow n + 1$$

$$t \leftarrow t + \Delta t$$

Figura 2.4: Algoritmo do Método Explícito Generalizado - α.

Note que neste método de integração a equação de movimento é atendida em um instante de tempo intermediário entre os passos n e $n+1$. Tal fato tem implicações sobre alguns estimadores de refinamento que utilizam informações em um passo genérico inteiro n. A Seção 2.3.2 apresenta maiores detalhes a respeito destes estimadores.

2.3 Adaptatividade

No desenvolvimento e utilização de programas de análise dinâmica estrutural diversos procedimentos são utilizados no intuito de otimizar o desempenho computacional dos algoritmos de integração envolvidos. Isso se faz necessário uma vez que a tarefa de integração

numérica temporal é em geral bastante dispendiosa computacionalmente e que portanto qualquer melhoria obtida neste processo reflete-se em reduções de tempo de análise. Essas reduções são proporcionais ao tamanho do modelo numérico simulado e podem viabilizar análises mais refinadas para estes modelos.

Um fator que tem fundamental importância neste processo, no caso de algoritmos de integração explícitos, é o incremento de tempo de integração (Δt). Esta variável é responsável pelo grau de refinamento da resposta numérica e tem forte influência sobre a eficiência do processo de integração. Isso ocorre uma vez que quanto menor é o valor do incremento de tempo mais avaliações instantâneas do comportamento estrutural do modelo são realizadas em um dado intervalo de tempo de análise. O número maior de avaliações por intervalo permite uma descrição mais detalhada e precisa do comportamento estrutural do modelo, no entanto acaba por aumentar o tempo gasto pela rotina computacional responsável pelo trabalho de integração no tempo.

A escolha do incremento de tempo a ser utilizado em uma análise dinâmica é, muitas vezes, baseada em procedimentos heurísticos. Um valor de incremento de tempo é estabelecido e este é utilizado ao longo de toda a análise. Embora seja possível ter a certeza de convergência da resposta, nem sempre é possível saber se esta apresenta um erro de discretização que seja considerado aceitável. Quando um problema envolve grandes variações nas condições de contorno e na geometria da estrutura (grandes deslocamentos), o analista, em geral, acaba optando pelo uso de incrementos de tempo de integração muito pequenos no intuito de assegurar a qualidade da resposta. Uma vez que o incremento de tempo é mantido constante durante toda a análise, o desempenho global do processo de integração decai em função, muitas vezes, de situações que ocorrem em um curto intervalo de tempo. Em problemas transientes e não-lineares é comum a obtenção de grandes variações como as citadas.

Com o objetivo de amenizar problemas deste tipo são utilizados procedimentos adaptativos para o controle automático do incremento de tempo ao longo da análise. Em geral estes procedimentos utilizam informações de passos de integração anteriores para estabelecer automaticamente o incremento de tempo a ser utilizado nos próximos passos. Isso, no entanto, ocorre de maneira otimizada, buscando sempre levar em consideração a precisão da resposta, a estabilidade do método e o desempenho da implementação computacional.

2.3.1 Considerações e Aspectos Importantes

Embora a idéia principal da adaptação no tempo seja bastante simples, de alterar o incremento de tempo ao longo da simulação, algumas exigências devem ser atendidas de

maneira que este procedimento ocorra de maneira robusta e eficiente. Deve-se assegurar que o mecanismo de adaptatividade busque de forma otimizada a precisão de respostas e a melhoria da eficiência, sempre que possível. Além disso, uma vez que o algoritmo de integração adaptativo faz uso de um algoritmo de integração numérica, todas as peculiaridades deste último algoritmo devem ser levadas em conta. Muitas vezes a simples alteração da variável que controla o tamanho do passo de integração requer a atualização de parâmetros dependentes.

Para que o algoritmo de integração adaptativo tenha sucesso no seu propósito é requerido que o custo complementar de implementação do seu mecanismo seja o mínimo possível. Assim, a eficiência do processo de integração temporal não é afetada e, portanto, os benefícios do uso da adaptação se tornam mais claros. Além disso, sugere-se que os parâmetros de controle do algoritmo adaptativo sejam simples a fim de facilitar o seu uso por parte do usuário que irá lidar com o modelo numérico avaliado.

Silveira (2001), baseado no trabalho de Bergan & Mollestad (1985), relaciona algumas outras particularidades que devem ser observadas em um procedimento de integração temporal adaptativa. De uma forma geral, pode-se destacar os seguintes aspectos:

- Os ajustes no incremento de tempo devem ser realizados tomando como referência um incremento de tempo inicial escolhido. O valor deste incremento deve ser portanto menor que o valor crítico que assegura a convergência da resposta. Assim sendo, as soluções iniciais são obtidas e passam então a fornecer informações para o decorrer do procedimento de adaptação.

- Durante o regime de solução permanente o incremento de tempo deve permanecer constante. Desta forma modificações desnecessárias e que podem comprometer a estabilidade e a precisão do algoritmo são evitadas.

- O critério de avaliação do incremento de tempo adotado (critério de erro) deve ter um custo computacional mínimo, de modo a não comprometer o desempenho do algoritmo.

- O algoritmo adaptativo deve reagir de maneira imediata a mudanças súbitas no comportamento da estrutura.

- O incremento de tempo apenas deve ser alterado quando for necessário.

Söderlind & Wang (2006) fazem ainda considerações a respeito das mudanças para o incremento de tempo de integração em um procedimento adaptativo. Segundo os autores, este procedimento tem influência negativa sobre a estabilidade computacional. O problema é

ainda potencializado quando as seqüências de incrementos de tempo utilizados na integração são desordenadas e desta forma prejudicial à estabilidade do processo de integração. O tratamento de regularização destas sequências é apontado portanto como uma solução para este problema. Melhorias significativas na estabilidade computacional são observadas com o uso deste procedimento.

2.3.2 Estimadores de Refinamento

Um procedimento necessário no estudo de adaptatividade temporal consiste no processo de estimação de refinamento. Isso se faz necessário uma vez que, em uma análise adaptativa, o conhecimento de alguma informação que forneça subsídios para o controle do incremento de tempo de integração é requerido. A técnica de adaptação é fundada nesta informação e, quanto mais esta revela da dinâmica do problema em questão, melhor é o procedimento adaptativo.

Os ditos estimadores de refinamento buscam de alguma forma quantificar a distância existente entre a solução numérica obtida e a solução exata para o problema em questão. Assim sendo, tais ferramentas podem ser utilizadas em uma análise adaptativa. Note que o incremento de tempo de integração pode ser controlado como função destes estimadores. Nos instantes da análise onde a distância, ou o valor do erro de integração, está acima de uma tolerância permitida faz-se necessário um decremento do intervalo de tempo de integração (Δt). Desta forma, aquele instante pode ser resolvido de uma maneira mais apurada, o que conduz a respostas de melhor qualidade. Quando o valor do erro está dentro de um intervalo de tolerância prescrita, um incremento em Δt é permitido desde que este varie de maneira suave e que portanto não venha a perturbar o algoritmo de integração. Logo, o processo de integração temporal pode agora ocorrer com um maior desempenho (mais rápido) e de maneira a não comprometer a qualidade da resposta.

Os estimadores de refinamento recebem denominações específicas dependendo da forma com que eles são obtidos ao longo dos passos de integração. Os ditos estimadores a *posteriori* são aqueles que são obtidos a partir do conhecimento da resposta numérica no instante em que o estimador é calculado. Zienkiewicz & Xie (1991) e Hulbert & Jang (1995) fazem uso de estimadores de refinamento com tal característica. Os estimadores de refinamento a *priori*, ao contrário dos apresentados anteriormente, são obtidos sem a necessidade do conhecimento da resposta numérica. Desta forma o incremento de integração pode ser selecionado adaptativamente sem a necessidade de processos de re-análise, muitas vezes requeridos quando se utiliza estimadores de refinamento a *posteriori*. Chung *et al.* (2003), por exemplo, fazem uso de um estimador a *priori* para o Metodo Explícito Generalizado-α.

A seguir são apresentados dois estimadores de refinamento: o primeiro baseado na solução local aproximada e o segundo no resíduo de forças desequilibradas. Uma técnica de estimação de refinamento baseada no indicador geométrico curvatura, alvo de estudo deste trabalho, é ainda introduzida na Seção 3.2.

Solução local aproximada

A estratégia de adaptação no tempo baseada em soluções locais aproximadas consiste em comparar a solução numérica obtida em um processo de integração com uma solução dita local aproximada. A distância entre as duas soluções é quantificada e esta informação pode ser utilizada como um bom estimador de refinamento a ser utilizado em uma estratégia de adaptatividade. A solução local aproximada em geral é obtida de uma forma mais apurada que a solução numérica utilizada na integração e esta diferença está associada com a sua precisão. No caso do algoritmo de Newmark, por exemplo, esta precisão é de segunda ordem e está associada ao número de termos em uma seqüência da série de Taylor utilizada. Segundo Romero & Lacoma (2006), a solução melhorada tem precisão no mínimo uma vez maior do que a obtida pela resposta numérica. O trabalho de Romero & Lacoma apresenta uma metodologia para a formulação de estimadores de refinamento aplicável aos métodos de integração direta mais comumente utilizados em dinâmica estrutural. A metodologia proposta é válida para o método de Newmark e Hilber-Hughes-Taylor (Hilber *et al.*, 1977). A técnica de estimativa de refinamento baseada em soluções locais no entanto não é restritiva, podendo ser utilizada em diversos métodos de solução, desde que se possa obter uma solução de maior precisão para ser tomada como referência. No trabalho de Wang *et al.* (2001) e Benderskaya *et al.* (2008), por exemplo, esta técnica é empregada no método de Runge-Kutta.

O estimador de refinamento baseado em erro local, $\boldsymbol{e}_{n+1}$, medido a partir do avanço da solução do tempo t_n ao tempo t_{n+1} é dado portanto por:

$$\boldsymbol{e}_{n+1} = \boldsymbol{d}_{n+1} - \boldsymbol{u}_{n+1} \tag{2.32}$$

onde $\boldsymbol{u}_{n+1} = \boldsymbol{u}_{t_{n+1}}$ é a solução local aproximada e $\boldsymbol{d}_{n+1}$ a resposta numérica obtida.

Observe que problemas que envolvem múltiplos graus de liberdade devem avaliar o erro local dado pela Equação (2.32) ao longo de todo o domínio do modelo. É necessário que se atente ao fato de que este indicador será utilizado para a definição do incremento de tempo e que portanto deve ser armazenado de maneira apropriada em uma implementação computacional. Isso se faz necessário uma vez que o processo de adaptação pode ocorrer de maneira global ou local com relação aos graus de liberdade. Para o primeiro caso, as trocas em Δt são válidas para

todo o modelo e portanto apenas a informação de erro local máximo observado no domínio é necessária para armazenamento. Em casos onde as alterações do incremento de tempo podem ocorrer de maneira local, o vetor de erros locais continua sendo utilizado como parâmetro para a adaptação. A diferença é que este vetor agora está dividido em partes, correspondentes aos subdomínios do modelo. A técnica de utilização de diferentes incrementos de tempo para diferentes subdomínios do modelo é chamada de subciclagem (Daniel, 1998).

Na Seção 2.3.3 apresenta-se uma expressão para o erro local de forma específica para o Método Explícito Generalizado - α.

Resíduo de forças desequilibradas

Um outro estimador de refinamento tradicionalmente utilizado em análises dinâmicas baseia-se na informação do resíduo de forças desequilibradas. Diferentemente daqueles estimadores que utilizam soluções locais aproximadas para avaliar a precisão de uma análise, este indicador utiliza apenas as informações da resposta numérica obtida para poder caracterizá-la e sugerir mudanças para o incremento de tempo de integração. Este procedimento, no entanto, é aplicável apenas aos casos em que a equação de movimento (2.21) é atendida de maneira aproximada pelos algoritmos de integração numérica, a exemplo do Método Explícito Generalizado-α. Para estes algoritmos, o resíduo de forças desequilibradas, correspondente ao numerador do segundo membro da Equação (2.33), é não necessariamente nulo. Quando isso for verificado o incremento de tempo de integração deve ser reduzido no sentido de atender melhor a equação que governa o problema. O estimador de refinamento baseado no resíduo de forças desequilibradas $r_f(t)$ é dado na forma:

$$r_f(t) = \frac{\|\boldsymbol{f}^{ext} - \boldsymbol{f}^{int} - \boldsymbol{M}\ddot{\boldsymbol{d}}\|}{\|\boldsymbol{f}^{ext}\|} \tag{2.33}$$

onde $\boldsymbol{f}^{ext}$ é o vetor de esforços externos atuantes, $\boldsymbol{f}^{int}$ é o vetor de esforços internos, $\boldsymbol{M}$ é a matriz de massa e $\ddot{\boldsymbol{d}}$ é o vetor da acelerações. Supõe-se que $\|\boldsymbol{f}^{ext}\| \neq 0$ no instante em que $r_f(t)$ é calculado. Para uma análise precisa estima-se que um resíduo em torno de 1% seja aceitável (Cook *et al.*, 1989).

Assim como para os estimadores de refinamento baseados em soluções locais aproximadas, o uso do resíduo de forças desequilibradas como ferramenta indicativa de ajuste do incremento de tempo de integração requer cuidados especiais. Deve-se assegurar que as mudanças em Δt não ocorram de maneira freqüente ou brusca com o objetivo de não afetar a estabilidade do algoritmo de integração. O estimador de refinamento baseado no resíduo de forças é utilizado

em um procedimento adaptativo no trabalho de Cintra & Silveira (2007).

Observe que para problemas envolvendo múltiplos graus de liberdade, o resíduo de forças desequilibradas também pode ser utilizado em estratégias de adaptação no tempo que ocorrem de maneira global ou local. Para o último caso, a Equação (2.33) é avaliada para subdomínios do modelo estudado.

2.3.3 Método Explícito Generalizado - α Adaptativo

Nesta seção é apresentada uma estratégia de adaptação no tempo que faz uso do algoritmo do Método Explícito Generalizado-α. O emprego deste mecanismo adaptativo será apresentado no capítulo deste trabalho onde os exemplos numéricos são apresentados. De qualquer maneira, alguns artifícios aqui tratados são usados como referência para o desenvolvimento da metodologia de adaptação no tempo baseada em curvatura, alvo de estudo desta dissertação.

O Método Explícito Generalizado - α Adaptativo é introduzido por Silveira (2001) no contexto de análises dinâmicas de linhas de ancoragem e *risers*. Seu desenvolvimento é motivado pela resolução desacoplada do processo de integração numérica. Desta forma, modelos numéricos com elevado nível de discretização podem ser simulados computacionalmente sem as restrições de memória que o armazenamento de uma matriz de rigidez poderiam impor.

De maneira similar aos trabalhos de Zienkiewicz & Xie (1991) e Hulbert & Jang (1995), o Método Explícito Generalizado - α Adaptativo faz uso de uma solução local aproximada para estabelecer um mecanismo de adaptatividade. Trata-se portanto de um método adaptativo de integração numérica explícita resultado da fusão do Método Explícito Generalizado - α com os procedimentos de adaptatividade do Método Implícito Generalizado - α apresentado por Hulbert & Jang.

De uma maneira geral, o Método Explícito Generalizado - α Adaptativo pode ser encarado como um algoritmo que engloba três tarefas básicas: estimação de erro local aproximado, controle do incremento de tempo e integração numérica propriamente dita. A última tarefa é realizada de maneira idêntica à apresentada na Seção 2.2.1. Por se tratar de um algoritmo explícito, logo condicionalmente estável, deve-se assegurar que as alterações no incremento de tempo de integração ocorram dentro de um intervalo que tem como limite superior o incremento de tempo crítico Δt_{crit} e limite inferior um percentual deste último incremento. Silveira (2001) sugere a utilização de um valor em torno de 10% do incremento de tempo crítico.

Para a tarefa de estimação de erro utiliza-se a proposta apresentada por Hulbert & Jang. Através de uma série de Taylor determina-se uma expressão para $\boldsymbol{u}_{n+1}$ que leva em consideração o algoritmo de integração numérica utilizado. Sendo $\Delta\ddot{\boldsymbol{d}}_n = \ddot{\boldsymbol{d}}_{n+1} - \ddot{\boldsymbol{d}}_n$ a variação de aceleração observada na integração de um passo, e fazendo uso da expressão de $\boldsymbol{u}_{n+1}$, o erro local pode ser obtido com uso da equação:

$$\boldsymbol{e}_{n+1} = \Delta t_{n+1}^2 \left[\left(\Omega - \frac{1}{6(1-\alpha_f)} \right) \Delta\ddot{\boldsymbol{d}}_n + \left(\frac{1}{6(1-\alpha_f)} - \frac{1}{2} \right) \boldsymbol{w}_n \right], \tag{2.34}$$

onde $\boldsymbol{w}_n$ é determinado a partir da expressão:

$$\boldsymbol{w}_n = \left(1 - \frac{\alpha_f}{1-\alpha_f} \right)^{-1} \frac{\alpha_m - \alpha_f}{(1-\alpha_f)} \Delta\ddot{\boldsymbol{d}}_n. \tag{2.35}$$

Os parâmetros α_f, α_m e Ω são aqueles utilizados na formulação do Método Explícito Generalizado-α, apresentado na Seção 2.2.1.

Uma vez definido o indicador de erro, cabe à estratégia de adaptação definir a maneira com que o incremento de tempo de integração é manipulado. Deve-se estabelecer um mecanismo que seja robusto o suficiente para observar o comportamento do erro ao longo do tempo e utilizar esta informação de maneira a não afetar a estabilidade do algoritmo de integração numérica. Isso se faz necessário uma vez que a função que descreve o erro não tem um comportamento suave ao longo do tempo.

Uma prática comum para evitar as alterações constantes do incremento de tempo é a utilização de intervalos de tolerância de tempo que levam em consideração o período de vibração da estrutura T. Usualmente a relação $\Delta t/T$ é utilizada para tal feito. Desta forma, para problemas lineares é comum a utilização de dez incrementos de tempo por período da máxima freqüência de interesse na resposta. Para problemas de natureza fortemente não-linear o valor desta relação pode chegar até a duzentos intervalos.

Outra medida necessária nesta estratégia de adaptatividade é a normalização do erro calculado através da Equação (2.34) por um fator apropriado. Isso é necessário uma vez que um fator de escala do modelo deve ser utilizado como referência. Hulbert & Jang (1995) propuseram que este fator fosse a norma do vetor de deslocamentos do passo corrente $\|\boldsymbol{d}_n\|$. Para evitar que o erro normalizado adote valores fora da realidade quando o valor de $\|\boldsymbol{d}_n\|$ for muito pequeno, deve-se utilizar valores de normas de deslocamentos de passos anteriores: $\|\boldsymbol{d}_i\|$, com $i < n$. Desta forma, Hulbert & Jang sugerem a seguinte expressão para o erro

normalizado:

$$\eta_{n+1} = \frac{\|\boldsymbol{e}_{n+1}\|}{s_n}, \tag{2.36}$$

onde

$$s_n = max(\|\boldsymbol{d}_n\| \, , 0.9 s_{n-1}), \tag{2.37}$$

e $s_{n-1} = 0$ quando $n = 0$.

Para uma relação $\Delta t/T$, um referencial de tolerância admissível para o erro pode ser obtido da equação:

$$\eta_{adm} = C\left(\frac{\Delta t}{T}\right)^2 , C = (2\pi)^2 \left| c_1 \frac{1-\alpha_f}{1-\alpha_m} \right| , c_1 = \Omega - \frac{1-\alpha_m}{6(1-\alpha_f)}, \tag{2.38}$$

onde α_f, α_m e Ω correspondem aos parâmetros de integração utilizado pelo algoritmo MEG-α, apresentado na Seção 2.2.1.

A estratégia de adaptatividade baseada no erro local aproximado consiste portanto em manter o valor do erro calculado dentro de um intervalo de erros admissíveis. Este intervalo leva em conta o valor do erro admissível η_{adm} obtido da Equação (2.38) e os valores de erros normalizados obtidos para todos os instante de tempo a partir da Equação (2.36). A faixa de valores a ser utilizada no controle do incremento de tempo é dada portanto na forma:

$$\mu \eta_{adm} < \eta_{n+1} < \eta_{adm}, \tag{2.39}$$

onde μ é um fator que define o limite inferior admissível ($0 \leq \mu \leq 1$).

Desta forma, uma vez que seja verificado que $\eta_{n+1} > \eta_{adm}$, o incremento de tempo deve ser reduzido no intuito de trazer o valor do erro dos próximos passos para o intervalo da Equação (2.39). De maneira contrária, quando $\eta_{n+1} < \mu \eta_{adm}$, o incremento de tempo pode ser aumentado já que o erro calculado é menor que o limite inferior admissível. Isso no entanto deve ser realizado de maneira cautelosa para que a própria alteração do incremento de tempo não venha a aumentar o valor do erro de maneira significativa, o que acabaria por acarretar um resultado contrário ao esperado. Nos instantes em que o valor do erro estiver dentro do intervalo citado o incremento de tempo deve ser mantido constante.

Observe que em uma relação direta entre erro e incremento de tempo, os problemas de desestabilização de algoritmo já citados podem ocorrer, inviabilizando o procedimento adaptativo buscado. Faz-se então necessário um procedimento intermediário que torne esta

relação possível. Hulbert & Jang propuseram a utilização de um contador que caracteriza o comportamento da função erro frente à sua permanência no intervalo dado pela Equação (2.39). Esse contador é reinicializado toda vez que a referida equação for satisfeita. Seu valor é alterado quando o valor do erro não pertence à faixa de erros admissíveis e indica quantas vezes o erro esteve fora deste intervalo. Na proposta apresentada por Hulbert & Jang o mesmo contador é utilizado para o aumento ou redução do incremento de tempo (Δt). Uma vez que este indicador atinge um valor limite, a alteração em Δt é então realizada. Na prática o valor limite deste contador pode ser definido de maneira prescrita, no entanto sugere-se que o período de vibração da estrutura (T) seja utilizado como referência. Fazendo uso da relação $\Delta t/T$, comentada anteriormente, o valor limite para o contador é dado por:

$$cont = int\left(\frac{1}{\Delta t/T}\right). \tag{2.40}$$

A determinação do novo incremento de tempo Δt_{new} faz uso do valor do erro admissível bem como do valor do incremento de tempo anterior à mudança Δt_{old}. A equação que correlaciona estas variáveis é dada por:

$$\theta = \sqrt{\frac{\eta_{adm}}{\eta_{n+1}^{max}}}, \quad \text{e} \tag{2.41}$$

$$\Delta t_{new} = \theta \Delta t_{old}, \tag{2.42}$$

onde η_{n+1}^{max} é o erro máximo obtido após a atribuição do valor nulo ao contador.

Uma segunda situação possível é a redução no valor do incremento de tempo quando $\eta_{n+1} > \eta_{adm}$. Nesta situação, dois casos particulares devem ser tratados. O primeiro caso diz respeito à situação em que o incremento de tempo tenha sido aumentado no passo anterior. O novo incremento de tempo escolhido deve ser o valor do incremento de tempo anterior ao seu aumento. Caso o incremento de tempo não tenha sido aumentado no passo anterior, a metodologia é similar à utilizada para o aumento do incremento de tempo, no entanto a variável que relaciona os incrementos de tempo (θ) é agora dada por:

$$\theta = \sqrt{\frac{\eta_{adm}}{\eta_{n+1}}}. \tag{2.43}$$

Para este caso, Hulbert & Jang sugerem ainda que a solução encontrada no passo anterior seja descartada e uma nova solução com um novo incremento de tempo obtida. No trabalho

destes autores ainda são sugeridos valores para alguns dos parâmetros livres baseados em estudos e casos analisados e apresentados. Entre esses valores sugere-se a utilização de $\mu = 0{,}75$ e de uma relação $\Delta t/T$ fornecida pelo usuário.

Uma vez que o incremento de tempo deve permanecer constante durante a resposta estacionária para soluções periódicas (Bergan; Mollestad, 1985), o contador deve ser maior que o número de incrementos de tempo entre os valores máximos do erro local normalizado. Como o período entre os valores máximos do erro local normalizado é metade do período natural da vibração, sugere-se que o contador deva ser maior que $\Delta t/2T$ (Silveira, 2001).

2.3.4 Indicadores de Desempenho e Qualidade

Estabelecida uma estratégia de adaptatividade, fica evidente a necessidade de sua avaliação. Critérios tais como robustez e eficiência devem ser atendidos para que a utilização desta estratégia seja justificada. Alguns indicadores podem ser utilizados visando avaliações deste tipo. Eles devem prover informações que permitam avaliar uma resposta acerca de sua eficiência e qualidade.

Uma resposta é obtida de forma eficiente quando consome o mínimo de recursos computacionais para o nível de precisão desejado. Assim sendo, considerando uma integração temporal explícita, esta situação ocorreria em uma situação hipotética em que o incremento de tempo máximo que garante a estabilidade é utilizado no processo de integração numérica. Como isso nem sempre é possível, o termo eficiência pode ser utilizado de uma maneira relativa. Quanto mais próximo da situação hipotética uma resposta numérica pode ser obtida mais eficiente é a mesma.

De forma prática, a eficiência de um algoritmo computacional pode ser medida através da associação direta deste fator com a performance de sua implementação. Assim sendo, o número de operações de ponto flutuante (*flop*) pode ser utilizado como referência. Em computação científica é comum a utilização desta medida uma vez que esta mede a quantidade de tarefas que deve ser efetuada pelos núcleos de processamento. A velocidade com que esta tarefa é realizada no entanto depende da capacidade de processamento dos referidos núcleos. A quantidade de *flops*, ou seja, do número de operações de ponto flutuante instruídos por segundo, revela portanto o quão rápido uma tarefa computacional pode ser obtida.

Algoritmos de integração numérica adaptativa podem ser entendidos de forma simplificada como sendo extensões de algoritmos de integração convencionais, tais como os apresentados na Seção 2.2.1. Desta forma, por melhor que tenha sido sua implementação estes não conseguirão

integrar um passo de maneira mais rápida que um algoritmo onde a adaptação no tempo não é utilizada. A melhoria em desempenho para o algoritmo adaptativo só é observada uma vez que o tamanho do incremento de integração pode ser aumentado em tempo de simulação. Desta forma, um número menor de passos de integração é necessário para descrever o intervalo de tempo de análise e, quando o processo global é considerado, acaba-se observando tal melhoria de desempenho.

Considerando agora o aspecto qualidade de resposta, um algoritmo de integração ótimo seria aquele que apresentasse uma resposta numérica mais próxima possível da solução exata, ou seja, mais preciso. Como isso nem sempre é possível, uma medida de comparação é estabelecida a fim de que respostas numéricas sejam confrontadas quanto à sua precisão. Para problemas em que a solução analítica do problema, representada por $\boldsymbol{u}(t)$, é conhecida pode-se definir o valor de um resíduo de resposta, chamado de norma euclidiana. Este indicador permite quantificar a distância entre a solução numérica $\boldsymbol{d}(t)$ e a solução exata conforme a equação:

$$\|\boldsymbol{e}(t)\| = \|\boldsymbol{u}(t) - \boldsymbol{d}(t)\|. \tag{2.44}$$

No próximo capítulo são tratados aspectos relacionados com a formulação do estimador de refinamento alvo de estudo deste trabalho. Os fundamentos apresentados até então desenvolvidos neste capítulo que se encerra serão posteriormente utilizados quando a formulação do estimador de refinamento baseado em curvatura estiver sido apresentada.

3 *Adaptação Baseada no Indicador Geométrico Curvatura*

3.1 Conceitos Preliminares

Ao longo deste trabalho são utilizados conceitos de geometria diferencial que embasam toda a estratégia de adaptação temporal aqui desenvolvida. Uma breve descrição destes itens faz-se portanto necessária a fim de que se possa entender a maneira pela qual este ferramental matemático é aplicado ao mecanismo de adaptação no tempo.

A base da formulação é fundada no conceito de curvas parametrizadas. Neste ponto, é comum que ocorra a associação do termo curva a subconjuntos de $\mathbb{R}^2$ e $\mathbb{R}^3$ tais como retas, circunferências ou elipses. De maneira geral, uma curva pode ser entendida como um conjunto de pontos descritos com um único parâmetro. Assim sendo, curvas com imagem em $\mathbb{R}^2$ e $\mathbb{R}^3$ fazem parte de um espaço restrito de curvas de $\mathbb{R}^n$ que podem ser visualizados ou plotados em um sistema cartesiano. Ao conjunto de pontos mapeados por uma curva dá-se o nome de traço, desta forma, duas curvas podem ter o mesmo traço dependendo da parametrização utilizada, ou seja, da maneira como ela é descrita. Uma curva de $\mathbb{R}^n$ é portanto uma função:

$$\begin{aligned} \boldsymbol{\tau} : I \quad &\subseteq \quad \mathbb{R} \rightarrow \mathbb{R}^n \\ z \quad &\mapsto \quad \boldsymbol{\tau}(z) \end{aligned} \tag{3.1}$$

Sendo $\boldsymbol{\tau}$ uma curva parametrizada, o vetor $\boldsymbol{\tau}'(z_0)$, se existir, é dito vetor velocidade da curva $\boldsymbol{\tau}$ no instante z_0. Quando $\boldsymbol{\tau}'(z_0) \neq \mathbf{0}$, este vetor define a direção da tangente ao traço da curva no ponto $\boldsymbol{\tau}(z_0)$. Ao escalar $v(z_0) = \|\boldsymbol{\tau}'(z_0)\|$ é usual a denominação "velocidade escalar no instante z_0". Uma curva diz-se curva regular se e só se $\boldsymbol{\tau}$ é uma função vetorial de classe C^1 (com primeira derivada existente e contínua) e além disso $\boldsymbol{\tau}'(z) \neq \mathbf{0}, \forall z \in I$.

Uma forma usual de se trabalhar com o vetor tangente é na sua forma normalizada, logo, o vetor tangente unitário à trajetória de $\boldsymbol{\tau}$ no instante z é escrito como:

$$\boldsymbol{\Gamma}(z) = \frac{\boldsymbol{\tau}'(z)}{\|\boldsymbol{\tau}'(z)\|}, \text{ onde } \|\boldsymbol{\tau}'(z)\| \neq 0. \tag{3.2}$$

Um ponto $\boldsymbol{\tau}(z_0)$ tal que $\boldsymbol{\tau}'(z_0) = \mathbf{0}$ diz-se ponto singular. Nesse caso diz-se que a curva $\boldsymbol{\tau}$ tem uma singularidade em $z = z_0$.

Observe que até o instante não existe nenhuma relação explícita entre a variável independente z e o comprimento de arco descrito ao longo da curva parametrizada nesta variável. Duas parametrizações de curvas com o mesmo traço podem, para uma mesma variação em z, descrever comprimentos diferentes. Note no entanto que o comprimento da curva é uma propriedade global e que portanto independe da parametrização utilizada. Sendo z_0 um instante qualquer fixo, pode-se definir uma função comprimento de $\boldsymbol{\tau}$, na variável φ, da seguinte forma:

$$h(z) = \int_{z_0}^{z} \|\boldsymbol{\tau}'(\varphi)\| d\varphi. \tag{3.3}$$

Se imaginarmos que uma partícula descreve uma trajetória ao longo do traço da curva parametrizada nesta variável, então a função $s = h(z)$ representa o comprimento percorrido pela partícula entre os instantes z_0 e z. Observe ainda que:

$$\frac{ds}{dz} = h'(z) = \|\boldsymbol{\tau}'(z)\| = v(z). \tag{3.4}$$

De posse da função comprimento de $\boldsymbol{\tau}$, o indicador geométrico curvatura circular pode então ser definido como a magnitude com a qual o vetor tangente varia ao longo do comprimento de uma curva, Equação (3.5). Seu valor é portanto definido para curvas regulares de classe C^2, ou seja aquelas que possuem segunda derivada definida e contínua.

$$k(s) = \|\boldsymbol{\tau}''(s)\| \tag{3.5}$$

Aplicando a regra da derivação composta ao vetor tangente unitário e substituindo a Equação (3.4), tem-se

$$\boldsymbol{\Gamma}'(z) = \boldsymbol{\Gamma}'(s(z))s'(z), \tag{3.6}$$

e portanto

$$\begin{aligned} \|\boldsymbol{\Gamma}'(z)\| &= \|\boldsymbol{\Gamma}'(s(z))s'(z)\|, \\ \|\boldsymbol{\Gamma}'(z)\| &= \|\boldsymbol{\Gamma}'(s(z))\| \, \|s'(z)\|, \\ \|\boldsymbol{\Gamma}'(z)\| &= k(s(z))v(z), \end{aligned} \tag{3.7}$$

logo

$$k(z) = \frac{\|\mathbf{\Gamma}'(z)\|}{v(z)} = \frac{\|\mathbf{\Gamma}'(z)\|}{\|\boldsymbol{\tau}'(z)\|}. \tag{3.8}$$

Uma vez que o vetor tangente unitário é função de $\boldsymbol{\tau}$, sua derivada é dada por:

$$\mathbf{\Gamma}'(z) = \frac{\boldsymbol{\tau}''(z)\|\boldsymbol{\tau}'(z)\| - \|\boldsymbol{\tau}'(z)\|'\boldsymbol{\tau}'(z)}{\|\boldsymbol{\tau}'(z)\|^2}. \tag{3.9}$$

Substituindo a Equação (3.9) em (3.8), tem-se:

$$k(z) = \left\| \frac{\boldsymbol{\tau}''(z)\|\boldsymbol{\tau}'(z)\| - \|\boldsymbol{\tau}'(z)\|'\boldsymbol{\tau}'(z)}{\|\boldsymbol{\tau}'(z)\|^3} \right\|. \tag{3.10}$$

É possível ainda provar que a Equação (3.10) pode ser escrita na forma apresentada por Kreyszig (2006), dada por:

$$k(z) = \frac{\sqrt{(\boldsymbol{\tau}'\cdot\boldsymbol{\tau}')(\boldsymbol{\tau}''\cdot\boldsymbol{\tau}'') - (\boldsymbol{\tau}'\cdot\boldsymbol{\tau}'')^2}}{(\boldsymbol{\tau}'\cdot\boldsymbol{\tau}')^{3/2}}. \tag{3.11}$$

A vantagem desta forma de apresentação do indicador curvatura está relacionada a aspectos computacionais. Os produtos internos necessários para o cálculo da curvatura podem ser feitos de forma simultânea, em um único "*loop*". A dedução da Equação (3.11) é apresentada no apêndice deste trabalho.

A Equação (3.10) apresenta uma forma geral para a curvatura de uma curva qualquer parametrizada na variável z. Observe que não necessariamente existe uma relação desta variável com o comprimento descrito ao longo da curva. Sendo $\boldsymbol{\tau}$ uma curva parametrizada de classe C^2, a função $k(z)$ indica um valor positivo que informa quão curvo é o traçado de $\boldsymbol{\tau}$. Tal fato pode ser observado se considerarmos, por exemplo, a parametrização de uma função polinomial do terceiro grau. A Figura 3.1 ilustra o comportamento da função $k(z)$ para a parametrização $\boldsymbol{\tau}(z) = (z\ , 0.5z^3 + 0.5z^2 - 10z)$.

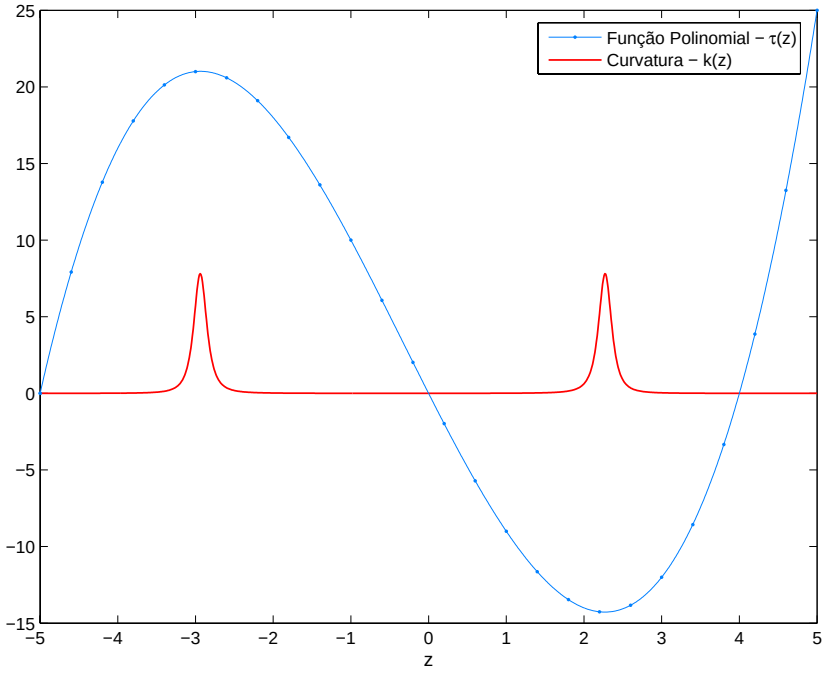


Figura 3.1: Curvatura de uma função polinomial do 3º grau.

Note que o valor da curvatura é praticamente nulo nos instantes em que o traçado da função polinomial se assemelha ao de uma reta. Para os demais instantes, o valor de $k(z)$ cresce ou decresce de acordo com a tendência de curvatura da função. Essa informação, apesar de simples, é bastante valiosa e é utilizada de forma intuitiva em nosso cotidiano.

Imagine por exemplo que se deseje fazer o esboço de um gráfico de uma função qualquer, a exemplo da apresentada anteriormente. O traçado deste gráfico será tão bem desenhado quanto maior for o detalhamento nas regiões onde a curvatura da função for maior. Nas regiões onde a curvatura é menor o traçado da curva tende ao de uma reta, logo um menor detalhamento é exigido em seu esboço. Se pensarmos no exemplo de um veículo que trafega em uma rodovia, supostamente livre de pedestres, veículos e em perfeitas condições de tráfego, a atenção do motorista ao volante será maior justamente nos instantes em que a curvatura do eixo da pista for mais elevada. O motorista então tomará algumas medidas a fim de que o veículo não tome o rumo que não era previsto.

3.2 Curvatura do Histórico de Deslocamentos

Em um processo de integração numérica temporal acontece algo semelhante aos exemplos do traçado do gráfico e do tráfego em uma rodovia. Desta vez o nível de atenção que se dá à resposta está associado ao incremento de tempo de integração que é utilizado (Δt). Quanto

menor o seu valor, maior a qualidade da resposta obtida. Quando o contrário acontece, a qualidade da resposta diminui. Uma estratégia de adaptação no tempo é aquela que consegue enxergar o comportamento de um modelo que está sendo integrado de forma a sugerir mudanças para o incremento de tempo. Estas mudanças buscam uma forma de otimizar a relação existente entre a qualidade da integração e o tempo gasto para a sua execução. Conforme apresentado anteriormente, estas estratégias utilizam estimadores de refinamento para que isso possa ser feito. Uma alternativa para o processo de estimativa de refinamento seria utilizar o valor da curvatura do histórico de deslocamentos.

De maneira particular, pode-se parametrizar uma curva $\boldsymbol{\tau}$ de maneira que esta represente um histórico de deslocamentos de um corpo qualquer na forma:

$$\boldsymbol{\tau}(t) = \begin{Bmatrix} t \\ u(t) \end{Bmatrix}. \tag{3.12}$$

Neste caso, a variável de parametrização t representa a grandeza tempo e a função $u(t)$ descreve o campo de deslocamentos deste corpo. Considerando uma forma discreta para a caracterização do campo de deslocamentos, a Equação (3.12) assume a forma particular:

$$\boldsymbol{\tau}(t) = \begin{Bmatrix} t \\ \boldsymbol{d} \end{Bmatrix}. \tag{3.13}$$

Diferenciando a função $\boldsymbol{\tau}$, tem-se:

$$\boldsymbol{\tau}'(t) = \begin{Bmatrix} 1 \\ \dot{\boldsymbol{d}} \end{Bmatrix}, \ \boldsymbol{\tau}''(t) = \begin{Bmatrix} 0 \\ \ddot{\boldsymbol{d}} \end{Bmatrix}, \tag{3.14}$$

logo,

$$\|\boldsymbol{\tau}'(t)\| = \sqrt{1 + \dot{\boldsymbol{d}} \cdot \dot{\boldsymbol{d}}}, \tag{3.15}$$

$$\|\boldsymbol{\tau}'(t)\|' = \frac{\ddot{\boldsymbol{d}} \cdot \dot{\boldsymbol{d}}}{\|\boldsymbol{\tau}'(t)\|}. \tag{3.16}$$

Substituindo a Equação (3.16) na equação de curvatura (3.10), temos a forma discreta da curvatura:

$$k(t) = \left\| \frac{\boldsymbol{\tau}''(t)\|\boldsymbol{\tau}'(t)\|^2 - (\ddot{\boldsymbol{d}} \cdot \dot{\boldsymbol{d}})\boldsymbol{\tau}'(t)}{\|\boldsymbol{\tau}'(t)\|^4} \right\|. \tag{3.17}$$

Sendo $\dot{\boldsymbol{d}}$ o vetor velocidades, e $\ddot{\boldsymbol{d}}$ o vetor acelerações, a Equação (3.17) pode ainda ser escrita na forma:

$$k(t) = \frac{\left\| (1+\dot{\boldsymbol{d}}\cdot\dot{\boldsymbol{d}})^2 \begin{Bmatrix} 0 \\ \ddot{\boldsymbol{d}} \end{Bmatrix} - (\dot{\boldsymbol{d}}\cdot\ddot{\boldsymbol{d}}) \begin{Bmatrix} 1 \\ \dot{\boldsymbol{d}} \end{Bmatrix} \right\|}{(1+\dot{\boldsymbol{d}}\cdot\dot{\boldsymbol{d}})^2}, \tag{3.18}$$

ou, usando a forma da equação de curvatura apresentada por Kreyszig, como:

$$k(t) = \frac{\sqrt{(\dot{\boldsymbol{d}}\cdot\dot{\boldsymbol{d}})(\ddot{\boldsymbol{d}}\cdot\ddot{\boldsymbol{d}}) - (\dot{\boldsymbol{d}}\cdot\ddot{\boldsymbol{d}})^2}}{(\dot{\boldsymbol{d}}\cdot\dot{\boldsymbol{d}})^{3/2}} \tag{3.19}$$

As equações (3.18) e (3.19) são idênticas e revelam portanto a intensidade da curvatura do histórico de deslocamentos de um corpo qualquer. Esta informação pode ser utilizada de maneira a controlar o valor do incremento de tempo a ser utilizado em um processo de integração direta. Uma vez que o valor da curvatura é sempre positivo, uma forma de se estabelecer uma relação entre estas duas variáveis é fazendo uso de uma função exponencial:

$$\Delta t = \Delta t_{max} e^{-c\ k(t)} \tag{3.20}$$

Sendo c uma constante positiva, para qualquer valor de curvatura obtém-se um correspondente incremento de integração variando de um valor máximo definido a zero, no caso limite onde a curvatura é infinita.

A Equação (3.20) pode ser incorporada ao algoritmo responsável pela tarefa de integração no tempo. Para todo passo avaliado pelo algoritmo, obtém-se um valor de curvatura com base na cinemática do modelo simulado e este valor é correlacionado com um incremento de tempo. Isso faz com que o algoritmo perceba o comportamento do modelo e proponha alterações no intuito de tornar o processo de integração otimizado quanto à qualidade de resposta e desempenho.

De qualquer maneira o valor do incremento obtido nesta correlação não deve ser utilizado na solução dos próximos passos sem que antes algumas considerações sejam feitas. Conforme comentado anteriormente, a alteração do valor do incremento deve ocorrer de forma controlada. O simples fato de fazer sua correlação com o valor da curvatura não vai garantir que esse controle aconteça. Visando a solução deste problema são propostos alguns mecanismos que fazem um processamento dos valores de curvatura antes do seu uso. A partir deste processamento são geradas seqüências de curvaturas ditas “regularizadas”. Uma vez que

estas seqüências tenham sido geradas, as mesmas podem ser utilizadas para a alteração do incremento de tempo. Algumas técnicas de regularização de curvatura são apresentadas na próxima seção.

3.3 Regularização da Curvatura

Em alguns casos, a curva *k(t)* pode apresentar um comportamento oscilatório indesejável que pode perturbar a estabilidade do algoritmo de integração empregado. Com o objetivo de tratar tal oscilação, sugere-se o emprego de uma metodologia de regularização da curva que descreve a curvatura do histórico $k(t)$. Diversas técnicas podem ser utilizadas com tal finalidade. Deve-se atentar, no entanto, ao fato de que a complexidade destas técnicas está relacionada tanto com a robustez quanto com o desempenho de suas implementações computacionais. Uma estratégia de regularização ótima é aquela em que ambos os fatores são bem favorecidos, e que portanto a tarefa de executar bem a regularização pode ser realizada com um alto desempenho computacional. A seguir são apresentadas algumas metodologias de regularização de curvatura.

3.3.1 Curva *k versus* Δt Regularizada

Consiste na utilização de uma função de arredondamento (*floor*) com o objetivo de evitar a troca do incremento de tempo para variações de curvatura dentro de intervalos de tamanho *dk*.

$$\Delta t = \Delta t_{max} e^{-c\ dk\ floor(k(t)/dk)} \tag{3.21}$$

Observe que a implementação computacional desta técnica é bem simples e ela atua diretamente na função que relaciona curvatura com incremento de integração. A Figura 3.2 ilustra como esta estratégia de regularização é aplicada.

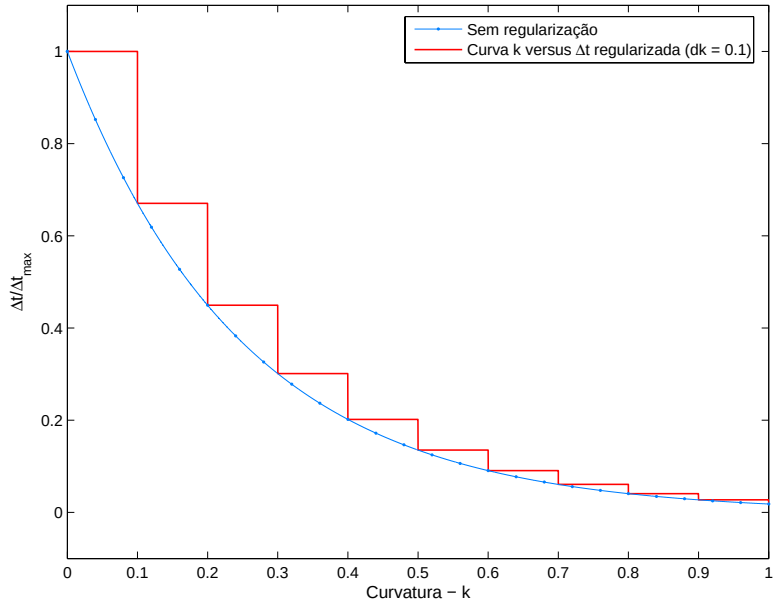


Figura 3.2: Curva *k versus* Δt regularizada.

São gerados patamares de incrementos de tempo para intervalos correspondentes de curvatura. Desta forma, variações em $k(t)$ que ocorrem dentro destes intervalos não acarretam mudanças para Δt. A utilização desta estratégia de regularização é indicada para os casos onde as mudanças de curvatura ocorrem de maneira menos constante ao longo de tempo. Quando essa condição não é atendida e, portanto, as oscilações de curvatura são de grandes amplitudes ou bastante freqüentes, o uso desta técnica de regularização continua a resultar em mudanças constantes para Δt, sendo desta forma não eficaz.

3.3.2 Regularização por Máximo Valor em Intervalos

Esta estratégia utiliza um procedimento de regularização da curvatura que leva em consideração o comportamento de $k(t)$ em intervalos regulares de tempo. Durante o decorrer de toda a integração numérica e para todos os instantes de tempo, investiga-se o valor da curvatura e o compara com o máximo valor de curvatura ocorrido no intervalo correspondente àquele instante. O maior valor dentre estes é então utilizado na consulta do incremento de tempo a ser utilizado na integração do passo corrente de acordo com a Equação (3.20).

Observe que, diferentemente da estratégia de regularização anteriormente apresentada, a equação que relaciona curvatura com incremento de tempo permanece inalterada. O que muda portanto entre esta metodologia e o procedimento padrão (sem regularização) é a forma com que o parâmetro k é fornecido à função de consulta de Δt. Sugere-se que o tamanho dos intervalos de tempo utilizados no processo de regularização seja função do valor do incremento de integração

crítico (Δt_{crit}), valor a princípio considerado constante ao longo do tempo.

A Figura 3.3 apresenta o escopo do algoritmo que implementa esta técnica de regularização em um processo iterativo que ocorre ao longo do tempo e conforme ilustrado na Figura 3.4.

Inicialização $n = 0$; $k_n = 0$

Processo Iterativo
$aux = floor(t/(m\Delta t_{crit}))$
$Se(aux \neq n)$
$n = aux$
$k_n = \alpha k_{n-1} + (1-\alpha) maxk(t_n,t)$
Senão
Se $(maxk(t_n,t) > k_n)$
$k_n = maxk(t_n,t)$
Fim
Fim
$\Delta t_n = \Delta t_{max} e^{-c\ k_n}$

onde,
$floor \Rightarrow$ Quociente da divisão
$m \Rightarrow$ Constante inteira positiva
$\Delta t_{crit} \Rightarrow$ Incremento de integração crítico
$maxk(t_n,t) \Rightarrow$ Máximo valor de $k(t)$ entre os instantes t_n e t
$\alpha \Rightarrow$ Coeficiente de ponderação $(0 \leqslant \alpha \leqslant 1)$

Figura 3.3: Algoritmo de regularização por máximo valor em intervalos.

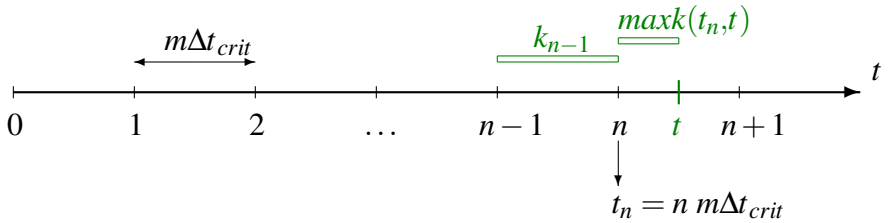


Figura 3.4: Linha temporal - Algoritmo de regularização por máximo valor em intervalos.

Note que o algoritmo apresentado faz recorrência à informação de curvatura máxima observada no intervalo de tempo que imediatamente precede o corrente (k_{n-1}). Trata-se de um artifício para minimizar o número de trocas para o incremento de tempo nos primeiros instantes do intervalo de regularização ($m\Delta t_{crit}$). Isso se faz necessário uma vez que o valor máximo de curvatura entre os instantes t_n e t , $maxk(t_n,t)$, tem pouca representatividade do comportamento da curva $k(t)$ quando o instante t é próximo de t_n. Uma vez que esta baixa representatividade

pode acarretar mudanças constantes no intervalo de integração neste período, seu tratamento fica portanto justificado.

A Figura 3.5 ilustra a aplicação desta técnica de regularização para o caso em que a curvatura é descrita de maneira hipotética a partir de uma função oscilatória - $k(t) = |(t+4)(t-4)(t-8)||cos(20t)|$.

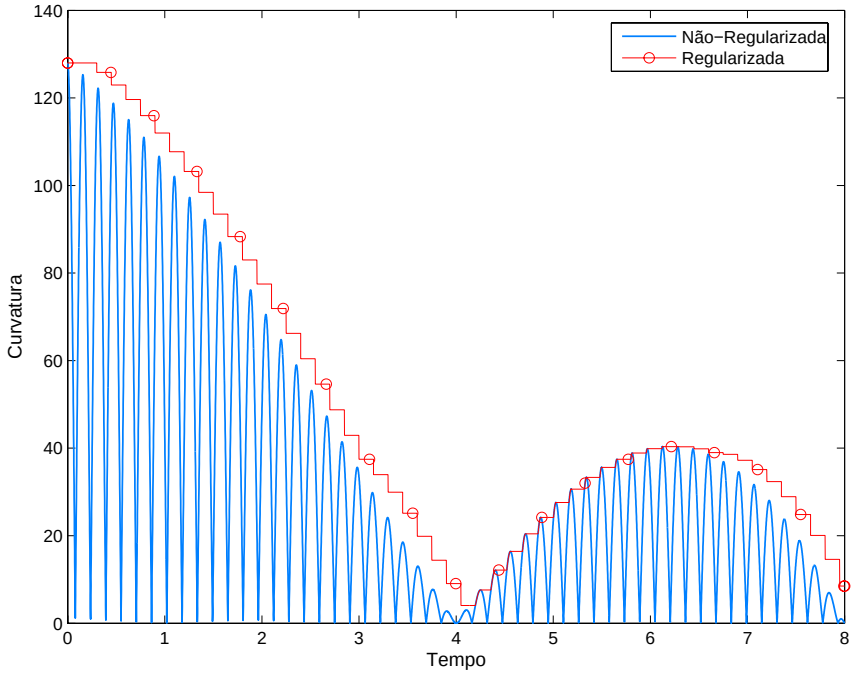


Figura 3.5: Regularização de curvatura por máximo valor em intervalos.

Observe que se o incremento de tempo fosse ajustado de maneira direta, a partir da informação de curvatura $k(t)$, as alterações da variável que regulam o tamanho do passo de integração herdariam um comportamento oscilatório que acabariam por desestabilizar o algoritmo de integração numérica. O uso desta técnica de regularização, conforme pode ser observado, permite que a informação de curvatura seja regularizada antes que ela venha a ser utilizada no controle do incremento de tempo. Desta forma as alterações ocorrem de maneira controlada e gradativa. A técnica assegura ainda que os valores de curvatura regularizados sejam sempre maiores que os não-regularizados. Desta forma, garante-se que em nenhum instante um valor de incremento de tempo seja utilizado em desacordo com o valor real de curvatura apresentado ao longo do histórico de integração.

4 *Implementação Computacional*

Neste capítulo são tratados alguns detalhes referentes à implementação computacional da estratégia de adaptação apresentada neste trabalho. Com o intuito de facilitar o entendimento desta etapa, sempre que possível são utilizadas ilustrações a respeito dos algoritmos utilizados, da arquitetura do sistema desenvolvido e de exemplos de trechos de códigos de aplicações que fazem uso deste recurso.

A idéia básica da implementação é desenvolver uma biblioteca extensível que incorpore todas as funcionalidades providas pela técnica de adaptação baseada no indicador geométrico curvatura. Desta forma, o uso deste mecanismo de adaptação em aplicações existentes dispende bem menos esforço de programação.

Uma vez que a técnica de adaptação pode ser utilizada em diversas aplicações que utilizam resolvedores baseados em soluções incrementais (desde que possuam formulações compatíveis), a arquitetura da biblioteca em questão foi concebida de maneira a permitir tal usabilidade. Para tanto, os conceitos de programação orientada a objetos são utilizados. O uso deste tipo de programação em geral oferece alguns benefícios relacionados com a legibilidade, facilidade de extensão, dentre outros. A linguagem de programação C++ (Stroustrup, 2000) contempla todos estes conceitos e foi utilizada para o desenvolvimento da biblioteca proposta.

4.1 Programação Orientada a Objetos

A denominação “orientação a objetos” é utilizada para especificar estruturas abstratas onde dados e operações são agrupados em um objeto. Os dados que caracterizam o objeto são chamados de “atributos” e as operações que manipulam estes dados recebem o nome de “métodos”. Este agrupamento ocorre de maneira a separar aspectos internos e externos a um objeto, sendo amplamente utilizado para impedir o acesso direto ao estado deste, ou seja, aos seus atributos. Desta forma, os dados de um objeto ficam encapsulados em um espaço de memória privado e de acesso controlável, impedindo que sua manipulação ocorra de maneira desordenada. O tipo de dado abstrato que concebe estes conceitos é chamado de classe.

A classe é a entidade básica a ser tratada em um programa orientado a objetos, cabendo portanto ao programador sua construção. Isso deve ser feito de maneira que exista uma boa relação entre o objeto que se deseja caracterizar e a formalização desta classe, através da definição dos atributos e métodos. Quando esta relação ocorre de maneira apropriada a composição do programa a partir da comunicação entres as classes ocorre de maneira otimizada. Deve-se buscar sempre que os problemas tratados nas classes sejam bem resolvidos dentro destas.

Um mecanismo bastante útil em orientação a objetos é o de generalização, também chamado de "herança". Este artifício permite que uma classe seja construída a partir de outra, herdando as funcionalidades da classe tomada como referência. Estas classes passam a compartilhar códigos e o processo de programação torna-se mais fácil. Uma vez que os atributos e métodos sejam compartilhados, os mesmos tornam-se disponíveis nas classes derivadas. Isso evita que rotinas sejam re-escritas, exigindo portanto um menor esforço de programação. Além disso, o código escrito é mais bem organizado, já que segue um processo de construção com uma lógica bem definida. Dependendo da forma com que as classes estejam associadas, as classes envolvidas no mecanismo de herança recebem denominações próprias. A classe base usada para a extensão recebe o nome de super-classe e a classe derivada é denominada sub-classe.

Para que este compartilhamento de métodos e atributos possa ocorrer, as classes devem definir a forma de acesso a estes recursos. São definidos três padrões de acesso: privado, protegido e público. Estes padrões são definidos para cada um dos métodos ou atributos que compõem uma classe no momento em que esta é construída. Quando um método ou atributo é definido como privado o acesso a este recurso só pode ocorrer dentro do contexto da classe que os contém. No padrão de acesso protegido estes recursos são compartilhados com as classes derivadas, permitindo o acesso aos mesmos. O padrão de acesso público estabelece um compartilhamento que vai além das relações de herança. Desta forma o acesso a um dado recurso de uma classe pode acontecer a partir de outra classe sem a necessidade do processo de generalização ou herança. A utilização de padrões de acesso assegura que a manipulação dos recursos da classe ocorre de maneira ordenada, impedindo que o resultado previsto em uma implementação seja afetado por acessos externos à classe e que ocorram de forma indevida.

Uma forma usual de representar uma classe é através de uma modelagem gráfica. Esta forma de representação permite que o resultado de uma implementação possa ser visualizado em diagramas de fácil entendimento. Uma linguagem bastante difundida no meio de desenvolvimento de *software* é a UML (*Unified Modeling Language*).

Segundo Rumbaugh *et al.* (1999), a UML é uma linguagem de modelagem visual para a especificação, visualização, construção e documentação dos artefatos de sistemas de *software*. Ela é usada para entender, projetar, configurar, manter e controlar informações a respeito destes sistemas. Ao longo deste trabalho são apresentadas algumas notações, baseadas na linguagem UML, que foram utilizadas na representação da arquitetura da biblioteca de adaptação a ser apresentada. A Figura 4.1, por exemplo, apresenta o diagrama de representação de uma classe qualquer.

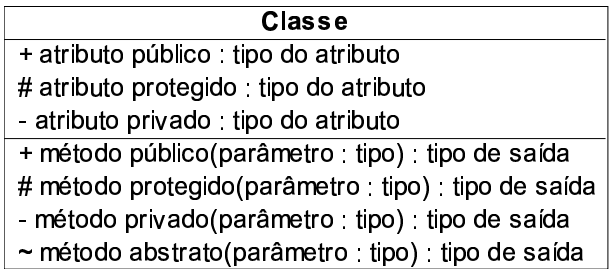
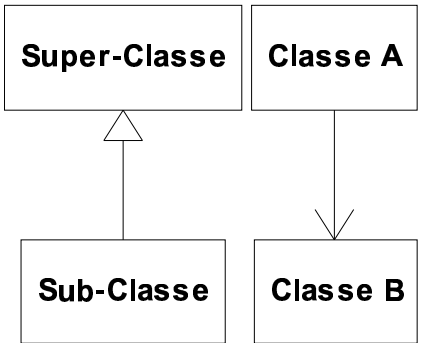


Figura 4.1: Diagrama de classe - métodos e atributos (*UML*).

Note que alguns dados relevantes da classe são apresentados nesta forma de representação. São destacadas informações tais como: o nome da classe, seus principais atributos e métodos, bem como a visibilidade destes últimos recursos. Observe que uma nova modalidade de método, a abstrata, é introduzida. Um método abstrato é aquele que não necessariamente é implementado em uma classe. Ele permite que esta implementação ocorra de maneira específica nas classes derivadas. Isso faz com que um mesmo método possa apresentar várias formas de acordo com seu contexto. Esta abstração recebe o nome de polimorfismo. Na linguagem de programação C++ os métodos abstratos são chamados de métodos virtuais e recebem denominações próprias quanto à sua forma de construção, podendo ser virtual puro ou não. Um método virtual puro é aquele que de fato não possue implementação, apenas a assinatura do método a ser definido nas classes derivadas. Quando um método virtual apresenta implementação ele deixa de ser puro e o código definido passa a ser utilizado pelas classes derivadas, caso a mesma não defina um método com a mesma assinatura. Nas linguagens de programação C# e Java, uma classe onde todos os métodos são abstratos recebe o nome de classe de interface (Fowler, 2003).

Embora o diagrama de classe forneça informações valiosas da sua estrutura, o mesmo não apresenta de forma visual a maneira com a qual esta classe relaciona-se com as demais na composição de um programa. Uma outra maneira de denotar as propriedades de uma classe é a partir da utilização de setas direcionais. A linguagem UML apresenta formas de representação para alguns tipos de relações entre classes, dentre elas as de generalização (Figura 4.2(a)) e de associação (Figura 4.2(b)).



(a) Generalização (b) Associação

Figura 4.2: Representação gráfica das relações entre classes (*UML*).

Na relação de generalização, a seta direcional indica que uma classe (sub-classe), como o próprio nome sugere, generaliza outra (super-classe). No caso da relação de associação a seta direcional indica que uma classe utiliza objeto(s) de outra. A representação da Figura 4.2(b) indica portanto que a classe "B" está sendo utilizada pela classe "A", ou seja, a classe "A" possue objeto(s) da classe "B".

A representação gráfica das relações entre classes utilizadas na estruturação de um programa fornece informações úteis aos seus desenvolvedores. A partir desta, tem-se a noção exata de onde está localizado um dado a ser manipulado por uma tarefa. Isso acaba conduzindo ao desenvolvimento de rotinas mais eficientes e precisas.

Na próxima seção são tratados aspectos acerca da implementação computacional da técnica de adaptação baseada no indicador curvatura previamente apresentada.

4.2 Arquitetura da Biblioteca

O projeto de uma biblioteca de adaptação deve atender a algumas peculiaridades relacionadas à sua metodologia de solução. De uma forma geral, esta biblioteca deve ser concebida de maneira a generalizar uma estratégia de solução não-adaptativa, ou seja, aquelas que fazem uso de incrementos com valores fixos durante todo o processo de solução. Desta forma, valores de incrementos de solução não necessariamente constantes podem ser utilizados em uma análise. A solução não-adaptativa passa a ser portanto um caso particular de análise que a biblioteca pode instanciar. O desafio agora passa a ser então a maneira como isso deve ser feito. Deve-se ter em mente que uma biblioteca é um recurso computacional que vem a complementar uma solução existente, jamais alterá-la. Assim sendo, uma biblioteca de adaptação não pode modificar as maneiras pelas quais as soluções não-adaptativas são obtidas. De que forma portanto isso pode ser feito?

Uma solução consiste em criar uma interface entre estes dois tipos de solução. Esta interface seria responsável pela comunicação entre dois resolvedores. O primeiro resolvedor incorpora uma técnica de solução iterativa onde a adaptação não é levada em conta e o segundo implementa um mecanismo puramente de adaptação. A partir da interface, o mecanismo de adaptação poderia consultar informações relevantes para o seu funcionamento e sugerir alterações para o valor do incremento de tempo utilizado pelo resolvedor não-adaptativo. Esta interface atuaria portanto no sentido de alterar a forma que o resolvedor não-adaptativo se comporta. Embora não exista alteração no escopo da rotina deste último o mesmo passa a se comportar de forma dinâmica e adaptativa em virtude das trocas de informações providas pela interface. A Figura 4.3 ilustra a interface de comunicação anteriormente proposta.

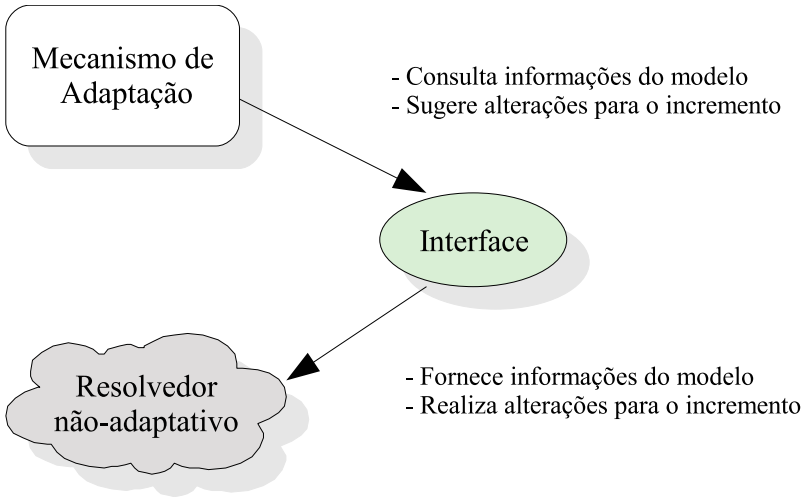


Figura 4.3: Interface para o mecanismo de adaptação.

Suponha que uma aplicação utilize um método de integração direta, a exemplo do método das diferenças centrais, para avaliar a evolução dos deslocamentos de um certo modelo. O resolvedor não-adaptativo corresponderia à seqüência de passos que atualizam as informações de deslocamentos, velocidades e acelerações do modelo, que para o caso do método citado corresponde ao algoritmo ilustrado na Figura 2.2. Este resolvedor faz uso de um incremento de integração de tempo (Δt) que, para um procedimento de integração não adaptativo, é constante durante todo o processo de simulação. Uma forma de fazer este resolvedor se comportar de forma adaptativa seria alterando o escopo de programação do algoritmo de maneira a levar em consideração uma técnica de adaptatividade qualquer. Isso no entanto exigiria um esforço de programação que teria que ser repetido caso outro método de integração numérica fosse utilizado. Estes esforços de reprogramação são dispensados com o uso da interface ilustrada na Figura 4.3, uma vez que as tarefas de resolução e adaptação ocorrem de maneira independente.

Para que isto ocorra é necessário que se estabeleça um padrão de comunicação entre o

resolvedor não-adaptativo e o mecanismo de adaptação. Uma vez que isso tenha sido feito, as tarefas de adaptação e de integração são realizadas de maneira desacoplada e todas as suas peculiaridades são atendidas. Os objetos responsáveis por estas tarefas ficam portanto encapsulados e trocando mensagens através da interface se pensarmos em orientação a objetos. Para o caso particular de adaptação no tempo baseada na curvatura do histórico de deslocamentos, as informações em trânsito na interface são basicamente os vetores de velocidades e acelerações, bem como os valores de incrementos de tempo. A estrutura de classes da biblioteca de adaptação deve então considerar apenas as tarefas que estão relacionadas com uma estratégia de adaptatividade. Supõe-se que a tarefa de integrar um passo para um dado incremento esteja bem resolvida no escopo da aplicação que faz uso desta biblioteca.

De uma forma geral, a metodologia de adaptação baseada em curvatura é composta por dois procedimentos básicos: determinar o valor da curvatura em um dado passo de solução e sugerir um novo incremento a ser utilizada no próximo passo. Entretanto, tarefas auxiliares são necessárias para que isso ocorra, dentre elas pode-se destacar o procedimento de regularização de curvatura. Observe ainda que a tarefa de estimar o valor da curvatura é algo que depende do tipo de problema que se esteja analisando. Conforme visto no Capítulo 3, o uso do indicador curvatura pode ser aplicado no estudo do histórico de deslocamentos de um corpo ou em qualquer outro tipo de problema, desde que sua expressão seja deduzida de maneira semelhante. Tomando como base estas premissas, uma organização de classes é desenvolvida para a biblioteca (ver Figura 4.4).

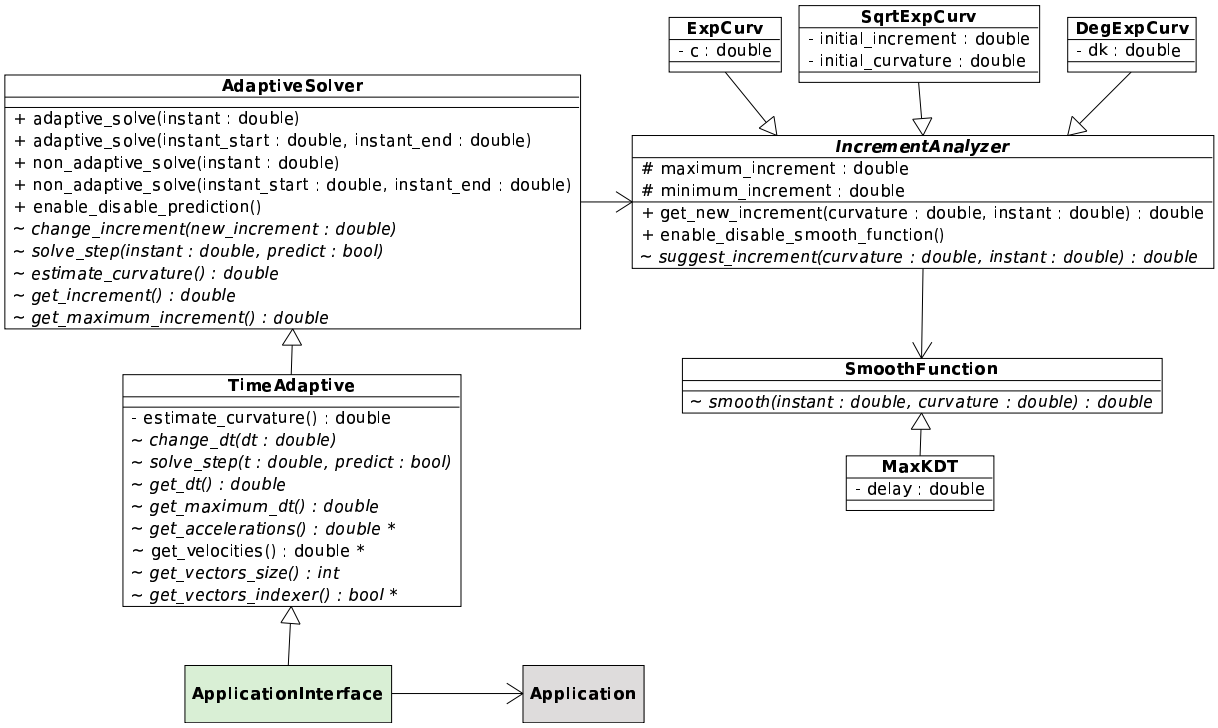


Figura 4.4: Diagrama de classes da biblioteca.

O núcleo da biblioteca se concentra na classe *AdaptiveSolver*. Desta classe nascem as principais tarefas a serem realizadas, assim como dela surgem as derivações que caracterizam o tipo de tratamento dado à curvatura. Seus métodos invocam os algoritmos disponíveis de resolução iterativa e, em virtude disso, um objeto desta classe deve ser construído a fim de que o sistema funcione. Embora esta classe possua implementação de métodos, a mesma não pode existir de maneira isolada. Isso acontece uma vez que alguns métodos desta classe são abstratos. Isso se faz necessário uma vez que o tratamento que se dá para o cálculo da curvatura não pode ser feito de uma maneira específica nesta classe. Além disso, para que as sugestões de alterações de incremento possam ser realizadas é preciso que exista a comunicação com uma interface, conforme comentado anteriormente. Assim sendo, relações de generalização se fazem necessárias para que se possa definir primeiramente que tipo de tratamento é dado à curvatura e, logo em seguida, a maneira pela qual esta classe se comunica com a aplicação.

Na primeira generalização o cálculo da curvatura assume um procedimento específico de obtenção, correspondente à obtenção do valor deste indicador geométrico para o caso onde o histórico de deslocamentos é alvo de investigação. A classe *TimeAdaptive* é portanto construída tomando como base a classe *AdaptiveSolver*, e novos métodos abstratos são estabelecidos. Isso se faz necessário uma vez que, para o cálculo da curvatura do histórico de deslocamentos, é necessário o conhecimento dos vetores de velocidades e acelerações nos graus de liberdade do modelo estudado (ver Seção 3.2). Observe na Figura 4.4 que a classe *TimeAdaptive*,

além de estabelecer novos métodos abstratos, assume uma nova denominação para o termo "incremento", que passa a ser tratado de forma específica como "dt" (em referência ao incremento de tempo de integração Δt).

A implementação dos métodos abstratos herdados da classe *AdaptiveSolver* e redefinidos na classe *TimeAdaptive* é feita na classe de interface da biblioteca com a aplicação. Esta classe, denominada *ApplicationInterface* apenas para fins ilustrativos, não é implementada no escopo da biblioteca. Ela é composta por uma série de métodos a serem implementados para que haja um acoplamento da biblioteca de adaptação com o resolvedor não-adaptativo provido pela aplicação. Na prática os métodos a serem implementados nesta classe de interface podem ser imaginados como perguntas em um questionário que devem ser respondidas pela aplicação. Se tomarmos o método *change_dt* como exemplo, a implementação deste deve conter uma chamada de funções contidas no escopo da aplicação que alterem o valor do incremento de tempo de integração Δt. Uma vez que o questionário é respondido adequadamente pela aplicação esta imediatamente incorpora o mecanismo de adaptação incluso na biblioteca.

Note que até o instante foram discutidos aspectos relacionados com as classes que especificam a classe base *AdaptiveSolver*. No entanto, outras classes estão envolvidas diretamente com esta última e são necessárias para que a metodologia de adaptação apresentada no Capítulo 3 seja implantada de fato na biblioteca. A classe *IncrementAnalyzer*, por exemplo, faz a correlação entre o valor da curvatura calculado em um certo passo e o valor de incremento sugerido para a sua solução. Uma vez que este procedimento pode ser realizado de diversas formas, esta classe concebe um método abstrato (*suggest_increment*) que permite que cada forma específica seja tratada em uma classe derivada.

No diagrama de classes da Figura 4.4 são apresentadas três classes derivadas responsáveis pela tarefa de correlação entre a curvatura e o incremento. A classe *ExpCurv* (*Exponential Curve*), por exemplo, utiliza uma forma exponencial para a função que correlaciona as duas informações citadas, Equação (3.20). Na classe *DegExpCurv* (*Degrees in a Exponential Curve*) a forma exponencial é associada a uma função que cria patamares para evitar mudanças constantes no valor do incremento (Equação (3.21)). E por fim, na classe *SqrtExpCurv* (*Square root and Exponential Curve*), a curva que relaciona a curvatura com o incremento é um função definida por partes, onde um trecho desta apresenta a forma de uma função raiz quadrada e o outro a de uma função exponencial.

4.3 Algoritmos de Solução Iterativa

Nesta seção são tratados aspectos relacionados aos principais algoritmos de solução iterativa presentes na biblioteca de adaptação. Observe que até o instante apenas aspectos macroscópicos foram discutidos quando a arquitetura da biblioteca foi apresentada. Embora as relações entre classes tenham sido mostradas, estas não revelam a maneira ou seqüência pela qual as tarefas são realizadas com o objetivo de tornar adaptativo um processo de integração.

São apresentados os três principais algoritmos responsáveis pela tarefa de comandar e avaliar o procedimento de integração iterativo presente na aplicação que faz uso da biblioteca. Com o intuito de facilitar o entendimento das rotinas computacionais envolvidas, sempre que possível são utilizadas notações gráficas baseadas em UML, a exemplo de diagramas de seqüência e de atividade.

4.3.1 Estratégia Não-Adaptativa

Conforme comentado anteriormente, uma biblioteca de adaptação deve ser construída no sentido de generalizar uma estratégia de solução não-adaptativa. Assim sendo, esta solução passa a ser considerada uma instância de uso da ferramenta computacional desenvolvida dentre as demais existentes. Uma vez que o incremento utilizado em cada iteração é constante ao longo de toda a análise, todas as tarefas relativas à alteração do valor do incremento são portanto dispensáveis. Não faz sentido, por exemplo, calcular o valor da curvatura para cada passo de análise se este dado não vai ser utilizado na correlação com um novo incremento de iteração.

A implementação realizada para o uso desta estratégia de solução é portanto mínima, e concebida apenas para permitir a comunicação entre a biblioteca e a aplicação através da interface. A Figura 4.5 ilustra o método da classe *AdaptiveSolver* responsável por esta tarefa.

```
1 //Método de iteração não-adaptativa
2 void AdaptiveSolver::non_adaptive_solve(double start, double end){
3
4    //Inicialização
5    double instant= start;
6
7    //Consulta o valor do incremento através da interface
8    double increment= get_increment();
9
10   //Iteração
11   while (instant <= end){
12
13      //Faz pedido de iteração através da interface
14      solve_step(instant);
15
16      //Incrementa o instante corrente
17      instant = instant + increment;
18   }
19 }
```

Figura 4.5: Método de iteração não-adaptativa.

Observe que a rotina é composta basicamente por duas chamadas de funções da interface, uma para consultar o valor do incremento de integração a ser utilizado, e uma para fazer o pedido de iteração para a aplicação. O algoritmo da Figura 4.5 pode ser representado graficamente a partir do diagrama de seqüência ilustrado na Figura 4.6.

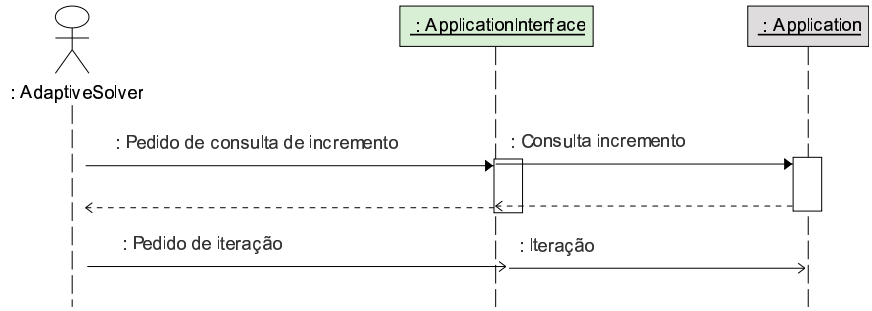


Figura 4.6: Diagrama de seqüência da estratégia de iteração não-adaptativa.

Na representação acima, as entidades apresentadas no topo da figura correspondem a objetos das classes especificadas pelos nomes apresentados. O objeto em destaque, representado pelo desenho de um indivíduo, é chamado de ator. É deste objeto que partem as interações com os demais. As linhas verticais representam as linhas de vida de um dado objeto. Estas retratam o histórico de suas atividades ao longo da execução de uma rotina. As linhas horizontais representam as trocas de mensagens entre os objetos envolvidos, ou seja, as chamadas de métodos.

4.3.2 Estratégia Adaptativa Sem Análise de Resposta

Nesta estratégia de solução o cálculo da curvatura é levado em consideração na solução de um passo. A partir do valor deste indicador geométrico, um novo incremento de integração é sugerido para a solução do passo posterior. Desta forma o procedimento de iteração adapta-se ao comportamento do modelo simulado. Nos instantes onde o valor da curvatura for pequeno, o incremento de iteração pode ser aumentado e, quando o contrário acontece, este deve incremento ser reduzido. A Figura 4.7 apresenta o algoritmo responsável por esta tarefa.

```
1 //Método de iteração adaptativo sem análise de resposta
2 void AdaptiveSolver::adaptive_solve(double start, double end){
3
4    //Inicialização
5    double instant= start;
6
7    //Iteração
8    while (instant <= end) {
9
10      //Faz pedido de iteração através da interface
11      solve_step(instant);
12
13      //Estima o valor da curvatura corrente
14      double curr_curvature= estimate_curvature();
15
16      //Consulta o valor do incremento corrente através da interface
17      double increment= get_increment();
18
19      //Especifica o valor do incremento a ser utilizado no próximo passo
20      double new_increment= inc_analyzer->get_new_increment(curr_curvature, instant);
21
22      //Incrementa o instante corrente
23      instant = instant + increment;
24
25      //Altera o valor do incremento a ser usado no próximo passo através da interface
26      if(new_increment != current_increment){
27         change_increment(new_increment);
28         current_increment = new_increment;
29      }
30
31   }
32 }
```

Figura 4.7: Método de iteração adaptativa sem análise de resposta.

Observe que o algoritmo apresentado não avalia a qualidade da resposta obtida em um dado passo. Ele apenas utiliza informações do passo corrente de iteração para que o próximo passo de iteração seja resolvido com a discretização apropriada. Em alguns tipos de problema, a exemplo daqueles que envolvem contato ou impacto, o uso deste tipo de algoritmo pode não ser apropriado.

Uma vez que a qualidade da resposta atual do passo de iteração não é levada em consideração, este algoritmo é dito como sem análise de resposta (sem *feedback*). O diagrama de seqüência deste algoritmo é apresentado na Figura 4.8.

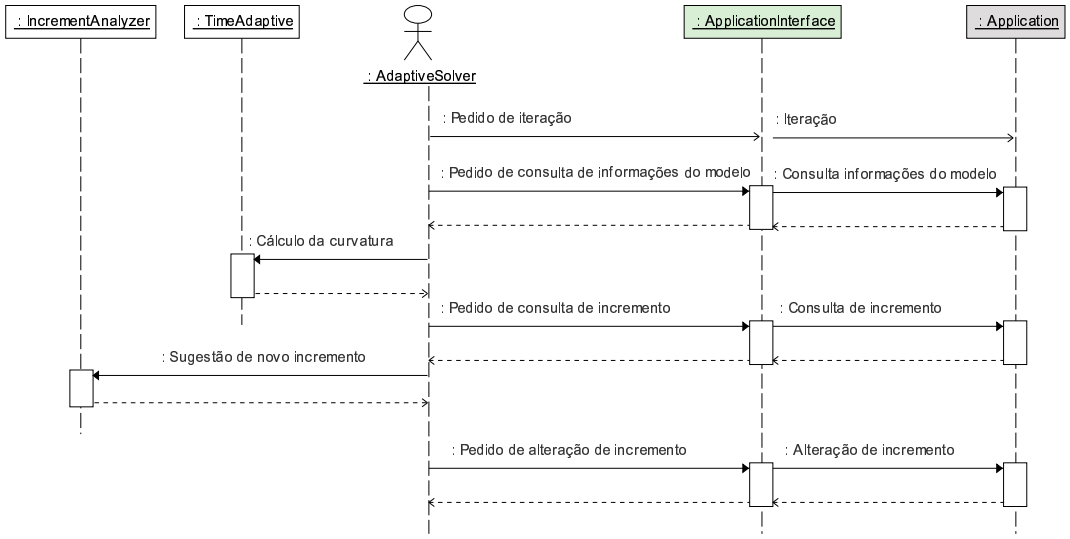


Figura 4.8: Diagrama de seqüência da estratégia de iteração adaptativa sem análise de resposta.

4.3.3 Estratégia Adaptativa Com Análise de Resposta

Esta estratégia de solução é um pouco mais sofisticada que as apresentadas anteriormente, porque é introduzido um mecanismo de avaliação de qualidade da resposta. Embora a curvatura continue sendo utilizada como parâmetro de avaliação do incremento de iteração, a mesma não é mais considerada de forma pontual como na estratégia adaptativa sem análise de resposta. São levados em consideração valores de curvatura que ocorrem em um dado intervalo de tempo. Estes intervalos são construídos de acordo com o valor do incremento máximo de iteração, e normalmente possuem uma extensão bem maior que as dos incrementos de análise.

A evolução da simulação ocorre de maneira condicional e dependente do máximo valor de curvatura observado no intervalo. Quando este valor diminue de um intervalo para o outro, as iterações que ocorreram em um intervalo são validadas e a simulação avança para a análise do próximo intervalo. Quando isso não ocorre, ou seja, quando o máximo valor de curvatura aumenta de um intervalo para o outro, as iterações realizadas no intervalo são descartadas. Novas iterações são então realizadas utilizando um incremento de iteração refinado de maneira compatível.

Para que este mecanismo iterativo possa ser implantado de fato recorre-se à utilização de uma entidade chamada de “estado”: o conjunto de informações que caracterizam um modelo em um dado instante. Se tomarmos como exemplo uma análise que avalia o histórico de deslocamentos de um corpo com propriedades elásticas, um estado é composto por vetores de deslocamentos, velocidades e acelerações, além das informações correspondentes ao tempo e ao incremento de tempo utilizado naquele instante.

A metodologia padrão de evolução de um algoritmo iterativo qualquer considera apenas o uso de um estado. Desta maneira, as variáveis que armazenam as configurações de um modelo em um dado instante são sempre atualizadas em um passo de iteração tomando como base informações do passo imediatamente anterior. Esta forma de evolução do algoritmo não permite que um dado instante seja simulado novamente de outra forma. Isso acontece uma vez que o valor do incremento de iteração é considerado sempre positivo.

Quando um algoritmo iterativo faz uso de mais de um estado, sua evolução ao longo da análise se torna mais flexível. Um mesmo intervalo de análise pode, por exemplo, ser analisado com diferentes valores de incrementos de iteração. Uma estratégia de iteração adaptativa com análise de resposta pode ser portanto utilizada neste caso. Deve-se portanto conceber um mecanismo que analise a resposta obtida durante um intervalo e avalie sua qualidade.

A Figura 4.9 apresenta o diagrama de atividades, em UML, de uma rotina que utiliza o valor da curvatura como critério de qualidade de resposta. Para tanto são utilizados dois estados que armazenam duas configurações do modelo em dois instantes quaisquer: o estado de simulação e o estado de predição.

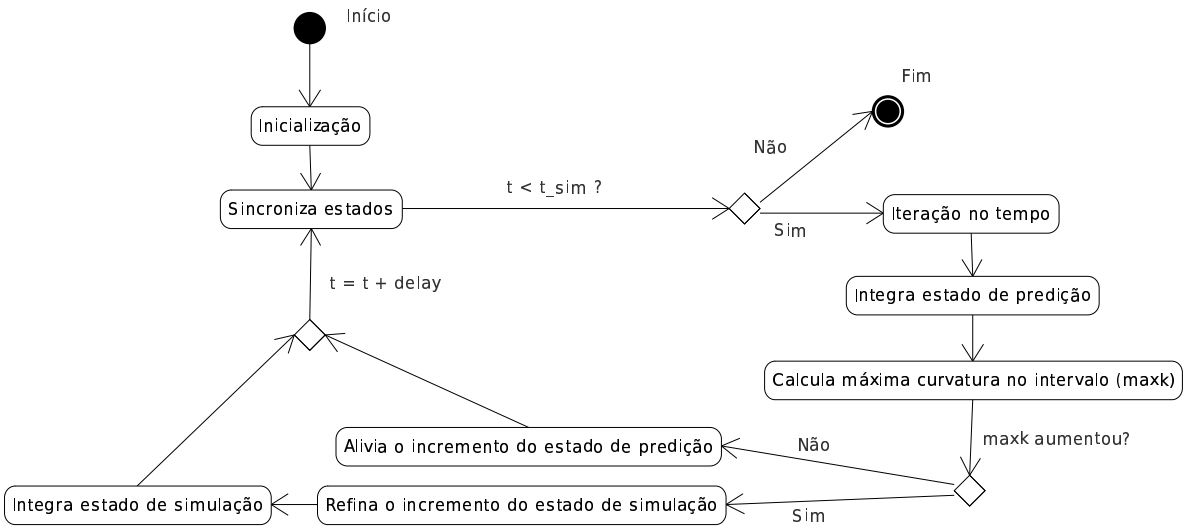


Figura 4.9: Algoritmo de integração com análise de resposta (*feedback*).

No início do processo iterativo, ambos os estados armazenam a configuração do modelo no instante inicial da análise. A partir deste instante, o estado de predição é atualizado de maneira a fazer a iteração ao longo da análise. A configuração do modelo é atualizada no processo iterativo e valores de curvatura são obtidos para cada instante de iteração. Ao fim de um certo intervalo de análise, extrai-se o máximo valor de curvatura observado. Caso este valor apresente uma tendência de queda em relação ao intervalo anterior, a análise realizada tomando como base o estado de predição é validada e este procedimento pode ser repetido para outro intervalo de

análise. As configurações do modelo armazenadas no estado de predição são copiadas para o estado de simulação e o procedimento de análise prossegue.

Quando em um dado intervalo de análise o máximo valor de curvatura observado aumenta em relação ao valor obtido no intervalo anterior, as iterações realizadas sobre o estado de predição são descartadas, e a análise é retrocedida até o início do intervalo. O estado de simulação é então resgatado e um novo processo de iteração, desta vez mais refinado, é utilizado para substituir a resposta anteriormente obtida. No final do intervalo, o estado de simulação é copiado para o estado de predição e o mecanismo que prevê e sincroniza os estados repete-se para o próximo intervalo.

Esta metodologia de adaptação é mais efetiva que a apresentada na seção anterior. Isso acontece uma vez que as alterações do incremento de iteração ocorrem antes mesmo que estas sejam necessárias. Entretanto, uma vez que dois estados de configurações do modelo são utilizados para a evolução da análise, esta metodologia requer o uso de memória auxiliar para armazenar o estado adicional.

No próximo capítulo são apresentados exemplos e aplicações dos conceitos, formulações e implementações relacionados com o processo de integração temporal adaptativo até então abordado.

4.4 *FASTTI*: Um *Framework* de Integração Adaptativa

Conforme comentado anteriormente, a análise de problemas robustos de dinâmica em geral utilizam recursos computacionais de forma intensa. Isso acontece uma vez que os métodos empregados nas soluções destes problemas costumam envolver rotinas computacionais dispendiosas, tais como soluções de problemas de autovalores e autovetores, resolução de sistemas de equações lineares, soluções iterativas, dentre outras.

O número de operações de ponto flutuante envolvidos na solução de problemas desta natureza está associado com da formulação empregada na solução e com o nível de discretização espacial que se utiliza para representar o modelo numérico. No Método dos Elementos Finitos, por exemplo, esta discretização é reflexo do nível de refinamento da malha de elementos utilizada. Quanto maior o nível de discretização, maior o número de operações efetuadas. Quando a solução ao longo do tempo dá-se a partir do uso de um Método de Integração Direta, o número de operações também é influenciado pelo nível de discretização que se utiliza na representação do intervalo de tempo de análise.

Conforme visto, os Métodos de Integração Direta propõem a solução de um histórico de resposta a partir de uma forma discreta, em um processo de obtenção passo-a-passo. Para o caso particular da análise da dinâmica de corpos sólidos, estes métodos resolvem o histórico do campo de deslocamentos de um corpo a partir da solução da equação de movimento (2.21). Esta resolução é feita para cada passo de integração envolvido no processo incremental e, portanto, tendo impacto sobre o tempo computacional envolvido na análise dinâmica do problema analisado.

No processo de integração desenvolvido, a cada passo de solução realizam-se duas tarefas distintas. A primeira é a atualização das variáveis envolvidas na equação de movimento, feita de forma particular e de acordo com a formulação empregada, envolvendo basicamente a determinação de vetores de forças internas e externas. A segunda etapa corresponde ao processo de integração temporal, onde o campo de deslocamentos e de suas correspondentes variações é determinado de forma apropriada (ver Seção 2.2.1). Esta última etapa pode ser realizada de diversas formas dependendo do método de solução empregado, no entanto pode ser aproveitada em diversos programas de análise dinâmica, uma vez que a metodologia empregada não vai mudar de um programa para o outro.

A etapa de integração temporal em si corresponde a uma tarefa altamente paralelizável computacionalmente. Isso acontece uma vez que as grandezas que estão sendo calculadas são armazenadas em vetores, permitindo portanto que seus elementos possam ser obtidos

independentemente.

Outro aspecto que deve ser levado em conta quando no uso de um Método de Integração Direta é a adaptatividade. Esta estratégia permite que o tamanho do passo de integração Δt seja regulado de forma automática, de acordo com a resposta que está sendo obtida no processo iterativo, buscando aliar dois fatores importantes envolvidos: tempo de processamento e qualidade de resposta.

Diante das características apresentadas do processo de integração, desenvolve-se o *framework FASTTI* (*Framework of Adaptive-Step Time Integration*). Trata-se de um sistema com arquitetura construída para facilitar o desenvolvimento de Métodos de Integração Direta, com suporte a estratégias adaptativas e técnicas computacionais de alto desempenho. A idéia principal do *FASTTI* é permitir que aplicações que utilizem estes métodos de integração possam usufruir dos recursos do *framework* de maneira fácil e rápida, exigindo o mínimo de esforço de programação possível.

O *FASTTI* atua de maneira a permitir que parte do processo de integração numérica seja feita no escopo do *framework* e uma outra parte seja feita no escopo do *software* ou sistema que faz o seu uso. Estas partes correspondem às etapas de integração observadas no uso de um Método de Integração Direta.

A primeira etapa consiste na atualização do estado de equilíbrio do modelo numérico simulado, ou seja, para uma dada configuração cinemática e um dado instante de tempo, quais são as forças externas atuantes e as forças internas calculadas. Esta etapa é realizada no escopo do *software*, uma vez que é este que conhece as relações constitutivas do modelo numérico simulado e suas condições de contorno. A segunda etapa corresponde à integração no tempo, realizada no escopo do *framework*. Nesta etapa os campos de deslocamentos, velocidades e acelerações são atualizados com base nos valores de forças obtidos na etapa anterior. Conforme comentado, esta etapa é comum para diversos softwares de dinâmica de corpos sólidos e justifica a sua implementação em um *framework*.

Ao longo deste trabalho são tratados casos de uso do *framework* que permitem tanto exemplificar a sua utilização em *softwares* existentes, como empregar a estratégia de adaptação alvo de estudo deste trabalho. Nas seções seguintes são tratados aspectos referentes à estrutura de classes atual do *FASTTI*, bem como as técnicas de computação de alto desempenho empregadas.

4.4.1 Arquitetura do Sistema Orientado a Objetos

Segundo Fayad & Schmidt (1997), um *framework* é um conjunto de classes que colaboram para realizar uma responsabilidade para um domínio de um subsistema da aplicação. Em outras palavras, trata-se de uma estrutura de suporte definida para que outro programa possa ser desenvolvido ou organizado em torno dela. Desta forma o desenvolvimento de um *software* se torna mais fácil, permitindo que os programadores invistam mais tempo na parte do código que realmente é o foco de investigação.

O framework *FASTTI* propõe que a tarefa de integração no tempo envolvida em um Método de Integração Direta qualquer seja inserida em um ambiente de desenvolvimento colaborativo. Sua estrutura de classes foi desenvolvida em torno de uma entidade que corresponde ao alvo de estudo de qualquer análise de dinâmica de corpos sólidos, o histórico de deslocamentos (classe *DisplacementHistory*). A Figura 4.10 apresenta organização de classes do *framework*.

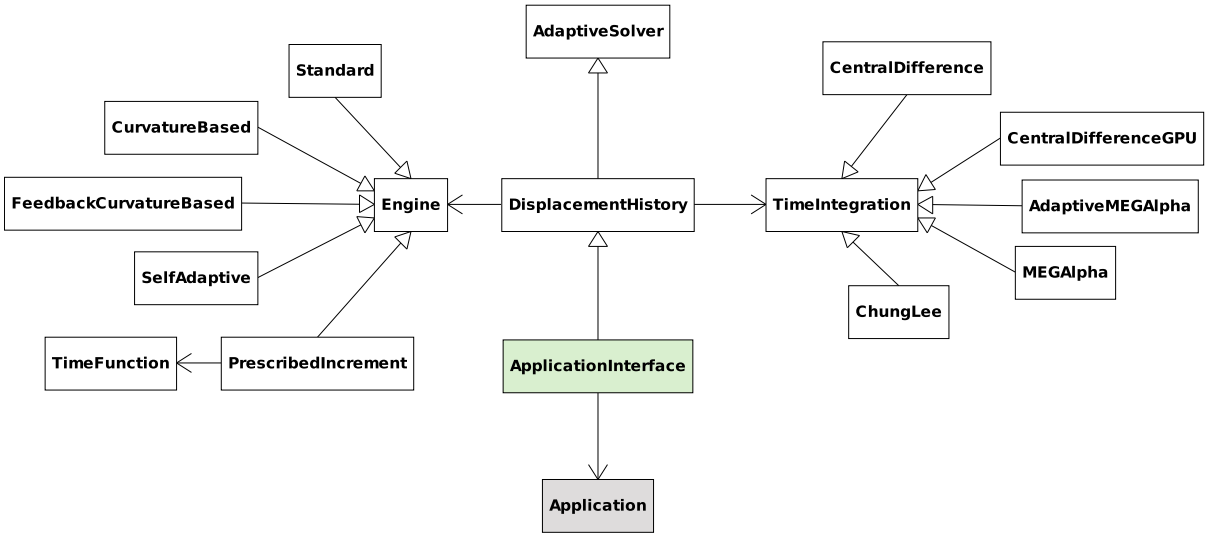


Figura 4.10: Diagrama de classes do *framework*.

Observe que a classe *DisplacementHistory* corresponde à parte central do sistema, de onde surgem todas as relações com as demais classes. Um objeto desta classe é capaz de caracterizar a cinemática de um objeto de estudo (um corpo) em um dado instante de tempo qualquer. Mas de que maneira isso é feito? Para responder esta pergunta é necessário entender o funcionamento de duas outras classes: *TimeIntegration* e *Engine*.

A classe *TimeIntegration* é responsável pela integração temporal propriamente dita. Nela, as informações de deslocamentos, velocidades e acelerações são calculadas em todos os graus de liberdade envolvidos na composição no modelo numérico simulado. Para um dado valor de incremento de integração Δt fornecido, os métodos desta classe atuam no sentido de

atualizar as informações cinemáticas citadas. Trata-se de uma classe abstrata, e que portanto assume formas específicas de acordo com o método de integração a ser empregado. As classes *CentralDifference*, *ChungLee*, *MEGAlpha* e *AdaptiveMEGAlpha* são exemplos de classes que especificam a forma geral de integração da classe *TimeIntegration*. Estas correspondem respectivamente, aos algoritmos de integração das diferenças centrais, Chung & Lee, Hulbert & Chung e MEG-α Adaptativo (Silveira, 2001).

Os objetos da classe *Engine* atuam no sentido de regular a forma com que a integração é feita ao longo do tempo. A idéia é permitir que estratégias de adaptação de formas diversas possam ser desenvolvidas, modificando a forma usual de integração, onde o valor do incremento Δt é mantido constante durante toda a análise. Estes objetos podem agir regulando o valor do incremento de tempo, comandando um processo de integração com controle de qualidade de resposta (*feedback*), ou de outra forma customizada qualquer. Assim como um motor de um automóvel permite a locomoção do veículo em uma rodovia, um objeto da classe *Engine* permite que um processo de integração avance ao longo do tempo de simulação.

Por se tratar de uma classe abstrata, a classe *Engine* não pode ser utilizada sem que antes assuma uma forma particular. No estágio atual de desenvolvimento do *framework*, as formas específicas da classe *Engine* são:

- Classe *StandardEngine*: Aplicada a análises onde um único valor de incremento é utilizado em toda a simulação.
- Classe *CurvatureBased*: Solução adaptativa baseada em curvatura e sem análise de resposta (Seção 4.3.2).
- Classe *FeedbackCurvatureBased*: Solução adaptativa baseada em curvatura e com análise de resposta (Seção 4.3.3).
- Classe *SelfAdaptive*: Técnica de integração adaptativa baseada em soluções locais aproximadas.
- Classe *PrescribedIncrement*: Aplicada a análises onde se deseja prescrever uma função temporal para o comportamento do incremento de integração.

Note que duas das classes acima apresentadas fazem referência à estratégia de adaptação baseada em curvatura. De fato, as funcionalidades embutidas na biblioteca de adaptação apresentadas no Capítulo 4 são utilizadas na construção do *framework FASTTI*. Observe na Figura 4.10 que a classe *DisplacementHistory* especifica um método de solução adaptativo geral baseado em curvatura (a classe de interface da biblioteca: *AdaptiveSolver*).

Conforme visto, as classes *TimeIntegration* e *Engine* encapsulam as principais atribuições do sistema responsável pelo processo de integração direta. Apesar disso a existência isolada de objetos destas classes não faz sentido. Isso acontece uma vez que estes devem estar associados a um objeto da classe *DisplacementHistory*, que fornece os dados necessários para os seus funcionamentos.

A classe *DisplacementHistory* corresponde portanto a uma interface de comunicação entre o *FASTTI* e um *software* genérico que faça estudo de problemas de dinâmica de corpos sólidos através de métodos de integração direta. Trata-se desta forma de uma classe abstrata, e atua de maneira semelhante à interface da biblioteca de adaptação apresentada na Seção 4.2, ou seja, fazendo a comunicação entre dois sistemas distintos. A integração do *framework* em *softwares* existentes é exemplificada em maiores detalhes na Seção 5.2.

As implementações que são realizadas no *framework FASTTI* podem ser classificadas de duas maneiras: internas ou modulares. A primeira denominação é utilizada para designar aquelas alterações que são realizadas no escopo do *framework*, ou seja, no conteúdo das classe que compõem este sistema ou na própria organização de classes. As implementações modulares são aquelas realizadas a partir da interface do *framework*. Estas últimas ocorrem de maneira a não se preocupar com o funcionamento interno do sistema, apenas com a classe de interface (*DisplacementHistory*). Estas implementações independentes recebem o nome de módulos. Na Figura 4.10 a entidade atuante no papel de um módulo é a classe *ApplicationInterface*.

As classes que atuam como módulos são construídas apenas com o intuito de fundir dois sistemas diferentes, o *framework* e o *software* cliente deste recurso. Lembre-se que a resolução de um passo de integração por um método direto é um processo que envolve duas etapas de tarefas distintas, executadas em locais distintos. Na prática a fusão entre estes dois sistemas é feita a partir da implementação de alguns métodos virtuais, tais como os ilustrados na Figura 4.11.

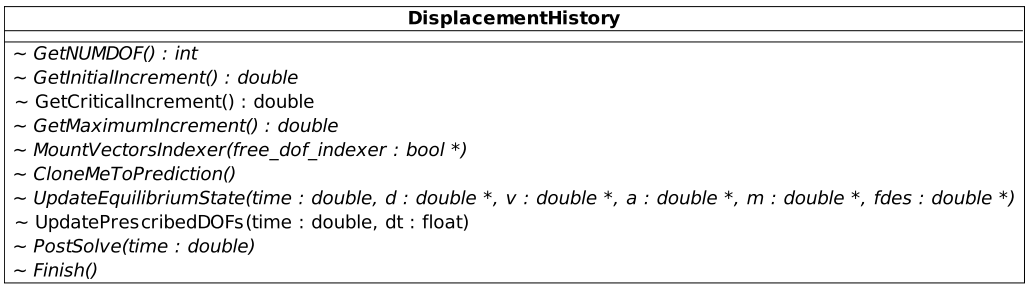


Figura 4.11: Classe de interface do *framework*.

Estes métodos tanto consultam informações importantes contidas no escopo do *software*, quanto ordena que este último execute determinadas ações. Por exemplo, enquanto o

método *GetMaximumIncrement* consulta da aplicação o valor máximo para o incremento de integração, o método *UpdateEquilibriumState* ordena que a aplicação atualize o vetor de forças desequilibradas do modelo numérico para uma dada condição cinemática.

4.4.2 Técnicas de Computação de Alto Desempenho Empregadas

A seguir são apresentadas algumas das técnicas de computação de alto desempenho empregadas no desenvolvimento do *framework*. A utilização das mesmas é justificada pela necessidade de viabilizar a análise de modelos numéricos robustos. Para análises dinâmicas de problemas deste tipo, o emprego destas técnicas permite uma redução de tempo de processamento, o que muitas vezes é um fator limitante. Com um processamento realizado com um maior desempenho, análises de modelos numéricos mais sofisticados podem se tornar viáveis, dando um nível de precisão maior a uma análise de um problema físico qualquer.

Conforme observado anteriormente, os dados manipulados pelo *framework* em geral estão armazenados na forma de vetores. Correspondem basicamente a informações de deslocamentos, velocidades, acelerações, forças, dentre outros. Os componentes destes vetores estão associados a graus de liberdade do modelo numérico. A obtenção destes dados pode ser realizada de forma independente em um processo de integração numérica, sobretudo quando são utilizados métodos explícitos. Desta forma, a tarefa de resolver um passo de análise pode ser feita de forma paralela e, assim sendo, o tempo de processamento necessário diminui.

Outra característica que pode ainda ser observada é que todos os componentes dos vetores recebem as mesmas instruções de cálculo. Por exemplo, o processo de determinação dos deslocamentos para todos os graus de liberdade é o mesmo. Isso quer dizer que o sistema pode processar diversos dados com a mesma instrução. Arquitetura de sistemas com esta característica são classificadas como *SIMD* (*Single Instruction, Multiple Data*), segundo a taxonomia de Flynn (apud Duncan, 1990). Esta classificação é válida quando os dados, neste caso particular chamados de *streams*, são naturalmente paralelizáveis, e que uma unidade de processamento pode efetuar uma mesma instrução para todos eles. Os *streams* possuem boa aplicabilidade em processadores vetoriais e *GPUs* (*Graphics Processing Units*), onde os mesmos podem ser processados de forma paralela. O uso de processadores destes tipos já são uma realidade em aplicações com propósitos mais gerais possíveis (Owens *et al.*, 2007).

Duas técnicas empregadas na construção do *FASTTI* permitem tomar partido desta forma de paralelismo de dados citada. Trata-se do *OPENMP* e da linguagem de programação *OpenGL Shading Language*.

O *OpenMP* (*Open Multi-Processing*) é uma *API* (*Application Programming Interface*) que define um conjunto de diretivas estabelecidas no intuito de permitir a utilização de programas paralelos em ambientes de memória compartilhada (Chapman *et al.*, 2007). O *OpenMP* define um padrão de programação de alto nível aplicável às linguagens C/C++ e Fortran e que facilita o desenvolvimento de aplicações paralelas, direcionadas tanto ao uso em computadores pessoais quanto em supercomputadores.

O uso das diretivas estabelecidas pelo padrão citado permite que determinadas tarefas possam ser realizadas de forma paralela, através de processos chamados de *threads*. As arquiteturas de computadores atuais permitem que as tarefas contidas nas *threads* sejam divididas entre núcleos de processamento, permitindo portanto que uma melhoria de performance seja observada. Na forma de programação convencional, sem o uso do *OpenMP*, todas as tarefas das *threads* são realizadas por apenas um núcleo de processamento. O uso de arquiteturas de processadores com múltiplos núcleos (*multicore*) tem se difundido entre os fabricantes deste produto. Esta é uma alternativa empregada para a solução de alguns problemas de superconsumo e emissão de calor, enfrentados para a tecnologia da época.

No contexto do *framework*, o *OpenMP* é utilizado no intuito de criar *threads* para a tarefa de integração no tempo. Na prática isso consiste em designar a cada *thread* a tarefa de integrar numericamente um conjunto de graus de liberdade. Desta forma, quando um programa que faz uso do *framework* é executado em uma máquina com arquitetura *multicore*, a tarefa de integração no tempo é divida entre os núcleos de processamento, permitindo desta forma que este processo seja realizado de forma mais rápida. Todos os métodos de integração disponíveis no *framework*, e que utilizam a *CPU* (*Central Processing Unit*) para o processamento, fazem o uso das diretivas do *OpenMP*.

Note que até então a técnica de paralelização apresentada utiliza apenas recursos da *CPU* para o processamento dos dados. Uma alternativa que vem surgindo nos últimos anos consiste no uso da *GPU* para efetuar cálculos numéricos com propósitos gerais. Este fato é motivado pela elevada capacidade de processamento que estes dispositivos apresentam. Uma *GPU* é composta internamente por diversos processadores, inicialmente desenvolvidos com o objetivo de dar suporte a computação gráfica. Alguns destes processadores, os chamados *fragment processors*, são *SIMD*. Linguagens de alto nível, a exemplo da *CG* (Mark *et al.*, 2003), *Brook* (Buck *et al.*, 2004), *CUDA* (Nickolls *et al.*, 2008) , e *OpenGL Shading Language* (Rost, 2004), permitem que estes processadores sejam programáveis e efetuem operações de ponto flutuante com um objetivo qualquer. No trabalho de Georgii & Westermann (2005), por exemplo, a *GPU* é empregada na solução de problemas de dinâmica estrutural. Tarefas como montagem

de matrizes de rigidez e integração temporal são apenas alguns exemplos de tarefas que podem ser paralelizadas com o uso da *GPU*.

No *FASTTI* a *GPU* é utilizada na solução de um método de integração no tempo, o clássico Método das Diferenças Centrais, e a linguagem de programação empregada é a *OpenGL Shading Language*. A implementação desta técnica até então tem apenas fins acadêmicos. Alguns fatores ainda devem ser tratados antes que esta forma de paralelização possa ser de fato disponibilizada pelo *framework* para o uso em *softwares* clientes. O principal consiste na forma de representação dos valores numéricos. Deve-se assegurar que a *GPU* seja capaz de representar estes valores com a mesma precisão utilizada na *CPU*. Desta forma a viabilidade de uso deste dispositivo gráfico não seria comprometida inicialmente por este fator.

É importante informar que o uso das técnicas de computação de alto desempenho citadas acima deve ser feito de forma apropriada. Para algumas situações, estas formas de processamento apresentam peculiaridades que devem ser levadas em consideração. Em alguns casos o uso destas técnicas não é viável ou mesmo não resulta em melhoria de desempenho. Isso acontece em virtude das perdas de tempo inerentes ao próprio processo de paralelização. No caso da paralelização por *OpenMP*, o simples fato de criar de uma *thread* já acarreta perda de desempenho. Quando a *GPU* é utilizada, e existe comunicação entre este dispositivo e a *CPU*, também são verificados retardos de tempo. Estes tempos de processamento perdidos devido a atividades deste gênero devem ser considerados, sobretudo quando se lida com problemas de pequeno porte. Em problemas mais robustos, estes retardos de tempo passam a não representar parcela significativa do tempo total de processamento. Em geral, os melhores resultados de desempenho em técnicas de paralelização são observados quando o número de dados a ser processados é elevado ou quando o número de instruções realizadas por dado é grande.

5 *Exemplos e Aplicações*

5.1 Choque Elástico Entre Partículas

Assim como na análise de problemas de contato-impacto, o estudo do choque elástico entre partículas deve atentar aos aspectos relacionados a esta categoria de problemas. As técnicas de solução empregadas devem assegurar que a interação entre os corpos ocorra de maneira compatível com as leis da física. Não se deve permitir, por exemplo, que dois corpos ocupem o mesmo local no espaço. Alguns métodos de solução numérica, a exemplo do Método dos Elementos Discretos (Cundall; Strack, 1979), admitem que esta sobreposição possa ocorrer. A partir desta sobreposição, forças resultantes do contato entre os corpos são definidas e a uma dinâmica resultante deste processo pode ser então obtida. Em uma situação ótima o contato entre os dois corpos ocorre de maneira instantânea, ou seja, a magnitude da sobreposição e o período em que os corpos estão sobrepostos tendem a zero. Para que este problema físico seja representado de maneira apropriada em um processo de integração numérica direta faz-se necessária a utilização de incrementos de integração bastante reduzidos, o que acaba refletindo-se diretamente no esforço de execução gasto por esta tarefa. Evidencia-se portanto uma relação entre a qualidade de representação de um modelo numérico e o seu tempo de solução.

O exemplo a seguir tem como objetivo ilustrar esta relação a partir do estudo de um problema simples e de solução analítica conhecida. Busca-se ainda avaliar o emprego de estratégias de adaptação temporal no método de solução numérica empregado. O problema consiste em obter o histórico de movimento de um corpo esférico indeformável submetido à ação da gravidade, inicialmente em repouso e lançado a uma dada altura de uma superfície, conforme ilustrado na Figura 5.1. Apresentam-se diversas soluções numéricas e realizam-se comparações com o objetivo de avaliar aspectos relacionados com a qualidade de resposta, como também com o tempo demandado para a análise (performance).

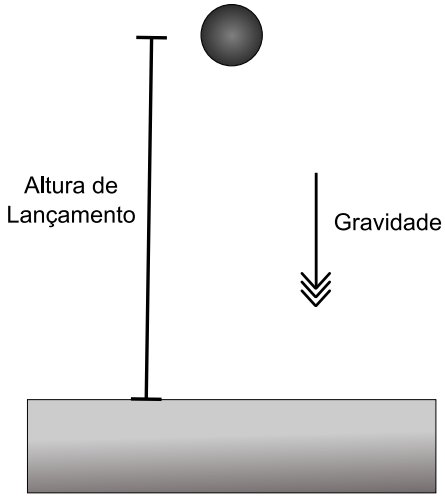


Figura 5.1: Queda livre com choque perfeitamente elástico.

A dinâmica do problema é bem conhecida. Em virtude da ação da força gravitacional, a esfera citada desenvolve um movimento que a conduz ao choque com a superfície. No instante do contato ocorrem mudanças na cinemática dos corpos em interação, uma vez que a energia cinética total pode ser convertida em outras formas de energia, a exemplo de energia térmica ou interna. Considerando que este choque seja elástico, ou seja, sem perda de energia, e que a superfície apresentada seja imóvel, pode-se definir de maneira direta a cinemática da esfera após a colisão. Neste caso particular, a intensidade do vetor velocidade da esfera conserva-se após o choque e sua direção inverte-se. A esfera é então lançada para cima iniciando um movimento que a conduz à sua posição inicial de lançamento. A partir deste instante, o movimento de queda e restituição repete-se indefinidamente.

Note que o movimento cíclico da esfera só é possível em virtude da natureza elástica do choque. Na prática este tipo de choque não existe em virtude das perdas de energia inevitáveis. Os choques que acontecem com transformação de energia são chamados inelásticos. Caso a natureza do choque em estudo fosse inelástica, a altura até a qual a esfera é restituída após o choque diminuiria ao longo do tempo até o instante em que este corpo repousaria sobre a superfície. Uma vez que o foco da investigação deste problema neste trabalho não está na cinemática do movimento da esfera, e sim na análise de uma técnica numérica de solução, logo, por conveniência, opta-se pela análise de um choque elástico.

5.1.1 Solução Analítica

Considerando que o choque da esfera com a superfície seja representado a partir de um modelo de contato elástico linear, o histórico da sua altura em relação à superfície pode ser

obtido analiticamente a partir da solução da equação de movimento (2.17). Para uma rigidez de contato k, uma altura de lançamento h_0 e assumindo um valor g para a gravidade, a posição desta esfera ao longo do tempo é dada por:

$$h(t) = \begin{cases} h_0 - \frac{1}{2} g t^2 & \text{se } t \leq t_q \\ \frac{mg}{k} \cos\left(t_c \sqrt{\frac{k}{m}}\right) - \sqrt{\frac{2mgh}{k}} \sin\left(t_c \sqrt{\frac{k}{m}}\right) - \frac{mg}{k} & \text{se } t_q < t \leq t_{ac} \\ t_r \sqrt{2gh} - \frac{1}{2} g {t_r}^2 & \text{se } t_{ac} < t \leq t_f \end{cases} \tag{5.1}$$

onde

$$t_c = t - t_q \ , \tag{5.2}$$

$$t_r = t - t_{ac} \ , \tag{5.3}$$

$$t_q = \sqrt{\frac{2h_o}{g}} \ , \quad t_{ac} = t_q + t_{cont} \ , \quad t_f = 2\, t_q + t_{cont} \ , \tag{5.4}$$

e t_{cont} é o tempo em que a esfera está em contato com a superfície hipotética. Este valor corresponde à primeira raíz positiva da solução de $h(t) = 0$ para o segundo intervalo da Equação (5.1).

Conforme visto, o movimento da esfera é descrito através de uma função definida em três partes. Esta função caracteriza o movimento da esfera dentro de um período que compõe os intervalos de tempo em que a esfera aproxima-se e afasta-se da superfície, bem como os instantes em que a mesma está em contato com este obstáculo. A extensão deste último intervalo de tempo é definido, dentro outros fatores, pela rigidez empregada no modelo de representação do contato. Quanto maior o valor da constante que define a rigidez de contato k, menor é o tempo em que a esfera permanece em contato com a superfície. Isto pode ser observado na Figura 5.2, que apresenta históricos de posições verticais da esfera para diferentes valores de k. Os valores de alturas apresentados no gráfico são normalizados pela altura de lançamento, e a escala de tempo pelo período t_f.

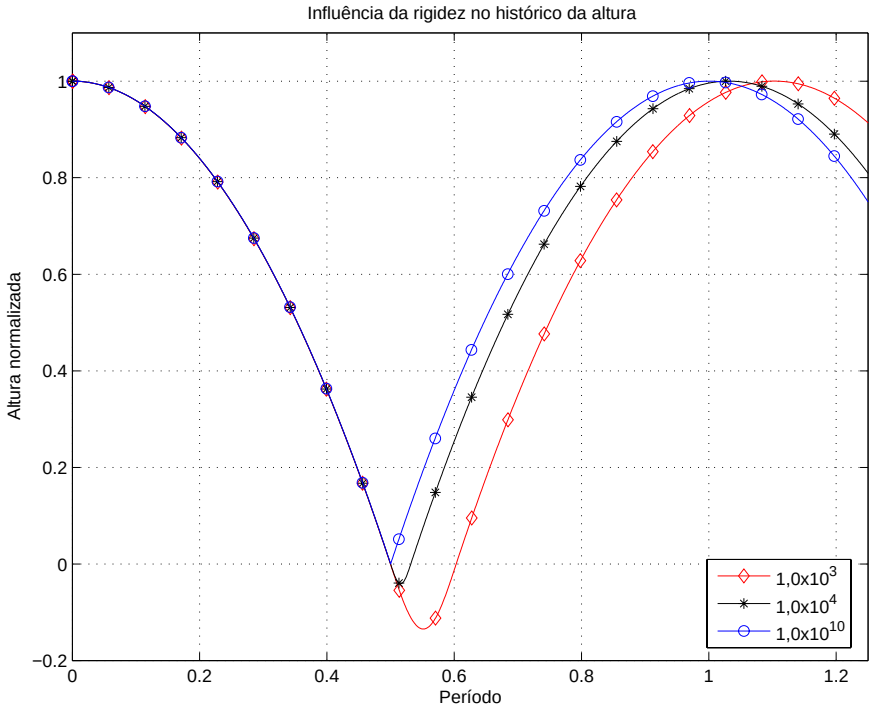


Figura 5.2: Influência da rigidez de contato no histórico da posição da esfera.

Os instantes em que o contato esfera-superfície está ocorrendo são aqueles em que as posições verticais da esfera assumem valores negativos. Note que o intervalo de tempo em que isso ocorre é de fato função do valor da rigidez de contato. Esta grandeza, conforme visto, tem influência direta no histórico de deslocamentos da esfera. Se tomarmos como referência o valor do indicador curvatura para a função que descreve este histórico, pode-se observar uma perturbação justamente no instante do choque. A Figura 5.3 apresenta o comportamento da função que descreve a curvatura do histórico de deslocamentos para diversas configurações de rigidez de contato.

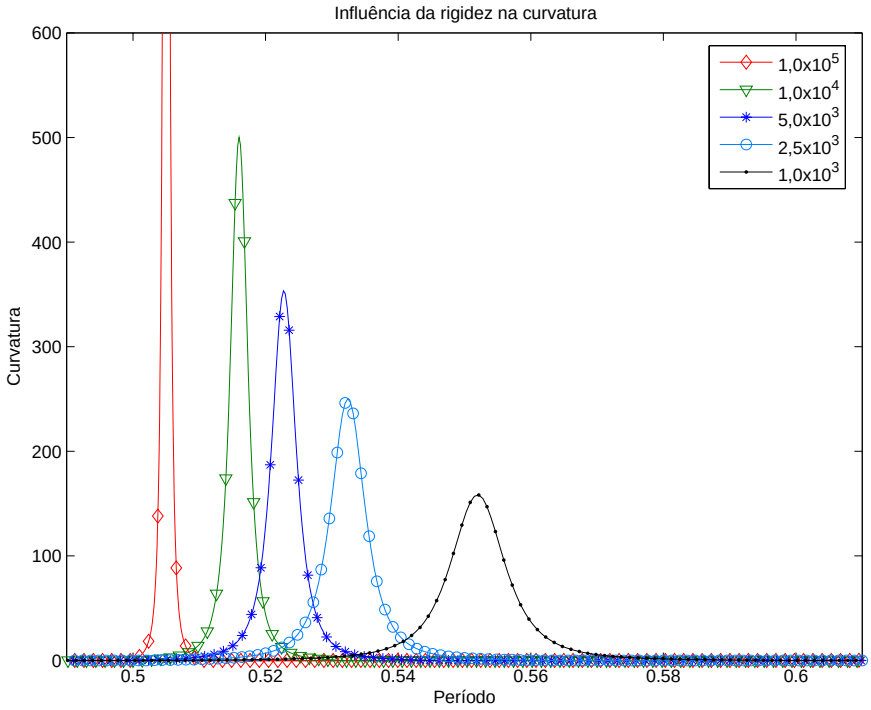


Figura 5.3: Influência da rigidez de contato no histórico de curvatura (instante do choque).

Veja que em todos os casos o valor da curvatura é afetado pelo choque da esfera com a superfície, no instante correspondente a aproximadamente meio período. A velocidade de variação deste indicador, assim como o seu máximo valor assumido, são funções novamente do valor de k. Quanto maior o valor da rigidez, maior o valor da curvatura do histórico. Mais adiante estas informações são utilizadas em um procedimento numérico de integração adaptativa no tempo.

5.1.2 Soluções Numéricas

As soluções numéricas propostas para a resolução do problema apresentado (Figura 5.1) são baseadas no Método dos Elementos Discretos. Este método permite avaliar a dinâmica do movimento de elementos, também chamados de partículas, que interagem entre si através de forças de contato. Estes elementos em geral são livres de conectividades, ao contrário de métodos que fazem uso de malhas, e seus históricos de movimento também podem ser obtidos a partir da solução da equação de movimento (2.21). Consideram-se forças externas atuantes sobre as partículas, bem como forças internas decorrentes do contato entre elas. Os mecanismos de tratamento de contato entre partículas são gerais e podem considerar, por exemplo, modelos visco-elastoplásticos. Tendo em vista a modelagem numérica do problema de choque elástico anteriormente apresentado, realizam-se algumas adequações e particularizações para o uso do

método. Supõe-se que o contato entre a esfera e a superfície seja representado pelo contato entre duas partículas. A primeira representa a esfera, cujo movimento se deseja analisar, e a segunda representa a superfície, necessária para a simulação do choque e admitida como imóvel. A Figura 5.4 ilustra o modelo numérico gerado.

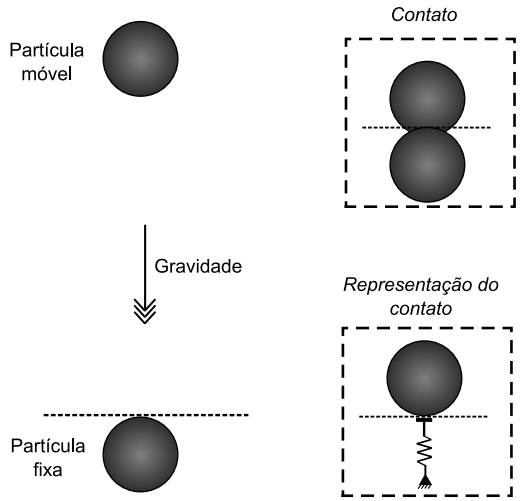


Figura 5.4: Modelagem por elementos discretos para o problema de contato.

Note que o contato é considerado a partir do uso de um modelo elástico representado por uma mola. Esta mola tem por objetivo restituir o movimento da partícula quando a mesma entra em contato com a superfície (representada pela partícula fixa). Uma força de restituição é aplicada à partícula livre toda vez que uma porção de seu volume esteja se sobrepondo ao volume da partícula fixa, conforme ilustrado na Figura 5.4. Para partículas esféricas esta sobreposição pode ser calculada a partir da diferença entre a soma dos raios das partículas em contato e a distância entre o centro das mesmas. A relação entre a sobreposição ou penetração das partículas e a força de restituição aplicada é linear. Quanto maior for a sobreposição, maior é o valor da força de restituição.

Para que haja uma boa representação do contato da esfera com a superfície a partir do modelo numérico, a rigidez de contato, ou seja a rigidez da mola, deve ser elevada. Esta penalização diminui as sobreposições fictícias entre partes dos volumes das partículas, reduzindo portanto os erros de modelagem. Além disso, uma outra questão deve ser levada em conta. Uma vez que a solução numérica é em geral obtida a partir do uso de um método de integração direta explícito, deve-se atentar ao valor do incremento de tempo de integração utilizado para a integração numérica. Já que os deslocamentos entre cada passo de integração são funções que dependem da sua cinemática (velocidades e acelerações) bem como do incremento de tempo utilizado, é prudente que a escolha do último seja feita de maneira criteriosa.

A escolha do incremento de tempo de integração na solução do histórico de movimento da partícula em estudo afeta diretamente questões tais como precisão de resposta e consumo de recursos de processamento. Para este caso específico, a definição do incremento de tempo é afetada por um agravante: o tratamento da numérico de problemas de contato. Uma vez que a cinemática do movimento é obtida de maneira discreta, a simulação deste fenômeno seria possível apenas com o uso de incrementos de tempo que bastante reduzidos. Isso tornaria muito cara a obtenção de uma resposta numérica para este problema, sobretudo quando não são utilizadas técnicas de adaptação no tempo para o processo de integração numérica. Embora exista a necessidade do uso de incrementos de integração muito pequenos para o tratamento do contato o mesmo só faz-se necessário durante um período de tempo muito pequeno. Na realidade este período também tende a zero. Como nem sempre é possível prever os instantes onde os contatos ocorrem ao longo de uma simulação, os programas de análise que fazem estudo deste fenômeno, e que utilizam técnicas de integração não adaptativa, recorrem ao uso de incrementos de tempo muito pequenos para toda a simulação.

Da Equação (5.1), observa-se que a função que descreve o deslocamento da esfera possui dois trechos onde a mesma assume formas quadráticas. O emprego de um método de integração de segunda ordem, a exemplo do Método das Diferenças Centrais, permite a obtenção exata da resposta de deslocamentos para estes trechos. Isso ocorre independentemente do valor do incremento de integração (Δt) empregado. O uso de valores de incrementos maiores permitem portanto que menos passos incrementais sejam necessários para a solução, diminuindo o tempo gasto para a obtenção de uma resposta.

De qualquer maneira a escolha do incremento de análise é limitada pelo trecho de solução correspondente aos instantes de contato. Para este intervaldo de tempo de análise, a exatidão do método de integração citado não ocorre de maneira incondicional. Cundall & Strack (1979) apresentam o valor do incremento de integração crítico que assegura a estabilidade da solução do problema neste intervalo, dado por:

$$\Delta t_{crit} = 2\sqrt{\frac{m_p}{k_c}}. \tag{5.5}$$

Este valor é definido a partir da massa da partícula m_p, e da rigidez de contato k_c. Trata-se apenas de um valor de referência para o processo de integração e que assegura a estabilidade do método de solução. Deve-se ter em mente que o valor do incremento de integração está associado com o valor da penetração entre as particulas e, assim sendo, com o valor da força de repulsão que surge do contato. Quanto maior for o valor do incremento, maior é o valor desta força. Em algumas situações o emprego de valores de incrementos elevados

pode acarretar valores de forças repulsivas fisicamente elevados e deve ser portanto evitado. Em geral percentuais do valor de Δt_{crit} são utilizados de maneira a obter o o nível de qualidade de resposta desejado. A escolha do incremento tem portanto influência no erro proveniente deste modelo numérico assumido. Outro fator que influencia a qualidade da resposta é o próprio método de integração empregado. Visando investigar estes aspectos, apresentam-se a seguir soluções numéricas que permitem discutir a influência destas variáveis na qualidade de resposta obtida em um processo de integração. Nas análises realizadas utilizam-se os parâmetros físicos e geométricos apresentados na Tabela 5.1.

Tabela 5.1: Propriedades físicas e geométricas do modelo numérico de contato

Propriedade	Símbolo	Unidade	Valor
Gravidade	g	m/s^2	10
Altura de lançamento	h_0	m	1,25
Rigidez de contato	k_c	kN/m	10^7
Massa da partícula	m_p	kg	1

A Figura 5.5 ilustra a influência do incremento Δt no valor do erro de integração numérica. Utiliza-se como referência o valor do erro relativo para o comprimento de arco da função histórico de deslocamentos. Os valores de comprimentos de arcos obtidos nas soluções numéricas são confrontados com os correspondentes comprimentos obtidos a partir da solução analítica, Equação (5.1). O valor de comprimento de arco foi escolhido como parâmetro de comparação de erro uma vez que permite retratar todo o histórico da qualidade da solução numérica ao longo da simulação. A comparação pontual de erro utilizando diretamente o dado de altura da partícula como parâmetro de referência poderia indicar valores de erros incoerentes em algumas situações, aquelas onde o valor do erro é instantaneamente pequeno mas historicamente grande.

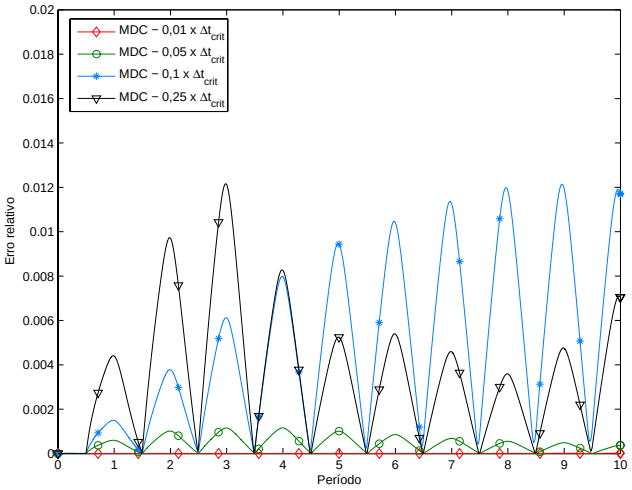
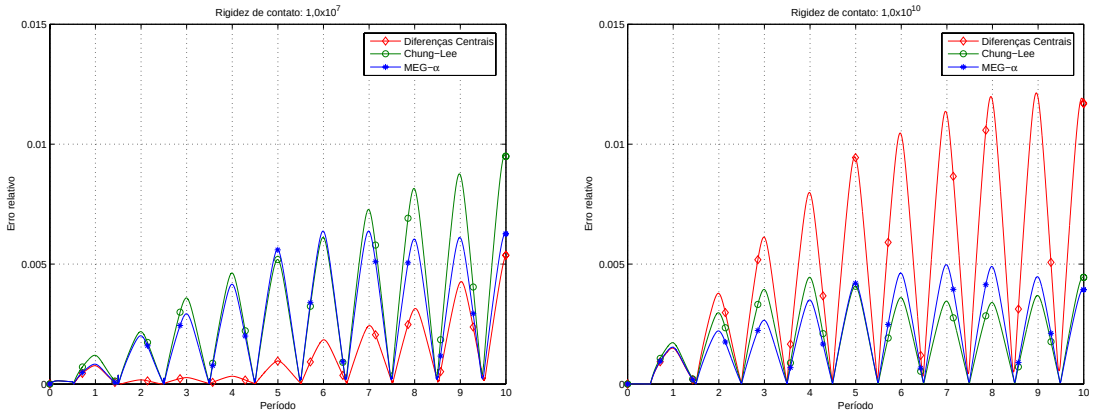


Figura 5.5: Influência do incremento de integração no erro relativo.

Observe que para um dado método de integração escolhido, neste caso o Método das Difereças Centrais, as melhores respostas numéricas são obtidas para os casos em que os valores dos incrementos são menores. Note também que até o momento do choque, no instante correspondente a meio período, o valor do erro de integração é desprezível, conforme já havia sido ressaltado.

A seguir avalia-se a influência do método de integração temporal na qualidade da resposta numérica obtida. São reproduzidas configurações de análise onde os algoritmos do MEG-α, de Chung & Lee e o do Método das Diferenças Centrais são confrontados quanto ao valor do erro de integração. Para cada uma das análises apresentadas nas Figuras 5.6(a) e 5.6(b) adota-se um valor constante para o incremento de integração, correspondente a 10% do valor de $(\Delta t)_{crit}$. Estes valores são obtidos levando em consideração massas unitárias para as partículas em questão e dois casos de rigidez de contato, $k_c = 10^7 N/m$ e $k_c = 10^{10} N/m$. Para estas condições os valores correspondentes de incrementos de integração críticos são $\Delta t = 0{,}0002\sqrt{10}\ s$ e $\Delta t = 2 \times 10^{-5}\ s$, respectivamente.



(a) Rigidez de Contato: $10^7 N/m$ (b) Rigidez de Contato: $10^{10} N/m$

Figura 5.6: Influência da rigidez de contato no erro para os métodos de integração.

Note que os melhores resultados observados para os algoritmos do MEG-α e de Chung & Lee ocorrem para a configuração de análise onde o valor da rigidez é maior (Figura 5.6(b)). Isso acontece em virtude da utilização de amortecedores numéricos empregada nestes algoritmos.

5.1.3 Soluções Numéricas Adaptativas

Nas soluções até então apresentadas o valor do incremento de integração é mantido constante durante todo o processo iterativo de solução. Não há portanto nenhum mecanismo que regule o valor deste incremento à medida que o processo de integração vai acontecendo. Com o intuito de investigar o emprego de técnicas que incorporem este mecanismo, apresenta-se a seguir um estudo onde algoritmos de integração adaptativos são empregados na solução do problema de choque de partículas até então tratado.

As soluções utilizam duas metodologias de adaptação distintas. A primeira emprega o valor da curvatura do histórico de deslocamentos, um mecanismo de simulação iterativa com análise de resposta (Figura 4.9) e o Método das Diferenças Centrais. A outra metodologia utiliza a sistemática de adaptação baseada em soluções locais aproximadas do MEG-α Adaptativo, apresentado na Seção 2.3.3. Esta última é apenas utilizada com o objetivo de situar o emprego da primeira metodologia, proposta de estudo deste trabalho, frente a outras técnicas de adaptação já existentes. Em alguns casos soluções não adaptativas também são utilizadas também com propósito de comparação.

Em todas as soluções, adaptativas ou não, emprega-se o *framework FASTTI* para a simulação computacional. O exemplo reproduzido, Figura 5.4, e os dados de análise (gravidade, altura de lançamento e massa da partícula), obtidos na Tabela 5.1, são os mesmos

em todas as situações. Assegura-se também que o incremento de integração estava sempre abaixo do valor crítico ($\Delta t_{crit} = 2 \cdot 10^{-5}$), determinado a partir da Equação (5.5).

A Figura 5.7 apresesenta o histórico da trajetória da esfera (partícula) para duas configurações distintas de análise. Uma utiliza a estratégia de adaptação no tempo baseada em curvatura comentada anteriormente e a outra faz uso de um operador de integração convencional. Para o último caso o valor do incremento utilizado durante toda a análise corresponde a 10% do valor de Δt_{crit}. O gráfico foi construído de maneira a apresentar o histórico de altura normalizada da partícula. Para tanto a altura inicial de lançamento é utilizada como referência. A escala de tempo também foi normalizada pelo período gasto pela esfera repetir seu movimento de queda livre e restituição após o contato. Com o objetivo de facilitar a visualização das respostas numéricas obtidas, apenas alguns pontos representativos destas soluções são apresentados. A distância entre estes pontos não tem portanto nenhuma relação com os valores dos incrementos de integração utilizados.

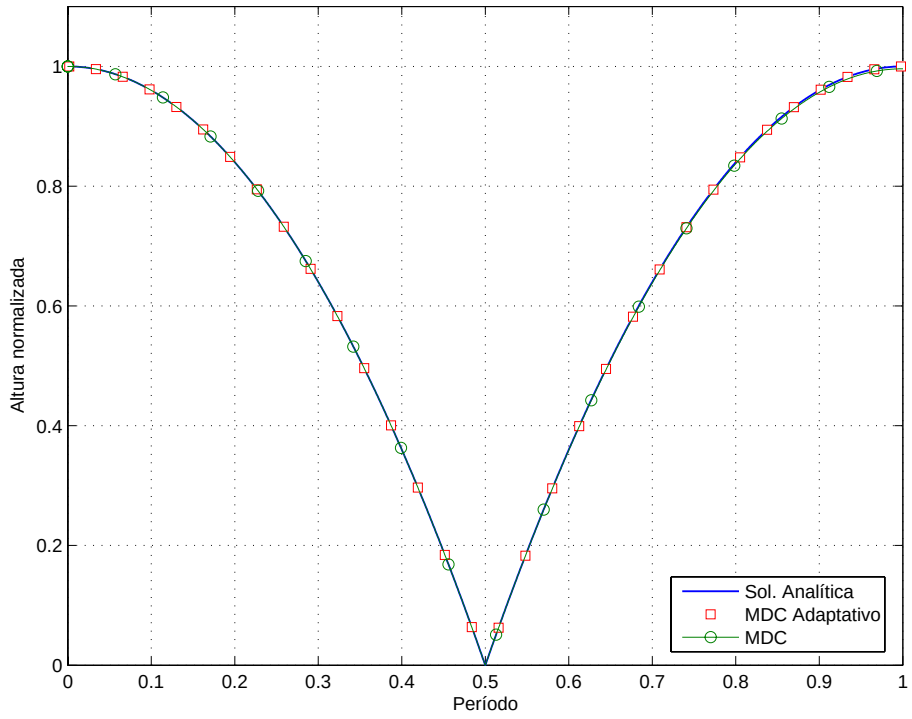
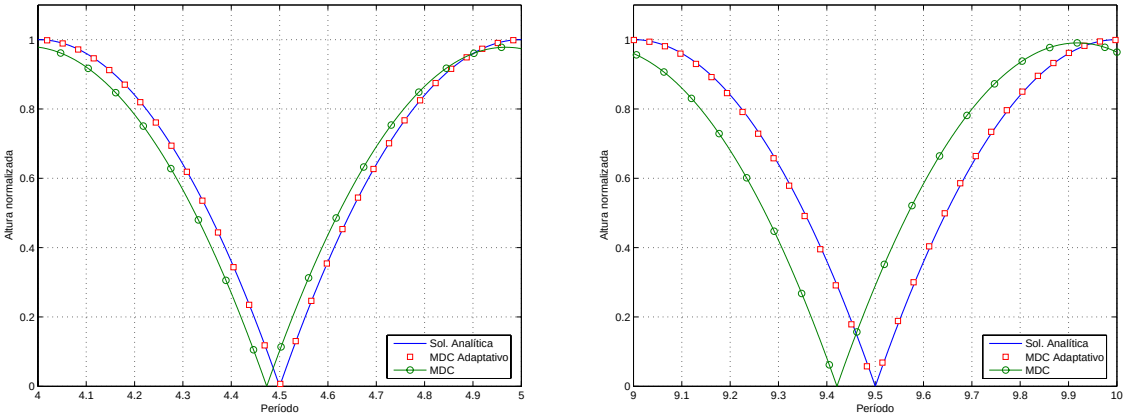


Figura 5.7: Trajetória do movimento - Primeiro Período.

Nos instantes iniciais da análise ambas as soluções conduzem a respostas muito próximas da solução exata. Em uma análise mais criteriosa pode-se observar que a solução não adaptativa apresenta uma precisão um pouco maior do que a adaptativa nestes instantes. No entanto, uma vez que o contato ocorre, no instante correspondente a meio período, a solução não adaptativa começa a apresentar uma tendência de divergência da solução exata. Essa tendência de divergência da resposta permanece ao longo de toda a simulação, conforme pode ser observado

nas Figuras 5.8(a) e 5.8(b).



(a) Quinto Período (b) Décimo Período

Figura 5.8: Trajetória do movimento.

Note que a solução adaptativa, mesmo após sucessivos contatos com a superfície, continua a apresentar uma melhor precisão de resposta que a solução não-adaptativa. Esta característica é mantida durante todo o processo de análise, conforme pode ser observado na Figura 5.9, que retrata o histórico do erro relativo à solução analítica para as soluções numéricas em questão.

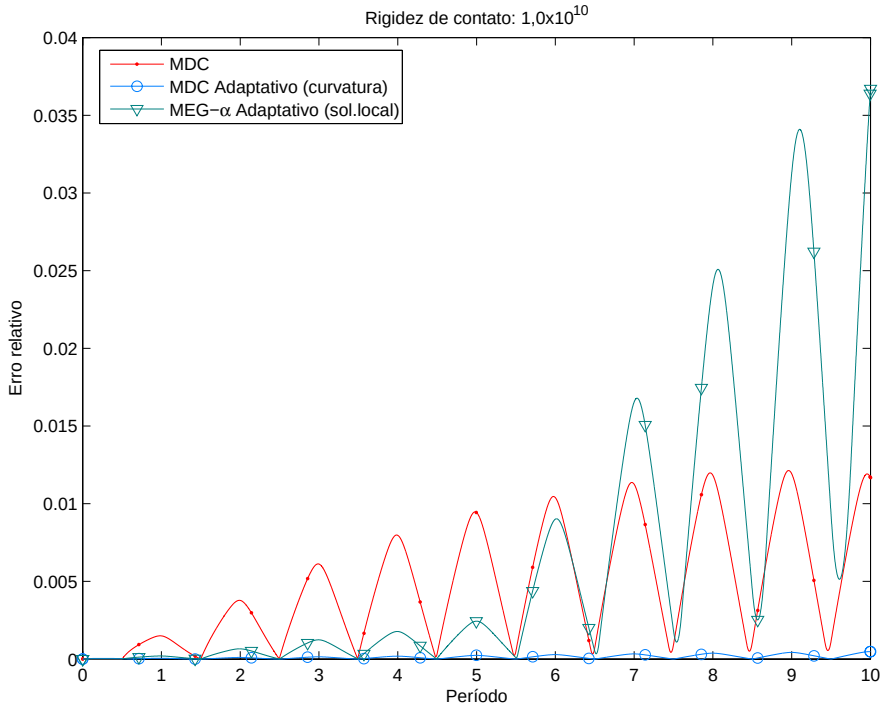


Figura 5.9: Histórico do erro relativo.

Conforme comentado anteriormente, o valor do erro verificado para a solução não-adaptativa nos instantes iniciais é bem pequeno. Entretanto, à medida que os choques com

a superfície vão ocorrendo, este valor cresce bastante, indicando que esta solução numérica diverge da solução analítica.

Quando um outro método de adaptação no tempo, o MEG-α Adaptativo, é empregado a divergência é inicialmente reduzida. Entretanto logo volta a crescer após sucessivos instantes de choques.

Os valores de erro para a solução adaptativa baseada em curvatura (MDC Adaptativo) continuam todavia abaixo de um patamar pequeno e sem tendência de crescimento brusco. Isso é possível uma vez que esta última solução faz um tratamento devido para a integração numérica nos momentos que circundam os instantes de contato (ver Seção 4.3.3). A Figura 5.10 retrata o histórico dos incrementos de tempo utilizado ao longo da simulação para esta solução é compara com a solução não-adaptativa.

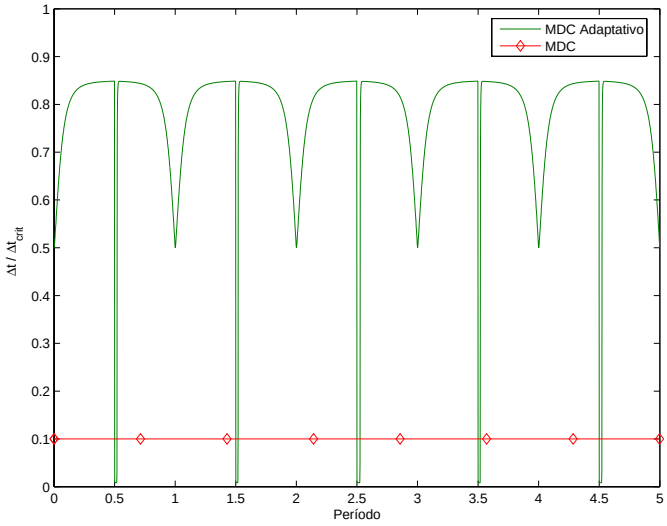


Figura 5.10: Histórico do incremento de tempo de integração.

Observe que, nos instantes onde ocorrem os choques da esfera com a superfície (instantes múltiplos de meio período), o incremento de tempo de integração para o algoritmo adaptativo tem seu valor bastante reduzido. Desta forma, assegura-se que estas ocasiões sejam tratadas de uma maneira mais apropriada, conforme comentado anteriormente. Isso no entanto só é possível uma vez que a solução adaptativa é dotada de um dispositivo que permite avaliar os instantes onde a simulação deve receber uma discretização temporal maior, a curvatura do histórico de deslocamentos. A Figura 5.11 apresenta o comportamento deste indicador geométrico ao longo da simulação.

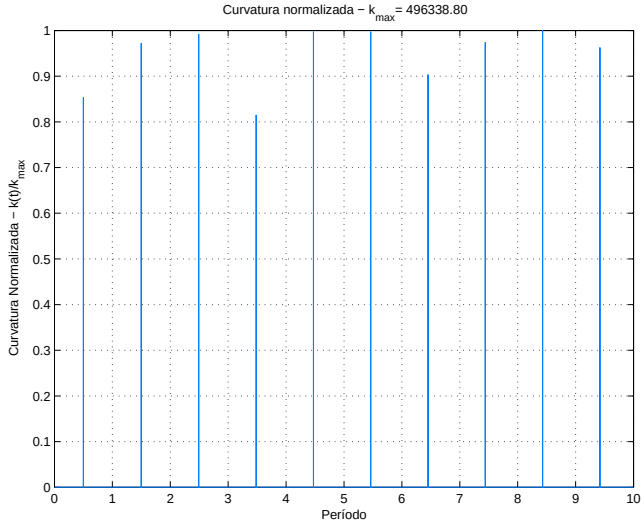


Figura 5.11: Histórico da curvatura.

Veja que a curva apresenta instantes onde a sua variação é bastante elevada. Estes correspondem exatamente aos instantes onde ocorrem os choques da esfera com a superfície, conforme já havia sido observado anteriormente na Figura 5.3. Estes instantes tambem são captados pelo estimador de refinamento utilizado no MEG-α Adaptativo, o erro local aproximado, ilustrado na Figura 5.12. Note que existe uma boa correlação entre o comportamento das funções que descrevem o histórico destes estimadores de refinamento.

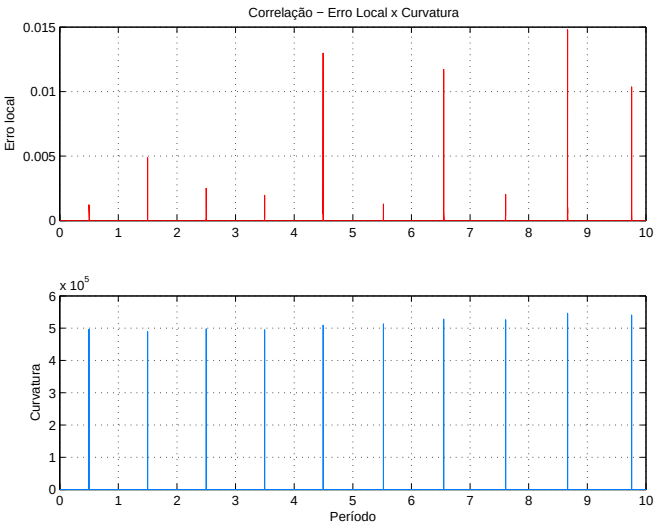


Figura 5.12: Correlação: Curvatura *versus* Erro Local.

O uso da curvatura em uma estratégia de controle automático do incremento de integração permite portanto que a resposta obtida seja mais apurada e obtida de uma forma mais otimizada. Em virtude do caráter instantâneo do fenômeno de contato, o uso de uma estratégia de adaptação com avaliação de resposta ao longo da simulação (*feedback*) faz-se necessária. Isso assegura que o valor do incremento de integração seja alterado antes do instante onde esta redução é

realmente necessária. Caso esse tratamento não houvesse sido realizado, o uso da técnica de adaptação poderia não trazer uma melhoria de resposta da maneira esperada. Para a análise em questão, o descarte de passos de integração ocorre nos pontos críticos da análise, os instantes de colisões.

Assim como o valor da curvatura do histórico é utilizado para a redução do incremento de tempo este também pode ser utilizado para aliviar a discretização temporal quando a análise assim permitir. Da mesma maneira que o incremento de tempo deve ser reduzido quando a curvatura se eleva ao longo da simulação, o mesmo também pode ser aumentado quando o valor da curvatura diminui. Isso ocorre até mesmo para o problema em estudo. A Figura 5.13 destaca de maneira mais aguçada a tendência da curvatura nos instantes que circundam o primeiro choque da esfera com a superfície.

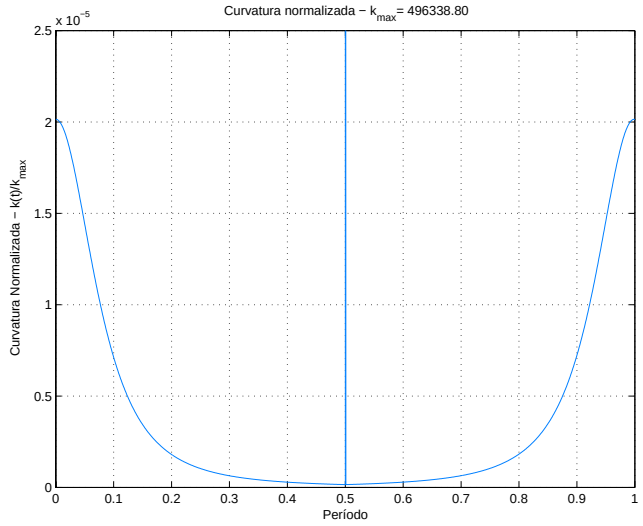


Figura 5.13: Tendência de crescimento e decrescimento da curvatura.

Nos instantes que antecedem o choque o valor da curvatura diminui. Isso acontece uma vez que nestes momentos o movimento da esfera não sofre variações em sua cinemática. Nestes instantes a tendência do corpo é aumentar indefinidamente a sua velocidade em virtude da queda livre que o mesmo está submetido. Essa tendência reflete-se no o valor da curvatura, que diminui. Este decrescimento é acompanhado pelo aumento do intervalo de tempo de integração (Δt), conforme pode ser observado na Figura 5.10 para o trecho de simulação correspondente. O incremento de integração varia de um certo valor inicial ($\Delta t_0 = 2{,}5 \times 10^{-6}$) até um valor máximo ($\Delta t_{max} = 5 \times 10^{-6}$) acompanhando a tendência de decrescimento da curvatura.

Um comportamento contrário é ainda verificado nos instantes posteriores ao choque. Isso ocorre uma vez que nestes instantes o movimento da esfera tende a ser contrário à ação gravitacional, e portanto com mudanças previstas em sua cinemática. A esfera tende a diminuir sua velocidade até o instante em que a mesma inverte o sentido do seu movimento

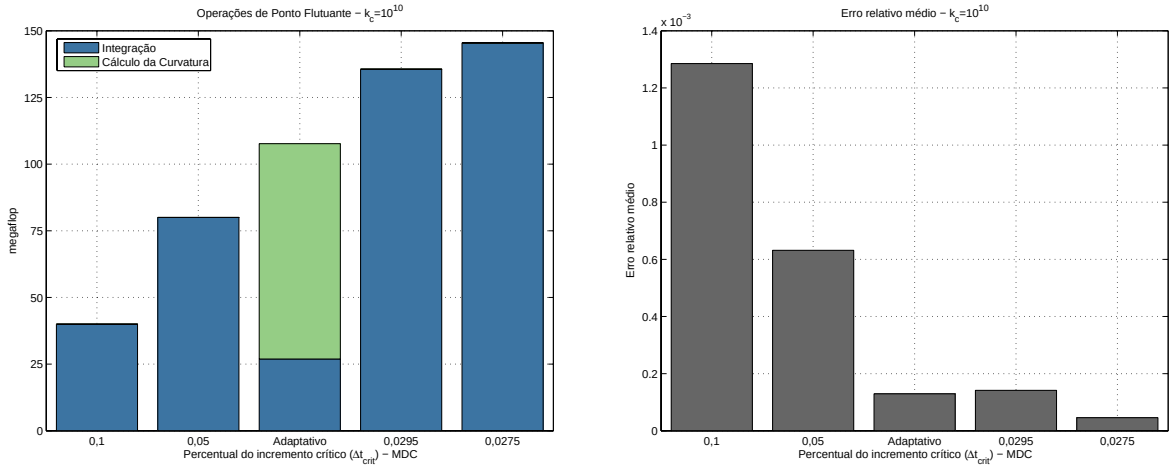
e começa a aumentar sua velocidade novamente. Todas estas tendências de mudança do movimento da esfera são retratadas no valor da curvatura de seu histórico de deslocamentos e conseqüentemente no processo de adaptação no tempo.

5.1.4 Estudo de Desempenho Computacional

Com o intuito de avaliar o desempenho computacional da técnica de adaptação no tempo, apresenta-se a seguir um estudo onde esta variável é alvo de investigação. O algoritmo adaptativo baseado em curvatura é confrontado com quatro casos de integração convencional, onde o valor do incremento de tempo é mantido constante. O objetivo é estabelecer relações entre duas variáveis relevantes em um processo de integração: o tempo de processamento e a qualidade da resposta. Em todas as análises realizadas emprega-se o Método das Diferenças Centrais para a tarefa de integração no tempo.

Utiliza-se como parâmetro de referência para a medição de desempenho o número de operações de ponto flutuante (*flop*) requerido pelos algoritmos para a tarefa de integração temporal. Este valor é diretamente proporcional ao tempo de processamento da simulação. Para a avaliação da qualidade de resposta faz-se o uso do valor médio do erro relativo em comprimento de arco observado durante todo o processo de simulação. Um algoritmo de integração ótimo é portanto aquele que consegue minimizar tanto o valor do erro como o número de operações.

A Figura 5.14 apresenta a performance do algoritmo adaptativo frente ao convencional para um caso de análise onde o valor da rigidez de contato é de $k_c = 10^{10} N/m$. Os casos não-adaptativos são ilustrados conforme o valor percentual do incremento de integração crítico Δt_{crit} empregado (10%, 5%, 2,95% e 2,75%).



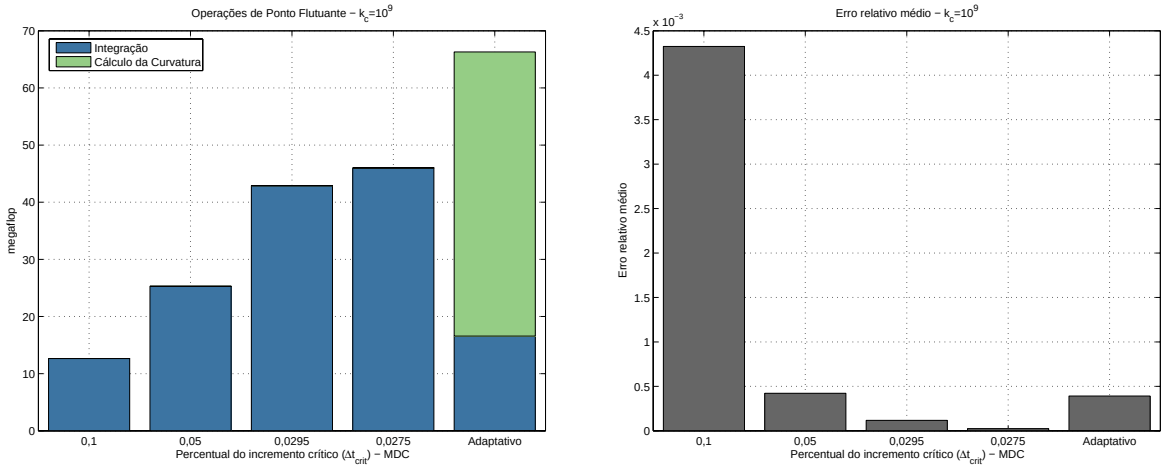
(a) Desempenho (b) Erro Relativo Médio

Figura 5.14: Performance do Algoritmo Adaptativo - $k_c = 10^{10} N/m$.

Note que na forma de integração convencional para que ocorra uma melhoria de qualidade de resposta é necessária a redução do incremento de tempo. Uma vez que a discretização temporal ocorre de maneira uniforme para este caso, o número de operações de ponto flutuante acaba crescendo com a redução do incremento, aumentando desta forma o tempo de processamento. O algoritmo adaptativo entretanto não segue esta regra. A qualidade da resposta obtida para este último depende apenas da robustez do método de adaptação empregado, ou seja, da maneira com que a relação qualidade-desempenho foi otimizada.

No caso da metodologia de adaptação baseada em curvatura, esta otimização está associada, dentre outros fatores, com a maneira com que este indicador geométrico relaciona-se com o incremento de tempo. Um aspecto que tem portanto influência nesta otimização é a forma da curva que relaciona curvatura com incremento (Seção 3.2). Para o mecanismo de adaptação até então utilizado, os melhores resultados de desempenho obtidos correspondem aos casos onde o valor da rigidez de contato k_c é elevado. Para estes casos a forma exponecial da curva incremento-curvatura utilizada se apresenta mais adequada.

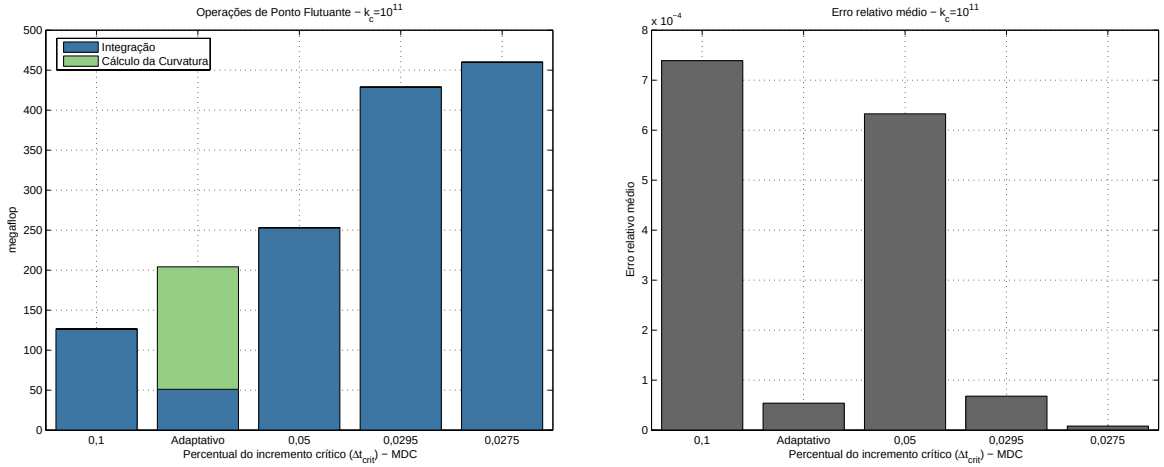
Note na Figura 5.15 que quando o valor da rigidez diminui para $k_c = 10^9 N/m$ o desempenho do algoritmo adaptativo diminui, embora o nível de qualidade de resposta continue comparável ao de um algoritmo convencional que utiliza um incremento correspondente a 5% do valor de Δt_{crit}. Observe ainda, que o algoritmo adaptativo realiza um número menor de operações em tarefas puramente de integração, que o correspondente algoritmo não adaptativo.



(a) Desempenho (b) Erro Relativo Médio

Figura 5.15: Performance do Algoritmo Adaptativo - $k_c = 10^9 N/m$.

Isso ocorre uma vez que a estratégia de adaptação permite que o valor de Δt aumente em instantes onde isto é possível, ver Figura 5.10. Este alívio no processo de integração conduz a melhorias de performance que ficam ainda mais evidentes quando o valor da rigidez de contato aumenta, conforme ilustra a Figura 5.16.



(a) Desempenho (b) Erro Relativo Médio

Figura 5.16: Performance do Algoritmo Adaptativo - $k_c = 10^{11} N/m$.

Note neste caso que o algoritmo adaptativo apresenta uma qualidade na resposta um pouco superior à do algoritmo convencional que usa um incremento correspondente a 2,95% de Δt_{crit}, gastando metade do tempo. O número de operações realizados em tarefas puramente de integração no tempo para este caso é cerca de 85% menor que a do algoritmo não adaptativo tomado como referência.

Conforme observado nos gráficos anteriores, o número de operações flutuantes apresentado

para o algoritmo adaptativo é composto de duas parcelas. Uma desta retrata as operações de ponto flutuante que são realizadas com o intuito de integração no tempo propriamente dita, ou seja, aquelas que são realizadas para a determinação dos dados de deslocamentos, velocidades e acelerações para o modelo numérico e um dado passo. A segunda parcela indica o número de operações efetuadas para o cálculo de um estimador de refinamento utilizado no mecanismo de adaptação, que para o caso em questão corresponde ao cálculo da curvatura do histórico de deslocamentos. A composição dos valores destas parcelas é feita tomando como base o número de graus de liberdade do modelo estudado, o número de passos de integração utilizado na análise, assim como o método de integração empregado.

A dimensão do problema tratado até então é bastante reduzida, uma vez que considera apenas o grau de liberdade que define a altura da particula em movimento. Em modelos numéricos onde o número de graus de liberdade envolvidos é bem mais elevado, a escolha do algoritmo de integração passa a ter um maior impacto sobre o desempenho computacional. A Figura 5.17 ilustra esta influência.

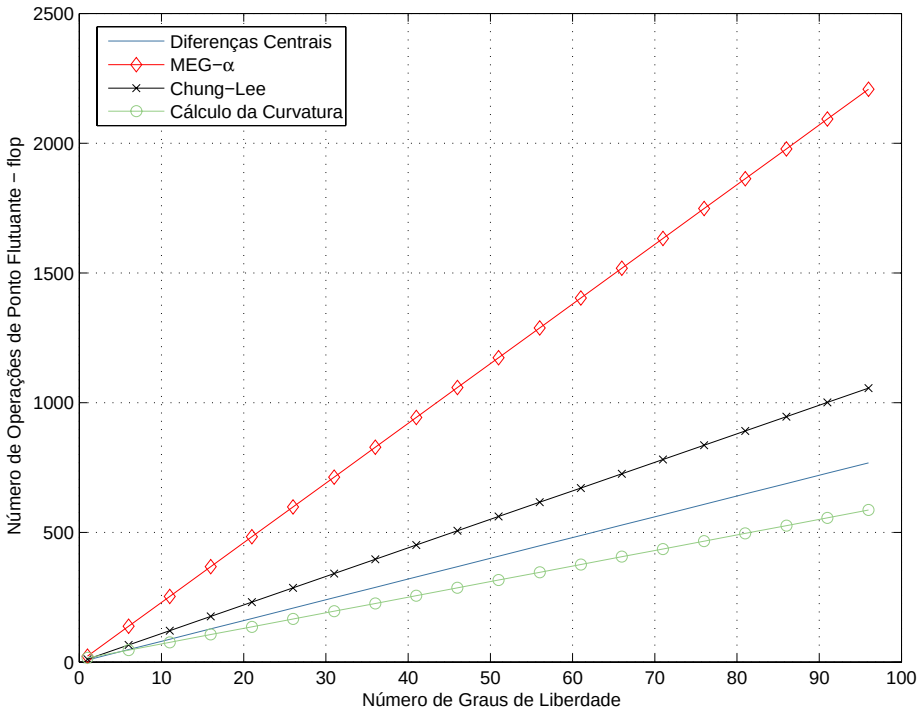


Figura 5.17: Número de operações de ponto flutuante por passo de integração.

Note que o algoritmo do MEG-α, para um dado modelo (um dado número de graus de liberdade), efetua mais cálculos durante o processo a integração de um passo que o clássico Método das Diferenças Centrais. Isso acontece em virtude do emprego dos seus amortecedores numéricos e preditores. A utilização de técnicas de adaptação para algoritmos mais robustos, tais como esse, permite que o ganho em desempenho seja portanto maior. Observe ainda que

a função que descreve o número de operações de ponto flutuante para o cálculo da curvatura tem uma inclinação menor que as dos métodos de integração. Ou seja, para modelos numéricos com uma discretização espacial mais refinada (maior número de graus de liberdade), o emprego da técnica de adaptação no tempo baseado em curvatura é potencialmente favorável ao desempenho computacional. Isso acontece uma vez este processo de estimação de refinamento passa a representar uma menor parcela de contribuição no total de cálculos efetuados.

5.2 Fraturamento Dinâmico - DyCOH

No exemplo a seguir, a técnica de adaptação no tempo proposta por este trabalho é utilizada na solução de um problema de fraturamento dinâmico. Para tanto, algumas implementações computacionais são desenvolvidas no intuito de acoplar um sistema existente, responsável por análises de problemas desta natureza, ao *framework* de integração adaptativa, apresentado na Seção 4.4.

O *software* utilizado como base para as implementações é o *DyCOH* (*Software for Simulation of Dynamic Cohesive Fracture*). Trata-se de um sistema computacional orientado a objetos que permite o tratamento unificado entre o Método dos Elementos Finitos (para pequenas ou grandes deformações) e modelos de Zona Coesiva (XU; NEEDLEMAN, 1993 apud Amorim, 2007). A versão apresentada por Amorim propõe a análise da dinâmica estrutural de problemas bidimensionais envolvendo ou não fraturamento.

A idéia básica do *sofware* é que, para um arquivo de dados fornecido como parâmetro de entrada, seja gerado um arquivo de saída para uma futura etapa de pós-processamento. O arquivo de entrada, quando compatível com o sistema, fornece todos os dados para que um modelo numérico seja gerado e uma análise de elementos finitos, envolvendo modelos coesivos e um processo de integração numérica temporal, seja realizada.

Na etapa de integração numérica temporal o *DyCOH* faz uso do clássico algoritmo das diferenças centrais. Embora este algoritmo em si não represente uma parcela considerável de processamento do programa, o mesmo encontra-se inserido em um processo que comanda toda a etapa de simulação, que é a integração temporal.

Conforme visto anteriormente, os métodos de integração direta permitem a obtenção do histórico de deslocamentos a partir do uso de técnicas de integração passo-a-passo. No caso específico dos métodos de integração direta explícitos, a exemplo do método das diferenças centrais, as operações realizadas em cada passo de integração não costumam representar grande parcela no total de esforço de processamento de um passo de integração. Olhando para uma escala um pouco maior, quando são considerados todos os passos de integração necessários para a caracterização de um dado intervalo de tempo, percebe-se que o esforço computacional está relacionado com o número de passos necessários para descrever o comportamento estrutural do corpo em estudo. Quanto maior o número de passos de integração, maior o esforço computacional demandado. Quando técnicas de adaptação no tempo são empregadas neste processo, pode-se de certa forma melhorar o impacto da etapa de integração nos custos computacionais envolvidos.

A técnica de integração no tempo utilizada no *DyCOH* não contempla um mecanismo que altere o número de passos de integração à medida que as respostas dos passos vão sendo obtidas. Isso significa que o intervalo de tempo de integração (Δt) se mantém constante durante toda a análise, não havendo portanto a adaptação deste valor à resposta do sistema. O controle da relação entre desempenho e qualidade de resposta é feito de forma prescrita, conforme comentado na Seção 2.3. A implementação de técnicas de adaptatividade permite portanto ao *software* uma maior flexibilidade quanto a este aspecto.

5.2.1 Aspectos Computacionais

Na arquitetura do sistema apresentado por Amorim a etapa de solução responsável pela tarefa de integração no tempo está concebida na classe *TimeIntegration*, conforme pode ser observado no diagrama de classes da Figura 5.18.

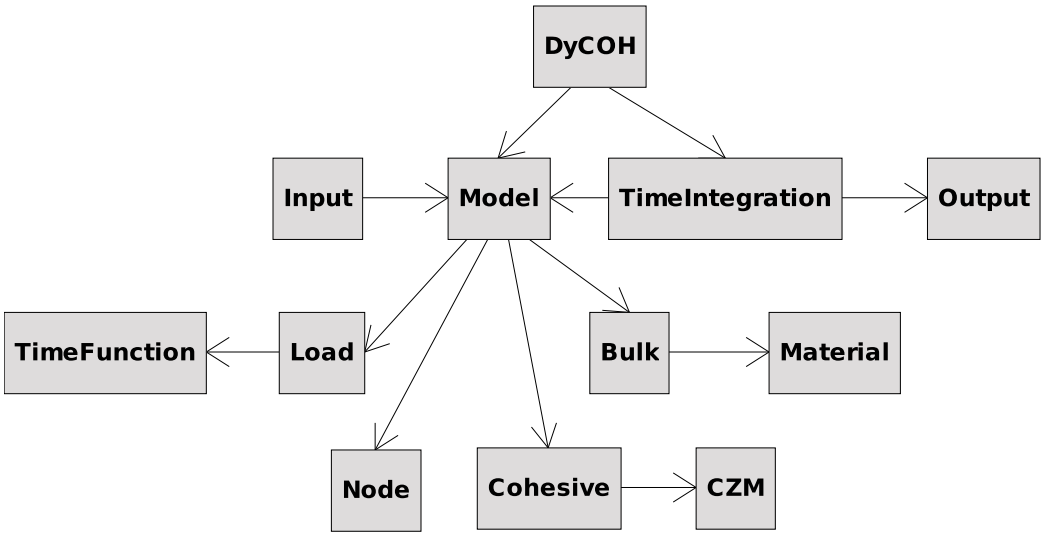


Figura 5.18: Diagrama de Classes - DyCOH.

A eventual incorporação de funcionalidades relacionadas à tarefa de integração no tempo, seja ela adaptativa ou não, a priori deve ser realizada nesta classe. Uma das formas que isto pode ser feito é através da alteração direta do escopo do código implementado neste local. Desta maneira, toda a técnica empregada no processo de integração teria que ser convertida em código para que a mesma pudesse ser utilizada. Isto demandaria um esforço de programação que possivelmente teria que ser continuado quando novas funcionalidades tivessem que ser incorporadas ao programa.

Uma outra maneira de incorporar recursos, algoritmos e técnicas de integração, é fazendo uso de um sistema desenvolvido de forma independente. Desta forma, o programa passa

a comunicar-se com este sistema, de maneira a utilizar suas funcionalidades. O *FASTTI*, apresentado na Seção 4.4, propõe este tipo de interação de sistemas. Conforme visto, a comunicação entre os dois sistema acontece através de uma interface. Uma vez que esta interface de comunicação esteja estabelecida, todas os recursos embutidos no *framework* são automaticamente herdados pela aplicação que faz o seu uso. No caso do *software DyCOH* a concepção desta interface pode ser realizada no local correspondente à classe *TimeIntegration* (Figura 5.18). A Figura 5.19 apresenta uma proposta para a nova estrutura de classes do programa.

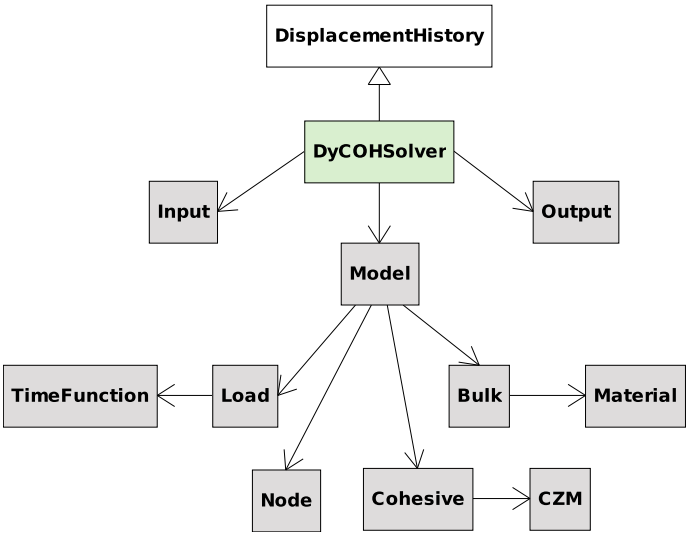


Figura 5.19: Módulo de interface do DyCOH com o *framework FASTTI*.

Observe que esta proposta pouco afeta a organização de classes do programa em si. Na prática, a implementação desta nova arquitetura consistiu apenas na criação da interface (*DyCOHSolver*) com o *framework* (*DisplacementHistory*). Todas as demais classes do programa foram aproveitadas, sem que houvesse a necessidade de alteração em qualquer parte do código.

5.2.2 Descrição do Problema e Validação das Implementações

Com o intuito de validar a implementação da nova arquitetura de classes, o exemplo apresentado por Amorim (2007) é novamente reproduzido. Trata-se da análise, em estado plano de tensões, do comportamento estrutural de um tirante de largura L e altura H, sujeito a um campo de força prescrita $P(t)$, conforme ilustrado na Figura 5.20.

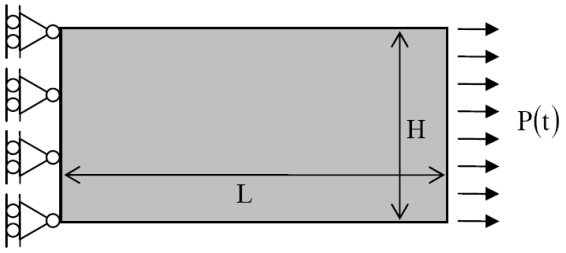


Figura 5.20: Condições de contorno do tirante.

As condições de contorno do problema foram definidas de maneira a impedir o movimento horizontal da extremidade esquerda do tirante. A carga atuante sobre o tirante, representada pelo campo $P(t)$, tem um histórico de intensidade definido pela função rampa:

$$P(t) = \begin{cases} P_0\, t/t_r & \text{para } t \leq t_r \\ P_0 & \text{para } t > t_r \end{cases} \tag{5.6}$$

onde $P_0 = 10^8 kN/m$ e $t_r = 10^5 \mu s$, correspondem aos valores da carga máxima e o tempo de crescimento da função rampa, respectivamente. Para as condições iniciais do problema dinâmico são especificados campos de deslocamentos e velocidades nulos.

O modelo numérico empregado (Figura 5.21) é composto por uma malha de 8 elementos finitos triangulares do tipo T3, 9 elementos coesivos e 24 nós, em um total de 48 graus de liberdade.

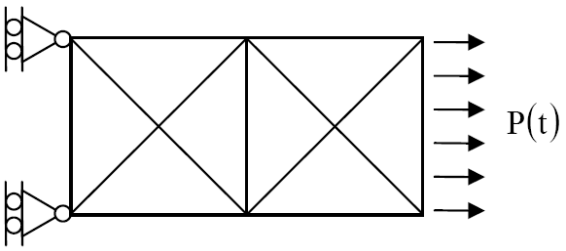


Figura 5.21: Modelagem numérica do problema.

O material constituinte do tirante é o *Polimetilmetacrilato*, material frágil e com propriedades apresentadas por XU & NEEDLEMAN (apud Amorim, 2007). A Tabela 5.2 resume as principais propriedades do material, bem como do modelo de fratura coesiva empregado.

Tabela 5.2: Propriedades do material do tirante e do modelo de fratura coesiva

Propriedade	Valor
Módulo de Elasticidade Longitudinal	3,24 GPa
Coeficiente de Poisson	0,35
Densidade Específica	1,190 kg/m^3
Tenacidade	352,3 N/m
Tensão normal máxima	324 MPa
Tensão tangencial máxima	755,4 MPa

Uma vez que o modelo numérico utilizado é o mesmo apresentado por Amorim, é de se esperar que a implementação da nova arquitetura de classes não altere os resultados obtidos. Isso é o que de fato ocorre quando se reproduzem os mesmos parâmetros do algoritmo de integração temporal no sistema que incorpora a nova estrutura de classes. As versões tomadas como referência para as comparações são aquelas com estruturas de classes apresentadas nas Figuras 5.18 e 5.19.

Na prática, a comparação entre os resultados obtidos para as duas versões do *software* é feita tomando como base os arquivos de saída de dados geradas por ambas versões.Assim feito, pôde-se observar que estas geravam arquivos de saída idênticos para um mesmo arquivo de dados de entrada.

A performance do novo sistema também é confrontada. A Tabela 5.3 apresenta os tempos de processamento do sistema com a nova arquitetura para diversas configurações de *hardware*. O valor do incremento de tempo utilizado nestas análises foi mantido constante e igual a $\Delta t = 10^{-9}s$ durante um tempo de simulação $t_{sim} = 6 \times 10^{-4}s$. O algoritmo empregado na solução é o do Método das Diferenças Centrais.

Tabela 5.3: Performance da nova versão do DyCOH em algumas configurações de *hardware*

Processador	Clock	RAM	Compilador	Arquitetura	Tempo Gasto (s)
Intel Pentium 4	3.26 Ghz	1GB	GCC 4.2.3	32 bits	20,98
Intel Pentium D	3.40 Ghz	1GB	GCC 4.2.3	32 bits	20,35
Intel Core 2 Duo	1.86 Ghz	1GB	GCC 4.2.1	64 bits	20,67
Intel Core 2 Duo	1.86 Ghz	1GB	Intel 10.1	64 bits	11,34
Intel Core 2 Duo	2.66 Ghz	1GB	GCC 4.2.3	32 bits	17,15
Intel Core 2 Duo	2.66 Ghz	2GB	GCC 4.2.1	64 bits	9,19
Intel Xeon E5320	1.86 Ghz	2GB	GCC 4.2.3	64 bits	13,16
Intel Xeon E5320	1.86 Ghz	2GB	Intel 10.1	64 bits	12,31

A configuração apresentada no primeiro dado da tabela foi reproduzida no intuito de comparar os tempos de processamento para as duas versões do *software DyCOH*. O tempo de

processamento da versão original do *software* é de 21 segundos, valor aproximadamente igual ao obtido para a nova arquitetura (20,98 segundos) na mesma configuração. Este dado informa que, a nova arquitetura do *software* proposta não teve influencia no tempo de processamento. Apesar da adição de todo um conjunto de classes (*framework*) ao sistema, não se observa perda de desempenho.

Este dado também é valioso para a paralelização computacional do processo de integração no tempo. Uma vez que o tempo gasto em ambas versões do *software* é o mesmo, espera-se que na nova arquitetura o tempo de processamento diminua quando os recursos de paralelismo disponíveis estejam em uso. Isso é possível uma vez que a tarefa de integração no tempo pode ser dividida entre processos, diminuindo portanto o seu tempo total de execução.

Deve-se tomar cuidado especial quanto à divisão de tarefas entre processos. É comum que o simples fato de dividir tarefas resulte em perda de desempenho computacional, sobretudo quando a tarefa dividida para cada processo é pequena. A Tabela 5.4 apresenta uma classificação das chamadas de funções com maior influência no tempo de processamento da análise aqui realizada.

Tabela 5.4: *Profile*: Diferenças Centrais + Integracao Padrão

Porcentagem Total	Tempo Gasto (s)	Método
32,68	1,83	DyCOH::MatrixLib::ProductMatrixVector
31,52	1,77	DyCOH::XuNeedleman::EvaluateTractions
6,61	0,37	DyCOH::COH4::GetBMatrix
5,89	0,33	DyCOH::COH4::GetCosines
3,75	0,21	•*FASTTI*::CentralDifference::ComputeDisVelAcc
2,86	0,16	DyCOH::T3::GetJacobianRST
2,32	0,13	DyCOH::COH4::GetCohesiveForces
2,14	0,12	DyCOH::Model::GetResidualForces
1,96	0,11	DyCOH::MatrixLib::SumDoubleVector
1,43	0,08	DyCOH::T3::GetBMatrix
0,98	0,06	DyCOH::T3::GetDerivationXY
0,98	0,06	DyCOH::T3::GetInternalForces
0,89	0,05	DyCOH::LimitedPiecewise::EvaluateFunction
0,71	0,04	DyCOH::T3::AssemblyGlobalVector
0,71	0,04	DyCOH::T3::GetElementDisplacement
0,54	0,03	•*FASTTI*::DyCOHSolver::UpdatePrescribedDOFs
0,54	0,03	DyCOH::XuNeedlemanUnloading::Getdelta_t
0,54	0,03	DyCOH::COH4::AssemblyGlobalVector
2,67	0,15	demais chamadas

Note que o cálculo dos deslocamentos, velocidades e acelerações (*FASTTI*::CentralDifference ::ComputeDisVelAcc) pouco representa em tempo de processamento para o problema avaliado.

A paralelização computacional da tarefa de integração no tempo apenas é válida quando se manipula um número de graus de liberdade elevado (milhares ou milhões, dependendo do algoritmo de integração utilizado). No problema em estudo o número de graus de liberdade, ou seja, a dimensão do vetor que deve ser integrado no tempo, é apenas 48. Para esta dimensão o tempo de criação de processos influencia negativamente na performance computacional.

Apesar da Tabela 5.3 apresentar configurações de *hardware* com processadores de múltiplos núcleos, apenas um núcleo foi utilizado para a análise do problema em curso. Os dados da tabela revelam apenas a influência da própria arquitetura do processador e do compilador utilizado. Os melhores resultados de performance são obtidos em máquinas 64 bits, indicando uma boa aplicabilidade do *software* a esta plataforma.

5.2.3 Adaptação no Tempo

Os resultados de análises obtidos até então em nada diferem daqueles obtidos por Amorim (2007). Conforme comentado anteriormente, os mesmos são apenas reproduzidos no intuito de validar a implementação de uma nova estrutura de classes para o *software DyCOH* que incorpora os recursos do *framework FASTTI*. Já que esta validação foi verificada, então o sistema de análise de modelos coesivos agora dispõe de mecanismos que permitem que o seu processo de integração no tempo seja mais flexível. A nova arquitetura de classes permite, por exemplo, que a integração no tempo ocorra de forma adaptativa, e não apenas da forma convencional, com o incremento de tempo constante, como até então era a prática.

A seguir o mesmo exemplo de ruptura do tirante é analisado utilizando-se uma estratégia de integração adaptativa baseada na curvatura do histórico de deslocamentos. A idéia do estudo é apresentar aspectos relacionados com a influência do emprego desta forma de integração no desempenho computacional, bem como na qualidade da resposta obtida para a análise. Antes que isso seja feito é necessário entender de que forma o indicador geométrico curvatura é utilizado no controle do incremento de tempo para este exemplo em particular.

Embora a formulação envolvida na análise de elementos coesivos seja sofisticada, a dinâmica do tirante que está sendo estudada é bastante simples. Inicialmente este elemento estrutural encontra-se em repouso, uma vez que as condições iniciais prescritas para o problema impõem campos de deslocamentos e velocidades nulos. Gradativamente este estado vai alterando-se, já que a intensidade da carga aplicada em sua extremidade aumenta com o decorrer do tempo (ver Figura 5.20). O aumento da carga é automaticamente traduzido em deslocamentos nodais e aumento das tensões nas interfaces entre os elementos finitos. Este aumento de tensões acontece até o instante em que o valor limite de resistência do material é

atingido e o tirante se rompe. As porções do tirante após a fratura passam a ter cinemáticas distintas. Uma parte do tirante passa a se deslocar sempre no sentido de aplicação da força e a outra porção do começa a apresentar um movimento oscilatório, semelhante a uma mola em vibração livre.

Analisando o histórico de deslocamentos do modelo, três momentos bem representativos estão definidos. O primeiro momento corresponde ao intervalo de tempo que vai do início da análise até o instante do rompimento do tirante. Neste intervalo os deslocamentos nos graus de liberdade nestes momentos variam bastante em decorrência das constantes interações entre elementos finitos através dos elementos coesivos. Esta variação reflete-se no comportamento da função que descreve a curvatura do histórico de deslocamentos.

O segundo momento corresponde ao instante da ruptura do tirante. Neste instante, mudanças bruscas nas condições de contorno do problema acontecem. Elementos coesivos que faziam a ligação entre elementos finitos vizinhos passam a não existir mais, deixando de transmitir a carga aplicada em uma extremidade do tirante a extremidade oposta

O terceiro momento corresponde ao intervalo de tempo que é posterior ao instante do fraturamento. O corpo, anteriormente constituído de uma única porção, passa a ser composto por duas porções com comportamento de deslocamentos distintos.

A Figura 5.22 retrata estes três momentos distintos da simulação. Trata-se do comportamento temporal da curvatura do histórico de deslocamentos nos graus de liberdade do modelo numérico, ilustrado na Figura 5.21.

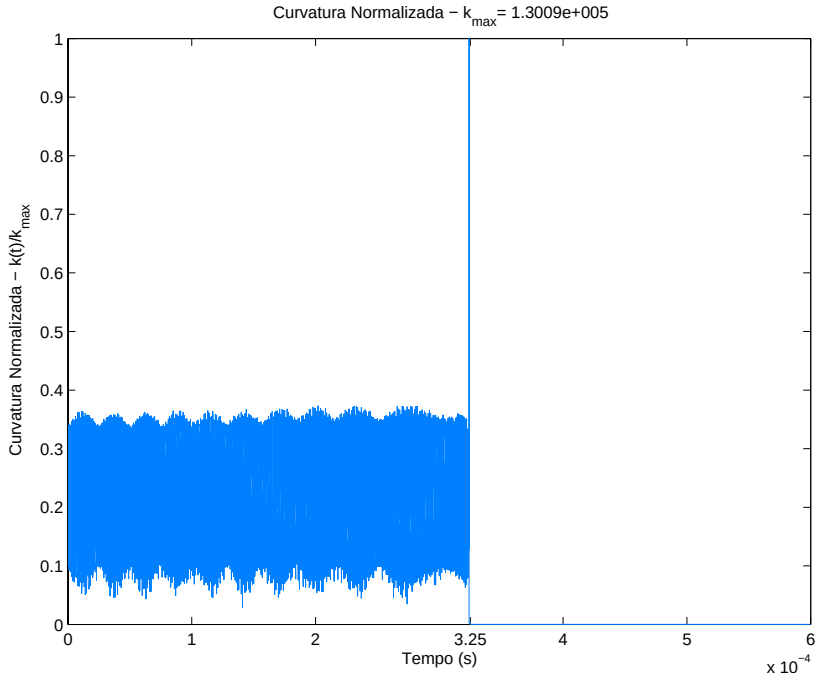


Figura 5.22: Histórico da curvatura.

Observe que até a iminência da ruptura ($t = 325\mu s$), o valor da curvatura oscila bastante. Esta oscilação é reflexo das interações entre os elementos finitos e coesivos, citadas anteriormente. No instante em que o tirante rompe a curvatura atinge o seu máximo valor, indicando a separação de elementos e uma mudança brusca nas condições de contorno. Após a ruptura, as porções que compõe o volume do tirante passam a apresentar um comportamento dinâmico distinto do até então desenvolvido. As mesmas passam a vibrar em uma freqüência mais baixa, o que acaba por diminuir o valor da curvatura.

O valor da curvatura do histórico de deslocamentos pode ser utilizado no controle do incremento utilizado no processo de integração temporal. Para tanto faz-se necessário um tratamento desta informação antes da sua utilização. Conforme comentado nas seções anteriores, este valor pode ser utilizado diretamente em uma relação com o incremento de tempo a ser utilizado em um dado passo de integração qualquer. Para tanto é necessário que o comportamento da função que descreve a evolução da curvatura ao longo do tempo seja o mais próximo possível de uma função suave. Como em boa parte dos problemas é comum que isso não ocorra, então faz-se necessário um processo de regularização de curvatura. Desta forma as alterações feitas para a variável incremento de tempo são feitas tomando como base as seqüências regularizadas de curvatura. Isto torna as mudanças para o valor desta variável menos freqüêntes, mais controladas e favoráveis à convergência da resposta.

A Figura 5.23 confronta o histórico de curvatura obtido para o problema em estudo

com a sua correspondente seqüência regularizada. Para a obtenção da função regularizada utiliza-se o algoritmo do Máximo Valor por Intervalo, apresentado na Seção 3.3. O intervalo de regularização empregado neste exemplo é 10000 vezes maior que o valor incremento de integração máximo $\Delta t_{max} = 10^{-9}s$, tomado como referência no trabalho de Amorim (2007).

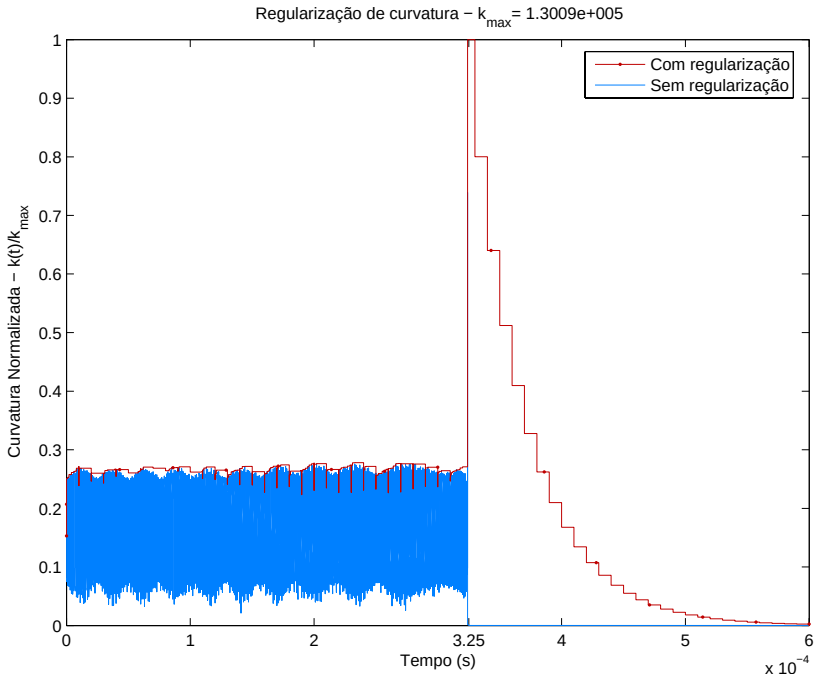


Figura 5.23: Histórico da curvatura regularizada.

O emprego desta técnica de regularização assegura que em um dado instante de tempo qualquer o valor da curvatura a ser utilizado no controle do incremento de tempo é sempre maior do que aquele que está ocorrendo de fato (sem a regularização). Isso significa que o valor de incremento obtido a partir da seqüência regularizada é sempre menor do que o de fato seria necessário, caso houvesse a relação direta sem o processo de regularização. A Figura 5.24 apresenta o histórico de incremento de tempo utilizado no processo de integração adaptativo baseada em curvatura. No gráfico, o valor do incremento é normalizado pelo valor do incremento de tempo de integração máximo, comentado anteriormente.

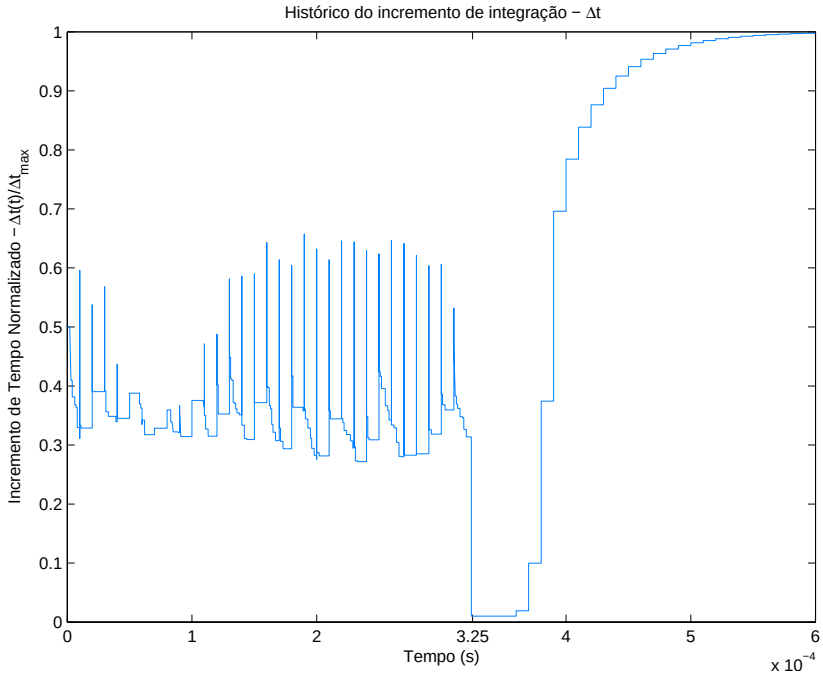


Figura 5.24: Histórico do incremento de integração temporal.

Note que o comportamento desta curva relaciona-se de maneira inversamente proporcional com a função de curvatura regularizada apresentada anteriormente. Os mínimos valores de incremento de tempo observados ocorrem imediatamente após o instante em que há a ruptura ($t = 325\mu s$), quando o valor da curvatura é máximo. Nos instantes em que a curvatura assume valores mais baixos, o incremento de tempo começa a crescer gradativamente até o valor máximo permitido.

O uso da curvatura na estratégia de adaptação no tempo permite identificar o comportamento da dinâmica do sistema e propor alterações para o incremento de tempo baseadas neste comportamento. Estas mudanças refletem-se tanto na qualidade da resposta obtida pelo processo de integração quanto no tempo gasto para a sua obtenção. A Tabela 5.5 apresenta o tempo de processamento para a solução do problema em estudo em diversas condições de análise. A idéia é confrontar a performance do algoritmo adaptativo com diversas situações de uso do algoritmo convencional, onde o incremento de tempo é mantido constante durante todo o processo de simulação. Em todos os casos de análise o Método das Diferenças Centrais é utilizado para a integração[1].

[1]Processador utilizado: Intel Core 2 Duo - 2.66 Ghz, Memória: 2GB, Compilador: GNU 4.2.1, Plataforma: 64 bits

Tabela 5.5: Tempo de processamento: Integração Adaptativa *versus* Convencional

Incremento utilizado	Tempo de processamento (s)
Δt_{max}	9,19
$0{,}5\Delta t_{max}$	18,41
$0{,}25\Delta t_{max}$	37,40
$0{,}125\Delta t_{max}$	73,23
Adaptativo	84,65
$0{,}109\Delta t_{max}$	84,81
$0{,}01\Delta t_{max}$	919,98

Observe que para o algoritmo não-adaptativo existe uma relação linear entre o incremento de tempo utilizado para a integração numérica e o tempo de processamento. Quando o incremento reduz em cem vezes, de Δt_{max} até $0{,}01\Delta t_{max}$ por exemplo, o tempo de processamento aumenta em cem vezes.

O algoritmo adaptativo no entanto não segue esta tendência linear, uma vez que ele assume diversos valores de incremento ao longo da simulação. Conforme visto na Figura 5.24, o valor do incremento varia entre Δt_{max} e $0{,}01\Delta t_{max}$. Quando o algoritmo adaptativo é comparado ao convencional, vê-se que o mesmo equivale em performance a um onde o incremento de tempo utilizado é $0{,}109\Delta t_{max}$. Isso quer dizer que qualquer integração que seja feita com um incremento constante e menor que $0{,}109\Delta t_{max}$ terá menor performance que o adaptativo.

A vantagem de utilização do algoritmo adaptativo reside portanto no fato que o incremento de integração não precisa necessariamente ser mantido em um patamar baixo durante toda a análise. Essa característica é o que permite que um algoritmo adaptativo possa ser mais eficiente em tempo de processamento. A Figura 5.25 apresenta o histórico do número de operações de ponto flutuante (*flop*) efetuados ao longo da simulação para as correspondentes configurações de análise apresentadas na Tabela 5.5.

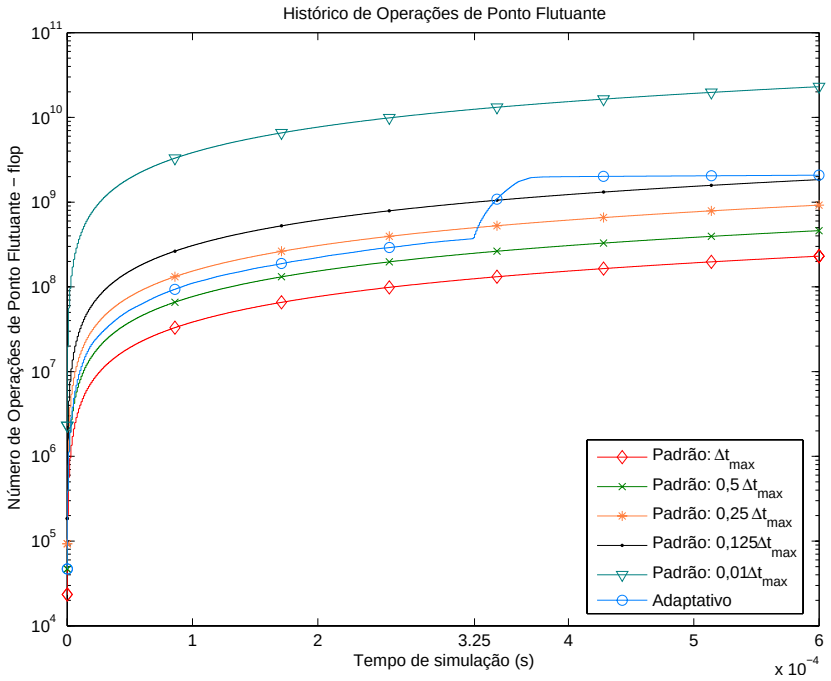


Figura 5.25: Número de operações de ponto flutuante.

Observe que o comportamento da curva correspondente ao algoritmo adaptativo tem uma forma diferente das demais. O salto observado no instante $t = 325\mu s$ corresponde ao instante em que o incremento de integração foi reduzido em virtude da mudança brusca que ocorreu no sistema devido à ruptura do tirante.

Convém lembrar que embora estejamos tratando de eficiência, e medindo isto em tempo de processamento, a utilização deste termo é relativa. Uma integração que utiliza incrementos pequenos demanda um tempo de processamento maior, no entanto a resposta obtida é melhor. Uma comparação mais justa de eficiência entre os algoritmos de integração convencional e adaptativo ocorreria se a qualidade da resposta obtida por ambos fosse a mesma.

Um outro fator que influencia a eficiência de um algoritmo adaptativo é o processo de estimação de refinamento. Conforme comentado anteriormente, esta etapa deve consumir o mínimo de recurso computacional (Bergan; Mollestad, 1985). Desta forma o ganho que se observa na utilização de incrementos de tamanhos variáveis não é prejudicado pelo tempo gasto na tarefa de estimar o incremento de integração.

O uso da curvatura como um estimador de refinamento neste exemplo não tem impacto significativo no tempo de processamento da análise adaptativa. Isso pode ser observado na Tabela 5.6, que classifica as chamadas de funções com maior consumo de tempo de processamento.

Tabela 5.6: Profile: Diferenças Centrais + Adaptação baseada em curvatura

Porcentagem Total	Tempo Gasto (s)	Método
32,23	9,47	DyCOH::MatrixLib::ProductMatrixVector
28,53	8,39	DyCOH::XuNeedleman::EvaluateTractions
5,99	1,76	DyCOH::COH4::GetBMatrix
4,93	1,45	DyCOH::COH4::GetCosines
4,70	1,38	•*FASTTI*::CentralDifference::ComputeDisVelAcc
4,41	1,30	DyCOH::T3::GetJacobianRST
1,94	0,57	DyCOH::COH4::GetCohesiveForces
1,87	0,55	DyCOH::MatrixLib::SumDoubleVector
1,80	0,53	DyCOH::Model::GetResidualForces
1,63	0,48	•*FASTTI*::DisplacementHistory::_EstimateCurvature
1,21	0,36	DyCOH::T3::GetInternalForces
1,04	0,31	DyCOH::T3::GetBMatrix
0,95	0,28	DyCOH::COH4::AssemblyGlobalVector
0,92	0,27	DyCOH::T3::GetDerivationXY
0,70	0,21	DyCOH::LimitedPiecewise::EvaluateFunction
0,65	0,19	DyCOH::MatrixLib::VectorPrintf
0,58	0,17	DyCOH::T3::GetDerivationRS
0,58	0,17	DyCOH::COH4::GetElementDisplacements
0,56	0,17	DyCOH::T3::ComputeElementStrainStress
0,56	0,17	DyCOH::MatrixLib::ProductEscalarVector
0,46	0,14	DyCOH::T3::AssemblyGlobalVector
6,06	1,78	demais chamadas

Note que o cálculo da curvatura (*FASTTI*::DisplacementHistory::_EstimateCurvature) apenas representa 1,63% do tempo computacional total de análise. Valor semelhante também é obtido quando, ao invés de utilizar o Método das Diferenças Centrais para a integração, utiliza-se o Método Explícito Generalizado-α (ver Tabela 5.7).

Tabela 5.7: Profile: MEG-α + Adaptação baseada em curvatura

Porcentagem Total	Tempo Gasto (s)	Método
28,12	8,37	DyCOH::MatrixLib::ProductMatrixVector
26,66	7,94	DyCOH::XuNeedleman::EvaluateTractions
9,34	2,78	•*FASTTI*::MegAlpha::ComputeDisVelAcc
5,27	1,57	DyCOH::COH4::GetBMatrix
4,84	1,44	DyCOH::COH4::GetCosines
3,90	1,16	•*FASTTI*::MegAlpha::ComputeFdesMass
3,31	0,99	DyCOH::T3::GetJacobianRST
2,05	0,61	DyCOH::MatrixLib::SumDoubleVector
1,75	0,52	DyCOH::Model::GetResidualForces
1,68	0,50	•*FASTTI*::DisplacementHistory::_EstimateCurvature
1,44	0,43	DyCOH::T3::GetBMatrix
1,41	0,42	DyCOH::COH4::GetCohesiveForces
1,14	0,34	DyCOH::T3::GetInternalForces
1,08	0,32	DyCOH::COH4::AssemblyGlobalVector
0,74	0,22	DyCOH::T3::AssemblyGlobalVector
0,74	0,22	DyCOH::COH4::GetElementDisplacements
0,71	0,21	DyCOH::T3::GetDerivationXY
0,67	0,20	•*FASTTI*::DyCOHSolver::UpdatePrescribedDOFs
0,60	0,18	DyCOH::MatrixLib::VectorPrintf
4,50	1,34	demais chamadas

Uma vez que o MEG-α efetua mais operações por passo de integração que o Método das Diferenças Centrais, já que faz uso de preditores e amortecedores numéricos, espera-se que o emprego de uma técnica de adaptação para este método conduza a uma maior melhoria de performance.

De uma forma geral o exemplo de fraturamento apresentado anteriormente permite avaliar diversos aspectos relacionados ao emprego de uma técnica de adaptação no tempo em um *software* previamente existente, o *DyCOH*.

A partir da implementação realizada e que define a interface do *software* com o *framework* todos os futuros desenvolvimentos para este último são automaticamente compartilhados. Inicialmente, o *framework* acrescenta ao *software DyCOH* técnicas de integração previamente não concebidas, tais como os algoritmos de Chung & Lee (1994), Hulbert & Chung (1996), os mecanismos de integração adaptativos e as técnicas de computação de alto desempenho.

5.3 Linhas de Ancoragem - Preadyn++

O estudo de linhas de ancoragem sob o ponto de vista estrutural tem sido alvo de diversas pesquisas, sobretudo motivadas pela indústria de petróleo quando na exploração deste recurso em água ultraprofundas. De uma forma geral, esse modo de exploração requer a utilização de flutuantes, tais como o ilustrado na Figura 5.26. Esses flutuadores são sujeitos às mais diversas condições ambientais, que envolvem fatores tais como ventos, ondas e correntes marítimas. Faz-se necessário portanto o estudo do impacto dessas condições sobre os diversos elementos estruturais que compõem os arranjos *offshore*.

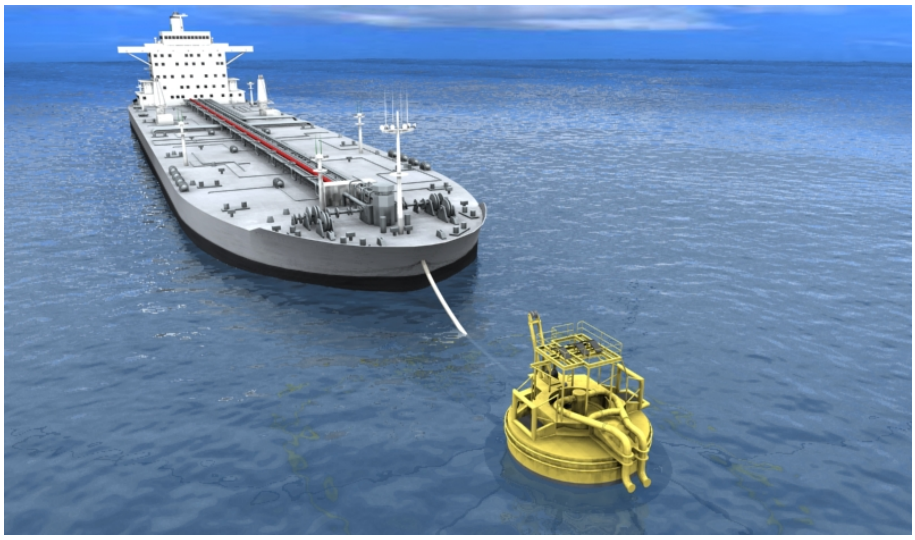


Figura 5.26: Exploração de petróleo em águas ultraprofundas (Fonte: TPN/USP).

As linhas de ancoragem são responsáveis em grande parte pela estabilidade dos flutuantes. Estas absorvem as intempéries atuantes na estrutura submersível e as transmite para o solo marinho, situado em alguns casos a milhares de metros do nível do mar. O estudo do comportamento estrutural dinâmico de sistemas embarcados tem sido objetivo de alguns *softwares*, a exemplo do *Dynasim* (Coelho *et al.*, 2001). O *software* engloba alguns sistemas computacionais, dentre os quais o *Preadyn*++ (Ferreira, 2005).

O *Preadyn*++ é um *framework* orientado a objetos, desenvolvido em C++, que contempla a análise dinâmica de linhas de ancoragem e *risers*. Com esse objetivo, o sistema faz uso do método dos elementos finitos, para a discretização espacial destas linhas, assim como de métodos de integração direta, para a determinação do comportamento ao longo do tempo destas entidades.

Os algoritmos de solução implementados no sistema *Preadyn*++ até então não fazem uso de estratégias adaptativas temporais em suas simulações. A tarefa de integração ao longo do

tempo para este sistema está encapsulada em uma classe específica, que pode assumir várias formas dependendo do algoritmo de solução empregado, mas essa considera que o incremento de integração temporal Δt mantem-se constante durante toda a análise.

Visando investigar o emprego de técnicas adaptativas no sistema *Preadyn++*, apresenta-se a seguir um exemplo de análise de linhas de ancoragem. De maneira análoga ao *software Dycoh*, apresentado na Seção 5.2, são realizadas implementações computacionais que permitam a fusão de dois sistemas computacionais distintos. No caso em questão esses correspondem ao *framework Preadyn++*, responsável pela análise espacial das linhas, e o *framework* FASTTI, atuante no processo de integração numérica temporal.

O exemplo consiste em reproduzir a operação de instalação de linhas denominada *hookup*. Nesse procedimento, um trecho de linha, inicialmente suspenso por dois navios "A" e "B", tem uma de suas extremidades liberada subitamente, conforme ilustrado na Figura 5.27.

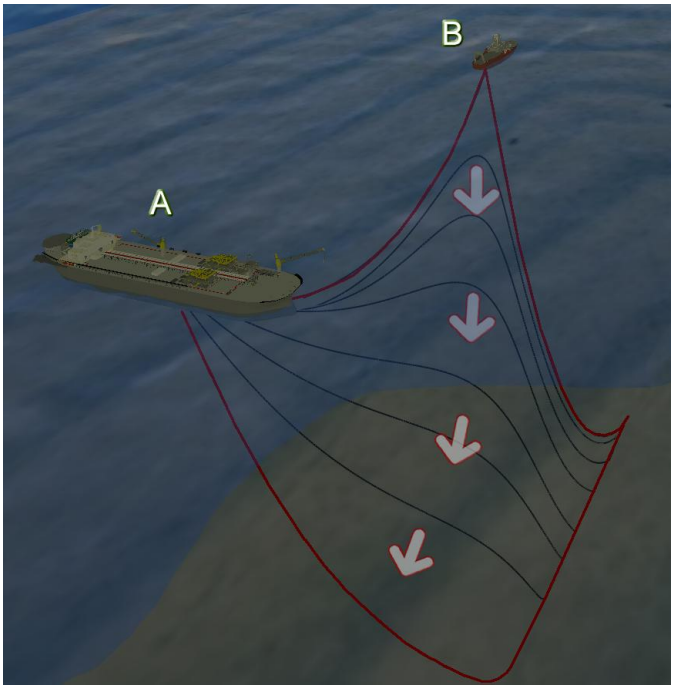


Figura 5.27: Esquema de lançamento da linha de ancoragem.

O momento da liberação da linha de ancoragem pelo navio situado em "B" é o momento mais crítico da operação. Do ponto de vista de modelagem numérica, a liberação da linha pelo rebocador corresponde a um instante de variação brusca nas condições de contorno do problema. Em geral, variações repentinas, tais como esta, contribuem mais intensamente em erro para uma dada discretização numérica. Uma solução adotada para amenizar esse problema consiste em reduzir o valor do incremento de integração (Δt).

Em algoritmos não adaptativos, essa redução é feita de forma global e portanto válida para

toda a simulação. Esta prática, apesar de ser mais simples, pode ter impacto negativo em tempo de processamento, uma vez que exige que uma quantidade maior de passos de solução seja obtida. Além disso, em muitos instantes da simulação pode-se utilizar um incremento de tempo maior sem que haja perda significativa da qualidade de resposta. Os algoritmos adaptativos atuam no sentido de identificar esses instantes e sugerir mudanças para o valor do incremento tentando aperfeiçoar a relação qualidade-desempenho. Dessa forma esses algoritmos permitem que níveis de discretização distintos sejam usados em uma análise sem que isso inviabilize a obtenção de uma resposta devido ao tempo de processamento demandado.

A seguir apresenta-se um estudo onde o algoritmo adaptativo baseado em curvatura é empregado na simulação numérica do procedimento de *hookup*. O objetivo é investigar aspectos relacionados com o desempenho computacional e com a qualidade da resposta numérica. Os dados obtidos para o algoritmo adaptativo são confrontados com análises onde o valor do incremento de tempo é fixo durante toda a simulação. Em todas as análises utiliza-se uma única malha de discretização espacial, composta por elementos de treliça com comprimentos uniformes no domínio e igual a 5 metros. Os parâmetros geométricos e físicos do modelo numérico empregados são apresentados na Tabela 5.8.

Tabela 5.8: Propriedades geométricas e físicas do modelo numérico de *hookup*.

Propriedade	Valor
Distância entre as embarcações	500 m
Lâmina d'água	1000 m
Comprimento do trecho de linha suspenso	550 m
Comprimento total da linha	1620 m
Rigidez longitudinal da linha (EA)	1000 kN
Peso no ar (linha)	1,50 kN/m
Peso submerso (linha)	1,26 kN/m
Coeficiente de arrasto de Morison	1,2
Coeficiente de inércia de Morison	0,06
Coeficiente de atrito estático	0,50
Coeficiente de atrito dinâmico	0,05

A Figura 5.28 ilustra a relação entre o valor do incremento de integração e o tempo de processamento demandado na simulação numérica do procedimento de *hookup*. Utilizam-se diferentes níveis de discretização temporal, determinados com base no valor do incremento de integração crítico (Δ_{crit}), que garante a convergência para uma solução. Estas discretizações são uniformes, ou seja, adota-se um único incremento de integração ao longo de toda a análise para cada um destes níveis. Os dados são ainda comparados com o obtido fazendo uso de uma estratégia de adaptação no tempo baseada na curvatura do histórico de deslocamentos.

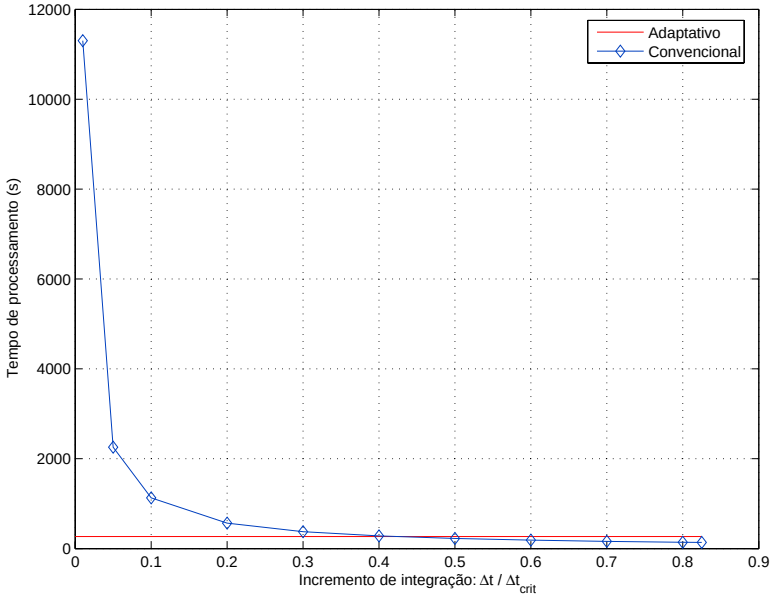


Figura 5.28: Comparação de tempo de processamento: convencional *versus* adaptativo.

Note que, para a estratégia de discretização convencional, o tempo de processamento está intimamente ligado com o valor do incremento de integração. O simples fato de reduzir o valor do incremento em dez vezes para este problema em particular aumenta, em proporcional escala, o tempo necessário para a obtenção da análise. Observe ainda que o algoritmo adaptativo tem o desempenho comparável a um convencional com malha uniforme de tamanho $\Delta t = 0{,}4 \times \Delta t_{crit}$.

Na Figura 5.29 é possível visualizar o histórico do incremento de tempo para o algoritmo adaptativo, assim como associar as mudanças para esta variável de acordo com o comportamento dinâmico do problema apresentado. Com este fim, apresenta-se o comportamento da força resultante atuante no topo da linha (embarcação) ao longo do tempo. Os dados de força são normalizados pelo valor máximo verificado na série temporal correspondente ao estágio final da análise, quando a linha está instalada.

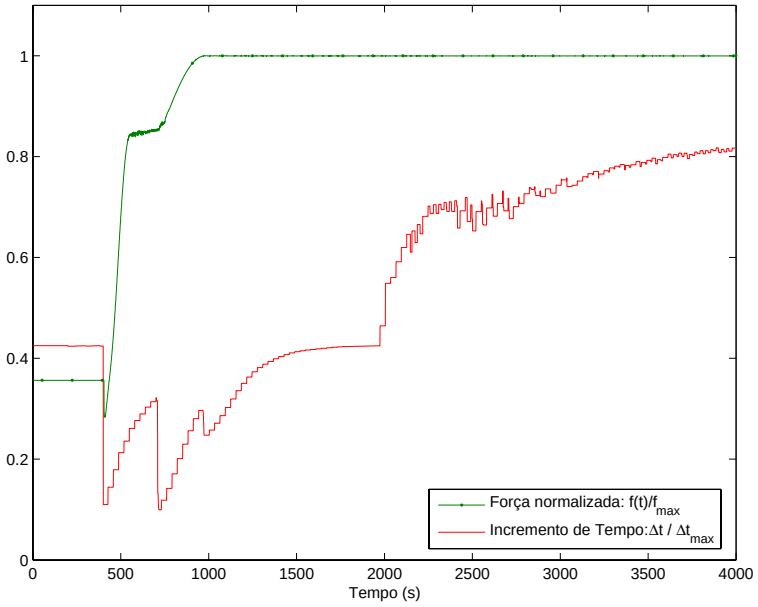


Figura 5.29: Correlação do incremento de tempo com o comportamento da força no topo.

Nos instantes iniciais da simulação, quando a linha ainda tem um trecho suspenso, o valor da força atuante no topo da linha (embarcação "A" - Figura 5.27) é constante e definido apenas pela contribuição de peso deste trecho. No instante em que a linha é liberada da embarcação "B" ($t = 400s$), o valor da força tem uma leve queda em função do movimento da linha, mas logo em seguida assume uma tendência de crescimento que logo se encerra, indicando o fim da instalação da linha.

São pontos cruciais da análise, em termos de erro numérico, os instantes de lançamento e os de choque da linha com o solo, em virtude das mudanças bruscas na cinemática da linha. Nos demais intervalos de tempo não há variação significativa de comportamento dinâmico. Note que o valor do incremento utilizado pelo algoritmo adaptativo é menor nos instantes mais críticos da análise, logo após o lançamento da linha. Isso ocorre em função da variação do histórico de deslocamentos do modelo analisado. Quando esta mudança é medida em termos de curvatura (Figura 5.30) verifica-se tal constatação.

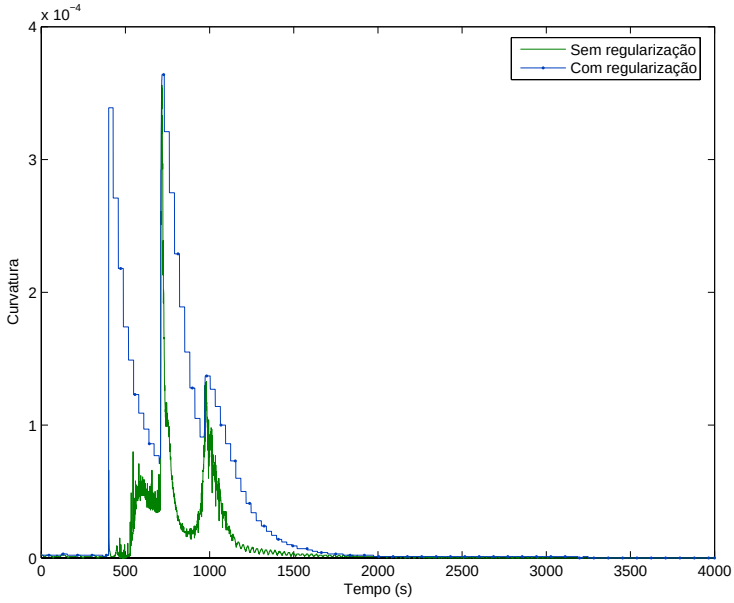


Figura 5.30: Histórico da curvatura (problema de *hookup*).

Note que os maiores valores de curvatura são observados após o lançamento. As mudanças de comportamento da função que descreve o indicador geométrico curvatura são utilizadas pelo algoritmo adaptativo tanto para diminuir como para aumentar o incremento de tempo de integração. A relação incremento-curvatura no entanto tem que ser feita a partir da série regularizada deste indicador, a fim de assegurar que as mudanças de incremento não afetem a estabilidade do processo de integração. Note que nos estágios finais da simulação, quando o comportamento da linha é quase estático, o algoritmo adaptativo aumenta o valor do incremento de tempo de integração, ultrapassando em alguns momentos $0{,}80 \times \Delta t_{crit}$.

Caso incrementos dessa magnitude sejam empregados ao longo de toda a análise, o valor do erro de discretização será grande nos instantes mais críticos. Tal constatação pode ser observada na Figura 5.31. Nessa, uma discretização temporal mais grosseira ($\Delta t = 0{,}825 \times \Delta t_{crit}$) é comparada com uma mais apurada ($\Delta t = 0{,}01 \times \Delta t_{crit}$) no instante em que a linha colide com o solo, instante onde são observados os maiores erros.

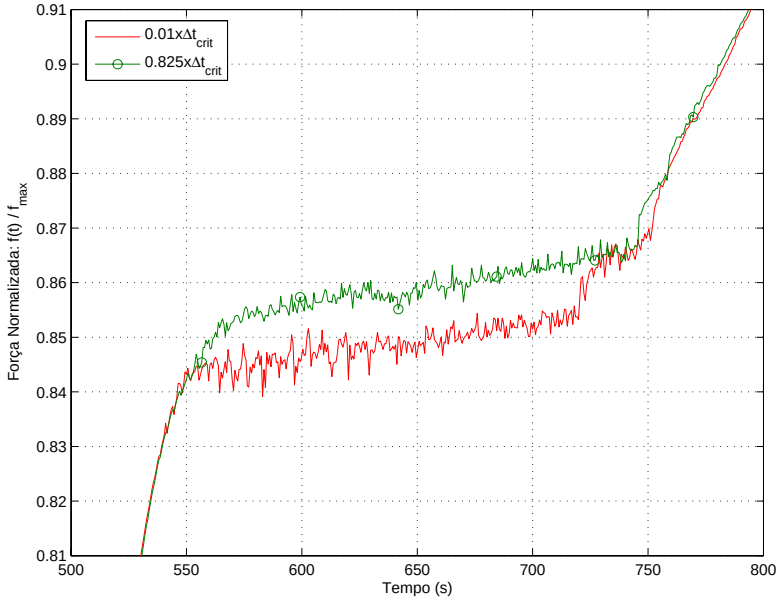


Figura 5.31: Erro do algoritmo convencional ($\Delta t = 0{,}825 \times \Delta t_{crit}$).

Para algumas situações de projeto esse nível de discretização pode ser suficiente. No entanto, muitas vezes é necessário um nível de detalhamento maior das grandezas envolvidas. O emprego de incrementos mais reduzidos pode ser utilizado com este intuito, conforme comentado anteriormente. A Figura 5.32 comprova esta afirmação.

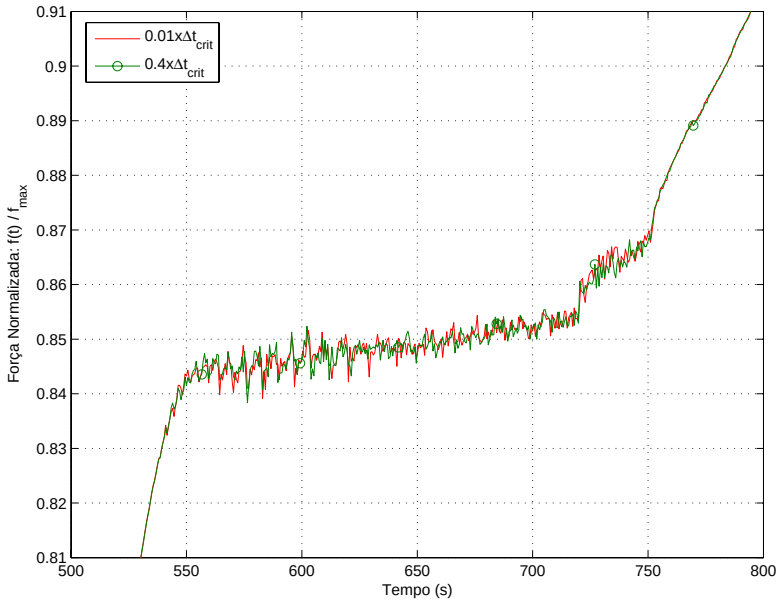


Figura 5.32: Erro do algoritmo convencional ($\Delta t = 0{,}40 \times \Delta t_{crit}$).

Note que ambas as respostas são bastante semelhantes entre si. Isso indica que o uso de um incremento mais reduzido, no caso $\Delta t = 0{,}40 \times \Delta t_{crit}$, é suficiente para garantir a qualidade da resposta. O grande desafio entretanto consiste em descobrir para uma dada análise qualquer que

valor é esse. Além disso, é preciso saber se o emprego desse incremento pode ocorrer durante todo o processo de análise sem comprometer o tempo de processamento.

Para algoritmos adaptativos essa indagação é respondida à medida que a resposta numérica é obtida. Quanto melhor for a relação entre o erro numérico e o estimador empregado pelo algoritmo adaptativo maior o proveito que se pode ter em qualidade e desempenho computacional.

Na Figura 5.33 apresenta-se uma comparação entre a resposta de força no topo da linha para o algoritmo adaptativo, baseado em curvatura do histórico de deslocamentos, e para um algoritmo que usa um incremento de integração fixo e refinado ($\Delta t = 0{,}01 \times \Delta t_{crit}$).

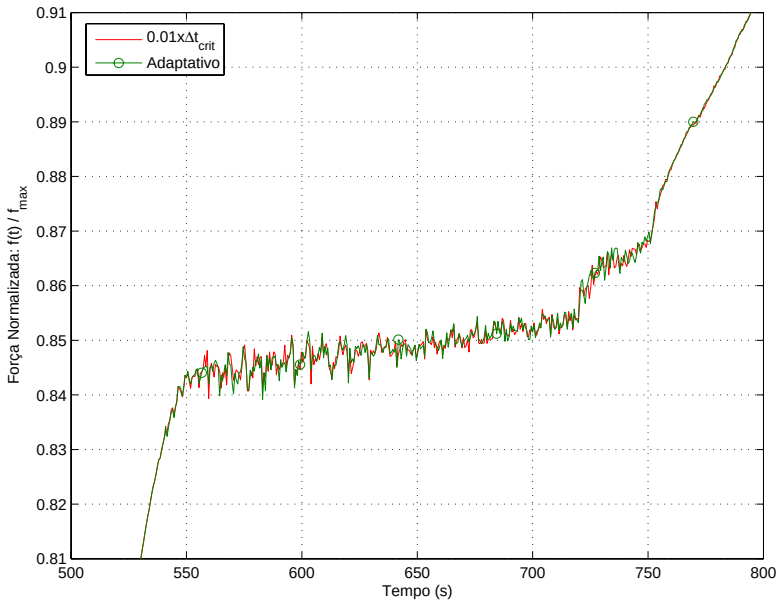


Figura 5.33: Erro do algoritmo adaptativo.

Observe que a diferença entre as respostas é bastante pequena, indicando que a informação fornecida pelo indicador geométrico pode portanto ser utilizada na melhoria da qualidade da resposta e no controle automático do incremento de integração. É necessário ter em mente que nem sempre é possível realizar análises com diferentes níveis de discretização, assim como feito neste trabalho, para uma dada simulação. O emprego de técnicas adaptativas apresenta-se portanto como uma forma alternativa e promissora de solução para esta realidade.

6 *Considerações Finais*

Conforme visto neste trabalho, o mecanismo de adaptação do tempo pode ser bastante útil na solução de problemas que fazem uso de métodos iterativos, a exemplo dos Métodos de Integração Direta. Isso acontece uma vez que em muitas situações a qualidade da resposta de um modelo numérico relaciona-se diretamente com o nível de discretização utilizado para a representação da variável tempo.

Na forma convencional de integração, a dinâmica que descreve o comportamento de um modelo numérico é obtida em instantes pontuais, obtidos de um processo de discretização uniforme. Esta prática, apesar de ser mais simples, pode demandar um esforço de solução muito grande, sobretudo quando se analisam grandes modelos de uma forma mais apurada. Os métodos adaptativos atuam no sentido de regular a discretização do tempo com o intuito de aperfeiçoar a relação entre a qualidade de resposta e o nível de discretização necessário para tanto.

Problemas dinâmicos com bom potencial de uso de técnicas de adaptatividade são aqueles que exigem que um nível de refinamento da solução mais elevado em apenas alguns instantes de análise. Simulações que envolvam variações bruscas nas condições de contorno ou mesmo na própria cinemática incógnita do problema são bons exemplos de aplicação destas técnicas.

O grande desafio na concepção de métodos adaptativos consiste em estabelecer algum parâmetro de referência que possa indicar a qualidade de uma resposta numérica, ou seja, um estimador de refinamento. A boa relação entre o estimador de refinamento e o nível de precisão obtido em uma resposta para um dado instante é essencial para que uma estratégia de adaptação seja válida. No escopo deste trabalho apresentam-se algumas técnicas adaptativas que fazem uso destes estimadores.

Uma dificuldade enfrentada na formulação de métodos adaptativos diz respeito ao esforço adicional requerido para o uso de um estimador de refinamento qualquer. Espera-se que a tarefa responsável pela determinação do incremento de tempo e seu correspondente tratamento exija um esforço de realização mínimo. Quando isso é verificado o esforço adicional exigido nas tarefas de adaptação é compensado, seja em qualidade de resposta ou em desempenho

computacional.

6.1 Principais Contribuições

Este trabalho colabora no âmbito dos métodos adaptativos de integração direta introduzindo uma estratégia de adaptação que utiliza o indicador geométrico curvatura. Dentre as principais contribuições pode-se destacar:

- **Desenvolvimento da formulação do estimador de refinamento e da estratégia adaptativa**
  O estimador de refinamento empregado consegue detectar mudanças cinemáticas que em geral contribuem para a redução da qualidade de resposta obtida em um processo de integração. Além disso, o cálculo deste indicador tem um baixo custo computacional, basicamente três produtos internos.

- **Implementação computacional de uma biblioteca de adaptação**
  Trata-se de um sistema orientado a objetos que incorpora a formulação do estimador de refinamento e a estratégia de adaptividade desenvolvida. Este sistema simplifica o aproveitamento da técnica adaptativa em sistemas que utilizam métodos iterativos com formulações compatíveis.

- **Implementação computacional de um *framework* de integração temporal adaptativa com suporte a computação de alto desempenho**
  O *framework* construído, *FASTTI*, usa o valor da curvatura de uma função que descreve o histórico de deslocamentos obtidos a partir de um método de integração direta. Trata-se, portanto, de um caso particular de uso da biblioteca anteriormente citada. O sistema permite que aplicações que utilizam métodos diretos possam herdar recursos de adaptatividade assim como técnicas de computação de alto desempenho.

- **Validação da técnica de adaptividade temporal e das implementações computacionais**
  Os exemplos numéricos apresentados indicam que o estimador de refinamento proposto pode ser facilmente implantado em diversos métodos de integração direta, uma vez que depende apenas de dados cinemáticos. A implantação do *FASTTI* em sistemas computacionais existentes (*DyCOH* e *Preadyn++*) ilustra ainda a facilidade de utilização do *framework* desenvolvido.

6.2 Sugestões para Trabalhos Futuros

Os tópicos tratados neste trabalho apenas elucidam a necessidade de estudos mais detalhados a respeito da estratégia de adaptação no tempo introduzida. Embora exemplos numéricos tenham sido apresentados de forma a demonstrar o potencial do estimador de refinamento formulado, alguns aspectos envolvidos ainda precisam ser objetos de investigações.

Como proposta inicial, indica-se o estudo de funções que relacionem o valor da curvatura com o do incremento de integração. Nas análises até então realizadas apenas emprega-se uma forma de função com este objetivo, a exponencial. Além disso, sugere-se que os parâmetros que definem estas funções sejam calibrados de maneira a maximizar o desempenho computacional para um dado nível de qualidade de resposta definido.

Outro estudo proposto consiste em investigar novas técnicas de regularização de curvatura. Conforme comentado, em muitos casos o uso destas técnicas é indispensável para garantir a estabilidade de um processo de integração. A técnica de regularização empregada neste trabalho (máximo valor em intervalos), apesar ter sido eficaz em assegurar a estabilidade, acaba tendo influência negativa no desempenho do processo de integração.

Quanto aos sistemas computacionais desenvolvidos, sugere-se a incorporação de novas funcionalidades, incluindo algoritmos de métodos diretos, sejam eles explícitos ou implícitos, novas técnicas de adaptação, dentre outros. Propõe-se ainda a elaboração de uma documentação técnica e de usuário, incluindo tutoriais e outros mecanismos que permitam uma maior facilidade de aplicação dos sistemas em *software* existente.

Referências Bibliográficas

AMORIM, J. A. *Aplicação de Modelos Coesivos Intrínsecos na Simulação da Propagação Dinâmica de Fraturas*. Dissertação (Mestrado) — Universidade Federal de Alagoas, Maceió-AL, 2007.

ARNOLD, R. R.; CITERLEY, R. L.; CHARGIN, M.; GALANT, D. Application of Ritz vectors for dynamic analysis of large structures. *Computers & Structures*, v. 21, n. 5, p. 901–907, 1985.

BATHE, K. J. M. *Finite Elements Procedures*. New Jersey: Prentice-Hall, 1996.

BELYTSCHKO, T.; HUGHES, T. J. R. *Computational Methods For Transient Analysis*. Amsterdam: North-Holland, 1983.

BENDERSKAYA, G.; GERSEM, H. D.; ACKERMANN, W.; WEILAND, T. Adaptive time integration for electromagnetic models with sinusoidal excitation. *Compel-The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, v. 27, n. 1, p. 122–132, 2008.

BERGAN, P. G.; MOLLESTAD, E. An automatic time-stepping algorithm for dynamic problems. *Computer Methods in Applied Mechanics and Engineering*, v. 49, n. 3, p. 299–318, 1985.

BUCK, I.; FOLEY, T.; HORN, D.; SUGERMAN, J.; FATAHALIAN, K.; HOUSTON, M.; HANRAHAN, P. Brook for gpus: Stream computing on graphics hardware. *ACM Transactions on Graphics*, v. 23, n. 3, p. 777–786, ago. 2004.

CHAPMAN, B.; JOST, G.; PAS, R. van der. *Using OpenMP: Portable Shared Memory Parallel Programming (Scientific and Engineering Computation)*. [S.l.]: The MIT Press, 2007. ISBN 0262533022, 9780262533027.

CHUNG, J.; CHO, E. H.; CHOI, K. A priori error estimator of the generalized-alpha method for structural dynamics. *International Journal for Numerical Methods in Engineering*, v. 57, n. 4, p. 537–554, maio 2003.

CHUNG, J. T.; LEE, J. M. A new family of explicit time integration methods for linear and nonlinear structural dynamics. *International Journal for Numerical Methods in Engineering*, v. 37, n. 23, p. 3961–3976, dez. 1994.

CINTRA, D. T.; SILVEIRA, E. S. S. Uma estratégia de adaptatividade no tempo para a análise estrutural de linhas e risers. In: CMNE CILAMCE 2007. *Métodos Numéricos e Computacionais em Engenharia*. Porto, Portugal, 2007. p. 432.

COELHO, L. C. G.; NISHIMOTO, K.; MASETTI, I. Q. Dynamic simulation of anchoring systems using computer graphics. In: *OMAE CONFERENCE*. Rio de Janeiro, RJ, Brazil: ASME, 2001.

COOK, R. D.; MALKUS, D. S.; PLESHA, M. E. *Concepts and Applications of Finite Element Analysis*. Madison: John Wiley & Sons, 1989.

CUNDALL, P. A.; STRACK, O. D. L. Discrete numerical-model for granular assemblies. *Geotechnique*, v. 29, n. 1, p. 47–65, 1979.

DANIEL, W. J. T. Subcycling first- and second-order generalizations of the trapezoidal rule. *International Journal for Numerical Methods in Engineering*, v. 42, n. 6, p. 1091–1119, jul. 1998.

DUNCAN, R. A survey of parallel computer architectures. *Computer*, IEEE Computer Soc, v. 23, n. 2, p. 5–16, fev. 1990.

FAHMY, M. W.; NAMINI, A. H. A survey of parallel nonlinear dynamic analysis methodologies. *Computers & Structures*, v. 53, n. 4, p. 1033–1043, nov. 1994.

FAYAD, M.; SCHMIDT, D. C. Object-oriented application frameworks. *Commun. ACM*, ACM, New York, NY, USA, v. 40, n. 10, p. 32–38, 1997. ISSN 0001-0782.

FERREIRA, C. M. A. *The Unsymmetric Tridiagonal Eigenvalue Problem*. Tese (Doutorado) — Universidade do Minho, Escola de Ciências, Braga/Portugal, 2007.

FERREIRA, F. M. G. *Desenvolvimento e Aplicações de um Framework Orientado a Objetos para Análise Dinâmica de Linhas de Ancoragem e de Risers*. Dissertação (Mestrado) — Universidade Federal de Alagoas, Maceió-AL, 2005.

FLYNN, M. J. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*, IEEE Computer Soc, C 21, n. 9, p. 948–&, 1972.

FOWLER, M. *UML distilled: a brief guide to the standard object modeling language, 3rd edition*. Boston, MA, USA: Addison-Wesley, 2003. ISBN 0321193687.

GEORGII, J.; WESTERMANN, R. Mass-spring systems on the GPU. *Simulation Modelling Practice and Theory*, v. 13, n. 8, p. 693–702, nov. 2005.

GOBAT, J. I.; GROSENBAUGH, M. A. Time-domain numerical simulation of ocean cable structures. *Ocean Engineering*, v. 33, n. 10, p. 1373–1400, jul. 2006.

HIBBIT, H. D.; KARLSSON, B. I. Analysis of pipe whip. In: ASME PRESSURE VESSELS AND PIPING CONFERENCE. San Francisco, CA, USA, 1979.

HILBER, H. M.; HUGHES, T. J. R.; TAYLOR, R. L. Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering & Structural Dynamics*, v. 5, n. 3, p. 283–292, 1977.

HUGHES, T. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Englewood, New Jersey: Prentice-Hall, 1987.

HULBERT, G. M.; CHUNG, J. T. Explicit time integration algorithms for structural dynamics with optimal numerical dissipation. *Computer Methods in Applied Mechanics and Engineering*, v. 137, n. 2, p. 175–188, out. 1996.

HULBERT, G. M.; JANG, I. Automatic time-step control algorithms for structural dynamics. *Computer Methods in Applied Mechanics and Engineering*, v. 126, n. 1-2, p. 155–178, set. 1995.

KREYSZIG, E. *Advanced Engineering Mathematics*. 9. ed. [S.l.]: John Wiley & Sons, 2006. 1246 p.

KRYSL, P.; BELYTSCHKO, T. Object-oriented parallelization of explicit structural dynamics with PVM. *Computers & Structures*, v. 66, n. 2-3, p. 259–273, jan. 1998.

LOWRIE, R. B. A comparison of implicit time integration methods for nonlinear relaxation and diffusion. *Journal of Computational Physics*, v. 196, n. 2, p. 566–590, maio 2004.

MARK, W. R.; GLANVILLE, R. S.; AKELEY, K.; KILGARD, M. J. Cg: A system for programming graphics hardware in a c-like language. *Acm Transactions on Graphics*, v. 22, n. 3, p. 896–907, jul. 2003.

NEWMARK, N. M. A method of computation for structural dynamics. *ASCE Journal of Engineering Mechanics Division*, v. 85, p. 67–94, 1959.

NICKOLLS, J.; BUCK, I.; GARLAND, M.; SKADRON, K. Scalable parallel programming with cuda. *Queue*, ACM, New York, NY, USA, v. 6, n. 2, p. 40–53, 2008. ISSN 1542-7730.

OURGHOULIAM, C.; POWEL, G. ANSR-III - general purpose computer program for nonlinear structural analysis. In: EARTHQUAKE ENGINEERING RESEARCH CENTER. Berkley, CA, 1982. p. 245.

OWENS, J. D.; LUEBKE, D.; GOVINDARAJU, N.; HARRIS, M.; KRUGER, J.; LEFOHN, A. E.; PURCELL, T. J. A survey of general-purpose computation on graphics hardware. *Computer Graphics Forum*, v. 26, n. 1, p. 80–113, 2007.

PANTALÉ, O. Parallelization of an object-oriented fem dynamics code: influence of the strategies on the speedup. *Advances in Engineering Software*, v. 36, n. 6, p. 361–373, jun. 2005.

PARK, K. C.; UNDERWOOD, P. G. A variable-step central difference method for structural dynamics analysis -part 1: Theoretical aspects. *Computer Methods in Applied Mechanics and Engineering*, v. 22, n. 2, p. 241–258, 1980.

RIO, G.; SOIVE, A.; GROLLEAU, V. Comparative study of numerical explicit time integration algorithms. *Advances in Engineering Software*, v. 36, n. 4, p. 252–265, abr. 2005.

ROMERO, I.; LACOMA, L. M. A methodology for the formulation of error estimators for time integration in linear solid and structural dynamics. *International Journal for Numerical Methods in Engineering*, v. 66, n. 4, p. 635–660, abr. 2006.

ROST, R. J. *OpenGL(R) Shading Language*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004. ISBN 0321197895.

RUMBAUGH, J.; JACOBSON, I.; BOOCH, G. *The Unified Modeling Language Reference Manual*. Boston, MA, USA: Addison-Wesley, 1999. ISBN 020130998X.

RYU, H. S.; BAE, D. S.; CHOI, J. H.; SHABANA, A. A. A compliant track link model for high-speed, high-mobility tracked vehicles. *International Journal for Numerical Methods in Engineering*, v. 48, n. 10, p. 1481–1502, ago. 2000.

SILVEIRA, E. S. S. *Análise Dinâmica de Linhas de Ancoragem com Adaptação no Tempo e Subciclagem*. Tese (Doutorado) — Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro/RJ, 2001.

STROUSTRUP, B. *The C++ Programming Language*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000. ISBN 0201700735.

SöDERLIND, G. Digital filters in adaptive time-stepping. *ACM Transactions on Mathematical Software*, v. 29, n. 1, p. 1–26, mar. 2003.

SöDERLIND, G.; WANG, L. Adaptive time-stepping and computational stability. *Journal of Computational and Applied Mathematics*, v. 185, n. 2, p. 225–243, jan. 2006.

WANG, H. Y.; TAYLOR, S.; SIMKIN, J.; BIDDLECOMBE, C.; TROWBRIDGE, B. An adaptive-step time integration method applied to transient magnetic field problems. *IEEE Transactions on Magnetics*, v. 37, n. 5, p. 3478–3481, set. 2001.

WILSON, E. L.; YUAN, M. W.; DICKENS, J. M. Dynamic analysis by direct superposition of Ritz vectors. *Earthquake Engineering & Structural Dynamics*, v. 10, n. 6, p. 813–821, 1982.

XU, X. P.; NEEDLEMAN, A. Void nucleation by inclusion debonding in a crystal matrix. *Modelling and Simulation in Materials Science and Engineering*, v. 1, n. 2, p. 111–132, jan. 1993.

ZHANG, D.; WHITEN, W. J. Step size control for efficient discrete element simulation. *Minerals Engineering*, v. 14, n. 10, p. 1341–1346, out. 2001.

ZIENKIEWICZ, O. C.; TAYLOR, R. L. *The Finite Element Method*. Fifth. [S.l.]: Butterworth-Heinemann, 2000.

ZIENKIEWICZ, O. C.; XIE, Y. M. A simple error estimator and adaptive time-stepping procedure for dynamic analysis. *Earthquake Engineering & Structural Dynamics*, v. 20, n. 9, p. 871–887, set. 1991.

APÊNDICE A – Dedução da equação de curvatura

Diferenciando a equação da norma da função derivada da curva parametrizada $\boldsymbol{\tau}$:

$$\|\boldsymbol{\tau}'\|' = \frac{\boldsymbol{\tau}' \cdot \boldsymbol{\tau}''}{\|\boldsymbol{\tau}'\|} \tag{A.1}$$

e substituindo na equação 3.10, tem-se:

$$k(z) = \left\| \frac{\boldsymbol{\tau}''\|\boldsymbol{\tau}'\| - \frac{\boldsymbol{\tau}' \cdot \boldsymbol{\tau}''}{\|\boldsymbol{\tau}'\|}\boldsymbol{\tau}'}{\|\boldsymbol{\tau}'\|^3} \right\|, \tag{A.2}$$

$$k(z) = \left\| \frac{\boldsymbol{\tau}''\|\boldsymbol{\tau}'\|^2 - \boldsymbol{\tau}' \cdot \boldsymbol{\tau}''\boldsymbol{\tau}'}{\|\boldsymbol{\tau}'\|^4} \right\|, \tag{A.3}$$

$$k(z) = \frac{\|\boldsymbol{\tau}''\|\boldsymbol{\tau}'\|^2 - \boldsymbol{\tau}' \cdot \boldsymbol{\tau}''\boldsymbol{\tau}'\|}{(\boldsymbol{\tau}' \cdot \boldsymbol{\tau}')^2}. \tag{A.4}$$

Fazendo:

$$a = \|\boldsymbol{\tau}'\|^2, \tag{A.5}$$

e

$$b = \boldsymbol{\tau}' \cdot \boldsymbol{\tau}'', \tag{A.6}$$

a equação de curvatura pode ser escrita na forma:

$$k(z) = \frac{\sqrt{a^2\boldsymbol{\tau}'' \cdot \boldsymbol{\tau}'' - 2ab\boldsymbol{\tau}' \cdot \boldsymbol{\tau}'' + b^2\boldsymbol{\tau}' \cdot \boldsymbol{\tau}'}}{a^2}. \tag{A.7}$$

Substituindo (A.5) e (A.6) em (A.7), tem-se:

$$k(z) = \frac{\sqrt{a^2\boldsymbol{\tau}'' \cdot \boldsymbol{\tau}'' - 2ab^2 + b^2a}}{a^2}, \tag{A.8}$$

$$k(z) = \frac{\sqrt{a(a\boldsymbol{\tau}'' \cdot \boldsymbol{\tau}'' - 2b^2 + b^2)}}{a^2}, \tag{A.9}$$

$$k(z) = \frac{\sqrt{a\boldsymbol{\tau}'' \cdot \boldsymbol{\tau}'' - b^2)}}{a^{3/2}}. \tag{A.10}$$

Usando novamente a igualdade das equações (A.5) e (A.6), chega-se à forma da equação de curvatura apresentada por Kreyszig (2006):

$$k(z) = \frac{\sqrt{(\boldsymbol{\tau}' \cdot \boldsymbol{\tau}')(\boldsymbol{\tau}'' \cdot \boldsymbol{\tau}'') - (\boldsymbol{\tau}' \cdot \boldsymbol{\tau}'')^2}}{(\boldsymbol{\tau}' \cdot \boldsymbol{\tau}')^{3/2}}. \tag{A.11}$$