UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA QUÍMICA
ÁREA DE CONCENTRAÇÃO SISTEMAS DE PROCESSOS QUÍMICOS E INFORMÁTICA

"SIMULAÇÃO E CONTROLE PREDITIVO LINEAR (COM MODELO DE CONVOLUÇÃO) E NÃO-LINEAR (COM MODELO BASEADO EM REDES NEURAIS ARTIFICIAIS) DE COLUNAS RECHEADAS DE ABSORÇÃO COM REAÇÃO QUÍMICA"

Autor: José Edson Lima e Silva

Orientador: Prof. Dr. Sergio Persio Ravagnani

Dissertação apresentada à Faculdade de Engenharia Química como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Química.

Setembro de 1997
Campinas - SP

UNICAMP
BIBLIOTECA CENTRAL

IDADE BC
CHAMADA:
T/UNICAMP
Si38s
Ex.
MBO BC/32003
OC. 281/97
C D X
ECO R$ 11,00
TA 12/11/97
CPD

CM-00102070-4

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Si38s

Silva, José Edson Lima e
Simulação e controle preditivo linear (com modelo de convolução) e não-linear (com modelo baseado em redes neurais artificiais) de colunas recheadas de absorção com reação química. / José Edson Lima e Silva.--Campinas, SP: [s.n.], 1997.

Orientador: Sergio Persio Ravagnani
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Química

1. Absorção. 2. Massa - transferência. 3. Controle preditivo. 4. Inteligência artificial. 5. Simulação (Computadores). I. Ravagnani, Sergio Persio. II. Universidade Estadual de Campinas. Faculdade de Engenharia Química. III. Título.

Agradecimentos

Ao professor Sergio Persio Ravagnani, pela orientação e pela compreensão quando o trabalho andou a passos de tartaruga;

à CAPES, pelo apoio financeiro que tornou viável a realização deste trabalho

aos amigos que contribuiram através de muitas conversas e sugestões, em especial: Jurandir (que me permitiu os conhecimentos básicos de redes neurais artificiais); Arlan (pela grande troca de informações sobre transferência de massa com reação química), Marcone (pela ajuda na utilização dos recursos computacionais, principalmente no ambiente UNIX) e ao Cesar (pelas constantes conversas sobre o trabalho);

ao Sérgio e ao Arlan, pela dicas de violão que ajudou (embora indiretamente) de forma decisiva na realização deste trabalho;

a todos os colegas do DESQ;

e principalmente aos meus irmãos, Maria Cristina, César e Sandra, e aos meus pais, Margarida e Eloi, pelo constante apoio sob as mais diversas formas.

NOMENCLATURA

Modelagem de colunas recheadas e transferência de massa com reação química

a'_v = Área interfacial por unidade de volume de recheio da coluna (m^2/m^3)
a_k = Coeficiente estequiométrico do componente "k"
C_i = Concentração do componente "i" ($kmol/m^3$)
D_i = Difusividade do componente i (m^2/h)
D_P = Diâmetro nominal da partícula do recheio (m)
F = Fluxo molar de gás (kmol/h)
H_L = Holdup de líquido (m^3/m^3)
k_G = Coeficiente de transferência de massa na fase gás ($kmol/m^2$ h bar)
k_L = Coeficiente de transferência de massa na fase líquida (m/h)
L = Vazão volumétrica de líquido (m^3/h)
M = Peso molecular (kg/kmol)
$N_k|_{y=y_G}$ = Fluxo de transferência de massa do componente "k" na interface gás-líquido ($kmol/m^2$ h)
$N_k|_{y=y_L}$ = Fluxo de transferência de massa do componente "k" na interface filme-seio da solução ($kmol/m^2$ h)
P_A = Pressão parcial do componente absorvido no gás (bar)
P_t = Pressão do gás (bar)
R = Constante dos gases ideais (bar m^3/kmol K)
r = Taxa de reação química ($kmol/m^3$ h)
T = Temperatura absoluta (K)
t = Tempo (h)
y = Comprimento do filme de transferência de massa (m)
Z = Altura do recheio na coluna (m)
z = Coordenada axial na coluna (m)

Subescritos

A = Referente ao componente absorvido
G = Referente a gás
L = Referente a líquido
P_k = Referente aos produtos "k"
R_k = Referente aos reagente "k"

Letras Gregas

ε = Porosidade do recheio (m^3/m^3)
μ = Viscosidade (kg/m h)
ρ = Massa específica do líquido (kg/m^3)
σ = Tensão superficial do líquido (m^2 /dina)
σ_c = Tensão superficial crítica do recheio (N/m)
Ω = Área da seção transversal da coluna (N/m)

Colocação Ortogonal

A_{ji} = Matriz de colocação para a primeira derivada
B_{ji} = Matriz de colocação para a segunda derivada
P(u)= Polinômio ortogonal
$P_j^{(\alpha,\beta)}(u)$ = Polinômio ortogonal de Jacobi
R(x,y) = Resíduo da aproximação de uma função
W_k = Peso no método dos resíduos ponderados
Sobrescrito
l = Referente ao elemento (no método da colocação ortogonal em elementos finitos)

Letras Gregas
α, β= Constantes do polinômio de Jacobi
δ = Função delta de Dirac

Controle preditivo com modelo

a = Coeficientes do modelo de convolução
f = Fator de amortização das ações de controle
h = Coeficientes do modelo de convolução (forma diferencial)
L = Horizonte de controle
N = Horizonte do modelo
R = Horizonte de predição
r_k = Valor desejado ("setpoint") no instante k
Δt = Período de amostragem
u_k = Variável de entrada (manipulada) do processo no instante k
y^c_k = Saída predit corrigida do processo para o instante k
y^d_k = Trajetória desejada do processo
y_k = Saída do processo (medida) no instante k
$\hat{y}_k$ = Saída predita para o processo para o instante k

Letras Gregas
α = Constante do filtro de 1ª ordem

Redes Neurais Artificiais

a = Saída de um neurônio
d = Saída desejada (real)
E = Erro de predição
f(.) = Função de ativação
I = Entrada passadas para a rede neural artificial
O = Saída da rede neural artificial
Th = Resíduo de ativação do neurônio ("*Threshold*")
w = Pesos da rede neural
x = Sinal total que chega a um neurônio

Sobrescrito
c = Referente a camada
m = Iteração do algoritmo Backpropagation

Subescrito

p = Referente ao par de treinamento

Letras Gregas

α = Momento ("memória" da rede), no algoritmo Backpropagation

δ = Gradiente (no algoritmo Backpropagation)

η = Taxa de aprendizado (no algoritmo Backpropagation)

RESUMO

O tratamento dado a problemas que envolvam transferência de massa com reação química tem o sido o mais diverso possível, sendo que a maioria dos trabalhos se preocupa em encontrar soluções analíticas que sirvam para alguns casos específicos, já que soluções analíticas genéricas não são possíveis. Apesar do grande número de trabalhos, as soluções para um grande número de sistemas são apenas aproximações da solução verdadeira. A resolução numérica das equações fundamentais de transferência de massa permite que qualquer tipo de regime cinético seja utilizado.

Desta forma, numa primeira etapa deste trabalho, o estudo dos perfis de concentração e fluxos mássicos ao longo de colunas de recheadas de absorção com reação química foi feito utilizando-se as equações fundamentais de transferência de massa com o termo de reação química. As equações foram resolvidas de forma que se adequem a qualquer tipo de sistema em ocorra uma reação química simultânea ao processo de absorção. O método numérico utilizado foi o da colocação ortogonal em elementos finitos, que se mostrou bastante adequado tanto em termos de precisão, como pela redução do sistema de equações obtido.

O tratamento dado a simulação dinâmica de colunas recheadas também tem sido bastante simplificado, sendo que características importantes do processo em geral não são levadas em consideração. A utilização do método das Características é uma alternativa para que se possa fazer uma simulação mais rigorosa. No entanto, este necessita do cálculo dos fluxos de transferência de massa em diversos pontos ao longo da coluna, o que torna impossível a sua utilização juntamente com a resolução númerica das equações de transferência de massa.

Para contornar este problema, as redes neurais artificiais foram utilizadas na predição dos fluxos de transferência de massa, juntamente com as equações de balanço de massa ao longo da coluna, constituindo assim um modelo neural híbrido. As RNA's mostraram-se bastante adequadas para este fim, apesar da alta não-linearidade apresentada pelos dados de transferência de massa com reação química.

Desta forma, o modelo resultante pode usar o rigor da transferência de massa com uma modelagem que inclui características hidrodinâmicas importantes do processo. Os resultados obtidos comprovaram a importância em se considerar o processo desta forma, ao invés das simplificações normalmente feitas.

Numa etapa final do trabalho, estudou-se o comportamente do controle preditivo com modelo linear e não-linear frente a este processo que é altamente não-linear. Este estudo mostrou que o processo, pelo fato de ser distribuido, apresenta um comportamento que torna difícil o bom desempenho do sistema de controle.

Ao final do trabalho, foi feita uma comparação entre os resultados obtidos com os dois modelos internos do controlador aqui utilizados, não só com relação aos resultados, como também em relação à dificuldade de aplicação e às limitações destes. Encontrou-se que, embora o controle não-linear apresentasse melhores resultados que o linear, a diferença de desempenho não foi tão grande, o que se deve principalmente à grande inércia do sistema frente variável manipulada (neste caso, a vazão de solvente). Conclui-se então que em alguns casos a utilização do controle preditivo com modelo de convolução pode ser justificado pela sua simplicidade de aplicação e resultados razoáveis, quando comparado ao controle com modelo baseado em RNA's.

ÍNDICE

1 - INTRODUÇÃO 1

2 - MÉTODOS NUMÉRICOS 4
2.1 - Redes Neurais Artificiais 5
2.1.1 - O Algoritmo Backpropagation utilizando a Regra do Delta Generalizado (GDR) 8
2.1.2 - O Algoritmo de Marquardt-Levenberg 10
2.1.3 - Adimensionalização das entradas e saídas 11
2.1.4 - O conceito de janela de tempo para simulação dinâmica 14
2.2 - Colocação Ortogonal 16
2.2.1 - Colocação Ortogonal Global 16
2.2.2 - Colocação Ortogonal em Elemento Finitos 19
2.3 - O Método das Características 21

3 - TRANSFERÊNCIA DE MASSA COM REAÇÃO QUÍMICA 27
3.1 - Aplicações da Transferência de Massa com Reação Química 28
3.2 - Teorias de Transferência de Massa 29
3.2.1 - Teoria do Filme 30
3.2.2 - Teoria da Penetração de Higbie 31
3.2.3 - Teoria da Superfície Renovável 32
3.2.4 - Outras Teorias de Transferência de Massa 32
3.3 - Resolução Analítica das Equações 33
3.4 - Resolução Numérica das Equações 34
3.5 - Dados Físico-Químicos 35
3.5.1 - Solubilidade de gases em líquidos 36
3.5.2 - Difusividades 37
3.6 - Absorção de CO_2 em Solução de Monoetanolamina (MEA) 38
3.7 - Cálculo dos Fluxos de Transferência de Massa utilizando a Teoria do Filme e o Método da Colocação Ortogonal em Elementos Finitos 38

4 - MODELAGEM E SIMULAÇÃO DE COLUNAS RECHEADAS 43
4.1 - Processos que utilizam a absorção com reação química 44
4.2 - Os diferentes tipos de equipamentos 45
4.3 - Modelos hidrodinâmicos de colunas recheadas 46
4.4 - Cálculo do estado estacionário 47
4.4.1 - Resultados 52
4.5 - Cálculo dos fluxos de transferência de massa a partir das redes neurais artificiais 54
4.6 - Simulação dinâmica de colunas recheadas de absorção com reação química utilizando modelo neural híbrido 57
4.6.1 - Desevolvimentos dos balanços dinâmicos 58
4.6.2 - Algoritmo dinâmico 60
4.6.3 - Dinâmica da base e do topo da coluna 62
4.6.4 - Resultados 63

5 - CONTROLE PREDITIVO 67
5.1 - Motivação do estudo 68
5.2 - A estratégia de controle 69
5.3 - Formulação matemática do Controle Preditivo 70
5.4 - O modelo baseado em RNA's 74
5.5 - O modelo de Convolução 79
5.6 - Resultados 83
5.6.1 - Sintonia do controle preditivo com o modelo de Convolução 84
5.6.2 - Sintonia do controle preditivo com modelo baseado em RNA 89
5.7 - Comparação entre os resultados obtidos com modelo baseado em RNA e com o modelo de convolução 92

6 - CONCLUSÕES E SUGESTÕES 93

REFERÊNCIAS BIBLIOGRÁFICAS 97

APÊNDICE A - 103

APÊNDICE B - 107

APÊNDICE C - 134

CAPÍTULO 1

Introdução

INTRODUÇÃO

O projeto de colunas recheadas de absorção com reação química tem sido bastante estudado na literatura, com um grande número de trabalhos publicados nesta área. No entanto, trabalhos que tratem de uma simulação dinâmica têm recebido bem menos atenção, sendo que os existentes, em geral, utilizam várias simplificações que comprometem os resultados.

Algumas simplificações típicas quando se trata de simulação de colunas de absorção recheadas, são: soluções diluídas, o que permite desprezar as variações de fluxos mássicos ao longo da torre; e a consideração de que as perturbações se propagam com velocidade infinita, ou seja, as variações que ocorrem nas alimentações atingem imediatamente todos os pontos da coluna. Embora haja casos em que considerações deste tipo sejam aceitáveis, para muitas colunas industriais estas são simplificações drásticas.

Em especial, no caso da absorção seguida de reação química, devido ao aumento da capacidade de absorção que esta proporciona, as correntes podem apresentar grandes variações de concentração ao longo da torre, o que pode causar uma variação considerável dos fluxos ao longo da torre, modificando assim, os coeficientes de transferência de massa.

Por outro lado, a transferência de massa seguida de reação química é um fenômeno difícil de tratar e importante neste tipo de processo, já que introduz uma grande não-linearidade no sistema. Abordagens clássicas como considerações de equilíbrio não se aplicam a este caso. Os fluxos de transferência devem ser calculados a partir das equações fundamentais de teorias de transferência de massa.

Soluções analíticas têm sido desenvolvidas ao longo dos anos para reações de diversas ordens e vários regimes cinéticos, de forma que grande parte dos trabalhos desenvolvidos nesta área se concentram sobre este tema. No entanto, estas soluções, além de só se aplicarem a determinado tipo de sistema, muitas vezes são aproximações da solução real, podendo apresentar um erro grande a depender do sistema. Além disto, estas muitas vezes não permitem a inclusão de fenômenos importantes, como a redução de solubilidade devido à presença de íons, por exemplo.

Logo, temos que o processo de absorção seguido de reação química em colunas recheadas apresenta várias características, tanto hidrodinâmicas como de transferência de massa, que são importantes para uma descrição mais rigorosa, mas que muitas vezes não são levados em consideração, principalmente pelas dificuldades numéricas que surgem na resolução das equações resultantes do fenômeno.

Estas características contribuem para um aumento da não-linearidade e para a presença de tempo morto nas respostas do sistema. Estes são fenômenos que dificultam bastante o desempenho de sistemas de controle e devem, portanto, estar presentes no projeto e análise destes.

No desenvolvimento do simulador procurou-se levar em consideração uma modelagem a mais rigorosa possível, para que se possa ter confiança nas informações obtidas a partir deste, sem no entanto tornar o modelo muito complexo, o que poderia tornar a sua resolução muito difícil e demorada, ou mesmo inviável. Isto foi possível graças a utilização de uma abordagem mais recente dada a problemas de simulação: os chamados modelos neurais híbridos, que consistem em utilizar Redes Neurais Artificiais (RNA) juntamente com equações que descrevem fenomenologicamente o processo.

A partir de uma representação mais fiel do sistema, o estudo do sistema de controle pode ser feito com mais segurança nos resultados obtidos. O sistema de controle aqui considerado foi do tipo SISO, ou seja, com uma única entrada (variável manipulada) e uma única saída (variável controlada).

Já que a absorção seguida de reação química é utilizada principalmente em processos que necessitam manter a concentração de gases ácidos em suas correntes em uma faixa bem estreita, é interessante ter um sistema de controle bastante robusto, sendo particularmente interessante os que se antecipam a resposta do sistema a perturbações.

Por estes aspectos, as técnicas de controle preditivo baseado em modelo são as mais indicadas, tendo sido as aqui estudadas. Além disto, estas permitem a manipulação de restrições, que aparecem naturalmente no caso de colunas recheadas devido à existência do ponto a partir do qual ocorre arraste de líquido e à exigência de um grau mínimo de molhamento para que não haja a formação de zonas secas na coluna.

O modelo interno do controlador é a parte principal desta técnica, já que é com base nas previsões futuras deste que as ações de controle são calculadas. Uma questão natural que surge neste caso diz respeito as implicações em se utilizar um modelo interno do controlador linear em lugar de um não-linear, já que o processo é altamente não-linear. Esta questão será aqui analisada com base em duas técnicas bastante utilizadas: o modelo de Convolução (linear) e o modelo baseado em RNA's (não-linear).

Embora os resultados obtidos com o modelo não-linear sejam superiores aos obtidos com o linear, o último é de mais fácil aplicação, de forma que se torna interessante saber até que ponto o resultado final de controle é influenciado pela utilização de um ou outro.

Este trabalho é dividido nas seguintes partes: no capítulo 2 será feita uma análise dos métodos numéricos aqui utilizados; no capítulo 3 alguns aspectos da transferência de massa com reação química são discutidos; no capítulo 4 os modelos hidrodinâmicos de colunas recheadas são analisados e uma metodologia é proposta para que se possa considerar características importantes do processo; no capítulo 5 é feito um estudo sobre o controle preditivo deste processo e, finalmente, no capítulo 6, com base nos resultados obtidos, são apresentadas algumas conclusões e sugestões para trabalhos futuros.

CAPÍTULO 2

Métodos Numéricos

MÉTODOS NUMÉRICOS

Neste capítulo serão discutidos alguns dos métodos numéricos aqui utilizados. Um enfoque maior será dado com respeito às aplicações, vantagens e limitações de cada método, o que ajudará a entender a utilização destes no presente trabalho. As formulações matemáticas serão tratadas apenas de forma superficial, já que estas podem ser encontradas de forma mais detalhada na literatura.

2.1 - Redes Neurais Artificiais.

As redes neurais artificiais (RNA) têm recebido uma crescente atenção em diversas áreas das ciências exatas. Sua simplicidade, aliada a capacidade de manusear dados complexos e altamente não-lineares, tem permitido que elas sejam aplicadas na resolução de uma série de problemas de diversas áreas. Na engenharia química as RNA's têm sido utilizadas com grande sucesso nas áreas de modelagem, controle e identificação de processos, (Bhat *et al* (1990); Narenda *et al* (1990); Ydstie (1990); Hunt *et al* (1992); Willis *et al* (1992); Hoskins *et al* (1992); Psichogios *et al* (1992); Thompson *et al* (1994); Morris *et al* (1994); Zbicinski *et al* (1996);), em detecção de falhas em equipamentos (Hoskins *et al* (1988); Hoskins *et al* (1991); Ungar *et al* (1995)), etc.

As RNA's são capazes de identificar e "aprender" a relação existente entre entradas e saídas num conjunto de dados. Este aprendizado não depende de nenhum conhecimento prévio, não sendo necessário, por exemplo, definir nenhuma função que aproxime estes dados. Desta forma, os dados são tratados por uma espécie de caixa preta. Por este motivo, as RNA's são consideradas aproximadores universais de funções.

O que chamamos de RNA é apenas um dos ramos da Inteligência Artificial, ciência que desenvolve sistemas que imitam características de funcionamento do cérebro humano. Assim, sabe-se que os neurônios no cérebro humano funcionam a partir de pulsos elétricos. Cada neurônio recebe e envia pulsos de vários outros, sendo que o pulso de saída de um neurônio é função dos vários pulsos recebidos. Sabe-se também que a conexão entre um neurônio pode ampliar ou reduzir os pulsos transmitidos, bem como os neurônios, internamente, podem estar mais ou menos ativos, ou seja, podem responder de formas diferentes a um mesmo pulso. Estas características são matematicamente imitadas pelas redes neurais artificiais.

Uma RNA é, portanto, um sistema composto de camadas de neurônios artificiais (no mínimo três), que são unidades capazes de receber e enviar sinais. O seu funcionamento pode ser resumido, de forma bastante simplificada, nos seguintes passos:

(i) Um dado neurônio recebe o sinal (numérico) de outros neurônios;
(ii) Esses sinais são ponderados por um peso e somados;
(iii) A essa soma é adicionada um resíduo de ativação do neurônio;
(iv) A soma é modificada por meio de uma função de ativação;
(v) O sinal final é enviado a outros neurônios, sendo que na conexão entre os neurônios, esse sinal é modificado (multiplicado) por um peso.

Embora, de uma forma geral, os neurônios possam estar conectados a outros neurônios de várias camadas, para fins de aplicações na engenharia química, o tipo de rede mais utilizado é a direta (*"feedforward"*), onde os neurônios de uma camada só recebem sinal da camada imediatamente anterior e só enviam sinais para a camada imediatamente posterior (fig 2.1). Este tipo de rede é o mais utilizado na simulação de processos. Além disto, redes diretas com uma única camada intermediária são suficientes para a representação de dados contínuos (Cybenko (1989)). Por este motivo, trataremos aqui exclusivamente de redes diretas com uma única camada intermediária. Na fig. 2.1 temos um exemplo deste tipo de rede, a qual apresenta uma topologia (3-4-2), ou seja, 3 neurônios na camada de entrada, 4 na camada escondida e 2 na camada de saída.

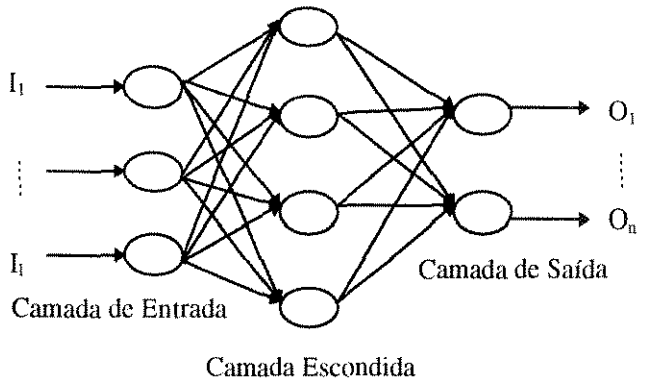

Fig 2.1 - Rede neural direta (*feedforward*), com três camadas: cada neurônio é ligado a todos os neurônios da camada imediatamente anterior e imediatamente posterior.

A fig. 2.2 mostra de forma mais detalhada um neurônio artificial. A função de ativação que se encontra representada é uma das mais comumente utilizada na literatura. A escolha de uma função de ativação deve seguir alguns conceitos de reconhecimento de padrões descritos na literatura. Dentre as características de maior interesse nesta função, temos:

(i) É uma função limitada (para $x \rightarrow -\infty$, $f(x)=0$; para $x \rightarrow \infty$, $f(x)=1$), o que confere uma maior estabilidade à rede;

(ii) É uma função de tipo sigmoidal, a qual apresenta características lineares e não-lineares, a depender da faixa, o que auxilia no treinamento dos diversos tipos de dados;

(iii) É facilmente derivada.

A saída de um neurônio de uma dada camada pode ser calculada a partir das saídas de todos os neurônios das camadas anteriores a partir do seguinte procedimento:

(i) O sinal total que chega ao j-ésimo neurônio da camada c (x_j^c), é a soma ponderada dos sinais de saída dos neurônios da camada anterior (a_i^{c-1}) subtraído ou somado de um resíduo (*"threshold"*). Quando se subtrai, esse resíduo é conhecido como *"bias"*. O fatores ponderais são os pesos da rede ($w_{i,j}^c$):

$$x_j^c = \sum_{i=1}^{l} a_i^{c-1} w_{i,j}^c + Th_j^c \qquad (2.1)$$

(ii) Esse sinal é então modificado pela função de ativação, dando a saída do neurônio (a_j^c). Caso a função sigmoidal seja utilizada, a saída é dada por:

$$a_j^c = f(x_j^c) = \frac{1}{1+e^{-x_j^c}} \qquad (2.2)$$

Desta forma, é necessário calcular a saída de todos os neurônios de uma camada para se calcular a saída dos neurônios da camada seguinte. Procedendo-se desta forma, temos a saída da última camada (NC), ou seja, a resposta da rede:

$$O_k = a_k^{NC} \qquad (2.3)$$

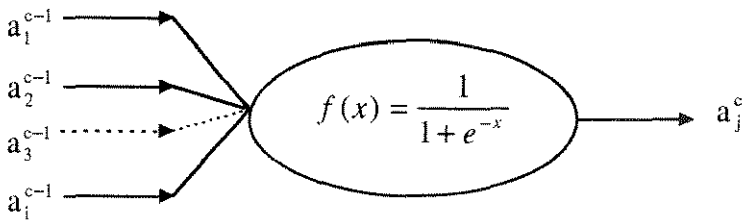

Fig 2.2 - Representação esquemática de um neurônio artificial.

Para que a rede consiga predizer as saídas corretamente, os pesos utilizados nas expressões anteriores devem estar ajustados. O processo de aprendizado consiste em ajustes sucessivos dos pesos e, em alguns algoritmos, dos resíduos de ativação, de forma a se obter as saídas da rede as mais próximas possíveis das desejadas. O treinamento caracteriza-se então como um problema de otimização onde se busca reduzir os erros de predição. Essa é a etapa mais demorada na utilização das redes neurais e tem gerado um grande número de trabalhos na literatura. No entanto, não há até o momento um algoritmo que se adeque a todas as situações. É essencial, portanto, saber as vantagens e limitações de cada método.

Grande parte dos algoritmos de treinamento existentes na literatura são baseados nos métodos de gradiente descendente e nos métodos de Newton (Vogl *et al* (1988)).

As abordagens baseadas nos métodos de Newton apresentam, em geral, melhores resultados pelo fato de serem métodos de segunda ordem, apresentando uma convergência quadrática próximo ao mínimo. No entanto, estes apresentam como limitação o grande espaço de memória requerido e o volume de cálculos matriciais envolvido, o que os torna praticamente inviáveis para redes de grandes dimensões. Diversos métodos quasi-Newton têm sido propostos com o intuito de reduzir a memória requerida e o volume de cálculos. Estes se baseiam principalmente em simplificações na matriz de Hessian (Robitaille (1996)), reduzindo o tamanho desta e simplificando a sua forma de cálculo.

Os métodos de gradiente descendente não apresentam problemas quanto à memória requerida. No entanto, a taxa de convergência é bem menor que a dos métodos

de segunda ordem, o que tem motivado o desenvolvimento de diversos trabalhos no sentido de melhorar o desempenho destes.

A seguir apresentamos os princípios de dois algoritmos baseados em métodos de otimização diferentes: o Backpropagation (com a regra do delta generalizado), que é um método de gradiente descendente bastante utilizado na literatura; e o método de Marquardt-Levenberg, baseado no método quasi-Newton de mesmo nome e que, a depender do tamanho da rede que se esteja utilizando, é muito mais eficiente que o primeiro.

2.1.1 - O Algoritmo Backpropagation utilizando a Regra do Delta Generalizado (GDR)

Este foi o primeiro método que surgiu para o treinamento de uma RNA direta, tendo sido desenvolvido por Rumelhart *et al* (1986). Uma grande eficiência é obtida quando o Backpropagation é utilizado com processamento em paralelo, para o qual ele foi originalmente criado, sendo que a sua utilização com processamento serial não apresenta as mesmas características. Observa-se que há uma convergência lenta quando os dados são de difícil mapeamento e/ou se requer uma alta precisão. As razões para a lentidão dos métodos de gradiente descendente (dos quais o Backpropagation é um exemplo) são discutidas por Akaike (1959).

Como este método não apresenta nenhum tipo de cálculo matricial (que está presente em muitos algoritmos), a sua velocidade em cada iteração é muito rápida em relação a iteração de outros métodos. Assim, apesar da pequena taxa de convergência, o tempo requerido para cada laço é bem menor. Por outro lado, a medida que o erro diminui, a taxa de atualização dos pesos se torna bem pequena, de forma que, em geral, um número muito maior de iterações é necessário para se conseguir uma mesma redução do erro de predição da rede. Uma compensação entre esses fatores e a consideração do pouco espaço de memória requerido é que vai determinar a vantagem ou desvantagem de utilizá-lo.

A seguir, damos os passos para se treinar uma RNA pelo algoritmo de Backpropagation com a Regra do Delta Generalizado. A demonstração matemática do método pode ser encontrado nos trabalhos de Rumelhart *et al* (1986) e Hagan (1994).

Os passos do Backpropagation, para uma rede de três camadas com configuração (l-m-n) são os seguintes:

(1) Um conjunto de pesos aleatórios é passado como estimativa inicial para a rede; Os resíduos ("*threshold*"), neste método, assumem valor zero na camada de entrada e valor 1 nas demais camadas;

(2) Os dados de entrada são submetidos à rede (que já deve ter a sua topologia definida), calculando-se a saída da rede da forma já exposta acima, através das expressões (2.1) e (2.2). A partir da saída assim obtida (O_k) e da saída desejada ($d_{k,p}$), calcula-se o erro do p-ésimo par de dados para o k-ésimo neurônio de saída:

$$E_{k,p}=d_{k,p}-O_k \qquad (2.4)$$

(3) Calcula-se o gradiente do k-ésimo neurônio da última camada pela seguinte expressão:

$$\delta_{k,p}^{(3)} = E_{k,p} \frac{\partial f(x_k^{(3)})}{\partial x_k^{(3)}} \tag{2.5}$$

onde f(x) é a função de ativação, dada pela equação (2.1) e $x_k^{(3)}$ é a entrada do k-ésimo neurônio da 3ª camada;

(4) Calcula-se o gradiente do j-ésimo neurônio da segunda camada através da expressão de recorrência abaixo:

$$\delta_{j,p}^{(2)} = \left(\sum_{k=1}^{n} \delta_{k,p}^{(3)} w_{j,k}^{(3)} \right) \frac{\partial f(x_j^{(2)})}{\partial x_j^{(2)}} \tag{2.6}$$

onde $w_{j,k}^{3}$ é o peso da conecção entre o j-ésimo neurônio da 2ª camada com o k-ésimo neurônio da 3ª camada;

(5) Calcula-se a atualização dos pesos através das seguintes expressões:

$$\Delta w_{j,k}^{(3),(m)} = \eta \delta_{k,p}^{(3)} a_j^{(2)} + \alpha \Delta w_{j,k}^{(3),(m-1)} \tag{2.7}$$

$$\Delta w_{i,j}^{(2),(m)} = \eta \delta_{j,p}^{(2)} a_i^{(1)} + \alpha \Delta w_{i,j}^{(2),(m-1)} \tag{2.8}$$

onde η e α são respectivamente, a taxa de aprendizado e o momento (parâmetros do método, cujos valores devem ser definidos pelo usuário), e *m* a interação. Desta forma, as atualizações anteriores (da iteração "m-1") devem ser guardadas para que se possa calcular as atualizações atuais;

(6) Os pesos são então atualizados pelas expressões abaixo:

$$w_{i,j}^{(2),(m)} = w_{i,j}^{(2),(m-1)} + dw_{i,j}^{(2),(m)} \tag{2.9}$$

$$w_{j,k}^{(3),(m)} = w_{j,k}^{(3),(m-1)} + dw_{j,k}^{(3),(m)} \tag{2.10}$$

(7) Os passos (2) a (6) são repetidos até que seja atingida a convergência. Neste ponto devemos adotar algum critério de convergência. Um critério comumente utilizado na literatura é dado pela seguinte expressão:

$$V = \sum_{k=1}^{n} \sum_{p=1}^{NP} E_{p,k}^2 \tag{2.11}$$

Este critério é global, ou seja se o valor de V for suficientemente baixo, o algoritmo terá convergido, mesmo que alguns pontos não estejam se ajustando tão bem quanto outros. Caso seja interessante que o maior erro não ultrapasse um determinado valor percentual, por exemplo, um outro critério baseado no maior erro dentre todos do conjunto de dados pode ser utilizado no lugar deste.

Embora para uma rede com três camadas a nomenclatura utilizada possa ser bem mais simples que a que foi aqui usada, preferimos colocá-la desta forma para uma compreensão de como o algoritmo pode ser generalizado para várias camadas escondidas. Para tal fim, todos valores referente à 3ª, 2ª e 1ª camada devem ser entendidos como referentes à c, (c-1) e (c-2)-ésima camada e o cálculo deve ser feito com c variando de 3 ao número total de camadas (NC). Alguns autores incluem ainda um conjunto de pesos antes dos neurônios da primeira camada. Neste caso, o calculo deveria ser feito de c variando de 2 a NC.

A taxa de aprendizado e o momento (η e α) são parâmetros que devem ser testados para cada conjunto de dados de forma a se ter um bom aprendizado. Valores típicos de η variam entre 0 e 10 e de α de 0 a 1, conforme utilizado por diversos autores.

Como já mencionamos, uma série de modificações têm sido propostas para melhorar a convergência do Backpropagation. Algumas delas propõem que outras

metodologias sejam utilizadas no lugar da Regra do Delta Generalizada (GDR). Um exemplo é o Backpropagation utilizando o Gradiente Conjugado (Leonard e Kramer (1989)), onde os parâmetros η e α são otimizados a cada passo do algoritmo e a atualização dos pesos, ao invés de ser feita para cada par de treinamento lido, é feito de uma única vez depois que todos as variações dos pesos (Δw's) são calculadas. Outras modificações sugerem simplesmente que se varie os parâmetros do algoritmo através de regras heurísticas simples, de acordo com o aumento ou a diminuição do erro quadrático médio (conforme definido pela equação (2.11)), como é feito por Vogl *et al* (1988).

No entanto, nenhuma destas técnicas melhora a convergência do método próximo ao mínimo (que é linear). Uma implementação deste algoritmo em Fortran 77 encontra-se no Apêndice C (sub-rotina TREIN1, a qual é gerenciada pela sub-rotina RNA).

2.1.2 - O Algoritmo de Marquardt-Levenberg.

Este algoritmo é baseado no método quasi-Newton de otimização de Marquardt-Levenberg e foi proposto por Hagan *et al* (1994).

Suponhamos que se deseje minimizar uma função $V(\mathbf{x})$ qualquer em relação ao vetor $\mathbf{x}$. Pelo método de Newton, temos que as atualizações nos elementos de $\mathbf{x}$ seriam dadas por:

$$\Delta\mathbf{x} = -\left[\nabla^2 V(\mathbf{x})\right]^{-1} \nabla V(\mathbf{x}) \tag{2.12}$$

Se esta função V for uma soma de quadrados (semelhante à equação (2.11)), pode-se mostrar que:

$$\nabla V(\mathbf{x}) = J^T(\mathbf{x})E(\mathbf{x}) \tag{2.13}$$

$$\nabla^2 V(\mathbf{x}) = J^T(\mathbf{x})J(\mathbf{x}) + S(\mathbf{x}) \tag{2.14}$$

onde $J(\mathbf{x})$ é o jacobiano de $V(\mathbf{x})$ e $S(\mathbf{x})$ é definido por:

$$S(\mathbf{x}) = \sum E_i(\mathbf{x})\nabla^2 E_i(\mathbf{x}) \tag{2.15}$$

No método de Gauss-Newton, assume-se que $S(\mathbf{x}) \approx 0$, de forma que a equação (2.12) pode ser reescrita da seguinte forma:

$$\Delta\mathbf{x} = -\left[J^T(\mathbf{x})J(\mathbf{x})\right]^{-1} J^T(\mathbf{x})E(\mathbf{x}) \tag{2.16}$$

A modificação do método de Marquardt-Levenberg ao método de Gauss-Newton é:

$$\Delta\mathbf{x} = -\left[J^T(\mathbf{x})J(\mathbf{x}) + \mu I\right]^{-1} J(\mathbf{x})E(\mathbf{x}) \tag{2.17}$$

onde I é a matriz identidade e μ é um escalar.

Desta forma, podemos calcular a variação dos pesos a cada iteração utilizando a expressão (2.17), desde que tenhamos disponível o jacobiano da soma dos quadrados dos erros em relação aos pesos da rede. Os passos para a aplicação deste algoritmo são os seguintes:

(1) Estima-se um valor inicial de μ . (neste trabalho utilizamos o valor sugerido por Hagan (1994) que é: $\mu = 0{,}01$);

(2) Calcula-se o valor da equação (2.11) bem como o jacobiano desta em relação ao pesos;

(3) Através da expressão (2.17), calcula-se os novos pesos;

(4) Se estes novos pesos representarem um aumento de V, abandona-se estes, retornando aos valores anteriores; μ é então multiplicado por um valor $\beta > 1$ (neste trabalho utilizou-se $\beta = 1{,}2$) e volta-se ao passo (2). Se os novos pesos reduzirem V, estes são aceitos e μ é dividido por β . Se ainda não foi atingida a convergência, retorna-se ao passo (2).

A chave deste algoritmo é, portanto, o cálculo do jacobiano de (2.11) em relação aos pesos da rede neural. Embora este possa ser calculado numericamente, isto representaria um número de chamadas da rede igual ao número total de pesos desta, o que tornaria o cálculo bastante pesado para uma arquitetura maior. No entanto, Hagan *et al* (1994) mostram que, a partir de pequenas modificações no algoritmo Backpropagation exposto anteriormente, é possível se calcular o jacobiano com uma única chamada a rede neural.

O Algoritmo de Marquardt apresenta um convergência mais rápida que o Backpropagation para redes com até um certo número de pesos. A medida que o número de pesos aumenta muito, a inversão de matriz da equação (2.17), bem como o espaço de memória necessário para o armazenamento desta, tornam este método inviável. Comparações feitas por Hagan *et al* (1994) mostram que o método é superior ao Backpropagation quando se tem uma rede com até algumas centenas de pesos.

Uma outra característica importante deste algoritmo é que, para valores altos de μ, o algoritmo se torna gradiente descendente e para valores pequenos, este se torna o método de Gauss-Newton. Assim este método alterna entre um método e outro, apresentando a vantagem de ambos. De fato, para pontos longe do mínimo, os métodos de gradiente descendente conferem melhor resultado que os métodos de Newton (que só apresentam convergência quadrática próximo ao mínimo), e é por isto que o algoritmo inicia com um valor alto de μ. A medida que o erro diminui, o valor de μ diminui e o método se aproxima do método Gauss-Newton, apresentando a convergência quadrática neste ponto. É de se esperar, portanto, que a não ser em casos onde o número de pesos é muito grande, os resultados obtidos com este método sejam muito superiores.

Uma implementação deste algoritmo em Fortran 77 encontra-se no apêndice C (sub-rotina TREIN2, a qual é gerenciada pela sub-rotina RNA).

2.1.3 - Adimensionalização de entradas e saídas.

Até agora tratamos as entradas e saídas como se estas pudessem ser diretamente apresentadas às redes neurais. No entanto, deve-se lembrar que as redes utilizam uma função sigmoidal cuja saída varia entre zero e um. Além disto, se valores muito altos ou muito baixos são utilizados, a saída da função sigmoidal é praticamente um dos seu limites (0 ou 1). Desta forma as entradas e saídas utilizadas nas etapas de treinamento e

inferenciação devem ser adimensionalizadas. Obviamente, a adimensionalização utilizada no treinamento deve ser a mesma da utilizada durante a etapa de predição.

Os valores entre os quais as entrada e saídas devem variar não são fixos e muitos autores utilizam de diferentes formas. Neste trabalho utilizamos as entradas e saídas variando sempre entre inf=0,1 e sup=0,9. Dois tipos de adimensionalização foram utilizadas para tanto: uma linear, e outra baseada na função logarítmica. A expressão linear para adimensionalizar as entradas de um determinado vetor **x** foi a seguinte:

$$y(i)=a+b*x(i) \tag{2.18}$$

onde os valores "a" e "b" são definidos, por:

$$a = \frac{(inf * maior - sup * menor)}{(maior - menor)} \tag{2.19}$$

$$b = \frac{sup - inf}{maior - menor} \tag{2.20}$$

onde sup e inf são ,respectivamente, os valores superior e inferior das variáveis após adimensionalização; e maior e menor são, respectivamente, o menor e o maior valor do vetor **x**.

De forma semelhante, a adimensionalização logarítmica utilizada é dada pelas seguintes expressões:

$$y(i) = \frac{\log_{10}(x(i) - a)}{b} \tag{2.21}$$

onde:

$$a = \frac{sup * \log_{10}(menor) - inf * \log_{10}(maior)}{sup - inf} \tag{2.22}$$

$$b = \frac{\log_{10}(maior / menor)}{sup - inf} \tag{2.23}$$

Os valores de "a" e "b" utilizados no treinamento devem ser os mesmos utilizados na inferenciação.

A adimensionalização logarítimica se torna melhor que a linear quando se utiliza um conjunto de dados onde o maior valor é muito superior ao menor valor. A tabela 2.1 mostra bem o caso em que a adimensionalização logarítmica dá melhores resultados que a linear. Nestes dados temos o maior valor mais de mil vezes superior ao menor. Quando aplicamos a transformação linear, os dados menores ficam bem próximos ao valor inferior estipulado, o que dificulta o treinamento das redes. Mas quando aplicamos a transformação logarítimica, a distribuição dos dados fica mais uniforme, com os valores iniciais mais espaçados entre si, o que facilita o treinamento da rede (como mostra o gráfico 2.1).

Tabela 2.1 - Adimensionalização linear e logaritmica para um conjunto de dados.

Conjunto Original	Adimensionalização linear	Adimensionalização logarítmica
7,13E-04	0,100000	0,100000
8,53E-04	0,100057	0,118058
1,67E-03	0,100391	0,185929
4,00E-03	0,101346	0,274262
7,82E-03	0,102909	0,341960
9,34E-03	0,103532	0,359933
1,12E-02	0,104277	0,377915
2,17E-02	0,108603	0,445243
5,13E-02	0,120723	0,532137
9,78E-02	0,139777	0,597362
1,16E-01	0,147142	0,614417
1,37E-01	0,155760	0,631288
2,50E-01	0,202252	0,692335
5,07E-01	0,307419	0,763675
7,93E-01	0,424477	0,808849
8,78E-01	0,459446	0,819185
9,66E-01	0,495356	0,828801
1,30E+00	0,630644	0,858527
1,68E+00	0,786357	0,884521
1,90E+00	0,879062	0,897320
1,95E+00	0,900000	0,900000

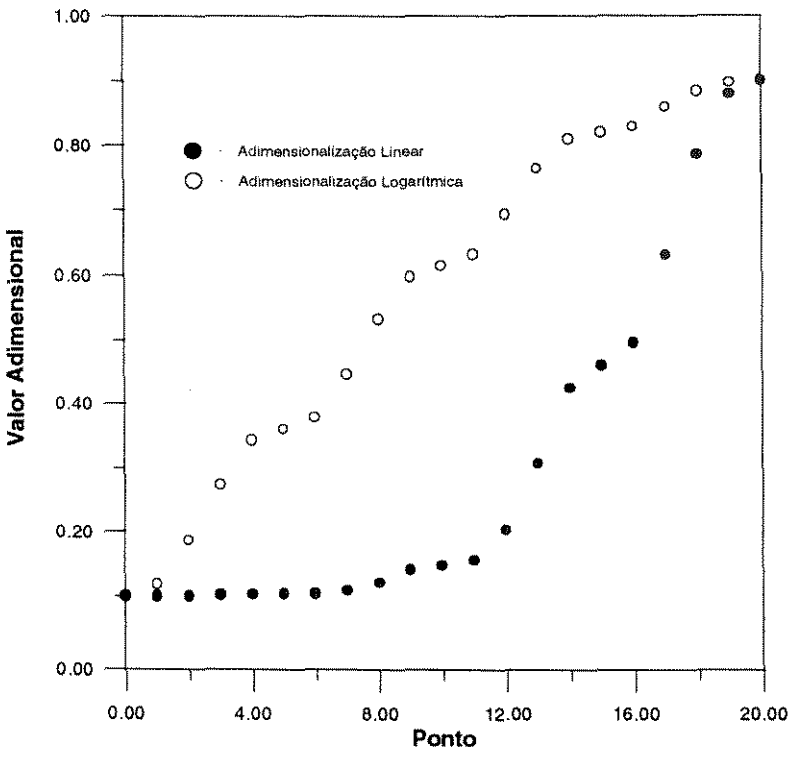

Gráfico 2.1 - Comparação entre a adimensionalização linear e a logarítimica.

2.1.4 - O conceito de janela de tempo para simulação dinâmica.

As redes neurais diretas, da forma como exposto até aqui, são essencialmente estáticas, não servindo para reter características de processos dinâmicos. Outros tipos de redes e algoritmos de treinamento têm sido propostos para tal fim. Hunt (1992) dá uma revisão de algumas redes inerentemente dinâmicas mais comumente utilizadas. Estas redes, no entanto, são recursivas, ou seja, os valores preditos por elas são utilizados com entrada para a predição de valores em instantes posteriores. As redes recursivas podem sofrer de problemas de instabilidade.

Por outro lado, as redes diretas também podem ser utilizadas em processos dinâmicos, desde que seja utilizada o conceito de janela de tempo móvel (Bhat *et al* (1990)). A fig. 2.3 ilustra este método para um janela de tamanho 8 (quatro valores passados e quatro futuros). Os dados que serão utilizados pela rede constituem de valores passados e futuros das entradas originais, bem como de dados passados dos valores de saída desejados (retângulos hachurados na fig. 2.3). As saídas utilizadas durante o treinamento são constituídas de valores futuros dos valores a serem preditos pela rede após a etapa de treinamento. O termo passado e futuro é aqui utilizado em relação a um tempo central, que se vai modificando para cada par de treinamento. Assim, o treinamento consiste em mover a janela por todo o conjunto de dados até que o critério de convergência seja atingido.

Deve-se observar que na etapa de inferenciação, se faz necessário conhecer um certo histórico do sistema, de forma que se tenham alguns valores passados para que se possa predizer os valores futuros.

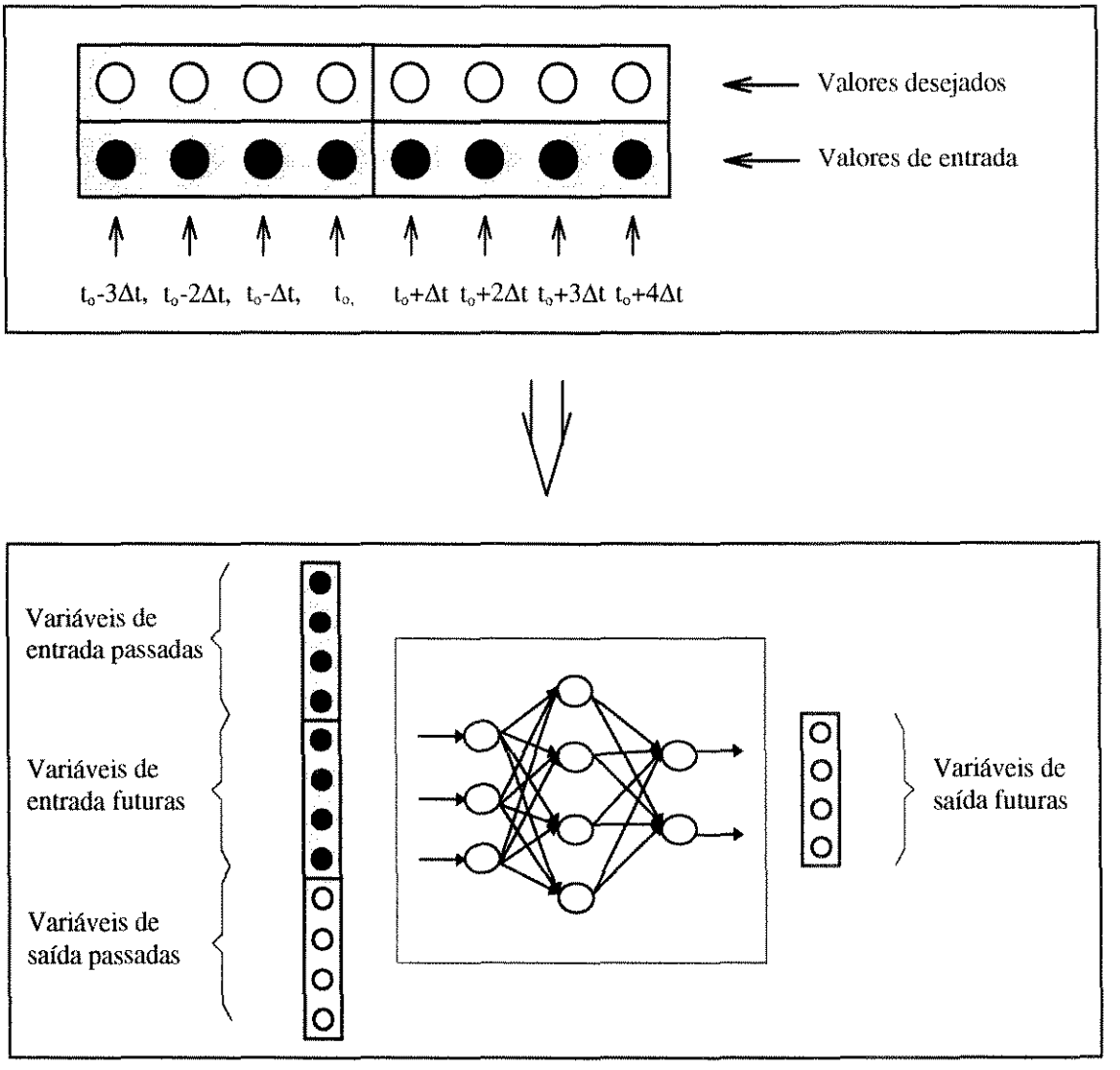

Fig 2.3 - Janela de tempo móvel: as áreas hachuradas correspondem aos valores de entrada; os círculos representam os dados.

2.2 - Colocação Ortogonal.

2.2.1 - Colocação Ortogonal Global.

Uma das abordagens mais utilizadas para resolução de equações diferenciais (principalmente as parciais) é a aproximação da solução em uma série de funções de coeficientes desconhecidos. Os coeficientes devem, então, ser ajustados por algum critério que aproxime esta função da melhor forma possível à solução.

Uma escolha "natural" para esta função seria um polinômio, tanto pelas suas características de aproximador universal de funções, como pela sua simplicidade. Vale aqui lembrar que uma função que seja diferenciável em qualquer ordem, pode ser expressa por uma série de taylor.

$$f\left(y^{(n)}, y^{(n-1)}, \ldots, y^{(1)}, y\right) = 0 \tag{2.24}$$

Assim, por exemplo, se tivermos uma equação diferencial do tipo descrito pela equação (2.24), uma "boa" escolha para a aproximação de "y", seria:

$$y_N = \sum_{i=1}^{N} a_i x^{i-1} \tag{2.25}$$

A equação (2.24) estaria sujeita a *n* condições de contorno que devem ser satisfeitas pela aproximação (2.25). Se substituirmos a solução aproximada (2.25) na equação (2.24), teremos um resíduo, R(x,y) que, *a priori*, apresenta um valor não nulo. No método dos resíduos ponderados (Finlayson (1980)), temos a seguinte condição:

$$\int_0^1 W_k R(x, y_N) dx = 0 \tag{2.26}$$

Logo, neste método, os resíduos ponderados por um peso, W_k, são forçados a serem zero ao longo de um intervalo (0,1). A escolha do peso é que vai dar as características de cada método. Por exemplo, no Método da Colocação, utiliza-se a função delta de dirac (2.27). Da mesma forma, podemos citar outros métodos, como: método de Galerkin, método dos momentos, método dos mínimos quadrados, etc.

$$W_k = \delta(x - x_k) \tag{2.27}$$

O uso da função delta de dirac na equação (2.26) nos dá a seguinte relação:

$$R(x_k, y_N) = 0 \tag{2.28}$$

Ou seja, no método da colocação o resíduo é zero nos pontos de colocação x_k.

O método da colocação, como descrito acima, tem a desvantagem de se ter que escolher os pontos de colocação, ou seja, se tivermos um polinômio de grau *n* , temos que escolher os *n* pontos onde o resíduo será zero. A partir da equação (2.26), teríamos então um sistema de *n* equações (dadas pela cálculo do resíduo para cada valor de x) e *n* incógnitas (que são os coeficientes a_i na equação 2.25). O cálculo dos resíduos e resolução do sistema resultante, se torna, portanto, algo extremamente tedioso. Outra desvantagem deste método, é que a escolha dos pontos não segue nenhum critério, tornando-se algo aleatória e podendo levar a resultados pouco satisfatórios.

Embora haja uma série de métodos baseados na soma de resíduos ponderados, a precisão destes varia muito, como mostra Villadsen e Michelsen (1978). Esses autores mostram que o método de Galerkin é o mais preciso comparado a outros métodos com polinômios de mesma ordem. No entanto, eles mostram que é possível obter um método de precisão comparável ao método de Galerkin quando se utiliza o método da colocação

com os pontos de colocação sendo as raízes de um polinômio do tipo $P_j^{(\alpha,\beta)}(u)$, onde este polinômio é definido pela relação:

$$\int_0^1 u^{\alpha}(1-u)^{\beta} P_j^{(\alpha,\beta)}(u) P_i^{(\alpha,\beta)}(u)du = 0 \qquad (2.29)$$

A equação (2.29) expressa a relação de ortogonalidade para os polinômios de Jacobi, onde α e β são constantes. De uma forma geral, dois polinômios são ditos ortogonais no intervalo (a,b) se obedecem a seguinte relação:

$$\int_a^b W(u)P_1(u)P_2(u)du = 0 \qquad (2.30)$$

A função peso, W(u), é que vai definir a família de polinômios.

No método da colocação, quando a solução for aproximada pela equação (2.25), temos que nos pontos de colocação, esta equação pode ser escrita da seguinte forma:

$$y = \sum_{i=1}^{N} a_i x_k^{\,i-1} \qquad (2.31)$$

Derivando-se esta equação, podemos escrever as derivadas na seguinte forma matricial (Finlayson, 1980):

$$\frac{\mathbf{dy}}{\mathbf{dx}} = \mathbf{CQ^{-1}} \equiv \mathbf{Ay} \qquad \frac{\mathbf{d^2y}}{\mathbf{d^2x}} = \mathbf{DQ^{-1}} \equiv \mathbf{By} \qquad (2.32a)$$

$$\frac{dy}{dx}(x_j) = \sum_{i=1}^{N} A_{ji} y_i \qquad \frac{d^2y}{d^2x}(x_j) = \sum_{i=1}^{N} B_{ji} y_i \qquad (2.32b)$$

$$Q_{ji} = x_j^{i-1} \qquad C_{ji} = (i-1)x_j^{i-2} \qquad D_{ji} = (i-1)(i-2)x_j^{i-3} \qquad (2.33)$$

Desta forma, temos que as derivadas podem ser expressas em função do próprio valor da função nos pontos de colocação e de coeficientes que não precisam ser calculados utilizando-se a equação (2.26), mas podem ser obtidos a partir dos pontos de colocação. A partir deste resultado e, como já discutido acima, de que um método muito eficiente é obtido quando os pontos de colocação são utilizados como sendo as raízes de um polinômio ortogonal, temos o método da colocação ortogonal global, ou simplesmente, colocação ortogonal.

Os polinômios ortogonais de Jacobi apresentam uma série de propriedades que permitem o cálculo dos seus zeros e as matrizes de colocação através de procedimentos numéricos. Villadsen & Michelsen (1978) descrevem algumas destas propriedades e mostram como elas podem ser utilizadas no cálculo dos zeros dos polinômios de Jacobi e de outros valores de interesse no método da colocação ortogonal (como as matrizes **A** e **B**). Estes autores também apresentam subrotinas em Fortran IV que fazem esses cálculos.

Desta forma, o método da colocação ortogonal apresenta uma precisão comparável ou igual a do método de Galerkin, com a vantagem de se ter todos os coeficientes facilmente determinados a partir dos zeros de polinômios ortogonais, o que justifica o grande número de trabalhos na literatura que tem utilizado a colocação ortogonal na solução de sistemas de equações diferenciais. Embora a nossa discussão tenha praticamente se restringido aos polinômios de Jacobi, existem outras famílias de polinômios que são utilizados, principalmente quando se tem um sistema discreto (como é o caso de colunas de prato), onde a utilização de polinômios contínuos só é possível a partir de certos artifícios.

Assim, podemos resumir a utilização do método da colocação nos seguintes passos:

•Adimensionalização das variáveis independentes, de forma que estas passem a variar nos intervalo (0;1), já que uma das propriedades dos polinômios definidos pela equação (2.26) é que os seus zeros estão todos no intervalo (0;1);

•Escolha do grau do polinômio e cálculo das matrizes de colocação (matriz **A** para derivadas de 1° ordem, **B** para as de 2° ordem, etc) a partir das raízes do polinômio;

•A partir das equações (2.32a), as derivadas são então transformadas em expressões algébricas. Todo o cálculo, deste ponto em diante, é feito nos pontos de colocação, sendo as equações diferenciais ordinárias transformadas em equações algébricas e as equações diferenciais parciais (com derivada em relação ao tempo) transformadas em equações diferenciais ordinárias. Resolvendo-se o sistema de equações resultante, temos a resposta nos pontos de colocação. Nos capítulos 3 e 4, bem como no apêndice A, mostraremos a aplicação do método a algumas equações resultantes da modelagem da absorção com reação química.

No caso de equações diferenciais parciais (com derivadas em relação ao tempo), há aplicações na literatura que adimensionalizam também o tempo, permitindo o uso da colocação ortogonal em toda a equação e reduzindo, desta forma, a equação diferencial parcial a uma equação algébrica. Este método (conhecido por dupla colocação) é discutido por Villadsen (1969).

Sobre a aplicação do método da colocação ortogonal, cabe fazer alguns comentários:

•Os pontos onde se tem a solução da equação ou do sistema de equações diferenciais são os zeros do polinômio ortogonal. Desta forma, escolhido o grau do polinômio, o único parâmetro que resta para ajustar a posição deste pontos são as constantes α e β, da equação (2.29). Quanto maior o valor de α, mais deslocados estarão os pontos para a direita no eixo x; da mesma forma, quanto maior o valor de β, mais deslocados os pontos estarão para a esquerda, conforme mostra a tabela (2.2) para três pontos de colocação internos. A escolha dos pontos de colocação é importante não só para se ter a solução em um local mais próximo do desejado, mas principalmente porque esta posição influencia na resposta obtida, já que, como vimos na equação (2.32a), a solução depende de todos os pontos de colocação. Assim, é importante se ter os pontos localizados em regiões que a resposta do sistema varie muito.

•Como no método da colocação os resíduos são zero nos pontos de colocação (2.28), seria de se esperar que quanto mais pontos de colocação fossem utilizados, mais a solução se aproximaria do valor real (Villadsen e Michelsen, 1978). No entanto, deve-se observar que a solução esta sendo aproximada por um polinômio. Desta forma, se tivermos, por exemplo, uma função que tenha regiões de grandes variações e regiões quase lineares (como é o caso dos perfis de concentração no filme de transferência de massa em absorção com reação química), a aproximação com um número muito alto de pontos pode levar a oscilações na resposta. O número muito alto de pontos também leva a um sistema maior de equações, o que torna a sua resolução mais lenta. Desta forma, deve-se ter um compromisso entre um número de pontos não muito baixo (o que levaria a uma resposta imprecisa) e um número muito alto (que levaria a uma resposta lenta e com possíveis oscilações).

Tabela 2.2 - Distribuição dos pontos com os valores de α e β.

		Pontos de Colocação		
α	β	x_2	x_3	x_4
0	0	0,1127	0,5000	0,8873
0	1	0,2123	0,5905	0,9114
1	0	0,0886	0,4095	0,7876
1	1	0,1727	0,5000	0,8273

2.2.2 - Colocação Ortogonal em Elementos Finitos.

A importância da localização dos pontos de colocação (já discutida acima), pode tornar-se, algumas vezes, bastante crítica. Um exemplo clássico é dado por Carey e Finlayson (1979), que mostra o problema de determinação do perfil de concentração ao longo de um poro de catalisador. Este perfil cai a praticamente zero rapidamente (em x=0,001). Assim, teríamos que ter um número muito alto de pontos de colocação para reconstituir a região de importância neste caso, ou seja, para se ter a concentração na região entre (0; 0,001).

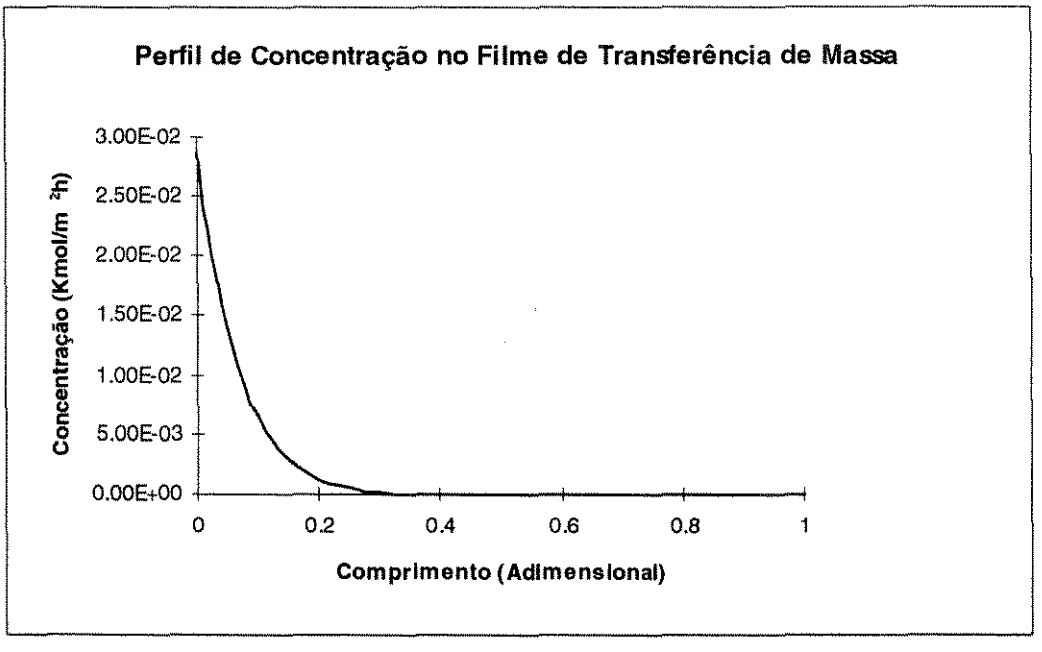

Gráfico 2.2 - Perfil típico no filme de transferência de massa na absorção com reação química.

Por outro lado, a existência de uma região onde a função varia muito seguida de uma outra onde ela é praticamente uma reta, pode causar oscilações na solução obtida por colocação ortogonal.

Situações semelhantes a do problema apresentado por Carey e Finlayson (1979) são obtidas no caso da absorção com reação química, como podemos observar no gráfico 2.2

Estes problemas são contornados utilizando-se o método da colocação ortogonal em elementos finitos. Neste método, ao invés de termos apenas um polinômio aproximador, o domínio é dividido em vários subintervalos e um polinômio é utilizado em cada um deste subintervalos (fig. 2.4).

Consegue-se assim as vantagens da colocação ortogonal aliadas a capacidade de se poder ter os pontos nos locais desejados (características do método das diferenças finitas).

A aplicação deste método segue os mesmos passos básicos da colocação ortogonal global, sendo que o domínio da variável independente, que deve estar adimensionalizada e variando no intervalo (0;1), é dividido em "l" subintervalos e a colocação é então aplicada de forma semelhante ao já exposto.

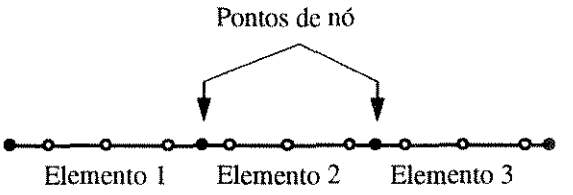

Figura 2.4 - Localização dos pontos: os pontos em negrito correspondem aos pontos de nó, enquanto os pontos claros são os pontos internos de colocação.

Desta forma, sendo "x" a variável espacial já adimensionalizada (conforme utilizamos acima), escolhemos "l+1" pontos (x_1, x_2, x_3,...,x_l,x_{l+1}) que serão os pontos de nós entre os "l" elementos (figura 2.4). Uma nova variável "u_l" deve ser então definida entre os pontos x_l e x_{l+1}, pela equação (2.20), de tal forma que esta varie no intervalo (0, 1) no elemento "l", tornando possível a aplicação da colocação ortogonal.

$$u_l = \frac{x - x_l}{x_{l+1} - x_l} = \frac{x - x_l}{\Delta x_l} \tag{2.34}$$

As condições de contorno, no entanto, precisam ser definidas para cada elemento, já que temos uma equação diferencial aplicada a cada elemento. Logicamente, estas condições devem ter um significado físico e devem ser em número compatível com a ordem da equação diferencial. No exemplo da figura acima, por exemplo, temos uma equação de segunda ordem. As condições utilizadas neste caso foram a da igualdade de concentrações e de fluxo dos componentes através dos elementos (igualdade da primeira derivada).

É importante notar que um ponto de nó corresponde ao último ponto de um elemento e ao primeiro ponto do elemento seguinte. Assim, a condição de igualdade de concentrações entre dois elementos é direta.

No apêndice A, a aplicação da colocação ortogonal em elementos finitos para a resolução de equações de transferência de massa é tratada em maiores detalhes.

2.3 - Método das Características.

Muitos processos em engenharia química se dão em meios contínuos, tais como adsorção, destilação e absorção em colunas recheadas, etc. As equações diferenciais que descrevem a dinâmica deste tipo de processo geralmente são parciais do tipo hiperbólica, possuindo derivadas no tempo e no espaço. Uma grande dificuldade na resolução destas equações surge devido ao fato de se ter as descontinuidades introduzidas no processo se propagando através do equipamento com uma velocidade finita. O perfil inicial que é passado como condição inicial na resolução do sistema de equações contêm, desta forma, uma descontinuidade.

Os métodos mais comumente utilizados na literatura para a solução deste tipo de problema são o das diferenças finitas e uso do modelo de células de mistura (Tan e Spinner, 1984a). Este método, no entanto, implicitamente acrescenta um termo de dispersão que amortiza as descontinuidades impostas ao sistema. Para uma melhor representação do sistema, um grande número de células ou maior grade no método das diferenças finitas deve ser utilizado, o que pode levar a um grande esforço computacional, sem que o problema seja totalmente contornado.

O método das características permite a resolução das equações diferenciais hiperbólicas sem amortizar as descontinuidades. Este método se vale do fato das equações diferenciais hiperbólicas terem curvas características ao longo das quais as descontinuidades se propagam. O conceito de derivada substantiva é utilizado no desenvolvimento deste método. Assim, temos que se a variável c for função do tempo e das cordenadas espaciais (x,y,z), a sua derivada substantiva é definida como (Bird, 1960):

$$\frac{Dc}{Dt} = \frac{\partial c}{\partial t} + v_x \frac{\partial c}{\partial x} + v_y \frac{\partial c}{\partial y} + v_z \frac{\partial c}{\partial z} \qquad (2.35)$$

onde v_x, v_y e v_z são respectivamente as velocidades de propagação de um elemento de fluido nas direções x, y e z.

Desta forma, temos que a derivada substantiva representa a derivada de uma variável do elemento do ponto de vista de um observador que se move com a mesma velocidade do elemento. É este aspecto que a torna interessante na resolução de equações diferenciais hiperbólicas, pois a integração é feita seguindo-se uma partícula de fluido que entra no equipamento. Uma variação em degrau na variável de interesse não causa problema já que cada elemento é integrado em separado, seguindo a sua própria característica. Além disto, o método transforma as equações diferenciais parciais em equações diferenciais ordinárias.

O método das características foi originalmente utilizado em problemas de fluidos compressíveis. Uma abordagem matemática mais rigorosa deste método é dada por Friedly (1972). É importante ressaltar que este é apenas uma aproximação que exige

(como será notado pelo exemplo a seguir) que as equações sejam colocadas sob determinada formato. A depender da complexidade das equações, a aplicação deste método se tornar bastante complicada.

As primeiras aplicações deste método a problemas de engenharia química foram feitas por Acrivos (1956). Dentre estas aplicações, foi apresentada a simulação de uma coluna de absorção recheada. Embora o modelo utilizado seja bastante simplificado, o exemplo é interessante para compreensão do método. A metodologia utilizada no exemplo a seguir não é exatamente a mesma a ser utilizada na modelagem, pois, como veremos, esta apresenta algumas limitações.

Considere uma coluna de absorção isotérmica e cujas misturas sejam diluídas e onde não ocorra reação química. As equações básicas que descrevem este processo são:

$$L\frac{\partial x}{\partial z} - H_L\frac{\partial x}{\partial t} = -R(x,y) \tag{2.36a}$$

$$G\frac{\partial y}{\partial z} + H_G\frac{\partial y}{\partial t} = R(x,y) \tag{2.36b}$$

onde:

- L = Vazão molar de líquido por área transversal de recheio;
- G = Vazão molar de gás por área transversal de recheio;
- x = Fração molar do soluto na fase líquida;
- y = Fração molar do soluto na fase gás;
- z = Distância da base do recheio;
- t = Tempo;
- R(x,y) = Taxa de transferência de massa;
- H_L = Acúmulo de líquido;
- H_G = Acúmulo de gás;

Da nomenclatura acima, temos que a velocidade do líquido e do gás podem ser expressas, respectivamente, pelas equações (2.37a,b):

$$\frac{dz}{dt} = -\frac{L}{H_L}, \text{ para a família de características I} \tag{2.37a}$$

$$\frac{dz}{dt} = \frac{G}{H_G}, \text{ para a família de características II} \tag{2.37b}$$

Desta forma, as equações (2.36a,b) abaixo podem ser transformadas, utilizando o conceito de derivada substantiva, para:

$$\left(\frac{dx}{dz}\right)_I = -\frac{R(x,y)}{L} \tag{2.38a}$$

$$\left(\frac{dy}{dz}\right)_{II} = \frac{R(x,y)}{G} \tag{2.38b}$$

onde a nomenclatura utilizada acima significa que as equações (2.38a,b) são válidas ao longo das características I e II descritas pelas equações (2.37a,b).

Se os fluxos e holdups de líquido e gás forem constantes, temos que as equações (2.37a,b) podem ser integradas:

$$t = -\frac{H_L}{L}z + l\Delta t \tag{2.39a}$$

$$t = \frac{H_G}{G} z + k\Delta t \qquad (2.39b)$$

As constantes l e k nas equações acima definem as características I (2.39b) e II (2.39b). A figura 2.5 mostra essas características para diferentes valores de l e k.

Os índices utilizados na fig. (2.5) têm o seguinte significado: uma partícula que entra no coluna irá se mover ao longo de uma das características citadas, ou seja, a sua posição irá variar com o tempo de acordo com equação (2.39a) ou (2.39b). Desta forma, uma partícula de gás que entra no instante inicial, no fundo da coluna, está no ponto [0,0]. No instante seguinte (em t=Δt) a partícula terá se movido no esquema acima, para o ponto [0,1], estando na posição z=Δz. De forma semelhante, um elemento de líquido que esteja na posição z=Δz, posição [0,1], no tempo inicial, terá se movido para o ponto [1,1] no instante t=Δt (estando na posição z=0).

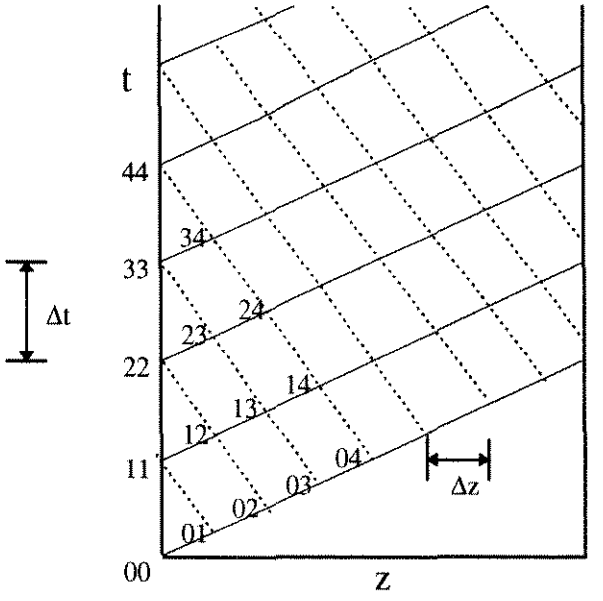

Fig. 2.5 - Família de características para diferentes valores de l e k.
——— Características I.
..... Características II.

Suponhamos que estejamos interessados em simular a partida da coluna. Desta forma, as condições iniciais são as seguintes: para t≤0, a fase líquida está livre de soluto (x=0 e y=y_o). Além disto, as composições x e y são conhecidas a qualquer tempo em z=0 e z=Z, respectivamente. A integração pode então ser feita da seguinte forma: a partir de x[0,0], o valor de x[0,1] pode ser calculado ao longo da característica I, a partir da equação (2.38), usando-se o método de Euler modificado:

$$x^1[0,1] = x[0,0] - \frac{R[0,0] + R^1[0,1]}{2}\Delta z \qquad (2.40)$$

O índice sobrescrito na equação (2.40) indica uma primeira aproximação, já que para se calcular R[0,1], deve-se conhecer x[0,1]. O método, é portanto, iterativo. No entanto, este converge rapidamente, com pouco esforço computacional. Os outros

valores de x ao longo desta característica podem então ser calculados de forma semelhante.

A integração para a fase líquida se dá da mesma forma, ao longo das características II. Sendo conhecido o valor de y[0,1], por exemplo, y[1,1] é calculado por:

$$y^{1}[1,1] = y[0,1] + \frac{R[0,1] + R^{1}[1,1]}{2}\Delta z \qquad (2.41)$$

Procedendo-se desta forma, construímos o restante da malha, obtendo os valores de x e y em qualquer ponto desejado.

Este método, da forma como foi exposto aqui - que é basicamente a abordagem utilizada por Acrivos (1956) - possui algumas limitações. Dentre estas, podemos citar:

•A consideração básica para que se pudesse traçar a família de características (fig. 2.5) é de que os fluxos e holdups de gás e de líquido sejam constantes. Variações na carga de líquido ou de gás causariam uma mudança na inclinação destas, o que tornaria a sua resolução bem mais complexa e para a qual o esquema anterior não se adequaria. Além disto, quando se trabalha com soluções não diluídas, os fluxos variam ao longo da coluna, e as inclinações não seriam conhecidas *a priori* como no caso anterior, variando ao longo da coluna. Um exemplo deste caso é citado por Acrivos. No entanto, ele cita o caso em que apenas um dos fluxos varia ao longo da coluna, enquanto no caso da absorção, os dois fluxos podem variar ao longo da coluna, o que complica bastante o procedimento.

•Da forma como foi construída, a malha nos dá os valores na fase gás e fase líquida em tempos diferentes (já que os fluxos têm velocidades diferentes, para que percorram um mesmo Δz, é necessário um tempo diferente). Assim, seria necessário construir boa parte da malha para então se ter a composição em instantes e posição desejados.

•O método nos propõe que seria possível determinar as composições nos pontos [0,1], [0,2], [0,3], ..., [0,N], onde N é o número de pontos na malha, a partir do valor x[0,0] e $y=y_0$. Após isto, ao longo das características II, seria possível determinar y nos pontos [1,1], [1,2], [1,3], etc; e por conseguinte, determinar x (ao longo da característica I e a partir da condição inicial x[1,1]) nestes mesmos pontos. Note que no entanto, para determinar o ponto [1,2], teríamos que ter determinado o valor de R(x[1,2],y[1,2]). Assim, o valor do fluxo vai depender de ambas composições, sendo que pelo raciocínio acima, consideramos que uma delas não variou para que se pudesse fazer o cálculo da outra. Desta forma, pequenos erros são introduzidos no cálculo. Estes erros não são citados por Acrivos e nenhum dos trabalhos seguintes que se utilizou deste método (como Tan e Spinner, 1984a,b; Bradley e Andre 1972, etc).

Tan e Spinner (1984a,b) propuseram variações no método da forma como foi colocado por Acrivos para evitar a iteratividade que aparece em alguns exemplos e as interpolações na malha de cálculo. No entanto, estes não propuseram nenhuma solução para os problemas citados acima. Desta forma, a resolução de um modelo que inclua variações de fluxos mássicos ao longo da coluna, necessita de uma metodologia diferente. Como não foi encontrado nenhum trabalho na literatura que se adequasse a resolução das equações obtidas na modelagem deste trabalho, utilizamos o método das características com as seguintes modificações:

•Ao invés de se escolher um Δz fixo, escolhemos um Δt e, a partir das equações de velocidade do gás e do líquido, calcula-se a posição da partícula no instante seguinte. Isto faz com que tenhamos a resposta do sistema em intervalos de tempos iguais, ao contrário do que teríamos com a metodologia acima. Por outro lado, fixando-se o intervalo de tempo, os pontos da coluna no qual estaremos calculando a resposta do sistema não serão igualmente espaçados, o que obviamente introduz uma outra dificuldade, já que as equações do gás e do líquido estão acopladas pelo fluxo de transferência de massa, não podendo ser resolvidas em separado. Para calcular os fluxos de transferência de massa em um determinado ponto, necessitamos das concentrações tanto no gás como na líquido.

•Para contornar o problema anterior, se faz necessário interpolar as concentrações nos pontos desejados. Esta interpolação, é claro, introduz um erro, mas que pode ser feito tão pequeno quanto se deseje através da redução do passo de integração (Δt).

•Por último, devido a interação entre a corrente de gás e de líquido, um método recursivo é única solução para evitar pequenos erros que podem vir a se acumular, como já citamos acima. Assim, para calcular o ponto x[1,2] na fig. 2.5, teria que se calcular simultaneamente os valores de x[1,2]e y[1,2].

A forma como utilizamos este método no presente trabalho é melhor compreendido através do esquema a seguir (fig. 2.6).

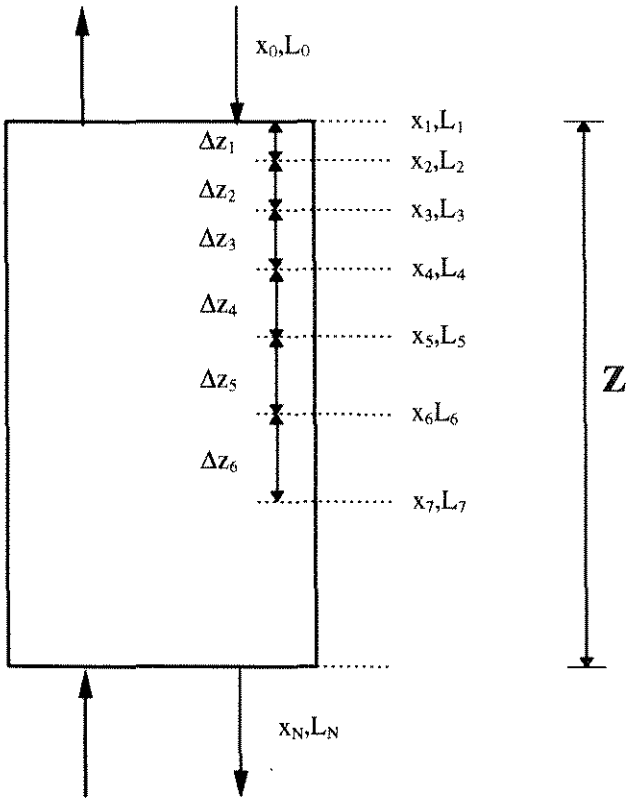

Figura 2.6 - Representação esquemática do método das características, da forma como foi utilizado no presente trabalho.

Na representação acima, considera-se que a quantidade de soluto absorvido pelo líquido não é desprezível. Desta forma, a vazão de líquido irá aumentar ao longo da torre e, consequentemente, a sua velocidade. Logo, as características não podem ser determinadas *a priori*, como foi feito no exemplo anterior. O cálculo dos Δz's é feito a cada intervalo de integração através das equações (2.37a,b), por tentativa e erro, da seguinte forma:

(1) Estima-se todas as concentrações e vazões ao longo da torre no instante seguinte (considerando-se, por exemplo, iguais ao instante anterior);

(2) Calcula-se então, uma primeira aproximação para os Δz's;

(3) A partir destes Δz's, calcula-se os fluxos de transferência de massa nestes pontos (interpolando-se as concentrações e vazões nos pontos onde o valor exato destas não são conhecidos);

(4) A partir destes fluxos, calcula-se um novo valor das concentrações e vazões ao longo da torre;

(5) Verifica se há diferença entre o valor calculado e o anterior. Se houver, calcula-se novos Δz's e repete-se o passo (3) em diante.

Embora o método da forma como descrito acima seja iterativo, devido a simplicidade dos cálculos envolvidos e a rápida convergência deste método, o esforço computacional é mínimo. Além disto, os problemas já citados anteriormente são contornados, tendo-se a cada tempo um perfil de concentrações ao longo da torre.

CAPÍTULO 3

Transferência de Massa com Reação Química

TRANSFERÊNCIA DE MASSA COM REAÇÃO QUÍMICA

Este capítulo visa dar uma explanação sobre alguns do tópicos deste vasto assunto que é a transferência de massa com reação química. Este tema tem recebido diferentes formas de abordagem na literatura. Enquanto alguns trabalhos tratam exclusivamente de obter dados empíricos para determinados sistemas, outros lidam exclusivamente com a formulação matemática de certos problemas, procurando obter soluções analíticas para estes.

Do ponto de vista da simulação, estaremos interessados em conseguir formular o problema da transferência de massa com reação química de uma maneira rigorosa, mas que seja facilmente tratável em termos numéricos. Todos os tópicos tratados neste capítulo visam analisar, a partir dos trabalhos disponíveis na literatura, qual a abordagem que pode ser utilizada de forma a se atingir este objetivo

Ao final, apresentamos a metodologia aqui utilizada para tratar a transferência de massa com reação química.

3.1 - Aplicações da transferência de massa com reação química.

A absorção com solventes químicos tem sido largamente utilizada em indústrias químicas na remoção de gases ácidos (CO_2, H_2S, SO_2), compostos organo-sulfurosos e outras impurezas. Este tipo de processo apresenta como vantagem, além das altas taxas de absorção que podem ser conseguidas, uma grande seletividade.

As impurezas mais constantemente encontradas são CO_2 e H_2S, as quais ocorrem em altas concentrações (5-50%) em processos como produção de amônia, hidrogênio e purificação de gás natural. Outros contaminantes ocorrem em proporções mais baixas.

As concentrações de gases ácidos no gás tratado varia muito de processo para processo. Por exemplo, a concentração de CO_2 no tratamento do gás natural deve estar em torno de 1%, enquanto que no processo de produção de amônia deve ser de 10ppm.

Alguns dos solventes químicos mais utilizados no tratamento de gases ácidos podem ser encontrados na tabela 3.1 a seguir (Astarita *et al*, 1983).

Tabela 3.1 - Solventes químicos mais comumente utilizados.

Mono-etanolamina (MEA)	$NH_2(CH_2CH_2OH)$
Di-etanolamina (DEA)	$NH(CH_2CH_2OH)_2$
Di-isopropanolamina (DIPA)	$NH(CH_2CHOHCH_3)_2$
β,β' Hidroxi-amino-etil-eter	$NH_2(CH_2CH_20CH_2CH_2OH)$
Carbonato de Potássio	K_2CO_3
Glicinato de Potássio	$NH_2(CH_2COOK)$
Soda Cáustica	NaOH

Astarita *et al* (1983) aponta que existiam cerca de 2000 instalações que utilizavam solventes químicos contra menos de 100 que utilizavam solventes físicos até o momento em que aquele trabalho foi publicado. No entanto, estes autores citam que no futuro, a medida que os custos de energia forem aumentando, deverá haver um aumento das unidades operando com solventes físicos, devido ao menor consumo energético destas unidades.

Deve-se observar que boa parte dos solventes mencionados na tabela 3.1 são amino-álcoois. De fato, o número de processos utilizando este tipo de solvente corresponde a quase metade do total de unidades que utilizam solventes químicos. Outro solvente bastante utilizado é o carbonato de potássio com promotores (Astarita *et al* (1983)).

3.2 - Teorias de Transferência de Massa.

Um estudo mais detalhado da absorção seguida de reação química passa necessariamente pelo estudo da transferência de massa em si.

O fenômeno de transferência de massa com reação química entre duas fases imiscíveis pode ser descrito pelos seguintes passos (Astarita, 1967):

(*i*) Difusão de um ou mais reagentes do seio de uma das fases para a interface, onde se pode assumir equilíbrio físico (descrito, por exemplo pela lei de Henry);
(*ii*) Difusão dos reagentes para o seio da outra fase;
(*iii*) Reação química nesta fase;
(*iv*) Difusão dos prdutos gerados na reação química para o seio da fase.

As três últimas etapas podem ocorrer simultaneamente e cada uma delas interfere mutuamente sobre as outras. Embora as duas fases possam ser quaisquer, restrigiremos a nossa discussão a sistemas gás-liquido por ser o sistema de interesse quando se trata de torres de absorção.

A partir do momento em que um elemento de fluido está na interface, a variação da sua concentração do componente i com o tempo e espaço varia de acordo com a seguinte equação genérica:

$$D_i \nabla_2 c_i = u \nabla c_i + \frac{\partial c_i}{\partial t} + r \qquad (3.1)$$

onde "∇" é o operador de Lagrange, "**u**" é o vetor velocidade, e "r" a taxa de reação.

É importante ressaltar que a variável "t" na equação acima se refere ao tempo de permanência do elemento de fluido na interface. O vetor de velocidade refere-se ao movimento do fluido dentro do elemento, de forma que **u**=0 não significa que o elemento está parado, mas sim que este age como um corpo rígido.

A forma como este elemento de fluido chega a superfície e como ele se comporta a partir daí ainda não é conhecida, apesar do grande número de trabalhos que tem surgido na literatura nos últimos tempos tentando explicar a hidrodinâmica da interface gás-líquido no fenômeno de transferência de massa. No entanto, é importante ressaltar que, embora não se saiba exatamente o que ocorre a nível de fenômeno, algumas das teorias existentes dão bons resultados tanto a nível qualitativo quanto quantitativo.

Dentre as teorias existentes, as mais conhecidas são: teoria do filme, teoria da penetração de Higbie e a teoria da superfície renovável. Discutiremos estas teorias de uma forma mais detalhada a seguir.

3.2.1 - Teoria do Filme.

Esta foi a primeira teoria a ser utilizada para explicar a hidrodinâmica da transferência de massa, tendo sido desenvolvida por Lewis e Whitman (1924). O desenvolvimento desta assume a existência de filme delgado junto a interface gás-líquido. Através deste filme a transferência de massa se dá unicamente por difusão molecular. Fora deste filme, no seio do fluido, a composição é mantida constante devido à turbulência.

A espessura do filme de transferência de massa é muito pequena, e portanto, possui uma capacitância em termos de massa desprezível. Desta forma, o perfil de concentração assume rapidamente o estado estacionário, e o termo transiente pode ser eliminado na equação (3.1). O filme delgado também permite tratar a geometria como plana, já que o raio de curvatura de uma superfície líquida qualquer em geral é bem maior que a espessura do filme. Por fim, como o fluido através do qual ocorre a transferência de massa está estagnado, temos que **u**=0. Estas considerações permitem escrever a equação (3.1) na seguinte forma (para um componente absorvido "A"):

$$D_A \frac{d^2 C_A}{dx^2} = r \tag{3.2}$$

sujeita as seguintes condições de contorno:

$$\left.\begin{aligned} C_A &= C_{Ai}, \quad x = 0 \\ C_A &= C_{Ab}, \quad x = y_L \end{aligned}\right\} \tag{3.3}$$

onde y_L é o comprimento do filme e os subíndices "i" e "b" significam, respectivamente, interface e seio (bulk).

Integrando-se a equação (3.2) sem o termo de reação química (ou seja, para a absorção puramente física) com as condições de contorno (3.3), obtemos a seguinte expressão para o fluxo de transferência de massa:

$$N_A = \frac{D}{y_L}(C_{Ab} - C_{Ai}) \tag{3.4}$$

Como a espessura do filme de transferência de massa é um parâmetro difícil de se medir experimentalmente, costuma-se englobar o termo D/y_L em um só parâmetro, o coeficiente de transferência de massa "k":

$$k = \frac{D}{y_L} \tag{3.5}$$

Logo, temos que pela teoria do filme, o coeficiente de transferência de massa é proporcional a difusividade do componente absorvido. No entanto, medidas experimentais mostram que na verdade este é proporcional a difusividade elevada a uma potência que varia entre 0,5 e 0,75, a depender de características do fluido e da hidrodinâmica do equipamento (Glasscock e Rochelle, (1989)).

Conclui-se que a teoria do filme está em desacordo com evidências experimentais. No entanto, é importante frisar que esta vem sendo utilizado com bons resultados em certos problemas de engenharia química.

3.2.2 - Teoria da Penetração de Higbie.

A teoria da penetração (Higbie (1935)) propõe um modelo hidrodinâmico mais real que o da teoria do filme. Esta abandona a hipótese de um filme muito delgado junto a interface e propõe que diversos elementos de espessura infinita estejam na interface e que haja um renovação constante destes. Desta forma, um elemento de fluido sai do seio da solução para a interface, onde troca massa durante um determinado tempo, voltando depois para o seio da solução e sendo substituído por outro. Higbie assumiu que o tempo de contato era o mesmo para todos os elementos.

Cada elemento que chega a interface estará à concentração inicial do seio e trocará massa em regime transiente, já que a espessura do elemento aqui é infinita. As considerações de geometria plana e de que o elemento se comporta como um corpo rígido são mantidas, de forma que a equação (3.1) reduz-se a:

$$D_A \frac{\partial^2 C_A}{\partial x^2} = \frac{\partial C_A}{\partial t} + r \tag{3.6}$$

sujeita as seguintes condições de contorno:

$$\left.\begin{array}{ll} C_A = C_{A0}, & t = 0 \\ C_A = C'_{Ai}, & x = 0 \\ C_A \text{ finito}, & x \to \infty \end{array}\right\} \tag{3.7}$$

É importante observar que, como tempo na equação acima se refere ao tempo em que a partícula permanece trocando massa na interface, o fluxo total que cada partícula troca é igual ao somatório dos fluxos ao longo do tempo. Assim, sendo N_A o fluxo de transferência de massa em um dado tempo e t^* o tempo de residência do elemento na interface, ou seja, o tempo em que ele permanece traocando massa, o fluxo médio trocado por esse elemento é:

$$\overline{N}_A = \frac{1}{t^*}\int_0^{t^*} N_A dt \tag{3.8}$$

O coeficiente de transferência de massa para o caso de absorção física que se obtém a partir desta teoria é dado por:

$$k = 2\sqrt{\frac{D}{\pi t^*}} \tag{3.9}$$

Esta teoria está em concordância com os alguns dados experimentais, já que prevê um coeficiente de transferência de massa proporcional à $D^{0,5}$. No entanto, como já foi citado acima, os coeficientes experimentais nem sempre são proporcionais a raiz quadrada da difusividade, de forma que esta teoria também não está em concordância com todos os dados experimentais.

3.2.3 - Teoria da Superfície Renovável.

Também conhecida como teoria da penentração de Danckwerts, esta é uma variação da teoria de Higbie, sendo que a hipótese de igual tempo de vida para todos os elementos é abandonada. Danckwerts (1951) propôs que o tempo de contato dos elementos seguisse uma função de distribuição tempo, $\psi(t)$. Assim, o fluxo médio que na teoria da penetração é calculado pela equação (3.7), seria calculado por:

$$\overline{N}_A = \int_0^\infty N_A \psi(t) dt \tag{3.10}$$

A forma proposta por Danckwerts para a função de distribuição é baseada na hipótese de que a probalidade de um elemento sair da interface é independente da sua idade. Desta forma, a taxa de saida de elementos da interface é proporcional ao número de elementos daquela idade que estão presentes. Logo:

$$-\frac{d\psi}{dt} = s\psi \tag{3.11}$$

onde "s" é uma constante de proporcionalidade e 1/s pode ser visto como um tempo médio de vida dos elementos na interface.

Independente do valor de "s", esta teoria também prediz que o coeficiente de transferência de massa é proporcional à difusividade elevada a uma potência de 0.5.

3.2.4 - Outras teorias de transferência de massa.

Embora as teorias acima citadas sejam as mais comumente utilizadas, existem outras teorias na literatura que procuram adicionar efeitos não englobados por estas. Um exemplo disto é a teoria da difusividade eddy, proposta por King (1966). As teorias anteriores consideram que a transferência de massa se dá por difusão através de uma camada laminar, desprezando-se, portanto, o termo de velocidade convectiva na equação (3.1). A teoria da difusividade eddy modifica o coeficiente de difusão para levar em consideração tanto a transferência de massa por regime turbulento como por difusão molecular. Em compensação esta teoria adiciona novos parâmetros cujos valores devem ser conhecidos para o sistema em questão.

Esta teoria prevê um coeficiente de transferência de massa proporcional a difusividade elevada a uma potência que pode variar, a depender dos parâmetros utilizados no modelo. Assim, este modelo, embora necessite de um volume de informações ainda não disponíveis para que possa ser utilizado, está em melhor concordância com evidências experimentais.

3.3 - Resolução analítica das equações.

Boa parte dos trabalhos que se encontram na literatura sobre transferência de massa com reação química se restringe a encontrar soluções analíticas para as equações resultantes das teorias acimas. Não obstante o esforço de muitos autores, estas soluções só podem ser obtidas em casos específicos de cinética química e de ordem de reação.

A classificação dos diversos sistemas é comumente baseada nos chamados regimes, os quais dependem de fatores como a ordem de reação, parâmetros de transferência de massa e cinética da reação química envolvida. Um parâmetro importante que engloba todos estes fatores é o chamado número de Hatta (De Leye e Froment, 1986a,b), que permite dividir os regimes em:

(1) Lentos (cuja reação se dá praticamente toda no seio da solução);

(2) Moderadamente rápidos (cuja reação acontece tanto no filme como no seio da solução);

(3) Rápidos (onde todo componente absorvido reage no filme);

(4) Instantâneos (onde existe um ponto onde a concentração de componente absorvido e reagentes cai a zero).

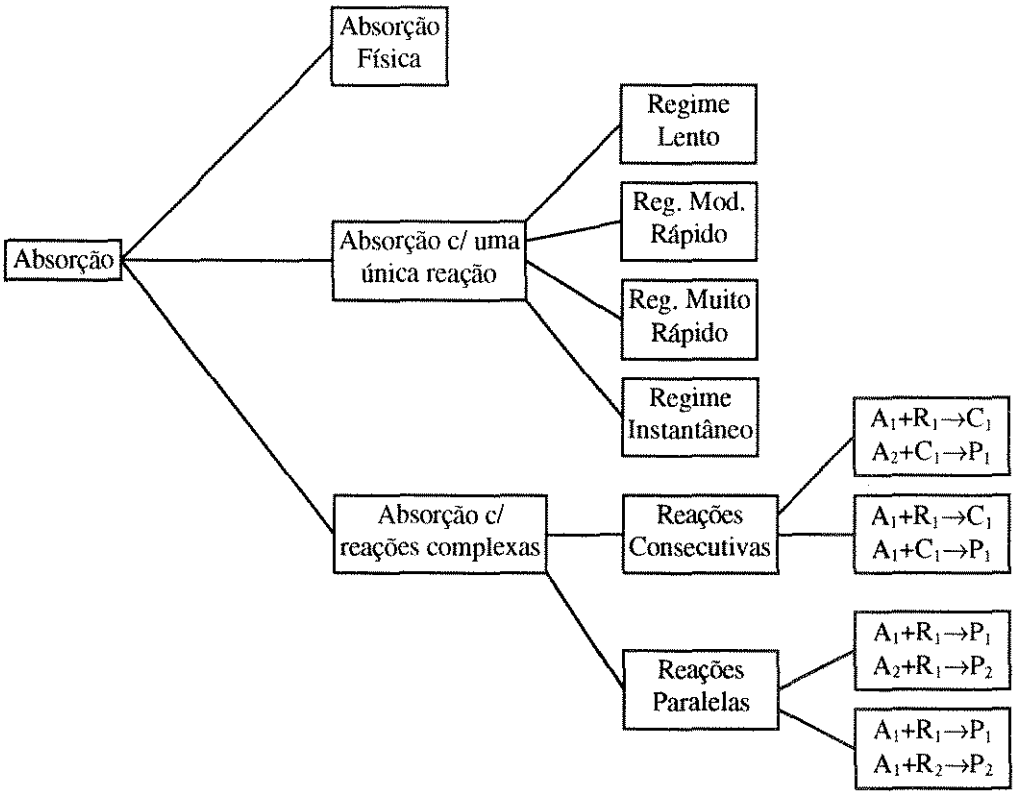

Figura 3.1 - Diversos tipos de absorção e seus regimes cinéticos.

A fig. 3.1 mostra alguns tipos de reações e regimes cinéticos. No caso mais geral, onde a reação acontece tanto no filme como no seio da solução não há uma solução analítica geral, cabendo algumas simplificações apenas nas condições de contorno. Alguns autores, no entanto, fazem simplificações mais drásticas e obtêm soluções aproximadas para alguns destes casos. O erro destas soluções cresce a medida que as condições se afastam das hipóteses simplificadoras. Um resumo destas soluções aproximadas e os casos para os quais se aplicam podem ser encontradas nos trabalhos de De Leye e Froment (1986) e de Charpentier (1982).

Embora as equações analíticas não estejam disponíveis para muitos casos, elas permitem, a partir do resultados existentes, uma análise prévia do tipo de equipamento que deve ser utilizado. De uma forma geral, a taxa de absorção pode depender dos seguintes fatores:

(i) Área interfacial;
(ii) Acúmulo de líquido;
(iii) k_L^0 (constante de transferência de massa sem reação química);
(iv) Taxa de reação;
(v) Concentração na interface.

Astarita (1967) mostra quais dos fatores acima são importantes para cada um dos diferentes tipos de regimes cinéticos.

É importante ressaltar que poucas expressões permitem um cálculo direto do fluxo de transferência de massa a partir de grandezas conhecidas. Muitas expressões apresentam a concentração na interface como uma das grandezas necessárias ao cálculo do fluxo. Desta forma, se faz necessário considerar também a resistência da fase gás e aplicar condições de igualdade de fluxo e condição de equilíbrio na interface (geralmente a Lei de Henry) para se ter informações suficientes para se efetuar os cálculos. Logo, a existência de uma solução analítica não implica em um cálculo direto do fluxo de transferência de massa, sendo muitas vezes necessário o uso de um método numérico para resolver o sistema implícito de equações que surge.

Em alguns casos a solução obtida é bastante complexa, como por exemplo no trabalho de Wang e Langemann (1994), que trata exclusivamente de resolver as equações resultantes da teoria do filme para o caso de uma reação de primeira ordem e obtém como solução uma complicada série infinita.

Além disto, algumas destas soluções não permitem a inclusão de alguns fenômenos, como por exemplo, a correção de solubilidade devido à presença de íons.

3.4 - Resolução numérica das equações.

A resolução numérica das equações resultantes dos modelos hidrodinâmicos vem recebendo uma crescente atenção na literatura. A razão disto está tanto nas lacunas e limitações deixadas pelas soluções analíticas, como no avanço dos computadores digitais, que vem permitindo se fazer um volume de cálculo cada vez maior. Além disto, as resolução numéricas das equações permitem a inclusão de efeitos que as soluções analíticas não consideram.

Quanto ao aspecto de esforço computacional, De Leye e Froment (1986 e 1986a) compararam as duas formas de solução e obtiveram que, para o mesmo caso, o tempo de computação da solução numérica foi cerca de 7 vezes maior que o da solução analítica, enquanto que os resultados obtidos foram muito próximos.

Vale ressaltar que o método numérico utilizado por estes autores foi o das diferenças finitas, que resulta num sistema de equações muito grande (principalmente no caso da absorção seguida de reação química, onde já vimos que os perfis de concentração costumam ter variações muito bruscas). Um sistema de equações menor resulta se utilizarmos o método da colocação ortogonal em elementos finitos. Este método já foi utilizado com bastante sucesso (tanto em termos de precisão dos resultados como em tempo de computação) em um problema bastante semelhante que é a determinação do perfil de concentração em um poro de catalisador por Carey e Finlayson (1979).

Glasscock e Rochelle (1989) também utilizaram este método para comparar algumas teorias de transferência de massa. Estes encontraram que as teorias dinâmicas (teorias da penetração) apresentam um tempo de computação bem maior em relação às teorias estáticas. O motivo para isto está na taxa de reação, que torna o sistema de equações diferenciais no tempo muito *stiff*, dificultando bastante a sua resolução. Estes autores sugerem que, devido ao menor tempo computacional envolvida nas teorias estáticas e dada as atuais incertezas sobre o mecanismo real de transferência de massa, as teorias estacionárias são de maior utilidade na interpretação de dados experimentais e no projeto de equipamentos de tratamento de gases.

Uma outra vantagem das soluções numéricas é que permitem o cálculo das concentrações e fluxos independente da expressão da taxa e regime cinético que se esteja trabalhando, o que permite manusear problemas extremamente complexos como por exemplo, reações auto-catalíticas.

Embora a resolução numérica das equações de transferência de massa permita o cálculo dos fluxos qualquer que seja o regime cinético, este não permite a generalização quando mais de uma reação está presente. De fato, o balanço de massa obtido pela Lei de Fick para duas reações consecutivas, por exemplo, não se aplica ao caso de duas reações em paralelo. Nestes casos, as soluções tem que ser desenvolvidas especificamente para uma determinada sequência de reações (embora seja genérica quanto ao regime cinético em que estas ocorrem). Assim, para cada uma das reações complexas mostradas na fig. 3.1, um balanço específico teria que ser feito, só cabendo uma generalização quanto ao regime cinético destas.

3.5 - Dados físico-químicos.

Como já citamos anteriormente, um dos problemas quando se trabalha com simulação de um processo que envolva absorção com reação química é a determinação de dados físico-químicos necessários aos cálculos.

Esses dados, muitas vezes, não são encontrados sob a forma de correlações que sirvam para diversos sistemas, mas principalmente sob a forma de dados experimentais de um dado sistema. Desta forma, a depender do sistema que se trabalhe, as propriedades físico-químicas podem ser de difícil determinação. Para gases mais

utilizados como CO_2, essas propriedades já são mais conhecidas. Trabalhos como o de Danckwerts (1966) e Charpentier (1982) servem como base para uma série de referências para algumas das substâncias mais comumente utilizadas em absorção com reação química. Para substâncias em geral, a compilação de Reid *et al* (1987) serve como base. A seguir discutiremos algumas destas grandezas.

3.5.1 - Solubilidade de gases em líquidos.

Equilíbrio interfacial é normalmente assumido na interface gás-líquido. Este equilíbrio é normalmente considerado sob a forma da lei de Henry, que para o caso da absorção física de gases com uma concentração de componente absorvido baixa, tem-se mostrado ser válida. No caso da absorção com reação química, esta lei não se aplica ao caso da concentração total, mais continua a ser válida para a concentração de gás não reagido (na interface).

Dizer que a lei de Henry pode ser aplicada na interface equivale a dizer que esta não oferece nenhuma resistência a transferência de massa. Quando se tem contaminantes na interface, no entanto, esta lei passa a não se aplicar. Asolekar *et al* (1985) mostra também que esta falha no caso de sistemas com taxas de absorção fortemente aumentada pela reação química. No entanto, não há uma forma de se predizer o comportamento real nestes casos.

A solubilidade de gases em líquidos pode variar bastante de caso para caso. Apesar das tentativas em se obter expressões que consigam prever a solubilidade de gases em líquidos, estas têm tido pouco sucesso (Reid *et al* (1987)). Isto acontece em parte por não existir uma teoria que descreva bem as soluções gás-líquido e também devido aos poucos dados experimentais existentes na literatura (principalmente a temperaturas muito diferentes de 25°C). Logo, dados experimentais de solubilidade são essenciais para muitos sistemas.

A redução de solubilidade devido à presença de eletrólitos pode ser estimada pelo método de van Krevelen and Hoftijzer com um erro em torno de 10% (Danckwerts *et al* (1966)), utilizando-se as seguintes equações:

$$\log_{10}(H / H_w) = -K_s I \qquad (3.12)$$

$$K_s = i_+ + i_- + i_G \qquad (3.12a)$$

onde:

H = Solubilidade na solução eletrolítica;
H_W = Solubilidade em água;
I = Força iônica da solução;
i_+, i. e i_G = Contribuições dos cátions, ânions e gases, respectivamente.

Uma tabela com valores de contribuições para diferentes íons pode ser encontrada no trabalho de Danckwerts *et al* (1966). Estes autores sugerem ainda que esta expressão possa ser generalizada para soluções com vários eletrólitos da seguinte forma:

$$\log_{10}(H / H_w) = -(K_s' I' + K_s'' I'' + \ldots) \qquad (3.13)$$

onde Ks', Ks'', etc, se refere as contribuições dos diferentes eletrólitos e I', I'', etc, se refere às forças iônicas destes eletrólitos.

3.5.2 - Difusividades.

Ao contrário do acontece com sistemas gasosos, ainda não existe uma metodologia satisfatória para determinar a difusividade em sistemas líquidos. Assim, a determinação da difusividade em diferentes casos exige diversas abordagens. Em certos casos, no entanto, a abordagem existente pode não ser satisfatória ou exigir parâmetros não disponíveis para uma dada substância. Além disto, a maioria dos trabalhos se preocupa em obter a difusividade em diluição infinita.

Na ausência de dados experimentais, para soluções não-eletrolíticas, Charpentier (1982) sugere o uso da equação de Wilke e Chang. Valores destas difusividades para a presença de outras substâncias em solução podem ser encontrados para alguns casos específicos, como por exemplo, para o CO_2 (Danckwerts (1966)).

Uma outra forma de se corrigir a difusividade é através da conhecida relação de Stoke-Einstein, que diz que para soluções aquosas, a difusividade e viscosidade da solução são inversamente proporcionais uma a outra. Assim, conhecendo-se a difusividade de uma substância a uma determinada viscosidade, pode-se determinar a difusividade desta em uma solução com uma viscosidade diferente. Esta regra é particularmente útil para corrigir a difusividade em soluções de aminas, cuja viscosidade é conhecida em função da concentração. Alguns autores no entanto, têm mostrado que de uma forma geral, $D^{\circ}_{AB} \propto \mu_B^q$, onde o valor de "q" varia de sistema para sistema.

Em soluções salinas, temos íons se difudindo ao invés de moléculas. Na ausência de potencial elétrico, a difusividade de um único sal pode ser tratada como difusão molecular (Reid *et al* (1987)). Estes autores advertem que a baixas concentrações, o comportamento das soluções salinas é bem conhecido sendo que, para soluções com contaminantes e concentrações normalmente encontradas industrialmente, faz-se necessário correções, havendo uma perda de precisão e generalidade.

Uma outra abordagem que pode ser utilizada quando se trata de difusão de íons é considerar a equação de Nernst-Einstein para relacionar as condutâncias elétricas aos coeficientes de difusão. Glasscock e Rochelle (1989) utilizaram esta abordagem e encontraram que o tratamento numérico das equações resultantes se torna mais difícil quando o efeito do potencial elétrico através da equação acima mencionada é incluída na equação da lei de Fick.

Como até o presente momento não há uma metodologia que permita obter valores confiáveis da difusão de íons, preferiu-se considerar neste trabalho que a difusão de íons obedece a lei de Fick, como sugerido por Reid *et al* (1987) e calcular as difusividades pelos métodos sugeridos por estes autores. Uma abordagem mais completa sobre a difusão de íons pode ser encontrada no trabalho de Krishna (1987).

3.6 - Absorção de CO_2 em solução de Monoetanolamina (MEA).

Como mencionamos acima, para o caso onde haja a presença de mais de uma reação, a resolução das equações é específica para cada uma das sequências utilizadas. Desta forma, não há meio de se utilizar um modelo generalizado que sirva para todos os sistemas. Neste trabalho, estudaremos os casos onde apenas uma única reação ocorre (que cobre grande parte dos processo químicos que utilizam absorção seguida de reação química). A modelagem aqui utilizada é válida para qualquer sistema onde haja apenas uma reação química (independente do fato desta ser reversível ou irreversível e do regime cinético). Como caso de estudo, será utilizado o sistema CO_2 - MEA.

A forma como o CO_2 reage em soluções aquosas de monoetanolamina é bastante complexa e ainda não é totalmente compreendida. Sabe-se que três reações principais devem ser levadas em consideração:

$$CO_2 + 2RNH_2 \rightarrow RNH_3^+ + RNCOO^- \quad (3.14)$$

$$CO_2 + RNH_2 + H_2O \rightarrow RNH_3^+ + HCO_3^- \quad (3.15)$$

$$RNCOO^- + CO_2 + 2H_2O \rightarrow RNH_2^+ + 2HCO_3^- \quad (3.16)$$

A reação (3.14), reação de formação de carbamato, tem sido bastante estudada para a MEA, sendo que a taxa de reação desta é dada por:

$$r = k[CO_2][RNH_2] \quad (3.17)$$

A reação (3.15), reação de formação de bicarbonato, ocorre via mecanismo direto. Desta forma, esta é de primeira ordem em relação ao reagente, ou seja:

$$r \propto [CO_2][RNH_2] \quad (3.18)$$

A reação (3.16), reação de reversão de carbamato, ocorre com um taxa semelhante a da equação (3.18).

Para as condições de interesse industrial, pode-se considerar que a única reação que ocorre na absorção de CO_2 em solução de MEA é a dada pela expressão (3.14). Desta forma, a taxa de reação pode ser calculada pela equação (3.17).

3.7 - Cálculo dos fluxos de transferência de massa utilizando a teoria do filme e o método da colocação ortogonal em elementos finitos.

Para o cálculo do fluxo de transferência de massa utilizaremos, conforme já discutimos no capítulo anterior, o modelo do filme, por ser muito mais fácil de tratar numericamente.

A nossa discussão se restringe a um único componente absorvido e uma única reação química. A absorção de vários componentes ou a existência de várias reações leva a equacionamentos diferentes, não havendo uma única equação que possa descrever as diversas possibilidades de reações complexas (reações em paralelo, reações em série, etc.).

Portanto, as equações na fase líquida para componente absorvido, reagentes e produtos são, respectivamente:

$$D_A \frac{d^2C_A}{dx^2} = a_A r \quad (3.19a)$$

$$D_{R_k}\frac{d^2C_{R_k}}{dx^2} = a_{R_k} r \tag{3.19b}$$

$$D_{P_k}\frac{d^2C_{P_k}}{dx^2} = a_{P_k} r \tag{3.19c}$$

sujeitas as seguintes condições de contorno:

$$\left.\begin{aligned} C_A &= C_{Ai}, \quad x = 0 \\ C_A &= C_{Ab}, \quad x = y_L \end{aligned}\right\} \tag{3.20a}$$

$$\left.\begin{aligned} \frac{dC_{R_k}}{dx} &= 0, \quad x = 0 \\ C_{R_k} &= C_{R_kb}, \quad x = y_L \end{aligned}\right\} \tag{3.20b}$$

$$\left.\begin{aligned} \frac{dC_{P_k}}{dx} &= 0, \quad x = 0 \\ C_{P_k} &= C_{P_kb}, \quad x = y_L \end{aligned}\right\} \tag{3.20c}$$

onde y_L é o comprimento do filme líquido.

As condições de contorno na interface para os reagentes e produtos significam que estes não se difundem através da interface.

Para a fase gás, devido a ausência de reação química, a expressão da lei de Fick pode ser integrada diretamente, sendo o fluxo de gás dado por:

$$N_A\big|_{y=y_G} = k_G(P_{Ab} - P_{Ai}) \tag{3.21}$$

Como as condições na interface não são conhecidas, se faz necessário outras relações para que se possa resolver o conjunto de equações. Estas relações são: a igualdade de fluxos na interface (não há acúmulo de "A" na interface) e o equilíbrio termodinâmico na mesma. Assumindo então que o equilíbrio termodinâmico possa ser expresso pela lei de Henry na interface (conforme já discutimos), temos:

$$P_{Ai} = HC_{Ai} \tag{3.22}$$

As soluções das equações (3.19b) e (3.19c) podem ser obtidas analiticamente se tivermos a solução da equação (3.19a). Para resolver a equação (3.19b), por exemplo, basta multiplicá-la por (a_A/a_{Rk}) e subtrair da equação (3.19a). Desta forma, elimina-se a taxa de reação e a equação pode ser integrada facilmente utilizando as condições de contorno (3.20a) e (3.20b), sendo que na interface, para o componente "A", utiliza-se a condição (3.25). De forma análoga, procedemos com a equação (3.19c), obtendo então:

$$C_{R_k} = C_{R_kb} + \frac{a_{R_k}}{a_A}\frac{D_A}{D_{R_k}}(C_A - C_{Ab}) - \frac{a_{R_k}}{a_A}\frac{N_A\big|_{y=0}}{D_{R_k}}(y_L - y) \tag{3.23}$$

$$C_{P_k} = C_{P_kb} + \frac{a_{P_k}}{a_A}\frac{D_A}{D_{P_k}}(C_A - C_{Ab}) + \frac{a_{P_k}}{a_A}\frac{N_A\big|_{y=0}}{D_{P_k}}(y_L - y) \tag{3.24}$$

Logo, uma equação diferencial de segunda ordem ainda tem de ser resolvida. Isto foi feito utilizando colocação ortogonal em elementos finitos e resolvendo-se o sistema de equações resultantes por Newton-Raphson. Os detalhes da aplicação do método e do cálculo do jacobiano analítico do sistema de equações não-lineares resultante encontram-se no Apêndice A.

Embora as soluções analíticas das equações (3.19b) e (3.19c) simplifiquem bastante o cálculo, a resolução da equação (3.19a) ainda requer a um esforço computacional razoável. Isto se deve principalmente ao fato das reações que

normalmente aparecem nos casos de absorção serem rápidas. Desta forma, a constante da taxa de reação é bastante grande, dificultando a convergência do método numérico. Pode ocorrer, inclusive, falha na convergência do método numérico. Glasscock e Rochelle (1989) tiveram este problema e propuseram que a taxa de reação fosse multiplicada por um parâmetro que iria variar progressivamente de 0 a 1. Quando este parâmetro fosse zero, a equação seria a mesma do caso da absorção física e a equação seria de fácil resolução. A solução assim obtida seria utilizada como condição inicial do mesmo problema com o parâmetro, desta vez, com um valor um pouco maior. Repetindo-se este procedimento sucessivamente até se utilizar o valor 1 para o parâmetro, chegamos a solução do problema.

Logo, resolvendo-se as equações anteriores, temos os perfis de concentração de todos os componentes. A partir destes perfis, os fluxos de transferência de massa na fase líquida podem ser calculados por:

$$N_A\Big|_{y=0} = -D_A \frac{\partial C_A}{\partial y}\Big|_{y=0} \tag{3.25}$$

$$N_A\Big|_{y=y_L} = -D_A \frac{\partial C_A}{\partial y}\Big|_{y=y_L} \tag{3.26}$$

$$N_{R_k}\Big|_{y=y_L} = -D_{R_k} \frac{\partial C_{R_k}}{\partial y}\Big|_{y=y_L} \tag{3.27}$$

$$N_{P_k}\Big|_{y=y_L} = -D_{P_k} \frac{\partial C_{P_k}}{\partial y}\Big|_{y=y_L} \tag{3.28}$$

Um resumo dos valores de propriedades físicas utilizadas a 315K é dado na tabela (3.2). Os valores contidos nesta tabela foram diretamente retirados dos trabalhos citados, quando experimentais, ou diretamente calculados pelas correlações disponíveis neste, com exceção da difusividade do CO_2 na solução de MEA, que foi obtido para a água e corrigida pela lei de Stokes-Einstein (Charpentier, 1982).

Tabela 3.2 - Propriedades físicas utilizadas na resolução da teoria do filme.

Propriedade	Valor	Referência bibliográfica
Constante da taxa de reação (k)	$5{,}183 \times 10^7 m^3/kmol\ h$	Hikita et al (1977)
Difusividade do CO_2 no líquido (D_A)	$5{,}184 \times 10^{-6} m^2/h$	Thomas e Furzer (1962)
Difusividade da MEA em solução (D_R)	$2{,}772 \times 10^{-6} m^2/h$	Thomas e Furzer (1962)
Constante de Henry (H)	$48{,}6 m^3 bar/kmol$	Danckwerts e Sharma (1966)
Contribuição do cátion, ânion e gás na eq. (3.13)	$i_+=0{,}31$ $i_-=0{,}21$ $i_g=-0{,}17$	Danckwerts e Sharma (1966)
Difusividade do CO_2 na fase gás (D_{AG})	$9{,}62894 \times 10^{-3} m^2/h$	Reid et al (1987)

Nos gráficos (3.1), (3.2) e (3.3) mostramos alguns dos perfis obtidos com a resolução do modelo do filme. Estes gráficos foram tirados de três pontos (base, meio e topo) de uma coluna de absorção que será utilizada como caso de estudo no capítulo 4.

Os pontos de mais difícil convergência do algoritmo utilizado são os próximos à base da coluna, onde o fluxo de transferência de massa é maior

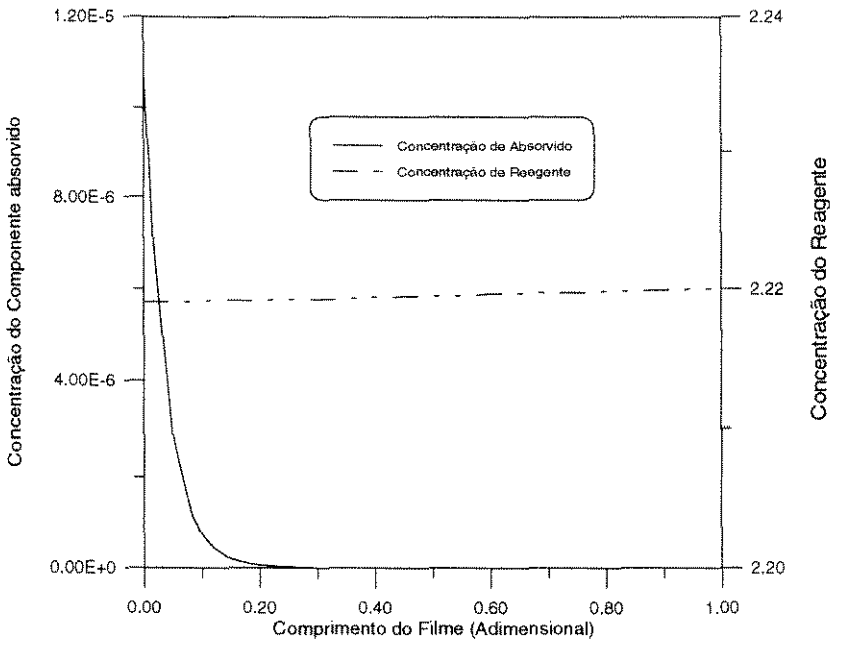

Gráfico 3.1 - Perfis do conponente absorvido e do reagente ao longo do filme (topo da coluna).

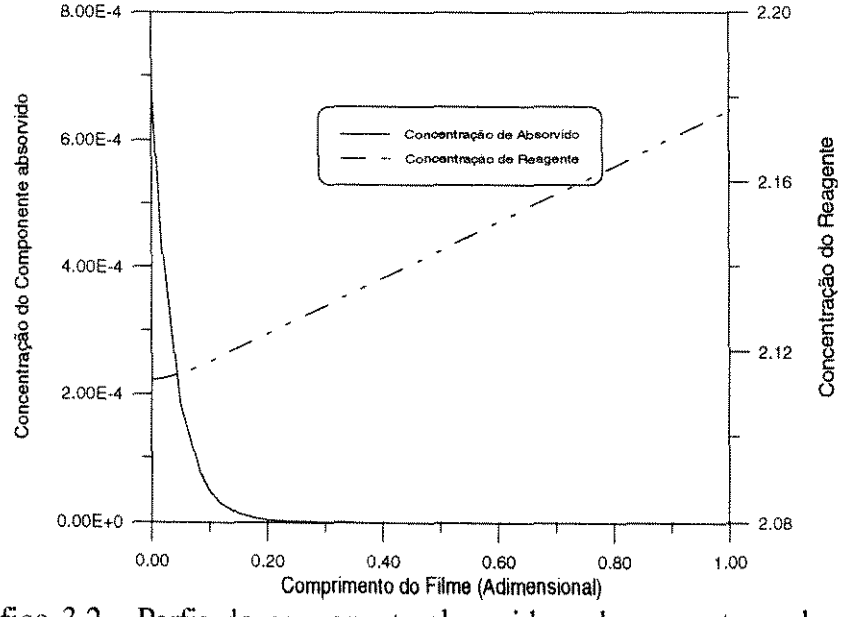

Gráfico 3.2 - Perfis do conponente absorvido e do reagente ao longo do filme (centro da coluna).

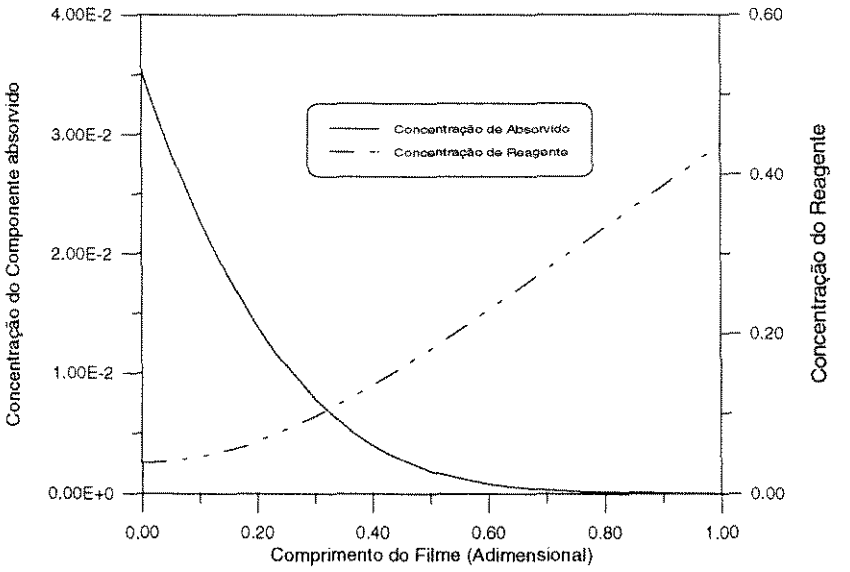

Gráfico 3.3 - Perfis do conponente absorvido e do reagente ao longo do filme (Base da coluna).

CAPÍTULO 4

Modelagem e Simulação de Colunas Recheadas

MODELAGEM E SIMULAÇÃO DE COLUNAS RECHEADAS

Neste capítulo serão discutidas as abordagens que têm sido utilizadas na simulação transiente de colunas de absorção recheada. Numa etapa seguinte, o cálculo da transferência de massa com reação química e as metodologias numéricas discutidas anteriormente serão utilizadas juntamente com o modelo hidrodinâmico da coluna, de forma a se ter uma simulação que consiga representar as principais características do processo.

Ao final do capítulo, serão apresentados os resultados de algumas simulações que mostram a importância de se incluir alguns efeitos que são comumente desprezados quando se estuda absorção em colunas recheadas, bem como características importantes dos processos que deverão ser levadas em consideração num estudo de controle do mesmo.

4.1 - Processos que utilizam a absorção com reação química.

A absorção com reação química é utilizada em diversos processos para a retirada de gases ácidos de correntes. Esta retirada pode ser tanto por motivos ambientais como por exigências do processo. A composição de entrada e saída dos gases pode variar bastante de um processo para outro. Na tabela (4.1) a seguir, mostramos alguns dos processos que utilizam este tipo de tecnologia, bem como as composições de saída exigidas por cada um (Astarita (1983)).

Tabela 4.1 - Processos de tratamento de gases ácidos e respectivas composições de saída.

Processo	Gás ácido a tratar	Composições de saída (% gás)
Produção de hidrogênio	CO_2	<0.1%
Dessulfuração do petróleo	CO_2+H_2S+CO	10ppm H_2S
Produção de amônia	CO_2	<16ppm
Produção de etileno	H_2S, CO_2	~1ppm H_2S, 1ppm CO_2
Dessulfuração de gás	SO_2	90% de remoção

O que se observa da tabela acima é que a composição final de gás ácido pode ir deste alguns ppm's até alguns porcentos. Por outro lado, o fato da absorção seguida de reação química, em geral, aumentar bastante a taxa de transferência de massa, permite que sejam tratados gases com altas concentrações de soluto, ou seja, a concentração pode variar bastante ao longo da coluna.

Como já mencionamos no capítulo 3, grande parte dos processos envolvem absorção de H_2S e/ou CO_2. Dentre os solventes mais utilizados na remoção destes gases estão as álcoolaminas. Em princípio, quanto maior a composição destes solventes na

solução absorvedora, menor serão as dimensões da coluna necessária para se ter uma dada separação. No entanto, a utilização de altas concentrações causa alguns problemas, dentre os quais o mais sério é a severa corrosão que se tem no equipamento, que é o principal limitante.

Desta forma temos que neste tipo de processo, é comum se ter a composição de gás ácido grande na entrada e pequena na saída, bem como colunas com grandes dimensões. Estas características são importantes no desenvolvimento de um simulador. Concluímos então que as considerações de que os fluxos são constantes ao longo da coluna e de que as perturbações introduzidas se propagam imediatamente por toda a sua extensão (simplificações bastante utilizadas na literatura), podem não se aplicar a este caso.

4.2 - Os diferentes tipos de equipamentos.

Uma ampla gama de equipamentos é utilizada para os diferentes casos de absorção em sistemas gás-liquido. Dentre estes podemos citar: Colunas de prato, de recheio, tipo "spray" e de borbulhamento, vasos agitados, venturis, etc. A escolha do equipamento depende da aplicação e de considerações de custo. Se por exemplo estivermos interessados no produto de uma reação rápida que forme uma lama viscosa, um vaso agitado é a escolha mais sensata, sendo que uma coluna de pratos, por exemplo, não se aplicaria a uma situação desta. No caso de tratamento de gases ácidos, os tipos mais comumente utilizados são as colunas de prato e de recheio.

Quando se trabalha com absorção seguida de reação química, o tipo de regime cinético pode tornar mais viável a utilização de um dado tipo de equipamento, como comentamos no capítulo anterior. Na ausência de um critério deste tipo, as considerações a serem feitas se baseiam no custo do equipamento para conseguir uma determinada eficiência. Como exemplo de critérios que devem ser levados em consideração, podemos citar:

- a área interfacial de troca de massa por unidade de volume é maior em um prato que num recheio. No entanto, como numa coluna de pratos o volume total não é preenchido, há espaços vazios entre os pratos. Assim, a área total numa coluna deste tipo depende do espaçamento utilizado. Quando não for necessário usar vertedouros, o espaçamento é pequeno o bastante para que a área interfacial seja maior numa coluna de pratos do que numa coluna de recheio das mesmas dimensões;
- As colunas de recheio apresenta uma pequena perda de carga, o que torna a sua utilização muito interessante em vários tipos de processos;
- Colunas de pratos suportam uma variação maior de carga.

Uma série de outros critérios poderiam ser citados. No entanto, este tipo de análise foge ao escopo deste trabalho. Vale salientar que as colunas de recheio são uma das mais utilizadas atualmente, havendo na literatura, no entanto, uma lacuna no que se refere a uma simulação mais rigorosa destas. Discutimos a seguir os diversos aspectos envolvidos no projeto e simulação de colunas recheadas.

4.3 - Modelos hidrodinâmicos de colunas recheadas.

Apesar do número de trabalhos feitos nesta área, ainda não há um consenso sobre o modelo hidrodinâmico mais adequado para descrever uma coluna recheada. Basicamente três modelos são encontrados na literatura: modelo de células de mistura, regime empistonado (*plug flow*) e modelo de dispersão axial.

O modelo de células de mistura é, na verdade, uma tentativa de "imitar" o procedimento que normalmente é adotado em colunas de pratos, evitando assim um conjunto de equações diferenciais parciais no tempo e no espaço que surge quando se estuda a dinâmica de uma coluna de absorção. Este consiste em dividir o recheio em várias seções e considerar cada uma destas como sendo um estágio de mistura perfeita. Desta forma, passa-se de um meio contínuo para um constituído de uma sucessão de estágios. Este modelo é apenas uma simplificação, não havendo comprovação experimental que o justifique. Além disto, o fato de se considerar mistura perfeita em cada "estágio" adiciona implicitamente uma dispersão axial, numa magnitude que irá variar de acordo com o número de células que se utilizar na simulação. Assim, um grande número de células se faz necessário para diminuir este erro, o que pode resultar num problema de grandes dimensões.

No regime empistonado, considera-se que o transporte convectivo é completamente dominante sobre o difusivo, em particular, na direção de fluxo, ou seja, todas as partículas do fluido movem-se na mesma direção e com a mesma velocidade. Os balanços de massa devem ser feitos em um volume infinitesimal. Assim, as equações que são geradas a partir deste apresentam (no caso de simulação dinâmica) termos no tempo e no espaço, o que torna a sua solução um pouco mais complicada. Um dos equipamentos que mais se aproxima do que este modelo propõe são as torres recheadas.

O modelo de regime empistonado é, na verdade, um caso limite, já que dificilmente encontraríamos um equipamento onde não houvesse uma certa turbulência que impedisse o fluido de se mover de forma empistonada. O outro extremo da modelagem (que obviamente não se aplica a modelagem de uma coluna de absorção) seria a mistura perfeita. A dispersão axial seria o comportamento intermediário entre estes dois casos extremos, ou seja, o fluido se comportaria como em regime empistonado, mas com uma certa mistura na direção axial. A modelagem neste caso é feita considerando-se que a dispersão axial apresenta um comportamento semelhante a um processo difusivo (obedecendo a lei de Fick), onde a difusividade seria substituída por uma constante: o número de Peclet. As equações resultantes seriam, então, as mesmas do regime empistonado, com o termo de dispersão axial.

Uma grande importância tem sido dada ao modelo de dispersão axial no caso de absorção em torres recheadas. Valores experimentais do número de Peclet têm sido levantados para diversas condições operacionais, com diversos tipos de recheios. Alguns autores têm sugerido correlações empíricas para o cálculo do número de Peclet.

No entanto, não há um consenso quanto à verdadeira relevância da dispersão axial em colunas recheadas. Associado a isto, os dados disponíveis na literatura não apresentam uma boa concordância entre si. Sater e Levenspiel (1966) mostram que, para condições operacionais idênticas, o número de Peclet varia por um fator de 10 entre os dados experimentais disponíveis na literatura. Mellish sugere que, para condições de

fluxos normalmente encontradas em operações industriais, a dispersão axial pode ser considerada desprezível.

Froment e Bischoff (1990) alertam que os desvios do regime empistonado em colunas de recheio são devidos principalmente a caminhos preferenciais ou a existência de zonas estagnadas, fenômenos estes que não são devidamente modelados pela dispersão axial. Estes concluem que, dado o pouco conhecimento que se tem a respeito de parâmetros do modelo de zonas estagnadas, a melhor alternativa para se modelar uma coluna de absorção recheada ainda é o de regime empistonado.

Dada estas incertezas sobre o modelo hidrodinâmico mais adequado, utilizaremos aqui o regime de fluxo empistonado.

4.4 - Cálculo do estado estacionário.

Na abordagem aqui utilizada, o desenvolvimento do simulador dinâmico utilizará, como veremos a seguir, dados de estados estacionários.

O cálculo do estado estacionário de uma coluna de absorção recheada tem sido bastante estudado na literatura. Dentre os trabalhos existentes, destaca-se pela abordagem mais rigorosa, o trabalho de De Leye e Froment (1986). Estes foram os primeiros a propor um algoritmo que utiliza as equações fundamentais de transferência de massa juntamente com balanços de massa do processo de absorção. A principal vantagem de se utilizar este tipo de resolução está no fato do algoritmo ser válido para qualquer tipo de reação e de regime cinético, ao contrário do que aconteceria se fossem utilizadas soluções analíticas.

Trataremos aqui o processo como sendo composto de três partes: seio do gás, seio do líquido e filme de transferência de massa. Obviamente, estas não são distintas, interferindo mutuamente uma sobre a outra. As equações devem portanto, ser resolvidas simultaneamente.

Fazendo um balanço de massa sobre as fatias infinitesimais mostradas na fig. 4.1, obtemos as seguintes equações:

Balanço global de gás:

$$\frac{dF}{dz} = -N_A\big|_{y=0}\, a'_V \Omega \tag{4.1}$$

Balanço para o componente absorvido:

$$\frac{F}{p_t - p_{Ab}}\frac{dp_{Ab}}{dz} = -N_A\big|_{y=0}\, a'_V \Omega \tag{4.2}$$

Balanço global de líquido:

$$\frac{\partial L}{\partial z} = \frac{M_A}{\rho_{L,A}} L \frac{\partial C_A}{\partial z} + \sum_{i=1}^{NR} \frac{M_{R_i}}{\rho_{L,R_i}} L \frac{\partial C_{R_i}}{\partial z} + \sum_{i=1}^{NP} \frac{M_{P_i}}{\rho_{L,P_i}} L \frac{\partial C_{P_i}}{\partial z} \tag{4.3}$$

Para o componente absorvido:

$$L\frac{\partial C_{Ab}}{\partial z} = -C_{Ab}\frac{\partial L}{\partial z} - N_A\big|_{v-y_L}\, a'_V \Omega + a_A r_l H_L \Omega \tag{4.4}$$

Para os reagentes:

$$L\frac{\partial C_{R_k b}}{\partial z} = -C_{R_k b}\frac{\partial L}{\partial z} + N_{R_k}\Big|_{y=y_L} a'_v\Omega + a_{R_k} r_l H_L \Omega \tag{4.5}$$

Para os produtos:

$$L\frac{\partial C_{P_k b}}{\partial z} = -C_{P_k b}\frac{\partial L}{\partial z} - N_{P_k}\Big|_{y=y_L} a'_v\Omega - a_{P_k} r_l H_L \Omega \tag{4.6}$$

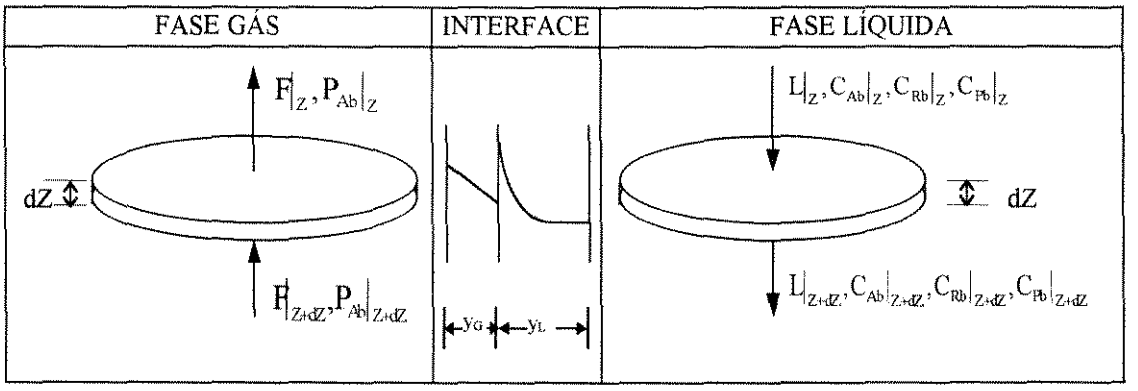

Figura 4.1 - Fatias infinitesimais sobre as quais foram feitos os balanços de massa na fase gás e no líquida.

O balanço global de líquido foi feito sob a forma volumétrica por facilitar o cálculo da velocidade deste, que é essencial no método das características a ser utilizado mais adiante na simulação dinâmica.

Os balanços para os componentes na fase líquida podem ser reduzidos a equações diferenciais ordinárias se for desprezado o termo de variação da vazão de líquido com a altura. O balanço global de líquido pode ser reduzido a uma equação algébrica. Desta forma, teríamos que as equações poderiam ser resolvidas por métodos como Euler, Runge-Kutta, etc. No entanto, deve-se observar que a cada passo de integração torna-se necessário calcular o fluxo de transferência de massa, o que pode tornar os cálculos bastante lentos.

Por outro lado, deve-se observar que alguns balanços contêm o termo da taxa de reação. Isto torna o sistema extremamente "*stiff*", o que dificulta bastante a sua resolução, já que métodos explícitos de integração como os anteriormente citados podem levar a instabilidade. Torna-se necessário utilizar métodos implícitos ou semi-implícitos, aumentando ainda mais o volume de cálculos a ser feito. Uma alternativa para este problema seria utilizar a taxa de reação no seio do líquido como sendo igual a zero, o que para reações instantâneas e rápidas é sempre verdade, já que todo componente absorvido é consumido no filme de transferência de massa. Isto no entanto não é válido para reações moderadamente rápidas ou lentas, o que torna esta consideração de uso limitado.

Para evitar perda de generalidade, não utilizaremos aqui os balanços na forma diferencial ordinária. Para a resolução destes utilizaremos o método da colocação ortogonal em elementos finitos, que já foi utilizado na resolução das equações da Teoria

do Filme. Este método, além de permitir a integração dos balanços sem que seja necessária nenhuma simplificação, tem ainda a vantagem de necessitar do cálculo dos fluxos de transferência de massa apenas nos pontos de colocação, o que reduz o esforço computacional. Além disto, a escolha da colocação ortogonal em elementos finitos em detrimento da colocação ortogonal global se deve aos perfis de concentração acentuados que são normalmente encontrados nesse tipo de processo, o que poderia levar a oscilações da solução em regiões onde se tivesse uma concentração pequena.

Aplicando colocação ortogonal em elementos finitos nas equações (4.1) a (4.6), obtemos as seguintes equações:

Balanço global de gás:

$$\sum_{i=1}^{ND} A_{ji} F_i^l = -N_A\big|_{y=0} a'_V \Omega Z \Delta x^l \tag{4.7}$$

Balanço para o componente absorvido:

$$\frac{F_j^l}{p_t - p_{Ab}^l} \sum_{i=1}^{ND} A_{ji} p_{Abj}^l = -N_A\big|_{y=0} a'_V \Omega Z \Delta x^l \tag{4.8}$$

Balanço global de líquido:

$$\sum_{i=1}^{ND} A_{ji} L_i^l = \frac{M_A}{\rho_{L,A}} L_j^l \sum_{i=1}^{ND} A_{ji} C_{Abi}^l + \sum_{k=1}^{NR} \frac{M_{R_i}}{\rho_{L,R_i}} L_j^l \sum A_{ji} C_{P_k bi}^l + \sum_{i=1}^{ND} \frac{M_{P_i}}{\rho_{L,P_i}} L_j^l \sum_{i=1}^{ND} A_{ji} C_{R_k bi}^l \tag{4.9}$$

Para o componente absorvido:

$$L_j^l \sum_{i=1}^{ND} A_{ji} C_{Abi}^l = -C_{Abj}^l \sum_{i=1}^{ND} A_{ji} L_i^l - N_A\big|_{y-y_L} a'_V \Omega Z \Delta x^l + a_A r_l H_L \Omega Z \Delta x^l \tag{4.10}$$

Para o k-ésimo reagente:

$$L_j^l \sum_{i=1}^{ND} A_{ji} C_{R_k bi}^l = -C_{R_k bi}^l \sum_{i=1}^{ND} A_{ji} L_i^l + N_{R_k}\big|_{y=y_L} a'_v \Omega Z \Delta x^l + a_{R_k} r_l H_L \Omega Z \Delta x^l \tag{4.11}$$

Para o k-ésimo produto:

$$L_j^l \sum_{i=1}^{ND} A_{ji} C_{P_k bi}^l = -C_{P_k bi}^l \sum_{i=1}^{ND} A_{ji} L_i^l - N_{P_k}\big|_{y=y_L} a'_v \Omega Z \Delta x^l - a_{P_k} r_l H_L \Omega Z \Delta x^l \tag{4.12}$$

Como as equações diferenciais são de 1ª ordem, uma única condição de contorno para cada equação é necessária para a resolução das mesmas. Estas são dadas por:

$$F_l^{l+1} = F_{ND}^l, \quad l = 1,\ldots,NE-1 \tag{4.13}$$

$$F_{ND}^{NE} = F^{in} \tag{4.14}$$

$$p_{Ab\,l}^{\,l+1} = p_{Ab\,ND}^{\,l}, \quad l = 1,\ldots,NE-1 \tag{4.15}$$

$$p_{Ab\,ND}^{\,NE} = p_{Ab}^{in} \tag{4.16}$$

$$L_{ND}^{l-1} = L_l^l, \quad l = 2,\ldots,NE \tag{4.17}$$

$$L_1^l = L^{in} \tag{4.18}$$

$$C_{Ab,ND}^{l-1} = C_{Ab,l}^l, \quad l = 2,\ldots,NE \tag{4.19}$$

$$C_{Ab,1}^l = C_{Ab}^{in} \tag{4.20}$$

$$C_{R_k b,ND}^{l-1} = C_{R_k b,l}^l, \quad l = 2,\ldots,NE \tag{4.21}$$

$$C_{R_k b,1}^l = C_{R_k b}^{in} \tag{4.22}$$

$$C_{P_k b,ND}^{l-1} = C_{P_k b,l}^l, \quad l = 2,\ldots,NE \tag{4.23}$$

$$C_{P_k b,1}^l = C_{P_k b}^{in} \tag{4.24}$$

As equações (4.7) a (4.23) formam um sistema não-linear. Deve-se observar que algumas destas possuem termos de fluxo de transferência de massa. Estes fluxos, por sua vez, podem ser calculados a partir das concentrações nos seios das soluções e das concentrações. O sistema foi resolvido pelo método de Newton-Raphson. Este método exige o cálculo do Jacobiano. Como já mencionamos ao falar da resolução das equações de transferência de massa, devido ao grande número de somatórios presentes nas equações acima, o cálculo do jacobiano de forma numérica se torna muito demorado e pode ter grandes erros de arredondamento. Desta forma foi utilizado jacobiano calculado de forma analítica, exceto no cálculo das derivadas do fluxo, onde isso seria impossível. A forma de resolução das equações pode ser resumida no fluxograma da fig.4.2.

A seguir é dado um detalhamento maior de cada bloco da fluxograma da fig. 4.2.

Estimativa inicial dos perfis de concentração e fluxos.

Uma estimativa inicial é requerida para que o método de Newton-Raphson possa prosseguir. Uma tentativa inicial bem grosseira seria um reta entre um valor inicial (condição de entrada) e um final (um valor também estimado). No entanto, uma estimativa rápida e bem mais próxima da solução é obtida quando as equações (4.1) a (4.6) são reduzidas a equações diferenciais ordinárias (desprezando o termo de variação da vazão de líquido ao longo da coluna) e considerando-se que a taxa de reação é sempre nula. Assim, temos as seguintes equações para os balanços na fase líquida:

Balanço global de líquido:

$$L(z+\Delta z) = L(z)\left\{1+\frac{M_A}{\rho_{L,A}}\left[C_{Ab}(z+\Delta z)-C_{Ab}(z)\right]+\right.$$

$$\left.\sum_{k=1}^{NR}\frac{M_{R_k}}{\rho_{L,R_k}}\left[C_{R_kb}(z+\Delta z)-C_{R_kb}(z)\right]+\sum_{k=1}^{NP}\frac{M_{P_k}}{\rho_{L,P_k}}\left[C_{P_kb}(z+\Delta z)-C_{P_kb}(z)\right]\right\} \qquad (4.25)$$

Para o componente absorvido:

$$L\frac{\partial C_{Ab}}{\partial z} = -N_A\Big|_{y=y_L} a'_V\Omega \qquad (4.26)$$

Para os reagentes:

$$L\frac{\partial C_{R_kb}}{\partial z} = N_{R_k}\Big|_{y=y_L} a'_V\Omega \qquad (4.27)$$

Para os produtos:

$$L\frac{\partial C_{P_kb}}{\partial z} = -N_{P_k}\Big|_{y=y_L} a'_V\Omega \qquad (4.28)$$

Neste formato, as equações podem ser integradas por Euler (ou qualquer outro método explícito). A integração começa da base, onde é conhecida a concentração de entrada do gás, mas não a de saída do líquido. Esta última é calculada considerando-se que é absorvido todo o soluto da corrente gasosa e que todo este é consumido pela quantidade estequiométrica equivalente de reagente do líquido.

O fluxo de transferência de massa é calculado no primeiro ponto de colocação (na base da coluna). As equações são integradas considerando-se que o fluxo não irá variar até o próximo ponto de colocação. No segundo ponto recalcula-se o fluxo de transferência de massa. A partir daí, considera-se que o fluxo irá variar de um ponto para o outro na mesma proporção em que ele variou entre os dois pontos anteriores.

A estimativa inicial assim obtida é bem próxima à solução do sistema, principalmente quando não há passagem do componente absorvido para o seio do líquido e quando a composição de topo do gás é baixa (ou seja, praticamente todo o soluto do gás foi absorvido). Desta forma, o algoritmo converge em um número bem menor de iterações.

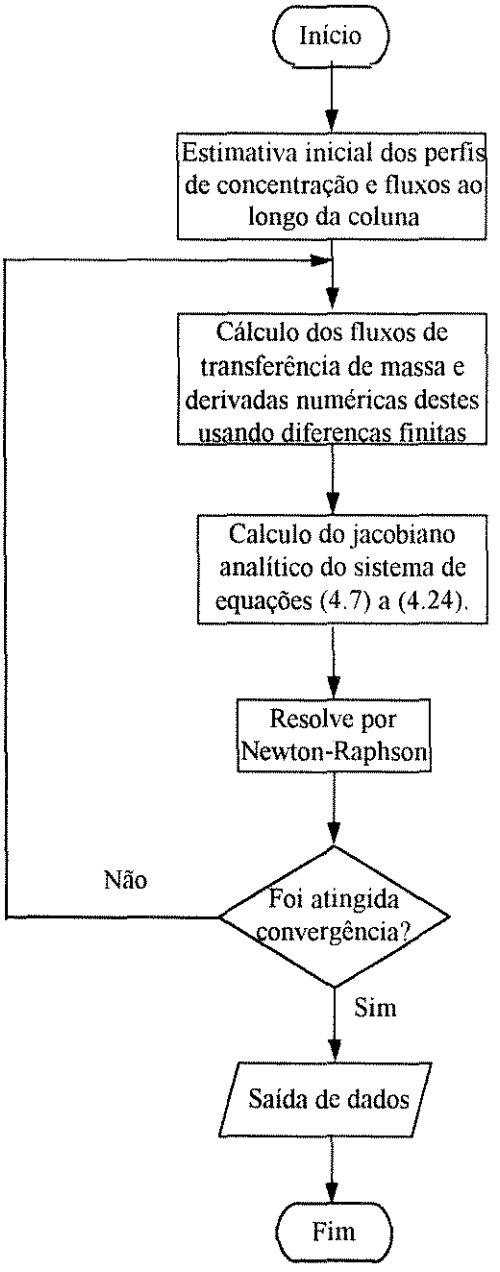

Figura 4.2 - Fluxograma de resolução das equações do estado estacionário.

Calculo dos fluxos de transferência de massa e das derivadas deste.

Como já foi mostrado no capítulo 3 e no apêndice A, a partir das composições nos seios do gás e do líquido, é possível calcular os fluxos de transferência de massa. No

entanto, para o cálculo do jacobiano, surgem termos de derivadas dos fluxos em relação as concentrações e fluxos mássicos. Estes foram calculados por diferenças finita.

Cálculo do jacobiano analítico do sistema de equações (4.1) a (4.24).

A partir das derivadas dos fluxos , o cálculo do jacobiano analítico é trivial, já que as funções são basicamente somatórios. A sub-rotina FCN1 (Apêndice B), escrita em Fortran 77, apresenta o cálculo deste. A taxa de reação foi considerada em uma sub-rotina em separado, de forma que, ao se mudar de sistema, basta modificar a equação da taxa nesta.

Resolução por Newton-Raphson.

A partir dos valores das funções (**F(x)**) e do jacobiano desta (**J(x)**), a atualização das variáveis é dada por:

$$\mathbf{x}_{k+1} - \mathbf{x}_k = -\frac{\mathbf{F(x)}}{\mathbf{J(x)}} \tag{4.30}$$

Critério de convergência.

O critério de convergência utilizado aqui foi tomado com base na diferença relativa do valor das variáveis entre duas iterações sucessivas. Para se considerar atingida a convergência, a maior diferença relativa (em valor absoluto) não poderia ser maior que uma dada tolerância. A tolerância aqui utilizada foi de 10^{-4}, ou seja, os resultados eram obtidos com quatro algarismos significativos.

O algoritmo acima descrito apresenta um mínimo de esforço computacional associado a uma grande precisão que é fornecida pelos métodos numéricos utilizados e por não ser necessário nenhum tipo de consideração na resolução das mesmas. No apêndice B encontra-se o programa EST.FOR, escrito em Fortran 77, que a partir das concentrações e vazões de entrada do gás e do líquido e da altura do recheio, calcula os perfis ao longo da coluna da forma aqui descrita.

4.4.1 - Resultados.

Como já foi mencionado, os resultados de diferentes estados estacionários serão utilizados mais a frente, no desenvolvimento de um simulador dinâmico. Embora tenha sido esta a principal finalidade do desenvolvimento deste programa, o mesmo pode ser utilizado para o projeto de colunas de recheadas, bastando para isto criar um laço que modificasse a altura até se obter a saída desejada. Outras etapas de projeto (escolha do tamanho e tipo do recheio, determinação do diâmetro da coluna, etc.), poderiam ser facilmente calculadas por regras simples (Caldas (1988)). No entanto, o projeto da coluna como um todo foge ao escopo deste trabalho.

Como estudo de caso (inclusive para as simulações dinâmicas e o estudo de controle), a coluna aqui utilizada será a mesma do exemplo de De Leye e Froment (1986). Esta é um exemplo típico de colunas de absorção com reação química. As especificações desta encontram-se resumidas na tabela 4.2. A coluna opera a temperatura de 315K e pressão de 14,3bar. A única consieradação utilizada é que a coluna é

isotérmica, o que limita a validade do modelo para sistema onde as concentrações de entrada não sejam muito grandes.

O sistema a ser estudado é o da absorção de CO_2 em solução aquosa de MEA. As propriedades físicas bem como as características deste sistema estão descritas no capítulo 3, que trata do tópico de transferência de massa com reação química.

Tabela 4.2 - Especificações da coluna utilizada como estudo de caso.

Altura	10,95m
Diâmetro da coluna	1,05m
Tipo de Recheio	Pall Ring
Área interfacial do recheio	105m^2
Tamanho nominal do recheio	0,05m

No cálculo dos coeficientes de transferência de massa foram utilizadas as correlações sugeridas por Onda *et al.* (1968a):

$$k_L\left(\frac{\rho_L}{\mu_L g}\right)^{1/3} = 0{,}0051\left(\frac{L}{a'_v \mu_L}\right)^{2/3}\left(\frac{\mu_L}{\rho_L D_L}\right)^{-1/2}\left(a_v D_P\right)^{0,4} \tag{4.31}$$

$$k_G\left(\frac{RT}{a_v D_G}\right) = 5{,}23\left(\frac{G}{a_v \mu_G}\right)^{0,7}\left(\frac{\mu_G}{\rho_G D_G}\right)^{1/3}\left(a_v D_P\right)^{-2,0} \tag{4.32}$$

onde L se refere a vazão de líquido e G a vazão de gás. As unidades devem ser escolhidas de forma que os grupos do lado direito das expressões sejam adimensionais.

Tabela 4.3 - Comparação entre as composições e vazões no topo e na base da coluna obtidas neste trabalho e os resultados de De Leye e Froment (1986).

	Resultados Obtidos		De Leye e Froment	
Fluxos	Base	Topo	Base	Topo
Vazão de gás (kmol/m^{2}h)	497,0	429,1	497,7	429,7
Pressão parcial de CO_2 (bar)	1,954	7,16x10^{-4}	1,954	7,15x10^{-4}
Vazão de líquido (m^3/h)	79,84	76,90	79,80	76,90
Concentração de reagente(kmol/m^3)	0,438	2,220	0,435	2,220
Concentração de produtos (kmol/m^3)	0,850	0,000	0,825	0,000
Altura do Recheio (m)	10,95		11,00	

A área interfacial de transferência de massa não é igual a área interfacial do recheio. Para corrigir esta, utilizou-se a expressão sugerida por Onda *et al.* (1969b):

$$\frac{a'_v}{a_v} = 1 - \exp\left[-1.45 \left(\frac{\sigma_c}{\sigma}\right)^{0,75} \left(\frac{L}{a_v \mu_L}\right)^{0,1} \left(\frac{L^2 a_v}{\rho_L{}^2 g}\right)^{-0,05} \left(\frac{L^2}{\rho_L \sigma a_v}\right)^{0,2} \right] \qquad (4.33)$$

onde novamente as unidades utilizadas devem ser tais que os termos entre parênteses sejam adimensionais.

A tabela 4.3 mostra as composições e vazões de entrada passadas como condições de contorno bem como as composições e vazões de saída calculados e os resultados obtidos por De Leye e Froment (1986). A excelente concordância entre os resultados mostra a validade do algoritmo aqui utilizado na resolução das equações.

4.5 - Cálculo dos fluxos de transferência de massa a partir das redes neurais aritificiais.

O Método das Características, o qual será utilizado na simulação dinâmica, necessita do cálculo do fluxo de transferência de massa em um grande número de pontos ao longo da coluna, além de ser um método iterativo, conforme vimos no capítulo 2. Desta forma, o cálculo dos fluxos de transferência de massa pela resolução direta das equações fundamentais se torna praticamente impossível devido ao grande tempo computacional envolvido.

Para evitar a utilização de soluções analíticas, que possuem algumas desvantagens já comentadas no capítulo 3, utilizaremos uma abordagem baseada em redes neurais artificiais, constituindo o que é conhecido na literatura como modelo neural híbrido (Morris Montague, 1994), onde equações que descrevem o fenômeno são utilizadas juntamente com as RNA's.

As redes neurais artificiais possuem características que são extremamente importantes para a utilização no cálculo dos fluxos de transferência de massa, dentre as quais destaca-se a capacidade de tratar dados altamente não-lineares, de ter ótimas características interpoladoras e de ser aproximador universal de funções. Todas essas características são extremamente importantes neste caso, já que o conjunto de dados é altamente não-linear e seria difícil definir uma função que pudesse aproximar estes.

Os princípios básicos de aplicação das redes neurais já foram discutidos no capítulo 2. Aqui discutiremos apenas como estas foram utilizadas.

O primeiro passo na utilização das RNA's diz respeito a definição das entradas. As entradas utilizadas devem dar informação suficiente sobre o conjunto de saídas que se está querendo inferenciar. No caso do fluxo de transferência de massa, as informações necessárias são todas as concentrações no seio do gás e do líquido e as vazões de gás e de líquido. Note que estas são as informações que são necessárias para se fazer os mesmo cálculos através das equações fundamentais da Teoria do Filme (as vazões de gás e de líquido são necessárias no cálculo dos coeficientes de transferência de massa).

As saídas desejadas neste caso são todos os fluxos de transferência de massa. Embora se pudesse utilizar uma única rede com várias saídas - já que as entradas são as mesmas - observou-se que o treinamento de várias redes com uma única saída era mais rápido (neste caso) que o treinamento de um única rede com várias saídas.

Para o sistema aqui utilizado (CO_2-MEA), as entradas são as seguintes: Concentração de reagente, concentração dos (dois) produtos, vazão volumétrica do líquido, pressão parcial de CO_2 no gás, e fluxo molar do gás. Assim, temos um total de seis entradas. A concentração de CO_2 no seio do líquido poderia ser incluída. No entanto para o regime cinético deste sistema, observou-se que a concentração de CO_2 no líquido é sempre zero. Caso fosse utilizado outro sistema onde isso não acontecesse, esta deveria ser incluída como entrada.

Deve-se observar que o número de combinações de valores possíveis com estas seis variáveis de entrada é muito grande. A utilização de todas as combinações possíveis proporcionaria um conjunto de dados muito grande e também desnecessário, já que nem todas as possibilidades são utilizadas. Por exemplo, gás a uma composição típica do topo da coluna nunca vai estar em contato com líquido contendo baixas concentrações de reagentes (encontradas na base da coluna).

Para contornar este problema, foram utilizados dados de estados estacionários. Para cada estado estacionário escolheu-se um determinado número de pontos ao longo da coluna (escolhidos, por conveniência, como sendo os pontos de colocação). Nesses pontos as composições e fluxos de transferência de massa foram passados para uma arquivo de dados, que será utilizado no treinamento da rede. Se considerarmos que no estado transiente a coluna transita entre estados estacionários, um conjunto de vários estados estacionários deve conter informação suficiente para o treinamento das RNA's e futura inferenciação (mais a frente faremos testes para verificar a veracidade disto).

Para gerar cada estado estacionário se faz necessário passar as composições e vazões mássicas de entrada. Estas, por sua vez, devem cobrir todo a faixa de variação das entradas que a coluna possa vir a ter no estado transiente. Assim, ainda resta um grande conjunto de entradas possíveis. Algumas considerações sobre o processo podem diminuir bastante o número de possibilidades.

Na fig. 4.3 temos a representação de um processo de absorção com reação química. Embora este processo seja bastante simplificado e possa variar bastante de caso para caso, o sistema básico mostrado aqui (coluna de absorção seguido de uma coluna de "*stripper*" para recuperação do solvente) é o mais utilizado em absorção com reação química. A partir deste podemos concluir que, sendo o sistema fechado, a concentração de reagente é praticamente constante na entrada da absorvedora. Além disto, a concentração de produtos deve ser sempre nula se o sistema de controle da "*stripper*" funcionar adequadamente.

Desta forma, o conjunto de dados a ser utilizado no treinamento das RNA's deve conter valores de entrada da vazão de gás, composição do gás e vazão de líquido tais que consiga cobrir toda a faixa de variação que possa vir a acontecer com a coluna durante o seu funcionamento. A suposição de que a concentração de reagente e de produtos na entrada da coluna não varia não implica que durante a simulação, se isto ocorrer, as redes não serão capazes de calcular o fluxo. A rede conseguirá predizer bem os valores de fluxos se as flutuações na concentração de reagentes e produtos estiverem dentro da faixa a qual a rede foi treinada. Como num sistema fechado estas variações são pequenas, conclui-se que não é necessário incluí-las nos dados de treinamento da redes.

Restam ainda três variáveis cujos valores devem ser escolhidos de tal forma que se consiga cobrir a maior faixa possível. Neste ponto as limitações de carga que uma coluna recheada pode suportar é importante. O ponto de inundação nos dá o valor máximo da vazão de líquido e de gás. Já o grau mínimo de molhabilidade nos dá um valor mínimo da vazão de líquido.

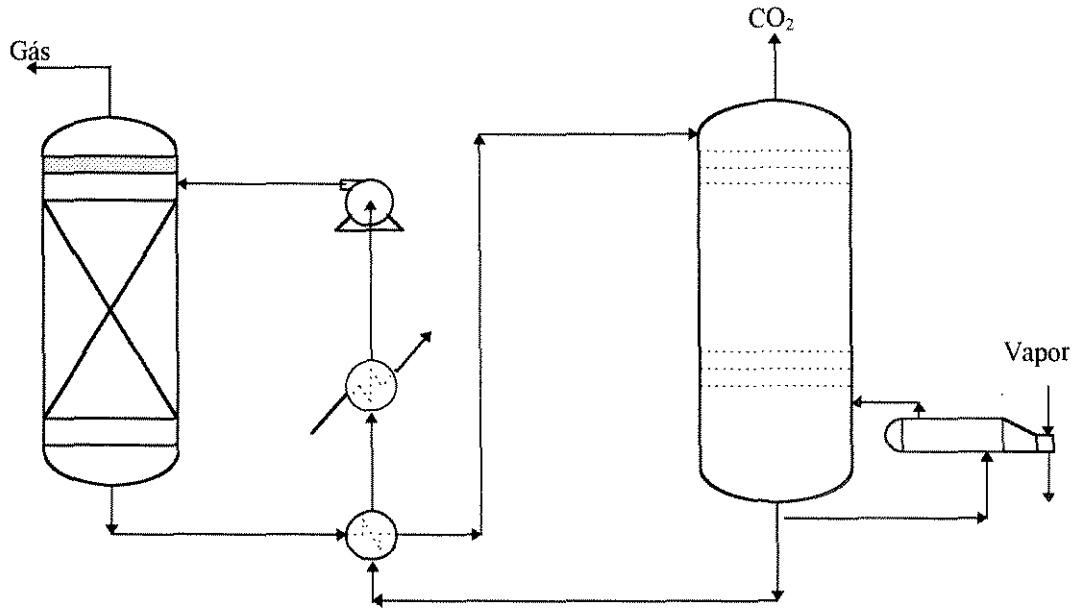

Figura 4.3 - Diagrama idealizado de um processo de absorção com reação química.

Tendo em vista que as colunas recheadas só podem variar em uma faixa próxima àquelas para os quais foram projetadas, a escolha dos estados estacionários deve ser feita em torno das condições de projeto. Desta forma, o seguinte critério na construção do conjunto de dados foi o seguinte:

• Adotou-se uma faixa de variação para a composição de entrada em torno de uma valor de projeto (tabela 4.3). A faixa aqui utilizada está entre -40% e +40% do valor de projeto, tendo sido utilizado valores em intervalos de 10%. Valores abaixo dos utilizados implicam que a menor vazão possível de líquido (necessária para garantir a taxa mínima de molhabilidade) ainda é grande o suficiente para manter a coluna em uma composição de topo do gás bem inferior à de projeto. Para valores acima da faixa a maior vazão de líquido que pode ser utilizada (para que não haja arraste) não é suficiente para que a composição de topo seja mantida próxima ao valor desejado, a não ser que a vazão de gás seja muito baixa;

• Para cada concentração de entrada do gás utilizou-se uma série de combinações de vazão do gás e do líquido. Os valores de vazão de gás e de líquido utilizados estavam em intervalos de 5% dos respectivos valores de projeto. A menor vazão de líquido utilizada foi tomada com base no grau mínimo de molhamento. O valor máximo utilizado foi tal que a máxima vazão de gás permitida seria muito pequena, resultando em uma composição de topo muito abaixo do desejado.

O arquivo de dados assim obtido é válido para a faixa operacional que a coluna pode vir a assumir. Caso sejam utilizados 12 pontos ao longo da coluna, o conjunto de dados assim obtido contêm cerca de 3.000 pares de treinamento.

Como uma grande precisão é requerida, utilizou-se o algoritmo de Marquardt-Levenberg, o qual é bem mais adequado para este caso. A rede utilizada tem uma configuração (6-22-1), ou seja, 6 neurônios de entrada, 22 na camada intermediária e 1

neurônio de saída. A quantidade de neurônios na camada intermediária foi determinada após sucessivos testes onde se aumentou o número destes até se obter a precisão desejada. Os valores de μ e β, parâmetros do algoritmo de treinamento, foram 0,01 e 1,2. O valor de μ utilizado foi o sugerido por Hagan *et al* (1994). O valor de β foi escolhido (por tentativas) de forma a garantir um maior número de iterações que representassem uma redução do erro.

Utilizou-se adimensionalização linear para todas as variáveis, exceto para a composição do gás e o fluxo de transferência de massa, os quais variam por um fator de cerca de 10.000 entre o maior e menor valor.

Quanto à convergência, foi utilizado um critério local: só se considera atingida a convergência quando o maior erro dentre todos fosse inferior a uma certa tolerância. Esse erro foi tomado de forma percentual em relação ao valor desejado e a tolerância adotada foi de 0,1%. O critério do erro quadrático, muito utilizado na literatura, foi abandonado pelo fato de ser um critério global e que dá portanto, pouca informação sobre como vai o erro em cada um dos pontos. Assim, embora este critério pudesse apresentar um valor aparentemente baixo, um erro grande poderia estar ocorrendo em alguns pontos. Em especial, foi observado que os pontos de menor fluxo de transferência de massa eram os que apresentavam maior erro percentual. Desta forma, a adoção de um critério global poderia mascarar um erro relativo grande, onde os fluxos de transferência de massa fossem baixos.

É importante ressaltar que, uma vez treinada, a RNA é capaz de predizer o fluxo de transferência de massa para todas as condições que a coluna possa a vir assumir e, mais importante, de forma bastante rápida. A etapa mais lenta na utilização das redes neurais é a criação do arquivo de dados e, principalmente, o treinamento da rede neural. A forma como o programa foi construído permite que as redes neurais seja substituídas por uma solução analítica.

Preferiu-se utilizar as redes neurais aqui tanto pela precisão dos resultados, como pela velocidade de inferenciação desta após treinada. Além disto, da mesma forma como dados obtidos da resolução das equações da Teoria do Filme, as RNA's poderiam ter sido treinadas com dados experimentais, sem a necessidade de especificar nenhum modelo.

4.6 - Simulação dinâmica de colunas recheadas de absorção com reação química utilizando modelo neural híbrido.

Como exemplo de simulação dinâmica de colunas recheadas podemos citar os trabalhos de Sakata e Prados (1972), Hoerner e Shiesser (1961), e Lees (1968). Estes trabalhos usam uma formulação matemática que só permite verificar o comportamento da coluna frente a uma única perturbação e não a um conjunto de perturbações sucessivas; Najim (1991), em um trabalho de estudo de controle da uma coluna de absorção de CO_2 em solução aquosa de MEA, utilizou um modelo simplificado para simular a coluna.

Dentre os trabalhos de simulação dinâmica de colunas de absorção com reação química recheadas, o mais completo talvez seja o de Bradley e Andre (1972). Estes autores fizeram a simulação com o sistema CO_2 - MEA e compararam com dados

experimentais obtidos em uma coluna piloto. Foi considerado que a variação do fluxo de líquido ao longo da coluna seria desprezível e que as mudanças na vazão de líquido seriam pequenas, o que simplifica bastante a resolução das equações pelo método das características, tornando o modelo, no entanto, limitado para casos onde a vazão de líquido possa variar em patamares maiores (o que pode vir a acontecer em uma coluna sob controle). No cálculo dos fluxos de transferência de massa, estes autores utilizaram uma função ajustada a dados experimentais de concentração de CO_2 na interface gás-líquido, o que torna os cálculos bastante simples, mas é de uso limitado a esse sistema unicamente.

No desenvolvimento aqui utilizado, procuraremos evitar as simplificações já discutidas até aqui, de forma que no estudo do sistema de controle tenhamos as principais características do processo presentes na simulação. Com isto, espera-se obter conclusões a respeito do sistema de controle muito mais confiáveis.

4.6.1 - Desenvolvimento dos balanços dinâmicos.

Para os balanços dinâmicos, utilizamos um desenvolvimento um pouco diferente do utilizado anteriormente para os balanços estáticos. Isto foi feito para que os balanços finais pudessem ser resolvidos pelo método das características. As razões para isto ficarão mais claras quando falarmos sobre o algoritmo de resolução das equações, mais adiante. Sendo ρ a densidade do líquido (aqui considerada constante), os balanços de massa sobre as fatias infinitesimais mostradas na fig. 4.1 são:

Balanços na fase líquida:

$$\frac{\partial L}{\partial t}+\frac{1}{\Omega}\frac{\partial L}{\partial H_L}\frac{\partial L}{\partial z}=\frac{\partial L}{\partial H_L}\frac{N_A\big|_{y=y_G}a'_v M_A}{\rho} \tag{4.34}$$

$$\frac{\partial C_A}{\partial t}+\frac{L}{H_L\Omega}\frac{\partial C_A}{\partial z}=\frac{N_A\big|_{y=y_L}a'_v}{H_L}-\frac{C_A M_A}{\rho H_L}N_A\big|_{y=y_G}a'_V-a_A r \tag{4.35}$$

$$\frac{\partial C_{R_K}}{\partial t}+\frac{L}{H_L\Omega}\frac{\partial C_{R_K}}{\partial z}=\frac{N_{R_K}\big|_{y=y_L}a'_v}{H_L}-\frac{C_{R_K}M_{R_K}}{\rho H_L}N_A\big|_{y=y_G}a'_V-a_{R_K}r \tag{4.36}$$

$$\frac{\partial C_{P_K}}{\partial t}+\frac{L}{H_L\Omega}\frac{\partial C_{P_K}}{\partial z}=\frac{N_{P_K}\big|_{y=y_L}a'_v}{H_L}-\frac{C_{P_K}M_{P_K}}{\rho H_L}N_A\big|_{y=y_G}a'_V-a_{P_K}r \tag{4.37}$$

Para a fase gás:

$$\frac{\partial F}{\partial z}-N_A\big|_{y=y_G}a'_V\Omega=0 \tag{4.38}$$

$$\frac{\partial P_{Ab}}{\partial t}-\frac{F}{H_G\Omega}\frac{\partial P_{Ab}}{\partial z}=-\frac{N_A\big|_{y=y_G}a'_V\left(P_t-P_{Ab}\right)}{H_G} \tag{4.39}$$

Através do conceito de derivada substantiva, as equações (4.35) a (4.37) e (4.39) podem ser transformadas em equações diferenciais ordinárias, já que velocidade do gás e do líquido são dadas respectivamente por:

$$\left.\frac{dz}{dt}\right|_{I} = \frac{L}{\Omega H_L} \tag{4.40}$$

$$\left.\frac{dz}{dt}\right|_{II} = -\frac{F}{\Omega H_G} \tag{4.41}$$

Desta forma, rescrevendo estas equações obtemos:

$$\left.\frac{dC_A}{dt}\right|_{I} = \frac{N_A|_{y=y_L} a'_V}{H_L} - \frac{C_A M_A}{\rho H_L} N_A|_{y=y_G} a'_V - a_A r \tag{4.42}$$

$$\left.\frac{dC_{R_K}}{dt}\right|_{I} = \frac{N_{R_K}|_{y=y_L} a'_V}{H_L} - \frac{C_{R_K} M_{R_K}}{\rho H_L} N_A|_{y=y_G} a'_V - a_{R_K} r \tag{4.43}$$

$$\left.\frac{dC_{P_K}}{dt}\right|_{I} = \frac{N_{P_K}|_{y=y_L} a'_V}{H_L} - \frac{C_{P_K} M_{P_K}}{\rho H_L} N_A|_{y=y_G} a'_V - a_{P_K} r \tag{4.44}$$

$$\left.\frac{dP_{Ab}}{dt}\right|_{II} = -\frac{N_A|_{y=y_G} a'_V (P_t - P_{Ab})}{H_G} \tag{4.45}$$

Os índices I e II nas equações (4.42) a (4.45) indicam que estas somente são válidas ao longo das características (4.40) e (4.41), respectivamente, ou seja, o tempo e a posição das grandezas obtidas na integração destes balanços estão relacionados pelas equações (4.40) e (4.41). Desta forma, acompanha-se todas as variações que ocorrem em um elemento de fluido que surge na base até o momento em que ele chega ao topo da coluna ou vice-versa. As descontinuidades não representam nenhum problema à solução das equações por este método, justamente porque se acompanha cada elemento isoladamente.

Os balanço globais para o gás e para o líquido não podem ser colocados no mesmo formato dos balanços para os componentes, pelo fato destes possuírem curvas características diferentes das velocidades do gás e do líquido. No entanto, estes podem ser colocados num formato que permita a sua resolução pelo método das características. Sendo conhecidas as concentrações dos componentes na fase líquida e a pressão parcial de "A" na fase gás, podemos desenvolver os seguintes balanços entre dois pontos distintos da coluna.

$$F(z+\Delta z) = F(z)\frac{\left(1 - \frac{P_{Ab}(z)}{P_t}\right)}{\left(1 - \frac{P_{Ab}(z+\Delta z)}{P_t}\right)} \tag{4.46}$$

$$L(z+\Delta z) = L(z)\left\{1 + \frac{M_A}{\rho_{L,A}}\left[C_{Ab}(z+\Delta z) - C_{Ab}(z)\right] + \sum_{k=1}^{NR} \frac{M_{R_k}}{\rho_{L,R_k}}\left[C_{R_k b}(z+\Delta z) - C_{R_k b}(z)\right] + \sum_{k=1}^{NP} \frac{M_{P_k}}{\rho_{L,P_k}}\left[C_{P_k b}(z+\Delta z) - C_{P_k b}(z)\right]\right\} \tag{4.47}$$

Desta forma, as equações (4.42) a (4.47) constituem o sistema que deve ser integrado ao longo das velocidades características (4.40) e (4.41) para o gás e para o líquido, respectivamente. Deve-se observar que as equações acima poderiam ser utilizadas também para o caso da absorção física, bastando para isto zerar as concentrações de reagentes e de produtos ao longo da coluna. O que no entanto vai

diferenciar a absorção física da absorção química é o cálculo dos fluxos de transferência de massa. Esta é a etapa mais lenta na simulação deste tipo de processo. Associado a isto, como foi visto no capítulo 2, o método das características, além de ser iterativo, necessita do cálculo dos fluxos de transferência de massa em um grande número de pontos ao longo da coluna. Isto praticamente torna a resolução deste modelo proibitivo se for utilizada a resolução das equações de transferência de massa em todos os pontos necessários.

A partir do cálculo do fluxo de transferência de massa pelas RNA's e dos balanços de massa no formato como exposto acima, a simulação transiente se torna direta a partir do Método das Características, na forma como descrito no capítulo 2. Uma descrição mais detalhada do algoritmo utilizado é dada a seguir.

4.6.2 - Algoritmo dinâmico.

O Método das Características necessita que uma malha de pontos com valores de concentração e fluxos mássicos ao longo da coluna seja passada como condição de contorno no tempo inicial. Esta malha inicial será aqui gerada a partir de um perfil estacionário da coluna, da forma como foi gerada pelo programa EST.FOR, descrito anteriormente. A saída deste perfil está na forma de concentrações e fluxos nos pontos de colocação, já que este foi o método numérico utilizado na resolução dos balanços. Para calcular o espaçamento entre os pontos da malha inicial, utilizou-se as equações (4.40) e (4.41). Assim, para um dado passo de integração no tempo, o procedimento para o cálculo na fase gás pode ser resumido no fluxograma da fig. 4.4. A interpolação polinomial a que se refere este fluxograma é feita com base no polinômio ortogonal que aproxima a solução do estado estacionário.

O mesmo procedimento utilizado na fase gás deve ser adotado na fase líquida, sendo que neste caso o início do cálculo da malha é pelo topo da coluna (z=0) e deve ser feita até se alcançar a base (z=Z).

Após se determinar os pontos da malha inicial, as concentrações do gás e do líquido devem ser obtidas nestes também por interpolação polinomial.

Após o cálculo da malha inicial, o processo de integração propriamente dito pode ser iniciado. Este segue basicamente o que já foi discutido no capitulo 2, onde se apresentou o Método das Características, sendo utilizadas as equações desenvolvidas anteriormente.

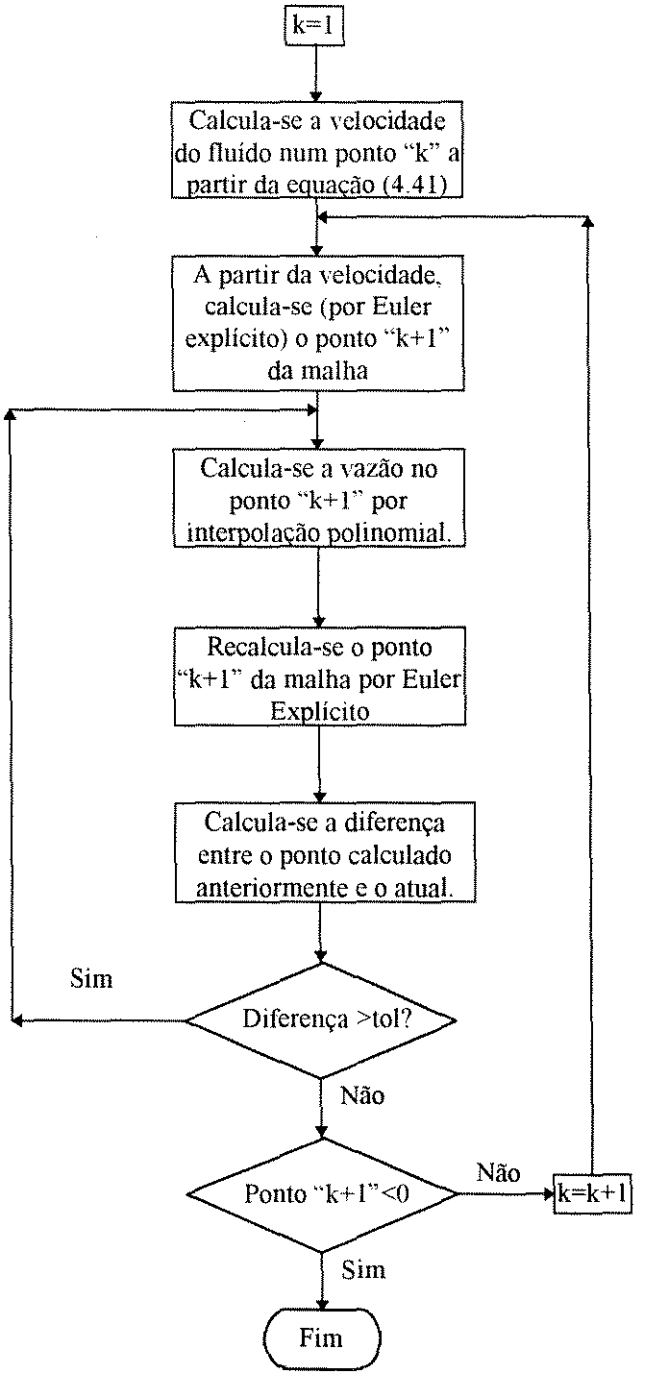

Figura 4.4 - Determinação da malha inicial do Método das Características.

4.6.3 - Dinâmica da base e topo da coluna.

No estudo do comportamento transiente de uma coluna de absorção, o reservatório de líquido na base da coluna e o abatedor de gotas no topo da coluna podem ter uma grande influência (Bradley e Andre (1972)).

Estes sistemas foram aqui modelados considerando que, em ambos, a hipótese de mistura perfeita seja aplicável.

Fazendo um balanço de massa sobre os elementos mostrados na fig. 4.5, obtemos as seguintes equações, para a base e para o topo, respectivamente:

$$\frac{dC}{dt} = \frac{L}{V}(C^{in} - C) \tag{4.48}$$

$$\frac{dP}{dt} = \frac{FRT}{VP_t}\left(P^{in} - P\right) \tag{4.49}$$

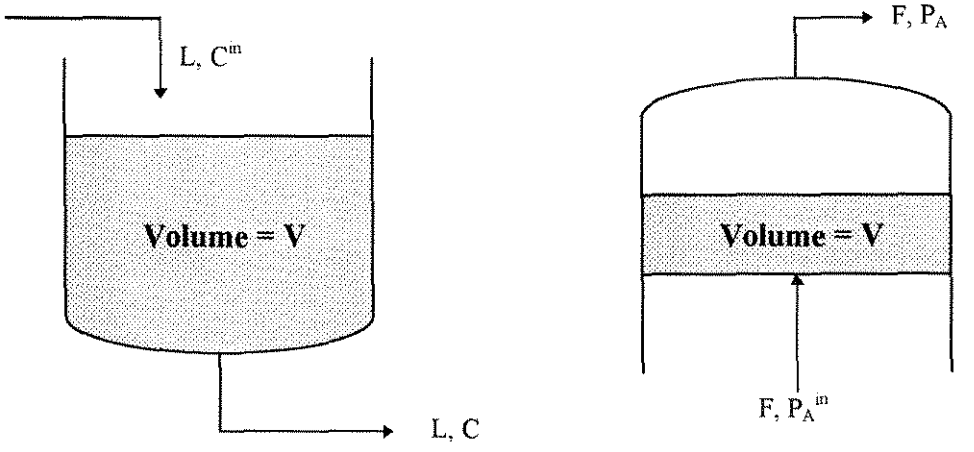

Figura 4.5 - Representação esquemática do fundo e topo de uma coluna.

Aplicando diferenças finitas nas derivadas das equações (4.48) e (4.49), a obtenção de expressões algébricas semelhantes a um filtro de 1ª ordem é direta.

4.6.4 - Resultados.

A seguir mostramos alguns resultados de simulações feitas a partir da modelagem acima descrita. Nestes podemos observar características importantes do processo e que são importantes no desenvolvimento e estudo de um sistema de controle.

A primeira característica a ser ressaltada é a alta não-linearidade do sistema. No gráfico (4.1) podemos observar que, para uma perturbação de 20% no valor da concentração de entrada do gás, a variação da saída é cerca de 11 vezes maior que para uma perturbação de 5% porcento na composição de entrada. Se o processo fosse linear, a variação na composição de entrada seria proporcional a variação na composição de saída.

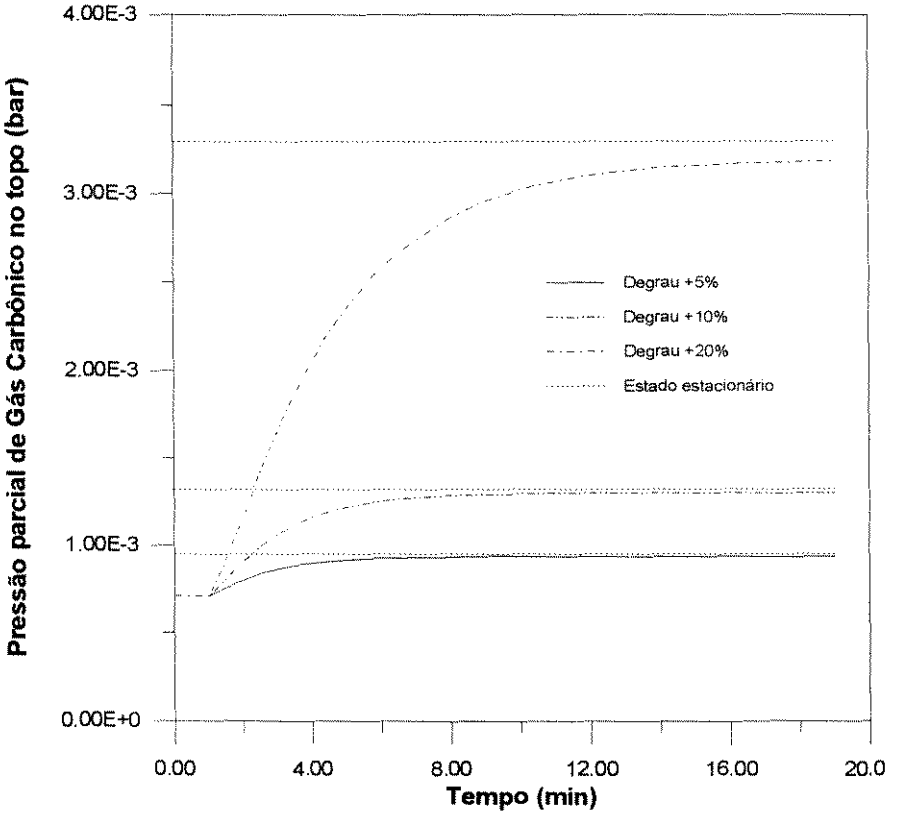

Gráfico 4.1 - Resposta da composição de topo a várias perturbações.

As linhas horizontais tracejadas no gráfico 4.1 mostram os estados estacionários gerados pelo programa EST.FOR. Podemos observar uma boa concordância entre os resultados obtidos pelo simulador dinâmico e o estacionário. De fato, a maior diferença percentual entre os resultados acima é de 3%. Este erro poderia ser reduzido utilizando-se um passo de integração menor. No entanto, o volume de cálculo aumenta bastante, pois, no método das características, o número de pontos na coluna em que os cálculos devem ser feitos é inversamente proporcional ao passo de integração no tempo.

A importância de se considerar o sistema como sendo distribuído pode ser visualizada no gráfico (4.2), onde duas perturbações simultâneas em degrau são

BIBLIOTECA CENTRAL

aplicadas no instante inicial na vazão de líquido e na concentração de CO_2 de entrada. As variações sucessivas que sofre a composição de topo antes de se estabilizar se deve à diferença de velocidade de propagação das perturbações. A velocidade do gás neste caso é cerca de cinco vezes maior que a velocidade do líquido. Desta forma, num instante inicial após a ocorrência das perturbações, a concentração de topo gás começa a aumentar, já que a vazão de líquido naquele ponto caiu. Como a velocidade do gás é bem maior que a do líquido, a perturbação nesta corrente rapidamente chega ao topo, causando uma brusca queda na concentração. A medida que a região com uma vazão menor de líquido se aproxima da base, aumenta a influência da fase líquida sobre a composição de saída, até chegar o ponto em que esta volta a aumentar, resultando num aumento da concentração, até que haja uma estabilização.

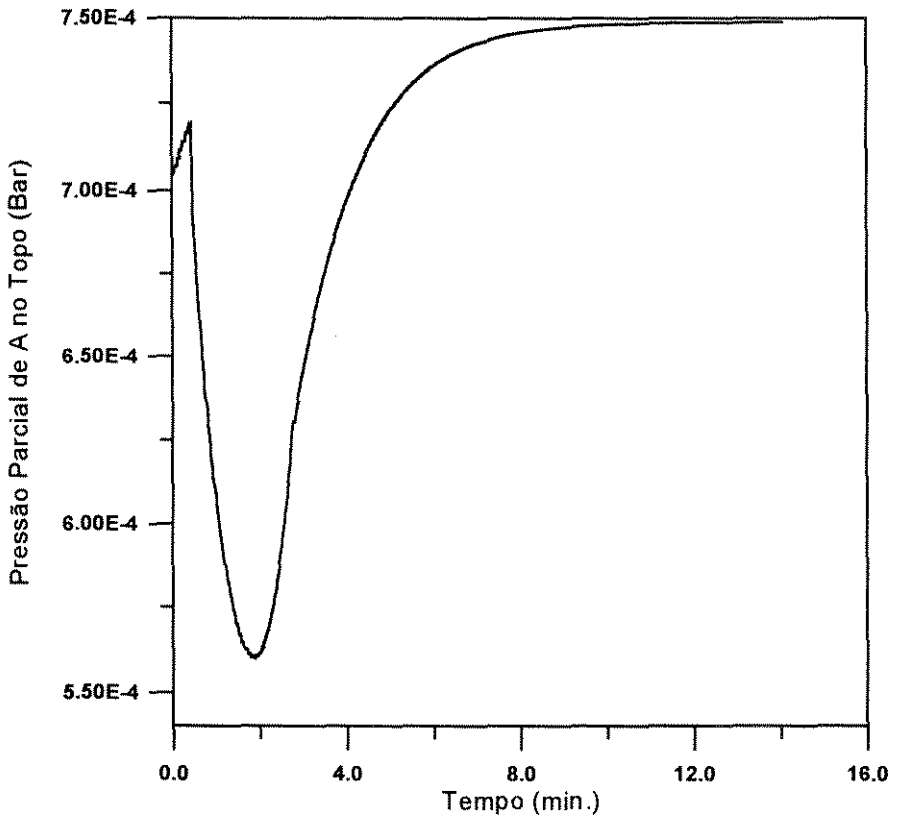

Figura 4.2 - Comportamento da composição de topo frente a duas perturbações simultâneas.

O gráfico 4.2 também nos dá uma importante informação quanto ao tempo de resposta do sistema a variações na fase gás e na fase líquida. Podemos observar que o tempo que o sistema leva para responder a uma variação na fase gás é bem menor que a uma variação na fase líquida. De fato, no exemplo acima, apenas num tempo em torno de 2 min. é que a redução da vazão de líquido começar a causar o aumento na concentração de saída do gás.

A forma como o Método das Características trata as perturbações pode ser visualizada através do gráfico (4.3). Neste, o perfil ao longo da coluna é mostrado no estado estacionário e após a perturbação em degrau negativo na composição do gás.

Pode-se verificar que a descontinuidade não influencia na região onde o gás, com uma nova composição, ainda não chegou.

Dos gráficos apresentados anteriormente, principalmente o gráfico (4.2), podemos concluir que, para colunas encontradas industrialmente, a consideração de que as variações nas entradas se propagam imediatamente, não é válida, já que o tempo de propagação destas ao longo da coluna pode ser bastante grande.

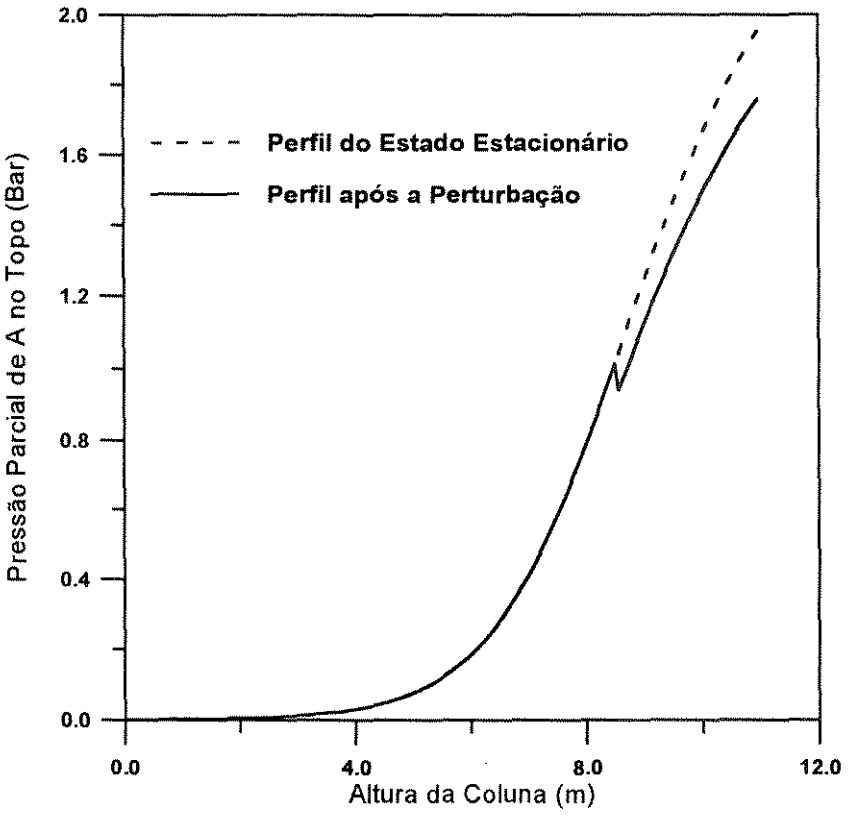

Gráfico 4.3 - Propagação de um perturbação ao longo da coluna.

O gráfico 4.4 mostra o comportamento do sistema frente a perturbações em degrau positivo na vazão líquido. Neste podemos observar que inicialmente a queda de concentração é lenta. A medida que a perturbação se aproxima da base, esta queda se torna mais acentuada. Isto é um observação importante para um bom entendimento do funcionamento do sistema de controle, como veremos no capítulo a seguir.

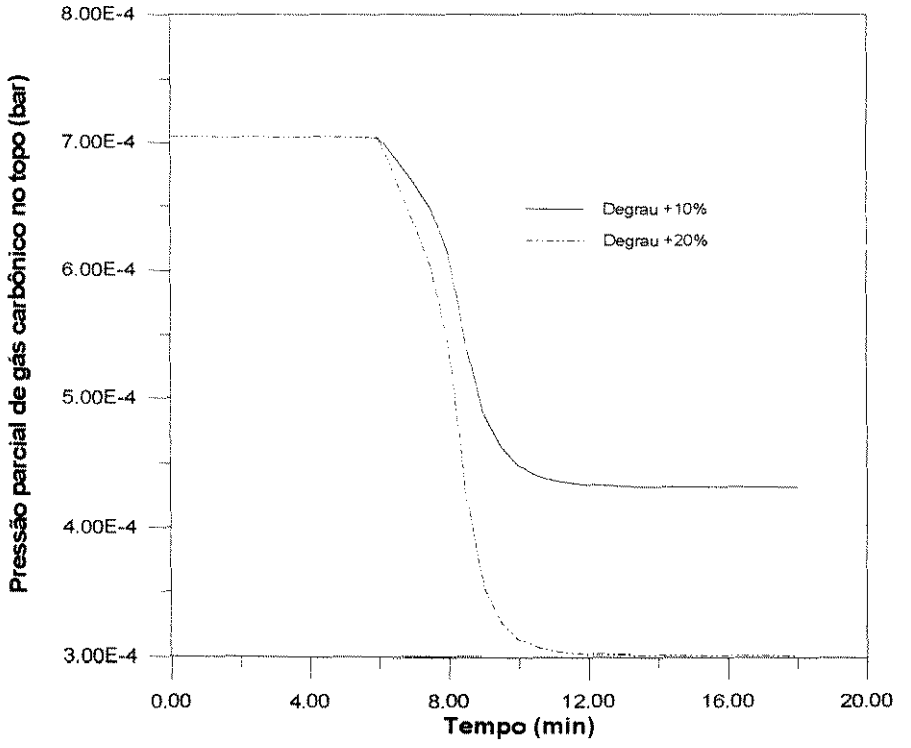

Gráfico 4.4 - Comportamento da composição de topo frente a perturbações em degrau na vazão de líquido.

CAPÍTULO 5

Controle Preditivo

CONTROLE PREDITIVO

No capítulo anterior viu-se que a resposta de colunas de absorção recheadas com reação química a perturbações é altamente não-linear, tanto pela presença da reação química, como pelo fato do processo ser distribuído.

Neste capítulo será analisado a importância de se utilizar uma estratégia de controle não-linear em lugar de uma linear para este sistema. O estudo será feito para o controle preditivo baseado em modelo, uma técnica avançada que tem sido aplicada com bastante êxito para diversos tipos de processo. Os modelos internos do controlador a serem testados são: o modelo baseado em Redes Neurais Artificiais (modelo não-linear) e o modelo de Convolução (linear).

5.1 - Motivação do estudo

Apesar da reação química aumentar bastante a taxa de absorção, a utilização deste tipo de processo em lugar da absorção puramente física apresenta algumas desvantagens. Dentre estas, as principais são: as altas taxas de corrosão que se tem no equipamento e o alto custo energético. A taxa de corrosão pode ser controlada a partir de aditivos ou pela limitação da concentração de solvente químico (Astarita *et al*, 1983). Já o consumo energético só pode ser minimizado a partir da redução da quantidade de solvente utilizada, já que a quantidade de utilidades (água de resfriamento e energia utilizada pela *stripper*) tem uma dependência direta desta.

Por outro lado, como vimos no capítulo anterior, este tipo de processo é utilizado principalmente quando se deseja manter a composição de topo em limites bem definidos, por limitação do próprio processo. Composições acima da especificação podem implicar em uma grande perda, como por exemplo o envenenamento de uma batelada de catalisador com a consequente parada de toda uma unidade.

A partir destas características do processo, o problema de controle pode ser colocado da seguinte forma: a composição final do gás deve ser mantida numa determinada especificação, utilizando para isto a menor quantidade de solvente possível.

Um outro aspecto do processo a ser levado em consideração diz respeito ao tempo de resposta do sistema. Como pudemos observar pelos gráficos de resposta da composição de topo a partir de perturbações na composição e fluxo da alimentação de gás, este apresenta um pequeno tempo morto após o qual a sistema varia rapidamente. Já a resposta da composição de topo a perturbações na fase líquida, embora seja sentida imediatamente, é mais lenta e só se torna mais significativa a partir do momento em que a perturbação se aproxima da base da coluna. Estas características sugerem que a forma mais eficiente de se controlar a coluna é utilizando um controle que se antecipe a modificação do processo frente a um determinada perturbação.

Desta forma, o controle preditivo é uma das técnicas mais indicadas para a utilização neste caso, tanto por atender as características acima descritas, como também pelo fato deste já ser largamente utilizado em escala industrial. Bequette (1991), numa revisão das diversas técnicas de controle não-linear, aponta que o controle preditivo é a estratégia que melhor consegue manusear as características gerais presentes nos diversos

processos químicos. Este autor ressalta ainda a capacidade desta técnica em manusear restrições do processo explicitamente.

A parte mais importante desta estratégia é o modelo interno do controlador, já que é com base nas predições deste que as ações de controle em instantes futuros são calculadas. Este modelo é quem vai diferenciar a estratégia linear da não-linear. O modelo de convolução, o qual é linear, foi o primeiro a ser utilizado, tendo sido empregado com sucesso mesmo para processo não-lineares. Uma técnica mais recente consiste em utilizar redes neurais artificiais. Do ponto de vista de aplicação, o modelo de convolução, como veremos mais a frente, é bem mais fácil de ser utilizado, requerendo um conhecimento menor do processo. No entanto, trabalhos que comparam os dois modelos (Willis *et al* (1992), Hoskins *et al*, 1992 e Morris *et al* (1994)) mostram que em geral o resultado obtido com redes neurais artificiais é bem superior.

O objetivo aqui será o de comparar os modelos tanto do ponto de vista de aplicação como do ponto de vista de resultados, de forma a dar bases para a escolha de um outra estratégia em sistemas onde a composição de saída do gás deve ser mantida em determinados limites.

5.2 - A estratégia de controle

O processo de absorção com reação química caracteriza-se como sendo de uma única saída, tendo em vista que o nosso objetivo é apenas manter a composição de topo no valor desejado.

Na definição das variáveis manipuladas, deve-se levar em consideração que estas devem ser tomadas da fase líquida, já que a fase gás é proveniente do processo anterior a coluna de absorção, sobre as quais o sistema de controle da absorvedora não pode interferir. Por outro lado, os processos de absorção com reação química, em geral, operam em sistema fechado (como mostrado na fig. 4.3), de forma que a única variável passível de manipulação é a vazão de líquido. Desta forma temos uma estratégia de controle com uma única entrada e uma única saída (estratégia SISO, *"Single Input, Single Output"*)

As restrições, quando se trata de colunas recheadas são extremamente importantes, tendo em vista a pequena faixa de operação em que este tipo de coluna pode operar (o que representa uma desvantagem em relação às colunas de pratos). Desta forma, temos que o desenvolvimento de sistemas de controle para estas deve levar sempre em consideração a presença de restrições intrínsecas do processo. As principais restrições dizem respeito ao ponto de arraste da coluna e ao grau mínimo de molhamento recomendado.

O ponto de arraste ocorre quando as vazões de gás e de líquido são grandes o suficiente para que o gás comece a arrastar o líquido para o topo da coluna. Diversas correções tem sido desenvolvidas para predizer o ponto de arraste de uma coluna. Aqui foi utilizada a correlação apresentada por Zens (1972):

$$\left(\frac{F'/\Omega}{10.86\sqrt{\rho_G \rho_L}}\sqrt{\frac{a_V}{\varepsilon^3}\mu_L^{0.2}}\right)^{1/3} + \left(\frac{L/\Omega}{2323}\sqrt{\frac{a_V}{\varepsilon^3}\mu_L^{0.2}}\right)^{1/2} = 18.91 \qquad (5.1)$$

onde as vazões de gás e de líquido devem estar em m^3/h e os outros valores em unidades do sistema internacional (SI).

O grau mínimo de molhamento refere-se a vazão mínima de líquido que deve circular pelo recheio de forma a não se ter a formação de áreas secas na coluna. O critério aqui adotado é o sugerido por Morris e Jackson (1953):

$$\frac{L}{a_v} \geq 0.8 m^3 / h \cdot m \tag{5.2}$$

Embora restrições de fluxo em válvulas ou de outros tipos pudessem ser adicionadas ao problema, estas duas são as mais importantes e as mais limitantes também, tendo em vista a pequena faixa operacional que as colunas recheadas podem operar. Desta forma, o problema de controle a ser aqui analisado consiste de um sistema SISO com restrições.

É importante ressaltar que tanto o modelo de convolução (quando utilizado junto com a otimização) como o modelo por RNA's permitem incluir também os distúrbios que atuam sobre o processo, ou seja, as variáveis que têm influência sobre o processo, mas sobre as quais não se tem controle. Os valores de entrada a serem passados para este modelo não são, desta forma, só as entradas do processo (variáveis controladas), mas sim toda variável que possa ter influência sobre este, o que pode tornar as predições muito mais precisas.

5.3 - Formulação matemática do Controle Preditivo

A idéia básica do controle preditivo com modelo é tratar o problema de controle como um problema de otimização, onde se busca minimizar a diferença entre a saída do processo e um valor desejado. Esta idéia não é nova, tendo sido primeiro apresentada por Zadeh e Whalen (1962). Na década de 70 as pesquisas neste campo se intensificaram, tendo a primeira aplicação sido descrita por Richalet *et al* (1978). Em 1979 engenheiros da Shell (Cutler e Ramaker (1979) e Prett e Gillette (1979)) apresentaram os princípios do DMC (controle por matriz dinâmica) e descreveram a aplicação deste a uma unidade de craqueamento catalítico.

Nos dois algoritmos de controle mencionados acima, a idéia básica era a mesma: utilizar um modelo explícito do processo que permitisse predizer as saídas do processo (variáveis controladas) a partir de determinadas entradas (variáveis manipuladas). As ações de controle são calculadas de forma a se ter as saídas as mais próximas possíveis do valor desejado. Desta forma, temos na verdade um problema de otimização onde se busca minimizar o erro de predição. A escolha de uma determinada função objetivo em detrimento de outra pode facilitar ou dificultar bastante o problema de otimização. A mais utilizada consiste em minimizar o quadrado das diferenças entre a variável de saída (predita por um modelo do controle) e uma trajetória desejada:

$$\min_{u(k),\cdots,u(+L-1)} J = \sum_{i=k+1}^{k+R} (y_i^d - y_i^{pred})^2 \tag{5.3}$$

onde R é o horizonte de predição, ou seja, o número de saídas futuras a serem preditas, e L é o horizonte de controle, que é o número de ações de controle futuras que se vai calcular.

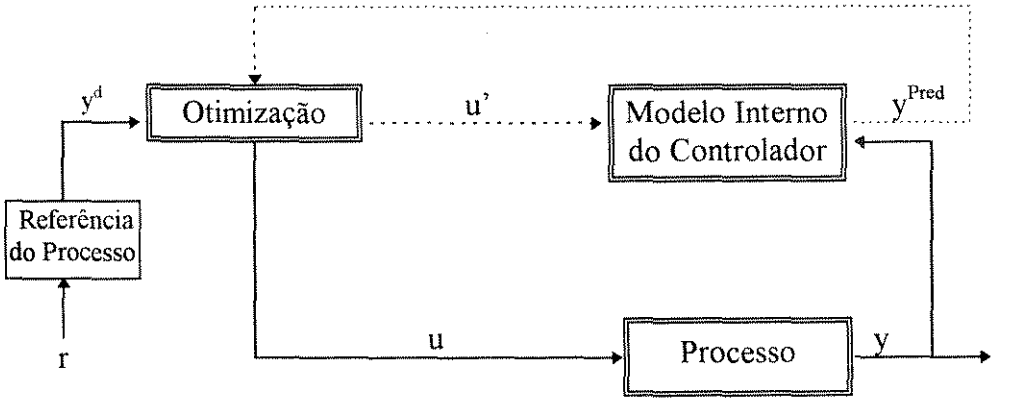

Figura 5.1 - Estrutura do Controle Preditivo.

A utilização da expressão anterior pode levar a ações de controle muito bruscas, já que o objetivo passado por esta é de simplesmente minimizar a diferença entre o valor predito e o valor dado por uma trajetória em determinados pontos. Assim, se houver um conjunto de ações que forcem os valores preditos a passarem exatamente sobre os valores desejados, mesmo que estes resultem em comportamento oscilatório e instável entre os instantes de amostragem, a resultado será aceito como o valor ótimo das ações de controle. Para evitar este problema, pode-se fixar uma variação máxima da ação de controle diretamente nas restrições do problema de otimização. No entanto, a utilização de restrições deste tipo pode fazer com que não haja uma solução factível para o problema de otimização em um determinado instante, o que representa um sério problema para o sistema de controle. Uma outra abordagem bastante utilizada e que elimina o problema anterior consiste em penalizar, na função objetivo, essas variações, através da adição de um segundo termo. Assim, a função objetivo do problema de controle pode ser escrita, da seguinte forma:

$$\min_{u(k),\cdots,u(+L-1)} J = \sum_{i=1}^{R} (y_{k+i}^{d} - y_{k+i}^{pred})^2 + \sum_{i=1}^{L} w_i \left(u_{k+i-1} - u_{k+i-2}\right)^2 \qquad (5.4)$$

Obviamente, quanto maior o valor dos pesos (w_i) utilizados na penalização das ações de controle na equação acima, mais suaves serão as ações de controles, o que pode fazer com que o sistema de controle demore muito para responder a uma determinada perturbação. Desta forma, o valores dos pesos devem ser o menor possíveis, desde que não se tenha oscilações. Note que neste caso não se estipulou um teto máximo de variação. Isto é muito importante, tendo em vista que o sistema deve responder de forma diferente para perturbações de magnitudes diferentes. Assim, a utilização de um valor máximo para a ação de controle poderia impedir o sistema de responder corretamente a uma perturbação de grande magnitude. Por outro lado, para perturbações de pequena magnitude, a ação de controle ainda poderia ser muito brusca, já que a variação máxima da ação de controle seria maior do que a realmente necessária.

O valor desejado (y_k^d) na equação acima poderia ser utilizado diretamente como sendo a saída desejado do processo ("*set point*"). No entanto, novamente para evitar que as ações de controle sejam muito bruscas, o valor utilizado não é diretamente o "*set*

point", mas sim uma valor compreendido entre este e a saída atual do processo, calculado utilizando a expressão de um filtro de primeira ordem:

$$y^{d}_{k+1} = \alpha y_k + (1-\alpha) r_k \tag{5.5}$$

Na expressão acima, α=0 implica em um valor desejado sem amortização. Para α=1, tem-se uma amortização máxima. Este é um parâmetro de sintonia do controlador.

Pela lei de controle acima, vê-se facilmente que a parte principal do controle preditivo está na capacidade de predizer o que irá ocorrer com o processo em instantes futuros. O valores preditos (y^{pred}_k) na equação acima poderiam ser obtidos diretamente de um modelo do processo. No entanto, caso o modelo não corresponda fielmente ao processo (o que geralmente ocorre), a utilização direta destes valores poderia resultar em um sistema de controle muito pouco robusto. Para evitar isto, o valor utilizado é corrigido pela seguinte expressão:

$$y^{pred}_{k+j} = y^{c}_{k+j} = \hat{y}_{k+j} + (y_k - \hat{y}_k) \tag{5.6}$$

ou seja, considera-se que a diferença entre o valor predito e o real no instante anterior seja válida para o instante atual.

Esta simples modificação torna o controlador bastante robusto, já que mesmo que seja utilizado um modelo interno do controlador não muito preciso, o sistema de controle, através de correções sucessivas dos desvios, acaba atingindo o valor desejado.

As constantes acima utilizadas na definição da função objetivo do controlador (horizonte de controle e horizonte de predição) são também parâmetros de sintonia que devem ser definidos pelo usuário.

O horizonte de controle define o número de ações de controle futuras que o otimizador deve calcular de forma a se ter a saída a mais próxima possível da desejada. Após a última ação de controle, a variável manipulada é considerada constante. Embora mais de uma ação possa ser calculada, apenas a primeira ação é implementada, tanto pelo fato de, no instante seguinte, as condições do processo já poderem ter se modificado, como pelo fato do modelo do processo não ser preciso, necessitando das correções já mencionadas a cada novo cálculo.

A partir dessas "L" ações de controle, o número de saídas do processo nos instantes futuros que poderiam ser levadas em consideração no somatório dos quadrados das diferenças na expressão (5.4) seria tão grande quanto se desejasse. No entanto, isto obviamente tornaria os cálculos muito extensos. Por outro lado, um número pequeno de predições futuras causaria uma perda de robustez, já que o controlador só conseguiria "enxergar" um curto intervalo de tempo a frente, podendo tomar ações de controle muito bruscas que satisfizessem a função objetivo naquele intervalo pequeno, mas que a longo prazo não conferissem bons resultados.

Por último, vale comentar sobre o tempo de amostragem do sistema. Este deve ser grande o suficiente para que se possa fazer os cálculos das ações de controle futuros, sem no entanto ser grande a ponto de se ter uma perda de informação do sistema. Em especial, no caso da absorção com reação química em torres recheadas, como se pode observar no capítulo anterior, a resposta do sistema a perturbações na fase gás possuem um retardo de tempo para serem sentidas na saída da coluna, após o qual a variação é grande. Isto sugere que um intervalo de amostragem inferior a esse tempo morto pode fazer com que o sistema de controle responda mais rápido e de forma mais eficiente a esse tipo de perturbação.

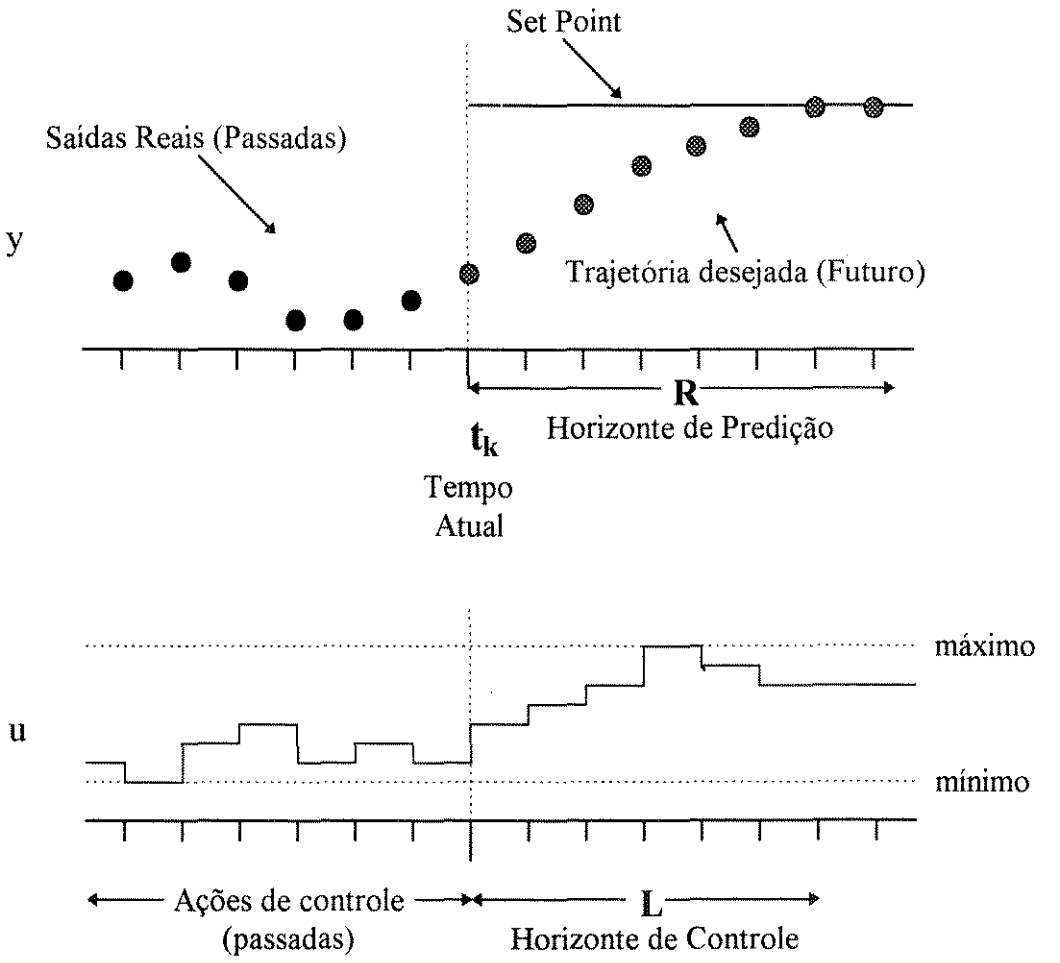

Figura 5.2 - O conceito de horizonte móvel no controle preditivo.

As características básicas de toda a família de controle preditivo baseado em modelos são as descritas aqui. Diversos tipos de controle surgiram na literatura a partir de modificações dessas idéias básicas. No entanto, em nenhum destes se observa uma mudança realmente significativa. Em linhas gerais, estas variações dizem respeito principalmente a formas de se obter uma solução analítica e de se tratar restrições. Dentre os diversos tipos de controladores desta família, podemos citar: DMC, QDMC, LDMC, MAC, IMC, DNC, etc.

O enorme avanço dos computadores digitais até hoje (e o que certamente ainda está por vir) sugere que seja muito mais interessante tratar o problema em sua forma essencial. A principal vantagem disso está no fato de se poder manusear as restrições explicitamente. Esta é a abordagem aqui utilizada. Nos resultados mostrados a seguir, a única modificação feita foi quanto ao modelo interno do controlador. A seguir falamos dos modelos aqui utilizados.

5.4 - O modelo baseado em RNA's

No capítulo referente aos métodos numéricos aqui utilizados, falou-se um pouco sobre as RNA's utilizando o conceito de janela de tempo e a sua utilização na simulação dinâmica de processo. Os fundamentos ali descritos são os mesmos que serão utilizados aqui no desenvolvimento de um modelo interno para o controlador baseado em redes neurais. Convém lembrar que esta não é a única abordagem utilizada em simulação dinâmica de processo por redes neurais. Outras abordagens como a utilização de redes recursivas ou outras formas de conseguir capturar a dinâmica de processos a partir das redes diretas também são utilizadas (Hunt *et al* (1992) e Narendra *et al* (1990)). Preferimos aqui utilizar as redes diretas com o conceito de janela de tempo móvel pela sua simplicidade e pelos bons resultados descritos na literatura que foram obtidos com esta para sistemas altamente não-lineares (Bath *et al* (1990) e Fileti (1995)).

A primeira etapa na identificação de um processo é a definição das variáveis que sejam relevantes para se conhecer o comportamento do mesmo e que sejam facilmente medidas ou estimadas. A utilização das redes neurais nessa identificação, tem a vantagem de se poder incluir os distúrbios (ao invés de só variáveis manipuladas e de saídas, como ocorre no modelo de convolução, descrito a seguir). Desta forma, pode-se passar um volume maior de informações sobre o processo, de forma a se ter uma melhor predição.

As variáveis utilizadas aqui como entradas para a rede neural foram:

(1) Vazão de alimentação de gás (F^{in});
(2) Composição de entrada do gás (P_{ab}^{in});
(3) Vazão de entrada do líquido (L^{in});

As variáveis de saída foram:

(1) Composição de saída do gás;
(2) Composição de saída dos produtos;

Note que a composição de saída dos produtos não é utilizada em nenhum momento na etapa de controle. No entanto, quando alguma perturbação na alimentação da fase gás acontece, a composição de topo do sistema demora um certo tempo morto para responder. Desta forma, a composição de saída não seria afetada, em um primeiro instante, pela perturbação na entrada de gás. A adição de uma das medidas de saída do líquido melhora pelo fato destas sentirem primeiro as variações na entrada do gás, dando uma informação extra sobre o sistema. A composição de saída dos produtos foi utilizada pelo fato desta ser facilmente mensurável - para uma determinada composição de amina - através do pH da solução (Dow Chemical, *apud* Bradley e Andre (1972)). Como está se utilizando um valor de saída do processo, esta foi utilizada como um valor de saída das redes neurais.

Todas as variáveis citadas acima são medidas por métodos simples, exceto a composição de componente absorvido na fase gás, que necessita de um método mais sofisticado, como por exemplo, a utilização de um aparelho de infravermelho em linha.

Definidas as variáveis a serem passadas para o modelo, a próxima etapa é a escolha do tamanho da janela de tempo para se conseguir uma boa descrição do processo. Deve-se observar que quanto maior o tamanho da janela, maior será o histórico de informações passadas para a rede neural. No entanto a medida que a janela

de tempo aumenta, maior será o número de pesos da rede neural, o que aumenta o tempo de treinamento de forma drástica. Por exemplo, se for utilizada uma janela de tempo de tamanho *n* (*n* instantes passados e *n* instantes futuros), com *ni* variáveis de entrada e *ns* variáveis de saída, o número de neurônios na 1ª e na última camada (ver figura 2.3) será, respectivamente, $2*ni*n+ns*n$ e de $ns*n$. Assim, se a janela de tempo tiver um tamanho 4, com um número de neurônios na camada intermediária igual ao da 1ª camada, teremos uma rede de configuração (32,32,8), que possui um total de 1.280 pesos. Caso a janela de tempo fosse de tamanho 5, o número total de pesos passaria para 2.000. Desta forma, deve-se utilizar a janela do menor tamanho possível, sem que isto comprometa os resultados.

A utilização de janelas muito grandes também não implica em uma boa predição, já que a medida que se inclui um número maior de dados passados para a rede, em princípio se faz necessário um número maior de neurônios para que se consiga reter essas informações. Assim, a rede pode aumentar de tal forma de proporção, que se torna mais viável utilizar uma de menores proporções.

O gráfico (5.1) mostra a utilização de uma RNA com janela de tempo na predição do próprio conjunto de dados utilizado no treinamento. Como podemos observar, as duas curvas praticamente se sobrepõem. De fato, o maior erro de predição neste caso é de 5% do valor real. No entanto, a rede foi utilizada da mesma forma com que foi treinada, ou seja, predizendo apenas os 3 instantes seguintes que a janela de tempo desta permite prever.

Se a rede for utilizada para predizer o mesmo conjunto de dados, sendo que agora de forma recursiva, obteremos os resultados do gráfico 5.2. Neste, apenas os primeiros dados de entrada foram utilizados para iniciar a predição e, a partir daí, os dados passados como entrada foram os anteriormente preditos pela RNA. Ou seja, a rede foi aqui utilizada da mesma forma como será utilizada no modelo interno do controlador.

O que se pode observar neste caso é que a predição é boa para uma faixa de dados iniciais. A partir de um certo ponto, os erros acumulados tornam-se grandes demais e os resultados obtidos são péssimos, oscilando entre o limite máximo e o mínimo para o qual a rede foi treinada. Além disto, mesmo na região em que os resultados são melhores, o erro é visivelmente maior que no caso anterior.

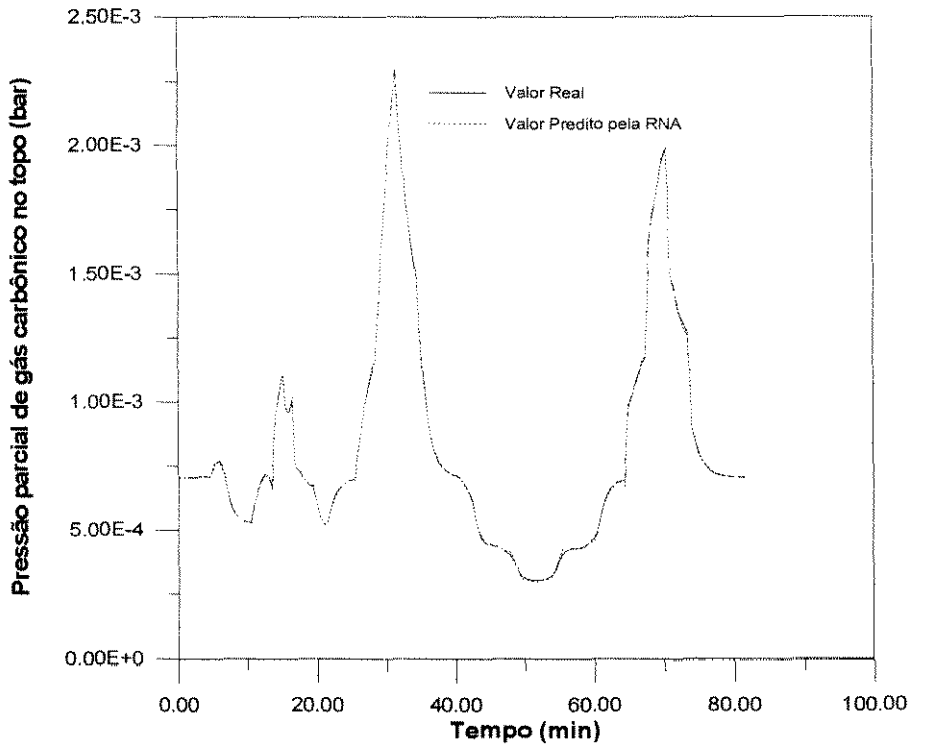

Gráfico 5.1 - Predição de uma RNA para o conjunto de dados com a qual foi treinada: a predição não é feita de forma recursiva.

Estes resultados, no entanto, não implicam que as RNA's não sirvam para serem utilizadas em sistemas de controle. O que ocorre aqui é que as variáveis de entrada passadas para a rede estão variando ao longo do tempo. Quando utilizadas no sistema de controles, estas variáveis irão variar um número de vezes bem inferior que os utilizados no conjunto de dados (definido pelo horizonte de controle). Desta forma, o acúmulo de erros esperados deve ser bem menor que neste caso. No entanto, a análise aqui feita serve para mostrar que um bom treinamento não implica em uma boa predição futura.

Como já discutimos no capítulo 2, para redes de grandes dimensões, o algoritmo Backpropagation (GDR) é muito mais indicado que o de Marquardt-Levenvenberg. No entanto, observou-se neste caso, que o treinamento de uma rede de menor porte utilizando Marquardt-Levennberg dava melhores resultados que o de uma rede maior utilizando o Backpropagation.

Desta forma, o treinamento da rede utilizada nos gráficos 5.1 e 5.2 foi feito com o algoritmo de Marquardt-Levenberg. A etapa de treinamento foi levada adiante até um total de 800 iterações, após o qual não se verificou uma redução significativa do erro na fase de treinamento.

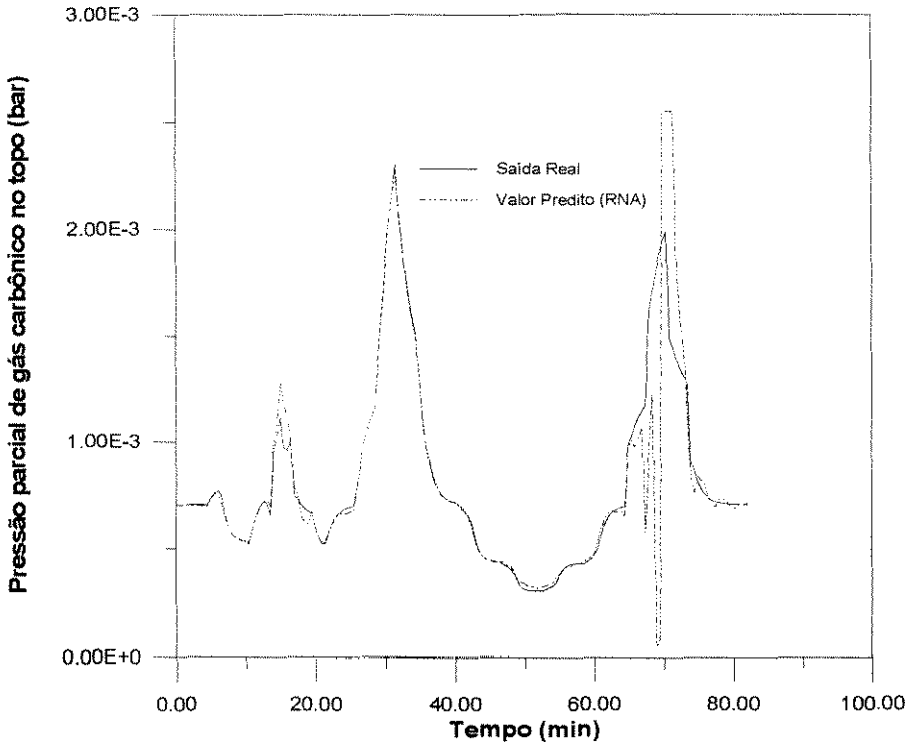

Gráfico 5.2 - Predição de uma RNA para o conjunto de dados com a qual foi treinada: a predição aqui é feita de forma recursiva.

O número de neurônios na camada intermediária foi fixado em 10. Este número foi adotado para que a rede não tivesse uma grande dimensão. O número de neurônios na camada de entrada e de saída da rede fica, então atrelado ao tamanho da janela de tempo. Testou-se os tamanhos de janelas de tempo descritos na tabela 5.1.

Tabela 5.1 - Janelas de tempos utilizadas e erros obtidos.

Tamanho da janela de tempo	Maior erro percentual	Arquitetura da RNA	Número total de pesos da RNA
2	7%	(16,10,4)	200
3	5%	(24,10,6)	300
4	10%	(32,10,8)	400

Não foram utilizadas janelas de tamanhos maiores pelo fato de se ter um aumento muito grande do número total de pesos da rede. A janela de tamanho 4 já implica num aumento muito grande do tempo total de treinamento em relação as outras duas. É bom lembrar que o algoritmo utilizado faz inversões de matrizes cujas dimensões são iguais ao número total de pesos. Assim, considerando-se que este é o passo mais lento no método, a grosso modo podemos dizer que o tempo gasto para realizar o mesmo número de iterações é proporcional ao quadrado do número de pesos.

Com base no que já foi discutido, os resultados da tabela 5.1 indicam que a janela de tamanho 2 não é suficiente para reter a dinâmica do sistema. Por outro lado, seria de

se esperar que uma janela de tamanho 4 desse melhores resultados. No entanto, a medida que se aumenta o número de neurônios nas camadas de entrada e saída, maior tem que ser, *a priori*, o número de neurônios na camada intermediária. Isto não foi tentando por já se ter obtido um bom resultado com a janela de tamanho 3.

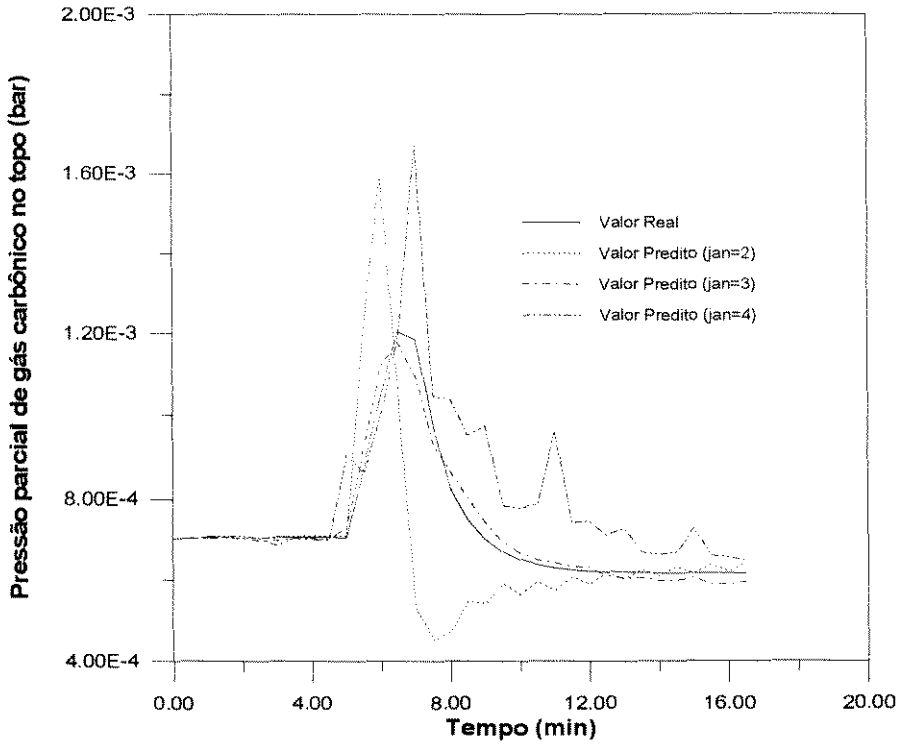

Gráfico 5.3 - Predição das RNA's com diferentes janelas de tempo para predição de dados que não foram utilizados no treinamento.

Convém lembrar que o resultado da tabela 5.1 não implica em um resultado bom. De fato, caso o resultado do gráfico 5.1 fosse feito com as três redes, não haveria uma melhora significativa. Para verificar se o resultado obtido com alguma destas é realmente bom, estas devem ser testadas com dados que não foram utilizados no conjunto de treinamento e de forma recursiva como será utilizada no sistema de controle. Uma comparação deste tipo encontra-se no gráfico 5.3. Neste podemos verificar que as redes com janela de tamanho 2 e 4 apresentam um resultado muito ruim. Por outro lado, o resultado obtido com uma janela de tempo de tamanho 3 foi muito bom.

O resultado irregular obtido no gráfico 5.3 pode parecer estranho a princípio. No entanto, deve-se observar que no processo de treinamento os ajustes são feitos de forma independente para cada um dos neurônios de saída, ou seja, para cada um dos valores de saída. Assim, cada neurônio vai predizer um valor com um erro diferente do outro, resultando em pontos que aparentemente não tenham uma continuidade, embora apresentem a mesma tendência de variação. Quando erro menores são obtidos na etapa de treinamento, a curva apresenta uma aparência mais suave (como podemos observar pela curva da janela de tamanho 3 no gráfico 5.3). No entanto, ainda assim persiste uma certa descontinuidade. Para melhorar a predição, durante a utilização das redes como modelo interno do controlador, as correções utilizadas pelo controle preditivo (equação

5.6) devem ser feitas de forma independente para cada um dos neurônios de saída da rede.

Um outro problema em se utilizar as redes neurais como modelo do controlador é que o número de dados requeridos para se efetuar a etapa de treinamento desta (para que se obtenha bons resultados na etapa de predição) deve ser suficientemente grande. Como, via de regra, quanto maior o número de dados de entrada, maior o número de neurônios para se conseguir convergência na etapa de treinamento, o número de neurônios na camada intermediária também pode ser bastante grande. Além disto, se os dados não cobrirem toda a faixa operacional, os resultados obtidos quando as condições saltarem fora desta faixa serão muito ruins, provavelmente piores do que os obtidos com um modelo linear, já que nestes casos a rede simplesmente prediz um valor próximo ao valor máximo ou mínimo que foi utilizado no conjunto de treinamento.

Embora o gráfico anterior mostre a existência de um desvio em relação ao valor real, deve-se lembrar que o próprio controle preditivo já supõe que está diferença irá existir e já coloca uma correção para esta. É claro que, quanto maior a esta diferença, pior será o resultado final de controle. A seguir apresentaremos as bases do modelo de convolução e, ao final, mostraremos o mesmo gráfico anterior utilizando o modelo linear para comparar os desempenhos.

5.5 - O Modelo de Convolução

A premissa básica no desenvolvimento do modelo de convolução é que o processo possa ser representado por um modelo linear. Assim, a partir de uma única resposta do sistema a uma perturbação em degrau unitário, é possível prever o comportamento deste frente a outras perturbações, já que estas serão simplesmente combinação linear da primeira.

Esta consideração torna o desenvolvimento deste modelo bastante simples, o que justifica o grande número de aplicações de controladores baseados no modelo de convolução (em especial, o DMC).

Suponha que a resposta de um dado sistema a uma perturbação em degrau unitário seja a mostrada na figura 5.3. Desta forma, a saída predita do sistema ($\hat{y}$) para uma perturbação na variável manipulada no instante inicial (Δm_0), é dada por:

$$\begin{aligned} \hat{y}_1 &= a_1 \Delta u_0 \\ \hat{y}_2 &= a_2 \Delta u_0 \\ &\vdots \\ \hat{y}_N &= a_N \Delta u_0 \end{aligned} \qquad (5.7)$$

onde N é o número de termos que deve ser utilizado para se conseguir reconstituir o sinal original (horizonte do modelo). Em geral se utiliza N suficiente para que se tenha a_N correspondente a 95-99% do valor do estado estacionário.

Caso a perturbação acima ocorresse no instante $1\Delta t$, ao invés do instante inicial, a resposta do sistema acima para esta seria dada por:

$$\begin{aligned}
\hat{y}_2 &= a_1 \Delta u_1 \\
\hat{y}_3 &= a_2 \Delta u_1 \\
&\vdots \\
\hat{y}_N &= a_{N-1} \Delta u_1
\end{aligned} \tag{5.8}$$

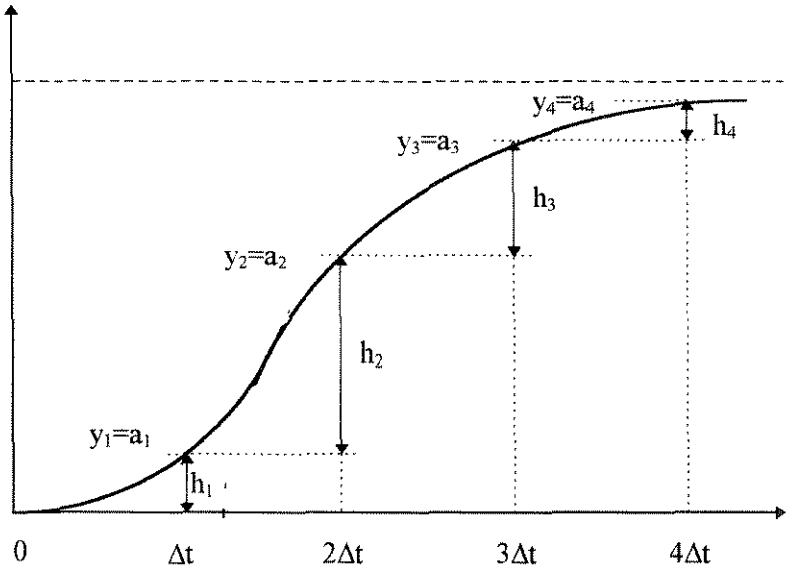

Figura 5.3 - Determinação dos coeficientes do modelo de convolução de um sistema a partir da resposta deste a um degrau unitário.

Caso as duas perturbações (Δm_0 e Δm_1) ocorressem no sistema, a resposta final seria a superposição das duas anteriores, ou seja:

$$\begin{aligned}
\hat{y}_1 &= a_1 \Delta u_0 \\
\hat{y}_2 &= a_2 \Delta u_0 + a_1 \Delta u_1 \\
&\vdots \\
\hat{y}_N &= a_N \Delta u_0 + a_{N-1} \Delta u_1
\end{aligned} \tag{5.9}$$

Generalizando a equação anterior para várias perturbações, e escrevendo-a de uma forma mais compacta, temos:

$$\hat{y}_{k+1} = \sum_{i=1}^{N} a_i \Delta u_{k+1-i} \tag{5.10}$$

A expressão anterior pode ser escrita de outra forma (conhecida como forma discreta) a partir dos coeficientes h's (figura 5.3):

$$h_i = a_i - a_{i-1} \tag{5.11}$$

Assim, expandindo o somatório da equação (5.10), e rearranjando os termos em função das variáveis manipuladas (e não mais das suas diferenças), obtemos:

$$\hat{y}_{k+1} = \sum_{i=1}^{N} h_i u_{k+1-i} \tag{5.12}$$

A mesma equação anterior, escrita para o instante "k", fica:

$$\hat{y}_k = \sum_{i=1}^{N} h_i u_{k-i} \tag{5.13}$$

Subtraindo as equações (5.12) e (5.13), obtemos:

$$\hat{y}_{k+1} = \hat{y}_k + \sum_{i=1}^{N} h_i \Delta u_{k+1-i} \tag{5.14}$$

A equação acima pode ser generalizada para um instante "k+j":

$$\hat{y}_{k+j} = \hat{y}_{k+j-1} + \sum_{i=1}^{N} h_i \Delta u_{k+j-i} \tag{5.15}$$

A partir da equação de correção dos erros (5.6), podemos rescrever a equação anterior da seguinte forma:

$$y^c_{k+j} = y^c_{k+j-1} + \sum_{i=1}^{N} h_i \Delta u_{k+j-i} \tag{5.16}$$

A equação anterior envolve ações de controle passadas e futuras. Estas podem ser separadas (o que é interessante, para fins de controle). Mostraremos um outro formato para a equação acima por indução. Assim, rescrevendo as equação (5.16) para j=1,2 e 3, e observando que, por definição, $y^c_k = y_k$, obtemos:

$$y^c_{k+1} = y_k + h_1 \Delta u_k + \sum_{i=2}^{N} h_i \Delta u_{k+1-i} \tag{5.17a}$$

$$y^c_{k+2} = y^c_{k+1} + h_1 \Delta u_{k+1} + h_2 \Delta u_k + \sum_{i=3}^{N} h_i \Delta u_{k+2-i} \tag{5.17b}$$

$$y^c_{k+3} = y^c_{k+2} + h_1 \Delta u_{k+2} + h_2 \Delta u_{k+1} + h_3 \Delta u_k + \sum_{i=4}^{N} h_i \Delta u_{k+3-i} \tag{5.17c}$$

Os somatórios na expressão acima só envolvem termos de ações de controle passadas. Podemos então definir uma nova variável, cujo valor é conhecido:

$$S_n = \sum_{i=n+1}^{N} h_i \Delta u_{k+n-i} \tag{5.18}$$

A partir desta definição, podemos rescrever as equações (5.17a-c) da seguinte forma:

$$y^c_{k+1} = y_k + h_1 \Delta u_k + S_1 \tag{5.19a}$$

$$y^c_{k+2} = y^c_{k+1} + h_1 \Delta u_{k+1} + h_2 \Delta u_k + S_2 \tag{5.19b}$$

$$y^c_{k+3} = y^c_{k+2} + h_1 \Delta u_{k+2} + h_2 \Delta u_{k+1} + h_3 \Delta u_k + S_3 \tag{5.19c}$$

Substituindo a equação (5.19a) em (5.19b) e esta última em (5.19c), obtemos:

$$y^c_{k+1} = y_k + h_1 \Delta u_k + S_1 \tag{5.20a}$$

$$y^c_{k+2} = y_k + h_1 \Delta u_{k+1} + (h_1 + h_2)\Delta u_k + S_1 + S_2 \tag{5.20b}$$

$$y^c_{k+3} = y_k + h_1 \Delta u_{k+2} + (h_1 + h_2)\Delta u_{k+1} + (h_1 + h_2 + h_3)\Delta u_k + S_1 + S_2 + S_3 \tag{5.20c}$$

Neste ponto, por indução, a generalização das equações acima para qualquer instante é direta. Para fins de simplificação, pode-se ainda fazer as seguintes substituições:

$$P_i = \sum_{n=1}^{i} S_n \tag{5.21}$$

$$a_i = \sum_{n=1}^{i} h_n \tag{5.22}$$

onde a equação (5.22) vem das definições utilizadas no desenvolvimento do modelo de convolução (figura 5.2).

Rescrevendo as equações (5.20a-c), obtemos:

$$y^c_{k+1} = y_k + a_1\Delta u_k + P_1 \tag{5.23a}$$

$$y^c_{k+2} = y_k + a_1\Delta u_{k+1} + a_2\Delta u_k + P_2 \tag{5.23b}$$

$$y^c_{k+3} = y_k + a_1\Delta u_{k+2} + a_2\Delta u_{k+1} + a_3\Delta u_k + P_3 \tag{5.23c}$$

Ou, de uma forma generalizada:

$$y^c_{k+j} = y_k + \sum_{i=1}^{j} a_i\Delta u_{k+j-i} + P_j \tag{5.24}$$

A equação anterior permite calcular o valor predito corrigido a qualquer instante "k+j" para um sistema SISO. Deve-se observar que na função objetivo de controle, dada pela equação (5.4), "j" deve variar de 1 a R, o horizonte de predição. No entanto, para instante posteriores ao horizonte de controle (j>L-1), a variável manipulada é mantida constante, ou seja, $\Delta u=0$.

Embora o modelo de convolução anterior tenha sido desenvolvido para um sistema SISO, a generalização deste para sistemas MISO (várias entradas e uma única saída) ou MIMO (várias entradas e várias saídas) é direta. Embora o sistema aqui utilizado seja SISO, o modelo pode ser utilizado de forma a incorporar também alguns variáveis de distúrbios do processo, da mesma forma como foi feito com o modelo por redes neurais. Isto só é possível por não estar se utilizando a solução analítica do problema de controle, e sim fazendo a otimização a cada tempo de amostragem. Caso fosse utilizado o DMC, por exemplo, o modelo de convolução apenas poderia incluir as variáveis manipuladas e controladas. Neste caso, no entanto, não se observou vantagem em se utilizar o modelo MISO, como podemos observar pelo gráfico 5.4. Concluímos então que a interação entre as variáveis que causam modificação no processo é muito não linear. Assim, preferiu-se utilizar o modelo na sua forma SISO, que reduz bastante o volume de cálculo.

Para calcular a saída do sistema MISO, bastar utilizar a superposição das saídas devida a cada uma das variáveis passadas como entradas do modelo.

A equação (5.24), juntamente com a definição de P_i e de S_i, constituem o modelo de convolução e permitem prever, a partir de um única resposta do sistema a uma perturbação degrau unitário (dada pelos coeficientes a_i), o comportamento do sistema a qualquer instante.

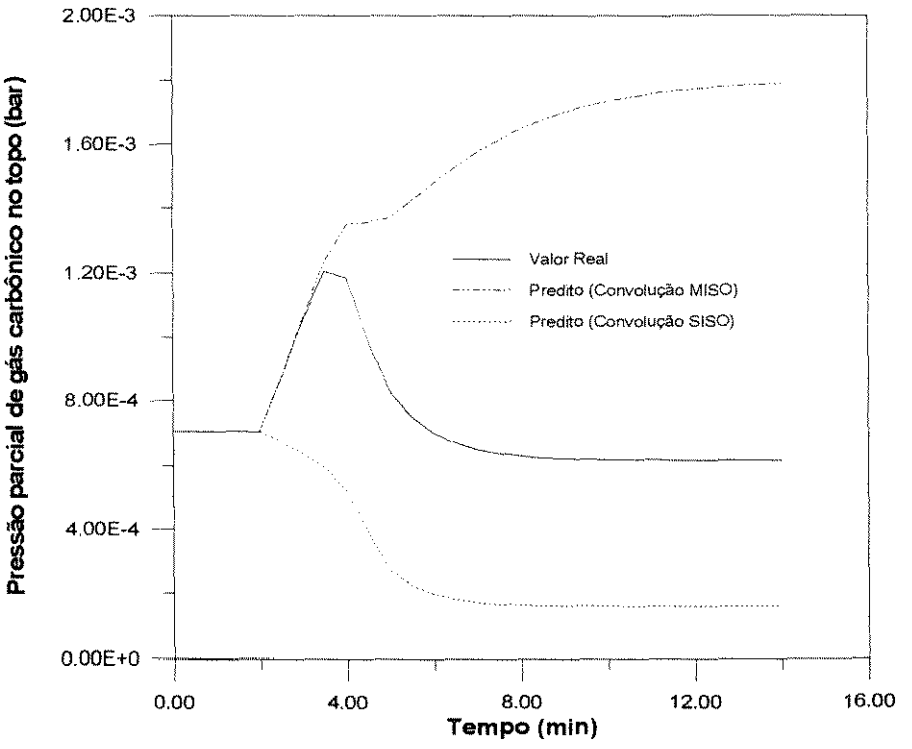

Gráfico 5.4 - Predição do modelo de convolução na forma SISO e MISO.

Perturbações unitárias podem ser difíceis de se obter em determinados sistemas, de forma que uma perturbação de qualquer magnitude pode ser utilizada, desde que os coeficientes sejam devidamente corrigidos. Deve-se observar que, pelo fato deste modelo ser linear, a escolha da perturbação deve ser feita em torno de um ponto intermediário da condição em que se vai operar.

5.6 - Resultados

Todos os resultados mostrados a seguir foram obtidos a partir da utilização de um dos modelos descritos anteriormente juntamente com uma sub-rotina de otimização. Para problemas de controle MIMO de grande porte, a escolha de um método de otimização que se adeque ao problema é essencial para que se tenha um mínimo de esforço computacional, já que o tempo de processamento pode ser um ponto limitante na utilização do controle preditivo. Neste caso, como o sistema em estudo é SISO, o tempo computacional foi pequeno (em torno de 2 seg. para o sistema de controle com os parâmetros ajustados, resultado obtido em computador PC Pentium 100MHz, com 16Mb de memória RAM). Desta forma, utilizamos para ambos os modelos internos do controlador, o método SQP (programação quadrática sucessiva), que resolve problemas de otimização de funções não-lineares sujeitas a restrições, embora o problema de controle com o modelo de convolução pudesse ser resolvido por um método QP (programação quadrática), já que este modelo é linear. A sub-rotina utilizada foi a E04UCF da NAG (1991).

A comparação entre os resultados foi feita inicialmente com base no critério da integral do erro absoluto (IEA), expresso por:

$$IEA = \int_0^\infty |e(t)|dt \tag{5.25}$$

onde e(t) é a diferença entre o valor desejado ("*set point*") e a saída do processo. Este critério foi escolhido por ser o mais amplamente utilizado na literatura. Embora o IEA não dê informações sobre algumas características importantes (como oscilações e sobre-elevação), este serve como base para a seleção inicial de um conjunto de parâmetros que confiram melhores resultados. Logo após, mostraremos os gráficos das saídas do processo, de forma a complementar a informação não coberta por este critério.

Os parâmetros aqui utilizados na sintonia do controlador foram: o horizonte de controle (L), o horizonte de predição (R) e os pesos para a penalização de ações de controle bruscas (w_i), que aqui consideraremos todos iguais (*f*). Preferimos utilizar estas penalizações em lugar de adicionar diretamente restrições ao problema de otimização pelo fato das últimas terem um valor fixo, que independe da magnitude das perturbações que ocorrem no sistema. Já as penalizações pressupõem um compromisso entre a minimização do erro e uma ação de controle não muito brusca.

Seborg *et al* (1989) sugere que estes parâmetros sejam suficientes para se conseguir uma boa sintonia do controlador, não sendo necessário incluir o tempo de amostragem como mais um parâmetro. Por este motivo, utilizamos este constante (Δt=0,5min) em todos os casos. Este valor é pequeno o suficiente para que se consiga descrever as características básicas do processo.

O filtro de 1ª ordem utilizado na determinação da trajetória desejada (equação (5.5)) não foi aqui incluído, já que este visa evitar ações de controle bruscas, função que já é desempenhada pelas penalizações acima citadas. Além disto, nem todos os autores o inclui na definição do problema de controle.

Procurou-se reduzir o número de parâmetros necessários para a sintonia tanto pelo fato dos aqui escolhidos serem os mais utilizados nos estudos existentes na literatura, como pelo fato de se ter uma quantidade muito grande de opções possíveis com todos estes, o que tornaria a visualização do efeito de cada um sobre os sistema de controle muito difícil.

No caso do modelo de convolução ainda é necessário definir mais um parâmetro: o horizonte do modelo (N), já discutido anteriormente. Adotou-se aqui N=25, que corresponde a um tempo onde 99% do valor de saída no estado estacionário já foi alcançado. Vale lembrar que o horizonte de predição (R) não pode ser superior ao horizonte do modelo.

A seguir discutiremos sobre a sintonia do controlador para ambos os modelos aqui utilizados.

5.6.1 - Sintonia do Controle Preditivo com Modelo de Convolução

Os dados mostrados a seguir foram gerados a partir de uma perturbação em degrau de +10% na composição de CO_2 do gás alimentado à coluna.

Via de regra, com já discutimos anteriormente, valores altos do horizonte de predição e baixos do horizonte de controle torna o sistema de controle mais robusto. Esta regra foi seguida nos resultados obtidos para o modelo de convolução. Os gráficos 5.5, 5.6 e 5.7 a seguir mostram que os melhores resultados são sempre obtidos com L=1. Por outro os maiores valores do horizonte de controle sempre resultaram em um IEA mais baixo.

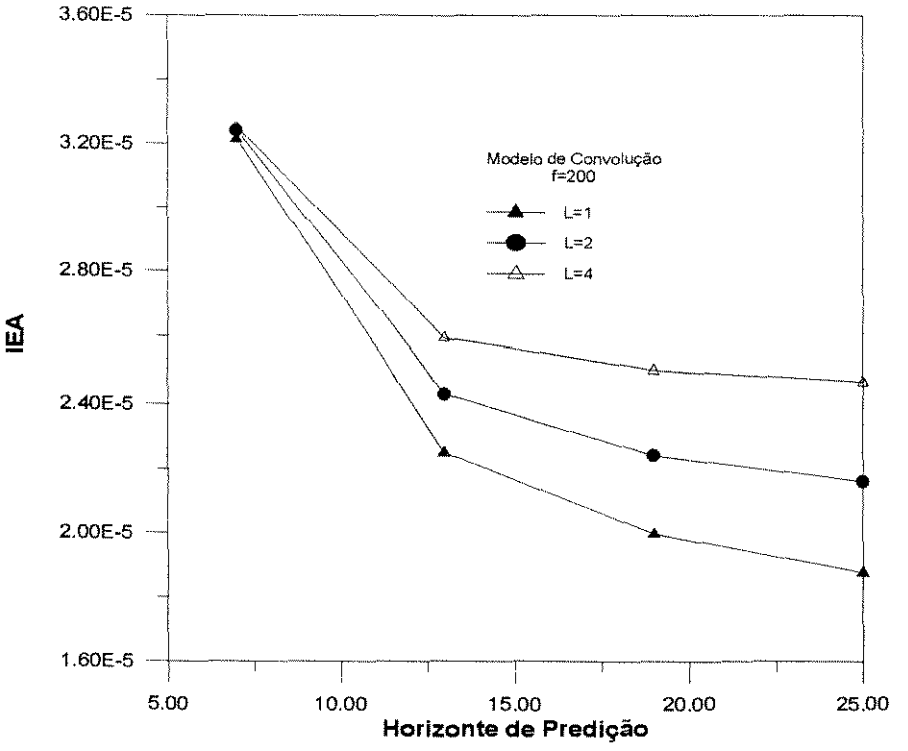

Gráfico 5.5 - Valores de IEA obtidos com o modelo de convolução para f=200, variando os valores de L e R.

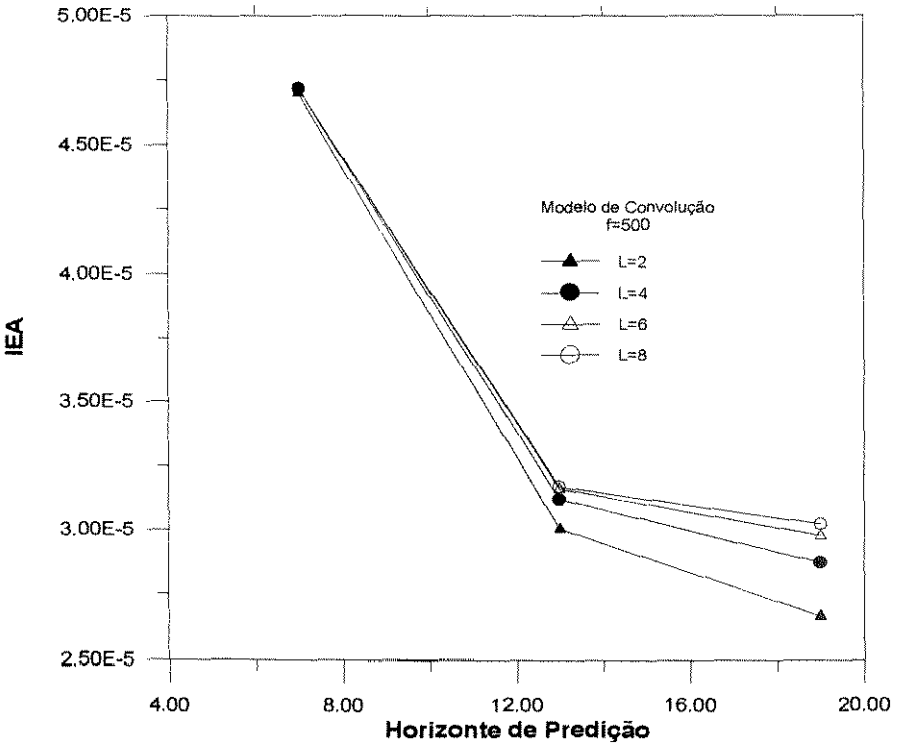

Gráfico 5.6 - Valores de IEA obtidos com o modelo de convolução para f=500, variando os valores de L e R.

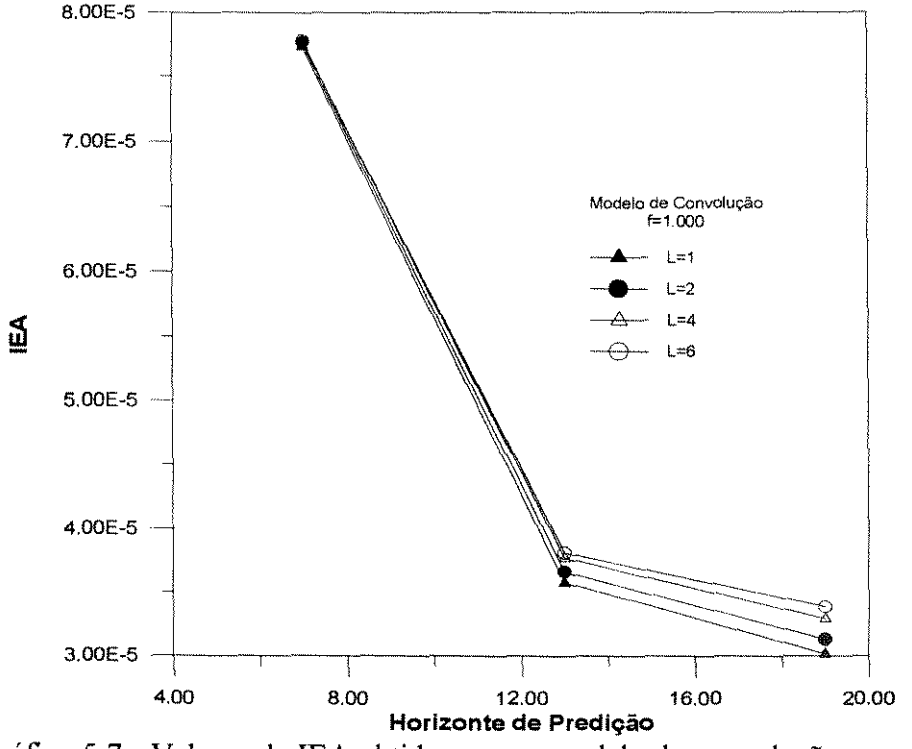

Gráfico 5.7 - Valores de IEA obtidos com o modelo de convolução para f=1.000, variando os valores de L e R.

Já o parâmetro de amortização do controle (*f*) não apresenta nenhuma regra heurística para a sua determinação. Pode-se observar que os maiores valores causam os piores resultados. Além disto, para valores muito grandes de *f*, o sistema de controle fica mais insensível a variações no horizonte de controle do sistema. Neste caso, no entanto,

a simples análise a partir do IEA não implica em uma melhora real, já que valores muito baixos de f podem implicar em uma excessiva oscilação na resposta do sistema, o que pode ser indesejável em alguns casos.

No gráfico 5.8 podemos observar um curioso comportamento do sistema para $f = 0$. Ao contrário do que ocorreu nos resultados anteriores, há um aumento da IEA a medida que se aumenta o horizonte de predição.

Quando há amortização muito grande, as ações de controle tendem a ser mais suaves do que o necessário. Assim, a medida que se aumenta o horizonte de predição, o número de pontos em que o erro deve ser minimizado aumenta, crescendo também a importância relativa deste termo na função objetivo definida pela equação (5.4). Esta tendência de uma maior variação do IEA com o horizonte de predição pode ser observado pelos gráficos 5.5 a 5.7. No entanto, quando a amortização torna-se pequena, o horizonte de controle passa a atuar de forma contrária, pois agora o otimizador está predizendo ações bruscas demais para o sistema. Assim, a utilização de um horizonte maior tende a amortizar as ações mais do que o necessário (já que o modelo apresenta imprecisões), aumentando o IEA. Neste caso, horizontes de tempo muito pequenos podem causar oscilações, pois não haverá nada amortizando as ações de controle. O gráfico 5.9 ilustra bem isso. Simulações feitas com R<7, mostraram que as oscilações tornam-se excessivamente grande, chegando a causar instabilidade.

O gráfico 5.9 mostra que para L=1, não há uma grande diferença nas repostas do sistema para f=0, f=100 e f=200. No entanto há uma pequena melhora, tanto em termos da sobre-elevação do sistema, como para o IEA, nos casos em que f=0. Como a resposta para R=7 teve uma certa tendência a ser mais oscilatória, preferimos utilizar R=13. Assim, consideraremos que o conjunto de parâmetros que confere melhores resultados para o modelo de convolução é: f=0, L=1, R=13. Deve-se observar que este conjunto apresenta por um lado uma amortização pequena, o que lhe permite tomar ações de controle mais bruscas, e por outro lado, um pequeno horizonte de controle, que restringe as ações de controle, já que o otimizador não pode corrigir uma ação brusca a partir das disponíveis nos casos em que L>1.

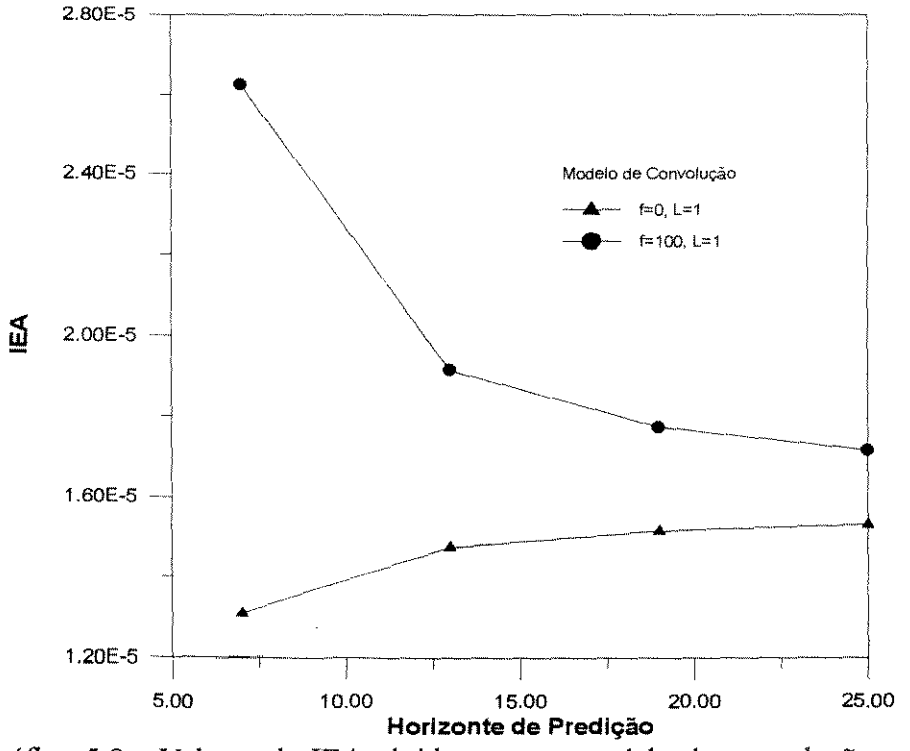

Gráfico 5.8 - Valores de IEA obtidos com o modelo de convolução para f=0 e f=100, variando-se R.

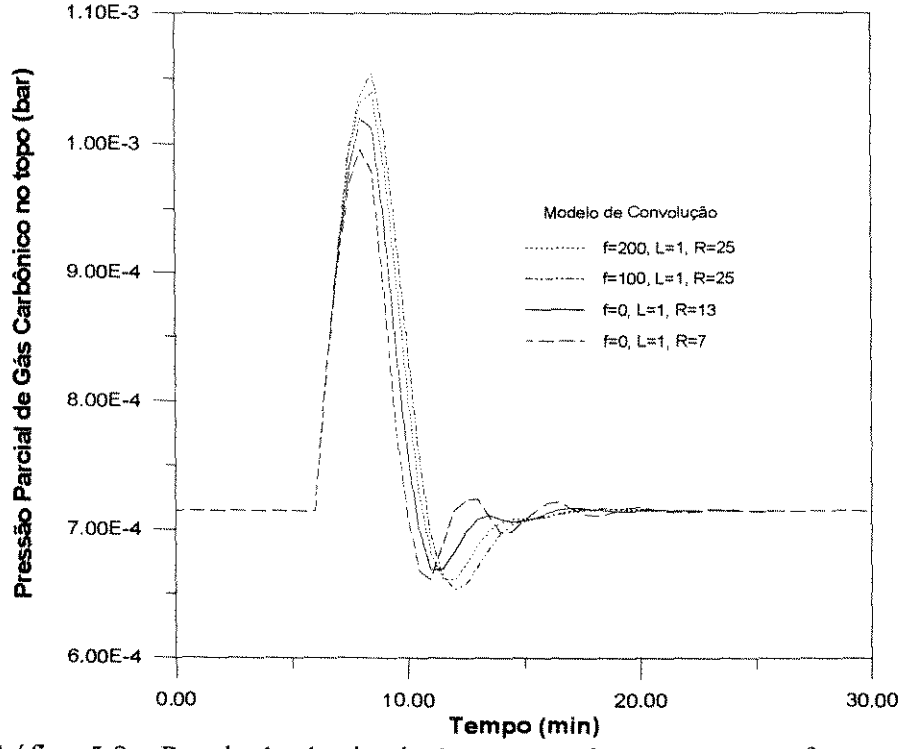

Gráfico 5.9 - Resultado da simulação para parâmetros que conferem uma maior estabilidade ao sistema de controle.

5.6.2 - Sintonia do Controle Preditivo com Modelo baseado em RNA

O modelo preditivo com modelo baseado em redes neurais mostrou um comportamento semelhante ao de convolução quanto a sintonia dos parâmetros. A mesma tendência observadas nos gráficos 5.6 e 5.7 podem ser observadas nos equivalentes 5.11 e 5.12. No entanto, este se mostrou um pouco mais sensível ao parâmetro de amortização, pois o aumento do IEA com o horizonte de controle ocorre já para f=200, embora somente para valores maiores de L (como mostra o gráfico 5.10). Novamente os menores valores de IEA foram obtidos com $f = 0$, como mostra o gráfico (5.13)

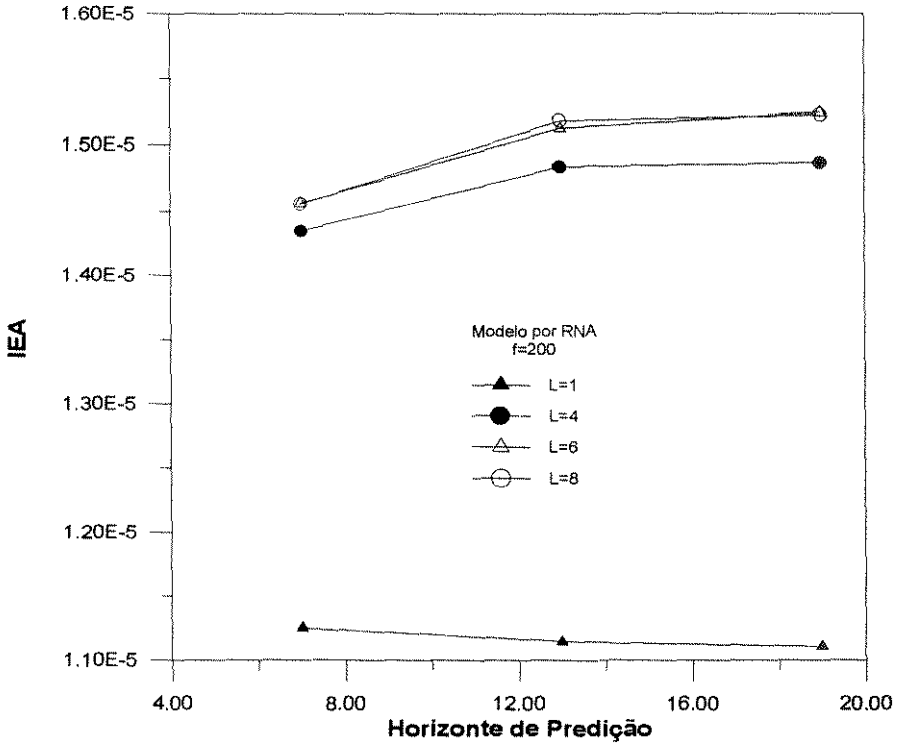

Gráfico 5.10 - Valores de IEA obtidos com o modelo baseado em RNA's para f=200, variando-se R e L.

O gráfico 5.14 mostra o resultado de simulações utilizando diferentes fatores de amortização. Podemos observar que quando valores menores são utilizados, as oscilações são grandes. No entanto, não há um ganho significativo com relação a sobre-elevação do sistema, sendo que os picos de concentração estão muito próximos uns dos outros. Desta forma, preferimos utilizar aqui f=200, L=1 e R=19, por estar numa faixa intermediária em relação as outras duas curvas.

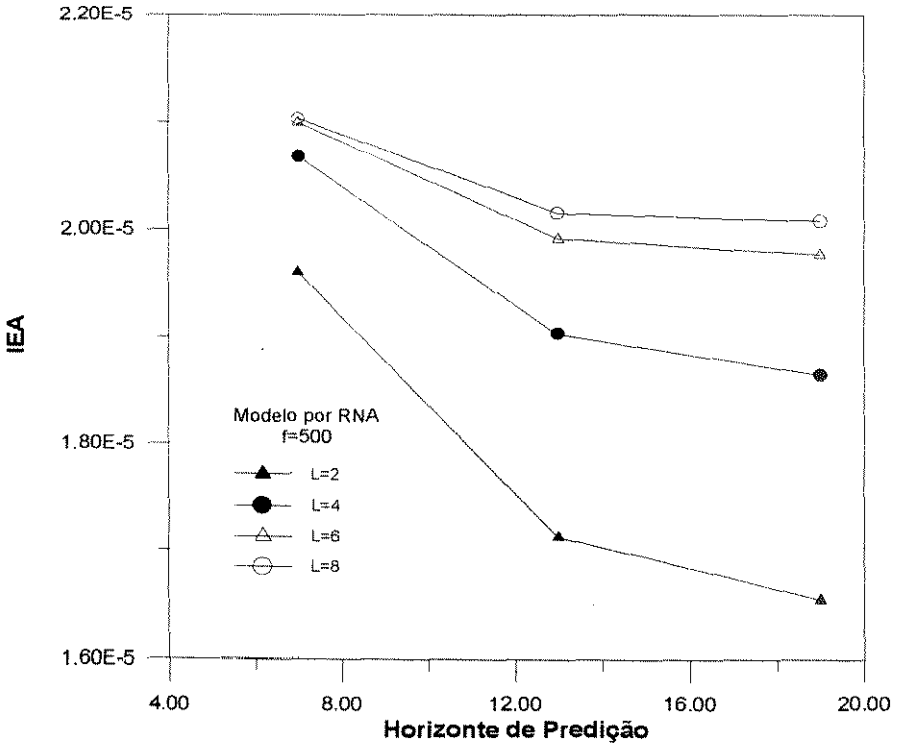

Gráfico 5.11 - Valores de IEA obtidos com o modelo baseado em RNA's para f=500, variando-se R e L.

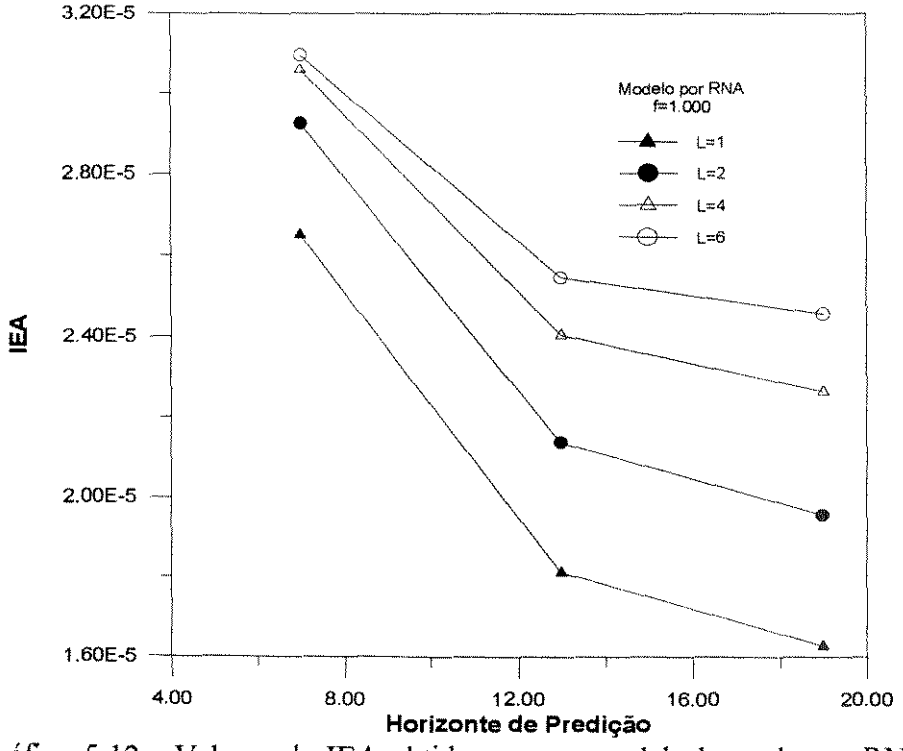

Gráfico 5.12 - Valores de IEA obtidos com o modelo baseado em RNA's para f=1.000, variando-se R e L.

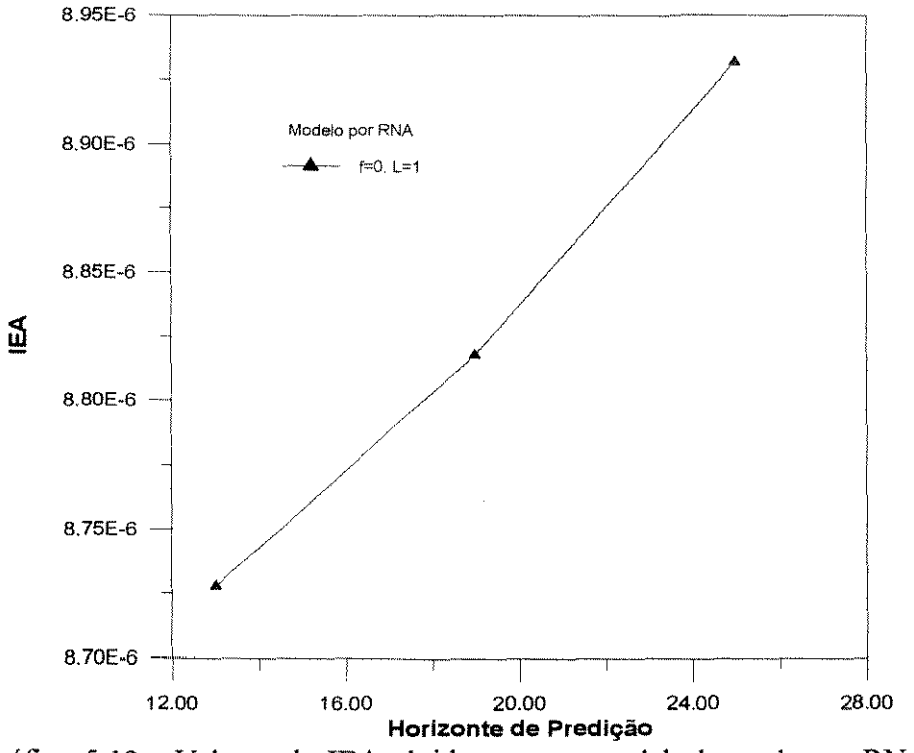

Gráfico 5.13 - Valores de IEA obtidos com o modelo baseado em RNA's para f=0 e L=1, variando-se R.

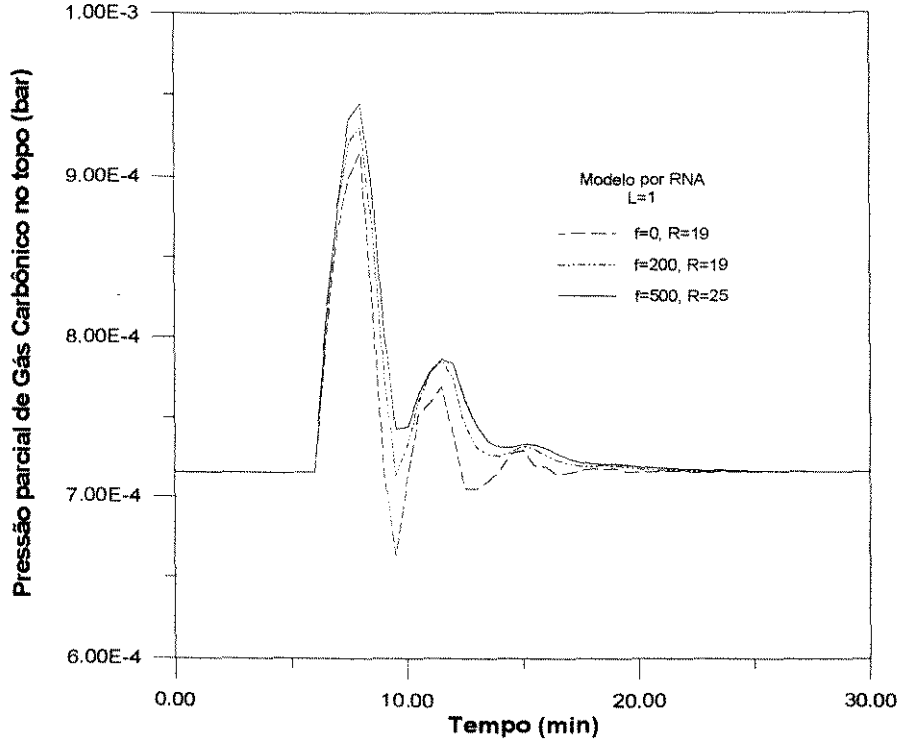

Gráfico 5.14 - Resultado da simulação para parâmetros que conferem melhores resultados.

5.7 - Comparação entre os resultados obtidos com modelo baseado em RNA e com o modelo de Convolução

Plotando os resultados obtidos com ambos os sistemas de controle para uma perturbação em degrau de +10% na composição de entrada do gás, obtemos o gráfico 5.15. Neste, podemos observar que a sobre-elevação do controlador utilizando o modelo de RNA's foi cerca de 70% da obtida com a utilização do modelo de convolução.

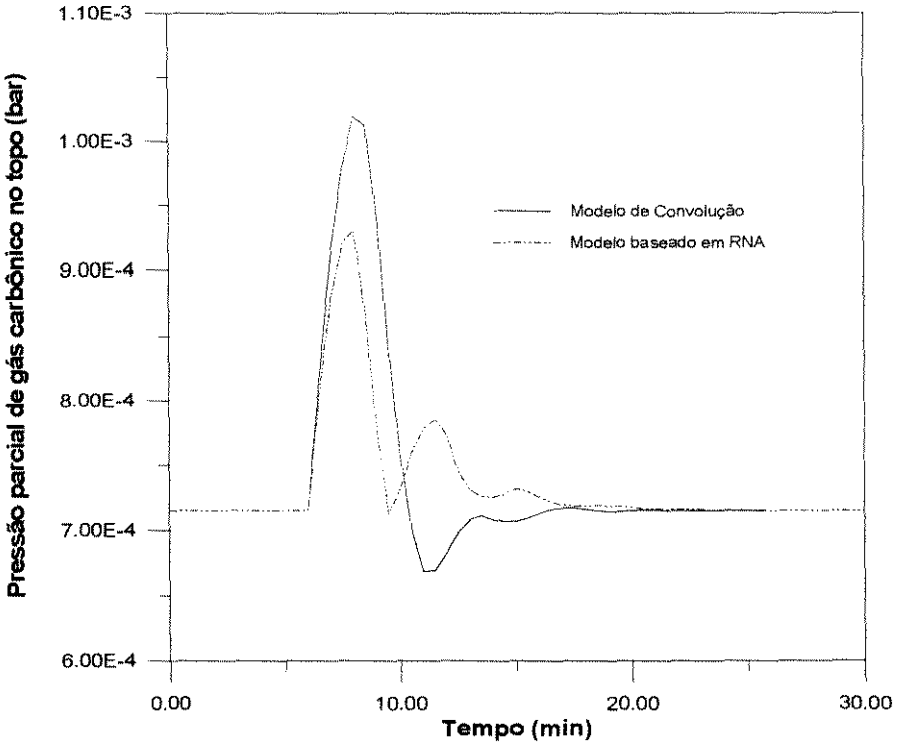

Gráfico 5.15 - Resposta do sistema para uma perturbação em degrau de +10% na composição de entrada do gás.

O resultado anterior parece realmente decepcionante, já que a diferença de precisão entre a predição dos dois modelos é muito grande (como vimos nas seções anteriores). No entanto, deve-se lembrar que, como foi discutido no capítulo anterior, a resposta da coluna a variações na fase líquida é bem mais lenta que a resposta a variações na fase gás. Isto faz com que o sistema tenha uma inércia grande e, desta forma, as variações na fase líquida somente são sentidas algum tempo depois. Concluímos então que a resposta de um processo deste tipo, quando perturbado, sempre irá apresentar uma certa sobre-elevação.

A seguir apresentamos mais alguns gráficos que mostram o comportamento do sistema frente a perturbações tanto na alimentação de gás como na composição deste.

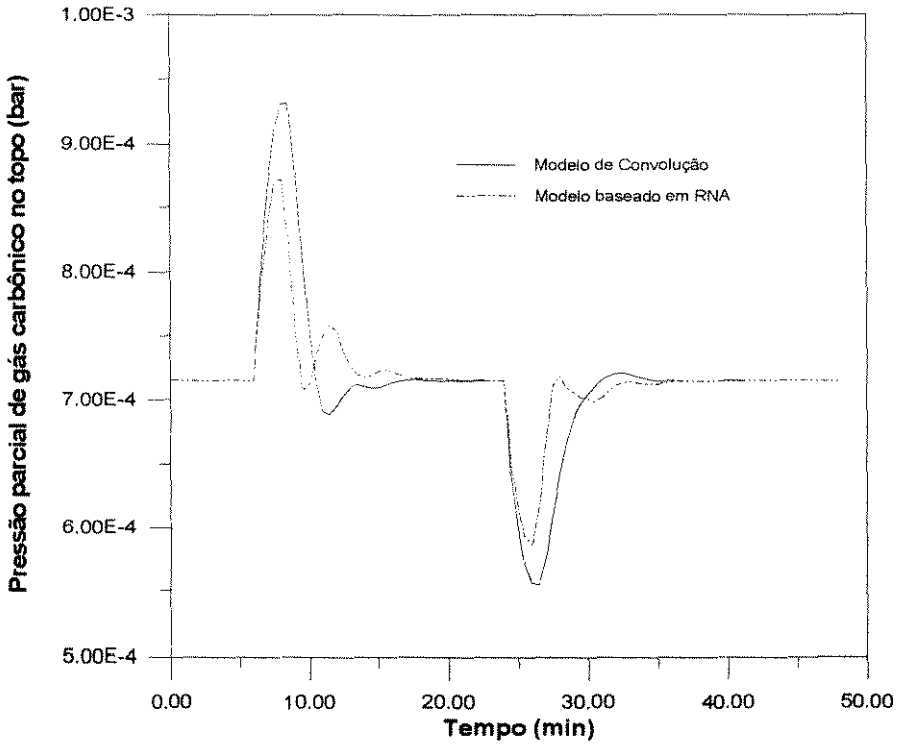

Gráfico 5.16 - Resposta do sistema para duas perturbações em degrau na composição de entrada do gás: +15% em t=6min e -15% em t=24min.

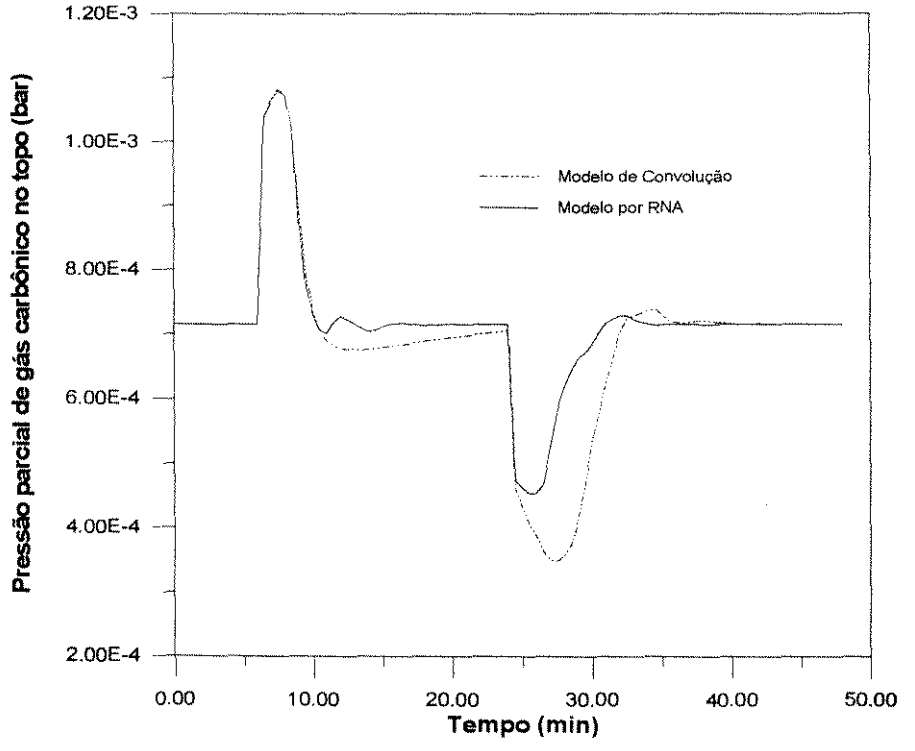

Gráfico 5.17 - Resposta do sistema para duas perturbações em degrau no fluxo de entrada do gás: +5% em t=6min e -5% em t=24min.

CAPÍTULO 6

Conclusões e Sugestões

CONCLUSÕES E SUGESTÕES

A presença da reação química na transferência de massa gera algumas dificuldades numéricas que tornam o estudo deste fenômeno bastante complicado. Ao longo dos anos, os trabalhos que surgiram nesta área se dedicaram em sua maioria a obtenção de soluções analíticas para os cálculos de fluxos de transferência de massa. Entretanto, com o aumento da capacidade de processamento dos computadores e o aparecimento de novas técnicas numéricas (notadamente o método da colocação ortogonal), o problema pôde ser tratado de novas formas.

A discretização das equações originárias dos balanços de massa, tanto no filme de transferência de massa, como ao longo da coluna de absorção, através do método da colocação ortogonal em elementos finitos, mostrou ser uma forma muito mais adequada de se tratar o problema de cálculo de perfis no estado estacionário. As principais vantagens em se resolver o problema desta forma está no tratamento rigoroso que se pode utilizar (permitindo a inclusão de vários efeitos que ocorrem na transferência de massa) associado a uma redução do conjunto de equações não-lineares resultante, o que implica em um esforço computacional menor. Além disto, a solução assim obtida é válida para qualquer regime cinético.

Entretanto, a extensão da metodologia acima ao estudo do comportamento transiente de colunas recheadas, não é válida. O principal problema aqui está no fato do sistema ser distribuído, de forma que as perturbações que são introduzidas nas alimentações da coluna se propagam a uma velocidade finita ao longo desta. Isto gera uma descontinuidade dos perfis ao longo da coluna. Desta forma, método numéricos como colocação ortogonal ou diferenças finitas (que supõem que uma função continua é passada como condição de contorno) não se aplicam a este caso. O método das diferenças finitas amortizam essas descontinuidades, enquanto que o método da colocação ortogonal sofre de problemas de oscilações na resposta.

O método das características consegue manusear descontinuidades ao longo do perfil, o que o torna indicado para a resolução deste caso. No entanto, o grande número de vezes em que este necessita calcular os fluxos de transferência torna inviável a sua utilização juntamente com a resolução numérica das equações de transferência de massa. Para contornar este problema, as redes neurais artificiais foram utilizadas com bastante sucesso na predição dos fluxos de transferência de massa. O modelo híbrido neural, que utiliza as redes juntamente com equações fenomenológicas do processo, mostrou ser capaz de tratar as características da absorção com reação química em colunas recheadas de forma rigorosa. Estas características mostraram ser muito importantes quando colunas de escala industrial são simuladas. De fato, alguns resultados aqui mostrados, como a resposta do sistema a perturbações simultâneas, não poderiam ser obtidos caso um modelo simplificado fosse utilizado.

Os resultados de simulação assim obtidos mostraram que colunas recheadas apresentam alguns fatores que tornam o desenvolvimento de um sistema de controle difícil. Dentre estes, os mais importantes são o atraso na resposta da composição de topo a variações na alimentação de líquido e a alta não-linearidade do sistema devido a presença da reação química.

Dentre as técnicas de controle utilizadas em processos químicos, o controle preditivo com modelo interno é o que melhor consegue manusear as dificuldades acima citadas (Bequette, 1991). Dentre os modelos mais utilizados, destacam-se o baseado em Redes Neurais Artificiais e o de Convolução. Ambos foram aqui testados.

O modelo baseado em Redes Neurais Artificiais mostrou excelentes resultados, quando devidamente escolhida a arquitetura da rede. No entanto, o volume de dados do processo que necessitam ser conhecidos e o grande tempo de treinamento tornam este tipo de abordagem mais difícil de ser implementada do que o modelo de Convolução. Quando a RNA é utilizada para prever um valor que esteja fora do conjunto de dados para o qual esta foi treinada, o resultado é muito pior do que o que seria obtido caso fosse utilizado um modelo linear. Desta forma, o controle baseado em RNA só dá bons resultados quando aplicados a processo que tenham um certo histórico e que não possa sair muito fora destas condições.

Já os resultados de predição obtidos com o modelo de convolução não foram bons. Este quando utilizado para predizer a saída a partir de duas perturbações simultâneas, apresentou uma resposta muito ruim.

Apesar disto, os resultados de controle obtidos não apresentaram uma diferença significativa (quando comparados com resultados da literatura). Isto se deve em parte ao modelo preditivo em si, que é bastante robusto independentemente do modelo, e de características do processo, que não permitem uma ação muito eficiente por parte do controlador. Para exemplificar: caso ocorra um aumento na concentração do gás de alimentação, o controlador toma imediatamente uma ação de controle. No entanto, esta ação só vai ter um efeito sobre a composição de topo alguns instantes após a variação na composição já ter chegado ao topo.

Desta forma, conclui-se que, independente do modelo interno do controlador, sempre haverá um aumento de concentração na saída devida as características do processo. Como o tempo de resposta do sistema é grande, o controle preditivo consegue, através de sucessivas correções, obter uma boa resposta, quando é utilizado o modelo de convolução.

É importante lembrar que para sistemas onde o pico de concentração não possa ser muito grande, o controle preditivo com modelo baseado RNA's torna-se uma melhor opção que o controle linear. Caso isto não ocorra, o modelo de Convolução (para este caso) é mais indicado, tanto pela sua simplicidade de utilização, como pelo pequeno volume de informação que este necessita em relação a outros modelos.

Toda a metodologia aqui descrita na parte de simulação foi feita considerando-se o sistema isotérmico. Esta é uma consideração aceitável caso as concentrações de entrada do gás e da solução líquida não sejam muito altas, caso contrário, o balanço de calor deve ser incluído no sistema. Um grande problema de se incluir os balanços de calor na modelagem é a grande quantidade de dados físico-químicos que precisa ser calculada ou estimada. Dentre outras coisas, podemos citar os coeficientes de película, que se necessita saber para estimar a troca de calor entre gás-líquido e líquido-sólido (recheio). As linhas gerais para o desenvolvimento de balanços de calor de uma forma rigorosa são dadas por Treybal (1969) e Feintuch (1978).

Embora o sistema aqui considerado no controle foi SISO, a temperatura de entrada das correntes pode ser incluída como um variável de entrada do sistema, já que a absorção com reação química é bastante influenciada pela temperatura. Assim, a inclusão de trocadores de calor na malha de controle da absorvedora pode ser uma alternativa para melhorar o desempenho da malha de controle, principalmente através da redução da temperatura da fase gás, tendo em vista o menor tempo de resposta do sistema a perturbações nesta.

REFERÊNCIAS BIBLIOGRÁFICAS

Acrivos, A., "Method of Characteristics Technique - Application to Heat and Mass Transfer Problems", *Industrial and Engineering Chemistry*, **48**, 703 (1956).

Asolekar, S. R., Desai D., Deshpande P. K. e Kumar R., "Effect of Surface Resistence on Gas Absorption Accompanied by a Chemical Reaction in Gas-Liquid Contactor", *The Canadian Journal of Chemical Engineering*, **63**, 336 (1985).

Astarita, G., *"Mass Transfer with Chemical Reaction"*, Elsevier, 1967.

Astarita, G.; Savage, D. W.; Bisio, A.; *"Gas Treating with Chemical Solvents"*, John Wiley & Sons, 1983.

Akaike, H. "On a Successive Transformation of Probability Distribuition ank its application to the analysis of the Optimum Gradient Method", *Ann. Inst. Statist. Math.*, **11**, 1 (1959).

Bequette, B. W., "Nonlinear Control of Chemical Processes: A Review", *Ind. Eng. Chem. Res.* , **30** (7), 1391 (1991).

Bhat, N. e McAvoy, T. J., "Use of Neural Nets for Dynamic Modeling and Control of Chemical Process Systems", *Computers and Chemical Engineering*, **14** (4/5), 573 (1990)

Bird, R. B., Stewart, W. E. e Lightfoot, E. N., *"Transport Phenomena"*, Jonh Wiley & Sons, 1960.

Bradley, K. J. e Andre, H., "A Dynamic Analysis of a Packed Gas Absorber", *The Canadian Journal of Chemical Engineering*, **50**, 529 (1972).

Buchanan, J. E., "Holdup in Irrigated Ring-Packed Towers Below the Loading Point", *I&EC Fundamentals*, **6**(3), 400 (1967).

Caldas, J. N. e Lacerda, A. I., "Torres Recheadas", JR Editora Técnica, 1988.

Carey, G. F. e Finlayson, B. A., "Orthogonal Collocation on Finite Elements", *Chemical Engineering Science*, **30**, 587-596 (1979).

Charpentier, J. C., "What's New in Absorption with Chemical Reaction", *Transactions of Instituition of Chemical Engineering*, **60**, 131 (1982)

Crider, J. E. e Foss, A. S., "Computational Studies of Transients in Packed Tubular Chemical Reactors", *AIChE Journal*, **12**, 515 (1966).

Cutler, C. R. e Ramaker, B. L., "Dynamic matrix control - a computer control algorithm", *AIChE National Meeting*, Houston, Texas (1979).

Cybenko, G., "Approximations by Superpositions of a Sigmoidal Function", *Math. Control Signal Systems*, **2**, 303 (1989)

Danckwerts, P. V. e Sharma, M. M., "The Absorption of Carbon Dioxide into Solutions of Alkalis and Amines (with some notes on Hydrogen Sulphide and Carbonyl Sulphide)", *The Chemical Engineer*, Outubro, 244 (1966).

Danckwerts, P. V., *"Gas-Liquid Reactions"*, McGraw-Hill, 1970.

Danckwerts, P. V., "Significance of Liquid-Film Coefficients in Gas Absorption", *Ind. Eng. Chem.*, **43** (6), 1460 (1951).

De Leye, L. e Froment, G. F., "Rigorous Simulation and Design of Columns for Gas Absorption and Chemical Reaction - I", *Computers and Chemical Engineering*, **10**, 493 (1986).

De Leye, L. e Froment, G. F., "Rigorous Simulation and Design of Columns for Gas Absorption and Chemical Reaction - II", *Computers and Chemical Engineering*, **10**, 505 (1986a).

Eckert, J. S., "Selecting the Proper Distillation Column Packing", *Chemical Engineering Progress*, **66** (3), 39 (1970).

Feintuch, H. M. e Treybal, R. E., "The Design of Adiabatic Packed Towers for Gas Absorption and Stripping", *Ind. Eng. Chem. Process Des. Dev.*, **17**, 505 (1978).

Fileti, A. M. F., "Controle em Destilação Batelada: Controle Adaptativo e Controle Preditivo com Modelo Baseado em Redes Neurais Artificiais", Tese de Doutorado em Engenharia Química, FEQ/UNICAMP, 1995

Finlayson, B. A., *"Nonlinear Analysis in Chemical Engineering"*, McGraw-Hill International Book Company, 1980.

Friedly, J. C., *"Dynamic Behavior of Processes"*, Prentice Hall, 1972.

Froment, G. F. e Bischoff, K. B., *"Chemical Reactor Analysis and Design"*, 2ª edição, John Wiley & Sons, 1990.

García, C. E., Prett, D. M. e Morari, M., "Model Predictive Control: Theory an Practice - A Survey", *Automatica*, **25** (3), 335 (1989)

Glasscock, D. A. e Rochelle, G. T., "Numerical Simulation of Theories for Gas Absorption with Chemical Reaction", *AIChE Journal*, **35** (8), 1271 (1989).

Gray, R. I. e Prados, J. W., "The Dynamics of a Packed Gas Absorber by Frequency Response Analiysis", *AIChE Journal*, **9** (2), 211 (1963).

Hagan, M. T. e Menhaj, M. B., "Training Feedforward Networks with the Marquardt Algorithm", *IEEE Transactions on Neural Networks*, **5** (6), 989 (1994).

Higbie, R., "The Rate of Absorption of a Pure Gas into a Still Liquid During Short Periods Of Exposure", *Transactions Am. Inst. Chem. Eng.*, **31**, 365 (1935).

Hoerner G. M. e Shiesser W. E., "Simultaneous Optimization and Transient Response Evaluation of Packed-Tower Gas Absorption", *Chemical Engineering Progress Syposium Series*, **55** (61), 115 (1961).

Hoskins J. C. e Himmelblau D. M., "Artificial Neural Networks Models of Knowledge Representation in Chemical Engineering", *Computers and Chemical Engineering*, **12** (9/10), 881 (1988).

Hoskins J. C., Kaliyur, K. M. e Himmelblau, D. M., "Falut Diagnosis in Complex Chemical Plants Using Artificial Neural Networks", *AIChE Journal*, **37** (1), 137 (1991).

Hoskins J. C. e Himmelblau D. M., "Process Control Via Artificial Nerual Networks and Reinforcement Learning", *Computers and Chemical Engineering*, **16** (4), 241 (1992).

Hunt, K. J., Sbarbaro, D., Zbikowski, R. e Gawthrop, P. J., "Neural Networks for Control Systems - A Survey", *Automatica*, **28** (6), 1083 (1992).

Jaswon M. A. e Smith W., "Countercurrent Transfer Processes in the Non-Steady State", *Proc. Royal Soc.*, **A225**, 226 (1954).

King, C. J., "Turbulent Liquid Phase Mass Transfer at a Free Gas-Liquid Interface", *Ind. Eng. Chem. Fund.*, **51** (1), 1 (1966).

Krishna R., "Diffusion in Multicomponent Eletrolyte Systems", *Chem. Eng. Journal*, **35**, 19 (1987).

Lakshmanan C. C. e Potter O. E., "Dynamic Simulation of Packed- and Tray-Type Absorbers", *Ind. Eng. Chem. Res.*, **28**, 1397 (1989).

Lees F. P., "The Frequency Response of a Packed Gas Absorption Colunm", *Chemical Engineering Science*, **23**, 97 (1968).

Leonard J. e Kramer M. A., "Improvement of the Backpropagation Algorithm for Training Neural Networks", *Computers and Chemical Engineering*, **14** (3), 337 (1990).

Lewis W. K. e Whitman W. G., "Principles of Gas Absorption", *Ind. Eng. Chem.*, **16** (12), 1215 (1924).

Liapis A. I. e McAvoy T. J., "Transient Solutions for a Class of Hyperbolic Counter-Current Distributed Heat and Mass Transfer Systems", *Trans. IChemE*, **59**, 89 (1981).

Marchetti, J. L., Mellichamp, D. A. e Seborg D. E., "Predictive Control Based on Discrete Convolution Models", *Ind. Eng. Chem. Process Des. Dev.*, **22** (3), 488 (1983).

Morris, G. A. e Jackson, J., *Absorption Towers*, Butterworth, 1953

Morris A. J., Montague G. A e Willis M. J., "Artificial Neural Networks: Studies in Process Modelling and Control", *Trans. IChemE*, **72** part A, 3 (1994).

NAG (*The Numerical Algorithms Group Limited*), The NAG Fortran Library Manual, Mark 15, (1991).

Najim K., "Modelling and Self-Adjusting Control of an Absorption Colunm", *International Journal of Adaptive Control and Signal Processing*, **5**, 335 (1991).

Narendra, K. S. e Parthasarathy K., "Identification and Control of Dynamical Systems Using Neural Networks", *IEEE Transactions on Neural Networks*, **1** (1), 4 (1990).

Onda K., Sada E. e Takeuchi H., "Gas Absorption with Chemical Reaction in Packed Columns", *Journal of Chemical Engineering of Japan*, **1** (1), 62 (1968a).

Onda K., Takeuchi H. e Okumoto Y., "Mass Transfer Coefficients Between Gas and Liquid Phases in Packed Columns", *Journal of Chemical Engineering of Japan*, **1** (1), 56 (1968b).

Potter O. E. e Sridhar T., "Predicting Diffusion Coefficients", *AIChE Journal*, **23** (4), 590 (1977).

Prett, D. M. e Gillette, R. D., "Optimization and Constrained Multivariable Control of a Catalytic Cracking Unit", *AIChE National Meeting*, Houston, Texas.

Psichogios, D. C. e Ungar, L. H., "A Hybrid Neural Networks-First Principles Approach to Process Modeling". *AIChE Journal*, **38** (10), 1499 (1992).

Reid R. C., Prausnitz J. M. e Poling B. E, "The Properties of Gases and Liquids", 4ª edição, McGraw-Hill, 1987.

Richalet, J., Rault, A., Testud, J. L. e Papon, J., "Model Predictive Heuristic Control Aplications to Industrial Processes", *Automatica*, **14**, 413 (1978).

Robitaille B., Marcos B., Veillette e Payre G., "Modified Quasi-Newton Methods for Training Neural Networks", *Computers and Chemical Engineering*, **20** (9), 1133 (1996).

Rumelhart D. E., Hinton G. E. e Williams R. J., "Learning Representations by Back-Propagation Errors", *Nature*, **323** (9), 533 (1986).

Sakata N. e Prados J. W., "The Dynamics of a Packed Gas Absorber by the Pulse Response Technique", *AIChE Journal*, **18** (3), 572 (1972).

Sater, V. e Levenspiel O., *I & E. C. Fund.* **5** (1), 86 (1966).

Seborg, D. E., Edgar, T. F. e Mellichamp, D. A., *"Process Dynamics and Control"*, Wiley Series in Chemical Engineering, 1989

Srisvastava R. K. e Joseph B., "Simulation of Packed-Bed Separation Processes Using Orthogonal Collocation", *Computers and Chemical Engineering*, **8**, 43, (1984).

Tan K. S. e Spinner I. H., "Numerical Methods of Solution for Continuos Countercurrent Process in the Nonsteady State. Part I: Model Equations and Development of Numerical Methods and Algorithms", *AIChE Journal*, **30** (5), 770 (1984a).

Tan K. S. e Spinner I. H., "Numerical Methods of Solution for Continuos Countercurrent Process in the Nonsteady State. Part I: Application of Numerical Methods", *AIChE Journal*, **30** (5), 780 (1984b).

Thomas W. J. e Furzer I. A., "Diffusion Measurements in Liquids By the Gouy Method", *Chemical Engineering Science*, **17**, 115 (1961).

Thompson, M. L. e Kramer, M. A., "Modeling chemical Processes Using Prior Knowledge and Neural Networks", *AIChE Journal*, **40** (8), 1328 (1994).

Tommasi G. e Rice P., "Dynamics of Packed Tower Distillation", *Ind. Eng. Chem. Process Des. Develop.*, **9** (2), 234 (1970).

Treybal R. E., "Adiabatic Gas Absorption and Stripping in Packed Towers", *Industrial and Engineering Chemistry*, **61** (7), 36 (1969).

Tseng Y. e Thompson A. R., "Densities and Refractive Indices of Aqueous Monoethanolamine, Diethanolamine, Triethanolamine", *Journal of Chemical and Engineering Data*, **9** (2), 264 (1964).

Ungar, L.H., Powell, A. "Adaptive Networks for Fault Diagnosis and Process Control", *Computers and Chemical Engineering*, **14** (4/5), 561 (1995).

Villadsen, J., Michelsen, M. L., *"Solutions of Differential Equation Models by Polynomial Approximations"*, Prentice-Hall Incorporation, 1978.

Villadsen J. Sϕrensen J. P., "Solution of Parabolic Differential Equations by a Double Collocation Method", *Chemical Engineering Science*, **24**, 1337 (1969).

Vogl, T. P., Mangis J. K., Rigler A. K., Zink W. T. e Alkon D. L., "Accelerating the Convergence of the Back-Propagation Method", *Biological Cybernetics*, **59**, 257 (1988).

Wang J e Langemann H., "Unsteady Two-Film Model for Mass Transfer Accompanied by Chemical Reaction", *Chemical Engineering Science*, **49**, 3457 (1994).

Willis, M. J., Montague, G. A., Di Massimo, C., Tham, M. T. e Morris, A. J., "Artificial Neural Networks in Process Estimation and Control", *Automatica*, **28** (6), 1181 (1992).

Ydstie, B. E., "Forecasting and Control Using Adaptive Connectionist Networks", *Computers and Chemical Engineering*, **14** (4/5), 583 (1990)

Zadeh, L. A. e Whalen B. H., "On optimal control and linear programming", *IRE Trans. Aut. Control*, 7 (4), 46 (1962).

Zbicinski I, Strumillo P. e Kaminski W., *"Hybrid Neural Model of Thermal Drying in a Fluidized Bed"*, Computers and Chemical Engineering, **20** (suplemento), S695 (1996).

Zens, F. A., *Chem. Eng.*, **13,** 120 (Nov. 13, 1972).

APÊNDICE A - APLICAÇÃO DA COLOCAÇÃO ORTOGONAL EM ELEMENTOS FINITOS NA RESOLUÇÃO DAS EQUAÇÕES DA TEORIA DO FILME.

Desenvolvimento do sistema de equações.

O primeiro passo para a aplicação da colocação ortogonal é a adimensionalização da variável independente. Assim, na equação (3.6a), temos que a variável x pode ser adimensionalizada da seguinte forma:

$$x^* = \frac{x}{y_L} \tag{A-1}$$

No entanto, quando queremos aplicar colocação ortogonal em elementos finitos, esta adimensionalização tem de ser feita no elemento, conforme a equação (2.34). Desta forma, definimos os "NE+1" nós (x_1^*, x_2^*, ... , x_{NE}^*, x_{NE+1}^*), que são os pontos extremos dos elementos a serem utilizados, de forma que temos um total de NE elementos. Definimos então uma nova variável (u_l) que irá variar entre 0 e 1 dentro do elemento "l":

$$u_l = \frac{x^* - x_l^*}{x_{l+1}^* - x_l^*} = \frac{x^* - x_l^*}{\Delta x_l^*} \tag{A-2}$$

A partir das equações (A-1) e (A-2), podemos escrever, para a variável independente, a seguinte expressão:

$$x = y_L x_l^* + u_l y_L \Delta x_l^* \tag{A-3}$$

Desta forma, a equação (3.6a) pode ser reescrita da seguinte forma:

$$\frac{D_A}{y_L^{\,2} \Delta x_l^*} \frac{\partial^2 C_A^l}{\partial u_l^{\,2}} - a_A r = 0 \tag{A-4}$$

onde o sobrescrito "l" indica que a variável está sendo aplicada no domínio do elemento "l".

Aplicando colocação em elementos finitos à equação (A-4) (de acordo com a equação (2.32a)), obtemos:

$$\frac{D_A}{y_L^{\,2} \Delta x_l^*} \sum_{i=1}^{ND} B_{ji} C_{Ai}^l - a_A r = 0 \tag{A-5}$$

onde ND é o número total de pontos de colocação, ou seja, o número de pontos internos (N) mais os dois pontos extremos.

$$ND = N + 2 \tag{A-6}$$

Deve-se ressaltar que a taxa de reação "r" na expressão acima é dependente das concentrações nos pontos de colocação, ou seja, $r = f(C_{Aj}^l, C_{Rkj}^l, C_{Pkj}^l)$. Preferimos aqui tratá-la de forma implícita, já que podemos ter diversas formas para a taxa de reação. Os valores da concentração de reagentes e de produtos, necessários ao cálculo da taxa de reação, podem ser obtidos a partir da concentração do componente absorvido, conforme já discutimos no capítulo 3. As equações (3.10) e (3.11) podem ser reescritas da seguinte forma:

$$C_{Rk,j}^l = C_{Rkb} + \frac{a_{R_k}}{a_A} \frac{D_A}{D_{R_k}} \left(C_{Ak,j}^l - C_{Ab} \right) - \frac{a_{R_k}}{a_A} \frac{N_A\big|_{y=0}}{D_{R_k}} y_L \left(1 - x_l^* - u_{l,j} \Delta x_l^* \right) \tag{A-7}$$

$$C_{Pk,j}^l = C_{Pkb} + \frac{a_{P_k}}{a_A} \frac{D_A}{D_{P_k}} \left(C_{Ak,j}^l - C_{Ab} \right) + \frac{a_{P_k}}{a_A} \frac{N_A\big|_{y=0}}{D_{P_k}} y_L \left(1 - x_l^* - u_{l,j} \Delta x_l^* \right) \tag{A-8}$$

onde $u_{l,j}$ é o ponto de colocação dentro do elemento "l" (o qual é calculado pelas sub-rotinas de colocação).

A equação (A-5) é válida para os pontos internos de colocação, ou seja, para $j = 2,\ldots,N+1$. Como temos uma equação diferencial de segunda ordem, devemos aplicar nos dois pontos extremos as condições de contorno da equação. No entanto, originalmente as nossas condições de contorno foram aplicadas apenas nos pontos extremos do filme de transferência de massa. Assim, temos de utilizar outras condições de contorno nos pontos extremos de cada elemento. Estas condições são dadas pela de igualdade de fluxo e de concentrações entre os elementos.

A condição de igualdade de concentração é direta, já que cada nó corresponde ao último ponto de um elemento e ao primeiro ponto do elemento posterior. A condição de igualdade de fluxo entre os elementos também pode ser vista como a condição de continuidade da função. Desta forma, sendo y_o um ponto de nó entre os elementos "l"e "l+1", a condição de igualdade de fluxo pode ser escrita como:

$$-D_A \left.\frac{\partial C_A}{\partial x}\right|_{x=x_o^+} = -D_A \left.\frac{\partial C_A}{\partial x}\right|_{x=x_o^-} \tag{A-9}$$

Substituindo as equações (A-1) e (A-2) e aplicando colocação de forma semelhante a que já foi utilizada à equação anterior, obtemos:

$$\frac{1}{\Delta x_l}\sum_{i=1}^{ND} A_{ND,i} C_{Ai}^{l} = \frac{1}{\Delta x_{l+1}}\sum_{i=1}^{ND} A_{1,i} C_{Ai}^{l+1} \tag{A-10}$$

A equação (A-10) é válida, obviamente, para $l = 1, Ne-1$.

A outra condição de contorno, a de igualdade de concentrações, é simplesmente:

$$C_{A,ND}^{l} = C_{A,1}^{l+1} \tag{A-11}$$

a qual também é válida para $l = 1, Ne-1$.

As condições de contorno citadas até aqui são apenas aquelas "criadas" entre os elementos para permitir a resolução do sistema por colocação ortogonal em elementos finitos. Além destas, temos as condições de contorno originais do problema que são: o equilíbrio na interface (expresso pela lei de Henry), a igualdade de fluxo na interface (ou seja, não há acúmulo nesta) e a igualdade de concentração entre o último ponto do filme e a concentração no seio da solução. As equações que expressão estas condições já foram discutidas no capítulo 3 (equações (3.7a), (3.8), (3.9) e (3.12)). Estas podem ser reescritas da seguinte forma:

$$P_{Ai} = HC_{A,1}^{l} \tag{A-12}$$

$$k_G(P_{Ab} - P_{Ai}) = -\frac{D_A}{y_L \Delta x_l}\sum_{i=1}^{ND} A_{1,i} C_{A,i}^{l} \tag{A-13}$$

$$C_{A,ND}^{NE} = C_{Ab} \tag{A-14}$$

A constante de Henry deve ser corrigida devido a presença de eletrólitos, quando houver formação destes na reação. Como a lei de Henry é válida na interface, a concentração de produto utilizada na correção deve ser $C_{P_k,1}^{l}$, ou seja, a concentração na interface. A expressão utilizada nesta correção é a sugerida por Danckwerts (1966), equação (3.13).

Resolução do sistema de equações.

A resolução simultânea das equações (A-5), (A-7) e (A-8), juntamente com as condições de contorno (A-10), (A-11), (A-12), (A-13) e (A-14) e a equação (3.13), permite encontrar os perfis ao longo do filme líquido e gasoso.

O sistema é altamente não-linear devido à presença da taxa de reação química na equação (A-5). Utilizamos, na resolução deste, a sub-rotina C05PCF da biblioteca matemática da NAG (1991), a qual é baseada no método híbrido de Powell, que apresenta uma melhor convergência quando a estimativa inicial está distante da solução. O método de Newton-Raphson também foi testado, mas observou-se que, nas condições em que o fluxo de transferência de massa era muito alto, este algorítimo não convergia. Além disto, o tempo gasto pela sub-rotina da NAG foi inferior ao gasto pelo método de Newton-Raphson, nos casos em que este último convergiu. Os índices I e II nas equações (9) a (12) indicam que estas somente são válidas ao longo das características (7) e (8), respectivamente, ou seja, o tempo e a posição das grandezas obtidas na integração das equações (9) a (12) estão relacionados pelas equações (7) e (8). Desta forma, acompanha-se todas as variações que ocorrem em um elemento de fluido que surge na base até o momento em que ele chega ao topo da coluna. As descontinuidades não representam nenhum problema à solução das equações por este método, justamente porque se acompanhar cada elemento isoladamente.

É importante ressaltar que, dado o grande número de somatórios que aparecem no sistema, o cálculo do jacobiano do sistema de forma numérica envolveria um cálculo muito intenso. Desta forma a resolução do sistema de equações acima foi resolvido utilizando jacobiano analítico.

A partir dos perfis de concentração, podemos calcular os fluxos a partir das equações (3.12) a (3.15), que podem ser reescritas, para o caso da colocação ortogonal em elementos finitos, da seguinte forma:

$$N_A\Big|_{y=0} = -\frac{D_A}{y_L \Delta x_1}\sum_{i=1}^{ND} A_{1,i} C_{A,i}^{I} \qquad \text{(A-15)}$$

$$N_A\Big|_{y=y_L} = -\frac{D_A}{y_L \Delta x_1}\sum_{i=1}^{ND} A_{ND,i} C_{A,i}^{NE} \qquad \text{(A-16)}$$

$$N_{R_k}\Big|_{y=y_L} = -\frac{D_{R_k}}{y_L \Delta x_1}\sum_{i=1}^{ND} A_{ND,i} C_{R_k,i}^{NE} \qquad \text{(A-17)}$$

$$N_{P_k}\Big|_{y=y_L} = -\frac{D_{P_k}}{y_L \Delta x_1}\sum_{i=1}^{ND} A_{ND,i} C_{P_k,i}^{NE} \qquad \text{(A-18)}$$

As equações acima são resolvidas pela subrotina FILME, escrita em fortran 77. Esta encontram-se em anexo juntamente com a listagem dos programas utilizados neste trabalho.

APÊNDICE B - LISTAGEM DO PROGRAMA "EST.FOR", QUE CALCULA OS FLUXOS E CONCENTRAÇÕES DE UMA COLUNA DE ABSORÇÃO COM REAÇÃO QUÍMICA NO ESTADO ESTACIONÁRIO.

```
************************************************************************
*     Este programa calcula o estado estacionario de uma coluna de     *
*     absorção com uma única reação química.                           *
*     Utiliza colocacao ortogonal em elementos finitos para resolver   *
*     os balancos na fase gas e liquida, bem como os balancos no       *
*     filme de transferencia de massa.                                 *
*     A estimativa inicial dos perfis ao longo da coluna é feita por   *
*     conjunto simplificado de equações.                               *
************************************************************************
      implicit real*8(a-h,o-z)
      real*8 mil,mig,molha
      character*12 arq,nome
      parameter (ncl=30,nel=11,nfl=20,nre=2,nov=60)
      dimension ar(nre),dr(nre),ca(ncl,nel,nfl),cr(ncl,nel,nfl,nre),
     &cp(ncl,nel,nfl,nre),er(nre),ap(nre),difp(nre),ft(nel,ncl),
     &pab(nel,ncl),vl(nel,ncl),cab(nel,ncl),crb(nel,ncl,nre),
     &cpb(nel,ncl,nre),densr(nre),densp(nre),pmr(nre),pmp(nre),
     &crbin(nre),cpbin(nre),crbout(nre),cpbout(nre),crbout1(nre),
     &cpbout1(nre),pab1(nel,ncl),ft1(nel,ncl),vl1(nel,ncl),
     &cab1(nel,ncl),crb1(nel,ncl,nre),cpb1(nel,ncl,nre),
     &aux(3*nov,6*nov),fluxo(ncl,nfl),fluxo1(ncl,nfl),
     &dfluxo(ncl,ncl,nfl)
c***  Colocacao Ortogonal
      dimension dif1(ncl),dif2(ncl),dif3(ncl),root(ncl),vect(ncl),
     &ni(ncl),alf(ncl),bet(ncl),a(nfl,nfl),b(nfl,nfl),dx(nel),
     &at(ncl,ncl),xq(ncl),x(nfl),xqt(ncl),dxt(nel),xt(ncl)
c***  Variaveis utilizadas pela subrotina da nag
      parameter (ndad1=160,ldfjac=ndad1)
      dimension fjac(ldfjac,ndad1),fvec(ndad1),xv(ndad1)
      external x02ajf
      common /bloco1/ b,dx
      common /bloco1a/ ar,ap,aa,dr,difp
      common /bloco2/ nd,nr,np
      common /bloco3/ ca,cr,cp
      common /bloco3a/er,ea
      common /bloco4/ ne
      common /bloco5/ a,x,xq
      common /bloco6/ ck
      common /difus/ da,dg
      common /const/ temp,r
      common /gastor/ pt
      common /gastor1/ ndt
      common /gastor2/ net
      common /proptor/ av,rol,mil,area
      common /propliq/ sigmac,sigma,d
      common /proprec/ dp,epslon
      common /propper/ densa,densr,densp,pma,pmr,pmp
      common /flutor/ ft,vl
      common /conct/ pab,cab,crb,cpb
      common /propgas/ mig
      common /propgasa/ rog
      common /pos/ l,j,nli
      common /solubilidade/ ho
      common /entrada/ fin,pabin,vlin,cabin,crbin,cpbin
      common /torre/ ht,at,dxt
      common /fluxo/ fluxo
      common /deriv/ dfluxo
      data pi/3.14159/
c***
c***  Entrada de dados atraves de arquivo
c***
c***  Propriedades fisicas e condicoes de contorno
      open(unit=10,file='dadodin.dat',status='old')
      read(10,*) d,z
      read(10,*) av,sigma,sigmac,epslon
      read(10,*) dp,diam,ht
      read(10,*) temp,pt
      read(10,*) fin,pabin
      read(10,*) vlin
      read(10,*) densa,pma,cabin
      read(10,*) nr
      read(10,*) (densr(i),pmr(i),crbin(i),dr(i),i=1,nr)
      read(10,*) np
      read(10,*) (densp(i),pmp(i),cpbin(i),i=1,np)
      read(10,*) r,ck
      read(10,*) aa,ea,da,dg
      read(10,*) (ar(i),er(i),i=1,nr)
      read(10,*) (ap(i),i=1,np)
      read(10,*) ho,rol,rog,mig,mil
      close(10)
      do i=1,np
```

```
        difp(i)=dr(l)
      enddo
c***  Dados p/ configuracao dos pontos de colocacao e elementos na torre
      open(unit=11,file='config2.dat',status='old')
      do i=1,2
        read(11,*) ni(i),alf(i),bet(i)
      enddo
      ndt=ni(1)+2
      nd=ni(2)+2
      read(11,*) ne
      npontos=(ne-1)*(nd-1)+nd
      read(11,*) (xq(l), l=2,ne)
      read(11,*) net
      npt=(net-1)*(ndt-1)+ndt
      read(11,*) (xqt(l), l=2,net)
      read(11,*) h
      close(11)
c***
c***  Calculo das matrizes e pontos de colocacao ortogonal
c***
      do k=1,2
        alfa=alf(k)
        beta=bet(k)
        n=ni(k)
        nt=ni(k)+2
c***  Calculo dos pontos de colocacao
        call jcobi(ncl,n,1,1,alfa,beta,difl,dif2,dif3,root)
        goto (1,2), k
    1   continue
          nopc=1
          do j=1,nt
            xt(j)=root(j)
          enddo
        goto 3
    2   continue
          nopc=2
          do j=1,nt
            x(j)=root(j)
          enddo
        goto 3
    3   continue
c***  Calculo da matriz a(j,i)
        id=1
        do j=1,nt
          call dfopr(ncl,n,1,1,j,id,difl,dif2,dif3,root,vect)
          do i=1,nt
            a(j,i)=vect(i)
          enddo
        enddo
c***  Calculo da matriz b(j,i)
        if (nopc.eq.2) then
          id=2
          do j=1,nt
            call dfopr(ncl,n,1,1,j,id,difl,dif2,dif3,root,vect)
            do i=1,nt
              b(j,i)=vect(i)
            enddo
          enddo
        endif
        if (k.eq.1) then
          do j=1,nt
            do i=1,nt
              at(j,i)=a(j,i)
            enddo
          enddo
        endif
      enddo
c***  Calculo dos elementos
      xq(1)=0.0
      xq(ne+1)=1.0
      do l=1,ne
        dx(l)=xq(l+1)-xq(l)
      enddo
      xqt(1)=0.0
      xqt(net+1)=1.0
      do l=1,net
        dxt(l)=xqt(l+1)-xqt(l)
      enddo
      area=pi*diam**2/4.
      g=9.81*3600.**2
   20 continue
```

```
c***
c***  Estimativa inicial dos perfis ao longo da torre
c***
      do k=1,nr
        crbout(k)=crbin(k)
      enddo
      do k=1,np
        cpbout(k)=cpbin(k)
      enddo
      vlout=vlin
   30 continue
      soma1=0.d0
      soma2=0.d0
      do k=1,nr
        soma1=soma1+pmr(k)/densr(k)*(crbout(k)-crbin(k))
      enddo
      soma2=0.d0
      do k=1,np
        soma2=soma2+pmp(k)/densp(k)*(cpbout(k)-cpbin(k))
      enddo
      vlout1=vlin*(1+soma1+soma2)
      do k=1,nr
        crbout1(k)=(crbin(k)*vlin-ar(k)/aa*((fin*pabin)/pt))/vlout1
      enddo
      do k=1,np
        cpbout1(k)=(cpbin(k)*vlin+ap(k)/aa*((fin*pabin)/pt))/vlout1
      enddo
      soma1=0.d0
      great=0.d0
      difer=dabs((vlout1-vlout)/vlout1)
      if (difer.gt.great) great=difer
      vlout=vlout1
      do k=1,nr
        difer=dabs((crbout1(k)-crbout(k))/crbout1(k))
        if (difer.gt.great) great=difer
        crbout(k)=crbout1(k)
      enddo
      do k=1,np
        difer=dabs((cpbout1(k)-cpbout(k))/cpbout1(k))
        if (difer.gt.great) great=difer
        cpbout(k)=cpbout1(k)
      enddo
      if (difer.gt.1.d-3) goto 30
      fout=fin*(1-pabin/pt)
      do k=1,nr
        crb(net,ndt,k)=crbout(k)
      enddo
      do k=1,np
        cpb(net,ndt,k)=cpbout(k)
      enddo
      vl(net,ndt)=vlout
      ft(net,ndt)=fin
      pab(net,ndt)=pabin
      cab(net,ndt)=0.d0
      passo=1.d-3
      do l=net,1,-1
        do j=ndt,2,-1
c***  Calcula filme de transferencia de massa
          n=(l-1)*(ndt-1)+j
          call filme(l,n)
          ponto=(xqt(l)+xt(j)*dxt(l))*ht
          j1=j-1
          l1=l
          ponto1=(xqt(l1)+xt(j1)*dxt(l1))*ht
          delta1=ponto1-ponto
          soma=0.d0
          do i=1,nd
            soma=soma+a(1,i)*ca(n,1,i)
          enddo
          fluxo(n,1)=-da/(yl(n)*dx(1))*soma
          soma=0.d0
          do i=1,nd
            soma=soma+a(nd,i)*ca(n,ne,i)
          enddo
          fluxo(n,2)=-da/(yl(n)*dx(ne))*soma
          do k=1,nr
            soma=0.d0
            do i=1,nd
              soma=soma+a(nd,i)*cr(n,ne,i,k)
            enddo
            soma=-dr(k)/(yl(n)*dx(ne))*soma
```

```
            fluxo(n,2+k)=-soma
          enddo
          do k=1,np
            soma=0.d0
            do i=1,nd
              soma=soma+a(nd,i)*cp(n,ne,i,k)
            enddo
            soma=-difp(k)/(yl(n)*dx(ne))*soma
            fluxo(n,2+nr+k)=soma
          enddo
c***  Integracao considerando que o fluxo nao varia
          if ((l.ne.net).and.(j.eq.ndt)) then
              pabint=pab(l+1,1)
              fint=ft(l+1,1)
              pab(l,ndt)=pab(l+1,1)
              ft(l,ndt)=ft(l+1,1)
              vl(l,ndt)=vl(l+1,1)
              cab(l,ndt)=cab(l+1,1)
              do k=1,nr
                crb(l,ndt,k)=crb(l+1,1,k)
              enddo
              do k=1,np
                cpb(l,ndt,k)=cpb(l+1,1,k)
              enddo
            else
              pabint=pab(l,j)
              fint=ft(l,j)
          endif
          if (n.ne.npt) then
              fator=(fluxo(n,1)/fluxo(n+1,1))
            else
              fator=1.d0
          endif
          if (fator.gt.1.d0) fator=1.d0
          z=ponto
          dz=passo
   40     continue
            fluxm=fluxo(n,1)*(1-fator)/(ponto-pontol)*(z-ponto)+
     &            fluxo(n,1)
            pabint1=pabint-fluxm*alv(l,j)*area*(pt-pab(l,j))/
     &              ft(l,j)*dz
            fint1=fint*(1.d0-pabint/pt)/(1.d0-pabint1/pt)
            z=z-dz
            pabint=pabint1
            fint=fint1
            if (dabs(dz-passo).lt.1.d-5*passo) then
              falta=z-pontol
              if (falta.lt.dz) then
                dz=falta
              endif
            endif
            dif=dabs(z-pontol)
          if (dif.gt.dz*1.d-3) goto 40
          delta=pontol-ponto
          pab(ll,jl)=pabint
          ft(ll,jl)=fint
          cab(ll,jl)=0.d0
          do k=1,nr
            crb(ll,jl,k)=crb(l,j,k)
          enddo
          do k=1,np
            cpb(ll,jl,k)=cpb(l,j,k)
          enddo
          vl(ll,jl)=vl(l,j)
   50 continue
          soma1=0.d0
          soma2=0.d0
          do k=1,nr
            soma1=soma1+pmr(k)/densr(k)*(crb(ll,jl,k)-crb(l,j,k))
          enddo
          soma2=0.d0
          do k=1,np
            soma2=soma2+pmp(k)/densp(k)*(cpb(ll,jl,k)-cpb(l,j,k))
          enddo
          vlout1=vl(l,j)*(1+soma1+soma2)
          do k=1,nr
            crbout1(k)=(crb(l,j,k)*vl(l,j)+ar(k)/aa*((ft(l,j)*pab(l,j)-
     &                 ft(ll,jl)*pab(ll,jl))/pt))/vlout1
          enddo
          do k=1,np
            cpbout1(k)=(cpb(l,j,k)*vl(l,j)-ap(k)/aa*((ft(l,j)*pab(l,j)-
```

```
     &                ft(ll,jl)*pab(ll,jl))/pt))/vloutl
          enddo
          great=0.d0
          difer=dabs((vloutl-vl(ll,jl))/vloutl)
          if (difer.gt.great) great=difer
          vl(ll,jl)=vloutl
          do k=1,nr
            difer=dabs((crboutl(k)-crb(ll,jl,k))/crboutl(k))
            if (difer.gt.great) great=difer
            crb(ll,jl,k)=crboutl(k)
          enddo
          do k=1,np
            difer=dabs((cpboutl(k)-cpb(ll,jl,k))/cpboutl(k))
            if (difer.gt.great) great=difer
            cpb(ll,jl,k)=cpboutl(k)
          enddo
          if (difer.gt.1.d-3) goto 50
        enddo
      enddo
c***
c***  Calculo dos perfis no filme de transferencia de massa
c***
      vl(1,1)=vlin
      cab(1,1)=cabin
      do k=1,nr
        crb(1,1,k)=crbin(k)
      enddo
      do k=1,np
        cpb(1,1,k)=cpbin(k)
      enddo
      floodb=flood(vlout,fin)*1.d2
      floodt=flood(vlin,fout)*1.d2
      write(*,*) 'Resolvendo balancos ao longo da torre'
      nli=0
      grande=0.d0
      ifalha=0
      iter=1
      nfalha=3
      open(100,file='avisos.dat',status='unknown')
    4 continue

c***  Solucao inicial
      do l=1,net
        nli=(l-1)*(4+nr+np)*ndt
        do j=1+nli,ndt+nli
          xv(j)=ft(1,j-nli)
        enddo
        nli=nli+ndt
        do j=1+nli,ndt+nli
          xv(j)=pab(1,j-nli)
        enddo
        nli=nli+ndt
        do j=1+nli,ndt+nli
          xv(j)=vl(1,j-nli)
        enddo
        nli=nli+ndt
        do j=1+nli,ndt+nli
          xv(j)=cab(1,j-nli)
        enddo
        do k=1,nr
          nli=nli+ndt
          do j=1+nli,ndt+nli
            xv(j)=crb(1,j-nli,k)
          enddo
        enddo
        do k=1,np
          nli=nli+ndt
          do j=1+nli,ndt+nli
            xv(j)=cpb(1,j-nli,k)
          enddo
        enddo
      enddo

C***  Calculo dos fluxos e derivadas ao longo da torre

      if (iter.ne.1) then
          call filme(0,npt)
        else
          call filme(1,1)
      endif
      do l=1,net
```

```
      do j=1,ndt
        n=(l-1)*(ndt-1)+j
        soma=0.d0
        do i=1,nd
          soma=soma+a(l,i)*ca(n,l,i)
        enddo
        fluxo(n,1)=-da/(yl(n)*dx(1))*soma
        soma=0.d0
        do i=1,nd
          soma=soma+a(nd,i)*ca(n,ne,i)
        enddo
        fluxo(n,2)=-da/(yl(n)*dx(ne))*soma
        do k=1,nr
          soma=0.d0
          do i=1,nd
            soma=soma+a(nd,i)*cr(n,ne,i,k)
          enddo
          soma=-dr(k)/(yl(n)*dx(ne))*soma
          fluxo(n,2+k)=-soma
        enddo
        do k=1,np
          soma=0.d0
          do i=1,nd
            soma=soma+a(nd,i)*cp(n,ne,i,k)
          enddo
          soma=-difp(k)/(yl(n)*dx(ne))*soma
          fluxo(n,2+nr+k)=soma
        enddo
      enddo
    enddo
    fator=dsqrt(x02ajf())
    do l=1,net
      do j=1,ndt
        n=(l-1)*(ndt-1)+j
        do il=1,4+nr+np
          if (il.eq.1) then
            delta=fator*ft(l,j)
            if (dabs(delta).lt.1.d-30) delta=1.d-08
            ft(l,j)=ft(l,j)+delta
          endif
          if (il.eq.2) then
            delta=fator*pab(l,j)
            if (dabs(delta).lt.1.d-30) delta=1.d-08
            pab(l,j)=pab(l,j)+delta
          endif
          if (il.eq.3) then
            delta=fator*vl(l,j)
            if (dabs(delta).lt.1.d-30) delta=1.d-08
            vl(l,j)=vl(l,j)+delta
          endif
          if (il.eq.4) then
            delta=fator*cab(l,j)
            if (dabs(delta).lt.1.d-30) delta=1.d-08
            cab(l,j)=cab(l,j)+delta
          endif
          do k=1,nr
            if (il.eq.4+k) then
              delta=fator*crb(l,j,k)
              if (dabs(delta).lt.1.d-30) delta=1.d-08
              crb(l,j,k)=crb(l,j,k)+delta
            endif
          enddo
          do k=1,np
            if (il.eq.4+nr+k) then
              delta=fator*cpb(l,j,k)
              if (dabs(delta).lt.1.d-30) delta=1.d-08
              cpb(l,j,k)=cpb(l,j,k)+delta
            endif
          enddo
          call filme(l,n)
          soma=0.d0
          do i=1,nd
            soma=soma+a(l,i)*ca(n,l,i)
          enddo
          fluxol(n,1)=-da/(yl(n)*dx(1))*soma
          soma=0.d0
          do i=1,nd
            soma=soma+a(nd,i)*ca(n,ne,i)
          enddo
          fluxol(n,2)=-da/(yl(n)*dx(ne))*soma
          do k=1,nr
```

```
            soma=0.d0
            do i=1,nd
              soma=soma+a(nd,i)*cr(n,ne,i,k)
            enddo
            soma=-dr(k)/(yl(n)*dx(ne))*soma
            fluxol(n,2+k)=-soma
          enddo
          do k=1,np
            soma=0.d0
            do i=1,nd
              soma=soma+a(nd,i)*cp(n,ne,i,k)
            enddo
            soma=-difp(k)/(yl(n)*dx(ne))*soma
            fluxol(n,2+nr+k)=soma
          enddo
          do k=1,2+nr+np
            dfluxo(n,k,il)=(fluxol(n,k)-fluxo(n,k))/delta
          enddo
          if (il.eq.1) then
            ft(l,j)=ft(l,j)-delta
          endif
          if (il.eq.2) then
            pab(l,j)=pab(l,j)-delta
          endif
          if (il.eq.3) then
            vl(l,j)=vl(l,j)-delta
          endif
          if (il.eq.4) then
            cab(l,j)=cab(l,j)-delta
          endif
          do k=1,nr
            if (il.eq.4+k) then
              crb(l,j,k)=crb(l,j,k)-delta
            endif
          enddo
          do k=1,np
            if (il.eq.4+nr+k) then
              cpb(l,j,k)=cpb(l,j,k)-delta
            endif
          enddo
        enddo
      enddo
    enddo
    call fcn1(ndad1,xv,fvec,fjac,ldfjac,1)
    call fcn1(ndad1,xv,fvec,fjac,ldfjac,2)
    do i=1,net*(4+nr+np)*ndt
      do j=1,net*(4+nr+np)*ndt
        aux(j,i)=fjac(j,i)
      enddo
    enddo
    ncol=net*(4+nr+np)*ndt+1
    do i=1,net*(4+nr+np)*ndt
      aux(i,ncol)=-fvec(i)
    enddo

c***  Resolucao por Newton-Raphson
    ns=1
    nc=net*(4+nr+np)*ndt
    call gauss(nc,ns,3*nov,aux)

c***  Passagem do resultados para as variaveis
    great=0.d0
    do l=1,net
      nli=(l-1)*(4+nr+np)*ndt
      ncol=net*(4+nr+np)*ndt+1
      do j=1+nli,ndt+nli
        ft1(l,j-nli)=ft(l,j-nli)+aux(j,ncol)
        dif=aux(j,ncol)
        if (dabs(ft(l,j-nli)).gt.1.d-10) then
            valor=dabs(dif/ft(l,j-nli))
          else
            valor=dabs(dif)
        endif
        if (great.lt.valor) great=valor
        ft(l,j-nli)=ft1(l,j-nli)
      enddo
      nli=nli+ndt
      do j=1+nli,ndt+nli
        pab1(l,j-nli)=pab(l,j-nli)+aux(j,ncol)
        dif=aux(j,ncol)
        if (dabs(pab(l,j-nli)).gt.1.d-10) then
```

```
          valor=dabs(dif/pab(l,j-nli))
        else
          valor=dabs(dif)
      endif
      if (great.lt.valor) great=valor
      pab(l,j-nli)=pabl(l,j-nli)
    enddo
    nli=nli+ndt
    do j=1+nli,ndt+nli
      vll(l,j-nli)=vl(l,j-nli)+aux(j,ncol)
      dif=aux(j,ncol)
      if (dabs(vl(l,j-nli)).gt.1.d-10) then
          valor=dabs(dif/vl(l,j-nli))
        else
          valor=dabs(dif)
      endif
      if (great.lt.valor) great=valor
      vl(l,j-nli)=vll(l,j-nli)
    enddo
    nli=nli+ndt
    do j=1+nli,ndt+nli
      cabl(l,j-nli)=cab(l,j-nli)+aux(j,ncol)
      dif=aux(j,ncol)
      if (dabs(cab(l,j-nli)).gt.1.d-10) then
          valor=dabs(dif/cab(l,j-nli))
        else
          valor=dabs(dif)
      endif
      if (great.lt.valor) great=valor
      cab(l,j-nli)=cabl(l,j-nli)
    enddo
    do k=1,nr
      nli=nli+ndt
      do j=1+nli,ndt+nli
        crbl(l,j-nli,k)=crb(l,j-nli,k)+aux(j,ncol)
        dif=aux(j,ncol)
        if (dabs(crb(l,j-nli,k)).gt.1.d-10) then
            valor=dabs(dif/crb(l,j-nli,k))
          else
            valor=dabs(dif)
        endif
        if (great.lt.valor) great=valor
        crb(l,j-nli,k)=crbl(l,j-nli,k)
      enddo
    enddo
    do k=1,np
      nli=nli+ndt
      do j=1+nli,ndt+nli
        cpbl(l,j-nli,k)=cpb(l,j-nli,k)+aux(j,ncol)
        dif=aux(j,ncol)
        if (dabs(cpb(l,j-nli,k)).gt.1.d-10) then
            valor=dabs(dif/cpb(l,j-nli,k))
          else
            valor=dabs(dif)
        endif
        if (great.lt.valor) great=valor
        cpb(l,j-nli,k)=cpbl(l,j-nli,k)
      enddo
    enddo
  enddo

  write(*,*) 'Residuo: ',great

  do l=1,net
    do j=1,ndt
      if (cab(l,j).lt.0.d0) cab(l,j)=0.d0
      if (pab(l,j).lt.0.d0) then
        pab(l,j)=1.d-3
      endif
      do k=1,np
        if (cpb(l,j,k).lt.0.d0) cpb(l,j,k)=0.d0
      enddo
      do k=1,nr
        if (crb(l,j,k).lt.0.d0) then
            crb(l,j,k)=tenta
            tenta=crb(l,j,k)*0.5
          else
            tenta=crb(l,j,k)*0.5
        endif
      enddo
    enddo
```

```
      enddo
      difer=(grande-great)/great
      if (dabs(difer).lt.1.d-1) then
        ifalha=ifalha+1
        if (ifalha.ge.nfalha) write(100,*) iprob,' Great= ',great
      endif
      grande=great
      iter=iter+1
      if ((great.gt.1.d-04).and.(ifalha.lt.nfalha)) goto 4
c***
c***  Saida de dados
c***
      molha=vl(1,1)/alv(1,1)
      floodt=flood(vl(1,1),ft(1,1))*1.d2
      floodb=flood(vl(net,ndt),ft(net,ndt))*1.d2
      write(*,5) 'Grau de molhamento =',molha,'m3/hr.m'
      write(*,5) 'Inundacao (topo) = ',floodt,'%'
      write(*,5) 'Inundacao (base) = ',floodb,'%'
    5 format(1x,a,1x,f5.2,1x,a)
      write(*,*) 'Nome do arquivo de saida de dados?'
      read(*,6) arq(1)
    6 format(a)
      open(12,file=arq(1),status='unknown')
      if (floodb.gt.80.d0) write(12,*) 'Flood na base: ',floodb
      if (molha.lt.0.8d0) write(12,*) 'Taxa de molhabilidade: ',molha
      do l=1,net
        do j=1,ndt-1
          n=(l-1)*(ndt-1)+j
          write(12,7) ft(l,j),pab(l,j),vl(l,j),crb(l,j,1),cpb(l,j,1),
     &                cab(l,j),fluxo(n,1),fluxo(n,3),fluxo(n,4)
          cpb(l,j,2)=cpb(l,j,1)
        enddo
        if (l.eq.net) then
          j=ndt
          n=(l-1)*(ndt-1)+j
          write(12,7) ft(l,j),pab(l,j),vl(l,j),crb(l,j,1),cpb(l,j,1),
     &                cab(l,j),fluxo(n,1),fluxo(n,3),fluxo(n,4)
          cpb(l,j,2)=cpb(l,j,1)
        endif
      enddo
      close(12)
    7 format(1x,f10.6,2x,e12.6,2x,f9.5,2x,f8.6,2x,f8.6,2x,e12.6,2x,
     &       e12.6,2x,e12.6,2x,e12.6)
      iprob=iprob+1
      if (iprob.le.niprob) goto 20
      open(13,file='saidinha.dat',status='unknown')
      do n=1,npt
        write(13,*)
        write(13,*) 'Posicao', n
        do l=1,ne
          do j=1,nd
            pos=xq(l)+x(j)*dx(l)
            write(13,8) pos,ca(n,l,j),cr(n,l,j,1),cp(n,l,j,1)
          enddo
        enddo
      enddo
      close(13)
    8 format(1x,f6.4,2x,e12.5,2x,f6.4,2x,f6.4)
      stop
      end
c-------------------------fim do programa principal--------------------

      subroutine posicao(nt,l,j)
      implicit real*8(a-h,o-z)
      logical volta
      common /gastor1/ ndt
      common /gastor2/ net
      k=0
      volta=.true.
    1 continue
      if (nt-k*(ndt-1).le.0) then
          k=k-1
          volta=.false.
        else
          k=k+1
      endif
      if (volta) goto 1
      j=nt-k*(ndt-1)
      l=k+1
      if (l.gt.net) then
        l=net
```

```
        j=ndt
      endif
      return
      end
c-------------------------fim da subrotina posicao-----------------------

      real*8 function hold()
      implicit real*8(a-h,o-z)
      real*8 mil
      parameter (ncl=30,nel=11)
      dimension ft(nel,ncl),vl(nel,ncl)
      common /proptor/ av,rol,mil,area
      common /propliq/ sigmac,sigma,d
      common /proprec/ dp,epslon
      common /pos/ l,j,nli
      common /flutor/ ft,vl
      g=9.81*3600.**2.
      vazaol=vl(l,j-nli)/area
      dpe=6*(1-epslon)/av
      fr=vazaol**2./(g*epslon*dpe)
      hold=fr**(1./3)
      return
      end
c-------------------------fim da subrotina hold--------------------------

      real*8 function dhold()
      implicit real*8 (a-h,o-z)
      real*8 x02ajf
      external x02ajf
      parameter (ncl=30,nel=11)
      dimension ft(nel,ncl),vl(nel,ncl)
      common /pos/ l,j,nli
      common /flutor/ ft,vl
      vl=hold()
      fator=dsqrt(x02ajf())
      delta=fator*vl(l,j-nli)
      vl(l,j-nli)=vl(l,j-nli)+delta
      v2=hold()
      vl(l,j-nli)=vl(l,j-nli)-delta
      dhold=(v2-vl)/delta
      return
      end
c-------------------------fim da subrotina dhold-------------------------

      real*8 function reac()
      implicit real*8(a-h,o-z)
      parameter (ncl=30,nel=11,nre=2)
      dimension cab(nel,ncl),crb(nel,ncl,nre),cpb(nel,ncl,nre),
     &pab(nel,ncl)
      common /pos/ l,j,nli
      common /conct/ pab,cab,crb,cpb
      common /bloco6/ck
      reac=ck*cab(l,j-nli)*crb(l,j-nli,1)
      return
      end
c-------------------------fim da subrotina reac--------------------------

      real*8 function dreac(n)
      implicit real*8(a-h,o-z)
      parameter (ncl=30,nel=11,nre=2)
      dimension cab(nel,ncl),crb(nel,ncl,nre),cpb(nel,ncl,nre),
     &pab(nel,ncl)
      common /pos/ l,j,nli
      common /conct/ pab,cab,crb,cpb
      common /bloco6/ck
      if (n.eq.4) dreac=ck*crb(l,j-nli,1)
      if (n.eq.5) dreac=ck*cab(l,j-nli)
      if ((n.ne.4).and.(n.ne.5)) dreac=0.d0
      veio=dreac
      return
      end
c-------------------------fim da subrotina dreac-------------------------

      real*8 function dflux(n,m)
      implicit real*8(a-h,o-z)
      parameter (ncl=30,nel=11,nre=2,nfl=20)
      dimension ft(nel,ncl),pab(nel,ncl),vl(nel,ncl),cab(nel,ncl),
     &crb(nel,ncl,nre),cpb(nel,ncl,nre),dfluxo(ncl,ncl,nfl)
      common /flutor/ ft,vl
      common /conct/ pab,cab,crb,cpb
      common /pos/ l,j,nli
```

```
      common /deriv/ dfluxo
      common /gastor1/ ndt
      j1=j-nli
      npt=(l-1)*(ndt-1)+j1
      dflux=dfluxo(npt,n,m)
      return
      end
c-----------------------fim da subrotina dflux-----------------------

      real*8 function kg(nt)
      implicit real*8(a-h,o-z)
      real*8 mil,mig
      parameter (ncl=30,nel=11,nre=2)
      dimension ft(nel,ncl),pab(nel,ncl),vl(nel,ncl),cab(nel,ncl),
     &crb(nel,ncl,nre),cpb(nel,ncl,nre)
      common /difus/ da,dg
      common /proptor/ av,rol,mil,area
      common /flutor/ ft,vl
      common /conct/ pab,cab,crb,cpb
      common /const/ temp,r
      common /proprec/ dp,epslon
      common /propgas/ mig
      common /propgasa/ rog
      common /gastor/ pt
      common /gastor1/ ndt
      common /gastor2/ net
      call posicao(nt,1,j)
      fm=rog*(ft(1,j)*r*temp/pt)/area
      kg=5.23*(fm/(av*mig))**0.7*(mig/(rog*dg))**(1./3)*
     &   (av*dp)**(-2.)/(r*temp/(av*dg))
      return
      end
c-----------------------fim da subrotina kg--------------------------

      real*8 function yg(nt)
      implicit real*8(a-h,o-z)
      real*8 kg
      external kg
      common /difus/ da,dg
      common /const/ temp,r
      yg=dg/(r*temp*kg(nt))
      return
      end
c-----------------------fim da subrotina yg--------------------------

      real*8 function kl(nt)
      implicit real*8(a-h,o-z)
      real*8 mil
      external alv
      parameter (ncl=30,nel=11)
      dimension vl(nel,ncl),ft(nel,ncl)
      common /proprec/ dp,epslon
      common /proptor/ av,rol,mil,area
      common /flutor/ ft,vl
      common /difus/ da,dg
      common /gastor1/ ndt
      common /gastor2/ net
      call posicao(nt,1,j)
      g=9.81*3600.**2
      vazaol=vl(1,j)*rol/area
      kl=0.0051*(vazaol/(alv(1,j)*mil))**(2./3)*(mil/(rol*da))**
     &   (-0.5)*(av*dp)**0.4*(rol/(mil*g))**(-1./3)
      return
      end
c-----------------------fim da subrotina kl--------------------------

      real*8 function yl(nt)
      implicit real*8(a-h,o-z)
      real*8 kl
      external kl
      common /difus/ da,dg
      yl=da/kl(nt)
      return
      end
c-----------------------fim da subrotina yl--------------------------

      real*8 function alv(l,j)
      implicit real*8(a-h,o-z)
      real*8 mil,vazaol
      parameter (ncl=30,nel=11)
      dimension vl(nel,ncl),ft(nel,ncl)
```

```
      common /proptor/ av,rol,mil,area
      common /propliq/ sigmac,sigma,d
      common /flutor/ ft,vl
      g=9.81*3600.**2
      vazaol=vl(l,j)*rol/area
      sig=sigma*3600.**2*1.d-03
      alv=av*(1-exp(-1.45*(sigmac/sigma)**0.75*(vazaol/(av*mil))**0.1*
     &(vazaol**2*av/(rol**2*g))**(-0.05)*(vazaol**2/(rol*sig*av))**0.2))
      return
      end
c-----------------------fim da subrotina alv--------------------------

      real*8 function dalv(l,j)
      implicit real*8(a-h,o-z)
      real*8 x02ajf
      external x02ajf
      parameter (ncl=30,nel=11)
      dimension vl(nel,ncl),ft(nel,ncl)
      common /flutor/ ft,vl
      vl=alv(l,j)
      fator=dsqrt(x02ajf())
      delta=fator*vl(l,j)
      vl(l,j)=vl(l,j)+delta
      v2=alv(l,j)
      vl(l,j)=vl(l,j)-delta
      dalv=(v2-vl)/delta
      return
      end
c-----------------------fim da subrotina dalv-------------------------

      real*8 function henry(nt)
      implicit real*8(a-h,o-z)
      parameter (ncl=30,nel=11,nfl=20,nre=2)
      dimension cae(ncl,nel,nfl),pab(nel,ncl),cab(nel,ncl),
     &crb(nel,ncl,nre),cpb(nel,ncl,nre),ar(nre),ap(nre),dr(nre),
     &difp(nre),a(nfl,nfl),b(nfl,nfl),dx(nel),xq(ncl),x(nfl),
     &cp(ncl,nel,nfl,nre)
      common /blocol/ b,dx
      common /bloco2/ nd,nr,np
      common /gastor1/ ndt
      common /gastor2/ net
      common /solubilidade/ ho
      common /conct/ pab,cab,crb,cpb
      common /blocola/ ar,ap,aa,dr,difp
      common /local/ cae
      common /bloco5/ a,x,xq
      common /difus/ da,dg
      call posicao(nt,l,j)
      soma=0.
      do i=1,nd
        soma=soma+a(l,i)*cae(nt,l,i)
      enddo
      do k=1,np
        cp(nt,l,l,k)=(cpb(l,j,k)-ap(k)/aa*da/difp(k)*(cae(nt,l,l)-
     &              cab(l,j))-ap(k)/aa*da/difp(k)*soma/dx(1)*(1-
     &              (x(1)*dx(1)+xq(1))))
      enddo
      cks=0.031+0.021-0.017
      si=0.
      do k=1,np
        si=si+0.5*cp((l-1)*(ndt-1)+j,1,1,k)
      enddo
      henry=ho/(10.)**(-cks*si)
      return
      end
c-----------------------fim da subrotina henry------------------------

      real*8 function dhenry(nt)
      implicit none
      external henry,x02ajf
      real*8 delta,cae,vl,v2,henry,fator,x02ajf
      integer ncl,nel,nfl,nt
      parameter (ncl=30,nel=11,nfl=20)
      dimension cae(ncl,nel,nfl)
      common /local/ cae
      vl=henry(nt)
      fator=dsqrt(x02ajf())
      delta=cae(nt,1,1)*fator
      if (dabs(delta).lt.1.d-30) delta=1.d-8
      cae(nt,1,1)=cae(nt,1,1)+delta
      v2=henry(nt)
```

```
      cae(nt,l,l)=cae(nt,l,l)-delta
      dhenry=(v2-v1)/delta
      return
      end
c-----------------------fim da subrotina dhenry----------------------

      real*8 function dreacl(n,l,j,i)
      parameter (ncl=30,nel=11,nfl=20,nre=2)
      implicit real*8(a-h,o-z)
      dimension ca(ncl,nel,nfl),cr(ncl,nel,nfl,nre),cp(ncl,nel,nfl,nre),
     &er(nre),rl(2),cae(ncl,nel,nfl)
      common /bloco6/ck
      common /bloco2/nd,nr,np
      common /bloco3/ca,cr,cp
      common /bloco3a/er,ea
      common /local/ cae
      rl(1)=reacl(n,l,j)
      fator=dsqrt(x02ajf())
      if (ca(n,l,i).le.1.d-20) then
          delta=1.d-14
        else
          delta=fator*ca(n,l,i)
      endif
      cae(n,l,i)=cae(n,l,i)+delta
      rl(2)=reacl(n,l,j)
      cae(n,l,i)=cae(n,l,i)-delta
      dreacl=(rl(2)-rl(1))/delta
      return
      end
c-----------------------fim da subrotina dreacl----------------------

      real*8 function reacl(n,l,j)
      parameter (ncl=30,nel=11,nfl=20,nre=2,nov=60)
      implicit real*8(a-h,o-z)
      dimension b(nfl,nfl),dx(nel),dr(nre),
     &ca(ncl,nel,nfl),cr(ncl,nel,nfl,nre),cp(ncl,nel,nfl,nre),er(nre),
     &a(nfl,nfl),x(nfl),xq(ncl),ap(nre),difp(nre),
     &ar(nre),cab(nel,ncl),crbl(nov,nre),cpb(nel,ncl,nre),cabl(nov),
     &pab(nel,ncl),mr(nre),crb(nel,ncl,nre),cae(ncl,nel,nfl)
      common /bloco1/b,dx
      common /blocola/ar,ap,aa,dr,difp
      common /bloco2/nd,nr,np
      common /bloco3/ca,cr,cp
      common /bloco3a/er,ea
      common /bloco4/ne
      common /bloco5/a,x,xq
      common /bloco6/ck
      common /difus/ da,dg
      common /const/temp,r
      common /conct/ pab,cab,crb,cpb
      common /gastor1/ndt
      common /gastor2/net
      common /local/ cae
      call posicao(n,l1,j1)
      cabl((l1-1)*(ndt-1)+j1)=cab(l1,j1)
      do k=1,nr
        crbl((l1-1)*(ndt-1)+j1,k)=crb(l1,j1,k)
      enddo
      prod=1.
      soma=0.
      do i=1,nr
        mr(i)=er(i)
      enddo
      ma=ea
      do i=1,nd
        soma=soma+a(l,i)*cae(n,l,i)
      enddo
      do nk=1,nr
        prod=prod*(crbl(n,nk)+ar(nk)/aa*da/dr(nk)*(cae(n,l,j)-cabl(n))+
     &       ar(nk)/aa*da/dr(nk)/dx(1)*soma*(1-(x(j)*dx(l)+xq(l))))**
     &       mr(nk)
      enddo
      reacl=ck*cae(n,l,j)**ma*prod
      return
      end
c-----------------------fim da subrotina reacl----------------------

      real*8 function flood(vl,ft)
      implicit none
      real*8 vl,ft,l,g,av,roi,rog,mil,area,dp,epslon,f1,f2,f3,temp,r,
     &pt,avl,roll,rogl,mill,f4,raiz,fator2
```

```
      common /proptor/ av,rol,mil,area
      common /proprec/ dp,epslon
      common /const/ temp,r
      common /gastor/ pt
      common /propgasa/ rog
      f1=1.d0/0.3048d0
      f2=1.d0/0.45359d0
      f3=1.d3
      l=(vl/area)*f1/60.d0
      g=(ft*r*temp/pt/area)*f1/60.d0
      av1=av/f1
      rol1=rol*f2/f1**3
      rog1=rog*f2/f1**3
      mil1=mil/3.6d3*f3
      fator2=.7481
      raiz=((av1/(epslon**3))*mil1**0.2)**0.5
      f4=(28.6*g/((rog1*rol1)**0.5)*raiz)**(1./3.)+
     &   (1/fator2*raiz)**0.5
      f4=(18.91-(28.6*g/(rog1*rol1)**0.5*raiz)**(1./3.))**2/
     &   (raiz/fator2)
      flood=vl/(f4/f1*60.d0*area)
      return
      end
c-----------------------fim da subrotina flood----------------------

      subroutine filme(kvar,npt)
      implicit real*8 (a-h,o-z)
      parameter (ndad=21,ldfjac=ndad,lr=ndad*(ndad+1)/2)
      external x02ajf
      external c05pcf,fcn
      external posicao
      parameter (ncl=30,nel=11,nfl=20,nre=2)
      dimension b(nfl,nfl),dx(nel),dr(nre),ca(ncl,nel,nfl),
     &cr(ncl,nel,nfl,nre),cp(ncl,nel,nfl,nre),a(nfl,nfl),x(nfl),xq(ncl),
     &ap(nre),difp(nre),ar(nre),cab(nel,ncl),crb(nel,ncl,nre),
     &cpb(nel,ncl,nre),pab(nel,ncl)
      dimension xv(ndad),fvec(ndad),fjac(ldfjac,ndad),rs(lr),qtf(ndad),
     &work(ndad,4),diag(ndad)
      common /bloco1/b,dx
      common /blocola/ar,ap,aa,dr,difp
      common /bloco2/nd,nr,np
      common /bloco3/ca,cr,cp
      common /bloco4/ne
      common /bloco5/a,x,xq
      common /difus/ da,dg
      common /conct/ pab,cab,crb,cpb
      common /gastor1/ndt
      common /gastor2/net
      common /solubilidade/ ho
      common /ponto/ kponto

      if (kvar.eq.1) then
          n1=npt
          n=npt
          call posicao(n,1,j)
          ca(n,1,1)=pab(1,j)/ho
          do k1=1,nr
            cr(n,1,1,k1)=crb(1,j,k1)
            cr(n,ne,nd,k1)=crb(1,j,k1)/10
          enddo
          do k1=1,np
            cp(n,1,1,k1)=cpb(1,j,k1)
            cp(n,ne,nd,k1)=cpb(1,j,k1)*10
          enddo
          do l1=1,ne
            do j1=1,nd
              pontol=xq(l1)+x(j1)*dx(l1)
              ca(n,l1,j1)=(1-pontol)*ca(n,1,1)
              do k1=1,np
                cp(n,l1,j1,k1)=cp(n,1,1,k1)+pontol*(cp(n,ne,nd,k1)-
     &                         cp(n,1,1,k1))
              enddo
              do k1=1,nr
                cr(n,l1,j1,k1)=cr(n,1,1,k1)+pontol*(cr(n,ne,nd,k1)-
     &                         cr(n,1,1,k1))
              enddo
            enddo
          enddo
        else
          n1=1
          do l=1,net
```

```
        do j=1,ndt
          n=(l-1)*(ndt-1)+j
          ca(n,1,1)=pab(l,j)/ho
          do k1=1,nr
            cr(n,1,1,k1)=crb(l,j,k1)
            cr(n,ne,nd,k1)=crb(l,j,k1)/10
          enddo
          do k1=1,np
            cp(n,1,1,k1)=cpb(l,j,k1)
            cp(n,ne,nd,k1)=cpb(l,j,k1)*10
          enddo
          do l1=1,ne
            do j1=1,nd
              pontol=xq(l1)+x(j1)*dx(l1)
              ca(n,l1,j1)=(1-pontol)*ca(n,1,1)
              do k1=1,np
                cp(n,l1,j1,k1)=cp(n,1,1,k1)+pontol*(cp(n,ne,nd,k1)-
&                              cp(n,1,1,k1))
              enddo
              do k1=1,nr
                cr(n,l1,j1,k1)=cr(n,1,1,k1)+pontol*(cr(n,ne,nd,k1)-
&                              cr(n,1,1,k1))
              enddo
            enddo
          enddo
        enddo
      enddo
    endif


    do n=n1,npt
      kponto=n
      do l=1,ne
        do j=1,nd
          xv((l-1)*(nd-1)+j)=ca(kponto,l,j)
        enddo
      enddo
      maxfev=100*(ndad+1)
      mode=1
      do i=1,ndad
        diag(i)=1.d6
      enddo
      factor=100.0d0
      nprint=0

      xtol=dsqrt(x02ajf())
      ifail=1

      call c05pcf(fcn,ndad,xv,fvec,fjac,ldfjac,xtol,maxfev,diag,mode,
&                 factor,nprint,nfev,njev,rs,lr,qtf,work,ifail)

      if (ifail.ne.0) then
          write(*,*) 'Ifail retornou ',ifail
          pause ' '
      endif
      do l=1,ne
        do j=1,nd
          ca(kponto,l,j)=xv((l-1)*(nd-1)+j)
        enddo
      enddo
      call posicao(n,l1,j1)
      soma=0.
      do i=1,nd
        soma=soma+a(1,i)*ca(n,l,i)
      enddo
      do l=1,ne
        do j=1,nd
          do k=1,nr
            cr(n,l,j,k)=(crb(l1,j1,k)+ar(k)/aa*da/dr(k)*(ca(n,l,j)-
&                        cab(l1,j1))+ar(k)/aa*da/dr(k)/dx(l)*soma*(1-
&                        (x(j)*dx(l)+xq(l))))
          enddo
          do k=1,np
            cp(n,l,j,k)=(cpb(l1,j1,k)-ap(k)/aa*da/difp(k)*(ca(n,l,j)-
&                        cab(l1,j1))-ap(k)/aa*da/difp(k)*soma/dx(l)*
&                        (1-(x(j)*dx(l)+xq(l))))
          enddo
        enddo
      enddo
    enddo
    return
```

```
      end
c-----------------------fim da subrotina filme----------------------

      subroutine fcn(ndad,xv,fvec,fjac,ldfjac,iflag)
      implicit real*8(a-h,o-z)
      real*8 kg,henry,dhenry
      parameter (ncl=30,nel=11,nfl=20,nre=2,nov=60)
      dimension b(nfl,nfl),dx(nel),dr(nre),ca(ncl,nel,nfl),
     &cr(ncl,nel,nfl,nre),cp(ncl,nel,nfl,nre),er(nre),a(nfl,nfl),x(nfl),
     &xq(ncl),cae(ncl,nel,nfl),ap(nre),difp(nre),ar(nre),pabl(nov),
     &cab(nel,ncl),crb(nel,ncl,nre),cpb(nel,ncl,nre),cabl(nov),
     &crbl(nov,nre),cpbl(nov,nre),pab(nel,ncl)
      dimension xv(ndad),fvec(ndad),fjac(ldfjac,ndad)
      common /bloco1/b,dx
      common /bloco1a/ar,ap,aa,dr,difp
      common /bloco2/nd,nr,np
      common /bloco3/ca,cr,cp
      common /bloco3a/er,ea
      common /bloco4/ne
      common /bloco5/a,x,xq
      common /bloco6/ck
      common /difus/ da,dg
      common /const/temp,r
      common /conct/ pab,cab,crb,cpb
      common /gastor1/ndt
      common /gastor2/net
      common /local/ cae
      common /solubilidade/ ho
      common /ponto/ npt
      external kg,henry,dhenry
      call posicao(npt,l,j)
      cabl((l-1)*(ndt-1)+j)=cab(l,j)
      pabl((l-1)*(ndt-1)+j)=pab(l,j)
      do k=1,nr
        crbl((l-1)*(ndt-1)+j,k)=crb(l,j,k)
      enddo
      do k=1,np
        cpbl((l-1)*(ndt-1)+j,k)=cpb(l,j,k)
      enddo
      do n=npt,npt
        do l=1,ne
          do j=1,nd
            cae(n,l,j)=xv((l-1)*(nd-1)+j)
          enddo
        enddo
      enddo
      n=npt
      if (iflag.eq.2) then
        do l=1,ne
          nint=(l-1)*(nd-1)
          do j=2+nint,nd-1+nint
            do i=1+nint,nd+nint
              fjac(j,i)=da/(yl(n)**2.*dx(l)**2.)*b(j-nint,i-nint)-aa*
     &                  dreacl(n,l,j-nint,i-nint)
            enddo
            do i=1,nint
              fjac(j,i)=0.d0
            enddo
            do i=nd+1+nint,(nd-1)*ne+1
              fjac(j,i)=0.d0
            enddo
          enddo
          j=1+nint
          do i=1,((nd-1)*ne+1)
            fjac(j,i)=0.d0
          enddo
          if (l.eq.1) then
              fjac(j,1+nint)=da/(yl(n)*dx(1))*a(1,1)-kg(n)*henry(n)
     &                       -kg(n)*dhenry(n)*cae(n,1,1)
              do i=2+nint,nd+nint
                fjac(j,i)=da/(yl(n)*dx(1))*a(1,i-nint)
              enddo
            else
              do i=1+nint-(nd-1),nint
                fjac(j,i)=a(nd,i-nint+(nd-1))/dx(l-1)
              enddo
              fjac(j,1+nint)=a(nd,nd)/dx(l-1)-a(1,1)/dx(1)
              do i=2+nint,nd+nint
                fjac(j,i)=-a(1,i-nint)/dx(l)
              enddo
          endif
```

```
          if (l.eq.ne) then
              j=((nd-1)*ne+1)
              do i=1,((nd-1)*ne+1)
                fjac(j,i)=0.d00
              enddo
              fjac(j,((nd-1)*ne+1))=1.d00
          endif
        enddo
      endif
c     ***************construcao da funcao do lado direito**************
      if (iflag.eq.1) then
        ncol=((nd-1)*ne+1)+1
        do l=1,ne
          nint=(l-1)*(nd-1)
          do j=2+nint,nd-1+nint
            soma=0.
            do i=1,nd
              soma=soma+b(j-nint,i)*cae(n,l,i)
            enddo
            soma=da/(yl(n)**2.*dx(l)**2.)*soma-aa*reacl(n,l,j-nint)
            fvec(j)=soma
          enddo
          if (l.eq.1) then
              soma=0.d00
              do i=2,nd-1
                soma=soma+a(l,i)*cae(n,l,i)
              enddo
              soma=da/(yl(n)*dx(1))*soma
              soma=soma+da/(yl(n)*dx(1))*a(1,nd)*cae(n,1,nd)
              fvec(1+nint)=(kg(n)*pabl(n)+(da/(yl(n)*dx(1))*
     &                    a(1,1)-kg(n)*henry(n))*cae(n,1,1)+soma)
          else
              j=1+nint
              soma=0.d0
              do i=1,nd-1
                soma=soma+a(nd,i)*cae(n,l-1,i)
              enddo
              soma=soma/dx(l-1)
              soma=soma+(a(nd,nd)/dx(l-1)-a(1,1)/dx(l))*cae(n,l,1)
              soma1=0.d00
              do i=2,nd-1
                soma1=soma1+a(1,i)*cae(n,l,i)
              enddo
              soma1=-soma1/dx(l)
              if (l.eq.ne) then
                  soma1=soma1-a(1,nd)*cae(n,ne,nd)/dx(l)
                else
                  soma1=soma1-a(1,nd)*cae(n,l+1,1)/dx(l)
              endif
              fvec(j)=(soma+soma1)
          endif
          if (l.eq.ne) then
              j=((nd-1)*ne+1)
              fvec(j)=(cae(n,ne,nd)-cabl(n))
          endif
        enddo
      endif
      return
      end
c-------------------------fim da subrotina fcn--------------------------

      subroutine fcn1(ndadl,xv,fvec,fjac,ldfjac,iflag)
      implicit real*8(a-h,o-z)
      real*8 mil
      parameter (ncl=30,nel=11,nre=2)
      dimension ar(nre),dr(nre),ap(nre),difp(nre),ft(nel,ncl),
     &pab(nel,ncl),vl(nel,ncl),cab(nel,ncl),crb(nel,ncl,nre),
     &cpb(nel,ncl,nre),densr(nre),densp(nre),pmr(nre),pmp(nre),
     &crbin(nre),cpbin(nre)
c***  Colocacao Ortogonal
      dimension at(ncl,ncl),dxt(nel)
      dimension xv(ndadl),fvec(ndadl),fjac(ldfjac,ndadl)
      common /blocola/ ar,ap,aa,dr,difp
      common /bloco2/ nd,nr,np
      common /gastor/ pt
      common /gastor1/ ndt
      common /gastor2/ net
      common /proptor/ av,rol,mil,area
      common /propper/ densa,densr,densp,pma,pmr,pmp
      common /flutor/ ft,vl
      common /conct/ pab,cab,crb,cpb
```

```
      common /entrada/ fin,pabin,vlin,cabin,crbin,cpbin
      common /torre/ ht,at,dxt
      common /pos/ l,j,nli
c***  Passagem de valores do vetor xv para as variaveis
      do l=1,net
        nli=(l-1)*(4+nr+np)*ndt
        do j=1+nli,ndt+nli
          ft(l,j-nli)=xv(j)
        enddo
        nli=nli+ndt
        do j=1+nli,ndt+nli
          pab(l,j-nli)=xv(j)
        enddo
        nli=nli+ndt
        do j=1+nli,ndt+nli
          vl(l,j-nli)=xv(j)
        enddo
        nli=nli+ndt
        do j=1+nli,ndt+nli
          cab(l,j-nli)=xv(j)
        enddo
        do k=1,nr
          nli=nli+ndt
          do j=1+nli,ndt+nli
            crb(l,j-nli,k)=xv(j)
          enddo
        enddo
        do k=1,np
          nli=nli+ndt
          do j=1+nli,ndt+nli
            cpb(l,j-nli,k)=xv(j)
          enddo
        enddo
      enddo
c***  Construcao da funcao
      if (iflag.eq.1) then
      do l=1,net
        nli=(l-1)*(4+nr+np)*ndt
c***    Ft
        do j=1+nli,ndt+nli-1
          soma1=0.d0
          do i=1,ndt
            soma1=soma1+at(j-nli,i)*ft(l,i)
          enddo
          fvec(j)=(soma1-flux(l)*alv(l,j-nli)*area*ht*dxt(l))
        enddo
        if (l.eq.net) then
            fvec(ndt+nli)=(ft(net,ndt)-fin)
          else
            fvec(ndt+nli)=(ft(l+1,1)-ft(l,ndt))
        endif
        nli=nli+ndt
c***    Pab
        do j=1+nli,ndt+nli-1
          soma1=0.d0
          do i=1,ndt
            soma1=soma1+at(j-nli,i)*pab(l,i)
          enddo
          fvec(j)=(soma1-flux(l)*alv(l,j-nli)*area*ht*dxt(l)*(pt-
     &             pab(l,j-nli))/ft(l,j-nli))
        enddo
        if (l.eq.net) then
            fvec(ndt+nli)=(pab(net,ndt)-pabin)
          else
            fvec(ndt+nli)=(pab(l+1,1)-pab(l,ndt))
        endif
c***    Vl
        nli=nli+ndt
        do j=2+nli,ndt+nli
          soma2=0.d0
          soma1=0.d0
          do i=1,ndt
            soma1=soma1+at(j-nli,i)*vl(l,i)
            soma2=soma2+at(j-nli,i)*cab(l,i)
          enddo
          soma2=soma2*vl(l,j-nli)*pma/densa
          soma4=0.d0
          do k=1,nr
            soma3=0.d0
            do i=1,ndt
              soma3=soma3+at(j-nli,i)*crb(l,i,k)
```

```
            enddo
            soma4=soma4+pmr(k)/densr(k)*vl(l,j-nli)*soma3
          enddo
          soma6=0.d0
          do k=1,np
            soma5=0.d0
            do i=1,ndt
              soma5=soma5+at(j-nli,i)*cpb(l,i,k)
            enddo
            soma6=soma6+pmp(k)/densp(k)*vl(l,j-nli)*soma5
          enddo
          fvec(j)=(soma1-soma2-soma4-soma6)
        enddo
        if (l.eq.1) then
            fvec(1+nli)=(vl(l,1)-vlin)
          else
            fvec(1+nli)=(vl(l-1,ndt)-vl(l,1))
        endif
c***  Cab
        nli=nli+ndt
        do j=2+nli,ndt+nli
          soma1=0.d0
          soma2=0.d0
          do i=1,ndt
            soma1=soma1+at(j-nli,i)*cab(l,i)
            soma2=soma2+at(j-nli,i)*vl(l,i)
          enddo
          fvec(j)=(vl(l,j-nli)*soma1+cab(l,j-nli)*soma2-flux(2)*
     &                alv(l,j-nli)*area*dxt(l)*ht+aa*reac()*hold()*area*
     &                dxt(l)*ht)
        enddo
        if (l.eq.1) then
            fvec(1+nli)=(cab(l,1)-cabin)
          else
            fvec(1+nli)=(cab(l-1,ndt)-cab(l,1))
        endif
c***  Crb
        do k=1,nr
          nli=nli+ndt
          do j=2+nli,ndt+nli
            soma1=0.d0
            soma2=0.d0
            do i=1,ndt
              soma1=soma1+at(j-nli,i)*crb(l,i,k)
              soma2=soma2+at(j-nli,i)*vl(l,i)
            enddo
            fvec(j)=(vl(l,j-nli)*soma1+crb(l,j-nli,k)*soma2+
     &                flux(2+k)*alv(l,j-nli)*area*dxt(l)*ht+ar(k)*
     &                reac()*hold()*area*dxt(l)*ht)
          enddo
          if (l.eq.1) then
              fvec(1+nli)=(crb(l,1,k)-crbin(k))
            else
              fvec(1+nli)=(crb(l-1,ndt,k)-crb(l,1,k))
          endif
        enddo
c***  Cpb
        do k=1,np
          nli=nli+ndt
          do j=2+nli,ndt+nli
            soma1=0.d0
            soma2=0.d0
            do i=1,ndt
              soma1=soma1+at(j-nli,i)*cpb(l,i,k)
              soma2=soma2+at(j-nli,i)*vl(l,i)
            enddo
            fvec(j)=(vl(l,j-nli)*soma1+cpb(l,j-nli,k)*soma2-
     &                flux(2+nr+k)*alv(l,j-nli)*area*dxt(l)*ht-ap(k)*
     &                reac()*hold()*area*dxt(l)*ht)
          enddo
          if (l.eq.1) then
              fvec(1+nli)=(cpb(l,1,k)-cpbin(k))
            else
              fvec(1+nli)=(cpb(l-1,ndt,k)-cpb(l,1,k))
          endif
        enddo
      enddo
      endif
      if (iflag.eq.2) then

c***
```

```
c***  Construcao do Jacobiano
c***
      do j=1,net*(4+nr+np)*ndt
        do i=1,net*(4+nr+np)*ndt
          fjac(j,i)=0.d0
        enddo
      enddo
c***  Ft
      ndesl=(4+nr+np)*ndt
      do l=1,net
        nli=(l-1)*(4+nr+np)*ndt
        nz=nli
        do j=1+nli,ndt+nli-1
          do i=1,ndt
            if (j-nli.eq.i) then
                fjac(j,i+nz)=at(j-nli,i)-dflux(l,l)*alv(l,j-nli)*area*
     &                       ht*dxt(l)
              else
                fjac(j,i+nz)=at(j-nli,i)
            endif
          enddo
          do i=ndt+1,(4+nr+np)*ndt
            fjac(j,i+nz)=0.d0
          enddo
          do k=2,4+nr+np
            i=j-nli+nz+(k-1)*ndt
            if (k.eq.3) then
                fjac(j,i)=-dflux(l,k)*alv(l,j-nli)*area*ht*dxt(l)-
     &                    flux(l)*dalv(l,j-nli)*area*dxt(l)*ht
              else
                fjac(j,i)=-dflux(l,k)*alv(l,j-nli)*area*ht*dxt(l)
            endif
          enddo
        enddo
        j=ndt+nli
        if (l.eq.net) then
            fjac(j,j)=1.d0
          else
            fjac(j,j)=-1.d0
            fjac(j,j+ndesl-ndt+1)=1.d0
        endif
c***  Pab
        nli=nli+ndt
        do j=1+nli,ndt+nli-1
          do i=1,ndt
            if (j-nli.eq.i) then
                fjac(j,i+nz)=-dflux(l,1)*alv(l,j-nli)*area*ht*dxt(l)*
     &                       (pt-pab(l,j-nli))/ft(l,j-nli)+flux(l)*
     &                       alv(l,j-nli)*area*ht*dxt(l)*(pt-
     &                       pab(l,j-nli))/ft(l,j-nli)**2
              else
                fjac(j,i+nz)=0.d0
            endif
          enddo
          do i=1+ndt,2*ndt
            if (j-nli.eq.i-ndt) then
                fjac(j,i+nz)=at(j-nli,i-ndt)-dflux(l,2)*alv(l,j-nli)*
     &                       area*ht*dxt(l)*(pt-pab(l,j-nli))/
     &                       ft(l,j-nli)+flux(l)*alv(l,j-nli)*area*ht*
     &                       dxt(l)/ft(l,j-nli)
              else
                fjac(j,i+nz)=at(j-nli,i-ndt)
            endif
          enddo
          do i=2*ndt+1,(4+nr+np)*ndt
            fjac(j,i+nz)=0.d0
          enddo
          do k=3,(4+nr+np)
            i=j-nli+nz+(k-1)*ndt
            if (k.eq.3) then
                fjac(j,i)=-dflux(l,k)*alv(l,j-nli)*area*ht*dxt(l)*
     &                    (pt-pab(l,j-nli))/ft(l,j-nli)-flux(l)*
     &                    dalv(l,j-nli)*area*ht*dxt(l)*
     &                    (pt-pab(l,j-nli))/ft(l,j-nli)
              else
                fjac(j,i)=-dflux(l,k)*alv(l,j-nli)*area*ht*dxt(l)*
     &                    (pt-pab(l,j-nli))/ft(l,j-nli)
            endif
          enddo
        enddo
        if (l.eq.net) then
```

```
            fjac(j,j)=1.d0
          else
            fjac(j,j)=-1.d0
            fjac(j,j+ndesl-ndt+1)=1.d0
        endif
c***  Vl
        nli=nli+ndt
        do j=2+nli,ndt+nli
          do i=1,2*ndt
            fjac(j,i+nz)=0.d0
          enddo
          do i=2*ndt+1,3*ndt
            if (j-nli.eq.i-2*ndt) then
                soma1=0.d0
                do i1=1,ndt
                  soma1=soma1+at(j-nli,i1)*cab(1,i1)
                enddo
                soma1=-pma/densa*soma1
                soma3=0.d0
                do k=1,nr
                  soma2=0.d0
                  do i1=1,ndt
                    soma2=soma2+at(j-nli,i1)*crb(1,i1,k)
                  enddo
                  soma3=soma3-pmr(k)/densr(k)*soma2
                enddo
                soma5=0.d0
                do k=1,np
                  soma4=0.d0
                  do i1=1,ndt
                    soma4=soma4+at(j-nli,i1)*cpb(1,i1,k)
                  enddo
                  soma5=soma5-pmp(k)/densp(k)*soma4
                enddo
                fjac(j,i+nz)=at(j-nli,i-2*ndt)+soma1+soma3+soma5
              else
                fjac(j,i+nz)=at(j-nli,i-2*ndt)
            endif
          enddo
          do i=3*ndt+1,4*ndt
            fjac(j,i+nz)=-pma/densa*vl(1,j-nli)*at(j-nli,i-3*ndt)
          enddo
          do k=1,nr
            do i=(3+k)*ndt+1,(4+k)*ndt
              fjac(j,i+nz)=-pmr(k)/densr(k)*vl(1,j-nli)*
     &                    at(j-nli,i-(3+k)*ndt)
            enddo
          enddo
          do k=1,np
            do i=(3+nr+k)*ndt+1,(4+nr+k)*ndt
              fjac(j,i+nz)=-pmp(k)/densp(k)*vl(1,j-nli)*
     &                    at(j-nli,i-(3+nr+k)*ndt)
            enddo
          enddo
        enddo
        j=1+nli
        if (1.eq.1) then
            fjac(j,j)=1.d0
          else
            fjac(j,j)=-1.d0
            fjac(j,j-ndesl+ndt-1)=1.d0
        endif
c***  Cab
        nli=nli+ndt
        do j=2+nli,ndt+nli
          do i=1,2*ndt
            fjac(j,i+nz)=0.d0
          enddo
          do k=1,2
            i=j-nli+nz+(k-1)*ndt
            fjac(j,i)=-dflux(2,k)*alv(1,j-nli)*area*dxt(1)*ht
          enddo
          do i=2*ndt+1,3*ndt
            if (j-nli.eq.i-2*ndt) then
                soma1=0.d0
                do i1=1,ndt
                  soma1=soma1+at(j-nli,i1)*cab(1,i1)
                enddo
                fjac(j,i+nz)=soma1+cab(1,j-nli)*at(j-nli,i-2*ndt)-
     &                       dflux(2,3)*alv(1,j-nli)*area*dxt(1)*ht-
     &                       flux(2)*dalv(1,j-nli)*area*dxt(1)*ht+aa*
```

```
     &                      reac()*dhold()*area*dxt(l)*ht
              else
                fjac(j,i+nz)=cab(l,j-nli)*at(j-nli,i-2*ndt)
              endif
            enddo
            do i=3*ndt+1,4*ndt
              if (j-nli.eq.i-3*ndt) then
                  somal=0.d0
                  do il=1,ndt
                    somal=somal+at(j-nli,il)*vl(l,il)
                  enddo
                  fjac(j,i+nz)=vl(l,j-nli)*at(j-nli,i-3*ndt)+somal-
     &                      dflux(2,4)*alv(l,j-nli)*area*dxt(l)*ht+aa*
     &                      dreac(4)*hold()*area*dxt(l)*ht
              else
                fjac(j,i+nz)=vl(l,j-nli)*at(j-nli,i-3*ndt)
              endif
            enddo
            do i=4*ndt+1,(4+nr+np)*ndt
              fjac(j,i+nz)=0.d0
            enddo
            do k=5,(4+nr+np)
              i=j-nli+(k-1)*ndt
              fjac(j,i)=-dflux(2,k)*alv(l,j-nli)*area*dxt(l)*
     &              ht+aa*dreac(k)*hold()*area*dxt(l)*ht
            enddo
          enddo
          j=1+nli
          if (l.eq.1) then
              fjac(j,j)=1.d0
            else
              fjac(j,j)=-1.d0
              fjac(j,j-ndes1+ndt-1)=1.d0
          endif
c***  Crb
          do k=1,nr
            nli=nli+ndt
            do j=2+nli,ndt+nli
              do i=1,2*ndt
                fjac(j,i+nz)=0.d0
              enddo
              do k1=1,2
                i=j-nli+nz+(k1-1)*ndt
                fjac(j,i)=dflux(2+k,k1)*alv(l,j-nli)*area*dxt(l)*ht
              enddo
              do i=2*ndt+1,3*ndt
                if (j-nli.eq.i-2*ndt) then
                    somal=0.d0
                    do il=1,ndt
                      somal=somal+at(j-nli,il)*crb(l,il,k)
                    enddo
                    fjac(j,i+nz)=somal+crb(l,j-nli,k)*at(j-nli,i-2*ndt)+
     &                        dflux(2+k,3)*alv(l,j-nli)*area*dxt(l)*ht+
     &                        flux(2+k)*dalv(l,j-nli)*area*dxt(l)*ht+
     &                        ar(k)*reac()*dhold()*area*dxt(l)*ht
                else
                  fjac(j,i+nz)=crb(l,j-nli,k)*at(j-nli,i-2*ndt)
                endif
              enddo
              do i=3*ndt+1,4*ndt
                if (j-nli.eq.i-3*ndt) then
                    fjac(j,i+nz)=dflux(2+k,4)*alv(l,j-nli)*area*dxt(l)*ht+
     &                        ar(k)*dreac(4)*hold()*area*dxt(l)*ht
                else
                  fjac(j,i+nz)=0.d0
                endif
              enddo
              do k1=1,nr
                if (k1.eq.k) then
                    do i=(3+k1)*ndt+1,(4+k1)*ndt
                      if (j-nli.eq.i-(3+k1)*ndt) then
                          somal=0.d0
                          do il=1,ndt
                            somal=somal+at(j-nli,il)*vl(l,il)
                          enddo
                          fjac(j,i+nz)=vl(l,j-nli)*at(j-nli,i-(3+k)*ndt)+
     &                              somal+dflux(2+k,4+k1)*alv(l,j-nli)*
     &                              area*dxt(l)*ht+ar(k)*dreac(4+k1)*
     &                              hold()*area*dxt(l)*ht
                        else
                          fjac(j,i+nz)=vl(l,j-nli)*at(j-nli,i-(3+k1)*ndt)
```

```
                endif
              enddo
            else
              do i=(3+k1)*ndt+1,(4+k1)*ndt
                if (j-nli.eq.i-(3+k1)*ndt) then
                    fjac(j,i+nz)=dflux(2+k,4+k1)*alv(l,j-nli)*area*
     &                           dxt(l)*ht+ar(k)*dreac(4+k1)*hold()*
     &                           area*dxt(l)*ht
                  else
                    fjac(j,i+nz)=0.d0
                endif
              enddo
          endif
        enddo
        do k1=1,np
          do i=(3+nr+k1)*ndt+1,(4+nr+k1)*ndt
            if (j-nli.eq.i-(3+nr+k1)*ndt) then
                fjac(j,i+nz)=dflux(2+k,4+nr+k1)*alv(l,j-nli)*area*
     &                       dxt(l)*ht
              else
                fjac(j,i+nz)=0.d0
            endif
          enddo
        enddo
      enddo
      j=l+nli
      if (l.eq.1) then
          fjac(j,j)=1.d0
        else
          fjac(j,j)=-1.d0
          fjac(j,j-ndesl+ndt-1)=1.d0
      endif
    enddo
c***  Cpb
    do k=1,np
      nli=nli+ndt
      do j=2+nli,ndt+nli
        do i=1,2*ndt
          fjac(j,i+nz)=0.d0
        enddo
        do k1=1,2
          i=j-nli+nz+(k1-1)*ndt
          fjac(j,i)=-dflux(2+nr+k,k1)*alv(l,j-nli)*area*dxt(l)*ht
        enddo
        do i=2*ndt+1,3*ndt
          if (j-nli.eq.i-2*ndt) then
              soma1=0.d0
              do il=1,ndt
                soma1=soma1+at(j-nli,il)*cpb(l,il,k)
              enddo
              fjac(j,i+nz)=soma1+cpb(l,j-nli,k)*at(j-nli,i-2*ndt)-
     &                     flux(2+nr+k)*dalv(l,j-nli)*area*dxt(l)*
     &                     ht-dflux(2+nr+k,3)*alv(l,j-nli)*area*
     &                     dxt(l)*ht-ap(k)*reac()*dhold()*area*
     &                     dxt(l)*ht
            else
              fjac(j,i+nz)=cpb(l,j-nli,k)*at(j-nli,i-2*ndt)
          endif
        enddo
        do i=3*ndt+1,(4+nr)*ndt
          fjac(j,i+nz)=0.d0
        enddo
        do k1=4,4+nr
          i=j-nli+nz+(k1-1)*ndt
          fjac(j,i)=-dflux(2+nr+k,k1)*alv(l,j-nli)*area*dxt(l)*ht-
     &              ap(k)*dreac(k1)*hold()*area*dxt(l)*ht
        enddo
        do k1=1,np
          if (k1.eq.k) then
              do i=(3+nr+k1)*ndt+1,(4+nr+k1)*ndt
                if (j-nli.eq.i-(3+nr+k1)*ndt) then
                    soma1=0.d0
                    do il=1,ndt
                      soma1=soma1+vl(l,il)*at(j-nli,il)
                    enddo
                    fjac(j,i+nz)=vl(l,j-nli)*at(j-nli,i-(3+nr+k1)*
     &                           ndt)+soma1-dflux(2+nr+k,4+nr+k1)*
     &                           alv(l,j-nli)*area*dxt(l)*ht-
     &                           ap(k)*dreac(4+nr+k1)*hold()*area*
     &                           dxt(l)*ht
                  else
```

```
                        fjac(j,i+nz)=vl(l,j-nli)*at(j-nli,i-(3+nr+kl)*
     &                                 ndt)
                      endif
                    enddo
                  else
                    do i=(3+nr+kl)*ndt+1,(4+nr+kl)*ndt
                      if (j-nli.eq.i-(3+nr+kl)*ndt) then
                        fjac(j,i+nz)=-dflux(2+nr+k,4+nr+kl)*
     &                               alv(l,j-nli)*area*dxt(l)*ht-ap(k)*
     &                               dreac(4+nr+kl)*hold()*area*dxt(l)*ht
                      else
                        fjac(j,i+nz)=0.d0
                      endif
                    enddo
                endif
              enddo
            enddo
            j=1+nli
            if (l.eq.1) then
                fjac(j,j)=1.d0
              else
                fjac(j,j)=-1.d0
                fjac(j,j-ndesl+ndt-1)=1.d0
            endif
          enddo
        enddo
        endif
        return
        end
c-------------------------fim da subrotina fcnl-------------------------

        real*8 function flux(nflux)
        implicit real*8(a-h,o-z)
        parameter (ncl=30,nfl=20)
        dimension fluxo(ncl,nfl)
        common /gastorl/ ndt
        common /pos/ l,j,nli
        common /fluxo/ fluxo
        jl=j-nli
        npt=(l-1)*(ndt-1)+jl
        flux=fluxo(npt,nflux)
        return
        end
c-------------------------fim da subrotina flux-------------------------
c
c***
c***  As sub-rotinas a seguir (GAUSS, JCOBI, DFOPR e INTRP) foram
c***  retiradas de Villadsen e Michelsen (1978)
c***

        subroutine gauss(n,ns,ncol,a)
        double precision a(ncol,2*ncol),x
        nl=n+1
        nt=n+ns
        if(n.eq.1) go to 50
        do 40 i=2,n
        ip=i-1
        il=ip
        x=dabs(a(il,il))
        do 10 j=i,n
        if(dabs(a(j,il)).lt.x) go to 10
        x = dabs(a(j,il))
        ip=j
 10     continue
        if(ip.eq.il) go to 30
        do 20 j=il,nt
        x = a(il,j)
        a(il,j) = a(ip,j)
 20     a(ip,j) = x
 30     do 40 j=i,n
        x = a(j,il)/a(il,il)
        do 40 k=i,nt
 40     a(j,k) = a(j,k)-x*a(il,k)
 50     do 70 ip=1,n
        i = nl-ip
        do 70 k=nl,nt
        a(i,k) = a(i,k)/a(i,i)
        if (i.eq.1) go to 70
        il = i-1
        do 60 j=1,il
 60     a(j,k) = a(j,k)-a(i,k)*a(j,i)
```

```
70    continue
      return
      end
c-------------------------fim da subrotina gauss-------------------------

      subroutine jcobi (nd,n,no,nl,al,be,difl,dif2,dif3,root)
      implicit real*8 (a-h,o-z)
      dimension difl(nd),dif2(nd),dif3(nd),root(nd)
c
c     calculo das raizes e derivadas do polinomio de jacobi
c
      ab=al+be
      ad=be-al
      ap=be*al
      difl(1)=(ad/(ab+2)+1)/2
      dif2(1)=0.
      if (n.lt.2) goto 15
      do 10 i=2,n
      z1=i-1
      z=ab+2*z1
      difl(i)=(ab*ad/z/(z+2)+1)/2
      if (i.ne.2) goto 11
      dif2(i)=(ab+ap+z1)/z/z/(z+1)
      goto 10
   11 z=z*z
      y=z1*(ab+z1)
      y=y*(ap+y)
      dif2(i)=y/z/(z-1)
   10 continue
c
c     determinacao das raizes pelo metodo de newton com supressao das
c     raizes determinadas anteriormente
c
   15 x=0.
      do 20 i=1,n
   25 xd=0.
      xn=1.
      xd1=0.
      xn1=0.
      do 30 j=1,n
      xp=(difl(j)-x)*xn-dif2(j)*xd
      xp1=(difl(j)-x)*xn1-dif2(j)*xd1-xn
      xd=xn
      xd1=xn1
      xn=xp
   30 xn1=xp1
      zc=1.
      z=xn/xn1
      if (i.eq.1) goto 21
      do 22 j=2,i
   22 zc=zc-z/(x-root(j-1))
   21 z=z/zc
      x=x-z
      if (dabs(z).gt.1.d-09) goto 25
      root(i)=x
      x=x+0.0001
   20 continue
c
c     acrescenta eventuais pontos de colocacao em x=0 e x=1
c
      nt=n+no+nl
      if (no.eq.0) goto 35
      do 31 i=1,n
      j=n+1-i
   31 root(j+1)=root(j)
      root(1)=0
   35 if (nl.eq.1) root(nt)=1.
c
c     calculo das derivadas do polinomio
c
      do 40 i=1,nt
      x=root(i)
      difl(i)=1.
      dif2(i)=0.
      dif3(i)=0.
      do 40 j=1,nt
      if (j.eq.i) goto 40
      y=x-root(j)
      dif3(i)=y*dif3(i)+3*dif2(i)
      dif2(i)=y*dif2(i)+2*difl(i)
      difl(i)=y*difl(i)
```

```
   40 continue
      return
      end
c-----------------------fim da subrotina jacobi----------------------


      subroutine dfopr(nd,n,no,nl,i,id,difl,dif2,dif3,root,vect)
      implicit real*8(a-h,o-z)
      dimension difl(nd),dif2(nd),dif3(nd),root(nd),vect(nd)
c
c     dfopr calcula as matrizes de discretizacao e os pesos da
c     quadratura gaussiana, normalizados para soma 1
c     id = 1;  matriz de dicretizacao para y(1) (x)
c     id = 2;  matriz de dicretizacao para y(2) (x)
c     id = 3;  pesos da quadratura gaussiana
c
      nt=n+no+nl
      if(id.eq.3) goto 10
      do 20 j=1,nt
      if (j.ne.i) goto 21
      if (id.ne.1) goto 5
      vect(i)=dif2(i)/difl(i)/2
      goto 20
    5 vect(i)=dif3(i)/difl(i)/3
      goto 20
   21 y=root(i)-root(j)
      vect(j)=difl(i)/difl(j)/y
      if (id.eq.2) vect(j)=vect(j)*(dif2(i)/difl(i)-2/y)
   20 continue
      goto 50
   10 y=0.
      do 25 j=1,nt
      x=root(j)
      ax=x*(1-x)
      if (no.eq.0) ax=ax/x/x
      if (nl.eq.0) ax=ax/(1-x)/(1-x)
      vect(j)=ax/difl(j)**2
   25 y=y+vect(j)
      do 60 j=1,nt
   60 vect(j)=vect(j)/y
   50 return
      end
c-----------------------fim da subrotina dfopr-----------------------

      subroutine intrp(nd,nt,x,root,difl,xintp)
      implicit real*8 (a-h,o-z)
      dimension root(nd),difl(nd),xintp(nd)
c
c     avaliacao dos coeficientes de interpolacao lagrangiana
c
      pol=1.
      do 5 i=1,nt
      y=x-root(i)
      xintp(i)=0.d0
      if (dabs(y).lt.1.d-10) xintp(i)=1.
    5 pol=pol*y
      if (dabs(pol).lt.1.d-10) goto 10
      do 6 i=1,nt
    6 xintp(i)=pol/difl(i)/(x-root(i))
   10 return
      end
c-----------------------fim da subrotina intrp-----------------------
```

APÊNDICE C - LISTAGEM DO PROGRAMA "DIN.FOR", CALCULA A RESPOSTA TRANSIENTE DE UMA COLUNA DE ABSORÇÃO COM UM ÚNICA REAÇAO QUÍMICA.

```
************************************************************************
*     Este programa calcula a resposta transiente de uma coluna de     *
*     recheio de absorcao com reacao quimica. Alem disto, este         *
*     possui tambem sub-rotinas que de controle preditivo com modelos  *
*     baseados em RNA's e no modelo de Convolucao.                     *
************************************************************************

      implicit double precision(a-h,o-z)
      double precision mil,mig,nagint,nagint1,inp,
     &nplint,nplint1,nrlint,nrlint1,nalint,nalint1,iea
      parameter (nef=6,npf=6,nec=4,npc=6,ncl=(npc-1)*nec+1)
      parameter (nre=1,npr=2)
      parameter (ngr=50,ngl=250)
      parameter (nl=32,nm=22,nn=6,nmp=30)
      parameter (ndm=20,nen=10)
      parameter (ndim=30,neconv=3)
      integer flag,tvar
      character*12 arq,arqs
      logical controla
      dimension ar(nre),dr(nre),ca(ncl,nef,npf),cr(ncl,nef,npf,nre),
     &cp(ncl,nef,npf,npr),er(nre),ap(npr),difp(npr),ft(nec,npc),
     &pab(nec,npc),vl(nec,npc),cab(nec,npc),crb(nec,npc,nre),
     &cpb(nec,npc,npr),densr(nre),densp(npr),pmr(nre),pmp(npr),
     &crbin(nre),cpbin(npr),cpbout(npr),tvar(10),tempol(10),nvar(10),
     &pulso(10),ktipo(10),fint(ngr),vlint(ngl),zg(ngr),zl(ngl),zgl(ngr),
     &zll(ngl),xintp(npc),pabint(ngr),crbaux(nre),crbaux1(nre),
     &cpbaux(npr),cpbaux1(npr),vlint1(ngl),pabl(ncl),pabint1(ngr),
     &cabl(ncl),crbl(ncl,nre),cpbl(ncl,npr),fint1(ngr),cabint(ngl),
     &crbint(ngl,nre),cpbint(ngl,npr),cabint1(ngl),cpbint1(ngl,npr),
     &crbint1(ngl,nre),npfag(2),concr(nre),concp(npr),arqs(3),
     &aconv(neconv,ndim),hconv(neconv,ndim)
c***  Colocacao Ortogonal
      dimension dif1(npc),dif2(npc),dif3(npc),root(npc),vect(npc),
     &ni(npc),alf(npc),bet(npc),a(npf,npf),b(npf,npf),dx(nef),
     &at(npc,npc),xq(ncl),x(npf),xqt(npc),dxt(nec),xt(npc)
c***  Redes Neurais
      dimension inp(nmp,nl),des(nmp,nn),res(nmp,nn),par(1),pvv(4,nl,nm),
     &pwv(4,nm,nn),carv(4,nl+nn),cbrv(4,nl+nn),ntav(4,nl+nn),pv(nl,nm),
     &pw(nm,nn),car(nl+nn),cbr(nl+nn),nta(nl+nn),infov(4,8)
c***  Vetores da sub-rotina de modelo por redes neurais
      dimension entradas(ndm,nen)
c***  Vetores utilizados nas subrotinas de controle
      common /bloco1/ b,dx
      common /bloco1a/ar,ap,aa,dr,difp
      common /bloco2/ nd,nr,np
      common /bloco3/ ca,cr,cp
      common /bloco3a/ er,ea
      common /bloco4/ ne
      common /bloco5/ a,x,xq
      common /bloco6/ ck
      common /difus/ da,dg
      common /const/ temp,r
      common /tempo/ t,h
      common /gastor/ pt
      common /gastor1/ ndt
      common /gastor2/ net
      common /proptor/ av,rol,mil,area
      common /propliq/ sigmac,sigma,d
      common /proprec/ dp,epslon
      common /propper/ densa,densr,densp,pma,pmr,pmp
      common /flutor/ ft,vl
      common /conct/ pab,cab,crb,cpb
      common /conct1/ pabl,cabl,crbl,cpbl
      common /dadomod/ nmod,nvar
      common /dadomoda/ tvar,tempo1,pulso,ktipo
      common /propgas/ mig
      common /propgasa/ rog
      common /solubilidade/ ho
      common /reder/ inp,des,res,par,pvv,pwv,carv,cbrv,pv,pw,car,cbr
      common /redei/ ntav,nta
      common /redinfo/ infov
      common /modelo/ njan,ninp,ns
      common /mdado/ entradas
      common /mentsai/ fin,pabin,vlin,cpbin,cpbout,pabout
      common /limite/ vlmin,vlmax,vlflood
      common /paracont/ ihc,ipred,imodcont
      common /setpoint/ set
      common /coefconv/ aconv,hconv
      common /intconv/ nent,nmodel
      data pi/3.14159/
c***
```

```
c***  Entrada de dados fisico-quimicos e de condicoes de entrada
c***  da coluna atraves de arquivo
c***
      open(unit=09,file='dadodin.dat',status='old')
      read(9,*) d,z
      read(9,*) av,sigma,sigmac,epslon
      read(9,*) dp,diam,ht
      read(9,*) temp,pt
      read(9,*) fin,pabin
      read(9,*) vlin
      read(9,*) densa,pma,cabin
      read(9,*) nr
      read(9,*) (densr(i),pmr(i),crbin(i),dr(i),i=1,nr)
      read(9,*) np
      read(9,*) (densp(i),pmp(i),cpbin(i),i=1,np)
      read(9,*) r,ck
      read(9,*) aa,ea,da,dg
      read(9,*) (ar(i),er(i),i=1,nr)
      read(9,*) (ap(i),i=1,np)
      read(9,*) ho,rol,rog,mig,mil
      close(9)
      do i=1,np
        difp(i)=dr(1)
      enddo
c***
c***  Entrada de dados de configuracao dos pontos de colocacao e
c***  elementos ao longo da torre (que devem ser iguais aos utilizados
c***  na geracao do perfil estacionario.
c***
      open(unit=11,file='config2.dat',status='old')
      do i=1,2
        read(11,*) ni(i),alf(i),bet(i)
      enddo
      tempao=0.d0
      ndt=ni(1)+2
      nd=ni(2)+2
      read(11,*) ne
      npontos=(ne-1)*(nd-1)+nd
      read(11,*) (xq(l), l=2,ne)
      read(11,*) net
      npt=(net-1)*(ndt-1)+ndt
      read(11,*) (xqt(l), l=2,net)
      read(11,*) h
      close(11)
c***
c***  Calculo das matrizes de colocacao
c***
      do k=1,2
        alfa=alf(k)
        beta=bet(k)
        n=ni(k)
        nt=ni(k)+2
c***
c***  Calculo dos pontos de colocacao
c***
        call jcobi(npc,n,1,1,alfa,beta,dif1,dif2,dif3,root)
        goto (1,2), k
    1   continue
          nopc=1
          do j=1,nt
            xt(j)=root(j)
          enddo
        goto 3
    2   continue
          nopc=2
          do j=1,nt
            x(j)=root(j)
          enddo
        goto 3
    3   continue
c***
c***  Calculo da matriz a(j,i)
c***
        id=1
        do j=1,nt
          call dfopr(npc,n,1,1,j,id,dif1,dif2,dif3,root,vect)
          do i=1,nt
            a(j,i)=vect(i)
          enddo
        enddo
c***
```

```
c***  Calculo da matriz b(j,i)
c***
        if (nopc.eq.2) then
          id=2
          do j=1,nt
            call dfopr(npc,n,1,l,j,id,dif1,dif2,dif3,root,vect)
            do i=1,nt
              b(j,i)=vect(i)
            enddo
          enddo
        endif
        if (k.eq.1) then
          do j=1,nt
            do i=1,nt
              at(j,i)=a(j,i)
            enddo
          enddo
        endif
      enddo
c***
c***  Calculo dos vetores de root e difl a serem usados pela rotina de
c***  de integracao
c***
      n=ni(1)
      call jcobi(npc,n,1,1,alf(1),bet(1),dif1,dif2,dif3,root)

      flag=1
      npfag(1)=0
      npfag(2)=0
      xq(1)=0.0
      xq(ne+1)=1.0
      do l=1,ne
        dx(l)=xq(l+1)-xq(l)
      enddo
      xqt(1)=0.0
      xqt(net+1)=1.0
      do l=1,net
        dxt(l)=xqt(l+1)-xqt(l)
      enddo
      area=pi*diam**2/4.
      idados=1

      ti=h
      ntrein=0
c***
c***  Leitura do perfil inicial ao longo da torre (no formato em que
c***  eh gerado pelo programa EST.FOR
c***
      write(*,*) 'nome do arquivo de entrada de dados?'
      read(*,23) arq
      if (arq.eq.'             ') then
        arq='perfinic.dat'
        write(*,*) 'Padrao = "perfinic.dat"'
        write(*,*)
      endif
   23 format(a)
      open(17,file=arq,status='old')
      do l=1,net
        do j=1,ndt-1
          read(17,*) ft(l,j),pab(l,j),vl(l,j),crb(l,j,1),cpb(l,j,1)
          cpb(l,j,2)=cpb(l,j,1)
        enddo
        if (l.eq.net) then
          j=ndt
          read(17,*) ft(l,j),pab(l,j),vl(l,j),crb(l,j,1),cpb(l,j,1)
          cpb(l,j,2)=cpb(l,j,1)
        endif
        if (l.ne.1) then
          ft(l-1,ndt)=ft(l,1)
          pab(l-1,ndt)=pab(l,1)
          vl(l-1,ndt)=vl(l,1)
          crb(l-1,ndt,1)=crb(l,1,1)
          cpb(l-1,ndt,1)=cpb(l,1,1)
          cpb(l-1,ndt,2)=cpb(l,1,2)
        endif
      enddo
      vlin=vl(1,1)
      do l=1,net
        do j=1,ndt
          cab(l,j)=0.d0
        enddo
```

```
      enddo
      cabin=cab(1,1)
      close(17)
      g=9.81*3600.**2

c***
c***  Valores iniciais para subrotinas de controle
c***
*     Variavel logica que define se o simulador deve rodar sob
*     controle ou nao.
      controla=.true.
*     Horizonte de predicao
      ihc=3
*     Horizonte de controle
      ipred=21
*     Tempo de amostragem do sistema
      tamost=0.5d0
*     Vazao minima p/ garatir grau de molhamento minimo
      call gmolha(0,vlmin,grau)
*     Vazao maxima (limitada da valvula)
      vlmax =100.d0
*     Variacao maxima da vazao de liquido permitida
      dvmax=7.7d0
*     Flag de atuacao do controle
      ncflag=0
*     Informacoes iniciais para as sub-rotinas de controle
      ifalha=-2
      nchama=-1
      ichama=0
*     Inicializacao do IEA (com valor negativo para indicar 1a. chamada
      iea=-1.d0
*     Modelo de predicao a ser utilizado (Convolucao=0, RNA=1)
      imodcont=0

c***  Leitura dos coeficientes do modelo de convolucao
      if (imodcont.eq.0) then
        open(10,file='conv.dat',status='old')
        read(10,*) nmodel,nent
        do k=1,nent
          do i=1,nmodel
            read(10,*) aconv(k,i),hconv(k,i)
          enddo
        enddo
        close(10)
      endif
c***
c***  Estimativa inicial dos perfis no filme de transferencia de massa
c***
      do l=1,net
        do j=1,ndt
          n=(l-1)*(ndt-1)+j
          ca(n,1,1)=pab(l,j)/ho
          do k1=1,nr
            cr(n,1,1,k1)=crb(l,j,k1)
            cr(n,ne,nd,k1)=crb(l,j,k1)/10
          enddo
          do k1=1,np
            cp(n,1,1,k1)=cpb(l,j,k1)
            cp(n,ne,nd,k1)=cpb(l,j,k1)*10
          enddo
          do l1=1,ne
            do j1=1,nd
              pontol=xq(l1)+x(j1)*dx(l1)
              ca(n,l1,j1)=(1-pontol)*ca(n,1,1)
              do k1=1,np
                cp(n,l1,j1,k1)=cp(n,1,1,k1)+pontol*(cp(n,ne,nd,k1)-
     &                         cp(n,1,1,k1))
              enddo
              do k1=1,nr
                cr(n,l1,j1,k1)=cr(n,1,1,k1)+pontol*(cr(n,ne,nd,k1)-
     &                         cr(n,1,1,k1))
              enddo
            enddo
          enddo
        enddo
      enddo
c***
c***  Carrega pesos da RNA utilizada na predicao do fluxo de
c***  transferencia de massa
c***
      write(*,*)
```

```
      write(*,*)
      write(*,*) 'Nome dos arquivos de entrada da rede treinada?'
      write(*,*)
      do i=1,3
      enddo
   16 format(1x,'Arquivo ',i1,' = ')
      do k1=1,3
        write(*,*)
        write(*,16) k1
        read(*,23) arqs(k1)
        if (arqs(k1).eq.'            ') then
          goto (17,13,14), k1
   17     continue
            arqs(k1)='peso1.dat'
            write(*,*) 'Padrao = "peso1.dat"'
          goto 18
   13     continue
            arqs(k1)='peso2.dat'
            write(*,*) 'Padrao = "peso2.dat"'
          goto 18
   14     continue
            arqs(k1)='peso3.dat'
            write(*,*) 'Padrao = "peso3.dat"'
          goto 18
   18     continue
        endif
        open(11,file=arqs(k1),status='unknown')
        read(11,*) infov(k1,3)
        read(11,*) infov(k1,4)
        read(11,*) infov(k1,5)
        do i=1,infov(k1,3)+infov(k1,5)
          read(11,*) carv(k1,i)
        enddo
        do i=1,infov(k1,3)+infov(k1,5)
          read(11,*) cbrv(k1,i)
        enddo
        do i=1,infov(k1,3)
          do j=1,infov(k1,4)
            read(11,*) pvv(k1,i,j)
          enddo
        enddo
        do j=1,infov(k1,4)
          do k=1,infov(k1,5)
            read(11,*) pwv(k1,j,k)
          enddo
        enddo
        close(11)
      enddo
c***  Tipo de adimensionalizacao das entradas/saidas
      do k=1,3
        do i=1,infov(k,3)+infov(k,5)
          ntav(k,i)=0
        enddo
        ntav(k,1)=1
        ntav(k,7)=1
      enddo
c***
c***  Carrega pesos do modelo interno do controle preditivo por RNA
c***
      write(*,*) 'Nome do arquivo do modelo por redes neurais?'
      read(*,23) arq
      if (arq.eq.'            ') then
        arq='pesomt.dat'
        write(*,*) 'Padrao = "pesomt.dat"'
        write(*,*)
      endif
      open(11,file=arq,status='unknown')
      read(11,*) njan,ninp,ns
      read(11,*) infov(4,3)
      read(11,*) infov(4,4)
      read(11,*) infov(4,5)
      do i=1,infov(4,3)+infov(4,5)
        read(11,*) carv(4,i)
      enddo
      do i=1,infov(4,3)+infov(4,5)
        read(11,*) cbrv(4,i)
      enddo
      do i=1,infov(4,3)
        do j=1,infov(4,4)
          read(11,*) pvv(4,i,j)
        enddo
```

```
      enddo
      do j=1,infov(4,4)
        do k=1,infov(4,5)
          read(11,*) pwv(4,j,k)
        enddo
      enddo
      close(11)
      do i=1,infov(4,3)+infov(4,5)
        ntav(4,i)=0
      enddo
c***
c***  Malha inicial a ser utilizada no metodo das caracteristicas.
c***  E calculada usando interpolacao polinomial a partir do perfil
c***  gerado para o estado estacionario
c***
      dt=h
      fint(1)=fin
      pabint(1)=pabin
      hgint=hg(vlin)
      alvint=alv(vlin)
      do kr=1,nr
        crbint(1,kr)=crbin(kr)
      enddo
      do kp=1,np
        cpbint(1,kp)=cpbin(kr)
      enddo
      cabint(1)=cabin
      vlint(1)=vlin
      zg(1)=ht
      zl(1)=0.d0
      iter=1
      k=1
      l=net
c***  Calculo da fase gas
   40 continue
        if (iter.eq.1) then
          zg(k+1)=zg(k)-fint(k)/(hgint*area)*dt
          fint(k+1)=fint(k)
        endif
        great=0.d0
        if (zg(k+1).gt.0.d0) then
          if (zg(k+1)/ht.lt.xqt(l)) l=l-1
        endif
        zx=(zg(k+1)/ht-xqt(l))/dxt(l)
        call intrp(npc,ndt,zx,root,difl,xintp)
        hgintl=hg(vlint(l))
        soma=0.d0
        do i=1,ndt
          soma=soma+xintp(i)*ft(l,i)
        enddo
        fint(k+1)=soma
        valor=zg(k)-(fint(k)/(hgint*area)+fint(k+1)/
     &         (hgintl*area))*dt/2.d0
        if (dabs((valor-zg(k+1))/zg(k)).gt.great)
     &    great=dabs((valor-zg(k+1))/zg(k))
        iter=iter+1
        zg(k+1)=valor
      if (great.gt.1.d-8) goto 40
      soma=0.d0
      do i=1,ndt
        soma=soma+xintp(i)*alv(vl(l,i))
      enddo
      alvintl=soma
      soma=0.d0
      do i=1,ndt
        soma=soma+xintp(i)*pab(l,i)
      enddo
      pabint(k+1)=soma
      if (zg(k+1).gt.0.d0) then
        k=k+1
        iter=1
        goto 40
      endif
      fator=zg(k)/(zg(k)-zg(k+1))
      fint(k+1)=fint(k)-fator*(fint(k)-fint(k+1))
      pabint(k+1)=pabint(k)-fator*(pabint(k)-pabint(k+1))
      zg(k+1)=0.d0
      kmaxg=k+1
      do k=1,kmaxg
        fintl(k)=fint(k)
        pabintl(k)=pabint(k)
```

```
        zgl(k)=zg(k)
      enddo
      k=1
      l=1
      iter=1
c***  Calculo da fase liquida
   41 continue
        if (iter.eq.1) then
          zl(k+1)=zl(k)+vlint(k)/(hgint*area)*dt
          vlint(k+1)=vlint(k)
        endif
        great=0.d0
        if (zl(k+1).lt.ht) then
          if (zl(k+1)/ht.gt.xqt(l+1)) l=l+1
        endif
        zx=(zl(k+1)/ht-xqt(l))/dxt(l)
        call intrp(npc,ndt,zx,root,difl,xintp)
        soma=0.d0
        do i=1,ndt
          soma=soma+xintp(i)*vl(l,i)
        enddo
        vlint(k+1)=soma
        hgintl=hg(vlint(k+1))
        valor=zl(k)+(vlint(k)/(hgint*area)+vlint(k+1)/
     &         (hgintl*area))*dt/2.d0
        if (zl(k).gt.1.d-15) then
          if (dabs((valor-zl(k+1))/zl(k)).gt.great)
     &      great=dabs((valor-zl(k+1))/zl(k))
        endif
        iter=iter+1
        zl(k+1)=valor
      if (great.gt.1.d-8) goto 41
      soma=0.d0
      do i=1,ndt
        soma=soma+xintp(i)*cab(l,i)
      enddo
      cabint(k+1)=soma
      do kr=1,nr
        soma=0.d0
        do i=1,ndt
          soma=soma+xintp(i)*crb(l,i,kr)
        enddo
        crbint(k+1,kr)=soma
      enddo
      do kr=1,np
        soma=0.d0
        do i=1,ndt
          soma=soma+xintp(i)*cpb(l,i,kr)
        enddo
        cpbint(k+1,kr)=soma
      enddo
      if (zl(k+1).lt.ht) then
        k=k+1
        iter=1
        goto 41
      endif
      fator=(zl(k)-ht)/(zl(k)-zl(k+1))
      vlint(k+1)=vlint(k)-fator*(vlint(k)-vlint(k+1))
      cabint(k+1)=cabint(k)-fator*(cabint(k)-cabint(k+1))
      do kr=1,nr
        crbint(k+1,kr)=crbint(k,kr)-fator*(crbint(k,kr)-crbint(k+1,kr))
      enddo
      do kr=1,np
        cpbint(k+1,kr)=cpbint(k,kr)-fator*(cpbint(k,kr)-cpbint(k+1,kr))
      enddo
      zl(k+1)=ht
      kmax1=k+1
      do k=1,kmax1
        do kr=1,nr
          crbint1(k,kr)=crbint(k,kr)
        enddo
        do kr=1,np
          cpbint1(k,kr)=cpbint(k,kr)
        enddo
        vlint1(k)=vlint(k)
        cabint1(k)=cabint(k)
        zl1(k)=zl(k)
      enddo
c***
c***  Inicio do programa principal
c***
```

```
c***  Define tempo inicial e tempo total de simulacao
      t=0.d0
      e=0.8d0
c***  Abre arquivo que guarda saidas do processo a cada "tamost"
c***  intervalo de tempo

      open(16,file='out.dat',status='unknown')

c***  Formatacao dos dados a serem impresso no arquivo anterior

   72 format(f8.5,1x,e12.6,1x,f6.4,1x,f8.4,1x,f8.4,1x,f7.5,1x,f7.5)

*************************************************************************
*                    Inicio do laco de integracao                       *
*************************************************************************

   10 continue

c***
c***  Chamada a sub-rotina que modifica condicoes de entrada da torre
c***

      call modifica(fin,pabin,vlin,cabin,crbin,cpbin,flag,npfag)

c***
c***  Modifica condicao de entrada do gas
c***
      if (npfag(1).eq.1) then
        fint1(1)=fin
        pabint1(1)=pabin
        zg1(1)=ht
        do k=1,kmaxg
          fint1(k+1)=fint(k)
          pabint1(k+1)=pabint(k)
          zg1(k+1)=zg(k)
        enddo
        kmaxg=kmaxg+1
        do k=1,kmaxg
          fint(k)=fint1(k)
          pabint(k)=pabint1(k)
          zg(k)=zg1(k)
        enddo
      endif

c***  Modifica as condicoes de entrada do liquido (as modificacoes
c***  no liquido ocorrem tanto por acao de controle, como por
c***  perturbacao externa)

      if ((npfag(2).eq.1).or.(ncflag.eq.1)) then
        vlint1(1)=vlin
        cabint1(1)=cabin
        do kr=1,nr
          crbint1(1,kr)=crbin(kr)
        enddo
        do kp=1,np
          cpbint1(1,kp)=cpbin(kp)
        enddo
        zl1(1)=0.d0
        do k=1,kmaxl
          vlint1(k+1)=vlint(k)
          cabint1(k+1)=cabint(k)
          do kr=1,nr
            crbint1(k+1,kr)=crbint(k,kr)
          enddo
          do kp=1,np
            cpbint1(k+1,kp)=cpbint(k,kp)
          enddo
          zl1(k+1)=zl(k)
        enddo
        kmaxl=kmaxl+1
        do k=1,kmaxl
          vlint(k)=vlint1(k)
          cabint(k)=cabint1(k)
          do kr=1,nr
            crbint(k,kr)=crbint1(k,kr)
          enddo
          do kp=1,np
            cpbint(k,kp)=cpbint1(k,kp)
          enddo
          zl(k)=zl1(k)
        enddo
```

```
        ncflag=0
      endif
      kmaxg1=kmaxg
      kmaxl1=kmaxl
c      iteracao=1
   29 continue
      great=0.d0
      loc=kmaxl
      k=1
      l=net
      iter=1
      zgl(1)=ht
   28   continue
c***  Predicao do fluxo interpolando-se os valores p/ fase liquida
      if (loc.gt.kmaxl) loc=kmaxl
    6   continue
        if (zg(k).gt.0.d0) then
            if ((zg(k).le.zl(loc)).and.(zg(k).ge.zl(loc-1)))
     &        then
              if (loc.ne.1) then
                fator=(zg(k)-zl(loc-1))/(zl(loc)-zl(loc-1))
                nentre=1
              endif
            else
              if (zg(k).lt.zl(loc)) then
                  loc=loc-1
              else
                loc=loc+1
              endif
              goto 6
            endif
          else
            loc=1
            if (dabs(zl(loc+1)-zl(loc)).gt.1.d-8) then
                fator=(zg(k)-zl(loc))/(zl(loc+1)-zl(loc))
              else
                loc=2
                fator=(zg(k)-zl(loc))/(zl(loc+1)-zl(loc))
            endif
            nentre=0
        endif

        hgint=hg(vlint(k))
        if (nentre.eq.1) then
            cabaux=cabint(loc-1)+fator*(cabint(loc)-cabint(loc-1))
            vlaux=vlint(loc-1)+fator*(vlint(loc)-vlint(loc-1))
            alvint=alv(vlaux)
            do kr=1,nr
              crbaux(kr)=crbint(loc-1,kr)+fator*(crbint(loc,kr)-
     &                   crbint(loc-1,kr))
            enddo
            do kp=1,np
              cpbaux(kp)=cpbint(loc-1,kp)+fator*(cpbint(loc,kp)-
     &                   cpbint(loc-1,kp))
            enddo
          else
            cabaux=cabint(loc)+fator*(cabint(loc+1)-cabint(loc))
            vlaux=vlint(loc)+fator*(vlint(loc+1)-vlint(loc))
            alvint=alv(vlaux)
            do kr=1,nr
              crbaux(kr)=crbint(loc,kr)+fator*(crbint(loc+1,kr)-
     &                   crbint(loc,kr))
            enddo
            do kp=1,np
              cpbaux(kp)=cpbint(loc,kp)+fator*(cpbint(loc+1,kp)-
     &                   cpbint(loc,kp))
            enddo
        endif
c***  Predicao do fluxo
        inp(1,1)=pabint(k)
        inp(1,2)=fint(k)
        do kr=1,nr
          inp(1,kr+2)=crbaux(kr)
        enddo
        do kp=1,np
          inp(1,kp+nr+2)=cpbaux(kp)
        enddo
        inp(1,np+nr+3)=vlaux
        nagint=flux(1)

        if (k.ge.kmaxg1) then
```

```
          zgl(k+1)=0.d0
          fint1(k+1)=fint1(k)
          pabint1(k+1)=pabint1(k)
        endif
        if (loc.gt.kmax11) loc=kmax11
    5   continue
        if (zgl(k+1).gt.0.d0) then
            if ((zgl(k+1).le.zl1(loc)).and.(zgl(k+1).ge.zl1(loc-1)))
     &        then
              if (loc.ne.1) then
                fator=(zgl(k+1)-zl1(loc-1))/(zl1(loc)-zl1(loc-1))
                nentre=1
              endif
            else
              if (zgl(k+1).lt.zl1(loc)) then
                  loc=loc-1
                else
                  loc=loc+1
              endif
              goto 5
          endif
        else
          loc=1
          if (dabs(zl1(loc+1)-zl1(loc)).gt.1.d-8) then
              fator=(zgl(k+1)-zl1(loc))/(zl1(loc+1)-zl1(loc))
            else
              loc=2
              fator=(zgl(k+1)-zl1(loc))/(zl1(loc+1)-zl1(loc))
          endif
          nentre=0
        endif

        hgint1=hg(vlint1(k+1))
        if (nentre.eq.1) then
            cabaux1=cabint1(loc-1)+fator*(cabint1(loc)-cabint1(loc-1))
            vlaux1=vlint1(loc-1)+fator*(vlint1(loc)-vlint1(loc-1))
            alvint1=alv(vlaux1)
            do kr=1,nr
              crbaux1(kr)=crbint1(loc-1,kr)+fator*(crbint1(loc,kr)-
     &                    crbint1(loc-1,kr))
            enddo
            do kp=1,np
              cpbaux1(kp)=cpbint1(loc-1,kp)+fator*(cpbint1(loc,kp)-
     &                    cpbint1(loc-1,kp))
            enddo
          else
            cabaux1=cabint1(loc)+fator*(cabint1(loc+1)-cabint1(loc))
            vlaux1=vlint1(loc)+fator*(vlint1(loc+1)-vlint1(loc))
            alvint1=alv(vlaux1)
            do kr=1,nr
              crbaux1(kr)=crbint1(loc,kr)+fator*(crbint1(loc+1,kr)-
     &                    crbint1(loc,kr))
            enddo
            do kp=1,np
              cpbaux1(kp)=cpbint1(loc,kp)+fator*(cpbint1(loc+1,kp)-
     &                    cpbint1(loc,kp))
            enddo
        endif
c***  Predicao do fluxo
        inp(1,1)=pabint1(k+1)
        inp(1,2)=fint1(k+1)
        do kr=1,nr
          inp(1,kr+2)=crbaux1(kr)
        enddo
        do kp=1,np
          inp(1,kp+nr+2)=cpbaux1(kp)
        enddo
        inp(1,np+nr+3)=vlaux1
        nagint1=flux(1)

        v1=1-pabint(k)/pt
        v2=1-pabint1(k+1)/pt
        valor=fint(k)*v1/v2
        if (dabs(valor-fint1(k+1))/fint1(k+1).gt.great)
     &    great=dabs(valor-fint1(k+1))/fint1(k+1)
        fint1(k+1)=valor
        v1=nagint*alvint*(pt-pabint(k))/hgint
        v2=nagint1*alvint1*(pt-pabint1(k+1))/hgint1
        valor=pabint(k)-(v1+v2)*dt/2.
        if (dabs(valor-pabint1(k+1))/pabint1(k+1).gt.great)
     &    great=dabs(valor-pabint1(k+1))/pabint1(k+1)
```

```
        pabint1(k+1)=valor

        valor=zg(k)-(fint(k)/(hgint*area)+fint1(k+1)/
     &        (hgint1*area))*dt/2.d0
        if (dabs(zg1(k+1)).ge.1.d-8) then
            if (dabs((valor-zg1(k+1))/zg1(k+1)).gt.great)
     &        great=dabs((valor-zg1(k+1))/zg1(k+1))
          else
            if (dabs((valor-zg1(k+1))).gt.great)
     &        great=dabs((valor-zg1(k+1)))
        endif
        zg1(k+1)=valor
        if (zg1(k+1).le.0.d0) then
            kmaxg1=k+1
            goto 27
          else
            k=k+1
            goto 28
        endif
c***
c***  Balanco global de liquido pelo metodo das caracteristicas
c***
   27 continue
      loc=kmaxg
      k=1
      l=1
      zl1(1)=0.d0
c***  Predicao do fluxo interpolando-se os valores p/ fase gas
      if (loc.gt.kmaxg) loc=kmaxg
   11   continue
        if (loc.gt.kmaxg) loc=kmaxg
        if (zl(k).lt.ht) then
            if ((zl(k).le.zg(loc-1)).and.(zl(k).ge.zg(loc)))
     &        then
              fator=(zl(k)-zg(loc-1))/(zg(loc)-zg(loc-1))
            else
              if (zl(k).gt.zg(loc)) then
                  loc=loc-1
                else
                  loc=loc+1
              endif
              goto 11
            endif
        endif

        alvint=alv(vlint(k))
        hlint=hold(vlint(k))
        pabaux=pabint(loc-1)+fator*(pabint(loc)-pabint(loc-1))
        faux=fint(loc-1)+fator*(fint(loc)-fint(loc-1))
c***  Predicao do fluxo
        inp(1,1)=pabaux
        inp(1,2)=faux
        do kr=1,nr
          inp(1,kr+2)=crbint(k,kr)
        enddo
        do kp=1,np
          inp(1,kp+nr+2)=cpbint(k,kp)
        enddo
        inp(1,np+nr+3)=vlint(k)
        nagint=flux(1)
        nalint=0.d0
        nrlint=-flux(2)
        nplint=flux(3)

        if (loc.gt.kmaxg1) loc=kmaxg1
   12   continue
        if (zl1(k+1).lt.ht) then
          if ((zl1(k+1).le.zg1(loc-1)).and.(zl1(k+1).ge.zg1(loc)))
     &      then
            fator=(zl1(k+1)-zg1(loc-1))/(zg1(loc)-zg1(loc-1))
            else
              if (zl1(k+1).gt.zg1(loc)) then
                  loc=loc-1
                else
                  loc=loc+1
              endif
              goto 12
          endif
          else
            fator=(zl1(k+1)-zg1(1))/(zg1(2)-zg1(1))
            loc=2
```

```
        endif

        hlint1=hold(vlint1(k+1))
        alvint1=alv(vlint1(k+1))
        pabaux1=pabint1(loc-1)+fator*(pabint1(loc)-pabint1(loc-1))
        faux1=fint1(loc-1)+fator*(fint1(loc)-fint1(loc-1))
c***  Predicao do fluxo
        inp(1,1)=pabaux1
        inp(1,2)=faux1
        do kr=1,nr
          inp(1,kr+2)=crbint1(k+1,kr)
        enddo
        do kp=1,np
          inp(1,kp+nr+2)=cpbint1(k+1,kp)
        enddo
        inp(1,np+nr+3)=vlint1(k+1)
        nagint1=flux(1)
        nalint1=0.d0
        nrlint1=-flux(2)
        nplint1=flux(3)

        conca=cabint(k)
        do kr=1,nr
          concr(kr)=crbint(k,kr)
        enddo
        do kp=1,np
          concp(kp)=cpbint(k,kp)
        enddo
        reacao=reac(conca,concr,concp,nr,np,ck)
        conca=cabint1(k+1)
        do kr=1,nr
          concr(kr)=crbint1(k+1,kr)
        enddo
        do kp=1,np
          concp(kp)=cpbint1(k+1,kp)
        enddo
        reacao1=reac(conca,concr,concp,nr,np,ck)
        v1=nalint*alvint/hlint-cabint(k)*pma/(rol*hlint)*nagint*
     &     alvint-aa*reacao
        v2=nalint1*alvint1/hlint1-cabint1(k+1)*pma/(rol*hlint1)*
     &     nagint1*alvint1-aa*reacao1
        valor=cabint(k)+(v1+v2)*dt/2.
        if (dabs(cabint1(k+1)).ge.1.d-8) then
          if (dabs(valor-cabint1(k+1))/cabint1(k+1).gt.great)
     &      great=dabs(valor-cabint1(k+1))/cabint1(k+1)
        else
          if (dabs(valor-cabint1(k+1)).gt.great)
     &      great=dabs(valor-cabint1(k+1))
        endif
        cabint1(k+1)=valor
        do kr=1,nr
          v1=nrlint*alvint/hlint-crbint(k,kr)*pma/(rol*hlint)*
     &       nagint*alvint-ar(kr)*reacao
          v2=nrlint1*alvint1/hlint1-crbint1(k+1,kr)*pma/(rol*hlint1)*
     &       nagint1*alvint1-ar(kr)*reacao1
          valor=crbint(k,kr)+(v1+v2)*dt/2.
          if (dabs(crbint1(k+1,kr)).ge.1.d-8) then
            if (dabs(valor-crbint1(k+1,kr))/crbint1(k+1,kr).gt.great)
     &        great=dabs(valor-crbint1(k+1,kr))/crbint1(k+1,kr)
          else
            if (dabs(valor-crbint1(k+1,kr)).gt.great)
     &        great=dabs(valor-crbint1(k+1,kr))
          endif
          crbint1(k+1,kr)=valor
        enddo
        do kp=1,np
          v1=nplint*alvint/hlint-cpbint(k,kp)*pma/(rol*hlint)*nagint*
     &       alvint-ap(kp)*reacao
          v2=nplint1*alvint1/hlint1-cpbint1(k+1,kp)*pma/(rol*hlint1)*
     &       nagint1*alvint1-ap(kp)*reacao1
          valor=cpbint(k,kr)+(v1+v2)*dt/2.
          if (dabs(cpbint1(k+1,kp)).ge.1.d-8) then
            if (dabs(valor-cpbint1(k+1,kp))/cpbint1(k+1,kp).gt.great)
     &        great=dabs(valor-cpbint1(k+1,kp))/cpbint1(k+1,kp)
          else
            if (dabs(valor-cpbint1(k+1,kp)).gt.great)
     &        great=dabs(valor-cpbint1(k+1,kp))
          endif
          cpbint1(k+1,kp)=valor
        enddo
        soma1=0.d0
```

```
        do kr=1,nr
          somal=somal+pmr(kr)/densr(kr)*(crbintl(k+1,kr)-crbint(k,kr))
        enddo
        soma2=0.d0
        do kp=1,np
          soma2=soma2+pmp(kp)/densp(kp)*(cpbintl(k+1,kp)-cpbint(k,kp))
        enddo
        vlintl(k+1)=vlint(k)*(1+pma/densa*(cabintl(k+1)-cabint(k))+
     &                somal+soma2)
        valor=zl(k)+(vlint(k)/(hlint*area)+vlintl(k+1)/
     &         (hlintl*area))*dt/2.d0
        if (dabs(zl(k)).ge.1.d-8) then
            if (dabs((valor-zll(k+1))/zl(k)).gt.great)
     &        great=dabs((valor-zll(k+1))/zl(k))
          else
            if (dabs((valor-zll(k+1))).gt.great)
     &        great=dabs((valor-zll(k+1)))
        endif

        zll(k+1)=valor
        if (zll(k+1).ge.ht) then
            kmaxll=k+1
          else
            k=k+1
            goto 11
        endif
       if (great.gt.1.d-04) then
         goto 29
       endif
c***
c***  Passagem dos valores calculados para os pontos de colocacao
c***
       ft(net,ndt)=fin
       pab(net,ndt)=pabin
       l=net
       j=ndt
       k1=1
       k2=kmaxll
    31 continue
         pos=(xqt(l)+dxt(l)*xt(j))*ht
    32   continue
         if ((pos.ge.zll(k2-1)).and.(pos.le.zll(k2))) then
             fator=(pos-zll(k2-1))/(zll(k2)-zll(k2-1))
             vl(l,j)=vlintl(k2-1)+(vlintl(k2)-vlintl(k2-1))*fator
             cab(l,j)=cabintl(k2-1)+(cabintl(k2)-cabintl(k2-1))*fator
             do kr=1,nr
               crb(l,j,kr)=crbintl(k2-1,kr)+(crbintl(k2,kr)-
     &                     crbintl(k2-1,kr))*fator
             enddo
             do kp=1,np
               cpb(l,j,kp)=cpbintl(k2-1,kp)+(cpbintl(k2,kp)-
     &                     cpbintl(k2-1,kp))*fator
             enddo
           else
             if (pos.lt.zll(k2)) then
                 k2=k2-1
               else
                 k2=k2+1
             endif
             goto 32
         endif
    33   continue
         if ((pos.le.zgl(k1)).and.(pos.ge.zgl(k1+1))) then
             fator=(pos-zgl(k1))/(zgl(k1+1)-zgl(k1))
             ft(l,j)=fintl(k1)+(fintl(k1+1)-fintl(k1))*fator
             pab(l,j)=pabintl(k1)+(pabintl(k1+1)-pabintl(k1))*fator
           else
             if (pos.gt.zgl(k1)) then
                 k1=k1-1
               else
                 k1=k1+1
             endif
             goto 33
         endif
         if (j.ne.1) then
             j=j-1
           else
             l=l-1
             j=ndt
         endif
       if (l.ne.0) goto 31
```

```
c***
c***  Atualizacao das variaveis para o proximo looping de integracao
c***
      do i=1,kmaxg1
        fint(i)=fint1(i)
        pabint(i)=pabint1(i)
        zg(i)=zg1(i)
      enddo
      kmaxg=kmaxg1
      do i=1,kmax1
        vlint(i)=vlint1(i)
        cabint(i)=cabint1(i)
        do kr=1,nr
          crbint(i,kr)=crbint1(i,kr)
        enddo
        do kp=1,np
          cpbint(i,kp)=cpbint1(i,kp)
        enddo
        zl(i)=zl1(i)
      enddo
      kmax1=kmax11
c***
c***  Passagem dos valores de saida (composicao do gas e de produto)
c***  para dados a serem utilizados na predicao por RNA
c***
      cpbout(1)=cpb(net,ndt,1)
      pabout=pab(1,1)
c***
c***  Verifica se eh tempo de amostrar
c***
      tm=t*60
      resto=dabs(dmod(tm,tamost))
      difer=dabs(tamost-resto)
      if ((resto.lt.0.5*(dt*60)).or.(difer.lt.0.5*(dt*60))) then

c***  Chamada a sub-rotina que faz a coleta de dados para as RNA's
        call coleta(ifalha)

c***  Chama a sub-rotina de controle (se houver dados suficientes
c***  para a predicao)
        if (ifalha.ne.-1) then
          mode=1
          set=7.15d-4
          vlflood=0.8/1.05d0*flood(ft(net,ndt))
          if (controla) then
            call controle(ichama,vlin,dvmax,ncflag)
          endif
          nchama=0
        endif

c***  Escreve resultados de saida para arquivo a cada "tamost"
c***  intervalo de tempo

        if ((resto.lt.0.5*(dt*60)).or.(difer.lt.0.5*(dt*60))) then
          write(16,72) tm,pab(1,1),pab(net,ndt),ft(net,ndt),vl(1,1),
     &                 cpb(1,1,1),cpb(net,ndt,1)
        endif
      endif

c***  Calculo do erro integral medio (IEA)

      if (ifalha.ne.-1) call calciea(pabout,t,iea,set)

c***
c***  Escreve perfil na tela a cada "ti" intervalos de tempo
c***
      resto=dabs(dmod(tm,ti))
      difer=dabs(tamost-resto)
      if ((resto.lt.0.5*(dt*60)).or.(difer.lt.0.5*(dt*60))) then
        write(*,*) '************************************************'
        do l=1,net
          do j=1,ndt-1
            write(*,56) ft(l,j),pab(l,j),vl(l,j),crb(l,j,1),cpb(l,j,1)
          enddo
          if (l.eq.net) then
            j=ndt
            write(*,56) ft(l,j),pab(l,j),vl(l,j),crb(l,j,1),cpb(l,j,1)
            write(*,*) '************************************************'
            tm=t*60.
            write(*,57) tm
          endif
```

```
        enddo
      endif
   56 format(3x,f6.2,3x,e10.4,3x,f5.2,3x,f5.3,3x,f5.3)
   57 format(3x,'     Tempo simulado= ',f5.2,'min.')

*************************************************************************
*                 Fim do laco de simulacao e controle                   *
*************************************************************************
      t=t+h
      if (t.lt.e) goto 10
      close(16)
      tempo=e*6.d1
      write(*,15) tempo
   15 format('Tempo total simulado: ', f5.2,'min.')
      stop
      end
c------------------------fim do programa principal---------------------

c-------------------------------------------------------------------------
c     Sub-rotinas de calculo de parametros e valores do modelo e
c     sub-rotinas auxiliares (MODIFICA, FLUX, HOLD, HG, REAC, ALV,
c     FLOOD E GMOLHA)
c-------------------------------------------------------------------------

      subroutine modifica(fin,pabin,vlin,cabin,crbin,cpbin,flag,npfag)
      implicit double precision(a-h,o-z)
      integer flag,tvar
      parameter (nre=1,npr=2,nps=10)
      dimension crbin(nre),cpbin(npr),tempo(nps),nvar(nps),pulso(nps),
     &k(nps),npfag(2),tvar(nps)
      common /tempo/ t,h
      common /dadomod/ nmod,nvar
      common /dadomoda/ tvar,tempo,pulso,k
c
c     se for a primeira chamada a sub-rotina, ler arquivo de dados
c
      if (flag.eq.1) then
        open(20,file='modifica.dat',status='old')
        read(20,*) nmod
        if (nmod.ne.0) then
          do i=1,nmod
            read(20,*) tvar(i),tempo(i),nvar(i),k(i),pulso(i)
          enddo
        endif
        close(20)
        flag=0
      endif
      npfag(1)=0
      npfag(2)=0
      do i=1,nmod
        if (dabs(tempo(i)-t).lt.0.5*h) then
          if ((nvar(i).eq.1).or.(nvar(i).eq.2)) then
              npfag(1)=1
            else
              npfag(2)=1
          endif
          goto (1,2,3,4,5,6), nvar(i)
    1     continue
            if (tvar(i).eq.0) fin=fin*(1.+pulso(i))
            if (tvar(i).eq.1) fin=fin+pulso(i)
            if (tvar(i).eq.2) fin=pulso(i)
          goto 7
    2     continue
            if (tvar(i).eq.0) pabin=pabin*(1.+pulso(i))
            if (tvar(i).eq.1) pabin=pabin+pulso(i)
            if (tvar(i).eq.2) pabin=pulso(i)
          goto 7
    3     continue
            if (tvar(i).eq.0) vlin=vlin*(1.+pulso(i))
            if (tvar(i).eq.1) vlin=vlin+pulso(i)
            if (tvar(i).eq.2) vlin=pulso(i)
          goto 7
    4     continue
            if (tvar(i).eq.0) cabin=cabin*(1.+pulso(i))
            if (tvar(i).eq.1) cabin=cabin+pulso(i)
            if (tvar(i).eq.2) cabin=pulso(i)
          goto 7
    5     continue
            if (tvar(i).eq.0) crbin(k(i))=crbin(k(i))*(1.+pulso(i))
            if (tvar(i).eq.1) crbin(k(i))=crbin(k(i))+pulso(i)
            if (tvar(i).eq.2) crbin(k(i))=pulso(i)
```

```
          goto 7
    6     continue
            if (tvar(i).eq.0) cpbin(k(i))=cpbin(k(i))*(1.+pulso(i))
            if (tvar(i).eq.1) cpbin(k(i))=cpbin(k(i))+pulso(i)
            if (tvar(i).eq.2) cpbin(k(i))=pulso(i)
          goto 7
    7 continue
        endif
      enddo
      return
      end
c-----------------------fim da sub-rotina modifica--------------------

      double precision function flux(n)
      implicit double precision(a-h,o-z)
      double precision inp
      external rna
      parameter (nl=32,nm=22,nn=6,nmp=30)
      dimension inp(nmp,nl),des(nmp,nn),res(nmp,nn),par(1),pvv(4,nl,nm),
     &pwv(4,nm,nn),carv(4,nl+nn),cbrv(4,nl+nn),ntav(4,nl+nn),pv(nl,nm),
     &pw(nm,nn),car(nl+nn),cbr(nl+nn),nta(nl+nn),info(8),infov(4,8)
      common /reder/ inp,des,res,par,pvv,pwv,carv,cbrv,pv,pw,car,cbr
      common /redei/ ntav,nta
      common /redinfo/ infov
c***  Carrega pesos da rede treinada
c      info(1)=infov(n,1)
      info(1)=1
      info(2)=1
c      infov(n,2)
      info(3)=infov(n,3)
      info(4)=infov(n,4)
      info(5)=infov(n,5)
      info(6)=1
c      infov(n,6)
      do i=1,info(3)
        do j=1,info(4)
          pv(i,j)=pvv(n,i,j)
        enddo
      enddo
      do j=1,info(4)
        do kr=1,info(5)
          pw(j,kr)=pwv(n,j,kr)
        enddo
      enddo
      do i=1,info(3)+info(5)
        car(i)=carv(n,i)
        cbr(i)=cbrv(n,i)
        nta(i)=ntav(n,i)
      enddo
      call rna(info,tol,tolp,inp,res,nta,pv,pw,car,cbr,par)
      flux=res(1,1)
      return
      end
c-----------------------fim da sub-rotina flux-------------------------

      double precision function hold(vazao)
      implicit double precision(a-h,o-z)
      double precision mil
      common /proptor/ av,rol,mil,area
      common /propliq/ sigmac,sigma,d
      common /proprec/ dp,epslon
      g=9.81*3600.**2.
      vazaol=vazao/area
      dpe=6*(1-epslon)/av
      fr=vazaol**2./(g*epslon*dpe)
      hold=fr**(1./3)
      return
      end
c-----------------------fim da sub-rotina hold-------------------------

      double precision function hg(vl)
      implicit double precision (a-h,o-z)
      common /const/temp,r
      common /gastor/pt
      common /proprec/ dp,epslon
      hg=(epslon-hold(vl))*pt/(r*temp)
      return
      end
c-----------------------fim da sub-rotina hg---------------------------

c!!!
```

```
c!!!  A sub-rotina a seguir deve ser modificada caso se utilize outra
c!!!  taxa de reacao
c!!!
      double precision function reac(conca,concr,concp,nr,np,ck)
      implicit double precision(a-h,o-z)
      dimension concr(nr),concp(np)
      reac=ck*conca
      do n=1,nr
        reac=reac*concr(n)
      enddo
      do n=1,np
        reac=reac*concp(n)
      enddo
      return
      end
c-----------------------fim da sub-rotina reac-------------------------

      double precision function alv(vl)
      implicit double precision(a-h,o-z)
      double precision mil,vazaol
      common /proptor/ av,rol,mil,area
      common /propliq/ sigmac,sigma,d
      g=9.81*3600.**2
      vazaol=vl*rol/area
      sig=sigma*3600.**2*1.d-03
      alv=av*(1-exp(-1.45*(sigmac/sigma)**0.75*(vazaol/(av*mil))**0.1*
     &(vazaol**2*av/(rol**2*g))**(-0.05)*(vazaol**2/(rol*sig*av))**0.2))
      return
      end
c-----------------------fim da sub-rotina alv--------------------------

      double precision function flood(ft)
      implicit none
      double precision ft,g,av,rol,rog,mil,area,dp,epslon,temp,r,pt,
     &raiz,fator1,fator2
      common /proptor/ av,rol,mil,area
      common /proprec/ dp,epslon
      common /const/ temp,r
      common /gastor/ pt
      common /propgasa/ rog
      g=(ft*r*temp/pt/area)
      raiz=((av/epslon**3)*mil**0.2)**0.5
      fator1=12.16734d0
      fator2=1.d0/28.167925845d0
      flood=(18.91-(fator1*g/((rog*rol)**0.5)*raiz)**(1./3.))**2/
     &    (raiz*fator2)*area
      return
      end
c-----------------------fim da sub-rotina flood------------------------

      subroutine gmolha(n,vl,grau)
      implicit none
      integer n
      double precision alv,vl,vlmin,vlmin1,erro,tol,grau
      external alv
c***  Esta sub-rotina calcula a vazao minima de liquido necessaria p/
c***  manter o grau de molhabilidade minimo (n=0) ou o grau de
c***  molhabilidade a partir de uma determinada vazao de liquido (n=1)
      if (n.eq.0) then
          vlmin=70.d0
          tol=1.d-7
    1     continue
            vlmin1=0.8*alv(vlmin)
            erro=dabs((vlmin1-vlmin)/vlmin1)
            vlmin=vlmin1
          if (erro.gt.tol) goto 1
          vl=vlmin
          grau=0.8d0
        else
          grau=vl/alv(vl)
      endif
      return
      end
c-----------------------fim da sub-rotina gmolha-----------------------

c--------------------------------------------------------------------------
c     Sub-rotinas de controle (CONTROLE, OBJFUN, OBJFUN1, COLETA e
c     CALCIEA)
c--------------------------------------------------------------------------

      subroutine controle(ichama,vlin,dvmax,ncflag)
```

```
      implicit none
      external objfun,objfun1
      external x04abf,e04udf
      double precision vlin,vlmin,vlmax,vlflood,fator1,
     &fator2,dvmax
      integer ihc,ipred,iuser,i,j,ncflag,ichama,njan,ninp,ns,imodcont
c***  Variaveis utilizadas pela sub-rotina ucf
      integer n,nclin,ncnln,nrowa,nrowj,nrowr,iter,istate,liwork,iwork,
     &lwork,ifail,nout,iclin
      parameter (n=10,nclin=9,ncnln=0,nrowa=nclin,nrowj=1,nrowr=n)
      parameter (liwork=3*n+nclin+2*ncnln,lwork=2*n*n+20*n+11*nclin)
      double precision a(nrowa,n),bl(n+nclin+ncnln),bu(n+nclin+ncnln),
     &c(1),cjac(nrowj,1),clamda(n+nclin+ncnln),objf,objgrd(n),
     &r(nrowr,n),x(n+1),work(lwork),user(1),bigbnd
      dimension istate(n+nclin+ncnln),iwork(liwork)
      dimension iuser(1)
      common /limite/ vlmin,vlmax,vlflood
      common /normaliza/ fator1,fator2
      common /modelo/ njan,ninp,ns
      common /paracont/ ihc,ipred,imodcont

*     Verifica se chamada eh .le. njan/2 (Valor a ser usado na correcao
*     da predicao na sub-rotina de calculo da funcao objetivo

      iuser(1)=ichama

*     Valores utilizados na normalizacao das entradas para a otimizacao

      fator1=100.d0
      fator2=1.d3

*     Define o arquivo de saida das mensagens da sub-rotina e04ucf

      nout=22
      open(22,file='lixao',status='unknown')
      call x04abf(1,nout)
      call x04aaf(1,nout)

*     Define valor 'infinito'.

      bigbnd=1.0D15

*
*     A  = Matriz de restricoes lineares
*     BL = Valores superiores em x,  a'x  and  c(x)
*     BU = Valores inferiores em x,  a'x  and  c(x)
*
      iclin=ihc-1
      do j=1,ihc
        do i=1,iclin
          a(i,j)=0.d0
        enddo
      enddo
      do i=1,iclin
        a(i,i)=1.d0
        a(i,i+1)=-1.d0
      enddo
      do i=1,ihc
        bl(i)=vlmin/fator1
        if (vlflood.lt.vlmax) then
            bu(i)=vlflood/fator1
          else
            bu(i)=vlmax/fator1
        endif
      enddo
      do i=ihc+1,ihc+iclin
        bl(i)=-dvmax/fator1
        bu(i)=dvmax/fator1
      enddo

*
*     x = Estimativas iniciais das solucoes

      do i=1,ihc
        x(i)=vlin/fator1*(1.d0+i/1.d2)
      enddo
      x(ihc+1)=vlin/fator1

*     Solucao do problema de otimizacao
      ifail = -1
```

```
*     Define opcao de gradiente nao disponivel
      call e04uef('   Derivative level              0')
      call e04uef('   Print Level =                 1')

      if (imodcont.eq.0) then
        call e04ucf(ihc,nclin,ncnln,nrowa,nrowj,nrowr,a,bl,bu,e04udf,
     &              objfunl,iter,istate,c,cjac,clamda,objf,objgrd,r,x,
     &              iwork,liwork,work,lwork,iuser,user,ifail)
      else
        call e04ucf(ihc,nclin,ncnln,nrowa,nrowj,nrowr,a,bl,bu,e04udf,
     &              objfun,iter,istate,c,cjac,clamda,objf,objgrd,r,x,
     &              iwork,liwork,work,lwork,iuser,user,ifail)
      endif
*      close()
*     Se a sub-rotina de otimizacao retornou alguma falha, escreve esta
*     na tela
      if (ifail.ne.0) write(*,*) ' Ifail retornou valor ',ifail
      if (dabs(vlin-x(1)*fator2).gt.0.005*vlin) then
          vlin=x(1)*fator1
          ncflag=1
        else
          ncflag=0
      endif
      close(22)
      if (ichama.le.njan/2+1) then
          ichama=ichama+1
        else
          ichama=ichama-1
      endif
      return
      end
c-----------------------fim da sub-rotina controle---------------------

      subroutine objfun(mode,n,x,objf,objgrd,nstate,iuser,user)
      implicit double precision (a-h,o-z)
      external rna
      integer mode,n,nstate,iuser(1)
      double precision x(n+1),objf,objgrd(n),user(1),inp
      integer i,nn,p,ichama
      parameter (nl=32,nm=22,nn=6,nmp=30)
      parameter (ndm=20,nen=10,nhc=40)
      double precision soma,set,fator1,fator2
      dimension ent(ndm,nen),ent1(ndm,nen)
      dimension inp(nmp,nl),des(nmp,nn),res(nmp,nn),par(1),pvv(4,nl,nm),
     &pwv(4,nm,nn),carv(4,nl+nn),cbrv(4,nl+nn),ntav(4,nl+nn),pv(nl,nm),
     &pw(nm,nn),car(nl+nn),cbr(nl+nn),nta(nl+nn),info(8),infov(4,8),
     &vl(nhc),yp(ndm,ndm),yc(ndm),
     &dy(ndm)
      common /modelo/ njan,ni,ns
      common /reder/ inp,des,res,par,pvv,pwv,carv,cbrv,pv,pw,car,cbr
      common /redei/ ntav,nta
      common /redinfo/ infov
      common /mdado/ ent
      common /normaliza/ fator1,fator2
      common /paracont/ ihc,ipred,imodcont
      common /setpoint/ set
      save yp
      save ichama
c***
      if (iuser(1).le.njan/2) then
          ichama=iuser(1)
        else
          if (ichama.ne.iuser(1)) then
            do i=1,njan
              do j=2,njan/2+1
                yp(i,j-1)=yp(i,j)
              enddo
            enddo
            ichama=iuser(1)
          endif
      endif
c***  Carrega saidas da rotina de otimizacao
      do i=1,ihc
        vl(i)=fator1*x(i)
      enddo
c***  Penalizacao das acoes de controle
      q=1.d3
c***  Informacoes iniciais para a rede neural artificial
      info(1)=1
      info(2)=1
      info(3)=infov(4,3)
```

```
      info(4)=infov(4,4)
      info(5)=infov(4,5)
      info(6)=1
c***  Carrega pesos da rede treinada
      do i=1,info(3)
        do j=1,info(4)
          pv(i,j)=pvv(4,i,j)
        enddo
      enddo
      do j=1,info(4)
        do kr=1,info(5)
          pw(j,kr)=pwv(4,j,kr)
        enddo
      enddo
      do i=1,info(3)+info(5)
        car(i)=carv(4,i)
        cbr(i)=cbrv(4,i)
        nta(i)=ntav(4,i)
      enddo
c***  Calcula as correcoes do valor predito pelo modelo de RNA
      if (ichama.lt.njan/2+1) then
          do i=1,ichama
            dy(i)=ent(njan,ni-ns+1)-yp(i,ichama-i+1)
            dy(i+njan/2)=ent(njan,ni)-yp(i+njan/2,ichama-i+1)
          enddo
          do i=ichama+1,njan/2
            dy(i)=0.d0
            dy(i+njan/2)=0.d0
          enddo
        else
          do i=1,njan/2
            dy(i)=ent(njan,ni-ns+1)-yp(i,njan/2-i+1)
            dy(i+njan/2)=ent(njan,ni)-yp(i+njan/2,njan/2-i+1)
          enddo
      endif
c***  Carrega dados do vetor de COLETA em vetor interno
      do k=1,ni
        do i=1,njan/2
          ent1(i,k)=ent(i+njan/2,k)
        enddo
        do i=njan/2+1,njan
          ent1(i,k)=ent(njan,k)
        enddo
      enddo
c***  Carrega dados de vazao de liquido em vetor interno
      k=ni-ns
      if (ipred.le.njan/2+1) then
          do i=1,ihc
            ent1(i+njan/2-1,k)=vl(i)
          enddo
          do i=ihc+1,njan/2+1
            ent1(i+njan/2-1,k)=vl(ihc)
          enddo
        else
          do i=njan/2+1,njan+1
            ent1(i-1,k)=vl(i-njan/2)
          enddo
      endif
      soma=0.d0
      do p=1,ipred,njan/2
c***    Desloca janela de tempo "njan/2" instantes a frente
        if (p.ne.1) then
          do k=1,ni-ns
            do i=njan/2+1,njan
              ent1(i-njan/2,k)=ent1(i,k)
            enddo
          enddo
          do k=1,ni-ns-1
            do i=njan/2+1,njan
              ent1(i,k)=ent1(njan,k)
            enddo
          enddo

c***    Passa as outras variaveis manipuladas se ihc > njan/2
          k=ni-ns
          if (ipred.gt.p) then
              ip=mod(ihc,p)
              if (ipred/(p).gt.1) ip=njan/2
              do i=njan/2+1,njan/2+ip
                ent1(i,k)=vl(p+i-njan/2)
              enddo
```

```
              do i=njan/2+ip+1,njan
                ent1(i,k)=vl(p+ip)
              enddo
            else
              do i=njan/2+1,njan
                ent1(i,k)=ent1(njan,k)
              enddo
          endif
c***    Variaveis de saida
          do k=ni-ns+1,ni
            kl=k-ni+ns
            do i=njan/2+1,njan
              il=i-njan/2
              ent1(il,k)=yc((kl-1)*(njan/2)+il)
            enddo
          enddo
        endif
c***    Passa dados para vetor de entrada
        do k=1,ni-ns
          do i=1,njan
            inp(1,(k-1)*njan+i)=ent1(i,k)
          enddo
        enddo
        do k=ni-ns+1,ni
          do i=1,njan/2
            inp(1,(ni-ns)*njan+(ns-ni+k-1)*(njan/2)+i)=ent1(i,k)
          enddo
        enddo
        call rna(info,tol,tolp,inp,res,nta,pv,pw,car,cbr,par)
        do i=1,njan
          yc(i)=res(1,i)+dy(i)
        enddo

c***    Calculo da funcao objetivo
        do i=1,njan/2
          soma=soma+(fator2*(set-yc(i)))**2
        enddo
        if (p.eq.1) then
          if (ichama.lt.njan/2) then
              do i=1,njan
                yp(i,ichama+1)=res(1,i)
              enddo
            else
              do i=1,njan
                yp(i,njan/2+1)=res(1,i)
              enddo
          endif
        endif
      enddo
      soma=soma+q*(x(1)-x(ihc+1))**2
      do i=2,ihc
        soma=soma+q*(x(i)-x(i-1))**2
      enddo
      objf=soma
      return
      end
c-----------------------fim da sub-rotina objfun-----------------------

      subroutine objfun1(mode,n,x,objf,objgrd,nstate,iuser,user)
      implicit double precision(a-h,o-z)
      integer mode,n,nstate,iuser(1),ichama
      double precision x(n+1),objf,objgrd(n),user(1)
      parameter (ndm=20,nen=10,nhc=40)
      double precision soma,set,fator1,fator2,q
      dimension ent(ndm,nen)
      dimension vl(nhc),q(nhc)
      double precision a,h,du,s,p,yc
      integer R,L,nmodel,nent
      integer i,j,k,ne,ndim
      parameter (ndim=30,ne=3)
      dimension a(ne,ndim),h(ne,ndim),du(ne,2*ndim),s(ne,ndim),
     &p(ne,ndim),yc(ndim)
      common /normaliza/ fator1,fator2
      common /paracont/ ihc,ipred,imodcont
      common /setpoint/ set
      common /coefconv/ a,h
      common /intconv/ nent,nmodel
      common /mdado/ ent
      common /modelo/ njan,ni,ns
      save ichama
      save du
```

```
c***  Variaveis de entrada da sub-rotina
      R=nmodel/2
      L=ihc
c***  Penalizacao das acoes de controle
      do i=1,ihc
        q(i)=1.d3
      enddo
c***  Valor inicial das variaveis manipuladas
      if (iuser(1).eq.0) then
          do k=1,nent
            do i=1,nmodel
              du(k,i)=0.d0
            enddo
            if (k.ne.1) then
              do i=nmodel+1,nmodel+ihc
                du(k,i)=0.d0
              enddo
            endif
          enddo
          ichama=iuser(1)
      endif
c***  Rola horizonte de tempo
      if (ichama.ne.iuser(1)) then
        do k=1,nent
          do i=2,nmodel
            du(k,i-1)=du(k,i)
          enddo
        enddo
        ichama=iuser(1)
      endif
c***  Carrega valores da sub-rotina de otimizacao
      vl(ihc+1)=x(ihc+1)
      do i=1,ihc
        vl(i)=x(i)
        if (i.eq.1) then
            du(1,nmodel+i-1)=(vl(i)-vl(ihc+1))
          else
            du(1,nmodel+i-1)=(vl(i)-vl(i-1))
        endif
      enddo
      du(2,nmodel)=(ent(njan,1)-ent(njan-1,1))/2.d0
      du(3,nmodel)=(ent(njan,2)-ent(njan-1,2))/500.d0
c***  Calculo dos S's
      do k=1,nent
        do i=1,R
          soma=0.d0
          do j=i+1,nmodel
            soma=soma+h(k,j)*du(k,nmodel+i-j)
          enddo
          s(k,i)=soma
        enddo
c***  Calculo dos P's
        do i=1,R
          soma=0.d0
          do j=1,i
            soma=soma+s(k,j)
          enddo
          p(k,i)=soma
        enddo
      enddo
c***  Calculo dos yc(k+j), j=1,R
      do i=1,R
        soma=0.d0
        do k=1,nent
          do j=1,i
            if (i-j.le.ipred-1) soma=soma+a(k,j)*du(k,nmodel+i-j)
          enddo
          soma=soma+p(k,i)
        enddo
        yc(i)=ent(njan,4)+soma
      enddo
      soma=0.d0
      do i=1,R
        soma=soma+(fator2*(set-yc(i)))**2
      enddo
      soma=soma+q(1)*(x(1)-x(ihc+1))**2
      do i=2,ihc
        soma=soma+q(i)*(x(i)-x(i-1))**2
      enddo
      objf=soma
      return
```

```
      end
c-----------------------fim da sub-rotina objfun1----------------------

      subroutine coleta(ifalha)
      implicit double precision(a-h,o-z)
      parameter (ndm=20,nen=10)
      parameter (npr=2)
      dimension ent(ndm,nen),cpbin(npr),cpbout(npr)
      common /modelo/ njan,ni,ns
      common /mdado/ ent
      common /mentsai/ fin,pabin,vlin,cpbin,cpbout,pabout
      save kdados
      ndados=njan
      if (ifalha.le.-2) kdados=1
      ifalha=0
c***  Passagem dos dados do simulador para vetor interno
        if (kdados.eq.ndados+1) then
            do j=2,ndados
              do i=1,ni
                ent(j-1,i)=ent(j,i)
              enddo
            enddo
            ent(ndados,1)=fin
            ent(ndados,2)=pabin
            ent(ndados,3)=vlin
            ent(ndados,4)=pabout
            ent(ndados,5)=cpbout(1)
          else
            ent(kdados,1)=fin
            ent(kdados,2)=pabin
            ent(kdados,3)=vlin
            ent(kdados,4)=pabout
            ent(kdados,5)=cpbout(1)
        endif
        if (kdados.lt.ndados+1) then
          kdados=kdados+1
          ifalha=-1
        endif
      return
      end
c-----------------------fim da sub-rotina coleta-----------------------

      subroutine calciea(pabout,t,iea,setpoint)
      double precision pabout,iea,t,setpoint,pabout1,t1,setpoint1,h,h1
      integer chamada
      save pabout1,setpoint1,t1
c***  Verifica se eh a primeira chamada a sub-rotina
      if (iea.lt.0.d0) then
          chamada=0
          iea=0.d0
        else
          chamada=1
      endif
c***  Calculo da integral do erro medio pela regra do trapezoide
      if (chamada.ne.0) then
        h1=dabs(pabout1-setpoint1)
        h=dabs(pabout-setpoint)
        iea=iea+(h1+h)/2*(t-t1)
      endif
      pabout1=pabout
      setpoint1=setpoint
      t1=t
      return
      end
c-----------------------fim da sub-rotina calciea----------------------

c-----------------------------------------------------------------------
c     Sub-rotinas de treinamento, predicao e calculo do erro de predicao
c     de Redes Neurais Artificias diretas com uma camada intermediaria.
c     A sub-rotina RNA gerencia as outras tres (TREIN1, TREIN2, PRED,
c     RNA)
c-----------------------------------------------------------------------

c***  Esta sub-rotina faz o treinamento de uma rede neural artificial
c***  usando o algoritmo backpropagtion.
c***  Dados de entrada: Topologia da rede (l,m,n)
c***                    Dados de treinamento (inp,d,npt)
c***                    Parametros de treinamento (eta,alp)
c***                    Criterios de convergencia (tol,tolp)
c***                    Tipo de adimensionalizacao (nta)
c***  Dados de saida: Pesos da rede treinada (v,w)
```

```
      subroutine treinl(l,m,n,npt,inp,d,eta,alp,tol,tolp,nta,v,w,ca,cb)
      implicit none
      integer i,j,k,nl,nm,nn,np,l,m,n,p,npt,nta
      double precision soma,erro,tol,tolp,eta,alp,errp,grande,v1,v2
      double precision v,w,a,b,c,inp,x1,x2,x3,d,th1,th2,th3,dif,delt2,
     &delt3,dv,dw,ca,cb
      parameter (nl=40,nm=40,nn=8,np=30)
      dimension v(nl,nm),w(nm,nn),a(nl),b(nm),c(nn),x1(nl),x2(nm),
     &x3(nn),th1(nl),th2(nm),th3(nn),inp(np,*),d(np,*),dif(nn,np),
     &delt2(nm,np),delt3(nn,np),dv(nl,nm),dw(nm,nn),ca(nl+nn),cb(nl+nn),
     &nta(nl+nn)

c***  Supoe-se que ja haja um valor inicial para os pesos e que seja
c***  conhecida a tabela para treinamento (inputs 'inp' e valores
c***  dos outputs desejados 'd')
c***
c***  Inicializacao dos fatores de ativacao

      do i=1,l
        th1(i)=0.d0
      enddo
      do j=1,m
        th2(j)=1.d0
      enddo
      do k=1,n
        th3(k)=1.d0
      enddo

c***  Inicializacao dos dv's e dw's

      do i=1,l
        do j=1,m
          dv(i,j)=0.d0
        enddo
      enddo
      do j=1,m
        do k=1,n
          dw(j,k)=0.d0
        enddo
      enddo

c***  Inicio do looping de treinamento da rede

    1 continue

c***  Calculo do erro quadratico
      errp=0.d0
      erro=0.d0
      do p=1,npt
        do i=1,l
          x1(i)=inp(p,i)
          a(i)=1.d0/(1.d0+exp(-x1(i)))
        enddo
        do j=1,m
          soma=0.d0
          do i=1,l
            soma=soma+a(i)*v(i,j)
          enddo
          x2(j)=soma+th2(j)
          b(j)=1.d0/(1.d0+exp(-x2(j)))
        enddo
        do k=1,n
          soma=0.d0
          do j=1,m
            soma=soma+b(j)*w(j,k)
          enddo
          x3(k)=soma+th3(k)
          c(k)=1.d0/(1.d0+exp(-x3(k)))
          dif(k,p)=d(p,k)-c(k)
c***  Calculo do erro percentual
          if (nta(k+l).eq.0) then
              v1=(c(k)-ca(k+l))/cb(k+l)
              v2=(d(p,k)-ca(k+l))/cb(k+l)
            else
              v1=10**(ca(k+l)+cb(k+l)*c(k))
              v2=10**(ca(k+l)+cb(k+l)*d(p,k))
          endif
          if (v2.gt.1.d-15) then
              grande=100*dabs((v1-v2)/v2)
            else
```

```
              grande=100*dabs(v1)
            endif
            if (errp.lt.grande) errp=grande
            erro=erro+dif(k,p)**2

c***  Calculo dos gradientes
            delt3(k,p)=dif(k,p)*(exp(-x3(k))/(1.d0+exp(-x3(k)))**2.)
          enddo
          do j=1,m
            soma=0.d0
            do k=1,n
              soma=soma+delt3(k,p)*w(j,k)
            enddo
            delt2(j,p)=soma*(exp(-x2(j))/(1.d0+exp(-x2(j)))**2.)
          enddo

c*** Calculo das variacoes dos pesos

          do i=1,l
            do j=1,m
              dv(i,j)=eta*delt2(j,p)*a(i)+alp*dv(i,j)
            enddo
          enddo
          do j=1,m
            do k=1,n
              dw(j,k)=eta*delt3(k,p)*b(j)+alp*dw(j,k)
            enddo
          enddo

c***  Atualizacao dos pesos

          do i=1,l
            do j=1,m
              v(i,j)=v(i,j)+dv(i,j)
            enddo
          enddo
          do j=1,m
            do k=1,n
              w(j,k)=w(j,k)+dw(j,k)
            enddo
          enddo
        enddo
        erro=(erro/npt)**0.5d0

c***  Verifica se foi atingida convergencia
        write(*,*) erro,'   ',errp
        eta=eta/1.0001
        if ((erro.gt.tol).and.(errp.gt.tolp)) go to 1
        return
        end
c-------------------------fim da sub-rotina treinl-------------------------

c***  Esta sub-rotina faz o treinamento da rede utilizando o algoritmo
c***  de Marquardt-Levenberg.
c***  Dados de entrada: Topologia da rede (l,m,n)
c***                    Dados de treinamento (inp,d,npt)
c***                    Parametros de treinamento (mi,beta)
c***                    Criterios de convergencia (tol,tolp)
c***                    Tipo de adimensionalizacao (nta)
c***  Dados de saida: Pesos da rede treinada (v,w)

        subroutine trein2(l,m,n,npt,inp,d,mi,beta,tol,tolp,nta,v,w,ca,cb)
        implicit none
        integer i,j,k,nl,nm,nn,np,l,m,n,p,npt,nta,i1,j1,k1,ncj,nlj
        double precision soma,erro,tol,tolp,mi,beta,errp,grande,v1,v2
        double precision v,w,a,b,c,inp,x1,x2,x3,d,th1,th2,th3,dif,delt2,
       &delt3,ca,cb,erro1,jac,prod1,prod2,prod3,aux,e,ve1,ve2
        parameter (nl=40,nm=40,nn=8,np=30)
        dimension v(nl,nm),w(nm,nn),a(nl),b(nm),c(nn),x1(nl),x2(nm),
       &x3(nn),th1(nl),th2(nm),th3(nn),inp(np,*),d(np,*),dif(nn,np),
       &delt2(nm,np),delt3(nn,np),ca(nl+nn),cb(nl+nn),nta(nl+nn),
       &erro(np,nn),erro1(np,nn),jac(nn*np,nl*nm+nm*nn),
       &prod1(nl*nm+nm*nn,nl*nm+nm*nn),aux(nl*nm+nm*nn,2*(nl*nm+nm*nn)),
       &e(nn*np),prod2(nl*nm+nm*nn),prod3(nl*nm+nm*nn)

c***  Supoe-se que ja haja um valor inicial para os pesos e que seja
c***  conhecida a tabela para treinamento (inputs 'inp' e valores
c***  dos outputs desejados 'd')
c***

c***  Inicializacao dos fatores de ativacao
```

```
      do i=1,l
        th1(i)=0.d0
      enddo
      do j=1,m
        th2(j)=1.d0
      enddo
      do k=1,n
        th3(k)=1.d0
      enddo

c***  Inicio do treinamento

    1 continue

c***  Calculo do vetor erro e do jacobiano (de(m)/dw(i,j))

      errp=0.d0
      do p=1,npt
        do i=1,l
          x1(i)=inp(p,i)
          a(i)=1.d0/(1.d0+exp(-x1(i)))
        enddo
        do j=1,m
          soma=0.d0
          do i=1,l
            soma=soma+a(i)*v(i,j)
          enddo
          x2(j)=soma+th2(j)
          b(j)=1.d0/(1.d0+exp(-x2(j)))
        enddo
        do k=1,n
          soma=0.d0
          do j=1,m
            soma=soma+b(j)*w(j,k)
          enddo
          x3(k)=soma+th3(k)
          c(k)=1.d0/(1.d0+exp(-x3(k)))
          dif(k,p)=d(p,k)-c(k)
          delt3(k,p)=-(exp(-x3(k))/(1.d0+exp(-x3(k)))**2.)
          erro(p,k)=dif(k,p)
        enddo
        do k=1,n
          do j=1,m
            soma=0.d0
            do k1=1,n
              if (k1.eq.k) soma=soma+delt3(k1,p)*w(j,k1)
            enddo
            delt2(j,p)=soma*(exp(-x2(j))/(1.d0+exp(-x2(j)))**2.)
          enddo
          do i1=1,l
            do j1=1,m
              jac((p-1)*n+k,(i1-1)*m+j1)=delt2(j1,p)*a(i1)
            enddo
          enddo
          do j1=1,m
            do k1=1,n
              if (k1.eq.k) then
                  jac((p-1)*n+k,(j1-1)*n+k1+1*m)=delt3(k1,p)*b(j1)
                else
                  jac((p-1)*n+k,(j1-1)*n+k1+1*m)=0.d0
              endif
            enddo
          enddo
        enddo
      enddo
      vel=0.d0
      do p=1,npt
        do k=1,n
          vel=vel+erro(p,k)**2
        enddo
      enddo
      vel=(vel/npt)**0.5

c***  Passagem dos erros para um vetor coluna

      nlj=npt*n
      ncj=l*m+m*n
      do p=1,npt
        do k=1,n
          e((p-1)*n+k)=erro(p,k)
```

```
        enddo
      enddo

c***  Label para o caso de se ter um aumento do erro

    2 continue

c***  Produto Jt*J+mi*I

      do j=1,ncj
        do i=1,ncj
          soma=0.d0
          do k=1,nlj
            soma=soma+jac(k,j)*jac(k,i)
          enddo
          if (i.eq.j) soma=soma+mi
          aux(j,i)=soma
        enddo
      enddo

c***  Inversao da matriz prod1

      do j=1,ncj
        do i=ncj+1,2*ncj
          aux(j,i)=0.d0
          if (i-ncj.eq.j) aux(j,i)=1.d0
        enddo
      enddo

      call gauss(ncj,ncj,nl*nm+nm*nn,aux)

      do j=1,ncj
        do i=1,ncj
          prod1(j,i)=aux(j,i+ncj)
        enddo
      enddo

c***  Calculo de Jt*E.

      do j=1,ncj
        soma=0.d0
        do k=1,nlj
          soma=soma+jac(k,j)*e(k)
        enddo
        prod2(j)=soma
      enddo

c***  Produto [inv(Jt*J+mi*I)]*[Jt*E]

      do j=1,ncj
        soma=0.d0
        do k=1,ncj
          soma=soma+prod1(j,k)*prod2(k)
        enddo
        prod3(j)=soma
      enddo

c***  Incremento dos pesos

      do il=1,l
        do jl=1,m
          v(il,jl)=v(il,jl)-prod3((il-1)*m+jl)
        enddo
      enddo
      do jl=1,m
        do kl=1,n
          w(jl,kl)=w(jl,kl)-prod3((jl-1)*n+kl+l*m)
        enddo
      enddo

c***  Calculo do erro apos a atualizacao dos pesos

      do p=1,npt
        do i=1,l
          xl(i)=inp(p,i)
          a(i)=1.d0/(1.d0+exp(-xl(i)))
        enddo
        do j=1,m
          soma=0.d0
          do i=1,l
            soma=soma+a(i)*v(i,j)
```

```
          enddo
          x2(j)=soma+th2(j)
          b(j)=1.d0/(1.d0+exp(-x2(j)))
        enddo
        do k=1,n
          soma=0.d0
          do j=1,m
            soma=soma+b(j)*w(j,k)
          enddo
          x3(k)=soma+th3(k)
          c(k)=1.d0/(1.d0+exp(-x3(k)))
          dif(k,p)=d(p,k)-c(k)
          errol(p,k)=dif(k,p)
c***  Calculo do erro percentual
          if (nta(k+1).eq.0) then
              v1=(c(k)-ca(k+1))/cb(k+1)
              v2=(d(p,k)-ca(k+1))/cb(k+1)
            else
              v1=10**(ca(k+1)+cb(k+1)*c(k))
              v2=10**(ca(k+1)+cb(k+1)*d(p,k))
          endif
          if (v2.gt.1.d-15) then
              grande=100*dabs((v1-v2)/v2)
            else
              grande=100*dabs(v1)
          endif
          if (errp.lt.grande) errp=grande
        enddo
      enddo
      ve2=0.d0
      do p=1,npt
        do k=1,n
          ve2=ve2+errol(p,k)**2
        enddo
      enddo
      ve2=(ve2/npt)**0.5

c***  Tomada de decisao (de acordo com a variacao do erro).

      if ((ve2.gt.tol).and.(errp.gt.tolp)) then
        if (ve2.lt.ve1) then
            mi=mi/beta
            write(*,*) ve1,'     ',errp
            goto 1
          else
            mi=mi*beta
            do i1=1,l
              do j1=1,m
                v(i1,j1)=v(i1,j1)+prod3((i1-1)*m+j1)
              enddo
            enddo
            do j1=1,m
              do k1=1,n
                w(j1,k1)=w(j1,k1)+prod3((j1-1)*n+k1+1*m)
              enddo
            enddo
            goto 2
        endif
      endif
      write(*,*) ve1,'     ',errp
      return
      end
c------------------------fim da sub-rotina trein2-----------------------


c***  Esta sub-rotina faz a predicao dos dados a partir dos pesos e
c***  topologia da rede treinada.
c***  Dados de entrada: Pesos da rede (w,v)
c***                    Topologia da rede (l,m,n)
c***                    Vetor dados de entrada (ent)
c***  Dados de saida: Vetor saida da rede: (sai)

      subroutine pred(w,v,l,m,n,ent,sai)
      implicit none
      double precision soma
      double precision w,v,ent,sai,x1,x2,x3,a,b
      integer i,j,k,l,m,n,nl,nm,nn
      parameter (nl=32,nm=22,nn=6)
      dimension v(nl,nm),w(nm,nn),ent(nl),sai(nn),x1(nl),x2(nm),x3(nn),
     &a(nl),b(nm)
      do i=1,l
```

```
        x1(i)=ent(i)
        a(i)=1.d0/(1.d0+exp(-x1(i)))
      enddo
      do j=1,m
        soma=0.d0
        do i=1,l
          soma=soma+a(i)*v(i,j)
        enddo
        x2(j)=soma+1.d0
        b(j)=1.d0/(1.d0+exp(-x2(j)))
      enddo
      do k=1,n
        soma=0.d0
        do j=1,m
          soma=soma+b(j)*w(j,k)
        enddo
        x3(k)=soma+1.d0
        sai(k)=1.d0/(1.d0+exp(-x3(k)))
      enddo
      return
      end
c-------------------------fim da sub-rotina pred-----------------------


c***  Esta sub-rotina faz o interfaciamento entre as subrotinas de
c***  treinamento e predicao e o programa principal. As variaveis
c***  inteiras sao passadas pelo vetor info, que deve conter:
c***  info(1) - opcao de treinamento, info(2) - numero de dados de dados
c***  de treinamento ou predicao, info(3) - no. de neuronios de entrada,
c***  info(4) - no. de neuronios na camada intermediaria, info(5) - no.
c***  de neuronios ultima camada, info(6) - tipo de chamada a sub-rotina,
c***  info(7) - algoritimo de treinamento a ser utilizado e info(8) -
c***  utilizacao de parametros de treinamento anteriores.
c***  As opcoes sao melhor detalhadas abaixo.
c***  Dados de ent/sai:
c***          Info(1):  Opcao de: treinamento (0); predicao (1) ou
c***                      verificacao do erro (2)
c***          Info(2):  Numero de dados a serem utilizados no
c***                      treinamento ou predicao (ndados)
c***          Info(6):  Se for primeira chamada (treinamento) a
c***                      sub-rotina, ncham.eq.0. Ncham.eq.1 indica que
c***                      os pesos e os coeficientes utilizados na adi-
c***                      mensionalizacao das variaveis nao serao modi-
c***                      ficados. Ncham.eq.-1 indica que os pesos devem
c***                      ser carregados de arquivo e nao aleatorizados.
c***                      Ncham.eq.2 indica que apenas os pesos nao serao
c***                      aleatorizados
c***          Info(7):  Opcao de treinamento: se ntrein=0, utiliza o
c***                      backpropagation (GDR), caso contrario, utiliza
c***                      o algoritmo de Marquardt-Levenberg
c***          Info(8):  Utiliza os parametros de treinamento (eta e alpha
c***                    no backpropagation e mi e beta no Marquardt)
c***                    anteriores se npar=1, senao incializa estes
c***          Tol     :  Tolerancia do erro quadratico medio
c***                      (=sqrt(erro/ndados)). Se tol=0.d0, o criterio
c***                      de convergencia fica em cima do erro maximo
c***                      percentual (tolp)
c***          Tolp    :  Tolerancia para o erro percentual maximo. Se
c***                      tolp=0.d0, o criterio de convergencia sera em
c***                      cima do erro quadratico medio. Se tanto 'tolp'
c***                      como 'tol' forem diferentes de zero, a rede
c***                      estara treinada quando for alcancado um dos
c***                      criterios.
c***          Aux1,aux2:Vetores de entrada e saida de dados. Se for
c***                      treinamento, aux1 e aux2 devem conter, respec-
c***                      tivamente, o vetor entrada e a saida desejada.
c***                      No caso de predicao, aux1 contera o vetor
c***                      entrada e aux2 a resposta da rede.
c***          Nta:      Tipo de adimensionalizacao a ser utilizado: se
c***                      nta(i)=0, adimensionalizacao linear; se
c***                      nta(i)=1, adimensionalizacao logaritmica.

      subroutine rna(info,tol,tolp,aux1,aux2,nta,v,w,ca,cb,par)
      implicit none

c***  Variaveis internas a esta sub-rotina e as subrotinas trein e pred.

      integer nopc,ncham,ntrein,l,m,n,nl,nm,nn,nmp,i,j,k,p,ndados,nta,
     &info,npar
      double precision random,inp,d,w,v,eta,alp,tol,tolp,maior,menor,ca,
     &cb,ent,sai,inf,sup,aux1,aux2,erro,errp,grande,mi,beta,par
```

```
      parameter (nl=32,nm=22,nn=6,nmp=30)
      dimension inp(nmp,nl),d(nmp,nn),nta(nl+nn),ca(nl+nn),cb(nl+nn),
     &ent(nl),sai(nn),v(nl,nm),w(nm,nn),aux1(nmp,nl),aux2(nmp,nn),
     &info(*),par(*)

c***  Este bloco common deve ser incluido no programa principal para
c***  permitir que valores nao seja perdidos ao sair da sub-rotina


c***  Atribuicao do vetor de informacoes as variaveis internas

      nopc=info(1)
      ndados=info(2)
      l=info(3)
      m=info(4)
      n=info(5)
      ncham=info(6)
      if (nopc.eq.0) then
        ntrein=info(7)
        npar=info(8)
      endif

c***  Treinamento da rede

      if (nopc.eq.0) then

c***  Salva valores de entrada

        do p=1,ndados
          do i=1,l
            inp(p,i)=aux1(p,i)
          enddo
          do k=1,n
            d(p,k)=aux2(p,k)
          enddo
        enddo

c***  Parametros de treinamento da rede para backpropagation e
c***  Marquardt-Levenberg

        alp=0.5d0
        eta=1.75d0
        beta=1.2d0
        if (npar.eq.0) then
            mi=0.01d0
          else
            mi=par(1)
        endif

c***  Se eh a primeira chamada a esta sub-rotina, faz a aleatorizacao
c***  dos pesos (valores entre -1 e 1)

        if (ncham.eq.0) then
          do i=1,l
            do j=1,m
              v(i,j)=1.+2.*(random()-1.)
            enddo
          enddo
          do j=1,m
            do k=1,n
              w(j,k)=1.+2.*(random()-1.)
            enddo
          enddo
        endif

c***  Determinacao dos coeficientes para adimensionalizacao de ent/sai.

        if ((ncham.eq.0).or.(ncham.eq.2)) then
          inf=0.2d0
          sup=0.8d0
          do i=1,l
            maior=inp(1,i)
            menor=inp(1,i)
            do p=2,ndados
              if (maior.lt.inp(p,i)) maior=inp(p,i)
              if (menor.gt.inp(p,i)) menor=inp(p,i)
            enddo
            if (nta(i).eq.0) then
                if (dabs(maior-menor).gt.1.d-30) then
                    maior=1.01*maior
                    menor=0.99*menor
```

```
                  ca(i)=(inf*maior-sup*menor)/(maior-menor)
                  cb(i)=(sup-inf)/(maior-menor)
                else
                  ca(i)=0.d0
                  if (dabs(maior).gt.1.d-10) then
                      cb(i)=1.d0/dabs(maior)
                    else
                      cb(i)=1.d0
                  endif
              endif
            else
              maior=1.01*maior
              menor=0.99*menor
              ca(i)=sup*dlog10(menor)/(sup-inf)-inf*dlog10(maior)/
     &          (sup-inf)
              cb(i)=dlog10(maior/menor)/(sup-inf)
          endif
        enddo
        do k=1,n
          maior=d(1,k)
          menor=d(1,k)
          do p=1,ndados
            if (maior.lt.d(p,k)) maior=d(p,k)
            if (menor.gt.d(p,k)) menor=d(p,k)
          enddo
          if (nta(k+1).eq.0) then
              ca(k+1)=(inf*maior-sup*menor)/(maior-menor)
              cb(k+1)=(sup-inf)/(maior-menor)
            else
              ca(k+1)=sup*dlog10(menor)/(sup-inf)-inf*dlog10(maior)/
     &          (sup-inf)
              cb(k+1)=dlog10(maior/menor)/(sup-inf)
          endif
        enddo
      endif

c***  Adimensionalizacao de ent/sai para mante-las no range [inf;sup]

      do p=1,ndados
        do i=1,l
          if (nta(i).eq.0) then
              inp(p,i)=ca(i)+cb(i)*inp(p,i)
            else
              inp(p,i)=(dlog10(inp(p,i))-ca(i))/cb(i)
          endif
        enddo
        do k=1,n
          if (nta(k+1).eq.0) then
              d(p,k)=ca(k+1)+cb(k+1)*d(p,k)
            else
              d(p,k)=(dlog10(d(p,k))-ca(k+1))/cb(k+1)
          endif
        enddo
      enddo

c***  Chamada a sub-rotina 'treinl' ou 'trein2' que fazem o treinamento
c***  da rede.
      if (ntrein.eq.0) then
          call treinl(l,m,n,ndados,inp,d,eta,alp,tol,tolp,nta,v,w,
     &                ca,cb)
        else
          call trein2(l,m,n,ndados,inp,d,mi,beta,tol,tolp,nta,v,w,
     &                ca,cb)
      endif
    endif

c***  Predicao de valores a partir da rede

    if (nopc.eq.1) then

c***  Salva valores de entrada

      do p=1,ndados
        do i=1,l
          inp(p,i)=auxl(p,i)
        enddo
      enddo

c***  Adimensionalizacao das entradas para mante-las no range [inf;sup]

      do p=1,ndados
```

```
          do i=1,l
            if (nta(i).eq.0) then
                inp(p,i)=ca(i)+cb(i)*inp(p,i)
              else
                inp(p,i)=(dlog10(inp(p,i))-ca(i))/cb(i)
            endif
            ent(i)=inp(p,i)
          enddo

c***  Chamada da sub-rotina que faz a predicao

          call pred(w,v,l,m,n,ent,sai)

c***  Passagem da saida para a forma nao adimensional

          do k=1,n
            if (nta(k+l).eq.0) then
                d(p,k)=(sai(k)-ca(k+l))/cb(k+l)
                aux2(p,k)=d(p,k)
              else
                d(p,k)=10.d0**(ca(k+l)+cb(k+l)*sai(k))
                aux2(p,k)=d(p,k)
            endif
          enddo
        enddo
      endif

c***  Calculo do erro (sem treinamento)

      if (nopc.eq.2) then

      erro=0.d0
      errp=0.d0

c***  Salva valores de entrada/saida

        do p=1,ndados
          do i=1,l
            inp(p,i)=aux1(p,i)
          enddo
          do k=1,n
            d(p,k)=aux2(p,k)
          enddo
        enddo

c***  Adimensionalizacao das ent/sai para mante-las no range (inf;sup)

        do p=1,ndados
          do i=1,l
            if (nta(i).eq.0) then
                inp(p,i)=ca(i)+cb(i)*inp(p,i)
              else
                inp(p,i)=(dlog10(inp(p,i))-ca(i))/cb(i)
            endif
            ent(i)=inp(p,i)
          enddo
          do k=1,n
            if (nta(k+l).eq.0) then
                d(p,k)=ca(k+l)+cb(k+l)*d(p,k)
              else
                d(p,k)=(dlog10(d(p,k))-ca(k+l))/cb(k+l)
            endif
          enddo

c***  Chamada da sub-rotina que faz a predicao

          call pred(w,v,l,m,n,ent,sai)

          do k=1,n

c***  Calculo do erro quadratico medio

            erro=erro+(sai(k)-d(p,k))**2

c***  Passagem da saida para a forma nao adimensional

            if (nta(k+l).eq.0) then
                sai(k)=(sai(k)-ca(k+l))/cb(k+l)
              else
                sai(k)=10.d0**(ca(k+l)+cb(k+l)*sai(k))
            endif
```

```
c***  Calculo do erro percentual

            if (dabs(aux2(p,k)).gt.1.d-10) then
                grande=dabs((aux2(p,k)-sai(k))/aux2(p,k))
              else
                grande=dabs(aux2(p,k)-sai(k))
            endif
            if (grande.gt.errp) errp=grande
          enddo
        enddo
        erro=(erro/ndados)**0.5
        errp=100*errp
c***  Escreve erro quadratico medio e erro percentual na tela

        write(*,*) erro,'     ',errp

        tolp=errp

      endif
c***  Devolve os parametros de treinamento finais.
      if (nopc.eq.0) then
        par(1)=mi
      endif
      return
      end
c-----------------------fim da sub-rotina rna-------------------------

c---------------------------------------------------------------------
c     Sub-rotinas extraidas de Villadsen e Michelsen (1978), utilizadas
c     nos calculos relativos a colocacao ortogonal e mais sub-rotina de
c     resolucao de sistemas lineares (GAUSS, JCOBI, DFOPR e INTRP)
c---------------------------------------------------------------------

      subroutine gauss(n,ns,ncol,a)
      double precision a(ncol,2*ncol),x
      n1=n+1
      nt=n+ns
      if(n.eq.1) go to 50
      do 40 i=2,n
      ip=i-1
      i1=ip
      x=dabs(a(i1,i1))
      do 10 j=i,n
      if(dabs(a(j,i1)).lt.x) go to 10
      x = dabs(a(j,i1))
      ip=j
10    continue
      if(ip.eq.i1) go to 30
      do 20 j=i1,nt
      x = a(i1,j)
      a(i1,j) = a(ip,j)
20    a(ip,j) = x
30    do 40 j=i,n
      x = a(j,i1)/a(i1,i1)
      do 40 k=i,nt
40    a(j,k) = a(j,k)-x*a(i1,k)
50    do 70 ip=1,n
      i = n1-ip
      do 70 k=n1,nt
      a(i,k) = a(i,k)/a(i,i)
      if (i.eq.1) go to 70
      i1 = i-1
      do 60 j=1,i1
60    a(j,k) = a(j,k)-a(i,k)*a(j,i)
70    continue
      return
      end
c-----------------------fim da sub-rotina gauss-----------------------

      subroutine jcobi (nd,n,no,n1,al,be,difl,dif2,dif3,root)
      implicit double precision (a-h,o-z)
      dimension difl(nd),dif2(nd),dif3(nd),root(nd)
c
c     calculo das raizes e derivadas do polinomio de jacobi
c
      ab=al+be
      ad=be-al
      ap=be*al
      difl(1)=(ad/(ab+2)+1)/2
      dif2(1)=0.
```

```
      if (n.lt.2) goto 15
      do 10 i=2,n
      z1=i-1
      z=ab+2*z1
      dif1(i)=(ab*ad/z/(z+2)+1)/2
      if (i.ne.2) goto 11
      dif2(i)=(ab+ap+z1)/z/z/(z+1)
      goto 10
   11 z=z*z
      y=z1*(ab+z1)
      y=y*(ap+y)
      dif2(i)=y/z/(z-1)
   10 continue
c
c     determinacao das raizes pelo metodo de newton com supressao das
c     raizes determinadas anteriormente
c
   15 x=0.
      do 20 i=1,n
   25 xd=0.
      xn=1.
      xd1=0.
      xn1=0.
      do 30 j=1,n
      xp=(dif1(j)-x)*xn-dif2(j)*xd
      xp1=(dif1(j)-x)*xn1-dif2(j)*xd1-xn
      xd=xn
      xd1=xn1
      xn=xp
   30 xn1=xp1
      zc=1.
      z=xn/xn1
      if (i.eq.1) goto 21
      do 22 j=2,i
   22 zc=zc-z/(x-root(j-1))
   21 z=z/zc
      x=x-z
      if (dabs(z).gt.1.d-09) goto 25
      root(i)=x
      x=x+0.0001
   20 continue
c
c     acrescenta eventuais pontos de colocacao em x=0 e x=1
c
      nt=n+no+n1
      if (no.eq.0) goto 35
      do 31 i=1,n
      j=n+1-i
   31 root(j+1)=root(j)
      root(1)=0
   35 if (n1.eq.1) root(nt)=1.
c
c     calculo das derivadas do polinomio
c
      do 40 i=1,nt
      x=root(i)
      dif1(i)=1.
      dif2(i)=0.
      dif3(i)=0.
      do 40 j=1,nt
      if (j.eq.i) goto 40
      y=x-root(j)
      dif3(i)=y*dif3(i)+3*dif2(i)
      dif2(i)=y*dif2(i)+2*dif1(i)
      dif1(i)=y*dif1(i)
   40 continue
      return
      end
c-------------------------fim da sub-rotina jcobi-------------------------

      subroutine dfopr(nd,n,no,n1,i,id,dif1,dif2,dif3,root,vect)
      implicit double precision(a-h,o-z)
      dimension dif1(nd),dif2(nd),dif3(nd),root(nd),vect(nd)
c
c     dfopr calcula as matrizes de discretizacao e os pesos da
c     quadratura gaussiana, normalizados para soma 1
c     id = 1;  matriz de dicretizacao para y(1) (x)
c     id = 2;  matriz de dicretizacao para y(2) (x)
c     id = 3;  pesos da quadratura gaussiana
c
      nt=n+no+n1
```

```
      if(id.eq.3) goto 10
      do 20 j=1,nt
      if (j.ne.i) goto 21
      if (id.ne.1) goto 5
      vect(i)=dif2(i)/dif1(i)/2
      goto 20
    5 vect(i)=dif3(i)/dif1(i)/3
      goto 20
   21 y=root(i)-root(j)
      vect(j)=dif1(i)/dif1(j)/y
      if (id.eq.2) vect(j)=vect(j)*(dif2(i)/dif1(i)-2/y)
   20 continue
      goto 50
   10 y=0.
      do 25 j=1,nt
      x=root(j)
      ax=x*(1-x)
      if (no.eq.0) ax=ax/x/x
      if (n1.eq.0) ax=ax/(1-x)/(1-x)
      vect(j)=ax/dif1(j)**2
   25 y=y+vect(j)
      do 60 j=1,nt
   60 vect(j)=vect(j)/y
   50 return
      end
c-------------------------fim da sub-rotina dfopr--------------------------

      subroutine intrp(nd,nt,x,root,difl,xintp)
      implicit double precision (a-h,o-z)
      dimension root(nd),difl(nd),xintp(nd)
c
c     avaliacao dos coeficientes de interpolacao lagrangiana
c
      pol=1.
      do 5 i=1,nt
      y=x-root(i)
      xintp(i)=0.d0
      if (dabs(y).lt.1.d-10) xintp(i)=1.
    5 pol=pol*y
      if (dabs(pol).lt.1.d-10) goto 10
      do 6 i=1,nt
    6 xintp(i)=pol/difl(i)/(x-root(i))
   10 return
      end
c-------------------------fim da sub-rotina intrp--------------------------
```