PAULO JOSÉ SAIZ JABARDO

DESENVOLVIMENTO DE UM SIMULADOR DE SISTEMAS TÉRMICOS

Trabalho de formatura apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do título de Graduação em Engenharia

Orientador:
Euryale de Jesus Zerbini

Área de Concentração:
Engenharia Mecânica
Energia e Fluidos

São Paulo
1999

...Uma possível explicação para o fato de saírem sons de uma vitrola é que esta possui uma alma. Esta explicação no entanto não me serve para nada pois não me permite aprender ou explicar nada além do fato de saírem palavras da vitrola...

Lev Landau

Agradecimentos

Eu certamente não poderia ter feito este trabalho sozinho. Inicialmente eu gostaria de agradecer meu orientador, o professor Euryale de Jesus Zerbini. As nossas conversas semanais realmente ajudaram muito no decorrer do trabalho. Suas sugestões e críticas ao longo do ano e principalmente no final foram muito importantes na elaboração deste trabalho.

Também gostaria de agradecer a meu Pai, professor José Maria Saiz Jabardo, que inspirou o trabalho, fez diversas sugestões, forneceu boa parte da literatura usada neste trabalho e aguentou numerosas discussões pacientemente.

Gostaria de agradecer a minha Mãe, Temis Rute Saiz Jabardo que me recebia muito bem em casa nos fins de semana (sempre com minha comida favorita, apesar do sorvete de maracujá).

Agradeço também as muitas pessoas que fizeram pequenas sugestões ao longo do trabalho mas, sem querer ofender ninguém, vou ficar por aqui.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O título deste trabalho é *Desenvolvimento de um Simulador de Sistemas Térmicos.* Neste título aparecem alguns conceitos básicos ao trabalho:

- Área Térmica
- Sistema
- Simulação

Este trabalho trata da simulação de sistemas na área térmica. Praticamente todos os conceitos a serem desenvolvidos neste trabalho são aplicáveis a qualquer área, como se pode observar na bibliografia, onde muitas referências são livros de cálculo numérico. A ênfase deste trabalho é a análise de processos termofísicos, ou seja, processos onde são utilizados os conceitos provenientes da termodinâmica, da mecânica dos fluidos e da transferência de calor. A área térmica envolve uma infinidade de processos e equipamentos, por exemplo: refrigeração e ar-condicionado, a maioria dos processos químicos e quase todos os ciclos de potência conhecidos (geração de energia e trabalho).

1.1 Sistemas

O segundo conceito listado é *Sistema*. O que é um sistema? Boa parte do curso de engenharia foi baseado em processos, fenômenos simples. Na maioria das vezes, o engenheiro trabalha com sistemas.

Os diversos equipamentos e ciclos usados na área térmica envolvem processos básicos. Estes processos, entretanto, não ocorrem isoladamente, e sim em conjunto, com cada processo influenciando o outro. Um sistema é basicamente isto, um conjunto de processos ocorrendo e interagindo simultaneamente. Isto cria sérias dificuldades pois os diferentes processos são por sí só extremamente complexos na sua maioria. Imagine agora juntar diferentes processos na mesma análise.

A complexidade da análise de sistemas tem como conseqüência a necessidade de se trabalhar com modelos simplificados da realidade. Isto introduz um novo conceito muito importante neste trabalho: *Modelagem*.

1.2 Simulação

Simulação nada mais é do que estudar o comportamento de sistemas através de um modelo adequado. A simulação é distinta do projeto, apesar de ser uma técnica muito importante no projeto de sistemas.

O objetivo do projeto é dimensionar um sistema que atenda a um conjunto de especificações. A simulação, no entanto, parte do pressuposto que o sistema já existe, ou que pelo menos, exista um modelo que represente a realidade.

Existem vários objetivos na simulação. Um primeiro objetivo é, dado um sistema, qual o seu comportamento variando alguma condição (ou condições) externa. Esta simulação permite um melhor uso dos recursos disponíveis.

Permite também conhecer em que outras condições o sistema pode ser usado ou modificado.

Um outro objetivo importante é no auxílio ao projeto de um sistema. Poder conhecer, mesmo de maneira aproximada, o comportamento de um sistema *a priori* traz diversas vantagens. Permite estudar, a baixo custo, diversas possibilidades para um mesmo projeto. Permite determinar condições de funcionamento do sistema, facilitando, assim a seleção e dimensionamento dos diferentes componentes.

A simulação também permite determinar possíveis problemas no sistema. Isto torna o sistema mais seguro e caso ocorra algum problema, a simulação pode permitir que o problema seja encontrado o mais rapidamente possível.

Um último aspecto da simulação, mas também muito importante é a modelagem: Os resultados da simulação só podem ser tão bons quanto o modelo usado nos diferentes componentes do sistema.

1.3 Modelagem de Sistemas Térmicos

Como foi dito acima, a modelagem é um aspecto muito importante na simulação, e na opinião deste autor, a mais importante. Sem um modelo adequado, a simulação tem pouco valor. Com um modelo muito refinado é possível que nenhum resultado seja alcançado, ou pelo menos, o esforço (custo e tempo) não valha a pena.

Um modelo adequado é fundamental. Como modelar o sistema? Esta é uma pergunta que apenas a experiência pode responder. A primeira tarefa do enegenheiro é encontrar um modelo adequado do sistema. Existem diversas possibilidades na modelagem. O objetivo deste trabalho não é modelar o sistema, e sim, simular o sistema para uma classe específica de modelos a ser

descrita a seguir.

1.3.1 Diferentes tipos de modelos

Um modelo nada mais é do que uma representação simplificada da realidade. Um diagrama esquemático é um modelo, por exemplo. Muitas vezes, quando o sistema é complexo e envolve altos custos, um modelo em escala reduzida é construído. Este modelo é, então, usado na simulação do sistema real.

Uma outra classe de modelos, o foco deste trabalho, são os modelos matemáticos. Um modelo matemático é obtido através de algum princípio físico aplicado a componentes do sistema. Pode-se fazer, em alguma parte do processo, um balanço de energia, um balanço de massa ou um balanço de quantidade de movimento, por exemplo. O modelo pode ser empírico, ou seja, o comportamento do componente em estudo pode ser modelado através de um ajuste de dados experimentais. Uma técnica para se fazer este ajuste será analisada no capítulo 5.

Aplicando as equações fundamentais a volumes de controle infinitesimais, o resultado, no caso geral, é uma equação diferencial parcial. Este tipo de equação é extremamente difícil de se resolver. Uma possibilidade é dividir o sistema em volumes de controle finitos, resultando em um sistema de equações diferenciais ordinárias no domínio do tempo. A solução deste tipo de sistema é muito mais simples que a solução de sistemas de equações diferenciais parciais. O capítulo 4 apresenta algumas técnicas na solução deste tipo de problema.

Agora, se o sistema operar em *regime permanente*, as simplificações acima resultarão em um sistema de equações algébricas. Estas equações, no caso geral, são não lineares. O objetivo fundamental deste trabalho é a simulação de sistemas modelados desta maneira.

O uso de parâmetros concentrados (volumes de controle finitos com propriedades uniformes) parece muito simplificado mas em diversas situações é a única solução possível. Em sistemas com muitos componentes, como a câmara frigrífica do capítulo 7, o uso parâmetros concentrados é essencial. Alguns componentes são modelados a partir de dados experimentais fornecidos por fabricantes. Em outros componentes, um modelo simpificado, que usa parâmetros concentrados é usado. Os resultados obtidos servem apenas como uma estimativa dos parâmetros principais do sistema. Estes resultados podem ser muito úteis no estudo do comportamento do sistema.

Um último aspecto fundamental à área térmica é a modelagem de propriedades termodinâmicas de fluidos. A maioria dos sistemas térmicos envolve algum fluido. As propriedades termodinâmicas (e de transporte) estão quase sempre presentes na modelagem do sistema. Existem diversas técnicas para a modelagem destas propriedades, algumas delas estão apresentadas no capítulo 6.

1.4 Motivações do Trabalho

Este trabalho surgiu de uma proposta do professor Jabardo. O autor e o professor Jabardo tinham desenvolvido um programa (o autor fez apenas a parte computacional) de seleção de equipamentos para uma câmara frigorífica. O resultado foi um programa muito simples. A proposta do professor Jabardo era de melhorar o modelo da câmara (e dos diversos componentes) e fazer algumas simulações. Com isso surgiu a idéia de se desenvolver um simulador um pouco mais geral. O autor conversou com o professor Zerbini que gostou da idéia e assim surgiu o trabalho.

O trabalho consiste em diferentes programas de computador. A interface

com o usuário foi inspirada por diferentes softwares existentes no mercado. Procurou-se implementar os diferentes programas de modo a facilitar o seu uso e permitir a integração com outros softwares de modo a tornar os programas de computador desenvolvidos úteis para diversos fins.

Capítulo 2

Interpretador de Expressões

Este capítulo descreve o interpretador de expressões desenvolvido neste trabalho. A interface com o usuário pretende ser bastante simples, de modo a permitir que o usuário use o *solver* com um programa compilado em qualquer linguagem.

A estratégia adotada foi criar um solver que lê arquivos texto (em formato ASCII), interpreta as equações lidas e gera um código binário eficiente.

2.1 Introdução

A leitura e interpretação de expressões matemáticas não é uma tarefa simples e diversas soluções são conhecidas. Uma possibilidade é ler as expressões e interpretá-las imediatamente. Esta solução talvez seja a mais simples a nível de programação mas certamente não é satisfatória no caso presente devido à imensa quantidade de cálculos a serem realizados. Como a mesma expressão é calculada diversas vezes, esta estratégia resultaria em um solver lento (cada vez que uma expressão é executada há a necessidade de interpretá-la).

A solução adotada foi criar um interpretador de expressões que gera um

código binário que pode ser executado diversas vezes de maneira eficiente.

A primeira etapa da interpretação da expressão será denominada, a partir de agora, *compilação da expressão*. Este processo é semelhante a compilação de um programa por um compilador (*FORTRAN*, *BASIC* ou *C* por exemplo). A execução da expressão, ou seja, a determinação do valor numérico da expressão, será denominada, de agora em diante, de interpretação da expressão. O resultado da compilação da expressão é um código binário cujo formato será descrito adiante (seção 2.5).

Geralmente, as pessoas escrevem expressões matemáticas em notação infixa. Nesta notação, os operadores situam-se entre os operandos. Esta notação apesar de fácil para as pessoas lerem e manipularem é extremamente complicada para análise no computador como resultado das inúmeras regras existentes, como por exemplo, a precedência de operadores. A solução adotada neste trabalho foi desenvolver um algorítimo que transforma uma expressão escrita em notação infixa em uma expressão em notação pós-fixa, também denominada *RPN* (*Reverse Polish Notation*). Na notação *RPN*, os operadores situam-se após os operandos e são executados na ordem que aparecem. Esta notação é muito usada: qualquer um que tenha trabalhado com uma calculadora *HP* a conhece, e certamente aprecia algumas de suas vantagens, apesar de ser um pouco desagradável aprendê-la no início (basta ouvir todos os *bips* em uma prova de engenharia de alunos de primeiro ano). Vale lembrar, no entanto, que o *usuário não trabalha com notação RPN* apenas na notação infixa, a que todos estão acostumados usar.

2.2 O Compilador

Como descrito no item anterior, o compilador gera um código binário da expressão matemática (em formato texto) para posterior execução. O compilador basicamente reescreve a expressão em notação *RPN*.

O compilador é formado por duas partes básicas: o analisador léxico(seção 2.3) e o compilador propriamente dito, também denominado analisador sintático (seção 2.4).

2.3 O Analisador Léxico

O analisador léxico recebe uma expressão em formato texto (seqüência de caracteres `ASCII`) e retorna um conjunto de *tokens*. Um *token* é a unidade léxica mínima de uma expressão matemática. Como exemplo, a expressão $\sin(2 * \exp(x))$ pode ser dividida nos seguintes *tokens*: sin, (, 2, $*$, exp,(, x e). Cada *token* tem um significado que considerado isoladamente pode não ter sentido mas quando considerado como parte de um todo e usado corretamente (o uso correto é dado pela gramática da linguagem) permite a interpretação de complexas expressões matemáticas, ou seja, o *token* representa uma informação que por sí não quer dizer nada mas é essencial à expressão como um todo. O significado pode ser, por exemplo, uma variável, uma constante numérica, uma constante simbólica, um operador ou uma função, entre outros. É tarefa do analisador léxico determinar o significado de cada *token*.

No programa de computador criado, o analisador léxico é implementado na função `Tokenize`. Esta função tem como entrada uma cadeia de caracteres (*string*) e apresenta como saída uma seqüência de tokens, armazenados num vetor de `TToken` (a definição de `TToken` pode ser vista na figura 2.2, página

23). Esta estrutura de dados (`TToken`) é composta por dois campos:

- *KIND*, que armazena o tipo de token (seu significado)

- *INDEX*, que armazena o *token* propriamente dito

A análise léxica da expressão é feita dividindo a expressão nos delimitadores. Os delimitadores são caracteres (ou conjunto de caracteres) que separam diversas partes de uma expressão, como por exemplo parênteses ou operadores.

A segunda etapa consiste em determinar se um separador é realmente um separador, mais especificamente, verificar se um operador + ou − é realmente um operador ou apenas parte de uma constante numérica escrita em notação científica (ex. $3,51E+01$). Esta segunda parte é a mais complicada e mais comprida da subrotina. Por fim, com os *tokens* separados, deve-se determinar seu significado. Isto é fácil para operadores e parênteses mas é um pouco mais complicado no caso de funções, variáveis ou constantes.

No início do programa, a subrotina `InitOperators` é chamada, de modo a inicializar a lista de funções declaradas no programa e constantes globais (constantes visíveis a qualquer sistema que porventura exista no programa). A leitura do sistema também cria uma lista de constantes locais. A última etapa consiste em verificar as listas de funções, constantes globais e constantes locais, nesta ordem, até encontrar o *token* em questão. Caso este *token* não seja encontrado, deve ser uma variável.

Desta forma, a partir de uma cadeia de caracteres contendo a definição da expressão matemática, constrói-se uma lista de unidades léxicas básicas da expressão (*tokens*) para posterior manipulação.

2.4 Implementação do Compilador

Nesta seção, é descrita a parte mais importante do compilador. Esta parte recebe uma seqüência de *tokens* em notação infixa e retorna um conjunto de *tokens* em notação *RPN*.

Esta parte do compilador é implementada na função `infix2postfix` do programa criado.

O algorítimo é muito simples. A idéia é precorrer a lista de *tokens*, armazenando os operadores até que suas posições na lista de *tokens* de saída (em notação *RPN*) sejam encontradas. O algorítimo básico é descrito por Kruse[7] de maneira simples. O algorítimo lê os *tokens* seqüencialmente. Caso o token seja um número (argumento, que pode ser uma constante ou variável), deve ser colocado diretamente na estrutura de dados. Caso o token seja um operador (binário ou unário), as coisas complicam-se um pouco: o problema é que existe a precedência de operadores e isto deve ser levado em consideração.

Caso todos os operadores tivessem a mesma precedência, não haveria nenhum problema: bastaria posicionar os operadores na estrutura de dados de saída, de maneira análoga aos operandos.

2.4.1 Precedência de operadores

Neste programa definiu-se a precedência de operadores, da maior para a menor, da maneira a seguir:

1. Exponenciação ($\hat{}$)
2. Operadores unários (+, -)
3. Os seguintes operadores binários: (*, /)

4. Os seguintes operadores binários: (+, -)

5. O operador comparação, (=)

2.4.2 Algorítimo básico

```
Expr - Lista de tokens com a expressão
Criar a pilha
DO
    PegarToken Token, Expr
    SELECIONAR CASO (tipo = TIPO(Token))
    CASO OPERANDO
        PorToken Token, Postfix
    CASO PARÊNTESE ESQUERDO
        PUSH Token, Pilha
    CASO PARÊNTESE DIREITO
        DO ENQUANTO TOPO DA PILHA <> PARÊNTESE ESQUERDO
            POP Token, Pilha
            PorToken Token, Postfix
        LOOP
    CASO OPERADOR
    DO
        SE A PILHA ESTIVER VAZIA, SAIR DO LOOP
        SENÃO
            StackTop TopEntry
            SE TIPO(TopEntry) = PARÊNTESE ESQUERDO
                SAIR DO LOOP
            SE Prioridade(TopEntry)<Prioridade(token)
                NÃO FAZER NADA
            SENÃO
                Pop Tmp, Pilha
                PorToken Tmp, postfix
            FIM SE
        FIM SE
    LOOP ENQUANTO VERDADEIRO
    Push Token, Pilha
    CASO FIM DA EXPRESSÃO
        LIMPAR A PILHA E COLOCAR TUDO EM Postfix
LOOP ENQUANTO HOUVEREM Tokens
```

Figura 2.1: Algorítimo para conversão de notação infixa para *RPN*

Como mencionado anteriormente, a precedência de operadores introduz

uma complicação ao problema. Para resolver este problema, construiu-se uma pilha[1]. Ao encontrar um operador ou parênteses na lista de tokens, o programa armazena este *token* na pilha (se esta estiver vazia).

Caso a pilha não esteja vazia, o programa verifica qual o *token* no topo da pilha. Se o *token* do topo da pilha for um parênteses esquerdo, o novo *token* será colocado no topo da pilha. Caso o novo *token* seja um parêntese direito, todos os operadores da pilha serão removidos e colocados na estrutura de dados de saída até encontrar o parêntese esquerdo correspondente.

Caso o *token* no topo da pilha seja um operador com precedência maior que o novo operador, o operador no topo da pilha será retirado e colocado na estrutura de dados de saída enquanto que o operador novo será empurrado na pilha.

Caso o final da expressão seja atingido, todos os operadores devem ser retirados da pilha e colocados sequencialmente na estrutura de dados de saída. O algorítimo básico pode ser visto na figura 2.1.

Até este momento, chamadas a funções não foram consideradas, apenas operadores binários e unários. A função na realidade representa um número e portanto deve ser posicionado na estrutura de dados de saída imediatamente. Para tanto, cada argumento da função é processado recursivamente pela função `infix2postfix` e finalmente, a função é colocada na estrutura de dados de saída.

[1]Pilha é uma estrutura de dados na qual o primeiro dado a entrar é o último a sair (*LIFO - Last In First Out*)

2.5 Estrutura de Dados

A estrutura de dados empregada no programa será brevemente descrita nesta seção. Na figura 2.2 pode-se observar os tipos de dados pré-definidos empregados para armazenar as equações.

```
TYPE TToken
     KIND AS LONG
     INDEX AS LONG
END TYPE

TYPE TFUNCTIONLIST
     ADD    AS LONG 'Endereço da função
     NARGS  AS LONG 'Número de argumentos
     PRIORITY AS LONG 'Prioridade da função
END TYPE

TYPE TEquation
     NTOKENS AS LONG         'Número de tokens na equação
     TOKENS AS TToken PTR    'Ponteiro para a estrutura binária
     NCONST AS LONG          'Número de constantes numéricas na eq.
     CONST(1 TO %MAXNUMCONST) AS DOUBLE 'Constantes numéricas
END TYPE

TYPE TSystem
     NVAR AS LONG   'Número de variáveis
     NCONST AS LONG 'Número de constantes simbólicas
     NEQUATIONS AS LONG  'Número de equações
     VAR(1 TO %MAXVAR) AS DOUBLE  'Armazenar variáveis
     VAR_MAX(1 TO %MAXVAR) AS DOUBLE 'Limite superior das var.
     VAR_MIN(1 TO %MAXVAR) AS DOUBLE 'Limite inferior das var.
     VAR_STR(1 TO %MAXVAR) AS ASCIIZ*%MAXCHARNUM
     CONST(1 TO %MAXCONST) AS DOUBLE  'Constantes simbólicas
     CONST_STR(1 TO %MAXCONST) AS ASCIIZ*%MAXCHARNUM
     EQUATION(1 TO %MAXEQUATIONS) AS TEquation PTR 'Ponteiro p/ as eqs.
END TYPE
```

Figura 2.2: Estrutura de dados usada para armazenar as equações

A estrutura `TToken` é usada para armazenar um *token* como já foi comentado. A estrutura `TFUNCIONLIST` armazena ponteiros para as funções e operadores pré-definidos, assim como suas prioridades.

Uma equação única é armazenada em uma variável tipo `TEquation`. Ob-

serve que esta estrutura tem o espaço para as constantes numéricas já alocado. O campo `TOKENS` é um ponteiro para um bloco de memória contendo a seqüência de *tokens* (`TToken`) no formato binário. A função `Evaluate` (seção 2.6) recebe este ponteiro e retorna o valor da expressão.

Um sistema de equações é armazenado na estrutura de dados `TSystem`. Esta estrutura tem espaço para as constantes numéricas locais ao sistema, variáveis e ponteiros para as equações (`TEquation`).

As estruturas de dados acima são empregadas na solução de sistemas de equações não lineares, assim como na integração numérica de sistemas de equações diferenciais não lineares.

2.6 Interpretação de Expressões

A interpretação das expressões matemáticas é feita pela função `Evaluate`. Esta função recebe como parâmetro um ponteiro (endereço de memória) de uma expressão matemática e retorna o seu valor numérico.

A interpretação de uma expressão numérica em formato *RPN* é muito simples. A subrotina percorre os *tokens* seqüencialmente. Todos os argumentos são empurrados numa pilha. Ao encontrar um operador, um número de argumentos correspondente ao número de operandos do operador são retirados da pilha (*POP*) e usados como argumentos do operador. O valor retornado pelo operador é, então, empurrado na pilha. Desta maneira o processamento prossegue até o fim da expressão.

Observe que o método para interpretação de expressões é exatamente o mesmo que o método empregado pelas calculadores *HP*. Um argumento é inserido digitando o número e pressionando `ENTER`. Isto equivale à operação `Push`. Quando uma função é executada (pressionando alguma tecla, por

exemplo sin, cos,log ou outros) os argumentos na memória da calculadora (no programa corresponde à pilha) são usados como argumentos da função. O valor retornado pela função, por sua vez é empurrado na pilha (na *HP* isto equivale a permanecer na tela).

2.7 Resultados

Esta seção analisa o desempenho do compilador, ou melhor, do código compilado. O objetivo deste teste é determinar se a idéia de empregar o compilador desenvolvido no *solver* é viável. Caso o código compilado não fosse eficiente, a idéia de usar arquivos texto para se fazer o programa seria praticamente impossível devido ao grande número de cálculos a serem feitos.

Neste teste, não será verificada a velocidade de compilação já que esta parte do programa é executada apenas uma vez e a velocidade de compilação é da ordem de alguns milésimos de segundo (20 milésimos de segundo para uma equação pequena em um *PENTIUM 100.*

O teste consiste em calcular o valor de uma expressão um certo número de vezes (10,000 ou 100,000) e comparar o tempo de execução em relação à mesma expressão compilada pelo *Power Basic.* Os resultados foram excelentes. O código criado é apenas 3,5 a 4,5 vezes mais lento que o código equivalente em *Power Basic.*

As seguintes expressões foram analisadas:

1. `3 ^ 2`

2. `3*COS(3))`

3. `(COS(SIN(3*(2+1))+1)+TAN(5+2)*COS(SIN(TAN(5 ^ 2)))`

Em todos os casos acima, a velocidade do código interpretado foi aproximadamente 4 vezes mais lento que o código executável. Este número no entanto variou bastante: em alguns casos apenas $1,5$ vezes mais lento e em outros chegou a ser 13 vezes mais lento mas em geral ficando em torno de 4.

Os testes foram realizados em duas máquinas diferentes (um *PENTIUM 100* e um *PENTIUM II 333*). Os resultados são praticamente idênticos nas duas máquinas. Observe que quando as máquinas estavam operando em *multi-task* (vários programas sendo executados simultaneamente) os resultados tinham uma dispersão considerável para cada teste.

O resultado pode ser considerado excelente. O compilador *Power Basic* é o compilador Win32 mais rápido que o autor conhece e a velocidade do código interpretado é da mesma ordem de grandez do código equivalente executável. Além disto, o código interpretado realiza uma verificação de erros o que não ocorre com o código executável.

Capítulo 3

Resolução de Sistemas de Equações não-Lineares

Sistemas modelados por parâmetros concentrados, em regime permanente podem ser representados por um sistema de equações não lineares. A solução deste tipo de sistema, no caso geral, não tem solução analítica e conseqüentemente uma solução numérica é necessária.

Existem vários métodos para a solução de sistemas de equações não-lineares. O mais conhecido, e provavelmente o mais usado, é o método de *Newton-Raphson*. Este método também serve de base para muitos outros métodos. Existem ainda outros tipos de métodos que procuram solucionar os problemas inerentes ao método de *Newton-Raphson*.

No curso deste trabalho, diversos métodos foram implementados e estudados. Neste capítulo serão descritos e analisados vários métodos utilizados. É importante ressaltar que é necessário, em alguns métodos, inverter matrizes, enquanto que em outros é necessário calcular derivadas numéricas (ou ambos).

3.1 Solução de Sistemas de Equações Lineares

Esta seção trata da solução de sistemas de equações lineares. Porque esta seção está localizada no início do capítulo referente a solução de sistema de equações não lineares? A resposta é simples: diversos métodos de solução de sistemas de equações não lineares requerem a solução de sistemas de equações lineares. O exemplo mais notável é o método de *Newton-Raphson* que será descrito adiante (seção 3.3). Outros métodos também fazem uso da solução deste tipo de sistema.

Diferentes métodos para a solução de sistemas de equações lineares são conhecidos. Estes métodos podem ser divididos em duas grandes famílias:

- Métodos Diretos
- Métodos Iterativos

Os métodos diretos procuram achar uma solução do sistema como um todo. Os métodos iterativos, por sua vez, começam com uma estimativa inicial da solução e procuram, de alguma maneira, achar uma estimativa melhor da solução do sistema. Quando a diferença entre as estimativas de duas iterações for inferior a um resíduo admissível, ϵ, a solução do sistema (ou melhor, uma boa estimativa) foi encontrada.

Para sistemas de pequena ordem (poucas equações) os métodos diretos são mais utilizados. Os métodos diretos, em geral, não têm problemas de convergência ou restrições nos coeficientes mas o número de cálculos realizados cresce rapidamente com o número de equações. Em geral os métodos diretos são recomendados para sistemas com um número de equações inferior a 50 ou 100. Vale lebrar que estas restrições não são válidas para sistemas de equações com certas características, como por exemplo os sistemas tridiagonais.

Os sistemas de grande porte normalmente são resolvidos com métodos iterativos. É importante ressaltar que, em certas condições, estes métodos podem não convergir.

3.1.1 Representação de sistemas lineares

Os sistemas lineares podem ser escritos como indicado na equação a seguir:

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \dots & & \\ a_{m1} & a_{m2} & \dots & a_{mm} \end{pmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{Bmatrix} = \begin{Bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{Bmatrix} \tag{3.1}$$

Esta equação pode ser escrita em formato matricial:

$$[A]\{u\} = \{r\} \tag{3.2}$$

A seguir, diferentes métodos de solução de sistemas de equações serão brevemente descritos.

3.1.2 Método de eliminação de Gauss

Este é um dos métodos mais conhecidos e utilizados. É recomendado para sistemas com até 50 equações. No caso do programa desenvolvido, os sistemas a serem resolvidos são pequenos na sua maioria e portanto este é o método *default* para a solução de sistemas de equações lineares.

Considerando o sistema 3.1, o método procura, através de combinações

lineares das equações reduzir o sistema ao seguinte formato:

$$\begin{aligned} u_1 + a'_{12}u_2 + a'_{13}u_3 + \ldots + a'_{1m}u_m &= r'_1 \\ u_2 + a''_{23}u_3 + \ldots + a''_{2m}u_m &= r''_2 \\ \vdots &= \vdots \\ u_m &= r_3^{(m)} \end{aligned} \tag{3.3}$$

Esta redução é feita seqüencialmente para cada equação. Inicialmente, a primeira equação é dividida por a_{11}. Desta maneira os coeficientes a'_{1j} e r'_1 são obtidos. O valor de u_1 obtido na primeira equação é substituído nas demais equações. Com isso, chega-se ao seguinte sistema de equações:

$$\begin{aligned} u_1 + \frac{a_{12}u_2}{a_{11}} + \frac{a_{13}u_3}{a_{11}} + \ldots &= \frac{r_1}{a_{11}} \\ \left(a_{22} - \frac{a_{21}a_{12}}{a_{11}}\right)u_2 + \left(a_{23} - \frac{a_{21}a_{13}}{a_{11}}\right)u_3 + \ldots &= r_2 - \frac{a_{21}r_1}{a_{11}} \\ \vdots & \\ \left(a_{n2} - \frac{a_{n1}a_{12}}{a_{11}}\right)u_2 + \left(a_{n3} - \frac{a_{n1}a_{13}}{a_{11}}\right)u_3 + \ldots &= r_n - \frac{a_{n1}r_1}{a_{11}} \end{aligned}$$

ou

$$\begin{aligned} u_1 + a'_{12}u_2 + a'_{13}u_3 + \ldots &= r'_1 \\ a'_{22}u_2 + a'_{13}u_3 + \ldots &= r'_2 \\ \vdots & \\ a'_{n2}u_2 + a'n3u_r + \ldots &= r'_n \end{aligned}$$

O mesmo procedimento deve ser feito, começando pela equação seguinte (no caso, a equação 2) até percorrer todo o sistema e chegar ao sistema 3.3. A solução deste sistema é rápida, bastando usar o algorítimo chamado de *retrosubstituição*, a ser descrito a seguir.

Retrosubstituição O sistema 3.3 é extremamente simples: A partir da última equação pode-se determinar o valor de u_n. Substituindo este valor na

equação $n-1$, determina-se o valor de u_{n-1}. Repetindo esta operação até a primeira equação, obtém-se a solução do sistema rapidamente. O algorítimo pode ser descrito com a seguinte equação:

$$u_i = \frac{1}{a_{ii}^{(n)}} \left[r_i^{(i)} - \sum_{j=i+1}^{n} a_{ij}^{(i)} u_j \right] \tag{3.4}$$

O método de eliminação de Gauss está implementado na subrotina **GAUSS** do programa criado. Este método é o *default*, ou seja, é o método que todos os algorítimos usam *a priori*.

3.1.3 Decomposição LU

Este método, também conhecido como método de Khaletsky, descrito por Press[5], procura separar a matriz dos coeficientes A em duas matrizes triangulares (superior e inferior):

$$[L] \cdot [U] = [A]$$

Onde $[L]$ é a matriz triangular inferior e $[U]$ é a matriz triangular superior. Esta decomposição pode ser usada para a resolução de sistemas lineares.

$$[A] \cdot \{x\} = ([L] \cdot [U]) \cdot \{x\} = [L] \cdot ([U] \cdot \{x\}) = \{b\}$$

primeiro resolvendo para o vetor $\{y\}$, tal que

$$[L] \cdot \{y\} = \{b\} \tag{3.5}$$

$$[U] \cdot \{x\} = \{y\} \tag{3.6}$$

As equações 3.5 e 3.6 podem ser resolvidas com o algorítimo de retrosubstituição descrito em 3.1.2. Observe também que tendo obtido a decomposição LU do sistema, pode-se resolver muito rapidamente o sistema para qualquer

vetor $\{b\}$. Ou seja, este método é recomendado para a inversão de matrizes. Outro ponto positivo deste método é a facilidade de obtenção de determinantes de matrizes: basta multiplicar as diagonais das matrizes $[L]$ e $[U]$.

Decomposição LU da matriz

A decomposição LU da matriz A resulta em:

$$\begin{pmatrix} \alpha_{11} & 0 & 0 & \dots & 0 \\ \alpha_{21} & \alpha_{22} & 0 & \dots & 0 \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & \dots & 0 \\ \vdots & & & & \\ \alpha_{n1} & \alpha_{n2} & \alpha_{n3} & \dots & \alpha_{nn} \end{pmatrix} \cdot \begin{pmatrix} \beta_{11} & \beta_{12} & \beta_{13} & \dots & \beta_{1n} \\ 0 & \beta_{22} & \beta_{23} & \dots & \beta_{2n} \\ 0 & 0 & \beta_{23} & \dots & \beta_{3n} \\ \vdots & & & & \\ 0 & 0 & 0 & \dots & \beta_{nn} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{31} & a_{32} & \dots & a_{3n} \\ \vdots & & & \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \tag{3.7}$$

Como determinar $[L]$ e $[U]$ a partir de $[A]$? O componente i, j da matriz $[A]$ vale (3.7)

$$a_{ij} = \alpha_{i1}\beta_{1j} + \dots$$

O número de termos nesta soma depende de i ou j ser maior. Três casos são possíveis:

$$i < j: \quad \alpha_{i1}\beta_{1j} + \alpha_{i2}\beta_{2j} + \cdots + \alpha_{ii}\beta_{ij} = a_{ij} \tag{3.8}$$

$$i = j: \quad \alpha_{i1}\beta_{1j} + \alpha_{i2}\beta_{2j} + \cdots + \alpha_{ii}\beta_{jj} = a_{ij} \tag{3.9}$$

$$i > j: \quad \alpha_{i1}\beta_{1j} + \alpha_{i2}\beta_{2j} + \cdots + \alpha_{ij}\beta_{jj} = a_{ij} \tag{3.10}$$

As equações acima formam um sistema de N^2 equações para $N^2 + N$ incógnitas αs e βs (a diagonal é representada duas vezes). Portanto pode-se especificar N variáveis, por exemplo,

$$\alpha_{ii} \equiv 1 \qquad i = 1, \dots, N \tag{3.11}$$

O *algorítimo de Crout* permite, de maneira simplificada, resolver o sistema de equações 3.7. Para tanto, basta reagrupar as equações da seguinte maneira:

- Fazer $\alpha_{ii} = 1, i = 1, \ldots, N$
- Para cada $j = 1, 2, 3, \ldots, N$ realizar os seguintes procedimentos:
  1. Para $i = 1, \ldots, j$ utilizar as equações 3.8, 3.9, 3.11 para calcular β_{ij}, ou seja,

$$\beta_{ij} = a_{ij} - \sum_{k=1}^{i-1} \alpha_{ik}\beta_{kj}$$

  2. Para $i = j+1, j+2, \ldots, N$ utilizar 3.10 para determinar α_{ij}:

$$\alpha_{ij} = \frac{1}{\beta_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} \alpha_{ik}\beta_{kj} \right)$$

O interessante deste procedimento é que os αs e βs necessários nas equações acima são conhecidos no momento em que são utilizados. Outro ponto a destacar é que todo o a_{ij} é usado apenas uma vez e portanto os parâmetros α_{ij} e β_{ij} podem ser armazenados na posição de a_{ij}.

A decomposição LU da matriz dos coeficientes está implementada na função `LUDCMP` do programa desenvolvido. Nesta função, a matriz dos coeficientes é que retorna a decomposição LU. A função `LUBKSB` faz a retrosubstituição, ou seja, resolve as equações 3.5 e 3.6.

Obtenção da matriz inversa

A inversão de matrizes é necessária no método de *Newton-Raphson* modificado na primeira iteração (seção 3.5). A inversão de uma matriz de ordem n envolve a resolução de n sistemas de equações lineares com a *mesma matriz*

de coeficientes. Dada a maneira como foi implementada a decomposição LU, com a decomposição sendo feita separado da retrosubstituição, as subrotinas desenvolvidas foram utilizadas para a inversão de matrizes.

Tendo determinado a decomposição LU da matriz $[A]$, a determinação da inversa é uma tarefa simples: basta aplicar o algorítimo de retrosubstituição para cada coluna (equações 3.5 e 3.6) nos seguintes sistemas:

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & & & \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \cdot \begin{Bmatrix} b_{1i} \\ b_{2i} \\ \vdots \\ b_{ni} \end{Bmatrix} = \begin{Bmatrix} \delta_{1i} \\ \delta_{2i} \\ \vdots \\ \delta_{ni} \end{Bmatrix} \qquad i = 1, \dots, n \qquad (3.12)$$

Na equação 3.12, δ_{ij} é o delta de Kronecker e vale

- $\delta_{ij} = 0, i \neq j$
- $\delta_{ij} = 1, i = j$

A inversão de matrizes usando a decomposição LU é feita pela subrotina `INVERTLU` do programa desenvolvido.

3.1.4 *Conjugate gradient*

Este é um método iterativo muito robusto e de rápida convergência. Este método é muito usado na solução de sistemas grandes,

$$[A] \cdot \{x\} = \{b\}$$

A vantagem deste método é que a matriz $[A]$ é referenciada apenas através de multiplicações desta matriz (ou sua transposta) por um vetor. Existem diferentes possibilidades. A mais simples procura minimizar o seguinte funcional:

$$f(\{x\}) = \frac{1}{2}\{x\} \cdot [A] \cdot \{x\} - [b] \cdot \{x\} \qquad (3.13)$$

Esta função é minimizada quando seu gradiente é nulo:

$$\{\nabla f\} = [A] \cdot \{x\} - \{b\} = \{0\} \tag{3.14}$$

3.1.5 Algorítimo de Thomas para sistemas tridiagonais

Sistemas tridiagonais ocorrem com freqüência na solução numérica de problemas. Um sistema tridiagonal é dado pela seguinte equação:

$$\begin{pmatrix} b_1 & c_1 & 0 & \dots & 0 \\ a_1 & b_1 & c_1 & \dots & 0 \\ \vdots & & & & \\ 0 & \dots & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & \dots & 0 & a_1 & b_1 \end{pmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{Bmatrix} \tag{3.15}$$

A vantagem do algorítimo de thomas na solução deste tipo de sistema é sua rapidez. Além disto, é muito eficiente no uso da memória do computador: Basta armazenar espaço para três vetores de tamanho n, a_i, b_i e c_i.

O algorítimo é muito simples. Resolvendo a primeira equação do sistema para x_1 e substituindo este valor na segunda equação, o sistema fica com mesmo formato que o sistema original mas com uma equação a menos. Isto pode ser feito sucessivamente até atingir a última equação. Desta maneira pode-se rapidamente chegar à solução do sistema.

Observe que no caso presente, os sistemas a serem resolvidos não terão formato tridiagonal mas resolvendo-se o sistema iterativamente, ou seja, estimando um valor inicial para as variáveis e resolvendo vários problemas tridiagonais sucessivamente pode-se chegar à solução do sistema.

3.2 Cálculo Numérico de Derivadas

O cálculo da matriz Jacobiana $J(\vec{x})$ é essencial à solução de sistemas de equações não-lineares em métodos como o *Newton-Raphson*. A matriz Jacobiana é definida por

$$J^m\left(x^m\right) = \left[\frac{\partial f_i}{\partial x_j}\right] \tag{3.16}$$

A definição de derivada parcial é dada por:

$$\frac{\partial f_i}{\partial x_j} = \lim_{h \to 0} \frac{f_i\left(x_1, \ldots, x_j + h, \ldots, x_n\right) - f_i\left(x_1, \ldots, x_j, \ldots, x_n\right)}{h} \tag{3.17}$$

No programa, a derivada foi estimada usando uma expressão de quarta ordem:

$$y'(x) = \frac{-y(x+2h) + 8y(x+h) - 8y(x-h) + y(x-2h)}{12h} + O\left(h^4\right) \tag{3.18}$$

3.3 O Método de *Newton-Raphson*

Este é o método básico para a resolução de sistemas de equações não linerares. Deste momento em diante, o método será denominado *NR*.

Este método, quando converge, é muito rápido em sistemas de pequena ordem (pequeno número de equações). O método consiste em tentar estimar a raíz do sistema fazendo uma expansão em série de taylor das equações do sistema. Esta série é truncada, permanecendo apenas os termos lineares (de primeira ordem). Considere um sistema de equações da seguinte forma:

$$f_i\left(x_1, x_2, \ldots, x_n\right) = 0 \qquad i = 1, \ldots, n \tag{3.19}$$

onde

f_i são as funções do sistema

x_i são as variáveis independentes

n é o número de equações que deve ser igual ao número de incógnitas

A expansão em série de taylor as funções do sistema 3.19 para a iteração m, resulta:

$$f_i\left(x^m + \Delta x^m\right) = f_i\left(x^m\right) + \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j} \Delta x_j^m + O\left(\Delta x^2\right) = 0 \qquad i = 1, \ldots, n \tag{3.20}$$

Usando a definição do Jacobiano (J, equação 3.16) , o incremento das variáveis dependentes vale

$$\Delta x^m = J^{-1}\left(x^m\right)\left[-f\left(x^m\right)\right] \tag{3.21}$$

Lembrando que $x^{m+1} = x^m + \Delta x^m$, determina-se uma estimativa melhor para as variáveis dependentes a cada iteração.

Observe a necessidade de cálculo de derivadas a cada iteração, assim como a inversão de uma matriz (J^{-1}) (vide seção 3.1). Estas são duas desvantagens para sistemas de grande porte. A inversão de matrizes de grande porte requer um trabalho computacional grande, assim como o cálculo de derivadas. Uma solução simples é, se a solução já estiver próxima do valor correto, empregar a matriz já calculada e invertida durante algumas iterações. Esta solução pode tornar o sistema instável mas certamente tornará o método mais rápido.

No programa, o método de *Newton-Raphson* está implementado na função `SOLVE_NEWTONRAPHSON`.

3.4 O Método da Continuação (ou *Homotopy*)

Este método, conhecido como *continuation method* ou *homotopy* é um método lento mas extremamente robusto, ou seja, converge quase sempre e é

muito insensível a valores iniciais. Este método descrito por Hanna[1] foi uma grata surpresa ao autor pois permite a solução de sistemas com valores iniciais muito pobres e, melhor, obtenção de valores iniciais bons para um método mais rápido porém menos robusto como por exemplo o método de *Newton-Raphson* ou *Newton-Raphson* modificado.

Supondo que o valor inicial seja x_0, pode-se construir a função $F(\lambda) = f[x(\lambda)]$ de modo que para $\lambda = 0$ esta função vale $f(x_0)$ e para $\lambda = 1$ a função vale 0. Uma possibilidade para esta função é:

$$f[x(\lambda)] = (1 - \lambda) f(x_0) \tag{3.22}$$

Derivando este sistema de equações em relação a lambda, temos

$$\frac{df_i(\lambda)}{d\lambda} = \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j} \frac{dx_j}{d\lambda} == f_i(x_0) \qquad i = 1, \ldots, n$$

A equação acima, usando a definição de jacobiano (equação 3.16), pode ser reescrita da seguinte maneira:

$$[J(x)] \left[\frac{dx}{d\lambda}\right] = - [f(x_0)] \tag{3.23}$$

A equação acima pode ser resolvida para $\left[\frac{dx}{d\lambda}\right]$:

$$\left[\frac{dx}{d\lambda}\right] = - \left[J^{-1}(x)\right] [f(x_0)] \tag{3.24}$$

A equação 3.24 pode ser facilmente integrada de $\lambda = 0$ ($x = x_0$) a $\lambda = 1$, determinando assim $x = x^*$ onde x^* é a solução do sistema (ou melhor, uma estimativa da solução).

A grande desvantagem deste método é a necessidade do cálculo da matriz jacobiana a cada iteração e a necessidade de invertê-la. Como o número de iterações é grande (para qualquer problema), uma boa estimativa da solução

do problema, se comparado ao método de *Newton-Raphson* (quando este converge), requer um trabalho computacional considerável. e portanto a solução é lenta.

Este método foi implementado em duas subrotinas separadas. A primeira, `SOLVE_METHCON`, o *core* do método, integra 3.24 para um número fixo de passos (uma discretização em λ fixa). A solução obtida pode ou não ser uma boa estimativa da solução do problema. A integração é feita para passos constantes pelo método de *Euler* (a ser descrito adiante). Nesta subrotina não é feito nenhum tipo de verificação de convergência.

A segunda subrotina (`SOLVE_METHCON2`) tem como parâmetro de entrada o resíduo desejado de convergência. Esta subrotina chama `SOLVE_METHCON` sucessivamente até ser obtida uma solução para o problema dentro do resíduo desejado.

Vale lembrar que a discretização em λ é um parâmetro extremamente importante na solução do sistema, como pode ser visto em 3.8.

3.5 O Método de *Newton-Raphson* Modificado

O grande problema do método de *Newton-Raphson* é a sua sensibilidade em relação ao valor inicial e também a necessidade de cálculo da matriz jacobiana e inversão desta matriz a toda iteração. Várias modificações foram propostas, entre elas, a descrita por Stoecker[14]. Este método é menos sensível ao valor inicial e também elimina a necessidade de inversão de matrizes e cálculo de derivadas. Neste trabalho, o método se mostrou tão robusto quanto o método de *Newton-Raphson* mas muito rápido, convergindo em poucas iterações e com tempo de processamento por iteração inferior ao método de *Newton-*

Raphson.

Neste método, a primeira iteração é idêntica ao método de *Newton-Raphson*, ou seja

$$\Delta x^1 = -J^{-1}(x_0) f(x_0)$$

Chamando $[H_0] = [J^{-1}(x_0)]$, o valor das variáveis são corrigidas pela seguinte equação:

$$[X] = -[H][F] \tag{3.25}$$

Onde $F_i = f(x_i)$ e $X_i = \Delta x_i$. A cada iteração, a matriz H é corrigida de acordo com a seguinte relação (Stoecker[14]):

$$H_{k+1} = H_k + \frac{(X_k - H_k Y_k)\, X_k^T H_k}{X_k^T H_k Y_k} \tag{3.26}$$

Onde:

$X_k = \Delta x^k$

$Y_k = F_{k+1} - F_k$

$F_k = f(x_k)$

Este método, denominado ST neste trabalho, está implementado na subrotina `SOLVE_STOECKER` do programa desenvolvido

3.6 O Método do Gradiente Máximo

Este método emprega o conceito de busca para determinar a solução do sistema de equações. Seja o sistema $f(x) = 0$. A solução deste sistema x^* é um mínimo da seguinte função:

$$g(x) = \sum_{i=1}^{n} [f_i(x))]^2$$

Agora, a direção da máxima variação de $g(x)$ é dada pelo gradiente de $g(x)$, $\nabla g(x)$. Desta forma, uma estimativa para a solução é dada por:

$$x^{(p+1)} = x^{(p)} - \lambda_p \nabla g(x^p) \tag{3.27}$$

Falta agora determinar $\nabla g(x)$ e λ_p. Pode-se mostrar (vide Demidovich[2]) que

$$\nabla g(x) = 2J^T(x)f(x) \tag{3.28}$$

Onde $J^T(x)$ é a transposta da matriz jacobiana. Chamando $\mu_p = 2\lambda_p$, obtém se a seguinte expressão (Demidovich[2]):

$$\mu_p = \frac{f^{(p)} \cdot J_p J_p^T f^{(p)}}{J_p J_p^T f^{(p)} \cdot J_p J_p^T f^{(p)}} \tag{3.29}$$

Assim, substituindo 3.28 e 3.29 em 3.27 obtém-se a seguinte relação para a nova estimativa da raíz do sistema:

$$x^{(p+1)} = x^{(p)} - \mu_p J_p^T f^{(p)} \tag{3.30}$$

Diversas referências e a experiência do autor permitem afirmar que este método é muito robusto com relação aos valores iniciais mas é lento (no entanto, esta lentidão só ocorre em alguns problemas de difícil convergência). Este método é excelente para a obtenção de valores iniciais que podem ser utilizados no método de *Newton-Raphson* ou *Newton-Raphson* modificado.

Este método, com algumas modificações, pode ser implementado de modo a resolver um sistema de equações lineares. Um exemplo é o método *Conjugate Gradient*. Este método é muito aplicado em *CFD* para a solução iterativa de sistemas de equações lineares esparsos de grande porte.

3.7 Critérios de Convergência

Esta seção trata do critério de convergência na solução de sistemas de equações não-lineares. Este é um tópico extremamente importante já que saber se a solução já convergiu é essencial (ainda mais num *solver* geral como o proposto).

Diversas alternativas existem, entre elas:

1. Verificar o valor do resíduo máximo ($f_i^{max}(x)$)
2. Variação máxima das variáveis dependentes ($[x^{i+1} - x^i]^{max}$)
3. Variação máxima relativa $\left[\frac{x^{i+1}-x^i}{x^{i+1}}\right]^{max}$
4. Composição das alternativas acima ou outros

Dado o caráter geral do solver, resolveu-se adotar duas alternativas como critério de parada:

- Verificar o valor do resíduo máximo ($f_i^{max}(x)$)
- Variação máxima relativa $\left[\frac{x^{i+1}-x^i}{x^{i+1}}\right]^{max}$

Desta forma, pode-se ter alguma segurança de que o problema realmente convergiu.

3.8 Comparação dos métodos

Diversos sistemas foram resolvidos para verificar as principais vantagens e desvantagens dos métodos implementados. As características a serem observadas são:

- Robustez do método em relação aos valores iniciais
- Número de iterações para a convergência
- Custo computacional associado

Diferentes sistemas, propostos por diferentes autores ([14], [1], [2] ou [3] por exemplo) foram resolvidos pelos diferentes métodos para diferentes valores iniciais. O *solver* desenvolvido também foi comparado a softwares comerciais (no caso *EES*) afim de verificar o desempenho real do programa desenvolvido.

3.8.1 Sistema de equações 1

Este sistema é composto pela equação abaixo:

$$x^2 - 5x + 1 = 0 \tag{3.31}$$

Esta equação é simples e tem solução analítica:

1. $x_1 = 0,208712$
2. $x_2 = 4,7912878$

Os diversos métodos implementados foram usados para a solução desta equação

Para diferentes valores iniciais, o problema convergiu para as duas diferentes soluções.

Método de *Newton-Raphson*

Para valores iniciais altos (acima de 2) o sistema convergiu rapidamente para x_2. Para valores inferiores, o sistema convergiu para x_1. O número de iterações foi considerável quando o valor inicial era próximo de 2.

Método de *Newton-Raphson* modificado

Os resultados para o método de *Newton-Raphson* modificado foram análogos ao método de *Newton-Raphson* exceto pelo número de iterações, que se mostrou bastante inferior em geral.

Método da continuação

Quanto à sensibilidade ao valor inicial, o método se mostrou idêntico aos métodos anteriores. Observe, no entanto a necessidade de um número muito alto de iterações para se obter uma boa estimativa.

O problema foi analisado para diferentes números de passos. Quanto mais baixo o número de passos, mais rapidamente o problema convergiu.

Método do gradiente máximo

Este método teve desempenho igual ao método de *Newton-Raphson* e *Newton-Raphson* modificado. Tanto no número de iterações quanto na sensibilidade do valor inicial.

3.8.2 Sistema de equações 2

Este segundo problema é um sistema de duas equações e duas incógnitas:

$$\begin{aligned} x^2 + y^2 &= 1 \\ x &= 2y \end{aligned} \tag{3.32}$$

Este problema também apresenta solução analítica (no caso duas):

1. $x_1 = 0,8944272$ e $y_1 = 0,4472136$

2. $x_2 = -0,8944272$ e $y_2 = -0,4472136$

Método de *Newton-Raphson* - NR

Para os valores *default*, o sistema convergiu em 18 iterações para a solução 1. Trabalhando com valores negativos de valores iniciais, o método convergiu para a solução 2 com aproximadamente a mesma velocidade

Método de *Newton-Raphson* modificado - ST

Convergência rápida para a solução do problema, aproximadamente 5 iterações. Mesmo comportamento do método de *Newton-Raphson* em relação aos valores iniciais.

Método da continuação - MC

Novamente, o método é muito dependente da discretização em λ utilizada. Os resultados podem ser observados na tabela 3.1.

Tabela 3.1: Número de iterações necessárias para convergência do sistema 2 para diferentes discretizações em λ

	Discretização em λ	Nº de iterações
1	5	48
2	10	66
3	30	124
4	50	153

Na tabela 3.1 observa-se claramente que quanto maior o número de passos, mais lentamente o problema converge. Será observado posteriormente que embora o método torne-se mais lento, este também se torna mais robusto.

Método do gradiente máximo

Neste caso, a convergência se deu muito rapidamente, em apenas 4 iterações. O comportamento deste método em relação ao valor inicial foi idêntico aos outros métodos.

3.8.3 Sistema de equações 3

Este sistema tem como propósito testar os diferentes métodos em um sistema sem solução:

$$\begin{aligned} x^2 + y^2 &= 2 \\ x^2 + y^2 &= 1 \end{aligned} \tag{3.33}$$

Neste caso, todos métodos acusaram alguma forma de erro sem abortar o programa de maneira abrupta.

3.8.4 Sistema de equações 4

Este sistema foi retirado de Stoecker[14], problema 14-7:

$$\begin{aligned} (-5 - t_e)\left(1 - e^{-\frac{UA_e}{wc}}\right) - \frac{q_e}{wc} &= 0 \\ (t_c - 30)\left(1 - e^{-\frac{UA_c}{wc}}\right) - \frac{q_c}{wc} &= 0 \\ 204 + 2,4t_e + 0,02t_e t_c - 1,5t_c - q_e &= 0 \\ q_c - 1,2q_e &= 0 \end{aligned} \tag{3.34}$$

Neste sistema, $UA_c = 13,31kW/K, UA_e = 7,621kW/K$ e $wc = 10kW/K$. Este sistema modela um condensador.

Neste problema, foram adotados como valores iniciais os valores default $(1,0)$.

Método de *Newton-Raphson*

O sistema convergiu para a resposta correta em 25 iterações.

Método de *Newton-Raphson* modificado

O sistema convergiu para a solução correta em 4 iterações, um resultado excelente.

Método da continuação

Novamente, o método é muito dependente da discretização em λ utilizada. Os resultados podem ser observados na tabela 3.2.

Tabela 3.2: Número de iterações necessárias para convergência do sistema 4 para diferentes discretizações em λ

	Discretização em λ	Nº de iterações
1	5	66
2	10	88
3	30	155
4	50	255

Método do gradiente máximo

O método do gradiente máximo convergiu para a solução correta mas em um número excessivo de iterações (acima de 1000 iterações).

3.8.5 Sistema de equações 5

Este é o último sistema de equações simples:

$$\begin{aligned} 0,0625 + 0,653Q^{1,8} &= P \\ 0,3 - 0,2Q^{2} &= P \end{aligned} \tag{3.35}$$

Novamente foram usados os valores *default* como valores iniciais.

Método de *Newton-Raphson*

O problema convergiu para a solução correta em 17 iterações.

Método de *Newton-Raphson* modificado

O problema convergiu muito rapidamente para a solução: apenas 5 iterações,

Método da continuação

A dependência da velocidade de convergência com a discretização pode ser observada na tabela 3.3.

Método do gradiente máximo

Neste problema, o método do gradiente máximo convergiu para a solução correta em 25 iterações.

Tabela 3.3: Número de iterações necessárias para convergência do sistema 5 para diferentes discretizações em λ

	Discretização em λ	N^o de iterações
1	5	48
2	10	55
3	30	124
4	50	153

3.8.6 Sistema de equações 6

Este sistema, retirado de Hanna[1] é um problema patológico. A convergência é extremamente complicada. As equações foram obtidas no modelamento de equilíbrio de um conjunto de reações químicas. As variáveis y_i correspondem a concentrações de espécies e as variáveis x_i correspondem a coordenadas da

reação. O sistema de equações está representado a seguir:

$$
\begin{aligned}
&\frac{y_3 y_4}{y_1^3 y_2} = 69,18 \\
&\frac{y_5 y_1}{y_2 y_4} = 4,68 \\
&\frac{y_2^2}{y_5} = 0,0056 \\
&\frac{y_6 y_4^2}{y_1^5 y_2^2} = 0,141 \\
&y_1 = \frac{3 - 3x_1 + x_2 - 5x_4}{D} \\
&y_2 = \frac{1 - x_1 - x_2 + 2x_3 - 2x_4}{D} \\
&y_3 = \frac{x_1}{D} \\
&y_4 = \frac{x_1 - x_2 + 2x_4}{D} \\
&y_5 = \frac{x_2 - x_3}{D} \\
&y_6 = \frac{x_4}{D} \\
&D = 4 - 2x_1 + x_3 - 4x_4
\end{aligned}
\tag{3.36}
$$

A primeira análise foi feita para valores iniciais *default*. A convergência foi extremamente difícil, ocorrendo apenas para o método da continuação com número de passos grande (acima de 50). O sistema também convergiu para o método de *Newton-Raphson* para valores de subrelaxação inferiores a 0,05. Como resultado, o número de iterações foi enorme: acima de 400 em ambos os casos. Os outros métodos não conseguiram convergir. O mesmo sistema com os mesmos valores iniciais foi testado no software *EES*: não houve convergência.

Uma segunda análise foi feita, adotando para todas as variáveis um valor inicial de 0,5. A convergência foi muito mais fácil. No método de *Newton-Raphson* com 0,1 de subrelaxação, o problema convergiu em 155 iterações. Usando estes mesmos valores iniciais no *EES*, o problema continuou não

convergindo. O método da continuação com 50 passos convergiu em 300 iterações.

A terceira análise consistiu em uma combinação de métodos: Usar o método da continuação para gerar um valor inicial melhor para o método de *Newton-Raphson* ou *Newton-Raphson* modificado. Os resultados para diferentes números de passos estão apresentados a seguir:

Combinação - método da continuação e método de Newton-Raphson modificado

Os resultados podem ser observados na tabela 3.4:

Tabela 3.4: Número de iterações para convergência usando o método da continuação para gerar valores iniciais para o método de Newton-Raphson modificado

	Número de passos	Número de Iterações (ST)
1	20	NÃO CONVERGIU
1	30	NÃO CONVERGIU
1	40	NÃO CONVERGIU
1	50	NÃO CONVERGIU
1	60	26

Combinação - método da continuação e método de Newton-Raphson

Os resultados podem ser observados na tabela 3.5:

Observe que na tabela 3.5 a subrelaxação foi de 1,0. Para uma subrelaxação de 0,5 o número de passos necessários seria 50 convergindo em 28 iterações. Adotando como subrelaxação 0,2 o sistema convergiu com 4o passos em 89 iterações.

Tabela 3.5: Número de iterações para convergência usando o método da continuação para gerar valores iniciais para o método de Newton-Raphson

	Número de passos	Número de Iterações (NR)
1	20	NÃO CONVERGIU
1	30	NÃO CONVERGIU
1	40	NÃO CONVERGIU
1	50	NÃO CONVERGIU
1	60	7

A tabela 3.6 mostra como o porblema é de difícil convergência. A tabela compara a solução do problema com o valor inicial gerado para 20 passos no método da continuação.

Tabela 3.6: Comparação entre a solução exata do problema com valor estimado pelo método da continuação com 20 passos

Variável	Valor exato	Estimativa N=20
x_1	0,681604	0,637
x_2	1,58962E-02	0,08
x_3	-0,128703	-0,066
x_4	1,40960E-05	0,055
y_1	0,387162	0,38
y_2	0,0179676	0,0063
y_3	0,271769	0,241
y_4	0,265442	0,251
y_5	0,0576549	0,0488
y_6	5,62034E-06	-1,76E-06

A tabela 3.6 mostra que mesmo para um número de passos pequeno (20) o valor gerado é próximo do valor exato mas mesmo assim o método de *Newton-Raphson* (com subrelaxação 1) ou *Newton-Raphson* modificado não convergem.

3.8.7 Sistema de equações 7

Este sistema, retirado de Carnahan[3], não apresenta nenhum problema de convergência se um valor inicial apropriado for fornecido. O problema descreve um equilíbrio químico e as variáveis são frações molares dos diversos componentes e portanto devem ser superiores a zero e inferiores a um.

O sistema é dado pelas equações a seguir:

$$
\begin{aligned}
&0,5x_1 + x_2 + 0,5x_3 - \frac{x_6}{x_7} = 0 \\
&x_3 + x_4 + 2x_5 - \frac{2}{x_7} = 0 \\
&x_1 + x_2 + x_5 - \frac{1}{x_7} = 0 \\
&-28837x_1 - 139009x_2 - 78213x_3 + 18927x_4 + 8427x_5 + \frac{13492}{x_7} - 10690\frac{x_6}{x_7} = 0 \\
&x_1 + x_2 + x_3 + x_4 + x_5 - 1 = 0 \\
&400x_1x_4^3 - 1,7837 \cdot 10^5 x_3x_5 = 0 \\
&x_1x_3 - 2,6058x_2x_4 = 0
\end{aligned}
\tag{3.37}
$$

Adotando como valor inicial de todas as variáveis 0.1, os resultados do diferentes métodos são descritos a seguir.

Método de Newton-Raphson

O sistema convergiu para a solução correta em 38 iterações.

Método de Newton-Raphson modificado

Inicialmente, o sistema não convergiu mas usando o método da continuação com 10 passos bastou para o sistema convergir em 12 iterações.

Método da Continuação

A convergência para diferentes número de passos pode ser vista na tabela 3.7.

Tabela 3.7: Número de iterações necessárias para convergência do sistema 7 para diferentes discretizações em λ

	Discretização em λ	N^o de iterações
1	5	78
2	10	99
3	30	217
4	50	306

Novamente, a tabela 3.7 mostra que para número de passos maiores levam a um número maior de iterações e portanto maior esforço computacional. Vale lembrar que o método se torna muito mais robusto.

Com estes mesmos valores iniciais o *EES* encontrou um resultado não realístico (solução 3, tabela 3.8).

Atribuindo agora o valor *default* (1,0) às variáveis (valor inicial) o problema se torna mais complexo: Outras soluções são encontradas. Estas outras soluções, apesar de corretas não têm sentido físico. A tabela 3.8 mostra as diferentes soluções obtidas.

Tabela 3.8: Diferentes soluções obtidas para o sistema de equações 7

Variáveis	Solução Correta(1)	Outra Solução(2)	Mais Outra(3)
x_1	0,322871	-0,1897470	0,4569306
x_2	0,009224	0,1897470	-4,071984E-04
x_3	0,046017	2,282343E-13	-2,125199E-03
x_4	0,618172	6,794155E-13	0,9151719
x_5	0,003717	1,0	-0,3695701
x_6	0,576715	0,9487348	9,487348E-02
x_7	2,977863	1,0	1,0

3.9 Comentários Finais

Neste capítulo, diferentes métodos para a solução de sistemas de equações não lineares foram analisados. Na seção 3.8 estes diferentes métodos foram testados para diferentes sistemas.

Os exemplos da seção anterior mostram algumas características dos diferentes métodos. Pode-se observar que os métodos mais rápidos certamente são os métodos de *Newton-Raphson* e *Newton-Raphson* modificado. O método de *Newton-Raphson* modificado em algumas circunstâncias é extremamente rápido (número de iterações muito pequeno) mas não é muito robusto. Além disto, este método, como foi visto na seção 3.5 não realiza cálculos de derivadas nem resolve sistemas de equações lineares, tornando muito atraente para sistemas com várias equações.

O método da continuação se mostrou uma grata surpresa ao autor. É um método muito lento mas excelente para a obtenção de estimativas iniciais mais apropriadas. O método é muito robusto. O método do gradiente máximo teve um desempenho inferior ao esperado mas funcionou relativamente bem. A possibilidade de variar a subrelaxação no método de *Newton-Raphson*

permite tornar este método mais robusto como se pôde observar na solução do sistema de equações 6. Press[5] propõe um método híbrido de solução de sistemas de equações não lineares que seria basicamente um método de *Newton-Raphson* com subrelaxação variável. Este parâmetro deria determionado de acordo com algum algorítimo de otimização (parecido com o método do gradiente máximo).

Outro ponto positivo do programa desenvolvido é a possibilidade de compor diferentes métodos para a solução de sistemas de equações não lineares. Ou seja, um método, que pode ser muito robusto, porém lento, pode ser utilizado em algumas iterações, com o objetivo de se obter uma solução aproximada. Esta solução aproximada seria usada como valor inicial para um método rápido, por exemplo, o método de *Newton-Raphson* modificado. Este procedimento foi adotado em alguns dos casos analisados, por exemplo o sistema de equações 7.

Capítulo 4

Solução de Sistemas de Equações Diferenciais Ordinárias

O uso de prâmetros concentrados para o modelagem de sistemas térmicos é caracterizado pela existência de variáveis de estado. Estas variáveis de estado determinam a configuração instantânea do sistema. Observe que no caso de modelos de parâmetros distribuídos, não é possível definir estas variáveis de estado, ou melhor, para se caracterizar o sistema, seriam necessárias infinitas variáveis de estado, o que é impossível.

Sistemas modelados por parâmetros concentrados em regime permanente resultam em sistemas de equações algébricas não lineares. Estes modelos, entretanto, não permitem tirar conclusões sobre a dinâmica do processo. Isto é muito importante quando se deseja saber quanto tempo leva para o sistema se estabilizar, o que ocorre durante a partida (inicialização) do sistema ou o que ocorre durante uma mudança na condição de funcionamento do sistema.

O uso das variáveis de estado resulta em sistemas de equações diferenciais ordinárias. No caso geral, este sistema é não linear, dificultando assim a análise do problema no domínio de *Laplace*. Desta forma, foi decidido

analisar o problema no domínio do tempo. Neste caso, a simulação consiste na integração de um sistema de equações diferenciais ordinárias com valor inicial.

4.1 Sistema de Equações Diferenciais Ordinárias

Existem diversas possibilidades para a representação dos sistemas de equações diferenciais ordinárias. A mais comum e também a de mais fácil manipulação pode ser observada na equação 4.1:

$$\frac{dx_i}{dt} = f_i\,(x_1, x_2, \ldots, x_n,) \qquad i = 1, \ldots, n \tag{4.1}$$

Neste trabalho, o formato da equação 4.1 será empregado.

Existem diversos métodos para a solução de sistemas de equações diferenciais ordinárias. Muitos destes métodos estão baseados na expansão em série de Taylor de uma função, ou seja, se uma função é conhecida no ponto x_0, o seu valor no ponto $x_0 + h$ é dado por:

$$y\,(x_0 + h) = y(x_0) + y'(x_0)h + \frac{y''(x_0)h^2}{2!} + \frac{y'''(x_0)h^3}{3!} + \cdots \tag{4.2}$$

Considerando, como exemplo, um sistema de uma única variável e equação,

$$y'(x) = f(x, y) \tag{4.3}$$

pode-se, com a regra da cadeia, calcular y'', y''' e outras derivadas de ordem superior:

$$y''(x) = f_x + f_y y' = f_x + f_y f \tag{4.4}$$

$$y'''(x) = (f_{xx} + f_{xy} f) + f_y(f_x + f_y f) + f(f_{yx} + f_{yy} f) \tag{4.5}$$

Neste caso, os subíndices representam derivadas parciais. Fazendo-se uso destas derivadas, pode-se fazer uma estimativa de $y(x+h)$. Observe que para se conseguir uma melhor aproximação (erro de ordem maior), basta incluir mais termos da série de Taylor (eq. 4.2).

Os diferentes métodos discutidos neste trabalho partem da expansão em série de Taylor do sistema com algumas modificações a serem discutidas. Observe que, sem perda de generalidade, os diferentes métodos serão deduzidos para o caso de uma única equação e variável.

4.2 Método de Euler

Este é o método mais simples para a solução de sistemas de equações diferenciais ordinárias. Este método mantém apenas o primeiro termo da expansão da série de Taylor, e portanto,

$$y(x+h) \approx y(x) + y'(x)h = y(x) + f\left(x, y(x)\right) \tag{4.6}$$

Este método, apesar de muito simples e de fácil implementação apresenta precisão de primeira ordem apenas, o que pode não ser conveniente. Existem outros métodos baseados no de Euler, que apresentam maior precisão. Um exemplo é o método de Euler melhorado (seção 4.4).

4.3 Métodos de Runge-Kutta

Observa-se das equações 4.4 e 4.5 que no caso geral é muito difícil calcular algumas poucas derivadas de uma equação diferencial $y' = f(x, y)$ para uso na equação de Taylor (eq. 4.2). A implementação direta da expansão em série de Taylor requer diferentes derivadas para cada sistema, tornando o método inviável para um programa geral como o proposto.

Existe, no entanto, um método inteligente para eliminar este problema, atribuído a Runge e Kutta (Hanna[1]). Esta abordagem consiste em calcular as derivadas necessárias indiretamente, combinando primeiras derivadas calculadas em diferentes pontos. Isto na realidade é um cálculo numérico de derivadas. Este método permite calcular a expansão em série de Taylor da solução para qualquer sistema. Desta forma, pode-se implementar um algorítimo geral para a solução de sistemas de equações diferenciais ordinárias.

4.3.1 Método de Runge-Kutta de segunda ordem (*RK2*)

Nesta seção será deduzida as equações para o método de Runge-Kutta de segunda ordem. Será considerado o caso geral $y' = f(x, y)$, expandindo em série de Taylor e ignorando os termos de ordem superior a 2. Usando a equação 4.4:

$$y(x+h) = y(x) + hf\left[x, y(x)\right] + \frac{h^2}{2}\left(f_x + f_y f\right)_{x,y(x)} + O(h^3) \tag{4.7}$$

Deseja-se escrever $y''(x) = (f_x + f_y f)$ como uma combinação de valores de f calculado em x e y apropriados. Expandindo $f(x, y)$ em série de taylor,

$$f\left(x + h_1, y + k_1\right) = f(x, y) + f_x h_1 + f_y k_1 + O\left(h_1^2 + h_1 k_1 + k_1^2\right) \tag{4.8}$$

Considerando as equações 4.7 e 4.8, observa-se que $(f_x h_1 + f_y k_1)$ pode ser aproximadamente igual a $(f_x + f_y f) \equiv y''$ se $k_1 = f h_1$. Tomando $h_1 = ah$ então,

$$f(x + ah, y + ahf) = f(x, y) + ahy''(x) + O(h^2) \tag{4.9}$$

Desta maneira, a equação 4.7 toma a seguinte forma:

$$y(x+h) = hf\left[x, y(x)\right]\left(1 - \frac{1}{2a}\right) + \frac{h}{2a} f\left[x + ah, y + ahf(x, y)\right] + O(h^3) \tag{4.10}$$

Da equação 4.10 observa-se que na realidade existem infinitas aproximações de Runge-Kutta de segunda ordem, dependendo do valor de a. Tomando $a = 1$, a equação 4.10 se torna:

$$y(x+h) = y(x) + \frac{h}{2}\{f[x, y(x)] + f[x+h, y+hf(x, y(x))]\} + O(h^3) \tag{4.11}$$

4.3.2 Método de Runge-Kutta de quarta ordem (*RK4*)

De maneira análoga ao método RK2, pode-se chegar ao método de Runge-Kutta de quarta ordem. A dedução, extremamente trabalhosa e comprida foi retirada de Hanna[1].

Este método que foi o mais usado no passado é uma aproximação de quarta ordem (*RK4*) que necessita de quatro cálculos de derivadas por passo, como mostrado a seguir:

$$y(x+h) = y(x) + \frac{h}{6}[k_0 + k_1 + k_2 + k_3] \tag{4.12}$$

Onde

$$k_0 = f[x, y(x)]$$

$$k_1 = f\left[x + \tfrac{h}{2}, y(x) + \tfrac{k_0}{2}\right]$$

$$k_2 = f\left[x + \tfrac{h}{2}, y(x) + \tfrac{k_1}{2}\right]$$

$$k_3 = f[x+h, y(x) + k_2]$$

Outros métodos de Runge-Kutta podem ser deduzidos. Os métodos de Runge-Kutta de segunda e quarta ordem são, entretanto, os mais conhecidos e usados.

4.4 Método de Euler Melhorado

Este método, relativamente novo, foi proposto por Hanna[1] e combina as melhores qualidades dos métodos de Euler e de Runge-Kutta de segunda ordem. Este método apresenta apenas uma etapa e necessita de apenas um cálculo de derivada por etapa mas resulta em precisão de segunda ordem (assim como o método de Runge-Kutta de segunda ordem).

O método é definido pelas simples regras de recorrência a seguir:

1. Valor inicial, $z(x_0) = y(x_0)$
2. Passo de Euler, $z(x+h) = y(x) + hf([x, z(x)]$
3. Cálculo da derivada $f\,[x+h, z(x+h)]$
4. Correção trapeziodal $y(x+h) = y(x) + \frac{h}{2}\{f\,[x, z(x)] + f\,[x+h, z(x+h)]\}$

A seqüência de cálculo é apenas um passo de Euler, começando do valor melhorado $y(x)$ e usando o coeficiente angular $f\,[x, z(x)]$. Em seguida, após o único cálculo de derivada $f\,[x+h, z(x+h)]$, a quadratura trapezoidal é aplicada, onde *ambos os coeficientes angulares são usados com os valores de Euler* $z(x)$, $z(x+h)$. Os coeficientes angulares não são corrigidos após o cálculo do valor melhorado. Ao não computar os valores corrigidos dos coeficientes angulares (usando $y(x)$), consegue-se uma eficiência considerável, com apenas um cálculo de derivada por etapa, chegando-se a uma precisão $O(h^2)$.

4.4.1 Exemplo de cálculo

O desempenho do método de Euler melhorado será mostrado na solução da seguinte equação diferencial ordinária:

$$y' = xy + 1$$

com $y(0) = 0$ e $h = 0,2$.

Os resultados podem ser observados na tabela 4.1.

Tabela 4.1: Solução numérica de $y' = xy + 1$; $y(0) = 0$; $h = 0,2$

x	y_{Euler}	y_{RK2}	$y_{EulerMelh.}$	y_{exato}
0,0	0,0	0,0	0,0	0,0
0,2	0,200	0,204	0,204	0,203
0,4	0,408	0,425	0,424	0,422
0,6	0,641	0,681	0,680	0,678
0,8	0,918	0,999	0,997	0,995
1,0	1,264	1,415	1.408	1,411

Neste exemplo percebe-se claramente que a precisão do método de Euler modificado é da mesma ordem da precisão do método de Runge-Kutta de segunda ordem.

4.5 Integração de Sistemas de Equações Diferenciais Ordinárias

Todos os métodos anteriores foram deduzidos para uma equação diferencial ordinária e não um sistema. O programa desenvolvido, no entanto, trabalha com sistemas de equações diferenciais ordinárias de dimensões variáveis.

A metodologia de solução de sistemas é idêntica à metodologia de solução de apenas uma equação. O procedimento geral consiste em aplicar os mesmos passos para as diferentes equações do sistema.

4.6 Implementação dos Métodos no Programa de Computador

O programa desenvolvido resolve sistemas de equações diferenciais ordinárias empregando os diferentes métodos discutidos anteriormente. Como exemplo, considere o seguinte sistema:

$$\begin{aligned} \frac{dx}{dt} &= xy^2 + tx \\ \frac{dy}{dt} &= \log x + y^{-\frac{1}{2}}t \end{aligned} \tag{4.13}$$

Este sistema deve ser digitado em um arquivo texto para a sua solução:

```
D(x, t) = x*y^2 + t*x
D(y, t) = log(x) + y^(-1/2)*t
```

Este sistema deve ser armazenado em um arquivo com extensão `.EQU`. Os valores iniciais devem ser armazenados em um arquivo com mesmo nome base mas com extensão `.IN`.

O método de Euler está implemntado na subrotina `Integrate_EULER`. O método de Runge-Kutta de segunda ordem está implementado na subrotina `Integrate_RUNGEKUTTA2` e o método de Runge-Kutta de quarta ordem está implementado na subrotina `Integrate_RUNGEKUTTA4`. Por último o método de Euler melhorado está implementado na subrotina `Integrate_IMPREULER`.

4.7 Aplicações: Equação de *Falkner-Skan*

Nesta seção será desenvolvida uma aplicação para os métodos de solução de sistemas de equações diferenciais ordinárias.

O problema a ser resolvido é conhecido como *equação de Falkner-Skan* Esta equação resolve o problema da camada limite laminar em uma cunha. A equação de Falkner Skan é dada pela seguinte relação:

$$f''' + ff'' + \beta\left(1 - f'^2\right) = 0 \tag{4.14}$$

As condições de contorno são dadas por:

$$f(0) = 0 \tag{4.15}$$

$$f'(0) = 0 \tag{4.16}$$

$$f'(\eta \to \infty) = 1 \tag{4.17}$$

Nesta equação,

$\beta = \frac{2m}{1+m}$

$\eta = y\sqrt{\frac{m+1}{2}\frac{U(x)}{\nu x}}$

$u(x,y) = U(x)f'(\eta)$

Observe que este não é um problema de valor inicial e sim um problema de condição de contorno. Este programa não foi desenvolvido para a solução deste tipo de problema.

Outro ponto é que esta não é uma equação diferencial de primeira ordem e sim uma equação de terceira ordem. A solução é trabalhar com o seguinte sistema equivalente:

$$\begin{aligned} \frac{df}{d\eta} &= g \\ \frac{dg}{d\eta} &= h \\ \frac{dh}{d\eta} &= -fh - \beta\left(1 - g^2\right) \end{aligned} \tag{4.18}$$

Neste caso as condições de contorno são dadas por

$$f(0) =, \quad g(0) = 0, \quad g(\eta \to \infty) = 1 \tag{4.19}$$

Uma possível solução deste problema é chutar um valor para $h(0)$ e determinar $g(\eta \to \infty)$. Com um algorítimo de busca, pode-se determinar valores mais próximos de $h(0)$. Este procedimento foi feito para o seguinte sistema (`FALKNER.EQU`):

```
D(f, eta) = g
D(g, eta) = h
D(h, eta) = -f*h-beta*(1-g^2)
```

Os valores de $h(0) = f''(0)$ para diferentes valores de β podem ser observados na tabela 4.2:

Tabela 4.2: Valores de $f''(0)$ para diferentes valores de β

β	$h(0) = f''(0)$
-0,19884	0,0
-0,18	0,12864
0,0	0,4696
0,3	0,77476
1,0	1,23259
2,0	1,68722

Os diferentes perfis de velocidade obtidos para diferentes valores de β podem ser visualizados na figura 4.1.

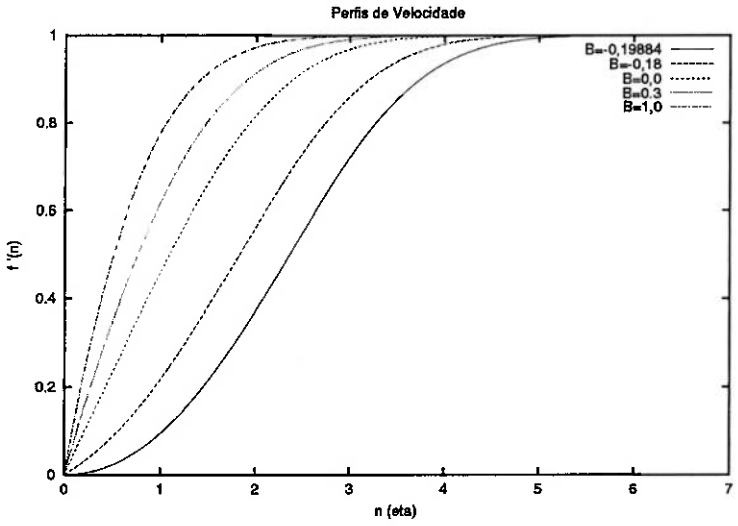


Figura 4.1: Perfis de velocidade para diferentes valores de β

Capítulo 5

Ajuste de Curvas

O ajuste de curvas consiste em aproximar pontos discretos por uma curva de formato conhecido (exceto por alguns parâmetros). Esta é uma tarefa extremamente importante na modelagem matemática de sistemas térmicos.

O ajuste de curva pode ser usado para modelar propriedades termodinâmicas de substâncias ou curvas de desempenho de equipamentos a partir de dados de fabricantes. O ajuste de curva também pode ser empregado para a obtenção de correlações a partir de dados experimentais.

O formato da curva de ajuste pode ser dado por um modelo teórico simplificado e alguns parâmetros podem ser ajustados de modo a representar o problema real. Um problema prático é a lei de *King*(Perry[32]). A lei de *King* é usada para modelar sensores de anemômetros de fio quente:

$$I^2 = \left(\frac{R-1}{R}\right)\left(X + Y\sqrt{V}\right) \tag{5.1}$$

Nesta relação, X e Y são funções das propriedades do sensor e fluido. Entretanto, este modelo é muito simplificado mas serve de base para a análise do problema. Uma possibilidade é usar a equação anterior na calibração de

anemômetros de fio quente. Neste caso,

$$I^2 = \left(\frac{R-1}{R}\right)(A + BV^n) \tag{5.2}$$

Os parâmetros A, B, n são obtidos através de uma calibração do instrumento.

Neste trabalho apenas o *método dos mínimos quadrados* será tratado. Este método distribui o erro da aproximação ao longo do intervalo de aproximação. Isto é feito minimizando o erro quadrático, daí o nome *método dos mínimos quadrados.*

5.1 Método dos Mínimos Quadrados

Como comentado anteriormente, o método dos mínimos quadrados aproxima um conjunto de pontos por uma função de formato conhecido. Esta aproximação é feita minimizando o erro global ao longo de um determinado intervalo (o domínio da função).

Considere um conjunto de pontos x_i e y_i. Deseja-se aproximar estes pontos por uma função da seguinte forma:

$$y = f(x, a_1, a_2, \ldots, a_m) \tag{5.3}$$

Neste caso, o objetivo do método dos mínimos quadrados é minimizar o erro, dado pela equação 5.4:

$$E = \frac{\sum_{i=1}^{N} [y_i - f(x_i, a_1, a_2, \ldots, a_m)]^2}{N - m} \tag{5.4}$$

Nesta equação $N - m$ é usado ao invés de N pois existem m parâmetros a serem determinados.

Um exemplo para o método dos mínimos quadrados é a regressão linear. Neste caso,

$$f(x, a_1, a_2) = a_1 + a_2 x$$

Outros exemplos de ajuste serão analisados posteriormente.

5.2 Método dos Mínimos Quadrados Linear

Neste caso, o termo *linear* se refere aos parâmetros indeterminados. Sem perda de generalidade, o método será deduzido para duas variáveis x e y. Neste caso,

$$y_i = \sum_{j=1}^{m} a_j \phi_j(x_i) \tag{5.5}$$

Onde $i = 1, \ldots, N$ sendo N o número de pontos. Neste caso, o erro quadrático vale:

$$E = \frac{\sum_{i=1}^{N} \left[y_i - \sum_{j=1}^{m} a_j \phi_j(x_i) \right]^2}{N - m} \tag{5.6}$$

Quando o erro é mínimo,

$$\frac{\partial E}{\partial a_k} = 0 \tag{5.7}$$

Substituindo a equação 5.6 na equação 5.7,

$$\sum_{i=1}^{N} \left[y_i - \sum_{j=1}^{m} a_j \phi(x_i) \right] \phi_k(x_i) = 0 \tag{5.8}$$

Ou seja,

$$\sum_{i=1}^{N} \sum_{j=1}^{m} a_j \phi_j(x_i) \phi_k(x_i) = \sum_{i=1}^{N} y_i \phi_k(x_i)$$

Invertendo as somatórias na equação acima, chega-se à seguinte relação:

$$\sum_{j=1}^{m} a_j \sum_{i=1}^{N} \phi_j(x_i) \phi_k(x_i) = \sum_{i=1}^{N} y_i \phi_k(x_i) \tag{5.9}$$

Com isso,

$$[M]\{a\} = \{f\} \tag{5.10}$$

Onde:

- $M_{ij} = \sum_{k=1}^{N} \phi_i(x_k)\phi_j(x_k)$
- $f_i = \sum_{k=1}^{N} y_k\phi_i(x_k)$

A equação 5.10 permite obter o ajuste de curvas de diversos tipos de equações.

5.2.1 Ajuste polinomial

O ajuste de curvas polinomial é extremamente usado na prática. Isto é resultado de sua simplicidade. O ajuste polinomial é dado pela equação 5.11:

$$y = a_0 + a_1x + a_2x^2 + \ldots + a_mx^m \tag{5.11}$$

Neste caso, as funções de ajuste são dadas por:

$$\phi_i(x) = x^i \qquad i = 0, \ldots, m \tag{5.12}$$

Onde m é o grau do polinômio que se deseja ajustar.

Substituindo 5.12 nas equações 5.9, para um polinômio de grau m a equação 5.10 fica:

$$\begin{bmatrix} N & \sum x & \sum x^2 & \ldots & \sum x^m \\ \sum x & \sum x^2 & \sum x^3 & \ldots & \\ \vdots & & & & \\ \sum x^m & \sum x^{m+1} & \ldots & & \sum x^{2m} \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{Bmatrix} = \begin{Bmatrix} \sum y \\ \sum xy \\ \vdots \\ \sum yx^m \end{Bmatrix} \tag{5.13}$$

Com a equação 5.13 determina-se facilemente os coeficientes $a_1, \ldots, a_m$.

5.2.2 Ajuste Potencial

O ajuste de curva potencial também é bastante usado. Este tipo de ajuste é capaz de modelar diversos fenômenos naturais. Como exemplo, pode-se citar

a equação de *Colburn* para transferência de calor em escoamento interno em tubos (Incropera[33]):

$$Nu_D = 0,023 Re_D^{4/5} Pr^{1/3}$$

onde Nu_D é o número de Nusselt (baseado no diâmetro interno), Re_D é o número de Reynolds e Pr é o número de Prandtl.

O ajuste potencial se resume em determinar os coeficientes a_1 e a_2 da equação 5.14:

$$y = a_0 x^{a_1} \tag{5.14}$$

A primeira vista, este ajuste é não linear. Este problema pode ser facilmente resolvido:

$$\ln y = \ln a_0 + a_1 \ln x$$

Desta maneira, a equação 5.10 fica:

$$\begin{bmatrix} N & \sum \ln x \\ \sum \ln x & \sum (\ln x)^2 \end{bmatrix} \begin{Bmatrix} A_0 \\ A_1 \end{Bmatrix} = \begin{Bmatrix} \sum \ln y \\ \sum \ln x \ln y \end{Bmatrix} \tag{5.15}$$

Onde $a_1 = A_1$ e $a_0 = \exp A_0$.

5.2.3 Ajuste Exponencial

Neste caso, a função de ajuste fica:

$$y = a_1 e^{a_2 x} \tag{5.16}$$

Novamente, tirando o logarítimo dos dois lados da equação 5.16, a equação 5.10 fica:

$$\begin{bmatrix} N & \sum x \\ \sum x & \sum x^2 \end{bmatrix} \begin{Bmatrix} A_0 \\ A_1 \end{Bmatrix} = \begin{Bmatrix} \sum \ln y \\ \sum x \ln y \end{Bmatrix} \tag{5.17}$$

Onde $a_1 = A_1$ e $a_0 = \exp A_0$.

5.2.4 Ajuste Logarítmico

Outra possibilidade para função de ajuste, também bastante usada na prática é o ajuste logarítmico:

$$y = a_0 \ln a_1 x \tag{5.18}$$

Esta equação também pode ser reduzida ao caso linear. Com isso, o ajuste é feito resolvendo o seguinte sistema:

$$\begin{bmatrix} N & \sum \ln x \\ \sum \ln x & \sum (\ln x)^2 \end{bmatrix} \begin{Bmatrix} A_0 \\ A_1 \end{Bmatrix} = \begin{Bmatrix} \sum y \\ \sum y \ln x \end{Bmatrix} \tag{5.19}$$

Onde $a_0 = A_1$ e $a_1 = \exp \frac{A_0}{A_1}$.

5.3 Método dos Mínimos Quadrados não Linear

Nesta seção será descrito método dos mínimos quadrados para o caso mais geral: função implícita de várias variáveis e não linear nos parâmetros de ajuste. Esta função pode ser representada pela seguinte equação:

$$g(x_1, x_2, \ldots, x_L, a_1, a_2, \ldots, a_m) = 0 \tag{5.20}$$

Na equação 5.20 L é o número de variáveis e m é o número de parâmetros independentes. Neste caso, o erro quadrático é dado por:

$$E = \frac{\sum_{i=1}^{N} \left[g(x_{1,i}, x_{2,i}, \ldots, x_{L,i}, a_1, a_2, \ldots, a_m) \right]^2}{N - m} \tag{5.21}$$

A equação 5.20 pode ser expandida em série de Taylor nos parâmetros a_j. Desprezando os termos de ordem superior, a equação 5.20 fica:

$$g(x_1, \ldots, x_L, a_1, \ldots, a_m) \approx g(x_1, \ldots, x_L, a_{1,0}, \ldots, a_{m,0}) + \sum_{j=1}^{m} \frac{\partial g}{\partial a_j} (a_j - a_{j,0}) \tag{5.22}$$

Onde $a_{i,0}$ é uma estimativa inicial para os parâmetros de ajuste.

Chamando o primeiro termo à direita da igualdade na equação 5.22 de g_0 e $\Delta a_j = a_j - a_{j,0}$ e substituindo estes valores na equação para o erro quadrático (equação 5.21) chega-se à seguinte relação:

$$E = \frac{\sum_{i=1}^{N} \left[g_0 + \sum_{j=1}^{m} \frac{\partial g}{\partial a_j} \Delta a_j \right]^2}{N - m} \tag{5.23}$$

Este erro deve ser minimizado em relação aos parâmetros a_j, com isso,

$$\frac{\partial E}{\partial a_k} = 0$$

Derivando a equação 5.23 em relação a a_k e igualando a zero, chega-se à seguinte relação:

$$\sum_{i=1}^{N} g_0 \frac{\partial g}{\partial a_k} + \sum_{i=1}^{N} \sum_{j=1}^{m} \frac{\partial g}{\partial a_k} \frac{\partial g}{\partial a_j} \Delta a_j = 0 \tag{5.24}$$

Passando o primeiro termo da equação acima para o outro lado da igualdade e invertendo as somatórias do segundo termo, chega-se à seguinte equação:

$$\sum_{j=1}^{m} \Delta a_j \sum_{i=1}^{N} \frac{\partial g}{\partial a_j} \frac{\partial g}{\partial a_k} = \sum_{i=1}^{N} g_0 \frac{\partial g}{\partial a_k} \tag{5.25}$$

Esta equação pode ser reescrita da seguinte maneira:

$$\begin{bmatrix} \sum \frac{\partial g}{\partial a_1}\frac{\partial g}{\partial a_1} & \sum \frac{\partial g}{\partial a_1}\frac{\partial g}{\partial a_2} & \cdots & \sum \frac{\partial g}{\partial a_1}\frac{\partial g}{\partial a_m} \\ \sum \frac{\partial g}{\partial a_2}\frac{\partial g}{\partial a_1} & \sum \frac{\partial g}{\partial a_2}\frac{\partial g}{\partial a_2} & \cdots & \sum \frac{\partial g}{\partial a_2}\frac{\partial g}{\partial a_m} \\ \vdots & & & \\ \sum \frac{\partial g}{\partial a_m}\frac{\partial g}{\partial a_1} & \sum \frac{\partial g}{\partial a_m}\frac{\partial g}{\partial a_2} & \cdots & \sum \frac{\partial g}{\partial a_m}\frac{\partial g}{\partial a_m} \end{bmatrix} \begin{Bmatrix} \Delta a_1 \\ \Delta a_2 \\ \vdots \\ \Delta a_m \end{Bmatrix} = - \begin{Bmatrix} \sum g \frac{\partial g}{\partial a_1} \\ \sum g \frac{\partial g}{\partial a_2} \\ \vdots \\ \sum g \frac{\partial g}{\partial a_m} \end{Bmatrix} \tag{5.26}$$

Observe que todas as derivadas na equação acima são calculadas no ponto $x_i, \ldots, x_L$ e $a_{1,0}, \ldots, a_{m,0}$. Resolvendo-se o sistema de equações lineares 5.26, pode-se chegar a uma estimativa melhor dos coeficientes a_j a partir de uma estimativa inicial $a_{j,0}$.

Critério de Convergência Como pôde ser observado acima, o método dos mínimos quadrados não linear é um processo iterativo e que deve convergir ao resultado correto após um certo número de interacões. A convergência deve ser determinada de alguma maneira. No programa desenvolvido, o critério aplicado é verificar se

$$max\{\Delta a_j\} < \epsilon \qquad j = 1, \ldots, m \tag{5.27}$$

Onde ϵ é o resíduo admissível.

5.4 Exemplos

O programa de ajuste de curvas geral `LSTSQRM` foi aplicado a alguns casos para que seu desempenho fosse demonstrado.

5.4.1 Correlações de transporte

Este primeiro exemplo, retirado de Hanna[1] é uma correlação de transferência convectiva de massa. Os dados foram obtidos por Gilliland e Sherwood. Os pontos experimentais podem ser observados na tabela 5.1 assim como o resultado ajustado.

A equação de ajuste usada foi a seguinte:

$$Sh = aRe^b Sc^c \tag{5.28}$$

O programa LSTSQRM retornou os segiuntes valores para os parâmetros a, b e c:

$$\begin{aligned} a &= 3,38 \cdot 10^{-2} \\ b &= 0,789 \\ c &= 0,436 \end{aligned} \tag{5.29}$$

Tabela 5.1: Dados de Transferência de Massa de Gilliland e Sherwood

Re	Sc	$Sh_{\text{Experimental}}$	$Sh_{\text{Calculado}}$
10800	0,6	43,7	41,16821
5290	0,6	21,5	23,44214
3120	1,8	24,2	24,95189
14400	1,8	88	83,39951
6620	1,875	51,6	45,98352
8700	1,875	50,7	57,0461
4250	1,86	32,3	32,30154
8570	1,86	56	56,17541
2900	2,16	26,1	25,50178
4950	2,16	41,3	38,88499
14800	2,17	92,8	92,45909
7480	2,17	54,2	53,96616
9170	2,26	65,5	64,50857
4720	2,26	38,2	38,19867
16300	1,83	93	92,63208
13000	1,83	70,6	77,49018
7700	1,61	42,9	48,4768
2330	1,61	19,8	18,87716

Os resultados da tabela 5.1 mostram que o ajuste é muito bom. Estes resultados podem ser visualizados na figura 5.1.

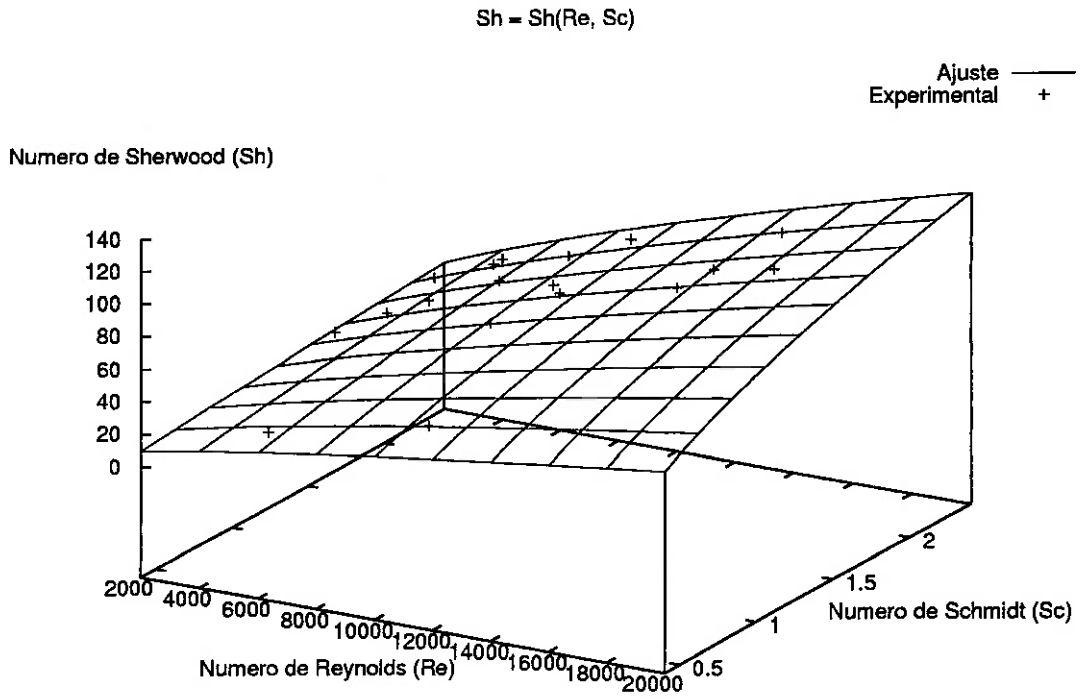


Figura 5.1: Número de Sherwood para diferentes números de Reynolds e Schmidt

5.4.2 Modelagem de uma unidade condensadora

No capítulo 7, será desenvolvido um programa de simulação de câmaras frigoríficas. A câmara possui três equipamentos que devem ser selecionados para uma dada condição de carga térmica: a unidade condensadora, a válvula de expansão termostática e o forçador de ar. Os parâmetros de desempenho destes equipamentos são essenciais na simulação da câmara, assim como no selecionamento das unidades. Nesta seção será descrita a modelagem da unidade condensadora e a seção seguinte será descrita a modelagem da válvula de expansão termostática.

As unidades condensadoras modeladas eram da Bitzer, operando com R134a. O catálog do fabricante([26]) fornece a capacidade das diferentes unidades para diferentes temperaturas de evaporação e temperaturas ambiente. Usando o programa `LSTSQRM` (programa de ajuste de curva não linear, multivariável), foi feito o seguinte ajuste de curva:

$$\begin{aligned} Q_{\text{Unidade Condensadora}}(T_e, T_c) = a_0 + a_1 \cdot T_e \\ + a_2 \cdot T_e^2 + a_4 \cdot T_e^3 \\ + a_4 \cdot T_c + a_5 \cdot T_c^2 + a_6 \cdot T_e \cdot T_c \end{aligned} \quad (5.30)$$

Observe que nestas curvas a temperatura de condensação foi fixada como sendo 10^oC superior a temperatura ambiente. Apesar de não estar indicado na equação 5.30 e figura 5.2, as variáveis independenetes são na realidade a temperatura de evaporação e a temperatura ambiente.

Este ajuste foi feito para todas as unidades da Bitzer operando com R134a. O resultado para duas unidades pode ser observado na figura 5.2

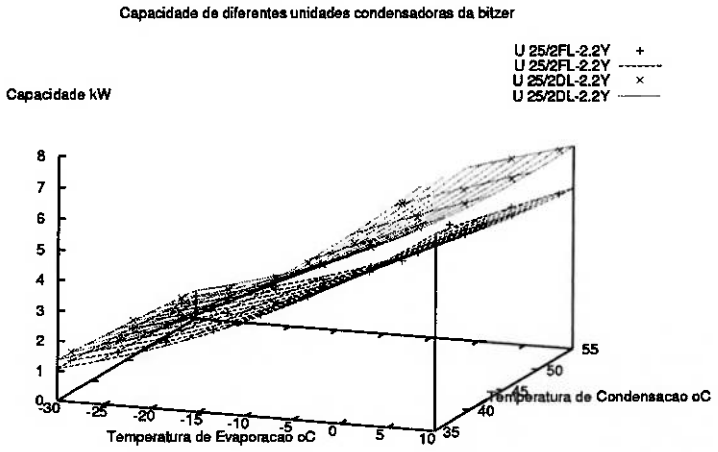


Figura 5.2: Capacidade das unidades condensadoras Bitzer para R134a para diferentes temperaturas de evaporação e de condensação. Pontos - Catálogo, Linhas - Ajuste

5.4.3 Modelagem de uma válvula de expansão termostática

Na simulação, apenas um fabricante foi usado, a Danfoss. Neste caso, será usada a mesma curva que a usada para a unidade condensadora, isto é,

$$\begin{aligned} Q_{\text{Válvula de Expansão}}(T_e, T_c) = a_0 &+ a_1 \cdot T_e \\ &+ a_2 \cdot T_e^2 + a_4 \cdot T_e^3 \\ &+ a_4 \cdot T_c + a_5 \cdot T_c^2 + a_6 \cdot T_e \cdot T_c \end{aligned} \quad (5.31)$$

Esta curva representa a capacidade para a válvula totalmente aberta. A figura 5.3 permite visualizar o ajuste obtido para uma unidade da Danfoss. Conforme a figura, pode-se concluir que o ajuste é apropriado.

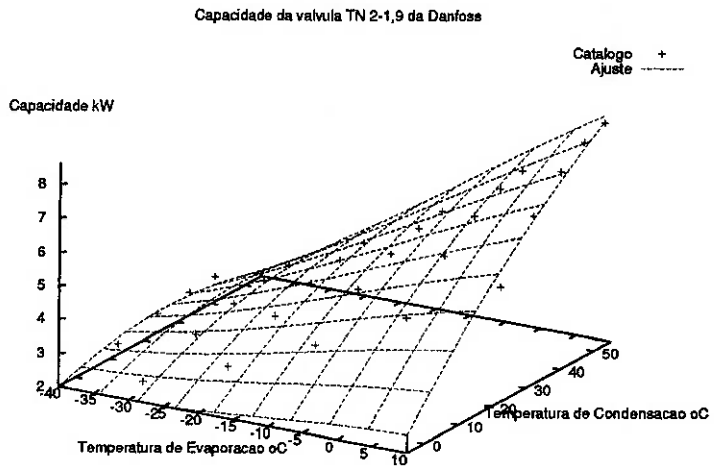


Figura 5.3: Capacidade da válvula de expansão TN2-1,9 da Danfoss para R134a

Capítulo 6

Modelagem de Propriedades Termodinâmicas e de Transporte

O capítulo 5 mostrou uma técnica para se obter aproximações para um conjunto de dados. A existência de correlações para propriedades termodinâmicas e propriedades de transporte é um aspecto extremamente importante na simulação de sistemas térmicos. Sem estas correlações fica muito difícil fazer qualquer tipo de simulação já que a cada iteração seria necessário procurar as propriedades em alguma tabela ou ábaco e entrar com estes dados na equação.

6.1 Técnicas de Modelagem

6.1.1 Interpolação

As propriedades termodinâmicas e de transporte são, em muitos casos, obtidas experimentalmente. Nestes casos, existem diversas abordagens para a obtenção de uma equação para estas propriedades. Uma primeira possibili-

dade é interpolar os dados experimentais. A interpolação deve ser feita com cuidado, de modo a representar os dados apropriadamente.

São conhecidas diversas técnicas de interpolação. As mais comuns são:

- Interpolação linear
- Interpolação de Lagrange
- Spline cúbico
- Interpolação de Newton

Estes métodos estão descritos em vários livros de cálculo numérico ([1], [3], [2], etc).

6.1.2 Método dos mínimos quadrados

O método dos mínimos quadrados, descrito no capítulo 5, é muito usado na modelagem de propriedades de substâncias e também na modelagem das curvas de desempenho de equipamentos. Neste tipo de modelagem, uma equação com parâmetros desconhecidos é ajustada pelo método de mínimos quadrados, determinando assim os valores dos parâmetros.

O formato da equação pode ser obtido de diferentes maneiras. Uma possibilidade muito usada é fazer um ajuste de curvas polinomial para representar faixas do domínio total dos dados. Isto é basicamente uma interpolação aproximada para parte dos dados disponíveis. Esta técnica requer muito cuidado pois polinômios, apesar de excelentes para manipulação matemática, não têm nenhuma relação com o fenômeno em si. Este problema se torna sério quando se trabalha com duas ou mais variáveis independentes.

Os dados experimentais muitas vezes revelam o comportamento da propriedade em estudo. Conhecendo este comportamento, pode-se propor equações

representativas do fenômeno e desta maneira, usando o método de mínimos quadrados, pode-se determinar os coeficientes desconhecidos da equação. Esta abordagem foi adotada no exemplo da seção 5.4.1(no capítulo que descreve o método dos mínimos quadrados). Neste exemplo, uma correlação para o número de Sherwood foi obtida ajustando parâmetros desconhecidos com dados experimentais.

Uma outra possibilidade, é formular um modelo simplificado para o comportamento da substância, e utilizar dados experimentais para ajustar alguns parâmetros, de modo que a correlação final esteja de acordo com os dados experimentais. Esta é a abordagem mais recomendada pois procura modelar a propriedade a partir de um comportamento aproximado da substância. Este tipo de modelo é conhecido como *modelo semi-empírico*. Neste capítulo alguns exemplos onde esta abordagem é usada serão descritos (seção 6.2.4 por exemplo).

6.2 Equações de Estado

Equações de estado são equações que correlacionam diferentes propriedades de uma substância. Para uma substância pura simples, duas propriedades bastam para o cálculo de qualquer outra. Inúmeras equações de estado foram propostas para diferentes regiões de validade e precisão. A seguir estão apresentadas algumas. Observe que qualquer equação de estado deve satisfazer o critério de estabilidade termodinâmica no ponto crítico:

$$\begin{aligned} \left(\frac{\partial P}{\partial v}\right)_{T_c} &= 0 \\ \left(\frac{\partial^2 P}{\partial v^2}\right)_{T_c} &= 0 \end{aligned} \tag{6.1}$$

6.2.1 Modelo de gás perfeito

Este modelo para a equação de estado é extremamente popular por sua simplicidade. Este modelo é válido para gases a altas temperaturas e baixas pressões. A equação de estado de gases perfeitos é dado pela seguinte relação:

$$pv = RT \tag{6.2}$$

Nesta equação, p é a pressão absoluta do gás, v é o volume específico, T é a temperatura absoluta do gás e R é constante do gás. Esta constante varia de gás para gás e vale $\frac{\Re}{M}$ onde $\Re$ é a constante universal dos gases e vale $8314 J/Kg \cdot mol \cdot K$ e M é a massa molar do gás.

6.2.2 Equação de Van der Waals

Esta equação sugerida por Van der Waals é uma das mais conhecidas,

$$\left(P + \frac{a}{v^2}\right)(v - b) = RT \tag{6.3}$$

Impondo as condições 6.1 na equação 6.3, obtém-se

- $a = \frac{27}{64}\frac{R^2 T_c^2}{P_c}$
- $b = \frac{RT_c}{8P_c}$

6.2.3 A equação de estado de Redlich-Kwong

Esta é, provavelmente, a equação de dois parâmetros mais bem sucedida. A equação tem a seguinte forma:

$$P = \frac{RT}{v - b} - \frac{a}{T^{\frac{1}{2}} v(v + b)} \tag{6.4}$$

Não existe uma tabulação extensiva de a e b, mas aplicando as condições 6.1, pode-se mostrar que

- $a = \frac{\Omega_a R^2 T_c^{2,5}}{P_c}$
- $b = \frac{\Omega_b R T_c}{P_c}$

Nestas equações, $\Omega_a = [9(2^{1/3} - 1)]^{-1}$ e $\Omega_b = \frac{2^{1/3}-1}{3}$

6.2.4 Compressibilidade de fluidos

A compressibilidade mede o quanto o fluido foge do comportamento de gás perfeito. A compressibilidade Z de um fluido é definida pela seguinte equação:

$$Z = \frac{pv}{RT} \tag{6.5}$$

Existem diversas relações para Z. Um fato interessante é que a compressibilidade em função da temperatura reduzida e pressão reduzida praticamente independe do gás. A temperatura e a pressão reduzida são definidas por:

$$T_r = \frac{T}{T_{cr}} \tag{6.6}$$

$$P_r = \frac{P}{P_{cr}} \tag{6.7}$$

Onde T_{cr} é a temperatura crítica do fluido e P_{cr} é a pressão crítica do fluido.

São conhecidas várias correlações para a compressibilidade de fluidos Z. Uma correlação é descrita a seguir.

Compressibilidade de gases puros segundo a API

Uma relação para Z muito usada é dada pela API (*American Petroleum Institute*)[24]:

$$\begin{aligned} Z = 1 + p_r \cdot T_r^{-1} \left[(0,1445 + 0,073\omega) - (0,330 - 0,46\omega) T_r^{-1} \right. \\ - (0,1385 + 0,50\omega) T_r^{-2} - (0,0121 + 0,097\omega) T_r^{-3} \\ \left. -0,0073\omega T_r^{-8}) \right] \end{aligned} \tag{6.8}$$

Nesta equação,

p_r é a pressão reduzida (p/P_c)

p é a pressão absoluta

P_c é a pressão crítica

T_r é a temperatura reduzida (T/T_c)

T é a temperatura absoluta

T_c é a temperatura crítica

ω é o fator acêntrico

Valores para pressão crítica, temperatura crítica e fator acêntrico podem ser encontrados na referência [24].

6.3 Relações para Fluidos Saturados

Uma expressão útil na determinação de propriedades de líquido e e vapor saturado é a equação de Clapeyron. Esta equação (veja a dedução apresentada por Stoecker[14]) tem a seguinte forma:

$$\left.\frac{dp}{dT}\right|_{sat} = \frac{h_{lg}}{T(v_g - v_l)} \tag{6.9}$$

Fazendo duas aproximações,

- $v_g >> v_l$
- O vapor saturado se comporta como gás perfeito, ou seja, $pv_g = RT$

chega-se à seguinte equação aproximada para fluidos na condição de saturação:

$$\left.\frac{dp}{p}\right|_{sat} = \left.\frac{dT}{T^2}\right|_{sat} \frac{h_{lg}}{R} \tag{6.10}$$

Integrando esta equação, a relação para a pressão em função da temperatura na saturação fica:

$$\ln p = A + \frac{B}{T} \tag{6.11}$$

Stocker[14] descreve um refinamento desta equação conhecida como equação de Antoine:

$$\ln p = A + \frac{B}{T - C} \tag{6.12}$$

6.4 Obtenção de Outras Propriedades a Partir da Equação de Estado

Dado um fluido puro e simples qualquer, a entropia pode ser calculada a partir da integração da seguinte equação

$$ds = c_p \frac{dT}{T} - \left(\frac{\partial v}{\partial T}\right)_p dp \tag{6.13}$$

Observe na equação 6.13 a necessidade do valor de c_p. Reid[23] apresenta várias correlações para esta propriedade.

De maneira análoga, pode-se determinar a entalpia de um fluido:

$$dh = c_p dT + \left[v - T\left(\frac{\partial v}{\partial T}\right) + p\right] dp \tag{6.14}$$

Novamente, é necessário c_p.

6.5 Correlações Propostas por Downing para Refrigerantes

Downing[20] propôs um conjunto de correlações para diferentes propriedades de diferentes refrigerantes.

6.5.1 Pressão de vapor

Downing[20] apresenta uma correlação para a pressão de saturação de refrigerantes que tem alguma similaridade com a equação de Clapeyron simplificada. Esta correlação também é recomendada pela DuPont[22].

$$\log_{10} P_{sat} = A + \frac{B}{T} + C \log_{10} T + DT + E \left(\frac{F - T}{T} \right) \log_{10}(F - T) \quad (6.15)$$

Nesta equação, A, B, C, D, E, F são coeficientes que dependem do refrigerante e do sistema de unidades utilizadas para as propriedades termodinâmicas. Para o $R134a$, pressão em kPa e temperatura em K os coeficientes têm os seguintes valores (DuPont[22]):

$$\begin{aligned} A &= 4,069889 \cdot 10^{1} \\ B &= -2,362540 \cdot 10^{3} \\ C &= -1,306883 \cdot 10^{1} \\ D &= 7,616005 \cdot 10^{-3} \\ E &= 2,342564 \cdot 10^{-1} \\ F &= 3,761111 \cdot 10^{2} \end{aligned} \quad (6.16)$$

6.5.2 Massa específica de refrigerantes líquidos

Downing[20] apresenta uma correlação para a massa específica de alguns refrigerantes no estado líquido. Os coeficientes também dependem do tipo de refrigerante e podem ser encontrados na referência [20].

$$d_L = A_L + B_L\left(1-\frac{T}{T_c}\right)^{\frac{1}{3}} + C_L\left(1-\frac{T}{T_c}\right)^{\frac{2}{3}} + D_L\left(1-\frac{T}{T_c}\right) + E_L\left(1-\frac{T}{T_c}\right)^{\frac{4}{3}} + F_L\left(1-\frac{T}{T_c}\right)^{\frac{1}{2}} + G_L\left(1-\frac{T}{T_c}\right)^{2} \quad (6.17)$$

6.5.3 Equação de estado para o vapor

No artigo de Downing ([20]), também é apresentada uma equação de estado para o vapor. A equação tem a seguinte forma:

$$P = \frac{RT}{v-b} + \frac{A_2 + B_2T + C_2e^{-KT/T_c}}{(v-b)^2} + \frac{A_3 + B_3T + C_3e^{-KT/T_c}}{(v-b)^3} + \frac{A_4 + B_4T + C_4e^{-KT/T_c}}{(v-b)^4} + \frac{A_5 + B_5T + C_5e^{-KT/T_c}}{(v-b)^5} + \frac{A_6 + B_6T + C_6e^{-KT/T_c}}{e^{av}\left(1 + C'e^{av}\right)} \quad (6.18)$$

Nesta equação, P é a pressão, T é a temperatura e v é o volume específico. Os outros coeficientes dependem do refrigerante e podem ser encontrado no artigo de Downing em unidades inglesas para diferentes refrigerantes ([20]).

6.5.4 Calor específico a volume constante do vapor

Downing também apresenta uma correlação para o calor específico a volume constante do vapor para diferentes refrigerantes (equação 6.19).

$$C_v = a + bT = cT^2 + dT^3 + \frac{f}{T^2} - \frac{JK^2Te^{-KT/Tc}}{T_c^2}\left[\frac{C_2}{v-b} + \frac{C_3}{2(v-b)^2} + \frac{C_4}{3(v-b)^3} + \frac{C_5}{4(v-b)^4} + \frac{C_6}{ae^{av}} - \frac{C_6C'}{a}(\ln 10)\log_{10}\left(1 + \frac{1}{C'e^{av}}\right)\right] \quad (6.19)$$

Novamente, T é a temperatura, v é o volume específico e C_v é o calor específico a volume constante. Os outros coeficientes podem ser encontrados no artigo de Downing ([20]) para diferentes refrigerantes.

6.5.5 Outras propriedades

As equações acima podem ser combinadas usando relações termodinâmicas exatas (vide, por exemplo, capítulo 6 de Holman[30] ou capítulo 13 de Stoecker[14]) de modo a obter-se correlações para outras propriedades. Algumas destas correlações são descritas a seguir. As diferentes constantes podem ser encontradas no artigo de Downing.

Entalpia de vaporização (h_{lg})

$$h_{lg} = JT(v_g - v_f)\left\{P\ln 10\left[-\frac{B}{T^2} + \frac{C}{T\ln 10} + D - E\left(\frac{\log_{10} e}{T} + \frac{F\log_{10}(F-T)}{T^2}\right)\right]\right\} \quad (6.20)$$

Entalpia do vapor

$$\begin{aligned} h = {} & aT + \frac{bT^2}{2} + \frac{cT^3}{3} + \frac{dT^4}{4} - \frac{f}{T} + JPv + J\left\{\frac{A_2}{v-b} + \frac{A_3}{2(v-b)^2}\right. \\ & \left. + \frac{A_4}{3(v-b)^3} + \frac{A_5}{4(v-b)^4} + \frac{A_6}{a}\left[\frac{1}{e^{av}} - C'(\ln 10)\log_{10}\left(1 + \frac{1}{C'e^{av}}\right)\right]\right\} \\ & + 1Je^{-KT/T_c}\left(1 + \frac{KT}{T_c}\right)\left[\frac{C_2}{v-b} + \frac{C_3}{2(v-b)^2} + \frac{C_4}{3(v-b)^3}\right. \\ & \left. + \frac{C_5}{4(v-b)^4} + \frac{C_6}{ae^{av}} - \frac{C_6C'(\ln 10)}{a}\log_{10}\left(1 + \frac{1}{C'e^{av}}\right)\right] + X \end{aligned} \quad (6.21)$$

6.5.6 Entropia do vapor

$$
\begin{aligned}
s = a(\ln 10)\log T + bT + \frac{cT^2}{2} + \frac{dT^3}{3} - \frac{f}{2T^2} + JR(\ln 10)\log_{10}(v-b) \\
- J\left\{\frac{B_2}{v-b} + \frac{B_3}{2(v-b)^2} + \frac{B_4}{3(v-b)^3} + \frac{B_5}{4(v-b)^4}\right. \\
\left. + \frac{B_6}{a}\left[\frac{1}{e^{av}} - C'(\ln 10)\log_{10}\left(1 + \frac{1}{C'e^{av}}\right)\right]\right\} \\
+ \frac{JKe^{-KT/T_c}}{T_c}\left[\frac{C_2}{v-b} + \frac{C_3}{2(v-b)^2} + \frac{C_4}{3(v-b)^3} + \frac{C_5}{4(v-b)^4}\right. \\
\left. + \frac{C_6}{ae^{av}} - \frac{C_6C'(\ln 10)}{a}\log_{10}\left(1 + \frac{1}{C'e^{av}}\right)\right] + Y \qquad (6.22)
\end{aligned}
$$

6.6 Correlações Aproximadas para Refrigerantes Saturados

Ribatsky[18] fornece algumas correlações aproximadas para refrigerantes. Estas correlações são bastante simplificadas mas apresentam resutlados excelentes. A Tabela 6.1 apresenta diversas correlações para diferentes propriedades termodinâmicas e de transporte em função das propriedades reduzidas. No trabalho de Ribatsky, a propriedade reduzida é definida como indicado a seguir:

$$
\text{(propriedade reduzida)} = \frac{\text{propriedade na temperatura de trabalho}}{\text{propriedade na pressão de saturação de referência}}
$$

Todas as correlações têm o seguinte formato:

$$
\phi = cte \cdot T_r^a \cdot (1 - T_r)^b \cdot p_r^c \qquad (6.23)
$$

Os expoentes a, b e c para as diferentes propriedades nos estados de vapor saturado e líquido saturado podem ser vistas na tabela 6.1.

Tabela 6.1: Coeficientes da equação 6.23 para diferentes propriedades. A *cte* depende do refrigerante

Propriedade	a_l	b_l	c_l	a_v	b_v	c_v
$C_p[J/(Kg \cdot K)]$	-0,4	-0,308	0,0166	-0,948	-0,527	0,07
$\mu[Pa \cdot s]$	-,614	0,285	-0,321	-0,013	-0,13	0,0877
$\rho[kg/m^3]$	-0,232	0,18	0,00414	-1,43	-0,236	1,05
$k[W/(m \cdot K)]$	-0,926	0,181	0,0217	1,39	-0,0375	0,0275
$h[J/Kg]$	1,0854	-0,0841	0,01219	0,347	0,0192	0,00923
$h_{lv}[J/Kg]$	0,107	0,405	-0,00412	-	-	-
$\sigma[N/m]$	0,189	1,2577	-0,0141	-	-	-

Na tabela 6.1, C_p é o calor específico a pressão constante, μ é a viscosidade dinâmica, ρ a densidade, k a condutividade térmica, h a entalpia, h_{lv} é a entalpia de vaporização, σ é a tensão superficial.

As correlações acima foram testadas para os eguintes refrigerantes:R11, R12, R123, R134a, R113, R114, R22, R124, R142b, R502, R125, R125a, R13. Os desvios médios são em geral baixos, geralmente inferiores a 2%.

6.7 Comentários Finais

A modelagem de propriedades termodinâmicas é uma tarefa extremamente importante e difícil. O objetivo deste capítulo foi introduzir algumas técnicas para a modelagem de propriedades termodinâmicas. Estas técnicas foram implementadas em subrotinas de computador (casos específicos).

No capítulo 7, as correlações de propriedades simplificadas, descritas na seção 6.6 foram usadas na simulação de uma câmara frigorífica.

As diversas referências consultadas possuem uma quantidade enorme de

correlações para diferentes propriedades termodinâmicas e de transporte. Deve-se tomar cuidado no uso destas correlações, de modo a garantir a validade das mesmas nas faixas de interesse.

Capítulo 7

Simulação de uma câmara frigorífica

Este capítulo implementa a simulação de uma câmara frigorífica em regime permanente. O objetivo deste capítulo é demonstrar o uso das diferentes ferramentas desenvolvidas deste trabalho.

7.1 Descrição e Modelagem do Sistema

O sistema consiste de diferentes componentes. A seguir estes diferentes componentes serão brevemente descritos.

O sistema de refrigeração da câmara consiste em um forçador de ar (evaporador), uma unidade condensadora (compressor e condensador) e a válvula de expansão termostática. Um modelo simplificado para o sistema é o ciclo padrão de refrigeração. Este ciclo é composto por dois trechos de troca de calor a temperatura constante (evaporador e condensador), uma compressão isoentrópica (compressor) e uma expansão isoentálpica do gás refrigerante. Este ciclo opera na região de saturação do fluido refrigerante.

7.1.1 A câmara

A câmara contém os produtos a serem resfriados ou mantidos a uma determinada temperatura. A interação entre a câmara e o sistema de refrigeração é dado pela carga térmica. A carga térmica é resultado de diversos fatores, por exemplo:

- Temperatura, umidade e pressão local
- Dimensões da câmara e tipo de isolamento
- Temperatura e umidade da câmara

Existem diferentes componentes da carga térmica:

1. **Carga de transmissão** Em geral, é uma parcela considerável da carga térmica. Consiste no calor transferido pelas paredes como resultado da condução de calor. Esta carga térmica é determinada pela diferença de temperatura (câmara e meio), isolante térmico (espessura e condutividade térmica) e dimensões da câmara (superfície de troca).

2. **Carga de Infiltração** A carga de infiltração é resultado da infiltração de ar externo na câmara. Esta carga depende do volume de ar infiltrado, temperatura do ar e da câmara (calor sensível) e umidade do ar e da câmara (calor latente)[1]. É muito difícil modelar o volume de ar infiltrado mas a Krack[15] faz algumas recomendações. A renovação de ar na câmara depende do volume da mesma.

3. **Carga do Produto** Caso a câmara necessite resfriar o produto, existe uma carga térmica associada. Esta carga térmica depende da carga diária de produto (quantidade), temperatura de entrada do produto e

[1]Calor sensível é afetivo. Calor latente é aquele que late (prof. Zerbini)

o calor específico do produto (c_p acima de 0^oC e abaixo de 0^oC e calor latente de vaporização (h_{lv}) e temperatura da câmara.

4. **Outras cargas** As outras cargas consideradas são: Ocupação humana (o ser humano transfere calor para o meio) e luminárias.

A carga térmica devido ao produto é nula em regime permanente pois nestas condições, o produto já foi resfriado e a diferença de temperatura entre a câmara (temperatura da câmara) e a temperatura do produto é nula. Na simulação, *a carga de produto é nula.* Lembre que a carga de produto é muito importante na seleção dos diferentes equipamentos da câmara. Um dos objetivos da simulação é determinar a "ociosidade" da câmara caso a carga de produto seja nula.

Na simulação, as caracterísiticas físicas da câmara são mantidas. Desta maneira, a carga térmica é dada pela seguinte expressão:

$$Q = Q_{\text{Carga Térmica}}(T_{bse}, T_{bue}, T_c, N_{\text{Horas}}) \tag{7.1}$$

Na equação 7.1 Q é a carga térmica, T_{bse} é a temperatura exterior, T_{bue} é a temperatura de bulbo úmido exterior, T_c é temperatura da câmara e N_{Horas} é o número de horas de operação diária da câmara.

7.1.2 A unidade condensadora

A unidade condensadora é composta pelo compressor, condensador e os ventiladores para circular o ar externo no condensador. O compressor pode ser caracterizado pela seguinte relação:

$$\dot{m} = f(P_E, P_{COND}) \tag{7.2}$$

Nesta equação P_E é a pressão de evaporação, P_{COND} é a pressão de condensação e $\dot{m}$ é a vazão em massa do refrigerante. Lembrando que a capacidade

do compressor vale $Q = \dot{m}\Delta h$ onde Δh é a variação de entalpia do refrigerante no evaporador, pode-se chegar à seguinte relação para esta capacidade:

$$Q = f(P_E, P_{COND}) \tag{7.3}$$

Como o fluido refrigerante está na região saturada (ou próximo a ele), pode-se determinar a temperatura de condensação e de evaporação. Estas temperaturas foram determinadas para o R134a usando a equação 6.15. As constantes, para o R134a, são dadas pela equação 6.16(DuPont[22]).

O condensador transfere calor para o ar externo. A taxa de transferência de calor pode ser aproximadamente dada pela seguinte equação:

$$Q_C = (UA)_{\text{Condensador}}(T_{\text{COND}} - T_{bse}) \tag{7.4}$$

Conhecendo a potência (P) fornecida ao compressor e fazendo um balanço de energia no sistema, chega-se à seguinte relação para o calor transferido no condensador:

$$Q_C = Q + P \tag{7.5}$$

Para uma dada temperatura de evaporação e uma dada temperatura ambiente, as equações 7.3, 7.4 e 7.5 formam um sistema de equações não linear. Este sistema pode ser reescrito de modo que

$$Q = Q_{\text{Unidade Condensadora}}(T_E, T_{bse}) \tag{7.6}$$

A Bitzer[26] fornece esta curva para suas unidades condensadoras operando com R134a em forma de tabela. Na seção 5.4.2 a equação 5.30 foi obtida a partir dos dados tabelados para as diferentes unidades condensadoras operando com R134a (referência [26]).

7.1.3 Forçador de ar

O forçador de ar é o elemento de refrigeração propriamente dito. O forçador de ar é composto por tubos aletados. No interior destes tubos, circula o refrigerante e no exterior circula, forçado por ventiladores, o ar. O forçador de ar nada mais é do que um trocador de calor.

Para um trocador de calor em que ocorre mudança de fase, a efetividade ($\epsilon = \frac{q}{q_{max}}$) vale

$$\epsilon = 1 - e^{-\frac{UA}{\dot{m}_{ar} C_{p,ar}}} \tag{7.7}$$

No caso do forçador de ar, $q_{max} = \dot{m}_{ar} C_{p,ar}(T_c - T_E)$. Com isso, a capacidade do forçador vale

$$Q = \epsilon q_{max} = C_{\text{Forçador}}(T_c - T_E) \tag{7.8}$$

Na equação 7.8 $C = \dot{m}_{ar} c_{p,ar} \left(1 - e^{-\frac{UA}{\dot{m}_{ar} C_{p,ar}}}\right)$.

Os valores de $C_{\text{Forçador}}$ para unidades da fabricante Mipal estão tabelados para diferentes temperaturas da câmra. Este parâmetro foi ajustado por uma reta neste trabalho.

7.1.4 A válvula de expansão termostática

A expansão do refrigerante é feita através de uma perda de carga localizada (no caso um orifício). Para que a maior parte da troca de calor ocorra na região de mudança de fase, para cargas térmicas diferentes, é necessário que haja algum mecanismo de controle. A válvula de expansão termostática faz este controle regulando a vazão de refrigerante no evaporador. Este controle é feito abrindo ou fechando a válvula de modo a manter um superaquecimento constante na saída do evaporador. Este mecanismo é descrito em maiores detalhes por Stoecker[13] ou Jabardo[12].

A vazão de refrigerante na válvula totalmente aberta é dada por

$$\dot{m} = C_D\sqrt{\Delta P} \tag{7.9}$$

Nesta equação, $\dot{m}$ é a vazão em massa do refrigerante, C_D é a constante da válvula e ΔP é a perda de pressão na válvula. Mas $\Delta P = P_{COND} - P_E$, ou seja, a vazão depende da temperatura de evaporação e de condensação. A capacidade da válvula totalmente aberta é dada por

$$Q = \dot{m}\Delta h \tag{7.10}$$

Δh é a variação de entalpia do refrigerante no evaporador (efeito de refrigeração).

A modelagem da válvula de expansão também foi feita partir de dados de fabricante (Danfoss[25]). O catálogo da Danfoss fornece a capacidade da válvula de expansão termostática totalmente aberta para diferentes temperaturas de evaporação e perda de carga. Com a equação 6.15, foi determinada a capacidade das diferentes válvulas em função da temperatura de evaporação e temperatura de condensação. Na seção 5.31 foi feito um ajuste de curva (equação 5.31) para cada unidade da Danfoss. Como resultado, o modelo da válvula de expansão usado neste trabalho é

$$Q = Q_{\text{Válvula}}(T_E, T_{COND}) \tag{7.11}$$

Caso a válvula seja fechada um pouco, a equação 7.11 fica:

$$Q = f \cdot Q_{\text{Válvula}}(T_E, T_{COND}) \tag{7.12}$$

O prâmetro f indica o fechamento da válvula devido a cargas térmicas parciais.

7.2 Simulação do Sistema

Com o equacionamento feito acima, existem duas possibilidades para a simulação da câmara frigorífica:

1. Desativar o termostato, manter as máquinas em operação e determinar qual o ponto de operação do sistema.

2. Manter fixa a temperatura da câmara e determinar quantas horas diárias as máquinas necessitam trabalhar para manter esta temperatura

A segunda simulação tem como objetivo determinar a ociosidade do sistema.

Fixadas a temperatura ambiente e temperatura de bulbo úmido ambiente e mantendo o sistema operando 24 horas por dia, pode-se determinar a carga térmica, temperatura da câmara e temperatura de evaporação através da solução do sistema composto pelas equações 7.1, 7.6 e 7.8. Observe que neste caso, a válvula de expansão não é necessária e também não é determinada a temperatura de condensação do sistema. Caso fosse desejado determinar a temperatura de condensação e o fator de fechamento da válvula de expansão seria necessário introduzir no sistema mais duas equações (duas variáveis a mais). A primeira equação é a da válvula, equação 7.12 e a outra seria a equação 7.4 que fornece o calor trocado no condensador. Com a curva do compressor (equação 7.3), a temperatura de condensação, calor trocado no condensador e fator de fechamento da vávlula de expansão seriam facilmente determinados. O problema é que estas duas curvas (condensador e compressor) não estão disponíveis e portanto não é possível determinar a temperatura de condensação, fator de fechamento da válvula e calor transferido no condensador.

O sistema foi projetado considerando a temperatura de condensação como sendo $T_{bse}+10°C$. Esta diferença de temperatura de 10°C é um valor razoável e que pode ser usado para se obter uma estimativa do fator de fechamento da válvula.

7.2.1 Simulação com termostato desativado e sistema operando continuamente

Neste caso, $N_{\text{Horas}} = 24$. Fixando as condições ambientes locais, o sistema de equações fica:

$$\begin{aligned} Q &= Q_{\text{Carga Térmica}}(T_{bse}, T_{bue}, T_c, N_{\text{Horas}}) \\ Q &= Q_{\text{Unidade Condensadora}}(T_E, T_{bse}) \\ Q &= C_{\text{Forçador}}(T_c) \cdot (T_c - T_E) \\ Q &= f \cdot Q_{\text{Válvula}}(T_E, T_{COND}) \\ T_{COND} &= T_{bse} + 10°C \\ N_{\text{Horas}} &= 24 \end{aligned} \tag{7.13}$$

Este é um sistema com 6 equações e 6 incógnitas que pode ser resolvido pelos diferentes métodos descritos no capítulo 3.

7.2.2 Sistema com termostato

Neste caso, um termostato desativa o sistema quando a temperatura da câmara for inferior à temperatura de controle. Como resultado, o sistema não opera continuamente. O objetvo desta simulação é obter uma estimativa do tempo que o sistema opera diariamente. Nesta simulação, a temperatura da câmara é fixada, assim como as condições ambientes locais. O sistema 7.14 é um modelo aproximado para esta situação.

$$
\begin{aligned}
Q &= Q_{\text{Carga Térmica}}(T_{bse}, T_{bue}, T_c, N_{\text{Horas}}) \\
Q &= Q_{\text{Unidade Condensadora}}(T_E, T_{bse}) \\
Q &= C_{\text{Forçador}}(T_c) \cdot (T_c - T_E) \\
Q &= f \cdot Q_{\text{Válvula}}(T_E, T_{COND}) \\
T_{COND} &= T_{bse} + 10^oC \\
T_c &= T_{\text{Termostato}}
\end{aligned}
\tag{7.14}
$$

Novamente, o resultado é um sistema com 6 equações e 6 incógnitas.

7.3 Resultados

Esta seção apresenta resultados de simulações para 6 câmaras diferentes. As câmaras foram selecionadas para mesma temperatura ambiente local (35°C), temperatura de bulbo úmido local (28°C) e mesma altitude local, referente a São Paulo, 700m. Todas as câmaras têm umidade relativa de 80% (projeto). Três casos foram analisados. Cada caso é composto por duas câmaras semelhantes mas projetadas com carga térmica devido a produtos diferentes.

7.3.1 Câmaras 1 e 2

As dimensões desta câmara são 5m × 5m × 3m. O isolante térmico é poliuretano com 6" de espessura. A câmara foi projetada para operar a uma temperatura de -10^oC. As características da câmara estão dadas na tabela 7.1.

Na figura 7.1 podem ser vistas algumas simulações destas câmaras com o termostato desativado e o sistema operando continuamente. A figura 7.4(a)

Tabela 7.1: Características das câmaras 1 e 2 com e sem carga de produto

Parâmetro	$M_{\text{Produto}} = 0$	$M_{\text{Produto}} = 1000kg/dia$
Carga térmica	1,93 kW	5,75 kW
Carga do Produto	0	3,81 kW
Temp. de evap.	-18	-18
Qtde. Forçadores	1	1
Forçador	SLR 72	SLR 75
Válv. de Expansão	TN 2-1,3	TN 2-2,5
Qtde de Un. Cond.	1	1
Un. Condensadora	U 25/2EL-2.2Y	L 443/4V-6.2Y

a apresenta o número de horas diárias que a câmara teria que operar para manter a temperatura constante.

7.3.2 Câmaras 3 e 4

Esta é uma câmara maior. As dimensões desta câmara são 25m × 15m × 4m. O isolante térmico é poliuretano com 6" de espessura. Esta câmara também foi projetada para operar a uma temperatura de $-10°C$. As características da câmara estão dadas na tabela 7.2.

Na figura 7.2 podem ser vistas algumas simulações destas câmaras com o termostato desativado e o sistema operando continuamente. A figura 7.4(b) a apresenta o número de horas diárias que a câmara teria que operar para manter a temperatura constante.

Tabela 7.2: Características das câmaras 3 e 4 com e sem carga de produto

Parâmetro	$M_{\text{Produto}} = 0$	$M_{\text{Produto}} = 3000kg/dia$
Carga térmica	15,7 kW	27,1 kW
Carga do Produto	0	11,4 kW
Temp. de evap.	-18	-18
Qtde. Forçadores	2	3
Forçador	SLR 77	SLR 78
Válv. de Expansão	TEN 5-3,7	TEN 5-5,4
Qtde de Un. Cond.	1	3
Un. Condensadora	U 862/4H-15.2Y	L 543/4P-10.2Y

7.3.3 Câmaras 5 e 6

Esta é uma câmara pequena mas opera a uma temperatura mais alta (4^oC). As dimensões desta câmara são 5m × 5m × 3m. O isolante térmico é poliuretano com 6" de espessura. As características da câmara estão dadas na tabela 7.3.

Na figura 7.3 podem ser vistas algumas simulações destas câmaras com o termostato desativado e o sistema operando continuamente. A figura 7.4(c) a apresenta o número de horas diárias que a câmara teria que operar para manter a temperatura constante.

Observe nas figuras 7.3 e 7.4 que as curvas são idênticas. Isto é fácil de entender olhando a tabela 7.3: todos os componentes são idênticos, apesar das cargas térmicas serem distintas. O problema é que a capacidade dos diferentes componentes não é contínua e sim discreta e com isso, se as cargas térmicas não diferirem muito, o programa de seleção poderá selecionar os

Tabela 7.3: Características das câmaras 5 e 6 com e sem carga de produto

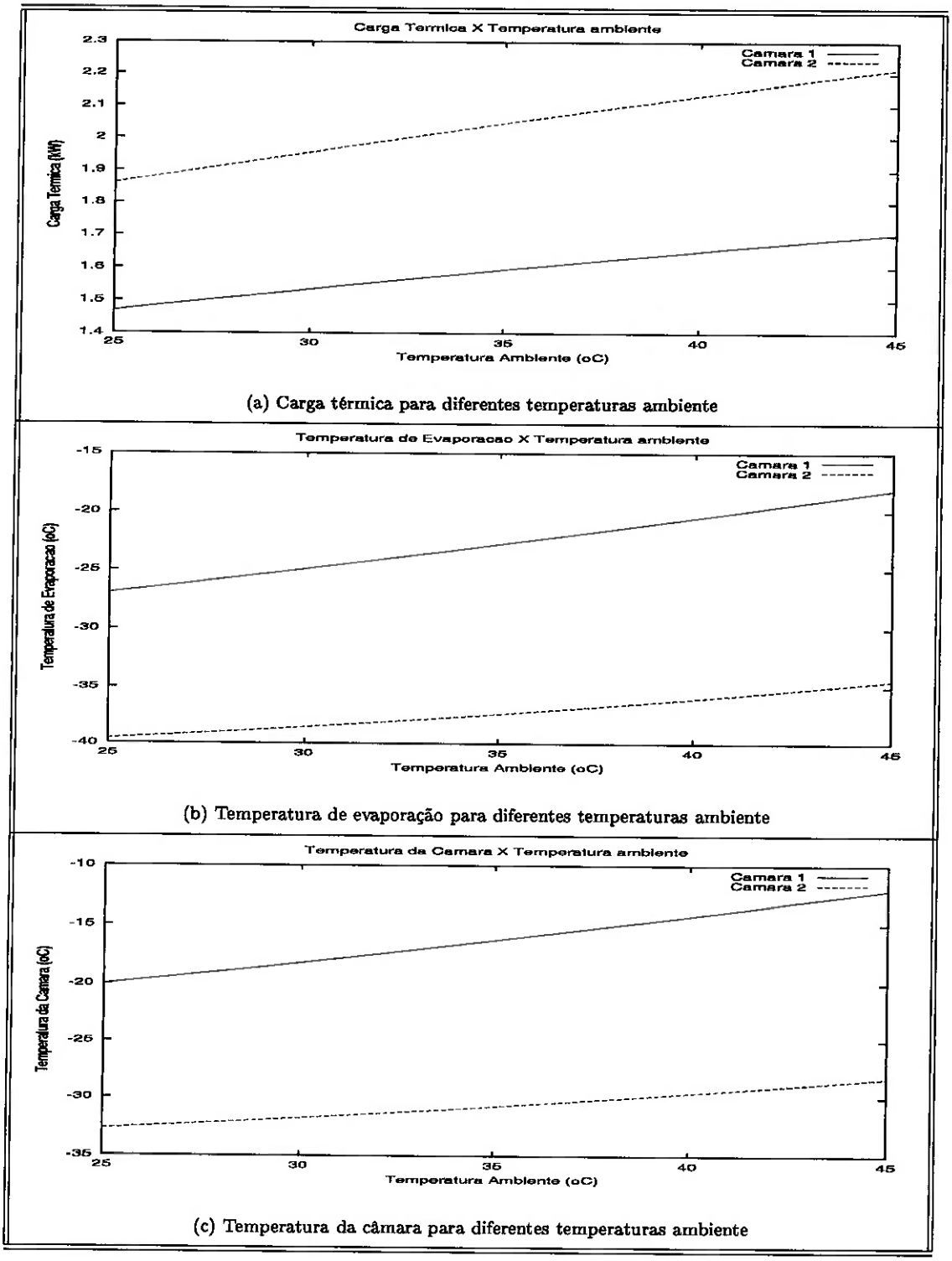
Parâmetro	$M_{\text{Produto}} = 0$	$M_{\text{Produto}} = 500kg/dia$
Carga térmica	1,46 kW	1,60 kW
Carga do Produto	0	0,14 kW
Temp. de evap.	-2	-2
Qtde. Forçadores	1	1
Forçador	SL 42	SLR 42
Válv. de Expansão	TN 2-0,5	TN 2-0,5
Qtde de Un. Cond.	1	1
Un. Condensadora	U 23/2HL-1.2Y	U 23/2HL-1.2Y

mesmos equipamentos.

7.4 Comentários

Este capítulo apresentou algumas simulações de câmaras frigoríficas. Um parâmetro muito importante no dimensionamento da câmara é a carga de produto diária. Pode-se observar nas tabelas 7.1, 7.2 e 7.3 que a carga térmica devido ao produto é considerável, principalmente quando há necessidade de congelamento. O resultado desta carga térmica são equipamentos grandes, ou seja, custo inicial e custo de operação altos. Com isso, pode-se perceber que é muito importante conhecer bem a carga de produto.

Outra observação interessante é o $\Delta T = T_c - T_e$. Este valor é um pouco diferente do valor de projeto pois os diferentes forçadores possuem capacidades discretas. Esta diferença pode representar algum problema pois a umidade do ar pode ser diferente da umidade de projeto.

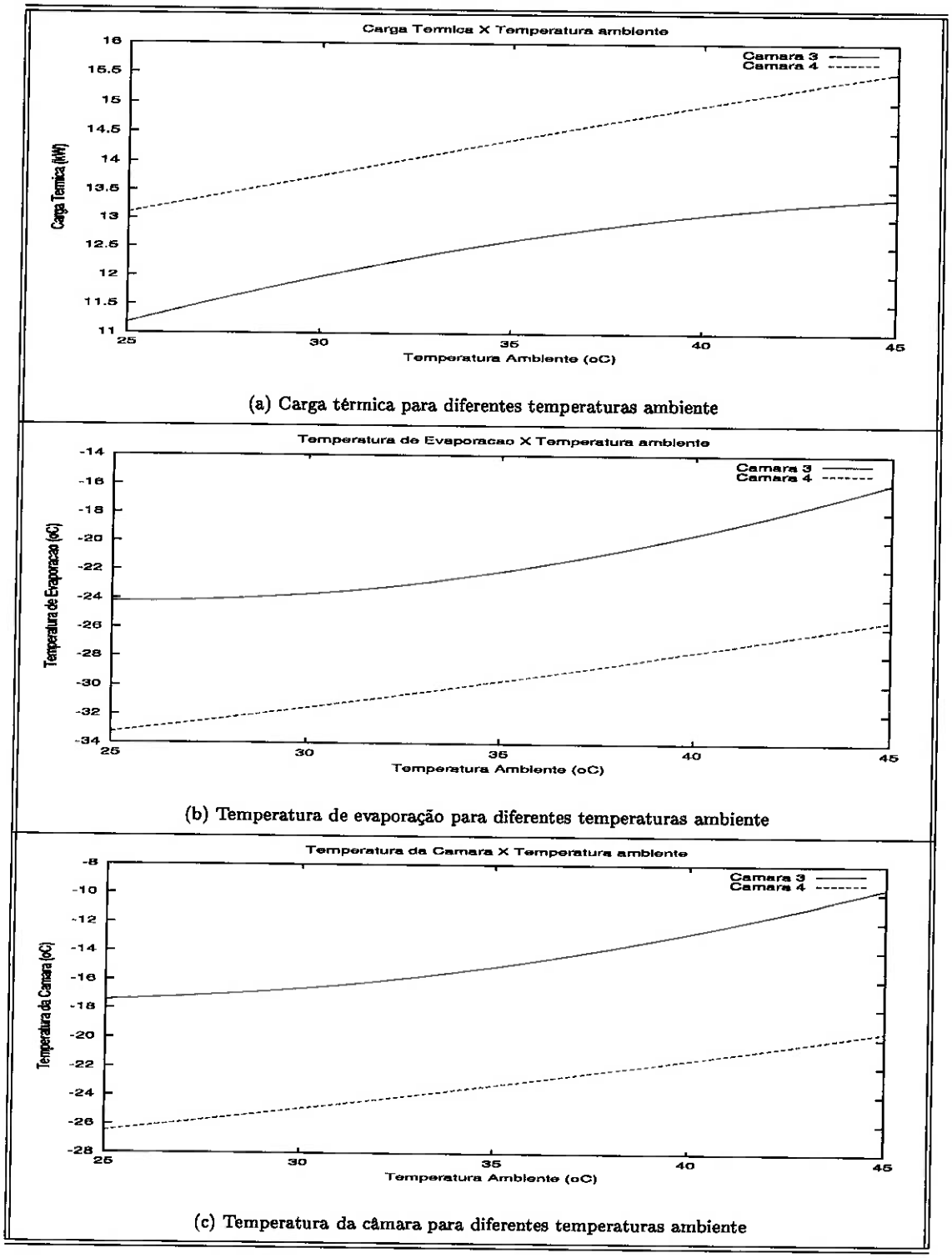


(a) Carga térmica para diferentes temperaturas ambiente

(b) Temperatura de evaporação para diferentes temperaturas ambiente

(c) Temperatura da câmara para diferentes temperaturas ambiente

Figura 7.1: Operação contínua das câmaras 1 e 2 sem termostato

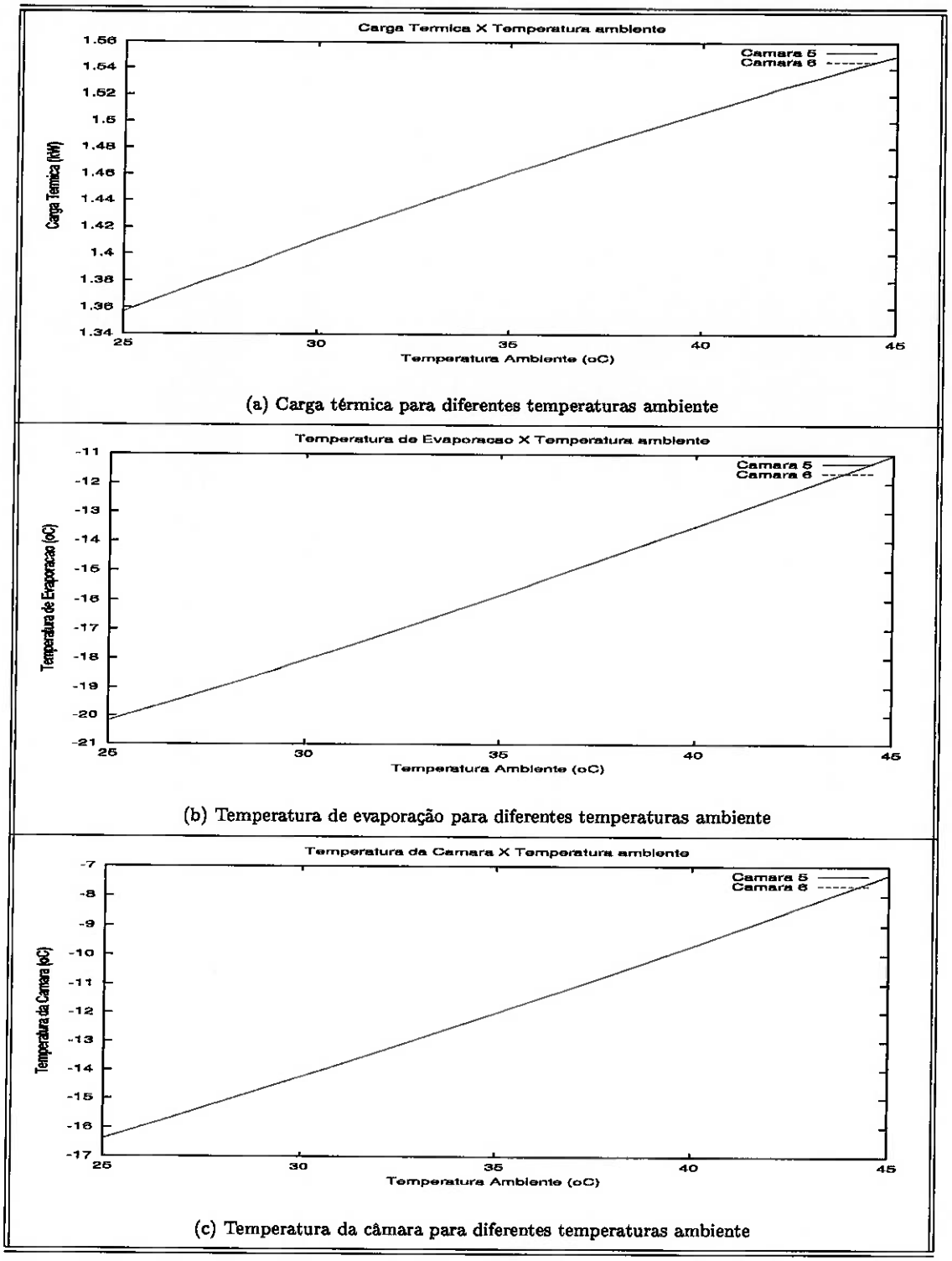


(a) Carga térmica para diferentes temperaturas ambiente

(b) Temperatura de evaporação para diferentes temperaturas ambiente

(c) Temperatura da câmara para diferentes temperaturas ambiente

Figura 7.2: Operação contínua das câmaras 3 e 4 sem termostato



(a) Carga térmica para diferentes temperaturas ambiente

(b) Temperatura de evaporação para diferentes temperaturas ambiente

(c) Temperatura da câmara para diferentes temperaturas ambiente

Figura 7.3: Operação contínua das câmaras 5 e 6 sem termostato

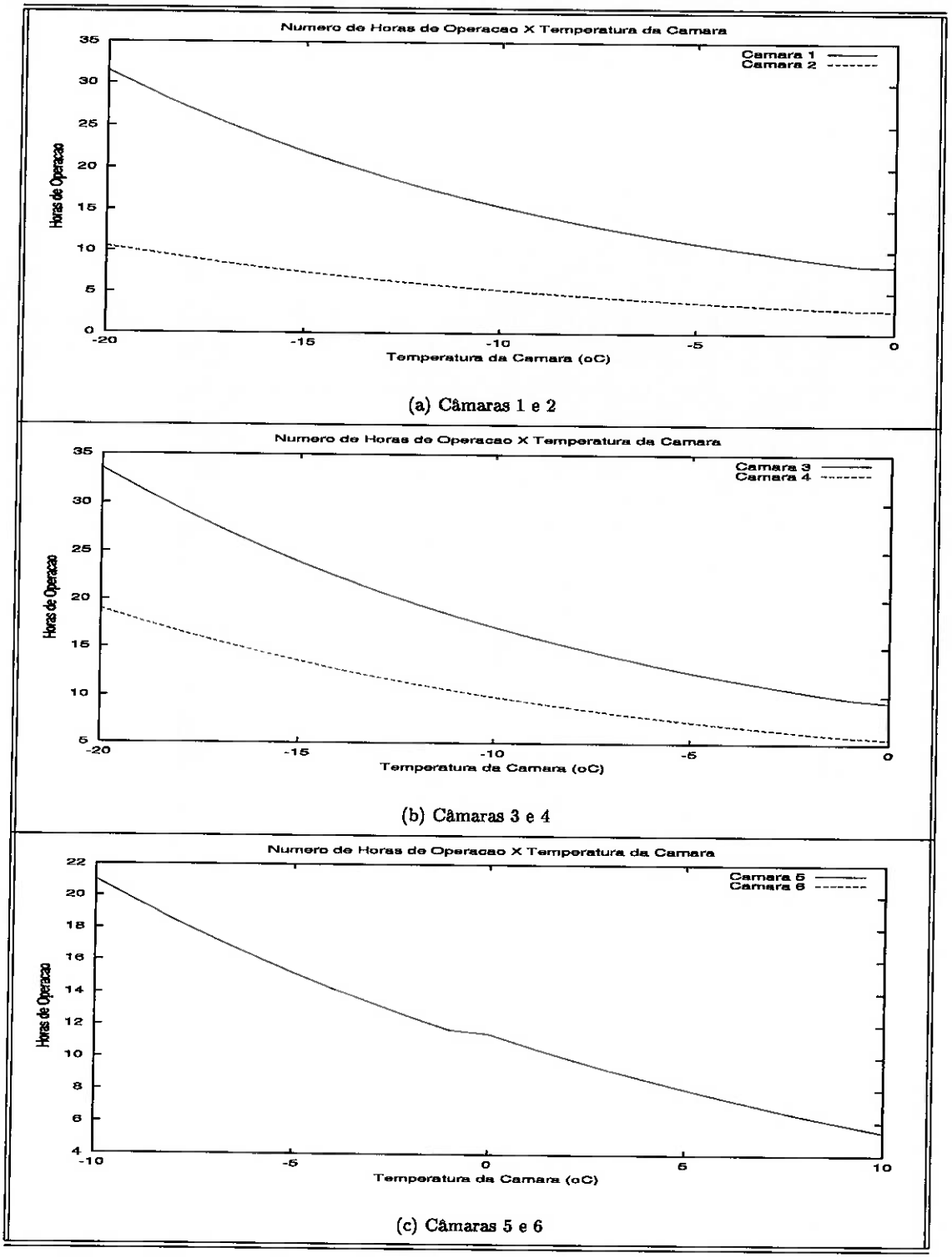


Figura 7.4: Número de horas diárias de operação para diferentes temperaturas de câmara

Capítulo 8

Conclusões e Comentários Finais

O objetivo inicial deste trabalho era a criação de um simulador de sistemas térmicos fácil de usar e útil na simulação numérica de sistemas termofluidos. Para isso, foram desenvolvidas diversas ferramentas para simulação. Foram desenvolvidas, também, ferramentas que auxiliam a modelagem de componentes e de propriedades termodinâmicas. Este trabalho apresenta uma boa revisão bibliográfica destas técnicas.

Diversas técnicas para a resolução de sistemas de equações algébricas foram analisadas, destacando pontos positivos e negativos de cada técnica. Esta análise serve como um "roteiro" na solução desta classe de problemas. Um dos trabalhos fundamentais do engenheiro é a modelagem de sistemas. Com o modelo em mãos, a simulação é apenas uma aplicação de algorítimos numéricos, mas de qualquer maneira, este processo é essencial pois é ela que fornece os resultados numéricos.

Antes da proliferação dos microcomputadores, a solução de problemas de engenharia normalmente se limitava a modelos simplificados que não necessitassem de cálculo extensos. Muitas vezes, a modelagem mesclava-se com a solução do problema.

O microcomputador mudou isto. Modelos mais complexos e refinados podem ser aplicados, mas é necessário fazer uma distinção clara entre modelo e solução numérica. O programa desenvolvido, de certa maneira incentiva esta distinção: a solução numérica do problema só pode acontecer após a modelagem apropriada do problema.

A interface com o usuário apresentou alguns problemas no momento da aplicação do programa. O interpretador, ao contrário de seu objetivo inicial, complicava a solução de qualquer problema diferente do originalmente proposto. Por exemplo, foram detectados problemas na solução da equação de Falkner-Skan (seção 4.7). Esta foi uma aplicação do programa de solução de sistemas de equações diferenciais ordinárias. Como o sistema não era um problema de valor inicial, as modificaçõe descritas na seção 4.7 são mais facilmente implementadas em um programa dedicado (caso esteja disponível um subrotina de integraçao numérica).

Outro problema encontrado na interface foi o tratamento de erros. O programa desenvolvido é muito extenso. Devido a esta extensão sempre existem alguns pequenos erros (*bugs*). Encontrar estes erros é um trabalho demorado e complicado. E pior, quando acontece algum erro na solução do problema, é difícil saber se o erro é inerente ao problema ou é algum *bug* do programa. Estes problemas certamente poderiam ser resolvidos com bastante trabalho, mas isto foge dos objetivos deste trabalho e da formação do autor. Estas correções devem ser realizadas por pessoas que trabalhem na área de computação. Além disto, existem muitas soluções alternativas que serão comentadas mais adiante.

Um trabalho interessante para o futuro seria criar um programa de cálculo de propriedades termodinâmicas geral com um interface simples e que possa ser usado em diferentes situações (programa dedicado ou biblioteca a ser

usada com alguma linguagem de programação).

Uma outra proposta de trabalho é usar os programas de computador *bison* e *flex* para a criação do interpretador. Estes programas criam um *parser* a partir de uma gramática. Estes programas estão em fase avançada de desenvolvimento, são gratuítos, e estão disponíveis para quase todas as plataformas de desenvolvimento conhecidas (DOS, WINDOWS, LINUX, e uma variedade de UNIX). Estes programas são muito fáceis de usar e não foram utilizados neste trabalho por que, no início do desenvovimento, eram desconhecidos ao autor. Assim, com poucas modificações, poderia ser desenvolvido facilmente um programa que tem como entrada os sistema de equações, como no trabalho atual, mas a saída seria o código em alguma linguagem de programação para a solução do sistema. Com poucas modificações, este programa poderia ser usado para criar código para a solução de sistemas de equações nas principais linguagens - FORTRAN, PASCAL e C. Estas propostas teriam como objetivo o desenvolvimento de um *solver* geral e robusto que pode ser usado em qualquer plataforma de desenvolvimento.

No geral, este trabalho foi muito positivo pois foram desenvolvidas diversas técnicas úteis no dia a dia de um engenheiro, o que no final das contas é o aspecto mais importante neste tipo de trabalho.

As diversas técnicas de solução de sistemas de equações algébricas não lineares foram implementadas com sucesso. Tendo em mente as vantagens e desvantagens apresentadas pelas diferentes técnicas, pode-se combiná-las de modo a se obter um *solver* robusto e rápido, permitindo a solução de variados problemas. Esta liberdade é realmente um ponto positivo do programa desenvolvido. Dada a maneira como foi escrito o programa de computador, a implementação destes métodos num programa dedicado é trabalho de poucos minutos, outro ponto positivo do trabalho.

O desempenho do interpretador, ao contrário do *solver*, não foi o esperado. Muitos problemas foram encontrados, mas a tentativa valeu, já que resultou em um conhecimento extra que dificilmente seria adquirido de outra maneira. Além disto, o autor, a muito tempo desejava desenvolver este interpretador. Vale lembrar, no entanto, que o interpretador foi apenas uma parcela do trabalho. O interpretador é apenas uma interface com o programador e existem muitas outras possibilidades (algumas já discutidas).

O uso de ferramentas numéricas adequadas certamente permite a utilização de modelos mais complexos e fiéis à realidade. A introdução destas técnicas na "cultura" do engenheiro é uma necessidade atualmente. Este objetivo foi atingido, pelo menos, para o autor. Este trabalho também permitiu (e incentivou) observar como os diferentes componentes de um sistema interagem entre sí, mostrando que na análise de um problema complexo, estas interações podem ser essenciais.

Um conclusão final deste trabalho é que o desenvolvimento de um programa de computador geral é uma atividade difícil e trabalhosa e, quando possível, o desenvolvimento de um programa simples e dedicado mas com interfaces bem especificadas pode trazer resultados excelentes com muito menos esforço.

Referências Bibliográficas

[1] Hanna, Owen T. e Sandall, Orville C.; *Computational Methods in Chemical Engineering*, Prentice Hall, 1995.

[2] Demidovich, B. P e Maron I. A.; *Computational Mathematics*, MIR, 1973.

[3] Carnahan, B., Luther, H.A., Wilkes, J.O.; *Applied Numerical Methods*, John Wiley, 1969.

[4] Faddeeva, V. N., *Computational Methods of Linear Algebra*, DOVER, 1959.

[5] Press, W., Teukolsky, S. Vetterling, W., Flannery, B.; *Numerical Recipes in Fortran, The Art of Scientific Computing*, 2nd Edition, University of Cambridge Press, Cambridge, 1992.

[6] Deutsch, David J.; *Microcomputer Programs for Chemical Engineers*, McGraw Hill, New York, 1984.

[7] Kruse, Robert; Tondo,C.L., Leung, Bruce; *Data Structures and Program Design in C*, 2^{nd} edition, Prentice Hall, 1997.

[8] Wirth, Niklaus; *Compiler Construction*, Addison-Wesley, 1996.

[9] Donnelly, Charles e Stallman, Richard; *Bison:The YACC-compatible Parser Generator v. 1.25*, Free Software Foundation, Boston, 1995.

[10] Paxson, Vern; *Flex: A fast scanner generator v2.5*, Berkeley, 1990.

[11] Smith, Norman E.; *Write Your Own Programming Language using C++*; WordWare Publishing, Plano, Texas, 1996.

[12] Stoecker, W. F. e Jabardo, J.M.S.; Refrigeração Industrial, ed. Edgar Blücher, 1994.

[13] Stoecker, W. F. e Jones J. W.; *Refrigeration and Air Conditioning*, 2nd edition, McGraw-Hill, 1982.

[14] Stoecker, W. F., *Design of Thermal Systems*, 3rd edition, McGraw-Hill, 1989.

[15] Krack Corporation, *Refrigeration Load Estimating Manual*, RLE-278, 1977.

[16] Copeland; *Copeland Refrigeration Manual*, Copeland, 1973.

[17] Giacomo, P.; *Equation for Determination of the Density of Moist Air*; International Bureau of Weights and Measures, F-92310, Sèvres, 1982.

[18] Ribatski, Gherhardt; *Estudo da Transferência de Calor em Ebulição Nucleada de Refrigerantes Halogenados*, Dissertação de Mestrado apresentada à Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 1998.

[19] Jung, D.; *Transport Properties and Surface Tension of Pure and Mixed Refrigerants*, ASHRAE Transactions, v.97, 3445, 1991.

[20] Downing, R. C.; *Refrigerant Equations*; ASHRAE Transactions, 2313.

[21] Kartsounes, G. T. e Erth, R. A.; *Computer Calculation of the Thermodynamic Properties of Refrigerants 12, 22 and 502*; Paper prepared for presentation at the ASHRAE Meeting, Washington D.C., Agosto 22-25, 1971

[22] DuPont, *Thermodynamic Properties of HFC-134a SI Units*, T-134a-SI, Technical Information, DuPont.

[23] Reid, Robert C., Prausnitz, John M., Sherwood, Thomas K.; *The Properties of Gases and Liquids*, 3rd ed., McGraw-Hill, 1977.

[24] N.V. Nederlandese Gasunie, *Physical Properties of Natural Gases*, Groningen, 1988.

[25] Danfoss; *Catálogo de Válvulas de expansão Termostática*, Refrigeração e Ar Condicionado.

[26] Bitzer, *Air-cooled Condensing Units*, Catálogo.

[27] Oetiker, T., Partl, H., Hyna, I. e Schlegl, E.; *The Not So Short Introduction to LaTeX2e*, 1999.

[28] Goossens, M., Mittelbach, F. e Samarin, A.; *The LaTeX Companion*, Addison Wesley, 1994.

[29] Lamport, Leslie; *LaTeX A Document Preparation System*, Addison-Wesley, 1995.

[30] Holman, J. P.; *Thermodynamics*, McGraw-Hill, New York, 1988.

[31] White, Frank, M.; *Viscous Fluid Flow*, McGraw-Hill, New York, 1983.

[32] Perry, A. E.; *Hot Wire Anemometry*; Claredon Press, Oxford, 1982.

[33] Incropera, Frank P. e De Witt, David P.; *Fundamentals of Heat and Mass Transfer*; 3^{rd} edition, John Wiley & Sons, New York, 1990.