UNIVERSIDADE FEDERAL DE SANTA CATARINA

DEPARTAMENTO DE ENGENHARIA DE PRODUÇÃO E SISTEMAS

PROGRAMAÇÃO LINEAR COM TROCAS MÚLTIPLAS DE BASE
IMPLEMENTAÇÃO DE UM SISTEMA COMPUTACIONAL USANDO A FPI

DISSERTAÇÃO SUBMETIDA À

UNIVERSIDADE FEDERAL DE SANTA CATARINA

PARA A OBTENÇÃO DO TÍTULO DE

MESTRE EM ENGENHARIA

SÉRGIO FERNANDO MAYERLE

FLORIANÓPOLIS

SANTA CATARINA - BRASIL

AGOSTO - 1984

PROGRAMAÇÃO LINEAR COM TROCAS MÚLTIPLAS DE BASE

IMPLEMENTAÇÃO DE UM SISTEMA COMPUTACIONAL USANDO A FPI

SÉRGIO FERNANDO MAYERLE

ESTA DISSERTAÇÃO FOI JULGADA ADEQUADA PARA A OBTENÇÃO DO TÍTULO DE

"MESTRE EM ENGENHARIA"

ESPECIALIDADE ENGENHARIA DE PRODUÇÃO, E APROVADA EM SUA FORMA FINAL PELO PROGRAMA DE PÓS-GRADUAÇÃO

PROF. ANTÔNIO DIOMÁRIO DE QUEIROZ, Dr.
COORDENADOR DO CURSO

EXAMINADORA:

PROF. WILHELM RÖDDER, Dr.
PRESIDENTE

PROF. PAULO RENÉCIO NASCIMENTO, M.Sc.

PROF. LUIZ GONZAGA DE SOUZA FONSECA, Dr.

PROF. ANTÔNIO SÉRGIO COELHO, M.Sc.

Dedico este trabalho à minha esposa e ao meu filho.

A G R A D E C I M E N T O S

Manifesto meus sinceros agradecimentos:

Ao Prof. Wilhelm Rödder, Dr., pela orientação e estímulo que me concedeu, para a realização deste trabalho.

Aos professores Paulo Renécio Nascimento, M.Sc., e Luiz Gonzaga de Souza Fonseca, Dr., pelos comentários e sugestões apresentados.

Ao Prof. Antônio Sérgio Coelho, M.Sc., pelas idéias e apoio dados na fase de testes do sistema programado.

Ao colega Carlos Ernani Fries, pelo apoio computacional prestado.

À Universidade Federal de Santa Catarina, pela oportunidade que me concedeu em realizar o curso de Pós-Graduação em Engenharia de Produção.

A todos que, direta ou indiretamente, contribuíram para a realização deste trabalho.

R E S U M O

Este trabalho tem como objetivo a implantação e avaliação de um algorítmo de programação linear, capaz de resolver problemas de médio e grande porte, usando o gradiente da função objetivo para efetuar trocas múltiplas de base.

Nele estão descritos os fundamentos matemáticos do método aplicado, bem como os instrumentos numéricos usados para tornar o sistema computacional mais eficiente.

O aspecto computacional também foi abordado, através da descrição das estruturas de armazenamento dos dados, das estruturas dos subprogramas, e da forma de como utilizar o sistema.

A título de avaliação do método proposto, será analisada a complexidade do algorítmo, fazendo um paralelo com o SIMPLEX. Ainda sob este enfoque, serão apresentados alguns resultados numéricos.

A B S T R A C T

The aim of this work is the implementation and evaluation of an LP-algorithm, to solve middle- and large-size problems using the gradient of the objective function to accomplish multiple bases exchanges.

The basic mathematics of the applied method, and the numerical instruments that make the computational system more efficient, are presented in this work.

A computational view also is presented through the description of the routines and data structures as well as the form in which the use of the system should be made.

The evaluation of the proposed method will be made through the analysis of the algorithm complexity and comparing it with SIMPLEX method. We shall give some numerical results for this purpose.

S U M Á R I O

L I S T A D E F I G U R A S

L I S T A D E Q U A D R O S

CAPÍTULO I

1 - INTRODUÇÃO

1.1 - Histórico e Origem

Na busca de algorítmos mais eficientes para a resolução de problemas de programação linear (PPL), Rödder desenvolveu um método que utiliza o gradiente da função objetivo para efetuar trocas múltiplas de base.

Assim como no método SIMPLEX, o PROJECT, como foi chamado, sofreu uma série de modificações, à medida que novos estudos foram sendo efetuados, como é visto a seguir:

- Em /RB1/ e /RB2/, Rödder e Blauth apresentam um algorítmo conceitual alternativo para a resolução de problemas de programação linear, caracterizando o início dos estudos sobre o PROJECT.

- Rödder, em /RO2/, apresenta uma forma revisada do algorítmo PROJECT, adaptada para o problema das duas fases, e em /RO1/, estuda o problema da dependência linear e degeneração, apontando medidas no sentido de manter uma base, quando efetuada a troca de fase em problemas com degeneração.

- Continuando os estudos acerca do PROJECT, Rödder demonstra, em /RO3/, que o algorítmo é finito, através do uso de um conceito

análogo ao da perturbação, também utilizado na demonstração da convergência do SIMPLEX.

- Em /COE/, Coelho efetua um estudo sobre análise de pós-otimalidade para o algorítmo PROJECT, apresentando opções de análise em relação aos vetores de custo e requerimento, além da inclusão e exclusão de variáveis e restrições no problema original.

- Rödder et alii sugerem, em /RCM/,um novo algorítmo conceitual, que utiliza a forma produto da inversa para efetuar mudanças na base, a ser desenvolvido a partir do algorítmo revisado.

Nesse contexto,esta dissertação visa dar uma colaboração ao método proposto, através da implementação computacional[1] do algorítmo sugerido em /RCM/. Pretende-se com isto caracterizar o PROJECT como um método de programação linear computacionalmente viável para a resolução de problemas de médio e grande porte.

1.2 - Considerações Gerais

É conhecido que,na pesquisa operacional, grande parte dos problemas são modelados segundo um PPL. Isto deve-se ao fato de que a programação linear é uma das técnicas mais difundidas, além de ter a sua disposição uma série de pacotes comerciais[2] que facilitam a sua resolução.

---

[1] Foi utilizado o computador IBM 4341, da Universidade Federal de Santa Catarina.

[2] Dentre os pacotes comerciais disponíveis no mercado, pode-se destacar o MPSX e o TEMPO.

Dada esta grande aplicação da programação linear na pesquisa operacional, justifica-se o desenvolvimento de novos métotos que possam resolver tais problemas com maior eficiência (menor tempo de CPU).

Além disso, tendo em vista que os pacotes disponíveis, capazes de resolver problemas de grande porte, são propriedades de empresas multinacionais, o desenvolvimento de tecnologia nacional, nesta área, poderá representar uma economia de divisas para o país.

O sistema computacional a ser desenvolvido será restrito, a nível deste trabalho, nos seguintes aspectos:

- não será implementada a análise de pós-otimalidade, embora o sistema deva ser estruturado para um posterior trabalho neste sentido;

- as dimensões dos problemas a serem resolvidos pelo sistema estarão limitadas pela disponibilidade de memória principal no computador, tendo em vista que não serão utilizadas memórias auxiliares para o armazenamento dos dados;

- não serão desenvolvidos sistemas auxiliares que possibilitem a montagem do arquivo de dados, a revelia do que ocorre com os sistemas comerciais.

CAPÍTULO II

2 - A FORMA PRODUTO DA INVERSA

2.1 - O Problema dos Métodos Tradicionais de Inversão

Todo e qualquer problema de programação linear, mesmo os de pequeno porte, envolve um número relativamente grande de operações matemáticas elementares. Este fato faz com que a utilização de computadores na resolução destes problemas seja imperativa.

Não raros são os problemas reais que possuem 1000 restrições e 3000 variáveis estruturais, ou até mais. É evidente que o armazenamento explícito de matrizes desta ordem, implica em áreas de armazenamento muito grandes, dificilmente supridas pela memória dos computadores atuais.

Considerando, entretanto, o elevado grau de esparsidade dessas matrizes, pode-se armazená-las em forma de listas através da memorização dos elementos não nulos.

Desta forma, fica resolvido o problema de armazenamento da matriz original. Porém, para a resolução do PPL, deve-se efetuar a inversão da matriz básica. A utilização de métodos convencionais[3] para este fim, fará com que o grau de esparsidade da inversa seja mui

---

[3] Métodos como o de Gauss-Jordan, por exemplo.

to baixo, inviabilizando o armazenamento desta na forma de listas.

A forma produto da inversa (FPI), que será apresentada a seguir, tem como característica marcante o fato de possuir um grau de esparsidade aproximadamente igual ao do problema original, o que viabiliza a utilização de listas para o seu armazenamento.

Em se tratando de memória auxiliar (disco ou fita magnética), para o armazenamento das matrizes citadas, a utilização das listas e da FPI também é vantajosa, pois o acesso a estas unidades é relativamente lento, e a minimização da transmissão de dados entre estas e a memória principal é um fator importante para a eficiência do sistema.

2.2 - Matrizes Elementares

Uma matriz E mxm, tal que:

$e_{ij} = 0$ para $i \neq j$ e $j \neq r$

$e_{ij} = 1$ para $i = j \neq r$

$e_{ij} \in \mathbb{R}$ para $j = r$

$e_{ij} \neq 0$ para $i = j = r$

onde $i,j \in \{1,2,\ldots,m\}$,
é uma matriz coluna elementar com pivô e_{rr}. Sua inversa η_{ij} sempre existe, e pode ser obtida através de:

$$\eta_{ij} = e_{ij} \qquad \text{para} \qquad j \neq r$$

$$\eta_{rr} = \frac{1}{e_{rr}}$$

$$\eta_{ir} = -\frac{e_{ir}}{e_{rr}} \qquad \text{para} \qquad i \neq r \qquad (2.1)$$

Assim sendo, para o armazenamento da inversa de uma matriz elementar, há necessidade somente do índice r e do vetor coluna η_{ir}, inclusive para i=r.

Com relação ao vetor η_{ir}, percebe-se que o grau de esparsidade permanece igual ao grau da coluna r da matriz elementar, o que sugere o armazenamento do mesmo na forma de listas.

2.3 - Técnica de Inversão Usando Matrizes Elementares

O conceito de matriz elementar pode ser utilizado para se obter, com economia de memória, a inversa de uma matriz não singular.

Este processo de inversão, conhecido por forma produto da inversa, será apresentado por meio de um exemplo.

Exemplo 2.1

Seja A uma matriz 3x3, não singular, para a qual deseja-se obter a inversa.

$$A = \begin{bmatrix} 1 & 3 & 0 \\ 0 & -1 & 1 \\ 2 & 0 & 5 \end{bmatrix}$$

Como $a_{11} \neq 0$, pode-se montar uma matriz elementar inversível, usando a primeira coluna de A, com a_{11} sendo o pivô.

$$E_1 = \begin{bmatrix} 1 & & \\ 0 & 1 & \\ 2 & & 1 \end{bmatrix} \qquad E_1^{-1} = \begin{bmatrix} 1 & & \\ 0 & 1 & \\ -2 & & 1 \end{bmatrix}$$

Tomando a segunda coluna de A, e pré-multiplicando-a pela inversa da primeira matriz elementar, obtém-se:

$$\alpha_2 = E_1^{-1} \cdot \begin{bmatrix} 3 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \\ -6 \end{bmatrix}$$

Como $\alpha_{22} \neq 0$, pode-se montar outra matriz elementar inversível, tomando este elemento como pivô.

$$E_2 = \begin{bmatrix} 1 & 3 & \\ & -1 & \\ & -6 & 1 \end{bmatrix} \qquad E_2^{-1} = \begin{bmatrix} 1 & 3 & \\ & -1 & \\ & -6 & 1 \end{bmatrix}$$

Pré-multiplicando a terceira coluna de A, pelas inversas das matri

zes elementares, na ordem em que foram criadas, obtêm-se:

$$\alpha_3 = E_2^{-1} \cdot E_1^{-1} \cdot \begin{bmatrix} 0 \\ 2 \\ -1 \end{bmatrix} 1 = \begin{bmatrix} 3 \\ -1 \\ -1 \end{bmatrix} 5$$

Como $\alpha_{33} \neq 0$, este poderá ser utilizado como pivô na geração da terceira matriz elementar inversível.

$$E_3 = \begin{bmatrix} 1 & & 3 \\ & 1 & -1 \\ & & -1 \end{bmatrix} \qquad E_3^{-1} = \begin{bmatrix} 1 & & 3 \\ & 1 & -1 \\ & & -1 \end{bmatrix}$$

Pode ser verificado, facilmente, que o produto destas inversas é a inversa de A, bastando para tanto efetuar a operação que se segue:

$$E_3^{-1} \cdot E_2^{-1} \cdot E_1^{-1} \cdot A = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix} = I$$

Não há necessidade de se saber previamente se a matriz a ser invertida é, ou não, singular. A singularidade da matriz poderá ser constatada, através da inexistência de um elemento não nulo para pivô. Note-se, entretanto, que a cada linha e a cada coluna, ao final da inversão da matriz, deverá ser associado um e somente um pivô.

2.4 - Operação FTRAN

Será chamada de FTRAN (*forward transformation*) a operação de pré-multiplicação de um vetor coluna, pela inversa de uma matriz, armazenada na sua forma produto.

Seja a_j^o a coluna a ser pré-multiplicada pela inversa

$$E_T^{-1} \cdot E_{T-1}^{-1} \cdots E_1^{-1}$$

que se encontra armazenada na forma:

$$r^1,\eta_1^1,\ldots,\eta_m^1;\ r^2,\eta_1^2,\ldots,\eta_m^2;\ldots;r^t,\eta_1^t,\ldots,\eta_m^t;\ldots;r^T,\eta_1^T,\ldots,\eta_m^T.$$

Então:

$$a_j^T = E_T^{-1} \cdot E_{T-1}^{-1} \cdots E_1^{-1} \cdot a_j^o$$

pode ser definida recursivamente como:

$$a_j^1 = E_1^{-1} \cdot a_j^o$$

$$a_j^t = E_t^{-1} \cdot a_j^{t-1}$$

$$a_j^T = E_T^{-1} \cdot a_j^{T-1} \qquad (2.2)$$

Nestas expressões, conhecendo-se a posição pivotal r^t, o cálculo

de a_{ij}^{t}, será efetuado como segue:

$$a_{ij}^{t} = a_{ij}^{t-1} + a_{rj}^{t-1} \cdot \eta_{i}^{t} \qquad \text{para } i \neq r = r^{t}$$

$$a_{rj}^{t} = a_{rj}^{t-1} \cdot \eta_{r}^{t} \qquad \text{para } r = r^{t} \qquad (2.3)$$

Este processo pode ser aplicado a qualquer coluna, e em especial, durante a própria montagem da forma produto da inversa.

2.5 - Operação BTRAN

Será chamada de BTRAN (*backward transformation*) a operação de pós-multiplicação de um vetor linha, pela inversa de uma matriz armazenada na sua forma produto.

Seja f_{i}^{o} o vetor linha em questão, e seja a inversa como apresentada no ítem 2.4. Então, a operação BTRAN é definida como:

$$f_{i}^{T} = f_{i}^{o} \cdot E_{T}^{-1} \cdot E_{T-1}^{-1} \cdots E_{1}^{-1}$$

A definição recursiva desta operação é:

$$f_{i}^{1} = f_{i}^{o} \cdot E_{T}^{-1}$$

$$f_{i}^{t} = f_{i}^{t-1} \cdot E_{T-t+1}^{-1}$$

$$f_i^T = f_i^{T-1} \cdot E_l^{-1} \qquad (2.4)$$

Nestas expressões, conhecendo-se a posição pivotal r^t, o cálculo de f_{ij}^t será efetuado através de:

$$f_{ij}^t = f_{ij}^{t-1} \qquad \text{para} \quad j \neq r^t$$

$$f_{ir}^t = \sum_k f_{ik}^{t-1} \cdot \eta_k^{T-t+1} \qquad \text{para} \quad r = r^t \qquad (2.5)$$

A fim de obter uma linha ℓ da inversa, a operação BTRAN é especialmente importante, pois basta submeter o ℓ-ésimo vetor unitário a esta operação, que o resultado obtido será a linha desejada.

CAPÍTULO III

3 - O MÉTODO PROJECT

3.1 - Conceituação do Método PROJECT

Várias são as tentativas encontradas na literatura, no sentido de evitar a procura da solução ótima, através das arestas do poliedro convexo de um problema de programação linear[4].

Assim como em diversas dessas tentativas, o método PROJECT combina a idéia de utilizar o gradiente da função objetivo com a de fazer trocas múltiplas de base.

O método PROJECT propõe-se a resolver o problema clássico de programação linear:

$$\max \; c^T x \quad , \quad Ax = b \quad , \quad x \geq 0, \tag{3.1}$$

sendo $c, x \in \mathbb{R}^n$; $b \in \mathbb{R}^m$; A uma matriz m x n.

"Seus principais passos são:

- gerar uma solução inicial viável (usando variáveis artifici

[4] Veja, por exemplo, as referências /PAR/,/COK/,/KUT/ e /KTZ/.

ais, se necessário);

- calcular a projeção ortogonal de c sobre o espaço nulo da matriz A;

- andar nesta direção até que uma ou mais das restrições de não negatividade sejam atingidas (tornam-se ativas);

- calcular a projeção de c, sobre o espaço nulo da matriz A aumentada pelos vetores unitários correspondentes às restrições atingidas;

- se o gradiente projetado for nulo, fazer um teste de otimalidade, e parar se a solução for ótima;

- caso não sejam satisfeitas as condições de otimalidade, relaxar as restrições de não negatividade indicadas pelo teste"[5].

3.2 - Fundamentos Matemáticos

A fim de resolver o problema 3.1, o método PROJECT utiliza os seguintes conceitos matemáticos básicos[6]:

i) A projeção ortogonal de um vetor $c \in \mathbb{R}^n$, sobre o espaço nulo

---

[5] /RCM/, op. cit., p.628.

[6] Todos os conceitos matemáticos apresentados neste ítem, foram obtidos das referências /RCM/, /RB1/ e /RB2/.

de uma matriz B pxn de rank p, é dada por:

$$C^T - C^T B^T (B B^T)^{-1} B \qquad (3.2)$$

ii) A matriz A, aumentada pelos vetores unitários correspondentes às restrições de não negatividade atingidas, referida no ítem 3.1, é da forma

$$\begin{bmatrix} A \\ e_J^T \end{bmatrix}, \quad \text{onde} \quad J = \{j_1,\ldots,j_k\} \subset \{1,\ldots,n\}$$

$$e_J = \{e_{j_1},\ldots,e_{j_k}\}$$

sendo e_{j_k}, o j_k-ésimo vétor unitário do $\mathbb{R}^n$.

Chamando de B a matriz aumentada, o cálculo da projeção de c sobre o espaço nulo de B será efetuado pela expressão 3.2, onde:

$$(B.B^T)^{-1} = \begin{bmatrix} D_J^{-1} & -D_J^{-1} A_J \\ -A_J^T D_J^{-1} & I + A_J^T D_J^{-1} A_J \end{bmatrix} \qquad (3.3)$$

sendo $D_J^{-1} = (A_J A_J^T - A_J A_J^T)^{-1}$, que será chamado de bloco central, e A_J as colunas de A com $j \in J$.

Seja $\bar{J}$ o complemento de J em {1,...,n}. Então, é possível demonstrar que:

$$D_J^{-1} = (A_{\bar{J}}^T A_{\bar{J}})^{-1} \qquad (3.4)$$

onde $A_{\bar{J}}$ são as colunas de A com $j \in \bar{J}$.

iii) Seja F uma matriz mxm e $a \in \mathbf{R}^m$. Existindo as inversas, tem-se:

$$(F \pm a\,a^T)^{-1} = F^{-1} \mp \frac{y\,y^T}{1 \pm a^T y}$$

onde $y = F^{-1} a$.

Este conceito matemático pode ser utilizado, de forma recursiva, na atualização do bloco central, após serem ativadas ou relaxadas restrições, evitando que se efetue uma reinversão. Assim, D_J^{-1} poderá ser calculado através de:

$$D_J^{-1} = D_{j_1,\ldots,j_k}^{-1} = (D_{j_1,\ldots,j_{k-1}}^{-1} - a_{j_k} a_{j_k}^T)^{-1} =$$

$$D_{j_1,\ldots,j_{k-1}}^{-1} + y_k y_k^T \frac{1}{1 - a_{j_k}^T y_k}$$

onde $y_k = D^{-1}_{j_1,\ldots,j_{k-1}} \cdot a_{j_k}$ (3.5)

iv) Seja x uma solução viável do problema 3.1, tal que $x_j = 0$ para $j \in J$, e seja nula a projeção do gradiente da função objetivo sobre o espaço nulo da matriz aumentada pelas restrições $e_j^T x = 0$. Então, aplicando as condições de Kuhn-Tucker, é necessário e suficiente, para a otimalidade de x, que[7]:

$$u_J^T = c^T (A^T e_J) \begin{bmatrix} -D_J^{-1} \cdot A_J \\ I + A_J^T D_J^{-1} A_J \end{bmatrix} \leqslant 0 \qquad (3.6)$$

onde u_J^T são as variáveis duais relativas às restrições $e_J^T x = 0$.

v) O problema 3.1 pode ser apresentado, na forma revisada, como:

$$\max \; z, \; \text{s.a} \begin{bmatrix} 1 & c^T \\ 0 & A \end{bmatrix} \begin{bmatrix} z \\ x \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}, \quad x \geqslant 0 \qquad (3.7)$$

---

[7] Relembra-se o fato de que os conceitos matemáticos em questão, foram obtidos das referências /RCM/,/RB1/ e /RB2/.

Nesta forma o gradiente da função objetivo é um vetor unitário. Redefinindo o problema 3.7, por motivos de simplificação, tem-se:

$$\max\ (1,0,\ldots,0)x\ ;\ \text{s.a.}\quad Ax = b\ ;\ x_2,\ldots,x_n \geq 0 \qquad (3.8)$$

onde:

$$A:=\begin{bmatrix} 1 & c^T \\ 0 & A \end{bmatrix}\ ;\ b:=\begin{bmatrix} 0 \\ b \end{bmatrix}\ ;\ x:=\begin{bmatrix} z \\ x \end{bmatrix}\ ;\ m:=m+1\ \text{ e }\ n:=n+1$$

Esta representação simplifica em muito os cálculos a serem efetuados.

vi) Desenvolvendo a expressão 3.2, obtem-se que a projeção do gradiente da função objetivo do problema 3.8 é dada por:

$$\left[\ 1-D_J^{-1}(1,1)\ ,\ -\ D_J^{-1}(1,\cdot)\ a_j\ \right] \qquad (3.9)$$

para $j \in \bar{J}$ e zero para os demais componentes, onde $D_J^{-1}(1,\cdot)$ é a primeira linha do bloco central, e $D_J^{-1}(1,1)$ é a primeira componente deste vetor.

vii) O cálculo de todas as variáveis duais, das restrições ativas do problema 3.1, é efetuado pela expressão abaixo:

$$(u_m^T\ ,\ u_J^T) = c_J^T\ (A_J^{-1}\ e_J)\begin{bmatrix} D_J^{-1} & -D_J^{-1}A_J \\ -A_J^T D_J^{-1} & I + A_J^T D_J^{-1} A_J \end{bmatrix}$$

onde u_m^T são as variáveis duais das restrições $Ax = b$, e u_J^T são as variáveis duais relativas às restrições $e_J^T x = 0$. Desenvolvendo esta expressão para o caso particular do problema 3.8, tem-se:

$$(u_m^T, u_J^T) = \left[D_J^{-1}(1,\cdot), -D_J^{-1}(1,\cdot)\, a_j \right] \qquad (3.10)$$

para $j \in J$.

3.3 - Melhoramentos Numéricos

3.3.1 - Estabilidade Numérica

O uso do bloco central como sugerido na expressão 3.4, para o cálculo dos vetores y, do gradiente projetado e das variáveis duais, tem a desvantagem de possuir uma baixa estabilidade numérica.

Entretanto, o cálculo do bloco central, quando o gradiente projetado fica nulo, pode ser efetivado através de:

$$D_J^{-1} = (A_{\bar{J}}^{-1})^T A_{\bar{J}}^{-1} \qquad (3.11)$$

pois neste caso particular, $|\bar{J}| = m$, o que possibilita o cálculo das inversas acima. Este procedimento possui uma estabilidade numérica maior que a forma anteriormente citada, conforme apresenta

do em /RCM/.

Além disso, as operações com o bloco central passarão a ser efetuadas de forma mais econômica, armazenando A_J na forma produto da inversa.

Exemplo 3.1

Resolver o sistema abaixo, supondo um instrumento computacional capaz de operar com apenas 4 dígitos significativos[8].

$$A\,A^T\,x = b, \text{ onde } A = \begin{bmatrix} 1 & 2 \\ 5 & 100 \end{bmatrix} \quad e \quad b = \begin{bmatrix} 46 \\ 2210 \end{bmatrix}$$

A) Usando $x = (A\,A^T)^{-1}\,b$, e efetuando a inversão pelo método de Gauss-Jordan, tem-se:

$$A\,A^T = \begin{bmatrix} 5 & 205 \\ 205 & 10020 \end{bmatrix}, \quad (A\,A^T)^{-1} = \begin{bmatrix} 1,240 & -0,02538 \\ -0,02538 & 0,0006191 \end{bmatrix}$$

$$x = (A\,A^T)^{-1}\,b = \begin{bmatrix} 0,9502 \\ 0,2007 \end{bmatrix}$$

[8] Os valores contidos após o quarto dígito significativo serão truncados, a exemplo do que ocorre em um computador digital.

B) Usando $x = (A^{-1})^T A^{-1} b$ e a forma produto da inversa de A, tem-se[9]:

$$A^{-1} = \begin{bmatrix} 1 & -0,02222 \\ 0 & 0,01111 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ -5 & 1 \end{bmatrix}$$

$$A^{-1} b = \begin{bmatrix} 2,004 \\ 21,99 \end{bmatrix} ; \quad x = (A^{-1})^T A^{-1} b = \begin{bmatrix} 1,005 \\ 0,1997 \end{bmatrix}$$

C) O resultado correto é:

$$x = \begin{bmatrix} 1,000 \\ 0,2000 \end{bmatrix} ,$$

e portanto a expressão usada em (B), apresenta uma maior estabilidade numérica.

3.3.2 - Armazenamento do Bloco Central

O armazenamento do bloco central em sua forma explícita, conforme sugerido nas expressões 3.4 e 3.5, além da desvantagem citada no

---

[9] Observa-se que esta expressão pode ser calculada, pré-multiplicando (FTRAN) o vetor b pela forma produto da inversa de A, seguido imediatamente da operação de pós-multiplicação (BTRAN) deste resultado pela mesma forma produto.

ítem anterior, implica no uso de áreas de armazenamento muito grandes, pois, embora sendo uma matriz simétrica, requer o armazenamento de $(m^2 + m)/2$ elementos, já que a sua esparsidade diminui rapidamente após relaxar ou ativar restrições do tipo $e_J^T x = 0$.

Com intuito de manter a forma produto já citada, aproveitando deste modo a esparsidade da matriz base, foi criada uma forma mista de bloco central, utilizando a FPI e mais um conjunto de vetores y que garantem uma atualização mais eficiente das grandezas envolvidas no processo.

Dado que se está em uma solução básica, chamando $A_J = B$ e tendo identificado uma restrição $e_{j_1}^T x = 0$ a relaxar, atualiza-se o bloco central através de:

$$y_1 = (B^{-1})^T B^{-1} a_{j_1} \quad \text{e} \quad f_1 = -1/(1 + a_{j_1}^T y_1)$$

Depois de ter relaxado $e_{j_1}^T x = 0$, e identicado a próxima restrição a relaxar, a atualização será:

$$y_2 = (B^{-1})^T B^{-1} a_{j_2} + y_1^T y_1 a_{j_2} f_1$$

e

$$f_2 = -1 / (1 + a_{j_2}^T y_2)$$

$$\vdots$$

$$y_k = (B^{-1})^T B^{-1} a_{j_k} + \sum_{\kappa=1}^{k-1} y_\kappa y_\kappa^T a_{j_k} f_\kappa$$

e

$$f_k = -1 / (1 + a_{j_k}^T y_k) \qquad (3.12\text{-a})$$

Em se tratando de ativar restrições, para a atualização do bloco central, deverá ser efetuada a troca dos sinais dos fatores f. Assim, por exemplo, após relaxar k restrições, e ativar k-1 restrições, o cálculo do último y_{2k} e f_{2k} será:

$$y_{2k} = (B^{-1})^T B^{-1} a_{j_{2k}} + \sum_{\kappa=1}^{k} y_\kappa y_\kappa^T a_{j_{2k}} f_\kappa + \sum_{\kappa=k+1}^{2k-1} y_\kappa y_\kappa^T a_{j_{2k}} f_\kappa$$

$$f_{2k} = +1 / (1 - a_{j_{2k}}^T y_{2k}) \qquad (3.12\text{-b})$$

3.4 - Aspectos Algorítmicos

3.4.1 - Procedimento Geral

Geralmente, quando o gradiente projetado da função objetivo é nulo, tem-se uma situação em que $|J| = n-m$, e $|\bar{J}| = m$, isto é, n-m variáveis são fixadas em zero, e as demais são livres[10]. Obviamen

[10] Verifica-se que isto equivale a solução básica do SIMPLEX.

te, neste caso o bloco central é calculado através da expressão 3.11, sendo armazenado implicitamente na forma produto da inversa da matriz A_J.

Em seguida recupera-se, explicitamente, a primeira linha do bloco central, a fim de efetuar o cálculo das variáveis duais. Esse vetor linha é recuperado através das seguintes operações:

- pré-multiplicar o vetor (1,0,...,0) pela forma produto da inversa de A_J (operação FTRAN);

- pós-multiplicar o vetor obtido pela mesma forma produto (operação BTRAN);

Considerando, entretanto, que o resultado da operação FTRAN, acima citada, é o próprio vetor (1,0,..,0)[11], as operações acima poderão ser substituídas pela operação de pós-multiplicação deste vetor unitário pela forma produto.

Assim, calculadas as variáveis duais para as restrições $e_J^T \; x = 0$, e definida qual a mais propícia[12] para ser relaxada, efetua-se a atualização do bloco central através das equações 3.12-a. Caso não seja encontrada nenhuma restrição a ser relaxada, a solução

---

[11] Este fato pode ser assegurado durante a montagem da forma produto da inversa de A_J, fazendo com que a variável a ser maximizada seja a primeira a ser introduzida na base.

[12] Algumas políticas de cálculo e escolha de variáveis duais *(pricing)* foram testadas. Mostrou-se mais eficiente, aquela em que se determina, entre todas as variáveis ativas que ao serem relaxadas possibilitam um incremento no valor da função objetivo, a variável dual de maior valor absoluto.

obtida é ótima.

Também a primeira linha do bloco central deverá ser atualizada, para que se possa calcular as novas variáveis duais, e consequentemente identificar a próxima restrição a relaxar. Isto poderá ser feito através de:

$$D_J^{-1}(1,\cdot) := D_J^{-1}(1,\cdot) + y_{s,1}\, y_s^T f_s \qquad (3.13)$$

onde y_s e f_s são os parâmetros de atualização do bloco central recém calculados.

Após relaxar k restrições, e tendo atualizado o bloco central e a primeira linha correspondente, calcula-se a projeção do gradiente da função objetivo (g), através da expressão 3.9. Esta projeção servirá de direção para a atualização da solução, conforme é apresentado abaixo:

$$x := x + \lambda_{max} \cdot g \qquad (3.14)$$

onde λ_{max} é um escalar obtido através de:

$$\lambda_{max} = \lambda_{j_r} = \min_{j \in \bar{J}} (-X_j / g_j \mid g_j < 0) \qquad (3.15)$$

Note-se que se não existir $g_j < 0$, a solução do problema é ilimitada.

Identificando a restrição $e_{j_r}^T x = 0$ a ser ativada, efetua-se a atualização do bloco central através da expressão 3.12-b. Também a projeção do gradiente será atualizada, usando-se para tanto a expressão abaixo:

$$g_j := g_j - y_{r,l}\, y_r^T a_j f_r \qquad (3.16)$$

O processo de ativar restrições tem continuidade até que a projeção do gradiente se torne nula.

Na figura 3.1, é apresentado um fluxograma geral do algorítmo PROJECT, utilizado como base no desenvolvimento do sistema computacional.

3.4.2 - Reatualização do Inversa

Observa-se que com o número crescente de vetores-y, a estabilidade numérica e a economia de memória ficam prejudicadas. Uma medida para evitar esta desvantagem é limitar o número 2k de vetores-y em cada iteração, ao final da qual reatualiza-se a inversa.

Mais detalhadamente, quando o gradiente projetado torna-se nulo, a situação equivale, normalmente, à base do SIMPLEX, onde $|\bar{J}| = m$ e $|J| = n-m$. Desta forma, pode-se atualizar a inversa B^{-1} fazendo-a igual a forma produto da inversa da matriz A_J atual, e a solução x_o usando a expressão bem conhecida:

$$x_o = B^{-1} b \qquad (3.17)$$

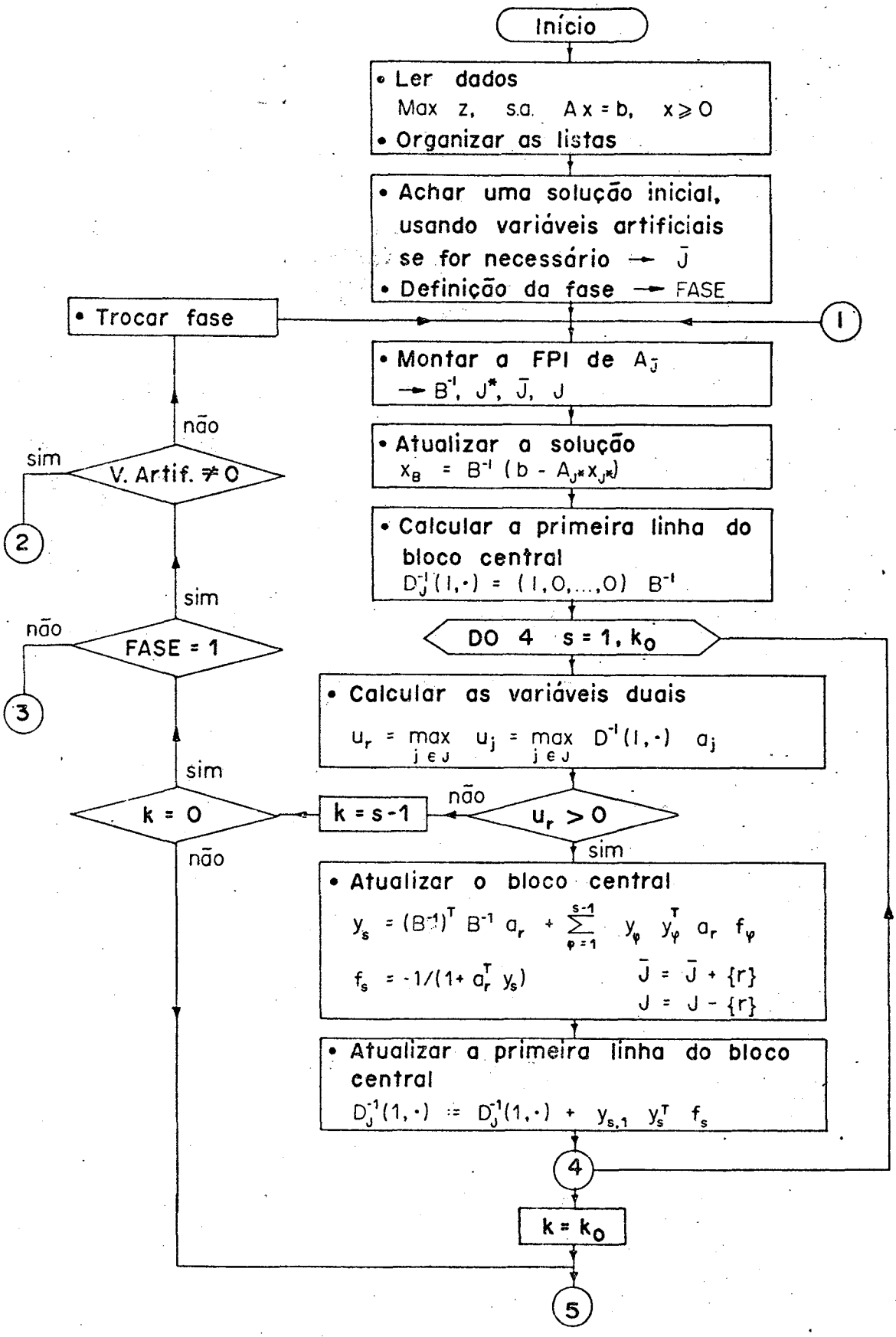

Fig. 3.1.a - Fluxograma do Algorítmo PROJECT

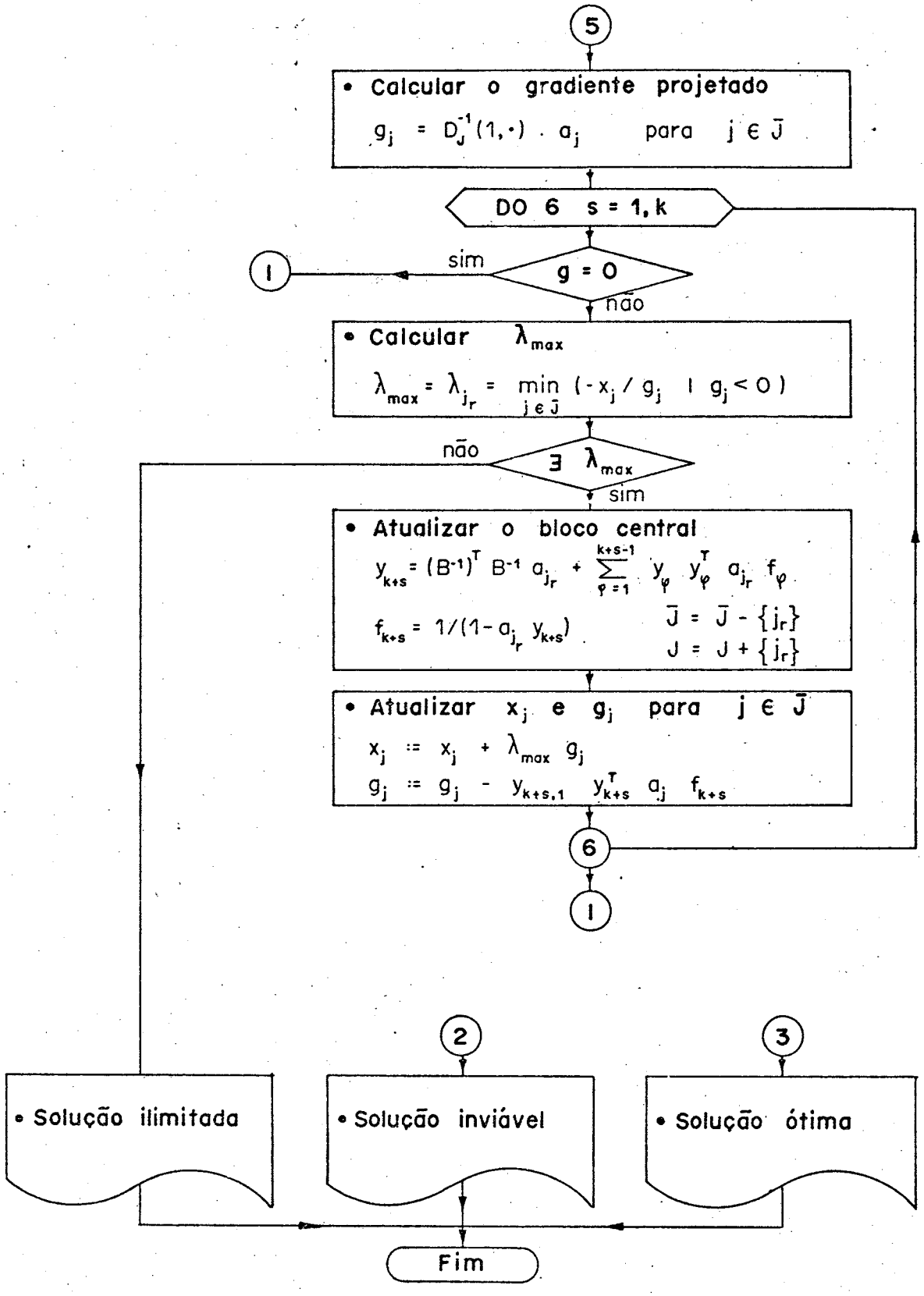

Fig. 3.1.b - Fluxograma do Algorítmo PROJECT (cont.)

O fato do gradiente projetado ficar nulo não significa,necessariamente, que $A_{\bar{J}}$ compõe uma base. Para permanecer com a idéia de atualização, após cada iteração, deve-se fazer algumas considerações.

Como $A_{\bar{J}}$ tem rank m, sempre existe uma submatriz mxm, não singular, obtida a partir de $A_{\bar{J}}$, que fará o papel da base B. Esta base não é necessariamente viável. Para obter-se uma solução viável, segue-se os seguintes passos:

- construir a FPI da base B escolhida;

- atualizar as variáveis básicas através da fórmula:

$$x_B = B^{-1} (b - A_J^* x_J^*) \qquad (3.18)$$

  onde A_J^* são as colunas não básicas de $A_{\bar{J}}$ e x_J^* são as variáveis correspondentes de $x_{\bar{J}}$, solução atual;

- com esta forma produto da inversa calculam-se as variáveis duais correspondentes a todas as variáveis ativas, inclusive aquelas eliminadas da base anterior, dando continuidade ao procedimento normal.

3.4.3 - Dependência Linear e Degeneração

Como já foi comentado, o fato do gradiente projetado ficar nulo não significa, necessariamente, que $A_{\bar{J}}$ é uma base.

Para os casos correntes, apresentou-se no ítem 3.4.2 uma forma de efetuar a reatualização da inversa.

Entretanto, este fato pode ocorrer ao final da primeira fase, quando ainda existem em base algumas variáveis artificiais nulas[13]. Estas variáveis são necessárias à manutenção do rank cheio da matriz básica.

Neste caso, a reatualização da inversa será feita de forma a incluir na base tais variáveis artificiais, não deixando, porém, que as mesmas saiam do zero. Na prática isto é conseguido através da inclusão implícita, na forma produto da inversa, dos vetores unitários relativos a estas variáveis, e da mudança dos índices que controlam os limites destas mesmas variáveis.

---

[13] Veja /ROl/.

CAPÍTULO IV

4 - INSTRUMENTAÇÃO NUMÉRICA

4.1 - Normalização

4.1.1 - Objetivos da Normalização

O processo de normalização de uma matriz consiste em dividir linhas e colunas por valores apropriados, a fim de obter as seguintes vantagens[14]:

- redução do número de condição[15] da matriz básica, o que implica em aumentar a estabilidade numérica;

- redução da dispersão da magnitude dos elementos do modelo original, possibilitando a identificação e eliminação dos elementos pequenos gerados por erros de arredondamento;

- simplificação dos testes de precisão, através da normalização dos vetores coluna.

---

[14] Segundo /BHR/, op. cit., p.296

[15] O número de condição de uma matriz A, dado por ko = $\lambda_{max}/\lambda_{min}$, onde λ_{max} (λ_{min}) é o seu maior (menor) auto-valor, expressa a ordem de grandeza do erro numérico na resolução de um sistema linear Ax = b.

Segundo a conclusão de Tomlin, em /TOM/, a normalização não deve ser usada se o problema de programação linear tiver sido formulada convenientemente. Entretanto, considerando que poucos são os problemas que se caracterizam como tal, um sistema proposto a resolver problemas genéricos de programação linear, deverá contar com opções de normalização que possibilitem a resolução de problemas não formulados apropriadamente.

4.1.2 - Normalização Comum

Consiste em dividir cada uma das linhas, inclusive a da função objetivo do modelo, pelo respectivo maior valor absoluto. Na obtenção deste valor, não se considera o elemento da mão-direita, embora este seja posteriormente dividido pelo valor encontrado.

Em seguida, efetua-se o mesmo procedimento para as colunas, dividindo-as pelo maior valor absoluto de cada uma delas. Este procedimento não se aplica aos elementos da mão-direita.

Cada um destes fatores de normalização é armazenado, para que no final do processo de otimização seja possível recuperar a solução do problema original.

Com relação aos limites superiores das variáveis estruturais, há necessidade de corrigí-los antes do processo de otimização, multiplicando-os pelos fatores de normalização das respectivas colunas, enquanto que os limites superiores das variáveis de folga(se existirem), deverão ser divididos pelos fatores de normalização das respectivas linhas.

4.1.3 - Normalização Geométrica

É um processo iterativo, que consiste em dividir linhas e colunas pela média geométrica entre o máximo e o mínimo valor absoluto, diferentes de zero, conforme as regras básicas já apresentadas para a normalização comum. Estes fatores de normalização são acumulados, sob a forma de produto, na memória do computador.

O processo tem continuidade até que não se obtenha alterações significativas, na matriz, de uma iteração para a outra. Esta situação é identificada quando os fatores de normalização em uma iteração possuirem em média ordem de grandeza entre 0,1 e 10, não excedendo individualmente, em hipótese nenhuma, o intervalo 0,01 a 100.

Posteriormente, passa-se a corrigir os limites superiores das variáveis estruturais e de folga, conforme já mencionado na normalização comum.

4.1.4 - Recuperação da Solução do Problema Original

Após o processo de otimização, há necessidade de se recuperar os valores relativos à solução do problema original. Sejam:

p_j = fator de normalização da j-ésima coluna da matriz;

q_i = fator de normalização da i-ésima linha da matriz;

q_o = fator de normalização da função objetivo;

x'_j , s'_i , c'_j , u'_i , z' = a solução ótima para as variáveis estruturais, de folga, e as respectivas variáveis duais, do problema transformado pelos fatores de normalização;

x_j , s_i , c_j , u_i , z = a solução ótima do problema original.

Então as seguintes relações são válidas:

$$x_j = x'_j / p_j$$

$$s_i = s'_i \cdot q_i$$

$$c_j = c'_j \cdot p_j \cdot q_o$$

$$u_i = u'_i \cdot q_o / q_i$$

$$z = z' \cdot q_o \qquad (4.1)$$

Recupera-se desta forma, todos os valores de interesse para a interpretação da solução do problema original.

4.2 - Tolerâncias

4.2.1 - Considerações sobre as Tolerâncias

A fim de contornar os erros de precisão, e consequentemente evitar problemas de instabilidade numérica, o sistema foi munido de alguns testes de tolerâncias, que se adaptam as situações particulares de cada modelo a ser resolvido.

A calibragem da precisão destes testes foi efetuada através da resolução de aproximadamente 20 problemas de porte e estrutura variados.

4.2.2 - Montagem da FPI

Durante a montagem da forma produto da inversa, é efetuada a operação de pré-multiplicação de vetores a_j, com os vetores-η já incluídos na FPI.

Normalmente nesta operação, podem ocorrer erros de precisão, que se não detectados, poderão causar problemas de instabilidade numérica. O caso mais crítico corresponde a escolha de um pivô cujo valor é diferente de zero, casualmente por erro de precisão.

Considere-se uma situação genérica em que um vetor a_j^{t-1} é pré-multiplicado pela matriz elementar E_t^{-1} de posição pivotal r^t. O vetor atualizado a_j^t é obtido através de:

$$a_{ij}^{t} = a_{ij}^{t-1} + a_{rj}^{t-1} \cdot \eta_{i}^{t} \qquad \text{para } i \neq r = r^{t}$$

$$a_{rj}^{t} = a_{rj}^{t-1} \cdot \eta_{r}^{t} \qquad \text{para } r = r^{t}$$

Nesta operação, se $a_{rj}^{t-1} \neq 0$, $a_{ij}^{t-1} \neq 0$ e $\eta_{i}^{t} \neq 0$, a_{ij}^{t} será considerado nulo se:

$$\frac{a_{ij}^{t}}{a_{ij}^{t-1}} \leqslant \varepsilon \qquad (4.2)$$

onde ε é uma tolerância fixada em função do número de dígitos significativos usados na operação. No sistema programado, utilizou-se $\varepsilon = 10^{-8}$, dado que as operações são efetuadas em precisão dupla.

Desta forma consegue-se evitar a escolha de um pivô potencialmente nulo, não deixando que o mesmo seja gerado.

4.2.3 - Cálculo das Váriaves Duais

O sistema programado prevê um teste de tolerância no valor das variáveis duais, a fim de evitar que entrem na base, variáveis cujo aparente incremento no valor da função objetivo é somente fruto de erros numéricos. Assim, será considerada nula, a variável dual que satisfizer a seguinte condição:

$$\frac{|D_J^{-1}(1,\cdot) \cdot a_j|}{\max_i |a_{ij}|} \leq \max_j |D_J^{-1}(1,j)| \cdot \sqrt{m} \cdot \varepsilon \qquad (4.3)$$

onde ε tem a mesma conotação no ítem 4.2.2.

Na expressão 4.3, $\max_i |a_{ij}|$ é uma boa estimativa para a norma do vetor a_j, tendo em vista o alto grau de esparsidade do mesmo. Já o vetor $D_J^{-1}(1,\cdot)$, que se caracteriza por ter um baixo grau de de esparsidade, pode ter sua norma estimada através de $\max_j |D_J^{-1}(1,j)| \sqrt{m}$, considerando que aproximadamente m componentes deste vetor possuem a mesma ordem de grandeza[16].

Assim sendo, na expressão apresentada, em analogia a forma trigonométrica do produto escalar, ε corresponde ao cosseno do ângulo formado pelos dois vetores. Considerando que $\varepsilon = 10^{-6}$, um valor relativamente pequeno, conclui-se que os vetores em questão, uma vez satisfeita a condição acima, são praticamente ortogonais entre si, e portanto possuem produto escalar nulo.

4.2.4 - Cálculo do Gradiente Projetado

A respeito do gradiente projetado, pode-se efetuar testes de tolerância, com a finalidade de identificar duas situações distintas.

---

[16] Observe que na expressão 4.3 existe uma compensação nos erros cometidos em cada estimativa, o que vem a reforçar a interpretação apresentada, e a validade do teste.

Situação 1

O gradiente projetado se torna nulo, antes de ativar um número de variáveis equivalente ao das variáveis relaxadas.

Em termos numéricos, o reconhecimento desta situação não poderá ser efetuado de forma direta, tendo em vista que erros de precisão são associados ao processo de atualização do gradiente.

Ocorrendo esta situação, e continuando a ativar variáveis, com base em um falso gradiente, as soluções a partir daí calculadas, poderão ser inviáveis, além de não se ter a certeza quanto aos incrementos da função objetivo.

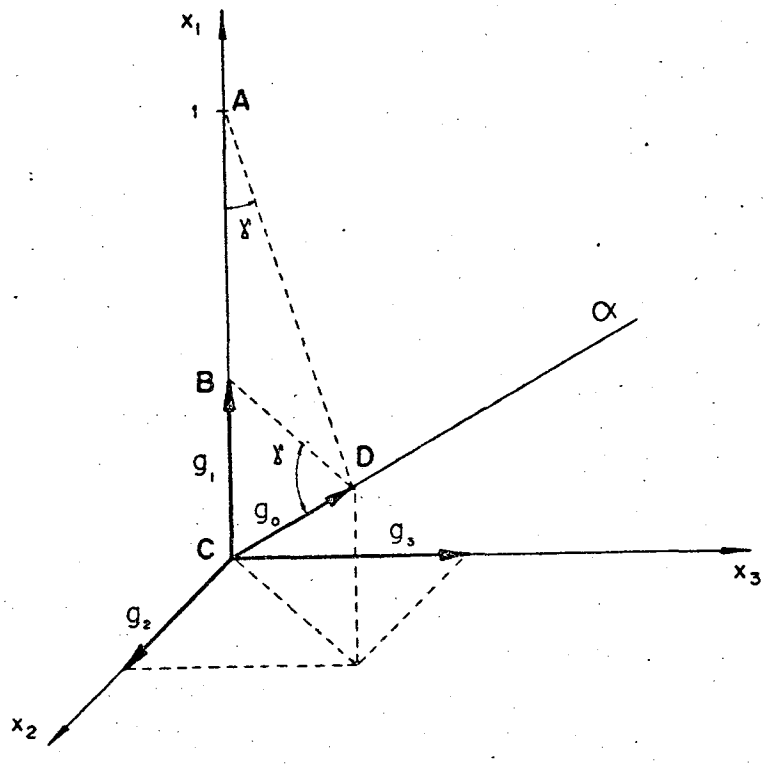

Fig. 4.1 - Gradiente projetado - esquema representativo.

Para evitar isto, considere o esquema da figura 4.1, onde o gradiente unitário é projetado no espaço nulo de uma matriz A, representado por um hiperplano α.

Considerando que existe uma relação entre o gradiente projetado g_o e sua componente g_1, dada por $g_1 = g_o.\text{sen}\gamma$[17], e que sen $\gamma \neq 0$, então pode-se afirmar que o gradiente projetado será nulo, quando a sua primeira componente também for nula.

Numericamente, a componente g_1 é atualizada, após relaxar r variáveis e ativar outras s variáveis, através da expressão:

$$g_1^s = g_1^{s-1} - (y_{r+s,1})^2 . f_{r+s} \qquad (4.4)$$

Dado a forma como g_1^s é obtida, não se pode esperar que a mesma tenha precisão numérica superior a $g_1^o.\varepsilon$[18], onde $\varepsilon = 10^{-8}$ para o sistema programado.

Portanto, o gradiente projetado será considerado nulo, quando a sua primeira componente for menor que a tolerância $g_1^o.\varepsilon$.

Situação 2

A variável escolhida para ser ativada corresponde a uma compo

---

[17] Eventualmente g_2 ou g_3 podem ser nulos, sem que g_o também o seja.

[18] A precisão de um somatório é sempre menor que a precisão da maior parcela.

nente, do gradiente projetado, nula.

Numéricamente, a identificação desta situação não é imediata, tendo em vista que, na atualização, as componentes do gradiente projetado estão sujeitas a erros de precisão.

Entretanto, se esta situação ocorrer, o posterior processo de inversão da base será numéricamente instável. Para avaliar a repercussão deste fato, pode-se fazer uma analogia ao método SIMPLEX, quando da escolha de um elemento pivotal muito pequeno ou nulo.

A fim de obter uma estimativa da tolerância, para cada componente do gradiente projetado, considere-se novamente a figura 4.1, a partir da qual obtém-se, genericamente para o $\mathbb{R}^n$, a seguinte expressão:

$$g_o^2 = g_1^2 + g_2^2 + g_3^2 + \ldots + g_n^2 \qquad (4.5)$$

Igualmente da figura 4.1, obtém-se, da semelhança dos triângulos ΔACD e ΔDCB, que:

$$g_1 = g_o^2 \qquad (4.6)$$

Considerando, na expressão 4.5, que das n componentes de g_o, aproximadamente m são diferentes de zero, e que estas possuem entre si a mesma ordem de grandeza, obtém-se para a expressão 4.6:

$$g_1 \cong m \cdot g_i^2 \qquad (4.7)$$

Relacionando esta expressão à tolerância da primeira componente do gradiente, obtém-se uma tolerância igual a $\sqrt{g_1^o \cdot \varepsilon/m}$ para as demais componentes. Portanto, as variáveis, cujas correspondentes componentes do gradiente projetado são menores que este valor, serão descartadas na escolha da variável a ser ativada.

4.3 - Critério para Manutenção da Esparsidade da Inversa

Basicamente, o problema da manutenção da esparsidade da inversa, consiste na escolha do elemento pivotal.

Considere uma série de vetores-coluna da matriz original, da qual deverá ser escolhida uma coluna para ser transformada em vetor-η.

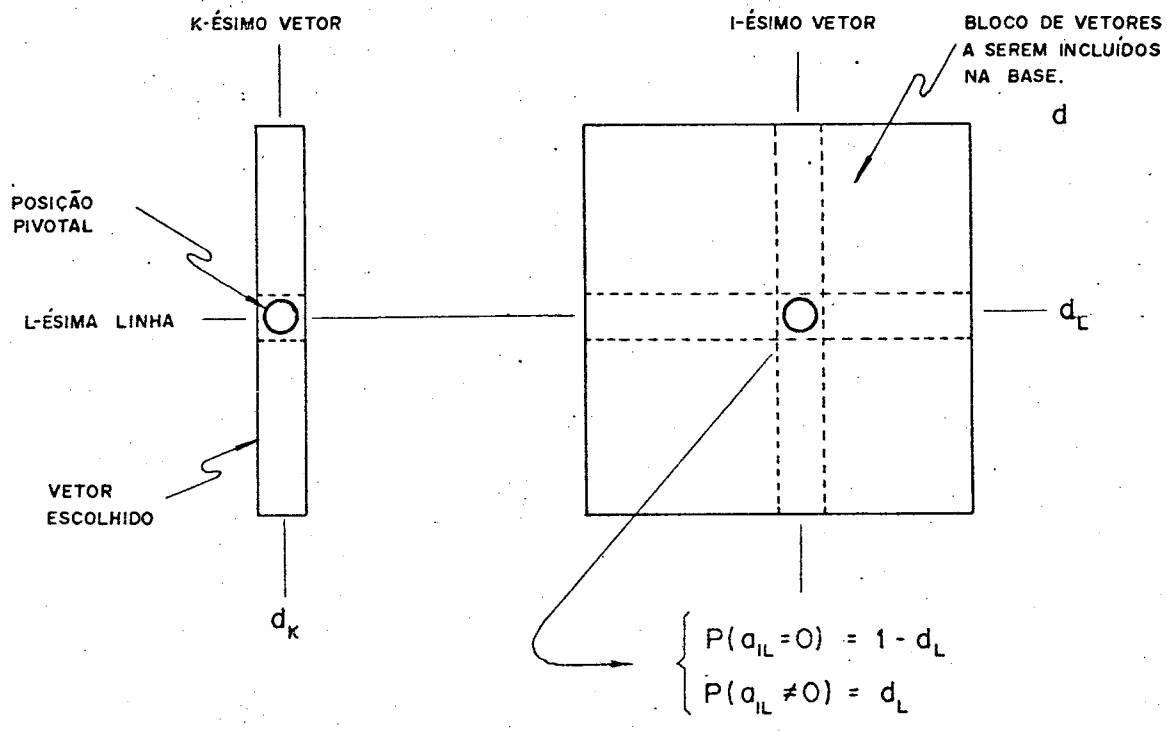

Fig. 4.2 - Esquema para escolha do elemento pivotal.

Sabe-se que a esparsidade dos vetores-η já incluídos na base não será alterada por esta escolha. Resta então, somente a possibilidade de otimizar a esparsidade das colunas ainda não incluídas na base, após a transformação que as mesmas sofrerão, ao serem submetidas à operação de pré-multiplicação (FTRAN) com o último vetor-η criado.

Na figura 4.2, sejam:

d_k = percentagem de elementos não nulos da coluna escolhida para ser transformada em vetor-η;

d_ℓ = percentagem de elementos não nulos da linha, correspondente ao pivô escolhido;

d = percentagem média de elementos não nulos dos vetores a serem incluídos na base, antes da operação FTRAN;

d' = percentagem média de elementos não nulos dos vetores a serem incluídos em base, após a operação FTRAN;

Ao ser submetido ao vetor-η recém-criado, os demais vetores passarão a ter uma esparsidade menor, conforme é mostrado a seguir:

se $a_{i\ell} = 0 \implies d' = d$

se $a_{i\ell} \neq 0 \implies d' = d + (1-d)\cdot d_k$

Dado a probabilidade de cada um dos casos ocorrer, tem-se:

$$E(d') = d \cdot (1-d_\ell) + \{ d + (1-d) \cdot d_k \} \cdot d_\ell \text{ , e portanto:}$$

$$E(d') = d + d_k \cdot d_\ell \cdot (1-d)$$

Portanto, para minimizar E(d'), deverá ser escolhido como pivô, o elemento para o qual o produto $d_k \cdot d_\ell$ é mínimo. Este processo de escolha é conhecido como método de Markowitz.

Entretanto, tal política, implica em se manter um controle sobre todos os elementos da matriz de vetores transformados, o que representa um custo computacional bastante elevado.

Procurando não fugir desta política de escolha, optou-se por uma solução heurística, que consiste em tomar uma coluna cujo d_k seja mínimo, e dentro desta coluna, escolher um elemento pivotal cujo correspondente d_ℓ seja mínimo também.

CAPÍTULO V

5 - IMPLEMENTAÇÃO COMPUTACIONAL

5.1 - Estruturas de Armazenamento dos Dados

5.1.1 - Armazenamento da Matriz Original

Tendo em vista a alta esparsidade da matriz original, que para problemas de médio porte é maior que 95%, optou-se por uma estrutura de listas, onde armazenam-se somente os valores não nulos.

A fim de tornar a entrada de dados facilmente interpretável pelo usuário, projetou-se a mesma dando ênfase às restrições (linhas da matriz). Considerando, entretanto, que em programação linear há necessidade de se recuperar rapidamente as colunas da matriz, foi criado um conjunto de apontadores para este tipo de operação.

Na figura 5.1 é apresentada a estrutura esquemática de armazenamento da matriz original, na qual se evidenciam os apontadores utilizados na recuperação das linhas e colunas, além dos índices de identificação dos elementos.

A cada elemento da matriz é associada uma célula de armazenamento composta por duas regiões distintas. Na região superior, dividida em quatro áreas, armazenam-se o apontador do próximo elemento não nulo da coluna, o índice da linha, o índice da coluna e o aponta-

dor do próximo elemento não nulo da linha, respectivamente. Na região inferior, armazena-se o próprio valor do elemento.

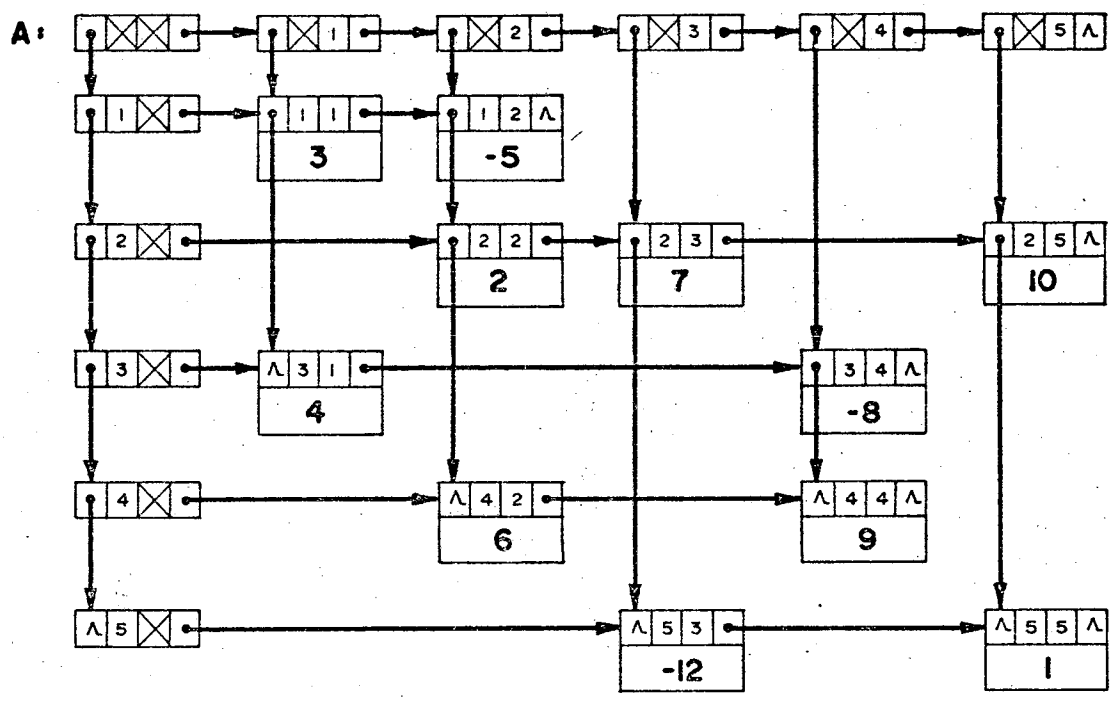

Fig. 5.1 - Representação esquemática do armazenamento da matriz original[19].

Para referenciar cada linha e cada coluna, são mantidos, à parte, apontadores que indicam a localização do primeiro elemento não nulo, da linha ou coluna, na lista. As células de armazenamento destes apontadores são mais simples que as anteriormente citadas. São compostas de três elementos apenas: apontador da próxima célula de referência, índice da linha ou coluna, e apontador do primeiro elemento não nulo da linha ou coluna.

[19] /PFA/, op.cit., p.231.

5.1.2 - Armazenamento da Matriz Básica Inversa

O processo de inversão utilizado (FPI), possibilita a obtenção de inversas a partir do armazenamento de vetores-η, que possuem uma esparsidade relativamente alta. A fim de aproveitar esta particularidade, foi projetado um conjunto de listas, apontadores e índices, que passamos a denominar de η-*file*.

No que diz respeito a sua montagem, efetua-se tratamentos distintos entre vetores unitários (variáveis de folga ou artificiais),e vetores não unitários (variáveis estruturais), dado que os primeiros, quando introduzidos *a priori* na base, não afetam, senão em sinal, os resultados das operações subsequentes.

A fim de identificar a necessidade (ou não) desta troca de sinal, em uma operação qualquer, é mantido um vetor com (-1,0,1), indicando as posições pivotais ocupadas por vetores unitários negativos, e portanto sujeitas às trocas de sinal, as posições pivotais não ocupadas e aquelas ocupadas por outros vetores, respectivamente.

Considerando que durante a montagem de η-*file* é efetuada a transformação das colunas básicas da matriz original, através de operações de pré-multiplicação com o último vetor-η gerado, dimensionou-se uma estrutura de lista a fim de armazenar tanto os vetores-η, quanto as colunas básicas transformadas, sem necessitar a duplicação da área de armazenamento. Isto pode ser feito tendo em vista que, ao ser transformada em vetor-η, a coluna básica deixa de ser necessária.

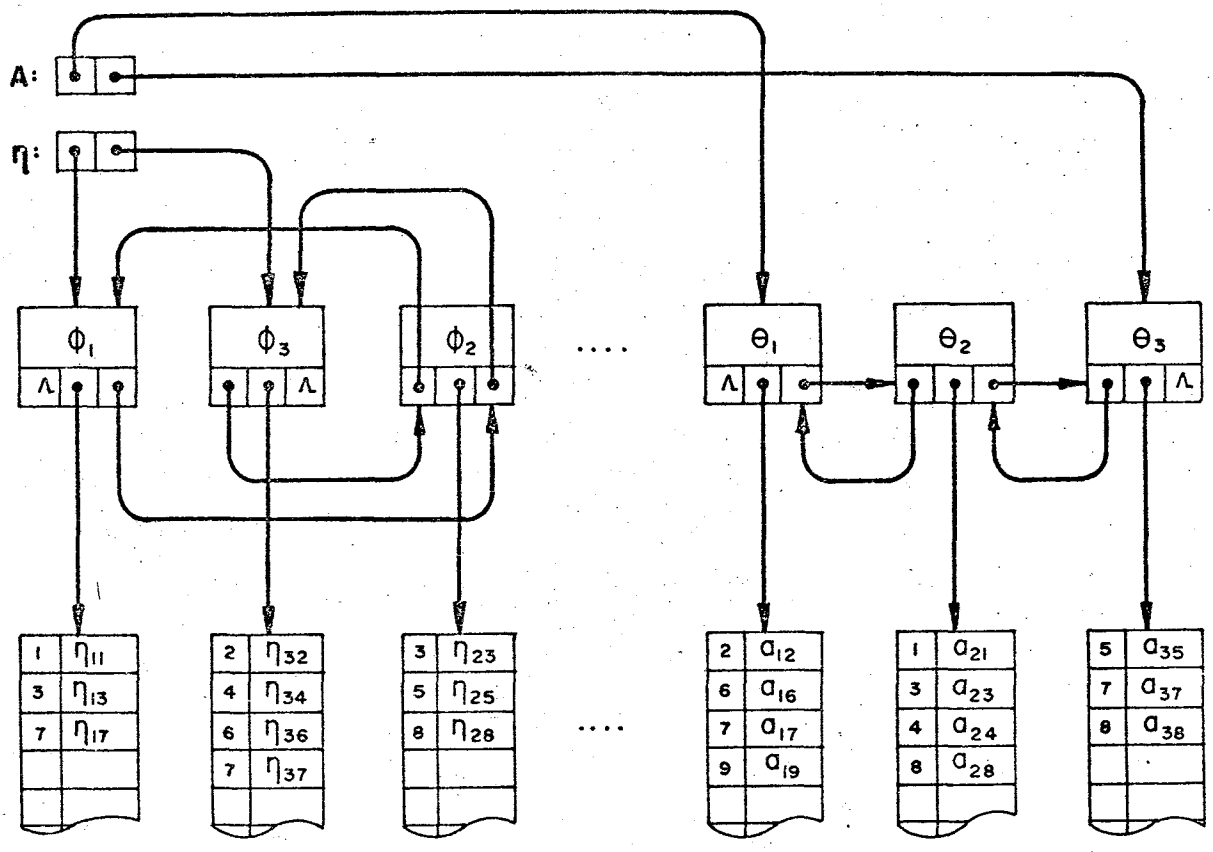

Fig. 5.2 - Lista de armazenamento dos vetores-η explicítos.

A estrutura esquemática da citada lista é apresentada na figura 5.2, onde a cada vetor é associada uma célula na qual está armazenado um índice de identificação do elemento pivotal (ϕi), em se tratando de vetor-η, ou de identificação da variável que originou a coluna básica transformada (θi), além de um conjunto de apontadores que possibilitam a recuperação do vetor em questão, e a identificação do vetor anterior e posterior a este.

Na mesma estrutura mantêm-se, ainda, apontadores que identificam o primeiro e o último vetor de cada tipo. Desta forma pode-se recuperar estes vetores na sequência correta em que foram criados.

A fim de armazenar os elementos não nulos destes vetores, a lista

é composta pelos pares (posição que o elemento ocupa no vetor, valor do elemento).

5.1.3 - Armazenamento dos Vetores y

Considerando que os vetores y possuem uma esparsidade muito baixa, optou-se pelo seu armazenamento na forma explícita. A cada iteração permite-se relaxar no máximo k variáveis, e ativar outras k, o que implica em gerar, a cada iteração, 2k vetores y com os correspondentes fatores.

5.1.4 - Outras Estruturas de Armazenamento

Além das listas e apontadores já citados, foram criados vetores com as seguintes finalidades:

- armazenar nomes de referência para as variáveis e as restrições do problema;
- armazenar os fatores de normalização, por linha e por coluna;
- armazenar o gradiente projetado relativo às variáveis relaxadas;
- armazenar o limite superior e inferior de cada uma das variáveis;

- armazenar a função objetivo de trabalho, e a função objetivo original;

- armazenar a situação de cada variável, isto é, armazenar um índice que identifique se a variável em questão esta livre (na forma produto da inversa ou não), ou se está ativa (em seu limite superior ou inferior);

- armazenar a primeira linha do bloco central.

5.2 - Estrutura do Sistema

5.2.1 - Subprograma ATFOB

Ao final de cada iteração, durante a primeira fase, este subprograma é chamado, a fim de eliminar do problema as variáveis artificiais que atingiram o seu limite inferior. Com isto, consegue-se que o vetor coluna da variável de folga correspondente, volte a situação de vetor unitário. As principais operações efetuadas são:

- identificação das variáveis artificiais ativadas na iteração precedente;

- soma dos coeficientes das linhas correspondentes a estas variáveis na, função objetivo de trabalho;

- eliminação destas variáveis artificiais do problema.

5.2.2 - Subprograma ATLIS

Este subprograma é chamado pelo subprograma ETAFIL[20], sempre que uma variável estrutural é escolhida para participar da forma produto da inversa, ou quando é identificada como linearmente dependente das variáveis já existentes na FPI. Sua finalidade é atualizar as listas de controle do η-*file*. As principais operações efetuadas são:

- atualização do número de elementos das linhas da matriz de colunas básicas transformadas;

- atualização dos apontadores de avanço e retrocesso da sequência de vetores básicos transformados.

5.2.3 - Subprograma ATUAX

Após a montagem do η-*file*, a solução do problema é atualizada, a fim de evitar que os erros de precisão se acumulem até o final do processamento. Esta operação é efetuada pelo subprograma ATUAX, através dos seguintes passos:

- inicializar um vetor de trabalho, com os coeficientes da mão direita;

- subtrair do vetor de trabalho, o produto das variáveis não in

[20] Veja subprograma ETAFIL, ítem 5.2.6.

cluídas na FPI, diferentes de zero, pelos respectivos vetores coluna;

- submeter o vetor de trabalho a operação FTRAN[21] com a forma produto da inversa;

- atualizar o vetor solução, com os valores encontrados na operação FTRAN.

5.2.4 - Subprograma BTRAN

Este subprograma efetua a operação BTRAN entre um vetor explícito e a matriz básica inversa, armazenada na sua forma produto (FPI). Nele são efetuadas as seguintes operações:

- produto do vetor explícito com cada uma das matrizes elementares (vetores-η), na ordem inversa em que foram geradas;

- alteração do sinal das componentes do vetor explícito, em função dos vetores unitários negativos existentes na FPI.

5.2.5 - Subprograma ESPIV

Este subprograma é chamado pelo subprograma ETAFIL, afim de determinar, para um vetor básico transformado, qual a melhor posição

---

[21] Ver subprograma FTRAN, ítem 5.2.8.

para o pivô. Esta escolha é feita de modo a satisfazer as seguintes condições:

- a posição escolhida não pode ter sido usada como pivô, por outro vetor-η;

- a posição escolhida deverá corresponder um elemento diferente de zero;

- a posição escolhida, satisfazendo os requesitos acima, deverá corresponder ã linha de maior esparsidade da matriz composta pelas demais colunas básicas transformadas.

Determinada esta posição ideal de pivô para o vetor em questão, é verificado se a esparsidade da linha em que este se encontra, é maior, e portanto melhor, que a de um possível vetor anteriormente eleito como ótimo.

5.2.6 - Subprograma ETAFIL

Com este subprograma inverte-se a base, armazenando-a na forma produto da inversa. A fim de manter a esparsidade, procede-se da seguinte forma:

- inclui-se na FPI as variáveis artificiais;

- inclui-se na FPI as variáveis de folga;

- inclui-se os vetores das variáveis estruturais na lista de vetores-η, após alterar o sinal das componentes em função dos vetores unitários negativos existentes na forma produto da inversa; nesta operação são inicializados os apontadores e índices necessários à posterior recuperação desses vetores;

- se ainda não existir uma base, escolhe-se, entre os vetores básicos transformados, aqueles que possuirem a maior esparsidade, submetendo-os ao subprograma ESPIV, que identificará o melhor elemento pivotal;

- atualiza-se as listas através do subprograma ATLIS, eliminando do conjunto de vetores básicos transformados, a coluna correspondente ao pivô escolhido;

- cria-se um vetor-η, tomando como pivô o elemento anteriormente escolhido;

- submete-se à operação FTRAN todos os vetores básicos transformados. Esta operação é feita chamando alternadamente os subprogramas FTRAN1 e FTRAN2[22].

5.2.7 - Subprograma FILEY

Este subprograma é responsável pela criação dos vetores y e seus respectivos fatores. Identificada uma variável a ser ativada ou

---

[22] Veja subprogramas FTRAN1 e FTRAN2, ítem 5.2.9.

relaxada, submete-se o índice da mesma, juntamente com um indicador do tipo de operação desejada, às seguintes operações básicas:

- recuperar o vetor coluna da variável a ser ativada, ou relaxada, em sua forma explícita;

- submeter este vetor à operação FTRAN[23], e o resultado desta à operação BTRAN;

- inicializar o novo vetor y, com o resultado desta operação;

- somar, a este vetor y, as parcelas relativas aos vetores y já existentes, conforme é apresentado nas equações 3.12.a e 3.12.b[24];

- calcular o fator correspondente ao vetor-y criado.

5.2.8 - Subprograma FTRAN

A operação de pré-multiplicação efetuada pelo subprograma FTRAN, consiste na realização do produto entre a matriz inversa, armazenada na forma de lista (FPI), e um vetor explícito. Para tanto

---

[23] Quando a variável a ser ativada compõe o η-*file*, a operação FTRAN pode ser dispensada, dado que o resultado é um vetor unitário.

[24] O produto escalar entre o vetor y e o vetor a_j é efetuado através do subprograma PROINT, apresentado no ítem 5.2.16.

são desenvolvidas as seguintes etapas:

- alterar o sinal das componentes do vetor explícito, em função dos vetores unitários negativos existentes na forma produto da inversa;

- efetuar o produto das matrizes elementares (vetores-η), na sequência em que foram geradas, com o vetor explícito que está sendo submetido.

5.2.9 - Subprogramas FTRAN1 e FTRAN2

Conforme foi apresentado no ítem 5.2.6, os subprogramas FTRAN1 e FTRAN2 são responsáveis pela atualização dos vetores básicos transformados em relação ao último vetor-η incluído na forma produto da inversa.

Considerando que a estrutura de ambos é similar, variando apenas a situação em que cada um é utilizado, passa-se a descrever de forma genérica, as operações efetuadas com cada um dos vetores básicos transformados:

- identificar, através de um algorítmo de busca binária, o elemento correspondente à posição pivotal do último vetor-η incluído na FPI. Se este elemento é nulo, efetuar a simples translação do vetor para uma nova posição dentro da lista, atualizando os seus apontadores;

- se o elemento não é nulo, efetuar, através da variação conveniente dos índices, o produto entre o vetor-η, por último incluído na FPI, e o vetor básico transformado, ambos armazenados na forma de lista. Eliminar nesta etapa, os valores não significativos, observando, tanto aqui quanto na etapa anterior, a disponibilidade de espaço para a inclusão do vetor atualizado, sem prejuízo dos valores ainda necessários ao processo de inversão em curso.

5.2.10 - Subprograma GRADI

Tendo identificado e relaxado as variáveis que incrementarão o valor da função objetivo, este subprograma é chamado pelo OPTIM[25], para calcular as componentes do gradiente da função objetivo. Para tanto, efetua as seguintes operações básicas:

- inicializa as componentes do gradiente, com o produto escalar entre a primeira linha da matriz básica inversa e os respectivos vetores colunas;

- se a componente do gradiente na direção da variável a ser otimizada é nula, retorna ao subprograma OPTIM, para efetuar o término do processamento. Em caso contrário, retorna para este mesmo subprograma, continuando com o processamento normal.

[25] Veja subprograma OPTIM, ítem 5.2.14.

5.2.11 - Subprograma IMPRO

Seguindo na direção do gradiente da função objetivo, ao se atingir uma restrição de não negatividade ou de limite superior de uma variável, há necessidade de se ativar a variável correspondente, e recalcular o gradiente na busca da solução ótima. Este processo é efetuado pelo subprograma IMPRO, que consiste, basicamente, nos seguintes passos:

- identificar qual variável básica atingirá, em primeiro lugar, uma restrição de não negatividade ou limite superior. Se isto não ocorrer, trata-se de um problema de solução ilimitada, e neste caso o processamento termina com a apresentação da última solução;

- ativar a variável identificada, atualizando-a quanto ao seu valor e situação. Gerar um novo vetor y através do subprograma FILEY;

- atualizar os valores das demais variáveis relaxadas;

- atualizar o gradiente da função objetivo, em relação ao último vetor y gerado.

Estes passos são efetuados até que sejam ativadas tantas variáveis quantas foram relaxadas, ou quando a primeira componente do gradiente for menor que a tolerância estabelecida[26].

---

[26] Veja ítem 4.2.4.

5.2.12 - Subprograma LEORG

Este subprograma é responsável pela leitura e organização dos dados, além da preparação de uma solução básica inicial. É um subprograma independente de todo o sistema, e pode ser substituído por outro quando o problema a ser resolvido tiver características especiais[27].

É compreendido das seguintes etapas:

- leitura das opções;
- leitura de comentários;
- leitura do tipo de otimização desejada;
- leitura da função objetivo original;
- leitura das restrições estruturais do problema;
- leitura dos limites especiais para as variáveis estruturais;
- viabilizar o vetor b, isto é, trocar o sinal das linhas quando for necessário;

[27] Em se tratando de problemas de transporte ou atribuição, ou ainda outra aplicação especial, pode-se alterar este subprograma sem prejuízo dos demais.

- executar a normalização, usando para tanto o subprograma SCALIN [28];

- preparar uma solução inicial, gerando, se necessário, variáveis artificiais;

- montar a função objetivo de trabalho;

- identificar o máximo valor absoluto existente em cada coluna da matriz original, para posterior utilização nos testes do subprograma PROSIG[29].

Ao longo da execução do subprograma LEORG, são efetuados diversos testes de consistência da massa de dados, conforme se apresenta no apêndice I, ítem C.

5.2.13 - Subprograma MONFOR

Quando é encontrada uma solução viável para o problema proposto, o subprograma MONFOR é chamado a fim de efetuar a montagem da função objetivo de trabalho, tomando como referência os coeficientes da função objetivo original. Para as variáveis de folga e artificiais, são assumidos coeficientes nulos.

---

[28] Veja subprograma SCALIN, ítem 5.2.17.

[29] Veja subprograma PROSIG, ítem 5.2.16.

5.2.14 - Subprograma OPTIM

É o subprograma que gerencia todo o sistema. Sua estrutura, basicamente, é igual a do algorítmo PROJECT, e compõe-se das seguintes etapas:

- inverter a base através do subprograma ETAFIL;

- calcular as variáveis duais, relativas às variáveis ativadas, e relaxar aquelas que possibilitem um incremento no valor da função objetivo. Isto é feito através do subprograma VARDU [30];

- caso nenhuma variável tenha sido relaxada no ítem anterior tem-se:

  se fase = 1 e se ainda existirem variáveis artificiais não nulas na base, a solução é inviável, e o processamento é interrompido com fracasso; caso não existam tais variáveis na base, a solução é viável, e o processamento continua com a troca de fase, montagem da nova função objetivo de trabalho, e o retorno a primeira etapa deste subprograma;

  se fase = 2, a solução é ótima - o processamento se encaminha para a saída dos resultados;

---

[30] Veja subprograma VARDU, ítem 5.2.18.

- caso alguma variável tenha sido relaxada, calcular a projeção do gradiente da função objetivo, sobre o espaço nulo da matriz de restrições ativas. Esta operação é feita pelo subprograma GRADI;

- chamar o subprograma IMPRO, a fim de andar na direção do gradiente calculado, ativando as variáveis à medida que um de seus limites seja atingido;

- atualizar a função objetivo de trabalho (somente se for fase = 1), eliminando do problema as variáveis artificiais ativadas no decorrer da iteração; este processo de eliminação das variáveis artificiais é efetuado pelo subprograma ATFOB;

- voltar a etapa de inversão da base, ou diretamente ao cálculo das variáveis duais, se ainda houver possibilidade de criar novos vetores y.

5.2.15 - Subprograma PRISOL

A fim de apresentar os resultados, foi criado um subprograma independente do sistema, e que pode ser substituido por outro, quando houver interesse em algum tipo de utilização especial.

Basicamente consiste em recuperar a solução, através dos fatores de normalização armazenados, apresentando-a sob a forma de relatório, conforme descrito no apêndice I, ítem D.

5.2.16 - Subprogramas PROINT e PROSIG

Estes subprogramas são utilizados para efetuar o produto escalar entre um vetor explícito e uma coluna da matriz original, que se encontra armazenada na forma de lista. A diferença básica entre ambos, é que o PROSIG efetua um teste de tolerância, a fim de avaliar se o resultado obtido é significativo no problema, ou se é consequência de erros de precisão. A utilização deste subprograma é específica para o cálculo das variáveis duais, enquanto que o PROINT é utilizado nos demais casos em que esta operação se faz necessária.

5.2.17 - Subprograma SCALIN

O subprograma SCALIN é utilizado para efetuar a normalização da matriz original. As operações efetuadas estão descritas no ítem 4.1, que trata das políticas de normalização disponíveis no sistema.

5.2.18 - Subprograma VARDU

O subprograma VARDU é utilizado para identificar e relaxar as variáveis que possibilitam um incremento no valor da função objetivo. Para tanto é escolhido, a cada vez, a variável dual que promete o incremento acima citado, e cujo valor absoluto, em relação a base generalizada, é máximo. Para esta variável, é gerado um

vetor y[31], o que corresponde a relaxá-la. A estrutura deste subprograma é a seguinte:

- calcular a primeira linha da matriz básica inversa, através da operação BTRAN de um vetor unitário, sobre a forma produto da inversa;

- atualizar a primeira linha da matriz básica inversa, em função dos vetores y gerados;

- verificar se o número máximo de vetores y a serem criados já foi atingido; em caso afirmativo, retornar ao subprograma OPTIM;

- para as variáveis ativadas, calcular o produto escalar entre a primeira linha da matriz básica inversa, e o vetor coluna correspondente; esta operação é feita através do subprograma PROSIG;

- dentre as variáveis duais calculadas, escolher as que produzem um incremento na função objetivo, e dentre estas, a que tiver o maior valor absoluto;

- relaxar a variável correspondente, e voltar a segunda etapa deste subprograma.

Este processo tem sequência, até que não existam mais variáveis a serem relaxadas, ou quando o número máximo de vetores y a serem criados tiver sido atingido.

---

[31] Veja subprograma FILEY, capítulo 5.2.7.

CAPÍTULO VI

6 - AVALIAÇÃO DO MÉTODO PROPOSTO

6.1 - Análise da Complexidade do Algorítmo

A complexidade do algorítmo proposto será analisada em função do número de operações do tipo multiplicação e/ou divisão, por iteração.

Entende-se por iteração, o processo de relaxar k restrições, ativando em seguida, outras k restrições.

Não serão considerados casos especiais, como a degeneração, nem os cálculos necessários a reinversão da base, tendo em vista que a eficiência desta depende muito do método usado[32].

Sejam:

ℓ_p = número de vetores-η na forma produto da inversa de A_J;

$1 - t$ = grau de esparsidade da matriz A;

$1 - t_{\eta p}$ = grau de esparsidade (média) dos vetores-η.

O número de operações necessárias à efetivação de uma iteração no algorítmo PROJECT, é dado pela soma das parcelas discriminadas a

---

[32] veja /ORH/, pp. 73-82.

baixo:

- cálculo da primeira linha do bloco central

$$\ell_p \quad t_{\eta p} \quad (m+1)$$

- cálculo das variáveis duais

$$k \quad t \quad (m+1) \quad (n-m-k/2)$$

- atualização do bloco central

$$k \quad \{ \; (m+1) \; [\; 4 \quad \ell_p \quad t_{\eta p} + 2k - 1 + (2\,k + 1)\; t \;] + (2k+1) \}$$

- atualização da primeira linha do bloco central

$$k \quad (m+2)$$

- cálculo do gradiente projetado

$$(m+k+1) \quad (m+1) \quad t$$

- verificação da restrição a ser ativada

$$k \quad [\; m + (k+1)/2 \;]$$

- atualização do gradiente projetado

$$k \quad [\; (m+1) + (k+1)/2] \quad [\; t \quad (m+1) + 2 \;]$$

- atualização da solução

$$k \quad [\; (m+1) + (k+1)/2 \;]$$

Considerando os valores correntes destes parâmetros, esta soma se aproxima à:

$$m\ k\ \ (\ 2k + n\ \ t + 4\ \ \ell_p\ \ t_{\eta p}\ )$$

Da mesma forma, o número de operações necessárias para efetuar uma iteração SIMPLEX é:

- obtenção da primeira linha da base inversa

$$\ell\ \ t_\eta\ \ (m+1)$$

- cálculo das variáveis duais *(pricing)*

$$t\ \ (m+1)\ \ \ (n-m)$$

- atualização da coluna pivotal

$$\ell\ \ t_\eta\ \ (m+1)$$

- atualização da solução x_B e coluna θ

$$2\ t_\eta\ \ (m+1)$$

- cálculo do novo vetor-η na FPI

$$t_\eta\ \ (m+1)$$

onde:

ℓ = número médio de vetores-η na forma produto da inversa;

$1 - t_\eta$ = grau de esparsidade (médio) dos vetores-η;

$1 - t$ = grau de esparsidade da matriz A.

A soma destas parcelas pode, tendo em vista os valores correntes dos parâmetros acima, ser aproximada para:

$$(2\ \ell + 3)\ t_\eta\ m + t\ m\ (n - m)$$

Estas duas somas não podem ser comparadas diretamente, pois:

I) enquanto que ℓ cresce com o número de trocas de base, ℓ_p fica constante dentro de uma iteração, permanecendo sempre menor ou igual a m;

II) t_η aumenta com o número de iterações, enquanto que $t_{\eta p}$ permanece igual a densidade inicial da FPI;

III) em uma iteração, o método proposto percorre um sub-espaço de dimensão k, enquanto que o SIMPLEX abre somente um sub-espaço de dimensão 1.

Exemplo 6.1

Sejam os parâmetros abaixo:

$t = 0,05$	$t_{\eta p} = 0,05$	$t_\eta = 0,08$	$k = 15$
$m = 400$	$n = 1200$	$\ell_p = 200$	$\ell = 225$

Assim, uma iteração no PROJECT custaria na ordem de 780000 operações do tipo multiplicação e divisão, enquanto que no SIMPLEX, seria da ordem de 31000 operações desta natureza.

Portanto, se uma iteração do PROJECT equivaler em média a 26 trocas de base do SIMPLEX, ou mais, o método proposto será mais eficiente.

Quanto a equivalência entre o número de iterações destes dois mé

todos, nada pode ser afirmado *a priori* , dado que a mesma está relacionada fortemente com a estrutura do problema a ser resolvido.

Na figura 6.1, são apresentados dois exemplos que demonstram este fato. No exemplo (A), o SIMPLEX efetua um número menor de iterações que o PROJECT, para obter a solução ótima. Já no exemplo (B) a situação é oposta.

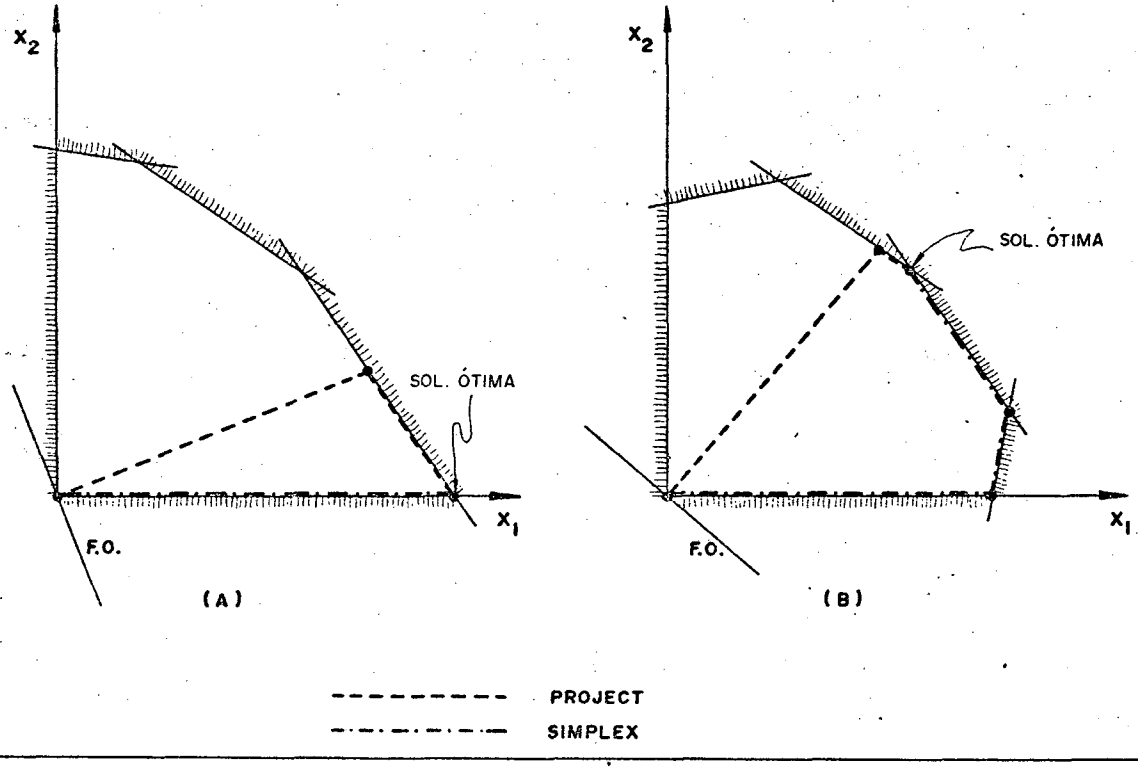

Fig. 6.1 - Iterações PROJECT x iterações SIMPLEX

6.2 - Resultados Numéricos

O algorítmo implantado foi avaliado segundo dois aspectos distintos:

- estabilidade numérica

- eficiência computacional

No que diz respeito à estabilidade numérica, /GLO/ sugere um caso clássico de problema numericamente instável. Aproveitando a estrutura do problema sugerido, foram elaborados e resolvidos com êxito, problemas de até 250 restrições e 250 variáveis estruturais.

Também foram resolvidos diversos problemas de atribuição, para os quais ocorrem casos de degeneração, e que as experiências com o PROJECT evidenciam a possibilidade de ocorrer instabilidade numérica. Os resultados obtidos, também foram satisfatórios.

Em geral, os resultados obtidos para os problemas acima, apresentam na ordem de 8 (oito) dígitos significativos.

Quanto a eficiência computacional, são apresentadas no quadro 6.1 as características de alguns problemas usados para teste do algo

PROB.	RESTR.	VARIÁVEIS			DENS.	TEMPO DE CPU	
		ESTR.	FOL.	ART.	(%)	SIMPLEX	PROJECT
EST40	40	40	0	40	7.38	1.06	2.58
ATR15	30	225	0	30	6.67	2.86	5.92
GER 1	70	211	70	0	1.90	9.48	11.59
GER 2	80	241	80	0	1.66	12.36	15.21
GER 3	120	242	120	0	1.65	20.61	22.47
GER 4	160	242	160	0	1.34	22.26	24.78
GER 5	200	242	200	0	1.40	*****	29.37

Quadro 6.1 - Comparação entre tempos de CPU do SIMPLEX e do PROJECT

rítmo PROJECT, e o respectivo tempo de CPU, em segundos, necessário a sua resolução. No referido quadro, consta ainda o tempo de CPU gasto por um algorítmo SIMPLEX primal-dual, que dispõe de tratamento especial para variáveis com limite superior.

A comparação destes tempos pode ser efetuada somente para problemas de pequeno porte, pois o SIMPLEX usado não comporta problemas de dimensões maiores com a memória disponível. Percebe-se entretanto, que o PROJECT tende a melhorar em relação ao SIMPLEX, à medida que as dimensões do problema aumentam e a densidade diminui. Esta tendência também é constatada para problemas de estrutura menos rígida, isto é, problemas onde as restrições são do tipo menor ou igual.

CAPÍTULO VII

7 - COMENTÁRIOS, CONCLUSÕES E RECOMENDAÇÕES

7.1 - Comentários

Esta não foi a primeira vez que se implantou, computacionalmente, um algorítmo PROJECT. Experiências anteriores, com sistemas capazes de resolver problemas de 80 restrições e 250 variáveis (estruturais, folga e artificiais), foram efetuadas, das quais obteve-se uma série de informações para o desenvolvimento deste trabalho. Ressalta-se que o novo sistema, para uma mesma partição da memória do computador (640 kbytes), é capaz de resolver problemas de 300 restrições e 750 variáveis estruturais, sendo na ordem de 5 vezes mais eficiente que seu antecessor.

Porém esta não é a capacidade máxima do mesmo. Com uma partição de 1500 kbytes, pode-se resolver problemas de até 1400 restrições e 4200 variáveis estruturais.

Na implementação do novo sistema, foram encontradas diversas dificuldades, das quais pode-se ressaltar:

- a formulação do algorítmo computacional, a partir do algorítmo conceitual apresentado em /RCM/;

- a idealização da estrutura de armazenamento dos dados, na qual

desejava-se manter apenas os elementos não nulos. Uma vez caracterizada a estrutura, com seus apontadores e índices, houve a necessidade de adequá-la à linguagem computacional utilizada;

- o desenvolvimento do *lay-out* da entrada dos dados e saída dos resultados, de forma a tornar o sistema confortável para o usuário;

- a manutenção da esparsidade da forma produto da inversa; inicialmente, idealizou-se uma estrutura que incluia na base as variáveis artificiais, de folga e estruturais, respectivamente, sem se preocupar com a ordem de inclusão destas últimas. Após uma rápida análise, concluiu-se que esta não era uma política razoável, tendo em vista que a esparsidade diminuiria muito rapidamente. Reestudou-se o problema, obtendo-se a política sugerida no ítem 4.3, que obrigou a reformular a estrutura de armazenamento da inversa;

- os problemas com a estabilidade numérica que, para serem resolvidos, necessitaram de um conjunto de testes para avaliação dos erros de precisão; estes testes foram formulados de forma a se adaptarem as situações particulares de cada problema.

Uma preocupação constante tida ao longo do projeto do sistema, foi a otimização do processo computacional, no sentido de economizar em memória e tempo de CPU. Não raras foram as situações em que estes dois objetivos se conflitaram, implicando na sub-otimização de um deles.

7.2 - Conclusões

Em termos puramente filosóficos, a idéia contida no algorítmo PROJECT é, sem dúvida nenhuma, melhor que o tradicional caminhamento via arestas do poliedro convexo do PPL. Entretanto, na prática, a maior barreira que tem sido encontrada, é a recuperação das informações para o cálculo do gradiente projetado da função objetivo, que tem um custo computacional relativamente alto. Neste sentido, a forma produto da inversa, utilizada neste trabalho, tem um papel importante, melhorando sensivelmente o desempenho do PROJECT.

Embora não seja possível afirmar que o método proposto é mais eficiente que o tradicional SIMPLEX, as comparações feitas, em termos de tempo de CPU, servem para caracterizar o PROJECT como um algorítmo computacional viável para a resolução de problemas genéricos de programação linear de médio e grande porte, o que é inédito entre os métodos não SIMPLEX. É neste fato que reside a maior contribuição dada por este trabalho.

7.3 - Recomendações

Evidentemente, este trabalho não esgota os estudos acerca do PROJECT. Os resultados obtidos até aqui são encorajadores, sendo que se recomenda a continuação do estudo, através dos seguintes trabalhos:

- programação de um SIMPLEX com os mesmos tratamentos e opções dados ao PROJECT, utilizando ao máximo as estruturas já implan

tadas, a fim de realizar uma avaliação mais justa do desempenho do novo método;

- estudar novas alternativas de *pricing*, pois as experiências ressaltam uma sensibilidade muito grande do PROJECT em relação a estas alternativas. Ficam como sugestões o *multiple pricing*, e o *devex* desenvolvido por Harris para o SIMPLEX (veja /HAR/);

- implementar o PROJECT utilizando a decomposição LU, no lugar da FPI, como sugere Gille (veja /GLO/);

- implementar o PROJECT utilizando memória auxiliar para armazenamento das matrizes e listas, a fim de tornar a capacidade do mesmo praticamente ilimitada;

- implementar as opções de análise de pós-otimalidade, já desenvolvidas por Coelho (veja /COE/);

- estudar novas formas de recuperação do gradiente projetado, que sejam mais eficientes que a utilizada atualmente;

- estudar a possibilidade de implantar o PROJECT, para a resolução de problemas de programação linear inteira e mista.

B I B L I O G R A F I A

/BGH/ BENICHOU, M.; GAUTHIER, J. M.; HENTGES, G.; RIBIERE, G.; "The Efficient Solution of Large-scale Linear Programming Problems - Some Algorithmic Techniques and Computational Results"; Mathematical Programming 13:280-322, (1977).

/COE/ COELHO, Antônio Sérgio; Uma Opção para Análise de Pós-Otimalidade no Algorítmo PROJECT; Teste de Mestrado UFSC; 1983.

/COK/ COOPER, L.; KENNINGTON, J.; "Nonextreme Point Solution Strategies for Linear Programs"; Naval Research Logistics Quaterly 3:447-461. (1979).

/CRH/ CROWDER, Harlan; HATTINGH, J. M.; "Patially Normalized Pivot Selection in Linear Programming"; Mathematical Programming Study 4:12-25. (1975).

/GLO/ GILLE, Philippe; LOUTE, Etienne; Updating the LU Gaussian Decomposition for Rank-one Corrections - Application to Linear Programming Basis Partitioning Techniques; Center for Operations Research & Econometric; Universite Catholique de Louvain; Belgium...; 1982.

/HAR/ HARRIS, Paula M. J.; "Pivot Selection Methods of the Devex LP Code"; Mathematical Programming Study 4:30-57. (1975).

/HEH/ HEHL, Maximilian Emil; Sistema de Programação FORTRAN IV G-H; São Paulo... 1972.

/HER/ HELLERMAN, Eli; RARICK, Dennis; "Reinversion with the Preassigned Pivot Procedure"; Mathematical Programming 1:195-216. (1971).

/KTZ/ KUNZI, H. P.; TZSCHACH, H. ; "The Duoplex-Algorithm"; Numerische Mathematik 7:222-225. (1965).

/KUN/ KUNZI, H. P.; "Die Duoplexmethode"; Untermehmensforschung 7:103-116. (1963)

/KUT/ KUNZI, H. P.; TAN, S. T.; "Lineare Optimierung Grosser Systeme"; Lecture Notes in Mathematics 4, Berlin, 1966.

/ORH/ ORCHARD-HAYS, William; Advanced Linear-programming Computing Techniques; New York...1968.

./PAR/ PARANJAPE, S. R.; "The Simplex Method: Two Basis Variables Replacement"; Management Science 12:135-141. (1965).

/PFA/ PFALTZ, John L.; Computer Data Structures; New York,1977.

/RB1/ "RODDER, W.; BLAUTH, M.; PROJECT - An Alternative LP-Algorithm; Artigo apresentado no 'International Congress on Mathematical Programming'; Rio de Janeiro; abril de 1981.

/RB2/ RODDER, W.; BLAUTH, M.; PROJECT - "An Alternative LP-Code"; Boletim de Produção e Sistemas - UFSC; vol. 2:24-35; Florianópolis. (1980).

/RCM/ RODDER, W.; COELHO, S.; MAYERLE, S.; A Forma Mista da Inversa para Troca Múltipla de Base em um LP-Algorítmo; Artigo apresentado no XV SBPO / I CLAPO; Rio de Janeiro; Novembro de 1982. pp 621-638.

/RO1/ RODDER, W.; "A Note on Linear Dependency in PROJECT"; Boletim de Produção e Sistemas - UFSC; Vol. 3:51-62; Florianópolis. (1981).

/RO2/ RODDER, W.; PROJECT and its Invertion-free Form; Submetido ao Operation Research Spektrum.

/RO3/ RODDER, W.; Finiteness of PROJECT under Perturbation; Artigo de circulação interna do Departamento de Engenharia de Produção e Sistemas; UFSC; Florianópolis. (1982)

/SCH/ SCHWARTZ, H. R.; et alii.; Numerical Analysis of Symmetric Matrices. London ..., 1973.

/TOM/ TOMLIN, J. A.; "On Scaling Linear Programming Problems"; Mathematical Programming Study 4:146-166; (1975).

APÊNDICES

APÊNDICE I - MANUAL DO USUÁRIO

A) NOÇÕES GERAIS

PROJECT é um sistema computacional para resolução de problemas de programação linear com estrutura matricial esparsa. Foi dimensionado para resolver problemas de médio e grande porte, sendo limitado em 1400 restrições e 4200 variáveis estruturais. Este limite foi fixado, dado que o sistema não utiliza memória auxiliar para o armazenamento dos dados. Com esta configuração, há necessidade de uma área de memória de aproximadamente 1500 kbytes.

A fim de obter uma economia em área de armazenamento e tempo de processamento, utiliza-se a forma produto da inversa, armazenando as matrizes elementares na forma de listas.

O método de resolução adotado efetua trocas múltiplas de base, levando em consideração o gradiente da função objetivo, para a obtenção da solução ótima.

A utilização do sistema pode ser feita via CMS, através de um comando de execução* , que efetua o processamento do arquivo de dados, editado conforme instruções contidas neste capítulo.

---

* Este comando de execução chama um procedimento que define os arquivos de entrada e saída, além de efetuar a carga do módulo do sistema.

B) ENTRADA DOS DADOS

A massa de dados para a alimentação do programa é dividida em sete blocos distintos, sendo cada um deles precedido de um registro de comando, conforme mostra o quadro I.1.

COMANDO	BLOCOS
OP	Opções
TC	Titulação e comentários
IP	Identificação do problema
FO	Função objetivo
RE	Restrições estruturais
BO	Limites especiais de variáveis
EX	Comando de execução

Quadro I.1 - Comandos para identificação dos blocos.

O formato destes registros é:

FORMAT (A2)

A fim de orientar o usuário quanto a edição do arquivo de dados é apresentada, a seguir, a descrição detalhada de cada bloco.

Bloco 1 - OPÇÕES (opcional)

Este bloco é composto de um registro que informa ao programa as opções desejadas. Sua formatação é a seguinte:

COM, OP1, OP2, OP3, TIME (REG. 1)

FORMAT (A2,(3(1X,I1),I5)

onde: COM = OP

OP1 = 0-não gera arquivo para conferência dos dados de entrada (*default*);

= 1-gera um arquivo para conferência dos dados de entrada;

OP2 = 0-não realiza a normalização dos dados (*default*);

= 1-realiza a normalização comum dos dados;

= 2-realiza a normalização geométrica dos dados;

OP3 = 0-não gera o arquivo TRACE com o resumo das operações efetuadas pelo sistema (*default*);

= 1-gera o arquivo TRACE a fim do usuário acompanhar as operações efetuadas pelo sistema;

TIME Tempo de CPU, em segundos, especificado como limite para obtenção da solução ótima; caso esta não seja encontrada com o tempo estipulado, a última solução encontrada é apresentada; se este bloco não for utilizado, o sistema assume que TIME = 10 segundos.

Bloco 2 - TITULAÇÃO E COMENTÁRIOS (opcional)

Este bloco é reservado para que sejam efetuados a titulação, os comentários, ou quaisquer outras anotações de interesse ao problema proposto. Sua formatação é a seguinte:

COM,(CAMPO(I),I=1,19) (REG. 2)

FORMAT(A2,2X,19A4)

onde: COM = TC

CAMPO(I) campo alfa-numérico com até 76 dígitos, destinado à titulação e aos comentários.

Este registro pode ser repetido.

Bloco 3 - IDENTIFICAÇÃO DO PROBLEMA

Por identificação do problema entende-se a definição do tipo de otimização desejada (maximização ou minimização), além da atribuição de um nome à função objetivo, e de um valor constante a ser somado na mesma. Este bloco possui um registro, cuja formação é a seguinte:

COM,OPTI,NFOB,ZO (REG. 3)

FORMAT(A2,1X,A4,2X,A8,2X,G12.5)

onde: COM = IP

OPTI = MAXI - para maximização da função objetivo;

= MINI - para minimização da função objetivo;

NFOB nome atribuido à função objetivo;

ZO valor constante a ser somado na função objetivo.

Bloco 4 - FUNÇÃO OBJETIVO

O bloco da função objetivo é composto de um registro que pode ser repetido tantas vêzes quanto se fizer necessário. A formatação é a seguinte:

COM,((NOME(I),COEF(I)),I=1,3) (REG. 4)

FORMAT(A2,3(2X,A8,1X,G12.5))

Onde: COM = FO

NOME(I) nome alfa-numérico de uma variável estrutural do problema;

COEF(I) coeficiente da variável NOME(I) na função objetivo.

Não há necessidade de se atribuir nomes e valores a todos os parâmetros do registro 4. Quando algum par (NOME, COEF) é omitido, o par seguinte é processado, até que seja identificado um novo bloco.

Bloco 5 - RESTRIÇÕES

Este bloco compõe-se de dois tipos de registros:

1) Registro de definição das restrições, formatado conforme segue:

COM,RNOME,REST,BI,BS (REG. 5.1)

FORMAT(A2,2X,A8,2X,A2,2X,2(G12.5,2X))

Onde: COM = RE

RNOME nome alfa-numérico atribuido à restrição;

REST = LE - para restrição com limite superior;

= GE - para restrição com limite inferior;

= EQ - para restrição de igualdade;

= RA - para restrição com limite superior e inferior;

BI coeficiente da mão direita (quando a restrição é do tipo 'RA' corresponde ao limite inferior;

BS limite superior da restrição (somente para o caso 'RA').

2) Registros de explicitação das restrições, que podem ser repetidos tantas vêzes quanto se fizerem necessários, e que são formatados conforme segue:

COM,((NOME(I),COEF(I)),I=1,3),RNOME (REG. 5.2)

FORMAT(A2,3(2X,A8,1X,G12.5),1X,A8)

onde: COM campo vazio;

NOME(I) nome alfa-numérico de uma variável estrutural do problema;

COEF(I) coeficiente da variável NOME(I) na restrição RNOME;

RNOME nome alfa-numérico atribuído à restrição a qual o registro está associado.

Assim, define-se uma restrição através de um registro de comando, um registro 5.1, e um grupo de registros 5.2 com o mesmo nome de referencia RNOME, agrupados nesta ordem. Finalmente, o bloco de restrições é obtido através da concatenação dos registros que definem as diversas restrições.

Bloco 6 - LIMITES ESPECIAIS (opcional)

Sete casos especiais de limites são considerados na entrada de dados, identificados através de um registro co

mando, e de um grupo de registros de explicitação das variáveis e seus respectivos limites. O quadro I.2 define os tipos de limites considerados no sistema, e os respectivos registros de explicitação a serem utilizados.

CASO	LIM. INFERIOR	LIM. SUPERIOR	REGISTRO
C1	zero	infinito (+)	6.2.1
C2	zero	LIM	6.2.2
C3	LIM	infinito (+)	6.2.2
C4	LIM1	LIM2	6.2.3
C5	infinito (-)	LIM	6.2.2
C6	infinito (-)	infinito (+)	6.2.1
C7	LIM	LIM	6.2.2

Quadro I.2 - Tipos de limites de variáveis tratadas de forma especial no sistema.

A formatação dos registros é dada abaixo:

1) Registro de identificação do caso a ser tratado:

CASO (REG. 6.1)

FORMAT (A2)

onde: CASO = C1, C2, C3, C4, C5, C6, ou C7, - conforme os limites a serem tratados. Quando não houver declaração explícita acerca dos limites de determinada variável, é assumido que se trata de uma variável do tipo C1.

2) Registros de explicitação das variáveis estruturais e seus respectivos limites.

```
COM,(NOME(I), I = 1,7),CASO                    (REG. 6.2.1)
COM,((NOME(I),LIM(I)),I=1,3),CASO              (REG. 6.2.2)
COM,((NOME(I),LIM1(I),LIM2(I),I=1,2),CASO      (REG. 6.2.3)
```

onde: COM campo vazio;

NOME(I) nome alfa-numérico de uma variável estrutural do problema a ser resolvido;

LIM(I) limite superior e/ou inferior da variável NOME(I);

LIM1(I) limite inferior da variável NOME(I), explicitada pelo registro 6.2.3;

LIM2(I) limite superior da variável NOME(I), explicitada pelo registro 6.2.3;

CASO tipo de limite tratado pelo registro.

A formatação destes registros segue, respectivamente,as declarações abaixo:

```
FORMAT(A2,7(2X,A8),6X,A2)
FORMAT(A2,3(2X,A8,1X,G12.5),7X,A2)
FORMAT(A2,2(2X,A8,2(1X,G12.5)),4X,A2)
```

A montagem deste bloco é obtida através da concatenação do registro de comando, que define o início do bloco, com o conjunto de registros que definem os diversos casos de limites tratados.

Para ilustrar a formatação dos blocos de entrada dos dados, veja o exemplo do apêndice II.

C) MENSAGENS DE ERROS E COMENTÁRIOS

A fim de detectar possíveis erros na entrada dos dados, o sistema PROJECT tem um conjunto de testes que geram mensagens identificando os problemas ocorridos, e que interrompem o processamento.

Em termos gerais, os testes efetuados detectam os seguintes erros:

- seguência incorreta dos blocos e registros;
- omissão de blocos não opcionais;
- tipo de otimização não compatível;
- dupla declaração de variável na função objetivo ou restrição;
- tipo de restrição não compatível;
- variável declarada no bloco de limites especiais, e não definida anteriormente;
- as dimensões do problema ultrapassam as dimensões previstas.

Além destes problemas, relativos à entrada dos dados, o PROJECT emite mensagem quando não existe área de memória suficiente para o armazenamento da matriz inversa (vetores-η).

D) RELATÓRIO DOS RESULTADOS

Dividiu-se o relatório dos resultados em três seções.

Na seção 1, são apresentados,além de estatísticas acerca das listas e do problema proposto, o valor e o tipo (ótima, inviável,ilimitada, ou problema cancelado por tempo de CPU) de solução encontrada.

Na seção 2, são apresentados os resultados relativos a cada uma das restrições do problema, conforme discriminação abaixo:

NUMBER — número sequencial associado a cada restrição;

ROWS — nome alfa-numérico atribuido pelo usuário a cada restrição do problema;

STATUS — = EQ - identifica uma restrição de igualdade;
= UL - indica que a restrição está em seu limite superior;
= LL - indica que a restrição está em seu limite inferior;
= BS - indica que a restrição está em um valor intermediário entre o seu limite inferior e superior;

ACTIVITY — valor assumido pela restrição;

SLACK ACTIVITY — valor da folga em relação ao limite superior, ou quando este não existir, representa o valor da folga em relação ao limite inferior;

LOWER LIMIT — limite inferior da restrição;

UPPER LIMIT — limite superior da restrição;

DUAL ACTIVITY — incremento relativo da função objetivo, com o acréscimo do limite ativo da restrição.

Na seção 3, são apresentados os resultados relativos a cada uma das variáveis estruturais do problema, conforme discriminação a baixo relacionada:

NUMBER — número sequencial associado a cada variável es

trutural do problema;

COLUMN — nome alfa-numérico atribuido pelo usuário a cada variável estrutural;

STATUS — = EQ - indica que a variável está fixada em um valor constante definido pelo usuário;
= UL - indica que a variável está fixada em seu limite superior;
= LL - indica que a variável está fixada em seu limite inferior;
= BS - indica que a variável assumiu um valor intermediário entre os respectivos limites inferior e superior;

ACTIVITY — valor da variável na solução;

INPUT COST — coeficiente da variável na função objetivo original;

LOWER LIMIT — limite inferior da variável;

UPPER LIMIT — limite superior da variável;

REDUCED COST — incremento relativo da funçãc objetivo, com o acréscimo do limite ativo da variável.

Como ilustração apresenta-se,no apêndice III, o relatório dos resultados do problema formulado no apêndice II.

E) ACOMPANHAMENTO DO PROCESSO DE OTIMIZAÇÃO

Ao usuário é dada a opção de acompanhar as operações efetuadas pelo sistema, através de um arquivo TRACE gerado durante o pro

cesso de otimização.

Neste arquivo constam indicações a respeito da FPI, das variáveis que são relaxadas, e das que são ativadas, servindo de subsídio para uma posterior análise do algorítmo.

No apêndice IV é apresentado, como exemplo, o arquivo TRACE gerado para o problema formulado no apêndice II.

APÊNDICE II - UM EXEMPLO DE ENTRADA DE DADOS

```
CP
CP 0 1 1 1000
IP
IP MINI  CUSTO
FO
FO  CEVADA      10000.      TRIGO-MO    30000.      MELACO      60.000
FO  FAR MAND    20000.      MANDIOCA    40000.      MILHO GR    19300.
FO  FAR MILH    250.00      MILHETO     50000.      AVEIA       80000.
FO  F ARROZ     130.00      F ARR DG    180.00      FO ARROZ    90000.
FO  GR SORGO    50000.      SEBO        70000.      TRIGO       10000.
FO  F TRIGO     130.00      F ALFAFA    80000.      F SANGUE    400.00
FO  F CCCC      70000.      F GLUTEN    90000.      F ALGOD     30000.
FO  F PENA      80000.      F PEIXE     10000.      F LINHAC    40000.
FO  F FIGADO    90000.      F CAR-CS    20000.      TANCAGEM    50000.
FO  F SOJA      256.00      F SOJA D    30000.      GR SOJA     60000.
FO  SO LEITE    200.00      F OSSO D    250.00      F OSSO A    250.00
FO  FOSF CAL    30000.      FOSF FLL    50000.      CAFE CAL    80000.
FO  CALC MOI    15.000      F ESTRA     60000.      SAL         60.000
FO  MIX         7293.0
FE
FE  ENERGIA   GE    3300.0
    CEVADA      3.3000      TRIGO-MO    3.0100      MELACO      2.6500    ENERGIA
    FAR MAND    3.4800      MANDIOCA    1.1700      MILHO GR    3.5000    ENERGIA
    FAR MILH    3.0400      MILHETO     3.2500      AVEIA       2.7400    ENERGIA
    F ARROZ     3.2600      F ARR DG    2.5000      FO ARROZ    3.8000    ENERGIA
    GR SORGO    3.3600      SEBO        7.9000      TRIGO       3.5000    ENERGIA
    F TRIGO     2.5200      F ALFAFA    1.9600      F SANGUE    2.6500    ENERGIA
    F CCCC      3.1800      F GLUTEN    3.5000      F ALGOD     2.7600    ENERGIA
    F PENA      2.7500      F PEIXE     3.2800      F LINHAC    2.8400    ENERGIA
    F FIGADO    3.2000      F CAR-CS    2.2000      TANCAGEM    2.4600    ENERGIA
    F SOJA      3.3700      F SOJA D    3.4600      GR SOJA     4.0000    ENERGIA
    SO LEITE    3.0700      F OSSO D    .70000      F OSSO A    .70000    ENERGIA
FE
FE  PROTEINA  GE    180.00
    CEVADA      .10200      TRIGO-MO    .11000      MELACO      .24000E-01 PROTEINA
    FAR MAND    .24000E-01  MANDIOCA    .10000E-01  MILHO GR    .88000E-01 PROTEINA
    FAR MILH    .73000E-01  MILHETO     .12200      AVEIA       .99000E-01 PROTEINA
    F ARROZ     .13400      F ARR DG    .17800      FO ARROZ    .12500     PROTEINA
    GR SORGO    .89000E-01  TRIGO       .12500      F TRIGO     .16200     PROTEINA
    F ALFAFA    .17400      F SANGUE    .74200      F CCCC      .26400     PROTEINA
    F GLUTEN    .40800      F ALGOD     .41600      F PENA      .84700     PROTEINA
    F PEIXE     .60500      F LINHAC    .34500      F FIGADO    .62000     PROTEINA
    F CAR-CS    .49600      TANCAGEM    .57100      F SOJA      .45000     PROTEINA
    F SOJA D    .49600      GR SOJA     .39600      SO LEITE    .13600     PROTEINA
    F OSSO D    .23200      F OSSO A    .20300                             PROTEINA
FE
FE  LISINA    GE    8.5000
    CEVADA      .40000E-02  TRIGO-MO    .62000E-02  MELACO      .10000E-02 LISINA
    FAR MAND    .80000E-03  MANDIOCA    .40000E-03  MILHO GR    .25000E-02 LISINA
    FAR MILH    .18000E-02  MILHETO     .40000E-02  AVEIA       .37000E-02 LISINA
    F ARROZ     .59000E-02  F ARR DG    .75000E-02  FO ARROZ    .55000E-02 LISINA
    GR SORGO    .24000E-02  TRIGO       .36000E-02  F TRIGO     .59000E-02 LISINA
    F ALFAFA    .93000E-02  F SANGUE    .19000E-01  F CCCC      .66000E-02 LISINA
    F GLUTEN    .83000E-02  F ALGOD     .16300E-01  F PENA      .16700E-01 LISINA
    F PEIXE     .48300E-01  F LINHAC    .12100E-01  F FIGADO    .48100E-01 LISINA
```

```
     F CAR-OS    .26000E-01  TANCAGEM    .38000E-01  F SOJA      .284CCE-01 LISINA
     F SOJA D    .31900E-01  GR SOJA     .22000E-01  SO LEITE    .97000E-02 LISINA
     F OSSO D    .20000E-01  F OSSO A    .20000E-01                         LISINA
RE
RE   MET-CIST   GE    5.0000
     CEVADA      .40000E-02  TRIGO-MO    .40000E-02  MELACO      .10000E-02 MET-CIST
     FAR MAND    .50000E-03  MANDIOCA    .30000E-03  MILHO GR    .28000E-02 MET-CIST
     FAR MILH    .28000E-02  MILHETO    .35000E-02  AVEIA       .34000E-02 MET-CIST
     F ARROZ     .41000E-02  F ARR DO    .50000E-02  PO ARROZ    .41000E-02 MET-CIST
     GR SORGO    .27000E-02  TRIGO       .38000E-02  F TRIGO     .42000E-02 MET-CIST
     F ALFAFA    .56000E-02  F SANGUE    .11000E-01  F COCO      .58000E-02 MET-CIST
     F GLUTEN    .15200E-01  F ALGOD     .15000E-01  F PENA      .10200E-01 MET-CIST
     F PEIXE     .24100E-01  F LINHAC    .17100E-01  F FIGADO    .21300E-01 MET-CIST
     F CAR-OS    .90000E-02  TANCAGEM    .12000E-01  F SOJA      .13000E-01 MET-CIST
     F SOJA D    .15200E-01  GR SOJA     .11000E-01  SO LEITE    .80000E-02 MET-CIST
     F OSSO D    .50000E-02  F OSSO A    .46000E-02                         MET-CIST
RE
RE   TRIPTOFA   GE    1.5000
     CEVADA      .11000E-02  TRIGO-MO    .19000E-02  MELACO      .50000E-03 TRIPTOFA
     FAR MAND    .20000E-03  MANDIOCA    .10000E-03  MILHO GR    .70000E-02 TRIPTOFA
     FAR MILH    .70000E-03  MILHETO    .15000E-02  AVEIA       .14000E-02 TRIPTOFA
     F ARROZ     .15000E-02  F ARR DO    .18000E-02  PO ARROZ    .14000E-02 TRIPTOFA
     GR SORGO    .10000E-02  TRIGO       .15000E-02  F TRIGO     .25000E-02 TRIPTOFA
     F ALFAFA    .23000E-02  F SANGUE    .38000E-02  F COCO      .20000E-02 TRIPTOFA
     F GLUTEN    .20000E-02  F ALGOD     .55000E-02  F PENA      .50000E-02 TRIPTOFA
     F PEIXE     .68000E-02  F LINHAC    .56000E-02  F FIGADO    .59000E-02 TRIPTOFA
     F CAR-OS    .20000E-02  TANCAGEM    .54000E-02  F SOJA      .63000E-02 TRIPTOFA
     F SOJA D    .67000E-02  GR SOJA     .49000E-02  SO LEITE    .18000E-02 TRIPTOFA
     F OSSO D    .10000E-02  F OSSO A    .11000E-02                         TRIPTOFA
RE
RE   CALCIO     RA    6.5000        8.5000
     CEVADA      .80000E-03  TRIGO-MO    .80000E-03  MELACO      .65000E-02 CALCIO
     FAR MAND    .15000E-02  MANDIOCA    .10000E-02  MILHO GR    .10000E-03 CALCIO
     FAR MILH    .90000E-03  MILHETO    .50000E-03  AVEIA       .10000E-02 CALCIO
     F ARROZ     .80000E-03  F ARR DO    .13000E-02  PO ARROZ    .20000E-02 CALCIO
     GR SORGO    .30000E-03  TRIGO       .11000E-02  F TRIGO     .24000E-02 CALCIO
     F ALFAFA    .14400E-01  F SANGUE    .55000E-02  F COCO      .15000E-02 CALCIO
     F GLUTEN    .29000E-02  F ALGOD     .19000E-02  F PENA      .25000E-02 CALCIO
     F PEIXE     .51000E-01  F LINHAC    .39000E-02  F FIGADO    .30000E-02 CALCIO
     F CAR-OS    .80100E-01  TANCAGEM    .61100E-01  F SOJA      .30000E-02 CALCIO
     F SOJA D    .27000E-02  GR SOJA     .27000E-02  SO LEITE    .97000E-02 CALCIO
     F OSSO D    .24500      F OSSO A    .25000      FOSF CAL    .23000     CALCIO
     FOSF FLU    .32000      CARB CAL    .38000      CALC MOI    .36000     CALCIO
     F ESTRA     .33000                                                     CALCIO
RE
RE   FOSFORO    GE    6.0000
     CEVADA      .35000E-02  TRIGO-MO    .32000E-02  MELACO      .17000E-02 FOSFORO
     FAR MAND    .80000E-03  MANDIOCA    .60000E-03  MILHO GR    .25000E-02 FOSFORO
     FAR MILH    .23000E-02  MILHETO    .28000E-02  AVEIA       .27000E-02 FOSFORO
     F ARROZ     .15000E-01  F ARR DO    .11000E-01  PO ARROZ    .56000E-02 FOSFORO
     GR SORGO    .29000E-02  TRIGO       .43000E-02  F TRIGO     .11200E-01 FOSFORO
     F ALFAFA    .22000E-02  F SANGUE    .32000E-02  F COCO      .45000E-02 FOSFORO
     F GLUTEN    .67000E-02  F ALGOD     .11100E-01  F PENA      .50000E-02 FOSFORO
     F PEIXE     .29000E-01  F LINHAC    .81000E-02  F FIGADO    .10400E-01 FOSFORO
     F CAR-OS    .41800E-01  TANCAGEM    .31700E-01  F SOJA      .65000E-02 FOSFORO
```

```
     F SOJA D    .62000E-02  GR SOJA     .54000E-02  SC LEITE    .76000E-02 FOSFORL
     F OSSO D    .10700      F OSSO A    .11200      FOSF CAL    .18400     FOSFORC
     FOSF FLU    .18000                                                     FOSFORC
RE
FE   REL CA-P    OE     .0
     CEVADA     -.27000E-02  TRIGO-MO   -.24000E-02  MELACO      .48000E-02 REL CA-P
     FAR MAND    .70000E-03  MANDIOCA    .40000E-03  MILHO GR   -.24000E-02 REL CA-P
     FAR MILH   -.14000E-02  MILHETO     -.23000E-02  AVEIA      -.17000E-02 REL CA-P
     F ARROZ    -.14200E-01  F ARR DG    -.97000E-02  PO ARROZ   -.36000E-02 REL CA-P
     GR SORGO   -.26000E-02  TRIGO       -.32000E-02  F TRIGO    -.88000E-02 REL CA-P
     F ALFAFA    .12200E-01  F SANGUE     .22000E-02  F COCO     -.30000E-02 REL CA-P
     F GLUTEN   -.39000E-02  F ALGOD     -.92000E-02  F PENA     -.25000E-02 REL CA-P
     F PEIXE     .22000E-01  F LINHAC    -.42000E-02  F FIGADO   -.74000E-02 REL CA-P
     F CAR-OS    .38300E-01  TANCAGEM     .29400E-01  F SOJA     -.38000E-02 REL CA-P
     F SOJA D   -.35000E-02  GR SOJA     -.27000E-02  SC LEITE    .21000E-02 REL CA-P
     F OSSO D    .13800      F OSSO A     .13800      FOSF CAL    .52000E-01 REL CA-P
     FOSF FLU    .14000      CARB CAL     .38000      CALC MOI    .36800     REL CA-P
     F ESTRA     .23000                                                     REL CA-P
RE
RE   SAL         EQ    2.5000
     SAL          1.0000                                                    SAL
RE
RE   MIX         EQ    5.0000
     MIX          1.0000                                                    MIX
FE
RE   PESO        EQ    1000.0
     CEVADA       1.0000      TRIGO-MO     1.0000      MELACO      1.0000   PESO
     FAR MAND     1.0000      MANDIOCA     1.0000      MILHO GR    1.0000   PESO
     FAR MILH     1.0000      MILHETO      1.0000      AVEIA       1.0000   PESO
     F ARROZ      1.0000      F ARR DG     1.0000      PO ARROZ    1.0000   PESO
     GR SORGO     1.0000      SEBO         1.0000      TRIGO       1.0000   PESO
     F TRIGO      1.0000      F ALFAFA     1.0000      F SANGUE    1.0000   PESO
     F COCO       1.0000      F GLUTEN     1.0000      F ALGOD     1.0000   PESO
     F PENA       1.0000      F PEIXE      1.0000      F LINHAC    1.0000   PESO
     F FIGADO     1.0000      F CAR-OS     1.0000      TANCAGEM    1.0000   PESO
     F SOJA       1.0000      F SOJA D     1.0000      GR SOJA     1.0000   PESO
     SC LEITE     1.0000      F OSSO D     1.0000      F OSSO A    1.0000   PESO
     FOSF CAL     1.0000      FOSF FLU     1.0000      CARB CAL    1.0000   PESO
     CALC MOI     1.0000      F ESTRA      1.0000      SAL         1.0000   PESO
     MIX          1.0000                                                    PESO
EO
C2
     TRIGO-MO   300.00        MELACO     300.00       FAR MAND   500.00          C2
     MANDIOCA   500.00        AVEIA      300.00       F ARROZ    400.00          C2
     F ARR DG   400.00        PO ARROZ   400.00       SEBO       50.000          C2
     F TRIGO    400.00        F ALFAFA    50.000      F SANGUE    30.000          C2
     F COCO     200.00        F GLUTEN   400.00       F ALGOD     80.000          C2
     F PENA      70.000       F PEIXE     70.000      F LINHAC    70.000          C2
     F FIGADO    60.000       F CAR-OS    60.000      TANCAGEM    60.000          C2
     SC LEITE   200.00                                                          C2
EX
     RACOES PARA SUINOS - TIPO INICIAL (10 A 20 KG DE PESO VIVO)
```

APÊNDICE III - UM EXEMPLO DE SAÍDA DOS RESULTADOS

SECTION 1 *** OPTIMAL SOLUTION AND PROBLEM STATISTICS

OPTIMAL SOLUTION

NAME	CUSTC
STATUS	OPTIMAL
VALUE	0.27993256D+06

STATISTIC OF LP-PROBLEM

STRUCTURAL VARIABLES	40
SLACK VARIABLES	8
ARTIFICIAL VARIABLES	11
ROWS	11

STATISTIC OF NON-ZERO ELEMENTS

NUMBER	311
DENSITY (ORIGINAL)	70.68

STATISTIC OF ETA-ELEMENTS

MEAN	36
MAX	61
DENSITY (MEAN)	25.26

STATISTIC OF Y-FILE

LENGHT (MEAN)	30.00

NUMBER OF ITERATIONS	9
NUMBER OF REINVERSIONS	8

SECTION 2 *** ROWS

NUMBER	..ROWS..	STATUS	...ACTIVITY...	SLACK ACTIVITY	.LOWER LIMIT.	.UPPER LIMIT.	DUAL ACTIVITY
1	ENERGIA	LL	0.33CCCCCCD+04	C.C	0.33CCCCCCE+04	NONE	-.642C3357D+C3
2	PROTEINA	BS	0.42142391D+03	C.24142391E+C3	C.18CCCCCCE+C3	NONE	C.C
3	LISINA	BS	C.2643369CD+C2	C.1793335CE+C2	C.85CCCCCCE+C1	NONE	C.C
4	MET-CIST	BS	C.12189C77D+C2	C.7169C76EE+C1	C.5CCCCCCCE+C1	NONE	C.C
5	TRIPTOFA	BS	0.58759387D+C1	C.43759297D+C1	C.15CCCCCCD+C1	NONE	C.C
6	CALCIO	BS	0.692C97C8D+C1	C.1579C292E+C1	C.65CCCCCCE+C1	C.85CCCCCCE+C1	C.C
7	FOSFORO	BS	0.692C9656D+C1	C.92C9655TE+CC	C.6CCCCCCCE+C1	NONE	C.C
8	REL CA-P	LL	C.C	C.C	0.C	NONE	-.51753787E+C4
9	SAL	EQ	0.25CCCCCCE+C1	C.C	C.25CCCCCCE+C1	0.25CCCCCCD+C1	-.193953921+C4
10	MIX	EQ	0.5CCCCCCCD+C1	C.C	C.5CCCCCCCE+C1	C.5CCCCCCCD+C1	-.5182535JE+C4
11	PESC	EQ	C.1CCCCCCCD+C4	C.C	C.1CCCCCCCE+C4	C.1CCCCCCCD+C4	C.1869539ZE+C4

SECTION 3 *** COLUMNS

NUMBER	.COLUMN.	STATUS	...ACTIVITY...	..INPUT COST..	.LOWER LIMIT.	.UPPER LIMIT.	.REDUCED COST.
1	CEVADA	LL	C.C	C.99999994D+C4	C.C	NONE	C.97E4E015D+C4
2	TRICO-MO	LL	C.C	C.29999998D+C5	C.C	C.3CCCCCCCD+C3	C.25969427D+C5
3	MELACO	LL	C.C	C.79999988D+C2	C.C	C.3CCCCCCCD+C3	C.2433C89CD+C3
4	FAR MAND	LL	C.C	C.19999999D+C5	C.C	C.5ECCCCCCD+C3	C.19651639D+C5
5	MANDIUCA	LL	C.C	C.39999998D+C5	C.C	C.5CCCCCCCD+C3	C.41136287D+C5
6	MILHO GR	LL	C.C	C.19299996D+C5	C.C	NONE	C.18994839D+C5
7	FAR MILH	LL	C.C	C.24999999D+C3	C.C	NONE	C.19CCC29ED+C3
8	MILHETO	LL	C.C	C.49999997D+C5	C.C	NONE	C.49E14831D+C5
9	AVEIA	LL	C.C	C.79999995D+C5	C.C	C.3CCCCCCCD+C3	C.8C1391C1D+C5
10	F ARROZ	BS	C.63816911D+C2	C.12999999E+C3	C.C	C.4CCCCCCCD+C3	C.C
11	F ARR DG	LL	C.C	C.17999999E+C3	C.C	C.4CCCCCCCD+C3	C.9146567CD+C3
12	PC ARROZ	LL	C.C	C.9CCCCCCCE+C5	C.C	C.4CCCCCCCD+C3	C.89468443D+C5
13	GR SORGO	LL	C.C	C.49999997D+C5	C.C	NONE	C.497457CCD+C5
14	SEBO	LL	C.C	C.69999996D+C5	C.C	C.5CCCCCCCD+C2	C.6681747CD+C5
15	TRIGO	LL	C.C	C.99999994D+C4	C.C	NONE	C.96589824D+C4
16	F TRIGO	LL	C.C	C.12999999E+C3	C.C	C.4CCCCCCCD+C3	C.447157EEE+C3
17	F ALFAFA	LL	C.C	C.79999995D+C5	C.C	C.5CCCCCCCD+C2	C.8C568CC9E+C5
18	F SANGUE	LL	C.C	C.39999984D+C3	C.C	C.3CCCCCCCD+C2	C.57676474D+C3
19	F CCCC	LL	C.C	C.69999996D+C5	C.C	C.2CCCCCCCD+C3	C.69E633945+C5
20	F GLUTEN	LL	C.C	C.9CCCCCCCE+C5	C.C	C.4CCCCCCCD+C3	C.89662C88D+C5
21	F ALGOD	LL	C.C	C.29999998D+C5	C.C	C.8CCCCCCCD+C2	C.3C1651C6D+C5
22	F PENA	LL	C.C	C.79999995D+C5	C.C	C.3CCCCCCCD+C2	C.8C13E881D+C5
23	F PEIXE	LL	C.C	C.99999994D+C4	C.C	C.7CCCCCCCD+C2	C.9669E1C6D+C4
24	F LINHAC	LL	C.C	C.39999995E+C5	C.C	C.7CCCCCCCD+C2	C.4CCE7E9ED+C5
25	F FIGADO	LL	C.C	C.9CCCC000D+C5	C.C	C.6CCCCCCCD+C2	C.8587333CD+C5
26	F CAR-OS	LL	C.C	C.19999999D+C5	C.C	C.ECCCCCCCD+C2	C.2C27E64EE+C5
27	TANCAGEM	LL	C.C	C.49999997D+C5	C.C	C.6CCCCCCCD+C2	C.5015797EE+C5
28	F SOJA	BS	C.91749442D+C3	C.25599998D+C3	C.C	NONE	C.C
29	F SOJA D	LL	C.C	C.29999998E+C5	C.C	NONE	C.2968621CD+C5
30	CR SOJA	LL	C.C	C.59999996E+C5	C.C	NONE	C.9933537CE+C5
31	SO LEITE	LL	C.C	C.19999998D+C3	C.C	C.2CCCCCCCD+C3	C.10762811D+C3
32	F OSSO D	LL	C.C	C.24999999E+C3	C.C	NONE	C.9759136CD+C3
33	F OSSO A	LL	C.C	C.24999999E+C3	C.C	NONE	C.9759136CD+C3
34	FOSF CAL	LL	C.C	C.29999999E+C5	C.C	NONE	C.3162C41EE+C5
35	FOSF FLU	LL	C.C	C.49999997D+C5	C.C	NONE	C.5116499CD+C5
36	CARB CAL	LL	C.C	C.79999995D+C5	C.C	NONE	C.7992289CD+C5
37	CALC MOI	BS	C.11188668D+C2	C.15CCCCCCD+C2	C.C	NONE	C.C
38	F OSTRA	LL	C.C	C.59999996E+C5	C.C	NONE	C.6018166EE+C5
39	SAL	BS	C.25CCCCCCD+C1	C.49999998D+C2	C.C	NONE	C.C
40	MIX	BS	C.5CCCCCCCD+C1	C.72929960D+C4	C.C	NONE	C.C

APÊNDICE IV - UM EXEMPLO DE ACOMPANHAMENTO DOS CÁLCULOS

```
REINVERSION AFTER ITERATION NUMBER .....          0
    PHASE ..............................          1
    SOLUTION ..........................-0.22968D+04
    GEN. BASE .........................         12
    ETA VECTORS .......................          0
    ETA ELEMENTS ......................          0


    RELAXED VARIABLES   NUMBER    INDEX    GOAL  ACTIVITY   STATUS     FACTOR Y
                             1       24     0.54752D+01        3      -0.27972D-01
                             2       36     0.31350D+00        3      -0.24240D+00
                             3       37     0.10611D+00        3      -0.36229D+00
                             4       26     0.84167D+01        3      -0.54669D+00
                             5       40     0.55508D-01        3      -0.38389D+00
                             6       34     0.46932D-01        3      -0.74301D+00
                             7       47     0.40524D-01        3      -0.59605D+00
                             8       41     0.43628D-01        3      -0.41103D+00
                             9       28     0.32885D-01        3      -0.74781D+00
                            10       33     0.27058D-01        3      -0.81409D+00
                            11       30     0.16641D-01        3      -0.74456D+00
                            12       38     0.11301D-01        3      -0.66014D+00
                            13       23     0.73945D-02        3      -0.65583D+00
                            14       29     0.37638D-02        3      -0.82903D+00
                            15       27     0.20282D-02        3      -0.86374D+00

    GRADIENT   0.87786D+00
    TOLERANCE  0.87786E-08
    ACTIVED VARIABLES   NUMBER    INDEX       GRADIENT        STATUS     FACTOR Y       SOLUTION
                             1       60     0.87307D+00        3      0.24728D+01   -0.22968D+04
                             2       37     0.87270D+00        3      0.19356D+01   -0.22968D+04
                             3       38     0.86625D+00        3      0.79941D+01   -0.22968D+04
                             4       34     0.86605D+00        3      0.14989D+01   -0.22968D+04
                             5       33     0.86527D+00        3      0.20414D+01   -0.22968D+04
                             6       36     0.86492D+00        3      0.99195D+01   -0.22968D+04
                             7       27     0.86468D+00        3      0.22193D+01   -0.22968D+04
                             8       28     0.86464D+00        3      0.27235D+01   -0.22968D+04
                             9       61     0.86207D+00        3      0.25680D+01   -0.22284D+04
                            10       47     0.85259D+00        2      0.21370D+01   -0.22272D+04
                            11       62     0.84736D+00        3      0.27991D+01   -0.21791D+04
                            12       26     0.84096D+00        2      0.15932D+01   -0.13917D+04
                            13       23     0.83712D+00        2      0.17238D+01   -0.13501D+04
                            14       55     0.16183D-01        3      0.48805D+02   -0.12042D+04
                            15       30     0.64196D-12        3      0.22570D+01   -0.11936D+04


REINVERSION AFTER ITERATION NUMBER .....          1
    PHASE ..............................          1
    SOLUTION ..........................-0.11936D+04
    GEN. BASE .........................         12
    ETA VECTORS .......................          4
```

ETA ELEMENTS 26

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y
	1	37	0.22052D+02	3	-0.71097D-03
	2	44	0.27182D+01	3	-0.59233D-01
	3	35	0.31318D+00	3	-0.30995D+00
	4	49	0.10564D+00	3	-0.28579D+00
	5	36	0.98617D-01	3	-0.59612D+00
	6	21	0.76809D-01	3	-0.64208D+00
	7	22	0.54093D-01	3	-0.75310D+00
	8	19	0.41728D-01	3	-0.71420D+00
	9	25	0.30955D-01	3	-0.80207D+00
	10	30	0.23650D-01	3	-0.80803D+00
	11	34	0.18753D-01	3	-0.79743D+00
	12	15	0.15424D-01	3	-0.48071D+00
	13	33	0.14588D-01	3	-0.83346D+00
	14	31	0.89657D-02	3	-0.89592D+00
	15	28	0.48829D-02	3	-0.82759D+00

GRADIENT 0.83267D+00
TOLERANCE 0.83267E-08

ACTIVED VARIABLES	NUMBER	INDEX	GRADIENT	STATUS	FACTOR Y	SOLUTION
	1	37	0.83235D+00	3	0.23201D+01	-0.11936D+04
	2	57	0.78649D+00	3	0.29627D+01	-0.11712D+04
	3	31	0.78624D+00	3	0.11332D+01	-0.11606D+04
	4	25	0.78457D+00	3	0.13010D+01	-0.11592D+04
	5	28	0.78440D+00	3	0.12208D+01	-0.11592D+04
	6	30	0.78225D+00	3	0.14019D+01	-0.11585D+04
	7	22	0.78092D+00	3	0.14170D+01	-0.11452D+04
	8	58	0.69577D+00	3	0.52895D+01	-0.11215D+04
	9	34	0.69506D+00	3	0.13564D+01	-0.10437D+04
	10	36	0.69320D+00	3	0.19853D+01	-0.10004D+04
	11	33	0.68985D+00	3	0.23997D+01	-0.98196D+03
	12	19	0.67692D+00	2	0.15072D+01	-0.91758D+03
	13	15	0.61751D+00	2	0.28478D+01	-0.87596D+03
	14	56	0.37431D+00	3	0.10858D+02	-0.86814D+03
	15	44	-0.38791D-12	3	0.39787D+02	-0.86043D+03

REINVERSION AFTER ITERATION NUMBER 2
PASE 1
SOLUTION .....-0.86043D+03
GEN. BASE 12
ETA VECTORS 6
ETA ELEMENTS 41

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y
	1	47	-0.32203D+01	2	-0.52091D-01
	2	45	0.88189D+00	3	-0.10086D+00
	3	46	0.58590D+00	3	-0.65153D-01
	4	44	0.30669D+00	3	-0.21698D+00
	5	27	0.13097D+00	3	-0.72874D+00
	6	13	0.10129D+00	3	-0.70983D+00
	7	36	0.77175D-01	3	-0.54555D+00
	8	20	0.70262D-01	3	-0.81188D+00
	9	11	0.49969D-01	3	-0.82722D+00
	10	7	0.39335D-01	3	-0.81802D+00
	11	16	0.31048D-01	3	-0.86269D+00
	12	28	0.26263D-01	3	-0.76707D+00
	13	5	0.23066D-01	3	-0.83646D+00
	14	14	0.19053D-01	3	-0.88766D+00
	15	31	0.17350D-01	2	-0.86959D+00

GRADIENT 0.69418D+00
TOLERANCE 0.69418E-08

ACTIVED VARIABLES	NUMBER	INDEX	GRADIENT	STATUS	FACTOR Y	SOLUTION
	1	59	0.65695D+00	3	0.27069D+01	-0.85756D+03
	2	36	0.65670D+00	3	0.20174D+01	-0.84906D+03
	3	54	0.59364D+00	3	0.55695D+01	-0.81516D+03
	4	47	0.56805D+00	2	0.16756D+03	-0.80222D+03
	5	28	0.56796D+00	3	0.15024D+01	-0.77798D+03
	6	44	0.56589D+00	3	0.77817D+01	-0.68628D+03
	7	45	0.56415D+00	3	0.11427D+02	-0.56049D+03
	8	11	0.56383D+00	3	0.17683D+01	-0.54459D+03
	9	20	0.56348D+00	3	0.13484D+01	-0.40931D+03
	10	29	0.54487D+00	3	0.30116D+01	-0.17702D+02
	11	31	0.48732D+00	3	0.54934D+01	-0.15465D+03
	12	53	0.60651D-02	3	0.22259D+02	-0.78810D+02
	13	13	0.11937D-02	3	0.21061D+01	-0.68162D+02
	14	16	0.56442D-03	3	0.22859D+01	-0.59285D+02
	15	7	-0.14454D-13	2	0.17608D+01	-0.56291D+02

REINVERSION AFTER ITERATION NUMBER	3
PASE	1
SOLUTION .....-0.56291D+02	
GEN. BASE	12
ETA VECTORS	8
ETA ELEMENTS	61

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y
	1	37	0.33638D+01	3	-0.58190E-04
	2	42	0.22834D+01	3	-0.67096D-01
	3	43	0.54789D+00	3	-0.57073D-01

4	45	0.30747D+CC	3	-C.14748D+00
5	47	-0.23514D+CC	2	-C.20863D+CC
6	6	0.18149D+CC	3	-C.74869D+CC
7	48	0.12145D+CC	3	-C.41828D+00
8	4	0.99974D-C1	3	-C.82489D+CC
9	10	C.74470D-C1	3	-C.85936D+CC
10	8	0.63517D-C1	3	-C.86940D+CC
11	16	0.56234D-C1	3	-C.87578D+CC
12	12	0.47430D-C1	3	-C.90947D+CC
13	17	C.40782D-C1	3	-C.89172D+CC
14	7	0.35500D-C1	3	-C.89770D+00
15	39	0.33738D-C1	3	-C.66543D+CC

GRADIENT 0.45305D+00
TOLERANCE 0.45305E-08

ACTIVED VARIABLES	NUMBER	INDEX	GRADIENT	STATUS	FACTOR Y	SOLUTION
	1	47	0.44689D+CC	3	C.25792D+C1	-C.75004D+C1
	2	63	0.11348D-1C	3	C.22377D+01	-C.29366D-02

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y

REINVERSION AFTER ITERATION NUMBER 4
PASE 2
SOLUTION .....-0.48926D+03
GEN. BASE 12
ETA VECTORS 5
ETA ELEMENTS 23

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y
	1	24	0.36603D+02	1	-C.44722D-05
	2	21	-0.10061D+C1	1	-C.31944D+00
	3	26	-0.59872D+CC	2	-C.38836D+CC
	4	38	0.40375D+CC	3	-C.54356D+CC
	5	29	0.23198D+CC	3	-C.65761D+CC
	6	10	-0.21854D+CC	1	-C.61737D+CC
	7	8	0.20510D+CC	1	-C.67492D+CC
	8	18	-0.17242D+CC	1	-C.77613D+CC
	9	32	0.15904D+CC	3	-C.80751D+00
	10	33	0.14070D+CC	3	-C.57618D+CC
	11	15	-0.12688D+CC	2	-C.57773D+CC
	12	11	0.12853D+CC	3	-C.83343D+CC
	13	12	0.10696D+CC	1	-C.85583D+00
	14	23	-0.11048D+CC	2	-C.61548D+CC
	15	17	0.96609D-01	1	-C.86180D+CC

GRADIENT C.75384D+CC
TOLERANCE C.75384E-08

ACTIVED VARIABLES	NUMBER	INDEX	GRADIENT	STATUS	FACTOR Y	SOLUTION
	1	37	0.73572D+00	3	C.16271D+01	-C.4817CD+03
	2	39	0.71333D+00	3	C.17711D+01	-C.46925D+03
	3	18	0.68104D+00	3	C.12402D+01	-C.46473D+03
	4	10	0.63065D+00	3	C.13178D+01	-C.4643CD+03
	5	43	0.55577D+00	3	C.38461D+01	-C.46261D+03
	6	45	0.54327D+00	3	C.71183D+01	-C.45466D+03
	7	21	0.47367D+00	3	C.57367D+01	-C.44C14D+03
	8	38	0.41429D+00	3	C.7441CD+01	-C.43527D+03
	9	42	0.30444D+00	3	C.4992CD+01	-C.41876D+03
	10	32	0.27490D+00	3	C.19774D+01	-C.41454D+02
	11	26	0.24009D-01	3	C.51793D+01	-C.37579D+02
	12	4	0.74792D-02	3	C.14770D+01	-C.37359D+03
	13	12	0.43354D-02	3	C.13492D+01	-C.37168D+03
	14	17	0.30622D-03	3	C.15469D+01	-C.37166D+03
	15	8	-0.92628D-12	3	C.27879D+01	-C.37125D+03

REINVERSION AFTER ITERATION NUMBER	5
FASE	2
SOLUTION .....-0.37125D+03	
GEN. BASE	12
ETA VECTORS	8
ETA ELEMENTS	54

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y
	1	43	0.23886D+02	3	-C.46939D-04
	2	24	-0.30721D+01	1	-C.41461D-02
	3	45	0.46286D+01	3	-C.32080D-02
	4	23	-0.62774D+00	1	-C.41921D+00
	5	44	0.23134D+00	3	-C.2194CD+00
	6	38	0.24219D+00	3	-C.15189D+00
	7	29	0.23479D+00	1	-C.69108D+00
	8	8	0.19498D+00	3	-C.75717D+00
	9	14	-0.14912D+00	1	-C.84618D+00
	10	4	0.15681D+00	3	-C.82353D+00
	11	32	0.13067D+00	3	-C.87579D+00
	12	17	0.11355D+00	3	-C.8674CD+00
	13	12	0.99030D-01	3	-C.88842D+00
	14	33	0.85567D-01	1	-C.69410D+00
	15	16	0.70179D-01	3	-C.91639D+00

GRADIENT 0.51185D+00
TOLERANCE 0.51185E-08

ACTIVED VARIABLES	NUMBER	INDEX	GRADIENT	STATUS	FACTOR Y	SOLUTION
	1	43	0.45948D+00	3	C.32506D+01	-C.37125D+03
	2	6	0.36936D+00	3	C.16498D+01	-C.35270D+03
	3	35	0.30462D+00	3	C.22768D+01	-C.34351D+03

4	24	0.29CCCD+CC	2	C.20618D+C1	-C.33395D+C3
5	38	0.28709D+CC	3	C.45131D+C1	-C.32855D+C3
6	23	0.17523D+CC	3	C.76196D+C1	-C.32704D+C3
7	7	0.17379D+CC	3	C.11098D+C1	-C.31721D+C3
8	12	0.17373D+CC	3	C.11795D+C1	-C.15605D+C3
9	49	0.16944D+CC	3	C.85826D+C3	-C.90361D+02
10	14	0.18C93D-C1	3	C.13535D+01	-C.78795D+02
11	17	0.12232D-C1	3	C.14470D+01	-C.77056D+02
12	15	0.4774CD-C2	2	C.59320D+C2	-C.63652D+C2
13	5	0.16748D-C2	3	C.27591D+C1	-C.59208D+02
14	32	0.99C14D-C3	2	C.12415D+01	-C.58552D+02
15	16	-C.1CC78D-11	3	C.11162D+C2	-C.53261D+C2

REINVERSION AFTER ITERATION NUMBER 6
BASE 2
SOLUTION ..... -0.53261D+C2
GEN. BASE 12
ETA VECTORS 7
ETA ELEMENTS 56

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y
	1	38	0.14793D+C1	3	-C.34226D-C3
	2	49	0.14313D+C1	3	-C.25752D-C2
	3	15	-0.17529D+01	2	-C.1C320D-01
	4	47	0.51826D+CC	3	-C.20136D+CC
	5	33	-0.59566D-C1	1	-C.6CC73D+CC
	6	16	0.26258D-C1	3	-C.84655D+00
	7	12	0.14767D-C1	3	-C.84458D+00
	8	17	0.42745D-C2	3	-C.85172D+CC

GRADIENT 0.94735D-01
TOLERANCE C.94735E-C9

ACTIVED VARIABLES	NUMBER	INDEX	GRADIENT	STATUS	FACTOR Y	SOLUTION
	1	38	0.80113D-C1	3	C.30543D+C1	-0.53261D+C2
	2	49	0.42441D-C1	3	C.16910D+02	-0.53261D+02
	3	12	0.4037CD-C1	3	C.12363D+01	-C.53261D+02
	4	16	0.40322D-C1	3	C.11735C+C1	-C.53261D+C2
	5	17	0.35953D-C1	3	C.13668D+C1	-0.53261D+C2
	6	47	0.358C7D-C1	3	C.38669D+03	-0.53261D+02
	7	27	0.48617D-C3	3	C.25430D+01	-C.47598D+02
	8	4	-0.72885D-13	3	C.29141D+C1	-C.46568D+C2

RELAXED VARIABLES	NUMBER	INDEX	DUAL ACTIVITY	STATUS	FACTOR Y
	1	38	0.46812D+CC	3	-C.36065D-02
	2	47	0.80305D+CC	3	-C.34466D-C1
	3	19	-0.49602D-C1	2	-C.18021D+CC
	4	24	-0.1513CD-C1	2	-C.47089D+00

```
GRADIENT    0.2353CD-C1
TCLERANCE   0.2353CE-09
ACTIVED VARIABLES   NUMBER    INDEX      GRADIENT        STATLS     FACTER Y       SCLUTION
                       1        33    0.437238-C2          3    C.18728D+01   -C.44982D+02
                       2        47    0.41824C-C2          2    C.39224D+C3   -C.40399D+C2
                       3        19    C.17963D-C2          3    C.6C981D+01   -C.37543D+C2
                       4        24   -0.24368D-12          3    C.956613D+01  -C.32C58D+C2

RELAXED VARIABLES   NUMBER    INDEX    DUAL  ACTIVITY    STATLS     FACTER Y
                       1        47   -0.37462D+CC          2    -C.56056D-C3
                       2        43    0.1CC58D+CC          3    -C.665C8D-01
                       3        32   -0.102E1D-C1          2    -C.73772D+CC

GRADIENT    0.83481D-03
TCLERANCE   0.83481E-11
ACTIVED VARIABLES   NUMBER    INDEX      GRADIENT        STATLS     FACTER Y       SCLLTICN
                       1         8    0.59697D-C3          3    C.15369D+C1   -0.13861D+C2
                       2        32    0.33845D-C3          3    C.16015D+01   -0.12232D+02
                       3        15    0.49624D-13          3    C.162E5D+04   -C.311C4D+C1


REINVERSICN AFTER ITERATICN NUMBER .....          5
    FASE .....                                    2
    SCLUTICN .....                    -0.311C4D+C1
    CEN. BASE .....                              12
    ETA VECTERS .....                             5
    ETA ELEMENTS .....                           30


RELAXED VARIABLES   NUMBER    INDEX    DUAL  ACTIVITY    STATLS     FACTER Y
```