ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

Departamento de Engenharia de Minas

ANTONIO CARLOS NUNES

SISTEMA COMPUTADORIZADO PARA AJUSTE DE BALANÇO DE MASSAS E METALÚRGICO

São Paulo, Julho de 1992.

A minha esposa e companheira, Marta, pelo estímulo e compreensão ao longo de nossas vidas.

AGRADECIMENTOS

Gostaríamos de agradecer às seguintes pessoas e entidades que colaboraram na confecção da dissertação ora apresentada:

Em primeiro lugar, ao nosso orientador, Prof. Dr. Arthur Pinto Chaves, cujas sugestões contribuíram decisivamente para a conclusão do trabalho, e à Mineração Taboca S.A., representada pelo seu Superintendente de Operações, Engo. Jairo Augusto Vasconcelos Reis, a cujo corpo técnico temos o orgulho de pertencer, pela oportunidade de desenvolver o "software" aqui apresentado, pela autorização de utilizar os resultados e colocar à nossa disposição toda a sua estrutura e todo o apoio.

Em particular, ao engenheirando Paulo Fernando de Toledo Damasceno e ao geólogo Pedro Luiz Côrtes, diretor executivo da Greensoft Informática, pela sua inestimável colaboração, sem a qual não teríamos concluído o estudo.

Em especial, à minha esposa, pelo apoio e carinho dispensado no dia-a-dia do desenvolvimento desta atividade.

Finalmente, a todos que direta ou indiretamente contribuiram para a confecção deste trabalho.

RESUMO

É apresentado um "software" para ajuste de Balanço de Massas e Metalúrgico, muito útil para avaliar e acompanhar o desempenho de uma usina de tratamento de minérios.

Inicialmente, são feitas considerações sobre a necessidade e importância do ajuste dos dados experimentais disponíveis para o estabelecimento dos balanços de massas e metalúrgico, em usinas de beneficiamento. A seguir, são apresentados os principais procedimentos, métodos matemáticos e "softwares" disponíveis para este cálculo.

Descreve-se o método proposto, apresentando o seu algoritmo, baseado no método dos mínimos quadrados e nos multiplicadores de Lagrange, e a sua rotina de solução, ilustrado pelo estudo do caso da Mina do Pitinga, da Mineração Taboca S.A.

ABSTRACT

The goal of this thesis is to develop a software able to adjust the data of mass and metallurgical balances of an ore dressing plant.

Initially, some considerations about the necessity and importance of experimental data adjustment in mass/metallurgical balances in preparation plants are presented. Next, the main available computational procedures, mathematical methods and the softwares to calculate material balances are discussed.

The proposed method, its algorithm, based upon weighted least-squares techniques and Lagrangian multipliers is described and the solution routine is presented. The procedures are ilustrated by the case study of the Pitinga Mine, Mineração Taboca S.A.

INDICE Página.

NOTA SOBRE A REDAÇÃO

Na inexistência de termos consagrados em língua portuguesa, foram utilizadas as palavras inglesas correntes no jargão mineiro.

Foram mantidas as aspas para maior rigor formal da dissertação.

Os termos são os seguintes:

"on-line"

"off-line"

"software"

"hardware"

"oversize"

"undersize"

"overflow"

"underflow"

"apex"

ROM - "Run-of-mine".

A simbologia utilizada para a descrição do Diagrama de Fluxo de Dados (DFD), Figura 4, foi a utilizada em Análise de Sistemas, por isso alguns blocos são abertos.

A impressão das Tabelas e Figuras 4 e 6 foi feita através dos "softwares" LOTUS, SIDEWAYS e FLOWCHART, que não permitem a acentuação das palavras, o uso de cedilha e a utilização de vírgula decimal.

SIMBOLOS E NOMENCLATURA

Wj*	Valor real da vazão no fluxo j, j=1,2,..,n.
Xj*	Valor real da análise de um componente no fluxo j.
Wj	Valor experimental da vazão no fluxo j.
Xj	Valor experimental da análise de um componente.
Ŵj	Valor ajustado da vazão no fluxo j.
X̂j	Valor ajustado da análise do componente no fluxo j.
Θj	Valor da variável de busca para a vazão.
Ξj	Valor da variável de busca para a análise de componente.
β	relação entre duas vazões em uma determinada partição.
di	resíduo da equação de partição de massa.
Σ	símbolo da somatória.
Swj	desvio padrão da vazão wj.
Sxj	desvio padrão da análise do componente Xj.
λ	multiplicador de Lagrange.
Dx	valor do passo para o método de busca direta.
$S_{erf}2$	variância do erro fundamental.
Pi	fator de ponderação.
ri	i-ésimo ponto de dado medido.
r̂i	i-ésimo ponto de dado ajustado.
Sj^2	variância associada ao j-ésimo elemento de fluxo.
DW	diferença entre os valores experimental e calculado da vazão.
DX	diferença entre os valores experimental e calculado do componente.
M	matriz que caracteriza o fluxograma de processo.
mij	elemento da linha i coluna j da matriz M.

I. INTRODUÇÃO

O balanço de massas calculado a partir de dados medidos nos vários locais de uma unidade de processos é a ferramenta fundamental para muitos objetivos, tais como medida de desempenho da usina, qualidade e controle de produtos, controle de processos, avaliação econômica de processos alternativos, ensaios de bancada e escala piloto e projeto de usinas.

É possível calcular o balanço de massas por vários procedimentos independentes. Se os valores fossem obtidos sem erros de medidas, uma condição nunca encontrada na prática, todos os balanços de massas calculados, pelos diferentes modos, teriam que estar em acordo, ou seja, todos os dados seriam consistentes. A situação real é que ocorrem erros nas medidas, de modo que os resultados do balanço de massas, determinados a partir de cada procedimento, diferem. Consequentemente, é necessário um ajuste.

O uso de métodos matemáticos para ajuste de dados não dispensa o conhecimento técnico por parte de quem formula a solução, pois é necessária uma análise crítica das respostas, que poderão ser corretas do ponto de vista matemático, mas nem sempre serão verdadeiras no que se refere ao tratamento de minerais. Sempre, porém, o uso de um programa computacional proporcionará velocidade ao processo e ganho de tempo (01).

A vazão de um mineral, que entra em determinado equipamento, é igual àquela que deixa o mesmo equipamento, desde que se tenham atingido as condições de equilíbrio do processo.

Com o objetivo de avaliar e modelar os processos de

beneficiamento, é necessário obter dados consistentes para satisfazer a conservação de massa.

Nas operações de processamento mineral, as informações de processo, requeridas ou disponíveis pelo engenheiro responsável, são usualmente restritas aos dados de entrada e saída, necessários para calcular os balanços metalúrgicos diários e semanais. Estes dados são utilizados para avaliar o desempenho econômico da usina e, valores desfavoráveis podem indicar um mau funcionamento do processo. Em muitas usinas de processamento mineral, o controle de curto prazo é baseado em algumas poucas medidas, obtidas em diferentes pontos do circuito; geralmente estão disponíveis a densidade de polpa, análises químicas e granulométricas ou medidas de vazão.

Avaliações sistemáticas do desempenho de equipamentos, de toda a usina ou do processo de beneficiamento mineral são extremamente úteis para poder operá-los apropriadamente. Os objetivos destas avaliações podem ser simplesmente o conhecimento das entradas/saídas de um processo, ou a implementação de uma estratégia de controle para uma determinada operação.

Os estudos começam invariavelmente pela tomada de medidas. As usinas de processamento mineral são, como regra geral, pouco instrumentadas e há uma considerável falta de informações sobre o estado do processo. Frequentemente, os dados devem ser obtidos através de amostragens manuais e análises.

Na realidade industrial, apenas algumas variáveis podem ser medidas e estas, na maioria das vezes, têm que ser feitas através de análises químicas e/ou físicas. Segundo Gy (02), a

medida experimental destas variáveis está sujeita a erros provenientes de várias fontes, e que acabam por não satisfazer os princípios do balanço de massas, ou seja, o da conservação total da massa e o da conservação após o material sofrer transformações físicas.

Quando as informações necessárias não estão disponíveis, o método usual aplicado é fazer medidas da vazão, apenas para alimentação e produto final, e ajustar os valores restantes. Às vezes, as vazões nos demais fluxos são calculadas a partir de outras características medidas, tais como análise de teores ou granulometria, em amostras retiradas em pontos apropriados. Este método é muito trabalhoso e pouco confiável, até mesmo para circuitos pequenos.

O número de amostras necessárias para dar uma estimativa precisa para os fluxos é geralmente grande. A importância relativa da amostragem de fluxos com grande vazão, do estado de operação da usina, as dificuldades de amostragem e dos erros de ensaio é diferente em condições de laboratório, escala piloto ou na usina. No laboratório, os processos são simulados independentemente; assim, não existem problemas de iteração. Uma amostra é geralmente todo o produto do ensaio e a precisão das estimativas é somente limitada pela análise e pela precisão experimental. Numa escala piloto, estes fatores podem ainda predominar, mas há um aumento de dificuldades a serem encontradas na usina.

Para se diminuir os efeitos de imperfeições de amostragem, pode ser utilizada instrumentação "on-line"; embora,

existam limitações, devido à natureza da polpa, normalmente com baixa porcentagem de sólidos. Instrumentos "on-line" são relativamente raros nas usinas de processamento mineral. Para aumentar as dificuldades, raramente os projetistas de usinas fazem previsão para tais medidas, quando estão estudando e construindo a unidade. O resultado é que a maioria das usinas de processamento mineral conta com amostragens, somente onde é possível sua realização (03).

Valores operacionais podem ser obtidos pela média de um número de medidas da usina em cada ponto do processo; mas, a sua utilização é limitada pelo esforço requerido para implementar tais sistemas de medida. Para processos de grande escala, o fator econômico de obtenção dos dados é limitante.

Técnicas de ajuste de dados e balanço de massas são, portanto, ferramentas muito úteis para obter sistematicamente a imagem do estado do processo. Estes procedimentos são de grande valor para o conhecimento do processo e para o desenvolvimento de modelos, enquanto diminuem a quantidade de experimentos necessários para obter a informação desejada. Elas são também ferramentas valiosas para determinar melhores condições operacionais e melhorar a economicidade da usina e a qualidade dos resultados.

Este trabalho mostra o modo como um computador pode ser usado para gerar o fechamento do balanço de massas para um dado fluxograma de processo, com base em dados experimentais.

O propósito do sistema computadorizado para ajuste do balanço de massas é dispor-se de um método sistemático e

versátil para calcular as vazões dos vários fluxos de um processo e corrigir os dados experimentais, a fim de torná-los consistentes.

O programa proposto é baseado na técnica dos mínimos quadrados, utilizando-se os multiplicadores de Lagrange e o método de eliminação de Gauss como ferramentas de solução do sistema linear montado.

Uma vez ajustados os valores para um referido processo, o sistema pode ser operado para as determinações do dia-a-dia da operação da usina.

Assim como o balanço de massas, o cálculo dos balanços metalúrgicos, a partir de dados analíticos imprecisos, é um problema que ocorre nas usinas. O algoritmo ora proposto, permite também a conciliação do balanço metalúrgico.

II. REVISÃO BIBLIOGRÁFICA

II.1. Revisão de conceitos (04),(06).

II.1.1. Cálculo da Partição da Massa

O método tradicional de cálculo da partição de massa, em uma operação unitária, é conhecido como regra dos n produtos e consiste em calcular as n variáveis correspondentes às vazões dos produtos.

O caso mais simples ocorre quando se tem apenas um produto, figura 1, e as equações são:

$W1 = W2$,

$W1\ X1 = W2\ X2$.

onde:

Wj = vazão no fluxo j,

Xj = análise de um componente do fluxo j.

Esse componente pode ser um teor, uma faixa granulométrica, a porcentagem de sólidos ou outro parâmetro eventualmente disponível.

Figura 1: Uma operação unitária I, com um fluxo de alimentação e um de produto.

Para calcular, digamos, W2 é necessário apenas conhecer W1 e nenhuma análise (química, granulométrica, % de sólidos) de Xj é necessária. Se, no entanto, conhecermos X2, pode-se calcular X1 e vice-versa. Em ambos os casos, o valor da variável

calculada contém o mesmo erro que o da variável medida.

Com dois produtos, o problema corresponde à figura 2 e as equações válidas são:

$$W1 = W2 + W3 ,$$

$$W1X1 = W2X2 + W3X3 .$$

Figura 2: Uma operação unitária I, com um fluxo de alimentação e dois de produto.

Pode-se calcular o sistema de equações, se tivermos, no máximo, duas incógnitas, geralmente W2 e W3:

$$W3 = \frac{X1 - X2}{X3 - X2} * W1 \qquad e \qquad W2 = \frac{X1 - X3}{X2 - X3} * W1 .$$

Para três produtos, de acordo com a figura 3, as equações válidas são:

$$W1 = W2 + W3 + W4 ,$$

$$W1X1 = W2X2 + W3X3 + W4X4 .$$

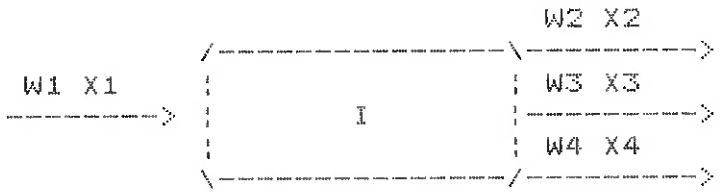


Figura 3: Uma operação unitária I, com um fluxo de alimentação e três de produto.

Se os fluxos correspondentes aos produtos devem ser calculados, deve-se medir um segundo Xj' em todos os fluxos de modo a obter a terceira equação:

$$W1X1' = W2X2' + W3X3' + W4X4' .$$

Por combinação das 3 equações, determina-se:

$$\frac{W4}{W1} = \frac{(X1' - X2')(X4 - X2) + (X1 - X2)(X2' - X4')}{(X3' - X2')(X4 - X2) + (X3 - X2)(X2' - X4')}$$

e as outras vazões, W3 e W2, por permutação.

Generalizando-se os casos acima, para n produtos, o problema pode ser resolvido se forem feitas n-1 análises de componentes diferentes em cada fluxo. Quando n cresce, o método de substituição para resolver o sistema torna-se demorado e trabalhoso.

Para estabelecer a partição, três tipos de variáveis devem ser consideradas:

1. Os valores experimentais (sem marcação), que são conhecidos com uma certa precisão, mas não são auto-consistentes, ou seja, por si só eles não satisfazem a lei de conservação de massas.
2. Os valores ajustados (que serão denotados por ^), que são numericamente auto-consistentes e podem ser obtidos a partir das medidas experimentais.
3. Os valores reais (que serão denotados por *), que são auto-consistentes, mas não podem ser obtidos a partir das medidas experimentais.

Tomando o caso dos dois produtos, tem-se que W1*, W2*, W3* são os valores reais das vazões da alimentação e dos produtos de um processo de separação. Os componentes dos fluxos são designados X1i*, X2i* e X3i*, respectivamente, onde i= 1 até n, e é o mesmo componente para um valor particular de i, mas podem ser vários componentes em cada fluxo. O processo pode ser escrito:

$$W1^* (X1i^*) \longrightarrow \begin{cases} W2^* (X2i^*) \\ W3^* (X3i^*) \end{cases} .$$

Desde que os valores são reais, o balanço de massas tem que ser exato para as vazões e componentes, as seguintes equações podem ser escritas:

$$W1^* = W2^* + W3^* \quad , \qquad (1)$$

$$W1^* X1i^* = W2^* X2i^* + W3^* X3i^* \quad . \qquad (2)$$

Dividindo a equação (1) por W1*

$$1 = \frac{W2^*}{W1^*} + \frac{W3^*}{W1^*} \quad ,$$

$$\frac{W3^*}{W1^*} = 1 - \frac{W2^*}{W1^*} \quad , \qquad (3)$$

fazendo $\beta = \dfrac{W2^*}{W1^*}$ (4) ,

combinando (2) e (4) :

$$X1i^* = \beta X2i^* + (1 - \beta) X3i^* \quad . \qquad (5)$$

Portanto, para o cálculo de uma partição de massas, utilizando-se os valores reais a equação (5) é igual a zero:

$$X1i^* - \beta X2i^* + (1 - \beta) X3i^* = 0 \quad . \qquad (6)$$

A regra dos n produtos apresenta algumas limitações: caso um experimento ou uma campanha de amostragem seja repetido, teoricamente, o aumento dessa repetição forneceria um valor médio, cada vez mais próximo do valor real. Na prática, isto não acontece, pois o valor experimental muda com o tempo, uma vez que, numa usina de beneficiamento, as condições nunca são estáveis por todo o tempo. Pode-se, no entanto, obter informações suficientes para o cálculo de várias estimativas de vazão em cada fluxo. Por exemplo, se as distribuições granulométricas da alimentação e dos produtos de um classificador forem conhecidas, pode-se estimar a vazão total que entra, a partir de cada fração granulométrica.

Se estas estimativas coincidirem, não haverá necessidade de análises adicionais, pois o conjunto de dados será consistente. Entretanto, isto raramente ocorre e, portanto, é necessário ajustar os valores de modo que se tornem coerentes entre si, satisfazendo todas as restrições de balanço de massas.

II.1.2. Métodos matemáticos para ajuste de balanço de massas

Dentre os métodos de ajustes de dados existentes, pode-se salientar os seguintes:

- minimização da soma dos quadrados dos resíduos do fechamento do balanço,
- método dos mínimos quadrados,
- método da busca direta,

que serão examinados em seguida.

II.1.2.1. Minimização da soma dos quadrados dos resíduos do fechamento do balanço

O método da soma dos resíduos do fechamento do balanço de massas assume que os valores experimentais das análises são suas melhores estimativas (05). Como o valor medido não coincide com os valores reais, o balanço de massas, equação (6), não será fechado, ou seja, a soma não será zero, mas terá um resíduo d.

Portanto, as equações de conservação de massa para os valores experimentais podem ser escritas:

(7) $W1 - W2 - W3 = di^{m}$,

(8) $W1\ X1i - W2\ X2i - W3\ X3i = di^{t}$.

onde di = são os resíduos na equação de fechamento de balanço de massas, gerados pelos erros experimentais nas medidas, com o "s" representando os sólidos e o "t" o teor.

Partindo da equação (6):

(9) $X1i - \beta\ X2i - (1 - \beta)\ X3i = di$.

O método consiste em encontrar a melhor estimativa de β, que minimize a soma dos quadrados de todos os resíduos:

(10) $F = \sum (di^{m})^{2} + \sum (di^{t})^{2} = mínimo$.

O valor de F não pode ser zero, a menos que os dados sejam consistentes (4). O método mais simples para encontrar o mínimo é diferenciar em relação a β e igualar a zero:

$$(11) \qquad X1i - \beta\, X2i + (1-\beta)\, X3i = di\,,$$

$$(12) \qquad \Sigma\, di^2 = \Sigma\, [\, X1i - \beta\, X2i + (1-\beta)\, X3i\,]^2\,,$$

(13)

$$\frac{d\ \Sigma di^2}{d\ \beta} = 2\,\Sigma\,[\, X1i - \beta\, X2i - (1-\beta)\, X3i\,]\,(X3i - X2i) = 0$$

(no ponto mínimo),

e

$$(14) \qquad \hat{\beta} = \Sigma\,(X1i - X3i).(X2i - X3i) \;/\; \Sigma\,(X2i - X3i)^2.$$

$\hat{\beta}$ é agora denominado $\hat{\beta}$, como a melhor estimativa do valor verdadeiro β^*, baseado no método da soma dos mínimos quadrados dos erros de fechamento do balanço de massas.

II.1.2.2. Método dos Mínimos Quadrados

O método dos mínimos quadrados dos resíduos do ajuste dos componentes utiliza valores obtidos por processo de cálculo, obtendo-se os resíduos entre esses valores e os obtidos experimentalmente.

Se a cada análise Xj ou medida de vazão Wj forem associados as melhores estimativas $\hat{X}j$ e $\hat{W}j$, então os resíduos são elevados ao quadrado e somados e o valor de ajuste é aquele que minimiza a soma F, dada pela equação:

$$(15) \qquad F = \sum_{j=1}^{m} \sum_{k=1}^{z} (\hat{X}i^k - Xi^k)^2 + \sum_{j=1}^{m} (\hat{W}j - Wj)^2 = \text{mínimo},$$

onde Wj é a vazão medida do fluxo j;

$\hat{W}j$ é a vazão ajustada do fluxo j;

Xj^k é o valor medido do componente k no fluxo j;

$\hat{X}j^k$ é o valor ajustado do componente k no fluxo j;

m é o número de fluxos;

z é o número de análises dos componentes em cada fluxo.

Um refinamento que pode ser utilizado no método dos mínimos quadrados é a introdução de ponderadores (06), que terão a finalidade de conduzir o ajuste das variáveis de acordo com algum critério pré-determinado, como por exemplo, a precisão conhecida de suas medidas.

O conceito de ponderadores para a soma dos quadrados não é, portanto, mais do que a inclusão da experiência do operador nos procedimentos de cálculo.

O peso de uma medida pode também ser inversamente proporcional à sua variância. O conhecimento do processo, condição da amostragem naquele ponto e a experiência qualitativa geram outras aproximações alternativas. Alguns fluxos podem ser convenientemente amostrados e a eles devem ser atribuídos maiores ponderadores do que a outros com maior dificuldade de amostragem. Portanto, o modelo de erros a ser adotado, depende da experiência e opção do usuário.

Logo, a equação (15) pode ser substituída por:

$$(16)\quad F = \sum_{j=1}^{m} \sum_{k=1}^{z} \frac{(\hat{X}i^k - Xi^k)^2}{S\ Xj^2} + \sum_{j=1}^{m} \frac{(\hat{W}j - Wj)^2}{S\ Wj^2} = \text{mínimo.}$$

Para minimizar a função F ponderada, pode ser utilizado o Cálculo Diferencial, que ensina que, para obter um ponto de mínimo da função F (Wj , Xj), inicialmente, devem-se encontrar

seus pontos criticos, ou seja, os (Wj , Xj), tais que:

$$(17)\quad \frac{dF}{dWj\ dXj} = 0 \quad ; j = 1, 2, \ldots, n ,$$

montando-se um sistema linear em função de (Wj , Xj), resolvendo-o e chegando ao valor ajustado.

Estes cálculos são um tanto trabalhosos, e o matemático francês Lagrange elaborou um método para abreviá-los.

II.1.2.2.1.. Método dos Multiplicadores de Lagrange

O método é utilizado para simplificar os problemas de minimização, que são sujeitos a condições e/ou restrições. As restrições são expressas de tal forma que se igualem a zero.

A função soma, equação (16), a ser minimizada é modificada pela adição de equações de restrição, multiplicadas por um fator chamado de Multiplicador de Lagrange. A equação 16 toma a forma:

(18)

$$L(Xj,Wj,\lambda_j) = \sum_{j=1}^{m} \sum_{k=1}^{z} \frac{(\hat{x}i^k - Xi^k)^2}{S\ Xj^2} + \sum_{j=1}^{m} \frac{(\hat{W}j - Wj)^2}{S\ Wj^2} + \sum_{j=1}^{m} \lambda_j\ Gj\ (\hat{x}j, \hat{W}j),$$

onde Gj ($\hat{x}$j , $\hat{W}$j) são as restrições (do circuito),

λ_j é o multiplicador de Lagrange,

L é o Lagrangeano.

A configuração do problema descrito pela equação (18) é chamada minimização sujeita à restrições.

O Lagrangeano é diferenciado em relação a cada um dos termos desconhecidos ($\hat{x}$j , $\hat{W}$j , λ_j) e os multiplicadores de

Lagrange são usados para se obter a solução. Os valores de x̂j e ŵj, que são solução da função, são também solução do problema.

II.1.2.3.. Método de busca direta

O método de busca direta, que não envolve derivadas e multiplicadores de Lagrange, é utilizado para obter o mínimo da função dada pela equação (15).

O método consiste em fixar valores iniciais, as variáveis de busca, denotadas por ^, cuja escolha é arbitrária, as quais, acrescidas de um valor Dx, chamado de passo, vão gerando valores para a função F até se atingir o mínimo.

Para se obter o mínimo da função F, pelo método da busca direta, deve-se ter, pelo menos, informações suficientes sobre o circuito a ser balanceado, ou seja, ter os valores experimentais de cada fluxo do circuito. Quanto maior o número de informações de que se disponha, maior será a probabilidade de se atingir um ajuste confiável.

Foram formulados vários algoritmos de busca direta, dentre eles tem-se os algoritmos de COGGINS, o de POWELL, o de ROSENBROCK e o de HOOKE-JEEVES. Serão descritos, a seguir, os métodos de COGGINS e POWELL.

II.1.2.3.1 Algoritmo de COGGINS

Este programa acha o mínimo de uma variável simples de função não-linear, sem restrições. O método é uma combinação das técnicas de variáveis simples, faz a comparação entre valores da função objetivo, para os pontos incrementados de Dx, de uma forma uni-direcional, portanto, é conveniente a escolha de um bom ponto de partida (07).

A descrição do método será feita, baseada em somente um dos termos da função F, dada pela equação (15), para a obtenção do valor mínimo relativo à vazão Wj. Para se obter o mínimo para os componentes Xj, o procedimento é idêntico.

O algoritmo de COGGINS é o seguinte:

-Rotina 0 : A avaliação da função F(W1) para o valor inicial da variável de busca: ponto W1.

-Rotina 1 : A variável de busca é incrementada por um passo Dx para dar o ponto W2. A função F(W2) é avaliada:

a) se F(W1) > F(W2), o passo é dobrado, somado a W1 e o programa transfere o valor para a rotina 2.

b) se F(W1) < F(W2) , o passo é diminuido de W1, os pontos W1 e W2 são permutados e o programa transfere o valor para a rotina 2.

Ao final da rotina 1, a função foi avaliada nos pontos W1 e W2 e F(W1) é maior do que F(W2).

-Rotina 2 : Usando o valor transferido da rotina 1, a função F(W3) é avaliada no ponto W3.

a) Se F(W1) > F(W3) , o passo é dobrado somado a W1, W1 e F(W1) são trocados por W2 e F(W2), que por sua vez são trocados por W3 e F(W3). Então, a rotina 2 é re-iniciada.

b) Se F(W1) < F(W3) , o ponto ótimo está entre os pontos W1 , W2 e W3, de modo que F(W1) e

F(W3) são maiores do que F(W2). O programa transfere o valor para a rotina 3.

-Rotina 3 : A equação quadrática F, é ajustada para os 3 pontos obtidos ao final da rotina 2. A localização do mínimo Wm, obtido por dF / dW = 0 é:

$$(19)\quad Wm = \frac{1}{2}\,\frac{(W2^2-W3^2)F(W1) + (W3^2-W1^2)F(W2) + (W1^2-W2^2)F(W3)}{(W2 - W3)F(W1) + (W3 - W1)F(W2) + (W1 - W2)F(W3)}\ .$$

A função é avaliada no ponto Wm. Se qualquer :

$$| W - Wi | < W\,lim\ , \qquad i = 1,2 \text{ ou } 3\ ,$$

Wlim = grau de precisão desejado pelo usuário e definido no início da rotina,

está satisfeito o processo de minimização.

II.1.2.3.2 Algoritmo de POWELL

O algoritmo de POWELL segue, basicamente, as mesmas rotinas do algoritmo de COGGINS. No entanto, o método de POWELL é baseado nas direções conjugadas, ou direções de busca (07), sendo, portanto, uma solução gráfica, para se determinar o mínimo da função objetivo F, dada pela equação (15).

A determinação do mínimo ao longo de uma direção é feita avaliando-se a função F na direção h, definida por p+W1h, p+W2h , p+W3h, onde p é um ponto de referência e h é definida por suas componentes de direção W1 , W2 , ... (valores experimentais das vazões).

O método consiste em ajustar uma curva passando pelos três pontos ((Ø1 , F(p+Ø1)) , ((Ø2 , F(p+~w2)) , ((Ø3 , F(p+Ø3)) e encontrar o mínimo que é localizado em p + Wmh, sendo Wm dado pela equação (19).

O procedimento é iterado até a precisão, na posição de mínimo pré-definida, para a direção h. Na primeira iteração do algoritmo de POWELL, as direções são aquelas das variáveis W1...Wn e o passo q é O,1.e.hi, onde e é denominado fator Escala e hi é a direção para cada iteração, e a função F é avaliada para:

p valor inicial de W

p + qh

p - qh ou p + 2qh, dependendo se S(p) é menor ou maior que S(p+qh).

Na situação geral (Ø1, Ø2 , Ø3), se a posição do mínimo da curva é tal que p + Wmh tem componentes maiores do que eh1 , eh2, ..., ehn , o máximo passo permitido é adotado no sentido de diminuir F e o valor da função no ponto, que é o mais distante de p + Wmh, é descartado, repetindo-se o processo.

O critério de convergência é satisfeito quando os valores de todas as variáveis entre duas sucessivas iterações são menores do que 10% da precisão determinada, para cada direção.

II.1.3. Métodos de Solução de Sistemas

II.1.3.1.. Método de Eliminação de Gauss

O método de Eliminação de Gauss consiste em transformar um sistema linear em um sistema linear equivalente, com matriz dos coeficientes triangular superior (08), pois esta é de resolução imediata.

Será apresentado, inicialmente, um algoritmo para a resolução de sistemas triangulares e, a seguir, será estudado como o método de Eliminação de Gauss efetua a transformação do sistema linear no sistema triangular equivalente.

II.1.3.1.1.. Resolução de sistemas triangulares

Seja um sistema linear: Ax = b, onde A: matriz n x n, triangular superior, com elementos da diagonal diferentes de zero. Escrevendo as equações deste sistema, tem-se:

$$\begin{aligned} a11x1 + a12x2 + \ldots\ldots\ldots + a1nxn &= b1 \\ a22x2 + \ldots\ldots\ldots + a2nxn &= b2 \\ &\vdots \\ annxn &= bn \quad , \end{aligned}$$

onde, aii ≠ 0, i= 1,2, ..., n é dito um sistema triangular.

A resolução do sistema é feita por:

$$xn = \frac{bn}{ann}$$

e:

$$xi = \left[bi - \sum_{j=i+1}^{n} aij\, xj \right] / aii \quad ,$$

para i = n-1 , n-2 , ... , 1.

II.1.3.1.2.. Descrição do método de Eliminação de Gauss

Usando a propriedade da Algebra Linear de que a solução de um sistema linear não se altera se subtrairmos de uma equação outra equação do sistema, multiplicada por uma constante e, sendo o det A ≠ 0, tem-se os seguintes passos para a triangularização da matriz (09):

- a eliminação é feita por colunas e chama-se de estágio k do processo à fase em que se elimina a variável xk das equações k+1, k+2, ..., n;
- usa-se a notação aij^{k}, para denotar o coeficiente da linha i e coluna j, no final do k-ésimo estágio, bem como bi^{k} será o i-ésimo elemento do vetor constante no final do estágio k;
- o elemento $aii^{(k-1)}$ é chamado o pivô de cada estágio e deve ser diferente de zero;
- os elementos $mij^{k} = \dfrac{aij^{(k-1)}}{aii^{(k-1)}}$

são os multiplicadores do k-ésimo estágio, para i,j,k = 1...n.

- os elementos finais de cada estágio são:

$$aij^{k} = aij^{(k-1)} - mij * aij^{(k-1)} \quad \text{para} \quad \begin{matrix} i = 2,.....,n \\ j = 1,.....,n \\ k = 1,.....,n \end{matrix} ,$$

$$bi^{k} = bi^{(k-1)} - mij * bi^{(k-1)} \quad \text{para} \quad i = 2,.....,n .$$

- Observação: Se durante a aplicação do método de Eliminação de Gauss a um sistema, ocorrer um pivô nulo e todos os elementos da coluna correspondente ao pivô, abaixo da diagonal principal, forem nulos, o determinante da matriz A é igual a zero. Portanto, o sistema é indeterminado ou inconsistente e a aplicação do método deve ser interrompida.

II.1.3.2.. Condensação Pivotal

Nas aplicações do método de Eliminação de Gauss, trabalhamos com frações, os computadores utilizam a representação dos números em ponto flutuante; portanto, torna-se inevitável o aparecimento de erros de arredondamento, que ocorrem nas operações com esses números e isto pode comprometer a solução obtida.

Para minimizar a influência destes erros de arredondamento, utiliza-se a condensação pivotal, que consiste em escolher o elemento pivô no início de cada etapa i, i=1,2,...n como sendo o número de maior valor absoluto dentre os elementos da i-ésima coluna, que estão na diagonal principal ou abaixo dela. Se o elemento escolhido para pivô não estiver na diagonal principal, devemos trocar sua linha de índice i. Os multiplicadores que estão colocados nos lugares dos elementos zerados também devem ser trocados.

O procedimento de solução do sistema, fazendo-se sempre o pivotamento a cada estágio, continua o mesmo para o método de Eliminação de Gauss.

II.1.3.3 Outros Métodos

Existem ainda vários outros métodos para o cálculo e resolução de sistemas, que são aplicados para a resolução de problemas mais complexos, tais como resolver equações não lineares, com polinômios de grau n, com o método de Newton-Raphson, e resolver sistemas "esparsos", sistemas com grande quantidade de elementos nulos (0's), que podem ser aplicados os métodos iterativos, de Gauss-Jacobi ou Gauss-Seidel. Como estes

métodos não são aplicados normalmente nas rotinas disponíveis, não serão detalhados neste trabalho.

II.1.4.. Avaliação de Erros

O problema é encontrar que tipos de erros são introduzidos nas medidas das variáveis do balanço de massas. Após uma avaliação dos erros na confiabilidade dos valores experimentais, é feita a análise de sensibilidade, que permite uma classificação dos resultados. Esta análise é importante, pois ela ajuda a definir uma estratégia apropriada de amostragem e a detectar desvios em relação ao valor calculado, baseado no modelo de erros assumido.

Será avaliado o problema da estrutura do erro, para dados experimentais, baseado nas observações e definições de Pierre Gy (02).

II.1.4.1. Componentes do erro total

O erro total é uma combinação de :

a. erro fundamental de amostragem,
b. erro devido às variações do lote no tempo,
c. erros devidos ao operador e ao amostrador,
d. erros devidos à preparação das amostras e à análise,
e. erros devidos à flutuação das operações unitárias.

II.1.4.2. Erro fundamental (EF)

O erro fundamental (EF) foi definido como o erro proveniente da constituição de um conjunto de particulas submetidas à amostragem, levando em consideração as propriedades

físicas destas partículas.

Para exemplificar quais propriedades influenciam este erro, a equação (20) dá a variância do EF, seu componente principal, demonstrado por Gy:

$$(20) \qquad S_{EF}^2 = C\ l\ f\ g\ d^3 \left(\frac{1}{Ms} - \frac{1}{Ml} \right) ,$$

onde: Ms = massa da amostra (g) ,

Ml = massa do lote (g) ,

l = fator de liberação (0 ou 1); para minério não liberado, l=1 ,

f = fator de forma da partícula (0,5 para a maioria dos sólidos) ,

g = fator da distribuição granulométrica (0,25 para materiais bem graduados e 1 para mal graduados) ,

d = tamanho máximo da partícula (cm) ,

C = fator de composição (g/cm3).

De todos os erros de amostragem, o erro fundamental nunca pode ser eliminado. Os únicos fatores que podem ajudar na minimização do EF são:

1. o peso da amostra Ms: quanto maior, menor o erro. Teoricamente, se fosse possível Ms = Ml, o que não ocorre na prática, o EF seria eliminado,
2. o diâmetro da partícula d: quanto menor o tamanho da partícula, menor será o erro.

II.1.4.3. Erro devido à variação do lote no tempo

As amostragens podem seguir rotinas que levem a pequenas ou grandes variações, cuja periodicidade poderá ser outra fonte de erro. A única maneira de reduzir tais erros é tirar amostras combinadas, fazendo-se ainda amostragem sistemática, em intervalos regulares.

II.1.4.4. Erro devido ao amostrador e operador

Este erro pode ser gerado em função da forma e posicionamento do amostrador, ou do manuseio inadequado do dispositivo de amostragem, não permitindo que as partículas tenham a mesma probabilidade de entrar no amostrador.

As regras básicas para minimizá-lo são:

- o cortador deve ser posicionado perpendicularmente ao eixo do fluxo.
- a largura da boca do cortador deve ter, pelos menos 10 mm e ser 3 vezes maior que a maior dimensão da partícula.

II.1.4.5. Erro devido à preparação e à análise da amostra

As fontes de tal erro são numerosas: contaminação, perda, alteração, retirada de alíquotas, método de análise, etc.

Pode ser minimizado, adequando-se o método de análise, máximo controle na identificação das amostras, limpeza e extremo cuidado com o material de manuseio das amostras e supervisão permanente.

II.1.4.6. Análise de Sensibilidade

A análise de sensibilidade consiste em estudar a relação entre o modelo do erro dos dados experimentais e o das estimativas. O cálculo do balanço de massas deve estar de acordo com uma precisão estimada e ideal, para os valores ajustados.

Pode ser feito através de dois métodos diferentes: o analítico e o de Monte-Carlo.

O princípio do método analítico é o cálculo da variância de uma combinação linear de duas variáveis aleatórias, W1 e W2, ou X1 e X2.

O método assume como hipótese de trabalho, que os fatores para a validade da variância das estimativas são: os erros medidos seguem uma distribuição normal, são independentes e a estimativa usada para o valor real tem boa precisão.

O método de Monte-Carlo baseia-se na capacidade do computador de gerar números aleatórios, em função das médias e desvios-padrão dos dados, podendo-se simular valores de análises, que seriam obtidas num processo de amostragem.

Depois que um primeiro cálculo de balanço for completado, pode-se recalcular, por simulação, todos os valores experimentais e resolver o problema para a nova série de valores calculados. Após realizar n vezes esse procedimento, calcula-se a média e o desvio-padrão, que quantificarão a sensibilidade dos valores de vazão calculados em relação aos desvios naturais das variáveis observadas.

Outras características da análise de sensibilidade

são a determinação do erro na estimativa e a avaliação dos resultados experimentais e do modelo de erro adotado para fazer o cálculo.

II.1.5.. Descrição Conceitual de Programas

II.1.5.1.. Caracteristicas dos Programas Computacionais

Um programa para cálculo de balanço de massas deve, no mínimo:

a. atender a um grande número de aplicações;
b. ser facilmente transferível;
c. ser bem documentado;
d. fornecer informações acessíveis sobre a entrada de dados;
e. ter uma saída de informações claramente compreensível ao usuário.

II.1.5.2.. Entrada

A síntese da entrada de dados, que qualquer programa requer, seria composta dos valores das análises e medidas (teores e vazões), do modelo de erro, isto é, os desvios-padrão associados a cada valor e os elementos da matriz que descreve o fluxograma. Os outros elementos dependem do programa e do seu uso específico.

O procedimento mais conveniente de entrada dos dados é o uso de tabelas ou planilhas, de modo que facilite a operação pelo usuário, permitindo-lhe introduzir alterações rapidamente e sem erros.

II.1.5.3.. Saída

A saída deve conter um resumo dos dados de entrada,

apresentando-os sob forma de tabelas; fornecendo os dados ajustados e, eventualmente, a distribuição de massa dos vários fluxos.

II.1.5.4.. Documentação

Tradicionalmente os documentos associados a programas computacionais descrevem as técnicas usadas pelo programa e aspectos relevantes, como memória necessária e sub-rotinas utilizadas.

As informações sobre como processar o sistema são frequentemente relegadas a segundo plano ou, quando existem, são tratadas em nível muito profundo, contendo detalhes supérfluos e informações incoerentes.

Considerando-se que o usuário de tais programas é, em geral, um elemento sem maiores conhecimentos no campo da computação, há a necessidade de se criar uma documentação do tipo hierárquico, ou seja, em vários níveis de detalhamento (06).

Assim, ao usuário direto do pacote seria reservado um documento contendo informações objetivas, ficando um outro volume destinado a técnicos qualificados da área de computação, que conteria detalhes a nível de "software" e "hardware", compatíveis com pesquisas, que eventualmente se necessitem fazer.

II.1.5.5.. "Hardware"

Os programas foram inicialmente desenvolvidos para uso em computadores de médio a grande porte e sua transferência para mini ou microcomputadores impôs uma limitação de tamanho dos fluxogramas a serem estudados e terem seus balanços de massas

ajustados, e também a eventual necessidade de uma nova versão em linguagem diferente.

No princípio, os microcomputadores tiveram grande dificuldade a nível de transferência do "software", porque cada fabricante de equipamentos implementava uma linguagem BASIC diferente na sua máquina, o que fazia com que um programa, desenvolvido para a máquina específica, não se adequasse a outra, a menos que fossem feitas modificações.

O uso crescente dos microcomputadores e seus baixos custos modificaram esse quadro. Atualmente, os equipamentos de pequeno porte são padronizados pelo modelo IBM e aceitam várias linguagens de programação, no momento, até as de quarta geração.

II.1.5.6.. Vantagens e Desvantagens

Os autores dos programas para cálculo de balanço de massas nunca deixaram de enfatizar as vantagens do seu uso, colocando as desvantagens sempre num plano marginal. Eles são unânimes em afirmar que o tempo requerido para o cálculo do balanço é minimizado, restando mais tempo para o estudo de resultados e eventuais modificações e/ou testes podem ser feitos sem maiores problemas. Além disso, qualquer engenheiro ou técnico, mesmo inexperiente, pode fazer os cálculos de circuitos complexos e uma vez estabelecido o fluxograma e preparada a entrada de dados, o pacote pode ser usado rotineiramente, apenas fazendo-se as modificações relativas à parte experimental.

Como desvantagens, ressalta-se que o uso mecânico do pacote pode levar a respostas incoerentes. Sem um estudo crítico dos resultados, pode-se tomar decisões desastrosas. Por outro lado, os engenheiros de operações, em sua maioria, ainda relutam em aceitar os resultados provenientes de um computador, provavelmente, por não terem tido tempo hábil para acompanhar a rapidíssima evolução da informática. Isto pode ser contornado, incentivando-o a participar da confecção do problema e da análise dos resultados.

II.2 Procedimentos e "softwares" disponíveis para o ajuste de dados de Balanço de Massas

Uma variedade de técnicas matemáticas está disponível para ajustar dados experimentais, obtidos na usina. Esses dados ajustados permitem o fechamento do balanço de massas. Os procedimentos de ajuste existentes minimizam a função objetivo F, dada pela equação (16), ou por alguma função semelhante a ela.

Em problemas onde nenhuma vazão, ou apenas uma, é conhecida pode-se trabalhar com vazões relativas , escolhendo-se uma como base (=100, 1 ou outro valor qualquer). Faz-se uma estimativa inicial para um conjunto de vazões relativas sendo W1 a vazão base, a partir da qual todas podem ser determinadas.

As técnicas para minimizar F, ou a forma apropriada de F, podem ser divididas em duas categorias, as que envolvem os multiplicadores de Lagrange e derivadas, e as que envolvem os métodos de busca direta.

Alguns autores, tais como White et al. (10) e Mular (11), usam diretamente as técnicas de minimização que trabalham as variáveis independentes, isto é, em um conjunto de variáveis a partir do qual todos os outros podem ser calculados, método de busca direta. Outros autores, tais como Wiegel (12), Cutting (13) e os usuários do método Wiegel (Laguitton (14)), montam a função Lagrangeano, que incorpora as restrições na função a ser otimizada, utilizando o recurso dos multiplicadores de Lagrange.

O primeiro programa para cálculo computacional de balanço de massas na indústria mineral, que envolvia múltiplas análises de espécies e valores de vazão, foi desenvolvido, em

1972, por Wiegel, utilizando os multiplicadores de Lagrange. Desde então, um grande número de pacotes similares têm surgido, alguns bastante parecidos com o original de Wiegel: Towsend, 1974; Reid e Rimmer, 1975; Cutting, 1976/1979; Wiegel, 1978/1979; Tippin; Laguitton e Wilson, 1979; Laguitton, 1980. Outros utilizam as rotinas de busca direta: White, 1976, 1977; White e Winslow, 1974, 1975, 1979; Winslow, 1980; Hockings e Callen, 1977; Tipman, 1979; Hodouin e Everell, 1979/1980 e Hodouin, 1981 (06).

A técnica que utiliza os multiplicadores de Lagrange foi introduzida por Nielson e Diaz, em 1970, referidos por (11), que descreveram um método para ajustar os dados para os balanços de massas e metalúrgicos. Empregaram a equação (16) com um ponderador igual a k / Sri^2, onde k é um fator de ponderação com valor zero ou um; e Sri^2 é uma estimativa da variância do erro associada com o i-ésimo ponto de dado. Os multiplicadores de Lagrange foram introduzidos para localizar os extremos de uma função objetivo diferenciável com o número de multiplicadores igual ao número de equações de restrição. O procedimento foi o seguinte:

(1) ajustar as vazões, até que um ajuste mínimo ocorra. Caso não se tenha o valor de um fluxo, pode se atribuir um peso zero (k=0) e o método calcula o valor da vazão naquele ponto.

(2) Fechar o balanço de massas, usando as vazões inicialmente calculadas.

(3) Fechar o balanço de cada componente.

Para o primeiro passo, são utilizados os valores experimentais das análises do componente, e as vazões experimentais são ajustadas. A função objetivo para minimizar foi dada:

$$(21)\quad \emptyset 1 = \sum_{j=1}^{b} k\,((\hat{w}j' - Wj)^2 / Sj^2) - \sum_{i=1}^{n} \lambda i \sum_{j=1}^{b} mj\, Xji\, \hat{w}j' \ ,$$

onde λi é um multiplicador para o i-ésimo balanço, Xji é a análise do componente i no fluxo j, Sj2 é a variância associada com o valor da vazão Wj para o j-ésimo fluxo, $\hat{w}j'$ é a primeira estimativa da vazão para o j-ésimo fluxo, e mj é $\pm$ 1 (-1 se o fluxo é uma saída e +1 se o fluxo é uma entrada). Derivadas parciais de $\emptyset$ 1 em relação a λi, i=1....n e Wj, j=1....b são determinadas e igualadas a zero. As equações resultantes são resolvidas, simultaneamente, para obter as estimativas iniciais $\hat{w}j'$.

O segundo passo, o balanço de massas, é feito para minimizar:

$$(22)\quad \emptyset 2 = \sum_{j=1}^{b} k\,((\hat{w}j - \hat{w}j')^2 / Sj^2) - \lambda o \sum_{j=1}^{b} rj\, \hat{w}j \ ,$$

onde λo é um multiplicador, e $\hat{w}j$ será a tonelagem ajustada no j-ésimo fluxo.

O terceiro passo é análogo ao passo dois, exceto que a função objetivo para o i-ésimo balanço de elemento é:

$$(23)\quad \emptyset i = \sum_{j=1}^{b} k\,((\hat{x}ji - Xji)^2 / Sj^2) - \lambda i \sum_{j=1}^{b} rj\, \hat{x}ji\, \hat{w}j \ ,$$

onde Xji é a análise do elemento i no fluxo j, $\hat{x}ji$ será o valor

ajustado final correspondente, e Sji^2 é a variância do erro correspondente. Xji é determinado para cada balanço, i=1,2,...,n pela minimização de Øi.

Devido a cada valor de análise e tonelagem medida ser sujeito a erros, os balanços de elementos são essencialmente não-lineares. Wiegel (12) utilizou uma técnica tal que F poderia ser minimizada, usando os Multiplicadores de Lagrange, linearizando de forma aproximada os termos do produto nas equações de balanço de massas, que incluiam os erros. Uma solução aproximada foi obtida, dada pela equação (24), e desenvolvida em uma forma iterativa para qualquer precisão que fosse desejada:

$$(24)\quad W1^* X1^* \simeq W1\,X1 + X1\,DW1 + W1\,D\,X1 \ ,$$

$$\text{onde } D\,W1 = W1^* - W1 \ ,$$

$$D\,X1 = X1^* - X1 \ .$$

Wiegel, em 1972, desenvolveu um programa de computador de objetivo geral, baseado em seu método, para resolver os problemas de balanço de massas. O programa foi denominado MATBAL e foi escrito em FORTRAN IV, para uso em Sistema Control Data 6600. Atualmente, tem-se a versão MATBAL3 (14), escrito em FORTRAN 77.

As informações de entrada para cada cálculo continham: título, identificações de fluxo, valor das medidas, balanço de massas global e individual e as medidas que devem ser mantidas constantes. Aquela época a entrada era feita com cartões perfurados, atualmente, na versão para microcomputadores é feita via teclado. A impressão de saída inclui um sumário das

informações de entrada, uma tabela para cada material balanceado, mostrando o nome do fluxo, valores experimentais, valores ajustados, porcentagem de pesos, componentes: produto e análises ajustadas, e porcentagem de material recuperado naquele fluxo.

O método dos multiplicadores de Lagrange foi usado para minimizar F de um modo diferente por Smith e Ichiyen (15), em 1973.

Este método usa um procedimento numérico para os balanços de massas e metalúrgico computadorizado, baseado no tratamento estatístico de todos os dados disponíveis, com uso "off-line" ou "on-line". Estima a estrutura do balanço de massa do processo a partir dos dados de entrada, usando o maior número possível de valores disponíveis.

Pode ser utilizado tanto a nível de laboratório, quanto em escala industrial.

O λ_i foi substituído na equação (23), pelo ajuste das análises, DXji, buscando-se a solução (15). Se uma estimativa inicial de um ensaio é denotado por Xji, então uma melhor estimativa será $\hat{X}ji = Xji + D\,Xji$. Xji é trocada por $\hat{X}ji$ e o método de busca direta de Rosenbrock é aplicado para encontrar um conjunto de valores de Wj que minimiza Øi. O método de Rosenbrock consiste de estágios, tendo uma minimização uni-dimensional, as direções de busca são giradas, mantendo-se a ortogonalidade, para que a primeira busca do estágio seguinte obtenha um valor convergente. Um novo conjunto de valores de D Xji é determinado e o procedimento repetido para adequar a convergência. A busca termina quando o mínimo é localizado, para

uma precisão pré-especificada.

Um outro procedimento numérico para uso "off-line", com grande número de variáveis foi descrito em 1976, estimando a estrutura do balanço de massas do processo a partir dos dados de entrada, utilizando o maior número de dados possível (16).

O método é generalizado para vazões, análises químicas, distribuições granulométricas e outros parâmetros. O número de medidas de cada ponto, usualmente é menor do que o conveniente para se avaliar com precisão a distribuição de erros, portanto, Cutting e Watson (16) assumiram que os erros teriam distribuição normal e que o método dos mínimos quadrados poderia ser aplicado, sujeito às restrições definidas pelo fluxograma de processos. Os fatores de ponderação são usados para definir os níveis de confiança associados a uma medida, em particular.

Definiram o fluxograma de processos por uma matriz M , de ordem m x n, onde m é o número de fluxos e n é o número de nós identificados (junções no fluxograma), cujos valores tomam a forma 0, +1 e -1, dependendo da existência e direção do fluxo, em um nó particular, convencionalmente:

mij = + 1 para fluxos j entrando no i-ésimo nó ,

- 1 para fluxos j saindo do i-ésimo nó ,

0 para fluxos j não se ligando ao i-ésimo nó .

As restrições são definidas pela equação matricial WM=0, onde W é um vetor m-coluna dos valores experimentais das vazões de m fluxos; o método dos multiplicadores de Lagrange é usado para gerar a função objetivo, dada pela equação (22) e obter o mínimo da função.

Hockings e Callen, citados em (11), empregaram o Método de Lagrange para um circuito geral, usando uma notação de matriz incidência, com as unidades de processos, chamadas de nós, e os respectivos fluxos. Sua função objetivo foi escrita como:

$$(25)\quad \Phi = \sum_{k=1}^{n} \sum_{j=1}^{b} P_{jk} \left((\hat{x}_{jk} - X_{jk}) / X_{jk} \right)^2 + \sum_{i=1}^{p} \Phi_i \sum_{j=1}^{b} m_{ij} \hat{w}_j + $$

$$+ \sum_{k=1}^{n} \sum_{i=1}^{p} \lambda_{ik} \sum_{j=1}^{b} m_{ij} \hat{x}_{jk} \hat{w}_j + \sum_{j} \gamma_j \hat{w}_j \quad ,$$

onde Pjk são os fatores de ponderação, mij são +1 ou -1 a partir da matriz do fluxograma, e os símbolos gregos são os multiplicadores. Os valores $\hat{w}_j$ no segundo e quarto termos da equação são tratados separadamente, similarmente a Nielson e Diaz. Se qualquer dos $\hat{w}_j$ é fixado, tais como a vazão na alimentação, um subconjunto, que é o das vazões experimentais, é determinado. Os valores iniciais para o subconjunto são selecionados e os $\hat{w}_j$ são calculados para gerar um novo conjunto, que é temporariamente referido como constante. A seguir, a equação é diferenciada em relação a cada $\hat{x}_{jk}$ e a cada um dos multiplicadores λ_{ik}. Cada equação é igualada a zero. O conjunto de equações é resolvido simultaneamente e o resultado é o conjunto dos valores $\hat{x}_{jk}$. Estes são agora a melhor estimativa. Os independentes $\hat{w}_j$ vêm a ser variáveis de busca para a rotina de busca de Powell. Cada avaliação da função objetivo, chamada pela busca, usa estimativas de $\hat{x}_{jk}$ encontradas a partir de equações, até que Φ é minimizada, com os adequados valores.

Este método, embora generalizado para qualquer circuito, é realmente similar àquele proposto por Smith e Ichiyen (15), exceto que as medidas de vazão podem ser introduzidas como constantes e a busca de Powell substitui a busca de Rosenbrock.

Segundo Lynch (04), as técnicas de ajuste de balanço de massa são modos de distribuir os erros de balanço di entre os vários valores medidos para dar valores (Ŵ1i,Ŵ2i,Ŵ3i) ajustados, que são numericamente consistentes com uma vazão calculada. Lynch discutiu as técnicas de minimização da soma dos quadrados dos resíduos e de minimização pelo método dos multiplicadores de Lagrange para uma função objetivo, baseada na equação (6). Contudo, concluiu que sua aplicação só é possível quando as medidas de componentes são disponíveis. O método consiste em encontrar, num primeiro estágio, as vazões relativas, para posteriormente, fazer o ajuste. Mas, em alguns casos, este procedimento não é iterado de modo suficiente.

O Julius Kruttchnitt Mineral Research Centre, da Universidade de Queensland, Austrália, desenvolveu os "softwares" JKMbal (17) e o JKSimMet (18), para ajuste de balanço de massas, baseado no método desenvolvido por Lynch.

Black, citado em (11), empregou a rotina de busca direta de Hooke-Jeeves para minimizar a equação (22) e ajustar as medidas de fluxo de massas de sólidos e água em um banco de células de flotação. As principais características do método foram o emprego da busca direta pela média das variáveis de busca e o fator de ponderação P, da equação (22), foi feito

inversamente proporcional ao quadrado dos valores medidos.

White et al. (10) projetaram um procedimento de ajuste de dados de balanço de massas, empregando o método de busca direta, baseado na rotina de COGGIN.

O procedimento ajusta automaticamente os valores dos dados para minimizar a função dos mínimos quadrados ponderada, de modo que os dados sejam ajustados o mínimo possível.

A função objetivo do procedimento de White é dada pela equação:

$$(26) \qquad SSQ = \sum_{i=1}^{n} \left[\frac{Fi \; (\; Wi - \hat{W}j \;)}{Wi} \right]^2 \; .$$

White escreveu, baseado no seu método, o programa SIMUMIL , que usa rotinas de busca direta, mas não é um método genérico, isto é, tem que ser projetado para cada uso particular, ou "doméstico".

O SIMUMIL pode ajustar até 19 nós e no máximo 50 fluxos.

A saída do SIMUMIL é um sumário dos cálculos, mostrando os valores experimentais e os ajustados para sólidos, água e porcentagem de sólidos em cada fluxo.

Tipman et al., citados por (11), descreveram o procedimento dos mínimos quadrados, para m avaliações múltiplas. A função objetivo foi escrita da seguinte forma:

$$(27) \qquad F = \sum_{i=1}^{m} \left(\sum_{k=1}^{n} \sum_{j=1}^{b} (Pj \; Xjk \; Wj)^2 + \sum_{j=1}^{b} (Pj \; Wj)^2 \right)^1 \; .$$

A função F é minimizada, usando um pacote IBM, programa para encontrar o melhor conjunto de vazões iniciais Wj, e

componentes Xjk, que dão o Fmin. Os fatores de ponderação Pj podem ser introduzidos como desejados.

Esta técnica não permite a introdução de variância dos erros associados às medidas individuais: a rotina IBM é que as calcula.

Hodouin e Everell (19) desenvolveram um procedimento que usa somente algumas das idéias encontradas na literatura, estende suas aplicações para uma solução geral do problema e o formula de um modo sistemático.

Escreveram o circuito de processamento mineral em termos de uma rede de fluxos e nós, a fim de formular as equações gerais de conservação e fluxo de material. Um fluxo corresponde a um elemento de processo, no qual somente ocorre transporte de material, e um nó é o local onde vários fluxos se conectam.

O procedimento proposto, por Hodouin e Everell (19), é minimizar a soma ponderada dos resíduos do fechamento do balanço, elevados ao quadrado e consiste em um programa de controle, trabalhando em um conjunto de variáveis independentes necessárias para definir todas as vazões de minério. Com este algoritmo, as variáveis são separadas em dois conjuntos: o das vazões e o dos componentes, cada conjunto correspondendo a parte do critério da soma dos quadrados para ser minimizado. O mínimo da parte dos componentes desta função pode ser encontrado diretamente desde que as restrições sejam lineares e a função de segundo grau. A parte das vazões é minimizada usando o algoritmo padrão de Powell, que é também usado como um algoritmo de controle. As restrições do balanço de material são escritas,

usando formalismo de matriz e o fluxograma descrito do circuito.

O método desenvolvido por Powell, que requer computação analítica, foi selecionado a fim de obter uma solução geral e simples para o problema sem introduzir rigidez no algoritmo. Foi utilizado na sua versão programada dada por Kuester e Mize, segundo Hodouin e Everell (19).

As estimativas iniciais das variáveis de busca das vazões são as vazões experimentais. As estimativas das variáveis de busca das vazões são usadas somente na primeira vez que o algoritmo é usado. Então, as estimativas iniciais das variáveis de busca são tomadas como os valores de solução do cálculo prévio, portanto, a convergência do algoritmo é mais rápida e os valores iniciais das variáveis de busca podem ser só grosseiramente estimadas.

Hodouin et al. (20) escreveram um programa de computador denominado de BILMAT, que atualiza os dados do processamento mineral, baseado no método descrito por eles, satisfazendo as restrições de conservação de massa; é aplicado para qualquer tipo de circuito de beneficiamento e pode utilizar vários dados do processo.

O programa BILMAT calcula variáveis não mensuráveis, tais como vazões em pontos inacessíveis, simultaneamente usa todos os dados experimentais e satisfaz todas as restrições de balanço de massas e corrige os dados medidos de acordo com o modelo de erros assumido. É um programa bastante complexo, trabalhando com vários níveis de balanço, o de massa e o metalúrgico.

O BILMAT foi programado em APL e FORTRAN e uma versão simplificada foi implementada em BASIC para microcomputador. As versões APL e BASIC são plenamente conversacionais e podem ser usadas por profissionais, que não tenham familiaridade com a matemática dos programas. A programação é geral, tal que qualquer tipo de circuito pode ser tratado. A estrutura é de uma forma modular, que permite modificações rápidas e adições ao programa.

O programa possui uma ajuda ao usuário, com uma série de explanações sobre a maneira de usá-lo e com a base da documentação. Inicializando, ele questiona sobre as medidas executadas (análises químicas, medidas de densidade de polpa, análise granulométrica, vazões, etc.). Após isto, o circuito deve ser descrito, esquematicamente, como uma rede de nós e fluxos. Uma sub-rotina constrói a matriz equivalente desta rede. Faz-se a distinção dos fluxos que são somente de água e só de minério. Os próximos módulos perguntam quais fluxos onde as medidas dos componentes foram feitas. Há possibilidade de definir até 5 sub-redes do circuito.

Então, o programa solicita os valores das medidas e as variâncias estimadas, associadas a cada uma delas. Antes de iniciar os cálculos computacionais, o programa necessita das estimativas iniciais das vazões experimentais. Em seguida, o operador introduz as variáveis de controle para os algoritmos de minimização, baseado na rotina de Powell: precisão requerida, máximo número de iterações, saída de impressões intermediárias. Após a execução dos cálculos computacionais estar completada, o usuário pode selecionar os resultados que deseja impressos,

inclusive com uma análise de sensibilidade, para avaliar a confiabilidade dos resultados.

Wills e Manser (05) propuseram um método para a conciliação do excesso de dados de uma usina, isto é, aproveitar os dados disponíveis, tais como análises granulométricas do material, taxas de diluição e análises químicas para calcular o melhor valor de Wj, dado pela equação (16).

Com todos os dados disponíveis, o balanço de massas e seu ajuste podem ser calculados por várias formas. Eles discutiram e compararam os resultados obtidos com os dois métodos basicos, comumente adotados, o método de minimização da soma dos quadrados dos resíduos do fechamento do balanço de massas e o método dos mínimos quadrados, da soma do ajuste dos componentes, e verificaram uma pequena diferença entre os resultados dos métodos, a qual não teria um significado prático real.

Propuseram um método alternativo, para a aproximação pelos mínimos quadrados, que é obter o melhor ajuste da estimativa de Wj, a partir da média ponderada dos valores calculados dos componentes individuais Xj disponíveis. O fator de ponderação utilizado é baseado na variância nos cálculos de Wj, a partir de cada equação do componente. A técnica, embora descrita para um nó simples, pode ser aplicada para qualquer circuito, até de configuração complexa, desde que facilidades computacionais estejam disponíveis.

Outros dois programas de ajuste de balanço de massas foram desenvolvidos, baseados no método dos mínimos quadrados, com utilização dos multiplicadores de Lagrange (21):

- O primeiro programa, chamado HANDBAL, foi desenvolvido para calculadoras HP, é composto somente de uma rotina para calcular os valores de análise ajustados correspondentes aos valores experimentais, a busca é feita de uma forma manual.
- O segundo programa, chamado SINBAL, foi escrito em BASIC+, inclue uma rotina de busca automática para calcular o valor da vazão ajustada. Este programa requer pequena memória de computador e pode facilmente ajustar-se em sistemas de microcomputador.

Outro "software" disponível é o METSIM, desenvolvido pela PROWARE , de Lakewood, Colorado-EUA (22). é um sistema de simulação aplicado para microcomputador, que possue um módulo de Balanço de Massas.

No Departamento de Engenharia de Minas da Escola Politécnica da Universidade de São Paulo, dois "softwares" para ajuste do balanço de massas foram desenvolvidos. O primeiro, baseado no algoritmo do MATBAL, é um programa em FORTRAN, que teve sua aplicação na Serrana Mineração (06), foi desenvolvido para microcomputador. O segundo, baseado na técnica de Cutting e Watson, foi escrito em BASIC e permite o ajuste dos dados do balanço de massas e metalúrgico, pelo método dos mínimos quadrados, com aplicação dos multiplicadores de Lagrange. O fator de ponderação utilizado é o inverso da variância de cada ponto de medida (23).

III. DESCRIÇÃO BÁSICA DO MÉTODO DE AJUSTE PROPOSTO

Os ajustes de valores medidos têm que ser feitos de modo que todas as equações do balanço sejam satisfeitas e os ajustes totais sejam mínimos. Para que as informações possam ser processadas por computador, é necessário expressar essas duas condições de forma matemática.

A interpretação matemática deve começar pelo fluxograma que, juntamente com os valores experimentais e com o modelo de erros adotado, irão formular o problema para o qual busca-se a solução, representando o melhor ajuste para os dados.

III.1. Tratamento matemático

O primeiro passo é descrever o fluxograma de processo de tratamento, usando a matriz M de dimensões n x m (nº de operações x nº de fluxos), cujos elementos mij são +1 , -1 ou 0 (zero), indicando se o j-ésimo fluxo entra, sai ou não se liga à i-ésima operação, respectivamente, e m = número de parâmetros de fluxo e n=número de operações no fluxograma.

Assim, o conjunto de equações de balanço de massa em cada operação, que descreve o processo, pode ser expresso na forma matricial:

(28) $$W\,M = 0 \quad ,$$

onde, W é um vetor m-linha dos parâmetros experimentais do processo.

As equações da regra dos n produtos poderão ser escritas da forma matricial com os valores ajustados para zerar o balanço de massas.

$$(29) \quad \sum_{i=1}^{n} \sum_{j=1}^{m} mij \; \hat{W}j = 0 \; ,$$

onde $\hat{W}j$ são os valores ajustados das vazões no fluxo j.

Se forem analisados k componentes, o balanço metalúrgico em cada operação i, para cada componente k, é:

$$(30) \quad \sum_{i=1}^{n} \sum_{j=1}^{m} mij \; (Xj^{k} \; \hat{W}j) = 0 \; .$$

O procedimento proposto é um algoritmo para fazer o ajuste tanto do balanço de massas, quanto do balanço metalúrgico, utilizando o método dos mínimos quadrados.

Para facilitar a análise e programação do procedimento proposto, a função dada pela equação (16) será simplificada, pelo critério para minimização da função objetivo, dado por Hodouin e Everell (19), que foi dividi-la em duas partes, uma das quais contendo as variáveis de vazões e a outra as variáveis dos componentes, teores e granulometria, como pode ser visto na equação (31). Isto é possível, desde que os erros nas medidas sejam independentes, então tem-se:

$$(31) \quad F = F(W) + F(X) \; .$$

Portanto, a função objetivo global pode ser escrita da seguinte forma:

$$(32) \quad F = \sum_{i=1}^{n} Pi \; (\hat{r}i - ri)^{2} \; ,$$

onde: Pi é o fator de ponderação associado a ri, o i-ésimo ponto de dado medido e $\hat{r}i$ é o correspondente ponto de dado ajustado.

A partir da equação genérica (32), para o cálculo de ajuste dos dados de vazão e de teores, sendo o elemento generalizado ri usado para o balanço de massas, onde ri é wj, e para o balanço metalúrgico, onde ri é Xj.Ẅj, chega-se à equação:

$$(33) \qquad Fn = \sum_{i=1}^{m} \sum_{j=1}^{k} Pij \; (\hat{r}i - rij)^2 \quad ,$$

onde,

rij = valor experimental do parâmetro de ri no fluxo j,

$\hat{r}i$ = valor ajustado de ri,

Pij = peso de rij, ou seja, peso de ri no fluxo j.

A função dada pela equação (33) é uma estimativa, pelo método dos mínimos quadrados, com os fatores de ponderação Pij sendo usados para definir os níveis de confiança associados a cada medida particular. Impondo as restrições de processos definidas pelo fluxograma, matriz M, e pelas equações (28) e (29) para o balanço de massas e, (28) e (30) para o balanço metalúrgico, e usando os Multiplicadores de Lagrange para gerar a função Lagrangeana, para minimizar o método dos mínimos quadrados, chega-se à equação matemática para o i-ésimo parâmetro ri :

$$(34) \qquad Fn = \sum_{i=1}^{m} \sum_{j=1}^{k} Pij \; (\hat{r}i - rij)^2 + RM.\underline{\lambda} \quad ,$$

onde:

$\underline{\lambda}$ = um vetor n-coluna dos multiplicadores de Lagrange.

Diferenciando a equação (34) em relação ao valor ajustado $\hat{r}i$ e aos multiplicadores de Lagrange, gera-se um

conjunto de m+n equações simultâneas (16), que podem ser organizadas na forma matricial:

(35) $$A\,S = Y \quad ,$$

onde:

$$A = \begin{bmatrix} B & M \\ M' & O \end{bmatrix} \qquad S = \begin{bmatrix} R \\ \lambda \end{bmatrix} \qquad Y = \begin{bmatrix} Q \\ O \end{bmatrix}$$

$$B = \begin{bmatrix} \sum_{j=1}^{k1} E\,2P1j & & & O \\ & \sum_{j=1}^{k2} E\,2P2j & & \\ & & \ddots & \\ O & & & \sum_{j=1}^{km} E\,2Pmj \end{bmatrix}$$

$$R = \begin{bmatrix} r1 \\ r2 \\ \vdots \\ rm \end{bmatrix} \qquad \lambda = \begin{bmatrix} \lambda 1 \\ \lambda 2 \\ \vdots \\ \lambda n \end{bmatrix} \qquad Q = \begin{bmatrix} \sum_{j=1}^{k1} E\,2P1j\,r1j \\ \sum_{j=1}^{k2} E\,2P2j\,r2j \\ \vdots \\ \sum_{j=1}^{km} E\,2Pmj\,rmj \end{bmatrix} .$$

A matriz A, de dimensões [(m+n) x (m+n)], é dada pela matriz diagonal B, (m x n), com os ponderadores multiplicados pelo fator 2, proveniente da derivada da função quadrática, com a matriz M, (m x n), do fluxograma, completando as colunas, e a inversa de M, M', de dimensões (n x m), completando as linhas. Os elementos $a_{m+1,m+1}$ até $a_{m+n,m+n}$ são zerados, para tornar a matriz retangular e ser possível solucionar o sistema gerado.

O vetor solução S, de dimensões [(m+n) x 1], contém um vetor m-coluna dos valores ajustados $\hat{f}i$ e um vetor n-coluna dos multiplicadores de Lagrange, que podem ser considerados como variáveis de recurso matemático, no número de n operações, para propiciar a solução do sistema e como tal, não têm valor em termos de processo mineral.

O vetor (m+n)-coluna dos termos independentes Y é formado pelo vetor m-coluna dos valores experimentais, multiplicados por duas vezes o fator de ponderação, e pelo vetor n-coluna de valores nulos.

A equação matricial , dada pela equação (35), é resolvida pelo método de Eliminação de Gauss para os valores $\hat{f}i$ e λi .

III.2 Rotina de Solução

Entrada dos dados:

- nº de operações — n ;
- nº de fluxos — m ;
- restrições — matriz M ,
- valores experimentais — média de vazão e desvio-padrão, média de teor e desvio-padrão. Opção: do modelo de erros de ser feito com o grau de certeza ou incerteza.

Para facilitar, a entrada de dados é feita atráves de uma só planilha, colocando-se para cada operação o valor da média da vazão do fluxo ligado a ela, com o seu respectivo sinal de entrada ou saída e o respectivo desvio-padrão. Para o teor é mantido o mesmo procedimento.

Sub-rotinas de Cálculos :

1. Zera a matriz A;
2. calcula os pesos Pij;
3. calcula os valores da matriz diagonal B

$$bii = \sum_{j=1}^{ki} 2\,Pij \;;$$

4. calcula valores da matriz Y

$$yij = \sum_{j=1}^{ki} 2\,Pij * ri \;;$$

5. resgata os valores da matriz M e calcula a M';
6. monta a matriz A;
7. triangulariza a matriz A, pelo método de Eliminação de Gauss;
8. calcula a solução da matriz triangularizada;
9. dá a resposta do Lagrangeano e fi.

III.3 Descrição do Diagrama de Fluxo de Dados (DFD)

O Sistema de Ajuste de dados objeto desta dissertação foi desenvolvido em linguagem C, de quarta geração, e é formado por várias rotinas/procedimentos. O compilador utilizado foi o Turbo C. Trata-se de uma linguagem de grandes recursos, permite uma melhor estruturação do programa, facilitando a manutenção do sistema, gera executáveis menores e de velocidade de processamento maior, o Turbo C permite gerar programas com o dobro da velocidade de processamento do Quick Basic (24). Possui maiores recursos gráficos e matemáticos, com maior precisão em ponto flutuante e permite a criação de funções pelo usuário. Ainda como recursos, a linguagem C tem uma portabilidade muito grande para sistemas operacionais Windows, Unix, CPM e Risk, facilidade de acessar a BIOS (Sistema de entrada/saída básico) do computador.

O Diagrama de Fluxo de Dados (DFD), figura 4, mostra o esquema de operação do programa de ajuste de Balanço de Massas, chamado BALMAS.

A primeira etapa é fazer a opção por consultar um balanço de massas já feito para uma usina, buscando no banco de dados, ou por iniciar novos cálculos, optando pelo ajuste do balanço de massas ou metalúrgico.

O passo seguinte é definir a usina, através do seu fluxograma, interpretado pela matriz M. A entrada dos dados do fluxograma é feita utilizando a forma de planilha, já simulando a matriz. O sistema está, inicialmente, dimensionado para 30

operações e 50 fluxos, mas a linguagem C permite uma fácil e rápida alteração do tamanho desta matriz.

O valor do elemento mij, ou seja, o valor do fluxo j, ou do componente, na operação i é o seu valor experimental com o sinal de + ou −, caso o fluxo entre ou saia do ponto de operação, respectivamente, ou zero se não se ligar a esse ponto. A montagem da matriz M, restrições, é feita automaticamente pela divisão do valor mij pelo seu valor absoluto, gerando os valores +1 e −1, a serem utilizados na matriz A. Com este procedimento, facilita-se a entrada de dados e as eventuais correções pelo usuário, permitindo uma visão ampla de todos os valores experimentais.

Os dados podem já estar calculados, com média e desvio-padrão, ou utiliza-se a rotina/procedimento de cálculo de média e desvio padrão, para prepará-los.

O ponderador Pij para o "software" proposto fica a critério do usuário. Poderão ser usados os desvios-padrão das medidas experimentais ou, então, dependendo do nível de conhecimento da tomada de medidas o usuário poderá optar pelo grau de certeza ou incerteza, medida em porcentagem, por exemplo, desvio-padrão dividido pela respectiva média, o que em muitos casos pode representar uma ponderação mais apurada.

Após a confirmação da entrada dos dados e do modelo de erros adotado, o programa calcula o ajuste dos dados, cujos resultados são armazenados, sob a forma de planilhas, referentes a cada unidade ensaiada, de modo que facilite sua futura consulta e utilização para um novo cálculo.

O usuário optará pela forma de impressão dos resultados, o que facilitará a análise do processo e eventual interferência na usina.

Para instalação do sistema é necessário um "hardware" com a seguinte configuração:

- MINIMA : PC XT,
  disco flexível,
  monitor monocromático, placa CGA,
  impressora (opcional).

- ADEQUADA : mínimo PC AT 286,
  disco rígido,
  disco flexível,
  co-processador aritmético,
  monitor colorido, placa EGA,
  impressora (opcional).

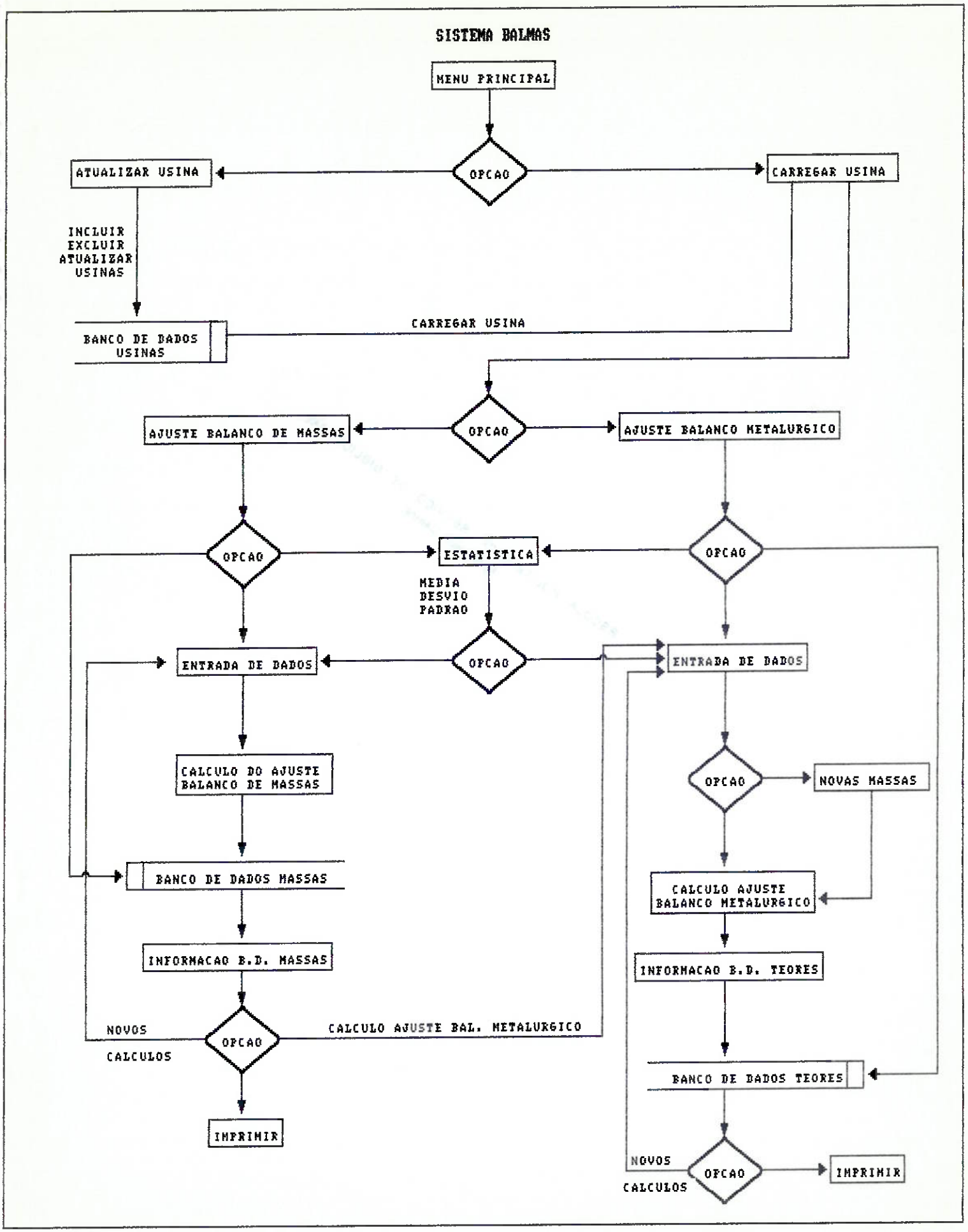


Figura 4 — DIAGRAMA DE FLUXO DE DADOS DO SISTEMA DE AJUSTES DO BALANCO DE MASSAS E METALURGICO

IV. ESTUDO DO CASO DA MINA DO PITINGA

O Pitinga é uma província mineral polimetálica localizada 280 km a N de Manaus, no Estado do Amazonas, como mostrado na Figura 5. A Mina do Pitinga é operada pela Mineração Taboca há vários anos. Está situada em uma região tropical, com chuvas abundantes durante seis meses do ano e relevo marcado por intrusões graníticas, de desníveis de até 400 m (25).

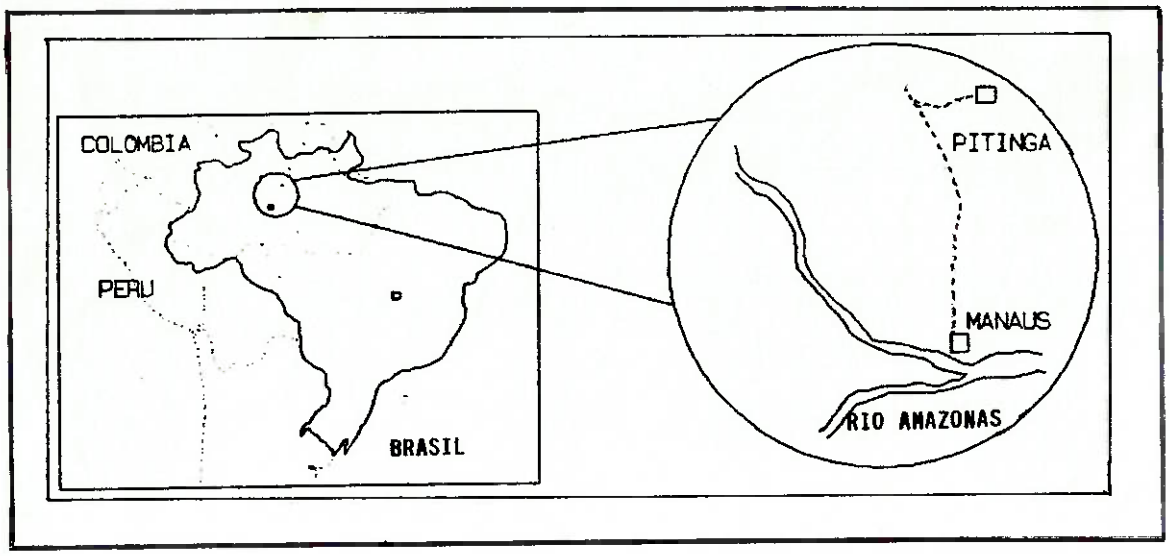


Figura 5 : Localização da Mina do Pitinga

A operação foi iniciada em 1982, no Igarapé Queixada, em lavra unicamente aluvionar. Atualmente, são lavrados os depósitos aluvionares e eluvionar, com uma produção , em 1991, de 16.500 t de estanho contido (25), o que representa cerca de 14% da produção mundial.

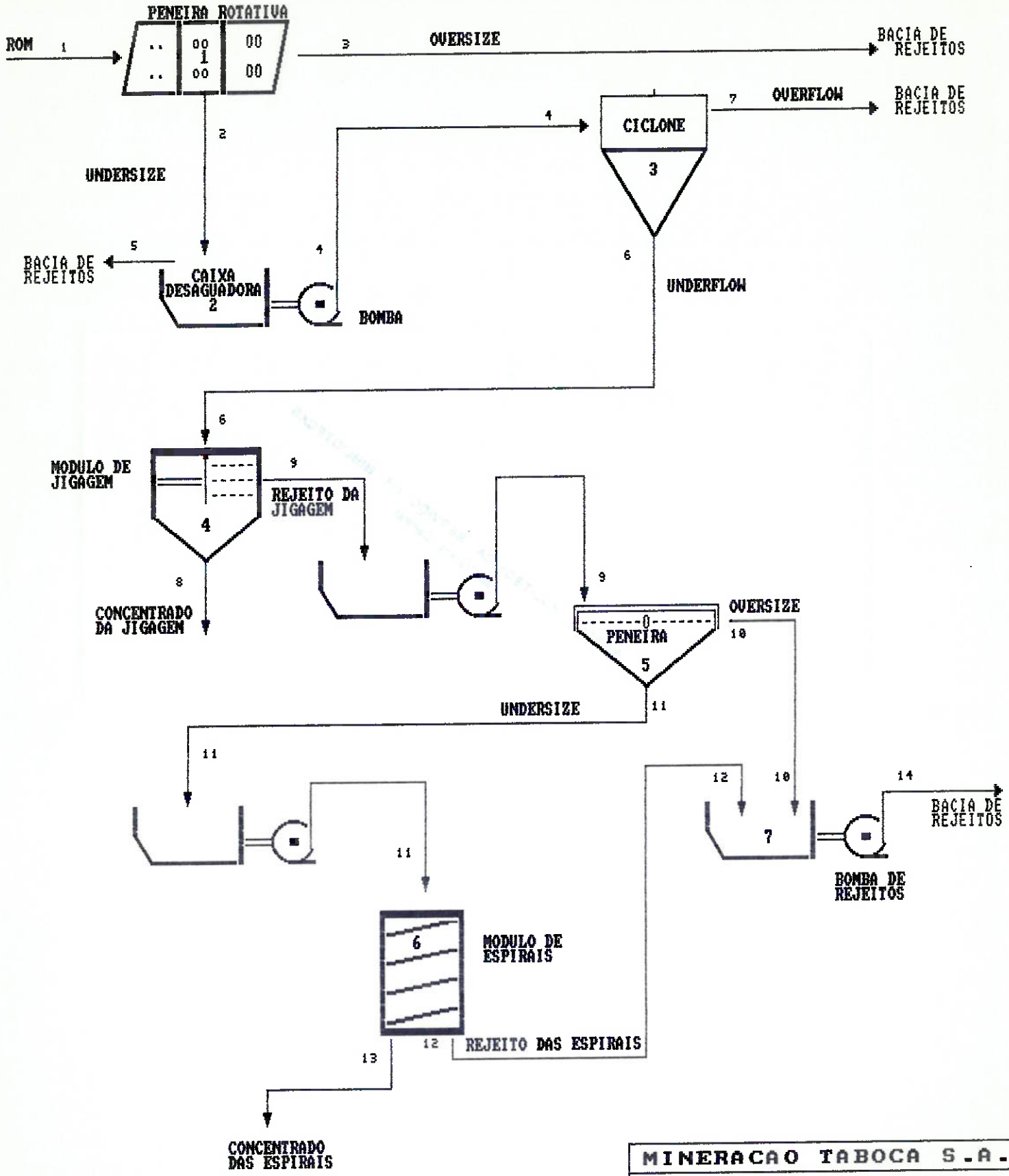
Todo o processamento para os materiais aluvionares e eluvionar é essencialmente físico (gravítico e separações eletromagnética e eletrostática), sem qualquer reagente químico. O processamento densitário é todo baseado em jigues e espirais.

Para evitar problemas de processamento específicos dos

depósitos e dos métodos de lavra diferentes, todas as unidades de lavra têm sua unidade de pré-concentração, cada uma das quais produz um pré-concentrado, que é levado por caminhões para uma usina de beneficiamento central. O produto das diferentes unidades de pré-concentração é processado e um concentrado final de alto teor é produzido. Esta usina de beneficiamento central utiliza jigues, mesas vibratórias e separadores magnéticos e eletrostáticos.

A usina objeto deste estudo será a de pré-concentração do material eluvionar. Estão operando seis unidades, denominadas de PL, que tratam o material proveniente do depósito eluvionar da Serra do Madeira. O fluxograma, dado pela Figura 6, é idêntico para todas as unidades e é composto esquematicamente por 7 operações e 14 fluxos, portanto, a matriz M é igual para todas as PL's.

O material ROM alimenta uma peneira rotativa, cujo "oversize" é rejeito final e cujo "undersize" vai para uma caixa desaguadora; o "overflow" é descartado e o "underflow" é bombeado para um ciclone. O "overflow" do ciclone é rejeito final e o "underflow" alimenta o módulo de jigagem, cujo concentrado é desaguado em classificador espiral e já é produto desta usina; o rejeito deste módulo sofre um peneiramento em uma peneira vibratória horizontal, onde o "oversize" é rejeito e o "undersize" alimenta um módulo de espirais, cujo rejeito junta-se com o "oversize" da peneira e forma mais um fluxo de rejeito final e o concentrado é desaguado em um classificador espiral e é produto desta usina.



MINERACAO TABOCA S.A.

MINA DO PITINGA

FLUXOGRAMA DA USINA DE BENEFICIAMENTO DO MINERIO ELUVIONAR PL's

FIGURA 6

FBALMAS

O diagrama de pontos da Figura 7 resume as operações e os fluxos, para melhor compreensão do fluxograma, caracterizando 7 operações e 14 fluxos.

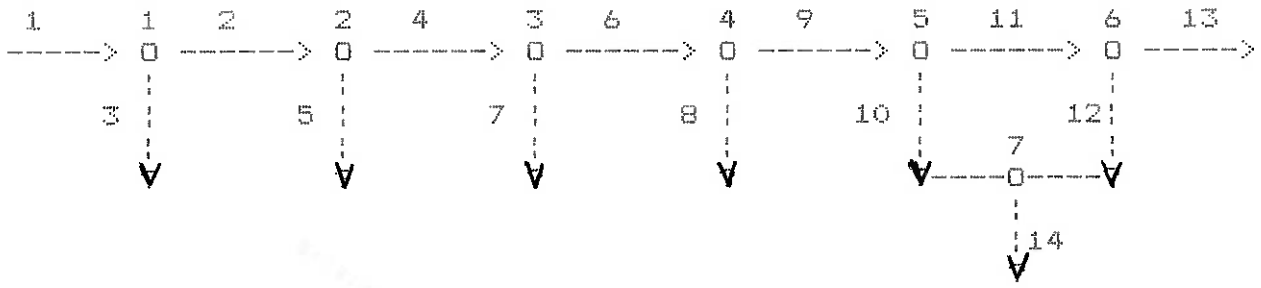


Figura 7: Diagrama de pontos do fluxograma das PL's

A matriz M é dada pelas seguintes restrições:

-operação 1: peneira rotativa - fluxos 1 - 2 - 3 = 0

-operação 2: caixa desaguadora - fluxos 2 - 4 - 5 = 0

-operação 3: ciclone - fluxos 4 - 6 - 7 = 0

-operação 4: módulo de jigagem - fluxos 6 - 8 - 9 = 0

-operação 5: peneira vibratória - fluxos 9 -10 -11 = 0

-operação 6: módulo de espirais - fluxos 11 -12 -13 = 0

-operação 7: caixa de junção de rejeitos- fluxos 10 +12 -14 = 0 .

A planilha da matriz M, referente ao fluxograma da usina PL é dada pela Figura 8.

fluxos / operações	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	-1	-1	0	0	0	0	0	0	0	0	0	0	0
2	0	1	0	-1	-1	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	-1	-1	0	0	0	0	0	0	0
4	0	0	0	0	0	1	0	-1	-1	0	0	0	0	0
5	0	0	0	0	0	0	0	0	1	-1	-1	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	-1	-1	0
7	0	0	0	0	0	0	0	0	0	1	0	1	0	-1

Figura 8 - Matriz M para as usinas PL's

Os pontos de amostragem, em que se baseou o balanço de massas, foram o "oversize" da peneira rotativa, o "overflow" da caixa desaguadora, o "overflow" do ciclone, a alimentação e o concentrado do jigue, a alimentação e o concentrado da espiral, o "oversize" da peneira e o rejeito do módulo das espirais. Foram feitas 12 séries de amostragens, para os fluxos de cada PL.

Será descrito apenas o caso da PL 01. Os valores da vazão, com respectivas médias e desvios-padrão, são dados na Tabela 01, e os valores de teor de SnO_2, na Tabela 02.

A planilha de entrada dos dados para o ajuste do balanço de massas é dada pela Tabela 03. O fluxo 1 foi tomado como base, portanto, sua média foi 100 e o desvio-padrão zero. Foi adotado para o cálculo, o desvio-padrão de 1%, pois não é possível trabalhar com zero.

A planilha de entrada dos dados para o balanço metalúrgico é dada pela Tabela 04.

As planilhas de entrada representam a matriz M', baseado nas restricões dadas pela Figura 8.

Os valores calculados, ajustados para o balanço de massas, são apresentados na Tabela 05 e os ajustados para o balanço metalúrgico pela Tabela 06.

A planilha dada pela Tabela 07 faz a comparação entre os valores experimentais e os calculados do balanço de massas. Observa-se que os valores para o concentrado do módulo de jigagem, alimentação e concentrado do módulo de espirais não sofreram nenhum ajuste, significando que a qualidade dos valores medidos foi boa. No entanto, os valores para o "oversize" da peneira rotativa e para o rejeito final, fluxo 14, sofreram correção bem maior, pois são os pontos de maior dificuldade de amostragem.

Cada série de dados foi composta de três dias de amostragem, sendo que a amostra de um dia foi composta de incrementos retirados de hora em hora, cujo volume variou de ponto a ponto em função da vazão e condições de cada fluxo.

A planilha dada pela Tabela 08 faz a comparação entre os valores experimentais e os calculados para o balanço metalúrgico. Observa-se que os valores, de um modo geral, não sofreram ajuste, ou somente de $\pm$ 0,01, representando a alta confiabilidade dos teores obtidos, exceto para os concentrados, com maior teor, que sofreram uma correção maior.

Foram feitos ensaios para as demais PL's 02, 03, 04, 05 e 06, seguindo o mesmo procedimento em todas as usinas. Os dados gerados pelos ensaios, fechando o balanço de cada operação, com sua média e desvio-padrão (modelo de erros adotado), foram planilhados em tabelas semelhantes às Tabelas 01 e 02.

As planilhas de entrada dos valores calculados, com a matriz M', seguiram o modelo das Tabelas 03 e 04. O modelo de erros adotado foi a utilização do desvio-padrão das 12 análises feitas.

As planilhas de resultados, com os valores ajustados para a vazão e teores, foram semelhantes às Tabelas 05 e 06. As planilhas comparativas entre os valores experimentais e os calculados, para o balanço de massas e o metalúrgico, seguiram o modelo das Tabelas 07 e 08.

TABELA 1: Valores de medidas experimentais, com respectivas medias e desvios-padrao de vazao da PL 01, para o ajuste do Balanco Massas

MINERACAO TABOCA S.A.
MINA DO PITINGA
VALORES PARA AJUSTE DO BALANCO DE MASSAS
(Normalizados para 100%)

USINA: PL 01 — 1991

FLUXO / AMOSTRA	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	100.00	94.95	5.05	90.85	4.10	48.42	42.43	1.89	46.53	32.10	14.43	12.86	1.57	44.96
2	100.00	95.10	4.90	94.30	0.80	45.65	48.65	1.65	44.00	31.40	12.60	11.99	0.61	42.02
3	100.00	98.93	1.07	97.40	1.53	47.78	49.62	1.84	45.94	28.90	17.04	16.00	1.04	44.90
4	100.00	98.95	1.05	96.99	1.96	50.20	46.79	2.49	47.71	27.22	20.49	19.94	0.55	47.16
5	100.00	98.85	1.15	97.03	1.82	44.08	52.95	2.01	42.07	21.76	20.31	16.77	3.54	38.53
6	100.00	98.30	1.70	96.09	2.21	50.13	45.96	2.10	48.03	27.23	20.80	19.10	1.70	46.33
7	100.00	98.27	1.73	95.99	2.28	52.16	43.83	3.00	49.16	37.96	11.20	8.38	2.82	46.34
8	100.00	98.20	1.80	96.32	1.88	52.00	44.32	3.76	48.24	31.91	16.33	14.15	2.18	46.06
9	100.00	98.17	1.83	94.77	3.40	53.98	40.79	3.09	50.89	31.14	19.75	17.61	2.14	48.75
10	100.00	98.11	1.89	93.60	4.51	40.97	52.63	2.19	38.78	27.93	10.85	10.21	0.64	38.14
11	100.00	96.06	3.94	93.99	2.07	57.18	36.81	5.20	51.96	33.87	18.11	17.13	0.98	51.00
12	100.00	96.00	4.00	94.63	1.37	54.79	39.84	4.17	50.62	34.89	15.73	13.77	1.96	48.66
MEDIA	100.00	97.49	2.32	95.16	2.33	49.78	45.39	2.78	47.00	30.53	16.47	14.83	1.64	45.24
DESVIO PADRAO	0.00	1.45	1.54	1.79	1.07	4.49	4.82	1.05	3.69	4.07	3.45	3.40	0.90	3.76

TABELA 2 : Valores de medidas experimentais, com respectivas medias e desvios-padrao de SnO2 da PL 01, para o ajuste do Balanco Metalurgico

MINERACAO TABOCA S.A.
MINA DO PITINGA
VALORES PARA AJUSTE DO BALANCO METALURGICO
Teor: %SnO2

1991

USINA: PL 01

FLUXO / AMOSTRA	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0.28	0.28	0.17	0.29	0.12	0.48	0.08	7.06	0.21	0.07	0.51	0.12	3.71	0.08
2	0.30	0.31	0.15	0.31	0.09	0.54	0.09	8.27	0.25	0.20	0.37	0.13	5.18	0.19
3	0.29	0.29	0.12	0.30	0.10	0.50	0.11	7.40	0.21	0.15	0.32	0.08	4.00	0.13
4	0.34	0.34	0.20	0.34	0.21	0.56	0.10	8.35	0.15	0.11	0.21	0.08	4.67	0.10
5	0.31	0.32	0.15	0.32	0.10	0.56	0.12	6.05	0.30	0.09	0.51	0.07	2.62	0.08
6	0.23	0.23	0.18	0.23	0.07	0.39	0.06	6.05	0.14	0.08	0.21	0.05	2.06	0.07
7	0.42	0.42	0.17	0.43	0.11	0.73	0.08	8.20	0.27	0.12	0.78	0.09	2.81	0.12
8	0.39	0.39	0.13	0.40	0.09	0.67	0.08	6.63	0.22	0.11	0.44	0.08	2.78	0.10
9	0.38	0.39	0.20	0.40	0.12	0.61	0.12	6.57	0.24	0.12	0.43	0.07	3.26	0.11
10	0.27	0.27	0.13	0.28	0.08	0.55	0.08	8.02	0.12	0.09	0.20	0.07	2.24	0.09
11	0.42	0.42	0.28	0.43	0.07	0.64	0.10	5.09	0.20	0.16	0.27	0.11	3.02	0.14
12	0.39	0.39	0.18	0.40	0.08	0.59	0.14	5.20	0.19	0.12	0.34	0.08	2.18	0.11
MEDIA	0.33	0.34	0.17	0.34	0.10	0.57	0.10	6.91	0.21	0.12	0.38	0.09	3.21	0.11
DESVIO PADRAO	0.06	0.06	0.04	0.06	0.04	0.09	0.02	1.12	0.05	0.03	0.16	0.02	0.95	0.03

TABELA 03 - Planilha de entrada de dados
Ajuste de Balanco de Massas

Usina: PL - 01 (Valores normalizados para 100%)

OPERACAO / FLUXO		1	2	3	4	5	6	7
1	Media	100						
	Desvio	1						
2	Media	-97,49	97,49					
	Desvio	1,45	1,45					
3	Media	-2,32						
	Desvio	1,54						
4	Media		-95,16	95,16				
	Desvio		1,79	1,79				
5	Media		-2,33					
	Desvio		1,07					
6	Media			-49,78	49,78			
	Desvio			4,49	4,49			
7	Media			-45,39				
	Desvio			4,82				
8	Media				-2,78			
	Desvio				1,05			
9	Media				-47,00	47,00		
	Desvio				3,69	3,69		
10	Media					-30,53		30,53
	Desvio					4,07		4,07
11	Media					-16,47	16,47	
	Desvio					3,45	3,45	
12	Media						-14,83	14,83
	Desvio						3,40	3,40
13	Media						-1,64	
	Desvio						0,90	
14	Media							-45,24
	Desvio							3,76

TABELA 04 - Planilha de entrada de dados
Ajuste de Balanco Metalurgico

Usina: PL - 01 Teor : % SnO2

OPERACAO / FLUXO		1	2	3	4	5	6	7
1	Media	0,33						
	Desvio	0,06						
2	Media	-0,34	0,34					
	Desvio	0,06	0,06					
3	Media	-0,17						
	Desvio	0,04						
4	Media		-0,34	0,34				
	Desvio		0,06	0,06				
5	Media		-0,10					
	Desvio		0,04					
6	Media			-0,57	0,57			
	Desvio			0,09	0,09			
7	Media			-0,10				
	Desvio			0,02				
8	Media				-6,91			
	Desvio				1,12			
9	Media				-0,21	0,21		
	Desvio				0,05	0,05		
10	Media					-0,12		0,12
	Desvio					0,03		0,03
11	Media					-0,38	0,38	
	Desvio					0,16	0,16	
12	Media						-0,09	0,09
	Desvio						0,02	0,02
13	Media						-3,21	
	Desvio						0,95	
14	Media							-0,11
	Desvio							0,03

TABELA 05 - Planilha de saida dos dados - Ajuste de Balanco de massas

Usina: PL - 01 (Valores normalizados para 100%)

OPERACAO FLUXO		1	2	3	4	5	6	7
1	Media	100						
	Desvio	1						
	Valor Calc	99.96						
2	Media	-97,49	97,49					
	Desvio	1,45	1,45					
	Valor Calc	-97.54	97.54					
3	Media	-2,32						
	Desvio	1,54						
	Valor Calc	-2.42						
4	Media		-95,16	95,16				
	Desvio		1,79	1,79				
	Valor Calc		-95.20	95.20				
5	Media		-2,33					
	Desvio		1,07					
	Valor Calc		-2.35					
6	Media			-49,78	49,78			
	Desvio			4,49	4,49			
	Valor Calc			-49.76	49.76			
7	Media			-45,39				
	Desvio			4,82				
	Valor Calc			-45.44				
8	Media				-2,78			
	Desvio				1,05			
	Valor Calc				-2,78			
9	Media				-47,00	47,00		
	Desvio				3,69	3,69		
	Valor Calc				-46.97	46.97		
10	Media					-30,53		30,53
	Desvio					4,07		4,07
	Valor Calc					-30.51		30.51
11	Media					-16,47	16,47	
	Desvio					3,45	3,45	
	Valor Calc					-16.47	16.47	
12	Media						-14,83	14,83
	Desvio						3,40	3,40
	Valor Calc						-14.82	14.82
13	Media						-1,64	
	Desvio						0,90	
	Valor Calc						-1.64	
14	Media							-45,24
	Desvio							3,76
	Valor Calc							-45.35

TABELA 06 - Planilha de saida dos dados - Ajuste do Balanco Metalurgico

Usina: PL - 01 Teor : % SnO2

OPERACAO FLUXO		1	2	3	4	5	6	7
1	Media	0,33						
	Desvio	0,06						
	Valor Calc	0.34						
2	Media	-0,34	0,34					
	Desvio	0,06	0,06					
	Valor Calc	-0,34	0,34					
3	Media	-0,17						
	Desvio	0,04						
	Valor Calc	-0,17						
4	Media		-0,34	0,34				
	Desvio		0,06	0,06				
	Valor Calc		-0,35	0,35				
5	Media		-0,10					
	Desvio		0,04					
	Valor Calc		-0,10					
6	Media			-0,57	0,57			
	Desvio			0,09	0,09			
	Valor Calc			-0,57	0,57			
7	Media			-0,10				
	Desvio			0,02				
	Valor Calc			-0,10				
8	Media				-6,91			
	Desvio				1,12			
	Valor Calc				-6,69			
9	Media				-0,21	0,21		
	Desvio				0,05	0,05		
	Valor Calc				-0,21	0,21		
10	Media					-0,12		0,12
	Desvio					0,03		0,03
	Valor Calc					-0,12		0,12
11	Media					-0,38	0,38	
	Desvio					0,16	0,16	
	Valor Calc					-0,39	0,39	
12	Media						-0,09	0,09
	Desvio						0,02	0,02
	Valor Calc						-0,09	0,09
13	Media						-3,21	
	Desvio						0,95	
	Valor Calc						-3.06	
14	Media							-0,11
	Desvio							0,03
	Valor Calc							-0,11

TABELA 07 - Planilha comparativa de valores experimentais e ajustados
Ajuste de Balanco de Massas

Usina: PL - 01 (Valores normalizados para 100%)

FLUXO	VALOR EXP.	VALOR AJUS.	DIFERENCA	DISTR.(%)
1 - Alimentacao Peneira Rotativa - RON	100.00	99.96	0.04	100.00
2 - Undersize Peneira Rotativa	97.49	97.54	0.05	97.58
3 - Oversize Peneira Rotativa	2.32	2.42	0.10	2.42
4 - Alimentacao do Ciclone	95.16	95.20	0.04	95.24
5 - Overflow da Caixa Desaguadora	2.33	2.35	0.02	2.35
6 - Alimentacao do Modulo de Jigagem	49.78	49.76	0.02	49.78
7 - Overflow do Ciclone	45.39	45.44	0.05	45.46
8 - Concentrado do Modulo de Jigagem	2.78	2.78	0.00	2.78
9 - Rejeito do Modulo de Jigagem	47.00	46.97	0.03	46.99
10 - Oversize da Peneira	30.53	30.51	0.02	30.52
11 - Alimentacao do Modulo de Espirais	16.47	16.47	0.00	16.48
12 - Rejeito do Modulo de Espirais	14.83	14.82	0.01	14.83
13 - Concentrado do modulo de Espirais	1.64	1.64	0.00	1.64
14 - Rejeito Final	45.24	45.35	0.11	45.37

TABELA 08 - Planilha comparativa de valores experimentais e ajustados
Ajuste de Balanco Metalurgico

Usina: PL - 01 Teor : % SnO2

FLUXO	VALOR EXPERIMENTAL	VALOR AJUSTADO	DIFERENCA
1 - Alimentacao Peneira Rotativa - ROM	0.33	0.34	0.01
2 - Undersize Peneira Rotativa	0.34	0.34	0.00
3 - Oversize Peneira Rotativa	0.17	0.17	0.00
4 - Alimentacao do Ciclone	0.34	0.35	0.01
5 - Overflow da Caixa Desaguadora	0.10	0.10	0.00
6 - Alimentacao do Modulo de Jigagem	0.57	0.57	0.00
7 - Overflow do Ciclone	0.10	0.10	0.00
8 - Concentrado do Modulo de Jigagem	6.91	6.69	0.22
9 - Rejeito do Modulo de Jigagem	0.21	0.21	0.00
10 - Oversize da Peneira	0.12	0.12	0.00
11 - Alimentacao do Modulo de Espirais	0.38	0.39	0.01
12 - Rejeito do Modulo de Espirais	0.09	0.09	0.00
13 - Concentrado do modulo de Espirais	3.21	3.06	0.15
14 - Rejeito Final	0.11	0.11	0.00

V. CONCLUSÕES

O método de ajuste dos valores dos balanços de massas e metalúrgicos, descrito neste trabalho, foi desenvolvido a partir das necessidades reais de ter as medidas de vazão e teores ajustadas, para permitir o fechamento e maior confiabilidade dos dados do balanço de massas da usina. Deste modo, o programa de computador pode ser visto como um enorme exercício de ajuste, onde as regras definidas regulam o cálculo.

Os benefícios do balanço de massas estão em dois pontos principais. Primeiro, o trabalho envolvido na escolha e determinação dos dados da usina, obtidos da investigação de um processo mineral é minimizado. Os dados do processo obtidos pelo programa de balanço de massas podem ser usados diretamente e interpretados em termos de processo global, devido ao fechamento assegurado em todas as operações. Este ponto deve ser enfatizado, pois por métodos manuais, o tempo gasto para os cálculos é frequentemente alto e pode-se chegar apenas a alguns balanços de baixa confiabilidade. Usando programas de balanço de massas, maior ênfase pode ser dada para os problemas de processo.

O segundo benefício vem do uso periódico do balanço de massas: o programa pode ser usado para verificar as flutuações nas condições operacionais do processo, a curto e longo prazos, auxiliando na identificação de quaisquer mudanças que possam ocorrer, por exemplo, mudança de qualidade de minérios, mau funcionamento de equipamentos e o desgaste em componentes do circuito, tal como: "apex" de ciclones ou revestimentos: a operação, nestas condições, pode levar a uma condição anti-

econômica. Estas flutuações ocorrem gradualmente e, usualmente, não podem ser detectadas no dia-a-dia, mas podem ser detectadas facilmente a partir de um balanço de massas rotineiro e as correções necessárias feitas assim que detectadas.

No caso em estudo, o balanço de massas e o metalúrgico evidenciaram perdas significativas de SnO_2, nos fluxos de rejeitos de todas as PL's, o que levou a Mineração Taboca a instalar um moinho de martelos no fluxo 3 - "oversize" da peneira rotativa, e um moinho de barras no fluxo 10 - "oversize" da peneira vibratória, e ainda a melhorar a deslamagem no fluxo 5 - "overflow" da caixa desaguadora, com o intuito de otimizar a recuperação de Sn.

É de grande importância a hierarquização dos dados, para uma melhor precisão nos resultados. A variância dos valores experimentais é usualmente escolhida como o mais satisfatório fator de ponderação, definindo o modelo de erros.

Portanto, o funcionamento do processo mineral e os métodos de obtenção de dados devem ser bem compreendidos pelo profissional que atribuir os fatores de ponderação, no algoritmo proposto, pois sua habilidade para enquadrar o processo, em termos do balanço de massa, em qualquer direção, é de fundamental importância para o sucesso da operação. Em outras palavras, um usuário inexperiente pode facilmente obter resultados errôneos; particularmente, onde os dados foram obtidos apressadamente ou com outro objetivo. Estas condições não são incomuns em usinas de processamento mineral.

O sistema de ponderação pode ser modificado para para tornar mais clara a compreensão de certas áreas do fluxograma de processo.

Nenhum dos métodos discutidos fornece mais do que uma estimativa do valor verdadeiro da vazão. Qualquer dos métodos não é confiável para o ajuste de dados de baixa confiabilidade.

O método numérico proposto oferece também um procedimento de obtenção da melhor estimativa dos balanços metalúrgicos, baseados no tratamento estatístico próprio dos dados disponíveis, da mesma forma da rotina para o balanço de massas.

Os dados experimentais obtidos em uma unidade de processamento mineral podem ser tratados pelo método apresentado. O algoritmo permite:

(a) a determinação das vazões em vários pontos do circuito,

(b) a correção dos dados experimentais a fim de torná-los consistentes do ponto de vista do balanço de massas e metalúrgico.

O método usado, dos mínimos quadrados ponderado, faz a minimização de forma hierárquica, ou seja, as variáveis são divididas em dois conjuntos: o das vazões e o dos componentes, análises de teores e granulométricas, e as partes correspondentes da função a serem minimizadas são independentemente minimizadas. As restrições do problema de minimização são as equações de balanço de massa. Elas são escritas de uma forma geral, usando redes de fluxo e cálculo matricial. As principais vantagens do método são:

(1) grande número de variáveis de componentes podem ser manuseadas, devido à existência da solução analítica, para a minimização da parte de componentes,

(2) o programa é de fácil utilização, até mesmo para profissionais não familiarizados com a técnica, devido ao seu desenvolvimento de forma estruturada e conversacional, tendo a grande vantagem de estar escrito em Português,

(3) o programa é geral, abrangendo vários processos e tipos de medidas diferentes,

(4) o programa é versátil. Devido à decomposição adotada e devido à estrutura modular do programa, é fácil implementar itens especiais com pouca programação.

SUGESTÕES

Para continuidade do desenvolvimento do sistema ora apresentado, sugerimos outros trabalhos a serem desenvolvidos, para aumentar os recursos deste programa:

- Introduzir outras metodologias de cálculo como alternativa,
- Novos módulos: . Simulação de circuitos,
  . Simulação de operações unitárias,
  . Circuitos de Britagem,
  . Circuitos de moagem.
- Desenvolvimento da parte gráfica, tanto relativo a fluxogramas, quanto a gráficos comparativos.

REFERÊNCIAS BIBLIOGRAFICAS

01 NUNES, A.C.; FERREIRA, F.M.; TOMI, G.F.C. Aplicação de microcomputadores na indústria mineral. São Paulo, EPUSP, 1983. / Trabalho apresentado ao Prêmio José Ermírio de Moraes, 1983/

02 GY, P. Sampling of particulate materials: theory and practice. Amsterdam, Elsevier, 1982. (development in Geomathematics, v.4.)

03 FREW, J.A. Computer aided design of sampling schemes. International Journal of Mineral Processing, v. 11, p. 255-65, 1983.

04 LYNCH, A. J. Mineral crushing and grinding Circuits: their simulation, optimization, design and control Amsterdam, Elsevier, 1977.(Developments in Mineral Processing, v.1.)

05 WILLS, B.A.; MANSER R.J. Reconciliation of simple node excess data. Transactions of the Institution of Mining and Metallurgy. Section C, v. 94, p. 209-14, Dec. 1985.

06 BAZAN, A.L.C. Cálculo computadorizado de balanços de massa e metalúrgico. s.l., s.n., 1984 / Trabalho apresentado ao Prêmio José Ermírio de Moraes, 1984/

07 LAGUITTON, D. Material Balance Computation for Process evaluation and modelling. BILMAT2 Computer Program, Report MRP/MSL 82-11, CANMET, Energy, Mines, and Resources, 1982.

08 HUMES,A.F.P. et al. Noções de cálculo número. São Paulo, McGraw-Hill, 1984.

09 RUGGIERO, M.A.G.; LOPES, V.L.R. Cálculo numérico: aspectos teóricos e computacionais. São Paulo, McGraw-Hill, 1988.

10 WHITE, J.W.; WINSLOW, R.L.; ROSSITER, G.J. A useful technique for metallurgical mass balances: applications in grinding. International Journal of Mineral Processing, v.4, p.39-49, 1977.

11 MULAR, A.L. Data adjustment procedures for mass balances. In: WEISS, A., ed. Computer methods for the 80's in the mineral industry. New York, Society of Mining Engineers, 1979. p.843-9.

12 WIEGEL, R.L. Advances in mineral processing material balances. Canadian Metallurgical Quarterly, v. 11, n.2, p.413-24, 1972.

13 CUTTING, G.W. Estimation of interlocking mass balances on complex mineral Beneficiation plants. International Journal of Mineral Processing, v.3, p.207-18, 1976.

14 LAGUITTON, D. Material Balance Computation for Process evaluation and modelling. MATBAL3 Computer Program, Report MRP/MSL 82-131, CANMET, Energy, Mines, and Resources, 1982.

15 SMITH, H.W.; ICHIYEN, N. Computer adjustment of metallurgical balances. The Canadian Mining and Metallurgical Bulletin, v. 66, n.737, p. 97-100, Sept. 1973.

16 CUTTING, G.W.; WATSON, D. Material flow balances around mineral processing plants. Stevenage, Warren Spring Laboratory, 1976.

17 JULIUS KRUTTSCHNITT MINERAL RESEARCH CENTRE. JK MBal: the mass balancing system. Indooroopilly, JK Tech, s.d. (JKT 2032)

18 JULIUS KRUTTSCHNITT MINERAL RESEARCH CENTRE. JK SimMet: the mineral processing simulator. Indooroopilly, JK Tech, s.d. (JKT 1641)

19 HODOUIN, D.; EVERELL, M.D. A hierarchical procedure for adjustment and material balancing of mineral processes data. International Journal of Mineral Processing, v. 7, p.91-116, 1980.

20 HODOUIN, D. ET AL. Bilmat: an algorithm for material balancing mineral processing circuits; applications to comminution, desliming and flotation circuits. The Canadian Mining and Metallurgical Bulletin, v.74, p.123-31, Sept. 1981.

21 VOLLER, V.R.; RYAN, P.J. Automated material balance and assay data adjustment around a piece of mineral processing equipment. International Journal of Mineral Processing, v. 10. p.279-88, 1983.

22 PROWARE. Metsim R. Lakewood, Proware, s.d.

23 CHAZIN, S. Simula (Programa de computação). São Paulo, Departamento de Engenharia de Minas da EPUSP, 1989. 1 Disquete, em Linguagem Basic.

24 CORTES, P.L. Turbo C - Ferramenta e Utilitários. v.1, São Paulo, Editora Erica, 1992.

25 TOMI, G.; NUNES, A.C. Computerized mine planning at Pitinga tin mine. In: INTERNATIONAL CONFERENCE ALLUVIAL MINING, London, 1991. Alluvial mining. London, Elsevier, 1991. p.53-62.

INDICE DE FIGURAS

INDICE DE TABELAS

TABELAS Página.

APENDICE 01 - LISTAGEM DO PROGRAMA EM LINGUAGEM C

```
#include <conio.h>

void abertura ()
{
 int i;

 clrscr ();

 tela ();

 titulo ();

 textattr (LIGHTGRAY + (LIGHTGRAY<<4));
}

#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include "d:\toninho\bal_def.h"
#include <string.h>
#include <mem.h>
#include <io.h>
#include <fcntl.h>
#include <conio.h>
#include <ctype.h>     /* para conversao de string em maiuscula      */
#include <sys\stat.h>  /* para verificar e mudar atributo read only  */
#include <sys\types.h> /* para verificar e mudar atributo read only  */
#include "d:\procalc\mcalc.h"
#include "d:\toninho\bal_def.h"

/* funcoes */

void balmas_1 ();
void opcao ();
void entradai ();
void limpam ();
void limpami ();
void restricoesi ();
void pesoi ();
void matrizi ();
void cond_tri ();
void calcula ();
void respostai ();
void entradaii ();
void restricoesii ();
void rest_restri ();
void pesoii ();
void matrizii ();
void respostaii ();
void impressaom ();
void impressaot ();
void confere ();
void load3 ();
```

```
int opcaoi;
int respp;
int respb;
int m;
int n;
int cont;

float  gi     [MAX_MATRIZ_I];
float  gc     [MAX_MATRIZ_I];
float  q      [MAX_MATRIZ_I];
float  peso   [MAX_MATRIZ_I];
float  massa  [MAX_MATRIZ_I];
float  teor   [MAX_MATRIZ_I];
float  r      [MAX_MATRIZ_I]; /**/
float  t      [MAX_MATRIZ_I];
float  rc     [MAX_MATRIZ_I];
float  tc     [MAX_MATRIZ_I];
float  dpt    [MAX_MATRIZ_I];
float  dpm    [MAX_MATRIZ_I]; /**/
float  y      [MAX_MATRIZ_I];
float  lag    [MAX_MATRIZ_I];

float a   [MAX_MATRIZ_I][MAX_MATRIZ_I];
float aux [MAX_MATRIZ_I][MAX_MATRIZ_I];

char auxiliar [15];
char filename [20];
char respi    [MAX_MATRIZ_I];
char resp;
char respe;

void balmas_1 ()
{

 register int i;
 register int j;

 /* Turbo C 2.01 bug killer ! */
 float bug = sin (0);
 if (bug != 0) exit (-1);

do {

    /* setando variaveis */

    cont   = 0;
    opcaoi = 1;
    respp  = 1;
    respb  = 0;
    resp   = 's';
    respe  = 'n';
    strcpy (respi,"");
```

```
    for (i=1; i<=MAX_MATRIZ_I; i++) {

         for (j=1; j<=MAX_MATRIZ_J; j++) {
              aux [i][j] = 0;
              a   [i][j] = 0;
                                        }

         gi    [i] = 0;
         gc    [i] = 0;
         peso  [i] = 0;
         massa [i] = 0;
         teor  [i] = 0;
         r     [i] = 0;
         t     [i] = 0;
         rc    [i] = 0;
         tc    [i] = 0;
         dpt   [i] = 0;
         dpm   [i] = 0;
         y     [i] = 0;
         lag   [i] = 0;

         timer ();
                        }


    m = 0;
    n = 0;

    clrscr ();

    opcao ();

    switch (opcaoi) {

      case 1:

       limpam ();
printf ("\n1 Processando entradai");
       entradai ();
       load3 ();
printf ("\n2 Processando restricoesi");
       restricoesi ();

printf ("\n3 Processando pesoi");
       pesoi ();
printf ("\n4 Processando matrizi");
       matrizi ();
printf ("\n5 Processando cond_tri");
       cond_tri ();
printf ("\n6 Processando confere");
       confere ();
printf ("\n7 Processando if (cont != 0) break;");
       if (cont != 0) break;
```

```
printf ("\n8 Processando calcula");
       calcula ();
printf ("\n9 Processando respostai");
       respostai ();
printf ("\n10 Processando break");
      break;

      case 2:

       entradaii ();
       limpam ();
       restricoesii ();
       pesoii ();
       matrizii ();
       cond_tri ();
       confere ();
       if (cont!=0) break;
       calcula ();
       respostaii ();
      break;

      case 3:
       printf ("\ndeseja  fazer  calculo  da  media  e  desvio  padrao?   (s/n)");
       respe = getch ();
       if (respe=='s' || respe=='s') {
           opcaoi=1;
           estatistica ();
           opcaoi=3;
                                    }

       else entradai ();
       limpam ();
       restricoesi ();
       pesoi ();
       matrizi ();
       cond_tri ();
       confere ();

       if (cont!=0) break;
       calcula ();
       respostai ();
       printf ("\ndeseja  fazer  calculo  da  media  e  desvio  padrao?   (s/n)");
       respe = getch ();
       if   (respe=='s' || respe=='s') {
             opcaoi = 2;
             estatistica ();
             opcaoi = 3;
                                      }

       else entradaii ();
       limpami ();
       rest_restri ();
       pesoii ();
```

```
        matrizii ();
        cond_tri ();
        confere ();
        if (cont != 0) break;
        calcula ();
        respostaii ();
       break;

       case 4:
        impressaom ();
        impressaot ();
       break;

       case 5:
        clrscr ();
        printf ("\nÂte´ breve !");
        exit (0);
       break;

                    } /* switch ... */


   } while (1);

}

/* ---------------------- */
/* balanco  de  massa */
void opcao ()
{
  opcaoi  =0;

  printf ("\nDeseja  fazer  qual  ajuste  ?");

  printf ("\n 1 - balanco  de  massas");

  printf ("\n 2 - balanco  metalurgico");

  printf ("\n 3 - balanco  de  massas  e  balanco  metalurgico");

  printf ("\n 4 - impressao de resultados");

  printf ("\n 5 - para  sair  do  programa");

  printf ("\nTecle a opcao desejada !");

  gets (auxiliar);
  opcaoi = atoi (auxiliar);

}
```

```
/* -------------------- */

/* comentario */

void entradai ()
{
 int i;

    printf ("\nQual a sua opcao de entrada de ponderadores ?");

    printf ("\n1 - desvio  padrao");

    printf ("\n2 - grau  de  incerteza   (%)");

    printf ("\n3 - grau  de  certeza   (%)");

    if  (respb==1)  respp=1;
    else {
          gets (auxiliar);
          respp = atoi (auxiliar);
         }
    printf ("\nQual o nome do arquivo a ser analisado ? ");
    gets (filename);

    printf ("\n\nLendo arquivo. Aguarde ... \n\n");


    load3 (); /* IDEM PARA BALANCO METALURGICO */

/*
    printf ("\nentre  com  os  valores  experimentais  das  massas  e  os  do");

    for (i=1; i<=m; i++) {
         printf ("\nValor r [%i]", i);

        gets (auxiliar);
        r [i] = atof (auxiliar);

        switch  (respp) {
          case 1:
           do {
              printf ("\ndpm[%i]",i);
              gets (auxiliar);
              dpm[i] = atof (auxiliar);
              if (dpm[i]==0) printf("\ndesvio padrao == 0 (zero!! parametro do !!");
              } while (dpm[i]==0);
          break;

          case 2:
           do {
               printf ("\ngi[%i]",i);
               gets (auxiliar);
               gi[i] = atof (auxiliar);
               if (gi[i]<=0) printf ("\n parametro  incompativel!!  gi>0  ");
```

```
            } while (gi[i]<=0);

           dpm[i]=gi[i]*r[i]/100;


            break;

          case 3:
           do {
               printf ("\ngc[%i]",i);
               gets (auxiliar);
               gc[i] = atof (auxiliar);
               if (gc[i]<=1 || gc[i]<0) printf("\nparametro incompativel!! 0<gc<1");
              } while (gc[i]<=1 || gc[i]<0);

           dpm[i] = (1-gc[i]/100)*r[i];

          break;
                       }  switch ...

                        }  for ...
*/
}
/* ---------------------- */
/* comentario */
void limpam ()

{
 int i,j;

 for (i=1; i<=m+n; i++) for (j=1; j<=m+n+1; j++) a[i][j]=0;

 for (i=1; i<=m; i++)  q[i]=0;

}

/* ---------------------- */
/* comentario */
void limpami ()
{
 int i,j;

 for (i=1; i<=m+n; i++) for (j=1; j<=m+n+1; j++) a[i][j]=0;
}

/* ---------------------- */
/* comentario */
void restricoesi ()
{
 int i,j;

 for (j=m+1; j<=m+n; j++) {

      for (i=1; i<=m; i++) {
```

```
            /*
            printf("\nentre com a restricao do fluxo ,i, relativo ao no %i",j-m);
            gets (auxiliar);
            aux[i][j] = atof (auxiliar);
            if (aux[i][j] != 0.0) a[i][j]=(fabs(aux[i][j])/aux[i][j]);

            else a[i][j] = (aux[i][j]);
            */
            a[i][j] = (aux[i][j]);
            q[i]=q[i]+1;
            a[j][i]=a[i][j];
                          } /* for ... */

                        } /* for ... */
}

/* ---------------------- */
/* comentario */
void pesoi ()
{
  int i;

  for (i=1; i<=m; i++) peso[i]  = (1/ (dpm[i]*dpm[i]));
}

/* ---------------------- */
/* comentario */
void matrizi ()
{
 int i,j;

 for (i=1; i<=m; i++) a[i][i]=2*peso[i]*q[i];

 for (i=1; i<=m; i++) {
      /* a[i][i]=2*peso[i]*q[i]; */
      j=m+n+1;
      a[i][j]=2*peso[i]*r[i]*q[i];
                      }

}

/* ---------------------- */
/* comentario */
void cond_tri ()
{
 int l,lin,i,p;
 float max,e,x;

 for (l=1; l<=m+n; l++) {
      max = a[l][l];
      lin=l;
      for (i=(l+1); i<=(m+n); i++) {
           if (fabs (a[i][l]) > fabs (max)) {
```

```
                max=a[i][1];
                lin=i;                          }
                                    }


        if  (lin != 1) {
             for (p=1; p<=m+n+1; p++) {

                  e=a[1][p];
                  a[1][p]   = a[lin][p];
                  a[lin][p] = e;
                                        }
                       }


        for (i=1+1; i<=m+n; i++) {
             x=a[i][1]/a[1][1];
             a[i][1]=0;
             for (p=1+1; p<=m+n+1; p++) {
                  e=a[i][p]-(a[1][p]*x);
                  a[i][p]=e;
                                          }
                                     }
                            } /* for ... */
}
/* ---------------------- */
/* comentario */
void calcula ()
{
 int i,p;
 float e;

 y[m+n]=a[m+n][m+n+1]/a[m+n][m+n];

 for (i=(m+n-1); i>=1; i--) {      /*    > =    */
      e = 0;
      for (p=i+1; p<=m+n; p++) e=e+a[i][p]*y[p];
      y[i] = (a[i][m+n+1]-e)/a[i][i];
                                 }

}

/* ---------------------- */
/* comentario */
void respostai ()
{
 int i;

 for (i=1;  i<=m; i++) {
      rc[i]=y[i];
      printf ("\nrc[%i] = %f ",i,rc[i]);
                        }
```

```
 for (i=m+1; i<=m+n; i++) {
      lag[i]=y[i];
      printf ("\n1 = %f",lag[i]);
                          }

 for (i=1; i<=m; i++) {
      massa[i] = (100*rc[i]/rc[1]);
      printf   ("\n%% massa = %f",massa[i]);
                     }

 getch ();

}

/* ---------------------- */
/* balanco  metalurgico */
void entradaii ()
{
 int i;

 do {
     printf ("\n sao  os  fluxos  existentes  no  processo  dessa  planta?");
     gets (auxiliar);

     m = atoi (auxiliar);
     if (m<=0) printf ("\n parametro  incompativel!!  m>0  ");
    } while (m<=0);

 do {
     printf ("\ne  o  numero  de  operacoes?");
     gets (auxiliar);
     n = atoi (auxiliar);
     if (n<=0) printf ("\n parametro  incompativel!!  n>0  ");
    } while (n<=0);

 printf ("\nqual  a  sua  opcao  de  entrada  de  do");
 printf ("\n1 - desvio  padrao");
 printf ("\n2 - grau  de  incerteza   (%)");
 printf ("\n3 - grau  de  certeza   (%)");

 if (respb==1) respp = 1;
 else {
       gets (auxiliar);
       respp = atoi (auxiliar);
      }

 printf ("\nentre  com  os  valores  experimentais  das  massas  e  os  do");

 for (i=1; i<=m; i++) {
      printf ("\nt[%i]", i);
      gets (auxiliar);
      t[i] = atof (auxiliar);
```

```
        switch  (respp) {
         case 1:
          do {
              printf ("\ndpt[%i]",i);
              gets (auxiliar);
              dpt[i] = atof (auxiliar);
              if(dpt[i]==0.0) printf("\ndesvio padrao==0 (zero!! parametro do !!");
             } while (dpt[i]==0.0);
         break;


         case 2:
          do {
              printf ("\ngi[%i]", i);
              gets (auxiliar);
              gi[i] = atof (auxiliar);
              if(gi[i]<=0) printf("\nparametro incompativel!! gi>0");
             } while (gi[i]<=0);

          dpm[i]=gi[i]*r[i]/100;

         break;

         case 3:
          do {
              printf ("\ngc[%i]", i);
              gets (auxiliar);
              gc[i] = atof (auxiliar);
              if (gc[i]<=1 || gc[i]<0) printf ("\n parametro  incompativel!!  0<gc<1  ")
;
             } while (gc[i]<=1 || gc[i]<0);

          dpm[i] = (1-gc[i]/100)*r[i];
         break;

                        } /* switch ... */

                        } /* for ... */
}

/* -------------------- */
/* comentario */
void restricoesii ()
{
 int i,j;
 float num;

 for (i=1;  i<=m; i++) {
      printf ("\nrc[%i]", i);
      gets (auxiliar);
      rc[i] = atof (auxiliar);
                        }

 for (j=m+1; j<=m+n; j++) {
```

```
      for (i=1; i<=m; i++) {
           printf("\nentre com a restricao do fluxo %i relativo ao no %i",
                 i,j-m);
           gets (auxiliar);
           num = atof (auxiliar);
           if (num !=0) a[i][j]=(fabs(num))/num;
           else a[i][j]=num;
           q[i]=q[i]+1;
           a[j][i]=a[i][j];
                          }
                         }
}

/* ---------------------- */
/* comentario */
void rest_restri ()
{
 int j,i;

 for (j=m+1; j<=m+n; j++) {
      for (i=1; i<=m; i++) {
           if (aux[i][j] != 0) a[i][j]=(fabs (aux[i][j]))/(aux[i][j]);
           else a[i][j]=aux[i][j];
           a[j][i]=a[i][j];
                          }
                         }
}

/* ---------------------- */
/* comentario */
void pesoii ()
{
 int i;

 for (i=1; i<=m; i++) peso[i]  = 1/(dpt[i]*dpt[i]*rc[i]*rc[i]);
}

/* ---------------------- */
/* comentario */
void matrizii ()
{
 int i,j;

 for (i=1; i<=m ; i++) a[i][i]=2*peso[i]*q[i];

 for (i=1; i<=m ; i++) {
      j=m+n+1;
      a[i][j] =2*peso[i]*rc[i]*q[i]*t[i];
                        }
}

/* ---------------------- */
/* comentario */
```

```
void respostaii ()
{
 int i;

 for (i=1; i<=m; i++) {
      tc[i]=y[i]/rc[i];
      printf ("\ntc[%i] = %f ",i, tc[i]);
                      }

 for (i=m+1; i<=m+n; i++) {
      lag[i]=y[i];
      printf ("\nl = %f",lag[i]);
                          }

 for (i=1; i<=m; i++) {
      teor[i]=(tc[i]*100*rc[i])/(rc[i]*tc[1]);
      printf ("%% teor = %f",teor[i]);
                      }

 getch ();

}

/* ---------------------- */

/* comentario */

void impressaom ()

{
 int i;

 printf  (lst,"Escola Politecnica da USP
           Depto. de Engenharia de Minas  ";

       printf  (lst,  "programa  de  ajuste  de  balanco  de  massas  e  metalurgico";
       printf  (lst,  "                    por  acn";
       printf  (lst;
       printf  (lst;
       if  opcaoi==1              printf  (lst  ,"      balanco  de  massas  "
       else           printf  (lst  ,"      balanco  de  massa   e   metalurgico";
       printf  (lst;
       printf  (lst,"planta    ",respi;
       printf  (lst,  "fluxos    ",m;
       printf  (lst,"operacoes    ",n;
       printf  (lst;
       printf  (lst,"valores  experimentais  das  massas  e  os  desvios";
       printf  (lst,"e  seus  respectivos  valores  do;
       printf  (lst;
       for (  i  =1;   m  do         {
          printf  (lst,"r[  ",i,"]==  ",r[i]  2 2;
         printf  (lst,"dpm[  ",i,"]==  ",dpm[i]  2 2;
         switch  respp  of
```

```
          2  printf  (lst,"gi[  ",i,"]==  ",gi[i]  2 2;
         3  printf  (lst,"gc[  ",i,"]==  ",gc[i]  2 2;
        }
        printf  (lst,"rc[  ",i,"]==  ",rc[i]  2 2;
        printf   (lst,"%  massa  == ",massa[i]  2 2;
        printf  (lst;
       }
     }
}

/* ---------------------- */

/* comentario */

void impressaot ()

{

  var  i  int;

  {
     if  opcaoi==2        {
        printf  (lst,"Escola Politecnica da USP
                  Depto. de Engenharia de Minas ";
       printf  (lst,"programa de ajuste de balanco de massas e metalurgico";
       printf  (lst,  "                    por  acn";
       printf  (lst;
       printf  (lst;
       printf  (lst  ,"      balanco  metalurgico  ";
       printf  (lst;
       printf  (lst,"planta    ", respi;
       printf  (lst,  "fluxos   ",m;
       printf  (lst,"operacoes   ",n;
    }
       else         printf  (lst;
       printf  (lst,"valores  experimentais  do  teores  e  os  desvios";
       printf  (lst,"e  seus  respectivos  valores  do;
       printf  (lst;
       for (  i  =1;   m  do          {
           printf  (lst,"t[  ",i,"]==  ",t[i]  2 2;
          printf  (lst,"rc[  ",i,"]==  ",rc[i]  2 2;
          printf  (lst,"dpt[  ",i,"]==  ",dpt[i]  2 2;
          switch  respp  of
            2  printf  (lst,"gi[  ",i,"]==  ",gi[i]  2 2;
           3  printf  (lst,"gc[  ",i,"]==  ",gc[i]  2 2;
          }

          printf  (lst,"tc[  ",i,"]==  ",tc[i]  2 2;
          printf   (lst,"%  teor  == ",teor[i]  2 2;
          printf  (lst;
         }

}
```

```
/* ---------------------- */

/* comentario */

void confere ()

{

 int i;

 for(i=1; i<=m+n; i++) if (a[i][i]==0) printf("\nsistema do ou inconsistente!");

}


/* ------------------------ */

/* le a planilhas e monta arquivo */

void load3 ()
{

 FILE *fp;
 /*char auxiliar [81];*/
 char check[81];
 char *name = MSGNAME;

 unsigned char format   [MAXCOLS] [MAXROWS];
 unsigned char colwidth [MAXCOLS];


 int size;
 int file;
 int i;
 int j;
 int max_i       = 0;
 int max_j       = 0;
 int conta_i     = 0;
 int conta_ii    = 0;
 int conta_iii   = 0;
 int lastcol;
 int lastrow;
 int curcol;
 int currow;
 int total_cell;                    /* numero total de celulas */
 int contador    = 0;               /* contador de celulas     */

 int flag_linha_valor       = 0;
 int flag_linha_dp          = 0;
 int flag_linha_leitura_val = OK;
 int flag_linha_leitura_dp  = OK;

 double v;
```

```
float matriz_valor [MAX_MATRIZ_I][MAX_MATRIZ_J];
float matriz_dp    [MAX_MATRIZ_I][MAX_MATRIZ_J];

struct CELLREC rec;
CELLPTR cell [MAXCOLS] [MAXROWS], curcell;
CELLPTR cellptr;

/* COMECO DOS CALCULOS */

for (i=0; i<=MAX_MATRIZ_I; i++) {
     for (j=0; j<=MAX_MATRIZ_J; j++) {
          matriz_valor [i][j] = 0.0;
          matriz_dp    [i][j] = 0.0;
                              }
                         }

i = j = m = n = 0;

          if (access(filename, 0)) {printf ("\nFalha na abertura de %s", filename);
          exit (-1);}

          if ((file = open(filename, O_RDWR | O_BINARY)) == -1) exit (-1);

          /*lendo (filename);*/
          /*timer ();*/

          read(file, check, strlen(name) + 1);
          if (strcmp(check, name) != 0) exit (-1);

               read(file, (char *)&size, 1);
               /*printf ("\n\rTamanho = %i", size);*/
               read(file, (char *)&lastcol, 2);
               /*printf ("\n\rLastcol = %i", lastcol);*/

               read(file, (char *)&lastrow, 2);
               /*printf ("\n\rLastrow = %i", lastrow);*/

               read(file, (char *)&size, 2);
               /*printf ("\n\rTamanho = %i", size);*/

               read(file, colwidth, sizeof(colwidth));
               /*printf ("\n\rColwidth = %i", colwidth);*/

               total_cell = (lastrow + 1) * (lastcol + 1);
               /*printf ("\n\rTotal de celulas = %i", total_cell);*/

               /*
               printf ("\nTecle Algo !");
               getch ();
               */

               clrscr ();
```

```
do {
     contador ++;

     /*timer ();*/

     /* zerando variaveis */
     strcpy (auxiliar, "");
     v = 0.0;

     if (read(file, (char *)&curcol, 2) <= 0);

     read(file, (char *)&currow, 2);

     read(file, &format[curcol][currow], 1);

     read(file, (char *)&size, 2);


     read(file, (char *)&rec, size);

     gotoxy (1,10);
     printf ("Coluna = %i    ", curcol);

     gotoxy (1,11);
     printf ("Linha = %i     ", currow);


     switch (rec.attrib) {

             case TEXT :

              strcpy (auxiliar,rec.v.text);
             break;

             case VALUE :

              v = rec.v.value;
              gcvt (v, 10, auxiliar);
             break;

             case FORMULA :

              v = rec.v.f.fvalue;
              gcvt (v, 10, auxiliar);
             break;
                        }  /* switch */

     if (curcol >= 1  &&
         curcol <= 26 &&
         currow >= 14 &&
         currow <= 195) {

         if (currow == flag_linha_valor+14) {
```

```
        i = abs (currow-14-flag_linha_valor) + conta_i;
        j = abs (curcol-2);

        if (v != 0) {

            if (flag_linha_leitura_val == OK) {
                matriz_valor [i][j] = fabs (v);
                flag_linha_leitura_val = NOK; }

            else matriz_valor [i][j] = 0;

            aux[i][j] = (fabs (v))/v;
            if (i >= m) m = i;
            if (j >= n) n = j;
            conta_ii++;

                    }


        else aux[i][j] = 0;


        if (max_i <= i) max_i = i;
        if (max_j <= j) max_j = j;

        if (curcol == 1) {
            flag_linha_valor += 3;
            conta_i++;
            flag_linha_leitura_val = OK;
                          }

                                    }

    if (currow == flag_linha_dp+15) {

        i = abs (currow-15-flag_linha_dp) + conta_i - 1;
        j = abs (curcol-2);

        /*matriz_dp [i][j] = v;*/

        if (v != 0) {

            if (flag_linha_leitura_dp  == OK) {
                matriz_dp [i][j] = fabs (v);
                flag_linha_leitura_dp = NOK; }

            else matriz_dp [i][j] = 0;

            dpm [conta_iii] = v;
            if (i >= m) m = i;
            if (j >= n) n = j;
            conta_iii ++;
                    }
```

```
                        if (curcol == 1) {
                            flag_linha_dp += 3;
                            flag_linha_leitura_dp  = OK;
                                        }
                                                }

                                    } /* if ... */


                } while (contador < total_cell && currow < 193);

            if (close (file) != 0) exit (-1);


    printf ("\nPassei 5");

    conta_i = 1;


    for (i=0; i<=m+1; i++) {

         for (j=0; j<=n+1; j++) {


              if (matriz_valor [i][j] > 0.0) {

                   r   [conta_i] = matriz_valor [i][j];
                   dpm [conta_i] = matriz_dp    [i][j];
                   gc  [conta_i] = matriz_dp    [i][j];

                   conta_i++;
                                              }
                                    }
                               }

           restricoesi ();
    printf ("\n3 Processando pesoi");
           pesoi ();
    printf ("\n4 Processando matrizi");
           matrizi ();
    printf ("\n5 Processando cond_tri");
           cond_tri ();
    printf ("\n6 Processando confere");
           confere ();
    printf ("\n7 Processando if (cont != 0) break;");
           if (cont != 0) exit (1);
    printf ("\n8 Processando calcula");
           calcula ();
    printf ("\n9 Processando respostai");
           respostai ();
    printf ("\n10 Processando break");

    } /* load3 */
```

```
/* ---------------------- */
/*

  (* abm  == ajuste  balanco  de  massas                          *
 (* abmet  == ajuste  balanco  metalurgico                        *
 (*

 ----------------------
 comentario
void  opcao    escolha  da  opcao  para  o  ajuste                *
 (*

 ----------------------
 comentario
void  rest_retri    recebe  as  restricoes  usadas  no  abm       *
 (*

 ----------------------

 comentario
void  entradai    entrada  de  do  para  o  abm                   *
 (*
 ----------------------
 comentario
void  entradaii    entrada  de  do  para  o  abmet                *
 (*

 ----------------------
 comentario
void limpam   limpa a matriz e um; auxiliar  para  os  calculos*
  (*

 ----------------------
 comentario
void  limpami    limpa  so  a  matriza  ser  utilizada           *
 (*

 ----------------------
 comentario
void  restricoesii    resticoes  para  o  abmet                  *
 (*

 ----------------------
 comentario
void  restricoesi    entrada  de  resticoes  para  o  abm        *
 (*

 ----------------------
 comentario
void  pesoi; calcula  o  1/v  para  o  abm                       *
 (*

 ----------------------
```

```
 comentario
void  pesoii     calcula  o  1/v  para  o  abmet                    *
 (*

 ____________________

 comentario
void  matrizi     preenche a matriz  para  os  calculos  do  abm  *
 (*

 ____________________

 comentario
void  matrizii    preenche a matriz para os  calculos  do  abmet *
 (*

 ____________________

 comentario
void  cond_tri    efetua  a  condensacao  pivotal  e  a
                  triangularizacao                                  *
  (*


 ____________________

 comentario
void  calcula     efetua  o  calculo  da  matriz  triangularizada *
 (*

 ____________________

 comentario
void  respostai   resposta  para  o  abm                            *
 (*

 ____________________

 comentario
void  respostaii    resposta  para  o  abmet                        *
 (*

 ____________________

 comentario
void  impressaom    impressao  do  abm                              *
 (*

 ____________________

 comentario
void  impressaot    impressao  do  abmet                            *
 (*

 ____________________

 comentario
void  flop    mostra;   sao  os  fluxos  e  as  operacoes          *
 (*

 ____________________

 comentario
```

```
void  confere   confere se a diagonal da mat.  a  "e  if  zero  *
 (*

 ____________________

 comentario
void  estatistica    calcula  media  e  desvio  padrao          *

 ____________________

 (* variaveis  globais                                          *
 (*  opcaoi  - bal.massas/bal.metalurgico/bal.massas  e  metal. *
 (*          m  - numero  de  fluxos                             *
 (*          n  - numero  de  operacoes                          *
 (*          respp  - desvio  padrao/grau  de  incerteza/grau de
                      certeza                                  *
 (*          respb  - char  balanco/ajustar  balanco             *
 (*          resp  - imprimir  ou  nao                           *
 (*          respe  - calculo  estatisticos  ou  nao             *
 (*          gi  - grau  de  incerteza                           *
 (*          gc  - grau  de  certeza                             *
 (*          q  - armazena  quantas  vezes  o  fluxo  aparece no
                  processo                                     *

 (*          peso  - do  do  a  partir  do  desvio  padrao e q   *
 (*          massa  - porcentagem  da  massa                     *
 (*          teor  - porcentagem  do  teor                       *
 (*          r  - valor  experimental  da  massa                 *
 (*          t  - valor  experimental  do  teor                  *
 (*          rc  - valor  do  da  massa                          *
 (*          tc  - valor  do  do  teor                           *
 (*          dpt  - desvio  padrao  do  teor                     *
 (*          dpm  - desvio  padrao  da  massa                    *
 (*          y  - auxiliar  no  calculo  da  matriz              *
 (*          lag  - lagrangiano                                  *
 (*          a  - matriz  usada  para  os  calculos              *
 (*          aux - matriz auxiliar para armazenar as retricoes   *
 (* variaveis  locais                                           *
 (*          i  - do  auxiliar                                   *
 (*          j  - do  auxiliar                                   *
 (*          l  - auxiliar  calculo  da  matriz                  *
 (*          lin  - auxiliar  calculo  da  matriz                *
 (*          p  - auxiliar  calculo  da  matriz                  *
 (*          max  - auxiliar  calculo  da  matriz                *
 (*          e  - auxiliar  calculo  da  matriz                  *
 (*          x  - aux */

/*
   Funcao de alerta ao usuario, fazendo soar tres beeps
*/

#include <dos.h>

void beep ()
```

```
{
 sound (523.25);
 delay (100);
 nosound ();
 sound (523.25);
 delay (100);
 nosound ();
 sound (523.25);
 delay (100);
 nosound ();
}

#include <conio.h>

unidade ()
{
 textattr (YELLOW + (BLUE<<4));
 gotoxy (7,2);
 cputs ("EPUSP - DEPTO. ENG. MINAS");

 gotoxy (18,23);
 cputs ("        Engenheiro Antonio Carlos Nunes       ");
}


#include <dos.h>
#include <string.h>

/* variaveis globais */
unsigned static int type_cursor_2;

/* funcoes */

/* apaga cursor */
void cursoff ();

/* restaura status original do cursor */
void curson (unsigned int shape);

/* Returns the shape of the current cursor */
unsigned int getcursor_2 (void);


void cursor (char tipo [10])
{
 if (strcmpi (tipo, "get")     == 0)  type_cursor_2=getcursor_2 ();

 if (strcmpi (tipo, "get_off") == 0) {type_cursor_2=getcursor_2 (); cursoff ();}

 if (strcmpi (tipo, "off")     == 0)  cursoff ();

 if (strcmpi (tipo, "on")      == 0)  curson (type_cursor_2);
}
```

```
/* ---------------------- */
/* apaga cursor */
void cursoff ()
{
 union REGS reg;

 reg.h.ah = 1;
 reg.h.ch = 0x20; /* ligando bit 5  */
 reg.h.cl = 0;    /*bios->cursBottom*/
 int86 (0x10, &reg, &reg);
}

/* ---------------------- */
/* restaura status original do cursor */
void curson (unsigned int shape)
{
 union REGS reg;

 reg.h.ah = 1;
 reg.x.cx = shape;
 int86(0X10, &reg, &reg);
}



/* ---------------------- */
/* Returns the shape of the current cursor */
unsigned int getcursor_2 (void)

{
 union REGS reg;

 reg.h.ah = 3;
 reg.h.bh = 0;
 int86(0X10, &reg, &reg);
 return(reg.x.cx);
} /* getcursor_2 */

/* ---------------------- */

/*
  Esta rotina oblitera o arquivo especificado e
  depois o apaga, modificando data e hora,  im-
  possibilitando sua recuperacao.

*/

#include <stdio.h>
#include <alloc.h>
#include <io.h>

void deletar (char nome_arquivo [12])
```

```
{
 FILE *delet_arquivo;
 char *buffer_del;
 int handle, buffsize_del;
 unsigned long tamanho;
 long posicao;
 char novo_nome [12];

 struct ftime filetime;

 novo_nome [0]  = 255;
 novo_nome [1]  = 255;
 novo_nome [2]  = 0;
 novo_nome [3]  = 1;
 novo_nome [4]  = 2;
 novo_nome [5]  = 3;
 novo_nome [6]  = 4;
 novo_nome [7]  = 5;
 novo_nome [8]  = 46;
 novo_nome [9]  = 255;
 novo_nome [10] = 255;
 novo_nome [11] = 255;


 delet_arquivo = fopen (nome_arquivo, "r+");

 if (delet_arquivo == NULL) return;

 /* ponteiro para o final do arquivo */
 fseek (delet_arquivo, 0L, SEEK_END);

 /* verifica tamanho do arquivo */
 tamanho = ftell(delet_arquivo);


 if (tamanho <= 32767) buffsize_del = tamanho;

 else buffsize_del = 32767;


 buffer_del = (unsigned char *) malloc (buffsize_del);

 if (buffer_del == NULL) return;


 handle = fileno (delet_arquivo);

 posicao = 0;
 do {

     fseek (delet_arquivo, posicao, SEEK_SET);
     write (handle, buffer_del, buffsize_del);
     posicao = posicao + buffsize_del;
```

```
    } while (posicao<tamanho);

 fclose (delet_arquivo);

 free (buffer_del);

 /* modifica dia, hora, etc. ... */
 delet_arquivo = fopen (nome_arquivo, "w");
 handle = fileno (delet_arquivo);
 filetime.ft_tsec  = 0;
 filetime.ft_min   = 0;
 filetime.ft_hour  = 0;
 filetime.ft_day   = 0;
 filetime.ft_month = 0;
 filetime.ft_year  = 0;
 setftime(handle, &filetime);

 fclose (delet_arquivo);

 /* modifica nome do arquivo */
 rename (nome_arquivo,novo_nome);

 /* deleta arquivo */
 unlink (novo_nome);

}


#include <conio.h>

/* termina o programa */
void EXIT_1 ()
{
  screen_d (47,19,77,23,14,4,7,1,4);
  gotoxy (48,20);
  cprintf ("ERRO DE PROCESSAMENTO EM PON-");
  gotoxy (48,21);
  cprintf ("TO FLUTUANTE !  PROGRAMA TER-");
  gotoxy (48,22);
  cprintf ("MINADO ! ");


  beep ();
  beep ();

  window (1,1,80,25);
  gotoxy (1,24);

  exit (1);
}

#include <bios.h>
```

```
int getkey_d  (int tipo)
{
  int key, lo, hi;

  if (tipo == 1) do { timer (); } while ( !kbhit ());

  if (tipo == 2) do { VGA_CARD (); timer (); } while ( !kbhit ());

  key = bioskey(0);
  lo = key & 0X00FF;
  hi = (key & 0XFF00) >> 8;
  return((lo == 0) ? hi + 256 : lo);
}
```

```
#include <bios.h>

/* recebe caracter/numero do teclado */
char getkey_s (int tipo)
{
  union key {
             int i;
             char ch [2];
            } k;

  if (tipo == 1) do { timer (); } while ( !kbhit ());

  if (tipo == 2) do { timer (); VGA_CARD (); } while ( !kbhit ());


  k.i = bioskey (0);
  return k.ch [1];
}

/*
  Utilizando  getkey_d ()  recebe conjunto de caracteres
  do teclado, nao aceitando algumas teclas especificadas

  Necessita da funcao getkey_d (...);
            da funcao rewrite  (...);
            da funcao beep        ();
            da funcao rewrite  (...);

  int cor_fore           cor do texto
  int cor_back           cor de fundo
  int x                  coluna inicial
  int y                  coluna final
  int num_caracteres     numero maximo de caracteres
  double valor_minimo    valor minimo
  double valor_maximo    valor maximo
  char *texto            ponteiro para texto
  int tipo               tipo = 1 (qualquer caracter ASCII),
                         tipo = 2 (idem,porem nao aceita ASCII 32 e 255)
```

```
                        tipo = 3 (aceita apenas numeros,ponto e sinais + ou -)
  int vga_timer         vga_timer = 1 (apenas relogio)
                        vga_timer = 2 (relogio e mensagem)

*/

#include <conio.h>
#include <stdlib.h>
#include <string.h>

/* definicoes */

#define MAX_VALOR     +1.7e+300
#define MIN_VALOR     -1.7e+300

#define BACKSPACE              8
#define ENTER                 13
#define BARRA                 32
#define CTRL_ALT_255         255
#define HOME                 327
#define SETA_ALTO            328
#define SETA_ESQUERDA        331
#define SETA_DIREITA         333
#define END                  335
#define SETA_BAIXO           336
#define INSERT               338 /* nao ativada */
#define DELETE               339
#define CTRL_SETA_ESQUERDA   371
#define CTRL_SETA_DIREITA    372
#define LETRAS                 1
#define LETRAS_N_BARRA         2

#define NUMEROS_N_BARRA        3
#define OK                     1
#define N_OK                   0

/* funcoes */
int check_caracter (int keyb_code, int tipo, int contador);
int check_range (double valor_minimo, double valor_maximo, char *texto, int tipo);

/* globais para verificar ponto em chek_range */
int conta_ponto,
    conta_sinal,
    posicao_ponto;

/* programa */
char *getsf (int cor_fore, int cor_back, int x, int y, int num_caracteres, double
            valor_minimo, double valor_maximo,char *texto,int tipo,int vga_timer)
{

unsigned char caracter;
int contador      = 0,
    flag_contador = OK,
```

```
    flag_signal    = OK,
    keyb_code,
    temporario;

/* inicializando variaveis */
conta_ponto    =  0;
conta_sinal    =  0;
posicao_ponto = -1;

 do {texto [contador] = 0; contador++;} while (contador<=num_caracteres);

 contador = 0;

 do {

  do {
    caracter = 0;

    flag_signal = OK;

    for (temporario = 0; temporario < num_caracteres; temporario++) {
             textattr (cor_fore + (cor_back<<4));
             gotoxy (x+temporario,y);
             cprintf ("%s", " ");}

    textattr (cor_fore + (cor_back<<4));
    gotoxy (x,y);
    cprintf ("%s", texto);

    if (contador < num_caracteres) rewrite (x+contador, y, 79);

    keyb_code = getkey_d (vga_timer);
    if (contador < num_caracteres) flag_contador = OK;

    else flag_contador = N_OK;

    if (keyb_code == ENTER      ||
        keyb_code == SETA_ALTO  ||
        keyb_code == SETA_ESQUERDA ||
        keyb_code == SETA_DIREITA  ||
        keyb_code == SETA_BAIXO  ) break;

    else {

          if (keyb_code == DELETE) {

              texto [contador] = ' ';

              textattr (cor_back + (cor_back<<4));
              gotoxy (x,y);
              cprintf ("%s", texto);

              textattr (cor_fore + (cor_back<<4));
```

```
        gotoxy (x,y);
        cprintf ("%s", texto);

        /* so para eventualmente resetar os flags de ponto */
        check_caracter (48, tipo, contador);

                        }

    if (keyb_code == CTRL_SETA_ESQUERDA && contador > 0) contador--;

    if(keyb_code==CTRL_SETA_DIREITA && contador<num_caracteres-1)contador++;

    if (keyb_code == HOME) contador = 0;

    if (keyb_code == END) {
        contador = strlen (texto);
        if (contador >= num_caracteres) flag_contador = N_OK;}

    if (keyb_code == BACKSPACE)  {

        if (contador > 0) {

            /* apagando texto anterior ao backspace */
            textattr (cor_back + (cor_back<<4));
            gotoxy (x,y);
            cprintf ("%s", texto);


            /* transferindo  */
            for(temporario=contador;temporario<=num_caracteres;++temporario)
            texto [temporario-1] = texto [temporario];

            /* para evitar que o backspace va' puxando o ultimo caracter */
            if (contador > num_caracteres) contador--;
            texto [contador] = 0;

            /* zerando resto */
            contador--;

            textattr (cor_fore + (cor_back<<4));
            gotoxy (x,y);
            cprintf ("%s", texto);

                        } /* contador > 0 */

                          } /* keyb_code */

    if (keyb_code >= BARRA              &&
        keyb_code <= CTRL_ALT_255       ||
        keyb_code == DELETE             ||
        keyb_code == CTRL_SETA_ESQUERDA ||
        keyb_code == CTRL_SETA_DIREITA  ||
        keyb_code == BACKSPACE          ||
```

```
                keyb_code == HOME                ||
                keyb_code == END ) flag_signal = OK;

            else {
                  beep ();
                  flag_signal = N_OK;
                 }

            if (keyb_code != DELETE             &&
                keyb_code != CTRL_SETA_ESQUERDA &&
                keyb_code != CTRL_SETA_DIREITA  &&
                keyb_code != BACKSPACE          &&
                keyb_code != HOME               &&
                keyb_code != END                &&
                check_caracter (keyb_code, tipo, contador) != N_OK) {

                if (flag_signal   == OK &&
                    flag_contador == OK ) {
                    caracter = keyb_code;
                    texto [contador++] = toupper (caracter);
                    textattr (cor_fore + (cor_back<<4));
                    gotoxy (x,y);
                    cprintf ("%s", texto);

                                        } /* if flag_ ... */

                                      } /* if keyb_code ... */

             } /* else */

       } while (OK);

     } while (check_range (valor_minimo, valor_maximo, texto, tipo)!=OK);

     /* zera o resto para evitar o envio de lixo */
     contador = strlen (texto);
     do {texto [contador++] = 0;} while (contador<num_caracteres);

     for (temporario = 0; temporario < num_caracteres; temporario++) {
          textattr (cor_fore + (cor_back<<4));
          gotoxy (x+temporario,y);
          cprintf ("%s", " ");}

     textattr (cor_fore + (cor_back<<4));
     gotoxy (x,y);
     cprintf ("%s", texto);

    return (texto);

    }
```

```
/* ---------------- */
/* compara keyb_code com intervalos de caracteres ASCII */

int check_caracter (int keyb_code, int tipo, int contador)
{

if (tipo == LETRAS) return OK;

if (tipo == LETRAS_N_BARRA)  {

    if (keyb_code == 255 ||
        keyb_code ==  32) {beep (); return N_OK;}
    else return OK;
                              }

if (tipo == NUMEROS_N_BARRA) {

    if ((contador  == posicao_ponto) &&
        (keyb_code != 46)) {posicao_ponto=-1;conta_ponto=0;}

    if (keyb_code >=  58 ||
        keyb_code <=  42 ||
        keyb_code ==  44 ||
        keyb_code ==  47) {beep (); return N_OK;}

    if (keyb_code == 46) {

        if (conta_ponto == 0) {conta_ponto++; posicao_ponto=contador;return OK;}
        else {beep (); return N_OK;}

                          }


    if (keyb_code == 43 ||
        keyb_code == 45) {

         if (contador == 0) {conta_sinal++; return OK;}
         else {beep (); return N_OK;}

                          }
                               }
return OK;

}
/* ---------------- */
int check_range (double valor_minimo, double valor_maximo, char *texto, int tipo)
{

 if (tipo == NUMEROS_N_BARRA     &&
     (atof(texto) > valor_maximo ||
      atof(texto) < valor_minimo ||
      atof(texto) > MAX_VALOR    ||
      atof(texto) < MIN_VALOR))  {
```

```
                                beep();
                                return N_OK;
                               }
 else return OK;

}
/* ---------------- */

/* efetua a impressao de arquivos */

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <alloc.h>
#include <dos.h>

#define NAO_OK   0
#define OK       1
#define NO_ACESS 2

#define LPT1     0


/* funcoes */
void menu_press ();              /* menu para chamar a rotina de impressao */
int tipo_letra  ();              /* set up do tipo de letra                */
int num_linhas  ();              /* set up do numeros de linhas por pagina */
int num_linhas_salto ();         /* set up do numeros de linhas a serem sal-
                                    tadas por pagina */


void show_setup (
     char arquivo [13],
     int letra,
     int linhas_pagina,
     int linhas_salto,
     int config);           /*mostra configuracao atual de impressao */

void not_found ();               /* avisa que determinado arquivo nao  foi
                                    encontrado                          */
void no_file   ();               /* avisa que nenhum arquivo foi  digitado
                                    ou encontrado */



void no_prn_file ();             /* avisa que o arquivo  CONFIG01.PRN  nao
                                    foi aberto/encontrado */

void no_prn_file_2 ();           /* avisa que o arquivo  CONFIG01.PRN  nao
                                    foi gerado */

void prn_file ();                /* avisa que o arquivo  CONFIG01.PRN  foi
                                    gerado */
```

```
int  show_printer_status ();      /* mostra situacao da impressora        */


/* variaveis globais */

char nome[15];
int tipo = 0, linhas_pagina = 60, linhas_salto = 7;
int config_flag = NAO_OK;


char *texto [] = {

        "A - Default                          ",
        "B - Default Sublinhado               ",

        "C - Wide                             ",
        "D - Wide Sublinhado                  ",

        "E - Bold                             ",
        "F - Bold Sublinhado                  ",

        "G - Enfatizado                       ",
        "H - Enfatizado Sublinhado            ",

        "I - Elite                            ",
        "J - Elite Sublinhado                 ",

        "K - Condensado                       ",
        "L - Condensado Sublinhado            ",

        "M - Subscrito                        ",
        "Tecle sua opcao  PgDn outras opcoes",

        "N - Superescrito                     ",
        "O - Qualidade Carta                  ",

        "P - Qualidade Carta Subl.            ",
        "Q - It lico                          ",

        "R - Wide & Bold                      ",
        "S - Wide & Bold Subl.                ",

        "T - Wide & Bold Qual. Carta          ",
        "U - Wide & Bold Qual. Carta Subl.    ",


        "V - Wide & Bold Elite                ",
        "                                     ",

        "                                     ",
        "                                     ",

        "                                     ",
```

```
        "Tecle sua opcao  PgUp outras opcoes",};

imprimir ()
{

FILE *fp, *fp_2;
char amostra [143], opcao_x, opcao_x2;
int opcao;

int conta_linha = 0, conta_salto = 0;

int flag_file = NAO_OK;

int bufsize;
char *buffer;


strcpy (nome, "___________");


bufsize = 2 * (79-47+1) * (21-13+1);
if ((buffer = (char *) malloc(bufsize)) != NULL) {
if (gettext (47,13,79,21, buffer)) {


if ((fp_2=fopen("config01.prn","rt"))==NULL) no_prn_file ();

else {
      prn_file ();

      fgets (amostra, 143, fp_2);
      tipo = atoi (amostra);

      fgets (amostra, 143, fp_2);
      linhas_pagina = atoi (amostra);

      fgets (amostra, 143, fp_2);
      linhas_salto = atoi (amostra);

      fclose (fp_2);

      config_flag = OK;
     }
     puttext (47,13,79,21, buffer);    }}

show_setup (nome, tipo, linhas_pagina, linhas_salto, config_flag);


menu_press ();


do {
```

```
do {

    opcao = 0;

    switch (opcao_x = getkey_s (2)) {

            case 2:  /* 1 arquivo       */
             opcao = 1;
            break;

            case 30: /* A arquivo       */
             opcao = 1;
            break;

            case 3:  /* 2 tipo letra    */
             opcao = 2;
            break;

            case 20: /* T tipo letra    */
             opcao = 2;
            break;

            case 4:  /* 3 linhas/pagina */
             opcao = 3;
            break;

            case 38: /* L linhas/pagina */
             opcao = 3;
            break;

            case 5:  /* 4 imprimir      */
             opcao = 4;
            break;

            case 23: /* I imprimir      */
             opcao = 4;
            break;

            case 6:  /* 5 salvar        */
             opcao = 5;
            break;

            case 31: /* s salvar        */
             opcao = 5;
            break;
            case 1: /* Esc - retorna */

             bufsize = 2 * (62-23+1) * (17-11+1);

             if ((buffer = (char *) malloc(bufsize)) != NULL) {
                 if (gettext (23,11,62,17, buffer)) {

                     screen_d (23,11,58,13,14,4,7,1,4);
```

```
                    textattr (YELLOW + (RED<<4));
                    gotoxy (24,12);
                    cputs ("Deseja  sair  deste modulo ? (S/N)");

                    opcao_x2 = getkey_s (2);

                    if (opcao_x2 == 31) return;

                    else puttext (23,11,62,17, buffer);

                                                    }
                                                        }

            break;

            case 19: /* R - retorna */

             bufsize = 2 * (62-23+1) * (17-11+1);
             if ((buffer = (char *) malloc(bufsize)) != NULL) {
                 if (gettext (23,11,62,17, buffer)) {

                    screen_d (23,11,58,13,14,4,7,1,4);
                    textattr (YELLOW + (RED<<4));
                    gotoxy (24,12);
                    cputs ("Deseja  sair  deste m"dulo ? (S/N)");

                    if (getkey_s (2) == 31) return;

                    else puttext (23,11,62,17, buffer);

                                                    }
                                                        }
             break;
                                }

    } while (opcao_x !=  2 && opcao_x != 30 && opcao_x !=  3 &&
             opcao_x != 20 && opcao_x !=  4 && opcao_x != 38 &&
             opcao_x !=  5 && opcao_x != 23 && opcao_x !=  1 &&
             opcao_x != 19 && opcao_x !=  6 && opcao_x !=  31);


if (opcao == 1) {

    screen_d (4,13,36,15,11,1,7,1,1);
    gotoxy (5,14);

    cputs ("Qual o arquivo ? ___________");
    getsf (11,1,22,14,12,0.0,0.0,nome,2,2);
     bufsize = 2 * (79-47+1) * (21-13+1);
     if ((buffer = (char *) malloc(bufsize)) != NULL) {

         if (gettext (47,13,79,21, buffer)) {
```

```
            if (procurar (nome) == 0) {
                not_found ();
                flag_file = NAO_OK;
                strcpy (nome, "___________");}

             else flag_file = OK;

          puttext (47,13,79,21, buffer);}}


    show_setup (nome, tipo, linhas_pagina, linhas_salto, config_flag);

    menu_press ();

              } /* if opcao == 1 */

if (opcao == 2) {tipo = tipo_letra (); show_setup (nome, tipo, linhas_pagina,
                 linhas_salto, config_flag); menu_press ();}

if (opcao == 3) {
    linhas_pagina = num_linhas ();
    show_setup (nome, tipo, linhas_pagina, linhas_salto, config_flag);
    linhas_salto = num_linhas_salto ();
    show_setup (nome, tipo, linhas_pagina, linhas_salto, config_flag);
    menu_press ();}

if (opcao == 4) {

    bufsize = 2 * (79-47+1) * (21-13+1);
    if ((buffer = (char *) malloc(bufsize)) != NULL) {
    if (gettext (47,13,79,21, buffer)) {

        if (flag_file == NAO_OK) {
                                  no_file ();

                                  show_setup (nome, tipo, linhas_pagina, linhas_salto,
                                              config_flag);
                                 }

        else {
              show_setup (nome, tipo, linhas_pagina, linhas_salto, config_flag);

              show_printer_status ();
             }

    puttext (47,13,79,21, buffer);


    menu_press ();}}


    if (flag_file == OK) {
```

```
reset_pr  ();

bufsize = 2 * (79-47+1) * (21-13+1);
if ((buffer = (char *) malloc(bufsize)) != NULL) {
if (gettext (47,13,79,21, buffer)) {

screen_d (47,13,77,16,0,3,7,1,3);
gotoxy (48,14);
cprintf (" Aguarde  alguns instantes ! ");
gotoxy (48,15);
cprintf ("  Imprimindo:  %s", nome);

if ((fp=fopen(nome,"r"))==NULL) {
    printf("\nNao posso abrir o arquivo solicitado!");
    exit(1);}

while (fgets (amostra, 143, fp) != NULL) {

 if (tipo == 0)  printer_ ("norm", amostra); /* IMPRESSAO NO MODO DEFAULT      */
 if (tipo == 1)  printer_ ("und_", amostra); /* DEFAULT SUBLINHADO/UNDERLINED */
 if (tipo == 2)  printer_ ("wide", amostra); /* WIDE                          */
 if (tipo == 3)  printer_ ("WDSU", amostra); /* WIDE SUBLINHADO               */
 if (tipo == 4)  printer_ ("bold", amostra); /* BOLD                          */
 if (tipo == 5)  printer_ ("BOSU", amostra); /* BOLD SUBLINHADO               */
 if (tipo == 6)  printer_ ("emph", amostra); /* ENFATIZADO                    */
 if (tipo == 7)  printer_ ("efud", amostra); /* ENFATIZADO SUBLINHADO         */
 if (tipo == 8)  printer_ ("elit", amostra); /* ELITE                         */
 if (tipo == 9)  printer_ ("elts", amostra); /* ELITE  SUBLINHADO             */
 if (tipo == 10) printer_ ("cond", amostra); /* CONDENSADO                    */
 if (tipo == 11) printer_ ("cdud", amostra); /* CONDENSADO SUBLINHADO         */
 if (tipo == 12) printer_ ("sub_", amostra); /* SUBSCRITO                     */
 if (tipo == 13) printer_ ("sup_", amostra); /* SUPERESCRITO                  */
 if (tipo == 14) printer_ ("NLQ_", amostra); /* NLQ - Near Letter Quality     */
 if (tipo == 15) printer_ ("NLQS", amostra); /* NLQ Near Letter
                                                Quality-Sublinhado            */
 if (tipo == 16) printer_ ("ital", amostra); /* ITALICO                       */
 if (tipo == 17) printer_ ("WIBO", amostra); /* WIDE & BOLD                   */
 if (tipo == 18) printer_ ("WIBS", amostra); /* WIDE & BOLD SUBLINHADO        */
 if (tipo == 19) printer_ ("WIBN", amostra); /* WIDE & BOLD & NLQ             */
 if (tipo == 20) printer_ ("WBNS", amostra); /* WIDE & BOLD & NLQ SUBLINHADO  */
 if (tipo == 21) printer_ ("WIBE", amostra); /* WIDE & BOLD & ELITE           */

        if ((conta_linha++) == linhas_pagina-2) {
            conta_linha = -1; /* foi necessario "zerar" em -1 */
            for (conta_salto=0; conta_salto <= linhas_salto-1; conta_salto++)
                                                printer_ ("norm","\n");
                                                 }
                                           } /* while */
fclose (fp);

puttext (47,13,79,21, buffer);}}

printer_ ("norm","\n");
```

```
     show_setup (nome, tipo, linhas_pagina, linhas_salto, config_flag);

     menu_press ();
     opcao = 19; /* para encerrar loop do-while */

     } /* if (flag_file == OK && flag_printer == OK)  */

                 } /* if opcao == 4 */


if (opcao == 5) {

     bufsize = 2 * (79-47+1) * (21-13+1);
     if ((buffer = (char *) malloc(bufsize)) != NULL) {
     if (gettext (47,13,79,21, buffer)) {


     if ((fp_2=fopen("config01.prn","wt"))==NULL) no_prn_file_2 ();

     else {
           prn_file ();

           fprintf (fp_2,"%i\n",tipo);
           fprintf (fp_2,"%i\n",linhas_pagina);
           fprintf (fp_2,"%i\n",linhas_salto);
           fprintf (fp_2,"%s\n","void");
           fprintf (fp_2,"%s\n","void");
           fprintf (fp_2,"%s\n","void");
           fprintf (fp_2,"%s\n","void");
           fprintf (fp_2,"%s\n","void");
           fprintf (fp_2,"%s\n","void");
           fprintf (fp_2,"%s\n","void");

           fclose (fp_2);
          }

      puttext (47,13,79,21, buffer);    }

      show_setup (nome, tipo, linhas_pagina, linhas_salto,  config_flag);
      menu_press ();}

      } /* if (opcao == 5) */

   } while (1);

}

/* ---------------------- */

/* menu para chamar a rotina de impressao */
void menu_press ()
{
```

```
menu (
        /* quadro */
        2,
        2,
        2,
        0,
        3,
        3,
        7,

        /* titulo */
        "IMPRESSAO DE ARQUIVOS",
        1,
        3,
        "c",
        1,

        /* texto */
        8,
        0,
        3,
        "e",
        "-Escolha  letra, numero ou Esc-",
        "",
        "1   - Selecionar Arquivo",
        "2   - Selecionar Tipo de Letra",
        "3   - Selecionar Linhas/Página",
        "4   - Imprimir",
        "5   - Salvar Configuracao",
        "Esc - Retorna ao menu principal",
        "",
        "",
        "",
        "",
        "",
        0,

        /* high-light */
        78,
        0,
        0,
        0,
        18,
        18,
        18,
        7,
        7,
        7,
        0,
        0,
        0,
        0,
        0);
}
```

```
/* ---------------------- */
/* set up do tipo de letra */
int tipo_letra ()
{
 int i, tipo_2 = 0;
 char opcao_x2;

 screen_d (3,3,33,13,14,4,7,1,14);
 gotoxy (4,4);
 cputs ("           ATENCAO           ");
 gotoxy (4,5);
 cputs ("Talvez nem  todos os tipos de");
 gotoxy (4,6);
 cputs ("letras  disponíveis na tabela");
 gotoxy (4,7);
 cputs ("ao lado poderao ser acessados");
 gotoxy (4,8);
 cputs ("pela impressora eventualmente");
 gotoxy (4,9);
 cputs ("conectada ao seu sistema.    ");
 gotoxy (4,10);
 cputs ("Verifique, no manual do usu -");
 gotoxy (4,11);
 cputs ("rio, tabela de compatibilida-");
 gotoxy (4,12);
 cputs ("de impressoras.              ");

 screen_d (35,3,71,19,14,1,7,1,1);

 textcolor (WHITE);
 gotoxy (40,4);
 cputs ("TIPOS DE LETRAS DISPONIVEIS");

 textcolor (YELLOW);
 for (i=5; i<=18; i++) {
      gotoxy (36,i);
      cprintf ("%s",texto [i-5]);}

 textcolor (WHITE);
 gotoxy (36,i-1);
 cprintf ("%s",texto [i-6]);

 do {

     switch (opcao_x2 = getkey_s (2)) {

            case 73:  /* opcoes de 1 a 12 */
             textattr (YELLOW + (BLUE<<4));
             for (i=5; i<=18; i++) {
                  gotoxy (36,i);
                  cprintf ("%s",texto [i-5]);}

             textcolor (WHITE);
```

```
                gotoxy (36,i-1);

                cprintf ("%s",texto [i-6]);
               break;

               case 81: /* opcoes de 13 a 21*/
                textattr (YELLOW + (BLUE<<4));
                for (i=5; i<=18; i++) {
                     gotoxy (36,i);
                     cprintf ("%s",texto [i+9]);}

                textcolor (WHITE);
                gotoxy (36,i-1);
                cprintf ("%s",texto [i+8]);
               break;

                                    }

    } while (opcao_x2 == 73 || opcao_x2 == 81);


 if (opcao_x2 == 30) tipo_2 = 0;  /* A */
 if (opcao_x2 == 48) tipo_2 = 1;  /* B */
 if (opcao_x2 == 46) tipo_2 = 2;  /* C */
 if (opcao_x2 == 32) tipo_2 = 3;  /* D */
 if (opcao_x2 == 18) tipo_2 = 4;  /* E */
 if (opcao_x2 == 33) tipo_2 = 5;  /* F */
 if (opcao_x2 == 34) tipo_2 = 6;  /* G */
 if (opcao_x2 == 35) tipo_2 = 7;  /* H */
 if (opcao_x2 == 23) tipo_2 = 8;  /* I */
 if (opcao_x2 == 36) tipo_2 = 9;  /* J */
 if (opcao_x2 == 37) tipo_2 = 10; /* K */
 if (opcao_x2 == 38) tipo_2 = 11; /* L */
 if (opcao_x2 == 50) tipo_2 = 12; /* M */
 if (opcao_x2 == 49) tipo_2 = 13; /* N */
 if (opcao_x2 == 24) tipo_2 = 14; /* O */
 if (opcao_x2 == 25) tipo_2 = 15; /* P */
 if (opcao_x2 == 16) tipo_2 = 16; /* Q */
 if (opcao_x2 == 19) tipo_2 = 17; /* R */
 if (opcao_x2 == 31) tipo_2 = 18; /* S */
 if (opcao_x2 == 20) tipo_2 = 19; /* T */
 if (opcao_x2 == 22) tipo_2 = 20; /* U */
 if (opcao_x2 == 47) tipo_2 = 21; /* V */

 /*if (opcao_x2 == 45) tipo_2 = 22; /* X */
 /*if (opcao_x2 == 21) tipo_2 = 23; /* Y */
 /*if (opcao_x2 == 44) tipo_2 = 24; /* Z */

 if (opcao_x2 == 28) tipo_2 = 0; /* enter */

 return tipo_2;
}
```

```
/* ---------------------- */
/* set up do numeros de linhas por pagina */
int num_linhas ()
{
 char nome [3];
 int numero;
 screen_d (4,13,36,15,11,1,7,1,1);
 gotoxy (5,14);
 cputs ("Numero  de  linhas/p gina:");
 getsf (11,1,32,14,3,0.0,100.0,nome,3,2);

 numero = atoi (nome);

 return (numero);

}

/* ---------------------- */
/* set up do numeros de linhas a serem saltadas por pagina */
int num_linhas_salto ()
{
 char nome [3];
 int numero;
 screen_d (4,13,36,15,11,1,7,1,1);
 gotoxy (5,14);
 cputs ("Linhas brancas entre p g.:");
 getsf (11,1,32,14,3,0.0,100.0,nome,3,2);

 numero = atoi (nome);

 return (numero);

}

/* ---------------------- */

/* mostra configuracao atual de impressao */
void show_setup (char arquivo [13], int letra, int linhas_pagina,
                 int linhas_salto, int config)
{

screen_d (2,14,51,22,15,4,7,1,4);

gotoxy (12,15);
cprintf ("Status da Configuracao de Impressao");

gotoxy (3,16);
cprintf ("Arquivo   : %s", arquivo);

gotoxy (3,17);
if (letra <= 12) cprintf ("Letra     : %s", texto [letra]);
else cprintf ("Letra     : %s", texto [letra+1]);
```

```
gotoxy (3,18);

cprintf ("Ln./P gina: %i", linhas_pagina);

gotoxy (3,19);
cprintf ("Ln./Salto : %i", linhas_salto);
gotoxy (3,20);
cprintf ("Impressora: Nao Solicitada  ");

gotoxy (3,21);
if (config == OK) cprintf ("Config.In.: CONFIG01.PRN");
else cprintf ("Config.In.: DEFAULT");

}
/* ---------------------- */

/* avisa que determinado arquivo nao foi encontrado */
void not_found ()
{
 screen_d (47,13,77,16,14,4,7,1,4);
 gotoxy (48,14);
 cprintf ("  Arquivo NO encontrado !   ");
 gotoxy (48,15);
 cprintf ("  VERIFIQUE NOME CORRETO !!  ");
 gotoxy (74,15);
 beep ();
 tecle (2);
}

/* ---------------------- */
/* avisa que nenhum arquivo foi digitado ou encontrado */
void no_file ()
{
 screen_d (47,13,77,16,14,4,7,1,4);
 gotoxy (48,14);
 cprintf ("  Arquivo NO especificado!  ");
 gotoxy (48,15);
 cprintf ("   DIGITE NOME CORRETO !!!   ");
 gotoxy (74,15);
 beep ();
 tecle (2);
}
/* ---------------------- */
/* avisa que o arquivo CONFIG01.PRN nao foi aberto/encontrado */
void no_prn_file ()
{
 screen_d (47,13,77,16,14,4,7,1,4);
 gotoxy (48,14);
 cprintf ("Imposs!vel ler CONFIG01.PRN !");
 gotoxy (48,15);
 cprintf ("Adotada configuraco default!");
 gotoxy (74,15);
 beep ();
```

```
 tecle (2);
}


/* ---------------------- */
/* avisa que o arquivo CONFIG01.PRN nao foi gerado */
void no_prn_file_2 ()
{
 screen_d (47,13,77,16,14,4,7,1,4);
 gotoxy (48,14);
 cprintf ("   Problemas na geracao do   ");
 gotoxy (48,15);
 cprintf ("  arquivo CONFIG01.PRN  !!!  ");
 gotoxy (74,15);
 beep ();
 tecle (2);
}

/* ---------------------- */
/* avisa que o arquivo CONFIG01.PRN foi gerado */
void prn_file ()
{
 screen_d (47,13,77,16,11,1,7,1,1);
 gotoxy (48,14);
 cprintf ("    ARQUIVO  CONFIG01.PRN    ");
 gotoxy (48,15);
 cprintf ("          GERADO !!          ");
 gotoxy (74,15);
}

/* ---------------------- */
/* mostra se a impressora esta, ou nao, pronta */
int show_printer_status ()
{

 textattr (WHITE + (RED<<4));

 gotoxy (3,20);
 cprintf ("Impressora: Nao Esta Pronta ");

 reset_pr  (OK);

 gotoxy (3,20);
 cprintf ("Impressora: Solicitada      ");

 return (OK);
}

/* ---------------------- */

/* monta o menu principal */
```

```
void m_menu ()
{
  menu (
        /* quadro */
        26,
        6,
        2,
        11,
        1,

        1,
        7,
        /* titulo */
        "SOFTWARE",
        14,
        1,
        "c",
        1,

        /* texto */
        9,
        15,
        1,
        "e",

        "   TECLE SUA OPCAO   ",
        "",
        "F-1 HELP",
        "F-2 SHELL",
        "F-3 PLANILHA",
        "F-4 BALANCOS",
        "F-5 ESTATISTICA",
        "F-6 IMPRESSAO",
        "Esc RETORNAR",
        "",
        "",
        "",
        "",
        "",
        1,

        /* high-light */
        78, /*sugestao de cor: amarelo com fundo vermelho */
        0,
        0,
        0,
        5,
        5,
        5,
        5,
        5,
        5,
        5,
        0,
```

```
        0,
        0,
        0,
        0);
}

/* ------------------- */

/*
  verifica se arquivo *.BMS, caso  nao  exista  e'
  criado um novo arquivo `a partir de BASEMAS.BMS */

#include <string.h>
#include <stdio.h>
#include <io.h>
#include <conio.h>
#include <alloc.h>     /* para alocacao de memoria                    */
#include <process.h>   /* para executar outros programas              */
#include <ctype.h>     /* para conversao de bmsing em maiuscula       */
#include <sys\stat.h>  /* para verificar e mudar atributo read only   */
#include <sys\types.h> /* para verificar e mudar atributo read only   */

/* funcoes */
void opcoes_false (int tipo);
void open_ok (char nome [12]);
void nao_procalc (char nome [12]);
void EXIT_1a (char filename [12]);

void new_old3 ()
{

int flag = 1000, opcao_toascii, opcao_rename_file, tipo;

char nome [14], nome_saida [14];
char opcao;
char tipo_x;
char *argv [3];

argv [0] = "procalc.exe";
argv [1] = nome;
argv [2] = NULL;

 do {
     opcao = getkey_s (2);

     switch (opcao) {

            case  2: /* Editar arquivo existente */
            case 18:
             tipo = 1;
            break;

            case  3: /* criar arquivo estatistica */
            case 20:
```

```
                tipo = 2;
               break;

               case  4: /* criar arquivo Balanco */
               case 48:
                tipo = 3;
               break;

               case  1: /* Retorna */
               case 19:

                screen_d (4,11,37,13,14,4,7,1,4);

                textattr (YELLOW + (RED<<4));
                gotoxy (5,12);
                cputs ("Deseja sair deste modulo ? (S/N)");

                opcao = getkey_s (2);

                if (opcao == 31) opcoes ();
                else opcoes_false (2);

               break;

                       } /* end switch */

    } while (opcao !=  2 &&
             opcao != 18 &&
             opcao !=  3 &&
             opcao != 20 &&
             opcao !=  4 &&
             opcao != 48 &&
             opcao !=  1 &&
             opcao != 19);

if (tipo == 1) {
    txtstor2 ('s', 1, 1, 80, 24);
    screen_d (6,14,37,16,11,1,7,1,1);
    gotoxy (7,15);
    cputs ("Qual o arquivo ? ________.BMS");
    getsf (11,1,24,15,8,0.0, 0.0, nome, 2, 2);

    if (strlen (nome) <= 0) {txtstor2 ('r', 1, 1, 80, 24); procalc3 (); new_old3 ();}
    else {txtstor2 ('r', 1, 1, 80, 24);strcat (nome,".BMS");}

    flag = procurar (nome);

    if (flag == 1) {
        screen_d (47,13,77,16,0,3,7,1,3);
        gotoxy (48,14);
        cprintf ("    Arquivo encontrado !     ");
        gotoxy (48,15);
        cprintf ("Chamando Procalc %s", nome);
        if (spawnvp (P_WAIT,"PROCALC.EXE", argv)==-1) nao_procalc (nome);
```

```
        opcoes_false (1);}

    if (flag == 0) {
        txtstor2 ('s', 1, 1, 80, 24);
        screen_d (35,13,77,16,14,4,7,1,4);
        gotoxy (36,14);
        cprintf ("  Arquivo %s NAO encontrado !  ", nome);
        gotoxy (36,15);
        cprintf ("Escolha opcao 2 ou 3 para criar arquivo !");
        beep ();
        delay (3000);
        txtstor2 ('r', 1, 1, 80, 24);

        new_old3 ();
                  }
              } /*if (tipo == 1)*/


if (tipo == 2) {

    txtstor2 ('s', 1, 1, 80, 24);
    screen_d (6,14,37,16,11,1,7,1,1);
    gotoxy (7,15);
    cputs ("Qual o arquivo ? ________.BMS");
    getsf (11,1,24,15,8,0.0,0.0,nome,2,2);

    if (strlen (nome) <= 0) {txtstor2 ('r', 1, 1, 80, 24); procalc3 (); new_old3 ();}
    else {txtstor2 ('r', 1, 1, 80, 24);strcat (nome,".BMS");}

    flag = procurar (nome);

    if (flag == 1) {
        txtstor2 ('s', 1, 1, 80, 24);
        screen_d (43,13,77,16,14,4,7,1,4);
        gotoxy (44,14);
        cprintf ("Existe arquivo %s !!!", nome);
        gotoxy (44,15);
        cprintf ("Deseja substituir arquivo ? (S/N)");
        beep ();
        do opcao = getkey_s (2); while (opcao != 31 && opcao != 49);
        txtstor2 ('r', 1, 1, 80, 24);
        if (opcao == 49) opcoes_false (2);
        else {
              opcao = 2;
              flag  = 0;
              deletar (nome);
             }
                   }

    /* verifica se arquivo BASECLAR.BMS existe, abrindo um novo arquivo */
    if ((opcao == 2 || opcao == 46) && flag == 0) {
        if (access ("baseclar.bms", 0) == 0) {
            if (system ("copy baseclar.bms temp.bms > NULL") == -1) EXIT_1a (nome);
```

```
            if (rename ("temp.bms", nome) == -1) EXIT_1a (nome);
            if (chmod (nome, S_IWRITE) == -1) EXIT_1a (nome);
            deletar ("temp.bms");
            open_ok (nome);
            if (spawnvp (P_WAIT,"PROCALC.EXE", argv)==-1) nao_procalc (nome);

            opcoes_false (1);                }

        else EXIT_1a ("BASE.BMS");                }


              } /*if (tipo == 2)*/

if (tipo == 3) {

    txtstor2 ('s', 1, 1, 80, 24);
    screen_d (6,14,37,16,11,1,7,1,1);
    gotoxy (7,15);
    cputs ("Qual o arquivo ? ________.BMS");
    getsf (11,1,24,15,8,0.0,0.0,nome,2,2);

    if (strlen (nome) <= 0) {txtstor2 ('r', 1, 1, 80, 24); procalc3 (); new_old3 ();}
    else {txtstor2 ('r', 1, 1, 80, 24);strcat (nome,".BMS");}

    flag = procurar (nome);

    if (flag == 1) {
        txtstor2 ('s', 1, 1, 80, 24);
        screen_d (43,13,77,16,14,4,7,1,4);
        gotoxy (44,14);
        cprintf ("Existe arquivo %s !!!", nome);
        gotoxy (44,15);
        cprintf ("Deseja substituir arquivo ? (S/N)");
        beep ();
        do opcao = getkey_s (2); while (opcao != 31 && opcao != 49);
        txtstor2 ('r', 1, 1, 80, 24);
        if (opcao == 49) opcoes_false (2);
        else {
              opcao = 2;
              flag  = 0;
              deletar (nome);
             }
                   }

    /* verifica se arquivo BASEBRUT.BMS existe, abrindo um novo arquivo */
    if ((opcao == 2 || opcao == 46) && flag == 0) {
        if (access ("basebrut.bms", 0) == 0) {
            if (system ("copy basebrut.bms temp.bms > NULL") == -1) EXIT_1a (nome);
            if (rename ("temp.bms", nome) == -1) EXIT_1a (nome);
            if (chmod (nome, S_IWRITE) == -1) EXIT_1a (nome);
            deletar ("temp.bms");
            open_ok (nome);
            if (spawnvp (P_WAIT,"PROCALC.EXE", argv)==-1) nao_procalc (nome);
```

```
            opcoes_false (1);                }

        else EXIT_1a ("BASE.BMS");                }

              } /* if (tipo == 3) */


}

/* ------------------- */

/* monta a tela  com  o  menu  principal
   (sem opcoes), mais o menu procalc3 ();
   e volta para o new_old3 (); */


void opcoes_false (int tipo)
{

 if (tipo == 1) {
    tela     ();  /* monta a tela de fundo */
    m_menu   ();  /* monta o menu principal*/
    procalc3 ();  /* monta o menu procalc  */
    new_old3 ();  /* funcao para Procalc   */
                }

 if (tipo == 2) {
    procalc3 ();  /* monta o menu procalc  */
    new_old3 ();  /* funcao para Procalc   */
                }

}

/* -------------------- */
/* avisa que arquivo foi gerado e esta chamando Procalc */
void open_ok (char nome [12])
{
 screen_d (47,13,77,16,0,3,7,1,3);
 gotoxy (48,14);
 cprintf ("    Novo arquivo aberto !    ");
 gotoxy (48,15);
 cprintf ("Chamando Procalc %s", nome);
}
/* -------------------- */
/* avisa que um programa nao pode ser executado */
void nao_procalc (char nome [12])
{
 screen_d (47,13,77,16,14,4,7,1,4);
 gotoxy (48,14);
 cprintf ("NAO  FOI  POSSIVEL   EXECUTAR");
 gotoxy (48,15);
 cprintf ("    Procalc %s", nome);
 beep ();
```

```
 delay (2000);
}
/* -------------------- */
/* termina o programa */
void EXIT_1a (char filename [12])
{
  screen_d (47,19,77,23,14,4,7,1,4);
  gotoxy (48,20);
  cprintf ("ERRO NA ABERTURA DO  ARQUIVO:");
  gotoxy (48,21);
  cprintf ("          %s", filename);
  gotoxy (48,22);
  cprintf ("    PROGRAMA ENCERRADO !!     ");

  beep ();
  beep ();


  window (1,1,80,25);
  gotoxy (1,24);

  exit (1);
}
/* -------------------- */

/* menu para chamar a planilha Procalc Plus */
void procalc3 ()
{

  menu (
        /* quadro */
        3,
        6,
        2,
        0,
        3,
        3,
        7,

        /* titulo */
        "PLANILHA Procalc Plus",
        1,
        3,
        "c",
        1,

        /* texto */
        7,
        0,
        3,
        "e",
        "- Escolha letra, numero ou Esc -",
        "",
```

```
            "1    - Editar arquivo existente",
            "2    - Criar arquivo ESTATISTICA",
            "3    - Criar arquivo BALANCO",
            "Esc  - Retorna ao menu principal",
            "",
            "",
            "",
            "",
            "",
            "",
            "",
            "",
            0,

            /* high-light */
            78,
            0,
            0,
            0,
            7,

            23,
            21,
            7,
            0,
            0,
            0,
            0,
            0,
            0,
            0);
}/*
   ROOTCOM.C FUNCAO PARA LISTAR ARQUIVOS
   DO DIRETORIO ATUAL, BASEADA EM SHOW.C
*/

#include <dir.h>
#include <dos.h>
#include <stdio.h>

#define OK              0
#define ENCONTRADO      1
#define N_ENCONTRADO    0

procurar (char nome [12])
{

 FILE *fp;
 struct ffblk store;
 int flag_file=0;

 /* procura o primeiro arquivo */
 if (findfirst (nome, &store, 0xFF) == -1) return N_ENCONTRADO;
 else ++flag_file;
```

```
 /*
   continua procurando outros arquivos
   (em caso de *.* ou similares)
 */
 while (findnext (&store) == OK) {++flag_file;}

 if (flag_file > 0) return ENCONTRADO;
 else return N_ENCONTRADO;

}

#include <conio.h>

produto ()
{

 textattr (YELLOW + (BLUE<<4));
 gotoxy (34,2);
 cputs ("BALMAS     1.00");
}

/*
  utiliza a interrupcao 10h da BIOS para
  verificar o caracter e  o  atributo em
  (col,row) (funcao 08h) e reescrevendo-
  o com o atributo  modificado  (attrib)
  utilizando a funcao 09h
*/

#include <dos.h>
#include <conio.h>

void rewrite (int col, int row, char attrib)

{

 union REGS reg;
 struct text_info text;

 gettextinfo (&text);

     gotoxy (col,row);
     reg.h.ah = 8;                  /* pega o caracter  */
     reg.h.bh = 0;                  /* pagina 0         */
     int86 (0x10, &reg, &reg);      /* via ROM BIOS     */
     reg.h.ah = 9;                  /* reescreve        */
     reg.h.bl = attrib;             /* hilite atributos */
     reg.h.bh = 0;                  /* pagina 0         */
     reg.x.cx = 1;                  /* um caracter      */
     int86 (0x10, &reg, &reg);

 gotoxy (text.curx, text.cury);     /* restore cursor   */

}/* DESENHA JANELAS PARA MENUS */
```

```
/* Descricao das variaveis
   xs = x superior esquerdo
   ys = y     "        "
   xi = x inferior direito
   yi = y     "        "
   corfore   = foreground color (p/modulra)
   corfore_2 =      "       "   (p/sombra) (if==-1 sem sombra)
   corback   = background   "   (p/moldura)
   tipo = 1 (linha simples) ou 2 (linha dupla)
   cor_interna = cor dentro da janela */

#include <conio.h> /* para cores                          */
#include <string.h>/* para concatenar strings            */

void screen_d (xs,ys,xi,yi,corfore,corback,corfore_2,tipo, cor_interna)

{
int tipo_vert, tipo_c1, tipo_c2, tipo_c3, tipo_c4;
register int horiz, vert, cont;
char line [80] = "";

/*
   efetua os testes antes de comecar a montagem  da tela
   pois os if's sao mais demorados e, se efetuados entre
   etapas de construcao da tela, podem retardar o efeito
   pop-up.
*/

if (tipo == 1) {
                tipo_vert = 179;
                tipo_c1 = 218;
                tipo_c2 = 191;
                tipo_c3 = 192;
                tipo_c4 = 217;}

if (tipo == 2) {
                tipo_vert = 186;
                tipo_c1 = 201;
                tipo_c2 = 187;
                tipo_c3 = 200;
                tipo_c4 = 188;}

if (tipo == 1) {
                horiz = xs+1;
                while (horiz <= xi-1) {
                       strncat (line,    "D", 1);
                       horiz++;
                                      }

              }

else {
      horiz = xs+1;
```

```
      while (horiz <= xi-1) {
             strncat (line,    "M", 1);
             horiz++;
                           }

      }

/* monta a area interna da janela */

if (cor_interna >= 0) {

    window (xs+1,ys+1,xi-1,yi-1);
    textbackground (cor_interna);
    clrscr ();
    window (1,1,80,25);

                      }


/* monta a sombra da janela */

if (corfore_2 >= 0) {

   for (cont = ys+1; cont <= yi+1; cont++) {

        rewrite (xi+1, cont, corfore_2);
        rewrite (xi+2, cont, corfore_2);

                                          }

   for (cont = xs+2; cont <= xi+1; cont++) rewrite (cont, yi+1, corfore_2);

                    } /* fim do if */

/* imprime linhas horizontais */

textcolor (corfore);
textbackground (corback);

gotoxy (xs+1,ys);
cprintf ("%s", line);

gotoxy (xs+1,yi);
cprintf ("%s", line);

/* imprime linhas verticais */

vert = ys+1;
while (vert <= yi-1) {
                       gotoxy (xs,vert);
                       cprintf("%c", tipo_vert);
                       gotoxy (xi,vert);
                       cprintf("%c", tipo_vert);
```

```
                    vert++;
                   }

gotoxy (xs,ys);
cprintf("%c",tipo_c1);
gotoxy (xi,ys);
cprintf("%c",tipo_c2);
gotoxy (xs,yi);
cprintf("%c",tipo_c3);
gotoxy (xi,yi);
cprintf("%c",tipo_c4);

}

/* ---------------------------------- */
/* SHELL PARA O SISTEMA               */
/* VERSAO 2 SALVA A TELA ANTES DO SHELL */
/* VERSAO UTILIZADA NA PROCALC PLUS   */

#include <conio.h>
#include <process.h>
#include <alloc.h>



void shell2 ()
{

char command [80]; /* para shell do sistema */
char path [20];
char *tbuffer_shell_2;
int  bufsize_shell_2;

 bufsize_shell_2 = 2 * (26) * (81);

 if((tbuffer_shell_2=(char *)malloc (bufsize_shell_2))==NULL){beep();beep();}

 else {
       if (!gettext (1,1,80,25,tbuffer_shell_2)) {beep (); beep ();}

       else {


             textattr (YELLOW + (BLACK<<4));
             clrscr ();

             textattr (YELLOW + (BLUE<<4));
             gotoxy (1,1);
             cputs ("Digite      <return> para retornar ...
     ");

             /* "exit" e' default para o shell do DOS */

             textattr (YELLOW + (RED<<4));
```

```
                gotoxy (8,1);
                cputs ("EXIT");
                gotoxy (40,1);
                textattr (YELLOW + (BLUE<<4));
                gotoxy (1,3);

                cursor ("on");

                system ("command"); /* mostra qual o sistema operacional e aciona o shell
                                       do DOS */

                cursor ("get_off");
              }

          if (!puttext (1,1,80,25,tbuffer_shell_2)) {beep (); beep ();}

          free (tbuffer_shell_2);

          cursor ("off");
        }

}

/* espera o usuario tecla algo,
   enquanto isso vai atualizan-
   do o relogio */

#include <conio.h>

tecle (int tipo)
{
 screen_d (26,17,55,19,0,3,7,1,3);
 gotoxy (27,18);
 cputs ("TECLE ALGO PARA PROSSEGUIR !");

 if (tipo == 1) getkey_s (1); /* apenas timer     */

 if (tipo == 2) getkey_s (2); /* vga_card & timer */

}

/* MONTA A TELA DE FUNDO */

#include <conio.h>

void tela ()
{

 register int i;


 textattr (BLACK + (BLACK<<4));
 clrscr ();
```

```
 window (1, 1, 80, 24);
 textattr (CYAN + (LIGHTGRAY<<4));
 clrscr ();
 window (1, 1, 80, 25);


 normvideo ();
 textattr (CYAN + (LIGHTGRAY<<4));
 screen_d (1,1,80,24,1,7,-1,2,7);


 textattr (BLUE + (BLUE<<4));
 for (i=3; i<=29; i++){
                       gotoxy (i,2);
                       cprintf ("%c",255);
                      }

 timer ();

 unidade ();

 produto ();

 textattr (BLACK + (GREEN<<4));



}#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <dos.h>
#include <process.h>
#include <math.h>

/* funcoes */
void opcoes ();
void balmas ();
void estatistica ();
char *def_arquivo_dat (char *arquivo_dat);
char *def_arquivo_txt (char *arquivo_txt);
void ADVICE_1 (char filename [15]);
void ADVICE_2 (char filename [20]);
void EXIT_2 (char filename [15]);
void proc_arq ();


main ()
{
 cursor ("get_off");

 textattr (BLACK + (BLACK<<4));
 clrscr ();
```

```
 abertura ();  /* apresenta o programa  */

 unidade ();

 tecle (1);

 opcoes ();

 }

/* ------------------------ */
/* monta a tela com o menu principal */
void opcoes ()
{
 char opcao;

 textattr (BLACK + (BLACK<<4));
 clrscr ();

 tela ();

 unidade ();

 m_menu ();

 do {


   switch(opcao=getkey_s(2)){
                     case 59: /* F 1 - help */
                     case 35: /* H   - help */
                      help ();
                      tela ();
                      m_menu ();
                     break;

                     case 60: /* F 2 - shell */
                     case 31: /* S   - shell */
                      shell2 ();
                     break;

                     case 61: /* F 3 - planilha */
                     case 25: /* P   - planilha */
                      procalc3 ();
                      new_old3 ();
                      tela ();
                      m_menu ();
                     break;

                     case 62: /* F 4 - balanco */
                     case 48: /* B   - balanco */
                      balmas ();
                      tela ();
```

```
                    m_menu ();
                   break;

                   case 63: /* F 5 - estatistica */
                   case 18: /* G   - estatistica */
                    estatistica ();
                    tela ();
                    m_menu ();
                   break;

                   case 64: /* F 6 - impressao */
                   case 23: /* G   - impressao */
                    imprimir ();
                    tela ();
                    m_menu ();
                   break;

                   case 1:  /* Esc - retorna */
                   case 19: /* R   - retorna */
                    txtstore ('s', 21, 11, 58, 15);

                    screen_d (21,11,56,13,14,4,7,1,4);
                    gotoxy (22,12);
                    cputs ("Deseja terminar o programa ? (S/N)");

                    if ((opcao = getkey_s (2)) == 31) {
                        txtstore ('r', 21, 11, 58, 15);
                        textattr (WHITE + (BLACK<<4));
                        clrscr ();

                        gotoxy (1,1);
                        cputs ("Ate breve !");
                        gotoxy (1,3);
                        cursor ("on");
                        exit (0); }

                    else txtstore ('r', 21, 11, 58, 15);

                   break;

                  } /* fim do switch */

   } while (opcao);

} /* fim do main */

/* ---------------------- */

/* recebe nome de arquivo de entrada */
char *def_arquivo_dat (char *arquivo_dat)
{
 strcpy (arquivo_dat, 0);
```

```
 screen_d (3,15,34,17,11,1,7,1,1);

 do {
     gotoxy (4,16);
     cputs ("Arquivo de dados:         .DAT");
     getsf (11,1,22,16,8,0.0,0.0,arquivo_dat,2,2);
     if (strcmpi (arquivo_dat,"NULL") == 0) {beep (); beep ();}
    } while (strcmpi (arquivo_dat,"NULL") == 0);


 if (strlen(arquivo_dat) < 1) opcoes ();

 else {
       strcat (arquivo_dat, ".DAT");
       if (procurar (arquivo_dat) != 1) {
           ADVICE_1 (arquivo_dat);
           strcpy (arquivo_dat,NULL);
           opcoes ();                    }

      }

 return (arquivo_dat);

}
/* ---------------------- */
/* recebe nome de arquivo de saida */
char *def_arquivo_txt (char *arquivo_txt)
{
 strcpy (arquivo_txt, 0);

 screen_d (3,20,35,22,11,1,7,1,1);

 do {
     gotoxy (4,21);
     cputs ("Arquivo relatorio:         .TXT");
     getsf (11,1,23,21,8,0.0,0.0,arquivo_txt,2,2);
     if (strcmpi (arquivo_txt,"NULL") == 0) {beep (); beep ();}
    } while (strcmpi (arquivo_txt,"NULL") == 0);
 if (strlen(arquivo_txt) < 1) opcoes ();

 else {
       strcat (arquivo_txt, ".TXT");

       if (procurar (arquivo_txt) == 1) ADVICE_2 (arquivo_txt);

      }

 return (arquivo_txt);

}
/* ---------------------- */

/* arquivo nao encontrado */
```

```
void ADVICE_1 (char filename [15])
{
  screen_d (47,19,77,23,14,4,7,1,4);
  gotoxy (48,20);
  cprintf ("NAO FOI ENCONTRADO O ARQUIVO:");
  gotoxy (48,21);
  cprintf ("         %s", filename);
  gotoxy (48,22);
  cprintf ("  VERIFIQUE NOME CORRETO !!  ");

  beep ();
  beep ();

  delay (2000);

}
/* ---------------------- */
/* arquivo encontrado */
void ADVICE_2 (char filename [20])
{
 char aux;

 txtstore ('s', 44, 11, 80, 24);


  do {

      screen_d (47,11,77,14,14,4,7,1,4);
      gotoxy (48,12);
      cprintf ("ATENCAO  ENCONTRADO   ARQUIVO");
      gotoxy (48,13);
      cprintf ("    COM NOME:  %s",filename);

      screen_d (44,16,77,19,14,4,7,1,4);

      gotoxy (45,17);
      cputs ("   DESEJA PROSSEGUIR (S/N)  ?   ");
      gotoxy (45,18);
      cputs ("S - O ARQUIVO SERA SUBSTITUIDO !");

      beep ();
      beep ();
      do {timer ();}while (!kbhit ());
      aux = tolower (getch ());
     } while (aux != 's' && aux != 'n');

  if (aux == 'n') {txtstore ('r', 44, 11, 80, 24);opcoes ();}

  else txtstore ('r', 44, 11, 80, 24);

}
```

```
/* ---------------------- */

/* termina o programa */
void EXIT_2 (char filename [15])
{
  screen_d (47,19,77,23,14,4,7,1,4);
  gotoxy (48,20);
  cprintf ("ERRO NA ABERTURA  DO  ARQUIVO");
  gotoxy (48,21);
  cprintf ("         %s", filename);
  gotoxy (48,22);
  cprintf ("    PROGRAMA ENCERRADO !!    ");

  beep ();
  beep ();

  window (1,1,80,25);
  gotoxy (1,24);

  exit (1);
}
/* ---------------------- */
/* avisa que um modulo nao pode ser executado */
void nao_exe_3 (char nome [20])
{
 screen_d (46,13,78,16,14,4,7,1,4);
 gotoxy (47,14);
 cprintf ("NAO  FOI  POSSIVEL  EXECUTAR  A");
 gotoxy (47-(strlen (nome)/2),15);
 gotoxy (47,15);
 cprintf ("      %s",nome);

 beep ();
 delay (2000);

 opcoes ();
}

/* ---------------------- */
/* balanco de massa */
void balmas ()
{
 balmas_1 ();
}
/* ---------------------- */
/* recebe nome e procura arquivo */
void proc_arq ()
{
 char nome_dat [15];
 char nome_txt [15];

 txtstor2 ('s', 1, 10, 40, 24);
```

```
 def_arquivo_dat (nome_dat);

 def_arquivo_txt (nome_txt);

 txtstor2 ('r', 1, 10, 40, 24);

 screen_d (41,5,75,12,14,1,7,1,1);

 textcolor (LIGHTGREEN);
 gotoxy (42,6);
 cprintf ("SISTEMA DE BALANCO DE MASSAS");

 textcolor (YELLOW);

 gotoxy (42,8);
 cprintf ("Arquivo Dados     : %s", nome_dat);

 gotoxy (42,9);
 cprintf ("Arquivo Relatorio : %s", nome_txt);

 textcolor (LIGHTRED);
 gotoxy (42,11);
 cprintf ("N.B.: O nome acima e uma sugestao");

 screen_d (44,17,71,19,0,3,7,1,3);
 gotoxy (45,18);
 cputs ("TECLE ALGO PARA RETORNAR !");

 getkey_s (1);

}
/* ---------------------- */

/* calcula a media e o desvio padrao */

void estatistica ()
{

}

/* ---------------------- */
#include <conio.h>   /* para cores em formato texto */
#include <time.h>    /* para funcao timer */
#include <dos.h>

timer()
{
/* DATA & HORA (UNIX) */
struct date date; /* recebe a data  */
struct time time; /* recebe a hora  */
time_t tunix;     /* para data/hora */

int i;
```

```
static int conta_tempo = 0;

conta_tempo++;

if (conta_tempo == 200) {conta_tempo = 0; wind ();} /* verifica pskey */

getdate (&date);
gettime (&time);
tunix = dostounix (&date, &time);
textattr (YELLOW + (BLUE<<4));
gotoxy (54,2);
cprintf ("%s\n", ctime (&tunix));
textattr (BLUE + (BLUE<<4));
gotoxy (54,2);

}

#include <conio.h>

titulo ()
{

 screen_d (19,9,61,12,9,7,8,2,7);

 textattr (BLUE + (LIGHTGRAY<<4));
 gotoxy (21,10);
 cputs ("        MODULO PARA CALCULO DE         ");

 gotoxy (21,11);

 cputs ("    BALANCO DE MASSA E METALURGICO     ");

}#include <alloc.h> /* para alocar memoria */
#include <conio.h> /* para salvar tela    */

#define OK   1
#define N_OK 0

int salvar_tela_txtstore_2 (int xs, int ys, int xi, int yi);
int recup_tela_txtstore_2 (int xs, int ys, int xi, int yi);


/* variaveis globais */

static int    int_screen_buffer_txtstore_2;  /* para salvar/recuperar tela */
static char *char_screen_buffer_txtstore_2;  /* para salvar/recuperar tela */


int txtstor2 (char tipo, int xs, int ys, int xi, int yi)
{
 if (tipo == 's') return salvar_tela_txtstore_2 (xs, ys, xi, yi);

 if (tipo == 'r') {
```

```
     recup_tela_txtstore_2  (xs, ys, xi, yi);
     return OK;
                }

 else return N_OK;

}

/* ---------------------- */
/* salva tela (modo texto) na memoria */
int salvar_tela_txtstore_2 (int xs, int ys, int xi, int yi)
{
int_screen_buffer_txtstore_2 = 2 * (xi-xs+1) * (yi-ys+1);

if ((char_screen_buffer_txtstore_2 = (char *) malloc(int_screen_buffer_txtstore_2))
                                        != NULL) {
     (gettext (xs,ys,xi,yi, char_screen_buffer_txtstore_2));
     return (OK);    }

else return (N_OK);

}

/* ---------------------- */
/* recupera tela (modo texto) da memoria */
int recup_tela_txtstore_2 (int xs, int ys, int xi, int yi)
{
 int retorno;

 retorno = puttext (xs,ys,xi,yi, char_screen_buffer_txtstore_2);
 free (char_screen_buffer_txtstore_2);

 return retorno;
}

/* ---------------------- */#include <alloc.h> /* para alocar memoria */
#include <conio.h> /* para salvar tela    */

#define OK   1
#define N_OK 0

int salvar_tela_txtstore  (int xs, int ys, int xi, int yi);
int recup_tela_txtstore  (int xs, int ys, int xi, int yi);

/* variaveis globais */

static int    int_screen_buffer_txtstore;  /* para salvar/recuperar tela */
static char *char_screen_buffer_txtstore;  /* para salvar/recuperar tela */
int txtstore (char tipo, int xs, int ys, int xi, int yi)
{
 if (tipo == 's') return salvar_tela_txtstore (xs, ys, xi, yi);

 if (tipo == 'r') {
```

```
     if (recup_tela_txtstore  (xs, ys, xi, yi) == OK);
     return OK;
                 }

 else return N_OK;

}

/* -------------------- */
/* salva tela (modo texto) na memoria */
int salvar_tela_txtstore (int xs, int ys, int xi, int yi)
{
int_screen_buffer_txtstore = 2 * (xi-xs+1) * (yi-ys+1);

if ((char_screen_buffer_txtstore = (char *) malloc(int_screen_buffer_txtstore))
                                    != NULL) {
     (gettext (xs,ys,xi,yi, char_screen_buffer_txtstore));
     return (OK);    }

else return (N_OK);

}

/* ---------------------- */
/* recupera tela (modo texto) da memoria */
int recup_tela_txtstore (int xs, int ys, int xi, int yi)
{
 int retorno;

 retorno = puttext (xs,ys,xi,yi, char_screen_buffer_txtstore);
 free (char_screen_buffer_txtstore);

 return retorno;
}

/* ---------------------- */
```