This is a repository copy of *For a Simulation of the Steering of Solid-Based Mining Structures*.

White Rose Research Online URL for this paper:
http://eprints.whiterose.ac.uk/77029/

---

**Monograph:**

---

**Reuse**
Unless indicated otherwise, fulltext items are protected by copyright with all rights reserved. The copyright exception in section 29 of the Copyright, Designs and Patents Act 1988 allows the making of a single copy solely for the purpose of non-commercial research or private study within the limits of fair dealing. The publisher or other rights-holder may allow further reproduction and re-use of this version - refer to the White Rose Research Online record for this item. Where records identify the publisher as the copyright holder, users can verify any specific terms of use on the publisher's website.

**Takedown**
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.
PROGRAM DESCRIPTION;

For a simulation of the steering of solid-based mining structures

J. B. Edwards
&M. Mazandarani

February 1986

Research Report No. 291.

Department of Control Engineering
University of Sheffield
Mappin Street
Sheffield S1 3JD.
**PROGRAM DESCRIPTION:**

for a Simulation of the Steering of Solid-Based Mining Structures

J. B. Edwards and M. Mazandarani

<table>
<thead>
<tr>
<th>CONTENTS</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Program Capabilities</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Document Layout</td>
<td>3</td>
</tr>
<tr>
<td>2 Steering Problem Formulation</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Machine Equations</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Location of Floor Break-points</td>
<td>5</td>
</tr>
<tr>
<td>2.3 Storage Intervals</td>
<td>6</td>
</tr>
<tr>
<td>2.4 Height constraints at the Breakpoints</td>
<td>7</td>
</tr>
<tr>
<td>2.5 Height constraints at the Base-ends</td>
<td>8</td>
</tr>
<tr>
<td>2.6 Fitting the Base</td>
<td>9</td>
</tr>
<tr>
<td>2.7 Control Laws</td>
<td>11</td>
</tr>
<tr>
<td>3 Specimen Results</td>
<td>12</td>
</tr>
<tr>
<td>4 List of Principle Symbols</td>
<td>13</td>
</tr>
<tr>
<td>5 References</td>
<td>14</td>
</tr>
<tr>
<td>6 Program Listing</td>
<td>15</td>
</tr>
<tr>
<td>7 Illustrations</td>
<td></td>
</tr>
</tbody>
</table>
1. **Introduction**

The effect of varying the geometry of coal-winning machines and machine systems on their vertical steering ability is the subject of increasing speculation, prompted by the advent of ranging-drum shearers and underplated (closed-bottomed) armoured face-conveyors (a.f.c's). Some doubt generally exists as to exactly how solid-based structures, like underplated a.f.c's ride over the steps cut by a vertically ranging drum and the attitudes they take up when resting on the stepped cut-floors produced.

The situation with earlier open-bottomed a.f.c's was much simpler in that it could reasonably be assumed that front-and rear-edges of the structure formed the principal points of contact with the floor and that intermediate high-spots were bridged or planed off by the structure. Rolling underframes for steering fixed-drum shearers were also simpler to analyse even with underplated conveyors since again, points of floor-contact were readily predictable.

With ranging-drum machines and underplated a.f.c's, the situation is complicated by the existence of geometrical factors additional to drum and conveyor widths, such as pick-to-pan distance, incomplete pushover, machine system centre-of-gravity etc., all of which can vary from one installation to another. On automatic systems, the coal-sensor location in line-of-advance is a further design variable.

Similar questions arise also in the case of tunnelling machines and roadheaders. Whilst relative dimensions differ from the power-loader situation, the basic geometrical arrangement still pertains. Namely, we have a solid machine base resting on high spots in a floor cut previously by the cutting-head which is located some distance ahead of the base and the depth of cut (the sumping distance) may not equal the length of the cutting head. Vertical steering is again attempted by raising or lowering the head relative to the projected underside of the base.
Fig. 1. illustrates the general situation, be it a coal-face or tunnelling problem, and identifies the basic problem parameters viz:

\[
\begin{align*}
W_b &= \text{base-length} \\
W_d &= \text{drum - (or cutting-head) length} \\
W_p &= \text{separation between drum and base (pick-to-pan distance)} \\
W_a &= \text{advance distance} \\
W_g &= \text{location of machine's centre of gravity from base-front} \\
W_s &= \text{location of height (coal) sensor, if any}
\end{align*}
\]

1.1 Program Capabilities

A general simulation program has been developed and written by the authors of this document that accept any practical values of \(W_d, W_b, W_p, W_a \ (>0.5 \ W_d)\), \(W_b\) and \(W_s\). It then computes and displays, with each advance of the machine, the height pattern, \(y\), of the cut-floor at each step created by raising or lowering the cutting-head in response to vertical steering action, \(J\). The steering actions may be applied manually from the keyboard by pressing 'raise' or 'lower' keys before each advance or they may be applied automatically, based on height-sensor, roof-follower and/or tilt-transducer measurements. The program also computes and displays the front and rear base heights \(h_1, h_2\) at each advance. The output is displayed graphically on the screen of a monochrome storage visual display unit or alternatively as a list of numerical height values. A list of the control actions, \(J\) applied manually or automatically is also available on request.

The program assumes no breakage of the cut-floor whatsoever and works by storing the cut-floor high-spots until these pass under the base, whereupon the height and tilt of the system is adjusted to minimise its total potential energy, without penetrating the cut-floor.

The program may be used at the face-design stage to investigate whether or not any proposed geometry is steerable (not all arrangements are steerable) or it may be used to obtain a recommended escape procedure (a list of \(J\)'s) for an
existing underground face or machine whose control has been lost temporarily.

The program will also assist in tuning up the height, tilt and integral gains of automatic systems before commissioning and will also check whether the system is indeed controllable (not all arrangements are controllable).

1.2 Document Layout

The purpose of this document is to explain the method of modelling the steering and floor-fitting process. The physics of the problem is described in Section 2 and the appropriate mathematical equations and height constraints developed. These are each numbered to allow cross reference to the comments in the program listing reproduced in Section 6. The comments also attempt to convey a physical appreciation of the operations being carried out in any particular program segment e.g. calculations of height ordinates, comparing heights at overlap points to determine high spots in the cut floor, storing and shifting crucial ordinates backwards by one step at each advance of the machine control they pass under the base etc. It is strongly recommended that Section 2 be read before and again in conjunction with the program listing, particularly if changes are contemplated by a purchaser of the source code.

The actual fitting of the base to the cut floor is effected using a standard DUOPLEX Linear Programming routine not provided here in source-code form (this not being the property of the Department of Control Engineering). Linear programming is a well-known optimisation technique however, and merely minimises a linear objective function of variables (here the base-end heights) within a set of linear constraint inequalities. The linear objective function in this context is the potential energy of the machine and base structure, and the parameters of the linear constraints are related to crucial heights in the cut floor: the derivation of which is explained in Section 2. Readers wishing to know more about Linear Programming are referred to References 1 and 2 in Section 5.
Section 3 provides some specimen results obtained by the authors from running the system with various parameters whilst Section 4 lists and defines the principle symbols used in Section 2. Their FORTRAN equivalents given in the program listing have been chosen to match the algebraic symbols as closely as possible.

2. Steering Problem Formulation

Machine Equations

Fig. 2. shows the basic variables specifying the behaviour of the cutting machine itself and the flat base-structure upon which it is mounted. The machine is shown making cut number \( n \) after making \( n-1 \) advances from left to right having started with front- and rear-base heights = \( h_1(0) \) and \( h_2(0) \) respectively. Small-angle geometry is assumed throughout this analysis so that the horizontal advance-distance can be taken as \( W_a \). Each advance is assumed to be constant from cut to cut furthermore. As shown, \( y(n) \) denotes the height of the leading edge of the cutting drum's underside and this is related to the base front height \( h_1(n) \), tilt \( \alpha(n) \) and steering control action \( J(n) \) thus

\[
y(n) = h_1(n) + (W_d + W_p) \alpha(n) + J(n)
\]

where

\[
\alpha(n) = \frac{h_1(n) - h_2(n)}{W_b}
\]

assuming small-angle geometry (as already emphasised). Parameters \( W_d \), \( W_p \) and \( W_b \) denote the drum width, drum-to-base spacing and base-width respectively whilst \( h_2(n) \) denotes the rear height of the base.

Given a value for \( J(n) \) from either an automatic control law or from a manual input therefore, the new cut-floor height \( y(n) \) may be calculated if the base coordinates \( h_1(n) \) and \( h_2(n) \) are also known. These depend on the heights of the high-spots created previously by the drum, presently lying beneath the base and on the position \( W_g \) of the centre of gravity of the machine + base measured back from the leading edge of the base: as shown in Fig. 2. A steering model for the overall system clearly can only be developed if the height and location of all the high spots created in the cut floor can first be determined.
This is the subject of the following Sections of this paper.

Location of Floor Break-Points

The cut-floor clearly comprises a sequence of piecewise-linear segments with upward and downward steps between the breakpoints. As Fig.2. demonstrates, however, the spacing of these steps is not necessarily constant despite the constancy of advance distance, $W_a$. A sequence of upward and downward movements of the drum with respect to its previous position, is shown in Fig.2. such movements being brought about by two causes generated by equations (1) and (2), viz;

(a) deliberate near-vertical adjustment $J(n)$ of the drum with respect to the base structure on which the machine currently rides

and

(b) displacement of the present base position $h_1(n), h_2(n)$ with respect to the previous cut-floor position produced during cut n-1.

A succession of downward movements yield break points of type A in Fig.2, produced by the trailing edge of the cutting head, whilst breakpoints of type B, produced by the leading edge are generated by a sequence of upward movements. Although of the same spatial frequency $W_a^{-1}$, the two breakpoint sequences are obviously phase shifted with respect to each other by drum width, $W_d$, so that when an upward sequence is followed by a sequence of downward movements, two successive high-spots are produced spaced at a distance of only

$$X = 2 W_a - W_d$$  \hspace{1cm} (3)

as typified by the two steps preceeding A in Fig (1) whilst the reverse situation produces two high-spots spaced at

$$Y = W_d$$  \hspace{1cm} (4)

as Fig.2 demonstrates two steps prior to step B.

A simulation model must therefore keep track of the drum position at its front and rear-edge locations taking due account of overlap distance

$$W_o = W_d - W_a$$  \hspace{1cm} (5)
and overlap is now considered in more detail. Before proceeding, however, we should note that, to keep spacing X positive, we shall restrict attention to situations where

\[ W_a > W_d / 2 \]  

i.e. where \[ W_o < W_a \]  

(6)

(7)

Otherwise the number of step positions increases causing greater complexity.

Fig.3. defines four ordinates per drum location: \( y(i) \) and \( y'''(i) \) = the front and rear heights of the drum respectively, produced during cut number \( i \) and intermediate heights \( y'(i) \) and \( y''(i) \) produced during the same cut but which clearly require comparison with overlapping ordinates \( y'''(i+1) \) and \( y(i-1) \) respectively to determine which of each pair represents the true high-spot in the cut floor and which therefore requires storage pending the arrival of the advancing base structure, one or several cuts later. Consideration of Fig.4, however, reveals that the high spot altitudes can only be \( y'(i) \) (not \( y'''(i+1) \)) and \( y''(i+1) \) (not \( y(i) \)) irrespective of whether the drum rises or falls on cut \( i+1 \) with respect to its previous position on cut \( i \). The only ordinates generated on cut \( n \) with the potential to ultimately affect the base position (apart from ordinates beneath its front and rear toes, to be considered later) are therefore:

\[ y'(n) = h_1(n) + \alpha(n) (W + W_a) + J(n) \]  

(8)

\[ y''(n) = h_1(n) + \alpha(n) (W + W_o) + J(n) \]  

(9)

It is now necessary to consider the number of advances between the generation of \( y' \) and \( y'' \) and the commencement and conclusion of their effect on base position i.e. the intervals over which these ordinates should be stored prior to and during their use in predicting the base position.

**Storage Intervals**

In considering the advance of the base from one cut to the next it is important to realise that two fundamentally different situations can occur as regards the location of the base front with respect to the breakpoints produced by the cutting drum. The system geometry determines which situation arises. As
illustrated in Fig. 5. (a), the front of the base may straddle both $y'(n)$ and $y''(n)$ after making a definite number, $p$, advances from the creation of $y'(n)$ and $y''(n)$, or, as shown in Fig. 5 (b), only the trailing breakpoint ordinate $y''(n)$ may be straddled first after, say, $r$ advances, and $y'(n)$ not straddled until $r+1$ advances have occurred. Defining $p$ as the number of advances needed for the base to first encounter $y'(n)$ (after its creation) and $r$ as the number for $y''(n)$ to be first encountered, it is clear that, if $r=p$, the base front always lies in the overlap region between the front and rear of the drum on successive cuts whilst $p=r+1$ corresponds to the base-front always falling in the nonoverlap region. It is readily deduced from Figs. 5(a) and 5(b) respectively that integers $p$ and $r$ are given by

$$\frac{W_p}{W_a} + 2 > p > \frac{W_p}{W_a} + 1$$  
and  
$$\frac{(W_p + W_r)}{W_a} > r > \frac{(W_p + W_r)}{W_a} + 1$$

(10)  

(11)

As Figs. 6(a) and 6(b) illustrate, a similar alternative pair of situations are possible (again dictated only by the fixed geometry of the system, this time involving base length $W_b$). If integer $s$ is defined as the maximum number of system advances for the base-end to straddle $y''(n)$ (following the creation of $y''(n)$) and $q$ = that maximum number to straddle $y'(n)$ then $s$ and $q$ are given by

$$\frac{(W_p + W_r + W_b)}{W_a} - 2 < s < \frac{(W_p + W_r + W_b)}{W_a} - 1$$  
and  
$$\frac{(W_b + W_r)}{W_a} < q < \frac{(W_b + W_r)}{W_a} + 1$$

(12)  

(13)

Hence, if $q=s$, the rear of the base always lies in overlap region but, if $s=q-1$, it will occupy the nonoverlap region. These two situations are illustrated in Figs. 6(a) and 6(b) respectively. (Figs. 5 and 6 are intended to show horizontal locations of ordinates only and the heights of the ordinates sketched carry no significance. For this reason, no tilts of the structure are shown).

Height constraints at the breakpoints

Thus, during cut number $n$, two sets of breakpoint ordinates lie beneath the base, viz
\[ f'(n,i) = y'(n-p-i+1), \quad 1 \leq i \leq q-p+1 \]  \hfill (14)

and
\[ f''(n,i) = y''(n-r-i+1), \quad 1 \leq i \leq s-r+1 \]  \hfill (15)

and to ensure that the base does not penetrate the breakpoints following the next advance, the following constraints must apply:

\[
h_a(n+1) + \{h_2(n-1)-h_1(n+1)\} \{(p+i-2)W_a - W_p \} / W_b \leq f'(n+1,i), \quad 1 \leq i \leq q - p + 1
\]  \hfill (16)

and

\[
h_a(n+1) + \{h_2(n+1)-h_1(n+1)\} \{(r+i)W_a - W_p - W_d \} / W_b \geq f''(n+1,i)
\]  \hfill (17)

These are readily deduced from the geometry of Figs. 5 and 6 by comparing breakpoint heights with base heights, recalling that tilt

\[ a(n+1) = [h_1(n+1) - h_2(n+1)] / W_b \]  \hfill (18)

Two additional constraints must also apply: namely that the ends of the base must not penetrate the cut surface either. We must therefore establish the horizontal location of the base ends with respect to the sequence \( y'(i) \) and \( y''(i) \) and the possible floor heights at these points in terms of \( y'(i) \) and \( y''(i) \).

Height constraints at the base-ends

As Figs 5 (a) and (b) show the front toe of the base may lie in the overlap or nonoverlap region depending on the fixed geometry of the system. The front toe length \( W_{TF} \) may be defined as the extent to which the base front overlaps the leading breakpoint ordinate \( f' \) or \( f'' \) respectively and from Fig.5 is readily deduced to be:

\[ W_{TF} = (p-1)W_a - W_p, \quad p = r \]  \hfill (19)

or
\[ W_{TF} = rW_a - W_p - W_o, \quad p = r + 1 \]  \hfill (20)

Similarly the rear toe length \( W_{TB} \) may be deduced from Fig.6, and found to be given by

\[ W_{TB} = W_p + W_o - bW_a, \quad s = q \]  \hfill (21)

or
\[ W_{TB} = W_p + W_b - qW_a, \quad s = q - 1 \]  \hfill (22)
Now in the regions of drum overlap, two possibilities exist for the height of the cut floor depending on whether the drum cuts lowest (at the point in question) during its first or second occupation of the overlap region. Obviously the base ends need only clear or contact the lowest of these two possible ordinates to avoid their penetration of the cut floor. In the nonoverlap region only one possibility exists since the drum cuts only once in this region. Careful consideration of the system geometry therefore shows that, if \( f_{df}(n-r) \) is the cut floor height at the location of \( h_1(n-r) \) then
\[
f_{df}(n-r) = \min \left[ y'(n) + \left((r-1)W_a - W_p \right) a(n) \right. \\
\left. \text{and} \ y'(n+1) + \left((r-2)W_a - W_p \right) a(n+1) \right], \quad p=r \tag{23}
\]
or
\[
f_{df}(n-r) = y'(n) + \left((r-1)W_a - W_p \right) a(n), \quad p=r+1 \tag{24}
\]
Similar considerations applied to the tail end of the base show that, if \( f_{dr}(n-s) \) denotes the cut floor height at the location of \( h_2(n-s) \) then
\[
f_{dr}(n-s) = \min \left[ y'(n) + \left((s-1)W_a - W_p - W_b \right) a(n) \right. \\
\left. \text{and} \ y'(n+1) + \left(sW_a - W_p - W_b \right) a(n+1) \right], \quad q=s \tag{25}
\]
or
\[
f_{dr}(n-s) = y'(n-1) + \left(sW_a - W_p - W_b \right) a(n-1), \quad q=s+1 \tag{26}
\]
Thus the two additional constraints on the base height during cut \( n+1 \) are
\[
h_1(n+1) \geq f_{df}(n+1) \tag{27}
\]
and
\[
h_2(n+1) \geq f_{dr}(n+1) \tag{28}
\]

**Fitting the Base**

The base will settle to a position of minimum potential energy on the cut floor beneath it and thus, if \( W_g \) is the distance of the centre-of-gravity of the system, measured back from the leading toe of the base, the potential energy function
\[
E(n+1) = h_1(n+1) + \left( h_2(n+1) - h_1(n+1) \right) W_g / W_b \tag{29}
\]
will be minimised in the fitting process by automatic adjustment of base-end-heights \( h_1(n+1), h_2(n+1) \), subject to the hard constraints imposed by conditions (16), (17), (27) and (28). These constraints clearly total \( q-p+s-r+4 \) in number.
Control Law

Automatic control is conventionally based on feedback measurements from a roof coal thickness-sensor, a tilt transducer (to provide derivative action) and a roof height sensor which measures any difference between the cut roof height and the base-height (both projected to a point beneath the thickness sensor). (The purpose of the roof-height follower is primarily to detect any deviation between the base and the cut floor beneath arising from the presence of fine coal left behind by the cutting drum, upon which the base may climb. Other factors can also cause such a deviation of course, not least the high spots produced earlier by the drum itself, and upon which the base now rests.)

Whilst the drum is making cut number \( n \) (i.e. whilst producing a drum height \( y(n) \) at the face side) the coal thickness \( y_c \) signal used may be one or more passes out of date depending on the sensor location. The control equations are therefore

\[
J(n) = k_h \{ y_{ref} - y_c(n) \} - k_g \{ h_1(n) - h_2(n) \} \\
+ k_r \left[ y(n-1) - h_1(n) - W_p \{ h_1(n) - h_2(n) \} / (W_p + W_b) \right]
\]  \hspace{1cm} (43)

where \( k_h, k_g \) and \( k_r \) are the thickness, tilt and roof-height gain settings, \( y_{ref} \) is the desired drum height and \( y(n-1) \) is obtained from previous base height, tilt and control values using equations (1) and (2) whilst \( y_c(n) = y'(n-cs) \), \( cs=1,2,3 \) etc. 

\[
y_c(n) = y'(n-cs)
\]  \hspace{1cm} (44)

the integer \( cs \) defining the prespecified sensor location shown in Fig.1., i.e.

\[
W_S = (cs-1) W_a
\]  \hspace{1cm} (45)

The reader will find most of the foregoing equations cross-referenced in the program listing of Section 6. A flowchart of the program is given in Fig.7.
3 Specimen Results

Fig. 8 shows the computed response of an automatic system having

\[
\begin{align*}
W_d &= 22 & W_s &= 0 + \\
W_b &= 48 & \text{height gain } k_h &= 0.5 \\
W_p &= 37 & \text{tilt gain } k_g &= 1.0 \\
W_a &= 15 & \text{integral gain } k_i &= 0.0 \\
W_g &= 24 & \text{roof follower gain } k_r &= 0.0
\end{align*}
\]

The relatively large ratio: \( W_p/W_d \) is clearly more appropriate to tunnelling machines than to shearers. The system clearly behaves in a stable manner and takes 25 advances to correct an initial height error by 80%.

Fig. 9 shows the response of a coal-face system, again on automatic control, with the parameters:

\[
\begin{align*}
W_d &= 68 & W_s &= 0 + \\
W_b &= 96 & k_h &= 0.5 \\
W_p &= 22 & k_g &= 1.0 \\
W_a &= 40 & k_i &= 0.0 \\
W_g &= 30 & k_r &= 0.0
\end{align*}
\]

(It is the ratios of the dimensions that dictates the performance of the system, of course, and here the \( W_d/W_b \) ratio is larger than used previously whilst the ratios \( W_p/W_d \) and \( W_p/W_b \) are much smaller, though still significant).

In this case the system is unsteerable, both height and tilt going progressively out of control despite the actions of the ranging drum. The trouble appears to be the failure of the base to reach the upward steps created by the upward ranging drum. Increasing \( W_a \) to, say, 60 does not cure the problem.

Fig. 10 shows the behaviour of the system as for Fig. 9 but with pick-to-pan distance \( W_p \) now reduced from 22 to 10. Control is now achieved as can be seen.

Fig. 11 shows the effect of introducing the roof-follower (i.e. setting \( k_r = 1.0 \)) to control the system of Fig. 9. The pick-to-pan distance is left set at 22. The system remains uncontrollable despite the introduction of this additional control device.
No obvious pattern for steerable geometry has yet emerged. In Fig. 8 for instance, a very large $W_p/W_b$ ratio is steerable as is the much smaller value for Fig. 10 whilst the intermediate value for Fig. 9 (and 11) fails. For the moment, therefore, it would appear that each new geometry should be assessed on its own merits. There are no safe rules-of-thumb as yet.

4. List of Principal Symbols

$a(n)$ = tilt of base (in radians) during cut $n$

$b_1, b_2$ = coefficients in Linear programming constraints.

$b_{11}, b_{12}, b_{13}$ = coefficients in Linear programming constraints.

$b_{j1}, b_{j2}, b_{j3}$ = objective function coefficients for Linear programming.

$c_s$ = integer $= W_s/W_a$ defining coal sensor location

$f'(n,i)$ = height of ith breakpoint associated with $y'$ sequence beneath base during cut $n$.

$f''(n,i)$ = height of ith breakpoint associated with $y''$ sequence beneath base during cut $n$.

$f_{df}(n)$ = height of cut floor beneath front of base during cut $n$.

$f_{dr}(n)$ = height of cut floor beneath rear of base during cut $n$.

$h_1(n)$ = height of front of base during cut $n$.

$h_2(n)$ = height of rear of base during cut $n$.

$J(n)$ = drum deflection (jack extension) applied during cut $n$.

$k_g$ = tilt gain of auto control system

$k_h$ = coal thickness gain of auto control system

$k_r$ = roof-follower gain of auto control system

$n$ = cut number

$p$ = integer minimum number of advances for base to straddle $y'(n)$

$q$ = integer number of advances before rear of base leaves $y'(n)$

$r$ = integer minimum number of advances for base to straddle $y''(n)$
s = integer number of advances before rear of base leaves y''(n)

v(p) = storage away for y' sequence before reached by advancing base

w(r) = storage array for y'' sequence before reached by advancing base

W_a = advance distance

W_b = base width

W_d = drum width

W_g = distance between front of base and system's centre of gravity

W_o = overlap distance

W_P = spacing between drum and base front

W_s = distance of coal-sensor from rear edge of drum

W_{TF} = front toe length

W_{TB} = back toe length

x(r) = storage array for cut floor ordinates beneath front of base
      (on arrival)

x_1, x_2 = Variables in Linear programming

y(n) = height of front of drum on cut n

y'(n) = height of floor cut by drum on cut n at leading overlap point

y''(n) = height of floor cut by drum on cut n at trailing overlap point

y''''(n) = height of rear of drum on cut n

y_{ref} = reference drum height for auto control system

z(r) = storage array for cut floor ordinates beneath rear of base
      (on arrival)

z = objective function in Linear programming

5. References

(1) Künzi, H.P., Tzschack, H.G. and Zehnder, C.A. "Numerical methods of

(2) Michell, G.H. "Operational research" English Universities Press,
6. Program Listing
************ WITH MANUAL OPTION ************

This listing is best understood by reference to University of Sheffield Control Eng. Research Report No. 291 Feb 1986

Equation numbers stated on this listing are those in the report

The Program uses a DUOPEX Linear Programming (L. P.) routine for fitting the base to the stored Cut Floor high spots. The SOURCE routine is NOT included, and therefore not documented here.

The variables Xn are the integer representations of variables Wn (n = A, B, D, G, etc)

List of Variable Names:
************************

AGN is Stores user reply to RUN program again
ANS is Stores user reply to a prompt
CNST is Constant added to heights to avoid negative heights (for L. P. purposes)
CODE is Code for Manual option to indicate whether or not JD has entered in a pass
CS is Coal Sensor position
FD1D is Final value of drum-rear height
FD2D is Final value of drum-front height
FDF is Cut floor height beneath front edge of base
FDR is Cut floor height beneath rear edge of base
JD is Drum deflection
H1 is Optimized height of front-edge of the base
H2 is Optimized height of rear-edge of the base
KG is Tilt gain
KH is Height gain
KR is Roof sensor gain
KYREF is Integer of reference horizone to be reached
M is Total number of the constraints
M1 is Number of constraints of type 1
M2 is Number of constraints of type 2
MPSS is Maximum pass number
NPSS is Current pass number
NSTR is No. of machine pictures saved at each run
OPTN is Auto or Manual switch
P is Number of advances needed for the base-front to reach the current Y1D
Q is Number of advances needed for the base-end to reach the current Y1D
ORY is Stores user reply to change of structure query
R is Number of advances needed for the base-front to reach the current Y2D
RSLT is Stores user reply to save results in a file
S is Number of advances needed for the base-end to reach the current Y2D
SAV is Stores user reply to save/quit/final question
SXIB is Integer of saved base X-coordinate
SXID is Integer of saved drum X-coordinate
SZ is Size of constraint tableau used by L. P.
XA is Advance distance
XB is Base length
XCL is X-coordinate of front edge of the base
XCR is X-coordinate of front edge of the drum
XD is Drum length
XG is Position of C. of G. from base front edge
X0 is Overlap distance
XOFST is Offset to be added to all X-coordinates
XS is X-coordinate of base
XSD is X-coordinate of drum
Y0D is Height of front-edge of drum (for drawing)
Y1D is Height of rear of the drum
Y2D is Height of front of the drum
Y3D is Height of rear-edge of drum (for drawing)
YBF is Potential Heights beneath front edge of base
YBR is Potential Heights beneath rear edge of base
YG is Height of centre of gravity
YOFST is Offset to be added to all Y-coordinates
YREF is reference horizon to be reached

List of Array Name:-
**************************************************************************
A is Constraint tableau in the form which L. P. takes
B is Tableau of all constraints
F1D is Storage for drum rear heights
F2D is Storage for drum front heights
L1 is Internal array of the L. P.
L2 is Internal array of the L. P.
L3 is Internal array of the L. P.
LP1 is Internal array of the L. P.
LP2 is Internal array of the L. P.
MOD is Mode for different cases of cut floor
SFD1D is Storage for FD1D
SFD2D is Storage for FD2D
SPSS is Storage for pass numbers of saved machines
SY is Storage for heights of saved machines
V is Storage for Y1D
W is Storage for Y2D
WE is Storage for overlap distances (for drawing)
X is Storage for YBF
XX is Optimized heights returned from L. P.
Y is Storage for height beneath drum (for drawing)
YC is Storage for coal sensor signal
Z is Storage for YBR

PROGRAM SBS
BYTE YES,NO
BYTE AGN,OPTN,CODE,AUTO,MANU,SAV,RSLT,ORY,ANS,FINL,QUIT
REAL V(10),W(10),X(10),Z(10),F1D(10),F2D(10),Y(10),JD
REAL KH,KG,KR,A(300),B(100,3),XX(3),SY(4,200)
REAL SFD1D(200), SFD2D(200), Y(5, 200), H2(200)

INTEGER LP1(3), LP2(100), L1(3), L2(100), L3(100), YOFS1, XOFS1
INTEGER XD, XB, XA, XO, XU, XSD, SXO, SXD, XG, YO
INTEGER MOD(200), P, Q, R, S, CNST, SZ, CS, SPSS(200)

COMMON/AREA 1/V, W, X, Z, F1D, F2D, FD1D, FD2D, Y, YOD, Y3D
COMMON/AREA 2/H1, H2, WD, WB, WP, WA, WD, HU
COMMON/AREA 3/BETA, Y1D, Y2D, JD, WE, MOD, P, Q, R, S, NPSS, FDF, FDR
COMMON/AREA 4/YC, SFD1D, SFD2D, W, CS, YREF, KH, KG, KR, MPSS
COMMON/AREA 5/AGN, RSLT, OPTN, YES, AUTO, MANU
COMMON/AREA 6/YOFS1, XOFS1

DATA AUTO, MANU, FINL, QUIT /* 'A', 'M', 'F', 'Q' */
DATA YES, NO /* 'Y', 'N' */

CALL TKINIT
AGN=NO
YOFS1=400
XOFS1=50
CNST=100
NPSS=0
NSR=0
JD=0.0
SAV=NO
CALL ERASE
CALL HEADER

C C C INITLZ is parameter entry and parameter calculation routine
C C (Listed below)
C C
C CALL INITLZ

C C FIND VALUES OF P, Q, R & S equations 10 to 13
C C and convert to INTEGER

C P=0
180 IF((P-((WP+WA)/WA)).GT.0) GOTO 185
   P=P+1
   GOTO 180

185 R=0
190 IF((R-((WP+WO)/WA)).GT.0) GOTO 195
   R=R+1
   GOTO 190

195 Q=0
200 IF((Q-((WP+WB)/WA)).GT.0) GOTO 205
   Q=Q+1
   GOTO 200

205 S=0
210 IF((S-((WP+WB+WD-2*WA)/WA)).GT.0) GOTO 215
   S=S+1
   GOTO 210

215 NQ=Q+1
   NS=S+1

C XCR=XOFS1+WB+WP+WD

C C FIND VALUES OF base front and rear Toe Length WF, WTR
C C equation 19 to 22

C WT = F * WA - WP - WO
   IF (P .EQ. R) WT = (P-1) * WA - WP
   WT = NO + WP + WB - Q * WA
   IF (S .EQ. Q) WTB = WP + WB + WO - S * WA

C C Initialize base front and rear heights and stored intermediate
heights
FDF=X(R)
FDR=Z(Q)
SFD1D(NQ)=V(P-1)
SFD2D(NS)=W(R-1)

Scale vertical display heights
FDMX=SFD1D(NQ)
N=2

For successive passes the program repeats from statement 220

220 IF(SFD1D(NQ).GE.FDMX) FDMX=SFD1D(NQ)
IF(SFD2D(NS).GE.FDMX) FDMX=SFD2D(NS)
IF(H1.GT.FDMX) FDMX=H1
IF(H2.GT.FDMX) FDMX=H2

Find Y and X scaling factors
YSCF=150.0/FDMX ! 150 max value of heights
XSCF=950.0/XCR ! 950 max value of advance

SCALE ALL THE M/C'S GEOMETRY horizontally
XB=INT(XSCF*WB+0.5)
XD=INT(XSCF*WD+0.5)
XA=INT(XSCF*WA+0.5)
XO=INT(XSCF*W0+0.5)
IF((XA+XO).NE.XD) XO=XD-XA
XU=INT(XSCF*WU+0.5)
IF((XO+XU).NE.XA) XU=XA-XO

PREPARE THE CONSTRAINT TABLEAU FOR LINEAR PROGRAM (DUOPLEX)

Find parameters of OBJECTIVE FUNCTION 34 using equations 41,42
I=1
B(I,1)=0.0
COFF=WG/WB
B(I,2)=-(1-COFF)
B(I,3)=-COFF

Find COEFFICIENTS OF CONSTRAINTS EQUATION 30 using equation 37
DO 230 J=(I+1),(I+Q-P+1)
B(J,1)=-(CNST+F1D(J-1))
COFF=((P+J-I-2)*WA-WP)/WB
B(J,2)=1-COFF
B(J,3)=COFF
230

Find COEFFICIENTS OF CONSTRAINTS EQUATION 31 using equation 38
I=I+Q-P+1
DO 240 J=(I+1),(I+S-R+1)
B(J,1)=-(CNST+F2D(J-I))
COFF=((R+J-I)*WA-WP-WD)/WB
B(J,2)=1-COFF
B(J,3)=COFF
240

Find COEFFICIENTS OF CONSTRAINTS EQUATION 32 using equation 39
I=(I+S-R+1)+1
B(I,1)=-(CNST+FDF)
B(1,2)=1.0  
B(1,3)=0.0

Find COEFFICIENTS OF CONSTRAINTS EQUATION 34 using equation 40

I=I+1
B(I,1)=-(CNST+FDR)
B(I,2)=0.0
B(I,3)=1.0
M1=I-1
M2=0
M=M1+M2

AUXILIARY OBJECTIVE FUNCTION (STORAGE SPACE)

I=I+1
DO 250 J=1,3
B(I,J)=0.0

SIZE OF A TABLEAU

SZ=(M+2)*(N+1)

PREPARE A TABLEAU FROM B ARRAY
IZSCHR = (N+1) & ISSCHR = 1 MEANS THAT A TABLEAU IS TAKEN ROW BY ROW FROM B ARRAY

DO 260 I=1,(M+2)
DO 260 J=1,(N+1)
K=(I-1)*(N+1)+J
A(K)=B(I,J)
CONTINUE

Instructions down to 350 are for Screen communication and plotting (Bypasses on auto and final pass demanded)

XCL=XCR-WD-WP
IF(OPTN.EQ.AUTO.AND.SAV.EQ.FINL.AND.NPSS.LT.MPSS) GOTO 350
CALL ERASE
CALL HEADER

CALCULATE X-COORDINATES OF BEAM, DRUM AND COORDINATE OF CENTER OF GRAVITY

XSB=INT((XSCF*XCL+0.5)+XOFST
XSD=INT((XSCF*XCR+0.5)+XOFST
XG=XSB-INT((XSCF*(W/G*COS(BETA))+0.5)
YG=INT((YSCF*H1+0.5)+YOFST-INT((YSCF*(W/G*(SIN(BETA)/COS(BETA))))

Plot floor heights
CALL HGHTS(0,XCR,SFD1D,NQ,XA,XO,XSCF,YSCF)
CALL HGHTS(1,XCR,SFD2D,NS,XA,XA,XSCF,YSCF)

Plot floor profile
CALL FLOOR(MOD,Y,(NPSS+1),XCR,XD,XO,XA,XU,WE,XSCF,YSCF)

Draw shape of base and drum
CALL SHAPE(1,XSB,H1,XB,H2,10,XSCF,YSCF)
CALL SHAPE(0,XSD,Y0D,XD,Y3D,25,XSCF,YSCF)
KYREF=INT((YSCF*KYREF+0.5)+YOFST

Plot reference line
CALL TPL0T(0,0,KYREF)
CALL TPL0T(1,1023,KYREF)
! 1023 IS MAX. X-AXIS VALUE

Mark the centre of gravity of the base

CALL MARKER(3,XG,(YG+5))

DRAW THE SAVED M/C AND DRUMS WHICH HAVE SELECTED PREVIOUSLY
IF(NSTR.NE.0.AND..(SAV.EQ.FINL.AND.NPSS.EQ.MPSS)) GOTO 275

OTHERWISE WRITE PASS NO AND INFORMATION

GOTO 282
DO 280 I=1,NSTR
SXK=SXB-(NPSS-SPSS(I))*XA
SXD=XSX-(NPSS-SPSS(I))*XA
CALL SHAPE(1,SXK,SY(1,I),XB,SY(2,I),10,XSCF,YSCF)
CALL SHAPE(0,SXD,SY(3,I),XD,SY(4,I),25,XSCF,YSCF)
280 CONTINUE
CALL TPL0T(0,400,700)
CALL ANMODE
WRITE(6,283)NPSS
283 FORMAT(T54,' PASS NO. = ',I4)
CALL TPL0T(0,800,700)
CALL ANMODE
WRITE(6,290)OPTN
290 FORMAT(T15,' OPTION = ',A1)
CALL TPL0T(0,0,270)
CALL ANMODE
WRITE(6,901)KH,KG,KR,CS
WRITE(6,902)P,Q,R,S
WRITE(6,903)WD,WB,WA,WH
WRITE(6,904)WG,WP,WTF,WTB
WRITE(6,913)NO,NS,JD

ON MANUAL OPTION READ IN THE DRUM DEFORMATION AND
SET CODE = NO (I.E. FALSE) INDICATING THAT VALUE
OF THE DRUM DEFORMATION IS READ IN ON THIS PASS

IF(OPTN.NE.AUTO.AND.CODE.EQ.YES) GOTO 292
GOTO 342
CODE=NO
MANU=YES
CALL TPL0T(0,1,XOFST)
CALL ANMODE
WRITE(6,295)
295 FORMAT(' ENTER DRUM DEFORMATION < JD >')
READ(6,919)JD

Shift all storage arrays to right by one value, to avoid
shifting left twice (on manual picture is drawn twice)

DO 300 J=1,P-1
V(J)=V(J+1)
DO 310 J=1,Q-1
W(J)=W(J+1)
X(J)=X(J+1)
DO 320 J=1,Q-P
Z(J)=Z(J+1)
DO 330 J=1,S-R
F1D(J)=F1D(J+1)
DO 340 J=1,S-R
F2D(J)=F2D(J+1)
GOTO 370
342 IF(NPSS.NE.0) GOTO 346
INITIALLY ASK THE QUESTION OF CHANGE OF STRUCTURE
START FROM TOP OF PROGRAM IF USER WANTS TO CHANGE

CALL TPLLOT(0,1,XOFST)
CALL ANMODE
WRITE(6,344)
FORMAT('ENTER Y TO CHANGE THE STRUCTURE')
READ(6,900)QRY
IF(QRY.EQ.YES) GOTO 100

WRITE ALL THE NECESSARY PARAMETERS AND ARRAY TO FILE
JUST FOR PRINTING AND TESTING PURPOSES
(FILE IS SET TO SCREEN)

IF(RSLT.EQ.YES.AND.NPSS.EQ.0) GOTO 348
GOTO 350

Write parameters at start each run only

WRITE(6,901)KH,KG,KR,CS
WRITE(6,902)P,Q,R,S
WRITE(6,903)WD,HB,W,A,W0
WRITE(6,904)WG,WP,WTF,NTB

RSLT = YES allows results to go to a file
future adaptation by BREITBY

IF(RSLT.NE.YES) GOTO 352
WRITE(6,905)NPSS,JD
WRITE(6,906)(V(I),I=1,10)
WRITE(6,907)(W(I),I=1,10)
WRITE(6,908)(X(I),I=1,10)
WRITE(6,909)(Z(I),I=1,10)
WRITE(6,910)(FID(I),I=1,10)
WRITE(6,911)(F2D(I),I=1,10)
WRITE(6,912)DFD,FDH,H1,H2

CARRY ON IF THE SELECTED MAX PASS NUMBER IS NOT REACHED

IF(NPSS.GE.MPSS) GOTO 450

Fit base to cut floor heights (by Linear programming)
for next pass

CALL DUOPLX(A,SZ,N,M1,M2,(N+1),1,IFAIL,
& LP1,N,LP2,M,L1,N,L2,M,L3,M,XX)
IF(IFAIL.NE.0) GOTO 990
DO 355 J=1,3
355 XX(J) = XX(J) - CNST

User communication for next instructions

IF(SAV.NE.FINL.AND.NPSS.NE.0) GOTO 360
GOTO 364

CALL TPLLOT(0,1,105)
CALL ANMODE
WRITE(6,361)
IF(OPTN.EQ.AUTO) WRITE(6,362)

Request whether to (SAV)E present machine display for
output later

WRITE(6,363)
READ(6,900)SAV

Do NOT accept final picture command on MANUAL option
IF(OPTN .NE. AUTO .AND. SAV.EQ.FINL) SAV=NO
IF(SAV.EQ.QUIT) GOTO 460.

Save current machine heights and pass number if requested

IF(SAV.NE.YES) GOTO 365
  NSTR=NSTR+1
  SPSS(NSTR)=NPSS
  SY(1,NSTR)=H1
  SY(2,NSTR)=H2
  SY(3,NSTR)=Y0D
  SY(4,NSTR)=Y3D

Load output of Linear Program into new base-end heights

H1=XX(1)
H2=XX(2)
NPSS=NPSS+1

Calculate AUTO control if selected using equation 43

IF(OPTN.NE.AUTO) GOTO 366
  JD= KH*(YREF-YC(CS))-KG*(H1-H2)
  & KX*(YC(CS)-H1-((WP*(H1-H2))/(WP+WB)))
GOTO 370

CODE=YES

UPDATE routine contains the floor cutting and shifting
equations (listed below)

CALL UPDATE

On AUTO shift coal sensor signal and load with
new value equation 44

IF(OPTN.EQ.AUTO .AND. CS.GT.1) GOTO 375
GOTO 442
DO 440 I=1,CS-1
440 YC(CS+1-I)=YC(CS-I)

On MANUAL option re-draw the picture with new
entered drum deflection

IF(MANU.NE.YES) GOTO 444
  MANU=NO
  SFD1D(NQ)=Y1D
  SFD2D(NS)=Y2D
  GOTO 220

increment to next pass, store front and rear of the drum
jump back for display and base fitting to high spots just
calculated in the update routine

444 NO=NO+1
  NS=NS+1
  SFD1D(NQ)=Y1D
  SFD2D(NS)=Y2D
  XCR=XCR+WA
GOTO 220

Terminate if requested, otherwise jump to 100 to start again

450 CALL TPLLOT(0,1,30)
        CALL ANMODE
READ(6,900)ANS
CALL ERASE
WRITE(6,462)
462 FORMAT(’ENTER Y TO RUN THE PROGRAM AGAIN’) READ(6,900)AGN
IF(AGN.EQ.YES) GOTO 100
GOTO 999
361 FORMAT(’ENTER Y TO SAVE THE CURRENT M/C’) READ(6,900)AGN
362 FORMAT(’F FOR FINAL PASS (I.E. ENTERED MAX PASS’) READ(6,900)AGN
363 FORMAT(’Q TO QUIT THE CURRENT SIMULATION’) WRITE(6,999)
999 FORMAT(’PASS NO. = ’,I3, ’ JD=’,F6.2)
900 FORMAT(A1)
905 FORMAT(’PASS NO. = ’,I3, ’ JD=’,F6.2)
906 FORMAT(’V ’,10(1X,F6.2))
907 FORMAT(’W ’,10(1X,F6.2))
908 FORMAT(’X ’,10(1X,F6.2))
909 FORMAT(’Z ’,10(1X,F6.2))
910 FORMAT(’FD ’,10(1X,F6.2))
911 FORMAT(’FD ’,10(1X,F6.2))
912 FORMAT(’FDF=’ ,F6.2,’ FDR=’ ,F6.2,’ H1=’ ,F6.2,’ H2=’ ,F6.2)
913 FORMAT(’NQ=’ ,I9,3X,’ NS=’ ,I9,3X,’ JD=’ ,F9.3)
914 FORMAT(F6.2)
915 WRITE(6,991)
916 CALL ANSINIT
917 CALL EXIT
918 END
C
999 C
INITILZ is used for parameter entry and parameter initialization
SUBROUTINE INITILZ
C
BYTE YES,N0
BYTE AGN,OPTN,AUTO,MANU,RPLY,RPT,RSLT
REAL U(10),W(10),X(10),Z(10),F1D(10),F2D(10),JD
REAL KH,KG,KG,KR,Y(5,200),WE(200),YC(10),SFD1D(200)
REAL SFD2D(200)
INTEGER P,Q,R,S,CS
COMMON/AREA 1/V,W,X,Z,F1D,F2D,FD1D,FD2D,Y,Y0D,Y3D
COMMON/AREA 2/H1,H2,WD,WD,WP,WA,WQ,WU
COMMON/AREA 4/YC,SFD1D,SFD2D,WG,CS,YREF,KH,KG,KR,MPSS
COMMON/AREA 5/AGN,RSLT,OPTN,YES,AUTO,MANU
BETA=0.0
C
IF(AGN.NE.YES) GOTO 102
WRITE(6,101)
101 FORMAT(’KEEPING THE EXISTING STRUCTURE? (Y/N)’) READ(6,900)RPLY
IF(RPLY.EQ.YES) GOTO 120
WRITE(6,103)
103 FORMAT(’ENTER Y FOR INITIAL VALUES FROM KEYBOARD’) READ(6,900)RPT
IF(RPT.NE.YES) GOTO 120
WRITE(6,105)
105 FORMAT(’ENTER I.V. FOR V, W, X, Z, F1D, F2D & YC’) DO 110 I=1,10
110 READ(6,919)U(I),W(I),X(I),Z(I),F1D(I),F2D(I),YC(I)
GOTO 142
C
Initialize all storage arrays and machine heights
C
120  DO 130 I=1,10
     V(I)=10.0
     W(I)=10.0
     X(I)=10.0
     Z(I)=10.0
     F1D(I)=10.0
     F2D(I)=10.0
     YC(I)=10.0
     SFD1D(I)=10.0
     SFD2D(I)=10.0
130  DO 140 I=1,4
     Y(I,J)=10.0
140  CONTINUE
     Y0D=10.0
     Y3D=10.0
     FD1D=10.0
     FD2D=10.0
     H1=10.0
     H2=10.0
     IF(RPLY.EQ.YES) GOTO 170
142  RSLT=NO
C142  WRITE(6,143)
C143  FORMAT( 'ENTER Y TO PRINT RESULTS IN A FILE')
C    READ(6,900)RSLT
    WRITE(6,144)
144  FORMAT( 'ENTER DRUM & BASE WIDTH < WD & WB >')
    READ(6,919)WD,WB
    WRITE(6,145)
145  FORMAT( 'ENTER PICK TO PAN DISTANCE < WP >')
    READ(6,919)WP
150  WRITE(6,151)
151  FORMAT( 'ENTER ADVANCE DISTANCE < WA >')
    READ(6,919)WA
    IF(WA.LE.WD) GOTO 155
    WRITE(6,152)
152  FORMAT( 'NOT POSSIBLE, CAN'T ADVANCE MORE THAN WD')
    GOTO 150
C
C    equation 5
C
C
155  WD=WD-WA
160  WRITE(6,161)
161  FORMAT( 'ENTER C OF G OF BASE < WG >')
    READ(6,919)WG
    IF(WG.GE.0.AND.WG.LE.WB) GOTO 165
    WRITE(6,162)
162  FORMAT( 'NOT POSSIBLE, C OF G SHOULD BE ON BASE')
    GOTO 160
165  WRITE(6,166)
166  FORMAT( 'ENTER COAL SENSOR POSITION & YREF')
    READ(6,920)CS
    READ(6,919)YREF
    IF(CS.GT.0) GOTO 170
    WRITE(6,167)
167  FORMAT( 'COAL SENSOR POS. SHOULD>0 SO IT TAKEN AS 1')
    CS=1
170  WRITE(6,171)
171  FORMAT( 'ENTER A FOR AUTO')
    READ(6,900)OPTN
    IF(OPTN.NE.AUTO) GOTO 174
    WRITE(6,172)
172  FORMAT( 'KH & KG')
    READ(6,919)KH,KG
    WRITE(6,173)
FORMAT(' ENTER ROOF SENSOR GAIN < KR >')
READ(6,919)KR
GOTO 175

WRITE(6,176)
FORMAT(' ENTER MAX. PASS(ES) NO.')
READ(6,920)MPSS
WU=WA-WO
RETURN

FORMAT(A1)
FORMAT(F8.2)
FORMAT(I4)
END

UPDATE routine is used for calculating floor cutting heights and shifting equations

SUBROUTINE UPDATE
REAL V(10),W(10),X(10),Z(10),F1D(10),F2D(10),JD
REAL KH,KG,KR,Y(5,200),WE(200)
INTEGER MOD(200),P,Q,R,S
COMMON/AREA 1/V,W,X,Z,F1D,F2D,F1D,F2D,Y,Y0D,Y3D
COMMON/AREA 2/H1,H2,W0,WB,WMP,WA,W0
COMMON/AREA 3/BETA,Y1D,Y2D,JD,WE,MOD,P,Q,R,S,NPSS,DF,DFR

NOTE BETA used rather than ALPHA in the report equations
equation 2

BETA=(H1-H2)/WB

equation 1

Y0D=H1+(W0+W0)*BETA+JD

equation 8,9

Y1D=H1+(W0+W0)*BETA+JD
Y2D=H1+(W0+W0)*BETA+JD

Y3D needed for display only

Y3D=H1+WP*BETA+JD

Save above ordinate in plotting arrays, used in the FLOOR routine

Y(1,(NPSS+1))=Y3D
Y(2,(NPSS+1))=Y2D
Y(3,(NPSS+1))=Y1D
Y(4,(NPSS+1))=Y0D
Y(5,(NPSS+1))=Y1D
WE(NPSS+1)=0.0

Identify whether or not extra floor cut in overlap region and any extra break points for plotting and set the CASE number (MOD) accordingly

MOD(NPSS)=1
IF(Y(3,NPSS).LE.Y(1,(NPSS+1))) MOD(NPSS)=2
IF((Y3D.GT.Y(3,NPSS)).AND.(Y2D.GT.Y(4,NPSS))) MOD(NPSS)=3 !CASE 3
IF((Y3D.GT.Y(3,NPSS)).AND.(Y2D.GT.Y(3,NPSS))) MOD(NPSS)=4 !CASE 4
IF(MOD(NPSS).EQ.3.OR.MOD(NPSS).EQ.4) GOTO 372
GOTO 374

DY1=Y(3,NPSS)-Y(1,(NPSS+1)) ! Y1D - Y3D
DY2=Y(2,(NPSS+1)) - Y(4,NPSS)  \quad \quad Y2D - Y0D
DY3=Y(2,(NPSS+1)) - Y(1,(NPSS+1))  \quad \quad Y2D - Y3D
WE(NPSS+1)=(W0*KDY1)/(DY1+DY2)
Y(5,(NPSS+1))=Y(1,(NPSS+1))+WE(NPSS+1)*DY3/W0

374
IF(WE(NPSS+1).GT.W0) WE(NPSS+1)=W0
IF(WE(NPSS+1).LT.0) WE(NPSS+1)=0

Calculate potential floor heights (in equation 23-26) beneath
front and rear edges of base
YBF1=Y1D+((R-1)*WA-WP)*BETA
YBF2=Y1D+((R-2)*WA-WP)*BETA
YBR1=Y1D+((Q-1)*WA-WP-WB)*BETA
YBR2=Y1D+((Q)*WA-WP-WB)*BETA

Load and shift V-array holding Y1D (i.e. drum rear) ordinates
and hold final value in FD1D for entry into F1D-array
(equation 23, 24)
FD1D=Y1D
IF(P.EQ.1) GOTO 380
FD1D=V(P-1)
DO 380 J=1,P-1
V(P+1-J)=V(P-J)
380 CONTINUE
V(1)=Y1D

Load and shift W-array holding Y2D (i.e. drum front) ordinates
and hold final value in FD2D for entry into F2D-array
(equations 25, 26)
FD2D=Y2D
IF(R.EQ.1) GOTO 390
FD2D=W(R-1)
DO 390 J=1,R-1
W(R+1-J)=W(R-J)
390 CONTINUE
W(1)=Y2D

Find and shift stored heights beneath front and rear of base
FDF=YBF1
IF(R.EQ.1) GOTO 402
IF(P.EQ.R.AND.X(1).GT.YBF2) X(1)=YBF2
DO 400 J=1,R-1
X(R+1-J)=X(R-J)
400 CONTINUE
FDF=X(R)
402 X(1)=YBF1
FDR=YBR1
IF(Q.EQ.1) GOTO 410
FDR=Z(Q)
IF(Q.NE.S.OR.Z(1).GT.YBR1) Z(1)=YBR1
DO 410 J=1,Q-1
Z(Q+1-J)=Z(Q-J)
410 CONTINUE
Z(1)=YBR2

Shift and load F1D and F2D arrays with outputs from Y1D and Y2D
previously stored in FD1D and FD2D equations 14, 15
DO 420 J=1,Q-P
F1D(Q-P+2-J)=F1D(Q-P+1-J)
420 CONTINUE
F1D(1)=FD1D
DO 430 J=1,S-R
SHAPE routine display shape of the base and drum depending on MOD
MOD = 1 displays base
MOD = 0 displays drum

SUBROUTINE SHAPE(MOD, XX1, YY1, XX2, YY2, HGT, XSCF, YSCF)
INTEGER Y1, X2, Y2, XX1, XX2, HT, HGT, YOFST, XOFST
COMMON/AREA 6/YOFST, XOFST
Y1 = INT(YSCF*YY1+0.5)+YOFST
XX=XX1-XX2
Y2 = INT(YSCF*YY2+0.5)+YOFST
HT = INT(HGT+0.5)
CALL TPLOT(0, XX1, Y1)
CALL TPLOT(1, X2, Y2)
CALL TPLOT(1, X2, (Y2+HT))
CALL TPLOT(MOD, XX1, (Y1+HT))
CALL TPLOT(1, XX1, Y1)
RETURN
END

HGHTS routine will display cut floor heights
MOD = 0 solid lines
MOD = 1 dotted lines

SUBROUTINE HGHTS(MOD, X, Y, N, XA, X1, XSCF, YSCF)
REAL Y(N)
INTEGER XA, X1, YOFST, XOFST
COMMON/AREA 6/YOFST, XOFST
IX = INT(XSCF*X+0.5)-X1+XOFST
DO 110 I = N, 1, -1
100 IY = INT(YSCF*Y(I)+0.5)+YOFST
CALL TPLOT(0, IX, YOFST)
110 IF (MOD.NE.1) GOTO 105
J = IY - YOFST
DO 100 K = 1, J, 5
M = 1
IF ((K/2).EQ.(K/2.0)) M = 0
CALL TPLOT(M, IX, (K+XOFST))
105 CALL TPLOT(1, IX, IY)
IX = IX - XA
RETURN
END

FLOOR routine will join the high spots of the cut floors to form the floor profile

SUBROUTINE FLOOR(MOD, Y, N, X, XD, XO, XA, XU, WE, XSCF, YSCF)
INTEGER MOD(N), XO, XA, XE, XD, XU, XX, YY, YOFST, XOFST
REAL Y(5, N), WE(N)
COMMON/AREA 6/YOFST, XOFST
XX = INT(XSCF*XX+0.5)-(N*XA+XO)+XOFST
YY = INT(YSCF*YY(1, 1)+0.5)+YOFST
CALL TPLOT(0, XX, YY) !PUT TO POINT A(1)
XX = XX+XO
YY = INT(YSCF*YY(2, 1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
IF(N.LE.1) GOTO 150
I=2
100 XX=XX+XU
YY=INT(YSCF*Y(3,(I-1))+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
IF(MOD(I-1).EQ.4) GOTO 130
IF(MOD(I-1).EQ.3) GOTO 120
IF(MOD(I-1).EQ.2) GOTO 110
IF(MOD(I-1).NE.1) GOTO 140
YY=INT(YSCF*Y(1,1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
XX=XX+XO
YY=INT(YSCF*Y(2,1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
GOTO 140
C
110 XX=XX+XO
YY=INT(YSCF*Y(4,(I-1))+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
YY=INT(YSCF*Y(2,1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
GOTO 140
C
120 YY=INT(YSCF*Y(1,1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
XE=INT(XSCF*WE(I)+0.5)
XX=XX+XE
YY=INT(YSCF*Y(5,1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
XX=XX+(XO-XE)
YY=INT(YSCF*Y(4,(I-1))+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
YY=INT(YSCF*Y(2,1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
GOTO 140
C
130 XE=INT(XSCF*WE(I)+0.5)
XX=XX+XE
YY=INT(YSCF*Y(5,I)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
XX=XX+(XO-XE)
YY=INT(YSCF*Y(2,1)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
GOTO 140
C
140 I=I+1
IF(I.LE.N) GOTO 100
150 XX=XX+XU
YY=INT(YSCF*Y(3,N)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
XX=XX+XO
YY=INT(YSCF*Y(4,N)+0.5)+YOFST
CALL TPLLOT(1,XX,YY)
RETURN
END

TPLLOT routine will draw or move to specified point on screen
MOD = 0 moves to a point
MOD = 1 draws to a point

SUBROUTINE TPLLOT(MOD,IX,IY)
IF(MOD.EQ.0) CALL MOVBAS (IX,IY)
IF(MOD.EQ.1) CALL DRWBAS (IX,IY)
RETURN
Initialize screen to graphic

SUBROUTINE TKINIT
INTEGER TPS
BYTE ESC,K,AA
DATA ESC,K,AA,TPS /27,'K','A',0/
CALL INITT(480)
CALL CHRSIZ(1)
WRITE(5,100)ESC,ESC
! Erase ANSI Screen
100 FORMAT(1H,'A1','[2J','A1','%!1')
WRITE(5,200)ESC,K,AA,TPS
! Change to TEXTRONIX mode
200 FORMAT(1H,'3A1,I1')
RETURN
END

Initialize screen to alphanumeric

SUBROUTINE ANSINT
BYTE ESC,FF
DATA ESC,FF /27,12/
WRITE(5,100)ESC,FF,ESC
! Clear Graphic Screen
100 FORMAT(1H,'A1','[2J',2A1)
RETURN
END

Clear both mode of the screen

SUBROUTINE ERASE
BYTE ESC,FF
DATA ESC,FF /27,12/
WRITE(5,100)ESC,ESC,FF
! Erase ANSI Screen
100 FORMAT(1H,'A1','[2J',2A1)
RETURN
END

Mark the centre of the gravity of the base

SUBROUTINE MARKER(K,IX,IY)
INTEGER MM,LH,I,K
BYTE ESC
DATA ESC,MM,LH,I /27,'MM','LH',0/
WRITE(5,100)ESC,MM,K
CALL TPLLOT(0,IX,IY)
CALL TPLLOT(1,IX,IY)
WRITE(5,100)ESC,MM,I
100 FORMAT(1H,'A1,A2,I1)
RETURN
END

SHEFFIELD UNIV.
APPLIED SCIENCE
LIBRARY
CALL TPLT(0,0,725)
CALL ANMODE
WRITE(6,200)
CALL TPLT(0,0,725)
CALL ANMODE
WRITE(6,300)
CALL TXTCOL(3)

100 FORMAT(T25,' Solid Base Structure Program')
200 FORMAT(T50,'University of Sheffield')
300 FORMAT(T5,'By J.B. Edwards & M. Mazandarani',//)
RETURN
END

SUBROUTINE TXTCOL(COLOUR)
INTEGER COLOUR,MT
BYTE ESC
DATA ESC,MT /27,'MT'/
WRITE(5,100)ESC,MT,COLOUR
100 FORMAT(1H,A1,A2,I1)
RETURN
END

SUBROUTINE GRFCOL(COLOUR)
INTEGER COLOUR,ML
BYTE ESC
DATA ESC,ML /27,'ML'/
WRITE(5,100)ESC,ML,COLOUR
100 FORMAT(1H,A1,A2,I1)
RETURN
END
7. Illustrations
Fig. 1. Showing geometrical parameters for Machine-steering simulation program.
Fig. 2. Showing machine variables and possible variation of break-point spacing in the cut floor.

Pattern of drum positions

- Cut-floor profile
- Extra floor cut by drum lowering
- Fresh air left by drum lifting
- Breakpoint A
- Downward step cut by drum lowering
- Breakpoint B
- Upward step left by raised drum
- Cutting Head (Drum)
- Base
- \( X = 2W_a - W_d \)
- \( Y = W_d \)
- \( C.o.f.g. \)
- \( W_b \)
- \( W_p \)
- \( W_g \)
- \( \alpha(n) \)
- \( J(n) \)
- \( h_1(n) \)
- \( h_2(n) \)
- \( x/W_a \)
Fig. 4. Showing breakpoint ordinates = $y'$ or $y''$ only.
(never $y$ or $y'''$).
Fig. 5. - Showing the two possible locations for the front of the base.
(a) Rear of base in overlap region (q=s)

(b) Rear of base in non-overlap region (s=q-1)

Fig. 6. Showing the two possible locations for the rear of the base.
Fig. 7. Outline Flowchart (Sheet 1)

(Details of user conversation, machine- and profile-plotting omitted).

Integer: \( p, q, r, s, cs \)

Arrays: \( v(10), w(10), x(10), z(10), f'(10), f''(10) \), (storage arrays for crucial floor heights), \( y_c(10) \).

Yes

Special initial conditions required?

Enter required initial elements in arrays \( v \) to \( f'' \)

No

\[
\begin{align*}
v(i) &= 10, \\
w(i) &= 10, \\
x(i) &= 10, \\
z(i) &= 10, \\
f'(i) &= 10, \\
f''(i) &= 10,
\end{align*}
\]

\( i = 1, 10 \)

\( y_c = 10 \)

Enter desired horizon height \( y_{ref} \) and cod 1 sensor position integer \( cs(=1, 2, 3, \ldots) \)

Enter machine and base parameters \( W_a, W_b, W_c, W_d, W_g \).

\[
W_o = W_d - W_d \quad (\text{eqn.}(5))
\]

\( p = 0, r = 0, q = 0, s = 0 \)

(Find size of height-storage arrays (between drum and base and under base)).

\[
\begin{align*}
\text{If } (p - W/p/a) &> 1 \ ? \quad (\text{eqn.}(10)) \\
p &= p + 1 \\
\text{No} &\quad \text{Yes}
\end{align*}
\]

\[
\begin{align*}
\text{If } (r - W/p/a) &> W_o/W_a \ ? \quad (\text{eqn.}(11)) \\
r &= r + 1 \\
\text{No} &\quad \text{Yes}
\end{align*}
\]

GO TO Sheet 2
If \((q > (W_p + W_b)/W_a)\)? (eqn. 13)

\[ q = q + 1 \]

No

Yes

If \((s > (W_p + W_b + W_d)/W_a - 2)\)? (eqn. 12)

\[ S = S + 1 \]

No

Yes

C. Find front and rear toe lengths

Yes

\[ p = r? \]

No

\[ W_{TF} = (p-1)W_a - W_p \quad \text{(eqn. 19)}, \quad W_{TF} = rW_a - W_p - W_o \quad \text{(eqn. 20)} \]

\[ p = r? \]

\[ W_{TB} = W_p + W_b - sW_a \quad \text{(eqn. 21)}, \quad W_{TB} = W_p + W_b - qW_a \quad \text{(eqn. 22)} \]

Yes

\[ s = q? \]

No

\[ f_{df} = x(r), \quad f_{dr} = z(q) \]

Print \(p, q, r, s, W_a, W_b, W_p, W_d, W_g, W_o, W_{TF}, W_{TB}\)

Draw machine outline

A

No

OK?

Yes

Call Linear Programming routine to minimise

\[ E = h_1 + (h_2 - h_1)W_g/W_b \quad \text{(eqn. 29)} \quad \text{subject to:} \]

\[ h_1 + (h_2 - h_1)(p+i-2)W_a - W_p /W_b > f'(i), \quad \text{(eqn. 16)} \]

\[ i = 1, q-p+1 \]

\[ h_1 + (h_2 - h_1)(r+i)W_a - W_d /W_b > f''(i), \quad \text{(eqn. 17)} \]

\[ i = 1, s-r+1 \]

\[ h_1 \geq f_{df}, \quad \text{(eqn. 27)} \quad h_2 \geq f_{dr}, \quad \text{(eqn. 28)} \]

GO TO Sheet 3
Get base heights $h_1$ and $h_2$ from L.P.

Control mode?

$J = k_h \left( y_{ref} - y_c(cs) \right) - k_g \left( h_1 - h_2 \right) + k_r \left( y_c - h_1 \right) \frac{W_1(h_1-h_2)}{(W_p + W_b)}$  \hspace{1cm} (eqn.43)

$\alpha = \frac{(h_1 - h_2)}{W_b}$  \hspace{1cm} (eqn.2)

$y' = h_1 + \alpha(W_p + W_a) + J$  \hspace{1cm} (eqn.8)

$y'' = h_1 + \alpha(W_p + W_o) + J$  \hspace{1cm} (eqn.9)

Display new floor profile

$y_c(cs+1-i) = y_c(cs-i) - \text{(store coal sensor reading)}$

$i = 1, cs-1$

$y_c(1) = y'$  \hspace{1cm} (eqn.44)

$y_{bf1} = y' + \alpha \left( (r-1)W_a - W_p \right)$  \hspace{1cm} (for eqn. (23) or (24))

$y_{bf2} = y' + \alpha \left( (r-2)W_a - W_p \right)$  \hspace{1cm} (for eqn. (23))

$y_{br1} = y' + \alpha \left( (q-1)W_a - W_p - W_b \right)$  \hspace{1cm} (for eqn. (25) or (26))

$y_{br2} = y' + \alpha \left( qW_a - W_p - W_b \right)$  \hspace{1cm} (for eqn. (25))

C. Store $y'$ in array $v$ for $p$ passes pending base arrival

$p = 1?$

No

$f_d' = v(p-1)$

Yes

$v(p+1-i) = v(p-i)$

$v(1) = y'$

GO TO Sheet 4
C. Store $y''$ in array $w$ for $r$ passes pending base arrival

- $f''_d = w(r-1)$
- $w(r+1-i) = w(r-i)$
- $i = 1, r-1$

- $w(1) = y''$

C. Implement equations 23 or 24 for front toe height via storage array $x$.  

- $p = r?\quad\text{No}$
- $y_{bf2} < x(1)?\quad\text{No}$
- $x(1) = y_{bf2}$

- $x(r-i+1) = x(r-i)$
- $f_d = x(r)$, $x(1) = y_{bf1}$
- $i = 1, r-1$

C. Implement equations 25 or 26 for rear toe height via array $z$.  

- $f_{dr} = z(q)$
- $q = s?\quad\text{No}$
- $y_{br1} < z(1)?\quad\text{No}$
- $z(1) = y_{br1}$

- $z(q+1-i) = z(q-i)$
- $i = 1, q-1$

- $z(1) = y_{br2}$

GO TO Sheet 5
C. Store $y'$ ordinates from array $v$ in array $f'$ beneath base, on arrival.

\[ f'(q-p+2-i) = f'(q-p+1-i) \]

\[ i = 1, q-p \]

\[ f'(1) = f'_d \]

C. Store $y''$ ordinates from array $w$ in array $f''$ beneath base, on arrival.

\[ f''(s-r+2-i) = f''(s-r+1-i) \]

\[ i = 1, s-r \]

\[ f''(1) = f''_d \]

Maximum number of passes complete?

\[ \text{No} \]

Return to L.P. on Sheet 2

\[ \text{Present final display} \]

\[ \text{STOP} \]

\[ \text{Yes} \]
Initial horizon

Desired horizon

Pass 3

Pass 6

Pass 17

Pass 30

Initial horizon

Desired horizon

Fig. 9 Unstable system response
Fig. 10 System response stabilised by reduction of pick-to-pan distance