Skew, Scatter, and Pitch in Music Rolls

Wayne Stahnke

November 12, 1999 (revised November 28, 2023)

Abstract

We present methods for determining the skew, scatter, and pitch of a music roll from a scanned image of the roll using the Method of Least Squares. The procedures are derived from the one given by Gauss for determining longitudes by chronometer. If a roll scan is improved by removing the skew and scatter, it is often possible to reconstruct the punch matrix, and we give a method for doing so.

1 Introduction

Music rolls form an important part of the musical legacy of the first quarter of the twentieth century. These long, perforated rolls of paper controlled the operation of the pneumatic automatic musical instruments of the period, some of which were highly sophisticated and capable of fine performances. Almost all of the great pianists of the Golden Age of the Piano made music roll recordings. We can play these on today's electronic reproducing pianos and synthesizers by scanning the rolls and processing the scans properly.

We are faced with numerous difficulties in bringing these classic recordings to life. Among the first to appear are errors in the scan that derive from three sources: imperfections in the original roll-perforating machinery, errors in the rolls themselves caused by age, damage, and improper storage over the century-plus time span since they were manufactured, and errors in the scanner. In the aggregate, these errors are often significant enough to result in audible degradation of the performance. In this note we give methods for determining the errors, which can then be removed from the scan, yielding an improved data set that eliminates the deleterious effects that would occur otherwise.

Music roll errors are of two types: systematic and random. We can determine the systematic errors by taking advantage of the fact that random errors are just that—random. Thus, if we form averages over many perforations they tend to cancel, whereas systematic errors do not, so the averages reveal the systematic errors. The Method of Least Squares does this averaging in an optimal way, so it is the method of choice. After finding and removing the systematic errors, the random errors can often be removed by exploiting the fact that the perforator advanced the paper in discrete steps, called *rows*. If the position of a perforation along the length of the roll is disturbed by a random error smaller than 1/2 row, its true position can be found by moving it to the nearest row.

Two systematic errors affect music roll scans: *skew*, the tilt of a single row of perforations caused by angular misalignment of the scan line with respect to the centerline of the punch-and-die set in the perforator, and *scatter*, the column-to-column deviations of the perforations from their nominal positions along the length of the roll due to tolerance and wear in the punch-and-die set. The scatter of the leading edges, called *actuate scatter*, is distinct from *release scatter*, the scatter of the trailing edges. In an ideal scan, the total of skew, actuate scatter, and release scatter would be 0. For our purposes, all of these are parameters to be determined so that they can be removed.

There are three further parameters required to characterize a music roll. One is *pitch*, the nominal row advance. Pitch varies slightly from roll to roll, even for rolls made on a single perforator in the same production run. Another is the *common offset*, the displacement of all of the perforations from the start of the scan. The final parameter is *punch length*, the apparent length of a perforation consisting of a single hole, considered separately in Section 7 of this note. In what follows, we denote the parameters other than punch length by these symbols:

$$
\begin{aligned}
c &= \text{common offset (meters)},\\
m &= \text{slope resulting from skew (meters/column)},\\
p &= \text{pitch (meters/row)},\\
s_k &= \text{scatter of column } k \text{ (meters)}.
\end{aligned}
$$

We choose *slope*, the tilt appearing across a single column (rather than skew, the tilt that appears across the entire roll width) for a clearer presentation. Skew is given by $(d-1)m$, where d is the total number of columns in the roll.

By using slope as a parameter, rather than skew, we eliminate the factor $d-1$ that would otherwise appear in many places in the derivations.

The parameters are of different natures. Skew is an error that we wish to find so that we can remove it. Skew often varies a great deal along the length of the roll, which requires considering it to consist of *static skew*, the mean skew over the length of the roll, and *dynamic skew*, the variations from the mean. Special techniques are required to find and remove dynamic skew, as addressed in Section 6.

Scatter is another error that we wish to find and remove. It does not vary along the roll length, so it can be dealt with in a straightforward manner.

Pitch is a fundamental characteristic of the roll. It can vary along the length of the roll, but so slightly that we can consider it to be constant. An accurate pitch estimate is required to find slope and scatter. It is also required for punch matrix reconstruction.

The common offset is an artifact of the scanning process. It is approximately the length of white space appearing after the start of the scan but before the first perforation. Since it results from an arbitrary choice of starting point by the scanner operator, it is not of interest.

2 Two Mathematical Models

The first step in processing a music roll scan is creating a file called a *reduced scan* that specifies the locations of each perforation's leading and trailing edges by an ordered sequence of indications, called *events*. Events are of two types: *actuate events* specify the locations of the leading edges, *release events* the locations of the trailing edges.

There are usually many more events than there are parameters, so we cannot calculate the parameters directly from a reduced scan because the problem is overdetermined. Instead, we find estimates of the parameters by applying the Method of Least Squares using the two mathematical models presented

below. To derive the models, we require several definitions. Let

$i =$ music roll event index,
$d =$ total number of columns in the music roll,
$n =$ total number of events,

where $0 \leq i < n$. There are four variables associated with each event. Let these be denoted by

$q_i =$ position of event i along roll length (rows)
$x_i =$ position of event i across roll width (columns),
$y_i =$ measured displacement of event i along roll length (meters),
$e_i =$ error in the measured displacement y_i (meters),

where $q_i = 0, 1, 2, \ldots$ and $x_i = 0, 1, 2, \ldots, d-1$. The variables q_i and x_i are integers and therefore exact, satisfying the requirement of the Method of Least Squares for the independent variables to be free of uncertainty. The displacement y_i and error e_i are real numbers. With these definitions, the parameters and variables are related by the *displacement equations*

$$\begin{aligned}
y_0 &= c + mx_0 + s_{x_0} + pq_0 + e_0, \\
y_1 &= c + mx_1 + s_{x_1} + pq_1 + e_1, \\
y_2 &= c + mx_2 + s_{x_2} + pq_2 + e_2, \\
&\vdots \\
y_{n-1} &= c + mx_{n-1} + s_{x_{n-1}} + pq_{n-1} + e_{n-1}.
\end{aligned}$$

The ultimate goal of the procedures explained here is to create a modified version of the reduced scan, called an *improved reduced scan*. The improved reduced scan is free of as many errors as possible. A first improvement removes the slope estimate m and scatter estimate s_{x_i} from the equations. If we assume that the estimates are very close to the true values, this yields

$$\begin{aligned}
y_0 &= c + pq_0 + e_0, \\
y_1 &= c + pq_1 + e_1, \\
y_2 &= c + pq_2 + e_2,
\end{aligned}$$

and so forth. In many cases, the substantial improvement resulting from removing skew and scatter can be furthered by punch matrix reconstruction,

which removes the common offset c and the random errors e_i, so that the equations become simply

$$y_0 = pq_0,$$
$$y_1 = pq_1,$$
$$y_2 = pq_2,$$

and so forth. These are the equations of an ideal scan, free of the errors introduced by imperfections in the perforating machinery, the roll itself, and the scanner. Substituting them for the original equations moves us closer to the original performance.

Returning to the original reduced scan, we take up determining the slope m and scatter s_{x_i} so that they can be removed.

Scatter occupies a unique place in the displacement equations because it can be regarded as either a systematic or random error. Gauss discusses errors of this sort in his landmark treatise on the Method of Least Squares, *Theoria Combinationis Observationum Erroribus Minimis Obnoxiae* [4] at the outset, in Article 1. The two viewpoints give rise to the two mathematical models derived in the current section.

When we consider scatter to be constant the scatter terms $s_0, s_1, s_2, \ldots, s_{d-1}$ are parameters to be estimated, but when we attempt to determine them from the displacement equations, we are faced with a difficulty: although the displacements are uniquely determined by the parameters for a given set of independent variables, the parameters are not uniquely determined by the displacements. In particular, the scatter terms are indeterminate for two reasons.

First, the common offset can take on any value by adjusting the scatter terms. To see this, let the common offset be $c = c' + u$, where u is any real number. Substituting gives

$$y_i = c' + mx_i + s'_{x_i} + pq_i + e_i,$$

where $s'_{x_i} = s_{x_i} + u$ is the adjusted scatter term. Since $c' = c - u$, it can assume any value. We resolve this ambiguity by assuming that scatter does not produce a wholesale shift of the perforations along the length of the roll,

which implies that the mean of the scatter terms is 0. Since we do not know the true scatter, we must approximate the assumption by taking the weighted mean of the scatter estimates to be 0. This gives the constraint $\sum W_k s_k = 0$, where s_k is the scatter estimate of column k, W_k is the weight of s_k, and the summation extends from 0 to $d-1$. We accommodate roll columns without perforations by assigning $W_k = 0$ to each such column.

Second, the slope too can take on any value by adjusting the scatter terms. Let the slope be $m = m' + v$, where v is any real number, so that

$$y_i = c + m'x_i + s'_{x_i} + pq_i + e_i,$$

where $s'_{x_i} = s_{x_i} + vx_i$ is the adjusted scatter term Since $m' = m - v$, it can take on any value. We resolve this ambiguity by assuming that scatter does not produce an overall tilt in the perforations, which implies that the scatter terms are uncorrelated with their corresponding column numbers. We do not know the true scatter, so we approximate this assumption by taking the weighted scatter estimates to be uncorrelated with their column numbers, yielding the additional constraint $\sum kW_k s_k = 0$.

Both constraints must be satisfied to resolve the ambiguities mentioned above (regardless of how the parameters are found), but since they complicate the calculations, we prefer to postpone imposing them for as long as possible. To that end, we abandon attempting to solve for slope and scatter at the same time. Instead we subsume the slope into the scatter to eliminate the need for the second constraint, and we consider the scatter of one of the columns to be known in lieu of the first constraint. The result is the *scatter model*. After applying it to find the pitch estimate and the scatter estimates, which will likely include some amount of slope and offset, we can impose the constraints to find the slope and offset, which we then remove. The procedure for doing this is presented in Section 5.5. Thus the scatter model can be applied to find all of slope, scatter, and pitch, but not in a single step.

If the goal is to produce an improved reduced scan for emulation or punch matrix reconstruction we can avoid the final step of finding and removing the slope and offset from the scatter, because removing scatter that contains slope achieves the same end as removing slope and scatter separately from the reduced scan (see Section 5.3). This avoids imposing the constraints.

Whether or not the constraints are ultimately imposed, the scatter model simplifies the calculations.

The scatter model is subject to false solutions if the scan contains a large amount of skew, so before applying it we must find and remove most of the skew. We therefore create a second model that considers scatter to be random, called the *skew model*, to find the skew separately. The two models are described in detail below.

2.1 The Scatter Model

To create the scatter model, we define the *composite scatter* $s'_{x_i} = mx_i + s_{x_i}$ in which the slope is subsumed into the scatter. Substituting into the equations for the event displacements gives

$$\begin{aligned}
y_0 &= c + s'_{x_0} + pq_0 + e_0, \\
y_1 &= c + s'_{x_1} + pq_1 + e_1, \\
y_2 &= c + s'_{x_2} + pq_2 + e_2, \\
&\vdots \\
y_{n-1} &= c + s'_{x_{n-1}} + pq_{n-1} + e_{n-1},
\end{aligned}$$

in which the slope terms mx_i do not appear. In the absence of error, these equations can be rewritten to read

$$\begin{aligned}
y_0 - s'_{x_0} - pq_0 &= c, \\
y_1 - s'_{x_1} - pq_1 &= c, \\
y_2 - s'_{x_2} - pq_2 &= c, \\
&\vdots \\
y_{n-1} - s'_{x_{n-1}} - pq_{n-1} &= c,
\end{aligned}$$

which can be combined to form the multiple equation

$$y_0 - s'_{x_0} - pq_0 = y_1 - s'_{x_1} - pq_1 = y_2 - s'_{x_2} - pq_2 = \dots$$

This is the first equation (with the terms in a slightly different order and the opposite sign for s'_{x_i}) in Gauss's *Chronometrische Längenbestimmungen* [5], in which Gauss applies weighted least squares to determining longitudes by chronometer. We can paraphrase Gauss's statement regarding its solution (see [7], p. 19) and adapt it to the notation used here as follows:

In order for these equations to be sufficient for determining the parameters $s'_{x_0}, s'_{x_1}, s'_{x_2}, \ldots, s'_{x_{n-1}}$ and p we must consider one of the scatter terms to be given, and in addition there must be at least two different events in the same column, so that two or more of the quantities $s'_{x_0}, s'_{x_1}, s'_{x_2}, \ldots, s'_{x_{n-1}}$ are identical. If exactly two are identical the problem is completely determined; otherwise it is overdetermined, and one must determine the unknown quantities such that the $n-1$ equations

$$\begin{aligned}(y_1 - y_0) - (s'_{x_1} - s'_{x_0}) - p(q_1 - q_0) &= 0,\\ (y_2 - y_1) - (s'_{x_2} - s'_{x_1}) - p(q_2 - q_1) &= 0,\\ (y_3 - y_2) - (s'_{x_3} - s'_{x_2}) - p(q_3 - q_2) &= 0,\end{aligned}$$

and so forth, are satisfied as closely as possible.

This brief statement provides the key to determining the parameters using the Method of Least Squares, which cannot be applied to the equations

$$y_i = c + s'_{x_i} + pq_i + e_i$$

without a loss of accuracy because the error terms e_i are serially correlated for reasons given in Section 3, violating the assumption of the Gauss-Markov theorem of independent errors. Gauss overcomes this difficulty by taking first differences to form the difference equations

$$y_j - y_i = (s'_{x_j} - s'_{x_i}) + p(q_j - q_i) + (e_j - e_i),$$

where $0 \le i < n-1$ and $j = i+1$, yielding a set of $n-1$ difference equations in which the error terms $e_j - e_i$ are independent, satisfying the requirement of the Gauss-Markov theorem. The common offset c has also been removed.

To see that taking first differences removes the serial correlation of the errors e_j and e_i from the difference error $e_j - e_i$, let $e_i = e'_i + \epsilon$, where ϵ is the portion of e_i that is carried over to row j, and similarly let $e_j = e'_j + \epsilon$. Then the difference equation error term $e_j - e_i$ is $(e'_j + \epsilon) - (e'_i + \epsilon) = e'_j - e'_i$, which is free of the mutual error ϵ. Since ϵ appears in e_j and e_i with the same sign, those errors suffer from positive serial correlation.

Thus, we can find estimates of the composite scatter and pitch by applying the Method of Least Squares to the residuals

$$r_{ji} = (y_j - y_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i),$$

where $0 \leq i < n-1$ and $j = i+1$, and where the square of each residual is weighted using the weighting given in Section 3.

2.2 The Skew Model

When we regard scatter as random each scatter term s_{x_i} can be subsumed into its corresponding error term e_i, after which the event displacements are given by the equations

$$\begin{aligned}
y_0 &= c + mx_0 + pq_0 + e'_0, \\
y_1 &= c + mx_1 + pq_1 + e'_1, \\
y_2 &= c + mx_2 + pq_2 + e'_2, \\
&\vdots \\
y_{n-1} &= c + mx_{n-1} + pq_{n-1} + e'_{n-1},
\end{aligned}$$

where $e'_i = s_{x_i} + e_i$. The error terms are serially correlated as before, so we take first differences to remove the serial correlation. This yields the $n-1$ difference equations

$$\begin{aligned}
y_1 - y_0 &= m(x_1 - x_0) + p(q_1 - q_0) + (e'_1 - e'_0), \\
y_2 - y_1 &= m(x_2 - x_1) + p(q_2 - q_1) + (e'_2 - e'_1), \\
y_3 - y_2 &= m(x_3 - x_2) + p(q_3 - q_2) + (e'_3 - e'_2),
\end{aligned}$$

and so forth. As with the scatter model, taking first differences removes the common offset c. If there are more than three event pairs (the usual case) there will be more than two difference equations and the problem will be overdetermined, so we solve for the values of m and p that minimize the sum of the squares of the residuals

$$r_{ji} = (y_j - y_i) - m(x_j - x_i) - p(q_j - q_i),$$

where $0 \leq i < n-1$ and $j = i+1$, and where the square of each residual is weighted using the weighting given in Section 3.

The skew model suffers from negative serial correlation, unlike the scatter model. Taking first differences removes positive serial correlation from both models, but it introduces negative serial correlation into the skew model. The error term of the first difference equation is $e_1 - e_0 = (e'_1 + s_{x_1}) - (e'_0 - s_{x_0})$, which we can rewrite as $(e'_1 - e'_0) + (s_{x_1} - s_{x_0})$, and for the second difference equation it is $e_2 - e_1 = (e'_2 - e'_1) + (s_{x_2} - s_{x_1})$. Both error terms contain s_{x_1}, but with opposite signs, indicating negative correlation.

Serial correlation does not bias the estimates, but it reduces their accuracy because the estimates are no longer the best linear unbiased estimates; there are other linear unbiased estimates with lower variance. The loss of accuracy is not a concern because scatter is small, so the decrease in accuracy is small, and because the skew model is used only to find an initial slope estimate. The final slope estimate is found using the scatter model, which does not suffer from negative serial correlation.

2.3 Iterative Refinement

The skew model estimates the slope m and pitch p by minimizing the sum of the weighted squares of the residuals

$$r_{ji} = (y_j - y_i) - m(x_j - x_i) - p(q_j - q_i).$$

When we attempt to form the sums, we encounter a fundamental difficulty: we do not know the row numbers q_j and q_i. In fact, these quantities are what we ultimately wish to find by reconstructing the punch matrix. If we knew them, there would be no need to apply the Method of Least Squares.

We do not need the row numbers, however, but only the differences between them, as we see from the pitch term $p(q_j - q_i)$. The residuals are formed from the displacement equations by taking first differences, which replaces row numbers with row spacings. The spacings are usually small integers because the displacement between successive events is rarely large. In the absence of error, the residuals are 0 and the row spacings are given by

$$q_j - q_i = \frac{(y_j - y_i) - m(x_j - x_i)}{p},$$

which suggests a way to proceed. Starting with initial approximations to m and p we calculate the row spacings $q_i - q_j$, which are almost never integers

because they are disturbed by errors. We round each row spacing to the nearest integer, giving the true row spacing if the approximations are close enough and the displacement errors are not too large. We use these spacings to find improved estimates for m and p and iterate until the desired accuracy has been achieved.

Rounding will result in faulty row spacing if the estimates for m and p are too far removed from their true values, even in the absence of other errors. We can find the circumstances in which this occurs as follows.

Define the slope error and pitch error by $m = m_0 + m_{\text{err}}$ and $p = p_0 + p_{\text{err}}$, where m_0 and p_0 are the true slope and pitch. If only the slope is in error, the rounded row spacing $q_j - q_i$ is correct if

$$|(m_0 + m_{\text{err}})(x_j - x_i)| < |m_0(x_j - x_i)| + p_0/2.$$

Solving for allowable slope error gives

$$-\frac{p_0}{2|x_j - x_i|} < m_{\text{err}} < \frac{p_0}{2|x_j - x_i|},$$

so small column spacings will accommodate large slope error. The weighting given in Section 3 gives more importance to closely-space event pairs than to widely-spaced ones.

Similarly, if only the pitch is in error, the rounded row spacing $q_j - q_i$ will be correct if

$$|(p_0 + p_{\text{err}})(q_j - q_i)| < |p_0(q_j - q_i)| \pm (p_0 + p_{\text{err}})/2,$$

where the positive sign applies for $p_{\text{err}} > 0$ and the negative sign for $p_{\text{err}} < 0$. Solving for allowable pitch error yields

$$-\frac{p_0}{2|q_j - q_i| + 1} < p_{\text{err}} < \frac{p_0}{2|q_j - q_i| - 1},$$

so small row spacings will accommodate large pitch error. As with slope, the weighting gives more importance to closely-spaced pairs than to widely-spaced ones.

If only the displacements are in error, the row spacing $q_j - q_i$ will be correct if $|e_j - e_i| < p_0/2$. This is always the case if all displacement errors are less than $p_0/4$ in magnitude.

It is clear that an algorithm that uses the skew model or scatter model must find the row spacings $q_j - q_i$ using already-available parameter estimates, so we must start with approximations to the parameters and refine them. This approach is consistent with Gauss's suggestion (see [7], p. 21, Remark 4) to consider each parameter as comprised of two parts: a known (approximated) part and an unknown (much smaller) part.

Starting with the approximations to slope and pitch as given in Section 4.5 (for the first iteration) or the previous slope and pitch estimates, we find the incremental changes, called *adjustments*, that minimize the sum of the weighted squares of the residuals. We update the prior parameter estimates to create the current estimates and repeat until the adjustments are as small as desired. Mistaken row spacings, known as *aliasing*, usually become fewer and fewer as iteration proceeds, but for poorly-made rolls (or poor scans) they may not be removed completely, so the number of iterations should be limited.

Since both models are used for iterative refinement, we introduce a change in notation. For the skew model, let M and P denote the current estimates or initial approximations to slope and pitch, and let m and p denote the adjustments. Using this notation, the adjusted estimates $M + m$ and $P + p$ minimize the sum. Substituting into the expression for the skew residual gives

$$r_{ji} = (y_j - y_i) - M(x_j - x_i) - P(q_j - q_i) - m(x_j - x_i) - p(q_j - q_i).$$

We can simplify this equation and return it to its original form by introducing the concept of *corrected displacements*. Let y'_j be the event displacement y_j corrected for the effects of M and P,

$$y'_j = y_j - Mx_j - Pq_j,$$

and similarly for y_i. Then the corrected displacement difference is

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - P(q_j - q_i),$$

and the skew residual becomes

$$r_{ji} = (y'_j - y'_i) - m(x_j - x_i) - p(q_j - q_i),$$

where the row spacing $q_j - q_i$ is approximated by

$$q_j - q_i = \frac{(y_j - y_i) - M(x_j - x_i)}{P}.$$

For the scatter model, let S'_0, S'_1, S'_2, ..., S'_{d-1} and P be the current estimates or initial approximations to the composite scatter and pitch, and similarly let $s'_0, s'_1, s'_2, \ldots, s'_{d-1}$ and p be the adjustments required to minimize the sum of the weighted squares of the residuals. With this notation, the adjusted estimates $S'_0 + s'_0$, $S'_1 + s'_1$, $S'_2 + s'_2$, ..., $S'_{d-1} + s'_{d-1}$ and $P + p$ minimize the sum. Substituting into the expression for the scatter residual and simplifying gives

$$r_{ji} = (y_j - y_i) - (S'_{x_j} - S'_{x_i}) - P(q_j - q_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i).$$

As with the skew model, we can return the residual equation to its original form by correcting the event displacements. Let y'_j be the event displacement y_j corrected for the effects of S'_{x_j} and P,

$$y'_j = y_j - S'_{x_j} - Pq_j,$$

and similarly for y_i. Then the corrected displacement difference is

$$y'_j - y'_i = (y_j - y_i) - (S'_{x_j} - S'_{x_i}) - P(q_j - q_i),$$

and the scatter residuals become

$$r_{ji} = (y'_j - y'_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i).$$

In practice, we cannot apply the scatter model without an approximation to slope, because the model is susceptible to spurious solutions due to aliasing in the presence of a large amount of skew. To avoid a false solution, we must find a slope estimate M separately (using the skew model is an excellent way to do this) and add a slope term to the corrected scatter displacement difference, which becomes

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - (S'_{x_j} - S'_{x_i}) - P(q_j - q_i),$$

where the row spacing $q_j - q_i$ is approximated by

$$q_j - q_i = \frac{(y_j - y_i) - M(x_j - x_i) - (S'_{x_j} - S'_{x_i})}{P}.$$

We use the revised notation that considers m, s', and p to be adjustments to their corresponding parameter estimates M, and S', and P in what follows.

3 Weighting for Least Squares Equations

As we saw in the last section, the Method of Least Squares refines estimates of the parameters of a music roll by minimizing the sum of the weighted squares of the residuals

$$r_{ji} = (y'_j - y'_i) - m(x_j - x_i) - p(q_j - q_i)$$

for the skew model, and

$$r_{ji} = (y'_j - y'_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i)$$

for the scatter model.

The weights associated with the residuals are inversely proportional to the variances of the errors in the difference equations used to create the residuals, and are thus a measure of the relative accuracies of the difference equations. In this section, we show how to find the weights.

The uncertainties in each of the variables y_j, y_i, m, and p contribute to the variance of the difference equation. Note that we do not consider scatter to be a separate source of uncertainty. In the skew model the scatter is subsumed into the error associated with the displacement, that is, s_{x_i} is subsumed into e_i, and its uncertainty is subsumed with it. In contrast, in the scatter model the uncertainty associated with the scatter, rather than the scatter itself, is subsumed in this way. We consider the contribution of each variable in turn.

The displacements y_i are subject to random errors arising from several sources: irregularly-shaped holes caused by dull punches, torn webbing, imperfections in the scanner camera, and other sources. The error of each measurement is independent of the error of all the other measurements, so in the absence of

other sources of uncertainty the variance of the displacement difference $y_j - y_i$ is twice the variance of y_j or y_i. We assign a weight of 1 to the uncertainty of the displacements of an event pair, following Gauss's statement (see [4], Article 7), that "the weight of one of the classes of observations should be set to one."

If the events in a pair are in different columns, nonzero slope will alter the difference displacement $y_i - y_j$. The events can appear in any column, so the column spacing $x_j - x_i$ can be positive, negative, or zero. The slope m adds a displacement $m(x_j - x_i)$ to $y_j - y_i$. The variance associated with this slope displacement is $(x_j - x_i)^2 \operatorname{var}(m)$, where $\operatorname{var}(m)$ is the variance of m. Let X be the column separation at which the variance due to slope variation equals the variance of the displacements of an event pair. Then the weight due to slope is $1/[(x_j - x_i)^2/X^2]$.

If the events in a pair are in different rows, the displacement difference $y_j - y_i$ is disturbed by row-to-row pitch variations. This disturbance is analogous to that of the balance wheel of a chronometer, so once again Gauss's work on determining longitudes by chronometer applies directly to music rolls. In that problem, the uniform rotation of the earth's surface was measured by a chronometer subject to irregular advance. Here the roles are reversed: the irregular row advance is measured by a highly uniform measuring roller, but the underlying problem is the same.

The variance is proportional to the number of rows separating the events in a pair, which we can see as follows. (This derivation parallels that of Gauss in [7], p. 7.) The displacement s between two events separated by n rows, where $n > 0$, is subject to random errors caused by irregular row advance, and consequently s is a random variable. However, s is itself the sum of n random variables, each one arising from a single row advance. Let e_i denote the error associated with the advance to row i from the preceding row. Then e_i is a random variable with zero mean, and the displacement between row s_i and the preceding row is $s_i = p + e_i$, where p is the mean pitch, so s is given by

$$s = (p + e_1) + (p + e_2) + (p + e_3) + \ldots + (p + e_n).$$

The expectation of s is $\mathrm{E}(s) = np$. The e_i are all of the same class and have the same variance, and each advance is independent of all the others, so the e_i

are independent and therefore uncorrelated. Thus the variance of their sum is the sum of their variances, so the variance of s is $n\,\mathrm{var}(e_i)$, where $\mathrm{var}(e_i)$ is the variance of e_i. Note that this result is independent of the probability density. Let Q be the row separation for which the variance due to irregular row advance is equal to the variance of the uncertainty of the displacement of an event pair. The weight associated with irregular row advance is then $1/(|q_j - q_i|/Q)$.

The three sources of error described above are independent and therefore uncorrelated, so the variance of their sum is the sum of their variances. Let w_{ji} denote the weight due to all of them. This composite weight is the inverse of the sum of the inverses of the individual weights, so it is given by

$$w_{ji} = \frac{1}{1 + \dfrac{(x_j - x_i)^2}{X^2} + \dfrac{|q_j - q_i|}{Q}}.$$

We do not need to include the slope contribution to the composite weight if the slope estimate is fairly accurate and we limit the event pairs to a short length of the roll, or after the dynamic skew has been removed (see Section 6). In both of these cases the weight is given by

$$w_{ji} = \frac{1}{1 + \dfrac{|q_j - q_i|}{Q}}.$$

The values of X and Q are determined by the magnitude of the errors in a given roll, and should ideally be chosen on a roll-by-roll basis. However, since the weight does not need to be especially accurate, one set of values can be used for many different rolls. I have found that considering a column separation of 8 and a row separation of 25 mm as having uncertainties equal to that of an event pair is suitable for a wide variety of rolls and scanners, so I use $X = 8$ and $Q = 25$ mm/pitch as default values in my work.

4 Refining Slope and Pitch Estimates with the Skew Model

The skew model, derived in Section 2, refines estimates of the slope and pitch of a music roll by minimizing the sum of the weighted squares of the residuals

$$r_{ji} = (y'_j - y'_i) - m(x_j - x_i) - p(q_j - q_i).$$

Let U be the sum and let w_{ji} be the weight as found in Section 3. Then the skew model minimizes

$$U = \sum w_{ji} r_{ji}^2 = \sum w_{ji} \left[(y'_j - y'_i) - m(x_j - x_i) - p(q_j - q_i)\right]^2.$$

We minimize U by equating its gradient to 0. It is well known that U is convex, so this approach finds a minimum, not a maximum or saddle point. To equate the gradient to 0, we take partial derivatives with respect to m and p and equate both to 0:

$$\begin{aligned}
\frac{\partial U}{\partial m} &= -2 \sum w_{ji}[(y'_j - y'_i) - m(x_j - x_i) - p(q_j - q_i)](x_j - x_i) = 0, \\
\frac{\partial U}{\partial p} &= -2 \sum w_{ji}[(y'_j - y'_i) - m(x_j - x_i) - p(q_j - q_i)](q_j - q_i) = 0.
\end{aligned}$$

These equations can be solved by substitution, since there are two equations in two unknowns, but we put them in matrix form instead for simplicity of presentation and parallelism with the derivation of the scatter model in Section 5. To do this, we first rewrite the equations as

$$m \sum w_{ji}(x_j - x_i)^2 + p \sum w_{ji}(q_j - q_i)(x_j - x_i) = \sum w_{ji}(x_j - x_i)(y'_j - y'_i),$$

$$m \sum w_{ji}(x_j - x_i)(q_j - q_i) + p \sum w_{ji}(q_j - q_i)^2 = \sum w_{ji}(q_j - q_i)(y'_j - y'_i),$$

which can be written in matrix form as

$$\begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \begin{bmatrix} m \\ p \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}.$$

For compactness, we write this as

$$\mathbf{A}[m\, p]^{\mathrm{T}} = \mathbf{b},$$

where $\mathbf{A}$ is the 2×2 coefficient matrix, $\mathbf{b}$ is the 2×1 product vector, and T indicates that the 1×2 row vector $[m\, p]$ is transposed to become the 2×1 column vector $\binom{m}{p}$ of solutions. The matrix elements are given by

$$\begin{aligned} a_{00} &= \sum w_{ji}(x_j - x_i)^2, \\ a_{01} &= \sum w_{ji}(q_j - q_i)(x_j - x_i), \\ a_{10} &= \sum w_{ji}(x_j - x_i)(q_j - q_i), \\ a_{11} &= \sum w_{ji}(q_j - q_i)^2. \end{aligned}$$

As is well known, $\mathbf{A}$ is symmetric and positive definite, so $a_{10} = a_{01}$. The elements of the product vector $\mathbf{b}$ are

$$\begin{aligned} b_0 &= \sum w_{ji}(x_j - x_i)(y'_j - y'_i), \\ b_1 &= \sum w_{ji}(q_j - q_i)(y'_j - y'_i). \end{aligned}$$

This derivation suggests the algorithm presented next for refining slope and pitch using the skew model. It makes use of actuate events or release events, but not both, because the punch length is unknown.

For each event pair, we must compute the column spacing $x_j - x_i$ and the row spacing

$$q_j - q_i = \text{round}\left[\frac{(y_j - y_i) - M(x_j - x_i)}{P}\right],$$

where $\text{round}(x) = \lfloor x + 1/2 \rfloor$ denotes the integer closest to x. We must also compute the corrected displacement difference

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - P(q_j - q_i)$$

and the weight w_{ji}. After initializing the slope and pitch approximations M and P, we proceed as follows.

Algorithm 1. A first algorithm for refining slope and pitch estimates M and P.

procedure
▷ Initialize

initialize 2×2 matrix $\mathbf{A} = \mathbf{0}$
initialize 2×1 vector $\mathbf{b} = \mathbf{0}$
initialize total number of events n
initialize event i = null event
select active event type (actuate or release)

▷ Fetch next event
for $j = 0$ to $n - 1$ **do**
fetch event j
if event j is not of selected type **then**
continue
end if

▷ Fill matrix and vector
if event i is null event **then**
event $i \leftarrow$ event j
else
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
compute corrected displacement difference $y'_j - y'_i$
compute weight w_{ji}
a_{00} += $w_{ji}(x_j - x_i)^2$
a_{01} += $w_{ji}(q_j - q_i)(x_j - x_i)$
a_{11} += $w_{ji}(q_j - q_i)^2$
b_0 += $w_{ji}(x_j - x_i)(y'_j - y'_i)$
b_1 += $w_{ji}(q_j - q_i)(y'_j - y'_i)$
event $i \leftarrow$ event j
end if
end for

▷ Solve matrix equation and update parameter estimates
$a_{10} \leftarrow a_{01}$
solve $\mathbf{A}[m\ p]^{\mathrm{T}} = \mathbf{b}$
M += m
P += p
end procedure

4.1 Multiple First Differences

For most rolls, most difference displacement errors are less than the 1/2 row error required to find the row spacing correctly in the absence of other errors and most of the row spacings will be correct, but an out-of-place perforation can disturb the row spacings in its vicinity. We can ameliorate the effects of a badly-placed perforation by pairing events more widely spaced than just the adjacent ones.

If we form first differences by subtracting each displacement equation from the one following its successor, rather than from the successor itself, we will effectively obtain two sets of equations: odd-numbered events will appear in one set, even-numbered events in the other. Each set traces a path through the roll, from (nearly) the first event to (nearly) the last. Similarly, the set of equations obtained by forming the usual first differences traces a full path, from the first event to the last. If we form first differences in both ways we will trace three separate paths through the roll, yielding nearly double the number of equations, because almost every event will be paired with four nearby events rather than two. This decreases the deleterious effects of a single badly-placed perforation and reduces the effects of aliasing.

If we pursue this idea further and pair each event with events further removed from it, we will further diminish the effects of a poorly-placed perforation. Let the number of pairings be N. Then the number of paths through the roll is $1 + 2 + \cdots + N = N(N + 1)/2$. If t is the separation of the events in a pair, the number of pairs for a given t is $n - t$ if $n \geq t$, 0 otherwise. Then for $n \geq N$ (the usual case), the number of pairs, and thus the number of difference equations, is

$$(n-1) + (n-2) + (n-3) + \ldots + (n-N) = nN - \frac{N(N+1)}{2}.$$

If $n \gg N$ (also the usual case), the number of difference equations is only slightly less than nN, so the total number of equations is increased by nearly a factor of N.

Note that we form first differences in every case. Subtracting a displacement equation from the equation following its successor forms a first difference of separation $t = 2$, not a second difference, which is the first difference of first differences, analagous to a second derivative.

4.2 Excluded Events

It sometimes happens that the events for one or more columns in a scan are not aligned with rows, reducing the accuracy of the estimates. By far the most common cause of unaligned events is torn edges, which create spurious actuate and release events. Another common cause is torn webbing, which can cause one or more columns to have many of the actuate or release events, or both, to be disturbed from their original positions.

Yet another cause is manually-inserted punches. These were often inserted at the factory to correct rolls in which the perforator failed to perforate all of the holes. Occasionally, they are inserted by a roll owner to correct the same type of perforator errors if they were not corrected at the factory, or to modify the performance to the owner's sense of what it should be.

The accuracy of the estimates can be improved by excluding the events in the affected columns from the computations. The columns to be excluded must be selected by the operator in advance, based on visual examination of the roll or scan.

4.3 A Closer Look at Estimating Pitch

The pitch estimate suffers from a loss of accuracy if there are multiple events in a single row. In particular, it will be low if the event stream has been sorted by displacement. This occurs because most data formats used for a reduced scan represent positions using the differential displacement between successive events, but they do not support negative differential displacements.

If the events in a pair are in different rows, the skew model residual

$$r_{ji} = (y'_j - y'_i) - m(x_j - x_i) - p(q_j - q_i),$$

attributes the corrected differential displacement $y'_j - y'_i$ to both the slope and the pitch, but for events in the same row the pitch term $p(q_j - q_i)$ disappears, and the displacement is attributed only to the slope. Sorting the events by displacement ensures that $y_j \geq y_i$, which introduces a positive bias to the mean of $y'_j - y'_i$, so when $y'_j - y'_i$ is not attributed to the pitch estimate, the estimate will be low.

The deficit in the pitch estimate depends on the value of N, the number of events paired with each actuate or release event. Larger values of N give rise to smaller deficits. For $N = 12$, the pitch estimate is low by typically one part in a few thousand, increasing the chance of faulty row assignments during punch matrix reconstruction following white space; the likelihood is proportional to the length of the white space. There is one place in most rolls where lengthy white space appears, risking faulty row assignments following it: between the end of the performance and the rewind perforation.

To overcome this difficulty, we find the pitch adjustment separately using the *pitch model*, a modification of the skew model that takes the slope to be known and considers only events in different rows. The pitch model minimizes the sum of the weighted squares of the residuals

$$r_{ji} = (y'_j - y'_i) - p(q_j - q_i),$$

where the corrected displacement difference is given by

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - P(q_j - q_i).$$

The procedure for doing this requires a ring buffer to contain the previous events. If the current event and the most recent event in the ring buffer are in the same row, the current event is discarded; otherwise, it is used for the calculations and subsequently placed in the buffer.

Since the pitch model is derived from the skew model, it suffers from negative serial correlation caused by subsuming the scatter into the error terms. In spite of this shortcoming, using it results in a sharp increase in the accuracy of the pitch estimate.

4.4 An Improved Algorithm for Refining Slope and Pitch Estimates

We can improve Algorithm 1 by using multiple first differences, excluding columns with unaligned events, and finding the pitch estimate using only events in different rows. The algorithm given here incorporates all of these improvements. It requires two ring buffers with a capacity of N events each, to contain the previous events for determining the slope and the pitch. In my work I use $N = 12$, which I have found provides many more event pairs,

but without pairing events too far removed from each other along the roll length.

Algorithm 2. An improved algorithm for refining slope and pitch estimates M and P.

procedure
▷ Initialize
initialize slope ring buffer empty
initialize pitch ring buffer empty
initialize 2×2 matrix $\mathbf{A} = \mathbf{0}$
initialize 2×1 vector $\mathbf{b} = \mathbf{0}$
initialize total number of events n
select event type (actuate or release)
select columns to be excluded

▷ Fetch next event
for $j = 0$ to $n - 1$ **do**
fetch event j
if event j is not of selected type **then**
continue
end if
if event j column is excluded **then**
continue
end if

▷ Fill matrix and vector slope elements
if slope ring buffer is empty **then**
install event j in slope ring buffer
else
for each event i in slope ring buffer **do**
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
compute corrected displacement difference $y'_j - y'_i$
compute weight w_{ji}
a_{00} += $w_{ji}(x_j - x_i)^2$
a_{01} += $w_{ji}(q_j - q_i)(x_j - x_i)$
b_0 += $w_{ji}(x_j - x_i)(y'_j - y'_i)$

end for
if slope ring buffer is full **then**
remove event at rear of slope ring buffer
end if
append event j to front of slope ring buffer
end if

▷ Fill matrix and vector pitch elements
if pitch ring buffer is empty **then**
install event j in pitch ring buffer
else
fetch event i at front of pitch ring buffer
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
if $(q_j - q_i) \neq 0$ **then**
for each event i in pitch ring buffer **do**
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
compute corrected displacement difference $y'_j - y'_i$
compute weight w_{ji}
a_{10} += $w_{ji}(x_j - x_i)(q_j - q_i)$
a_{11} += $w_{ji}(q_j - q_i)^2$
b_1 += $w_{ji}(q_j - q_i)(y'_j - y'_i)$
end for
if pitch ring buffer is full **then**
remove event at rear of pitch ring buffer
end if
append event j to front of pitch ring buffer
end if
end if
end for

▷ Solve matrix equation and update parameter estimates
solve $\mathbf{A}[m\ p]^{\mathrm{T}} = \mathbf{b}$
M += m
P += p
end procedure

This algorithm is extremely robust. When it is provided with good initial approximations, it unfailingly delivers accurate estimates of slope and pitch, even from low-quality scans and scans of badly damaged rolls.

4.5 Finding the Initial Approximations

The procedures given above require initial approximations M and P to slope and pitch. Finding an estimate for P does not present any difficulties. The pitch is well known for the more common types of music rolls. For other rolls, it is usually possible to find an approximation by examining the roll itself, or a scan of the roll, if the pitch is not too fine. When this fails, the best approach is to examine a histogram of the spacing of actuate events or release events by eye, which gives excellent results in every case.

Finding an initial approximation to the slope is more difficult. If the skew is known to be small, we can take the initial slope to be 0. In most cases, however, the skew is unknown, and since it can be large, the only way to find the initial slope reliably is by exhaustive search. After finding the pitch approximation, we find the sum U of the weighted squares of the residuals for a number of equally-spaced trial slopes encompassing the range of possible skew, and we select the trial slope that has the minimum sum as the initial approximation. As with estimating slope and pitch, this procedure benefits from pairing each event with a number of preceding events.

This approach finds the global minimum, thereby avoiding the possibility of a false solution caused by starting with a slope approximation near a local minimum. If the slope estimate found in this way is one of the extreme trial slopes, we cannot know if we have found a minimum. That minimum may be beyond the range of the trial slopes, in which case the procedure should halt and the operator should be alerted to the failure.

If we form the matrix and vector elements a_{00}, a_{01}, a_{10}, a_{11}, b_0, b_1 for each trial slope as in Algorithm 2 in addition to forming the sum U, we can refine the selected trial slope and the pitch approximations immediately using the update procedure of Algorithm 2. A less ambitious approach dispenses with refining the pitch estimate. Instead, it finds only U, a_{00}, and b_0 for each trial slope. The slope adjustment is then $m = b_0/a_{00}$, and the selected trial slope M is refined by $M \mathrel{+}= m$ as before.

To find the slope approximation, let the number of trial slopes be T, where $T \geq 2$, let t be the trial index, where $0 \leq t < T$, and let a_t and b_t be the values of a_{00} and b_0 for the index t. In addition, let $M_{\max}$ be the maximum trial slope and let $\Delta M = 2M_{\max}/(T-1)$ be the slope increment. To ensure finding a global minimum, the skew increment $(d-1)\Delta M$, where d is the total number of roll columns, should not exceed the pitch. The following algorithm finds the slope approximation M and refines it.

Algorithm 3. Find approximation to slope M.

procedure

▷ Initialize

initialize ring buffer empty
initialize $T \times 1$ vector $\mathbf{a} = \mathbf{0}$
initialize $T \times 1$ vector $\mathbf{b} = \mathbf{0}$
initialize $T \times 1$ vector $\mathbf{U} = \mathbf{0}$
initialize total number of events n
select event type (actuate or release)
select columns to be excluded

▷ Fetch next event

for $j = 0$ to $n - 1$ **do**
fetch event j
if event j is not of selected type **then**
continue
end if
if event j column is excluded **then**
continue
end if

▷ Fill vector elements

if ring buffer is empty **then**
install event j in ring buffer
else
for each event i in ring buffer **do**
compute column spacing $x_j - x_i$
if $(x_j - x_i) = 0$ **then**

```
                continue
            end if
            initialize M = M_max
            for t = 0 to T − 1 do
                compute row spacing q_j − q_i
                compute corrected displacement difference y'_j − y'_i
                compute weight w_ji
                a_t += w_ji(x_j − x_i)^2
                b_t += w_ji(x_j − x_i)(y'_j − y'_i)
                U_t += w_ji(y'_j − y'_i)^2
                M −= ΔM
            end for
        end for
        if ring buffer is full then
            remove event at rear of ring buffer
        end if
        append event j to front of ring buffer
    end if
end for

▷ Find slope approximation
initialize i = 0
for t = 1 to T − 1 do
    if U_t < U_i then
        i ← t
    end if
end for
M = M_max − iΔM
m = b_i/a_i
M += m
end procedure
```

5 Refining Scatter and Pitch Estimates with the Scatter Model

The scatter model, derived in Section 2, refines estimates of the composite scatter and pitch of a music roll by minimizing the sum of the weighted squares of the residuals

$$r_{ji} = (y'_j - y'_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i).$$

Let U be the sum and let w_{ji} be the weight as found in Section 3. The sum is then

$$U = \sum w_{ji} r_{ji}^2 = \sum w_{ji} \left[(y'_j - y'_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i)\right]^2,$$

which we minimize by forming partial derivatives with respect to s'_{x_j}, s'_{x_i}, and p and equating them to 0, yielding the normal equations that we write in matrix form as

$$\mathbf{M}[s'_0\ s'_1\ s'_2\ \ldots\ s'_{d-1}\ p\ ]^{\mathrm{T}} = \mathbf{c}.$$

The number of scatter terms is d, the total number of columns in the music roll. The dimensions of the matrix $\mathbf{M}$ are $(d+1) \times (d+1)$, and both the solution vector $[s'_0\ s'_1\ s'_2\ \ldots\ s'_{d-1}\ p\ ]^{\mathrm{T}}$ and product vector $\mathbf{c}$ have dimensions $(d+1) \times 1$. The matrix is symmetric and positive definite.

To find the estimates we fill the matrix and product vector, both initially 0, one event pair at a time. After all of the event pairs have been included, we solve the matrix equation to yield the estimates. The equations resulting from setting the partial derivatives to 0 are

$$\begin{aligned}
\frac{\partial U}{\partial s'_{x_j}} &= -2 \sum w_{ji}[(y'_j - y'_i) - (s'_{x_j} + s'_{x_i}) - p(q_j - q_i)] = 0, \\
\frac{\partial U}{\partial s'_{x_i}} &= -2 \sum w_{ji}[(y'_j - y'_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i)](-1) = 0, \\
\frac{\partial U}{\partial p} &= -2 \sum w_{ji}[(y'_j - y'_i) - (s'_{x_j} - s'_{x_i}) - p(q_j - q_i)](q_j - q_i) = 0.
\end{aligned}$$

These equations correspond to rows x_j, x_i, and d of the matrix.

The first two partial derivative equations are additive inverses. If the events in a pair are in the same column, updates are not required for these equations because they cancel, reflecting the fact that there is nothing to be learned about scatter from two events in the same column. Otherwise, we rewrite the equations as

$$\begin{aligned} s'_{x_j} \sum w_{ji} - s'_{x_i} \sum w_{ji} + p \sum w_{ji}(q_j - q_i) &= \sum w_{ji}(y'_j - y'_i), \\ -s'_{x_j} \sum w_{ji} + s'_{x_i} \sum w_{ji} - p \sum w_{ji}(q_j - q_i) &= -\sum w_{ji}(y'_j - y'_i). \end{aligned}$$

We include the contribution of an event pair to the sums in the first of these equations by updating row x_j of the matrix and product vector as follows:

$$\begin{aligned} m_{x_j,x_j} &+= w_{ji}, \\ m_{x_j,x_i} &-= w_{ji}, \\ m_{x_j,d} &+= w_{ji}(q_j - q_i), \\ c_{x_j} &+= w_{ji}(y'_j - y'_i). \end{aligned}$$

Similarly, we include the contribution of an event pair to the sums in the second of the equations by updating row x_i of the matrix and product vector in this way:

$$\begin{aligned} m_{x_i,x_j} &-= w_{ji}, \\ m_{x_i,x_i} &+= w_{ji}, \\ m_{x_i,d} &-= w_{ji}(q_j - q_i), \\ c_{x_i} &-= w_{ji}(y'_j - y'_i). \end{aligned}$$

Finally, we rewrite the last partial derivative equation as

$$\begin{aligned} s'_{x_j} \sum w_{ji}(q_j - q_i) - s'_{x_i} \sum w_{ji}(q_j - q_i) + p \sum w_{ji}(q_j - q_i)^2 = \\ \sum w_{ji}(q_j - q_i)(y'_j - y'_i). \end{aligned}$$

We include the contribution of an event pair to the sums in this equation by updating row d (the pitch row) of the matrix and product vector in the following way:

$$\begin{aligned} m_{d,x_j} &+= w_{ji}(q_j - q_i), \\ m_{d,x_i} &-= w_{ji}(q_j - q_i), \\ m_{d,d} &+= w_{ji}(q_j - q_i)^2, \\ c_k &+= w_{ji}(q_j - q_i)(y'_j - y'_i). \end{aligned}$$

If the events in a pair are in the same column, the first two updates cancel because $x_j = x_i$. No updates take place if $q_j = q_i$, so contributions from events in the same row are neglected, resulting in a low pitch estimate if the events are sorted by displacement (see Section 4.3).

After all of the event pairs have been included, the matrix $\mathbf{M}$ and product vector $\mathbf{c}$ are complete. Before we can solve the matrix equation, however, we must impose a constraint on the matrix because the system is indeterminate. To see this, let $\sigma'_k = s'_k + u$ for $0 \leq k < d$, where u is any real number. Then if $s'_{x_j} - s'_{x_i}$ is the scatter difference of an event pair, so is $\sigma'_{x_j} - \sigma'_{x_i}$ because

$$\sigma'_{x_j} - \sigma'_{x_i} = (s'_{x_j} + u) - (s'_{x_i} + u) = s'_{x_j} - s'_{x_i}.$$

This ambiguity was introduced by taking first differences. Following Gauss, we resolve it by taking the scatter of one of the columns to be known. It is simplest to consider the scatter to be 0. There is no loss of generality in this choice, which offsets the mean composite scatter by an amount equal to the composite scatter of the selected column, because if the scatter is taken to be some value other than 0, that value can be added to all of the scatter terms.

There are several ways to set the scatter of a selected roll column t to 0. The simplest approach removes row t and column t from the matrix, which can be achieved by overwriting row and column t with 0 after the matrix is filled, or by refraining from filling them. Both approaches leave a vacant row and column that must be accommodated by the procedure that solves the matrix equation. However, this is not a new requirement, because most rolls have many unused columns, leaving other vacant rows and columns in the matrix.

5.1 Refining Composite Scatter and Pitch Estimates

With these preliminary results established, we are prepared to present an algorithm for refining composite scatter and pitch estimates. Much of this procedure parallels that of Section 4 for refining slope and pitch, and both algorithms can operate only on actuate events or release events.

The scatter model is subject to false solutions caused by aliasing if the scan contains a significant amount of skew, so the skew model should be applied first as described in Section 4 to provide initial estimates M and P of the

slope and pitch.

Since scatter is small, we can set the initial values of the composite scatter estimates S'_0, S'_1, S'_2, ..., S'_{d-1} to 0. There is no danger of a spurious solution due to aliasing, as there is with skew.

The algorithm requires a ring buffer with a capacity of N events, where N is the number of previous events to be paired with each current event, a matrix $\mathbf{M}$ with dimensions $(d+1) \times (d+1)$, and a column vector $\mathbf{c}$ with dimensions $(d+1) \times 1$. For each event pair, we must compute the column spacing $x_j - x_i$ and the row spacing

$$q_j - q_i = \text{round}\left[\frac{(y_j - y_i) - M(x_j - x_i) - (S'_{x_j} - S'_{x_i})}{P}\right],$$

where $\text{round}(x) = \lfloor x + 1/2 \rfloor$ denotes the integer closest to x. We must also compute the corrected displacement difference

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - (S'_{x_j} - S'_{x_i}) - P(q_j - q_i)$$

and the weight w_{ji}. In addition, we must select a roll column with at least one perforation along the length of the roll as the column t for which the scatter is considered to be 0. The algorithm follows.

Algorithm 4. A first algorithm for refining composite scatter and pitch estimates S' and P.

procedure
▷ Initialize
initialize ring buffer empty
initialize $(d+1) \times (d+1)$ matrix $\mathbf{M} = \mathbf{0}$
initialize $(d+1) \times 1$ vector $\mathbf{c} = \mathbf{0}$
initialize total number of events n
select active event type (actuate or release)
select columns to be excluded

▷ Fetch next event
for $j = 0$ to $n - 1$ **do**
fetch event j

if event j is not of selected type **then**
continue
end if
if event j column is excluded **then**
continue
end if

▷ Fill matrix and vector
if ring buffer is empty **then**
install event j in ring buffer
else
for each event i in ring buffer **do**
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
compute corrected displacement difference $y'_j - y'_i$
compute weight w_{ji}
if $(q_j - q_i) \neq 0$ **then**
$m_{d,d}$ += $w_{ji}(q_j - q_i)^2$
c_d += $w_{ji}(q_j - q_i)(y'_j - y'_i)$
end if
if $(x_i - x_j) = 0$ **then**
continue
end if
if $x_j \neq t$ **then**
m_{x_j,x_j} += w_{ji}
m_{x_j,x_i} −= w_{ji}
$m_{x_j,d}$ += $w_{ji}(q_j - q_i)$
m_{d,x_j} += $w_{ji}(q_j - q_i)$
c_{x_j} += $w_{ji}(y'_j = y'_i)$
end if
if $x_i \neq t$ **then**
m_{x_i,x_j} −= w_{ji}
m_{x_i,x_i} += w_{ji}
$m_{x_i,d}$ −= $w_{ji}(q_j - q_i)$
m_{d,x_i} −= $w_{ji}(q_j - q_i)$
c_{x_i} −= $w_{ji}(y'_j - y'_i)$
end if
end for

```
            if ring buffer is full then
                remove event at rear of ring buffer
            end if
            append event j to front of ring buffer
        end if
    end for

▷ Solve matrix equation and update parameter estimates
    solve M[s'_0 s'_1 s'_2 ... s'_{d-1} p]^T = c
    for k = 0 to d − 1 do
        S'_k += s'_k
    end for
    P += p
end procedure
```

Note that we do not require the index i to perform the updates to the matrix and vector, only the values y_i and x_i, so there is no need to store i in the ring buffer.

5.2 Minimizing Matrix and Vector Storage

For the procedure of the preceding section, much of the matrix storage—usually more than half—is unused for two reasons: roll columns that lack perforations leave vacant rows and columns in the matrix, and the matrix is symmetric. In this section, we improve the procedure by removing the unused rows and columns and the redundant elements to minimize the storage.

Most rolls have many columns that do not have perforations anywhere along the length of the roll. In particular, columns corresponding to notes near the extremes of the keyboard are often unused. To remove the resulting unused matrix and vector elements, we define a mapping function $f(k)$ that maps each roll column k to a positive integer for use as a matrix or vector index. The mapping is arbitrary; it is only necessary for each active column x_i (a column with at least one perforation along the roll length) to be mapped to a unique integer $X_i = f(x_i)$ in the range $0 \leq X_i < h$, where h is the number of active columns.

Any active roll column can be selected as the one for which scatter is taken

to be known. Let the column number be t. The implementation is simplified slightly by assigning $f(t) = h - 1$, a choice that we adopt in what follows. There is no loss of generality in this choice because any active roll column can be mapped to $h - 1$. Since matrix row and column $h - 1$ are otherwise unused, they must be assigned to the pitch to avoid introducing a vacant row and column.

The column numbers of inactive roll columns (those for which there are no perforations) should be mapped to a value beyond the range of the mapped active columns to allow finding the column x_i corresponding to the index X_i by searching the mapping. This obviates the need for the inverse mapping. Mapping inactive roll columns to a positive value beyond the active range (such as h or d), rather than to a negative value, ensures that the mapped index of a roll column is less than $h - 1$ only for active roll columns other than t, simplifying the implementation slightly.

The mapping removes unused rows and columns from the matrix, relieving the procedure that solves the matrix equation of the need to accommodate vacant rows and columns, which simplifies the procedure. To implement it, we define the composite scatter of a mapped column X_i as $\sigma'_{X_i} = s'_{x_i}$.

Mapping typically reduces the storage requirements by about 30%. Whether or not mapping is used, the storage requirements can be reduced further by almost half by solving the matrix equation using a procedure that exploits the symmetry of the matrix, such as symmetic Gaussian elimination, $\mathrm{LDL}^{\mathrm{T}}$ decomposition, or Cholesky decomposition (see [1], p. 157ff), allowing us to store only the upper or lower triangle. The only modification required to do this is to refrain from updating matrix elements outside of the selected triangle.

Here is the relevant portion of the algorithm above, modified to use mapping and the upper triangle of the matrix to reduce the storage requirements to a minimum.

Algorithm 5. A second algorithm for refining composite scatter and pitch estimates S' and P.

procedure

▷ Initialize

. . .

initialize upper triangle of $h \times h$ matrix $\mathbf{M} = \mathbf{0}$

initialize $h \times 1$ vector $\mathbf{c} = \mathbf{0}$

. . .

▷ Fetch next event

for $j = 0$ to $n - 1$ **do**

. . .

▷ Fill matrix and vector

if ring buffer is empty **then**

install event j in ring buffer

else

for each event i in ring buffer **do**

compute column spacing $x_j - x_i$

compute row spacing $q_j - q_i$

compute corrected displacement difference $y'_j - y'_i$

compute weight w_{ji}

if $(q_j - q_i) \neq 0$ **then**

$m_{h-1,h-1}$ += $w_{ji}(q_j - q_i)^2$

c_{h-1} += $w_{ji}(q_j - q_i)(y'_j - y'_i)$

end if

if $(x_j - x_i) = 0$ **then**

continue

end if

$X_j \leftarrow f(x_j)$

$X_i \leftarrow f(x_i)$

if $X_j < h - 1$ **then**

if $X_i < X_j$ **then**

m_{X_i,X_j} −= w_{ji} [*sic*]

end if

m_{X_j,X_j} += w_{ji}

$m_{X_j,h-1}$ += $w_{ji}(q_j - q_i)$

c_{X_j} += $w_{ji}(y'_j - y'_i)$

end if

if $X_i < h - 1$ **then**

if $X_j < X_i$ **then**

m_{X_j,X_i} −= w_{ji} [*sic*]

end if

```
                $m_{X_i,X_i}$ += $w_{ji}$
                $m_{X_i,h-1}$ −= $w_{ji}(q_j - q_i)$
                $c_{X_i}$ −= $w_{ji}(y'_j - y'_i)$
            end if
        end for
        ...
    end if
end for

▷ Solve matrix equation and update parameter estimates
solve $\mathbf{M}[\sigma'_0\ \sigma'_1\ \sigma'_2\ \ldots\ \sigma'_{h-2}\ p]^{\mathrm{T}} = \mathbf{c}$
for $k = 0$ to $d - 1$ do
    $X_k \leftarrow f(k)$
    if $X_k < h - 1$ then
        $S'_k$ += $\sigma'_{X_k}$
    end if
end for
$P$ += $p$
...
end procedure
```

5.3 An Improved Algorithm for Refining Composite Scatter and Pitch Estimates

A further improvement finds the pitch separately from the scatter to improve the accuracy of the pitch estimate, in the same manner in which it is done in Section 4 for the skew model. The two preceding algorithms can be modified to do this by introducing the use of two ring buffers, one each for scatter and pitch.

Here is the algorithm of the previous section, which minimizes storage and uses the upper triangle, modified to find the matrix pitch entries separately. The last row (the pitch row) and the last column (the pitch column) are no longer transposes of each other, so the last row must be of full width.

Algorithm 6. An improved algorithm for refining composite scatter and pitch estimates S' and P.

procedure
▷ Initialize
initialize pitch ring buffer empty
initialize scatter ring buffer empty
initialize upper triangle and last row of $h \times h$ matrix $\mathbf{M} = \mathbf{0}$
initialize $h \times 1$ vector $\mathbf{c} = \mathbf{0}$
initialize total number of events n
select active event type (actuate or release)
select columns to be excluded

▷ Fetch next event
for $j = 0$ to $n - 1$ **do**
fetch event j
if event j is not of selected type **then**
continue
end if
if event j column is excluded **then**
continue
end if

▷ Fill matrix and vector scatter elements
if scatter ring buffer is empty **then**
install event j in scatter ring buffer
else
for each event i in scatter ring buffer **do**
compute column spacing $x_j - x_i$
if $(x_j - x_i) = 0$ **then**
continue
end if
compute row spacing $q_j - q_i$
compute corrected displacement difference $y'_j - y'_i$
compute weight w_{ji}
$X_j \leftarrow f(x_j)$
$X_i \leftarrow f(x_i)$
if $X_j < h - 1$ **then**
if $X_i < X_j$ **then**
m_{X_i,X_j} $-=$ w_{ji} $[sic]$
end if

m_{X_j,X_j} += w_{ji}
$m_{X_j,h-1}$ += $w_{ji}(q_j - q_i)$
c_{X_j} += $w_{ji}(y'_j - y'_i)$
end if
if $X_i < h - 1$ **then**
if $X_j < X_i$ **then**
m_{X_j,X_i} −= w_{ji} [*sic*]
end if
m_{X_i,X_i} += w_{ji}
$m_{X_i,h-1}$ −= $w_{ji}(q_j - q_i)$
c_{X_i} −= $w_{ji}(y'_j - y'_i)$
end if
end for
if scatter ring buffer is full **then**
remove event at rear of scatter ring buffer
end if
append event j to front of scatter ring buffer
end if

▷ Fill matrix and vector pitch elements
if pitch ring buffer is empty **then**
install event j in pitch ring buffer
else
fetch event i at front of pitch ring buffer
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
if $(q_j - q_i) \neq 0$ **then**
for each event i in pitch ring buffer **do**
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
compute corrected displacement difference $y'_j - y'_i$
compute weight w_{ji}
$X_j \leftarrow f(x_j)$
$X_i \leftarrow f(x_i)$
if $X_j < h - 1$ **then**
m_{h-1,X_j} += $w_{ji}(q_j - q_i)$
end if
if $X_i < h - 1$ **then**

```
                    m_{h-1,X_i} −= w_{ji}(q_j − q_i)
                end if
                m_{h-1,h-1} += w_{ji}(q_j − q_i)^2
                c_{h-1} += w_{ji}(q_j − q_i)(y'_j − y'_i)
            end for
            if pitch ring buffer is full then
                remove event at rear of pitch ring buffer
            end if
            append event j to front of pitch ring buffer
        end if
    end if
end for

▷ Solve matrix equation and update parameter estimates
solve M[σ'_0 σ'_1 σ'_2 ... σ'_{h-2} p]^T = c
for k = 0 to d − 1 do
    X_k ← f(k)
    if X_k < h − 1 then
        S'_k += σ'_{X_k}
    end if
end for
P += p
end procedure
```

5.4 Solving the Matrix Equations

A key step in the algorithms presented thus far is solving a matrix equation. A 2×2 matrix can be solved by substitution, but it is simpler to use the algorithm below, which must be programmed to solve larger matrices.

The matrix equation of Algorithm 4 can be solved by Gaussian elimination or its variants LU decomposition, Doolittle's method, or Crout's method if they are modified to accommodate empty rows and columns. More elaborate techniques such as QR decomposition or SVD decomposition could also be used, but they are not needed to achieve the required accuracy.

The triangular matrix of Algorithm 5 is symmetric and positive definite, so the matrix equation can be solved using symmetric Gaussian elimination,

U^TDU decomposition (the upper-triangle equivalent of LDL^T decomposition), or Cholesky decomposition. The matrix of Algorithm 4 is also symmetric and positive definite, so the same techniques can alternately be used to solve its matrix equation, if they are adapted to accommodate empty rows and columns.

None of the standard approaches can be used to solve the matrix equation of Algorithm 6, but it is a simple matter to create the necessary procedure. If we exclude the last row and column of the $h \times h$ matrix (the pitch row and column), the $(h-1) \times (h-1)$ remaining submatrix is symmetric and positive definite. We call a square matrix such as the one of Algorithm 6 that contains a symmetric positive definite matrix in its upper-left corner a *mixed matrix*. The submatrix can be decomposed using any of the procedures that can be applied to a symmetric positive definite matrix. The remaining row or rows can be decomposed using Gaussian elimination, resulting in a complete decomposition.

Let $\mathbf{A}$ be a square matrix with dimensions $n \times n$. If $\mathbf{A}$ contains a symmetric positive definite matrix in its upper-left corner, let the submatrix dimensions be $m \times m$. Only the upper triangle of the submatrix is used, saving nearly half of the storage. The following algorithm solves the matrix equation $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{b}$ is the $n \times 1$ product vector and $\mathbf{x}$ is the $n \times 1$ solution vector, which is placed in the storage occupied on entry by $\mathbf{b}$. It uses symmetric Gaussian elimination to decompose the $m \times m$ submatrix, saving nearly half of the operations, and uses ordinary Gaussian elimination for the remaining $n - m$ rows. The algorithm solves a matrix equation for a full square matrix if $m = 0$, a mixed matrix equation if $0 < m < n$, and a symmetric positive definite matrix equation if $m = n$, so it can be used to solve any of the matrix equations that appear in this note, with the understanding that it must be modified to skip over empty rows to solve the matrix equation of Algorithm 4.

Algorithm 7. Solve mixed matrix equation using Gaussian elimination.

procedure

▷ Forward elimination

for $j = 0$ to $n - 1$ (or $n - 2$) **do**

$i \leftarrow j + 1$

```
▷ Symmetric rows
        while i < m do
            r ← a_ji / a_jj
            for k = i to n − 1 do
                a_ik −= r a_jk
            end for
            b_i −= r b_j
            i += 1
        end while

▷ Asymmetric rows
        while i < n do
            r ← a_ij / a_jj
            for k = j + 1 to n − 1 do
                a_ik −= r a_jk
            end for
            b_i −= r b_j
            i += 1
        end while
    end for

▷ Back substitution
    for i = n − 1 to 0 do
        for j = i + 1 to n − 1 do
            b_i −= a_ij b_j
        end for
        b_i ← b_i / a_ii
    end for
end procedure
```

The pitch row entries of Algorithm 6 are found using only events in different rows, so the absolute value of the multiplier (r in the algorithm above) can be larger than 1 for the pitch row, increasing roundoff error. This issue could be remedied by performing partial pivoting for $|r| > 1$, but doing so is not necessary. Partial pivoting is not required for the symmetric rows because they are positive definite. The only asymmetric row is the pitch row, which is strongly diagonally dominant because of the relatively weak interaction between scatter and pitch, counteracting the effects of the large multiplier

and obviating the need for pivoting.

5.5 Finding and Removing Residual Slope

The method of Algorithm 6 finds composite scatter and pitch, which is all that is required for creating an improved reduced scan for emulation or punch matrix reconstruction. The composite scatter may contain a small amount of slope and its weighted mean will likely not be 0, but when it is removed from the reduced scan, the slope will be removed with it and the nonzero mean will alter the common offset, which is not of interest.

However, we may also wish to find *simple scatter*, that is, scatter free of slope and with a weighted mean of 0, along with an improved slope estimate that includes the slope from the composite scatter, if only to present the operator with an accurate skew estimate or an accurate display of the scatter. In this section, we find the *residual slope* (slope in the reduced scan not accounted for in the slope estimate), add it to the slope estimate, and remove it from the composite scatter, yielding simple scatter.

We find the residual slope by taking simple scatter (not composite scatter) to be random and considering it to be an error, exactly the approach of the skew model. The constraints of Section 2 ensure that simple scatter exhibits the necessary randomness properties. Finding the slope in this way avoids the negative serial correlation of the skew model (see Section 4), increasing the accuracy of the slope estimate. Composite scatter S'_k and simple scatter S_k are related by

$$S'_k = S_k + m'k + b',$$

where m' is the residual slope and b' is the offset introduced by taking the scatter of one of the roll columns to be 0. To determine m' and b', we map the composite scatter to points on the Cartesian plane and find the line of best fit using weighted least squares.

If there were perforations in all of the roll columns, the coordinate pairs of the points would be $(0, S'_0)$, $(1, S'_1)$, $(2, S'_2)$, $\ldots$, $(d-1, S'_{d-1})$, where d is the total number columns in the roll. If there are no perforations in roll column k, the corresponding point (k, S'_k) does not exist, so the total number of

points is h, the number of active columns in the roll, rather than d. Let the best-fit estimate be $\hat{S}_k$. The standard equation $y = mx + b$ for a line on the Cartesian plane gives it as $\hat{S}_k = m'k + b'$, so the residuals are

$$r_k = S'_k - \hat{S}_k = S'_k - m'k - b'.$$

A straightforward application of weighted least squares produces the matrix equation

$$\begin{bmatrix} \sum W_k & \sum kW_k \\ \sum kW_k & \sum k^2 W_k \end{bmatrix} \begin{bmatrix} b' \\ m' \end{bmatrix} = \begin{bmatrix} \sum W_k S'_k \\ \sum kW_k S'_k \end{bmatrix},$$

where W_k is the weight of S'_k and the summations extend over $0 \le k < d$. We write this equation as $\mathbf{A}[b'\, m']^{\mathrm{T}} = \mathbf{b}$ by defining $\mathbf{A}$ and $\mathbf{b}$ in the obvious way. The matrix $\mathbf{A}$ is symmetric and positive definite.

We can show that solving this equation imposes the constraints of Section 2 as follows. The simple scatter is $S_k = S'_k - m'k - b'$, so the first constraint is $\sum W_k S_k = \sum W_k (S'_k - m'k - b') = 0$, which we rewrite as

$$b' \sum W_k + m' \sum kW_k = \sum W_k S'_k,$$

the upper row of the matrix equation. Similarly, the second constraint is $\sum kW_k S_k = \sum kW_k (S'_k - m'k - b') = 0$, which can be rewritten as the lower row.

After solving for m' and b', the slope estimate is updated by $M \mathrel{+}= m'$. The simple scatter estimates are $S_k = S'_k - m'k - b'$.

The following algorithm determines the simple scatter by finding the slope and offset of the composite scatter and removing them. It also updates the slope estimate. The algorithm is divided into two parts. The first part finds the weights, which we approximate by the diagonal elements of the matrix $\mathbf{M}$ of Section 5.1. It accesses events in the same way as previous algorithms, so finding the weights can be included in any of them, saving a pass through the reduced scan. The second part finds the residual slope and scatter offset, includes the residual slope in the slope estimate, and removes the slope and offset from the composite scatter to yield simple scatter.

Algorithm 8. Refine slope estimate M and find simple scatter estimate S.

procedure
▷ Initialize
    initialize ring buffer empty
    initialize $d \times 1$ vector $\mathbf{W} = \mathbf{0}$
    initialize total number of events n
    select active event type (actuate or release)
    select columns to be excluded

▷ Fetch next event
    for $j = 0$ to $n - 1$ **do**
        fetch event j
        if event j is not of selected type **then**
            continue
        end if
        if event j column is excluded **then**
            continue
        end if

▷ Fill weights
        if ring buffer is empty **then**
            install event j in ring buffer
        else
            for each event i in ring buffer **do**
                compute column spacing $x_j - x_i$
                if $(x_j - x_i) \neq 0$ **then**
                    compute row spacing $q_j - q_i$
                    compute weight w_{ji}
                    W_j += w_{ji}
                    W_i += w_{ji}
                end if
            end for
            if ring buffer is full **then**
                remove event at rear of ring buffer
            end if
            append event j to front of ring buffer
        end if
    end for

▷ Initialize and fill matrix and vector
  initialize upper triangle of 2×2 matrix $\mathbf{A} = \mathbf{0}$
  initialize 2×1 vector $\mathbf{b} = \mathbf{0}$
  initialize total number of roll columns d
  for $k = 0$ to $d - 1$ **do**
    if $W_k = 0$ **then**
      continue
    end if
    a_{00} += W_k
    a_{01} += kW_k
    a_{11} += $k^2 W_k$
    b_0 += $W_k S'_k$
    b_1 += $kW_k S'_k$
  end for

▷ Solve matrix equation and update estimates
  solve $\mathbf{A}[b'\, m']^{\mathrm{T}} = \mathbf{b}$
  M += m'
  for $k = 0$ to $d - 1$ **do**
    if $W_k > 0$ **then**
      $S_k \leftarrow S'_k - m'k - b'$
    end if
  end for
end procedure

6 Finding and Removing Dynamic Skew

The skew in a music roll scan varies along the length of the roll, sometimes to an astonishingly large degree. The peak-to-peak variation in skew can be as large as several rows, whereas the magnitude of the mean skew is rarely more than a single row unless the scanner camera is badly misaligned. Skew of this magnitude will cause audible errors in timing and dynamics. It will also thwart all attempts at punch matrix reconstruction if it is not characterized and removed.

The approaches in the preceding sections determine mean skew, scatter, and pitch with a high degree of accuracy. The next step is to to find the dynamic

skew (the variation in skew from the mean along the roll length) so that it can be removed.

Some of the variation in skew is caused by warpage in the roll itself, largely due to improper storage, but in many cases most of the variation is caused by faulty paper handling in the scanner roll transport. If the scanner advances the roll by pulling it with a takeup spool, the paper must be centered at the start of the scan. The tabs at the start of the roll for affixing the roll to the takeup spool were not always properly centered at the factory, which can cause the roll to oscillate from side to side. The lateral variation in paper position causes a proportionate variation in skew.

Dynamic skew changes slowly over the length of the roll, whatever its source, which allows us to consider the skew in a short length of paper to be uniform. This suggests a way to determine dynamic skew. We assume that the mean skew over a length of the roll about equal to its width, called a *page*, is the skew of the events near the center of the page. Beginning at the start of the roll, we find the mean skew of the first page and record it, after which we advance the page boundaries along the roll length by a small amount, say 1/10 of the page length, and repeat the process. This results in a table of dynamic skew as a function of page displacement that can be used to remove the dynamic skew from the scan. To find the skew, we determine the slope in each page by minimizing the sum of the weighted squares of the residual

$$r_{ji} = (y'_j - y'_i) - m(x_j - x_i),$$

where the corrected displacement difference is given by

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - (S_{x_j} - S_{x_i}) - P(q_j - q_i)$$

for simple scatter, or

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - (S'_{x_j} - S'_{x_i}) - P(q_j - q_i)$$

for composite scatter, using estimates for M, S or S', and P found in the previous sections. The weight w_{ji} should not include the skew term, because the slope estimate is always fairly accurate and a page is only a short section of the roll; see Section 3 for the simplified expression for the weight. The approximate slope of the first page must be found by exhaustive search as in

Section 4, but each succeeding page can refine the slope of its predecessor. There may not be enough events in the first page to find the approximate slope reliably, in which case the length of the first page must be increased.

Removing the dynamic skew can reintroduce a small amount of static skew and scatter, so after dynamic skew is removed the slope and scatter should be refined once again.

7 Refining Release Scatter and Punch Length Estimates

The one parameter not addressed in the previous sections is punch length, the apparent displacement between the actuate event and the release event for a perforation consisting of a single hole. To create an improved reduced scan for emulation or punch matrix reconstruction, the release events must be moved by an amount equal to the difference between the mean punch length and the punch length intended by the manufacturer.

Release scatter would be identical to actuate scatter if the punch length were uniform across the width of the roll. Thus, we can think of release scatter as consisting of the sum of actuate scatter and column-to-column differences in punch length, which are small and unvarying for a given roll. (The choice of actuate scatter as the reference is arbitrary; it is equally valid to refer actuate scatter to release scatter.) In this view, there is nothing to be gained from determining actuate scatter and release scatter separately because they differ by a small fixed amount, so we can focus our attention on finding the mean punch length and punch length differences instead. In what follows, we adopt this approach with actuate scatter as the reference scatter.

We start by finding estimates of the slope M, the scatter S, and the pitch P of the actuate events and removing the dynamic skew, all as presented in the previous sections. With these estimates at hand, let L be the mean punch length estimate, let T_0, T_1, T_2, $\ldots$, T_{d-1} be the estimates of release scatter with a weighted mean of 0, and let t_k be the adjustment to T_k. We initialize L to the nominal punch length and set $T_k = S_k$ for $0 \leq k < d$, after which we refine the estimates by minimizing the sum of the weighted squares of

the residuals $r_{ji} = (y'_j - y'_i) - t_j$. For each residual, we must compute the corrected displacement difference

$$y'_j - y'_i = (y_j - y_i) - M(x_j - x_i) - P(q_j - q_i) - (T_{x_j} - S_{x_i}) - L),$$

with the understanding that event j is a release event and event i is an actuate event. In addition to finding the corrected displacement difference, we must compute the row spacing for each event pair,

$$q_j - q_i = \text{round}\left[\frac{(y_j - y_i) - M(x_j - x_i) - (T_{x_j} - S_{x_i}) - L}{P}\right],$$

where $\text{round}(x) = \lfloor x + 1/2 \rfloor$ denotes the integer closest to x, and the weight w_{ji}. We update the release scatter estimates by adding the adjustments and subtracting the weighted mean of the adjustments, which we add to the mean punch length. This yields release scatter estimates with a weighted mean of 0 and a refined estimate of mean punch length.

Algorithm 9. Refine mean punch length L and release scatter T estimates.

procedure

▷ Initialize

initialize ring buffer empty
initialize $(d+1) \times 1$ vector $\mathbf{t} = \mathbf{0}$
initialize $(d+1) \times 1$ vector $\mathbf{W} = \mathbf{0}$
initialize total number of events n
select columns to be excluded

▷ Fetch next event

for $j = 0$ to $n - 1$ **do**
fetch event j
if event j column is excluded **then**
continue
end if

▷ Fill vectors

if event j is actuate event **then**
if ring buffer is full **then**
remove event at rear of ring buffer
end if

append event j to front of ring buffer
else if event j is release event **then**
for each event i in ring buffer **do**
compute column spacing $x_j - x_i$
compute row spacing $q_j - q_i$
compute corrected displacement difference $y'_j - y'_i$
compute weight w_{ji}
t_{x_j} += $w_{ji}(y'_j - y'_i)$
t_d += $w_{ji}(y'_j - y'_i)$
W_{x_j} += w_{ji}
W_d += w_{ji}
end for
end if
end for

▷ Update estimates
L += t_d/W_d
for $k = 0$ to $d - 1$ **do**
if $W_k > 0$ **then**
T_k += $t_k/W_k - t_d/W_d$
end if
end for
end procedure

It is not necessary to find the simple scatter S. We can replace it by the composite scatter S' if we also replace the release scatter by the *composite release scatter* T'_0, T'_1, T'_2 ..., T'_{d-1}. To make this change, substitute S' for S and T' for T in the derivation and algorithm above.

Finding the mean punch length and release scatter is the last step required before removing the errors from the reduced scan to create an improved reduced scan, free of skew and scatter. The improved file can be created in several different ways. The procedure that I use in my work follows these steps:

- Find initial approximations to M and P (Algorithm 3, Section 4.5).
- Refine these approximations using the skew model to form the initial estimates of M and P (Algorithm 2, Section 4.4).

- Initialize the actuate composite scatter S' and refine it and the pitch estimate P using the scatter model (Algorithm 6, Section 5.3).
- Find the residual slope, add it to the slope estimate M, and remove it from the composite scatter S' to create simple scatter S (Algorithm 8, Section 5.5). Display the simple scatter for the operator's review.
- Find the dynamic skew (Section 6), display it, and remove it.
- Refine the estimates of actuate composite scatter S' and pitch P for a second time.
- Find the residual slope, add it to the slope estimate M, create the simple scatter S and display it, all for the second time.
- Find the dynamic skew, display it, and remove it for the second time.
- Determine the mean punch length L and release scatter T (Algorithm 9, Section 7).
- Create an improved reduced scan by removing M, S, and T from the reduced scan and moving the release events using L. Adjust the punch length and pitch to the values intended by the manufacturer.
- Reconstruct the punch matrix if possible (Section 8).

8 Reconstructing the Punch Matrix

The estimates of slope, actuate scatter, release scatter, and punch length found using the methods presented above can be removed from the reduced scan, yielding an improved reduced scan free of systematic errors. After this is done, the only remaining errors are random disturbances caused by the irregular shapes of individual holes in the roll, random variations in skew, scatter, and pitch, and random errors in the scanner. For many rolls, these residual errors can also be removed, yielding the pattern that controlled the perforator during production. The pattern is called the *punch matrix*, after the X-Y array of perforations in the master roll used to control synchronous perforators. Since the reconstructed punch matrix is free of the systematic and random errors introduced by the perforator, roll, and scanner, it is the

preferred starting point for all further processing, including emulation.

It is usually possible to reconstruct the punch matrix from the scan of a well-made roll if the pitch is not too fine. Rolls with a pitch of about 0.75 mm or more, which includes most American-made rolls, are candidates for punch matrix reconstruction unless the roll is poorly made or suffers from severe paper slips.

Reconstructing the punch matrix removes the residual errors by rounding the displacement of each actuate event and release event in the improved reduced scan to the nearest row. If the error is less than 1/2 row, the displacement will be restored to its intended position. We can judge the reliability of this operation by the amount by which the event is moved. Small moves are more reliable than large ones, and the row assignment for any event that is moved by an amount approaching 1/2 row is suspect. In this case, the operator should be alerted to confirm or alter the row assignment by visual inspection of the roll or the scan.

Before we can round an event to the nearest row, we must determine the row displacement. The only information we have available for doing this is the improved reduced scan, so we must find the row displacements from the event displacements. Fortunately, there is an analogous problem in the field of digital communications, namely recovering a local clock from an incoming data stream, that has been the subject of study for many decades, resulting in a well-known solution that we can apply to punch matrix reconstruction.

The long-established solution uses a phase-locked loop, abbreviated PLL, to generate the local clock. A PLL is a closed-loop control system in which the forward path of the loop consists of a phase comparator, a loop filter, and an oscillator whose frequency is controlled by the loop. We can consider the feedback path as having unity gain. The loop ensures that the phase of the oscillator coincides with the phase of the incoming data stream, which implies that the frequency too matches that of the data stream.

To apply the known solution to our problem, we must replace all phase-locked loop variables related to time with variables related to displacement. (This is identical to the change of variables required to apply Fourier transforms, which are usually presented as a method for representing a function of time

in terms of frequency, to represent a function of displacement in terms of spatial frequency.) The relevant quantities and their units are:

Temporal quantity	Spatial quantity
cycles (unitless)	rows (unitless)
period (seconds/cycle)	pitch (meters/row)
frequency (cycles/second)	spatial frequency (rows/meter)
phase (cycles)	phase (rows)
time (seconds)	displacement (meters)

The spatial units are found from the temporal units by replacing cycles by rows and seconds by meters.

With this change in variables, we can apply a phase-locked loop to punch matrix reconstruction. The first question that arises is which PLL to use. In order of increasing capability (and decreasing stability), the three most widely-used PLL configurations are known as Type 1, Type 2, and Type 3, which differ only in the complexity of their loop filters.

Loop stability is maximized by using the simplest PLL that will accommodate the expected disturbances in the incoming data steam. Here are the steady-state phase errors in the locally-generated clock in response to three different disturbances for each of the three standard PLL types:

Disturbance	Type 1	Type 2	Type 3
Phase step	zero	zero	zero
Frequency step	constant	zero	zero
Frequency ramp	increasing	constant	zero

Zero phase error signifies that the phase of the local clock generated by the PLL is identical to that of the incoming signal.

A constant phase error signifies a fixed phase difference between the local clock and the input signal, with a magnitude proportional to the magnitude of the frequency step.

Increasing phase error signifies that there is a time rate of change in the local clock phase. When this occurs, the PLL is not locked to the incoming signal.

For most music rolls the pitch, and thus the spatial frequency, is remarkably uniform from start to finish. (The variable-pitch case is addressed below.) Thus we do not expect frequency steps or acceleration steps, which create frequency ramps. However, step changes in phase due to paper slips during perforating are common and must be accommodated.

Even though the pitch is nearly constant, it is still subject to small variations caused by stretch in the roll. The steady-state response to a pitch error is a constant phase error inversely proportional to the magnitude of the pitch error. However, since the pitch estimate produced by the method of Section 5 is highly accurate, this phase error will be very small and will not disturb the operation of the loop.

These considerations imply that the correct choice for reconstructing the punch matrix is a Type 1 PLL, the simplest type. However, since the pitch is extremely uniform and its value is known with a high degree of accuracy, it can be simplified further. This is accomplished by replacing the variable-frequency oscillator with one that generates a fixed frequency, along with a new element that provides variable delay. The PLL variant that incorporates this replacement is known as a delay-locked loop, or DLL.

In essence, a DLL is identical to a PLL except that phase is the only state variable, so a first-order control loop in which the loop filter consists of an integrator yields a steady-state phase error of 0 in response to a step change in phase; the steady-state response to a pitch error is a constant phase error. Both of these responses are identical to those of a Type 1 phase-locked loop. The additional phase ϕ in response to a step change of phase ϕ_0 is a simple exponential $\phi = \phi_0(1 - e^{s/s_0})$, where s is displacement and s_0 is a constant. Any procedure that yields this response, as the procedure given below does, effectively implements a delay-locked loop. Widespread interest in delay-locked loops is fairly recent, but I have used them for punch matrix reconstruction since the 1990s.

Before reconstructing a punch matrix, there are two adjustments that must be made to the improved reduced scan.

First, we must increase the lengths of unusually short perforations. These are caused by poorly-shaped holes due to dull punches, torn webbing resulting

in partially-filled holes, and manually-inserted undersize holes. The length of each short perforation should be increased to equal the mean punch length by moving its onset and release events in equal amounts, but in opposite directions.

Second, we must adjust the displacement of each release event by moving it toward the start of the roll by an amount equal to the difference between the punch length and the pitch. This aligns the release events with rows, a necessary condition for assigning rows to them.

These adjustments can be performed one at a time in order, or they can be performed in a single pass that lengthens short perforations before moving the release events. Either way, the adjustments can cause the displacements of the events to be out of order. The punch matrix reconstruction method can accommodate out-of-order events, but it is usually simpler to sort them in advance. If this is not done, there is the danger of assigning row 0 to the first event only to discover that a succeeding event precedes it, which would require a negative row assignment.

To reconstruct the punch matrix using a delay-locked loop, we define these variables:

$$
\begin{aligned}
i &= \text{event index } (1, 2, 3, \dots), \\
s_i &= \text{displacement of event } i \text{ (meters)}, \\
S_i &= \text{predicted displacement of event } i \text{ (meters)}, \\
\sigma_i &= \text{adjusted displacement of event } i \text{ (meters)}, \\
n_i &= \text{row advance between event } i - 1 \text{ and event } i \text{ (rows)}, \\
r_i &= \text{row assigned to event } i \text{ (unitless)}.
\end{aligned}
$$

We also define the following constants:

$$
\begin{aligned}
p &= \text{pitch (meters)}, \\
w &= \text{weight (unitless)}.
\end{aligned}
$$

Using this notation, we first find the predicted displacement S_i of event i from the adjusted displacement σ_{i-1} of the preceding event $i - 1$ by forming

$$S_i = \sigma_{i-1} + n_i \, p,$$

where the row advance n_i is given by $(s_i - \sigma_{i-1})/p$ rounded to the nearest integer. The row r_i assigned to event i is $r_i = r_{i-1} + n_i$.

The amount by which the event must be moved to coincide with the row is $s_i - S_i$. If the absolute value of this quantity approaches $p/2$, the operator should be alerted to a questionable row assignment.

Next we form the adjusted displacement $\sigma_i = S_i + w(s_i - S_i)$, which will be used to find the predicted displacement of the next event. The weight w must fall in the range $0 \leq w \leq 1$ to ensure that the adjusted displacement will fall between the predicted event displacement S_i and the true displacement s_i.

The loop weight (not to be confused with the weight of residual terms when forming the least-squares sums in the previous sections) determines the loop gain, and must be chosen with care. If it is too small, the loop will not track changes in phase and pitch properly, resulting in faulty row assignments. If it is too large, a single out-of-place event will disturb subsequent predicted displacements, again risking wrong row assignments. My experience suggests that the weight can be as large as 1/2 for well-made rolls, but that it should be sharply smaller for rolls that are poorly made. A value of 1/4 appears to be a good choice for general use.

The procedure given above applies to all events after the first one, so we must provide an initial value for σ_1. The simplest choice is to set $\sigma_1 = s_1$, which assumes that the first event is well placed. A better approach starts by setting $\sigma_1 = s_1$ and finds the sum of the squared residuals for the events in a short length of roll roughly equal to the roll width for a number of equally spaced trial offsets, all less than the pitch. When doing this σ should not be adjusted, so that the estimated row displacements will be removed from the first event displacement σ_0 by integral multiples of the pitch. The offset that yields the minimum squared residuals should then be added to σ_1, yielding an improved initial value σ_1.

In practice, only a single variable is required for each of S_i, σ_i, n_i, and r_i. Let the variables be S, σ, n, and r. We initialize σ to σ_1 as described above and set $r = 0$. This initialization allows us to process all of the events, including the first, in the same way. After initialization, we update the variables as

follows:

$$\begin{aligned} n &\leftarrow \lfloor (s_i - \sigma)/p + 1/2 \rfloor, \\ S &\leftarrow \sigma + np, \\ r &\leftarrow r + n, \\ \sigma &\leftarrow S + w(s_i - S). \end{aligned}$$

After processing all of the events, the punch matrix is complete.

If a roll contains paper slips, the row assignments can often be improved by including a parameter q that specifies the magnitude of the maximum anticipated slip, where a positive value for q accommodates short rows and a negative value accommodates long ones. (In practice, short rows are much more common than long rows.) Let n_{slip} denote the slip row advance. Then the updates given above are replaced by the following:

$$\begin{aligned} n &\leftarrow \lfloor (s_i - \sigma)/p + 1/2 \rfloor \text{ as before}, \\ n_{\text{slip}} &\leftarrow \lfloor (s_i - \sigma)/p + q + 1/2 \rfloor, \\ S &\leftarrow \sigma + n_{\text{slip}}\, p, \\ r &\leftarrow r + n_{\text{slip}}, \\ \sigma &\leftarrow S + w(s_i - S) \text{ as before}. \end{aligned}$$

The operator should be alerted to the possible existence of a paper slip if $n \neq n_{\text{slip}}$.

The slip update procedure can be used in the absence of paper slips by setting $q = 0$, so there is no need to choose the procedure depending on whether we wish to accommodate paper slips or not because the latter procedure suffices for both cases.

The delay-locked loop accommodates paper slips, which appear to the loop as step changes in phase, by making an immediate phase adjustment. Since a phase-locked loop adjusts the pitch, rather than the phase, its response to a phase step takes place over many rows instead. We therefore expect a delay-locked loop to exhibit superior performance when paper slips occur, and my experience shows that to be the case.

The update procedure can optionally provide an estimate of the phase noise in the scan by keeping a running sum of $(s_i - S_i)^2$. Let N be the total

number of events. Then the root-sum-square phase noise is approximately $[\sum(s_i - S_i)^2/N]^{1/2}$. The resulting estimate is slightly too small, because the predicted row displacements S_i are derived from the events.

As a practical matter, it is valuable to provide means for selectively excluding one or more columns from the update procedure. This is especially useful for columns near the edges of the roll, which can have spurious actuate and release events resulting from torn edges. The events for excluded columns are assigned rows using the current values of r and σ, but are not used to update these variables. The row r' assigned to an excluded onset or release event is found as:

$$\begin{aligned} n_{\text{slip}} &\leftarrow \lfloor (s_i - \sigma)/p + q + 1/2 \rfloor \text{ as before,} \\ r' &\leftarrow r + n_{\text{slip}}. \end{aligned}$$

Finally, we consider the case in which the pitch increases along the length of the roll. The only such rolls were made by the Rudolph Wurlitzer Company for the Wurlitzer Model 165 band organ, under U.S. Letters Patent 1085986 to August de Kleist and Frank L. McCormick, granted February 3, 1914. These rolls contain multiple selections that play for several minutes each, with a total playing time of 20 minutes or more.

One approach to accomodating the increasing pitch in these rolls that I have used successfully is finding the punch matrix separately for each selection. Since the selections are short, the pitch increase over the length of a single selection is relatively small. The completed individual reconstructions are concatenated to form the reconstructed whole.

An alternate approach would use a Type 2 phase-locked loop to track changes in phase and frequency. Since the response of phase-locked loops to paper slips is poor, this approach should not be used if the rolls are known to have many paper slips.

Yet another approach to accomodating increasing pitch would extend the methods of Sections 4 and 5 to find the acceleration along with the other parameters, and remove the acceleration from the improved reduced scan prior to punch matrix reconstruction. Such extended methods are beyond the scope of this note.

References

[1] G. Dahlquist and Å. Björck, tr. N. Anderson (1974), *Numerical Methods*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

[2] F. M. Gardner (1967), *Phase Lock Techniques*, Second Edition. Wiley, New York.

[3] R. W. Farebrother (1988), *Linear Least Squares Computations*. Marcel Dekker, New York.

[4] C. F. Gauss (1821), *Theoria combinationis observationum erroribus minimis obnoxiae: Pars prior*. Commentationes Societatis Regiae Scientiarum Gottingensis Recentiores, **5**:33–62. Reprinted in Gauss's *Werke*, Königlichen Gesellschaft der Wissenschaften zu Göttingen, **4**:1–26.

[5] C. F. Gauss (1826), *Chronometrische Längenbestimmungen*. Astronomische Nachrichten, **5**:227–234. Reprinted in Gauss's *Werke*, Königlichen Gesellschaft der Wissenschaften zu Göttingen, **6**:455–458.

[6] G. Nash (July, 1993), *Phase-Locked Loop Design Fundamentals*. Application Note AN535, Motorola, Inc.

[7] W. L. Stahnke (June, 2022), *Gauss's Method for Determining Longitudes by Chronometer* (unpublished).