SCORE Staff Placement

Craig Stuart Sapp
23 October 2012 (preliminary draft)

This document provides a detailed analysis of the SCORE editor's (version 4.01000-555) staff placement in Encapsulated PostScript (EPS) output, and explains how to calculate staff placement in bitmaps converted from these EPS files. Analysis of the printing method in the SCORE editor consists of two primary stages: (1) continuous vector graphics printing, and (2) quantization effects. Quantization effects in turn can be split into two sub-types: (1) 4000-DPI print quantization imposed internally by SCORE, and (2) 600-DPI quantization (or any arbitrary rendering resolution) related to stroke width adjustment used to create uniformly thick staff lines when converting vector graphics into a bitmapped image. Modeling of the two quantization types allows for pixel-level localization accuracy of staff lines within bitmapped conversions from SCORE EPS output. This model can then be used to accurately locate staff lines in bitmaps converted from SCORE EPS files.

Table of Contents

1 Overview of SCORE PMX data for staff objects.

SCORE represents graphical objects as lists of floating-point numbers. Each object stored in a data file or internally in memory is represented explicitly on the page with these numbers. The convention for naming an object's numbers is "parameter one" for

the first value, "parameter two" for the second value, and so on. This is usually abbreviated as P1, P2, P3, etc. The positions of parameters in the list indicate their meaning. When the list is shorter than a particular parameter number, the parameter's value is implicitly 0, but this implicit value can also indicate that the default setting for the parameter should be used.

P1 for any object in SCORE has a consistent meaning: the integer part of the number describes the object category that the data represents (any the fractional value is ignored). The value 8 for P1 signifies a staff. P2 for all objects indicates the staff to which the object belongs. There are default positions on the page for each staff, so this value also influences the vertical placement of objects on the page. For example if the object data starts "8 1", this means that the object is a staff, and it is on staff #1 of the page (this is the lowest staff if staves are in their default positions).

P3 controls the left horizontal position of the staff, and P6 controls the right horizontal position of the staff. The horizontal units are not physical, but instead range from 0.0 for the left margin of the printout, and 200.0 for the right margin of the printout. When printing at the default page scaling, the length from 0 to 200 is 7.5″. When scaling the page contents from the print menu by 50%, then the physical length from 0 to 200 is 3.25″.

P4 controls the vertical offset of the staff from its default position on the page. The vertical offset is also controlled by the vertical scaling value stored in P5. The product P4 * P5 controls the final vertical offset from a staff's default position, so the vertical offset (P4=1, P5=1) is equivalent to the physical offset generated by the parameter pair (P4=0.5, P5=2).

Other parameters for staves are not particularly important for staff positioning since they are not used often in standard 5-line staff printing. These extra parameters are not considered in the following analysis:

- P7 indicates how many staff lines on the staff (starting with staff line 1 at the bottom of the staff). If P7=0 then use the default setting of 5 lines. P7=−1 will make the staff invisible (zero staff lines). If P7 is a 3-digit number, then the 100's digit indicates which staff line on the standard 5-lined staff you want to be the bottom of the displayed staff, and the other two digits indicate how many staff lines. For example 304 means that the bottom line would be positioned where the 3rd (middle) line of the 5-line staff would be, and there are 3 lines above this line (for a total of 4 lines).
- P8 is used to indicate the distance from the bottom of the bottom staff on the page to the bottom of the bottom staff on a separate page that is to be placed above the current page (used for splitting very large scores across multiple pages).
- P9 is used to store an instrument number. This information is useful for extracting parts from a full score where the parts may drop out on systems where they are resting.
- P10 is an alternate method of positioning staves vertically (as opposed to the P2/P4/P5 method described below. A non-zero value in P10 indicates the dis-

tance from the bottom of the staff 1 to the bottom of the current staff in units of inches or centimeters (the measurement unit will be given explicitly in binary SCORE .MUS files, but is lost when saving SCORE data to ASCII PMX files).

- P11 controls the thickness of staff lines and the default thickness for ledger lines. Ledger lines are two pixels (based on the target print resolution) thicker than staff lines by default (unless P5 < 0.7, or the ledger lines are associated with a grace note). The thickness of ledger lines is controlled independently from the staff thickness by altering P16 for notes (P1=1).
- P12 has no printing purpose. It is used to hide the staff number in the SCORE editor.

Further details about staff parameters can be found in the SCORE reference manual (version 3.0). Additional parameters may be defined for the Windows version of SCORE (starting with P13).

"PMX" is a command in the SCORE editor that saves all objects on the page to an ASCII text file. Typically .PMX will be the file extension used to distinguish from the binary data of .MUS/.PAG files, or text-based macro files. "Parameter MatriX" is the meaning of the abbreviation PMX. In the resulting text file, each line (except for text and embedded PostScript objects which include a second line of plain text) represents a musical object as a list of numbers separated by spaces. Figure 2 gives example PMX data for a page of music containing only staves and some text. This data can be read into the SCORE editor using the RE command. PMX data is nearly equivalent to the binary .MUS/.PAG files. The only difference is that the binary file format for PMX data also stores the unit (inches or centimeters) that is needed for certain parameters (such as P10 for staves). Note that musical objects can be sorted in any order in PMX/MUS files, not necessarily in time or spatial order.

2 Printing variables

In order to determine the position of staff lines on a page, several variables are specified at print time. These variables are not stored within the SCORE data file describing a page of music (MUS or PMX formats). Instead these variables are set from the print menu, or they can be loaded from a separately stored text file as shown in Figure 1.

- The **SIZE** print setting controls the scaling of music on the page. 1.000 is the default size; 0.5 will cause the music to be displayed at 50%, 2.0 will display at 200%, etc. Staff lines will be 7.5″ (540pt) long when the staff is 200

```
OUTPUT          LPT1:
ROTATE          No
SIZE            1.000
PAPER           LTR
OFFSETS         .50     .75
FREEZE          No
RESOLUTION      600
LINE_width      4
HEADER/FOOTER   None
MIRROR          No
SETSTROKE       Yes
MANUAL_feed     No
SETPAPERTRAY    0
DESC            0
```

Figure 1: Default SCORE print parameter file settings.

horizontal SCORE units long and the SIZE variable is set to 1.0.[1] If the size is 0.5, full-length staff lines will be 3.75″ long, and if the size is 2.0, full-length staff lines will be 15″ long. Note that the SIZE value does not scale the entire page. Margins offsets (see next parameter below) are applied first to move the origin from the bottom left corner of the page to the OFFSETS position before the SIZE parameter is applied to scale the music's size.

- **OFFSETS** controls the width of the left and bottom margins. Figure 1 shows the default left margin of 0.50″, and the default bottom margin is 0.75″. Extra fixed offsets are added to these margins when printing EPS files as described below.

The **RESOLUTION** setting gives the target physical printing resolution. The default resolution is 600 dots per inch. This value is not directly used in the output Encapsulated PostScript data, which only contains vector graphics; but rather it is used to calculate the width of line strokes, such as for staff lines. In this case staff lines have the **LINE_width** of 4 pixels at 600 DPI **RESOLUTION**, or 0.00667″ (0.48pt). An extra 0.0007206pt is added to the line width in SCORE EPS output for some reason. Adding 0.0007206pt to the stroke width causes all staff lines at their default sizes to be 5 pixels wide when converting an EPS file into a 600 DPI bitmap. Note that 0.00072pt is equal to 0.00001″.

The edges of all graphical objects (except for beams which have a separate stroke-width control in the SCORE preferences file) will be outlined with a line of this thickness (which will cause the objects' sizes to be increased by 0.2403603pt on all edges from their basic fill outline). In other words, staff lines have a width of 0.4807206pt, with 0.2403603pt extending above the unstroked line/edge, and 0.2403603pt extending below the unstroked line/edge. Staff lines do not have endcaps, so the left and right edges of staff lines do not have an extra horizontal adjustment of 0.2403603pt. Note that the P5 value of staff lines will adjust the line thickness as described in Section 8 starting on page 16.

The PAPER and ROTATE settings control page selection and orientation but are not directly relevant to staff positioning. Any paper type or rotation (No = portrait/Yes = landscape) will still use the bottom left corner of the page as the origin, and all music printed on the page will be in reference to this origin.

3 Default positions of staff lines

Once the printing variables OFFSETS and SIZE are determined, the default physical positions of staves on a page can be calculated. First, note that the physical margin from the left edge of the page to the left-hand side of the staff lines is equal to the first OFFSET number (0.5″ or 36pt default), plus an additional 0.025″ (1.8pt) shift to the right. Therefore when the left offset is 0.5″ (36pt), the actual left margin will be 0.525″ (37.8pt). The

[1] PostScript printing units are in points, so most of the physical length units in this document are given in points as well as inches. One inch is equivalent to 72 points (1″=72pt).

extra shift of 1.8 points may be done in order to allow extra space for system brackets (which actually ends up being 31.464pt–0.2403603pt = 31.224pt or 0.433667″ from left page edge), or braces (30.168pt–0.24pt = 29.928pt or 0.415667"). Since the full length of staff lines is 7.5″ (540pt), the right margin (distance from the right edge of the page to the right side of the staff lines) will be 8.5″–7.5″–0.5″–0.025″= 0.475″ (34.2pt).

The bottom margin is the second number in the OFFSETS print setting, which is 0.75″ by default. To this margin setting, an extra 4.5pt (0.0625″) offset is added. Therefore the default bottom margin is 0.8125″ (58.5pt) from the bottom edge of the page to the center of the bottom line of the first staff when it is in its default position (subtract 0.2403603pt if the default stroke width is applied to the staff line).

Once the bottom margin is specified, the default vertical positions of staves on the page can be calculated. Staff 1 is the bottom staff on the page. The center of the bottom line on this staff is placed BM+Bx above the bottom edge of the paper, where BM is the "Bottom Margin" as specified in the OFFSETS print setting, and Bx is a fixed length of 4.5pt (0.0625″). Each successive staff number is placed 56.7pt (0.7875″) higher than the previous staff number. This distance is from the center of the bottom line of one staff to the center of the bottom line on the next staff. In other words, the equation that calculates the vertical physical position in inches or points for a staff (referenced to the center of the bottom line on the staff) in its default position is:

$$Vpos = BM + Bx + (P2–1) * Vx * S$$

Where Vx is the constant default vertical spacing between staves (56.7pt, 0.7875″), P2 is the staff number (P2=1 for the bottom staff, P2=2 for the next higher staff, etc.), and S is the SIZE print parameter that scales the page. Note that the scaling factor, S, does not affect the bottom margin or extra bottom offset values. Here are the vertical positions of the left sides of staves from the bottom edge of the page, using the default bottom margin of 0.75″ and default size of 1.0:

Staff 1: BM + Bx + (1–1) * Vx * 1 = BM + Bx = 58.5pt
Staff 2: BM + Bx + (2–1) * Vx = 58.5pt + 56.7pt = 115.2pt
Staff 3: BM + Bx + 2 * Vx = 171.9pt
Staff 4: BM + Bx + 3 * Vx = 228.6pt
Staff 5: BM + Bx + 4 * Vx = 285.3pt
Staff 6: BM + Bx + 5 * Vx = 342.0pt
Staff 7: BM + Bx + 6 * Vx = 398.7pt
Staff 8: BM + Bx + 7 * Vx = 455.4pt
Staff 9: BM + Bx + 8 * Vx = 512.1pt
Staff 10: BM + Bx + 9 * Vx = 568.8pt
Staff 11: BM + Bx + 10 * Vx = 625.5pt
Staff 12: BM + Bx + 11 * Vx = 682.2pt

The distance between staff lines at the default size is 6.3pt (0.0875″), so the height of the default sized staff from the center of the bottom line to the center of the top line is 4 * 6.3pt = 25.2pt (0.35″), plus an additional 0.480726pt for the distance from the bottom edge of the bottom staff line to the top edge of the top staff line. Thus the center of the top line on staff 12 is 682.2pt+25.2pt = 707.4pt above the bottom edge of the paper. If the paper is letter sized (8.5″x11″), then the page height is 792pt, which means that the distance between the top edge of the page and the center of the top line of staff 12 is 792pt–707.4pt = 84.6pt. Note that SCORE quantizes the vertical space between staff lines into 7 * 25 units at 4000 DPI for each diatonic step: 7 * 25 / 4000 * 72 * 2 = 6.3pt.

The horizontal position of the default left staff position is simpler to calculate:

Hpos = LM + Lx

Where LM is the "Left Margin" as specified in the OFFSETS print setting, and Lx is a fixed left margin offset to the right of 1.8pt (0.025″). The length of the staff at the default size is 7.5″ (540pt), so the horizontal position of the right side of the full-length staff will be 7.5″ greater than Hpos.

4 Staff positioning parameters

Five parameters of staff objects (P1=8) control the position of standard 5-line staves on the page (unless P10 is non-zero).

- **P2** is the staff number that controls the default vertical position of the staff on the page as described above.
- **P3** is the horizontal position of the left side of the staff in terms of horizontal SCORE units.
- **P4** is the vertical position of the staff in terms of vertical SCORE units (vertical scale steps, or ½ of the distance between staff lines).
- **P5** is the scaling of the staff, which adjusts the vertical height of the staff but does not affect the horizontal length of the staff (which is controlled by P3/P6).
- **P6** is the horizontal position of the right side of the staff in terms of horizontal SCORE units. If P6=0, then this means that P6 is the default value of 200 for the right margin.

SCORE horizontal units range from 0.0 for the left side of the default staff positions to 200.0 at the right side of the default staff position. These values are not physical, but rather are used to describe ratios of the final physical length. For example, to indent the first system, P3 may be set to 15.0, this means that there is an additional indent from the left margin equivalent to 15/2% = 7.5% of the length for the default staff. At the default length at the default size, this would be 7.5″ * 0.075 = 0.5625″ (40.5pt) to the right of the left margin (including the left margin fixed offset of 0.025″). P6 controls the position of the right side of the staff. So if P6 = 200.0 – 15.0 = 185.0, then the right side of the default staff length would be an extra 0.5625″ to the left of the right margin of the paper.

P4 is the vertical adjustment from the default position for a staff. The physical units of P4 are dependent on two scalings: (1) the SIZE print setting and (2) the P5 value which is the vertical scaling of the staff. In other words, if P4=2 and P5=1 and the SIZE print setting is 1.0, then the staff will be raised on the page by 6.3pt. If P4=4 and P5=0.5 and SIZE is 1.0, then the staff would also be raised on the page by 6.3pt.

As an aside, note that at the default vertical scaling P5=1, the units of P4 are 7/6 the size of the horizontal units of P3. Or in other words, if P5=6/7=85.7142857142857…, the vertical (P4) and horizontal (P3) units have the same physical length. If you want a vertical line to have the same physical length as a horizontal line, then the equation would be: ΔP4 = ΔP3 * 6/7 / P5. For example, a 1-inch line has a horizontal length of 1/7.5*200 = 26.667 P3 units. This would be 22.857 P4 units when the vertical scaling P5 is 1.0.

Considering these 5 staff parameters, the full equation for calculating the physical position of the left edge of a staff, referenced from the center of the bottom line of the staff to the bottom edge of the page is:

$$Vpos = BM + Bx + [(P2–1) * Vx + P4 * P5 * Step] * S$$

Where BM = bottom margin from OFFSETS print setting, Bx is a fixed distance of 4.5pt (0.0625″), P2 is the staff number, Vx is the default distance between the bottom line of successive staff lines (56.7pt, 0.7875″), P4 is the vertical offset in SCORE vertical units, P5 is the staff scaling, Step is the default distance between staff lines divided by 2 (3.15pt, 0. 04375″), and S is the SIZE print setting. Note that if P5=0, then use a value of 1 for P5.

The vertical position of the top of a particular staff is given by the following equation, referenced to the center of the top of a 5-line staff:

$$Vtop = Vpos + Step * 8 * P5 * S$$

Vpos and Vtop are both in reference to the center of staff lines. To calculate the vertical position of the optical boundary of the bottom and top of the staff, subtract ½ of the stroke width for the staff lines which is 0.4807206pt/2 = 0.2403603pt for the default thickness of 4 pixels at 600 DPI (plus 0.00072pt):

$$\text{Optical staff bottom edge} = Vpos – Pixels/DPI/2 \text{ inches} – \varepsilon/2$$
$$\text{Optical staff top edge} = Vtop + Pixels/DPI/2 \text{ inches} + \varepsilon/2$$

where DPI is the dots per inch RESOLUTION setting in the print menu, and Pixels is the LINE_width print setting. The constant ε = 0.0007206pt (about 0.00001″) is an extra width probably used to improve staff-line quantization behavior when converting SCORE EPS output into bitmaps.

The horizontal position of the left side of a staff line is given by the fully generalized equation:

Hpos = LM + Lx + S * (Len * P3/200)

Where Len is the default length of staff lines, which is 7.5″ (540pt). The fully generalized equation that calculates the distance from the left edge of the paper to the right side of a staff is similarly:

Hright = LM + Lx + S * (Len * P6/200)

If P6=0, then use the default value of 200 for the full-length width. Hpos and Hright do not need extra adjustment related to the line stroke-width, although the addition of barlines at a staff edge may increase the apparent width of the staff due to stroking effects on the barlines.

5 Basic simulation of SCORE staff printing

Based on the description of the physical placement of staves on the page in SCORE in the previous sections, the following PERL script will accept any SCORE PMX data file (in ASCII format) and output a US-Letter sized page with identical placement of staves on the page when compared to the SCORE output (excluding quantization errors addressed in later sections).

scrstaff.pl

```
#!/usr/bin/perl
use strict;

# Print variables:
my $S          = 1.0;              # global music scaling
my $LM         = 0.50 * 72.0;      # left margin
my $BM         = 0.75 * 72.0;      # right margin
my $DPI        = 600;              # target print resolution
my $LINE_width = 4;                # pixel width of line strokes
my $Stroke     = (1.0 * $LINE_width) / $DPI * 72.0;  # stroke width
$Stroke += 0.0007206;  # match behavior in SCORE

# Constants:
my $Lx         = 0.025   * 72.0;   # left margin buffer
my $Bx         = 0.0625  * 72.0;   # bottom margin buffer
my $Len        = 7.5     * 72.0;   # default full length of staff
my $Step       = 0.04375 * 72.0;   # vertical diatonic step size
my $Vx         = 0.7875  * 72.0;   # default spacing of successive staves

print "%!PS-Adobe-2.0 EPSF-1.2\n"; # print PostScript marker
print "$Stroke setlinewidth\n";    # set line stroke width
print "gsave\n";
while (my $line = <>) {
   next if $line !~ /^8/;
   printStaffObject($line);
}
print "grestore\n";
exit(0);
```

```
##############################
##
## printStaffObject -- Place a staff on the page based on its SCORE
##      PMX data.
##

sub printStaffObject {
   my ($line) = @_;
   chomp $line;
   my @data   = split(/\s+/, $line);
   my ($P1, $P2, $P3, $P4, $P5, $P6) = @data;
   $P1 *= 1; $P2 *= 1; $P3 *= 1; $P4 *= 1; $P5 *= 1; $P6 *= 1;
   $P5 =   1.0 if $P5 == 0.0;
   $P6 = 200.0 if $P6 == 0.0;
   my $hpos  = $LM + $Lx + $S * ($Len * $P3 / 2.0);
   my $width = $S  * ($Len*($P6-$P3)/200.0);
   my $vpos  = $BM + $Bx + $S*(($P2-1)*$Vx+$P4*$P5*$Step);
   my $linespacing = $Step * 2 * $S * $P5;

   print "   gsave\n";
   print "      %SCORE%", join(" ", @data), "\n";
   print "      $hpos $vpos translate\n";
   printStaffLines(5, $linespacing, $width);
   print "   grestore\n";
}

##############################
##
## printStaffLines -- print the specified number of lines with the given
##     vertical spacing between lines.  Starting point before calling
##     this function is (0, 0).
##

sub printStaffLines {
   my ($count, $vspace, $width) = @_;
   my $hleft  = 0.0;
   my $hright = $hleft + $width;
   my $vpos = 0.0;
   for (my $i=0; $i<$count; $i++) {
      $vpos = $vspace * $i;
      print "      $hleft $vpos moveto\n";
      print "      $hright $vpos lineto\n";
   }
   print "      stroke\n";
}
```

Figure 2 shows example SCORE PMX data for staves in their default positions on a page that can be input into the *scrstaff* program above to generate the EPS file output found further below. A visual representation of the data after being converted to EPS by SCORE is displayed in Appendix I (page 27):

```
8  1  0
8  2  0
8  3  0
8  4  0
8  5  0
8  6  0
8  7  0
8  8  0
8  9  0
8 10  0
8 11  0
8 12  0
t 12 56.360   20. 1 1.327 0 0 0 0 0
_00Default staff positions and sizes
t 12 40.268   15. 1 1.000 0 0 0 0 0
_00Page size 8.5"x11", left margin 0.5", bottom margin 0.75"
```

Figure 2: SCORE PMX data used to generate the page in Appendix I.

Lines starting with "8" are staff objects. Lines starting with "t" are text objects (P1=16 in the SCORE editor), with the text found on the following line. Font "_00" at the start of the text string means Times-Roman. Here is the output of the *scrstaff* program given the PMX data found in Figure 2:

```
%!PS-Adobe-2.0 EPSF-1.2
0.4807206 setlinewidth
gsave
   gsave
      %SCORE%8 1 0
      37.8 58.5 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 2 0
      37.8 115.2 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 3 0
      37.8 171.9 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 4 0
      37.8 228.6 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore

gsave
      %SCORE%8 5 0
      37.8 285.3 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 6 0
      37.8 342 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 7 0
      37.8 398.7 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 8 0
      37.8 455.4 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore

gsave
      %SCORE%8 9 0
      37.8 512.1 translate
      0 0 moveto
```

```
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 10 0
      37.8 568.8 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 11 0
      37.8 625.5 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
   gsave
      %SCORE%8 12 0
      37.8 682.2 translate
      0 0 moveto
      540 0 lineto
      0 6.3 moveto
      540 6.3 lineto
      0 12.6 moveto
      540 12.6 lineto
      0 18.9 moveto
      540 18.9 lineto
      0 25.2 moveto
      540 25.2 lineto
      stroke
   grestore
grestore
```

6 Analysis of SCORE EPS output

As a comparison to the simulated graphical output generated in the last section, the following EPS code represents a single staff at the bottom of the page. This is equivalent to the first staff drawn in the above PostScript code. The SCORE PMX data for the following EPS code is:

```
  8 1 0
```

where P1=8 means a staff object, P2=1 means staff #1, and all other parameters are set to zero (which may mean use the default setting, as for example P6 and P5 in the case of staff objects).

This program contains a section of code at the start of the file with function defines and aliases. For example "`/m /moveto load def`" is a function definition which creates and alias "`m`" which can be used in place of the built-in function "`moveto`". Note that there is an extra space at the start of each line. This is added in SCORE EPS output to avoid problems caused by end-of-line differences on MS-DOS/Windows and Apple systems.

```
%!PS-Adobe-2.0 EPSF-1.2
%%Creator: SCORE (tm) Ver. 4.00, Serial #      0
%%Title: SCORE.MUS
%%BoundingBox: 37 57 579 85
%%DocumentFonts: (atend)
%%EndComments
 /scoredict 200 dict def scoredict begin
 save
 /m /moveto load def /l /lineto load def
 /setstrokeadjust where
 { pop true setstrokeadjust }
 { /m { transform .25 sub round .25 add exch .25 sub round .25 add exch
 itransform  moveto } bind def
 /l { transform .25 sub round .25 add exch .25 sub round .25 add exch
 itransform  lineto } bind def } ifelse
 /tr /translate load def /aw /awidthshow load def
 /e /eofill load def /s /stroke load def /g /gsave load def /r /grestore load def
 /f /findfont load def /sf /setfont load def
 /mkf /makefont load def /lw /setlinewidth load def
```

```
 newpath /SCORE  {
 /size  .01800 def  /wdl   26.7067 def
 size dup scale   wdl lw  1 setlinejoin
 /lmar   2100 def /bmar  27250 def
 lmar bmar tr} def
 g SCORE
 0 -24000 m
 30000 -24000 l
 0 -23650 m
 30000 -23650 l
 0 -23300 m
 30000 -23300 l
 0 -22950 m
 30000 -22950 l
 0 -22600 m
 30000 -22600 l
 s
showpage
r restore
end
%%Trailer
%%DocumentFonts:
```

Simple function renames in the header:

```
/m   /moveto       load def
/l   /lineto       load def
/tr  /translate    load def
/aw  /awidthshow   load def
/e   /eofill       load def
/s   /stroke       load def
/g   /gsave        load def
/r   /grestore     load def
/f   /findfont     load def
/sf  /setfont      load def
/mkf /makefont     load def
/lw  /setlinewidth load def
```

The main non-trivial function definition is `/SCORE`, which is used to do basic setup at the start of a page:

```
/SCORE  {
     /size  .01800 def
     /wdl   26.7067 def
     size dup scale
     wdl lw
     1 setlinejoin
     /lmar   2100 def
     /bmar  27250 def
     lmar bmar tr
} def
```

First the units for the page are set so that "1" represents 0.01800pt rather than 1pt. In other words the point is divided into 55.555555… units. The reason for this scaling is that SCORE prints internally at 4000 DPI, which could be residual behavior from the plotting method that SCORE used for printing before PostScript. The scaling of 0.018 for the page makes each integer unit equal to 1/4000″: 0.018 points/unit / (72 points/inch) = 1/4000 inch/unit, which is a resolution of 4000 DPI. In any case, since 55.5555… is a

repeating fraction, there will be round-off error ≤ 0.00025″ (0. 0000034722pt) in the layout precision since all SCORE position coordinates in the EPS file are integers.

Besides adjusting the page scale by 0.018 to emulate 4000 DPI, the /SCORE function moves the origin up from the bottom left edge of the page to (2100, 27250). Since the printing units are equivalent 1/4000″, the origin it moved to (0.525″, 6.8125″). The horizontal value of 0.525″ matches the measured distance between the left edge of the paper and the left side of the staves when they are in their default positions. The reason for the origin's vertical position of 6.8125″ above the bottom edge of the page is unknown. It is equivalent to the bottom margin specified in the print menu (0.75″ is the default value used in this case, plus 0.0625″ fixed vertical margin offset plus 6.0″). Again, this 6″ vertical offset from the bottom margin may be a remnant of the printing process before PostScript was used. It also may be used to minimize the maximum page coordinate value from 44000 to 24000 for some reason (such as to keep coordinate sizes less than a signed short which is 32767).

The line width that is used to stroke the staff lines is set to 26.7067 units or 0.4807206pt. The expected width is 0.48pt. The significance of the extra 0.0007206pt is uncertain, but may be related to the internal quantization in SCORE EPS position calculations, since 0.0007206pt is 0.0000100083333″. Since 1/4000″ is 0.00025″, the difference of 0.0007206 is equivalent to 1/24.979184 of the length 1/4000″. Excluding the round-off error of 0.0000006pt, the difference is exactly 1/25 of a 4000 DPI pixel. So the default width of a line stroke in SCORE EPS output is 0.48pt plus 1/25 of 1/4000″, which is 0.48072pt plus a round-off error of 0.0000006pt, making the final thickness 0.4807206pt.

When the print setting SETSTROKE is set to YES (see Figure 1), the following code is inserted into SCORE EPS output to redefine the "`l`" and "`m`" functions. The code switches to the rendering coordinate system, then does a quarter pixel shift followed by a rounding to the nearest pixel before shifting back a quarter pixel and switching back into the PostScript coordinate system.

```
/setstrokeadjust where
   { pop true setstrokeadjust }
   { /m {
          transform
          .25 sub round .25 add exch
          .25 sub round .25 add exch
          itransform  moveto
       } bind def
     /l {
          transform
          .25 sub round .25 add exch
          .25 sub round .25 add exch
          itransform  lineto
       } bind def
   } ifelse
```

This code gives a consistent pixel width to stroked lines and is used by most PostScript renderers "to improve line thickness [quantization] consistency in lower resolutions."[2] Official description and motivation of the setstrokeadjust feature which this code is copied from can be found at Adobe's website.[3]

7 Bitmap comparison of simulation to SCORE EPS

A useful method for analyzing the simulated staff printing generated by the program listed in Section 5 is to convert the SCORE Encapsulated PostScript file into a bitmapped image, convert the simulated printing result to a bitmapped image, and then compare the two images by comparing the pixels of the two images. To generate a bitmap from an EPS file, GhostScript is a good choice. GhostScript is typically installed on Linux computers by default (and accessible via the "`gs`" command-line program). To install GhostScript on an Apple OS X computer, you should first install MacPorts.[4] Once MacPorts is installed, type this command in /Applications/Utilities/Terminal.app:

```
sudo port install ghostscript
```

After installing ghostscript, the `gs` command should be accessible. Type the following command in the terminal to see if the `gs` command can be found in the command search path list:

```
which gs
```

The following (or similar) text should then be displayed:

```
/opt/local/bin/gs
```

If no text is displayed, then GhostScript is not likely to be installed on the computer. If GhostScript is installed, you can convert an EPS file into a TIFF image with this terminal command:

```
gs -r600 -dNOPAUSE -dBATCH -sPAPERSIZE=letter \
      -sDEVICE=tifflzw -sOutputFile=output.tif input.eps
```

This command will convert `input.eps` into `output.tif`. The options given to `gs` are:

`-r600`	convert to 600 DPI bitmap
`-dNOPAUSE -dBATCH`	don't go into interactive mode with the gs interpreter

[2] http://www.creativepro.com/article/acrobat-tips-graphics-in-pdfs?page=0%2C2

[3] http://partners.adobe.com/public/developer/en/ps/sdk/5111.Stroke_Adj.pdf, "Emulation of the setstrokeadjust Operator" Technical Note #5111, 31 March 1992, Adobe Systems, San Jose, California.

[4] Install MacPorts from the website http://www.macports.org by downloading and installing the most recent installation package for your particular version of OS X, such as:
https://distfiles.macports.org/MacPorts/MacPorts-2.1.0-10.7-Lion.pkg
which would be the installation file for MacPorts version 2.1.0-10.7 for OS X Lion.

`-sPAPERSIZE=letter`	set the paper size to US Letter. This is required, since SCORE does not store the paper size in its output EPS file, and gs usually uses A4 as the default paper size
`-sDEVICE=tifflzw`	output content is a TIFF image using LZW compression
`-sOutputFile=output.tif`	set output filename to output.tif
`input.eps`	the input EPS file

The following PERL script (called *maketiff*) can be used to convert EPS files into TIF without the need to remember all of the GhostScript options:

```
#!/usr/bin/perl
use strict;
foreach my $file (@ARGV) {
   convert($file);
}
sub convert {
   my ($file) = @_;
   my $basename = $file;
   $basename =~ s/\.[^.]*//;
   my $gsopts  = " -r600";                 # 600 dpi
   $gsopts    .= " -dNOPAUSE -dBATCH";     # non-interactive use of GhostScript
   $gsopts    .= " -sPAPERSIZE=letter";    # use 8.5" by 11" paper
   $gsopts    .= " -sDEVICE=tifflzw";      # B&W TIFF with LZW compression
   $gsopts    .= " -sOutputFile=$basename.tif";
   my $result  = `gs $gsopts $file`;
   print "gs $gsopts $file\n";
   print $result;
}
```

The `convert`, `compare` and `composite` commands, which are part of the ImageMagick package, can be used to examine differences between two image files.[5] These programs are typically pre-installed on Linux systems, and can be installed using MacPorts in OSX with the following command:

```
sudo port install ImageMagick
```

The first comparison method uses `composite` to subtract the two images from each other. Then the `convert` command is used to switch black/white.

```
composite file1.tif file2.tif  -compose difference output.png
convert output.png –negate output.tif
```

The output file in the composite command cannot be a TIFF file; otherwise, the entire image is black for some reason. So in this case the intermediate output is saved to a .PNG file that is then negated and converted into a .TIF file. An alternate method of displaying differences is with the compare command. This command highlights differences using red pixels along with unaltered pixels when they match between the two images:

```
compare file1.tif file2.tif output.png
```

[5] Documentation for these programs can be found on the web at:
http://www.imagemagick.org/script/convert.php,
http://www.imagemagick.org/script/compare.php,
http://www.imagemagick.org/script/composite.php

```
convert output.png output.tif
```

Figure 3 shows the resulting difference image comparing the TIFF image of the SCORE PMX data found in Figure 2 which is printed directly from SCORE with the output from the *scrstaff* PERL script. All pixels (at 600 DPI resolution) are equivalent between the two images, other than the text at the top of the page, which is ignored by the *scrstaff* program.

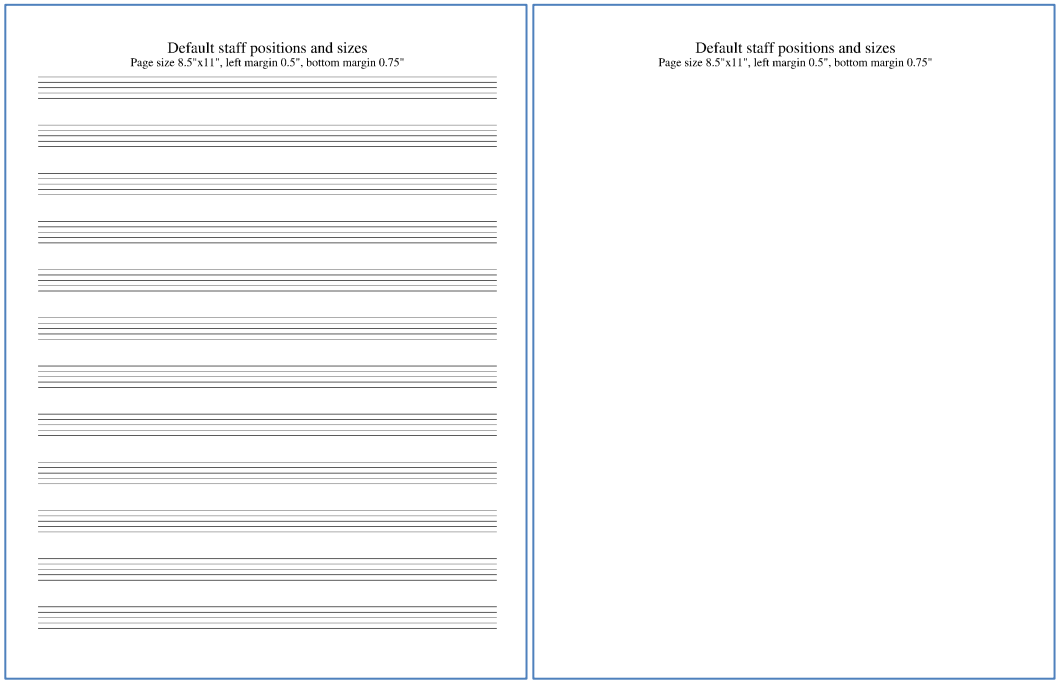


Figure 3: The left image represents the output SCORE EPS file using the PMX data from **Figure 2**. The right image shows the difference between the SCORE EPS output and the simulated EPS output found at the end of Section 5. The simulation was accurate to the pixel in placing the staff lines since there are no different pixels.

8 4000-DPI quantization effects

The exact 600-DPI alignment in the previous section turns out to be a coincidence. Testing a more complicated example with randomly positioned staff lines show 4000-DPI pixel quantization differences between SCORE EPS files and the simulated EPS output from the *scrstaff* program. Figure 4 shows the input SCORE PMX data for the following test of a wider range of staff placement.

```
8  6  20   -23   3.00  178.60
8  2   0    11
8  4 120     0    .25  180.00
8  7   0     0   2.00
8  8 101.45  0    .00  102.36
8  9   0     0   3.00  100.00
8 10 100    -6   3.00
8  1   0
8  5   0     0    .00   10.00
8  6  40     0    .75
8 11  45.63  0   4.00   95.24
8 12  -3     0    .00   30.00
t 12 111.01 14 1 1.327 0 0 0 0 0
_00Random staff placement test
```

Figure 4: SCORE PMX data for a set of randomly placed staff lines.

Figure 5 shows several staff lines that differ by one pixel due to quantization effects. The differences are caused by the SCORE editor printing method, which imposes a quantization of 4000 DPI when printing an EPS file. When this 4000-DPI quantization rounds in such a way that its 600 dpi rounding will be different from the pre-quantized position (about 1/6 of the time for random coordinates).

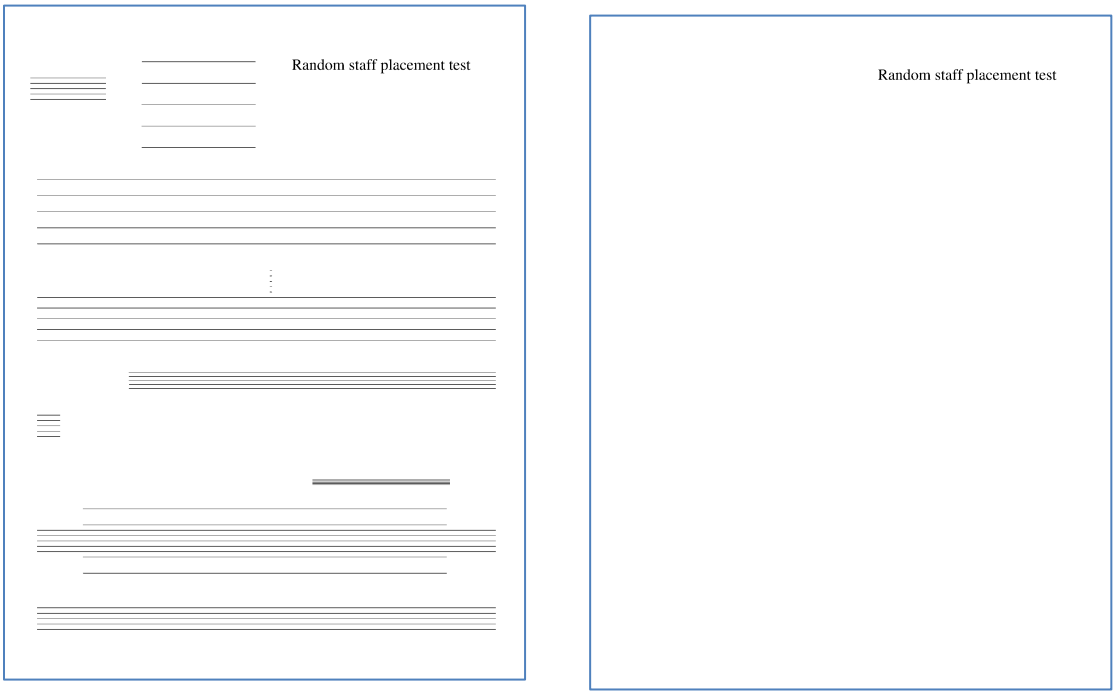


Figure 5: The left image show randomly placed staff lines created from the SCORE PMX data listed in **Figure 3**. The right image shows the difference between the SCORE EPS and simulated EPS after both are converted to 600 DPI images. Notice the occasional one-pixel difference between the two images caused by SCORE 4000 DPI quantization.

To correct for the 4000 DPI quantization used by the SCORE editor to print EPS files, here is a revised version of *scrstaff*. This version of the staff-printing script uses the SCORE EPS method of first switching to a 4000-DPI coordinate system where the origin is shifted to the left margin and six inches above the bottom margin.

```
#!/usr/bin/perl
# scrstaffq = Print staff lines from SCORE PMX data, applying 4000 DPI
#    quantization.  Use -Q option to turn off the quantization.
#
```

scrstaffq.pl

```
# Only one vertical position quantized incorrectly from a test set
# of 1000 staves:
#    8.000000 2.000000 134.886353 -5.738704 1.465246 195.576492
# 4th line of staff is:
#    20232 -20782 m
#    29336 -20782 l
# but should be:
#    20232 -20783 m
#    29336 -20783 l
#

use strict;
use POSIX;

use Getopt::Long;
my $noquantize = 0;
Getopt::Long::Configure("bundling");
GetOptions ( 'Q' => \$noquantize );  # -Q means turn off 4000-DPI quantization

# Command-line options:
my $quant4000 = !$noquantize;      # simulate SCORE EPS quantization at 4000 DPI

# SCORE print menu variables:
my $S          = 1.0;              # global music scaling
my $LM         = 0.50 * 72.0;      # left margin
my $BM         = 0.75 * 72.0;      # right margin
my $DPI        = 600;              # target print resolution
my $LINE_width = 4;                # pixel width of line strokes
my $Stroke     = (1.0 * $LINE_width) / $DPI * 72.0;  # stroke width

# Constants:
my $Lx         = 0.025   * 72.0;   # left margin buffer
my $Bx         = 0.0625  * 72.0;   # bottom margin buffer
my $Len        = 7.5     * 72.0;   # default full length of staff
my $Step       = 0.04375 * 72.0;   # vertical diatonic step size
my $Vx         = 0.7875  * 72.0;   # default spacing of successive staves

my $dpi72to4000  = 4000.0 / 72.0;  # conversion factor from points to 4000 DPI
my $hoffset4000 = ($LM + $Lx) * $dpi72to4000;           # should be 2100
my $voffset4000 = ($BM + $Bx + 6 * 72) * $dpi72to4000;  # should be 27250

$Stroke += 0.0007206;  # match line thickness behavior in SCORE
$Stroke *= $dpi72to4000;
my  $LW = $Stroke;     # Variables for keeping track of the stroke width which
my  $oldLW = $Stroke;  #    is needed since gsave/grestore are not used.

print "%!PS-Adobe-2.0 EPSF-1.2\n"; # print PostScript marker

# print PostScript functions:
print "/m {moveto} def\n";
print "/l {lineto} def\n";
print "/s {stroke} def\n";
print "/lw {setlinewidth} def\n";
print "/tr {translate} def\n";

print "\ngsave\n";
print 1.0/$dpi72to4000, " dup scale\n";  # coordinates from 72 DPI to 4000 DPI
print "$hoffset4000 $voffset4000 tr\n";
print "$LW lw\n";

while (my $line = <>) {
   next if $line !~ /^8/;
   printStaffObject($line);
}
```

```
print "grestore\n";
print "showpage\n";

exit(0);


##############################
##
## printStaffObject -- Place a staff on the page based on its SCORE
##      PMX data.
##

sub printStaffObject {
   my ($line) = @ ;
   chomp $line;
   my @data   = split(/\s+/, $line);
   my ($P1, $P2, $P3, $P4, $P5, $P6) = @data;
   $P1 *= 1; $P2 *= 1; $P3 *= 1; $P4 *= 1; $P5 *= 1; $P6 *= 1;
   $P5 =   1.0 if $P5 == 0.0;
   $P6 = 200.0 if $P6 == 0.0;
   my $width  = $S  * ($Len*($P6-$P3)/200.0);
   my $hlpos  = $LM + $Lx + $S * ($Len*$P3/200.0);
   my $vpos   = $BM + $Bx + $S*(($P2-1)*$Vx+$P4*$P5*$Step);
   my $lspace = $Step * 2 * $P5 * $S;  # spacing between staff lines

   # scale to 4000 DPI coordinates (from 72 DPI coordinates):
   $hlpos  = $hlpos  * $dpi72to4000;
   $vpos   = $vpos   * $dpi72to4000;
   $width  = $width  * $dpi72to4000;
   $lspace = $lspace * $dpi72to4000;

   # print the original SCORE PMX data for the staff:
   print "%SCORE%", join(" ", @data), "\n";

   # if the stroke width has changed, print it.  This code is hard-coded
   # and shoudl be geralize.  But 20.03 is equal to 3 pixels at 600 dpi
   # (one less than default staffline width), plus a small extra amount.
   $LW = $Stroke;
   $LW = 20.03 if $P5 < 0.65;
   if ($LW != $oldLW) {
      printf("    %.4lf lw\n", $LW);
      $oldLW = $LW;
   }

   # quantize new origin (6" above bottom right margin origin)
   my $hoffset4000q = int($hoffset4000);
   my $voffset4000q = int($voffset4000);

   # print the staff on the page according to the PMX parameters:
   printStaffLines(5, $hlpos-$hoffset4000q, $width, $vpos-$voffset4000q, $P5);
}


##############################
##
## printStaffLines -- print the specified number of lines with the given
##     vertical spacing between lines.
##

sub printStaffLines {
   my ($count, $hlpos, $width, $vbottom, $P5) = @_;
   my $hshift = 0.00075;   # hack value to fix 22 /1000 quantizations
   #$hshift = 0.0;   # hack value to fix 22 /1000 quantizations
```

```
   # variables for quantized versions of values:
   my $hlposq   = $hlpos;
   my $vbottomq = $vbottom;
   my $widthq   = $width;

   $hlposq -= 0.001 if $hlposq < 0;
   $hlposq += 0.001 if $hlposq > 0;

   if ($quant4000) {
      $hlposq    = int($hlposq);
      $widthq    = int($widthq);
      $vbottomq  = int($vbottomq);
   }
   my ($vpos, $vposq);
   my $vorigin = $vbottom;

   # The bottom staff line of the staff is at vertcial unit "3" in SCORE.
   # Shift the orgin down 3 steps from the bottom line of the staff.
   # the diatonic step is 175 units at 4000 DPI (3.15pt)
   my $diatonicstep = 175 * $P5;
   $vorigin = $vorigin - 3.0 * $diatonicstep;

   print "% Diatonic step: $diatonicstep\n";
   # print staff lines at diatonic steps 3, 5, 7, 9, 11:
   for (my $i=3; $i<=11; $i+=2) {
      $vpos = $vorigin + $i * $diatonicstep;
      $vposq = $vpos;

      # suppress quantization problems near integers
      $vposq -= 0.0001 if $vposq < 0;
      $vposq += 0.0001 if $vposq > 0;

      $vposq = int($vposq) if $quant4000;
      my $hrposq = $width + $hlpos;

      # suppress quantization problems near integers
      $hrposq -= 0.0001 if $hrposq < 0;
      $hrposq += 0.0001 if $hrposq > 0;

      $hrposq = int($hrposq + $hshift) if $quant4000;

      if (!$quant4000) {
         # avoid small numbers such as -1e8 when not quantizing:
         $hlposq = 0 if $hlposq =~ /e/i;
         $hrposq = 0 if $hrposq =~ /e/i;
         $vposq  = 0 if $vposq  =~ /e/i;
      }

      print " $hlposq $vposq m\n";
      print " $hrposq $vposq l\n";
   }
   print " s\n";
}
```

Figure 6 shows a 600-DPI difference analysis when printing the PMX data from SCORE and from *scrstaffq*. In this case there are no pixel differences between the two output methods, and the quantization effect seen in Figure 5 is now removed.

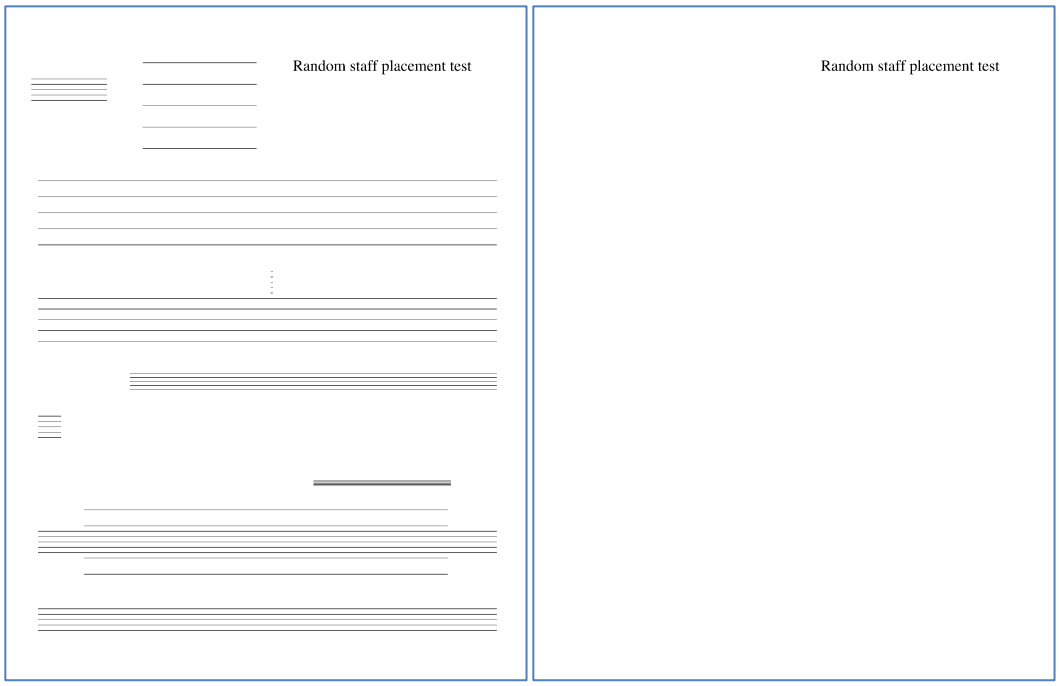


Figure 6: EPS output for the PMX data from **Figure 3**. Image on the right shows the difference between the quantized simulated output compared to the SCORE EPS output. Quantization error pixels from **Figure 5** are no longer present.

Using a more extensive random test described in Section 8.1, only one vertical position on the fourth staff line of the following PMX data is off by one 4000-DPI pixel:

```
8.000000 2.000000 134.886353 -5.738704 1.465246 195.576492
```

The vertical position of the fourth line of the staff is at -20782 in the 4000-DPI coordinate system, when the output from SCORE is at -20783. This is out of a test set of 1000 staves, or 5000 staff lines (so a measured error rate of 0.02%).

An additional complication added to the quantized version of the *scrstaffq* program is that SCORE will shrink the stroke width for staves when P5 is less than 0.65. For example when P5=1, the line thickness is 26.7067 SCORE printing units (0.4807206pt, or 0.006676675″). When P5=0.25, the thickness is 20.03 SCORE printing units (0.36054pt, or (0.0050075″). In other words, when P5 is less than 0.65, the pixel width of staff lines will be decreased by 1 pixel. For example a 0.4807206pt stroke width is 4.006005 pixels at 600 DPI, while 0.36054pt is 3.0045 pixels at 600 DPI.

8.1 Detailed random quantized staff positioning test

To exhaustively verify the quantization boundaries, the following PERL script was used to generate truly (quasi-)random staff positions. This program generates random values for P2–P6 of staff line objects. Figure 7 shows sample output from *makerandomstaff*: the text on the right lists 26 random staff objects, and the image on the right of the figure shows the graphic result of placing 1000 randomly generated staves on a page.

makerandomstaff.pl

```
#!/usr/bin/perl
my $count = 1000;
for ($i=0; $i<$count; $i++) {
   my $P1 = 8;                # P1=8 means staff object
   my $P2 = int(rand(12))+1; # staff number
   my $P3 = rand(200);        # left horizontal position
   my $P4 = rand(20) - 10;    # vertical offset
   my $P5 = rand(3);          # vertical scale
   my $P6 = rand(200);        # right horizontal position
   if ($p6 < $p3) {           # Put $P3 and $P6 is correct order
      my $temp = $p6;
      $p6 = $p3;
      $p3 = $ptemp;
   }
   printf("%f %f %f %f %f %f\n", $P1, $P2, $P3, $P4, $P5, $P6);
}
```

```
8 11 6.198888 9.573973 0.283635 168.128405
8 11 2.873254 -4.797876 2.927321 70.601575
8 7 87.639369 7.994754 0.598376 165.502491
8 7 41.646249 2.260639 0.923335 81.354952
8 12 0.694142 8.659682 1.671497 9.091288
8 7 141.401669 3.957960 1.272665 156.293134
8 6 56.481524 4.920927 0.341730 167.857785
8 10 72.348165 0.396136 1.237993 160.695259
8 6 33.471906 2.824829 2.398684 35.839999
8 1 8.250568 7.964336 0.584617 174.530481
8 12 32.757540 9.446832 2.429939 87.284526
8 10 162.437601 -6.53652 2.742277 170.27050
8 8 27.373900 -3.664126 1.156347 36.968902
8 9 96.360833 2.974862 2.310525 152.645578
8 9 108.015245 -4.306424 2.667651 113.96283
8 3 55.261755 -6.727791 0.970704 190.964604
8 1 52.988382 6.902256 1.430236 97.383340
8 2 2.663138 2.782467 0.111959 147.923603
8 4 139.913069 -3.690463 0.956186 143.73427
8 8 37.481259 -9.526205 0.102742 108.354876
8 7 40.561432 2.533573 1.711105 54.101457
8 4 36.023047 -7.778810 0.266028 152.194445
8 8 122.386297 8.136932 2.105072 176.946497
8 6 139.798412 -8.796644 0.437483 182.64005
8 1 49.466780 -8.467895 2.451192 57.357401
8 10 161.137762 7.532659 2.825202 165.72935
...
```

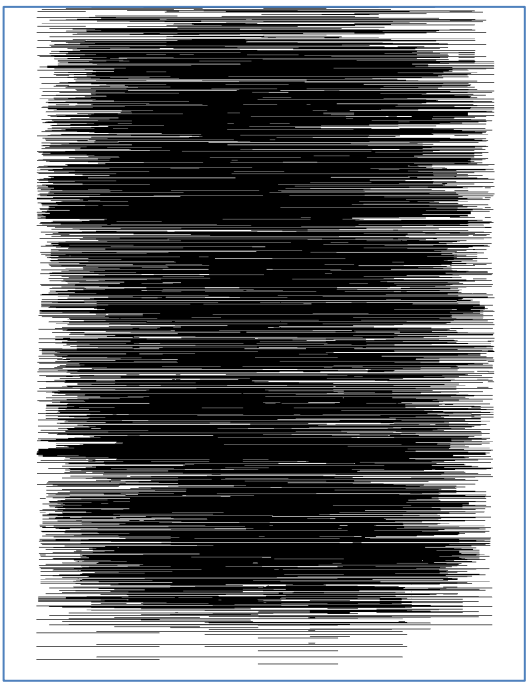


Figure 7: Sample output from *makerandomstaff* program. The page image on the right shows 1000 randomly placed staves on the page at the same time. **Appendix II** starting on page 29 contains the full listing of the 1000 staves.

For careful comparisons between SCORE EPS and simulated EPS files, it is necessary to quantize the above staff data to 32-bit floating-point numbers. SCORE processes all data numbers in this format, so the input to both printing programs should be quantize to 32-bit floats first in order to avoid trivial differences due to quantizing 64-bit floating-point numbers into 32-bit floats. Here is a C program that will quantize the example PMX data (but cannot handle SCORE text objects):

floatize.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

void printFloatLine(char* buffer) {
   float value;
```

```
   char* ptr;
   int count = 0;
   ptr = strtok(buffer, " \n\t");
   while (ptr != NULL) {
      if (!sscanf(ptr, "%f", &value)) {
         printf("ERROR\n");
         exit(1);
      }
      if (count++ > 0) {
         printf(" ");
      }
      printf("%f", value);
      ptr = strtok(NULL, " \n\t");
   }
   printf("\n");
}

int main(int argc, char** argv) {
   if (argc < 2) {
      return 1;
   }
   FILE *input;
   input = fopen(argv[1], "r");
   char buffer[1024] = {0};
   while (fgets(buffer, 1024, input) != NULL) {
      printFloatLine(buffer);
   }

   fclose(input);
}
```

Below is sample input and output from *floatize*. Note that the first line of the input data ends in "5" while the float-quantized output's first line ends in "3".

```
8 11 6.198888 9.573973 0.283635 168.128405
8 11 2.873254 -4.797876 2.927321 70.601575
8 7 87.639369 7.994754 0.598376 165.502491
8 7 41.646249 2.260639 0.923335 81.354952
8 12 0.694142 8.659682 1.671497 9.091288
8 7 141.401669 3.957960 1.272665 156.293134
8 6 56.481524 4.920927 0.341730 167.857785
8 10 72.348165 0.396136 1.237993 160.695259
8 6 33.471906 2.824829 2.398684 35.839999
8 1 8.250568 7.964336 0.584617 174.530481
8 12 32.757540 9.446832 2.429939 87.284526
8 10 162.437601 -6.536518 2.742277 170.270498
8 8 27.373900 -3.664126 1.156347 36.968902
8 9 96.360833 2.974862 2.310525 152.645578
8 9 108.015245 -4.306424 2.667651 113.962831
8 3 55.261755 -6.727791 0.970704 190.964604
8 1 52.988382 6.902256 1.430236 97.383340
8 2 2.663138 2.782467 0.111959 147.923603
```

```
8.0 11.0 6.198888 9.573973 0.283635 168.128403
8.0 11.0 2.873254 -4.797876 2.927321 70.601578
8.0 7.0 87.639366 7.994754 0.598376 165.502487
8.0 7.0 41.646248 2.260639 0.923335 81.354950
8.0 12.0 0.694142 8.659682 1.671497 9.091288
8.0 7.0 141.401672 3.957960 1.272665 156.293137
8.0 6.0 56.481525 4.920927 0.341730 167.857788
8.0 10.0 72.348167 0.396136 1.237993 160.695251
8.0 6.0 33.471905 2.824829 2.398684 35.840000
8.0 1.0 8.250568 7.964336 0.584617 174.530487
8.0 12.0 32.757542 9.446832 2.429939 87.284523
8.0 10.0 162.437607 -6.536518 2.742277 170.270493
8.0 8.0 27.373899 -3.664126 1.156347 36.968903
8.0 9.0 96.360832 2.974862 2.310525 152.645584
8.0 9.0 108.015244 -4.306424 2.667651 113.962830
8.0 3.0 55.261757 -6.727791 0.970704 190.964600
8.0 1.0 52.988380 6.902256 1.430236 97.383339
8.0 2.0 2.663138 2.782467 0.111959 147.923599
```

To compare the finale EPS output from both SCORE and scrstaffq, the following PERL script was used:

epscompare.pl

```
#!/usr/bin/perl
use strict;

my $epsfile1  = $ARGV[0];
my $epsfile2  = $ARGV[1];
my @epshpos1  = getEpsCoordinates($epsfile1);
my @epshpos2  = getEpsCoordinates($epsfile2);
my $size1     = @epshpos1;
```

```
my $size2     = @epshpos2;
my @coord1;
my @coord2;
my @diff;

die "Coordinate count mismatch: $size1 $size2\n" if $size1 != $size2;

for (my $i=0; $i<$size1; $i++) {
   @coord1  = split(/\s+/, $epshpos1[$i]);
   @coord2  = split(/\s+/, $epshpos2[$i]);
   $diff[0] = $coord1[0] - $coord2[0];         # horizontal difference
   $diff[1] = $coord1[1] - $coord2[1];         # vertical difference
   if (abs($diff[0]) + abs($diff[1]) != 0.0) { # print only if different
      my $line = int(($i % 10) / 2.0)+1;
      my $staff = int($i / 10);
      my $bin = $i % 2;
      print "$staff: $line-$bin: ";
      print "($diff[0], $diff[1])";
      print " =\t($coord1[0], $coord1[1])$coord1[2] -\t";
      print "($coord2[0], $coord2[1])$coord2[2]\n";
   }
}

exit(0);


##############################
##
## getEpsCoordinates -- extract all moveto and lineto coordinates from a
##    SCORE EPS file.
##

sub getEpsCoordinates {
   my ($file) = @_;
   my @output;
   open (FILE, $file) or die "Cannot read $file";
   while (my $line = <FILE>) {
      chomp $line;
      if ($line =~ /^\s*([-\d\.]+)\s+([-\d\.]+)\s+m\s*$/) {
         $output[@output] = "$1\t$2\tm";
      } elsif ($line =~ /^\s*([-\d\.]+)\s+([-\d\.]+)\s+l\s*$/) {
         $output[@output] = "$1\t$2\tl";
      }
   }
   close FILE;
   return @output;
}
```

The *epscompare* script reports two quantization problems for the set of 1000 random staves (staff #592, vertical position of 4th staff line).

```
592: 4-0: (0, 1) =      (20232, -20782)m -      (20232, -20783)m
592: 4-1: (0, 1) =      (29336, -20782)l -      (29336, -20783)l
```

This quantization error could not be compensated for without causing quantization errors on other staves. It is most likely cause by difference in quantization methods for the vertical spacing between SCORE and *scrstaffq*.

9 Strokeadjust quantization effects

Section 8 covers the 4000-DPI internal quantization of SCORE EPS coordinates caused by scaling the coordinate system by 0.018 and rounding coordinates of all points to integers. An important secondary quantization effect takes place when the `STROKEADJUST` setting in the SCORE print menu is set to "`Yes`". By default the stroke-adjust setting is turned on, and this will cause the following PostScript function to be inserted into the EPS output data.

```
/setstrokeadjust where
     { pop true setstrokeadjust }
     { /m {
             transform
             .25 sub round .25 add exch
             .25 sub round .25 add exch
             itransform  moveto
          } bind def
       /l {
             transform
             .25 sub round .25 add exch
             .25 sub round .25 add exch
             itransform  lineto
          } bind def
     } ifelse
```

This code imposes a quantization spacing for all coordinate points used by the `m` and `l` functions. The coordinate values are shifted ¼ of a pixel down, then rounded to the nearest pixel, then back by shifting up ¼ of a pixel:

x y moveto => [int(device(x, dpi)-0.25) + 0.25] [int(device(y,dpi)-0.25)+0.25] moveto

The purpose of this code is to create staff lines (and other lines) with uniform thicknesses. If this code is not used, any line placed at a pixel boundary in the rendering device will have a width that is an even number of pixels, while lines placed at ½ pixels will have a width that is an odd number of pixels. Quantizing coordinates to the ¼ pixel positions allows for pixelated lines to have the full range of even and odd widths. Alternatively ¾ pixel positions could also be used (quantize to the renderer's ½ pixel resolution, then shift ¼ pixel). Here is the Adobe explanation for setstrokeadjust:

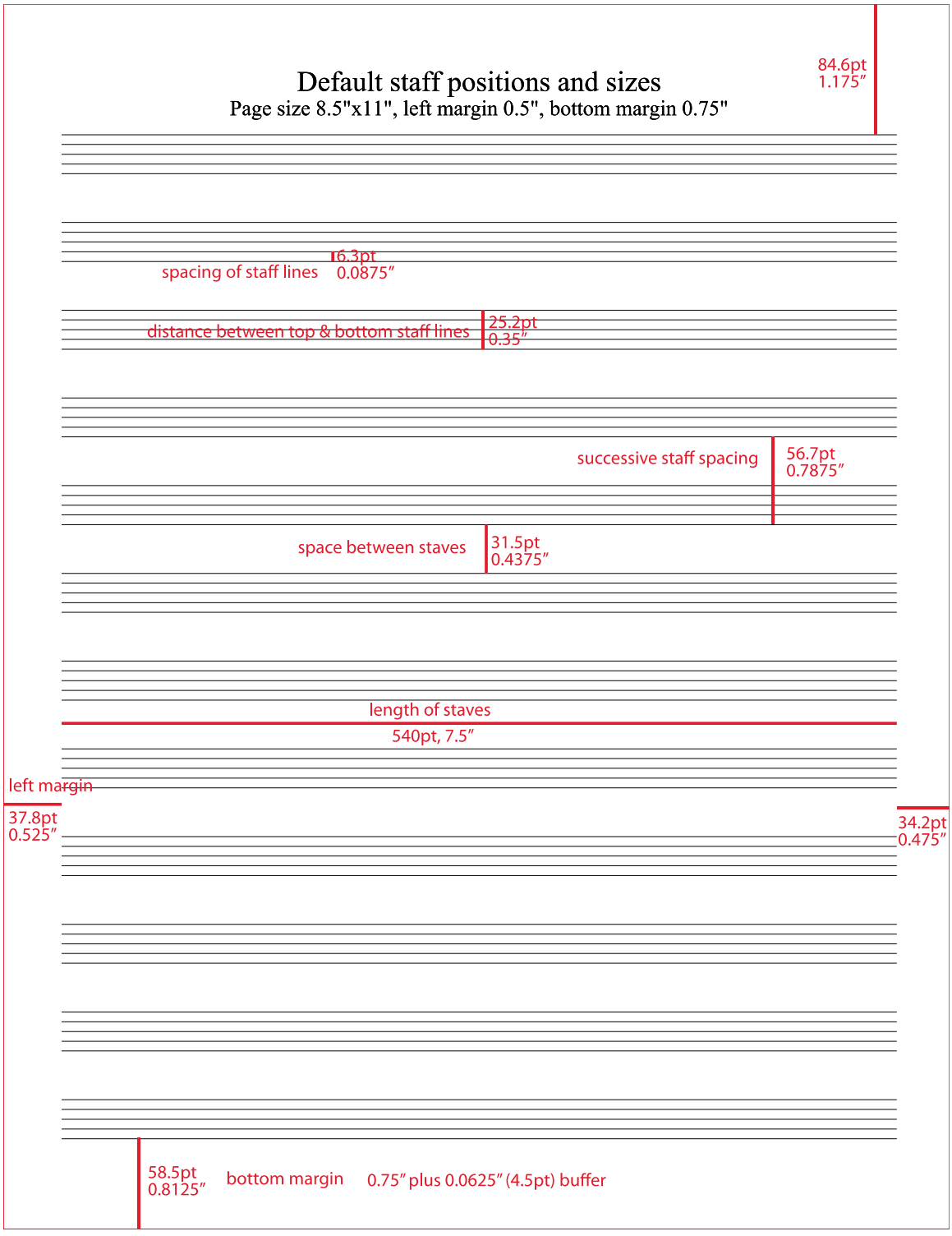
> "Why adjust to one quarter? Placing the path off center within the pixel makes it possible for the device space line width to grow one pixel at a time as the specified line width increases. Placing the path along the pixel boundary forces all line widths to use an even number of pixels and to grow two pixels at a time. Centering the path within the pixel forces all line widths to use an odd number of pixels and also to grow two pixels at a time."[6]

[6] http://partners.adobe.com/public/developer/en/ps/sdk/5111.Stroke_Adj.pdf, "Emulation of the setstrokeadjust Operator" Technical Note #5111, 31 March 1992, Adobe Systems, San Jose, California.

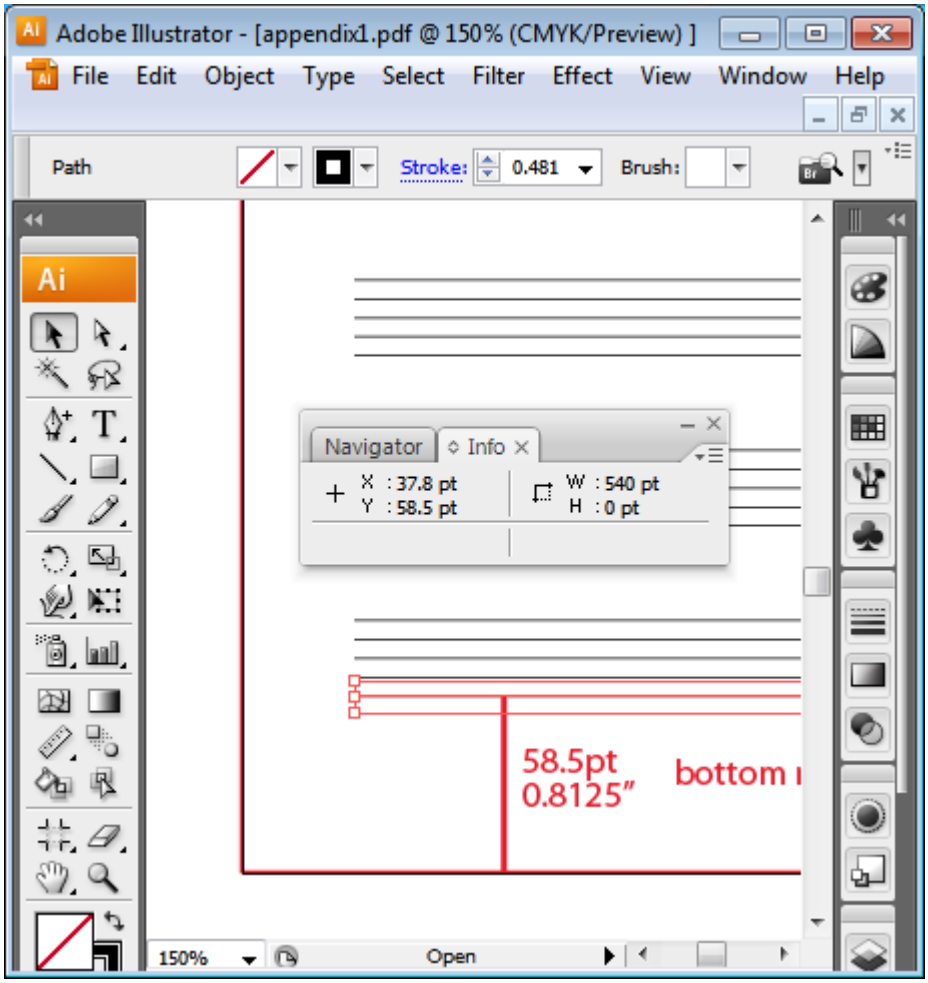
The program *scrstaffq* described in the previous section will match the output from SCORE when comparing with 600 DPI bitmaps and when the strokeadjust setting is turned off. If the strokeadjust setting in SCORE is turned on, the pixel behavior will be slightly different. It is preferable to turn on the strokeadjust setting in SCORE printing (it is turned on by default). The following program (called *scrstaffqq*) shows the final staff printing emulation program. This program calculates the continuous position of staves, then it applies a 4000 DPI quantization to match SCORE's printing behavior, then it applies a 600 DPI (or any arbitrary rendering resolution) quantization to match the behavior of the strokeadjust function usually included in SCORE EPS output.

Appendix I

The following figure shows measurements taken on a page with staves in their default positions and at their default size. See Figure 2 for the SCORE PMX data used to create this page.



The above sample page was measured and marked up in Adobe Illustrator:



To measure in Adobe Illustrator, select the measure tool which looks like a ruler rotated 45° from horizontal (the 10th tool down in the toolbar at the left in the above display). This will cause the measuring window to be displayed (small window shown in center of above figure). For example, the bottom staff has been selected, and the measure window states that the left side of the staff line is at horizontal position 37.8pt and vertical position 58.5pt on the page (in reference to the bottom left corner of the page. Likewise, the width of the staff line is 540pt, and the height is 0pt. The visual height of the line is controlled by the stroked line at its edges. In this case the stroke width is 0.481pt (actually 0.4807206 in the EPS code), which is visible at the top center of the main window.

Appendix II

The following PMX data for 1000 randomly generated staves was used to evaluate 4000-DPI quantization effects in Section 8.1.

8.0 11.0 6.198888 9.573973 0.283635 168.128403
8.0 7.0 41.646248 2.260639 0.923335 81.354950
8.0 6.0 56.481525 4.920927 0.341730 167.857788
8.0 1.0 8.250568 7.964336 0.584617 174.530487
8.0 8.0 27.373899 -3.664126 1.156347 36.968903
8.0 3.0 55.261757 -6.727791 0.970704 190.964600
8.0 4.0 139.913071 -3.690463 0.956186 143.734268
8.0 4.0 36.023048 -7.778810 0.266028 152.194443
8.0 1.0 49.466782 -8.467895 2.451192 57.357403
8.0 1.0 147.526260 -3.608124 2.806991 168.656876
8.0 3.0 40.992905 -2.417932 1.146453 166.101425
8.0 9.0 142.804077 0.054899 0.616394 175.324585
8.0 12.0 6.951823 8.712811 0.128427 181.867584
8.0 1.0 66.063492 5.738389 1.363535 157.752777
8.0 4.0 87.123947 -5.208183 0.467257 191.443710
8.0 3.0 6.238120 -2.370818 0.889234 72.832794
8.0 1.0 168.429443 6.261368 1.828918 190.589752
8.0 11.0 123.745979 6.332990 1.516397 198.792435
8.0 9.0 15.821316 -1.638870 1.455516 81.945145
8.0 7.0 33.353012 1.396795 2.534305 88.232147
8.0 11.0 18.498833 9.255879 2.704088 82.977425
8.0 3.0 3.903445 0.102748 0.305033 78.770164
8.0 5.0 25.594097 6.803292 2.698321 192.119293
8.0 3.0 91.754387 7.412782 2.697732 96.293327
8.0 8.0 23.666191 7.623250 2.462144 122.595360
8.0 12.0 51.833282 -0.124504 2.005128 182.354935
8.0 12.0 165.984650 6.040213 0.100030 170.573761
8.0 9.0 173.203766 -0.466391 2.549927 175.192688
8.0 1.0 24.353874 -0.371537 2.921825 196.329666
8.0 9.0 41.823765 0.663665 1.005246 107.636658
8.0 6.0 8.426390 1.355992 0.594715 121.191467
8.0 1.0 4.868407 8.722693 2.072404 83.056061
8.0 8.0 119.646637 1.135458 1.819793 123.064629
8.0 4.0 11.996562 8.099634 1.571574 85.321175
8.0 3.0 184.882019 8.974306 0.973613 196.811508
8.0 10.0 8.789088 -0.766710 0.924580 55.033581
8.0 6.0 132.828613 1.399292 2.912798 139.956406
8.0 3.0 53.107815 8.433582 0.562963 140.615540
8.0 6.0 97.156876 -5.459025 2.162644 115.181114
8.0 6.0 45.175087 3.615963 2.401092 97.849190
8.0 6.0 42.426208 -2.156638 2.967922 108.140968
8.0 8.0 8.587366 -4.131536 1.283458 58.713299
8.0 11.0 53.731918 3.716945 2.804798 81.564621
8.0 2.0 169.364014 9.350571 2.809417 178.406784
8.0 6.0 80.697533 3.719324 2.038636 179.228180
8.0 10.0 15.067068 -6.294972 2.625801 74.518311
8.0 6.0 46.475506 -8.635746 0.441633 50.059490
8.0 2.0 48.885788 9.670249 2.142379 99.150841
8.0 3.0 11.875489 -6.942601 1.804323 118.476990
8.0 4.0 13.197751 -6.990136 1.302105 173.773148
8.0 8.0 68.472229 9.598122 1.215920 144.504181
8.0 8.0 60.340813 -8.747022 2.502209 78.289024
8.0 12.0 125.369873 -7.855205 2.107706 143.773804
8.0 3.0 40.404610 3.195834 1.002441 166.094330
8.0 10.0 71.628502 -1.119869 0.055156 115.864403
8.0 5.0 63.050636 -4.481711 1.429135 153.737335
8.0 9.0 161.249527 2.111606 0.387793 169.057831
8.0 12.0 86.988899 4.598871 0.134817 172.939713
8.0 2.0 91.886932 -8.624597 0.668231 195.364014
8.0 8.0 106.620407 9.083251 1.190930 126.399506
8.0 4.0 46.638554 -4.814556 1.283052 52.222378
8.0 5.0 89.877396 5.117635 1.137677 195.250183
8.0 11.0 2.873254 -4.797876 2.927321 70.601578
8.0 12.0 0.694142 8.659682 1.671497 9.091288
8.0 10.0 72.348167 0.396136 1.237993 160.695251
8.0 12.0 32.757542 9.446832 2.429939 87.284523
8.0 9.0 96.360832 2.974862 2.310525 152.645584
8.0 1.0 52.988380 6.902256 1.430236 97.383339
8.0 8.0 37.481258 -9.526205 0.102742 108.354874
8.0 8.0 122.386299 8.136932 2.105072 176.946503
8.0 10.0 161.137756 7.532659 2.825202 165.729355
8.0 4.0 114.831512 5.774232 2.960993 142.368820
8.0 6.0 54.388432 6.239888 2.155237 155.652725
8.0 2.0 17.244064 -6.116035 1.885361 103.534569
8.0 3.0 20.031265 8.174537 2.115587 196.422897
8.0 7.0 9.377641 0.306433 2.541582 86.846817
8.0 11.0 2.331089 -7.717156 2.989127 69.230522
8.0 7.0 22.225761 -3.726212 1.593250 33.698196
8.0 7.0 84.881432 -5.191155 1.373822 86.943558
8.0 2.0 17.034014 -9.663969 0.265752 55.238831
8.0 5.0 80.533249 6.816965 2.110899 171.737808
8.0 9.0 119.414444 -0.217567 2.989400 155.924240
8.0 11.0 55.898617 -2.761884 2.443034 132.886612
8.0 12.0 107.808632 -1.094476 1.155876 130.665359
8.0 11.0 134.719528 -3.617043 2.575112 150.647720
8.0 3.0 97.292732 8.969493 0.694807 176.026993
8.0 10.0 91.751389 8.345641 0.366993 161.527908
8.0 10.0 111.864891 9.510755 0.027516 123.199890
8.0 10.0 159.737900 9.172223 0.173098 187.664825
8.0 5.0 70.985161 9.456027 1.716067 134.445587
8.0 5.0 3.691837 3.173079 2.038601 63.263123
8.0 8.0 16.489143 3.788120 1.344147 171.345673
8.0 12.0 6.166741 -8.814771 2.113602 40.555847
8.0 11.0 20.381676 -3.534865 0.535705 170.166199
8.0 10.0 41.834785 4.651078 1.159365 137.842880
8.0 2.0 125.119186 -5.537930 2.574008 144.523880
8.0 1.0 84.038879 0.005561 0.274519 89.176605
8.0 1.0 111.765106 -1.966664 1.681191 144.124893
8.0 6.0 77.514320 0.213602 2.531075 193.389145
8.0 12.0 43.322056 -5.422617 0.277374 53.351299
8.0 4.0 90.943214 -7.384304 1.007765 138.269272
8.0 8.0 47.832199 -4.499476 0.005907 128.902496
8.0 10.0 128.640274 0.617173 0.195471 150.647369
8.0 6.0 14.400546 5.970420 1.837977 103.014969
8.0 1.0 50.878475 0.792366 2.495129 97.902405
8.0 1.0 16.529488 -5.346721 1.949634 116.796783
8.0 6.0 20.152380 -5.888117 1.820123 105.919991
8.0 7.0 50.231308 1.280961 0.761224 60.836155
8.0 7.0 16.468596 9.280887 0.772771 115.074356
8.0 10.0 129.589325 -4.534252 2.844929 178.703278
8.0 9.0 91.449493 3.972556 0.716614 170.817795
8.0 12.0 20.812304 4.200317 1.727013 107.883606
8.0 1.0 34.945992 -3.071671 0.147771 47.790596
8.0 1.0 116.710922 -6.273032 2.401908 175.663422
8.0 11.0 22.705318 5.784137 1.453986 25.568111
8.0 8.0 107.738449 -7.341516 1.366944 194.129501
8.0 3.0 16.624325 2.712113 1.344250 118.385574
8.0 8.0 13.959438 -5.626602 1.543595 135.396484
8.0 11.0 39.713173 -9.538968 2.195183 69.798279
8.0 9.0 21.610228 -4.222854 0.373531 136.007812
8.0 8.0 67.146400 -9.722098 2.752115 189.993912
8.0 4.0 41.500107 2.908332 1.436238 47.443615
8.0 7.0 121.604385 9.628999 2.901616 149.554794
8.0 12.0 83.353165 -4.902263 0.209591 111.489517
8.0 7.0 87.639366 7.994754 0.598376 165.502487
8.0 7.0 141.401672 3.957960 1.272665 156.293137
8.0 6.0 33.471905 2.824829 2.398684 35.840000
8.0 10.0 162.437607 -6.536518 2.742277 170.270493
8.0 9.0 108.015244 -4.306424 2.667651 113.962830
8.0 2.0 2.663138 2.782467 0.111959 147.923599
8.0 7.0 40.561432 2.533573 1.711105 54.101456
8.0 6.0 139.798416 -8.796644 0.437483 182.640045
8.0 5.0 32.158031 -0.263913 2.999005 180.464996
8.0 11.0 107.031319 0.885436 1.729410 123.432526
8.0 4.0 25.159283 -6.164561 1.452633 139.740005
8.0 2.0 124.579742 -9.632725 0.028093 127.571960
8.0 3.0 126.520973 5.826864 1.477694 185.196609
8.0 5.0 35.528816 -7.923759 2.256310 83.712776
8.0 1.0 56.913700 8.659498 2.611049 145.973984
8.0 4.0 125.203056 -8.071726 0.546871 179.361725
8.0 6.0 157.588135 -6.391901 2.693684 192.980133
8.0 7.0 151.902786 -1.313248 2.683772 190.401306
8.0 9.0 40.627911 5.630449 2.350412 42.394787
8.0 5.0 86.503304 5.764833 0.512668 184.011749
8.0 5.0 6.592648 -2.193129 0.552463 133.957123
8.0 9.0 42.917736 -7.188578 2.150805 191.198090
8.0 8.0 83.498772 9.371502 2.805825 179.685501
8.0 4.0 38.387726 4.658899 2.709346 71.430374
8.0 4.0 82.341675 0.663294 1.571000 142.436035
8.0 12.0 6.728599 0.440169 0.666060 64.271217
8.0 2.0 11.939249 9.954610 0.495088 78.716515
8.0 5.0 132.236481 9.601337 0.323410 161.955994
8.0 3.0 78.907532 -9.183715 1.735404 131.235382
8.0 10.0 100.851784 7.122757 1.623421 195.231003
8.0 1.0 20.990101 -9.971948 2.590970 52.971676
8.0 10.0 116.046921 -9.111825 0.474188 160.109634
8.0 8.0 127.220459 -2.294082 2.515220 179.765976
8.0 8.0 23.918112 4.642719 2.977477 26.818214
8.0 2.0 137.074112 0.754651 2.940141 156.289429
8.0 11.0 18.165243 8.369194 1.534829 170.121429
8.0 1.0 75.870537 -9.107709 0.541848 192.892517
8.0 5.0 26.022585 -9.664699 1.454841 87.831917
8.0 4.0 66.256104 -3.249930 2.855365 192.081223
8.0 6.0 26.134062 3.439064 0.453696 142.655197
8.0 10.0 34.437576 4.963188 0.193202 120.222527
8.0 12.0 50.295952 2.145760 1.799494 190.561462
8.0 11.0 87.524948 -7.213889 1.732291 90.031349
8.0 3.0 68.184914 -4.102852 0.691362 85.281731
8.0 5.0 78.136879 0.816708 0.714503 101.095810
8.0 7.0 109.969177 -1.532546 1.409007 190.541656
8.0 1.0 131.153076 -4.877442 0.914802 152.590378
8.0 9.0 61.329205 -8.710005 1.426371 87.251518
8.0 9.0 104.029221 -0.591822 0.330169 186.825409
8.0 12.0 169.671570 -0.413325 1.703140 178.188110
8.0 5.0 186.368530 -7.105634 0.245545 195.193771
8.0 4.0 29.519829 -5.877676 1.032047 108.306084
8.0 12.0 109.068443 -5.671877 2.551476 133.522888
8.0 11.0 95.643837 1.744007 0.082317 174.756622
8.0 1.0 45.434669 -9.112237 0.260902 176.571152
8.0 12.0 2.591644 4.397594 2.556135 67.254662
8.0 8.0 0.002439 -6.947422 0.499436 173.643570
8.0 3.0 57.572063 2.138375 1.051679 175.349030
8.0 10.0 90.257835 6.105478 2.220700 153.076523
8.0 5.0 9.217006 -0.964986 2.254963 57.739231
8.0 3.0 65.438019 5.038711 1.685677 111.299606
8.0 3.0 58.180672 7.575591 2.193330 103.147781

8.0 10.0 19.073053 -8.536414 2.772692 86.789627
8.0 8.0 155.616226 5.491441 1.058782 174.050476
8.0 7.0 49.713753 0.864830 2.406403 120.020988
8.0 7.0 35.951218 -2.790097 0.116173 126.301155
8.0 9.0 32.593300 -2.510605 2.349072 97.280434
8.0 8.0 102.591522 -5.193213 2.003271 117.684578
8.0 7.0 87.906624 6.660782 2.384040 139.049133
8.0 4.0 10.217552 -6.343678 0.524038 171.939590
8.0 11.0 26.610806 -7.088907 2.925656 196.480423
8.0 1.0 90.470123 -7.575600 0.731878 126.295433
8.0 8.0 28.638708 -8.892422 1.746098 181.803558
8.0 10.0 79.555542 0.840379 0.222761 140.417587
8.0 2.0 108.479980 -1.260164 2.169207 127.066536
8.0 1.0 58.074249 3.043499 0.211461 147.022919
8.0 8.0 17.551544 -9.673802 1.778907 137.593719
8.0 12.0 116.684860 6.872620 1.318086 191.387589
8.0 12.0 19.567379 -4.218198 2.085061 106.599472
8.0 11.0 48.273045 1.911502 2.196523 186.390732
8.0 3.0 11.319019 0.353892 2.944220 180.901474
8.0 1.0 35.886246 -6.585683 1.660847 140.264328
8.0 6.0 106.595322 -8.641242 2.807013 122.826546
8.0 2.0 17.068535 2.109258 0.753339 141.830200
8.0 1.0 8.518073 8.484611 1.966735 28.936291
8.0 4.0 80.160538 -1.074503 2.655068 154.508545
8.0 6.0 56.694054 5.988776 1.130470 90.790070
8.0 7.0 79.559090 -5.081699 2.533234 108.571602
8.0 12.0 12.430821 -8.311004 2.042010 100.163246
8.0 4.0 48.880157 6.356507 2.532656 131.496246
8.0 6.0 12.309075 9.492307 2.160990 30.913565
8.0 6.0 12.155829 8.492864 1.823342 27.864023
8.0 11.0 38.050209 -1.571744 1.255017 98.043274
8.0 12.0 87.681305 8.790649 2.412442 126.623856
8.0 1.0 82.277954 0.122550 1.717702 177.025604
8.0 7.0 13.075804 -1.359603 1.173347 40.816235
8.0 6.0 25.407930 2.409933 0.985896 172.471619
8.0 7.0 166.277802 5.684893 1.284429 188.735931
8.0 6.0 18.072041 -4.987008 2.653144 71.654823
8.0 2.0 49.622311 2.170296 0.161153 185.418900
8.0 4.0 100.133850 9.559130 2.652844 145.981262
8.0 12.0 66.858948 3.290832 1.460887 163.772446
8.0 10.0 70.381592 -5.663016 2.186963 188.071152
8.0 3.0 18.129599 -2.135259 0.383626 63.829628
8.0 9.0 16.738810 -9.862246 0.269390 125.141739
8.0 8.0 60.296165 3.400823 2.568127 166.169754
8.0 1.0 26.935295 -2.630755 0.563065 145.853134
8.0 4.0 54.442177 -0.291263 1.820974 143.640610
8.0 7.0 4.943226 9.915509 0.219852 185.822067
8.0 2.0 124.953903 2.274350 0.760724 165.818985
8.0 5.0 23.440071 6.850019 0.143474 32.148045
8.0 1.0 80.031738 0.826973 2.870690 100.849937
8.0 7.0 134.358109 1.896265 1.955609 145.989395
8.0 7.0 103.258675 -1.336132 1.608597 152.994232
8.0 1.0 56.863239 2.839393 0.966109 183.122665
8.0 2.0 15.640855 -8.961320 1.640952 186.416885
8.0 5.0 69.920395 -6.414148 1.637433 142.573380
8.0 11.0 50.738560 -1.827479 1.012292 120.931580
8.0 4.0 42.231506 -8.790400 1.289371 192.276260
8.0 3.0 80.122177 -5.328413 2.860989 156.519135
8.0 7.0 30.652641 -4.656266 1.574996 192.973907
8.0 11.0 120.211372 -2.680075 1.159252 145.445587
8.0 4.0 23.864252 1.092225 1.898871 93.549301
8.0 9.0 93.426765 4.073196 0.350038 172.042633
8.0 5.0 0.103934 -9.879662 1.701350 170.028763
8.0 9.0 168.649902 0.936503 1.950627 179.871094
8.0 9.0 64.231064 4.447705 2.588568 196.447067
8.0 10.0 88.236755 6.110023 0.495481 182.531082
8.0 12.0 14.123473 -8.089642 1.653474 72.594971
8.0 10.0 133.308716 -8.191624 0.651754 185.875977
8.0 2.0 77.362747 7.366564 0.081220 183.687958
8.0 3.0 42.095062 9.029211 0.248598 138.351471
8.0 12.0 62.372597 4.030689 1.662204 152.101181
8.0 7.0 102.925789 0.170836 0.226196 163.459564
8.0 12.0 43.387093 8.761097 2.701263 190.504456
8.0 7.0 42.888153 -2.786601 0.114431 63.176968
8.0 5.0 98.815552 4.525210 1.801789 176.811417
8.0 2.0 18.342968 7.638640 1.472514 190.193863
8.0 11.0 42.049610 -0.425631 2.044714 49.356045
8.0 3.0 127.294510 2.806988 2.373009 184.486130
8.0 3.0 15.414627 3.083256 2.419085 40.953606
8.0 6.0 70.444962 9.572105 1.570834 107.338356
8.0 2.0 54.866516 -6.912787 0.342938 141.749222
8.0 11.0 84.965439 6.478159 2.076344 100.362663
8.0 9.0 143.558502 7.120744 0.080074 163.767929
8.0 6.0 91.991142 -2.772331 2.571283 136.884842
8.0 8.0 8.885323 7.022020 0.417309 141.191086
8.0 1.0 78.224213 3.822393 1.835477 110.506142
8.0 7.0 132.476227 5.178447 1.351940 163.778580
8.0 5.0 126.053551 -4.875822 1.375518 145.730804
8.0 5.0 30.183990 -8.285598 0.139426 180.862030
8.0 11.0 80.485229 -5.867422 2.674936 169.267685
8.0 8.0 97.095596 0.386696 1.161423 176.138550
8.0 6.0 61.169224 -5.593039 0.483856 128.582916
8.0 11.0 23.765106 5.347541 2.703864 100.731224
8.0 6.0 63.067642 6.895772 0.510680 91.276192
8.0 12.0 51.433426 -5.986856 2.481012 108.487038
8.0 9.0 15.467099 -8.367566 0.583645 105.339539
8.0 10.0 150.925797 -2.656467 2.185254 177.492203
8.0 9.0 66.374023 -8.042041 0.175666 138.309982
8.0 11.0 12.201890 -2.511644 1.649109 37.258499
8.0 1.0 23.847221 -0.861766 1.608767 95.332420
8.0 8.0 45.621738 5.476515 0.771606 127.701584
8.0 1.0 24.010956 -8.230528 2.685645 106.583122
8.0 12.0 47.408829 7.208623 1.250868 128.589859
8.0 5.0 72.854996 -4.644601 2.444797 105.017586
8.0 8.0 34.442215 -7.724564 0.743916 140.076340
8.0 7.0 130.271194 -9.617993 2.708467 171.286819
8.0 3.0 0.232733 -3.501532 1.643674 101.786530
8.0 1.0 85.320320 -1.817302 2.645279 118.293137
8.0 5.0 76.135880 -3.703938 1.058617 82.229424
8.0 9.0 85.123871 -0.338557 2.710533 174.807205
8.0 10.0 94.632973 3.083891 0.615113 129.946640
8.0 3.0 19.480112 -6.461064 2.005350 41.147964
8.0 11.0 43.539463 -2.212245 2.971004 149.946533
8.0 11.0 31.431393 -3.737312 1.780208 74.617157
8.0 1.0 30.831003 -2.323967 1.912418 89.721092
8.0 7.0 1.827714 4.534658 0.206286 89.329239
8.0 7.0 72.854507 -7.725797 0.466421 187.194550
8.0 5.0 12.744419 -8.024952 0.274674 54.493782
8.0 5.0 20.824875 -6.124417 1.721935 174.433990
8.0 2.0 119.720131 7.895227 0.298176 191.192245
8.0 8.0 23.593691 2.909264 1.309563 130.912659
8.0 3.0 25.969736 9.718051 2.386417 34.740021
8.0 7.0 39.531319 -4.649463 0.237004 144.083557
8.0 8.0 16.602167 -4.704538 2.280899 63.758347
8.0 8.0 137.294830 -1.352248 2.014158 172.012360
8.0 12.0 16.164261 -6.685463 0.988954 115.583374
8.0 2.0 56.032818 1.570793 2.005207 93.526337
8.0 11.0 82.928459 3.542064 2.431794 179.823395
8.0 9.0 86.230919 2.962104 2.224446 103.058617
8.0 8.0 102.742668 -7.305727 2.287588 158.870712
8.0 3.0 66.691414 -3.552878 2.454364 99.201988
8.0 5.0 100.362274 -7.088506 0.221201 173.020584
8.0 4.0 29.741161 -8.097264 1.401117 147.267563
8.0 9.0 101.111290 -0.171468 0.286689 172.259033
8.0 3.0 41.022381 -2.148752 2.164573 75.620796
8.0 12.0 72.787384 4.703096 0.266624 186.285812
8.0 1.0 100.625084 2.793222 1.847730 152.192657
8.0 11.0 2.758569 2.843094 2.885543 126.262199
8.0 7.0 39.229359 -7.292233 1.775392 48.714867
8.0 8.0 57.165535 2.922584 1.868043 103.793297
8.0 4.0 2.849893 4.501800 0.378617 76.692558
8.0 7.0 18.811386 -4.799361 0.244731 74.872505
8.0 12.0 0.791212 8.125234 2.422788 37.017357
8.0 4.0 20.780622 7.011534 1.617229 169.839996
8.0 4.0 24.755211 -0.146110 2.204085 152.146820
8.0 11.0 3.706129 -6.431927 0.775581 80.730522
8.0 8.0 1.156159 -3.526417 1.669290 30.598566
8.0 11.0 30.548683 -7.738230 0.777604 176.005524
8.0 12.0 115.762146 8.555594 1.343168 147.119980
8.0 10.0 68.674385 9.758743 2.641241 99.546616
8.0 5.0 79.658730 -4.474988 0.906208 175.433136
8.0 3.0 140.625870 -2.867226 2.340339 149.739197
8.0 8.0 13.474706 7.740390 0.965016 183.003494
8.0 1.0 81.642052 0.159782 1.080603 134.779785
8.0 10.0 95.374733 -2.439326 2.048009 173.167862
8.0 6.0 13.225158 5.850715 2.876324 180.007950
8.0 6.0 105.784767 1.064916 1.871115 136.961670
8.0 10.0 49.425297 -7.876217 2.370777 55.465866
8.0 2.0 47.063717 9.284201 2.819768 138.554367
8.0 10.0 8.861547 1.785305 2.551129 181.536819
8.0 7.0 32.387238 9.155203 0.263475 104.038300
8.0 5.0 37.982262 1.051118 2.957610 70.016029
8.0 10.0 84.792435 7.684317 2.597097 150.009964
8.0 10.0 4.715578 9.139266 1.712642 63.828957
8.0 5.0 28.027681 -1.486840 1.870408 112.459518
8.0 9.0 19.584618 -9.438375 0.613720 150.503174
8.0 2.0 94.262573 8.001296 2.970977 187.331146
8.0 12.0 43.224537 7.736801 0.141553 121.904007
8.0 8.0 108.317413 -7.082566 0.037731 198.767960
8.0 11.0 131.736191 -5.914645 1.090767 193.724915
8.0 7.0 67.427811 -2.924162 0.034823 74.912643
8.0 2.0 21.054779 5.722900 2.036193 70.803566
8.0 3.0 38.520512 -9.715551 0.310604 145.196945
8.0 5.0 125.651314 -6.363761 2.267466 188.626663
8.0 10.0 124.946617 4.299715 1.317535 130.754898
8.0 3.0 152.810196 0.492560 1.130445 177.056061
8.0 7.0 2.554950 1.773529 2.686136 78.213120
8.0 5.0 82.910973 -4.140019 2.754076 94.737770
8.0 9.0 19.629000 -8.150378 0.828679 162.451874
8.0 1.0 2.018962 5.305942 1.283989 178.882263
8.0 7.0 15.622020 0.753544 2.417074 147.899384
8.0 6.0 71.748482 -2.062187 2.554484 104.144539
8.0 7.0 66.600296 5.341584 2.718371 173.609711
8.0 9.0 18.228680 -9.199691 2.796436 184.700150
8.0 1.0 39.333649 0.163806 1.628060 179.667862
8.0 12.0 114.674927 -7.462067 2.106096 166.282013
8.0 6.0 42.046913 -1.596706 1.665813 166.050919
8.0 12.0 127.629700 -2.841869 0.454957 134.363983
8.0 1.0 31.798223 -3.197064 2.781870 141.877197
8.0 4.0 90.660660 -2.618990 0.103574 134.386642
8.0 8.0 11.447336 1.273456 0.331029 27.562880
8.0 3.0 76.747192 -3.603732 1.068472 136.675232
8.0 2.0 0.376517 -9.892484 2.745925 31.053766
8.0 9.0 182.498871 -8.542801 0.893802 194.139481
8.0 4.0 37.873837 9.185788 0.771434 41.526646
8.0 8.0 138.177902 4.654421 2.125635 170.859985
8.0 11.0 35.518917 -3.083571 2.979063 187.340073
8.0 6.0 78.258217 6.519065 0.225542 174.112457
8.0 1.0 83.317261 4.203425 2.828902 130.143646
8.0 4.0 51.510681 3.220374 1.159553 92.364441
8.0 5.0 33.613110 -0.626684 0.633538 64.942703
8.0 8.0 37.437962 3.884839 1.208170 168.039688
8.0 8.0 108.633842 0.997254 2.527468 159.491486
8.0 6.0 4.486347 -6.739302 1.846068 138.559570
8.0 1.0 79.239861 3.252391 2.673712 104.641800
8.0 10.0 115.192413 7.213905 0.830100 145.782272
8.0 7.0 103.742470 -2.093201 1.833476 140.353119
8.0 5.0 7.764113 3.941867 0.506473 182.488159
8.0 12.0 107.008263 2.293412 1.805421 187.795959
8.0 11.0 9.569684 -9.156461 1.300314 144.347473

8.0 6.0 20.970785 -6.445845 2.773738 79.830612
8.0 4.0 74.037804 -7.501632 1.787903 86.962265
8.0 7.0 81.471878 -6.906965 2.505467 112.837082
8.0 3.0 149.690353 2.735057 2.642796 161.922516
8.0 9.0 94.981903 9.035721 1.793351 140.284042
8.0 9.0 72.139023 1.306074 1.198920 143.937027
8.0 12.0 131.178574 -9.132643 2.886434 185.388748
8.0 10.0 2.311179 -8.643561 2.679339 140.011627
8.0 11.0 135.283936 -7.964407 0.977617 180.356918
8.0 12.0 85.057129 2.420851 1.751415 128.108826
8.0 4.0 45.964485 -9.264365 0.873804 56.018337
8.0 7.0 143.953064 9.688150 1.280566 189.627151
8.0 5.0 48.380596 3.016836 1.302457 159.403656
8.0 4.0 139.326385 -5.827009 2.695003 196.249557
8.0 12.0 135.861847 -5.864012 0.269570 163.483612
8.0 11.0 62.145256 8.605392 2.066681 106.029015
8.0 4.0 73.532333 9.305886 2.068374 97.996758
8.0 1.0 25.022299 -1.705256 2.140466 130.679306
8.0 6.0 61.902554 3.434312 0.923627 100.114960
8.0 5.0 87.615814 -5.010316 1.421782 124.853127
8.0 2.0 38.127113 -4.831174 2.346534 140.619415
8.0 1.0 26.633226 1.346365 0.373763 164.034531
8.0 11.0 22.370785 4.187649 0.038964 66.547768
8.0 7.0 21.670822 2.366665 0.145438 69.668541
8.0 9.0 31.834366 6.406661 1.474334 164.640594
8.0 11.0 26.438004 0.712412 0.832386 137.929031
8.0 2.0 154.326447 -6.269589 0.122323 168.084900
8.0 2.0 163.052795 -5.169689 1.171386 187.892868
8.0 10.0 49.402405 -7.933344 1.485621 144.530731
8.0 8.0 115.079376 0.287240 0.832130 151.373962
8.0 7.0 79.465546 -5.881222 2.186860 197.486176
8.0 5.0 11.541221 1.589485 0.123722 19.513119
8.0 4.0 150.151642 -4.005259 1.962366 194.283508
8.0 6.0 99.218674 3.290474 0.705951 107.059425
8.0 12.0 142.284637 0.379161 1.769781 154.108734
8.0 6.0 111.031616 7.484904 0.888005 135.489578
8.0 2.0 53.576492 5.577767 2.247183 185.173691
8.0 2.0 28.433941 4.415303 2.706054 124.913521
8.0 7.0 65.709412 -6.327274 1.982421 149.599319
8.0 4.0 88.502678 8.138285 2.930236 172.444412
8.0 6.0 63.515141 -7.477004 0.754614 152.330643
8.0 11.0 27.331301 3.234173 2.019844 148.528427
8.0 5.0 98.047424 9.132443 0.728208 171.224899
8.0 1.0 53.003716 1.248905 0.596591 152.262802
8.0 11.0 80.885895 -9.788152 0.086955 111.133896
8.0 6.0 31.810196 -7.325034 0.895021 199.929749
8.0 12.0 70.990074 8.911957 2.173972 109.682800
8.0 7.0 50.385811 -0.628154 0.382668 86.855362
8.0 1.0 131.663986 -0.700216 1.698793 171.543060
8.0 5.0 185.638809 7.330552 1.384373 199.417053
8.0 6.0 2.955523 0.004500 0.263826 138.494644
8.0 10.0 11.472920 -2.089885 2.749913 36.389610
8.0 3.0 75.306183 4.005403 1.692989 196.735107
8.0 12.0 75.232483 -2.042333 0.472990 148.529953
8.0 4.0 127.515114 -5.319727 1.947101 177.095673
8.0 11.0 94.931702 -4.169656 2.081656 102.199944
8.0 1.0 136.685028 -1.931512 2.277121 169.921143
8.0 5.0 27.060843 -7.972971 2.596490 80.552269
8.0 12.0 99.085167 -1.575229 0.730553 147.052261
8.0 5.0 63.140717 8.757283 1.170221 147.296112
8.0 2.0 38.821125 4.725176 2.099103 59.698704
8.0 4.0 3.541814 1.562403 1.195914 195.739655
8.0 9.0 6.329517 8.948663 2.241318 106.560951
8.0 5.0 112.520279 4.791398 1.083810 187.383118
8.0 8.0 42.197189 -1.747489 1.827818 69.913445
8.0 9.0 10.891364 4.396532 2.271241 94.098129
8.0 9.0 121.636452 -8.470328 2.735002 134.843964
8.0 8.0 132.179733 -5.009893 1.517065 178.821060
8.0 2.0 64.490280 -4.384120 1.490966 108.764900
8.0 5.0 112.893250 -6.291477 1.819762 194.330643
8.0 6.0 22.509535 0.009632 2.295281 70.408493
8.0 3.0 87.090698 -4.279736 0.853327 162.602325
8.0 2.0 8.259247 -0.645727 0.727911 89.461266
8.0 3.0 131.408798 -9.870153 0.509546 195.944534
8.0 8.0 17.320646 -6.457944 1.293761 19.647551
8.0 2.0 1.845281 -5.966280 1.411651 73.332878
8.0 7.0 134.949677 3.058056 2.066512 176.844406
8.0 5.0 165.391403 1.464837 1.527489 192.398254
8.0 3.0 100.040176 -7.011033 0.617689 109.486214
8.0 2.0 17.694695 -0.545289 0.899836 96.497002
8.0 7.0 139.337143 3.538032 0.630762 174.198761
8.0 4.0 34.303619 -5.258511 0.857778 122.541496
8.0 4.0 2.954489 -4.907717 2.870178 13.267231
8.0 9.0 72.910622 -0.546833 1.821372 184.960052
8.0 2.0 45.270184 6.338645 1.911072 55.694321
8.0 11.0 9.155297 -2.255427 1.762394 111.487595
8.0 1.0 85.106407 -0.128859 2.325823 196.853622
8.0 3.0 19.038099 -3.240804 2.605368 83.969345
8.0 4.0 4.710771 8.895531 1.519800 196.099533
8.0 8.0 75.443787 -0.611518 1.279141 139.496277
8.0 10.0 23.094055 9.031259 2.160289 43.683872
8.0 9.0 120.471771 -5.247067 2.878407 135.945129
8.0 6.0 12.007719 2.765085 1.430533 79.215340
8.0 8.0 61.114399 0.547701 2.372818 196.507462
8.0 6.0 59.262203 -7.022418 1.213863 144.264969
8.0 6.0 90.482590 9.955416 0.971781 106.445465
8.0 5.0 167.484985 6.093943 2.291119 180.005798
8.0 12.0 129.835220 7.931376 1.628079 167.815445
8.0 10.0 32.308464 8.991512 1.982445 142.698700
8.0 6.0 75.816032 7.078796 0.406891 141.517334
8.0 4.0 117.389160 -5.000824 0.039843 141.207550
8.0 10.0 36.629505 -6.428488 1.504381 39.106190
8.0 4.0 141.829010 4.688651 1.043582 175.497940
8.0 7.0 56.865044 2.477256 1.155857 90.150047
8.0 10.0 69.433998 -6.177194 1.709043 183.715652
8.0 8.0 130.961533 9.986216 2.565369 198.155930
8.0 4.0 25.490601 6.674485 1.543587 112.788185
8.0 4.0 129.351074 -8.925571 0.591416 160.593262
8.0 7.0 79.227318 6.609559 1.536894 144.690140
8.0 3.0 124.861023 -8.036462 2.545375 190.212448
8.0 9.0 124.387634 5.476968 1.896751 193.313828
8.0 3.0 43.503010 3.620184 2.074513 113.110367
8.0 1.0 130.132111 4.716710 1.817444 179.725204
8.0 2.0 15.899833 8.723133 0.174234 191.567459
8.0 6.0 72.374924 -7.342531 0.696890 188.439774
8.0 4.0 94.207275 -4.921562 2.512296 113.553078
8.0 10.0 199.054489 -3.031211 2.182857 199.350037
8.0 11.0 42.326508 -8.075220 1.351332 139.230621
8.0 3.0 71.575340 -5.026848 1.998382 89.597198
8.0 3.0 15.792538 1.634391 1.714083 163.483994
8.0 5.0 46.226105 2.675094 2.252940 74.112122
8.0 8.0 146.574265 -3.985104 1.746665 192.697006
8.0 7.0 3.338199 1.260460 2.087495 55.988224
8.0 2.0 88.337540 9.738338 2.817350 181.156815
8.0 3.0 157.449646 7.278662 2.718229 195.072281
8.0 10.0 97.597107 -3.417744 1.369984 118.823517
8.0 2.0 141.479813 0.194189 1.399731 183.383453
8.0 4.0 49.424725 -6.784719 2.116729 72.014389
8.0 9.0 65.553902 -6.766178 1.498105 138.649170
8.0 3.0 72.246124 2.017314 1.951028 140.026978
8.0 5.0 33.521721 8.300452 0.125552 84.281715
8.0 4.0 19.870064 7.143879 0.551587 114.262573
8.0 3.0 96.435982 2.435107 1.578799 184.148621
8.0 4.0 95.259277 4.558477 2.429930 133.974487
8.0 9.0 61.243534 -3.197007 2.830927 120.020966
8.0 2.0 134.886353 -5.738704 1.465246 195.576492
8.0 5.0 136.327896 -6.649775 0.572767 153.207443
8.0 5.0 133.750092 -7.076439 1.985607 192.229584
8.0 4.0 126.637260 -0.513525 0.162339 141.292496
8.0 3.0 59.252331 1.336045 2.692563 108.271065
8.0 12.0 31.832249 8.019896 0.768602 93.380264
8.0 6.0 70.801605 -4.871140 1.247726 92.805634
8.0 8.0 183.191498 -9.771203 1.021252 196.307114
8.0 4.0 129.204224 -0.727945 1.187638 155.602921
8.0 6.0 30.380274 7.930118 1.334694 143.536041
8.0 8.0 37.613937 4.624686 0.936728 90.131508
8.0 6.0 17.931763 6.646077 0.157562 197.273773
8.0 9.0 21.531483 9.895826 1.364460 85.660530
8.0 12.0 10.623493 4.688988 0.713845 28.941355
8.0 6.0 2.362376 -6.359510 0.569281 193.441574
8.0 2.0 46.079922 2.156064 0.367941 56.354633
8.0 1.0 62.939243 1.300595 2.527600 71.444130
8.0 1.0 51.422970 3.110980 2.406709 113.748375
8.0 11.0 59.360096 2.725445 1.623426 129.725327
8.0 5.0 53.057777 9.972609 1.012038 185.206955
8.0 9.0 95.156631 -8.896892 1.622749 183.861526
8.0 8.0 116.657944 8.298489 1.729092 117.451767
8.0 5.0 17.551441 0.789143 1.961363 143.139755
8.0 8.0 138.194199 -1.667406 2.883145 189.588211
8.0 3.0 14.782940 8.056971 1.284518 146.334717
8.0 12.0 109.381897 -7.246218 0.216761 180.803467
8.0 7.0 45.617832 -8.645006 0.809899 47.371208
8.0 5.0 41.181011 -4.488753 0.026151 108.221344
8.0 4.0 22.676432 5.083725 1.006810 111.184868
8.0 2.0 43.125423 7.895810 2.009708 68.980896
8.0 1.0 138.612152 -0.782953 2.867923 175.957809
8.0 8.0 56.996407 -3.228145 2.780599 141.993729
8.0 8.0 93.197708 4.960991 2.653369 141.014648
8.0 12.0 5.402694 8.799148 0.171538 22.580589
8.0 6.0 26.048943 -4.152093 1.487906 177.660324
8.0 7.0 145.335342 1.621929 2.101265 158.303345
8.0 2.0 4.057752 5.176256 0.222282 72.741020
8.0 8.0 14.521362 4.555463 0.821942 84.941055
8.0 6.0 4.910694 9.022774 2.999629 146.977936
8.0 12.0 43.754715 7.358728 2.218376 110.441399
8.0 1.0 93.213112 9.206882 1.997740 146.083328
8.0 7.0 24.728905 -0.702523 0.048370 127.390831
8.0 2.0 104.311485 -0.827537 0.196679 154.205429
8.0 7.0 142.328766 -0.857794 2.187228 180.664078
8.0 1.0 78.081825 9.171314 2.004895 79.389175
8.0 12.0 68.576195 8.531470 0.148072 155.170227
8.0 9.0 161.411346 -2.867732 2.913713 167.307846
8.0 2.0 102.867554 -8.263254 0.806208 104.852341
8.0 10.0 63.815540 -9.942202 0.567362 71.924187
8.0 6.0 4.891162 8.344670 1.117734 166.408127
8.0 12.0 94.273193 2.577007 1.874738 174.028946
8.0 3.0 110.318939 8.556031 1.545296 193.045898
8.0 11.0 62.981148 -6.092865 1.930455 155.622345
8.0 9.0 43.341648 1.073558 1.442449 184.661392
8.0 11.0 45.284924 -5.923649 1.994996 93.078049
8.0 7.0 27.471666 0.606786 2.893880 84.591080
8.0 9.0 159.713196 -1.575287 1.925944 190.090500
8.0 10.0 33.383499 -1.639206 1.579015 189.659805
8.0 9.0 78.120384 -2.893155 1.242279 199.222641
8.0 7.0 54.537758 3.173535 2.605676 192.927795
8.0 2.0 21.737976 8.231281 2.226030 94.326744
8.0 1.0 45.139282 -7.147657 2.724179 192.262604
8.0 1.0 53.656727 -0.556374 0.694301 104.094269
8.0 1.0 100.501656 -0.628360 2.465803 198.316772
8.0 1.0 63.310490 7.726147 2.465779 170.213104
8.0 4.0 140.075668 3.190076 1.103127 181.756195
8.0 4.0 73.658829 -9.760891 1.519101 187.989212
8.0 6.0 16.309629 -4.288683 2.035343 128.346771
8.0 11.0 9.980968 6.412787 2.645829 36.428658
8.0 12.0 119.971550 -3.026165 2.407001 159.199631
8.0 12.0 37.983685 3.078499 0.223280 44.928566
8.0 9.0 55.309322 0.189838 2.524495 177.133011
8.0 11.0 98.902496 -5.705263 0.392471 185.548950
8.0 5.0 22.787519 4.973252 2.903532 104.033264
8.0 6.0 163.363083 -0.128692 0.391442 187.775345

8.0 9.0 28.650900 5.423591 1.640258 146.635284
8.0 11.0 164.950821 2.507019 2.937588 193.021408
8.0 6.0 58.736149 1.390045 1.348215 77.774925
8.0 8.0 3.872991 -6.690140 0.897145 97.800682
8.0 9.0 37.242596 -3.432184 0.487529 198.238495
8.0 10.0 101.871727 -1.366438 1.923136 131.287552
8.0 7.0 48.573105 -5.765879 1.911244 175.707687
8.0 11.0 75.225281 -2.678143 2.277886 124.599411
8.0 11.0 89.602448 9.956639 0.318995 197.422714
8.0 9.0 25.665512 -0.640876 0.477287 166.472641
8.0 10.0 93.120644 1.536794 1.649897 121.818535
8.0 7.0 113.594673 -2.969506 1.668524 116.840286
8.0 5.0 108.886177 5.990815 2.265765 174.376984
8.0 12.0 29.428701 3.672730 2.517172 42.695763
8.0 7.0 107.642807 -5.898223 1.299772 154.797073
8.0 4.0 40.464085 -6.102719 0.132814 185.908295
8.0 4.0 30.436449 -2.670382 1.456803 155.016312
8.0 2.0 37.323013 2.972070 2.863248 107.626923
8.0 6.0 12.216588 -5.456238 2.443910 31.217234
8.0 4.0 20.681087 4.716594 1.204703 112.577202
8.0 4.0 11.732592 3.274464 1.294961 188.176743
8.0 7.0 100.164680 0.112611 2.411328 193.760590
8.0 3.0 166.702972 1.978529 1.440424 199.479385
8.0 4.0 45.069084 -3.802151 1.614989 149.441422
8.0 1.0 113.262558 -2.100609 2.104787 137.151993
8.0 5.0 13.653194 -5.503553 0.730340 84.535912
8.0 12.0 87.885994 8.121116 2.147930 98.688789
8.0 9.0 35.279476 -5.040557 1.054058 46.173508
8.0 1.0 35.323872 4.862755 2.883709 172.942963
8.0 5.0 24.837357 -3.969338 2.370103 152.225235
8.0 9.0 163.079422 3.494617 0.428365 176.583572
8.0 10.0 114.839615 2.882653 0.989705 116.299553
8.0 9.0 1.091883 -1.628183 1.735105 143.183914
8.0 2.0 118.748222 2.829127 2.962109 146.342896
8.0 8.0 8.183634 -9.521950 2.827422 12.667070
8.0 1.0 176.166443 -5.492836 1.700212 192.492157
8.0 12.0 112.458809 -3.150151 2.343964 157.331009
8.0 9.0 36.983257 -5.571917 2.798783 199.737137
8.0 9.0 35.008663 -3.298670 2.219608 46.009682
8.0 3.0 2.933636 4.862610 0.953436 18.510489
8.0 2.0 45.428802 -2.365610 0.592027 71.991783
8.0 7.0 72.751633 0.460389 0.544460 92.279091
8.0 8.0 146.048676 8.082449 0.709155 161.425842
8.0 8.0 46.560478 -5.282976 0.638300 86.644638
8.0 4.0 113.718102 0.112764 1.071043 163.941605
8.0 4.0 28.542988 2.799178 1.891518 46.458668
8.0 4.0 2.287769 -3.211876 2.679278 149.709000
8.0 10.0 24.680946 -7.734604 1.251975 38.884899
8.0 5.0 155.008347 -2.887717 0.514934 182.486465
8.0 11.0 150.151886 -7.060324 0.170557 187.856201
8.0 12.0 10.311120 2.713112 1.138099 163.595642
8.0 10.0 95.555618 -7.950796 1.971020 185.963928
8.0 9.0 86.119690 -5.419801 1.162148 138.976074
8.0 1.0 17.440889 -6.173627 0.762554 139.077271
8.0 7.0 79.994164 -4.218322 0.305749 176.918533
8.0 1.0 170.506500 -5.652208 0.735251 177.398438
8.0 11.0 120.803879 5.522686 2.109367 174.775085
8.0 5.0 102.200249 -8.372641 2.767596 116.180946
8.0 9.0 43.087505 4.481154 0.687552 88.306313
8.0 12.0 147.426910 2.059428 1.887318 164.578308
8.0 1.0 23.373299 2.170593 2.111170 99.567184
8.0 4.0 79.266045 5.471767 2.148786 97.755623
8.0 10.0 37.443192 -1.516400 0.474793 80.750122
8.0 4.0 82.559998 -7.474284 1.617422 171.485077
8.0 5.0 19.372000 0.187460 1.004566 80.526665
8.0 4.0 14.276244 6.450630 2.896439 147.623749
8.0 1.0 25.923326 -9.123878 0.377498 140.451752
8.0 1.0 13.250803 -6.132621 1.330847 165.803452
8.0 3.0 24.168806 8.655135 0.432953 155.107468
8.0 1.0 56.121864 -2.025624 0.817687 117.502388
8.0 10.0 37.189671 -2.161546 2.926301 99.130936
8.0 4.0 13.587741 2.763134 1.303033 58.873943
8.0 6.0 3.845816 -9.619049 0.735202 197.719589
8.0 7.0 67.442001 -9.539523 0.042092 135.539276
8.0 11.0 147.594482 -3.954651 1.652680 198.499786
8.0 5.0 58.091869 -0.830347 1.372527 174.036560
8.0 10.0 46.727715 5.509433 0.963000 174.487503
8.0 8.0 39.115562 -9.323033 0.871786 124.378059
8.0 4.0 59.596756 -0.975518 0.490364 70.234314
8.0 10.0 84.986191 -8.095092 1.653499 164.236298
8.0 6.0 69.292809 -4.816455 0.980122 86.333839
8.0 11.0 78.662659 -3.166421 2.099130 86.601234
8.0 11.0 39.389301 -1.195852 2.627320 48.491459
8.0 5.0 45.971642 6.778442 0.752572 85.995705
8.0 8.0 42.601147 7.556629 0.568451 198.436203
8.0 7.0 142.554367 -0.930138 2.101406 167.996704
8.0 9.0 25.972715 -4.863897 0.631466 178.151886
8.0 4.0 85.574203 -7.931014 1.645749 136.503281
8.0 1.0 98.071831 7.712434 2.005870 146.962418
8.0 1.0 7.968076 2.802939 1.841634 48.649960
8.0 2.0 16.659899 2.466424 2.662264 40.137501
8.0 10.0 63.100983 -3.185331 1.403694 171.604507
8.0 5.0 29.347260 -5.968417 2.919303 188.051544
8.0 4.0 27.359941 -3.602934 1.070561 58.415703
8.0 7.0 111.387650 -5.224852 2.478925 138.248688
8.0 12.0 10.093424 6.428528 2.298655 19.267057
8.0 5.0 33.026928 -8.324348 1.216494 153.504654
8.0 6.0 41.368511 3.672728 1.803308 47.274738
8.0 10.0 61.793751 9.548324 2.416578 133.823898
8.0 12.0 102.975174 -7.916235 1.224582 198.438828
8.0 4.0 97.131325 -4.314402 1.176142 97.183746
8.0 8.0 143.690216 8.798094 1.727100 186.860489
8.0 6.0 178.860275 -5.331815 2.473535 189.016724
8.0 11.0 85.920586 3.755967 0.645357 130.803635
8.0 12.0 12.721689 9.520845 0.247231 43.764301
8.0 4.0 8.938370 0.597612 0.868583 48.511787
8.0 4.0 37.864830 -2.483747 1.681840 189.599335
8.0 4.0 111.684792 6.242248 1.599461 169.394150
8.0 12.0 91.487236 2.168248 2.513304 107.869217
8.0 10.0 140.936172 6.538585 0.418037 157.396332
8.0 3.0 112.121658 7.849881 1.151322 148.469467
8.0 6.0 67.849663 -5.127299 1.885848 87.404472
8.0 3.0 16.248276 -6.285224 2.349385 196.658173
8.0 3.0 68.712730 9.183251 0.084173 84.039619
8.0 8.0 181.160934 -6.507923 0.856892 199.319214
8.0 4.0 6.164089 8.397712 0.891347 84.265839
8.0 4.0 63.808388 8.179790 2.287302 73.168938
8.0 2.0 4.158579 1.224892 2.825935 158.543381
8.0 5.0 70.887497 8.711919 0.542948 127.650848
8.0 8.0 8.478434 4.523382 1.003643 135.192993
8.0 3.0 68.901215 8.305599 1.523886 160.986252
8.0 9.0 165.065155 -8.196522 2.193909 171.971802
8.0 8.0 68.283478 -7.799463 0.851129 162.078827
8.0 10.0 19.134129 -3.938421 2.298271 177.455399
8.0 3.0 150.611237 -8.040744 0.886326 181.264404
8.0 12.0 144.965179 -5.002015 2.308575 189.525391
8.0 5.0 21.649097 -2.714970 1.132650 192.596786
8.0 8.0 110.655121 -1.616341 0.758334 170.164108
8.0 12.0 54.080463 8.042671 1.205570 56.133877
8.0 12.0 96.192978 0.535394 1.529058 163.623367
8.0 1.0 122.564758 -1.323196 1.719088 157.343384
8.0 7.0 41.290462 -9.022078 2.450573 90.220589
8.0 6.0 114.132118 -7.232230 0.253767 131.602509
8.0 12.0 173.664154 2.094244 1.025574 186.154388
8.0 9.0 38.293201 4.516220 1.921169 127.521507
8.0 6.0 63.974682 -4.992108 2.562460 81.822945
8.0 1.0 17.064535 -1.558024 2.252366 68.159241
8.0 6.0 36.430813 6.305052 2.490860 189.829346
8.0 9.0 174.091644 1.334177 2.570311 179.623230
8.0 3.0 94.258911 -4.113177 1.427140 105.334717
8.0 10.0 67.270142 -6.270813 0.017476 118.087257
8.0 8.0 104.087433 5.046164 2.622763 135.255249
8.0 2.0 115.110680 2.473783 0.097937 186.078934
8.0 5.0 97.567436 4.269776 2.528670 186.326218
8.0 5.0 16.333847 -7.494029 2.309548 92.539528
8.0 6.0 68.138420 6.241155 2.601197 130.985138
8.0 1.0 126.922745 4.091016 1.541376 193.483047
8.0 6.0 181.735809 -4.737432 1.030715 197.560135
8.0 9.0 48.833813 -5.679303 0.035910 178.145416
8.0 8.0 16.453764 -8.179356 2.585354 90.065430
8.0 6.0 38.564648 3.967503 2.083636 103.467438
8.0 12.0 54.540611 -4.273872 2.274285 179.583588
8.0 10.0 37.119247 5.003695 0.568788 54.430969
8.0 5.0 40.548458 4.705437 0.935245 182.802460
8.0 7.0 108.639954 -0.300105 2.610896 168.937607
8.0 7.0 132.930710 -8.735169 1.638062 161.830383
8.0 1.0 40.457924 -9.728800 1.656184 179.673752
8.0 1.0 68.137215 4.235130 2.539634 116.450218
8.0 12.0 39.567921 -5.208981 0.366856 135.618576
8.0 12.0 116.944000 -6.939561 2.745569 189.381241
8.0 11.0 13.001359 -0.318533 2.374712 87.408859
8.0 12.0 71.429886 -9.179692 2.418184 193.734802
8.0 1.0 47.750271 -1.520406 0.547920 109.433189
8.0 10.0 14.526658 8.294592 0.440789 198.199051
8.0 7.0 59.198242 -1.411397 1.899304 198.959564
8.0 4.0 103.805328 -9.578688 2.756881 165.553711
8.0 11.0 138.099258 4.504475 2.915357 144.481491
8.0 5.0 134.713058 -6.194101 2.059307 141.965637
8.0 4.0 165.421768 -6.619981 2.661872 189.584610
8.0 12.0 25.699247 -4.519383 0.182829 25.784744
8.0 5.0 25.175001 1.183469 2.057220 34.708553
8.0 4.0 65.509659 -4.015998 0.300117 174.314163
8.0 2.0 69.432449 -2.121647 0.815052 160.407700
8.0 6.0 44.802647 2.694475 2.680522 164.676483
8.0 6.0 18.068810 -2.961038 2.241510 189.412262
8.0 7.0 4.446296 2.662418 2.810120 36.627079
8.0 10.0 59.244442 3.594436 2.749784 161.679871
8.0 10.0 40.386772 7.157528 0.983271 196.920227
8.0 6.0 135.190765 0.431471 1.063657 138.569534
8.0 4.0 102.191444 3.313466 0.104373 172.392120
8.0 6.0 11.341682 -6.524371 1.778678 123.080025
8.0 8.0 54.244152 -2.716489 2.969197 57.707840
8.0 8.0 96.665543 3.053025 2.543251 97.578850
8.0 6.0 27.523117 1.722771 1.315640 94.337189
8.0 7.0 84.907295 3.782637 2.611013 134.321060
8.0 10.0 17.860991 5.515649 2.370556 198.600159
8.0 12.0 28.248240 -1.092462 0.657488 38.965099
8.0 6.0 70.939949 2.266138 0.917604 95.422760
8.0 7.0 27.291960 -1.435877 0.858055 132.590088
8.0 3.0 81.091751 0.898728 2.927988 127.271278
8.0 12.0 65.418594 -0.774691 0.405558 76.911179
8.0 8.0 14.624646 -4.146360 1.933772 116.031387
8.0 4.0 37.968987 8.635957 0.545506 93.743729
8.0 7.0 104.708527 -4.514191 1.838214 130.973984
8.0 10.0 27.074511 4.721091 0.225873 162.868195
8.0 7.0 131.755676 -4.341987 1.430776 177.293594
8.0 5.0 111.242119 5.702153 2.327864 118.462898
8.0 3.0 6.304066 -3.621953 2.446009 133.317169
8.0 7.0 107.813591 8.244928 0.486422 176.581879
8.0 9.0 93.713631 4.207499 2.518256 158.629517
8.0 5.0 13.895805 -8.093559 2.229792 156.384384
8.0 12.0 29.736752 -2.284284 1.500637 49.291710
8.0 6.0 68.206818 0.164922 2.840569 138.288361
8.0 12.0 164.122055 2.785559 1.621104 187.546616
8.0 12.0 28.823435 6.243329 2.043115 61.348576
8.0 5.0 159.355103 -4.390786 0.955997 179.176514
8.0 2.0 89.280037 -3.828688 1.692356 153.950027
8.0 3.0 52.376389 -0.982915 1.952733 175.149811
8.0 9.0 4.205302 8.389340 0.823615 172.613113
8.0 9.0 19.168665 6.965286 1.155431 48.194099

8.0 2.0 182.504547 -9.780095 1.621134 195.077789
8.0 6.0 5.059651 -1.100721 1.795775 49.698895
8.0 1.0 97.234947 -8.775687 1.454486 190.367935
8.0 10.0 131.126617 -8.192533 1.036571 176.080002
8.0 12.0 129.416214 2.146293 2.607545 187.243683
8.0 5.0 119.957062 3.972363 2.140945 169.631470
8.0 10.0 176.032227 -2.700865 2.541890 178.969193
8.0 11.0 144.236176 -6.124504 1.141840 174.278015
8.0 3.0 101.290283 3.166792 1.254289 155.072769
8.0 4.0 95.657280 9.232198 2.698198 129.887192
8.0 2.0 104.768616 0.971828 0.714088 162.792633
8.0 4.0 27.551029 4.563350 0.111643 125.024849
8.0 12.0 100.544090 -9.548505 2.468407 178.278091
8.0 10.0 43.415958 -5.442785 1.955736 46.342152
8.0 6.0 3.436999 -8.392205 0.837612 69.528557
8.0 9.0 19.230721 -4.817082 1.409667 24.703112
8.0 7.0 46.901749 -3.916349 1.091310 127.857307
8.0 3.0 87.252174 0.301964 2.276449 92.381165
8.0 3.0 60.496223 -1.086138 1.453747 191.557831
8.0 9.0 84.764603 8.261230 0.234874 156.014282
8.0 2.0 145.534973 -5.769293 1.559783 145.772186
8.0 11.0 105.886299 2.065960 0.204986 171.274994
8.0 7.0 72.513374 0.850096 2.734771 111.923317
8.0 4.0 58.874809 -0.852022 0.331629 105.669830
8.0 12.0 19.601698 7.225636 0.881056 140.366638
8.0 10.0 17.241337 6.869156 1.441129 65.135307
8.0 10.0 6.820424 2.153957 0.961933 135.830200
8.0 4.0 84.200661 2.216003 0.558879 85.362335
8.0 3.0 18.634275 -9.894376 1.312576 179.657593
8.0 11.0 91.712807 -3.139698 1.589754 172.471466
8.0 1.0 131.447052 -9.742850 2.314296 153.896393
8.0 5.0 60.917309 0.867734 0.872042 133.247192
8.0 8.0 66.961029 2.024557 1.692849 197.122162
8.0 5.0 50.102242 3.221377 2.444276 195.322128
8.0 4.0 123.773094 3.150597 2.029869 181.719803
8.0 4.0 85.852028 8.090543 0.300969 196.497696
8.0 3.0 26.429346 2.395243 1.089777 195.958359
8.0 8.0 6.234479 4.154498 2.546395 114.558350
8.0 9.0 149.302002 -2.110948 0.370202 176.606308
8.0 8.0 17.530943 -2.962172 0.988821 79.690491
8.0 4.0 20.505005 -4.181765 1.546849 146.940170
8.0 7.0 16.689756 8.329253 2.456661 43.387657
8.0 9.0 12.636844 -0.040728 1.211680 90.986908
8.0 2.0 0.766002 9.851117 1.482813 42.596310
8.0 1.0 49.009037 4.945451 0.022959 110.811264
8.0 10.0 55.777512 0.197760 0.324366 184.565674
8.0 12.0 60.294842 -7.863108 0.878969 110.664925
8.0 4.0 127.605339 -9.330128 1.096865 143.521942
8.0 5.0 102.046150 7.474882 0.274263 157.735260
8.0 1.0 18.011301 -7.890971 0.859359 82.675453
8.0 1.0 65.056786 -0.362521 1.609607 183.864151
8.0 10.0 106.123955 -4.324207 2.672290 157.379593
8.0 9.0 110.067215 -8.518485 0.630180 149.701736
8.0 4.0 6.755666 -4.182184 2.470325 192.957413
8.0 9.0 164.313416 7.674267 0.327671 180.068710
8.0 9.0 2.007798 1.450487 1.085411 22.126699
8.0 5.0 28.727030 -7.799535 1.412460 142.038239
8.0 7.0 40.061359 3.118286 2.572854 72.893700
8.0 11.0 11.784277 2.753941 0.446139 81.835251
8.0 3.0 57.472672 -7.559776 2.045711 160.441528
8.0 10.0 10.199279 -0.929753 0.756226 149.895126
8.0 2.0 148.820572 -8.041183 1.705385 176.468170
8.0 12.0 158.515182 7.327429 1.046495 181.792862
8.0 8.0 65.221397 7.789611 1.418302 91.209496
8.0 7.0 42.301712 -5.927191 2.768978 91.794579
8.0 10.0 43.897896 -5.503969 0.504097 192.887207
8.0 1.0 110.400154 -4.326507 1.726950 156.473557
8.0 8.0 117.800323 -4.597236 2.372756 138.503738
8.0 12.0 17.786137 9.448895 1.070920 139.865433
8.0 7.0 114.498383 -4.309084 2.824533 120.287514
8.0 12.0 101.030823 7.851832 0.637626 120.889030
8.0 11.0 114.881180 1.247499 1.198250 178.881790
8.0 4.0 91.429962 4.349438 1.859193 151.861252
8.0 9.0 31.473722 9.506461 2.685706 64.282265
8.0 1.0 40.209221 0.010143 0.473118 45.125931
8.0 10.0 160.056244 -3.031496 2.686761 189.571854
8.0 5.0 115.449287 1.951136 1.157048 121.261047
8.0 2.0 25.656448 -6.221125 0.125256 95.254440
8.0 5.0 23.137754 -3.644823 1.703816 53.920830
8.0 6.0 44.103924 -2.830258 2.994170 88.611099
8.0 10.0 80.535133 -3.126294 2.813883 154.720154
8.0 6.0 0.592676 9.964326 1.115263 84.382385
8.0 5.0 45.201649 -9.183793 2.784984 181.076782
8.0 4.0 115.888939 4.364284 1.481684 197.805756
8.0 12.0 80.945671 -8.381749 2.415249 180.381378
8.0 12.0 17.822901 3.419805 0.879953 91.044266
8.0 3.0 54.056274 -8.647423 2.948606 141.799377
8.0 6.0 137.013641 6.403788 2.209637 177.147659
8.0 11.0 58.888359 -5.799335 2.723095 121.644852
8.0 8.0 83.874039 -9.587542 1.607307 198.448700
8.0 5.0 118.649696 4.666414 2.544294 156.736053
8.0 1.0 22.970961 9.541767 2.370384 93.362068
8.0 8.0 104.275040 -9.651664 1.951182 176.971146
8.0 1.0 57.301044 5.177618 1.143156 89.481674
8.0 2.0 106.074234 -6.134956 1.606160 173.772705
8.0 10.0 158.622559 7.788235 2.450329 162.144104
8.0 4.0 8.283471 1.655901 1.102729 166.090622
8.0 1.0 62.269833 -1.575273 0.529536 63.905930
8.0 10.0 102.355324 -8.824838 2.059513 181.125565
8.0 1.0 26.162342 2.579809 0.119009 64.991302
8.0 3.0 11.814712 7.149282 0.465376 95.674294
8.0 2.0 13.881954 -5.795333 2.183232 70.238258
8.0 7.0 13.589021 -0.181602 2.743708 28.936586
8.0 7.0 112.356483 -7.477275 0.094167 122.373634
8.0 10.0 66.533630 -7.068519 1.585504 77.184113
8.0 7.0 26.510862 3.849252 1.764728 40.229694
8.0 8.0 75.025513 -6.521385 1.480021 109.001266
8.0 4.0 56.941525 -1.491983 0.693137 196.432129
8.0 3.0 151.294037 0.810460 0.683547 165.674301
8.0 2.0 37.690765 -3.940980 2.724502 80.463211
8.0 7.0 57.811352 -4.523915 2.036335 77.992645
8.0 7.0 72.989510 -1.772883 1.197588 101.632530
8.0 2.0 3.515356 -0.441439 0.968570 177.859390
8.0 8.0 179.583740 1.871595 0.948177 189.824158
8.0 3.0 112.601036 -7.949279 0.079629 112.999680
8.0 2.0 107.366882 0.348812 0.603990 168.063217
8.0 1.0 14.225032 3.931235 0.248926 106.552979
8.0 5.0 90.811867 6.349968 2.873042 130.349182
8.0 10.0 14.751127 -8.528277 0.624238 57.860500
8.0 8.0 106.698906 2.608750 1.957584 110.829353
8.0 2.0 130.001236 -1.324991 2.091922 196.356613
8.0 12.0 27.834629 7.985704 2.887564 139.035110
8.0 10.0 162.086655 1.647442 2.905789 163.229752
8.0 12.0 83.621788 -8.550568 0.005961 129.399536
8.0 1.0 27.762323 -1.103833 1.664699 198.635269
8.0 12.0 4.634602 -4.919667 0.926310 134.876526
8.0 6.0 8.191989 8.257698 1.860283 115.298836
8.0 8.0 114.666718 8.620451 2.085782 115.021477
8.0 9.0 73.846245 -7.625037 2.858799 161.464539
8.0 4.0 33.127441 7.538258 0.936612 147.362457
8.0 9.0 51.226383 0.032928 2.900163 156.659637
8.0 7.0 178.439682 -9.784158 2.062018 198.121765
8.0 8.0 156.559647 6.224923 1.068649 168.784973
8.0 12.0 120.437439 -5.043176 0.270801 171.592422
8.0 8.0 71.493309 -0.463032 1.327465 191.590576
8.0 12.0 123.189568 -6.105417 1.160184 190.533615
8.0 5.0 38.624069 0.626485 1.010073 76.758614
8.0 2.0 24.480732 8.944241 1.784063 137.918533
8.0 6.0 87.798210 -0.806372 0.824542 143.620728
8.0 6.0 41.390095 -7.271372 1.883447 45.961639
8.0 11.0 52.978157 -4.545469 1.690749 147.838623
8.0 12.0 84.045029 8.283856 2.917364 140.610474
8.0 1.0 34.788818 -0.660127 0.355327 95.555527
8.0 12.0 152.767929 9.707571 2.168732 180.262650
8.0 7.0 51.889381 -5.694790 1.702060 66.084251
8.0 7.0 23.320990 -0.460943 1.793333 139.827118
8.0 4.0 27.739908 -4.100799 0.680897 142.640625
8.0 9.0 38.484066 9.620125 1.019218 89.828751
8.0 5.0 103.690453 -3.056522 0.798548 115.960236
8.0 10.0 42.508018 -5.036370 2.621787 127.596481
8.0 5.0 43.671085 -1.493124 0.517994 93.253105
8.0 9.0 113.914459 0.568323 2.898131 184.824341
8.0 9.0 0.052993 9.215666 2.326613 32.356567
8.0 7.0 29.946115 8.886800 1.261506 86.888184
8.0 3.0 108.554718 -7.364392 0.848893 149.582611
8.0 8.0 41.107445 -2.092142 0.075873 100.947906
8.0 10.0 57.682850 0.930798 1.583985 112.863358
8.0 1.0 34.257195 2.164788 1.098224 165.941666
8.0 12.0 19.782562 5.838943 2.869043 101.622124
8.0 6.0 142.525955 5.169804 0.641753 172.478821
8.0 5.0 137.280396 -8.680936 1.680089 182.265808
8.0 12.0 68.935539 2.306858 0.409811 136.664581
8.0 12.0 4.765196 8.351755 2.354773 65.869141
8.0 9.0 69.492264 -2.272137 1.963379 144.916718
8.0 9.0 40.794067 5.164119 0.595817 63.405968
8.0 9.0 40.452629 -3.690134 2.296350 44.191849
8.0 7.0 24.074213 1.669899 1.057234 138.389175
8.0 4.0 11.933031 7.291333 0.144660 37.361149
8.0 7.0 40.298420 -5.496592 2.095791 120.208572
8.0 12.0 6.078831 1.825183 0.331837 95.800911
8.0 2.0 52.971943 -5.231248 1.180998 130.980728
8.0 7.0 92.885918 -1.175133 0.594052 101.766998
8.0 9.0 135.917770 7.010903 1.379157 143.478165
8.0 12.0 49.909462 8.726936 2.449803 64.973816
8.0 9.0 124.631386 -9.604907 2.827746 171.965652
8.0 11.0 3.814993 -0.243003 0.967952 68.733261
8.0 2.0 157.769928 6.120197 2.334093 175.976959
8.0 7.0 37.375477 2.079745 1.890117 163.668610
8.0 8.0 44.010777 7.148112 0.608991 142.662689
8.0 6.0 86.448662 -3.687105 0.752934 149.849426
8.0 3.0 93.845444 -4.898887 0.219882 100.222305
8.0 1.0 21.413153 4.483871 0.319180 128.014618
8.0 11.0 120.110519 5.310277 0.729262 121.473969
8.0 12.0 134.357727 8.903199 1.050888 199.002655