Reverse Conducting Plots and Analyses

Craig Stuart Sapp <craig@ccrma.stanford.edu>

25 Sept 2005 -- 8 October 2005

■ Loading the raw data for analysis

```
        SetDirectory["D:\revcond"];
```

Load the reverse conducting durations for each beat from the preprocessed data file for Mazurka Op. 7, No. 3. The data is stored in an array which contains the duration between beats in milliseconds. Each row in the array represents a separate reverse conducting trial, and each element in the trial is duration to the *n*th beat of the trial.

```
In[480]:= << id5667230-10.ma

        data = Transpose[id5667230x10];
```

Here are the durations in millseconds between the first and seconds beats (measure 1, beats 1 and 2) of Chopin's Mazurka Op. 7, No. 3 performed by Ignaz Friedman in 1930 for each of the twenty trials:

```
        data[[1]]

        {399, 336, 405, 363, 389, 297, 302, 347, 378,
         343, 343, 441, 481, 415, 456, 383, 482, 373, 421, 391}
```

■ Looking at basic statistics for a beat

Finding the minimum and maximum duration of the first beat:

```
        Max[data[[1]]]

        482

        Min[data[[1]]]

        297
```

The average value for the duration of the first beat:

```
        Mean[data[[1]]] // N

        387.25
```

Find the average beat duration for each beat in all trials:

```
        durationmean = Map[ Mean, data] // N;
```

Now, look at the timing spread between the average beat and the actual beats to determine the basic reproducibility accuracy of the reverse conductor.

```
        beatdiff = (data – durationmean);
```

Here are the differences between the first average beat duration and each individual trial's first beat (in milliseconds):

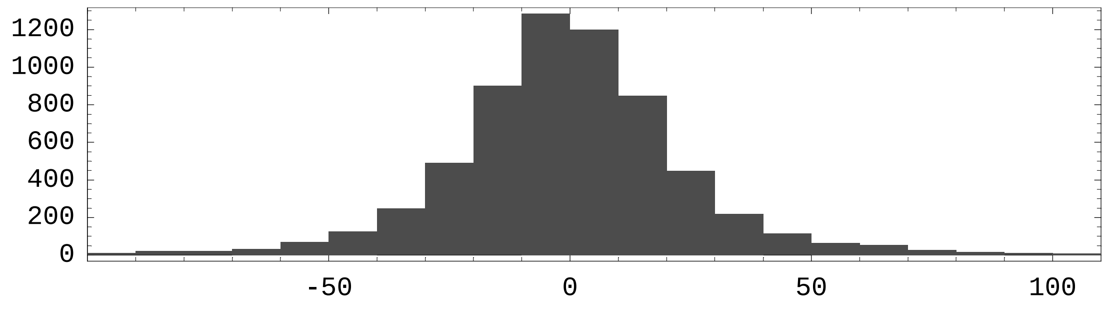
```
beatdiff[[1]]
```

```
{11.75, -51.25, 17.75, -24.25, 1.75, -90.25, -85.25, -40.25, -9.25,
 -44.25, -44.25, 53.75, 93.75, 27.75, 68.75, -4.25, 94.75, -14.25, 33.75, 3.75}
```

Display the distribution of all beats from all trials with respect to the average beat position of each beat:

```
<< Graphics`Graphics`
```

```
Histogram[Flatten[beatdiff], AspectRatio → 1 / 4, Frame → True,
  HistogramRange → {-100, 100}, BarEdges → False, HistogramCategories → 25];
```



This data has a nicely shaped distribution, so meausre some basic statistics about it:

```
avgbeatstats = DispersionReport[Flatten[beatdiff]]
```

```
{Variance → 897.148, StandardDeviation → 29.9524, SampleRange → 932.75,
 MeanDeviation → 18.9415, MedianDeviation → 12.925, QuartileDeviation → 12.9625}
```

The standard deviation of the beat locations are not necessarily constant for each beat, but it is reasonable to assume that the beat accuracy is constant according to the picture above where 95% of all beats between different trials fall within +/- 60 milliseconds of the average beat positions for each trial. This means that the reverse conductor in this can reliable repeat his beats for this particular performance within 60 millseconds of his average beat location 95% of the time.

Now calculate the confidence bounds based on the Student *t* distribution with a confidence level of 0.95, by first loading the *Mathematica* package which calculates confidence intervals:

```
<< Statistics`ConfidenceIntervals`
```

```
? MeanCI
```

```
MeanCI[list, options] returns a list {lower, upper} representing a confidence interval for
  the population mean, using the entries in list as a sample drawn from the population.
```

```
Options[MeanCI]
```

```
{ConfidenceLevel → 0.95, KnownStandardDeviation → None, KnownVariance → None}
```

```
MeanCI[data[[1]], KnownStandardDeviation → 29.9524]
```

```
{374.123, 400.377}
```

```
eachbeatstats = Map[DispersionReport, data] // N;
```

```
confidence =
  Map[MeanCI[#[[1]], KnownStandardDeviation → (KnownStandardDeviation /. #[[2]])] &,
   Transpose[{data, eachbeatstats}]];
```

```
confidence[[1]]
```

```
{362.711, 411.789}
```

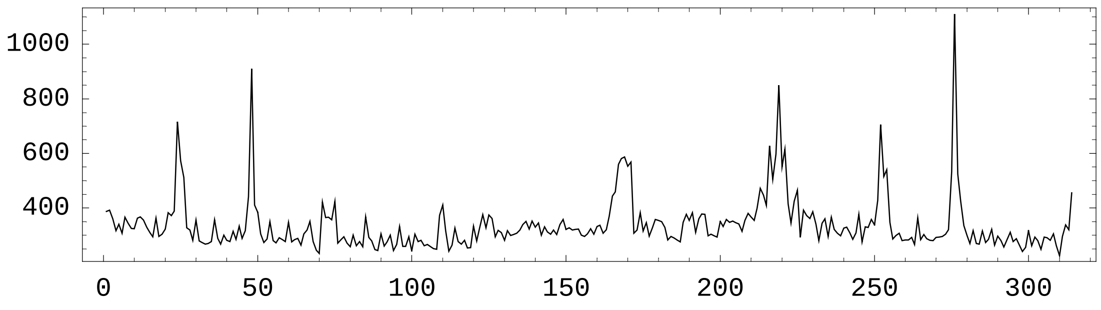
```
In[498]:= durationmean[[1]]

Out[498]= 387.25
```

This confidence interval signifies that the actual mean beat duration of the first beat has a 95% probability of being between 363 and 412 milliseconds.

■ Plotting duration curves for the composition

Here is a plot of the average durations for each beat over the 20 trials of reverse conducting on the composition.

```
meanplot = ListPlot[durationmean,
   PlotJoined → True, PlotRange → All, AspectRatio → 1 / 4, Frame → True];
```

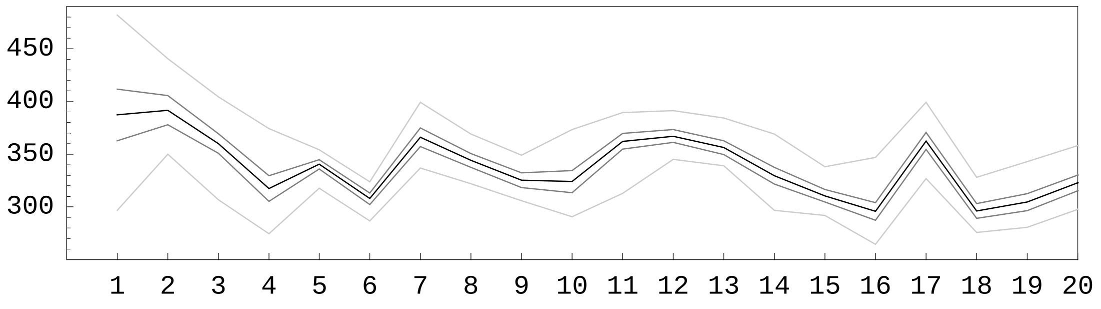


Now add the 95% confidence ranges to the plots as well as the maximum and minimum range for the duration of each beat. The confidence levels are displayed in dark gray, while the min and max range is displayed in light gray.

```
In[992]:= meanhighplot = ListPlot[Transpose[confidence][[1]], PlotJoined → True,
             DisplayFunction → Identity, PlotStyle → RGBColor[.5, .5, .5]];
          meanlowplot = ListPlot[Transpose[confidence][[2]], PlotJoined → True,
             DisplayFunction → Identity, PlotStyle → RGBColor[.5, .5, .5]];
          maxplot = ListPlot[Map[Max, scaleddata], DisplayFunction → Identity,
            PlotJoined → True, PlotStyle → RGBColor[0.8, 0.8, 0.8]];
          minplot = ListPlot[Map[Min, scaleddata], DisplayFunction → Identity,
            PlotJoined → True, PlotStyle → RGBColor[0.8, 0.8, 0.8]];
```

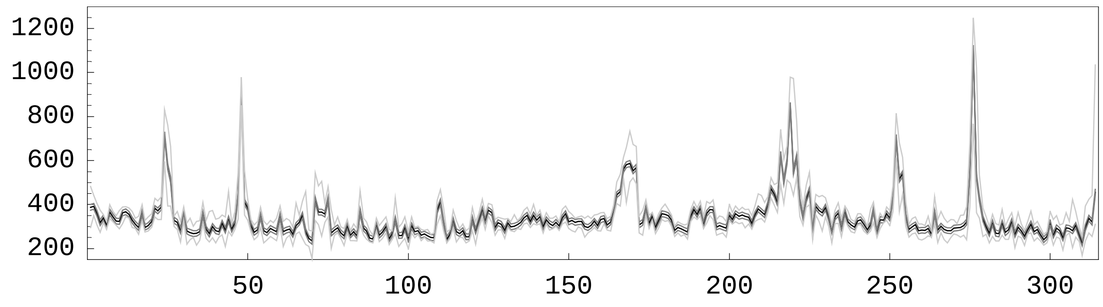
First, zoom in on the first 20 beats so that detail in the plot can be seen:

```
Show[meanplot, meanlowplot, meanhighplot, maxplot, minplot, Frame → True,
  PlotRange → {{0, 20}, {250, 490}}, FrameTicks → {Range[20], Automatic, None, None}];
```



Now, below is shown the beat-duration curve for the entire piece. Notice that the tempo of the opening 50 beats (The A section) is repeated in the return of the A section starting at beat 230.

```
Show[meanplot, meanlowplot, meanhighplot, maxplot,
  minplot, Frame → True, PlotRange → {{0, 315}, {150, 1300}},
  FrameTicks → {Table[ i - 1, {i, 1, 314, 50} ], Automatic, None, None}];
```

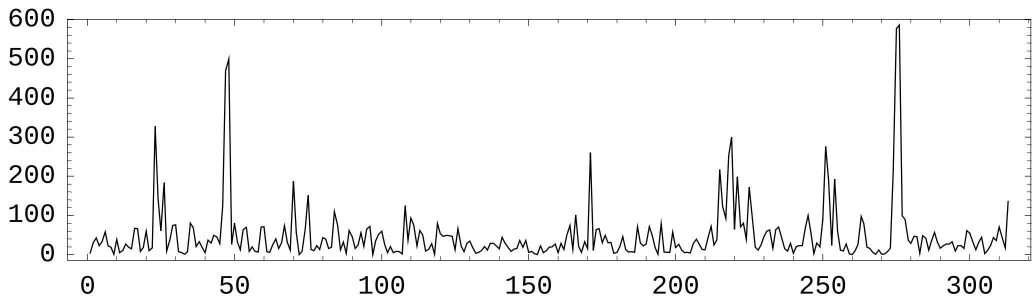


■ Displaying duration (tempo) changes in the performance

As a first approximation, the expressivness of the performance can be interpreted as changes in the tempo. The greater the tempo change, the greater the expressiveness. So, let's display the changes of the duration plots given in the previous section. The following plots display the acceleration and ritard of the performance on a beat by beat level.

```
durchange = Map[(#[[1]] - #[[2]]) &,
   Drop[Transpose[{durationmean, RotateRight[durationmean, 1]}], 1] ];
```

```
meanchplot = ListPlot[Sqrt[durchange * durchange],
   PlotJoined → True, PlotRange → All, AspectRatio → 1 / 4, Frame → True];
```



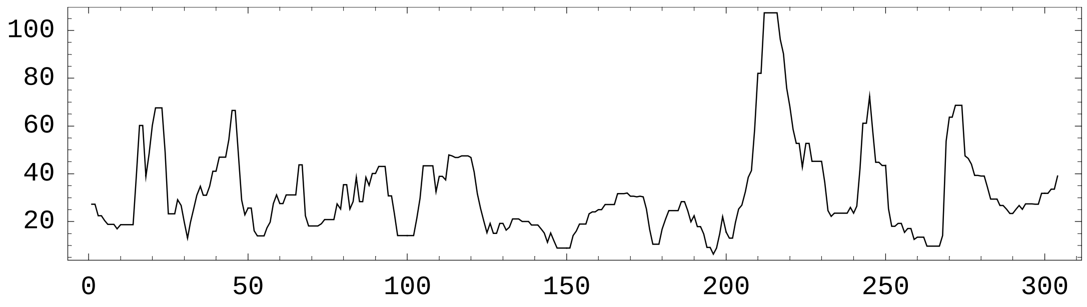
Now, Average the data with a moving median of 10 beats to smooth out the previous plot.

```
<< Statistics`DataSmoothing`
```

```
smoothdur = MovingMedian[Sqrt[durchange * durchange], 10];
```

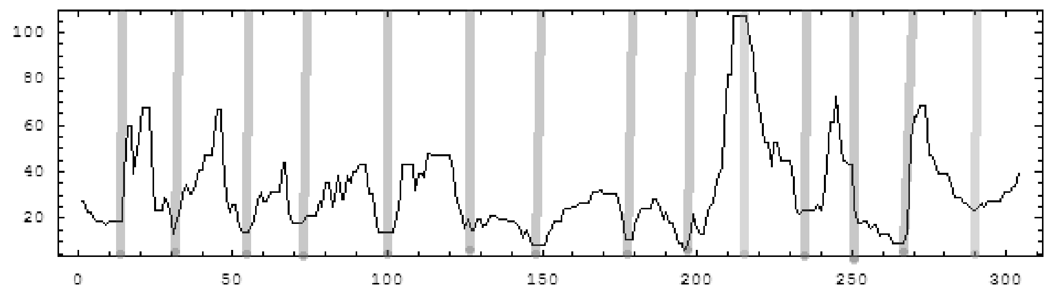
Phrase structure is visible in the following plot of the smoothed changes in duration. Each arch in the plot below represents a phrase (although some are elided towards the end). In particular note that the A section phrases are clearly marked, while the return of the A section starting at beat 255 shows a blurring of the phrases.

```
meansmoothchplot = ListPlot[Sqrt[smoothdur * smoothdur], PlotJoined → True,
   PlotRange → All, AspectRatio → 1 / 4, Frame → True, Axes → False];
```



In the above plot, the phrase starting around beat 125 to beat 150 (bar 41 to 50) is flatest phrase which means that the tempo remained fairly constant throughout that region of the music. This portion of the music is the beginning of the B section of the piece where the music becomes more fanfare-like and military with wide ranges in dynamic. The most "expressive"phrase starts at beat 205 (measures 68 -76). This is the point where there is a second repeat of a melody in the left hand, and marks the recapitulation to the A section in measure 77.

Here is an interpretive segmentation of the music. Red lines indicate phrase boundaries identified by eye; the green lines indicate "missing" phrase boundaries. The first missing phrase boundary is at the peak around beat 215, and the seoncd missing phrase boundary is in the valley at beat 290 where there may be a intentional phrase boundary, but it is very shallow and broad. It is interesting to note that the opening phrase of the piece and its return at beat 250 are similar, but the music of the 2nd and third phrases which is repeated starting at beat 270 have very different expressive tempo qualities.

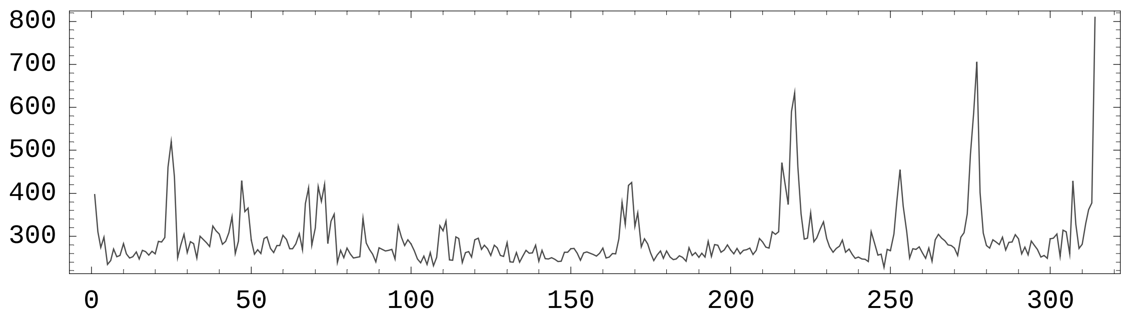


■ Displaying uncertainty in reverse conducting

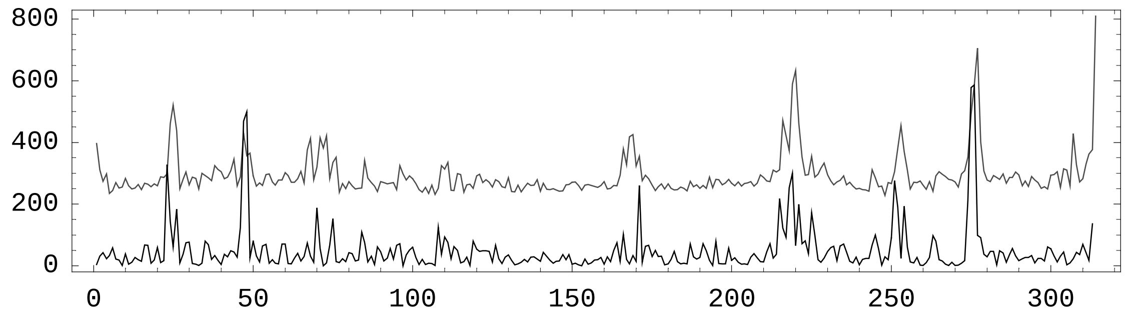
Here is a plot which shows how difficult it was for the reverse conductor to match the beat of the performer at each beat in the composition. The red line is a plot of the difficulty of following the performer. The black line in the second plot below shows the tempo (beat duration) fluctuation plot from above for comparison. Peaks in the black plot line up well with the peaks in the red plot.

```
uncertainty = Map[(#[[2]] - #[[1]]) &, confidence];
```

```
uncertaintyplot = ListPlot[ 4 * uncertainty + 200 , PlotJoined → True,
   PlotRange → All, AspectRatio → 1 / 4, Frame → True, Axes → False, PlotStyle → Hue[0]];
```



```
Show[uncertaintyplot, meanchplot];
```



Not much interpreting of this type of data yet. Basically the more the tempo changes between beats, the more inaccurate the reverse conductor is at locating the next beat properly, which is to be expected. It would be interesting to determine if certain categories of tempo change cause uncertainty in the reverse conducting while other categories of tempo change do not cause uncertainty. For example, offbeats can be used by the performer to indicate the rate of a tempo change before the new beat actually occurs.

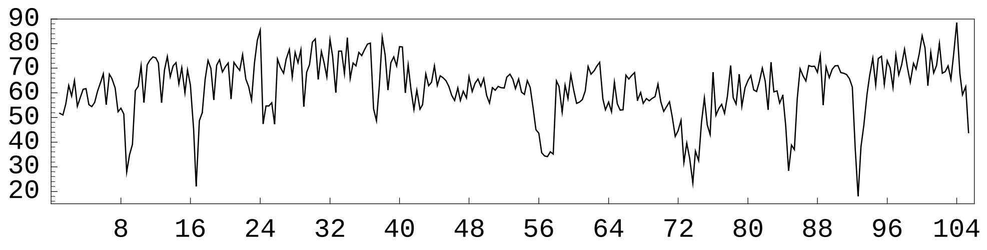
■ Displaying data as tempo per measure rather than as duration per beat

Musicians are more familiar with tempo, so the following plots convert duration in milliseconds into tempo in beats per minute. Also, since this composition is fast, and the tempo is indicated as 54 measures per minute (dotted half note tempo) in the 1915 Schimer edition of the mazurkas edited by Rafael Joseffy, the tempo values are divided by three (three beats per measure) to compare to this metronome marking.

```
len = Length[durationmean]
```

```
314
```

```
dummyplot = Plot[0, {x, 1, 1.001}, Frame → True,
   FrameTicks → {Range[len / 8] * 8, Automatic, None, None},
   Axes → None, DisplayFunction → Identity, AspectRatio → 1 / 5];
meantempoplot = Show[dummyplot, Graphics[
    Line[Transpose[{Table[N[x / 3] + 1, {x, 0, 313}],
         1000 / durationmean / 3 * 60}]]], PlotRange → {{0, 106}, {15, 90}},
   Frame → True, AspectRatio → 1 / 5, DisplayFunction → $DisplayFunction];
```

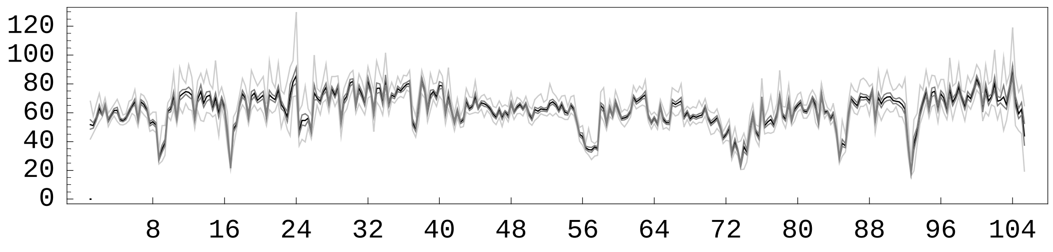


Now display the min, max, and confidence intervals on the tempo plot for the entire composition:

```
points = Transpose[
   {Table[N[x / 3] + 1, {x, 0, 313}], 1000 / Transpose[confidence][[2]] / 3 * 60}];
meanlowtempoplot = Show[dummyplot,
   Graphics[{RGBColor[0.5, .5, .5], Line[points]}]];
points = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}],
    1000 / Transpose[confidence][[1]] / 3 * 60}];
meanhightempoplot = Show[dummyplot,
   Graphics[{RGBColor[.5, .5, .5], Line[points]}]];
points = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}], 1000 / maxrange / 3 * 60}];
mintempoplot = Show[dummyplot, Graphics[{RGBColor[.8, .8, .8], Line[points]}]];
points = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}], 1000 / minrange / 3 * 60}];
maxtempoplot = Show[dummyplot, Graphics[{RGBColor[.8, .8, .8], Line[points]}]];
```

Numbers on x-axis in plot below indicate the measure numbers.

```
finalplot =
  Show[meantempoplot, maxtempoplot, mintempoplot, meanlowtempoplot, meanhightempoplot,
   DisplayFunction → $DisplayFunction, PlotRange → All, AspectRatio → 1 / 5];
```

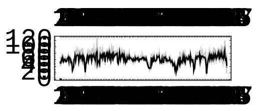


■ Zooming in on the tempo curve for every 8 bars

The previous plot showing the tempo curve for the entire piece is difficult to see detail, so now break up the tempo plots into 8 measure segments.

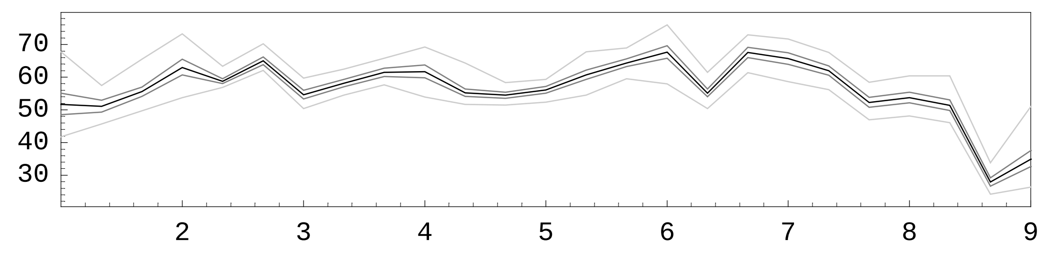
```
dummyplot =
  Plot[0, {x, 1, 1.001}, Frame → True, FrameTicks → {Range[400], Automatic}, Axes → None,
   AspectRatio → 1 / 4, DisplayFunction → Identity];
```

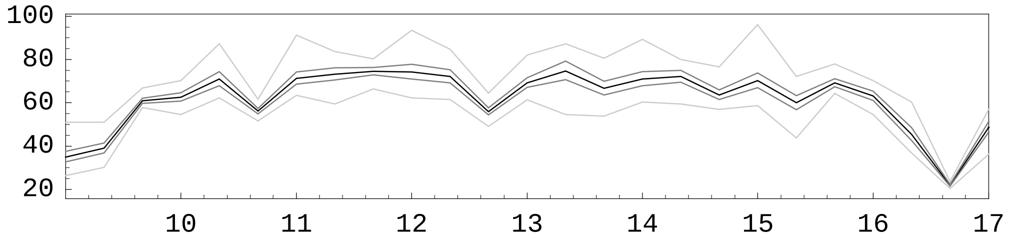
```
finalplot = Show[dummyplot, maxtempoplot,
   mintempoplot, meanlowtempoplot, meanhightempoplot,  meantempoplot
   (*, GridLines→{False, Automatic} *) , DisplayFunction → $DisplayFunction];
```

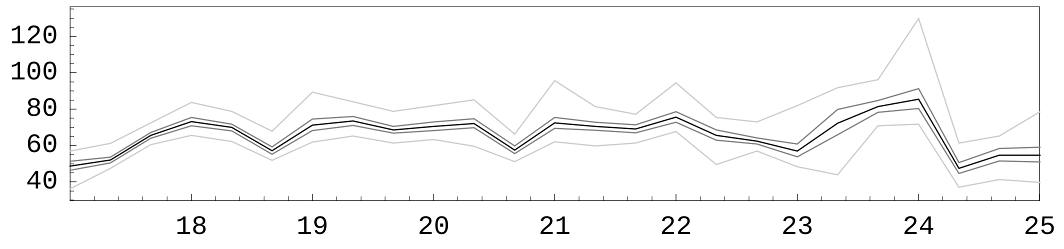


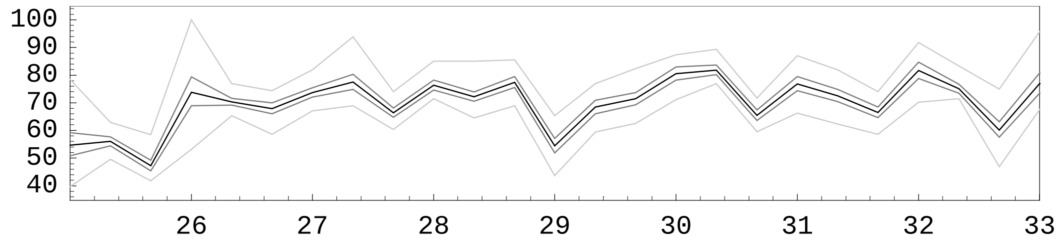
```
ShowFrame[finalplot_, minrange_, maxrange_, len_, frame_, width_] := Module[
  {framecount, startpoint, endpoint, rmin, rmax, starti, endi, delta},
  framecount = len / width / 3.0 + 0.99 // Floor;
  startpoint = frame * width + 1;
  endpoint =  (frame + 1) * width + 1;
  starti = (startpoint - 1) * 3 + 1;
  endi = (endpoint - 1) * 3 + 1;
  If[endi > len, endi = len];
  If[starti > len, starti = len];
  rmax = Max[60000.0 / 3.0 / Take[minrange, {starti, endi}]];
  delta  = rmax * 0.05;
  rmax = rmax + delta;
  rmin = Min[ 60000.0 / 3.0 / Take[maxrange, {starti, endi }]];
  rmin = rmin - delta;
  Show[finalplot, PlotRange → {{startpoint, endpoint}, {rmin, rmax}},
   FrameTicks → {Automatic, Automatic, None, None}]
 ]
```

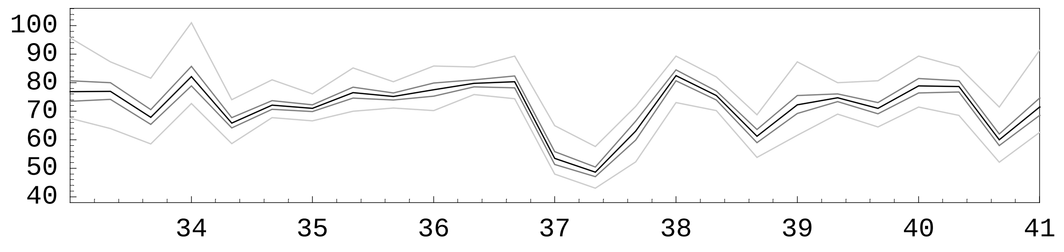
```
Table[ShowFrame[finalplot, minrange, maxrange, len, i, 8],
  {i, 0, Floor[ len / 8.0 / 3.0 + .99 ] - 1}];
```

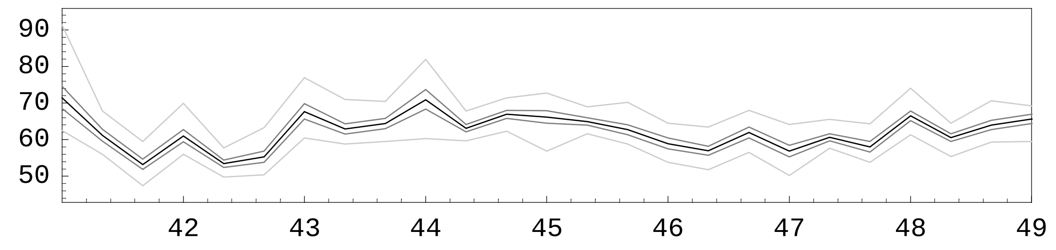

70
60
50
40
30
2 3 4 5 6 7 8 9

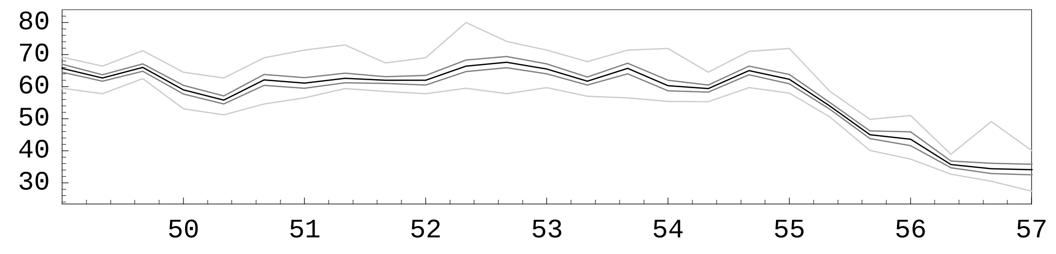

100
80
60
40
20
10 11 12 13 14 15 16 17

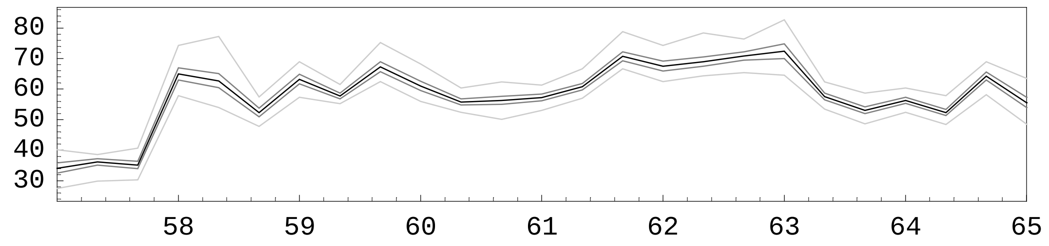

120
100
80
60
40
18 19 20 21 22 23 24 25

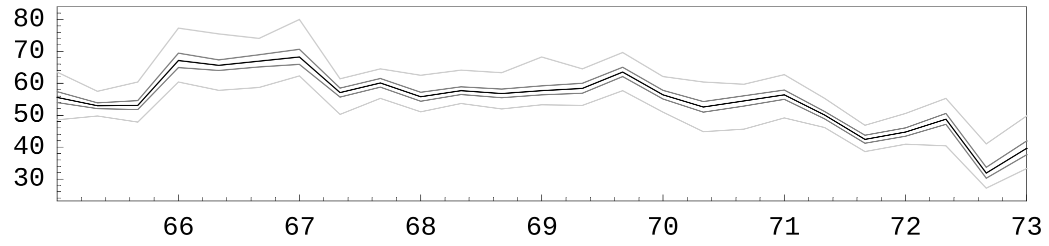

100
90
80
70
60
50
40
26 27 28 29 30 31 32 33

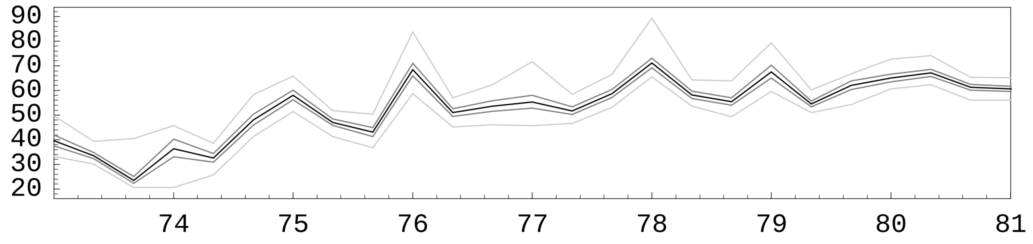

100
90
80
70
60
50
40
34 35 36 37 38 39 40 41

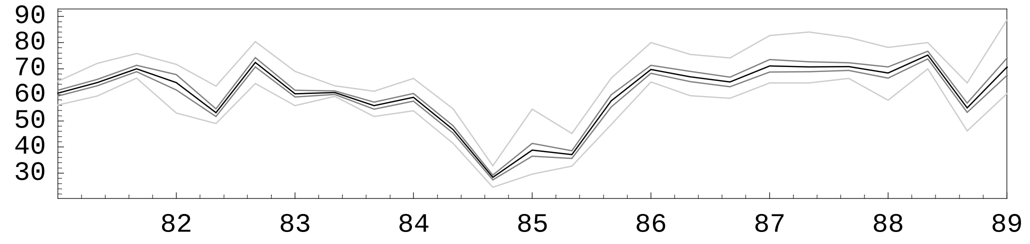

90
80
70
60
50
42 43 44 45 46 47 48 49

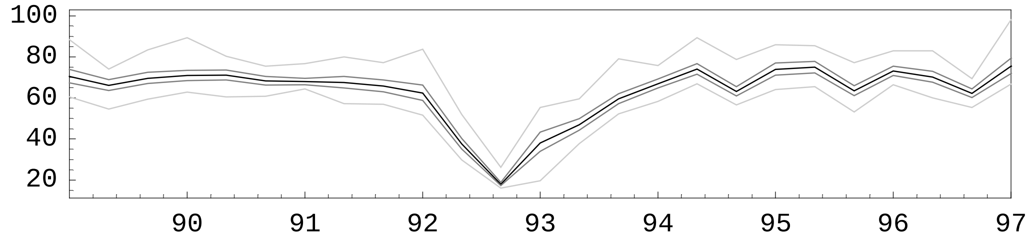

80
70
60
50
40
30
50 51 52 53 54 55 56 57

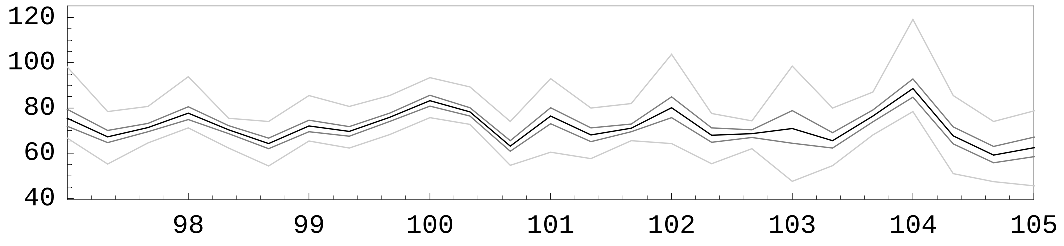

80
70
60
50
40
30
58
59
60
61
62
63
64
65

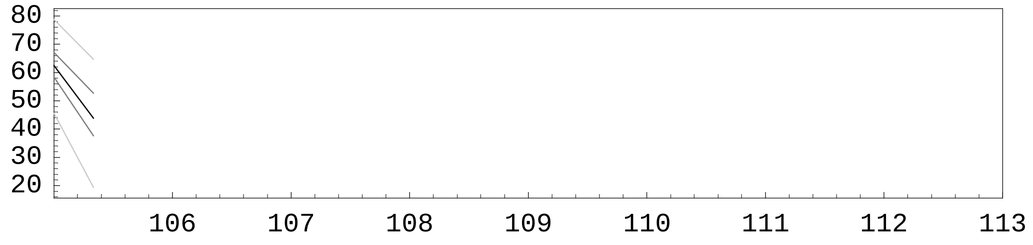

80
70
60
50
40
30
66
67
68
69
70
71
72
73


90
80
70
60
50
40
30
20
74
75
76
77
78
79
80
81


90
80
70
60
50
40
30
82
83
84
85
86
87
88
89

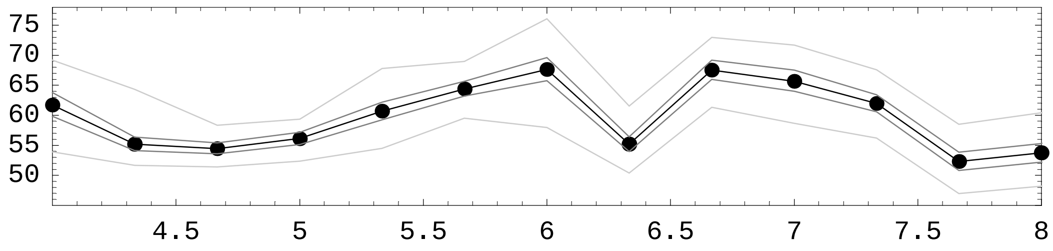

100
80
60
40
20
90
91
92
93
94
95
96
97


120
100
80
60
40
98
99
100
101
102
103
104
105


80
70
60
50
40
30
20
106
107
108
109
110
111
112
113

■ Plotting variations: adding dots at beat points

```
psize = 10;
points = Transpose[
   {Table[N[x / 3] + 1, {x, 0, 313}], 1000 / Transpose[confidence][[2]] / 3 * 60}];
meanlowtempoplot = Show[dummyplot,
   Graphics[{RGBColor[0.5, .5, .5], Line[points]}]];
points = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}],
    1000 / Transpose[confidence][[1]] / 3 * 60}];
meanhightempoplot = Show[dummyplot,
   Graphics[{RGBColor[.5, .5, .5], Line[points]}]];
points = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}], 1000 / maxrange / 3 * 60}];
mintempoplot = Show[dummyplot, Graphics[{RGBColor[.8, .8, .8], Line[points]}]];
points = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}], 1000 / minrange / 3 * 60}];
maxtempoplot = Show[dummyplot, Graphics[{RGBColor[.8, .8, .8], Line[points]}]];
points = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}], 1000 / durationmean / 3 * 60}];
meantempoplot = Show[dummyplot,
   Graphics[{Line[points], PointSize[0.015], Map[Point, points]}],
   PlotRange → {{4, 8}, {45, 78}}, Frame → True, AspectRatio → 1 / 5];
Show[meantempoplot, mintempoplot, maxtempoplot, meanlowtempoplot,
  meanhightempoplot, DisplayFunction → $DisplayFunction];
```

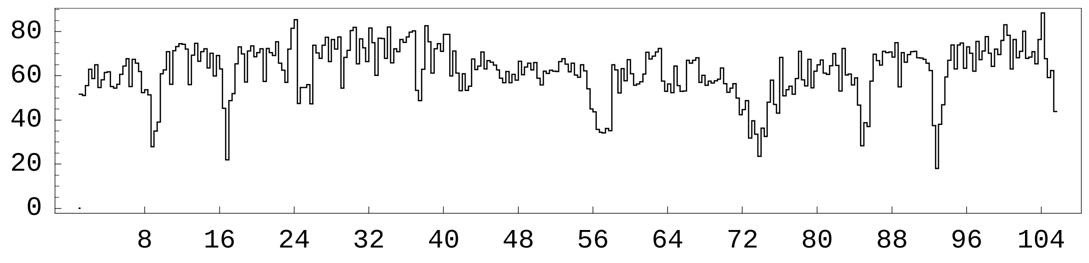


■ Plotting variation: Squared plot

```
dummyplot = Plot[0, {x, 1, 1.001}, Frame → True, FrameTicks →
    {Range[len / 4] * 4, Automatic, None, None}, Axes → None, DisplayFunction → Identity];
```

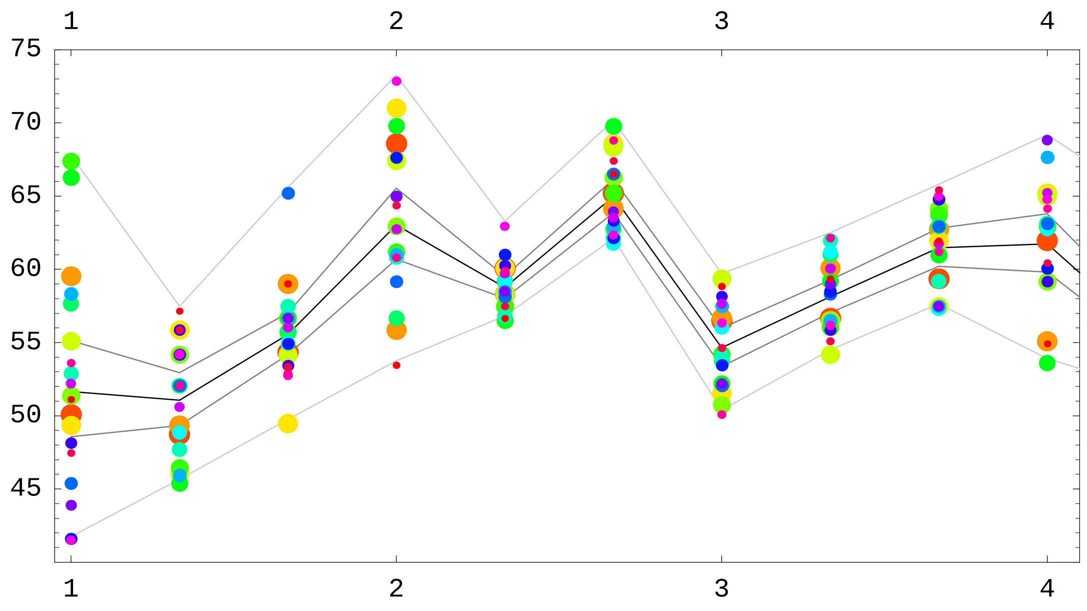
```
squarecurve1 = Transpose[{Table[N[x / 3] + 1, {x, 0, 313}],
      1000 / durationmean / 3 * 60}];
squarecurve2 = Transpose[{Table[N[x / 3] + 1, {x, 1, 314}],
      1000 / durationmean / 3 * 60}];
squarecurve = Flatten[Transpose[{squarecurve1, squarecurve2}], 1];
meantempocurve = Graphics[Line[squarecurve]];
```

```
meantempoplot = Show[dummyplot, meantempocurve, PlotRange → All,
   Frame → True, AspectRatio → 1 / 5, DisplayFunction → $DisplayFunction];
```



▪ Plotting Variations: displaying all trials points in a tempo plot

```
alltestpoints =
  Transpose[{(Range[Length[scaleddata]] - 1.0) / 3.0 + 1.0, scaleddata}];
dpoints = Map[Transpose[{Table[#[[1]], {x, 1, Length[#[[2]]]}],
      60000.0 / 3.0 / #[[2]] } ] &, alltestpoints];
tpoints = Flatten[Map[Transpose[{Range[Length[ #]], #}] & , dpoints] , 1];
gpoints =
  Graphics[{Map[{PointSize[(21 - #[[1]] + 9) * 0.0007], Hue[#[[1]] / 20.0] ,
       Point[#[[2]]]} &, tpoints]}];
Show[finalplot , gpoints, PlotRange → {{0.95, 4.1}, {40, 75} }, AspectRatio → 1 / 2];
```



Size and color indicate the individual trials of reverse conducting so that you can distinguish individual tempo curves. Earlier trials are large red circles, later trials become smaller circles which progress through the colors of the rainbow.

▪ Plotting Variations: Aligning and plotting absolute time of beats

```
datat = Transpose[data];
```

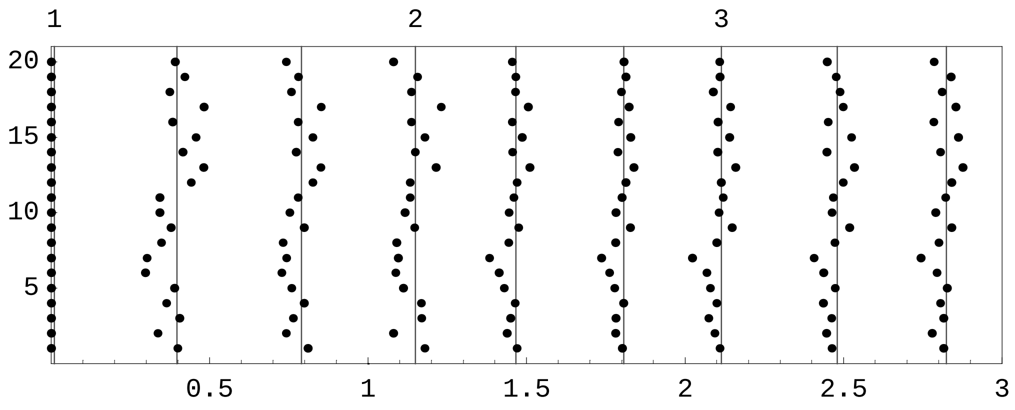
```
meanpoints = Map[Mean, data] // N;
```

```
sum = 0.0;
tempoutput = 0.0;
meansum = 0.0;
meanseq = {};
seqdata = {};
For[i = 1, i ≤ Length[datat], i++,
  sum = 0.0;
  tempoutput = {};
  llen = Length[datat[[i]]];
  For[j = 1, j ≤ llen, j++,
    tempoutput = Append[tempoutput, { sum / 1000.0, i}];
     sum += datat[[i]][[j]];
    If[i == 1, meanseq = Append[meanseq, meansum];
   meansum += meanpoints[[j]] / 1000.0; ];
   ];
  seqdata = Append[seqdata, tempoutput];
]
```

```
dummyplot = Plot[0, {x, 1, 1.001}, Frame → True,
   FrameTicks → Automatic, Axes → None, DisplayFunction → Identity];
```

```
beatlines = Map[Line[{{#, 0}, {#, 21}}] & , meanseq];
measures = Transpose[Partition[meanseq, 3]][[1]];
toplabels = Transpose[{measures, Range[Length[measures]]}];
```

```
Show[dummyplot, Graphics[
   {{Hue[0], beatlines}, PointSize[0.009], Map[Point, Flatten[ seqdata, 1]]}],
  PlotRange → {{0, 3}, {0, 21}}, AspectRatio → 1 / 3,
  DisplayFunction → $DisplayFunction,
  FrameTicks → {Automatic, Automatic, toplabels, None}];
```



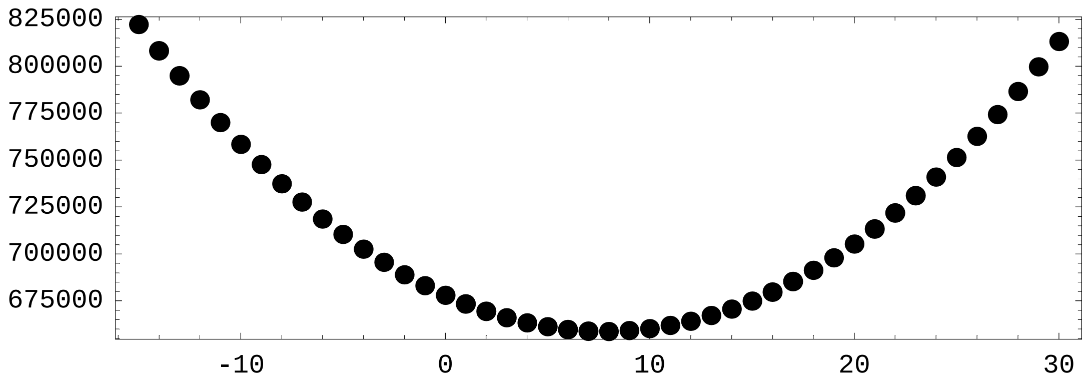
The above plot shows the absolute duraton location of each beat for each trial. The bottom axes labels are in seconds, the top axes labels are in measures, and the right side axes numbers are the trial numbers, starting with the first trial at the bottom of the plot. This plot assumes that all trials started beating at the same point in the recording, which is not physically possible, so the next section tries to identify the offset between each trial by minimizing the difference between beats in each trial compared to the first trial.

■ Aligning the absolute-value beat times between trials

First, do a test alignment between the first and 18th trials.

```
SquareDiff[x_, y_] := Apply[Plus, (x - y) * (x - y)];
```

```
plotpoints = ListPlot[Table[{x, SquareDiff[
        Transpose[ seqdata[[1]] ] [[1]] * 1000.0 ,
        Transpose[seqdata[[18]] ] [[1]] * 1000.0 + x
      ]}, {x, -15, 30, 1} ], Frame → True, PlotRange → All,
   PlotStyle → PointSize[0.02], AspectRatio → 1 / 3, Axes → False];
```



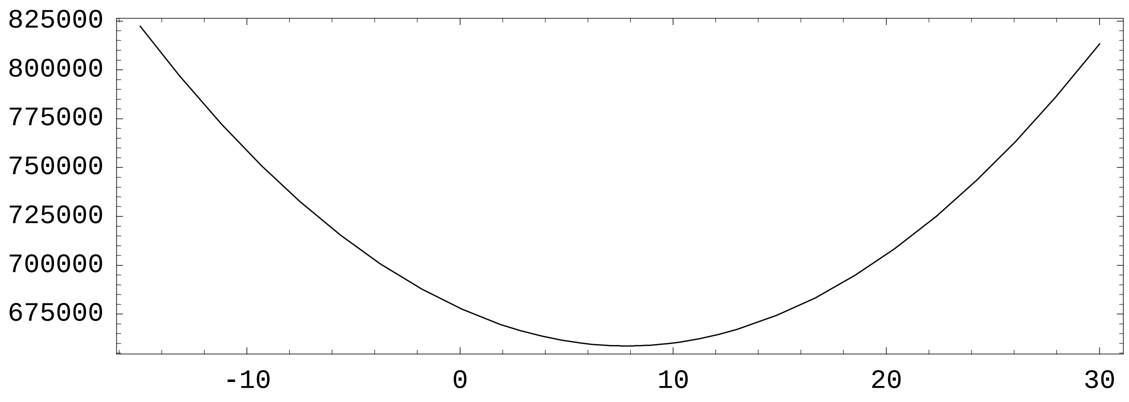
```
dd = Table[{x, SquareDiff[
       Transpose[ seqdata[[1]] ] [[1]] * 1000.0 ,
       Transpose[seqdata[[18]] ] [[1]] * 1000.0 + x
    ]}, {x, -15, 30, 1} ];
```

```
func = Fit[dd, {1, x, x^2}, x]
```

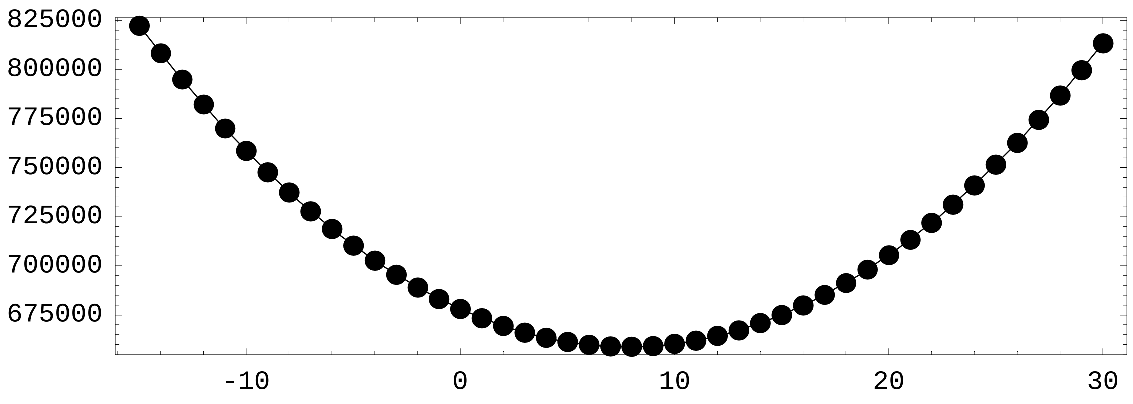
$677907. - 4910.\,x + 314.\,x^2$

```
f[x_] := 677907 - 4910 x + 314 x^2
```

```
funcplot = Plot[f[x], {x, -15, 30}, Frame → True, AspectRatio → 1 / 3, Axes → False];
```



```
Show[funcplot, plotpoints];
```



Identify the lowest point in the parabola which indicates the relative shift between performances.

```
slope = D[f[x], x]
```

```
-4910 + 628 x
```

```
N[Solve[slope == 0, x]]
```

```
{{x → 7.81847}}
```

```
(x /. Solve[D[Fit[dd, {1, x, x^2}, x], x] == 0, x])[[1]]
```

```
7.81847
```

Therefore, the 18th trial starts beating the first beat 8 milliseconds before the first trial does. Now, calculate the offsets for all other trials against the first trial.

```
diffsets =
  Table[
    Table[{x, SquareDiff[
       Transpose[ seqdata[[1]] ] [[1]] * 1000.0 ,
       Transpose[seqdata[[trial]] ] [[1]] * 1000.0 + x
     ]}, {x, -20, 20, 1} ],
   {trial, 1, 20}
  ];
```

```
offsets = Table[
     (x /. Solve[D[Fit[diffsets[[trial]], {1, x, x^2}, x], x] == 0, x])[[1]]
   , {trial, 1, 20}
  ] // Chop
```

```
{0, 2.14331, 19.551, -6.24204, 10.9363, 32.2675, 83.6911, 27.1847, -35.0478, 29.4045, 15.879,
 3.51274, -35.5064, 12.1019, 3.77389, 31.6815, 1.63694, 7.81847, -17.9459, 18.086}
```

The average shift is 20 milliseconds:

```
Mean[ Abs[offsets]]
```

```
19.7205
```

```
datato = datat;
sum = 0.0;
tempoutput = 0.0;
meansum = 0.0;
meanseq = {};
seqdatao = {};
For[i = 1, i ≤ Length[datato], i++,
   sum = offsets[[i]];
   tempoutput = {};
   llen = Length[datato[[i]]];
   For[j = 1, j ≤ llen, j++,
     tempoutput = Append[tempoutput, { sum / 1000.0, i}];
      sum += datato[[i]][[j]];
     If[i == 1, meanseq = Append[meanseq, meansum];
    meansum += meanpoints[[j]] / 1000.0; ];
    ];
   seqdatao = Append[seqdatao, tempoutput];
]
```

```
meanoffset = Mean[offsets]
```

```
10.2463
```

```
seqdata[[2]][[1]]
```

```
{0., 2}
```

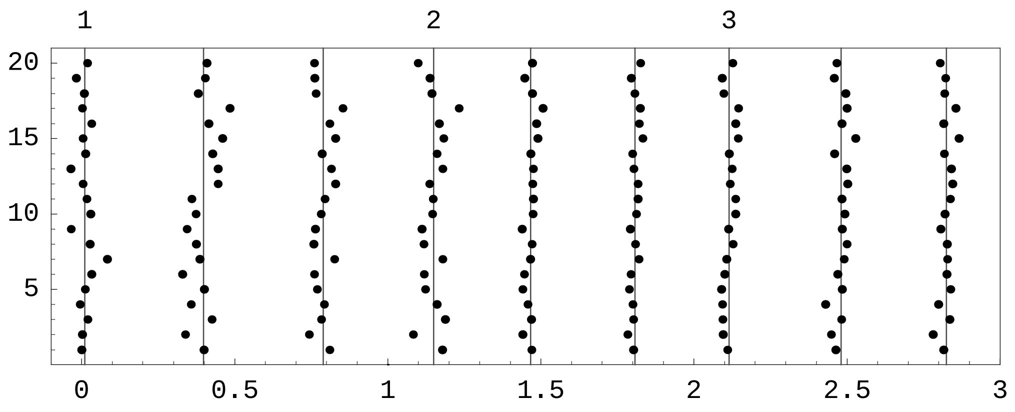
```
seqdatao[[2]][[1]]
```

```
{0.00214331, 2}
```

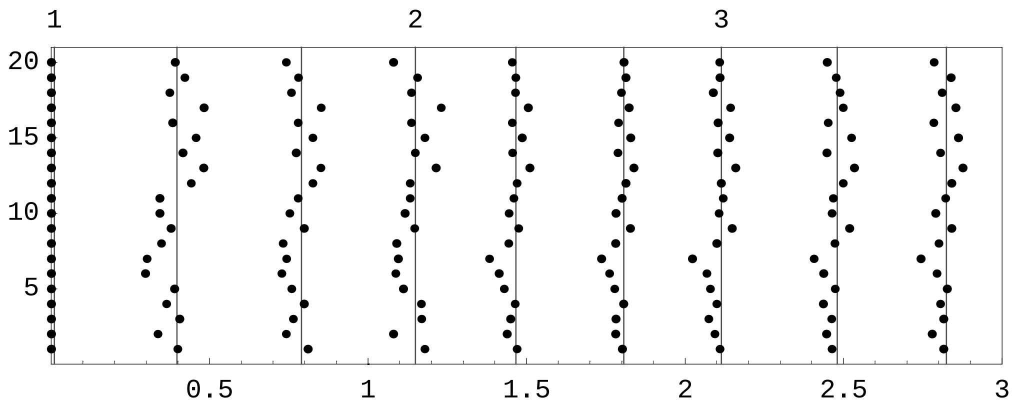
```
meanseq = meanseq + meanoffset / 1000.0;
```

```
beatlines = Map[Line[{{#, 0}, {#, 21}}] &, meanseq];
measures = Transpose[Partition[meanseq, 3]][[1]];
toplabels = Transpose[{measures, Range[Length[measures]]}];
```

```
Show[dummyplot, Graphics[
    {{Hue[0], beatlines}, PointSize[0.009], Map[Point, Flatten[ seqdatao, 1]]}],
  PlotRange → {{-.1, 3}, {0, 21}}, AspectRatio → 1 / 3,
  FrameTicks → {Automatic, Automatic, toplabels, None},
  DisplayFunction → $DisplayFunction];
```



Compare to the non-offset trial data plot which is repeated below, and notice that the points representing conducted beat locations fit to the average beat positions better in the above plot:



■ Offset adjustments to timing values to align with audio recording

Up to now the timing data for the human beats has been referenced to the first beat in the score. This beat was initially assigned 0 milliseconds. In the section above, a relative shifting of each trial was done against the first trial using a least-squares fit to coordinate the individual trials in absolute time.

In the actual recording, the music does not start at 0 milliseconds into the file, but rather starts after an initial segment of silence. In theory, the duration of this silence is easy to measure from the soundfile; however, the actual 0 point in the human beats does not necessarily coincide with the beginning of the first note, and it would be expected to be about 50 to 250 milliseconds later due to human reaction time.

Therefore, a more accurate method of determining the initial offset of the mean human beat timings into the soundfile is to use a least-squares fit to align the human beats with beats measured accurately in the audio file. First, load the beat data measured directly from the soundfile into a graphical display from an audio editing program:

```
<< audiobeats.dat
```

This data contains the most accurate measurement of the beat positions from audio events in the soundfile, probably within an accuracy of 1 millisecond, but definitely within an accuracy of 3 milliseconds which is better than the accuracy of 5 milliseconds of the input tapping method using the computer keyboard in Windows XP.

The events were chosen due to their visual clarity (mostly of the amplitude envelope), caused by accented beats, or beats starting in silence where the amplitude change is clear. The first colum in the table below is the measure number, second column is the beat in the measure, and the last column is the time in milliseconds from the start of the audio recording (which includes the initial silence at the start of the soundfile).

```
fixedpos // TableForm
```

```
1       1      2290
12      3      14888
13      3      15759
14      3      16607
18      3      21216
20      3      22965
24      3      26465
34      3      35344
37      2      37653
38      3      38844
41      1      40722
42      1      41729
45      1      44630
48      1      47579
49      1      48528
53      1      52364
56      3      56420
64      3      65260
68      1      68671
73      1      74496
74      1      76478
75      1      77938
76      1      79144
77      1      80282
86      3      90647
88      3      92436
96      3      100940
101     3      105125
102     3      105923
```

Now, calculate the beat location from the start of the piece (where beat one of measure one is absolute beat 0).

```
audiobeats = Map[{ ((#[[1]] - 1) * 3 + #[[2]] - 1), #[[3]]} &, fixedpos];
```

The first beat is not accurately performed by the human, so remove it from the calculations.

```
audiobeats = Drop[audiobeats, 1]
```

```
{{35, 14888}, {38, 15759}, {41, 16607}, {53, 21216}, {59, 22965}, {71, 26465},
 {101, 35344}, {109, 37653}, {113, 38844}, {120, 40722}, {123, 41729},
 {132, 44630}, {141, 47579}, {144, 48528}, {156, 52364}, {167, 56420}, {191, 65260},
 {201, 68671}, {216, 74496}, {219, 76478}, {222, 77938}, {225, 79144}, {228, 80282},
 {257, 90647}, {263, 92436}, {287, 100940}, {302, 105125}, {305, 105923}}
```

Extract the same beat positions from the mean beat sequence:

```
allhumanbeats = meanseq * 1000.0;

humanbeats = Map[{#[[1]], allhumanbeats[[ #[[1]] + 1 ]]} &, audiobeats]
```

```
{{35, 12382.2}, {38, 13295.9}, {41, 14155.2}, {53, 18704.7},
 {59, 20460.3}, {71, 24076.1}, {101, 32846.8}, {109, 35065.1}, {113, 36300.7},
 {120, 38295.4}, {123, 39278.3}, {132, 42164.9}, {141, 45135.}, {144, 46079.},
 {156, 49914.7}, {167, 53979.8}, {191, 62849.8}, {201, 66153.3},
 {216, 72028.9}, {219, 73981.7}, {222, 75563.3}, {225, 76797.6}, {228, 77855.3},
 {257, 88213.3}, {263, 89927.5}, {287, 98465.6}, {302, 102680.}, {305, 103505.}}
```

Seq1 and seq2 below are the two timing sequences for carefully measured beat locations in the audio dat. The first sequence are from the human beat timings which are reference from the first beat. The second sequence are from the audio and are referenced to the start of the soundfile which contains some initial silence.

```
seq1 = Transpose[humanbeats][[2]]
```

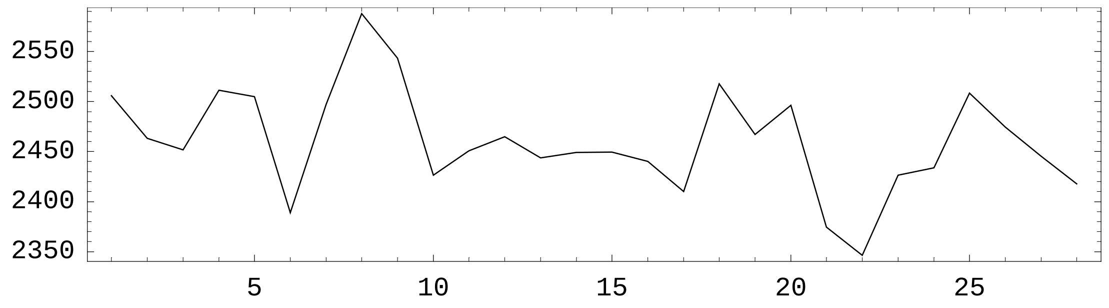
```
{12382.2, 13295.9, 14155.2, 18704.7, 20460.3, 24076.1, 32846.8, 35065.1, 36300.7, 38295.4,
 39278.3, 42164.9, 45135., 46079., 49914.7, 53979.8, 62849.8, 66153.3, 72028.9,
 73981.7, 75563.3, 76797.6, 77855.3, 88213.3, 89927.5, 98465.6, 102680., 103505.}
```

```
seq2 = Transpose[audiobeats][[2]]
```

```
{14888, 15759, 16607, 21216, 22965, 26465, 35344, 37653, 38844,
 40722, 41729, 44630, 47579, 48528, 52364, 56420, 65260, 68671, 74496,
 76478, 77938, 79144, 80282, 90647, 92436, 100940, 105125, 105923}
```

Here are the time difference between the human beats and the audio beats which range between 2350 and 2550 milliseconds. The average if this difference will be used to shift the human beats to their positions in the audio file.

```
ListPlot[seq2 - seq1, PlotJoined → True,
  PlotRange → All, Frame → True, Axes → False, AspectRatio → 1 / 4];
```



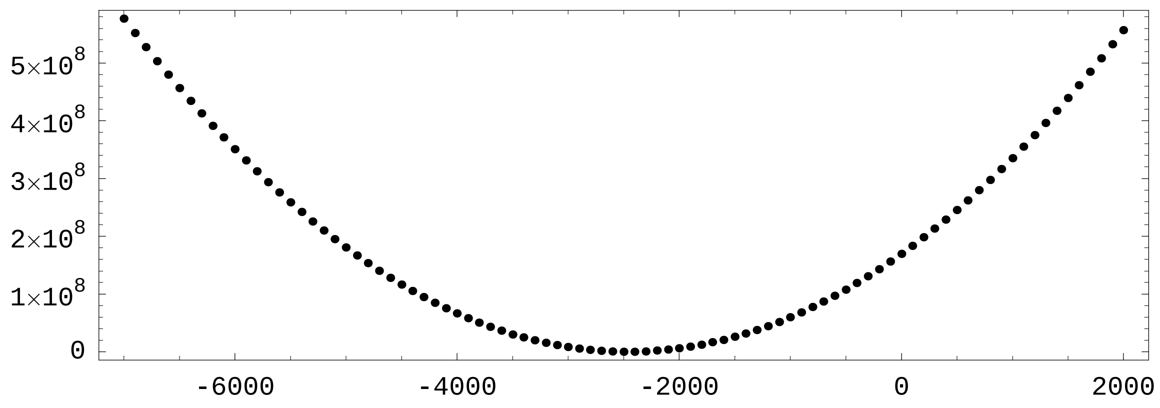
```
Mean[seq2 - seq1]
```

```
2460.61
```

Now, let's calculate the needed shift (2460.61 ms) the more difficult way just to check that this is the best shift to minimze the offset between the audio and human beats.

```
SquareDiff[x_, y_] := Apply[Plus, (x - y) * (x - y)];
```

```
plotpoints =
  ListPlot[Table[{x, SquareDiff[ seq1,  seq2 + x]}, {x, -7000, 2000, 100} ],
   Frame → True, PlotRange → All, PlotStyle → PointSize[0.008],
   AspectRatio → 1 / 3, Axes → False];
```



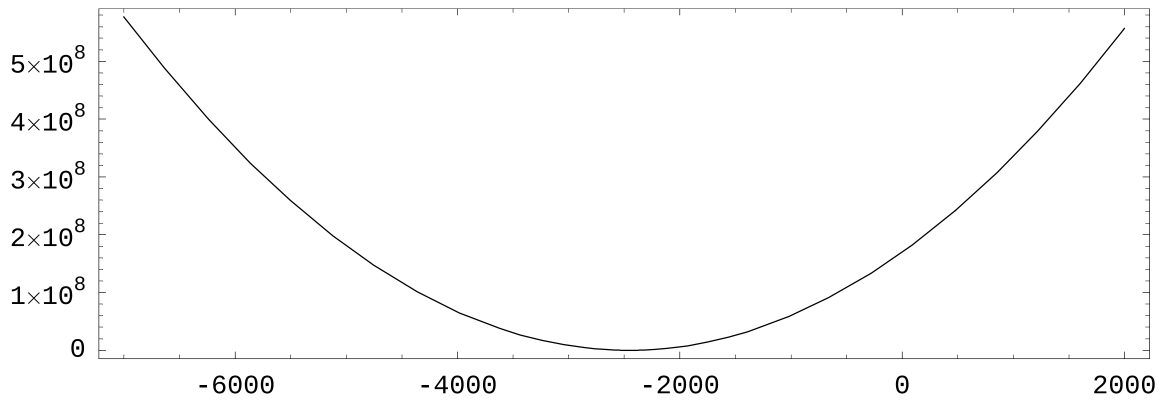
```
dd = Table[{x, SquareDiff[ seq1 , seq2 + x]}, {x, -7000, 2000, 100} ];
```

```
func = Fit[dd, {1, x, x^2}, x]
```

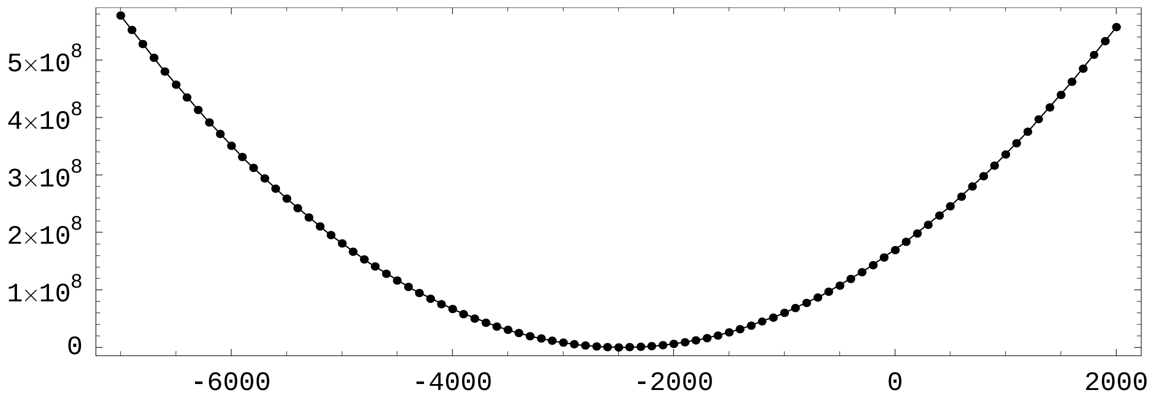
$1.69601 \times 10^8 + 137794. x + 28. x^2$

```
f[x_] := 169601000 + 137794 x + 28 x^2
```

```
funcplot = Plot[f[x], {x, -7000, 2000}, Frame → True, AspectRatio → 1 / 3, Axes → False];
```



```
Show[funcplot, plotpoints];
```



```
slope = D[f[x], x]
```

```
137794 + 56 x
```

```
N[Solve[slope == 0, x]]
```

```
{{x → -2460.61}}
```

```
beatoffset = - 1.0 * (x /. Solve[D[Fit[dd, {1, x, x^2}, x], x] == 0, x])[[1]]
```

```
2460.61
```

```
newseq1 = seq1 + beatoffset
```

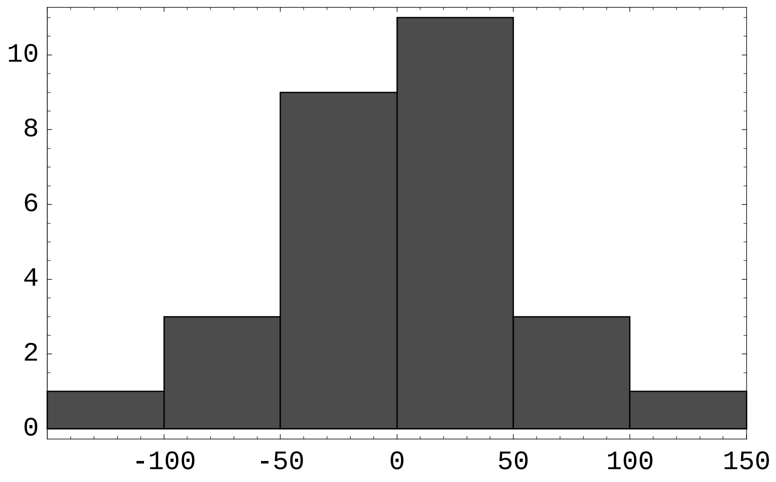
```
{14842.9, 15756.5, 16615.9, 21165.3, 22920.9, 26536.8, 35307.5, 37525.8, 38761.3, 40756.,
 41739., 44625.6, 47595.7, 48539.6, 52375.3, 56440.5, 65310.5, 68613.9, 74489.5,
 76442.4, 78023.9, 79258.3, 80316., 90674., 92388.1, 100926., 105140., 105966.}
```

```
newseq1 - seq2
```

```
{-45.1482, -2.49821, 8.85179, -50.6982, -44.0982, 71.7518,
 -36.5482, -127.248, -82.6982, 34.0018, 9.95179, -4.44821, 16.6518,
 11.6018, 11.3018, 20.4518, 50.4518, -57.0982, -6.49821, -35.6482,
 85.9018, 114.252, 33.9518, 26.9518, -47.8982, -13.7982, 15.4518, 42.8018}
```

```
Histogram[Flatten[beatdiff], AspectRatio → 1 / 4, Frame → True,
  HistogramRange → {-100, 100}, BarEdges → False, HistogramCategories → 25];
```

```
Histogram[newseq1 - seq2, Frame → True];
```



On the average, the measured beat positions is 40 milliseconds away from the measured position in the audio file. The worst case for the beat difference for the measured audio beats is 127 milliseconds.

```
Mean[Abs[newseq1 - seq2]]
```

```
39.5948
```

```
Max[Abs[newseq1 - seq2]]
```

```
127.248
```

■ Output data

Store the data analysis information into a text file for future reference:

```
finalbeattimings = allhumanbeats + beatoffset // Round;
outputconfidence = confidence // Round;
outputmeanbeat = meanseq * 1000.0 // Round;
outputmax = Map[Max, data];
outputmin = Map[Min, data];
outputstandarddeviation = Map[(StandardDeviation /. #) &, eachbeatstats];
```

```
finaldata = Map[Flatten,
   Transpose[{finalbeattimings, durationmean,
     outputmin, outputmax, outputconfidence, outputstandarddeviation}]];
```

The full analysis output file contains an information entry for each beat (except the last beat in the input) with seven numbers on a line:
(1) the expected position of the beat in the audio file in milliseconds
(2) the average beat duration for all trials
(3) the minimum beat duration in all trials
(4) the maximum beat duration in all trials
(5) the minimum location of the true average beat location with 95% confidence
(6) the maximum location of the true average beat location with 95% confidence
(7) the standard deviation of beat durations for all trials

Here is the information for the first beat in the Mazurka:

```
finaldata[[1]]
```

```
{2471, 387.25, 297, 482, 363, 412, 52.4313}
```

```
Export["output.dat", finaldata];
```