

# Pitch Perception: Missing fundamentals and Shepard tones

Ctrl-A followed by Shift-Enter to Evaluate all cells to ensure proper sound

Please Ignore code snippets and read text highlighted in blue and figures only.

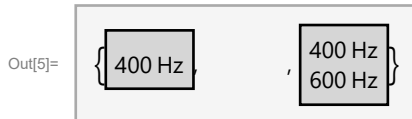
Init code

## Missing fundamental

Click Rectangular Buttons in notebook to hear sound

```
In[5]:= Panel[{playbutton[{{400, 1}}], "\t", playbutton[{{400, 1}, {600, .25}}]},  
  Style["Example 1", Background → Lighter[Cyan, .8]]]
```

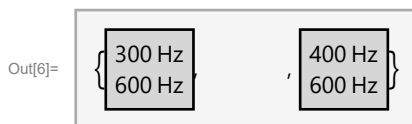
Example 1



Each button represents a sound which is a mixture of pure sine waves with the given frequencies. Introducing a *higher* frequency of 600 Hz surprisingly makes the perceived pitch sound lower.

```
In[6]:= Panel[{playbutton[{{300, 1}, {600, 1}}], "\t", playbutton[{{400, 1}, {600, .4}}]},  
  Style["example 2", Background → Lighter[Cyan, .9]]]
```

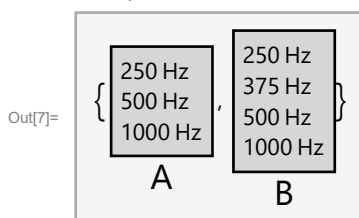
example 2



Replacing 300 Hz by the *higher* 400 Hz again, surprisingly makes the perceived pitch sound *lower*

```
In[7]:= Panel[{Labeled[playbutton[{{250, 1.}, {500, 0.25}, {1000, .08}}],  
  Style["A", 18]],  
  Labeled[playbutton[{{250, 1.}, {375, 1.75}, {500, 1.5}, {1000, .25}}], Style["B", 18]],  
  Style["example 3", Background → Lighter[Cyan, .9]]]
```

example 3



A: The first 3 overtones of a 250 Hz pitched sound. Therefore A is perceived to have a pitch of 250 Hz.

B: By introducing 350 Hz which is *higher* than 250 Hz we might expect B to sound *higher* in pitch than A. Again, surprisingly B is perceived to have a pitch one octave *lower* than A.

As reference, we provide the sound of pure sine waves:

```
In[8]:= DynamicModule[{fa},
  fa = {{250, 1}, {375, 2}, {500, 2.5}, {1000, .5}};
  Map[playbutton[{#}] &, fa]
]
```

```
Out[8]= {250 Hz, 375 Hz, 500 Hz, 1000 Hz}
```

## Frequency domain discussion

Any periodic waveform consists of frequencies which are integer multiples

$$1 f_0 \ 2 f_0 \ 3 f_0 \ 4 f_0 \dots$$

of a fundamental frequency  $f_0$ . A naturally produced sound from a linear instrument will have a dominant contribution at the fundamental  $f_0$  and smaller contributions in the overtones  $2 f_0 \ 3 f_0 \ 4 f_0 \dots$ . However synthetic waveforms where the  $f_0$  contribution is missing may be created. This is quite obvious in Examples 1 and 2 where the second tone has a missing fundamental of 200Hz.

Example 3 has the structure of:

$$\begin{array}{ll} 1 f_0 \ 2 f_0 \ 3 f_0 & \text{with } f_0 = 250 \text{ for (A)} \\ \text{--- } 2 f_0 \ 3 f_0 \ 4 f_0 \dots 8 f_0 & \text{with } f_0 = 125 \text{ for (B)} \end{array}$$

which shows that B has the missing fundamental of 125 Hz. (It is also missing harmonics 5,6,7)

A standard explanation for why such synthetic periodic waveforms with a missing fundamental like B are still perceived to have a pitch of  $f_0$  is that the ear somehow fills in for this missing fundamental.

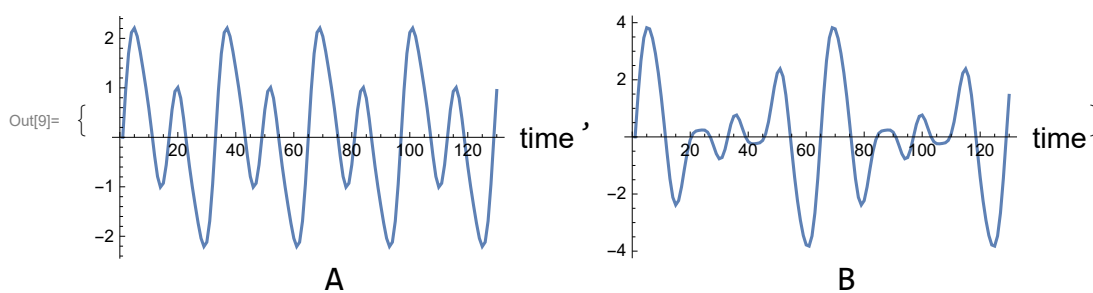
## Time domain discussion

But it is the time domain that reveals exactly why B is perceived to have a pitch one octave lower than A, in example 3 and similarly in example 1 and 2

```

In[9]:= Module[{ts1, ts2, is = ImageSize -> 250},
  ts1 = timeseries[{{250, 1.}, {500, 1.5}, {1000, .25}}];
  ts2 = timeseries[{{250, 1.}, {375, 1.75}, {500, 1.5}, {1000, .25}}];
  {Labeled[
    ListLinePlot[Take[ts1, 130], is, AxesLabel -> {Style["time", 15], ""}], Style["A", 18]],
    Labeled[ListLinePlot[Take[ts2, 130], is, AxesLabel -> {Style["time", 15], ""}],
      Style["B", 18]]}]
]

```



Let us look the waveforms for sounds A and B of example 3 in Figure above. We observe that:

Waveform A repeats every 32 samples or 250 times a second - its perceived pitch.

Waveform B repeats every 64 samples or 125 times a second - its perceived pitch.

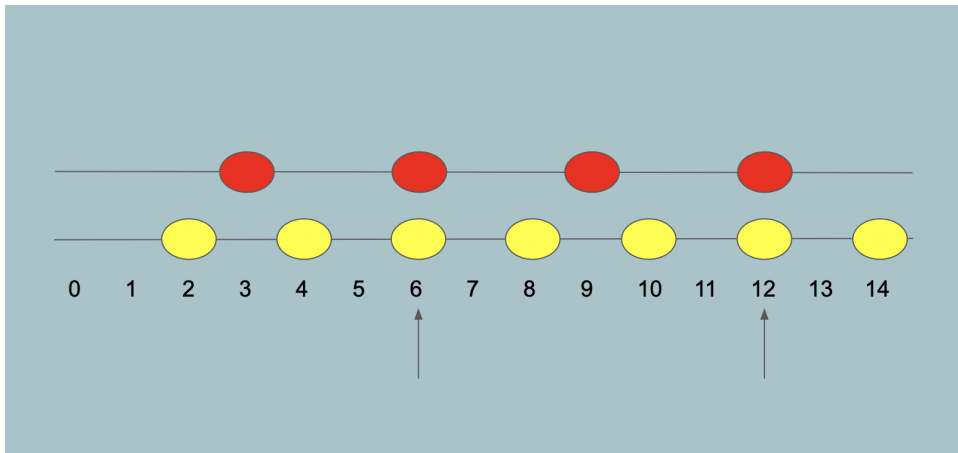
Suddenly, everything is clear. Introducing the faster 350Hz component into A makes the new waveform B repeat at a *slower* rather than faster rate! Why is this? Let us look more closely. As we saw the addition of 375 Hz transforms the harmonic series

into

$1 f_0$	$2 f_0$	$3 f_0$	with $f_0 = 250$ for (A)
$2 f_0$	$3 f_0$	$4 f_0$	with $f_0 = 125$ for (B)

Thus B is just another harmonic series except with a missing fundamental. Removing a fundamental frequency from a waveform cannot change its periodic nature nor indeed the frequency at which it repeats which is still  $f_0$  cycles a second.

The metaphor in the figure below may help drive the point home. A red light flashed every 3 seconds. We combine this with a yellow light that flashes faster - every 2 seconds. But the combined pattern “flashes” (i.e. repeats) *slower!* once every 6 seconds.



Measuring the peak to peak distance in autocorrelation functions of the waveforms is a more principled way of seeing this.

```
In[10]:= Module[{ts1, ts2, ac1, ac2},
  ts1 = timeseries[{{250, 1.}, {500, 1.5}, {1000, .25}}];
  ts2 = timeseries[{{250, 1.}, {375, 1.75}, {500, 1.5}, {1000, .25}}];
  ac1 = ListCorrelate[ts1, ts1, {1, -80}, 0];
  ac2 = ListCorrelate[ts2, ts2, {1, -80}, 0];
  Print["\t\t\t\t", Style["Autocorrelation functions", 18]];
  GraphicsRow[{
    ListPlot[ac1, Filling -> Axis, AxesLabel -> {"time", ""},
      Epilog -> Text[Style["A", 18], Scaled[ {.5, .1} ]]],
    ListPlot[ac2, Filling -> Axis, AxesLabel -> {"time", ""},
      Epilog -> Text[Style["B", 18], Scaled[ {.5, .1} ]]]
  ]
]
```

### Autocorrelation functions

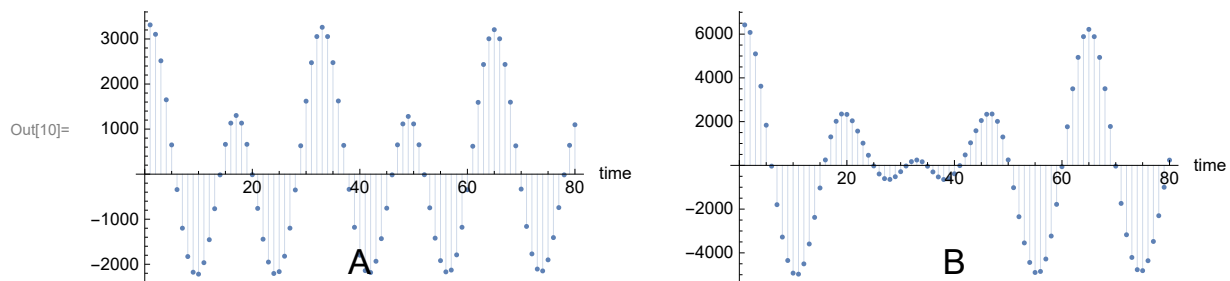


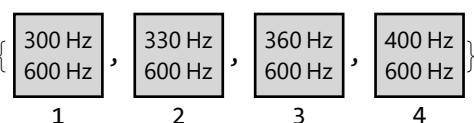
Figure above shows the autocorrelation functions for waveforms A and B of example 3  
The peak to peak distance in the autocorrelation functions confirms that the waveforms repeat

every 32 and 64 samples respectively i.e. with frequencies of 250 and 125 Hz resp as was already evident by direct inspection of the waveforms.

Satisfying though our time domain explanation is, one nevertheless is left with the nagging feeling: that surely pitch should have *something* to do with frequency spectrum? For example, of two spectra, should not the one whose frequency spectrum's center-of-mass is larger also sound higher pitched? In spite of us showing that this is not the case in all that has preceded we now demonstrate there is indeed some merit to this intuition also! Let us return to example 2. Instead of making a sudden change from 300 Hz to 400 Hz let us creep gradually up onto 400 Hz.

```
In[11]:= {Labeled[playbutton[{{ 300, 1}, {600, .4}}], 1],
          Labeled[playbutton[{{ 330, 1}, {600, .4}}], 2],
          Labeled[playbutton[{{ 360, 1}, {600, .4}}], 3],
          Labeled[playbutton[{{ 400, 1}, {600, .4}}], 4]
        }
```

```
Out[11]:= {
  { 300 Hz, 600 Hz }, { 330 Hz, 600 Hz }, { 360 Hz, 600 Hz }, { 400 Hz, 600 Hz }
}
```



From left to right each successive note sounds decidedly higher in pitch. Listen to the sequence 1-2-3-4 and then 1 again. To the author at least, now 1 sounds lower than 4, in contradiction to what was perceived in example 2 above. The reader is invited to return to example 2 and convince herself that the corresponding notes there are exactly the same. It appears that there is a sort of hysteresis, a kind of memory as it were, in pitch perception, where we are influenced by the history of what we have recently heard. While this conclusion of ambivalence may be disappointing, it presents one with the possibility of exploiting it in order to lead to some striking aural illusions. Here is the idea.

## Cyclic tones (Shepherd tones)

Imagine continuously transforming the set of frequencies

$$1 f_0 \ 2 f_0 \ 3 f_0 \ 4 f_0 \dots$$

into the set of frequencies

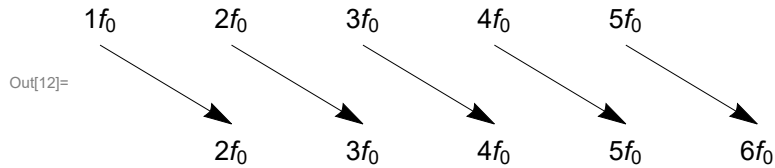
$$2 f_0 \ 3 f_0 \ 4 f_0 \dots$$

as indicated schematically by the arrows in the Figure below.

```

In[12]:= Module[{a, b, fs = 14, arr},
  a = Table[Text[Style[ToString[n] <> "f0", fs], {n, 1}], {n, 1, 5}];
  b = Table[Text[Style[ToString[n] <> "f0", fs], {n, 0}], {n, 2, 6}];
  arr = Table[Arrow[{n, 1 - .2}, {n + 1, 0 + .2}], {n, 1, 5}];
  Graphics[{a, b, arr}]

```



Each small step in the direction of the arrows must surely increase the perceived pitch. Yet our final configuration of frequencies is identical to the starting one except that  $f_0$  is missing. But this is just the missing fundamental! So we might expect the final pitch to be the same as the starting pitch! Incredibly enough, this illusion is real. There are many ways of implementing it but the above remains the central idea.

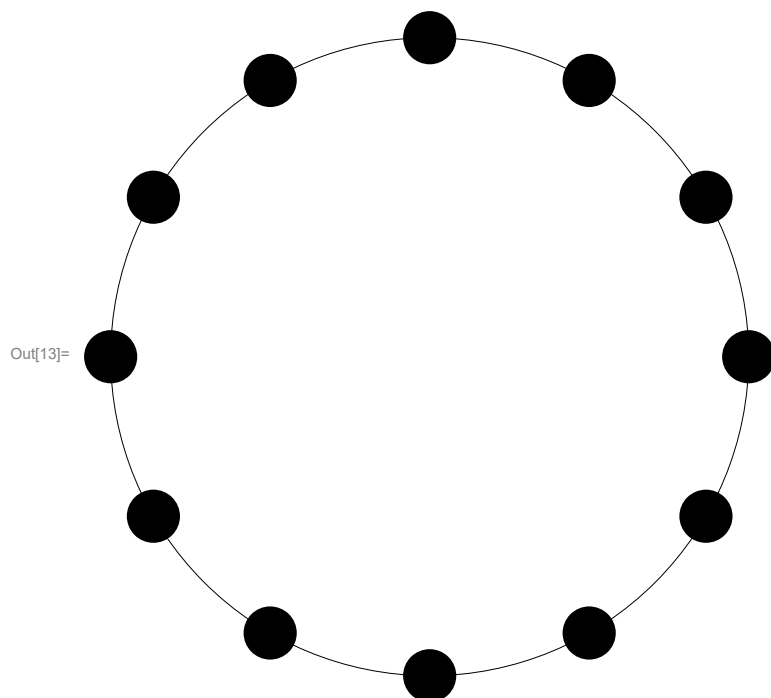
Instead of a purely continuous transformation we choose to divide up our transformation into 12 equal logarithmic steps so as to give a sense of moving in musical semi tones. The listener is invited to decide which of these 12 notes is the starting one and which the ending in the sense indicated in the above figure (click on dot). It is difficult to tell. The illusion, of course, is that each successive note clockwise is always increasing!

Click Dots to hear sound

```

In[13]:= DynamicModule[{ $\theta_1 = \text{Pi}$ ,  $\theta_2 = 3 \text{ Pi}$ ,  $d\theta = 2 \text{ Pi} / 12$ ,  $\theta_{\text{posn}}$ , pt, keys},
   $\theta_{\text{posn}} = \text{Table}[\theta_1 + n d\theta, \{n, 0, 12\}]$ ;
  pt = Table[{Cos[ $\theta$ ], Sin[ $\theta$ ]}, { $\theta$ ,  $\theta_{\text{posn}}$ ]];
  keys = Table[With[{p = pt[[i]], s = scale[[i]]},
    EventHandler[Point[p], {"MouseClicked"  $\Rightarrow$  EmitSound[s]}]], {i, 1, 12}];
  Dynamic[Graphics[{Circle[{0, 0}, 1], PointSize[.08], keys}], SaveDefinitions  $\rightarrow$  True]
]

```



Our approach is somewhat differently motivated from the standard approach ascribed to Mr. Shepard that does not rely on the missing fundamental *per se*. We implemented this other point of view<sup>1)</sup> also (below) for comparison. This method simply relies on a fade-in of the rising lower frequencies and a fade-out of the rising upper ones. Each tone consist of 3 frequencies an octave apart as shown in the spectrogram. The changing line-spectrum in the animation below clearly reveals how these three frequencies are gradually faded in and faded out to help achieve the magic trick.

```

In[14]:= Hyperlink["1) wiki", "https://en.wikipedia.org/wiki/Shepard_tone"]

```

Out[14]= 1) wiki

### An Infinite Keyboard

Click on any key to hear sound.

“do” “re” “...labels guide you to play a major scale

Top Left Tab selector allows you to select any desired key as the tonic “do”

Out[ ]=

