JOURNAL OF
EMERGING INVESTIGATORS

Article

Open string vibrato: does it exist?

Puyuan Ge[1], Li Zhang[2]

[1] Eton College, Windsor, Berkshire, UK

[2] University of Leeds, Woodhouse Lane, Leeds, West Yorkshire, UK

SUMMARY

Vibrato, defined as a rapid and subtle oscillation in pitch, is a technique that is commonly used by musicians to add expression and colour to notes. However, on stringed instruments, there are certain notes (open string notes) on which it is impossible to perform the technique. Without vibrato, they can sound angular and unpleasant, especially when juxtaposed against other notes played with vibrato. String players therefore use an alternative to achieve the same vibrato effect on the open string — a technique referred to as "open string vibrato". While the technique is widely used, it is unknown how much of a physical effect it has on the sound waves produced, if any at all. The purpose of this study is to analyse open string vibrato using a statistical approach to provide evidence to characterize the physical effect of the technique, and then compare it to normal vibrato. We hypothesised that it would have a noticeable and measurable effect on the sound waves produced because of the technique's widespread usage. To test this, notes, with and without either open string vibrato or normal vibrato, were recorded on the violin. We analyzed the audio recordings using a computational and statistical approach. The results of the study partially agreed with our hypothesis: while the technique has an observable physical effect on the sound waves, the effect is weaker than expected. We concluded that open string vibrato does work, but has quite a subtle effect, and thus should only be used when there is no other option.

INTRODUCTION

For almost every piece of music there exists a myriad of unique interpretations. There are numerous ways in which musicians create expression within their performances. Vibrato is one such technique. It is widely used to add expression to vocal and instrumental music through continuous and periodic oscillation of a sound's frequency to produce a pleasing mellowness and richness of tone (1, 2). Without vibrato, a sound could be perceived as sharp, angular, or unpleasant, especially if it is juxtaposed by other notes that have vibrato.

Open-string vibrato, a vibrato technique used on stringed instruments. In bowed stringed instruments (violin, viola, cello, double bass, etc.), sound is produced by pulling the bow across one (or sometimes more) of the strings, causing it to vibrate. The pitch is varied by pressing the string against the fingerboard in different positions with the fingers on the player's left-hand to modulate the length of the string that is able to vibrate. This technique changes the frequency of the vibration of the string, as the length of the vibrating string is inversely proportional to the frequency of sound produced by Mersenne's First Law (Eqn 1):

$$f_0 \propto \frac{1}{L}$$

where f_0 is the natural/fundamental frequency of the sound produced, and L is the length of string which is vibrating (3, 4). The frequency of the sound wave produced affects the pitch of the note with a base-2 logarithmic relationship — in general, the greater the frequency, the higher the pitch (3, 4).

An open string note refers to the note produced by a fixed string length, played without pressing the string against the fingerboard in any place to change its length (11). In this study, we used a violin. For the sake of clarity in the rest of the paper, on a violin, the open string notes are G3 (196 Hz), D4 (293.7 Hz), A4 (440 Hz), and E5 (659.3 Hz), played on the G, D, A, and E strings, respectively (11). For the rest of this paper, all frequency-pitch conversions use A4 at 440 Hz as a benchmark.

Normally, vibrato is performed by quickly rocking the finger on the string back and forth while maintaining pressure, rapidly and periodically lengthening, and shortening the vibrating part of the string, causing a regular oscillation in the frequency of the sound (12). However, such an effect is not possible on an open string, as the length of the vibrating string is fixed (11, 12).

Here, the problem arises that a note played without vibrato, especially when contrasted against other notes played with vibrato, can be perceived unpleasantly. This is a problem for the lowest open string note, because, by definition, there are no lower strings to play on. Moreover, most often, performing normal vibrato in the way described in the example requires an inconvenient shift in finger position, which poses another problem, where musicians may not be able to use conventional vibrato.

To solve this problem, musicians commonly use open string vibrato. To perform this technique, when playing an open string note, the player places their finger on the position for pitch one octave above (the second harmonic overtone of the original open string note). The player then performs vibrato on this one-octave higher note. Consequently, the frequency of the sound heard from the open string seems to oscillate subtly, as if there is vibrato on that note. One of the most famous examples where this this technique is used is the opening of the first movement of Bruch's 1st violin concerto in G minor (Op. 26). The piece begins with a long, sustained G3 note, which is impossible to perform using traditional vibrato

https://doi.org/10.59720/24-001

(5). Therefore, most soloists performing this concerto will use the open string vibrato. They place a finger on the D-string, in the position that would produce a G4 (392 Hz) note when played. They then pull the bow across the G-string, playing the open G string while performing vibrato with their left hand on the D-string. An example of this can be found in the 2016 recording of renowned violinist Hillary Hahn performing the concerto with the Frankfurt Radio Symphony Orchestra (6). For convenience, in this paper, this technique is referred to as "open string vibrato". Although it is a widely known technique among musicians, there is little scientific research supporting its existence as a distinct physical phenomena or providing an explanation for the similar sound produced in comparison to traditional vibrato.

Using a mathematical and computational approach, we investigated a method of quantifying vibrato, the physical existence of the effect of open string vibrato and compared open string vibrato to normal vibrato. We first hypothesized that the method proposed for quantifying vibrato is effective, showing a strong positive correlation between presence of vibrato and the quantity. Then secondly, we hypothesized that the open string vibrato technique has a measurable effect on the sound waves produced, showing a correlation between presence of open string vibrato and the amount of vibrato measured in the sound quantified previously. Finally, we compared the characteristics of open string vibrato with its conventional counterpart. Traditional vibrato is characterised by three factors: the amount of pitch variation ("vibrato extent"), the speed at which the pitch is varied ("vibrato rate/vibrato period"), and the shape of its pitch-time curve (6). We compared the two forms of vibrato according to each of these factors. We hypothesized that the open string vibrato effect produces an effect similar in nature to conventional vibrato. This work could be useful to musicians as well, by not only scientifically showing whether or not the open string vibrato technique actually works, but also by comparing to traditional vibrato.

RESULTS

To quantitatively determine whether open string vibrato has an effect on the sound waves and then subsequently compare its effect to that of normal vibrato, we used a statistical measure of dispersion (standard deviation) to quantify by how much the pitch varied, and thus how much vibrato there was on a note. To ensure that this quantification of vibrato was valid, we tested normal vibrato against notes without vibrato. The results were positive: in every test, and around each harmonic frequency, the spread of pitch in notes played with vibrato was consistently much greater than the spread of

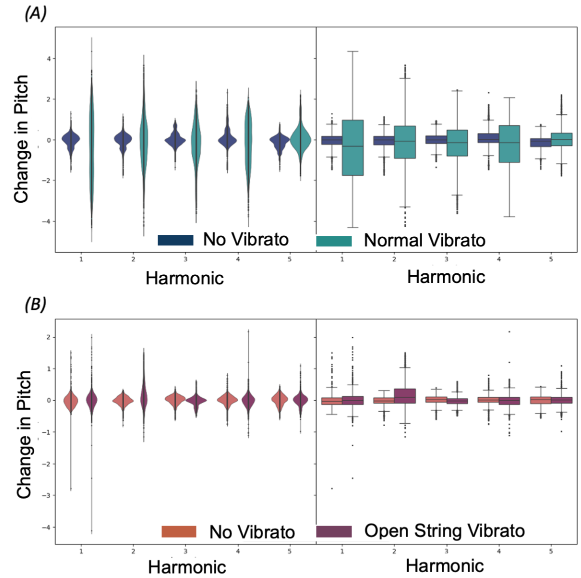

Figure 1: Comparison of the distribution of pitch in notes played with and without vibrato for different types of vibratos. A) Violin plot (left) and box and whisker plot (right) showing the distribution of pitch around the first five harmonic frequencies (n=5). Fingered notes were played with either no vibrato or normal vibrato. B) Violin plot (left) and box and whisker plot (right) showing the distribution of pitch around the first five harmonic frequencies (n=5). Open string notes were played with either no vibrato or open string vibrato.

pitch in notes played without any vibrato (**Table 1A, Figure 1A**). On average, the spread of pitch in normal vibrato was 405% more than the spread in notes without vibrato in the first harmonic, 199% more in the second harmonic, 273% more in the third harmonic, 233% more in the fourth harmonic, and 97.5% more in the fifth harmonic. Taking no vibrato as 0 and normal vibrato as 1, we calculated the Pearson correlation coefficient (PCC). The linear relationship between vibrato and pitch spread had r = 0.789. Thus, there was strong positive correlation between the presence of normal vibrato and the spread of pitch in the note. Therefore, the devised method is an effective way of quantifying vibrato, and thus we used it for the rest of the research.

To determine whether open string vibrato had any impact on the pitch spread of a note, we tested open string vibrato against open string notes played without vibrato. The spread of pitch was greater in notes with open string vibrato than it was in notes without any vibrato: on average 25.8% greater in the first harmonic, 180% greater in the second harmonic, 15.8% greater in the third harmonic, 24.3% greater in the

		(a) Spread of Regulated Frequency (Normal Vibrato)						(b) Spread of Regulated Frequency (Open String Vibrato)					
Harmonic	Vibrato	Test 1	Test 2	Test 3	Test 4	Test 5	Average	Test 1	Test 2	Test 3	Test 4	Test 5	Average
1st	With	1.69	1.76	1.55	1.63	1.5	1.626	0.22	0.3	0.21	0.15	0.29	0.234
	Without	0.28	0.32	0.41	0.35	0.25	0.322	0.19	0.19	0.17	0.2	0.18	0.186
2nd	With	1.16	0.90	1.02	0.81	1.25	1.028	0.33	0.25	0.38	0.42	0.44	0.364
	Without	0.28	0.32	0.49	0.33	0.3	0.344	0.18	0.14	0.11	0.08	0.14	0.13
3rd	With	0.94	1.07	0.73	1.21	1.01	0.992	0.13	0.13	0.12	0.11	0.17	0.132
	Without	0.16	0.25	0.32	0.36	0.24	0.266	0.15	0.13	0.11	0.06	0.12	0.114
4th	With	1.05	1.16	1.06	1.04	0.95	1.052	0.2	0.14	0.15	0.16	0.22	0.174
	Without	0.25	0.26	0.5	0.46	0.11	0.316	0.2	0.15	0.13	0.08	0.14	0.14
5th	With	0.75	0.42	0.27	0.28	0.63	0.47	0.13	0.14	0.15	0.16	0.23	0.162
	Without	0.17	0.29	0.35	0.26	0.12	0.238	0.19	0.16	0.12	0.07	0.14	0.136

Table 1: Spread of pitch in fingered notes played with and without normal vibrato, rounded to the nearest 0.01.

JOURNAL OF
EMERGING INVESTIGATORS
https://doi.org/10.59720/24-001

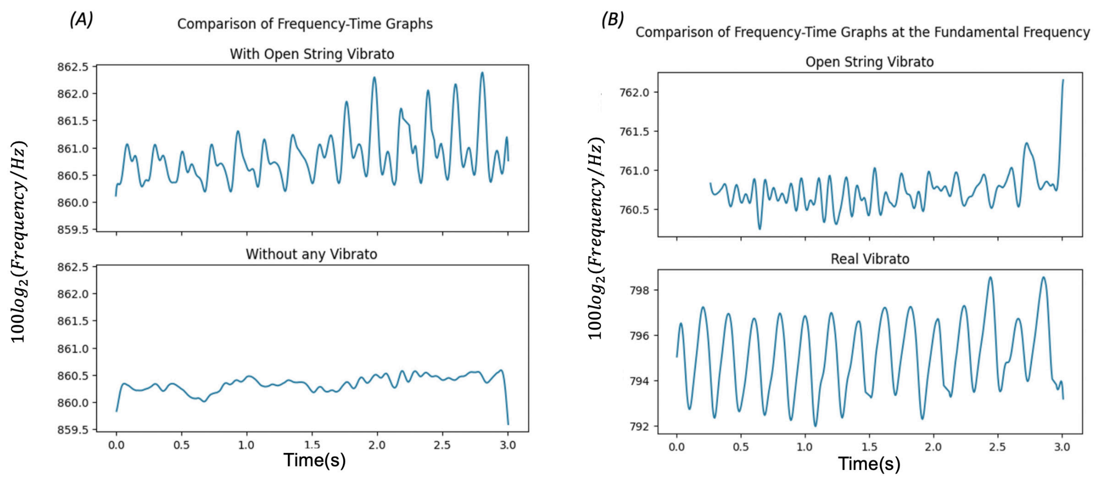

Figure 2: Pitch-time curves of notes played with and without different types of vibratos. A) Line graphs showing pitch-time curves of notes with open string vibrato (top) and without any vibrato (bottom) (n=5). Open string notes were recorded and analysed with either no vibrato or open string vibrato. B) Line graphs showing pitch-time curves of notes with open string vibrato (top) and with normal vibrato (bottom) (n=5). Open string notes were recorded and analysed with open string vibrato and fingered notes were recorded and analysed with normal vibrato.

fourth harmonic, and 19.1% greater in the fifth harmonic (**Figure 1B**, **Table 1B**). The difference in spread was quite small, and while on average it is true that the spread of pitch was greater in notes with open string vibrato than it was in notes without any vibrato, there were several instances where the spread of pitch in the non-vibrato notes was greater (e.g., test 4, 1st harmonic, **Table 1B**). Taking open string vibrato as 1 and no vibrato as 0, the calculated Pearson correlation coefficient was r = 0.445. This shows that open string vibrato has a weaker, yet still positive, correlation with the spread of pitch in a note. The greatest spread of pitch in open string vibrato was around the second harmonic (**Figure 1B**). On average, it had 180% greater spread of pitch than its control group counterpart, which is in stark contrast to the other harmonics, which had lower increases in spread, on average 21.25%, an almost ninefold difference in percentage increase. This was in contrast to normal vibrato where the harmonic with greatest spread of pitch was the first harmonic (**Figure 1A**). The pitch-time curves of open string vibrato and non-vibrato notes provide a clear qualitative insight into the effect of open string vibrato. From there it is clear to see that, in open string vibrato notes, the distribution of pitch in the sound waves is more dispersed, showing that there is more of a vibrato effect (**Figure 2A**).

To get a sense of how much difference open string vibrato made to a note, we compared the spread of pitch in fingered notes played without vibrato (a benchmark 'standard' tone produced by the violin) and open string notes played without any vibrato. The spread was consistently greater in fingered notes without vibrato, showing a 73% increase in the first harmonic, a 165% increase in the second harmonic, a 133% increase in the third harmonic, a 126% increase in the fourth harmonic, and a 75% increase in the fifth harmonic (**Figure 3B**, **Table 1A, Table 1A**). Furthermore, the spread of pitch in open string vibrato was far lower than it was in real vibrato, being 595% higher in the first harmonic, 182% in the second harmonic, 652% in the third harmonic, 505% in the fourth harmonic, and 190% in the fifth harmonic (**Figure 3A**, **Table 1A, Table 1B**). Comparing the spread of pitch in non-vibrato fingered notes and open string vibrato notes, we see that the spread of pitch in the non-vibrato fingered note was higher in the first, third, fourth and fifth harmonic overtones, but slightly lower in the second overtone, where the spread of pitch was higher in the open string vibrato note. The spread of pitch in the open string vibrato was, in the five harmonics, respectively: 27.3% less, 5.81% more, 50.4% less, 44.9% less, and 31.9% less (**Figure 4**, **Table 1A, Table 1B**). These percentage differences in spread are less than those between

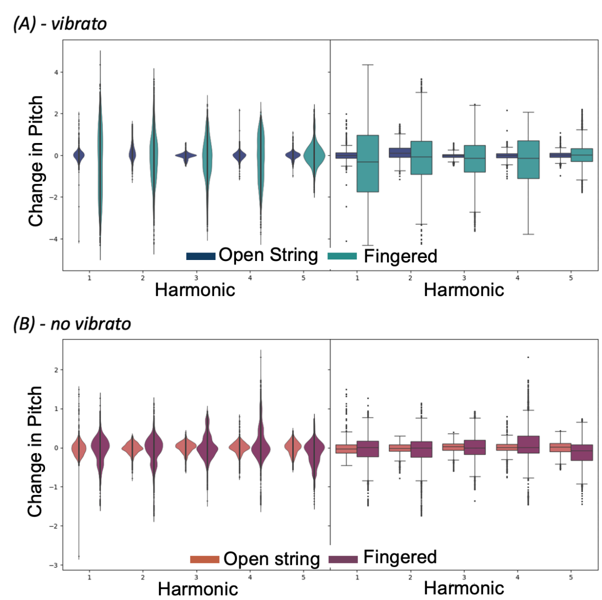

Figure 3: Comparison of pitch distribution in open string notes and fingered notes, with and without open string vibrato and normal vibrato respectively. A) Violin plot (left) and box and whisker plot (right) showing the distribution of pitch around the first five harmonic frequencies (n=5). Notes were played with either normal vibrato or open string vibrato. B) Violin plot (left) and box and whisker plot (right) showing the distribution of pitch around the first five harmonic frequencies (n=5). Notes played were either open string notes with no vibrato or fingered notes with no vibrato.

https://doi.org/10.59720/24-001

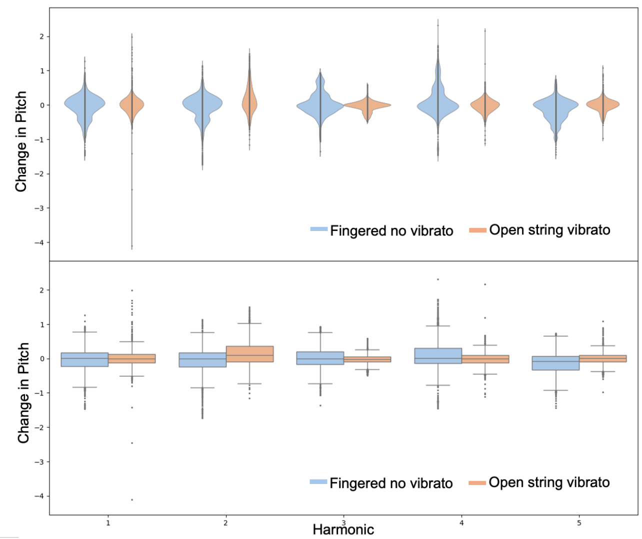

Figure 4: Comparison of the distribution of pitch in open string notes played with open string vibrato with the distribution of fingered notes with no vibrato. Violin plot (top) and box and whisker plot (bottom) showing the distribution of pitch around the first five harmonic frequencies (n=5). Notes were either fingered notes with no vibrato or open string notes played with open string vibrato.

the previously compared classes (open string vibrato notes, open string notes, and non-vibrato fingered notes), suggesting that these two classes (non-vibrato fingered notes and open string vibrato notes) are the most similar to each other.

After we confirmed the existence of open string vibrato by testing its pitch spread, we characterised it by comparing it to normal vibrato, according to three factors: extent, rate, and shape. Average vibrato extent is a slightly different measure from pitch spread discussed previously. Pitch spread is a standard deviation calculation on all the frequency values in a sound wave, whereas average vibrato extent is more specific to the sinusoidal waveform of a vibrato pitch-time curve — it measures the time averaged amplitude of that sinusoidal curve. Comparing the average extent of real vibrato and open string vibrato, we see that normal vibrato had a wider average extent: normal vibrato had a 776%, a 468%, a 273%, a 724%, and a 286% wider extent in the respective five harmonics than open string vibrato did, respectively (**Table 2A**). Similar to the comparison of pitch spread between open string vibrato and normal vibrato, here, normal vibrato had, on average, a wider extent than open string vibrato.

On the other hand, the rate and period of the vibrato in open string vibrato and real vibrato are very similar, showing little variation, suggesting that they are not affected by whether the vibrato played is open string vibrato or normal vibrato (**Table 2B**).

Finally, in comparing the shapes of the pitch-time curves of both vibratos, we see that normal vibrato, on average, had a higher "similarity" (quantified by Pearson's standard coefficient of correlation between the pitch values of the vibrato and the corresponding values of a fitted cosine curve) with the fitted sinusoidal curve than open string vibrato did, showing a 698% increase in the first harmonic, a 42.6% increase in the second harmonic, a 354% increase in the third harmonic, a 49.4% increase in the fourth harmonic, and a 51.7% increase in the fifth harmonic (**Table 3**). These results as well as a qualitative interpretation of shape (**Figure 2B**) suggest that the shape of the pitch-time curve in real vibrato was more similar to a sine wave than the curve in open string vibrato.

DISCUSSION

Our study aimed to accomplish three things: validate our proposed method of quantifying vibrato; confirm or disprove the existence of the effects of open string vibrato; and characterise open string vibrato (if it does indeed exist) by comparing it to conventional vibrato. We compared open string vibrato to conventional vibrato according to the three defining characteristics of vibrato: extent, rate and shape (6).

In our study, we first found a strong positive correlation between presence of normal vibrato and pitch spread thus supporting that this method of quantifying vibrato is effective. We then further demonstrated a positive correlation between the presence of open string vibrato and spread of pitch in the sounded note, which supports that physical effect of open string vibrato on notes does exist. Open string vibrato, compared to open string notes played with no vibrato, had a greater spread of pitch and thus more detectable vibrato. This supports our second hypothesis. Furthermore, the greatest percentage increase in pitch spread between vibrato and no vibrato for open string notes was around the second harmonic. This is in stark contrast to conventional vibrato, where the greatest percentage increase in pitch spread was around the fundamental. This is a tell-tale symptom of the effect of the technique of open string vibrato, which involves vibrating the finger on the note one octave above (the second harmonic overtone). This further supports the claim of the existence of the physical effect of open string vibrato and confirms the second hypothesis.

		(a) Average Vibrato Extent						(b) Average Vibrato Period					
Harmonic	**Form of Vibrato**	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Average**	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Average**
1st	**Normal**	2.28	2.46	2.10	2.17	2.03	2.208	0.19	0.19	0.2	0.2	0.21	0.198
	Open s.	0.30	0.27	0.24	0.15	0.30	0.252	0.2	0.21	0.2	0.17	0.14	0.184
2nd	**Normal**	1.61	1.19	1.44	1.14	1.78	1.432	0.2	0.19	0.2	0.2	0.21	0.2
	Open s.	0.54	0.35	0.58	0.59	0.61	0.534	0.2	0.2	0.19	0.21	0.21	0.202
3rd	**Normal**	1.28	1.52	1.04	1.60	1.43	1.374	0.21	0.19	0.2	0.2	0.21	0.202
	Open s.	0.06	1.12	0.04	0.08	0.11	0.082	0.21	0.2	0.23	0.18	0.19	0.202
4th	**Normal**	1.35	1.64	1.47	1.46	1.25	1.434	0.21	0.29	0.24	0.22	0.21	0.234
	Open s.	0.17	0.15	0.15	0.19	0.21	0.174	0.21	0.2	0.19	0.2	0.14	0.188
5th	**Normal**	0.90	0.59	0.38	0.33	0.93	0.626	0.22	0.24	0.21	0.27	0.21	0.23
	Open s.	0.09	0.15	0.13	0.19	0.25	0.162	0.2	0.18	0.18	0.18	0.16	0.18

Table 2: Average vibrato extent of notes played with different types of vibratos, rounded to the nearest 0.01. Open s. stands for open string (vibrato).

https://doi.org/10.59720/24-001

		Absolute Coefficient of Correlation					
Harmonic	Form of Vibrato	Test 1	Test 2	Test 3	Test 4	Test 5	Average
1st	Normal	0.53	0.75	0.81	0.72	0.78	0.718
	Open s.	0.1	0.02	0.23	0.10	0.00	0.09
2nd	Normal	0.13	0.02	0.71	0.38	0.80	0.408
	Open s.	0.06	0.17	0.08	0.6	0.52	0.286
3rd	Normal	0.02	0.51	0.72	0.19	0.74	0.436
	Open s.	0.04	0.26	0.04	0.13	0.01	0.096
4th	Normal	0.22	0.01	0.05	0.11	0.85	0.248
	Open s.	0.12	0.16	0.15	0.38	0.02	0.166
5th	Normal	0.16	0.05	0.21	0.06	0.43	0.182
	Open s.	0.02	0.14	0.09	0.23	0.12	0.12

Table 3: Absolute coefficient of correlation to a fitted sinusoidal curve of the pitch-time curves of notes played with different types of vibratos, rounded to the nearest 0.01. Open s. stands for open string (vibrato).

Fingered notes played without vibrato had consistently greater spread of pitch than open string notes played without vibrato. This is explained by the fact that, since the fingered note was played without vibrato, the pitch was being altered ever so slightly by human inconsistency. On the other hand, when the open string is played without vibrato, no finger was placed on the fingerboard and thus there is no pitch change possible due to human error. However, this discrepancy does not invalidate the results as it still holds true that when open string vibrato is played, there is more oscillation in the frequency than when the same note is played without any vibrato. Furthermore, the pitch spread in fingered notes played with normal vibrato was, on average, higher than the pitch spread in open string notes played with open string vibrato. This reveals that open string vibrato, while it does exist, is weak and less prominent than normal vibrato. In fact, open string vibrato was so weak that it was far more comparable to fingered notes played with no vibrato. Open string vibrato had less pitch spread than fingered notes with no vibrato, all except for the second harmonic, where the spread was slightly higher. This phenomenon further provides evidence for the effect of the open string vibrato technique on the sound produced. Similarly, open string vibrato on average had a vibrato extent that was narrower than that of normal vibrato.

In all, what can be learned from the comparisons of spread involving open string vibrato is that, while it indeed makes a difference compared to the straight open string that would otherwise have to be played, it has comparable spread to a fingered note without any vibrato. This contradicts the third hypothesis. So, the effect exists, but it is not a strong effect: conventional vibrato and even a non-vibrato fingered note produces a more pronounced effect than open string vibrato and so where possible, conventional vibrato can be used for a greater effect.

We analysed the shape of the open string vibrato pitch oscillations as shape is one of the key ways in which different vibratos differentiate themselves (6). The pitch-time curve of open string vibrato had, on average, a lower coefficient of correlation with the fitted sine wave than normal vibrato. This has two implications. Firstly, open string vibrato had a less sinusoidal shape, compared to normal vibrato. Secondly, normal vibrato maintained its sinusoidal shape more consistently than open string vibrato, whose shape was volatile and inconsistent. This, again, contradicts the third hypothesis. For musicians, this may mean that open string vibrato produces a less consistent sound than normal vibrato. However, since open string vibrato is relatively weak, the consistency of its pitch oscillation may not be noticeable without close analysis.

In conclusion, open string vibrato does exist and adds the oscillation in pitch desired in vibrato. However, it is far weaker than normal vibrato. Furthermore, it has an inconsistent and uneven shape, unlike normal vibrato, and moreover, it is narrower with a comparable level of pitch oscillation as a fingered note with no vibrato. Therefore, it is advantageous to use it, but only when there is no option, or it is too inconvenient to play normal vibrato (i.e. on the G-string on the violin, and on the C-string on the viola and cello).

Despite the success of this experiment in supporting effect of open string vibrato and analysing its characteristics, there were limitations, and there are areas for improvement and future study. Firstly, only the violin was tested. Different string instruments have different sizes, different timbres, lower ranges, and notably much thicker strings with drastically different linear densities. Whether these factors have an effect on open string vibrato is unknown, and the existence of the physical effect of open string vibrato is yet to be verified on these other instruments. This is an area which can be further studied. Secondly, only the G-string was tested for open string vibrato on the violin. In the future, the effect of the natural pitch of the string (and thus its linear density) can be tested to see if it affects vibrato and open string vibrato. Thirdly, what could be tested is whether the prominence of open string vibrato is affected by volume. Finally, only one player performed the vibrato in these experiments. It is unknown to what extent different styles of playing vibrato affects open string vibrato and its prominence. Vibrato technique varies in two key ways: the width of the vibrato (how far back and forth the finger rocks) and the rate of the vibrato (how many often per unit time the finger rocks back and forth), and these aspects could be tested to see if they affect the intensity of vibrato and open string vibrato.

In summary, we validated the method for quantifying vibrato using the calculation proposed. We showed that that open string vibrato has a physical effect on the sound waves produced by the violin, and we have characterised the open string vibrato in comparison with conventional vibrato. While the open string vibrato is a well-known technique, the value of our work lies within confirming its existence in an empirical and a scientific manner. Furthermore, by comparing it to conventional vibrato, familiar to many musicians, we hope our characterisation of open string vibrato will be useful to them.

MATERIALS AND METHODS

Data Collection and Analysis

First, using a microphone, the sounds of the violin were recorded. Second, the audio files of these recordings were loaded into Sonic Visualiser (8). Using Sonic Visualiser, the audio files were converted to csv file format, which consisted of many triplet data points. Each sound file thus has its own dataset. Each data point, in the form (*t, f, PSD*), is defined by its time (*s*), its frequency (*f*) in Hz, and its power spectral density (*PSD*) in W/Hz. Finally, the data in the csv files were analysed using a Python script.

The Python script analysed the data in the following steps. First, the data were filtered around the loudest (most powerful) few peak frequencies using a band pass filter. The loudest frequencies in the spectrum of a bowed violin sound are the

https://doi.org/10.59720/24-001

fundamental (f_1), the second harmonic (f_2), the third harmonic (f_3), the fourth harmonic (f_4) and the fifth harmonic (f_5) (**Figure 5**). For each sound wave, these five frequencies were used to filter and analyse the data. It is important to note that the relative prominence of harmonic overtones can vary a little, but it is generally true that the fundamental and the first few harmonic overtones are louder than the rest. Nevertheless, since the same overtones were used to analyse all of the datasets, there was no effect on the results of this research. The mean and standard deviation of the PSD values of the datapoints were calculated. Low-power noise, which would have been inaudible, was filtered away. Any data points with a PSD value lower than the cut-off value were deleted, with the cut-off PSD value being equal to the arithmetic mean minus the standard deviation of PSD values. After that, a smooth curve of best fit was added using the Savitzky-Golay smoothing filter to reduce the effect of random error and anomalies (9). Following that, the frequency was regulated so that it had a linear relationship with perceived pitch and could more accurately provide a numerical representation for pitch. Perceived pitch is proportional to the logarithm base 2 of frequency. Therefore, if the perceived pitch is represented numerically by p and the measured frequency of that pitch is represented by f_p, then p is directly proportional to the logarithm base 2 of f_p. So, the regulated frequency, a, is defined as:

$$a = k \cdot log_2(f_p) \propto p$$

in which k is a constant, taken as k = 100 for the purpose of visualisation. Since it is logarithmic, the value is unitless. Finally, the regulated frequency replaced the original frequency value in every data point.

Spread of Regulated Frequency

To quantify the amount of vibrato present in a sound, the spread of regulated frequencies in the sound can be measured. The wider the spread of regulated frequencies, the more it deviates from the mean (the original note) and thus the more it fluctuates. This means it has more vibrato. To measure spread, the standard deviation is taken, using the following equation:

$$\sigma = \sqrt{\frac{\sum(a_i - \bar{a})^2}{N}}$$

where σ is the spread of regulated frequency (standard deviation of regulated frequency values in the dataset), α_i is each regulated frequency value in the data, $\bar{\alpha}$ is the mean of the regulated frequency in the data, and N is the number of data points.

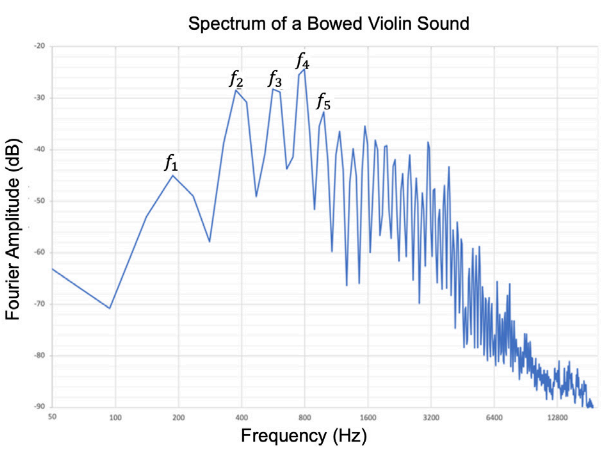

Figure 5: Spectrum of a bowed violin sound. Line graph showing the spectrum of a bowed violin sound with pitch G3 (n=1). Harmonic frequencies labelled. Open G string was played on the violin, recorded, and analysed.

Correlation Calculations

When comparing two different classes of notes, like non-vibrato open string and open string vibrato, it was useful to calculate their correlation coefficients. For this, the Pearson product moment correlation coefficient was calculated on all the data found in the tables, taking the control group in that situation (for example, non-vibrato open string note) as 0 and the experimental group (open string vibrato) as 1.

Vibrato Extent

When dealing with periodic oscillations in pitch like that found in traditional vibrato, measuring extent and period of the vibrato is useful. The extent of vibrato is defined as the maximum displacement of regulated frequency from the original regulated frequency of the sound. Here, the average extent for each period is calculated, as it is highly unlikely that any sound will have a perfectly consistent curve of pitch against time. It is calculated thus: first, local maxima and minima data points of regulated frequency in each wave cycle are detected in the dataset, then their regulated frequency values are taken for calculation, and finally, the following equation is applied to find the extent in each wave cycle and average the results. The equation is as follows:

$$\alpha = \frac{\sum(B_i - b_i)}{n}$$

where α is average vibrato extent, B_i is the regulated frequency value of each local maxima data point, b_i is the regulated frequency value of each local minima data point, and n is the number of wave cycles.

Vibrato Period and Rate

Similar to the previous calculation, the vibrato time period and its frequency of oscillation can be determined thusly: first, local maxima and minima data points of regulated frequency in each wave cycle are detected in the dataset, then their time values are taken for calculation, and finally, the following equation is applied to find the time period in each wave cycle and average the results:

$$\tau = \frac{\sum(T_i - t_i)}{n}$$

where τ is the average vibrato time period, T_i is the time value of each local maxima data point, t_i is the time value of each local minima data point, and n is the number of wave cycles. Once the period is known, the rate of vibrato, ψ , can be calculated using $\psi = \tau^{-1}$.

Sinusoid Similarity

First the curve of vibrato is centred around 0. Using τ and α, a sinusoidal wave $s(t)$, can be constructed in the graph of the sound wave with the regulated frequency α on the y-axis and time t on the x-axis using the equation:

$$s(t) = \alpha \cdot \cos\left(\frac{2\pi}{\tau} \cdot t\right)$$

The reason a sinusoidal wave is constructed is because vibrato waves are generally sinusoidal in shape, and so this allows open string vibrato's shape to be compared to the typical shape of normal vibrato (6).

It is important to note that this equation applies only to time values greater than the time value of the first local extreme, so that the start of the cosine wave lines up with the original curve. Once the sinusoidal wave is constructed, the coefficient of correlation between it and the original vibrato wave is calculated. The absolute value of the coefficient of correlation was taken because the direction of correlation has no effect on its shape since sinusoidal curves are symmetric and periodic.

Testing Conventional Vibrato

The purpose of this investigation was to validate the devised measurement and data analysis methods, in particular, the proposed method of vibrato intensity quantification. In order to do so, regular vibrato was played on a B3 (246.9 Hz) note, recorded, and analysed. Next, the same note was played with no vibrato (as a control group), recorded, and analysed. After repeating this five times, the values of regulated frequency spread for each were compared. Since the purpose of calculating the spread of regulated frequency is to quantify the amount of vibrato there is on a note, the spread value calculated for all the notes played with normal vibrato was expected to be higher on average than those calculated for all the notes played without any vibrato. To conclude that the method of vibrato intensity quantification was valid, this would have to have been the case.

All other conditions remained constant: the string played (its thickness, length, linear density, and the frequency at which it vibrates), the instrument of which this string was a part, the environment in which the experiment was performed (temperature, humidity and air pressure, string, time, place) and the speed of the bow on the string. The experiment was repeated to increase the reliability of the results and minimise the effect of human error, as the playing of the notes was performed by a human.

Testing Open String Vibrato

This study was conducted to confirm the physical existence of the effect of open string vibrato. First, the open G string (a G3 note, 196 Hz) was played without performing any vibrato, as a control group. Then, the open G string was played while performing open string vibrato. After repeating both recordings five times, values for vibrato intensity in each were calculated and compared. As before, all other conditions remained constant.

Received: December 28, 2023
Accepted: July 28, 2024
Published: May 01, 2025

REFERENCES

1. Timmers, Renee, and Peter Desain. "Vibrato: Questions and Answers from Musicians and Science." *Proceedings of the Sixth International Conference on Music Perception and Cognition*, 2000.
2. Seashore, Carl E. "The Natural History of the Vibrato." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 17, no. 12, 1931, pp. 623–626. *JSTOR*.
3. Mersenne, Marin. "Harmonie Universelle". *Éditions du CNRS*, 1986. Originally published in 1636.
4. Jeans, James H. "Mersenne's Laws." *Science and Music*. Dover, 1968, pp. 64-65. https://doi.org/10.1080/15436314.1938.11666448.
5. Bruch, Max. *Violin Concerto No. 1 in G minor, Op. 26*. C.F. Peters, n.d.
6. hr-Sinfonieorchester - Frankfurt Radio Symphony. "Bruch: 1. Violinkonzert · hr-Sinfonieorchester · Hilary Hahn · Andrés Orozco-Estrada." *YouTube*, 21 Dec. 2016, https://www.youtube.com/watch?v=KDJ6Wbzgy3E.
7. Desain, Peter, et al. "Rhythmic Aspects of Vibrato." *Rhythm Perception and Production*, 1999, pp. 203-216.
8. Cannam, Chris, et al. "Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files." *Proceedings of the 18th ACM International Conference on Multimedia*, 2010, pp. 1467–1468. https://doi.org/10.1145/1873951.1874248.
9. Savitzky, Abraham, and Marcel J. E. Golay. "Smoothing and Differentiation of Data by Simplified Least Squares Procedures." *Analytical Chemistry*, vol. 36, no. 8, 1964, pp. 1627-1639. https://doi.org/10.1021/ac60214a047.
10. Sundberg, Johan. "Acoustic and Psychoacoustic Aspects of Vocal Vibrato." *STL-QPSR*, vol. 35, no. 2-3, 1994, pp. 45-68.
11. Williams, Victoria. "Instruments (ABRSM Grade 7)." *MyMusicTheory.com*, https://www.mymusictheory.com/orchestration/instruments-abrsm-grade-7. Accessed 19 Jul. 2024
12. Pankhurst, Tom. "String Writing 2: Open Strings and Vibrato." *A Level Music*, https://www.alevelmusic.com/alevelcompositionhelp/4-texture-and-sonority/1-string-writing-introduction/string-writing-2-open-strings-and-vibrato. Accessed 19 Jul. 2024

APPENDIX

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
import numpy as np
from scipy.signal import savgol_filter, argrelextrema
import scipy.optimize

def rolling_window(a, window):
    shape = a.shape[:-1] + (a.shape[-1] - window + 1, window)
    strides = a.strides + (a.strides[-1],)
    return np.lib.stride_tricks.as_strided(a, shape=shape, strides=strides)

def reformat(df_o, approx, width):  # df_o MUST be a dataframe, approx MUST
be a float
    # Reformat the CSV file into suitably formatted dataframe

    df_o = df_o.iloc[:, :width]

    length = len(list(df_o.columns))

    # Split data frame
    df_freq = df_o.iloc[:, [0] + list(range(1, length - 1, 2))]
    df_mag = df_o.iloc[:, [0] + list(range(2, length, 2))]

    # Melt first half
    df_f = pd.melt(df_freq, id_vars=["TIME"],
value_vars=df_freq.columns[1:].tolist(),
                   var_name='VAR', value_name='Frequency')
    df_f['VAR'] = df_f['VAR'].str.replace('FREQ ', '')

    # Melt second half
    df_m = pd.melt(df_mag, id_vars=["TIME"],
value_vars=df_mag.columns[1:].tolist(),
                   var_name='VAR', value_name='PSD')
    df_m['VAR'] = df_m['VAR'].str.replace('MAG ', '')

    # Merge both halves
    df = pd.merge(df_f, df_m, how="left", on=['TIME', 'VAR'])

    # Finishing touches
    df = df.drop(["VAR"], axis=1)
    df = df.fillna(0)

    # Approximate Frequencies
    if approx == 0:
        pass
    else:
        df['Frequency'] = df["Frequency"].apply(lambda x: np.round((x /
approx), decimals=0) * approx)
    # df['PSD'] = df['PSD'].apply(lambda x: x * (10 ** 5))

    return df

def zero_centre(df):
    centre = np.average(df["Curve"].mode().to_numpy())
    df["Curve"] = df["Curve"] - centre
```

```
    return df

def frequency_filter(df, base, plusminusx):
    upper_bound = base + plusminusx
    lower_bound = base - plusminusx
    filtered_df = df.loc[(df["Frequency"] <= upper_bound) &
(df["Frequency"] >= lower_bound)]

    return filtered_df.sort_values(by="TIME", ignore_index=True)

def noise_reduce(df):
    mean = np.average(df["PSD"])
    var = np.average((df["PSD"] - mean) ** 2)
    std = np.sqrt(var)
    return df[df.PSD >= (mean-std)]

def plot_PSD_count(df, approx):
    mean = np.average(df["PSD"])
    var = np.average((df["PSD"] - mean) ** 2)
    std = np.sqrt(var)

    df['PSD'] = df["PSD"].apply(lambda x: np.round((x / approx), decimals=0)
* approx)
    unique, counts = np.unique(df["PSD"], return_counts=True)
    plt.plot(unique, counts)
    plt.axvline(x=mean, color="red", label="Mean")
    plt.axvline(x=mean - std, color="purple", label="-STD")
    plt.axvline(x=mean + std, color="purple", label="+STD")
    plt.show()

def smooth_curve(df):
    df["Curve"] = savgol_filter(df["Frequency"], 23, 5)
    df["Curve"] = df['Curve'].apply(lambda x: 100 * np.log2(x))
    return df

def find_max(df, window):
    df['localmax'] = df.iloc[argrelextrema(df.Curve.values, np.greater_equal,
order=window)[0]]['Curve']
    df = df.fillna(0)
    return df[df.localmax != 0].drop(columns=["Frequency", "PSD"])

def find_min(df, window):
    df['localmin'] = df.iloc[argrelextrema(df.Curve.values, np.less_equal,
order=window)[0]]['Curve']
    df = df.fillna(0)
    return df[df.localmin != 0].drop(columns=["Frequency", "PSD"])

def get_time_periods(df):
    df_peaks = find_max(df, 16)
    times = df_peaks["TIME"].to_numpy()
    time_periods = list()
    for i in range(0, len(times)-1):
        time_periods.append(times[i+1] - times[i])
    return time_periods

def freq_std(df):
    mean = np.average(df["Curve"])
```

```
    var = np.average((df["Curve"] - mean) ** 2)
    return np.sqrt(var)

def rolling_ranges(df, width):
    freqs = df["Curve"].to_numpy()
    ranges = np.ptp(rolling_window(freqs, width))
    return ranges

def get_amplitudes(df):
    peaks = find_max(df, 16)["localmax"].to_numpy()
    troughs = find_min(df, 16)["localmin"].to_numpy()
    if len(peaks) <= len(troughs):
        length = len(peaks)
    else:
        length = len(troughs)
    peaks = peaks[0:length-1]
    troughs = troughs[0:length - 1]
    return (np.subtract(peaks, troughs))/2

def data_prep(filename, freq):
    base_path = "measurements/"
    df = pd.read_csv(base_path + filename + ".csv")
    df = reformat(df, 0.1, 100)
    df = frequency_filter(df, freq, 15).reset_index()
    df = noise_reduce(df)
    return smooth_curve(df)

def raw_to_freq_std(filename, freq):
    df = data_prep(filename, freq)
    mean = np.average(df["Curve"])
    var = np.average((df["Curve"] - mean) ** 2)
    return float(np.sqrt(var))

def raw_to_av_amplitude(filename, freq):
    df = data_prep(filename, freq)
    amps = get_amplitudes(df)
    return np.average(amps)

def raw_to_av_period(filename, freq):
    df = data_prep(filename, freq)
    periods = get_time_periods(df)
    return np.average(periods)

def sinusoid_similarity(df):
    df = zero_centre(df)

    T = np.average(get_time_periods(df))
    A = np.average(get_amplitudes(df))

    dfmax = find_max(df, 16)
    peakminT = dfmax["TIME"].min()
    dfmin = find_min(df, 16)
    troughminT = dfmin["TIME"].min()

    if troughminT <= peakminT:
        df["SinTimes"] = df["TIME"][df.TIME >= troughminT]
        df["Sin"] = -1*A*np.cos((np.radians(360)/T)*df["SinTimes"])
```

```
        df = df.drop(columns="SinTimes")
    else:
        df["SinTimes"] = df["TIME"][df.TIME >= peakminT]
        df["Sin"] = A * np.cos((np.radians(360) / T) * df["SinTimes"])
        df = df.drop(columns="SinTimes")

    corr = df["Curve"].corr(df["Sin"])
    return df, corr

def plot(filename1, freq):
    df = data_prep(filename1, freq)
    df_peaks = find_max(df, 16)
    df_troughs = find_min(df, 16)
    fig, ax = plt.subplots(figsize=(6, 2))
    ax.plot(df["TIME"], df["Curve"], "-b")
    ax.scatter(df_peaks["TIME"], df_peaks["localmax"], 20, "r", "o")
    ax.scatter(df_troughs["TIME"], df_troughs["localmin"], 20, "r", "o")
    plt.show()


# comparison charts
def plot2(filename1, filename2, freq):
    matplotlib.style.use("seaborn-v0_8-pastel")
    df1 = data_prep(filename1, freq)
    df2 = data_prep(filename2, freq)
    fig, (ax1, ax2) = plt.subplots(2, sharex=True, sharey=True)
    fig.set_size_inches(7, 6)
    ax1.plot(df1["TIME"], df1["Curve"], "#257ca3")
    ax2.plot(df2["TIME"], df2["Curve"], "#257ca3")
    fig.suptitle("Comparison of Frequency-Time Graphs")
    ax1.set_title("With Open String Vibrato")
    ax2.set_title("Without any Vibrato")
    ax1.set_ylabel("$100$·$log_2(Frequency/Hz)$")
    ax2.set_ylabel("$100$·$log_2(Frequency/Hz)$")
    plt.xlabel("$time [s]$")
    plt.show()

def raw_sin(filename, freq):
    df = data_prep(filename, freq)
    return sinusoid_similarity(df)

exitloop = False
f = 0
investigation = input("Which investigation? ")
while exitloop == False:
    if investigation == "prelim":
        f = 247
        exitloop = True
    elif investigation == "prim":
        f = 196
        exitloop = True
    else:
        exitloop = False
        print("Input Error!")

def std(investigation):
```

```
    df = pd.DataFrame(columns=["Test1", "Test2", "Test3", "Test4", "Test5"],
index=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

    for i in range(0, 5):
        number = str(i+1)
        vib1 = raw_to_freq_std(investigation + "/vib" + number, f)
        novib1 = raw_to_freq_std(investigation + "/novib" + number, f)

        df.iloc[0, i] = vib1
        df.iloc[1, i] = novib1

        vib2 = raw_to_freq_std(investigation + "/vib" + number, 2*f)
        novib2 = raw_to_freq_std(investigation + "/novib" + number, 2*f)

        df.iloc[2, i] = vib2
        df.iloc[3, i] = novib2

        vib3 = raw_to_freq_std(investigation + "/vib" + number, 3*f)
        novib3 = raw_to_freq_std(investigation + "/novib" + number, 3*f)

        df.iloc[4, i] = vib3
        df.iloc[5, i] = novib3

        vib4 = raw_to_freq_std(investigation + "/vib" + number, 4*f)
        novib4 = raw_to_freq_std(investigation + "/novib" + number, 4*f)

        df.iloc[6, i] = vib4
        df.iloc[7, i] = novib4

        vib5 = raw_to_freq_std(investigation + "/vib" + number, 5 * f)
        novib5 = raw_to_freq_std(investigation + "/novib" + number, 5 * f)

        df.iloc[8, i] = vib5
        df.iloc[9, i] = novib5


    df = df.astype(float)
    df = df.round(2)
    df["Mean"] = df.mean(axis=1)

    return df

def amp(investigation):
    df = pd.DataFrame(columns=["Test1", "Test2", "Test3", "Test4", "Test5"],
index=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

    for i in range(0, 5):
        number = str(i + 1)
        vib1 = raw_to_av_amplitude(investigation + "/vib" + number, f)
        novib1 = raw_to_av_amplitude(investigation + "/novib" + number, f)

        df.iloc[0, i] = vib1
        df.iloc[1, i] = novib1

        vib2 = raw_to_av_amplitude(investigation + "/vib" + number, 2 * f)
        novib2 = raw_to_av_amplitude(investigation + "/novib" + number, 2 *
f)
```

```
        df.iloc[2, i] = vib2
        df.iloc[3, i] = novib2

        vib3 = raw_to_av_amplitude(investigation + "/vib" + number, 3 * f)
        novib3 = raw_to_av_amplitude(investigation + "/novib" + number, 3 *
f)

        df.iloc[4, i] = vib3
        df.iloc[5, i] = novib3

        vib4 = raw_to_av_amplitude(investigation + "/vib" + number, 4 * f)
        novib4 = raw_to_av_amplitude(investigation + "/novib" + number, 4 *
f)

        df.iloc[6, i] = vib4
        df.iloc[7, i] = novib4

        vib5 = raw_to_av_amplitude(investigation + "/vib" + number, 5 * f)
        novib5 = raw_to_av_amplitude(investigation + "/novib" + number, 5 *
f)

        df.iloc[8, i] = vib5
        df.iloc[9, i] = novib5

    df = df.astype(float)
    df = df.round(2)
    df["Mean"] = df.mean(axis=1)

    return df

def period(investigation):
    df = pd.DataFrame(columns=["Test1", "Test2", "Test3", "Test4", "Test5"],
index=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

    for i in range(0, 5):
        number = str(i + 1)
        vib1 = raw_to_av_period(investigation + "/vib" + number, f)
        novib1 = raw_to_av_period(investigation + "/novib" + number, f)

        df.iloc[0, i] = vib1
        df.iloc[1, i] = novib1

        vib2 = raw_to_av_period(investigation + "/vib" + number, 2 * f)
        novib2 = raw_to_av_period(investigation + "/novib" + number, 2 * f)

        df.iloc[2, i] = vib2
        df.iloc[3, i] = novib2

        vib3 = raw_to_av_period(investigation + "/vib" + number, 3 * f)
        novib3 = raw_to_av_period(investigation + "/novib" + number, 3 * f)

        df.iloc[4, i] = vib3
        df.iloc[5, i] = novib3

        vib4 = raw_to_av_period(investigation + "/vib" + number, 4 * f)
        novib4 = raw_to_av_period(investigation + "/novib" + number, 4 * f)
```

```
        df.iloc[6, i] = vib4
        df.iloc[7, i] = novib4

        vib5 = raw_to_av_period(investigation + "/vib" + number, 5 * f)
        novib5 = raw_to_av_period(investigation + "/novib" + number, 5 * f)

        df.iloc[8, i] = vib5
        df.iloc[9, i] = novib5

    df = df.astype(float)
    df = df.round(2)
    df["Mean"] = df.mean(axis=1)

    return df

def sinsim(investigation):
    df = pd.DataFrame(columns=["Test1", "Test2", "Test3", "Test4", "Test5"],
index=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

    for i in range(0, 5):
        number = str(i + 1)
        vib1 = raw_sin(investigation + "/vib" + number, f)[1]
        novib1 = raw_sin(investigation + "/novib" + number, f)[1]

        df.iloc[0, i] = vib1
        df.iloc[1, i] = novib1

        vib2 = raw_sin(investigation + "/vib" + number, 2 * f)[1]
        novib2 = raw_sin(investigation + "/novib" + number, 2 * f)[1]

        df.iloc[2, i] = vib2
        df.iloc[3, i] = novib2

        vib3 = raw_sin(investigation + "/vib" + number, 3 * f)[1]
        novib3 = raw_sin(investigation + "/novib" + number, 3 * f)[1]

        df.iloc[4, i] = vib3
        df.iloc[5, i] = novib3

        vib4 = raw_sin(investigation + "/vib" + number, 4 * f)[1]
        novib4 = raw_sin(investigation + "/novib" + number, 4 * f)[1]

        df.iloc[6, i] = vib4
        df.iloc[7, i] = novib4

        vib5 = raw_sin(investigation + "/vib" + number, 5 * f)[1]
        novib5 = raw_sin(investigation + "/novib" + number, 5 * f)[1]

        df.iloc[8, i] = vib5
        df.iloc[9, i] = novib5

    df = df.astype(float)
    df = df.round(2)
    df["Mean"] = df.mean(axis=1)

    return df
```

```
std(investigation).to_csv("output/" + investigation + "-spreads.csv")
amp(investigation).to_csv("output/" + investigation + "-extents.csv") #Ignore
"novib" rows in file
period(investigation).to_csv("output/" + investigation + "-periods.csv")
#Ignore "novib" rows in file
sinsim(investigation).to_csv("output/" + investigation + "-shapes.csv")
#Ignore "novib" rows in file

import numpy as np
import pandas as pd

import matplotlib as mpl
import matplotlib.pyplot as plt

import seaborn as sns
import seaborn.objects as so

import json

from ver3 import *


def get_var_name(var):
    for name, value in globals().items():
        if value is var:
            return name


def harmonics_pitches_tuples(filename, base_freq):
    pitches1 = data_prep_delta(filename, 1 * base_freq)["Curve"].tolist()
    len1 = len(pitches1)
    harms1 = [1] * len1

    pitches2 = data_prep_delta(filename, 2 * base_freq)["Curve"].tolist()
    len2 = len(pitches2)
    harms2 = [2] * len2

    pitches3 = data_prep_delta(filename, 3 * base_freq)["Curve"].tolist()
    len3 = len(pitches3)
    harms3 = [3] * len3

    pitches4 = data_prep_delta(filename, 4 * base_freq)["Curve"].tolist()
    len4 = len(pitches4)
    harms4 = [4] * len4

    pitches5 = data_prep_delta(filename, 5 * base_freq)["Curve"].tolist()
    len5 = len(pitches5)
    harms5 = [5] * len5

    harms = harms1 + harms2 + harms3 + harms4 + harms5
    pitches = pitches1 + pitches2 + pitches3 + pitches4 + pitches5

    return zip(harms, pitches)
```

```
def quadruplets(filename, base_freq, vib, string):
    tuplelist = harmonics_pitches_tuples(filename, base_freq)
    return [list(tup) + [vib] + [string] for tup in tuplelist]


def multifiledataprep(basename, base_freq, vib, string):
    ls = []
    for i in range(1, 6):
        trip = quadruplets(basename + str(i), base_freq, vib, string)
        ls = ls + trip
    return ls

def filesaveystuff():
    fingered_novib = multifiledataprep("prelim/novib", 247, "no vibrato",
"fingered")
    fingered_vib = multifiledataprep("prelim/vib", 247, "normal vibrato",
"fingered")
    open_novib = multifiledataprep("prim/novib", 196, "no vibrato", "open
string")
    open_vib = multifiledataprep("prim/vib", 196, "open string vibrato",
"open string")

    with open("output/fingered_novib.json", 'w') as a:
        json.dump(fingered_novib, a, indent=2)
    with open("output/fingered_vib.json", 'w') as b:
        json.dump(fingered_vib, b, indent=2)
    with open("output/open_novib.json", 'w') as c:
        json.dump(open_novib, c, indent=2)
    with open("output/open_vib.json", 'w') as d:
        json.dump(open_vib, d, indent=2)


with open("output/fingered_novib.json", 'r') as e:
    fingered_novib = json.load(e)
with open("output/fingered_vib.json", 'r') as f:
    fingered_vib = json.load(f)
with open("output/open_novib.json", 'r') as g:
    open_novib = json.load(g)
with open("output/open_vib.json", 'r') as h:
    open_vib = json.load(h)


def comparison_chart(comp1, comp2):
    df = pd.DataFrame(columns=["harmonic", "change in pitch", "vibrato"],
data=comp1 + comp2)
    g = sns.FacetGrid(df, col="harmonic", sharex=False, sharey=False,
height=6)
    g.figure.set_size_inches(10, 7)
    g.map_dataframe(sns.violinplot, x="harmonic", y="change in pitch",
hue="vibrato", palette="crest")
    g.set_titles(col_template="")
    g.tick_params(axis="y", direction="inout", labelsize="small")
    g.add_legend(loc="upper center", ncol=2, frameon=True)
    g.figure.set_constrained_layout(constrained=True)
    filepath = "output/" + get_var_name(comp1) + "+" + get_var_name(comp2) +
".png"
    g.savefig(filepath)
```

```
    plt.show()


def vlnplot_comparison(comp1, comp2, axis, col, legend, hue):
    df = pd.DataFrame(columns=["harmonic", "change in pitch", "vibrato",
"string"], data=comp1 + comp2)
    return sns.violinplot(data=df, x="harmonic", y="change in pitch",
hue=hue, palette=col, ax=axis,
                          linewidth=0.5, inner="point", legend=legend)


def boxplot_comparison(comp1, comp2, axis, col, legend, hue):
    df = pd.DataFrame(columns=["harmonic", "change in pitch", "vibrato",
"string"], data=comp1 + comp2)
    return sns.boxplot(data=df, x="harmonic", y="change in pitch", hue=hue,
palette=col, ax=axis, fliersize=2,
                       flierprops={"marker": "*"}, legend=legend)

def fig1():
    fig = plt.figure(figsize=(15,13))

    (topfig, bottomfig) = fig.subfigures(2, 1)

    topfig.suptitle('(1a)', x=0.1, y=0.95, fontweight="heavy")

    bottomfig.suptitle('(1b)', x=0.1, y=0.95, fontweight="heavy")

    (topax1, topax2) = topfig.subplots(1, 2, sharex=True, sharey=True)
    topfig.subplots_adjust(wspace=0)
    (bottomax1, bottomax2) = bottomfig.subplots(1, 2, sharex=True,
sharey=True)
    bottomfig.subplots_adjust(wspace=0)

    vlnplot_comparison(fingered_novib, fingered_vib, topax1, "mako", False,
"vibrato")
    boxplot_comparison(fingered_novib, fingered_vib, topax2, "mako", "brief",
"vibrato")
    sns.move_legend(topax2, "lower center", ncol=2, bbox_to_anchor=(0, 0),
title=None, frameon=True)

    vlnplot_comparison(open_novib, open_vib, bottomax1, "flare", False,
"vibrato")
    boxplot_comparison(open_novib, open_vib, bottomax2, "flare", "brief",
"vibrato")
    sns.move_legend(bottomax2, "lower center", ncol=2, bbox_to_anchor=(0, 0),
title=None, frameon=True)

    fig.savefig('output/fig1', bbox_inches='tight')

def fig3():

    fig = plt.figure(figsize=(15, 13))

    (topfig, bottomfig) = fig.subfigures(2, 1)

    topfig.suptitle('(3a) - vibrato', x=0.1, y=0.95, fontweight="heavy")
```

```
    bottomfig.suptitle('(3b) - no vibrato', x=0.1, y=0.95,
fontweight="heavy")

    (topax1, topax2) = topfig.subplots(1, 2, sharex=True, sharey=True)
    topfig.subplots_adjust(wspace=0)
    (bottomax1, bottomax2) = bottomfig.subplots(1, 2, sharex=True,
sharey=True)
    bottomfig.subplots_adjust(wspace=0)

    vlnplot_comparison(open_vib, fingered_vib, topax1, "mako", False,
"string")
    boxplot_comparison(open_vib, fingered_vib, topax2, "mako", "brief",
"string")
    sns.move_legend(topax2, "lower center", ncol=2, bbox_to_anchor=(0, 0),
title=None, frameon=True)

    vlnplot_comparison(open_novib, fingered_novib, bottomax1, "flare", False,
"string")
    boxplot_comparison(open_novib, fingered_novib, bottomax2, "flare",
"brief", "string")
    sns.move_legend(bottomax2, "lower center", ncol=2, bbox_to_anchor=(0, 0),
title=None, frameon=True)

    fig.savefig('output/fig3', bbox_inches='tight')

def fig4():
    fig = plt.figure(figsize=(15,13))
    gs = fig.add_gridspec(2, hspace=0)
    (ax1, ax2) = gs.subplots(sharex=True, sharey=True)
    df = pd.DataFrame(columns=["harmonic", "change in pitch", "vibrato",
"string"], data=fingered_novib + open_vib)
    df["vibrato"] = df["vibrato"].replace("no vibrato", "fingered no
vibrato")
    sns.violinplot(data=df, x="harmonic", y="change in pitch", hue="vibrato",
palette="pastel", ax=ax1,
                          linewidth=0.5, inner="point", legend="auto")
    sns.boxplot(data=df, x="harmonic", y="change in pitch", hue="vibrato",
palette="pastel", ax=ax2, fliersize=2,
                flierprops={"marker": "*"}, legend=False)
    sns.move_legend(ax1, "lower center", ncol=2, bbox_to_anchor=(0.5, 1),
title=None, frameon=False)
    fig.savefig('output/fig4', bbox_inches='tight')

fig4()
plt.show()
```