tonespace

v2.5 user guide

mucoder

www.mucoder.net/tonespace

Table of Contents

Welcome

Thank you for using mucoder tonespace 2.5.

Can't find what you need in this guide?

- Visit us on the web: www.mucoder.net
- Mail us: support@mucoder.net

What is tonespace ?

- it's a different, more intuitive music keyboard, combined with a chorder
- It supports over 50 different grids, 14 scales, 18 keys, 30+ chord types and 25 chord voicings
- it can function as a VST or Audio Unit plugin, processing incoming MIDI events and/or generating outgoing MIDI
- you don't need to use MIDI though: there is also a standalone executable which contains its own little synth for a fast start

What can you do with it ?

- you can audition and play chords while you move the mouse around in a grid-like space
- you can learn how musical scales and keys work, and how chords relate to these, using the simple octave-based spaces
- you can visually discover how chords map onto surprisingly simple geometric shapes in the more advanced spaces

- you can use the chord-generation algorithm to generate chords from these simple shapes in the advanced spaces
- you can also generate all chords that fit a chosen scale/key, then take the best chord, or alternate between all of them

Getting Started

Using tonespace is easy. After having installed tonespace, please read this to get started.

Installing tonespace

To proceed with installation please read the following sections:

System requirements

tonespace runs on Microsoft Windows, Apple Mac OS X and Linux.

Windows Requirements

tonespace for Windows requires :

- Windows XP with Service Pack 3 (32-bit)
- Windows Vista with Service Pack 2 (32-bit)
- Windows 7 with Service Pack 1 (32-bit or 64-bit)
- Windows 8 (32-bit or 64-bit)
- A compatible 32-bit or 64-bit VST host application (unless the standalone version is used)
- An audio interface (preferably one with an ASIO driver)

Mac OS X Requirements

tonespace for OS X requires:

- A Mac with Intel processor
- OS X 10.5 Leopard (except for the AU plugin version, which requires 10.6 or higher)
- OS X 10.6 Snow Leopard
- OS X 10.7 Lion
- OS X 10.8 Mountain Lion
- Either a compatible 32-bit or 64-bit Audio Unit or VST host application (unless the standalone version is used)
- An audio interface

Linux Requirements

tonespace for Linux requires:

- Ubuntu 12.04 Precise Pangolin 64-bit (other recent 64-bit Linux versions might work too, but your mileage may vary)
- A compatible 64-bit VST host application capable of loading native Linux VST plugins (unless the standalone version is used)
- An audio interface (supporting ALSA or Jack)

Miscellaneous

Miscellaneous issues:

- Direct MIDI output from tonespace to a DAW host requires that you
  - use the VST plugin version
  - use a DAW host that is capable of processing outgoing midi from a plugin (eg Ableton Live, Cubase, Sonar, EnergyXT, ...)
- With AU or standalone versions you need to route MIDI output to the host via a MIDI loopback adapter (eg Midi Yoke on Windows, or the IAC bus on OS X)

Installation

Prerequisites

Please verify that your configuration meets the system requirements listed in the System Requirements Section.

Installing on Windows

First extract the windows .zip archive to a directory on your file system.

On Windows you can use either the

- Sandalone 32-bit version (*tonespace.exe*) or 64-bit (*tonespace_64.exe*). This needs no installation, just run it and you are set.
- VST plugin 32-bit version 32-bit (*tonespace.dll*) or 64-bit (*tonespace_64.dll*). Copy this dll into your VST directory of your DAW host and run your VST host.

Please note: if you plan on using the older 1.0 VST version of tonespace side-by-side with tonespace 2, it is recommended to rename the VST dll of one of these versions. This is only an issue for Windows users, as tonespace 1.0 was Windows-only.
How?

- Rename the v1.0 tonespace.dll to tonespace1.dll if you have no old projects that depend on that plugin version
- Or (less preferred) rename the v2.0 tonespace.dll to tonespace2.dll if you have old projects that depend on tonespace 1.0

Installing on Mac OS X

First mount the OS X .dmg archive on your mac by double clicking it

On OS X you can use either the

- Standalone version (*tonespace.app*). This needs no installation, just run it and you are set. This is a universal binary containing both 32 and 64-bit versions.
- Audio Unit and VST plugin version (*tonespace.component*). This is a universal binary containing both AU and VST versions, in both 32 and 64-bit versions.
- Setting up the AU version: Copy the *tonespace.component* file into your AU directory (*/Users/yourname/Library/Audio/Plug-ins/Components*) and run your AU host.
- Setting up the VST version: Copy the *tonespace.component* file into your VST directory (*/Users/yourname/Library/Audio/Plug-ins/VST*) and rename it to *tonespace.vst*. Then run your VST host.

Installing on Linux

First extract the Linux .zip archive to a directory on your file system.

On Linux you can use either the

- Sandalone version (*tonespace*). You first need to set the executable bit on this binary using the command 'chmod +x tonespace'. Then just run it and you are set.
- VST plugin version (*tonespace.so*). Copy this file into your VST directory of your DAW host and run your VST host.
- All binaries for Linux are currently 64-bit only.

License agreement & thank you

mucoder tonespace 2.x freeware
license agreement and thank you

A) mucoder tonespace 2.x License Agreement

Informal summary for mortals:

You can
-use it free of charge
-use it in your performances, either commercial or non-commercial

You cannot (unless you have written consent from the author)
-distribute it (please link to www.mucoder.net/tonespace instead)
-sell it
-modify it or remove any part of it
-hold the author or publisher responsible for any damages

The author cannot guarantee
-bug fixes and other support

The small print:

BY USING THE SOFTWARE, YOU ARE CONSENTING TO BE BOUND BY AND BECOME A PARTY TO THIS AGREEMENT AS THE "LICENSEE." IF YOU DO NOT AGREE TO ALL OF THE TERMS OF THIS AGREEMENT, YOU MUST NOT INSTALL OR USE THE PRODUCT, AND YOU DO NOT BECOME A LICENSEE UNDER THIS AGREEMENT.

1. License Agreement.

As used in this Agreement, "Licensor" shall mean Altalogix BVBA, Karel Schurmansstraat 40, 3010 Leuven (Kessel-Lo), Belgium. "Product" shall mean mucoder tonespace version 2.x

Licensor grants Licensee a non-exclusive and non-transferable license to reproduce and use for personal or internal business purposes, on a single computer and by a single person at a time, the executable code version of the Product, provided any copy must contain all of the original proprietary notices. This license does not entitle Licensee to receive from Licensor hard-copy documentation, technical support, telephone assistance, or enhancements or updates to the Product, although Licensor may decide to provide these anyway. Licensor may terminate this Agreement at any time, for any reason or no reason. Licensor may also terminate this Agreement if Licensee breaches any of its terms and conditions. Upon termination, Licensee shall destroy all copies of the Product and of the personalised license files that were received by Licensee after purchase.

2. Restrictions.

Without Licensor's prior written consent, Licensee may not: (i) modify or create any derivative works of the Product or documentation, including customization, translation or localization; (ii) decompile, disassemble, reverse engineer, or otherwise attempt to derive the source code for the Product (except to the extent applicable laws specifically prohibit such restriction); (iii) redistribute, encumber, sell, rent, lease, sublicense, or otherwise transfer rights to the Product, (iv) remove or alter any trademark, logo, copyright or other proprietary notices, legends, symbols or labels in the Product. In case Licensee purchased a license to continue using the product and received a personalised license file to unlock the Product, Licensee will ensure that this license file is kept in a secure place and does not fall into the hands of unauthorised users. If a license file is found to be published anywhere this is considered sufficient reason for immediate termination of that license.

3. Proprietary Rights.

Title, ownership rights, and intellectual property rights in the Product shall remain in Licensor and/or his

suppliers. The Product is protected by copyright and other intellectual property laws and by international treaties.

4. Disclaimer of Warranty.

THE PRODUCT IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE WARRANTIES THAT IT IS FREE OF DEFECTS, VIRUS FREE, ABLE TO OPERATE ON AN UNINTERRUPTED BASIS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS AGREEMENT. NO USE OF THE PRODUCT IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

5. Limitation of Liability.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT WILL Licensor OR HIS AFFILIATES OR SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE THE PRODUCT, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOST PROFITS, LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF, AND REGARDLESS OF THE LEGAL OR EQUITABLE THEORY (CONTRACT, TORT OR OTHERWISE) UPON WHICH THE CLAIM IS BASED. IN ANY CASE, Licensor'S COLLECTIVE LIABILITY UNDER ANY PROVISION OF THIS AGREEMENT SHALL NOT EXCEED IN THE AGGREGATE THE SUM OF THE FEES LICENSEE PAID FOR THIS LICENSE (IF ANY). SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL, CONSEQUENTIAL OR SPECIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

6. Miscellaneous.

All disputes relating to this Agreement are subject to the exclusive jurisdiction of the courts of Leuven, Belgium.

B) Third Party Component Thank You's and license agreements
---

a) Harmony Space

What:
a major part of tonespace was inspired by the innovative Harmony Space paper by Simon Holland, describing an early implementation of an educational software tool that uses an intuitive navigation in a grid-like space to generate music, based on Longuet-Higgins' and Balzano's theories.

Source:
http://mcl.open.ac.uk/sh/uploads/Harmony%20Space%20ICMC%2087.pdf

Harmony space author's notice:
Harmony Space: http://harmonyspace.co.uk/
The Music Computing Lab: http://mcl.open.ac.uk/musiclab
Simon Holland: http://mcl.open.ac.uk/sh

b) Mutopia Project

What:
a great site and community that among other things collects and provides access to public domain MIDI files.

tonespace only uses public domain midi files from Mutopia

Source:
http://www.mutopiaproject.org/

Mutopia notice:
Disclaimer: The Mutopia Project is run by volunteers, and the material within it is provided "as-is". NO WARRANTY of any kind is made, including fitness for any particular purpose. No claim is made as to the accuracy or the factual, editorial or musical correctness of any of the material provided here.

c) Kiss FFT

What:
a powerful, yet easy to use C library for fast FFT processing

Source:
http://kissfft.sourceforge.net/

KISS FFT author's notice:

d) PADsynth

What:
a great algorithm and C++ sample for generating pad-like synth sounds

Source:

http://zynaddsubfx.sourceforge.net/doc/PADsynth/PADsynth.htm

PADsynth author's notice:
Implementation of PADsynth algorithm.
The algorithm and the implementation released under Public Domain by Nasca O. Paul.

e) Steinberg

What:
tonespace uses the industry-leading VST plugin and ASIO technology by Steinberg. This needs probably no introduction.

Source:
http://www.steinberg.net

Steinberg notice:
VST and ASIO are a trademark of Steinberg Soft- und Hardware GmbH.

Getting started

For a quick start try the following three scenarios:

- View Music on the Grid
- Play and Learn with C Major Scale and Chord
- Generate Chords with Geometry
- Automatically fit chords to scale/key

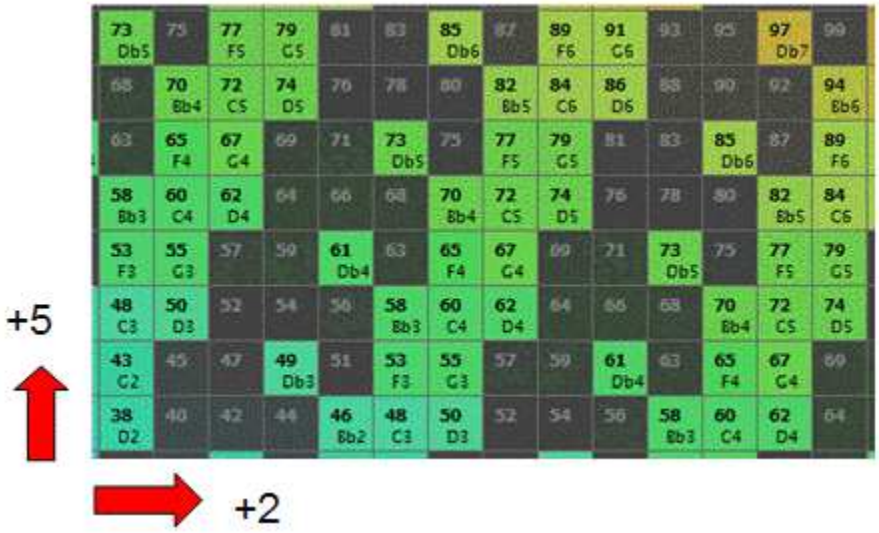
If you are ready for some more theory, proceed to the Concepts

Concepts

We will now discuss some concepts that help you get a fuller understanding of tonespace.

Introducing spaces

The main element in tonespace is the colored on-screen grid, which we call a space. A space is like a two-dimensional spreadsheet which contains midi note numbers. When you click on a cell in the space the corresponding midi note is played (or a chord based on that note, as you will see later).



Also, some cells are colored, others are not. This means that only the colored notes are part of the currently selected scale and key (Blues scale and key of G in the picture below). Only colored notes can be played. But more about that later.

The note numbers increase from left to right and from bottom to top. In the picture above the increase is 2 semitones in horizontal direction and 5 semitones in vertical direction. This space is called M2-P4 [2:5] or [2:5] for short [1].

As you can guess, other combinations of horizontal/vertical increases are possible, leading to other spaces. You can select these spaces by choosing a value for the space combobox in the parameter section of the screen.

There are two wellknown spaces, named after the researchers who discovered them:

- [4:7] or Longuet-Higgins' space
- [4:3] or Balzano's space

These two spaces have special properties. One of those properties is for instance that certain common chords look like very simple shapes when drawn on the space.

How chords fit to a space

Let's look at a grid based on Balzano's space. If we draw a major chord in that space, it will have a horizontally mirrored L-shape:

62 D4	66 F#4	70 A#4	74 D5	78 F#5	82 A#5
59 B3	63 D#4	67 G4	71 B4	75 D#5	79 G5
56 G#3	60 C4	64 E4	68 G#4	72 C5	76 E5
53 F3	57 A3	61 C#4	65 F4	69 A4	73 C#5

This is easy to see if you look at the offsets of the three notes within a major chord, which are [0 4 7]. So, starting the chord from middle C (midi 60) we get midi notes 60+0, 60+4 and 60+7. These offsets we also call intervals.

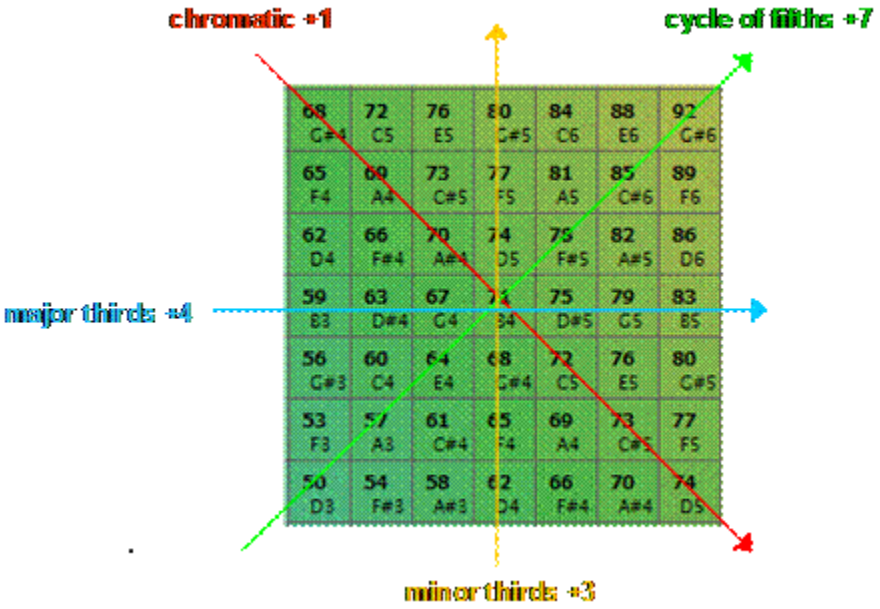
If we draw a minor chord (with intervals [0 3 7]) , it will have a vertically mirrored L-shape:

62 D4	66 F#4	70 A#4	74 D5	78 F#5	82 A#5
59 B3	63 D#4	67 G4	71 B4	75 D#5	79 G5
56 G#3	60 C4	64 E4	68 G#4	72 C5	76 E5
53 F3	57 A3	61 C#4	65 F4	69 A4	73 C#5

What's so useful about this, is that compact shapes in the space (shapes consisting of cells close to eachother) apparently correlate with interesting chords. We can now turn this around : if you would draw a compact shape at random in the space, there is a good chance that this shape turns out to be a useful chord. This is one of things the tonespace chord fitting algorithm is based on. More about that later.

Navigation along the axes of the space

Another interesting property of the Balzano space is that one of its diagonals turns out to be the chromatic scale (1 semitone increase) and the other diagonal appears to be the cycle of fifths (7 semitone increase). This means you can navigate (move the mouse) easily along a chromatic sequence and along the cycle of fifths simply by just following the diagonals. Plus there is a useful progression on the horizontal and vertical axes too, in steps of major and minor thirds respectively (4 and 3 semitone increases resp.).



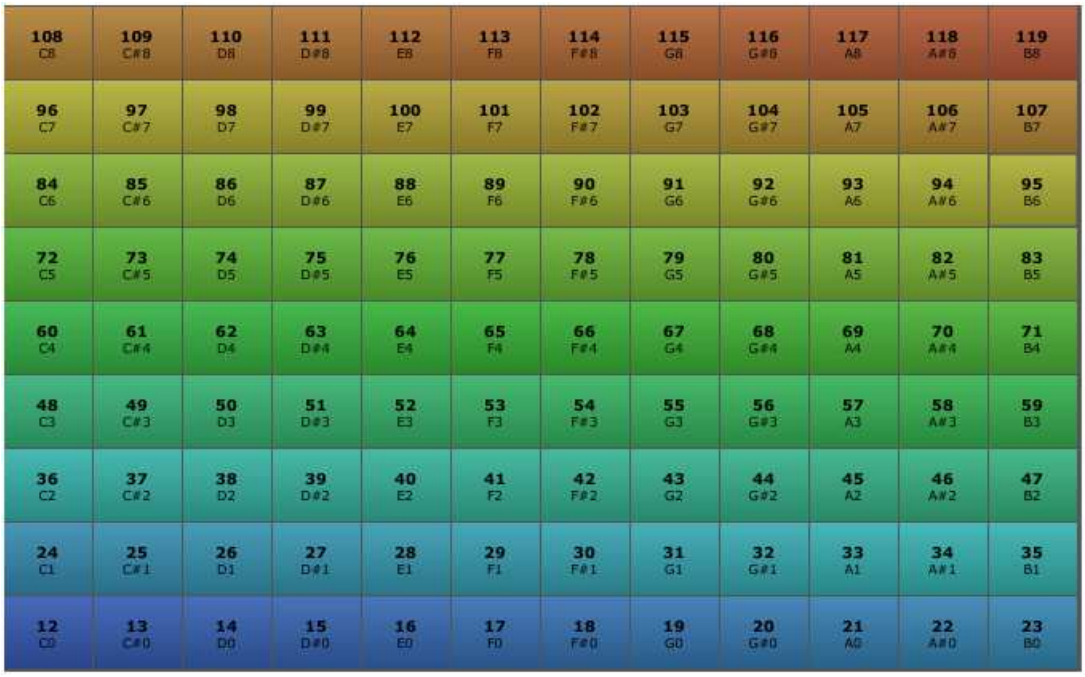
The Longuet-Higgins space has comparable properties, which we will not describe in detail here.

You can find a much more thorough treatment of the theory behind these spaces and a description of an earlier implementation of these principles, called Harmony Space, in the paper by Simon Holland [Holland, 1987]. Note that while tonespace is greatly indebted to the work on Harmony Space, the tonespace implementation deviates in some respects from what is described in this paper, for instance by generalizing spaces to other intervals besides Longuet-Higgins. But the general principles still apply.

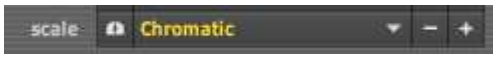
We will now explain a bit more about scales, keys and chord intervals.

Understanding scales

Before moving on to scales, let's first select a suitable space. Among the selectable spaces there is one that is particularly useful for understanding the concept of a scale, which is Octaves [1:12]. This space has a row for each of the 9 octaves, with each octave containing 12 semitone columns.



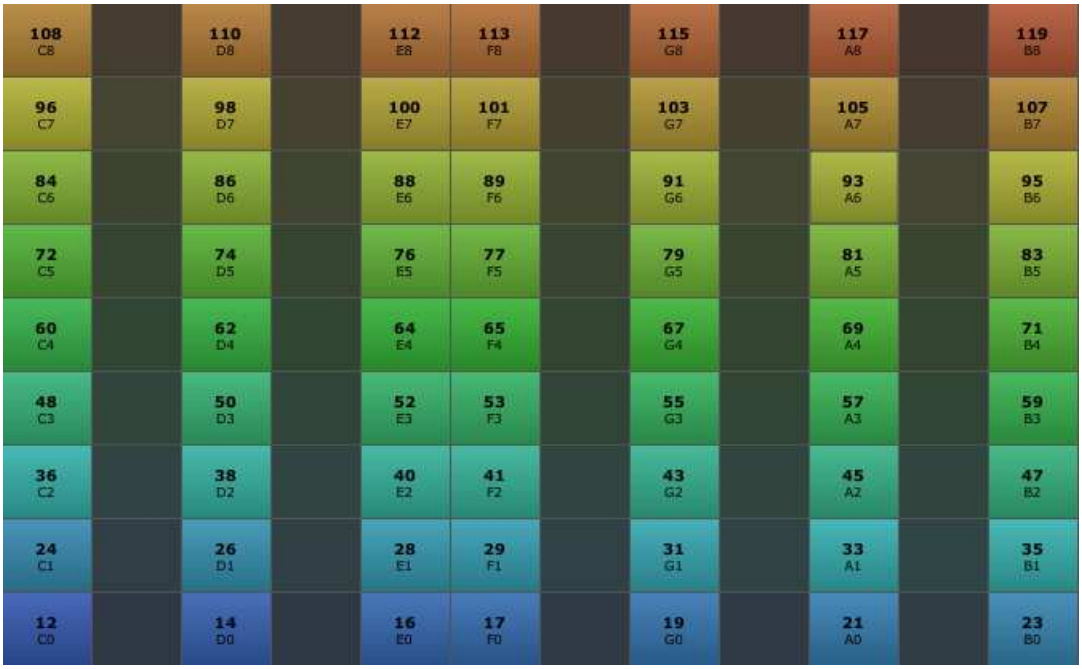
Now let's choose a scale to apply to this space. You can control the scale using the scale parameter in tonespace.



Shown above is the chromatic scale, which is just a fancy word for saying that all notes are allowed for playing [2]. So there are no black cells here, only colored ones.

Musicians do not often use the chromatic scale, because if you would use all of these notes, it is possible to select two or more notes that do not sound well together (they are dissonant). Therefore, it is standard procedure to throw away a bunch of notes from each octave and only work with the remaining set. Such a set we call a scale. Notes within a scale tend to sound pleasant when played together.

For example, let's choose another scale called the Major scale, also known as the Ionian scale. By applying the major scale, some columns are blackened in the space. These won't react any more when you click them with the mouse. Their notes are forbidden.



The picture above makes clear what a scale really is : it is a set of note offsets, or intervals, that defines which are the "good" notes. For the major scale these intervals are [0 2 4 5 7 9 11] [3]. When these seven intervals are applied to the octave of middle C, it means we only get to keep midi notes 60+0, 60+2, 60+4, 60+5, 60+7, 60+9 and 60+11.

The major scale happens to contain 7 intervals, yielding 7 midi notes when added to the midi start note of the octave. Many other scales do too. But there are also scales with just 5 notes (e.g. pentatonic scales). There is no hard rule about how many notes there can be in a scale. You have to find a balance: if there are too many, dissonances occur more easily. If there are too few, less variation can be used in the chords and melody.

Understanding keys

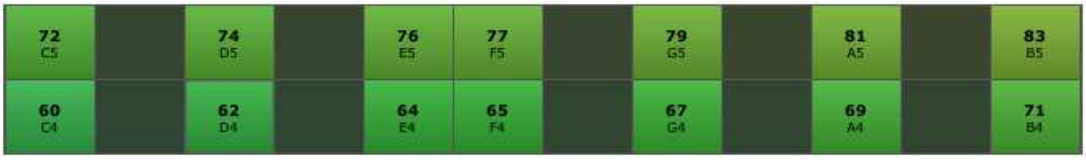
So musicians that play together have to agree first on the scale to use (if they want to avoid playing notes that don't go well together). But this is not enough. They also have to agree on the key they will use. The key (in tonespace at least) is just a fancy word for the starting note or better, the starting pitch class of the scale [4]. A key of C means we start at the first pitch class (being zero). A key of C# means we start at the second pitch class (being 1). A key of D means the third pitch class (being 2). And so forth.

You can control the key using the key parameter in tonespace.

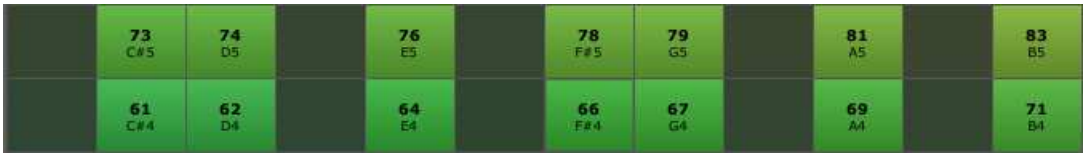
Note: when you hear someone saying : "let's play in the key of C Major", actually this says two things at once:

- the scale to use, i.e. the intervals between valid notes (Major)
- the starting pitch class (C or offset 0)

This means that, starting from C (offset 0), you will use only notes that you find at the relative offsets [0 2 4 5 7 9 11]. For the key of C that starting pitch class would be 0, therefore, for the middle octave which starts at midi 60, we get to keep midi 60+0+0, 60+0+2, 60+0+4, 60+0+5, 60+0+7, 60+0+9 and 70+0+11. This is shown in the space fragment below:

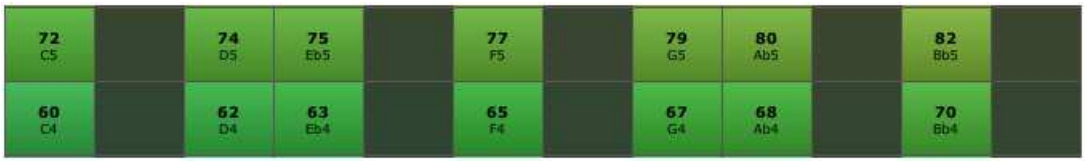


Should you instead agree on D Major, then the intervals would remain the same (major), but the starting pitch class would be different (offset 2), leading to the following midi notes in the scale/key : 60+2+0, 60+2+2, 60+2+4, 60+2+5, 60+2+7, 60+2+9 and 60+2+11. The result looks like this:



Notice how you get the same pattern of allowed notes starting from either C in the first picture and starting from D in the second picture. It is just shifted upward by 2 semitones.

Should you agree instead on C Minor, then the starting note remains the same as C Major (offset 0), but now the intervals that you add to that starting note will be different (minor) : [0 2 3 5 7 8 10], yielding midi notes 60+0+0, 60+0+2, 60+0+3, 60+0+5, 60+0+7, 60+0+8 and 60+0+10. Which looks like this:



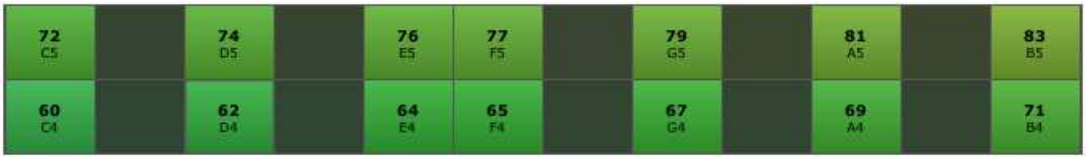
So keep in mind that the resulting note selection is always a function of these two things : the scale (the intervals) and the starting note to which the intervals are added. You are encouraged to play a bit with the scale and key parameters in tonespace to get a feel of how scales and keys interact to select a bunch of notes.

Notice also how the underlying space never changes. Applying a scale and key is in fact just putting a filter onto the space, making certain cells black and adjusting the note labels in each cell. It does however not alter the location of the midi notes within the space. Midi notes are an absolute pitch notation whereas scales, keys and note names are relative to the point of view of the musician [5].

Understanding scale degrees

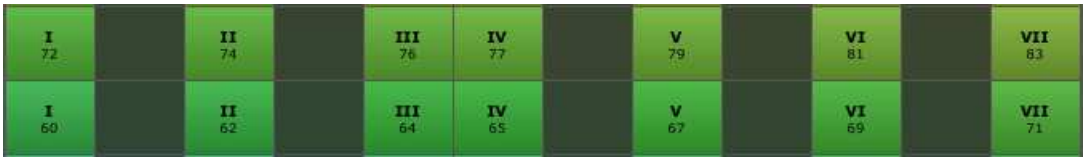
Often you will encounter the set of roman numerals I to VII in music literature. What are these? Very simple, they are nothing but names for the the seven notes in a scale (assuming you use a scale with seven notes – 5 note scales use only I..V). They are called scale degrees.

For instance, look again at the notes in the C major scale: they are C, D, E, F, G, A and B respectively.



Now let's display them as roman numerals You can do this by selecting the appropriate value in the labels combobox:

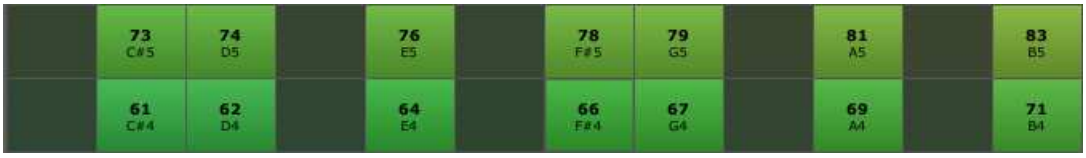
The result looks like this:



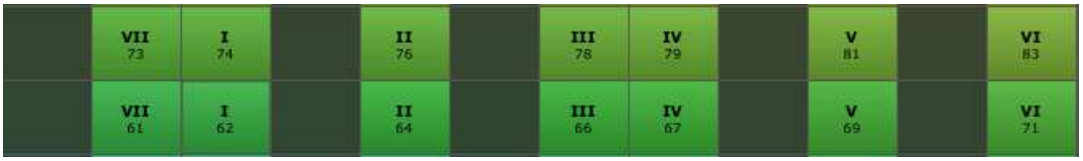
This the same scale, the same key, but just different labels. Why bother then? Well, this allows you to indicate the n-th note of a scale regardless which type of scale/key you are using.

For instance compare with D Major:

Original notation: D, E, F#, G, A, B, C#



And its roman numeral notation:



Did you notice how the I .. VII set has shifted two positions to the right? That's because

we now start at D instead of C.

Why is scale/key-independent note labeling useful? One example is when you want to specify the root notes of a series of successive chords to play (called a chord progression) within a scale/key, without knowing up-front what that scale/key is. That way it suffices to write down the chord progression just once as roman numerals (e.g. I–III–IV) which you can then apply to a myriad of scale/key combinations later.

Understanding chords

Okay, you now know how to choose a space to navigate in. You also know how to limit the number of allowed notes in that space by choosing a scale and key. And you are able to adjust the note label display as needed. Now let's examine the last bit that's missing from the picture : chords.

Like a scale, a chord is defined by a set of intervals. However, the number of notes in this set is typically much lower, with three or four being common.

We can classify chords after the number of notes in them:

- monads: single note (or unison)
- dyads : have two notes
- triads : have three notes
- sevenths: have four notes (also known as tetrachords)
- quintads : have five notes

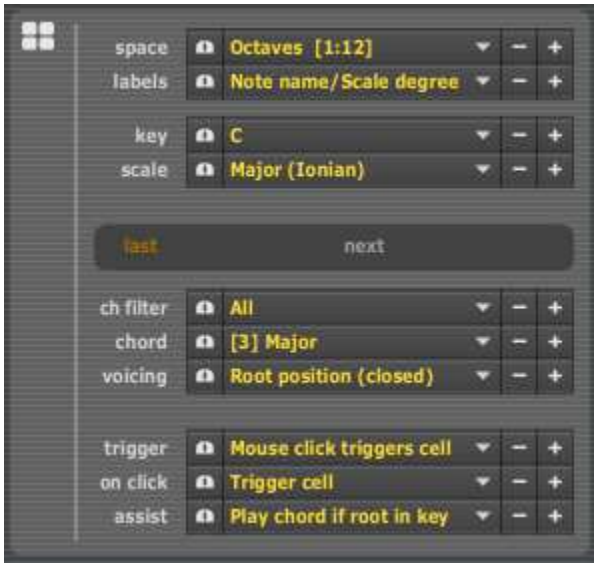
In tonespace 2 we support monads, dyads, triads and sevenths only.

You can select a chord in tonespace by setting three parameters. First set the ch filter parameter to All, which lets you choose from all possible chords.

Then set the assist parameter to Play chord if root in key. This will let us play the chord if we click on a cell that is in the current scale/key.

Then set the chord parameter to any of the chords in the combobox. For instance let's set it to [3] Major:

A major chord is a triad defined by the intervals [0 4 7]. Let's see what happens when we try to play this chord on our space. To make things easy, set your other parameters as follows :



Note: click also once on the little mouse wheel symbol before the chord combobox. This will become useful in a moment.

Now point your mouse in the space at midi note 60. You should see this :

The grey highlighted cells indicate how your chord maps onto the space. Since the major chord was defined by intervals [0 4 7] and you pointed at midi 60 as root note for the chord (starting note), it will highlight midi notes 60+0, 60+4 and 60+7. The highlights are not only in the grid, but also on the piano keyboard. This makes it really easy to see the relationship between the space cells and the piano keys.

Now press the left mouse button. If tonespace is set up right, you will now hear a major chord. The grey cells and piano keys should become orange too as long as the chord is sounding.

One thing you can spot visually immediately is that the major chord with root note midi 60 (C4) maps nicely onto the C major scale/key. This is no coincidence, since the major chord intervals are a subset of the major scale intervals.

Now move the mouse to midi note 62 (notice how you cannot move it to the black cell midi 61, since this is not part of the scale/key). You will now see these highlights (rest of screen omitted) :

Now we have a problem: the chord does not fit entirely onto the scale/key anymore, since the middle interval maps to a black cell. So if you would play this chord, the middle note would be a forbidden note. Whether that is a problem or not is not a hard rule and depends really on your intention and preferences as a composer, but let's say for simplicity that you would really like to stick to notes within the selected scale/key. In that case we should change the chord to something else that does fit at that root note.

What could we use then? Well, go back to your chord parameter and select [3] Minor instead of [3] Major. The minor chord is a triad defined by intervals [0 3 7]. Now point again to midi note 62 within the space. That should give you :

Now you can see that this chord does fit into the C major scale/key at that position.

This is one of the uses of tonespace : a visual tool that helps you spot easily how a range of chords map onto a given scale and key. And then shows you how that chord maps back onto a piano keyboard, so you could play it on an external keyboard.

Mouse wheel technique

To come back to the litte mouse wheel button : there is a quick way to select different chords while navigating in the space : just roll the mouse wheel and you will see that the chord parameter is being varied without you having to leave the space. In fact you can vary any parameter in this way by clicking its corresponding mouse wheel button first.

Try it :

We assume you have set the ch filter parameter to All and the trigger parameter to Mouse click triggers cell.

Then set the chord parameter to the very first value in of the combobox list, being [1] Unison (this means no chord but a rather a single note) Then point in the space and left-click once. After having done that, roll down your mouse wheel one step. Left-click again. Repeat this until you went through all of the chord values. What you will hear is a chord progression starting from unison, through all available dyads, triads and tetrachords/sevenths in that order.

Automatic chord fitting

Up to now we discussed mostly manual selection and auditioning of chords. However, tonespace can also help to automatically select appropriate chords. This process we call *chord fitting*. Two chord fitting algorithms are available.

Fitting chords to a scale/key

As explained earlier in Quick Start Scenario 4, a new feature of tonespace is that it can automatically select the best chord (most common chord) that fits at a certain root note in the scale. You point the cell with the chord root note that you wish and tonespace will try to fit a chord at that root note that is in key with the currently selected scale/key.

For this you need to set the assist parameter to Fit chord to scale (best). Variants of this parameter let you also cycle through all candidate chords by clicking repeatedly on the same root note cell (Fit chord to scale (cycle)), which is great for auditioning all chords that fit. Or let you select a random chord from the possible candidates (Fit chord to scale (rand)), which is useful for generating repetitive chord sequences that are less predictable.

Please note that at all times the range of chords that tonespace can use can be controlled using the ch filter parameter. This allows you for instance to only generate triads or only sevenths or anything up to triads or sevenths.

Geometric chord fitting

Tonespace can also perform automatic chord fitting using a geometric algorithm. In this mode tonespace will try to find chords that fit optimally wherever you point in the space. Here chords are selected based on their shape within the space. The chord with the most compact shape "wins" and will be played.

For automatic geometric chord fitting to generate interesting chords, you'd better select another space than the educational space Octaves [1:12] that we have been using so far. For instance let's take Balzano [4:3] instead [6]. Also keep scale set to Major (Ionian) and key to C.

One more essential setting here is to set assist to Fit most compact shape and also to set the ch filter parameter to Sevenths.

Note how the chord parameter combobox has been disabled in that case. This is to indicate that the chord parameter now is driven by tonespace instead of by the user. The chord will be automatically selected from all chords allowed by the chord filter. In this case we limit chords to sevenths or four-note chords. You can see which chord is played/selected by the last/next chord display indicators.

Now go to the space and play (left-click) the following series of cells: midi note 64, 65, 62, 50 and 53. You will hear a progression of 5 seventh chords chosen by tonespace, with the root notes being the cells that you left-clicked.

By setting the chord filter to other chord types, you can have tonespace play triads or dyads instead of sevenths.

One more thing. If you try the example above with sevenths and you click the same cell repeatedly, you will see that the chords will not be the same everytime. This is because at some spots there would be multiple chords that would fit the space equally well (that is, they are equally compact). Tonespace then alternates between these equally appriopriate chords. This can give a very pleasant effect if you use a pulsing note-on/note-off scheme to trigger cells (like the built-in MIDI track Pulses... does). If you are interested, read more about how this algorithm works in the next section.

Varying the space and scale

The above example used the [4:3] Balzano space. As you will now understand, this space favors chords that have either 4 or 3 or 4+3=7 semitones distances between them. These lead to quite common chords, like major and minor ones. Its sound we would describe as 'classic'.

You can influence the type of chords that are selected by the gemoetric chord fitting algorithm by changing the space. For instance, try the same procedure above using the spaces [2:5] or [2:7]. This will select another kind of chords, favoring distances of 2, 5 and 7 semitones. These chords sound darker and are more interesting for certain kinds of electronic music.

Another way of influencing chord selection is to change the scale. For instance, try the blues scale and the 5-note scales (pentatonic scales). By lowering the number of allowed notes in a scale, the chord fitting algorithm has to adapt to that.

Chord voicings

Yet another way to add variety to chords is to change the chord voicings. In tonespace we use the term voicing to indicate that we transpose individual chord notes by multiples of 12, so by 1 or more octaves. This leaves the character of the chord intact –it will still sound like the same chord- but will allow you to either transpose the chord (or parts of it) up or down and/or give it a more open sound.

Voicing is controlled using the voicing combobox. The default voicing is set to Root position (closed). This means that the cell you click will become the lowest or root note of the chord and the other notes will be cells that coincide exactly with the other intervals in the chord.

A number of other voicing types are available:

- Inversions
- Open voicings
- Root note transpositions
- Spreading

Inversions take the root note of the chord and put it at the top of the chord (i.e adding an octave to the root note). This is the first inversion. You can repeat this same trick with the inverted chord, leading to the second inversion (i.e. adding an octave to the second note of the original chord). And so on. Note that the fourth inversion requires at least a seventh chord (with 4 notes) to make a difference. Applying fourth inversion to a triad (or third inversion to a dyad), will not make a difference.

Open voicings take some *other* note than the root note of the chord and add an octave to that one. This will widen the total interval between lowest and highest chord note, making it sound more 'open'. When no open voicing is used the chord takes its normal 'closed' form.

Inversions and open voicings can be combined, by first inverting and then further opening up the inverted chord. Tonespace offers a combination of 4 inversions and 4 closed/open voicings.

Root note transpositions allow you to move the root note of the chord up or down by one or more octaves, while keeping the non-root notes in place. This is a quick technique for adding a bass line to the chord progressions. In tonespace you can transpose the root by -1 octave, -2 octaves and +2 octaves.

Spreading: tries to space the notes of the chord even more apart than with Open/Closed voicings. In tonespace you can choose from three settings, called Spread, Spread+ and Spread++.

Tip: while playing in automatic chord fit mode, you could assign the mouse wheel to the voicing parameter. That way, you can vary the voicing of the chord while you play. If you'd like to get a good visual idea of what voicing is about, set your space to Octaves [1:12] and then spin the mouse wheel : you will see how certain chord notes jump up and down vertically to another octave.

Note that when voicing is set to something other than Root position, the highlighted cells in the space will show the chord after applying voicing. This has no effect however on

the chord fitting algorithms, which are applied before voicing.

MIDI input

There is one more tonespace aspect we haven't dealt with yet : MIDI input. So far we have been mostly using the mouse to play chords. However, it is also possible to use an external keyboard, or for that matter, any external source of MIDI data that you can connect to the tonespace MIDI input. A MIDI track in your host application would be another example of this.

When you feed MIDI data to tonespace a number of things can happen, depending on the setting of the trigger parameter:

- If trigger is set to Mouse click triggers cell, the MIDI input merely forwarded either to the tonespace MIDI out port or to the built-in synth (depending on the midi out parameter setting). Apart from that it will also highlight the cells corresponding to the incoming MIDI notes. These highlights will be green, in order to distinguish them from the interactively played cells, which will be highlighted in orange. If a space contains more than one cell with the same MIDI note number, all of those will be highlighted. This mode can be used to visualize incoming midi on the tonespace grid. Also, if you want to control tonespace completely using mouse clicks, this is the mode to choose.

- If trigger is set to Midi triggers clicked cell, then if the on click parameter is set to Trigger cell, tonespace will use the incoming MIDI notes to trigger a chord in the cell where you are currently left-clicking and holding down the mouse button. If the on click parameter is set to Trigger & hold cell then you do not have to keep the mouse button down, and the last-clicked cell remains active until you click a new cell.
  What is really useful about this mode is that you can get the rhythm and dynamics (velocity) from an external keyboard or even a MIDI track, while still navigating interactively using the mouse for selecting the pitches to be played (ie select the cells you want with their root chord notes). This is a good setting to try out with the built-in Pulses... midi track. Please note that the incoming midi should be preferably monophonic in this mode.

- If trigger is set to Midi triggers hovered cell, this is very similar to the previous mode, but here it merely suffices to hover the mouse pointer over a cell to have the incoming midi notes trigger that cell.

- If trigger is set to Midi triggers note then the incoming midi controls both the rhythm and the pitches. This means that tonespace will look up an enabled cell in the current space that has the same note number as the incoming midi note and it will trigger that cell. This means that mouse input will have no longer any effect in the tonespace grid, and you should now do all the playing using an external midi keyboard or midi track. The only effect the mouse still has in this mode is that, when there are multiple cells in the space that have the same midi note number, tonespace will select the cell closest to your mouse cursor.

- If trigger is set to Midi triggers note (mod.) then the behavior is almost identical to the previous setting, with one difference: the incoming midi note will be assumed to be played in the CMajor scale (so white keys on the piano keyboard) and will first be modulated/transposed to the selected scale/key in tonespace before it will trigger any chords. This allows you to play the white keys on a keyboard and still generate chords that are in key with another scale/key.

When using midi input, please make sure that also the midi in parameter is set to the right

source and channel. Typically midi in is set to:

- Midi track if you want to use one of the built-in midi tracks as the source (eg the Pulses... track)
- VST/AU host if you want to input midi coming from your host application/DAW
- <your device name> if you want midi input to come from a hardware midi device such as an external keyboard.
  Note: before devices can be selected you need to enable them using the Configure... option on the midi in parameter drop down menu.

MIDI output

As said in the introduction, any chords generated by playing cells will be sent either to the tonespace MIDI output port or to the built-in synth.

While you can send this MIDI out straight to a synth, there are also a few techniques for post-processing tonespace MIDI output before sending it to the synth. These are general techniques and not part of tonespace itself.

One of those techniques is to send the output to an arpeggiator plug-in. If your arpeggiator has a chord modus, it will take tonespace chords and arpeggiate them. This arpeggiated output is then routed to a synth. This can lead to very interesting sounds.

A second technique is to send the output chords to a strummer, which is quite useful if you like to feed tonespace chords to a guitar emulation synth.

Of course it should also be possible to record tonespace output to a MIDI track in your host, if your host supports this.

Footnotes

[1] (M2 means Major 2nd which equals 2 semitones, and P4 means Perfect 4th which equals 5 semitones).

[2] When we say all notes we mean all typical notes in our Western 12-tone, equal temperament system or 12TET, which is the default tuning of most midi systems.

[3] Tip: you can hover over the space combobox to show a tooltip explaining the intervals for any chosen space.

[4] A pitch class is an offset within the octave. It is simply a number between 0 and 11. The pitch class of any midi note is its midi note number modulo 12.

[5] maybe you observed already that there are more keys in tonespace (18) than pitch classes (12), meaning that two keys can end up starting at the same pitch class or offset. For instance, the keys of C# and Db both start at pitch class 1. Therefore the colored grid will be exactly the same for both, with the same pattern of colored and disabled cells. Why bother then in distinguishing these two keys at all? Well, it appears that musicians do make a distinction between two such keys, but the distinction only affects note naming, but not which notes are allowed. So it affects not which absolute pitches are included in the key, but only the labels applied to those pitches. Again this has to do with the fact that traditional note names are just a viewpoint of a musician on an underlying world of absolute pitches.

[6] This is because the geometric chord fitting algorithm will try to make 'compact' shapes in the space, i.e. will try to make a shape consisting of cells that are close to eachother. For this to sound pleasantly, both the horizontal and vertical semitone distance between nearby cells should best be equal to common semitone distances between chord notes. With a [1:12] space this is not the case, since not many chords have 1 or 12 semitones distances between their notes. With a [4:3] space it is.

Parameter reference

The following parameters can be set in tonespace:

- **PRESET**. Selects the preset program to use. In hosts that support it you can rename programs yourself and save/load either single presets or complete preset banks to/from .fxp, .fxb files respectively.

- **SPACE**. Selects the space to use in the central grid. Spaces differ in how they distribute midi note numbers across the grid. Generaly the [N:M] notation means that the grid increases by N semitones horizontally and M semitones vertically. See also Introducing spaces.

- **LABELS**. Controls how cell labels are displayed within the space. A cell label always has a main label and a subscript. For either main label or subscript you can select
  - MIDI note number
  - Note name (only for cells that are part of the current scale/key)
  - Roman numeral for scale degree (only for cells that are part of the current scale/key)

- **KEY**. Selects the key or starting note from which we will apply the SCALE intervals. You can hover the mouse over the combobox to see which interval set corresponds to each possible value. The KEY parameter works in conjunction with the SCALE parameter for enabling/disabling notes within the space. See also Understanding keys.

- **SCALE**. Selects the scale or set of note intervals to apply to the space. You can hover the mouse over the combobox to see which interval set corresponds to each possible value. The SCALE parameter works in conjunction with the KEY parameter for enabling/disabling notes within the space. See also Understanding scales.

- **CH FILTER**. Restricts the kind of chords that can be selected by the user and by the automatic chord fitting algorithms. You can turn the filter off by setting it to All. Typical filter settings allow to use dyads, triads and sevenths only or up to dyads, triads and sevenths.

- **CHORD**. Shows the currently active chord name. This parameter can only be set interactively when the ASSIST parameter is set to any of the manual modes (Play chord ...). If ASSIST is set to any of the automatic chord fitting modes, the CHORD parameter will be overridden by tonespace. See also ASSIST.

- **VOICING**. Controls transposition of chord member notes by multiples of 1 octave. Examples are inversions, root note transpositions and making chords open/closed. For each possible value you can see the exact transpositions used by hovering above the combobox. See also Chord voicings.

- **ASSIST**. Controls whether the user can select chords manually (Play chord ...) or tonespace will select chords automatically (Fit ...).
  - Play chord anywhere: the chord root can be played anywhere in the space (by clicking any cell, even when cell is disabled)
  - Play chord where root in key: a chord can only played if the root note (cell) is part of the the active scale/key (only on enabled cells can be clicked)
  - Play chord if fully in key: a chord can only be played if all of its notes (cells) are part of the active scale/key (all chord cells are enabled cells)
  - Fit most compact shape: tries to fit a chord automatically at the current root cell, using the geometric chord fitting algorithm
  - Fit chord to scale (best): tries to fit a chord automatically at the current root cell, using the common chord fitting algorithm, and selecting the best candidate chord.
  - Fit chord to scale (cycle): like previous mode, but cycles through all candidate chords when repeatedly clicking the root cell
  - Fit chord to scale (rand): like previous mode, but picks a random chord from all fitting candidate chords each time the root cell is clicked

  See also Understanding chords and Automatic chord fitting.

- **TRIGGER**. Controls how mouse/MIDI input affects triggering of cells in tonespace.
  - Mouse click triggers cell: the MIDI input is forwarded either to the tonespace MIDI out. Apart from that it will also highlight the cells corresponding to the incoming MIDI notes. In this mode tonespace can be fully controlled by mouse movements and clicks.
  - Midi triggers clicked cell: tonespace will use the incoming MIDI notes to trigger a chord in the cell where you are currently left-clicking and holding down the mouse button. If the on click parameter is set to Trigger & hold cell, then you do not have to keep the mouse button down.
  - Midi triggers hovered cell: this is very similar to the previous mode, but here it merely suffices to hover the mouse pointer over a cell to have the incoming midi notes trigger that cell.
  - Midi triggers note: the incoming midi controls both the rhythm and the pitches, triggering a cell corresponding to the incoming midi note number. In this mode tonespace can be completely controlled from an external keyboard or midi track.
  - Midi triggers note (mod.) :the behavior is almost identical to the previous setting, with one difference: the incoming midi note will be assumed to be played in the CMajor scale and will first be modulated/transposed to the selected scale/key in tonespace before it will trigger any chords. This allows you to play the white keys on a keyboard and still generate chords that are in key with another scale/key.

  See also Midi input

- **ON CLICK**. Controls how mouse clicks trigger chords (parameter is enabled only when mouse clicks can trigger cells). Please note that with the standalone tonespace version the space bar acts as a mouse click as well. Some, but not all plugin hosts will also allow the VST/AU version of tonespace to use the keyboard in which case the space bar can be used.
  - Trigger cell: a cell is triggered while clicking that cell and holding down the left mouse button
  - Trigger & hold cell: a cell is triggered when clicking that cell and keeps

playing that cell until you click either a new cell, or you click a disabled cell (stops previous cell).

- **MIDI IN**. Selects the source for incoming midi. You can select either an external midi device (or loopback adapter), the VST/AU host in case of the plugin version or one of the built-in midi tracks.

  See also Midi input

- **CHANNEL (input)**. Selects the midi input channel were tonespace listens on

- **MIDI OUT**. Selects the destination for outgoing midi. You can select either an external midi device (or loopback adapter), the VST host in case of the plugin version or the built-in synth. Note that not all VST hosts and no AU hosts support midi output from a plugin.

- **CHANNEL (output)**. Selects the midi input channel were tonespace sends midi to

- **GAIN**. Controls the volume of the built-in synth