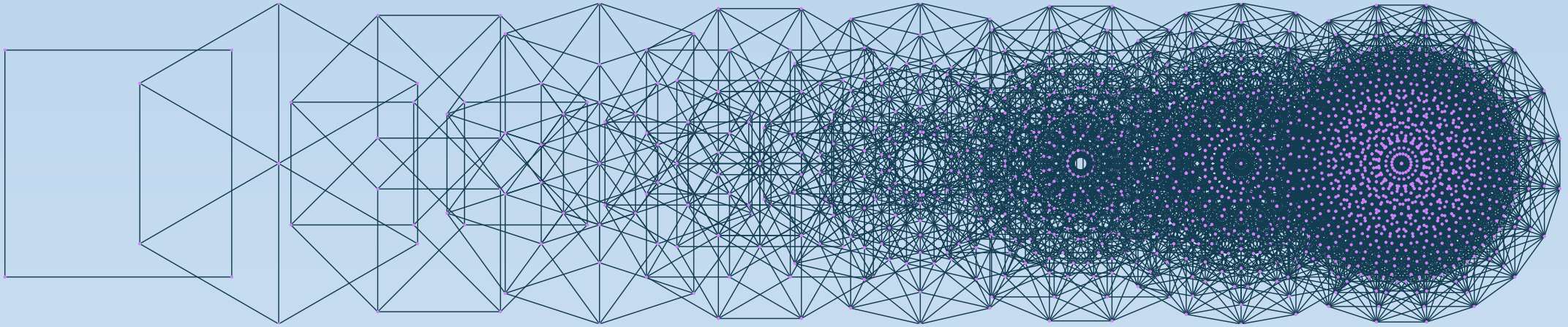


Tesseract Plugin

by Überton

User Guide

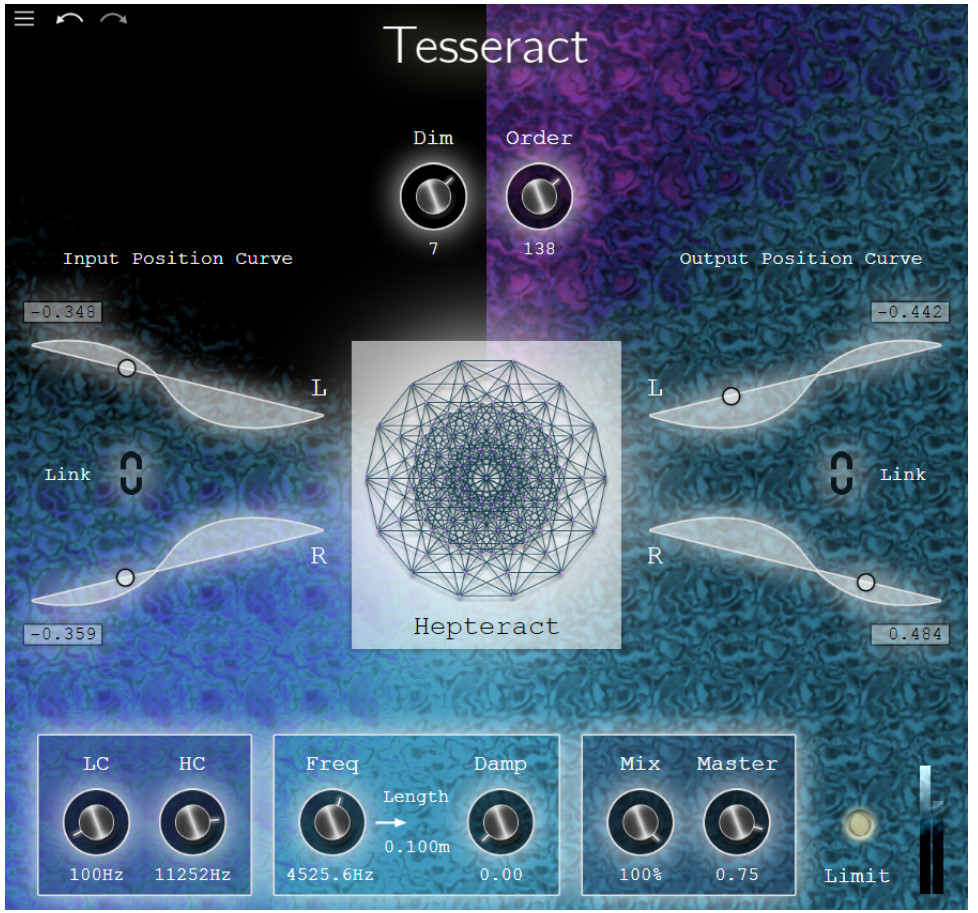
Überton

Contents

Introduction

Thanks for downloading Tesseract. For best (and savest) usage, please read through this manual.
This user guide contains three sections:

- a practical manual that explains how to use the plugin;
- physical interpretations;
- and finally, a detailed mathematical description of the background of the implemented algorithms.



1 Manual

1.1 Safety Warning

Caution: The very nature of this plugin is the unrestrained use of strong resonances!

Be careful with expensive speakers as this plugin can produces massive bass when the resonance frequency is set to low values.

Start at low volumes and always use a limiter after this plugin: either activate the built-in one or insert another limiting plugin after the Tesseract in the signal chain. Also use the low- and high-cut filters or add third party filters to tame the beast.

Even with these precautions be a bit careful because very high signal levels will produce considerable distortion when run through a limiter.

Moreover, some parameters can cause audible cracks during changes when the resonance frequency is set to low values. This is partly due to high levels in the bass range, where oscillations are slow and even small changes cause substantial discontinuities in the signal. On the other hand, imagine a sudden change of the world's dimension from e.g. five to nine. Of course this is a drastic rupture of spacetime accompanied by audible artefacts.

Überton is not liable for any damages that result from the use of this plugin.

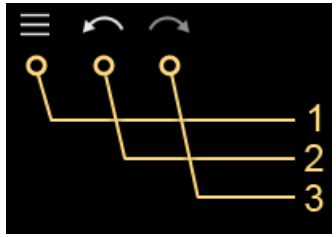
1.2 Concept

Tesseract simulates an ideal resonator like a perfect guitar string. A guitar string has certain modes or frequencies that it can vibrate on – the resonance frequencies. The same applies to a metal plate, a hollow sphere or any other geometric form. This concept can be generalized to higher dimensions. In the world we live in, the story ends with three dimensions but mathematics gives us formulas to compute resonances of bodies in arbitrary dimensions and software can easily execute these algorithms in the bounds of processing power which is limited by the computers resources.

In particular, this plugin models the regular polytopes with perpendicular line segments known as *hypercubes* which are higher-dimensional generalisations of the three-dimensional cube. The plugin borrows its name from the four-dimensional hypercube, the *Tesseract*.

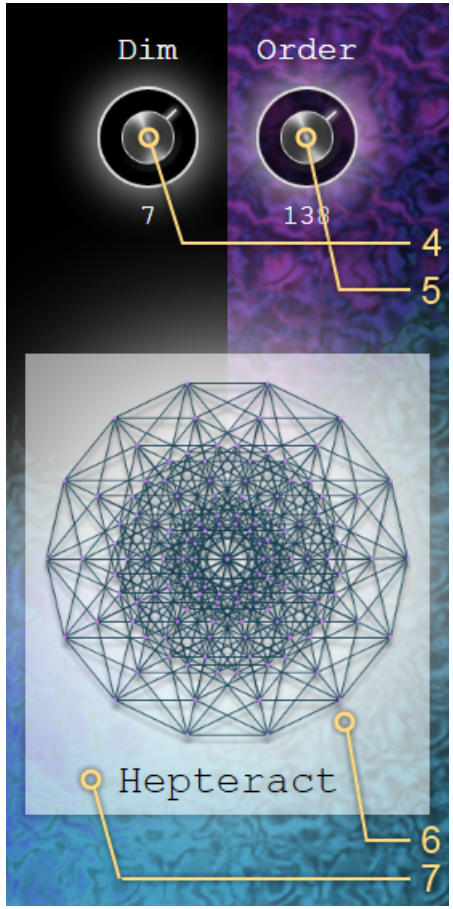
1.3 Interface and parameters

1.3.1 Menu controls



1. **Menu Button**: Open the menu to control the size of the user interface, open this manual and visit the website of Überton.
2. **Undo Button**: Überton allows you to undo the most recent 200 parameter changes.
3. **Redo Button**: Redo any undone changes.

1.3.2 Dimension and order



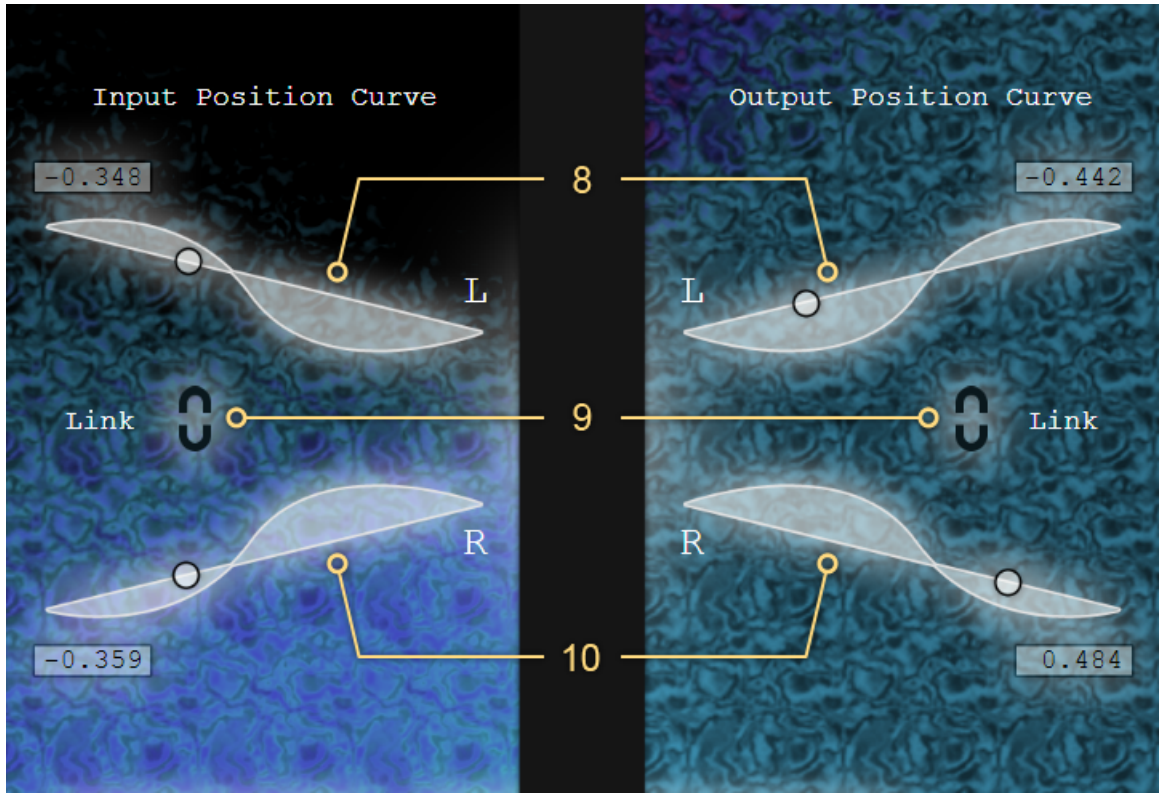
4. **Dimension**: Change the dimension of the hypercube resonator. The dimension can be set to integers from 1 (a one-dimensional string) to 10 (a ten-dimensional hypercube).

5. **Resonator Order**: Set the number of evaluated modes of the resonator (see section 2 and 3 for more detailed information). This effectively increases the number of audible frequencies. Setting this parameter to high values requires more CPU power for computation. This is the only parameter that noticeably changes the CPU usage. More specifically, the amount of computation time increases with the resonator order. Setting this parameter a bit lower may solve issues with cracks during real-time playback.

6. **Hypercube Graph**: Here, a projection of the currently selected (through the dimension parameter) hypercube is displayed.

7. **Hypercube Name**: Apart from the known one-dimensional string, the two-dimensional square and the three-dimensional cube, the hypercubes of dimension four to ten also have special names which are shown in this field.

1.3.3 Input and output position curve parameters



8. **Left In and Out Position Curve Parameters**: The left channel of the input signal is fed into the resonator at a certain position in d-dimensional space that can be controlled through the input position curve parameter. The left channel of the output is taken from the position in d-dimensional space specified by the output position parameter. You could also say that the input audio is played inside the hypercube at the input position and listened to from the output position. Changing these parameters basically changes the ratio of the loudness between individual resonance frequencies. For example, at some points in space, a sound wave of certain wavelength has knots where the amplitude vanishes.

9. **Link**: Linking the left and right curve parameters enables dragging both at the same time and by the same amount.

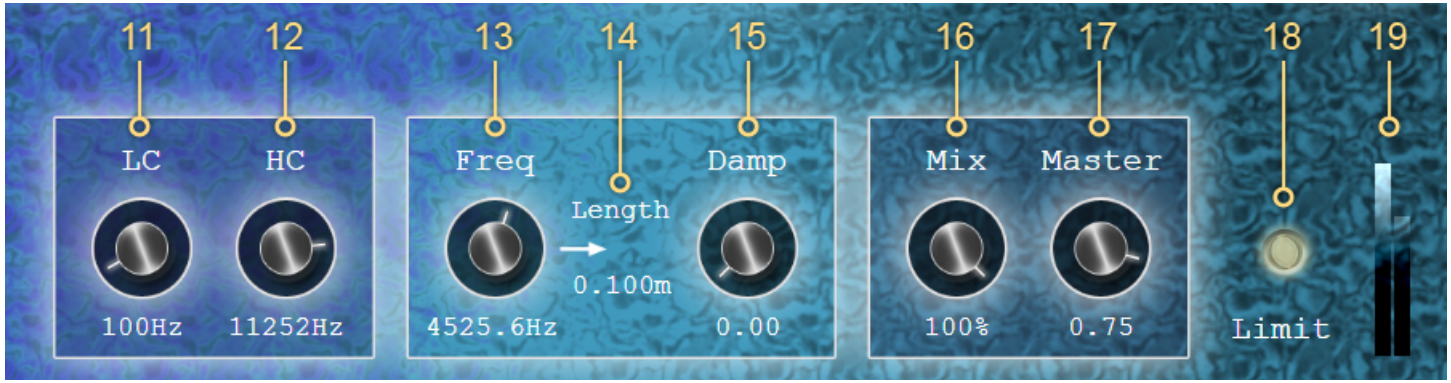
10. **Right In and Out Position Curve Parameters**: These work just like the others but for the right channels respectively.

To control four position vectors in up to ten dimensions, you would need $10 \cdot 4 = 40$ parameters. With all these, the interface would look a total mess and the controlling would be confusing and tedious.

Instead, each position is controlled by only one parameter which sets it according to a curve in (up to) ten-dimensional space. Changing the parameter means moving along this curve through d-dimensional space.

If needed, you actually have access to the mentioned 40 parameters. They are hidden but accessible through automation lanes in your DAW or by right-clicking on this plugins interface and choose *Switch to generic editor* (or similar, depending on your DAW).

1.3.4 Other parameters



11. **Low-cut Frequency**: Engage the low-cut filter to remove low frequencies below the low-cut frequency.
12. **High-cut Frequency**: The high-cut filter can be used to remove frequencies above the high-cut frequency.
13. **(Lowest) Resonance Frequency**: Control the frequency of the resonances in the hypercube by setting the frequency of the lowest resonance.
14. **Hypercube Size**: The edge length of the hypercube is derived from the set lowest resonance frequency, the dimension d and the dampening factor. This is a read-only parameter that displays the size in meters.

15. **Dampening Factor**: This parameter controls the absorbance of the resonator, or rather the "air" in the resonator. Setting this parameter to zero will remove all absorbance and each excited frequency will sound forever.

16. **Wet/Dry Mix**: The mix parameter allows to blend the dry input signal with the wet (processed) output signal.

17. **Master Volume**: Controls the master output volume.

18. **Limiter**: Activate the limiter to prevent values above 0 dB. It is recommended to use this option or insert a third party limiter to prevent speaker (or hearing) damage. When the limiter works hard, the signal is overdriven and new frequencies enter the game. This can be desirable or not. In case of high signal levels, you should probably decrease the master volume.

19. **Output VU Meter**: Displays the output levels of the left and right channel.

2 Physical remarks

In this plugin, the actual modeling of the d-dimensional cube is achieved through what is called *modal analysis*. The system is characterized by determining the spatial eigenfunctions and normal modes as well as the corresponding eigenfrequencies. The response of the system to an applied signal can then be calculated in terms of the eigenfunctions and eigenfrequencies as a linear combination.

In theory, there is an infinite number of possible normal modes for the d-dimensional cube and also infinitely many eigenfrequencies. A computer however has limited computing capacity and can only compute around a few hundred modes in real-time.

This leads to a representation that is not exact but nevertheless interesting for audio manipulation. You could say, the more eigenmodes can be computed, the more correct the physical modeling of the system. In practise, settings with a far less then possible number of eigenmodes also give great results, as they are not as harsh or piercing. Tesseract has a parameter that enables the user to set the number of eigenmodes to be computed: the *resonator order*.

When using the plugin, you will probably notice some effects:

For higher dimensions, there are (seemingly) less overtones: As said above, in theory there are infinitely many overtones, regardless of the dimension but only a number N of normal modes is actually computed. For the one-dimensional string, the order N is exactly the same as the number of overtones that can be heard. For higher dimensions however, the degeneracy of the eigenfrequencies (different modes have the same frequency) increases and although N spatial eigenfunctions are still evaluated, only a considerably smaller number of different frequencies are computed. For ten dimensions the degeneracy is so high that only six resonating frequencies are processed for a total of 200 eigenmodes!

Some frequencies decrease with increasing dimension: Try setting the order to 2, the dimension to 1 and then increasing the dimension while i.e. passing white noise through the plugin. Then you will see the effect. The plugin is designed so that the base frequency ω_1 (lowest frequency) is always the same. But the other frequencies do not keep the same ratio. This can be demonstrated with the second-highest frequency ω_2:

$$\frac{\omega_2}{\omega_1} = \frac{\sqrt{(d-1)+2^2}}{\sqrt{d}} = \sqrt{1+\frac{3}{d}}$$

The formula for ω_i is derived in section 3, Eqs. (4) and (7) and b is set to zero to simplification. The ratio ω_2/ω_1 decreases with increasing dimension d. With fixed ω_1, ω_2 thus is reduced, leading to the described effect.

A word should also be spoken about the term *overtone*. In music theory, most of the time this word is used to talk about *harmonics* of a given fundamental. Harmonics are integer multiples of the fundamental frequency. For example, all eigenfrequencies of the ideal one-dimensional string are harmonics of the lowest frequency. Generally, the term overtone also includes inharmonic frequencies. The ideal vibrating square, and the three or higher-dimensional cubes produce mainly inharmonics, namely square roots of integer multiples of the lowest frequency. Section 3 derives the exact eigenfre-

quencies.

By the way: the eigenvalue problem of the hypercubes and its solutions are completely analogous to that of a quantum mechanical (d-dimensional) well with infinite walls.

3 Mathematical description

In this last section, the underlying mathematics shall be explained and the implemented formulas derived. This is achieved by solving a differential equation through separation of variables.

3.1 Solutions of the wave equation

A free, linear system with dampening is described through the following differential equation:

$$0 = \left(\frac{1}{c^2}\frac{d^2}{dt^2} + \frac{2b}{c^2}\frac{d}{dt} - \Delta\right)\psi(\vec{x}, t), \tag{1}$$

where c describes the propagation speed of a wave $\psi(\vec{x}, t)$ and b is a constant factor of dampening. $\Delta = \nabla^2$ denotes the second spatial derivative in the d dimensions of $\vec{x}$. Note, that b has the unit m^{-1}; this form has been chosen deliberately to get better independence of parameters later. This equation is a wave equation that shall describe audio waves (i.e. through air pressure) in d-dimensional space.

Eq. (1) can be solved using the method of separation of variables. Let's assume $\psi(\vec{x}, t)$ can be separated into a spatial function $\phi(\vec{x})$ and a time-dependent function $\chi(t)$:

$$\psi(\vec{x}, t) = \phi(\vec{x})\chi(t)$$

Then, Eq. (1) takes the following form:

$$\begin{aligned} 0 &= \left(\frac{1}{c^2}\frac{d^2}{dt^2} + \frac{2b}{c^2}\frac{d}{dt} - \Delta\right)\phi(\vec{x})\chi(t) \\ \Leftrightarrow \chi(t)\Delta\phi(\vec{x}) &= \phi(\vec{x})\frac{1}{c^2}\frac{d^2}{dt^2}\chi(t) + \phi(\vec{x})\frac{2b}{c^2}\frac{d}{dt}\chi(t), \end{aligned}$$

because $\chi(t)$ does not depend on $\vec{x}$ and $\phi(\vec{x})$ has no dependency of t. We can rewrite this as

$$\frac{1}{c^2\chi(t)}\frac{d^2}{dt^2}\chi(t) + \frac{2b}{c^2\chi(t)}\frac{d}{dt}\chi(t) = \frac{\Delta\phi(\vec{x})}{\phi(\vec{x})} \equiv -k^2. \tag{2}$$

This equation has to be satisfied for all $\vec{x}$ and t independently and thus must be constant. We call this constant $-k^2$. From the first part

$$\frac{1}{c^2}\frac{d^2}{dt^2}\chi(t) + \frac{2b}{c^2}\frac{d}{dt}\chi(t) + k^2\chi(t) = 0 \tag{3}$$

that is a homogeneous, linear differential equation of second order in t, we obtain solutions of the form

$$\chi(t) = \hat{\chi}e^{i\gamma t}$$

with $\gamma \in \mathbb{C}$. Inserting this approach into Eq. (3) yields

$$\begin{aligned} 0 &= -\frac{\gamma^2}{c^2}\hat{\chi}e^{i\gamma t} + \frac{2ib\gamma}{c^2}\hat{\chi}e^{i\gamma t} + k^2\hat{\chi}e^{i\gamma t} \\ 0 &= -\frac{\gamma^2}{c^2} + \frac{2ib\gamma}{c^2} + k^2 \\ \Leftrightarrow \gamma_{1,2} &= ib \pm \sqrt{k^2c^2 - b^2}. \end{aligned}$$

The second part of Eq. (2) is an eigenvalue problem which gives us spatial eigenfunctions $\phi_j(\vec{x})$ to corresponding eigenvalues $\lambda_j = -k_j^2$:

$$\Delta\phi_j(\vec{x}) = \lambda_j\phi_j(\vec{x})$$

These are individual to the specific problem and depend on the geometry.

Altogether, solutions of the original differential equation have the following form:

$$\psi(\vec{x},t) = \hat{\chi}e^{(-b\pm i\sqrt{k^2c^2-b^2})t}\phi_j(\vec{x}) = \hat{\chi}e^{-bt}e^{\pm i\sqrt{k^2c^2-b^2}t}\phi_j(\vec{x})$$

and linear combinations of those. We can identify an exponential dampening e^{-bt} that depends on the dampening coefficient b and an oscillation $e^{\pm i\sqrt{k^2c^2-b^2}t}$ at the rate

$$\omega = \pm\sqrt{k^2c^2 - b^2}. \tag{4}$$

For this purpose, we are only interested in positive frequencies.

As you can see, the frequencies ω_j depend on the eigenvalues $-k_j^2$ and these usually depend on the geometry of the system. We can name a lowest resonance frequency ω_1 which corresponds to the lowest eigenvalue $\lambda_1 = -k_1^2$. For a plugin, it makes sense to be able to scale the frequencies so that ω_1 matches a desired frequency ω_1^* which means that i.e. the size or other parameters of the system and thus the eigenvalues need to be changed accordingly.

The given form for ω is a bit unpractical though for usage in a plugin that provides knobs for scaling the resonance frequencies and the dampening because the dampening coefficient b actually lowers the oscillation frequency ω making it dependent on two parameters. The user of the plugin might first want to first adjust the lowest resonance frequency ω_1 and then change the dampening factor, but this will again modify the frequency $\omega = \sqrt{k^2c^2 - b^2}$. It is better to provide a knob for ω altogether, in a way that the actual frequency is independent of the dampening. We would like to get a representation like this

$$\psi(\vec{x},t) = \hat{\chi}e^{c(b+i\omega)t}\phi_j(\vec{x})$$

with independent b and ω. Solving the equation $\omega = \sqrt{k^2c^2 - b^2}$ for k yields

$$k^2 = \frac{\omega^2 + b^2}{c^2}.$$

This tells us how to adjust the eigenvalues – and with it properties of the system like the size – to get the desired (lowest) output frequency. I.e. in the case of a one-dimensional string with eigenvalues $k^2 = (n\pi/l)^2$ where l is the length of the string, this length needs to be set to $l = \pi c/\sqrt{\omega^2 + b^2}$.

3.2 Example 1 – Vibrating string

For one dimension the Laplace operator Δ is just twice the differential operator in x direction. Thus, the eigenvalue problem looks like this:

$$\frac{\partial^2}{\partial x^2}\phi(x) = \lambda\phi(x).$$

This differential equation is solved for example by sinusoidal functions

$$\begin{aligned}\phi(x) &= A\sin(kx)\\ 0 &= A\sin(kx) - \lambda A\sin(kx)\\ 0 &= A(k^2+\lambda)\sin(kx)\\ \Rightarrow \phi(x) &= A\sin(kx) = A\sin(\sqrt{-\lambda}x).\end{aligned}$$

Let us assume a string with length l is attached at the beginning and at the end. This states a boundary condition (the deflection needs to be zero at $x = 0$ and $x = l$):

$$\begin{aligned}0 &= \phi(0) = \phi(l)\\ 0 &= \sin(k\cdot 0) = \sin(k\cdot l)\end{aligned}$$

Therefore, the argument $k \cdot l$ must be a multiple of π which is the first zero crossing of the function $\sin x$ for $x > 0$. All in all, we get eigenvalues

$$-\lambda_n = k^2 = (n\pi/l)^2, \quad n \in \mathbb{N}.$$

We could also use an exponential approach:

$$\begin{aligned}\phi(x) &= Ae^{ikx}\\ 0 &= -Ak^2e^{ikx} - \lambda Ae^{ikx}\\ 0 &= Ae^{ikx}(k^2+\lambda)\\ \Rightarrow \phi(x) &= Ae^{ikx} = Ae^{i\sqrt{-\lambda}x}\end{aligned}$$

In conclusion, the complete solutions are

$$\begin{aligned}\psi(\vec{x},t) &= \hat{\chi}e^{-bt\pm i\sqrt{k^2c^2-b^2}t}\phi_i(\vec{x})\\ \psi(\vec{x},t) &= \hat{\chi}e^{-bt}e^{\pm i\sqrt{\left(\frac{n\pi}{l}\right)^2-b^2}\,t}e^{i\frac{n\pi}{l}x}.\end{aligned}$$

3.3 Example 2 – d-dimensional cube

First, we search for eigenfunctions $\phi(\vec{x})$ of the cube of length L with boundary condition

$$\phi(\vec{x}) = 0 \tag{5}$$

if at least one component x_i of $\vec{x}$ equals zero or L.

The problem can again be solved by separation of variables:

$$\begin{aligned}\phi(\vec{x}) &= \phi_1(x_1)\cdot\ldots\cdot\phi_d(x_d)\\ \Delta\phi(\vec{x}) &= \partial^2_{x_1}\phi(\vec{x}) + \cdots + \partial^2_{x_d}\phi(\vec{x}),\end{aligned}$$

so

$$\begin{aligned}-k^2\phi(\vec{x}) &= \Delta\phi(\vec{x})\\ -k^2\prod_{i=1}^{d}\phi_i(x_i) &= \partial^2_{x_1}\phi(\vec{x}) + \cdots + \partial^2_{x_d}\phi(\vec{x})\\ -k^2\prod_{i=1}^{d}\phi_i(x_i) &= \phi_2(x_2)\ldots\phi_d(x_d)\partial^2_{d_1}\phi_1(x_1) + \cdots\\ &\qquad + \phi_1(x_1)\ldots\phi_{d-1}(x_{d-1})\partial^2_{x_d}\phi_d(x_d)\end{aligned}$$

which rewrites to

$$-k^2 = \underbrace{\frac{\partial_{x_1}^2 \phi_1(x_1)}{\phi_1(x_1)}}_{E_1} + \cdots + \underbrace{\frac{\partial_{x_d}^2 \phi_d(x_d)}{\phi_d(x_d)}}_{E_d}. \tag{6}$$

This equation needs to be independently fulfilled for each variable x_i and therefore each summand on the right hand side is constant. These constants shall be named E_1 through E_d, following conventions in quantum mechanics where this quantity has the unit of an energy.

For each spatial direction $\vec{e}_i$ the resonance condition for standing waves is

$$L = n_i \frac{\lambda_i}{2}, \quad n_i \in \mathbb{N}$$

meaning, that the length L can be filled with multiples of half the wavelength $\lambda_i = 2\pi/k_i$. This condition naturally fulfills the given boundary condition (5). Here, we have identified the *quantum numbers* $n_1, ..., n_d$ (if we shall call them this way), which span the wave vector

$$\vec{k} = \begin{pmatrix} k_1 \\ \vdots \\ k_d \end{pmatrix} = \frac{\pi}{L} \begin{pmatrix} n_1 \\ \vdots \\ n_d \end{pmatrix}.$$

In each direction $\vec{e}_i$ the functions $\phi_i(x_i)$ are sine waves

$$\phi_i(x_i) = \sin\left(\frac{\pi}{L} n_i x_i\right)$$

with boundary condition $\phi_i(0) = \phi_i(L) = 0$ fulfilled. This gives us the eigenfunctions

$$\phi_{n_1,...,n_d}(\vec{x}) = \prod_{i=1}^{d} \sin\left(\frac{\pi}{L} n_i x_i\right).$$

The corresponding eigenvalues $-k_j^2$ described by the eigenvalue equation

$$\Delta \phi_{n_1,...,n_d}(\vec{x}) = -k_j^2 \phi_{n_1,...,n_d}(\vec{x})$$

are the sum of eigenvalues of the component functions (see Eq. (6)). In total:

$$k_{n_1,...,n_d} = \frac{\pi}{L} \sqrt{n_1^2 + \cdots + n_d^2} \tag{7}$$

The lowest eigenvalue is always

$$k_{1,...,1} = \frac{\pi}{L} \sqrt{\underbrace{1^2 + \cdots + 1^2}_{d \text{ times}}} = \frac{\pi}{L} \sqrt{d}.$$

With the equation

$$k^2 = \frac{\omega^2 + b^2}{c^2}$$

from above we thus find that the length L of the cube needs to be set to

$$L = \pi c \sqrt{\frac{d}{\omega^2 + b^2}}$$

for a given effective lowest frequency ω and dampening factor b.

Developed by Überton ©

uberton.org

mail@uberton.org