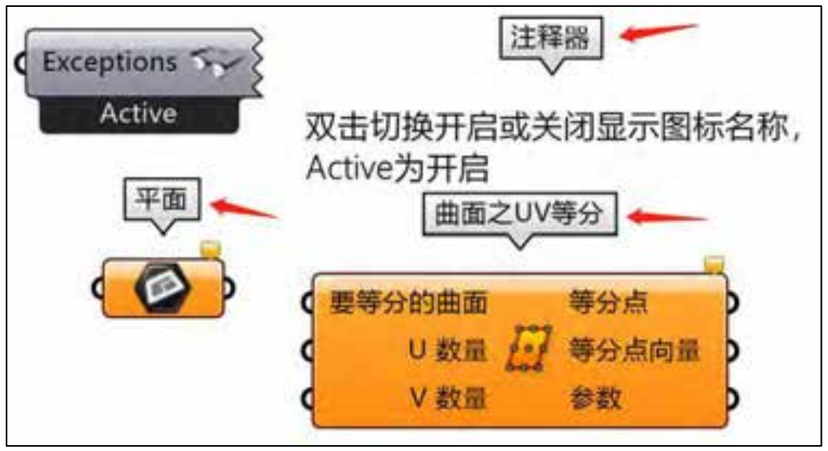
Bifocals 远近两用眼镜/双光眼镜：

在入门学习前，强烈推荐大家先安装下面这个运算器

Bifocals 远近两用眼镜/双光眼镜：非原生 GH 运算器，用于在 GH 工作区中显示运算器的名称，可以一目了然知道运算器的作用，毕竟初学记不住全部的运算器图标。

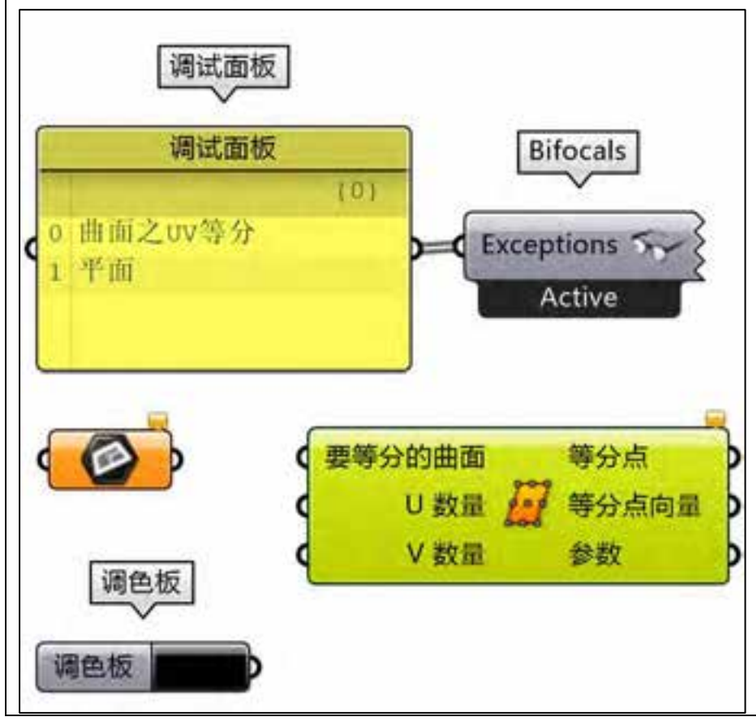
安装好后需要拖到 GH 工作窗口就可以了，双击 Active 处切换图标名称的显示或关闭。

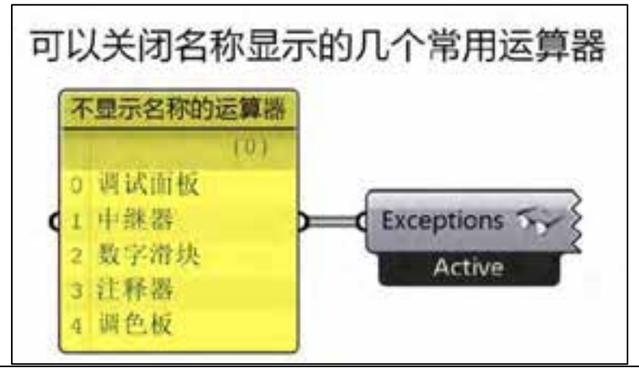
（这是犀流堂上的下载地址 http://www.rhinostudio.cn/course/706）下载插件后直接拖到 GH 工作区就可以了，会自动放在 Params 分类下。



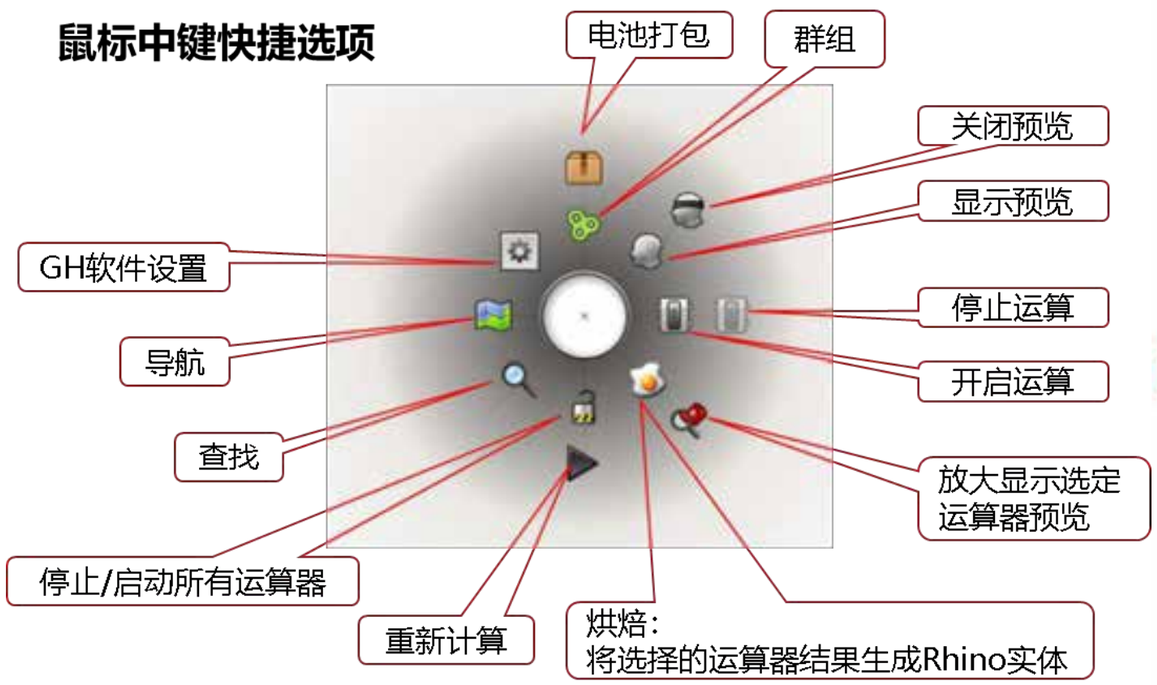
这两个都可以显示运算器名称

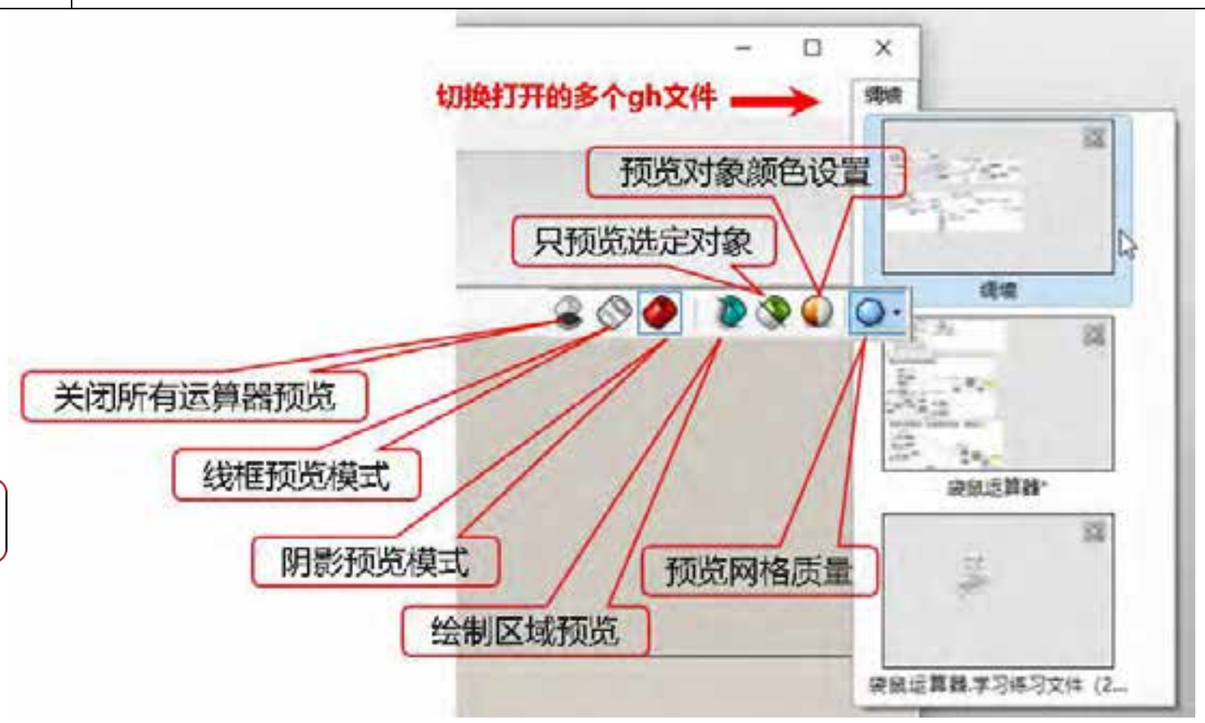
可以通过调试面板输入运算器名称，关闭不需要显示名称的运算器。右键两个英文菜单可以自己试一下，一般不需要改动。





常用的一些操作：	
1、按住鼠标右键可以在工作窗口中移动。按住 Alt 键后，在窗口空白处按住左键拖动，可以扩大指定位置的工作空间。	2、Ctrl+G 将选择的运算器组成群组，方便电池群组移动和查看。如果需要添加新的电池到群组，则选中要添加的电池，在指定群组空白处右键点击添加到组，或者直接全选要成组的运算器，再点击添加到组。右键还可以设置群组的背景颜色等。
3、按住 Shift 键可以增加多个电池输入端的连线，有些数据要注意连接的先后顺序；按住 Ctrl 键则是减少电池的连接线，也可以右键菜单里断开指定的连接线。.	4、同时按住 Ctrl+Shift 键，可以同时拖动连接在一个点的一条或多条电池线。双击电池的连接线会自动插入一个中继器。电池复制 Ctrl+C、粘贴 Ctrl+V。按住 Alt 键后拖动电池，也可以复制电池。
5、同时按住 Ctrl+Alt 键，然后点击工作区的运算器。这个时候 GH 就会自动显示出运算器在工具列中的位置。	6、同时按住 Ctrl+Alt 键，然后点击工作区的运算器。这个时候 GH 就会自动显示出运算器在工具列中的位置。
7、鼠标箭头放在运算器的输入端和输出端文字上，会自动显示相关注释，及输出端数据结构。	





Grasshopper 快速从目录中找到对应运算器须知：

1、 初始（Primitive）：创建曲线、曲面和多重曲面、网格，或者导入数据。

2、 分析（Analysis）：分析几何体的点、线、面的各种参数。

3、 公用（Util）：对几何体上的点、线、面进行处理，或是对数字进行计算、及其他数据（变形数据）的处理。

4、 创建点、向量、平面、网格线（三角形网格、矩形网格等）、磁场线是在向量分类中。

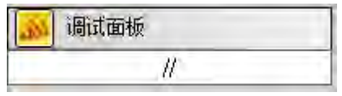
5、 树形数据及结构的处理在集合分类中。

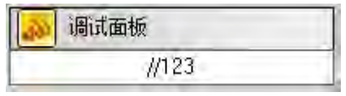
Grasshopper 主要菜单说明

双击工作区搜索电池的快捷命令

单点　　**句号**　　**加减乘除**　　**大于、小于、等于**

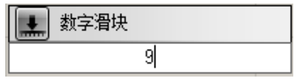
调试面板：双// （可直接输入数据文字）





注释器：～符号（shift+~键）

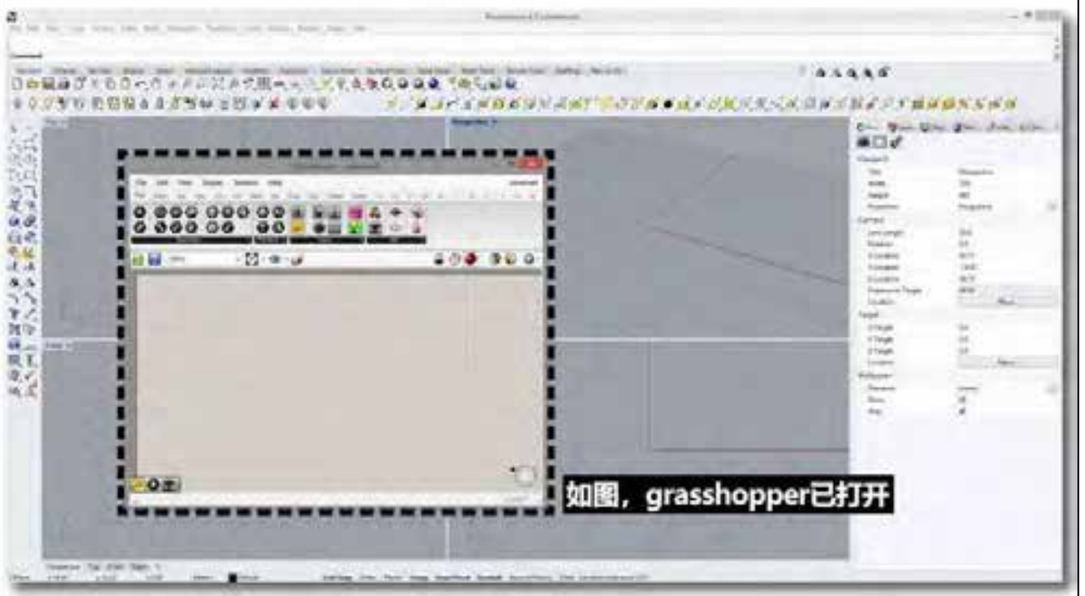
数字滑块：任意数字输出对应结果



如果输入“-10<10”则自动插入取值范围在-10—10 的数字滑块。

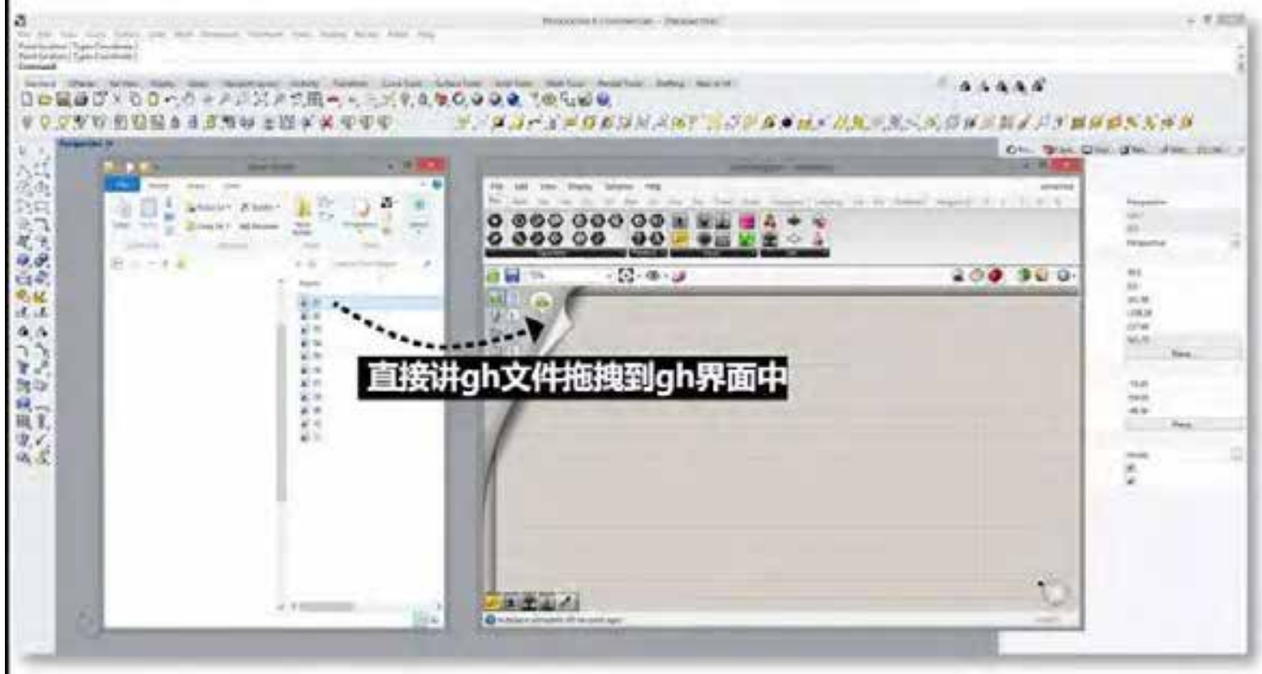
如果在调节过程需要取小数，则一开始就要带上小数位数，如输入 10.00，就可以在 0-10 之间取带小数点后两位的数字。

打开grasshopper

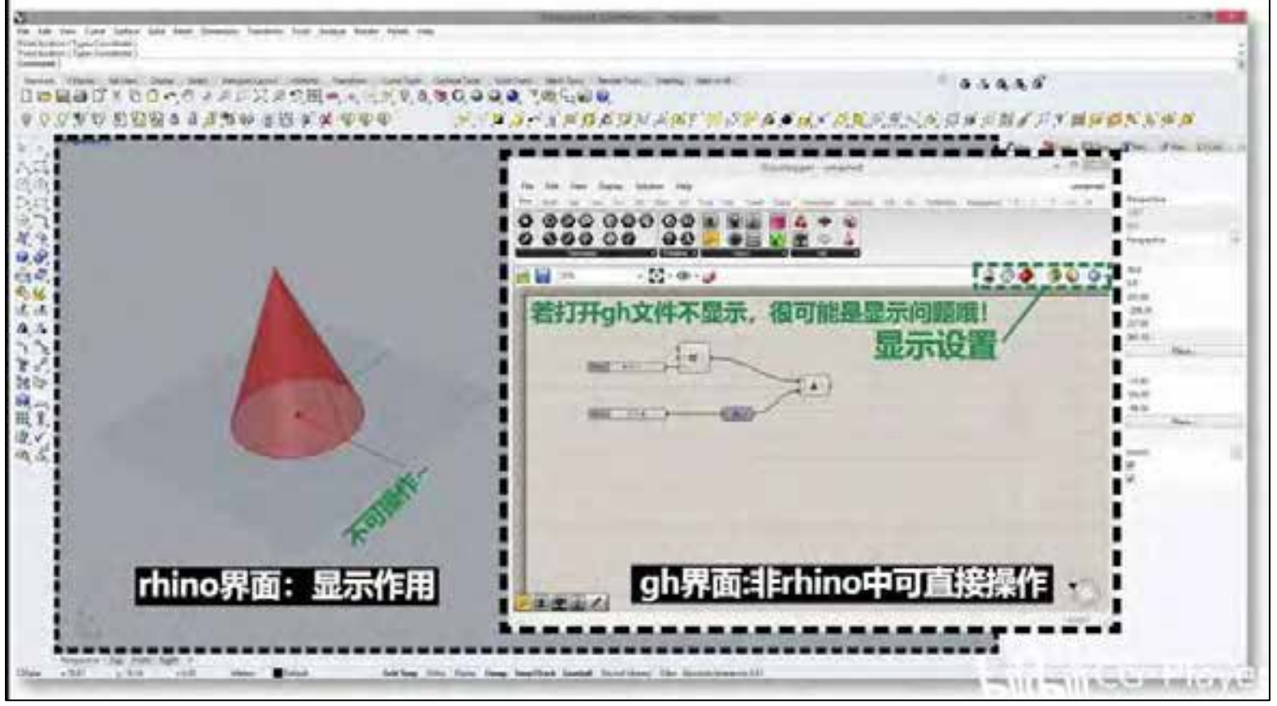


grasshopper界面介绍

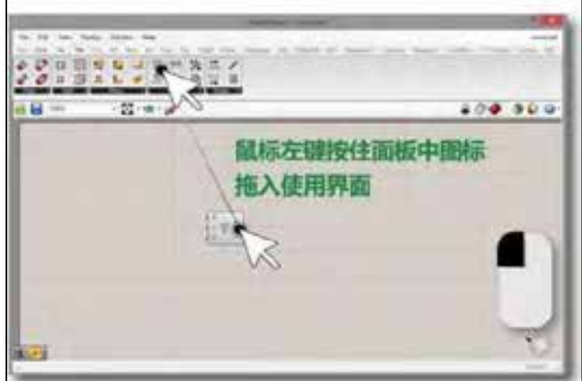
打开grasshopper文件

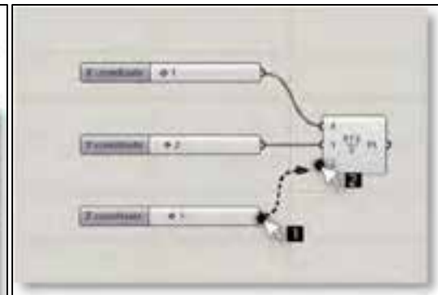


GH和rhino的关系



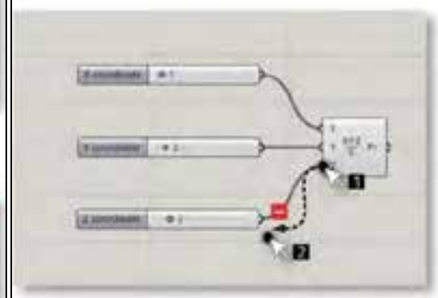
3种调用运算器的方式





连线

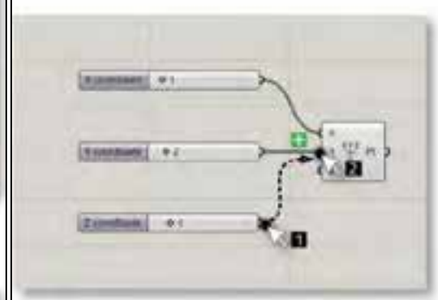
左键 拖动



取消连线

ctrl+左键 拖动

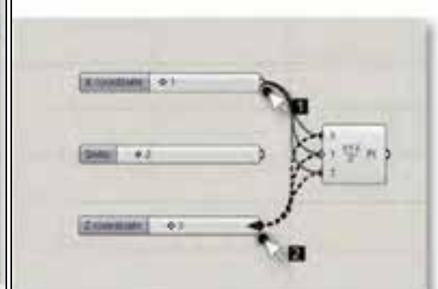
CTRL +



同时连输入端

shift+左键 拖动

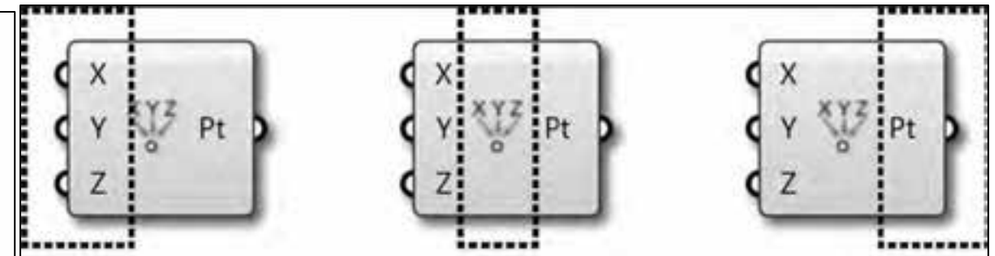
SHIFT +



同时连输出端

ctrl+shift+左键 拖动

CTRL + SHIFT +



输入端　　运算器名称/图标　　输出端

正常使用

运算器中含有正常数据
在rhino中会呈现红色(默认显示设置下)

隐藏

运算器中输出的物体
在rhino界面中不显示(隐藏)

不能使用

运算器不输入和输出任何数据
也不执行任何操作

选中

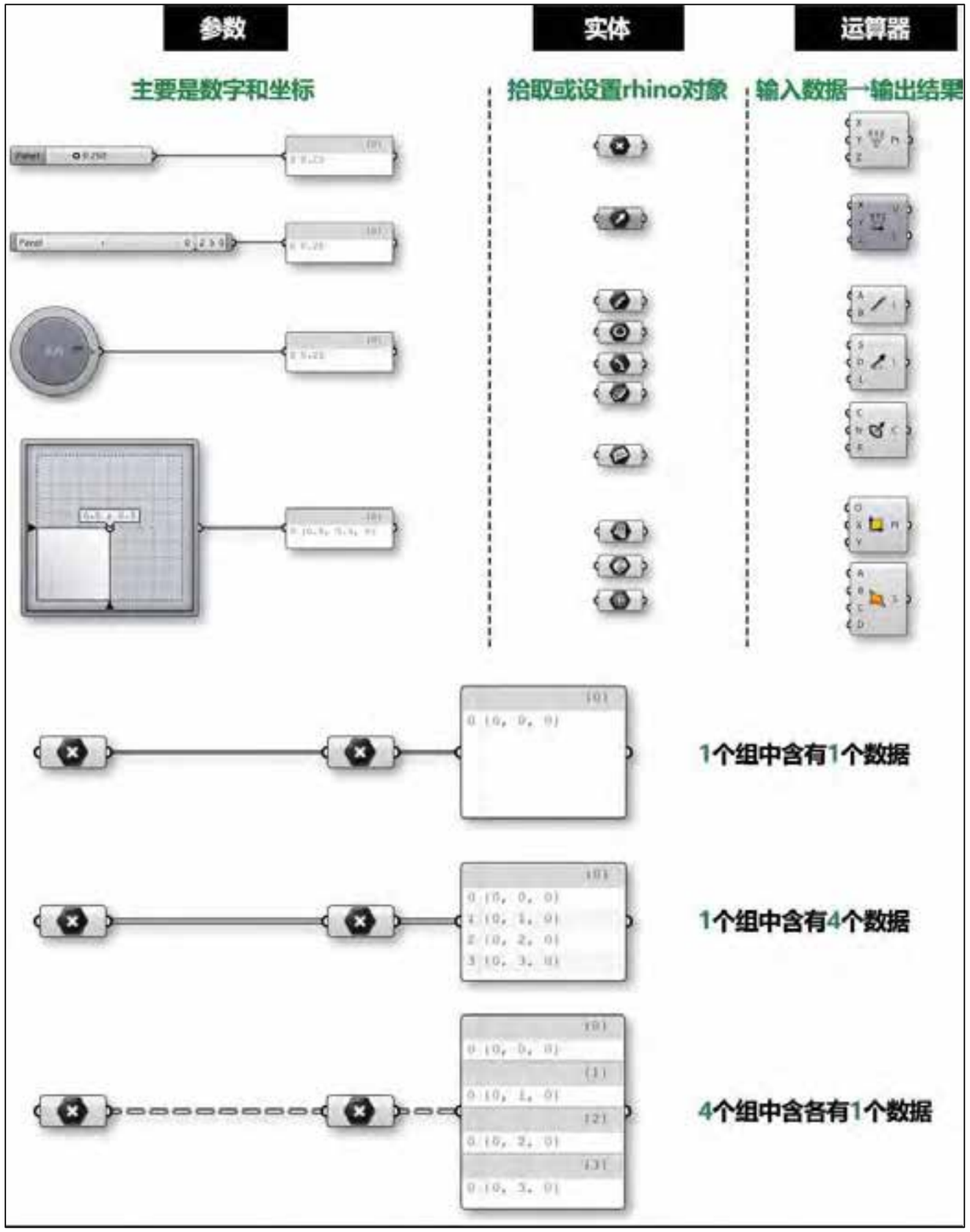
运算器不输入和输出任何数据
也不执行任何操作

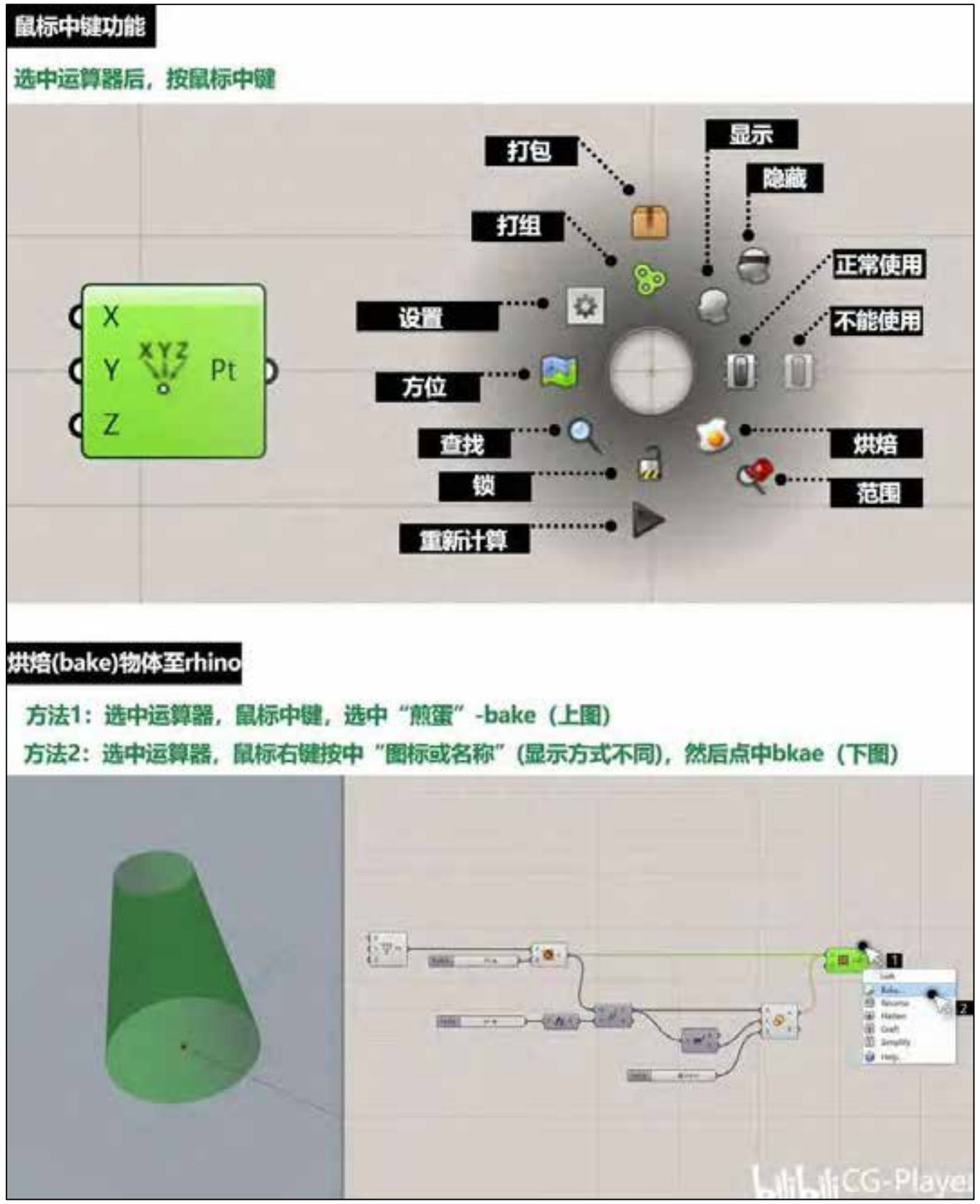
警告

运算器中不含有数据
需要设置数据才能够正常使用

错误

运算器中含有错误数据


参数
实体
运算器
主要是数字和坐标
拾取或设置rhino对象
输入数据→输出结果
1个组中含有1个数据
1个组中含有4个数据
4个组中含各有1个数据


鼠标中键功能
选中运算器后，按鼠标中键
打包
显示
隐藏
打组
正常使用
设置
不能使用
方位
查找
烘焙
锁
范围
重新计算
X
Y
Z
Pt
烘焙(bake)物体至rhino
方法1：选中运算器，鼠标中键，选中“煎蛋”-bake（上图）
方法2：选中运算器，鼠标右键按中“图标或名称”(显示方式不同)，然后点中bkae（下图）
bilibili CG-Player

网页中英双语显示翻译软件——彩云小译，电脑截图翻译软件——网易有道词典

另外推荐有需要上英文网站学习教程的朋友安装下面的浏览器翻译插件——**彩云小译**， 360 极速浏览器（基于谷歌 Chrome 内核）的应用商店就有可以以中英双语显示来翻译国外网站，可以设置自动翻译英文或日本网站，非常方便。这样有些中文翻译表达不准确的，还可以根据英语判断准确意思。

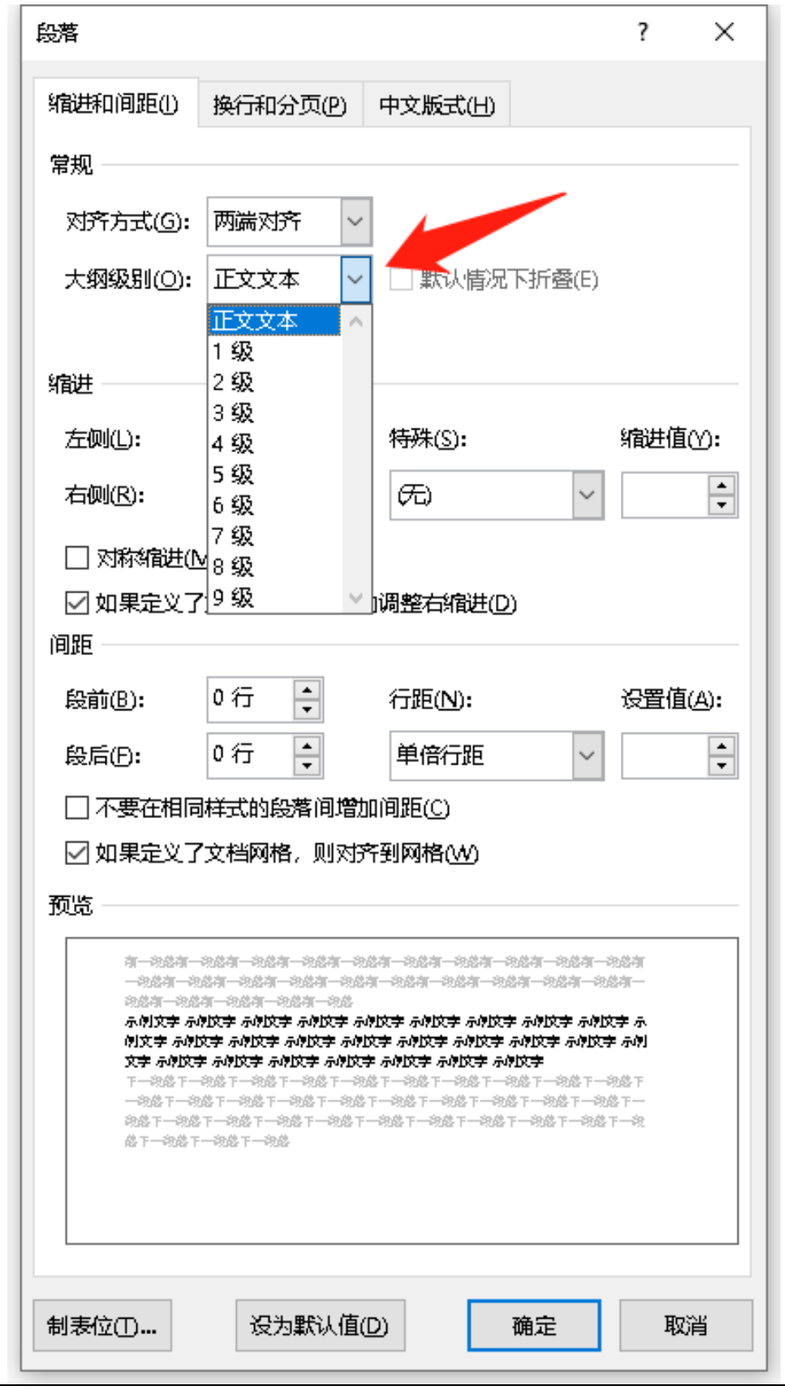
Edge 浏览器也有这个插件，但是用一段时间会要收费。Edge 另外还有万能翻译也能双语显示网页。

另外我用过比较好的电脑翻译软件有网易有道词典，可以截图自动翻译，遇上不认识的电池，截图翻译一下大概意思就懂了，推荐自行下载去广告版软件。

http://kangaroo3d.com/这是国外的袋鼠插件论坛

添加目录标签的方法:

在非表格和非文本框的位置输入文字（或者选择已有的文字），选择文字后点击“段落”，在段落设置窗口的大纲级别，选择要设置的目录级别，即可自动生成目录标签。



Rhino 和 Grasshopper 的插件安装方式详解

转自：犀牛坊

在三维建模软件中，Rhino 作为 Nurbs 曲面建模的杰出代表，越来越多的设计人员都开始使用，这次我们就 Rhino 和 Grasshopper 的插件安装方式做一个详细的解释。

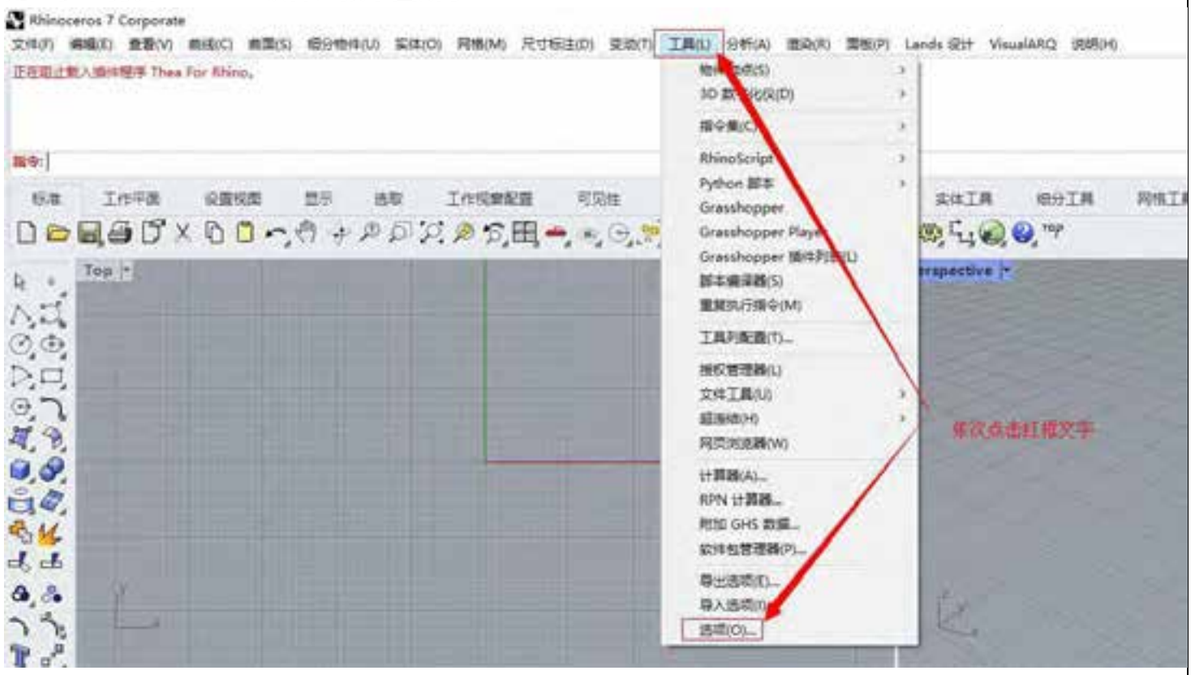
先说 Rhino 插件：

Rhino 的插件扩展名一般 为 *.rhp ，常用的安装方式分为 2 种：

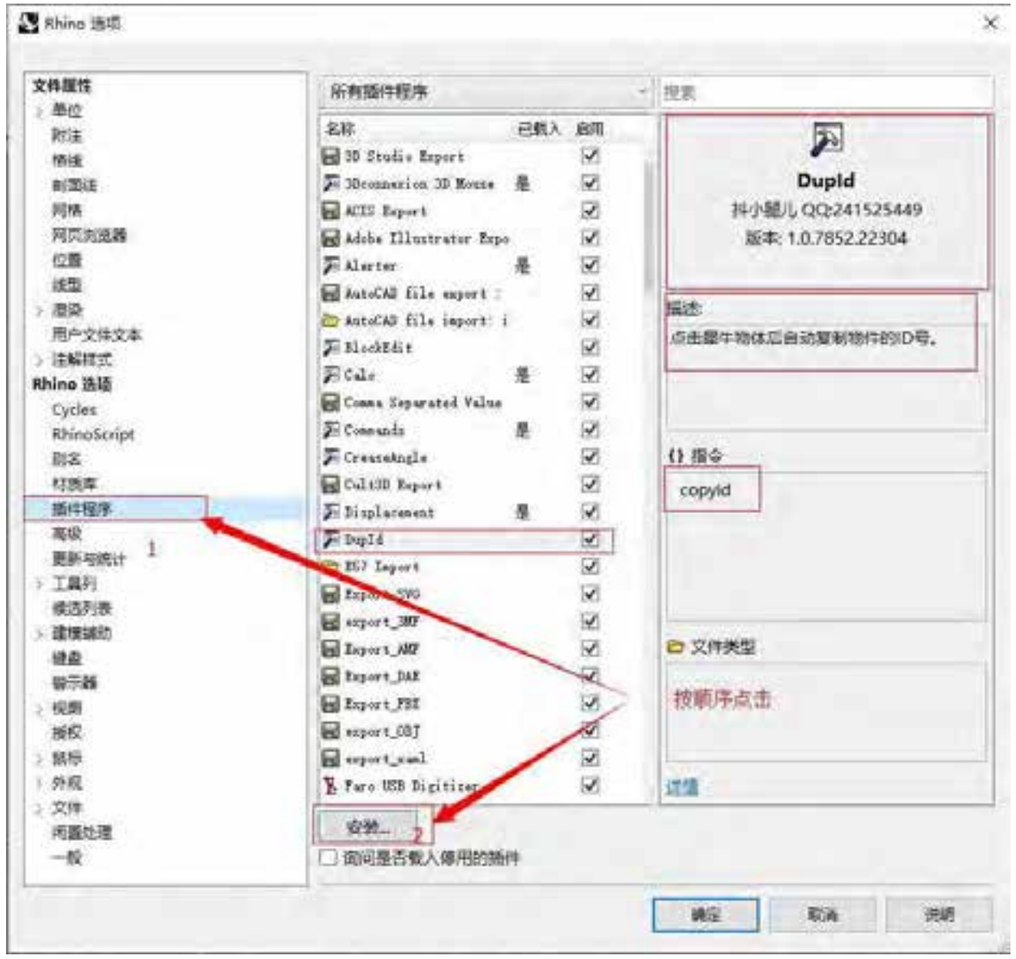
第一种是开发者直接做好了可执行文件安装包，直接一步步的安装即可。

第二种是插件只有一个 rhp 文件，这时就需要我们自己手动加载安装，我们以 Rhino 7.X 版本为例，具体安装方式如下：

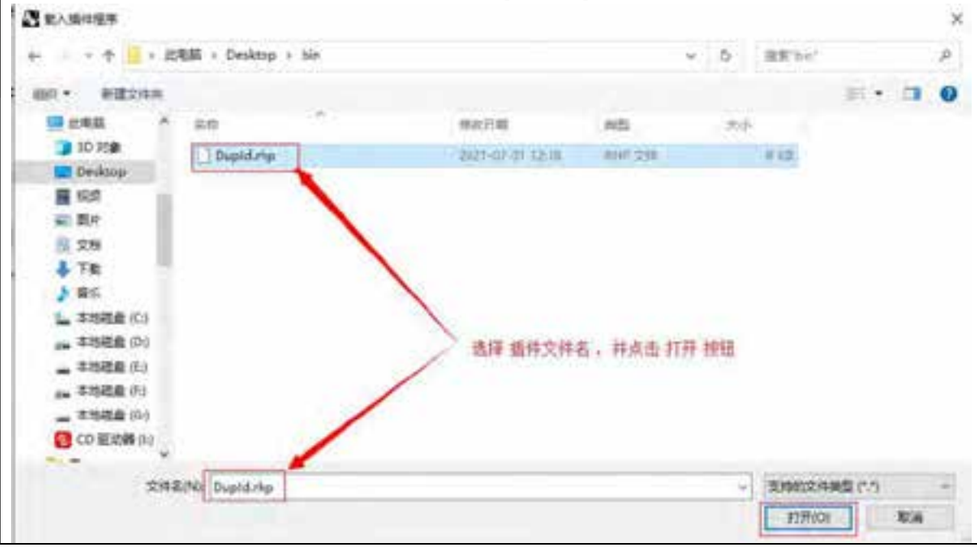
第一步：打开 Rhino 软件，按下图所示：



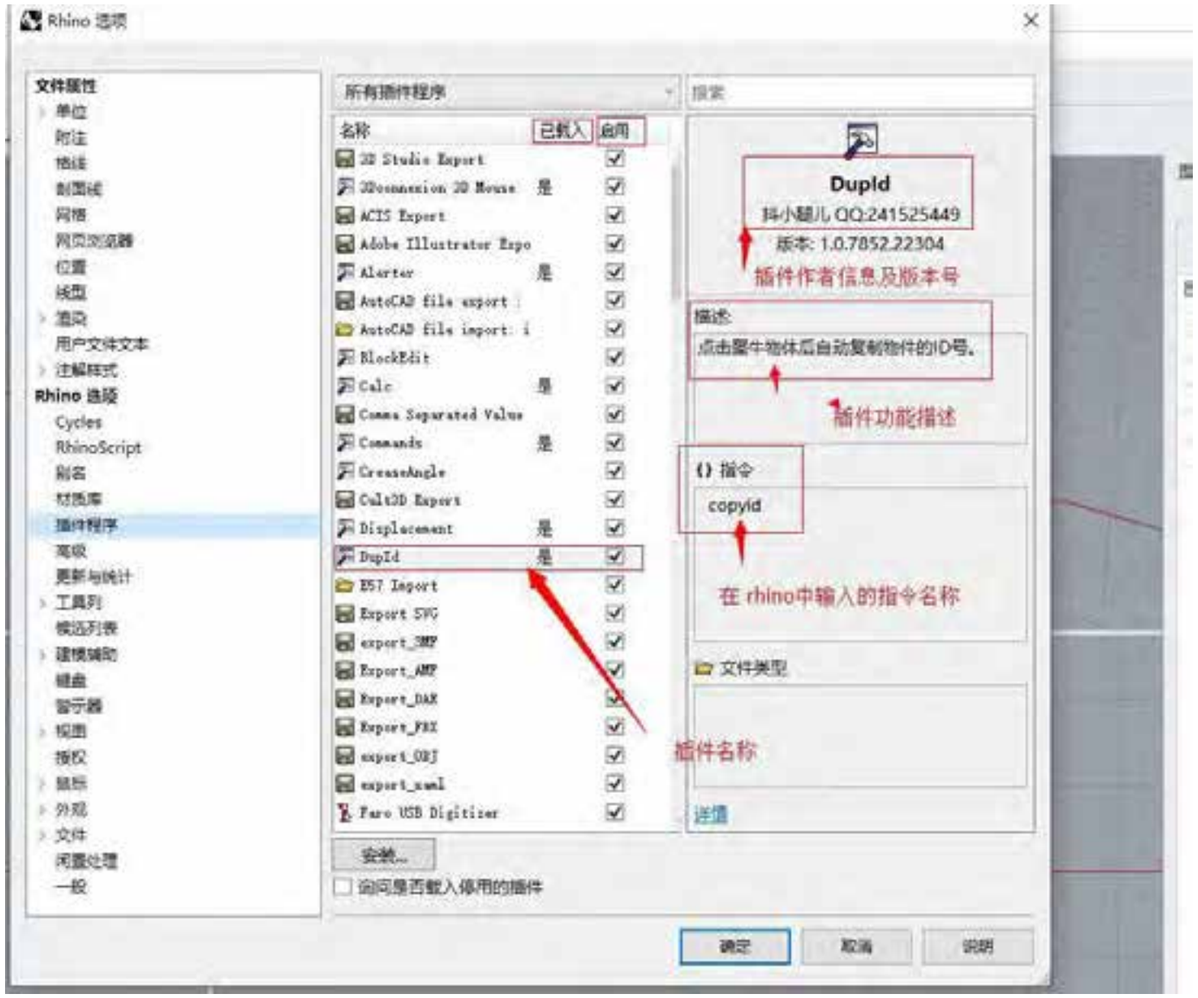
第二步：准备好我们下载的插件，并按下图所示：



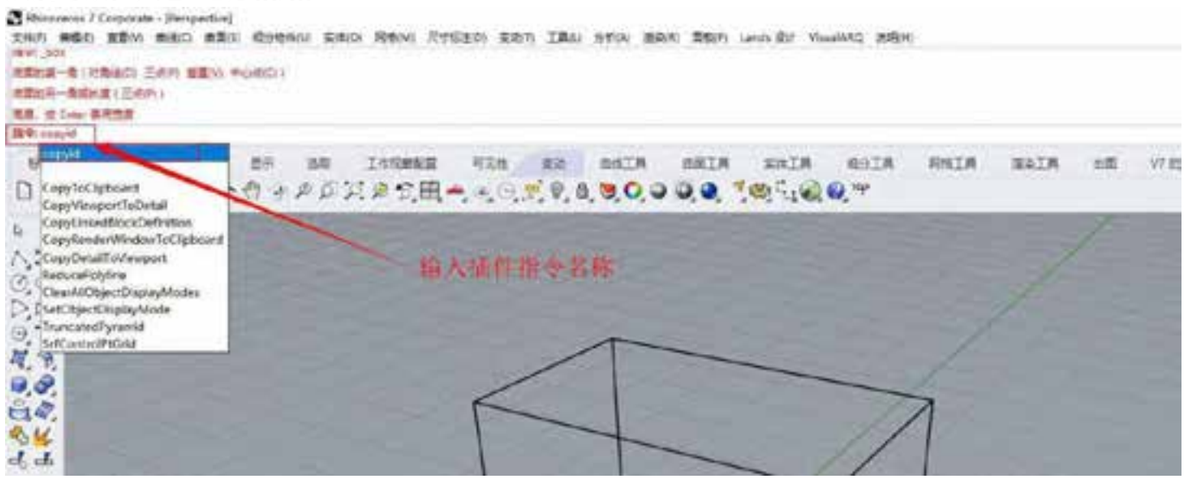
第三步：点击 安装 按钮以后，弹出 文件选择对话框，这时我们选择准备好的 rhp 文件，以小编开发的这个插件为例，插件文件名为 DupId.rhp ，按下图操作：



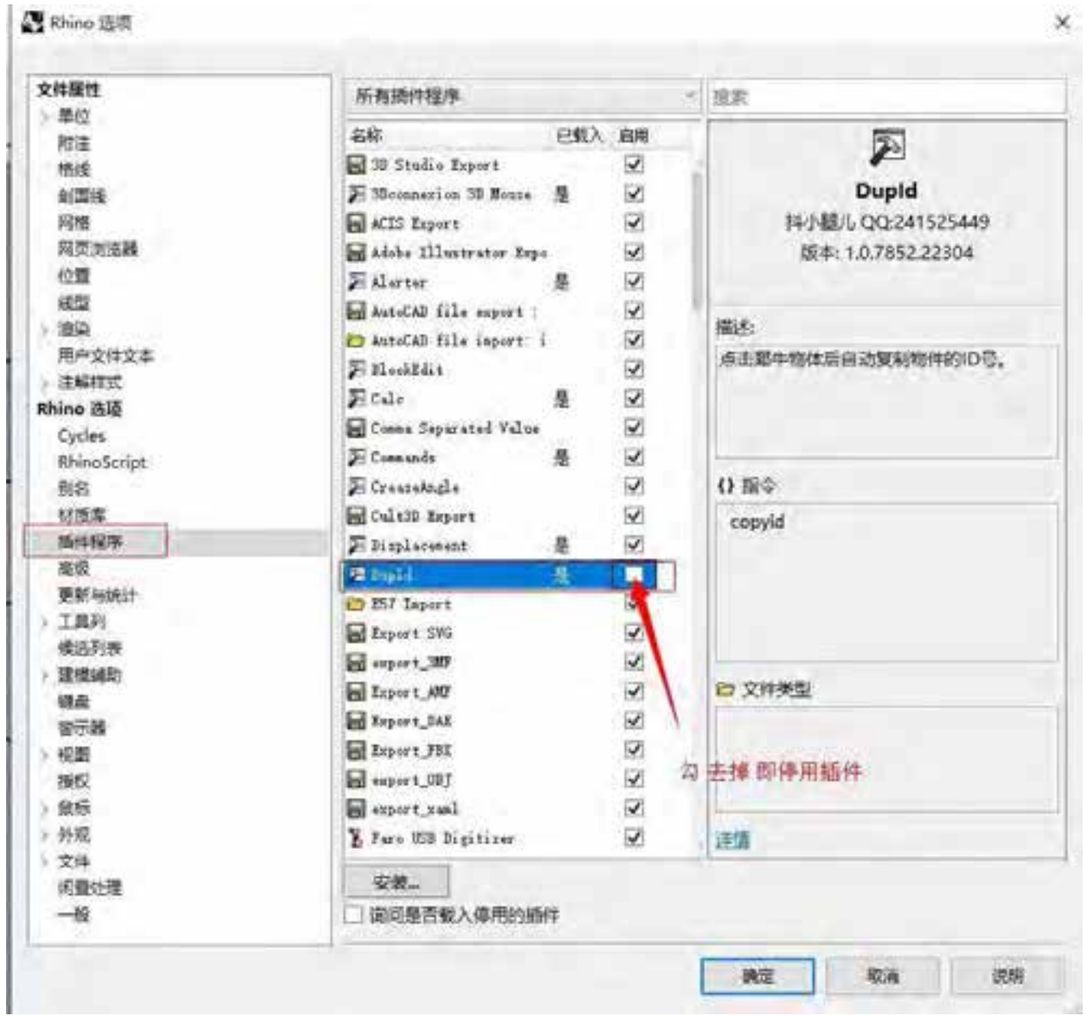
第四步：点击打开按钮即安装了插件，这时我们再查看下插件是否安装成功：



第五步：插件安装完成之后，我们再根据插件的提示信息，测试下插件能否正常工作，比如本插件的 犀牛指令为 copyid ，功能就是 点击了 犀牛物体后，自动复制 犀牛物体的 ID 号到 Windosws 的剪贴板，然后可以在其它 地方粘贴。



以上就是常用的 Rhino 插件的安装及调用方式，如果想停用插件呢？按下图操作：



再说 Grasshopper 插件：

Grasshopper 的插件扩展名比较多，常见的有 3 种：

第 1 种：图标为 绿色图标 ，扩展名为 *.gha ，图标样式：

第 2 种：图标为 橙色图标 ，扩展名为 *.ghuser ，图标样式：

第 3 种：无图标 ，扩展名为 *.ghpy 。

其中 扩展名为 *.gha 的是 C++、C#、VB 语言编写并编译的；*.ghuser 为各种语言编写的未编译插件，有些可以从其中看到编写源代码，或者是一种打包电池组；扩展名为 *.ghpy 的是 Python 语言编写并编译的独有 文件格式，看不到源代码。

这 3 种插件所放的磁盘位置并不相同：

*.gha 和 *.ghpy 格式的通用性比较强，可以放在 磁盘任意位置，如果插件为 单个文件时，安装时直接拖动插件文件到 Grasshopper 界面即安装完成，如果插件有多个文件的，则需要把插件整个文件夹复制到 GH 的插件库目录，插件库目录一般在 C 盘的 “C:\Users\电脑用户名\AppData\Roaming\Grasshopper\Libraries ” 下。而橙色的 *.ghuser 插件则必须安装在 “C:\USERS\电脑用户名\AppData\Roaming\Grasshopper\UserObjects" 下。

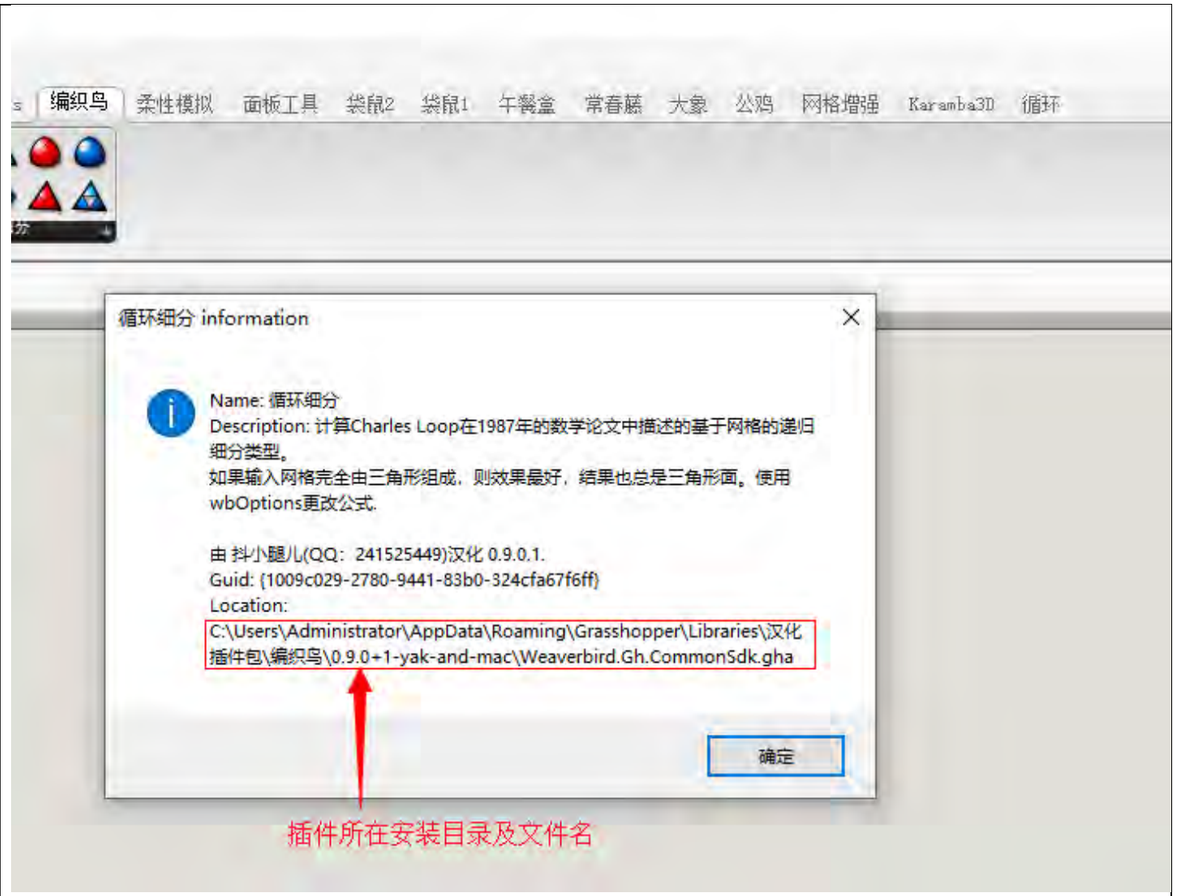
由于 Rhino 的版本比较多，有 4.X、5.X、6.X、7.X、8.X(WIP 版，即开发版)，各种插件所支持的 Rhino 版本也不尽相同，比如著名的插件 T-Splines 只支持到 Rhino 的 5.X 版本，所以 Rhino 开发者为管理各种插件的安装目录，在 "C:\Users\电脑用户名\AppData\Roaming\McNeel\Rhinoceros\packages" 目录，为各插件指定了安装目录：

如果一款插件支持多个 Rhino 版本，则在 每个目录中都含有该插件文件。

还有一些收费插件，既提供了 Rhino 插件，又提供了 GH 插件，属于定制化的插件，如：Lands Design、卡兰巴(Karamba3D)，通常会有自己的自定义安装目录，需要安装者自己分辨安装目录。

怎样查找插件安装目录?

很多人在安装插件时都是通过直接拖放的方式安装的，日子久了就忘记插件的安装路径了，特别是通过多种途径下载安装了插件后，还有版本新旧问题导致冲突，这时就可以通过打开 GH，在想找的插件任意一个电池点鼠标右键，可以查看该插件的安装位置，比如编织鸟插件，如下图所示：



通过这种查找方式，可以精准的删除或者和小编一样的整理插件。

以上就是各种插件的安装及删除方式，

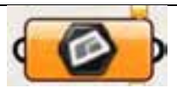
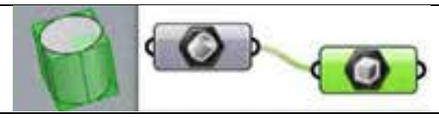
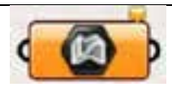
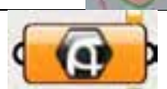
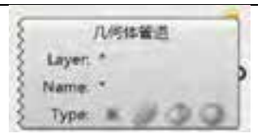
Grasshopper 中文版运算器详解 2022（中英文对照）

参数 Params

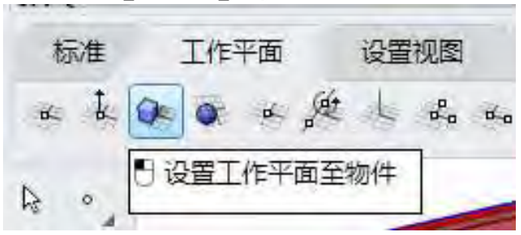
几何（Geometry）

参数(Params)

几何（Geometry）- （参数）

向量 Vector		包含方向及大小的矢量。可以直接拾取直线为向量，要注意直线的起始方向。（下方详解）	**点 Point**		点数据，可以右键点击拾取点，如果当前场景当中有点，就会让你去拾取，如果当前场景没有，就会让你去自行绘制。可以从其他数据中（向量，平面，颜色等）自动提取点数据（下方详解）
圆 Circle		只能通过手动绘制圆。如果是 Rhino 中已有的圆形，可以用曲线或几何体运算器拾取。可以把曲线接给 circle，看是否爆红来快速判断曲线是否是标准圆。			
圆弧 Arc		拾取手动绘制的一个或多个圆弧。或者用曲线拾取 Rhino 中的圆弧	**曲线 Curve**		曲线拾取，包含直线、多段线、多重曲线、圆、圆弧等。可以直接拾取曲面的边缘线
平面 Plane		拾取手动设置的一个或多个平面。平面是一个二维空间，没有边界。直接连接点的话，会自动生成以该点为原点的 XY 平面。			
直线 Line		拾取手动绘制的直线。Rhino 中已有的直线可以用曲线拾取	**矩形 Rectangle**		拾取手动绘制的一个或多个矩形。Rhino 已有的矩形同样可以用曲线拾取
盒子 Box		拾取手动绘制的一个或多个盒子。也可以直接接入任意物体（直线、曲面、多重曲面等），得到以物体所在的立方空间生成盒子。			
多重曲面 BRep		拾取一个或多个多重曲面，也可以拾取曲面。	**扭曲的盒子 Twisted box**		拾取手动绘制的一个或多个扭曲的盒子
曲面 Surface		拾取一个或多个曲面。可以接入封闭的平面曲线，会自动生成面。	**网格 Mesh**		拾取一个或多个网格
			网格面 Mesh Face		拾取一个或多个网格面
细分 SubD		拾取一个或多个细分物体。可以直接连接网格、曲面或者多重曲面生成细分物体。	**几何体管道 Geometry pipeline**		用来自动拾取 Rhino 对应图层中的物体。在 layer 输入对应图层名称后，运算器自动拾取图层中对应的物体， rhino 中物体发生改变，gh 中会即时发生改变。不过可拾取类型只有点，线，brep，网格，文字啊，注解之类的拾取不了，想拾取还得依靠插件。（下方详解）
几何体 Geometry		拾取一个或多个几何体。包含点、线、面、体。但是最好不要都用这个，避免数据多了很难区别数据类型。			
变形 Transform		包含三维变形数据的集合。可以将多个变形数据组合起来。可以参考变形工具组。	**几何体缓存 Geometry Cache**		将 GH 中的物体烘焙到 Rhino 中进行操作，再把操作后的结果拾取到 GH 中进行下一步处理（下方详解）
场 Field		包含场向量的集合。与电磁场相关的运算器	**群组 Group**		包含几何体群组的集合，这个 Gruop 不能拾取不能绘制任何物件。多个物体组成群组要用变形工具中的群组运算器。

关于平面：很多人对比 rhino 和 su 建体块或者方盒子的时候都会提到，su 可以直接在面上绘制曲线，rhino 不行，啊，su 好方便。没错，这一点 su 确实方便，但你要知道，市面上我还没见过其他软件可以直接在面上绘制，都是要先基于工作平面，所以这其实算是 su 的奇技淫巧。那 rhino 呢？也不麻烦，咋地，没这个操作，其他软件日子就不过了么?rhino 里很简单，一般我们可以直接设置工作平面到物件就行了，给命令'_CPlane _Object 设置一个快捷键，做起来也很快。

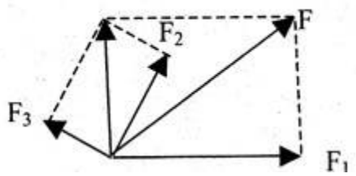


点 Point:

向量 Vector:

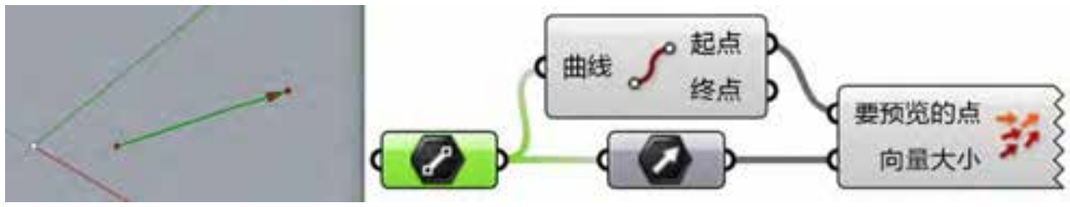
矢量（vector）是一种既有大小又有方向的量，又称为向量。

矢量可以加减，可以拆分，可以合并，可以通过平行四边形原则计算向量。

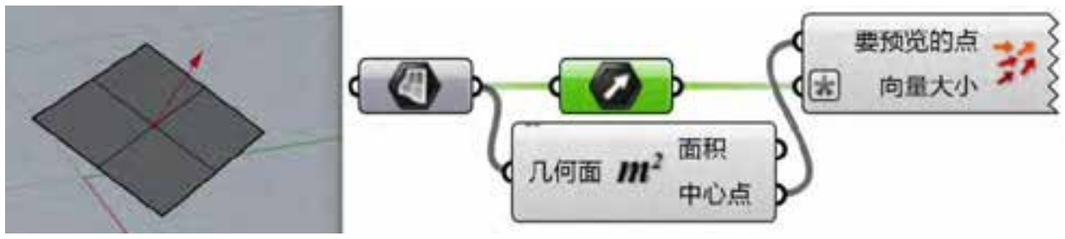


可以在 rhino 中指定一个点，连接点就能生成一个以指定点为起点，原点到指定点的方向，两点间距为长度的向量。

可以直接拾取直线转为向量。

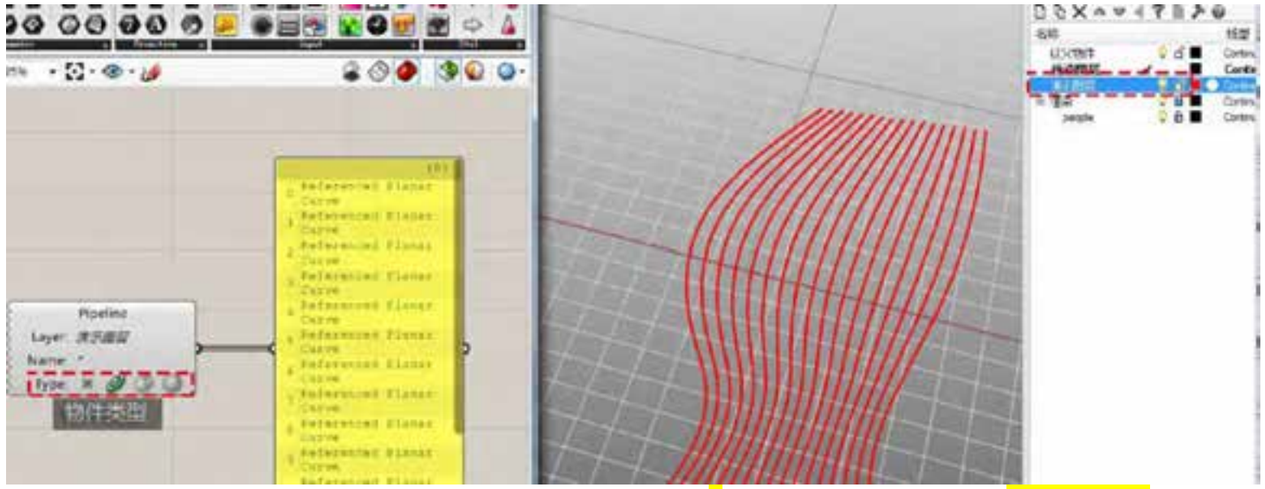


可以提取任意平面曲线或曲面的法线向量。

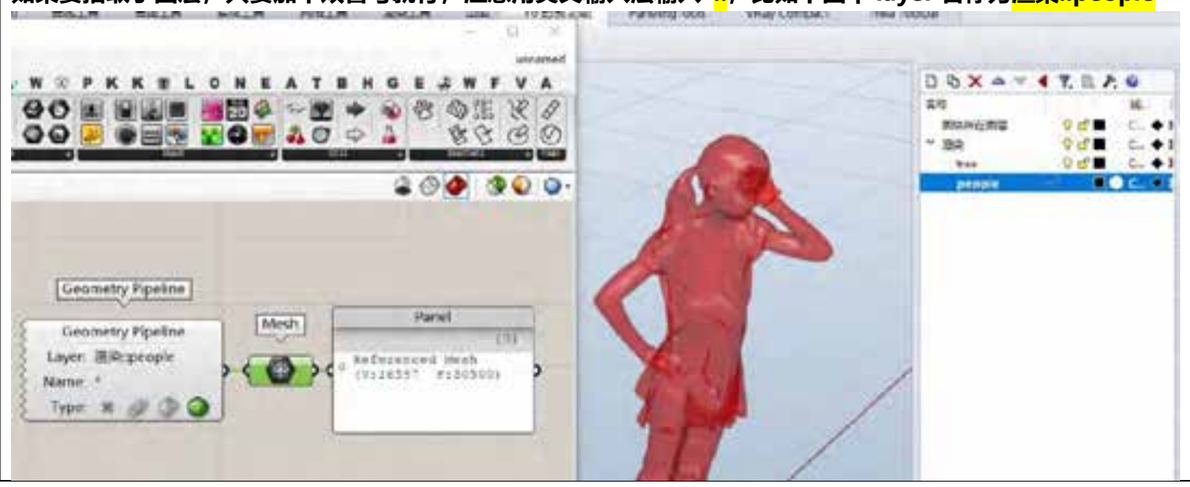


几何体管道 Geometry pipeline:

用来自动联动 rhino 和 GH，并非字面意思用来成管。可以根据输入的图层名字或者物件名字，以及物件属性来自动拾取 rhino 中对应图层中的物件。

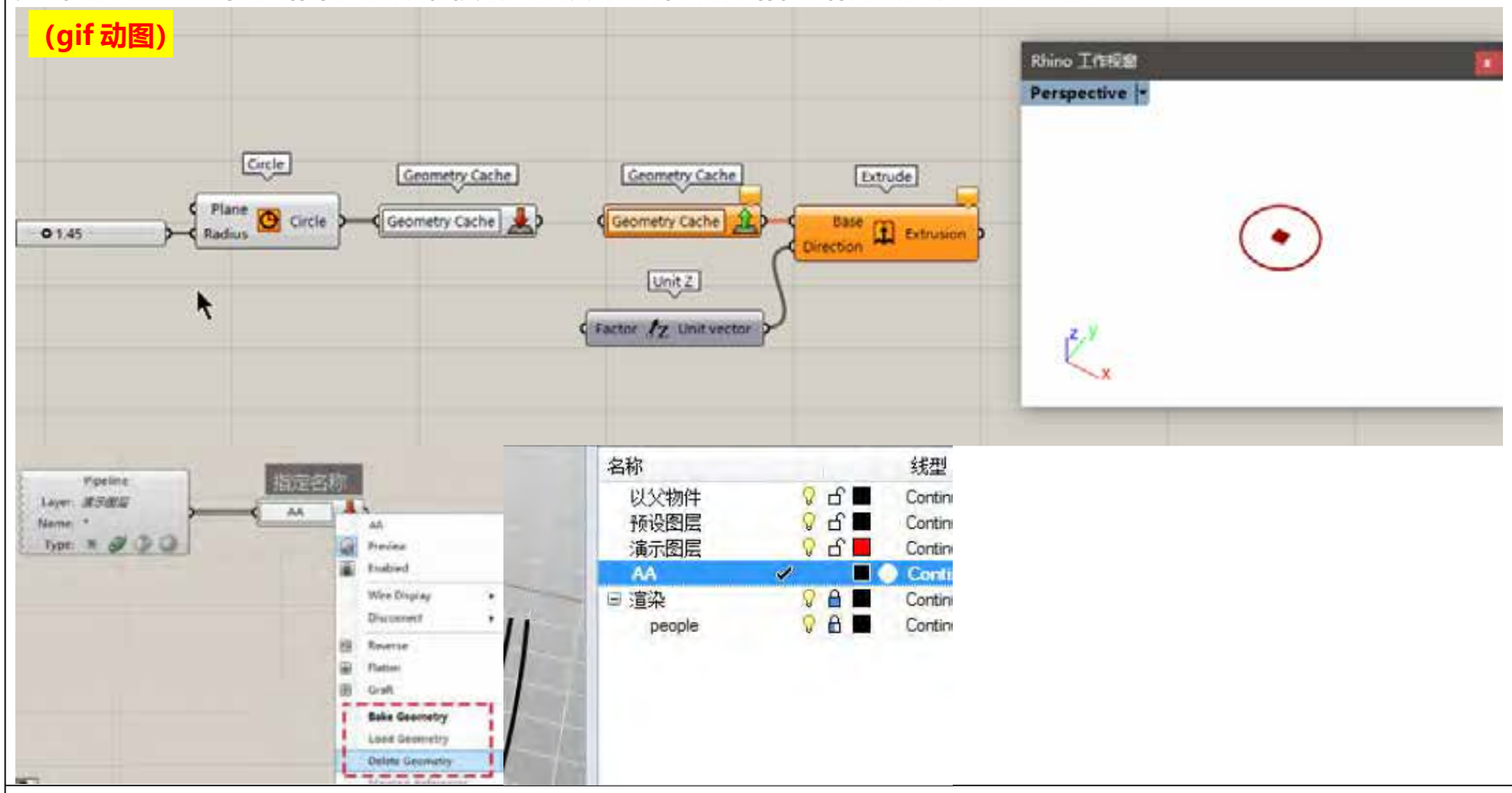


如果要拾取子图层，只要加个双冒号就行，注意用英文输入法输入 ::，比如下图中 layer 名称为渲染::people

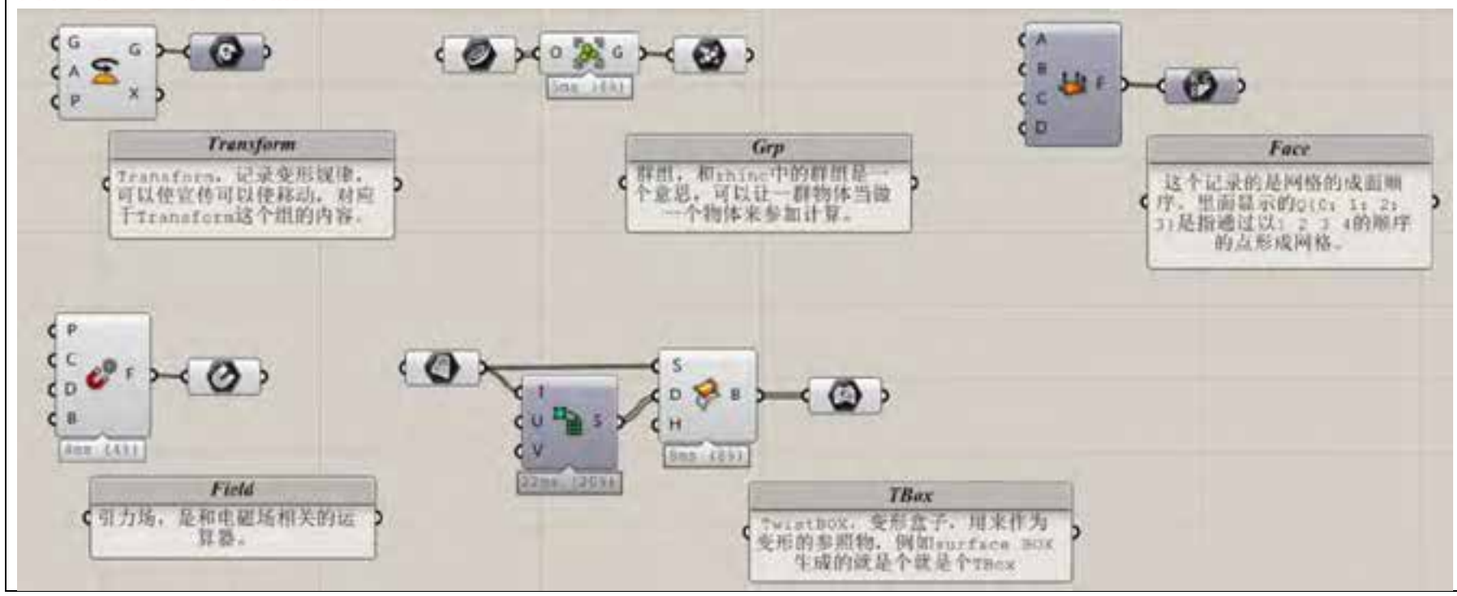


几何体缓存 Geometry Cache：

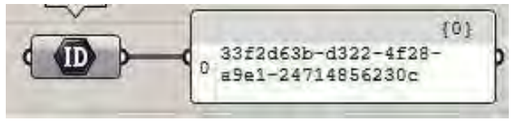
几何缓存简单说，可以把 gh 当中的物体先 bake（烘焙）到 rhino 当中，利用 rhino 手工对物体进行一定操作之后再重新拾取回 GH 当中继续深化处理。毕竟有的时候 rhino 当中操作要比 gh 直接操作方便很多。但是 在 rhino 中的操作不能是炸开，修剪分割之类的破坏物体的操作，这样是没办法拾取回 GH 的。



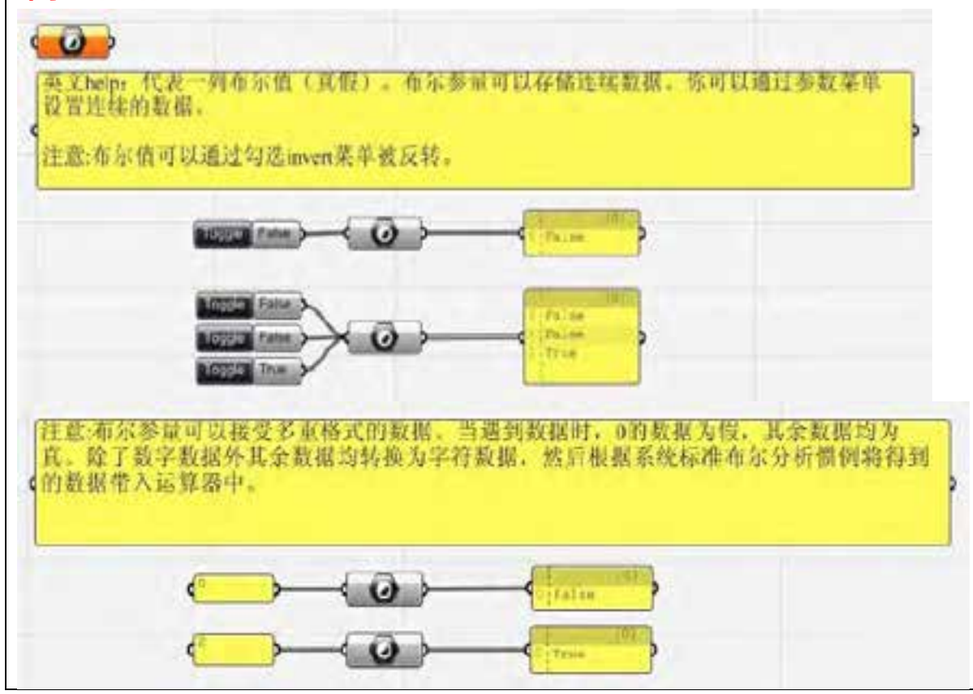
变形、群组、场、扭曲的盒子：



初始（Primitive）- （参数）

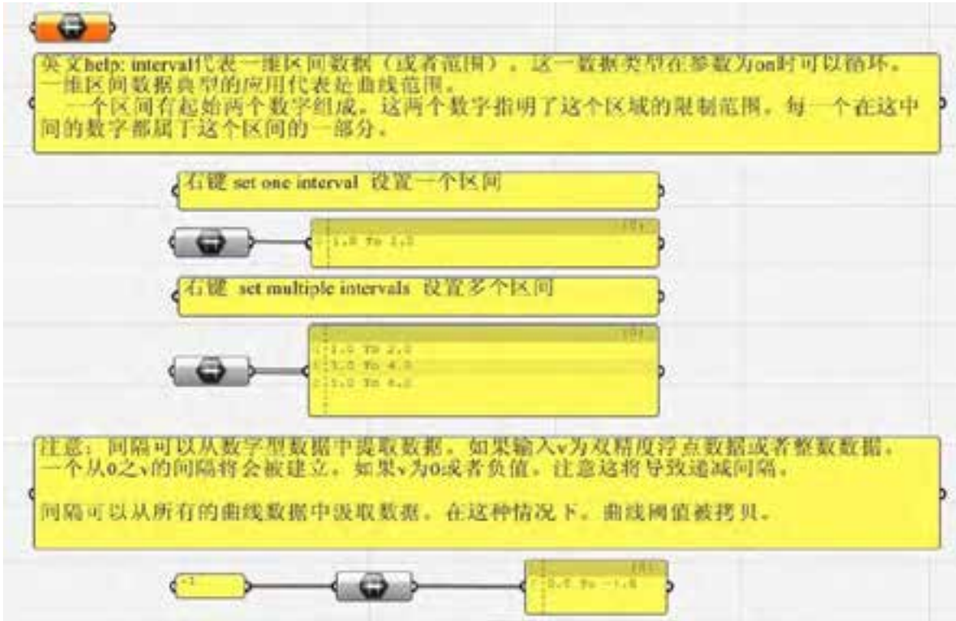
小数 Number		双精度浮点数据，简单理解就是小数	**布尔 Boolean**		True 和 False，对应 1 和 0。 所有非 0 的数，都会判定为 True（下方详解）
整数 Integer		提取整数，非整数会自动四舍五入取整	**文本 Text**		文本数据
一维区间 Domain		设置一个数值区间，即范围。右键设置一个或多个范围（下方详解）	**二维区间 Domain²**		二维区间，用来作为曲面坐标的定义。 右键设置一个或多个区间（下方详解）
复数 Complex		我们把形如 z=a+bi（a、b 均为实数）的数称为复数。其中，a 称为实部，b 称为虚部，i 称为虚数单位。当 z 的虚部 b＝0 时，则 z 为实数；当 z 的虚部 b≠0 时，实部 a＝0 时，常称 z 为纯虚数。	**文化 Culture**		这个运算器储存了全世界所有国家的名称，可以右键通过不同方式来选择。比如国家，语言，区域来选择不同的国家。（下方图解）
时间 Time		储存时间数据	**物件 ID Guids**		储存物件名称（ID）数据，可以右键运算器来选择任意物件获得其唯一物件 ID。 
矩阵 Matrix		由数字组成的方阵（下方详解）			
颜色 Color		包含 RGB 颜色的集合，右键可以设置颜色			
数据 Data		可以接受任何数据，反而不常用，会导致难以区分数据类型（下方详解）	**数据路径 Data Path**		储存数据结构的路径， 数据结构路径的格式为{A；B；C}
文件路径 Text		右键获取文件路径	**材质 Shader**		获取材质，右键可以直接选取 rhino 中默认存在的材质

布尔 Boolean:

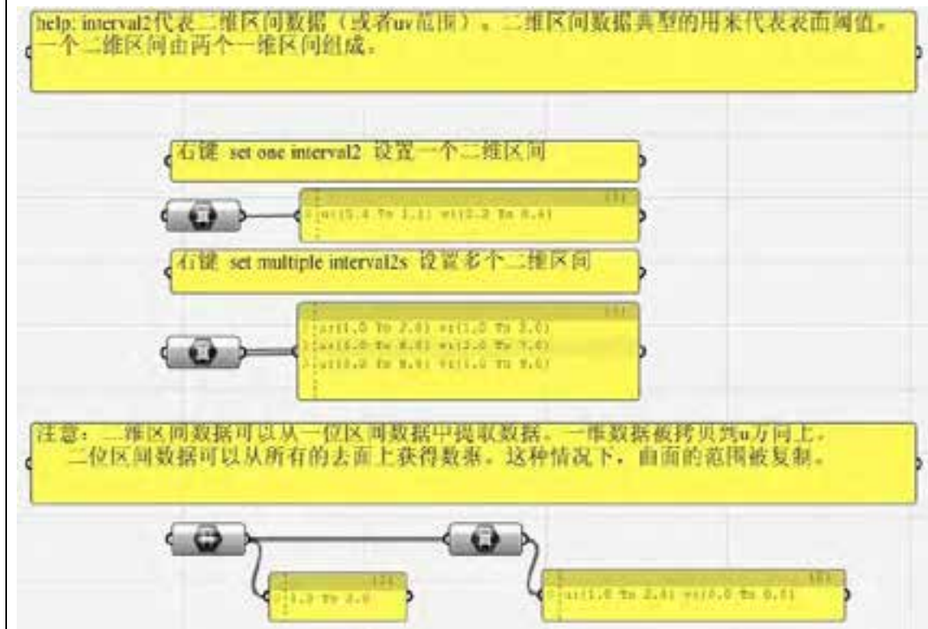


文化 Culture:

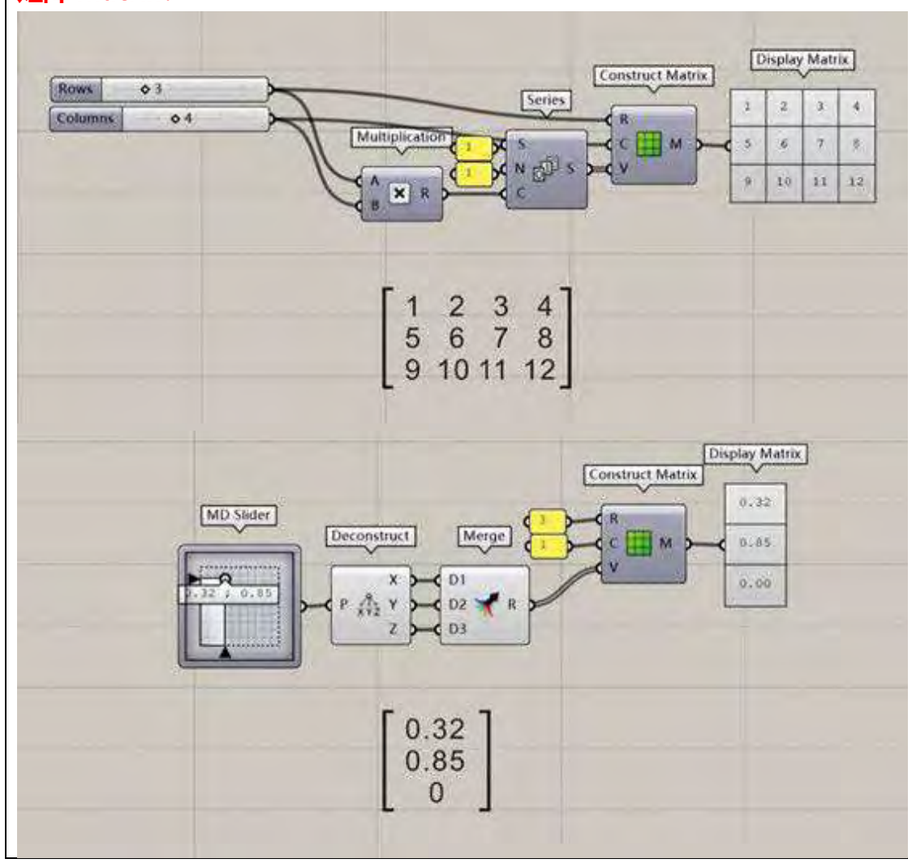
一维区间 Domain:



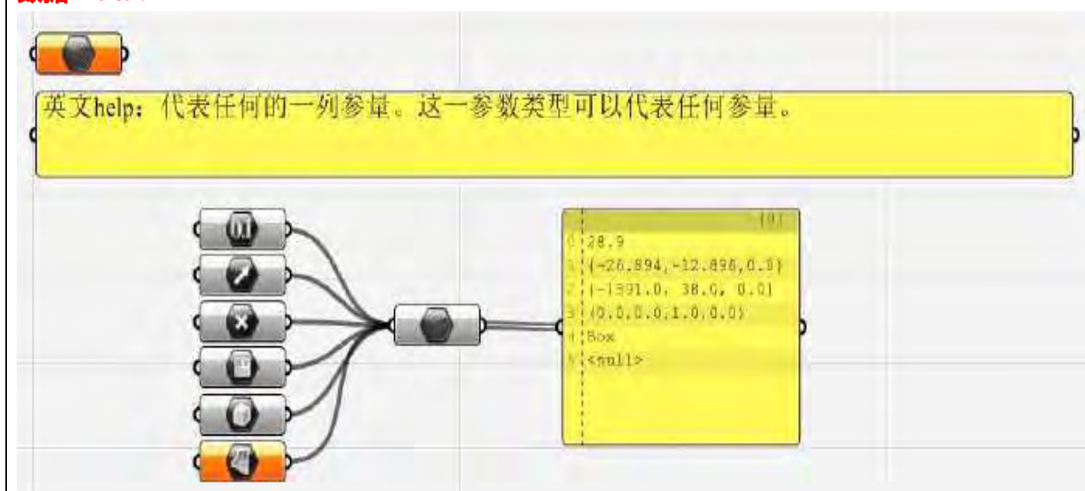
二维区间 Domain²:



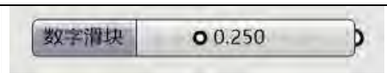
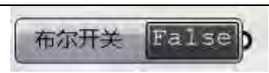
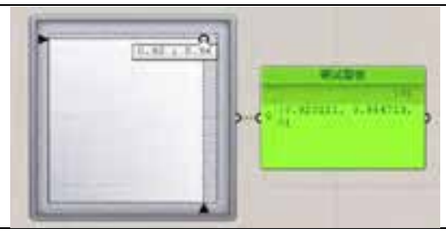
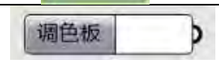
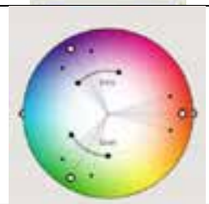
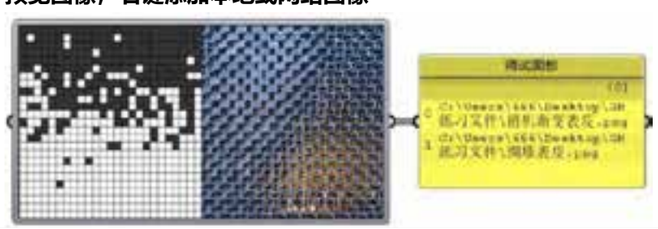
矩阵 Matrix:

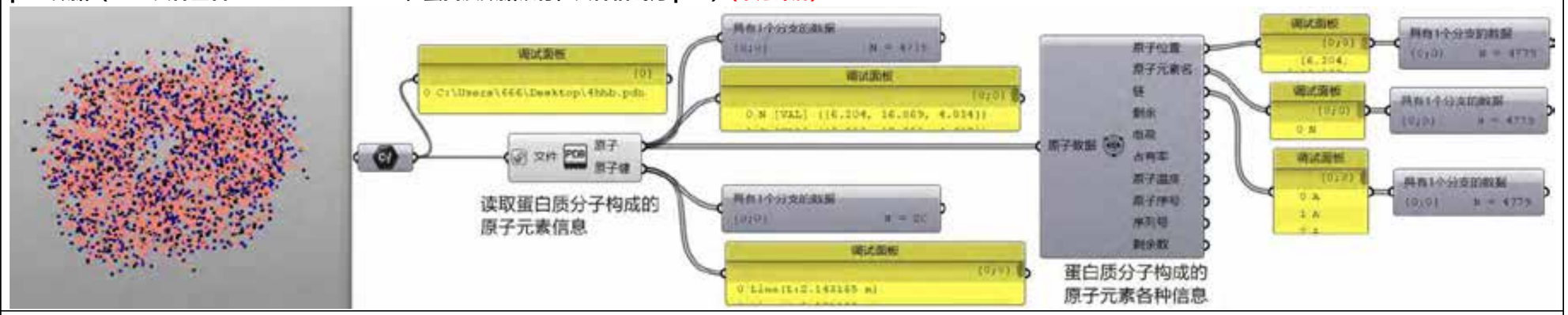
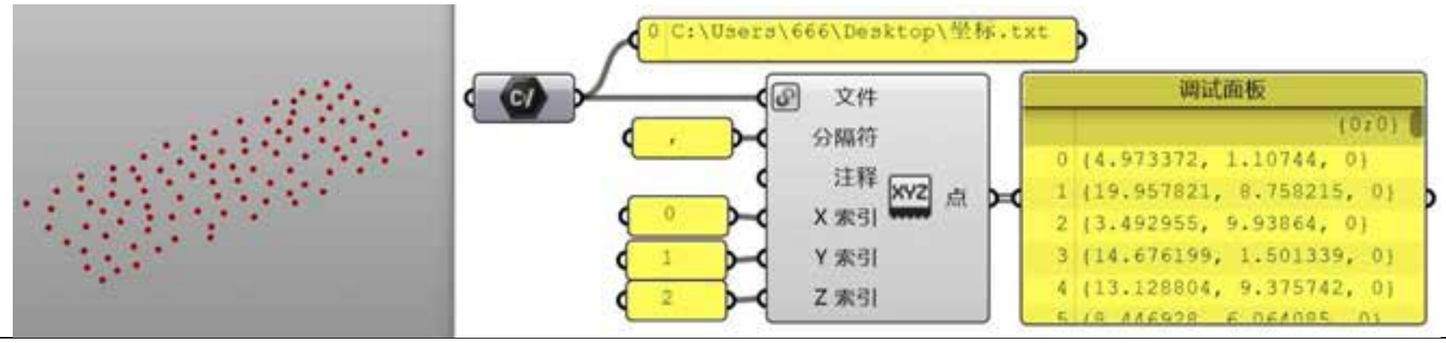
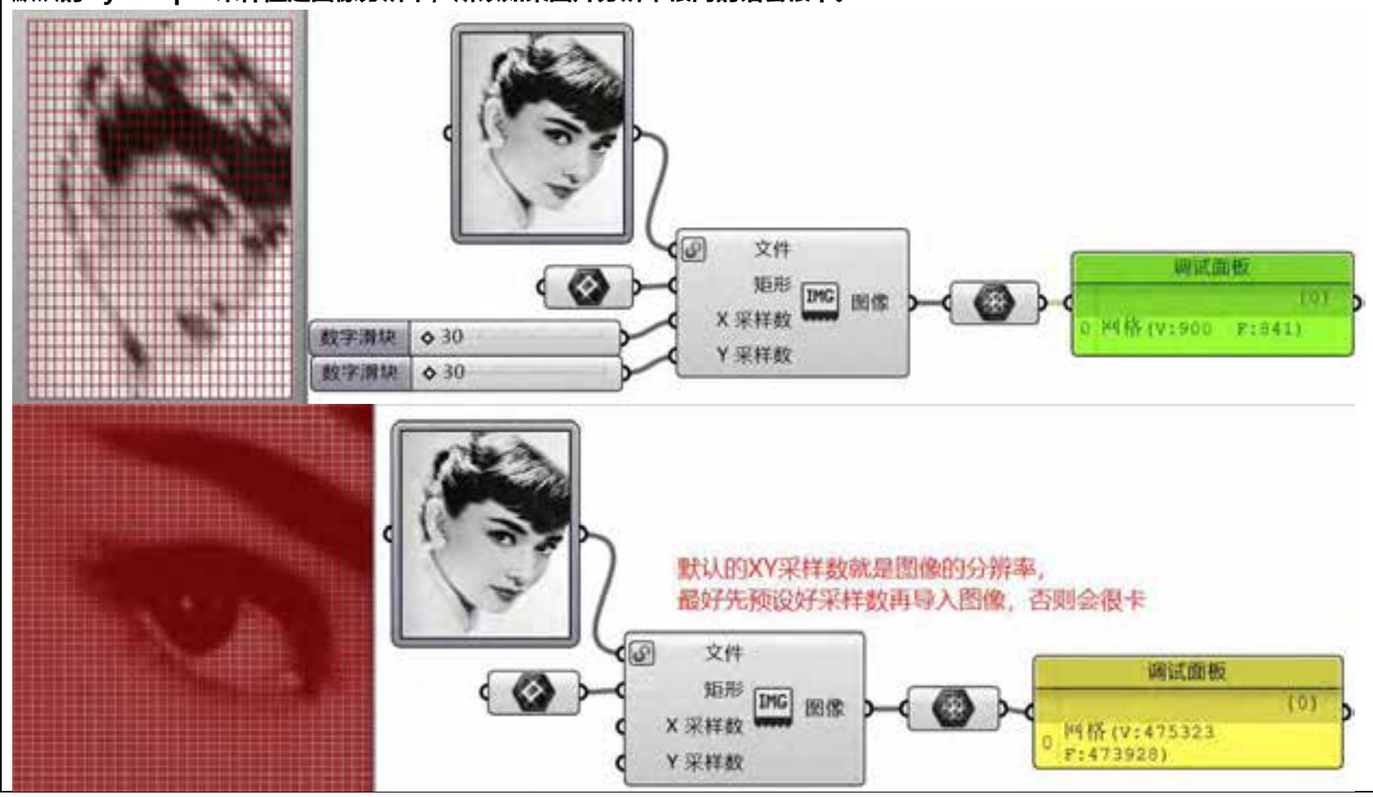


数据 Data:



输入 Input-（参数）

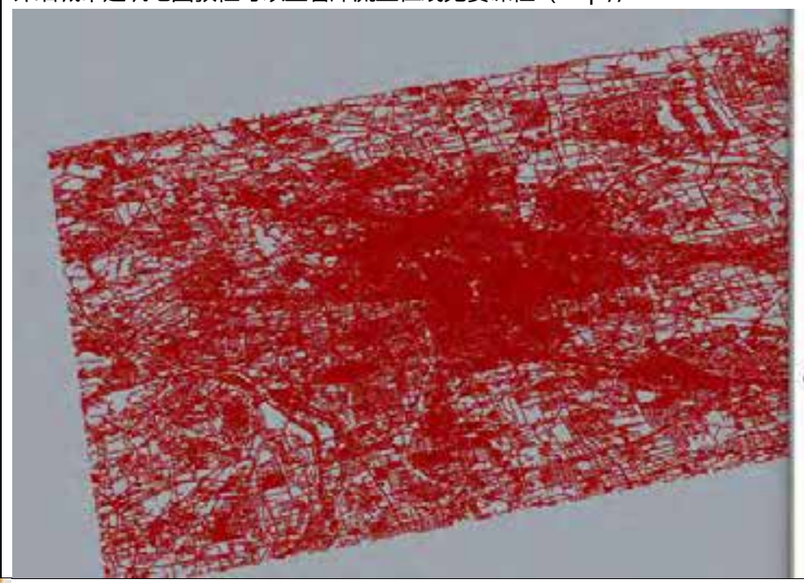
数字滑块 Slider		双击 gh 工作区空白处，直接输入数字回车可以调出（下方详解）	**布尔开关 Boolean**	 双击切换 True 或 False
预设值列表 Value List	下拉有预设好的数值、公式（下方详解）		**按钮 Button**	单击按住为 True，松开为 False。右键可以设置默认状态和点击状态。
多维滑块 MD Slider	在默认 0 到 1 的区间取二维坐标值，可以双击修改自定义区间。多用于曲面归一化后找曲面上点的位置。 		**调试面板 Panel**	常用于查看运算器数据，或者输入各种文字文本数据。双击 gh 工作区输入 “//” 调出空白面板，“//+文字” 可以直接输入对应的数据（下方详解）
数字滚动器 Digit Scroller	滚动运算器取值，或者双击自定义数值。		**控制旋钮 Control Knob**	手动转动按钮取值，或者双击进入编辑设定数值
函数映射器 Graph Mapper		右键可以选择不同的函数图表类型，用来改变数据趋势、控制曲线等，双击操作面板进入编辑，很常用的运算器（下方详解）	**取色器 Colour Slider**	设定颜色输出，但是图标太大，不常用。一般用面板较小的调色板
时钟 Clock		用来控制时间，比如日照分析	**调色板 Colour Swatch**	 右键指定颜色
图像采集器 Image Samper		图像采样（下方详解） 可以查看文档末尾的实际案例 ——利用图像采集器 image samper 进行图片干扰	**日历 Calendar**	输出日期数据
调色轮 Colour Wheel		调色轮，通过预设的 1-4 种颜色（默认为三色，空心点为取色点），调节颜色范围（实心点）及饱和度（Sat），亮度（Lum），输出对应数量的颜色值（下方详解）	**对象详细信息 Object Details**	可以获得拾取物件的图层，名称、颜色等各种信息。
色彩渐变器 Gradient		颜色渐变器，默认 0-1 区间，通过输入 0 到 1 之间的参数在渐变色带上取色。与 PS 的渐变颜色编辑类似，右键可以选择多种预设渐变色，单击彩色标签插入颜色，右键点击控制点编辑颜色，也可以双击控制面板打开编辑窗口（下方详解）		
图片资源 Image		预览图像，右键添加本地或网络图像 	**读取文件 Read File**	类似 Windows 的记事本打开任意类型的文件，所以任何类型的文件都能读取。

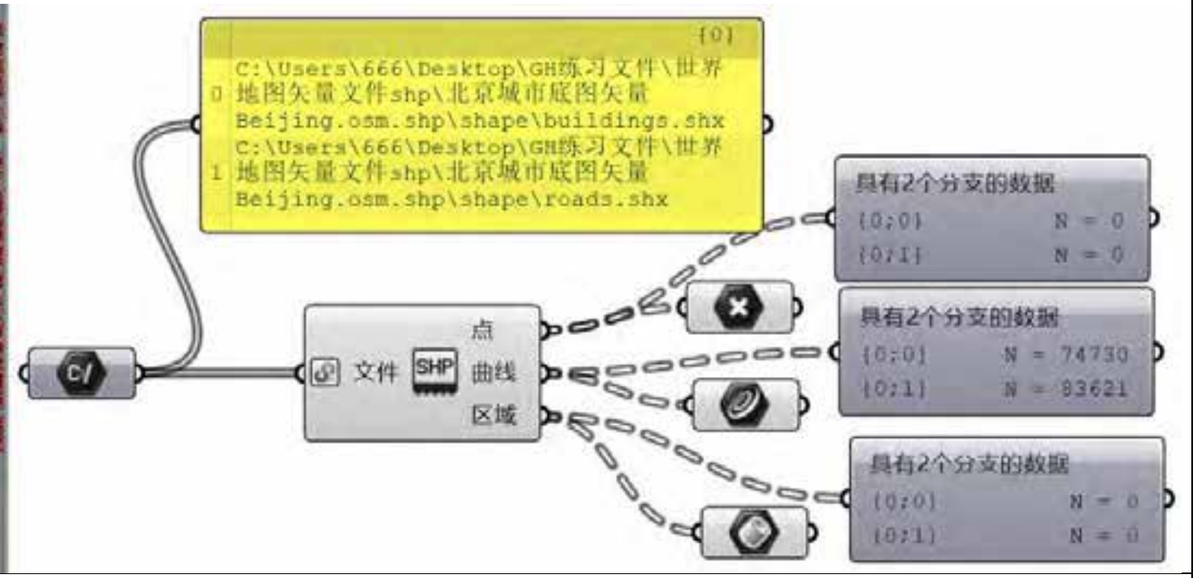
导入 PDB 文件 **Import PDB**		**pdb 数据（PDB 文件全名 Protein Data Bank，蛋白质数据银行，文件格式为 pdb）（下方详解）** 
原子数据 **Atom Data**		
导入坐标 **Import Coordinates**		**从包含坐标信息的文本导入坐标信息，生成点。有时候文件里的 xy 坐标是反过来的，这时候就 XY 的索引 0 和 1 可以对调下（XYZ 分别对应 012）。** 坐标.txt - 记事本 文件(F) 编辑(E) 格式(O) 查看(4.973372, 1.10744, 0 19.957821, 8.758215, 0 3.492955, 9.93864, 0 14.676199, 1.501339, 0 13.128804, 9.375742, 0 8.446928, 6.064085, 0 18.967846, 4.124965, 0 
导入图像 **Import Image**		**在指定矩形范围内，用导入的图像生成着色网格，可以自定义网格的数量。** **默认的 xy sample 采样值是图像分辨率，所以如果图片分辨率很高的话会很卡。** 

导入 SHP
Import SHP

导入 GIS（地理信息系统）地形数据。导入的数据尺寸默认是缩小了 0.000001 倍，所以要放大 100000 倍才是正常尺寸。城市的不同地图数据（建筑、道路等）处理要分开用多个运算器导入才能方便处理。(https://download.bbbike.org/osm/bbbike/这里可以下载北京的 shp 文件**)**

详细城市建筑地图教程可以查看犀流堂在线免费课程（http://www.rhinostudio.cn/course/1385）。

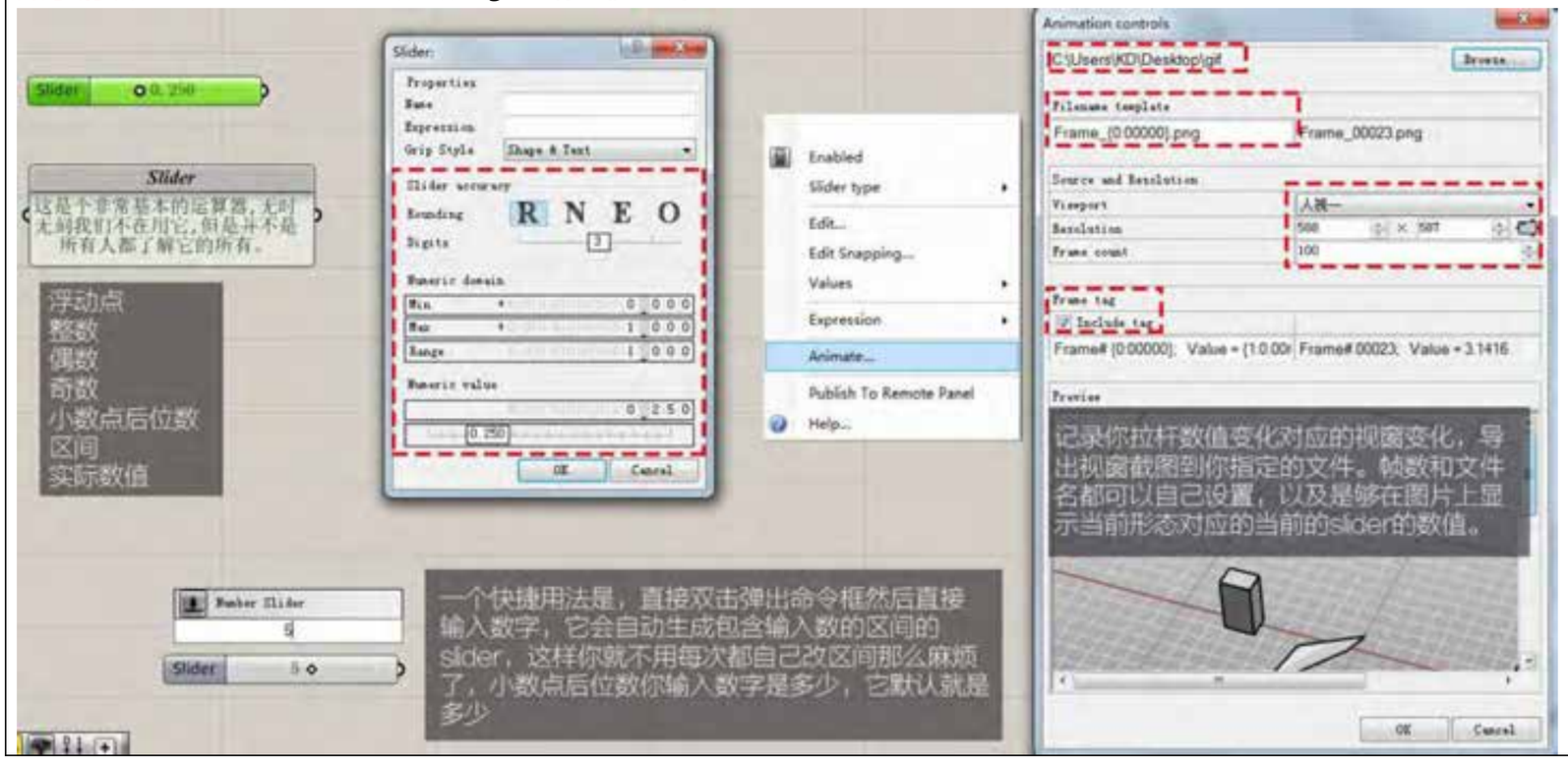




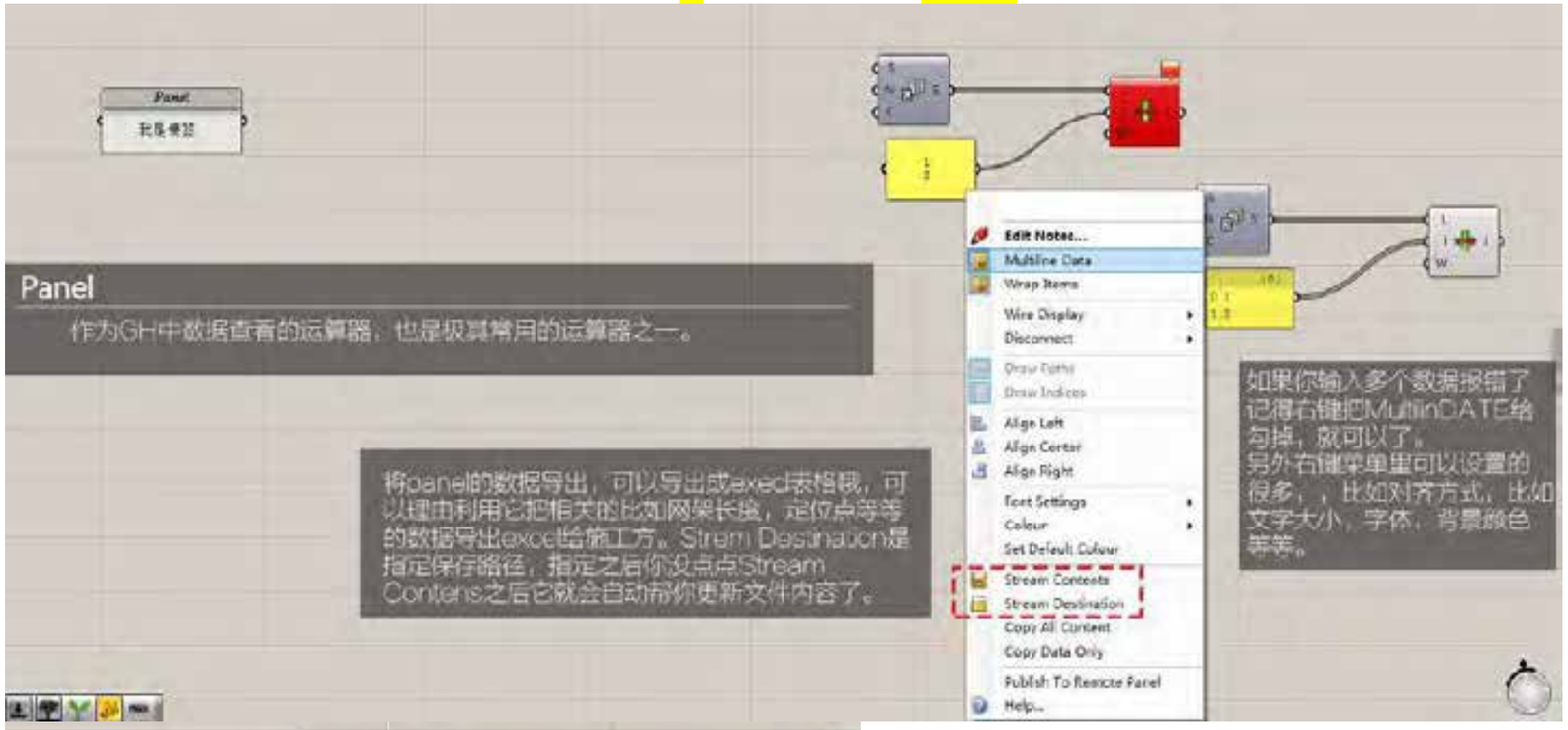
导入 3DM
Import 3dm

导入 rhino 模型，可以设置导入指定图层或名称的物体，但是材质会丢失

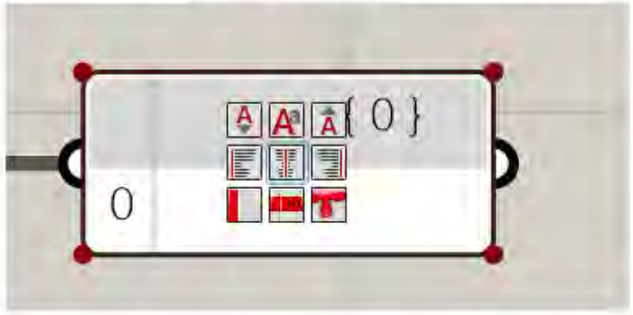
数字滑块 Slider: 快捷启动方式：双击 gh 工作区输入数值。



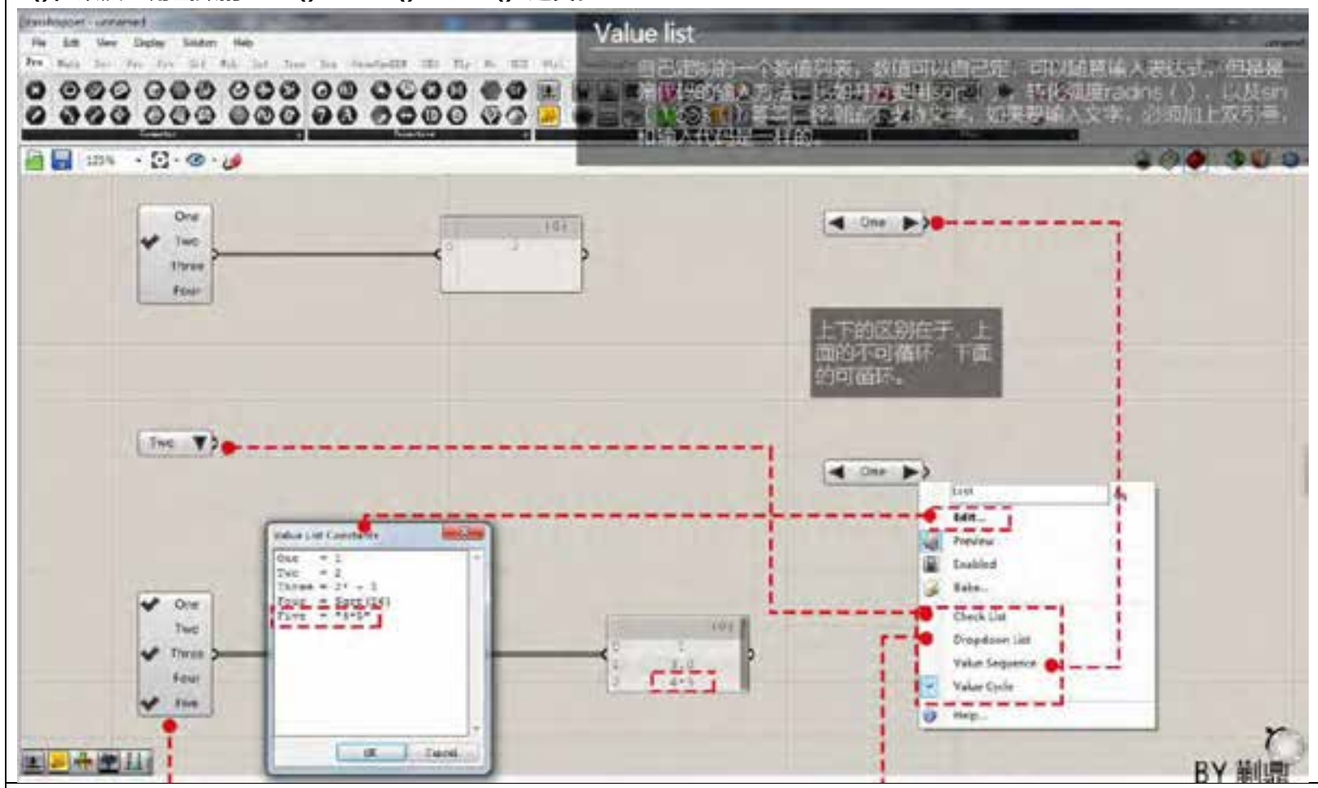
调试面板 Panel: **快捷启动方式：双击 gh 工作区输入 “//”调出空白面板，“//+文字”可以直接输入对应的数据**



另外，当panel放大到一定程度的时候，你就可以看到这些小按钮，可以方便的直接点击来修改。

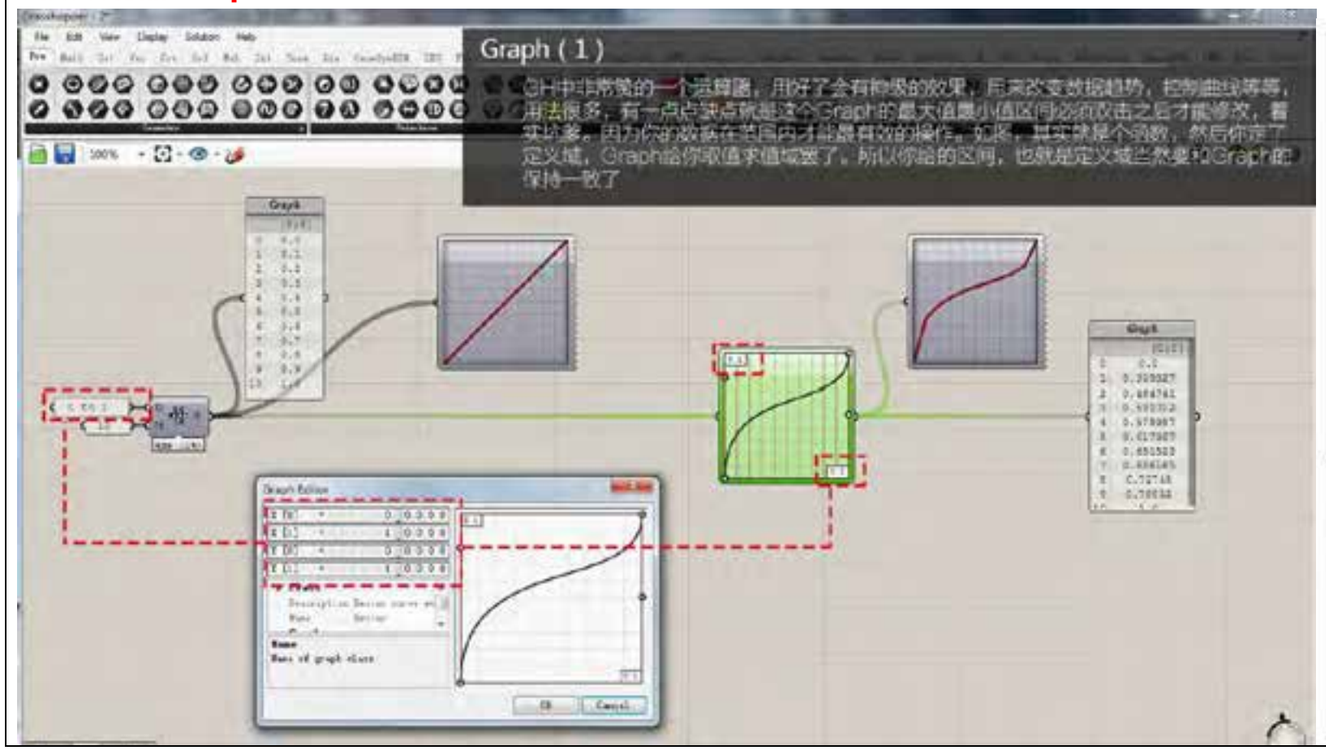


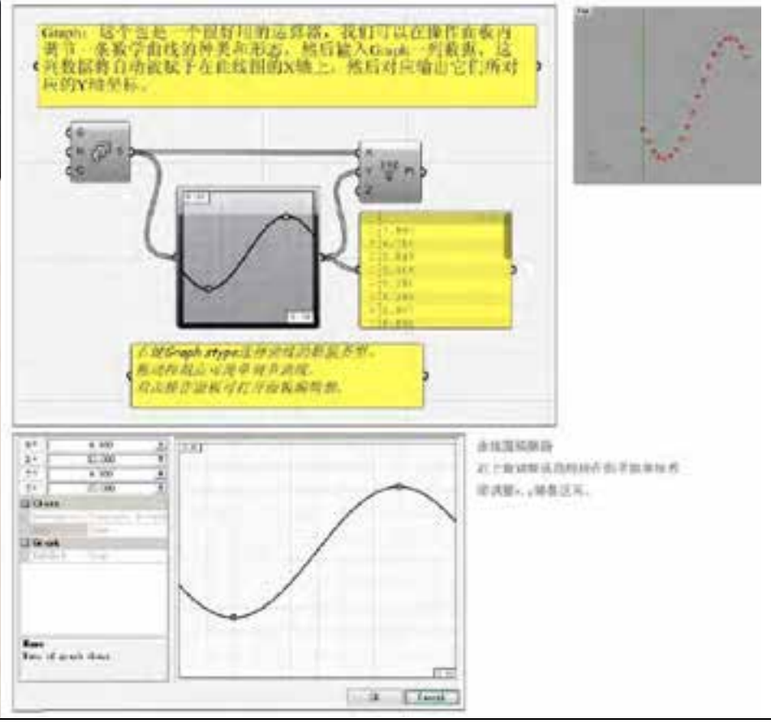
预设值列表 Value List: 单击展开列表选项，双击进入编辑窗口，默认的列表数值有如下四种，可以选择默认或输入自定义的表达式，运算器会计算出表达式的结果。常用的表达式比如求平方根 sqrt（），转化弧度 radins（），以及三角函数的 sin（） cos（） tan（）之类。



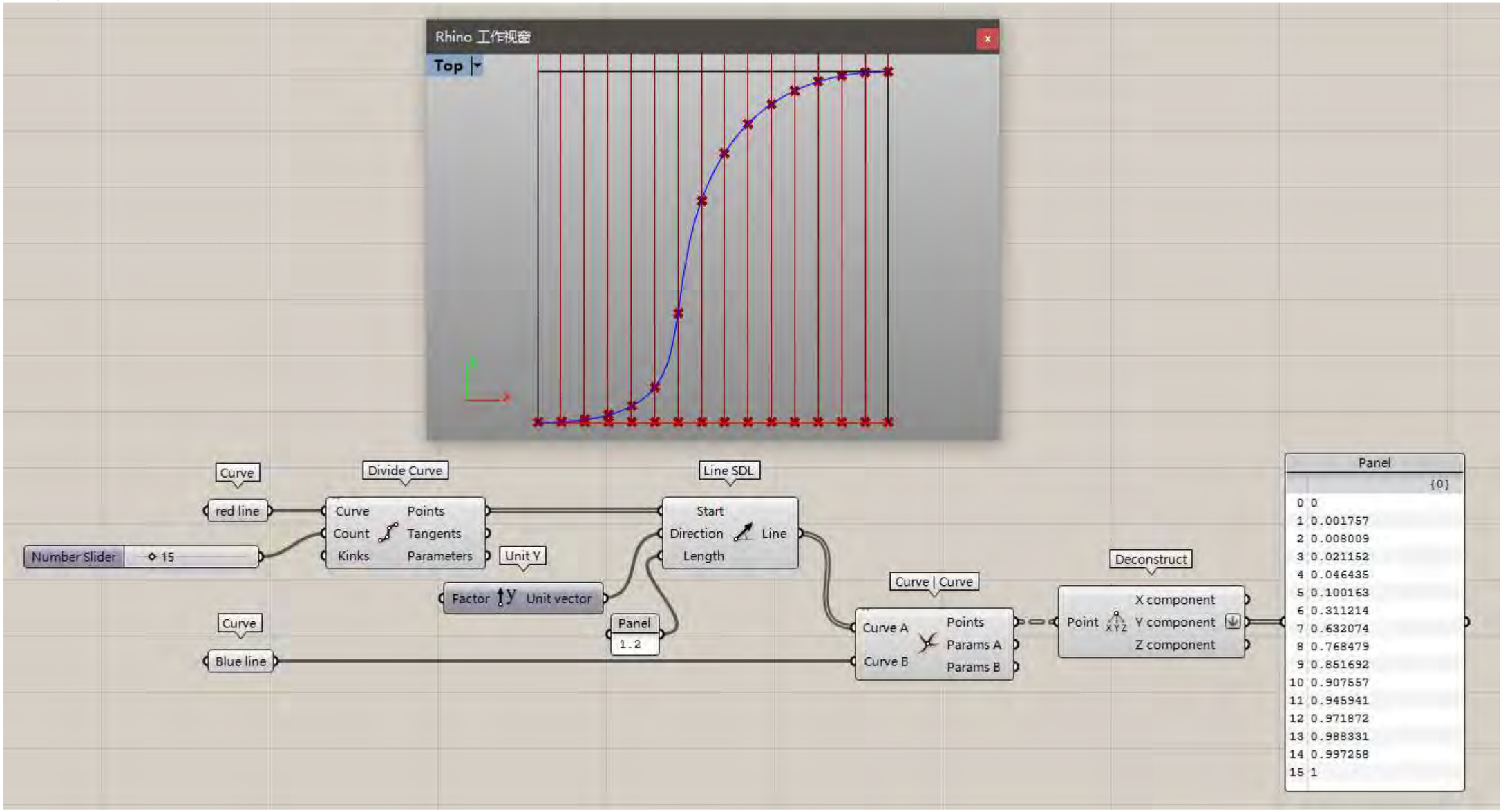
右键可以切换列表显示方式

函数映射器 Graph:





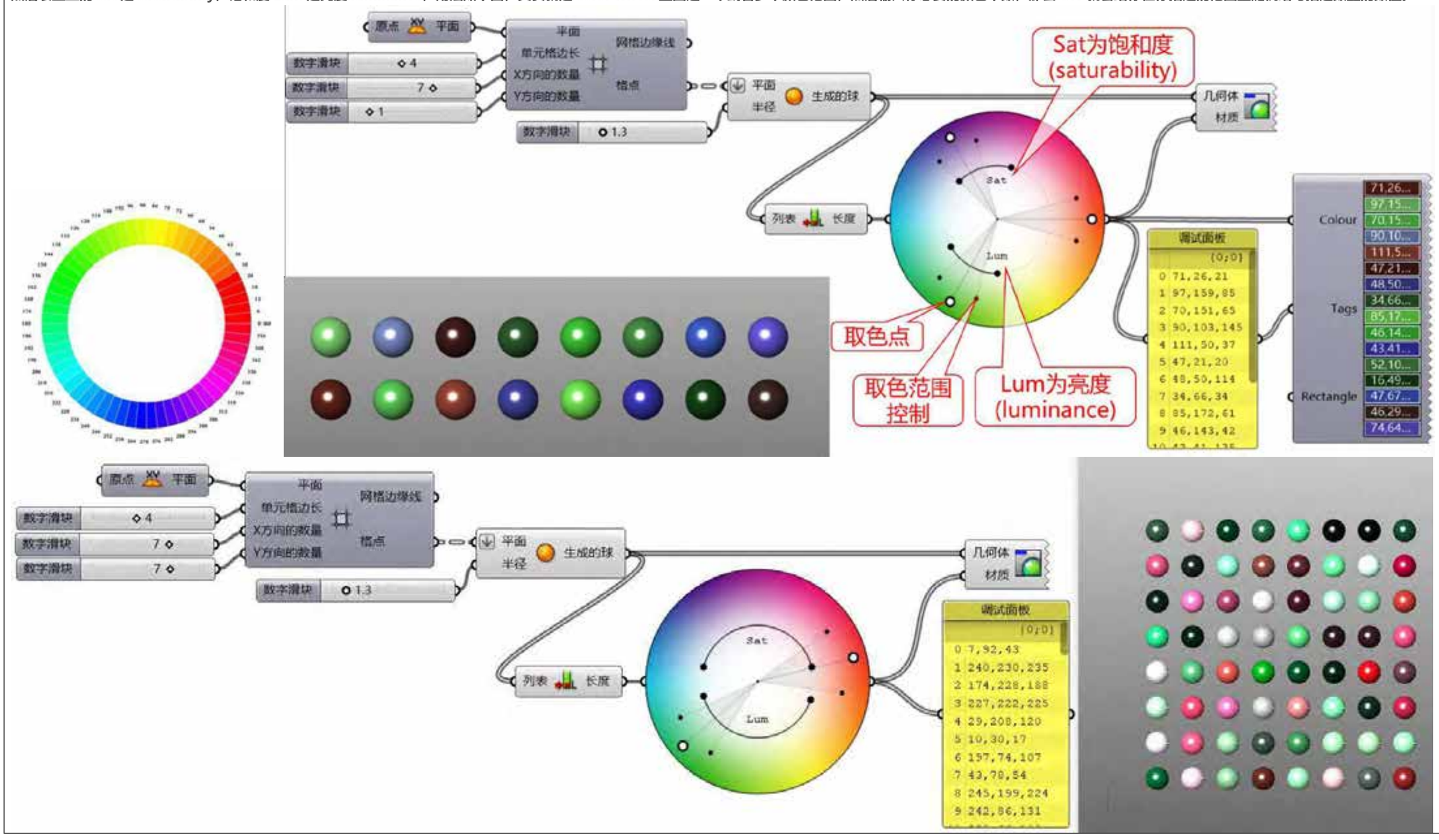
不过还有个问题，这个运算器自带了很多函数，这些函数大部分情况下都可以满足使用需求，但是并不能满足所有的需求，有的时候我需要一些特别自由的曲线时该怎么办？这个运算器就做不到了，不过没关系，秉承我们一贯的宗旨，没有 rh&gh 不能做的，只有我们不知道的。这个时候就需要我们自己去人为搭建一个图形映射器了。如下图做法，人为在 rhino 中绘制 1*1 矩形，自己绘制曲线作为函数曲线，然后通过直线相交的方式获得 y 值。从而得到想要的结果。

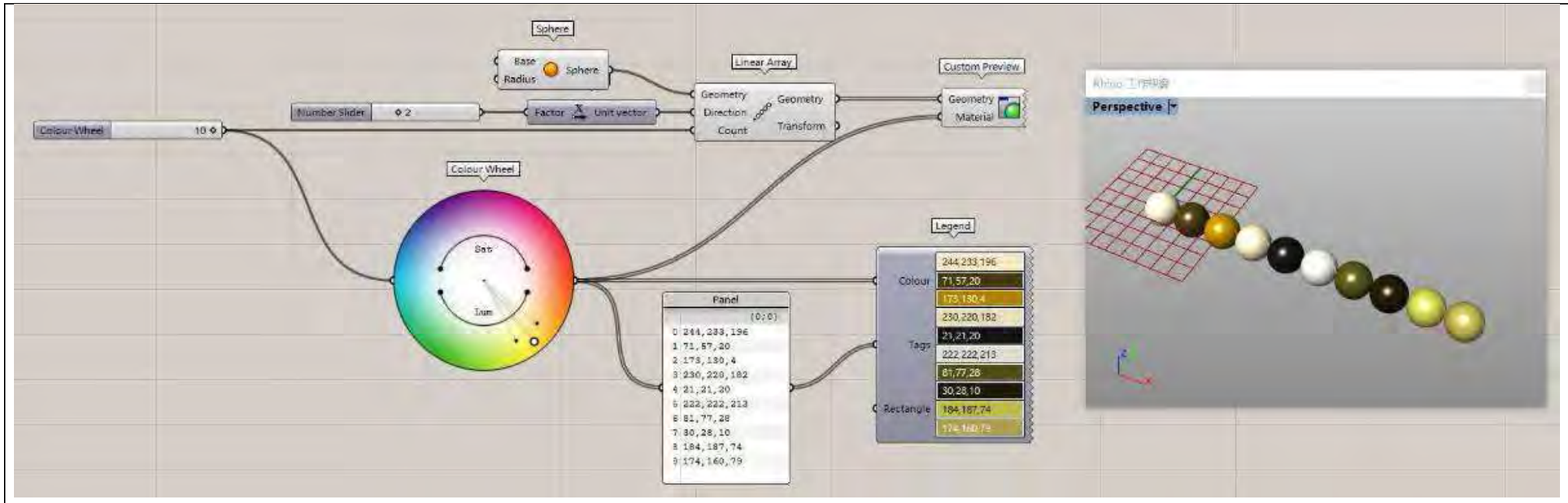


调色轮 Colour Wheel(摘自犀流堂)：

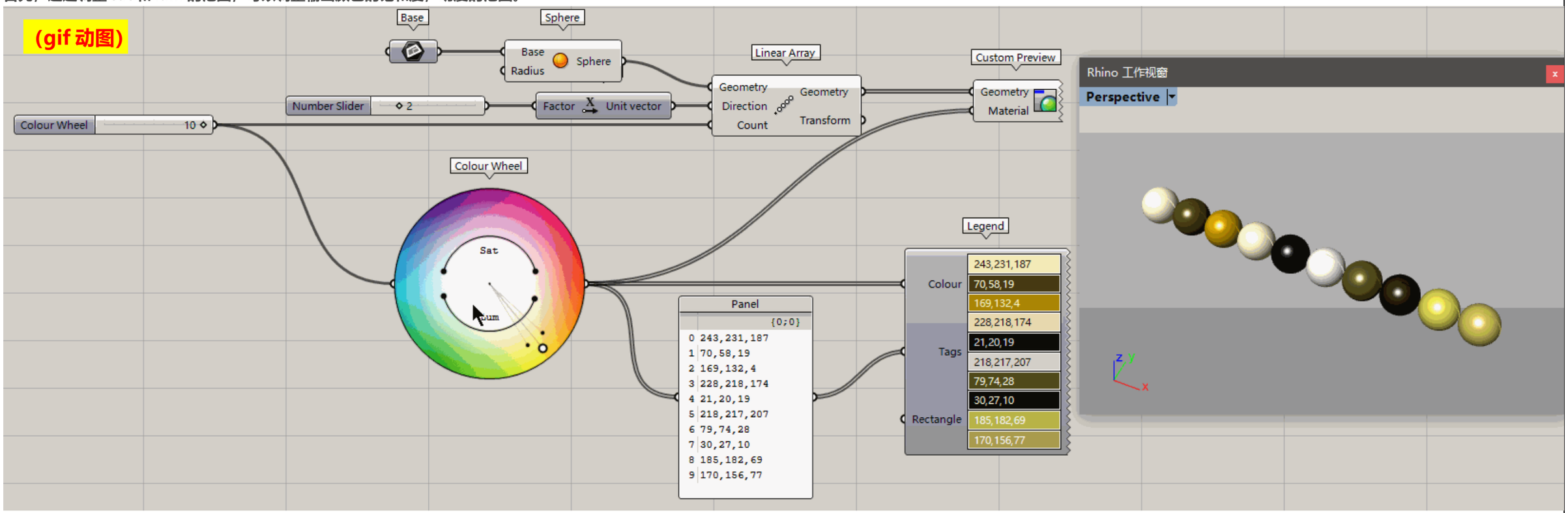
GH 当中颜值最高的运算器之一，也是很多人会忽视的运算器，其实挺好用的，虽然看起来挺吓人的。先来简单介绍一下这个界面，背景盘是一个色轮，只不过我们 ps 里比较常见的是方形的，都是一种颜色空间表达方式。

然后表盘上的 sat 是 saturability，饱和度 Lum 是亮度 luminance，用法如下图，其实就是 color wheel 里圈定一个或者多个颜色范围，然后输入你想要的颜色个数，那么 CW 就会给你在你指定的范围里随机给与指定数量的数值。

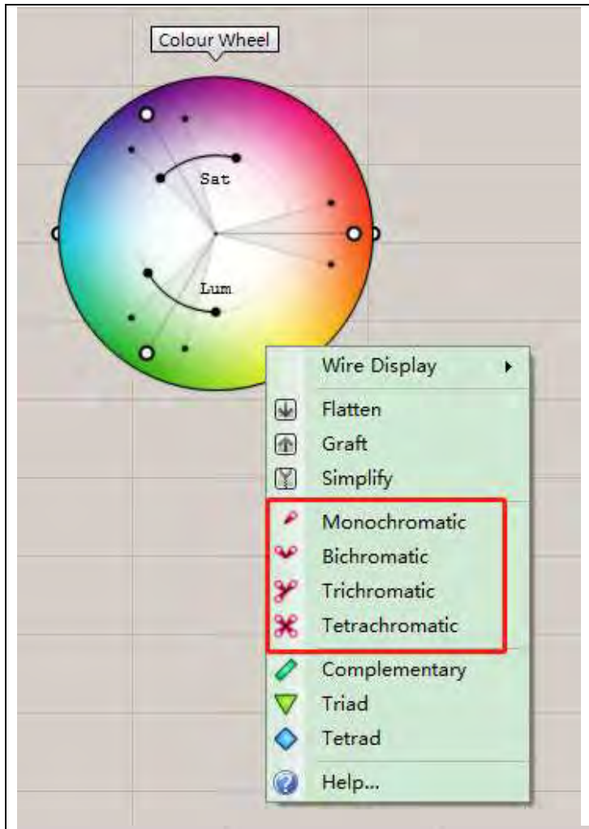


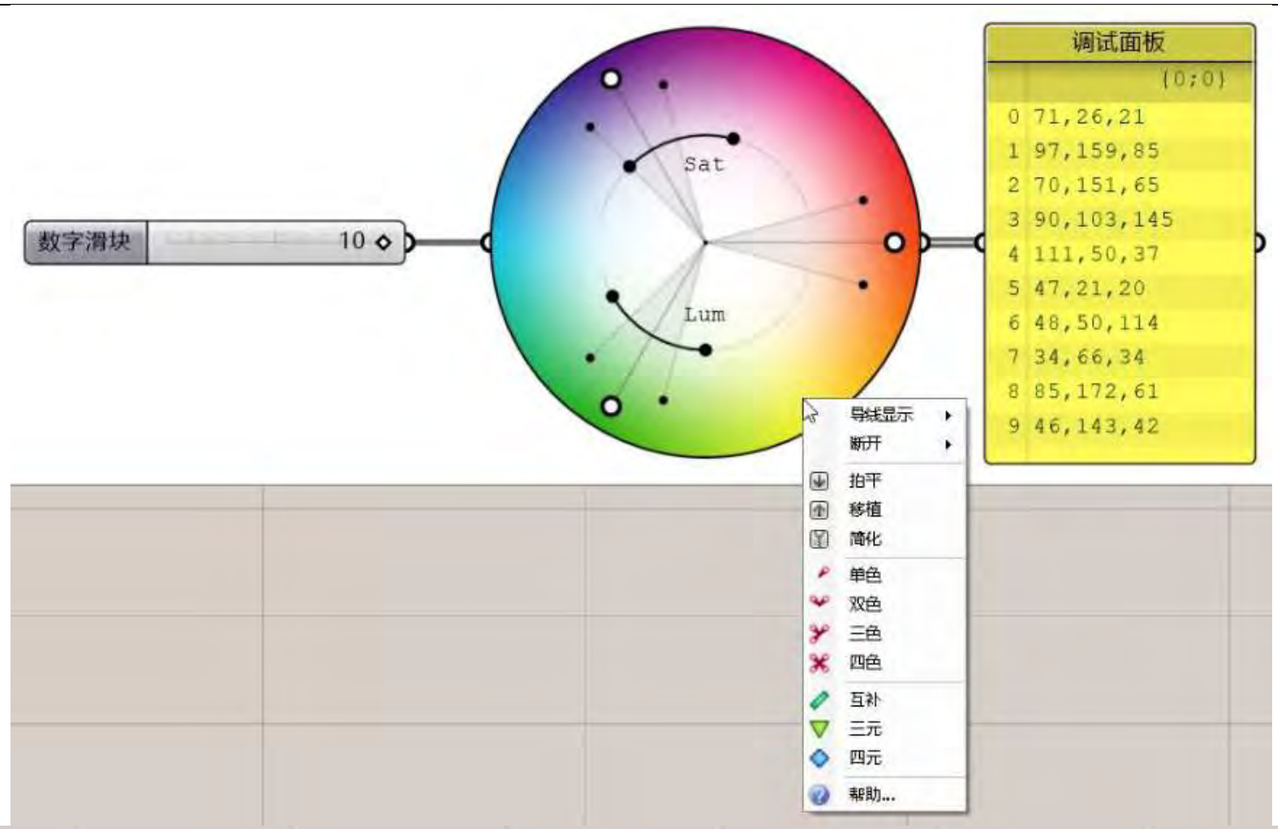


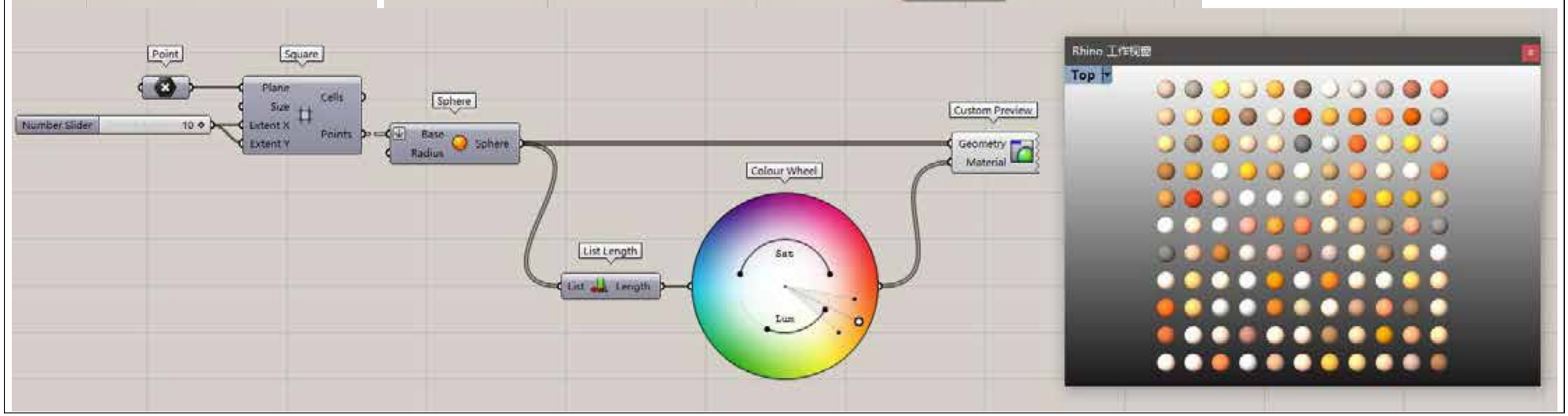
首先，通过调整 sat 和 lum 的范围，可以调整输出颜色的饱和度，明度的范围。

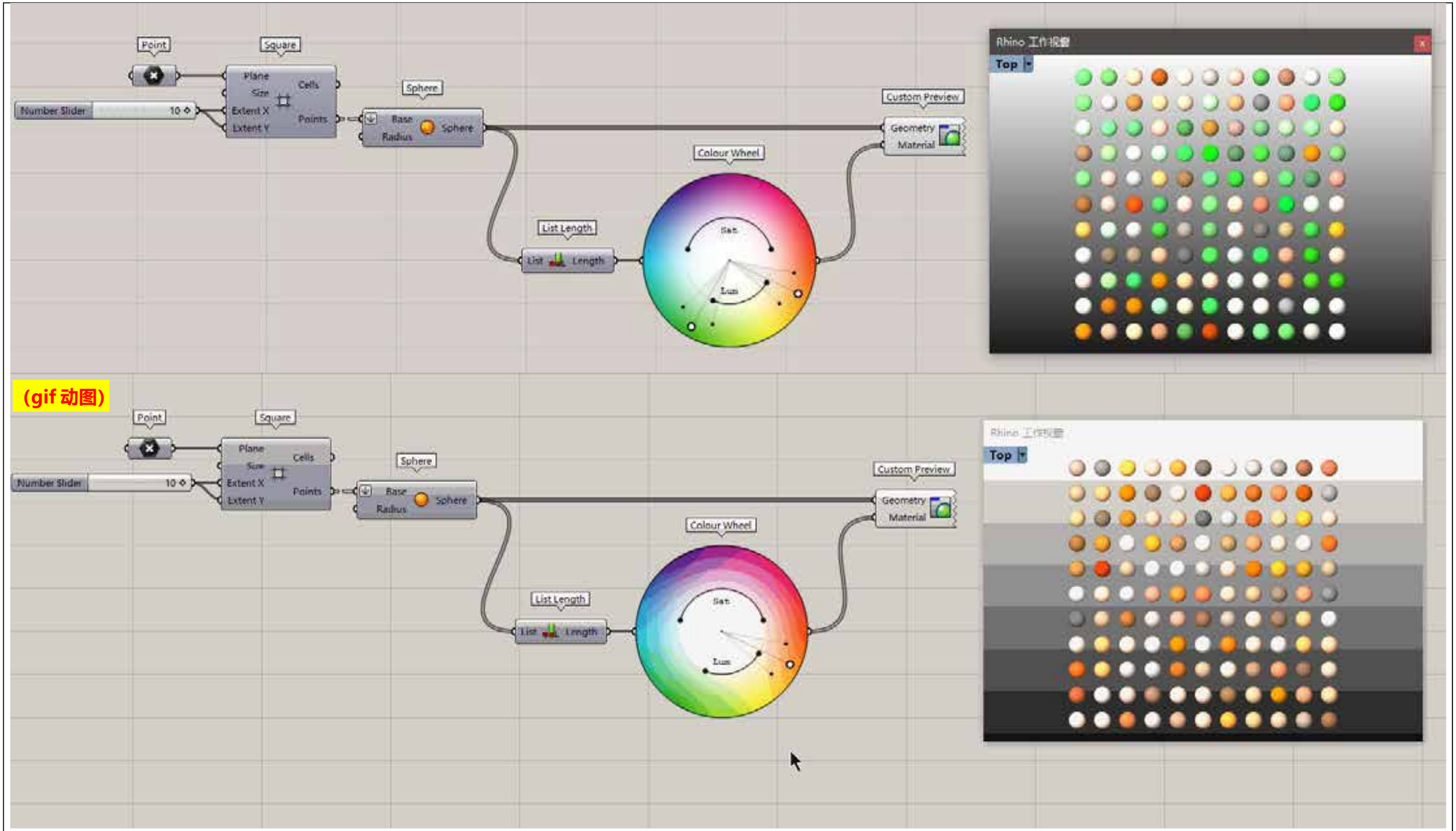


然后，你可以调整颜色范围和数量，然后你可以右键选择需要几个颜色区间 1-4，图标做成花瓣状，多讲究。


Colour Wheel
Sat
Lum
Wire Display
Flatten
Graft
Simplify
Monochromatic
Bichromatic
Trichromatic
Tetrachromatic
Complementary
Triad
Tetrad
Help...

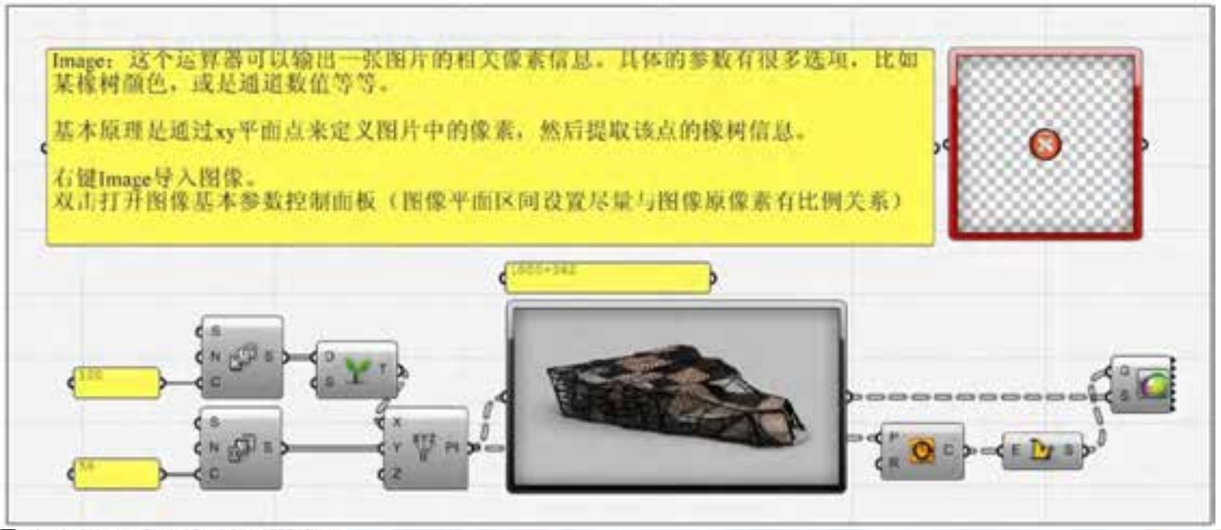

数字滑块
10
Sat
Lum
调试面板
{0;0}
0 71,26,21
1 97,159,85
2 70,151,65
3 90,103,145
4 111,50,37
5 47,21,20
6 48,50,114
7 34,66,34
8 85,172,61
9 46,143,42
导线显示
断开
拍平
移植
简化
单色
双色
三色
四色
互补
三元
四元
帮助...


Point
Square
Plane
Size
Extent X
Extent Y
Cells
Points
Number Slider
10
Sphere
Base
Radius
Sphere
Custom Preview
Geometry
Material
Colour Wheel
Sat
Lum
List Length
List
Length
Top

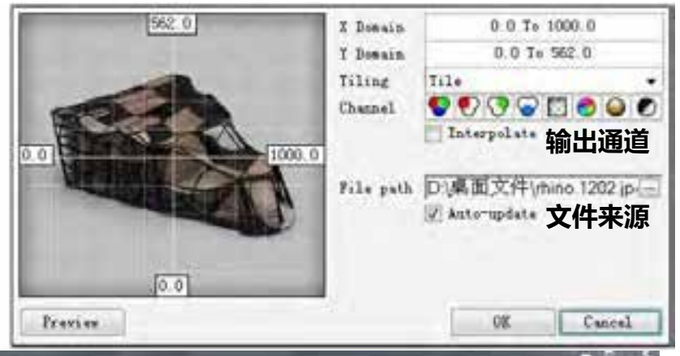


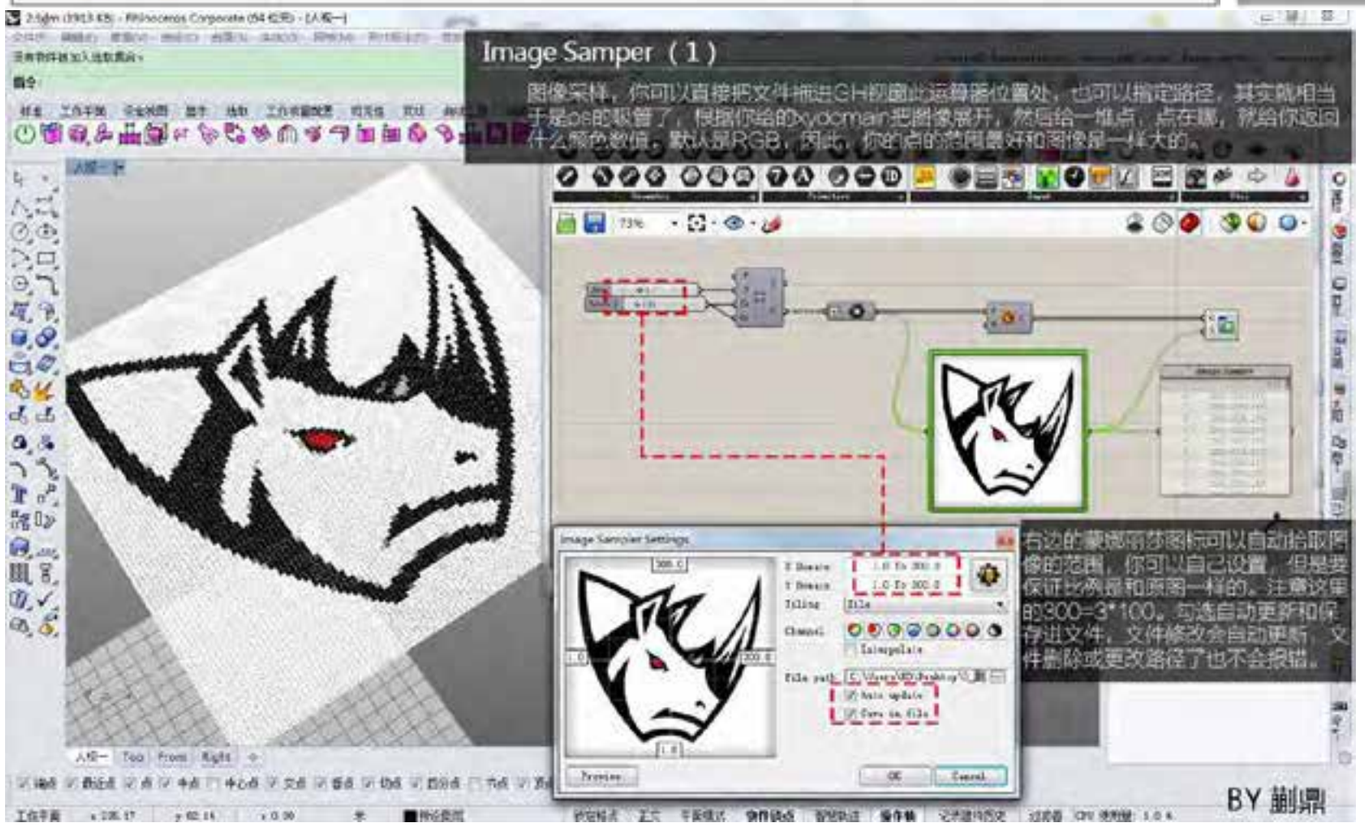
所以呢，colour wheel 是一个很好的得到随机颜色的运算器，，逼格也够高，不知道的人拿到你的运算器一定一脸懵逼，所以大家平常需要随机颜色的时候就可以搞起来了。

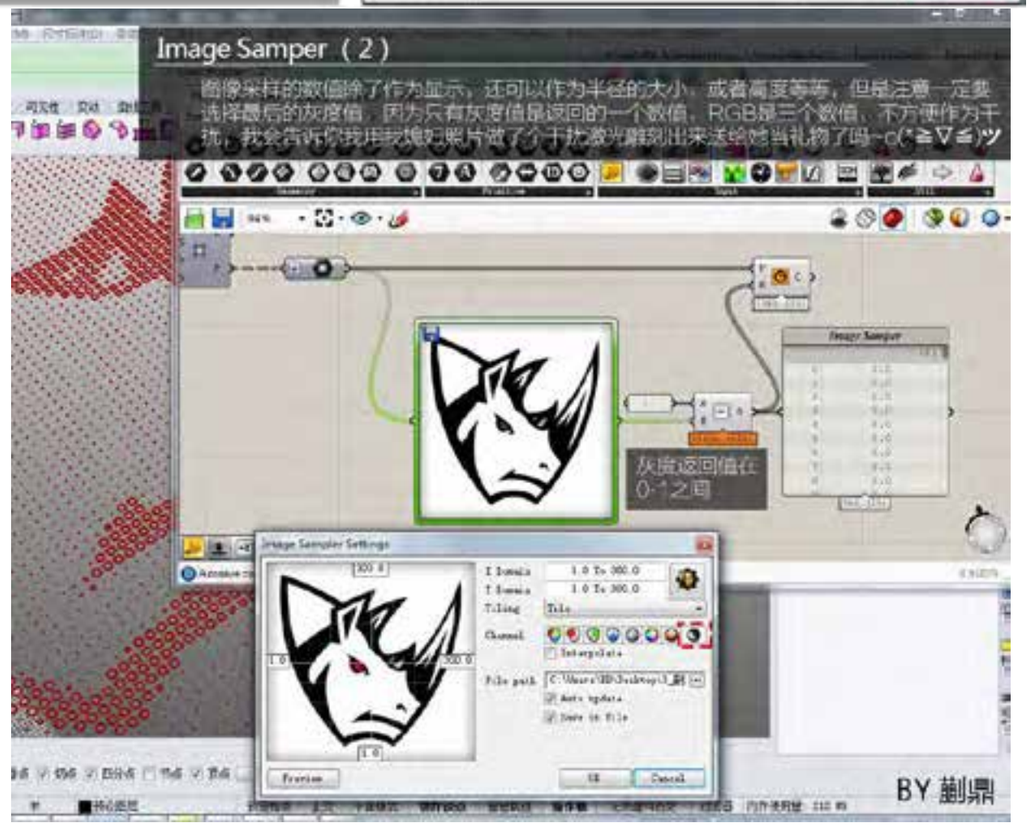
图像采集器 Image Samper：（文档末尾有实际案例演示）



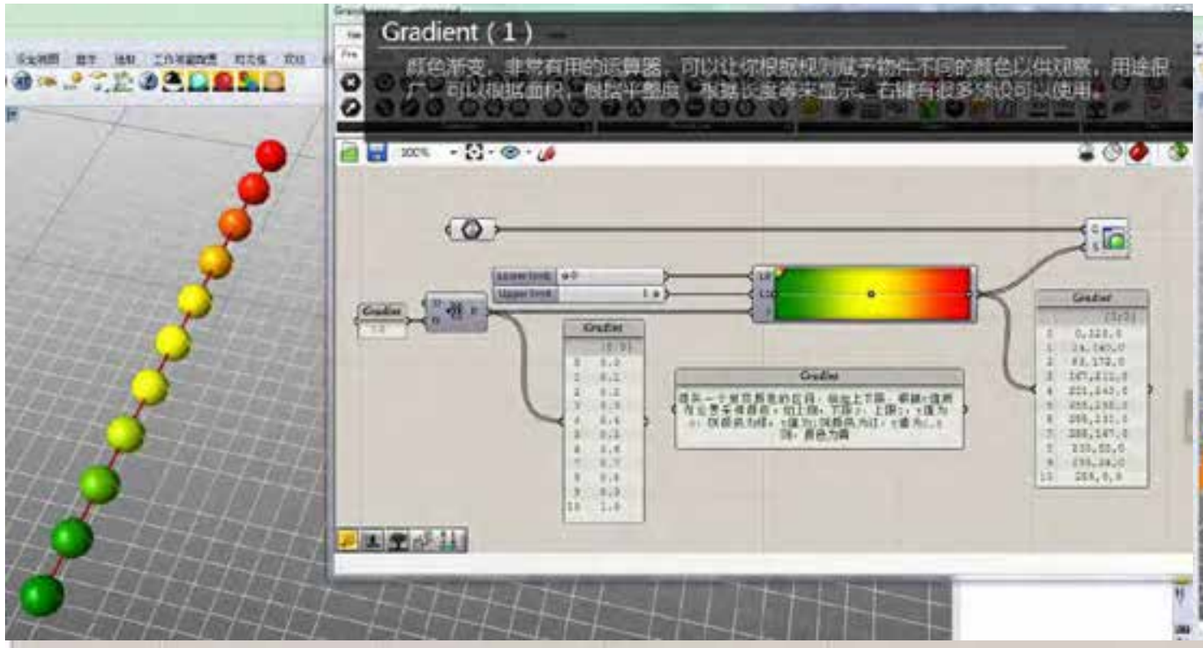
编辑器内的平面坐标区间（尽量符合图像像素数量的大小）

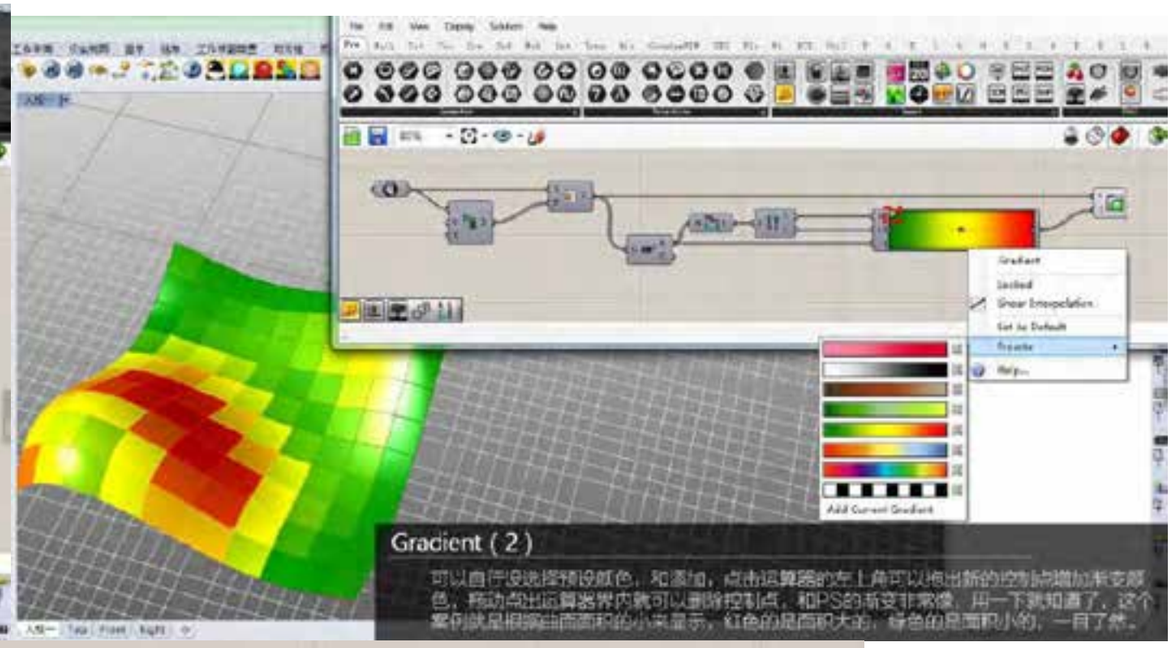


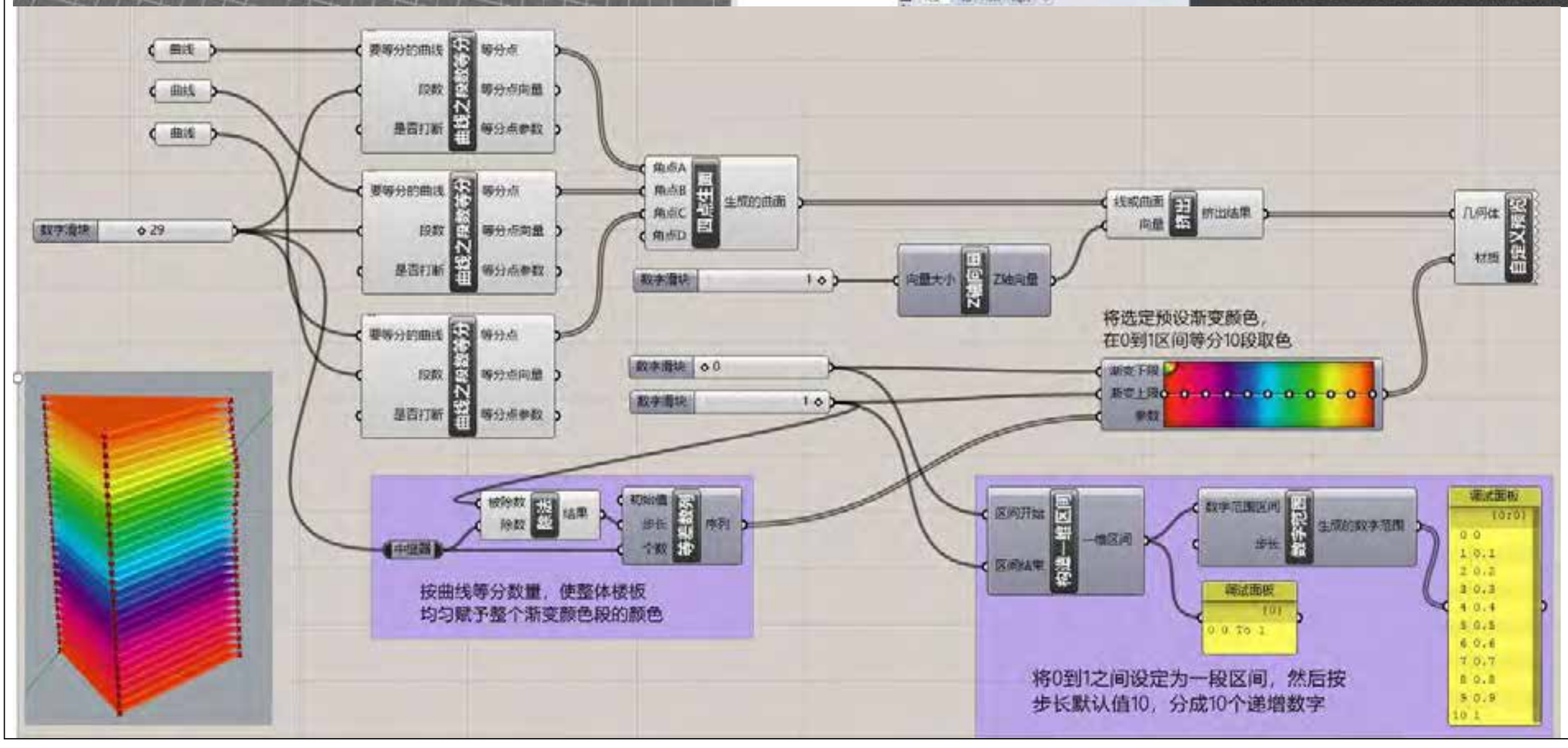




色彩渐变器 Gradient:



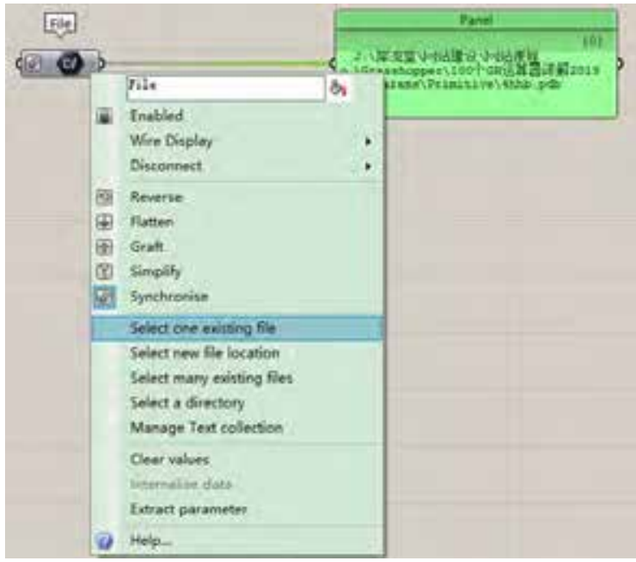




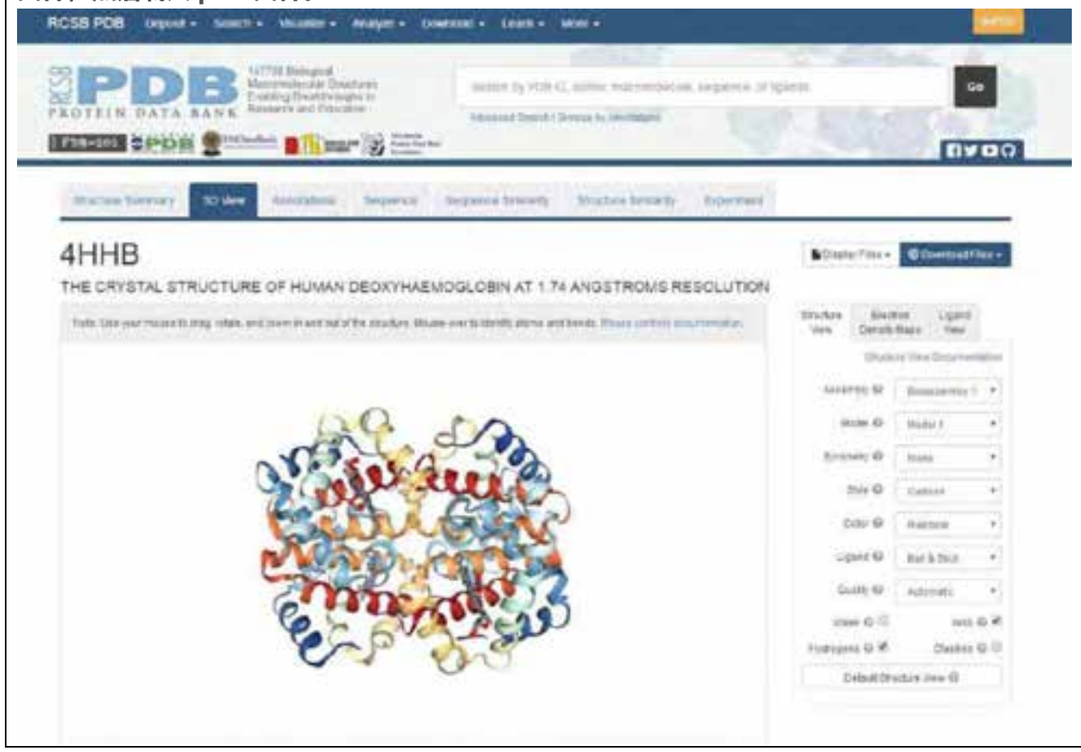
File 文件路径　　Import PDB 导入 PDB 文件　　Atom Data 原子数据

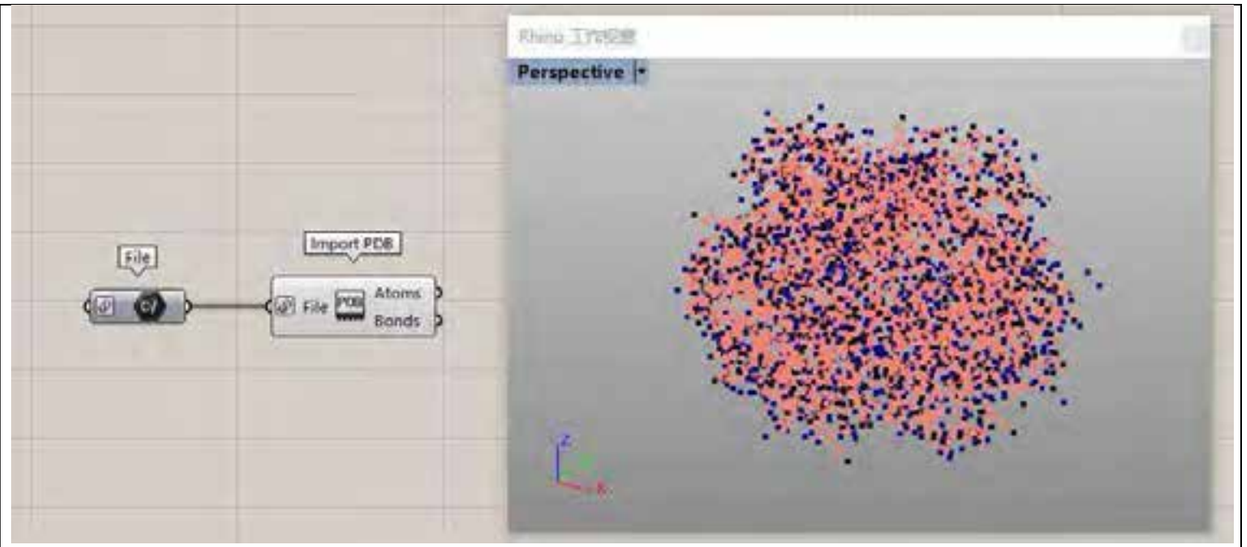
（网上找的 pdb 下载路径，5P22.pdb 为文件名，可以修改文件名下载不同的 pdb 文件 https://files.rcsb.org/download/5P22.pdb）

这三个我们放在一起讲吧，正好可以凑个例子。File 很简单，右键拾取文件路径，可以选一个文件，也可以选一个文件夹。

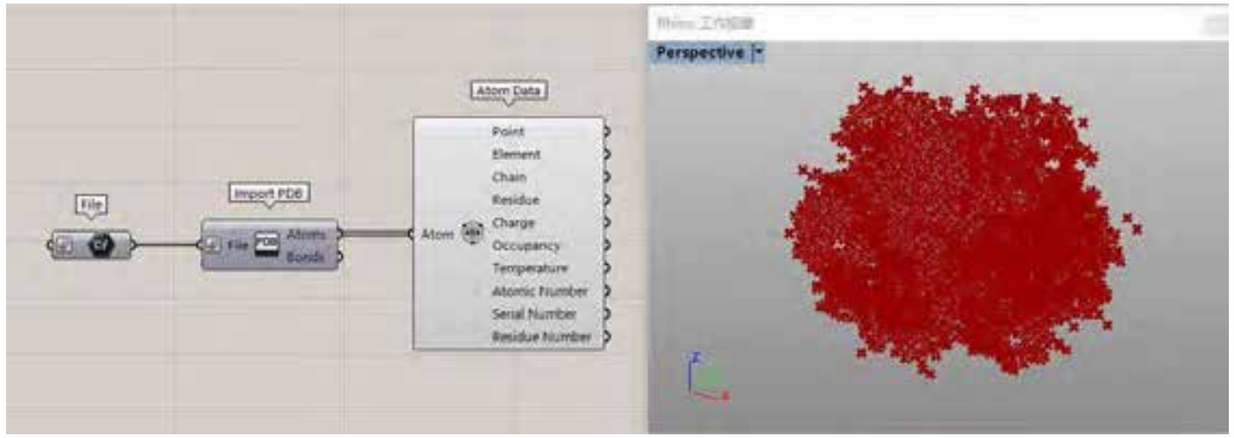


import PDB 是用来导入 PDB 文件的，而所谓的 PDB 文件全名 Protein Data Bank，蛋白质数据银行，文件格式为 pdb，网上随便一搜就有了，你可以在这个网站（https://www.rcsb.org/）下载 pdb 文件，然后载入 pdb 文件。

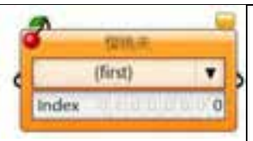
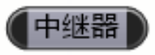
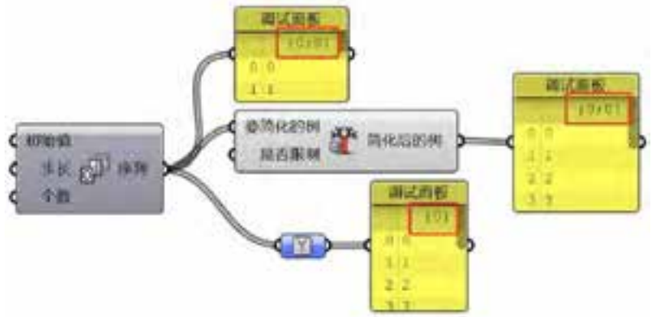
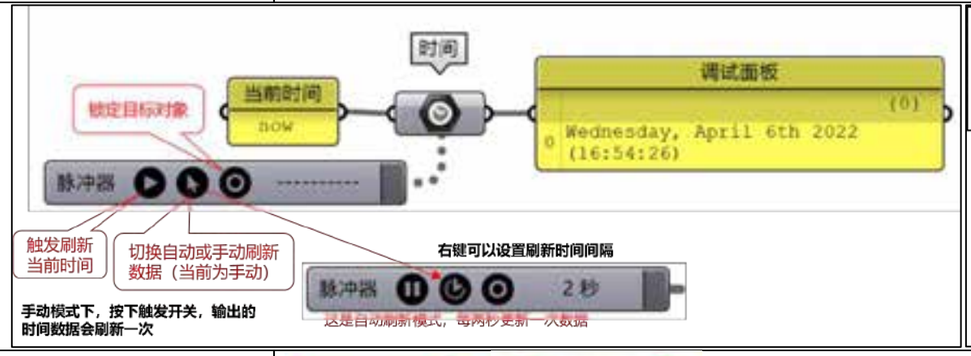


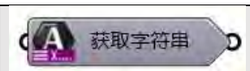
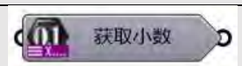
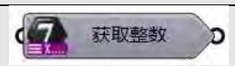
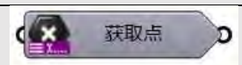
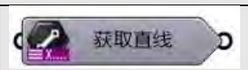


再接到 Atom Data，就可以获取这个蛋白质信息了，然后，emmm 就是另一个领域的事情了，大家就看看就好，除非，万一，你做个蛋白质博物馆，然后想从这里找灵感……

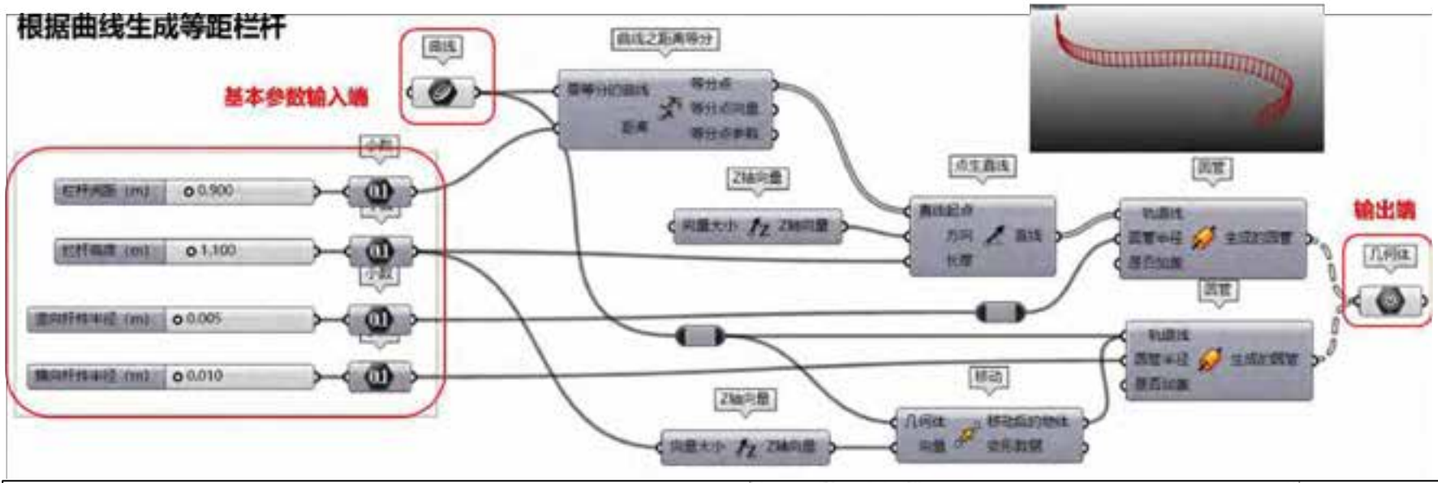


公用 Util-（参数）

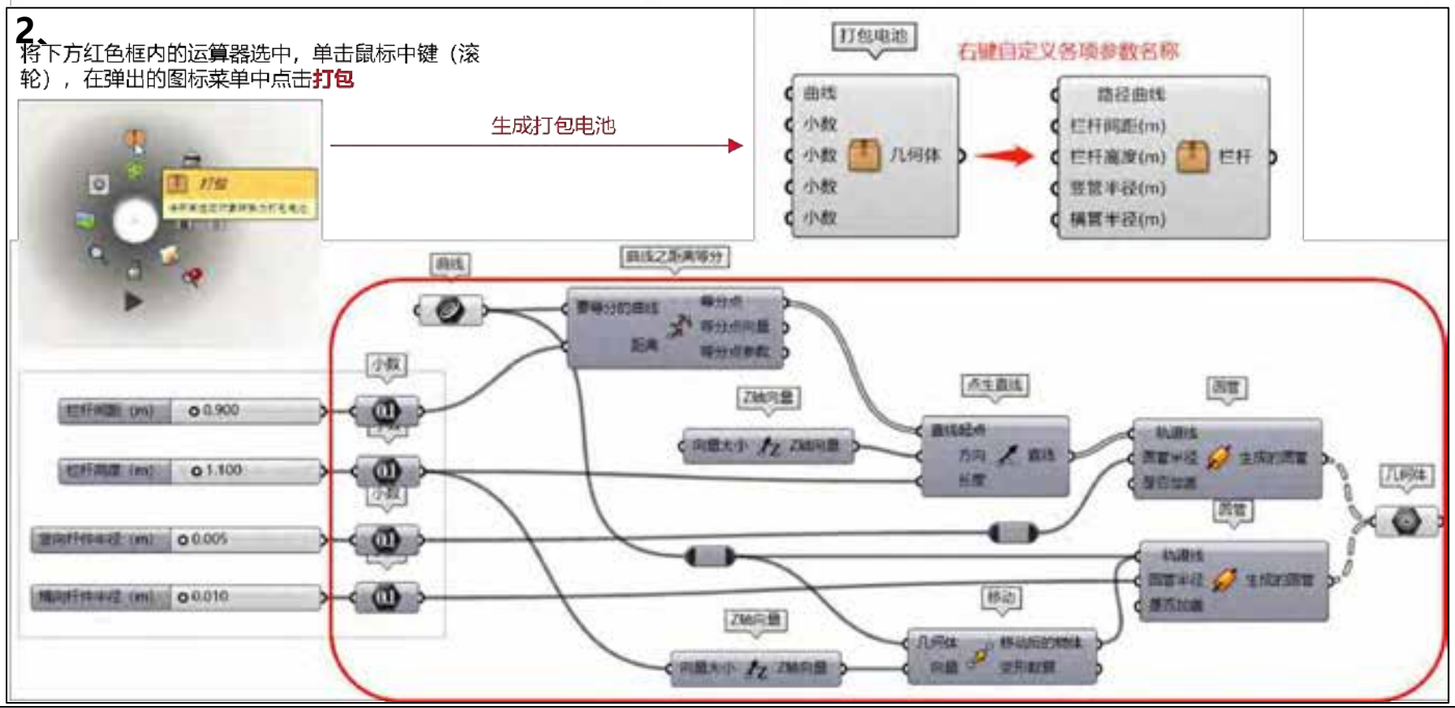
数据结构查看器 Param Viewer		**树形数据查看器，比较常用（下方图解）**	**樱桃夹 Cherry Picker**		**拾取指定数据结构分支下的指定序号的数据（下方图解）**
注释器 Scribble	注释器（双击编辑）	**添加文字标注。双击 GH 工作区，输入~符号就能调出来**	**跳转 Jump**		**传送门，双击可以在两个图标的视图位置切换**
中继器 Relay		**双击电池连接线会自动插入，用于整理电池连线，使视图美观**	**数据坝 Data Dam**		**数据流控制器（下方图解）**控制数据传输，一旦数据发生变化，就会停止传输，除非点击播放键；右键可以选择预设的时间间隔，控制数据传输间隔
数据记录器 Data Recorder		**记录数据的变化（下方图解）**			
简化器 Suirify					**简化数据结构,是简化树的强化版。单组数据时，树的简化运算器无法简化，这时用简化器就可以。**
脉冲器 Trigger					
电池输入端 Cluster Input:		**用于打包电池的输入和输出端。详见下方的电池组打包教程**			
电池输出端 Cluster Output:					
数据输入 Data Input		**可以把 GH 当中的包含数据结构的数据作为文件形式输出，然后再读取进 gh，适合多人之间配合使用。比如可以导出建筑日照数据发给其他人。（下方图解）**			
数据输出 Data Output					
加拉帕戈斯/遗传运算器 Galapagos:		**遗传运算器（计算最优解）** **基因组上右键可以选择并控制多个变量数据输入，找出适合度数据最接近最大值或最小值，返回变量数据（下方详解）**	**基因池 Gene Pool**		**存储一组指定区间和个数的随机数字，配合遗传计算器使用，计算最优解（下方详解）**

景观适合度 Fitness Landscape:		专门用来分析景观地形的运算器。高度，坡度，都可以分析，自动生成等高线，就是颜色比较难看不好改，所以我们还是习惯自己连，或者用插件。**（下方详解）**				
获取几何体 Get Geometry		**获取字符串 Get String**		**上下文打印 Context Print**		Grasshopper Player 运算器组，电池详细说明见下方。**（下方图解）**
获取小数 Get Number		**获取整数 Get Integer**		**上下文烘焙 Context Bake**		
获取点 Get Point		**获取直线 Get Line**				

电池组打包： 1、比如下方实例为根据指定的曲线，生成等间距栏杆。



2、将下方红色框内的运算器选中，单击鼠标中键（滚轮），在弹出的图标菜单中点击**打包**



3、图标右键可以编辑相关的属性，炸开电池，给电池加密；双击打包好的电池，可以进入电池内部编辑。

双击打包的电池组进入编辑界面，编辑完成后，右键点击下方图标选择保存。

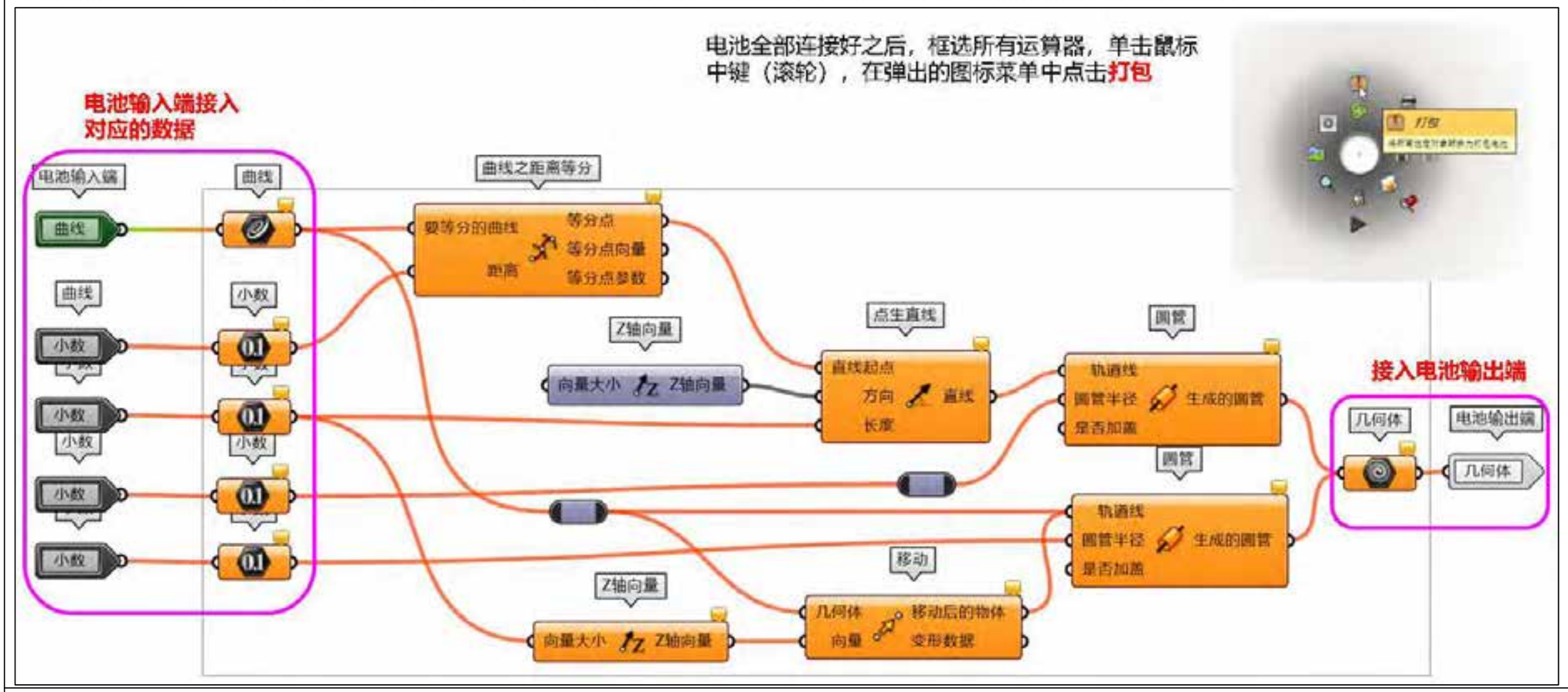
保存并关闭

取消保存并关闭

直接返回GH界面

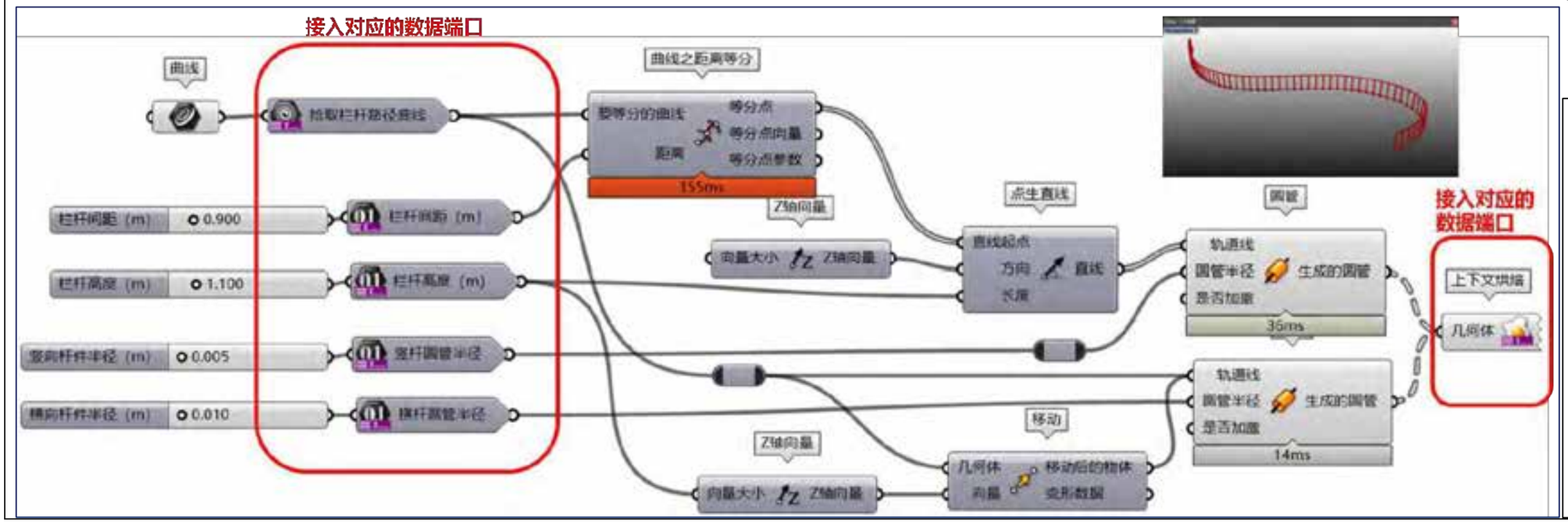
将打包好的电池添加至工具栏

4、打包的第二种方法是接入相应的电池输入端和电池输出端，然后全选打包。但显然这个比较麻烦，所以还是用前面的方法打包电池就好。



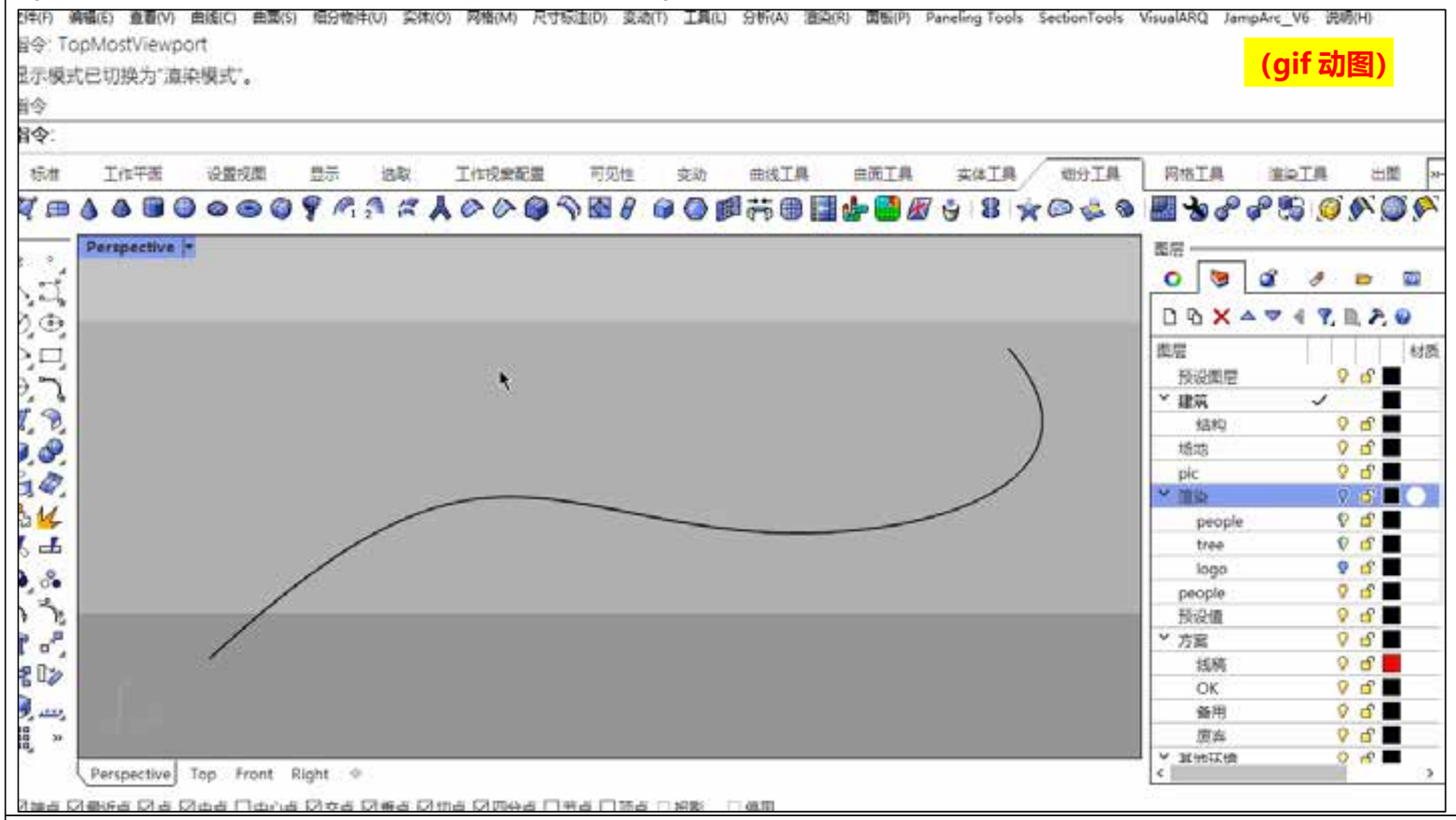
Grasshopper Player 运算器组

1、还是用前面的曲线生成栏杆实例。比如获取几何体，右键可以筛选拾取数据类型，以及拾取数量范围。保存好 GH 文件，比如下方 GH 文件命名为”等间距栏杆.gh”

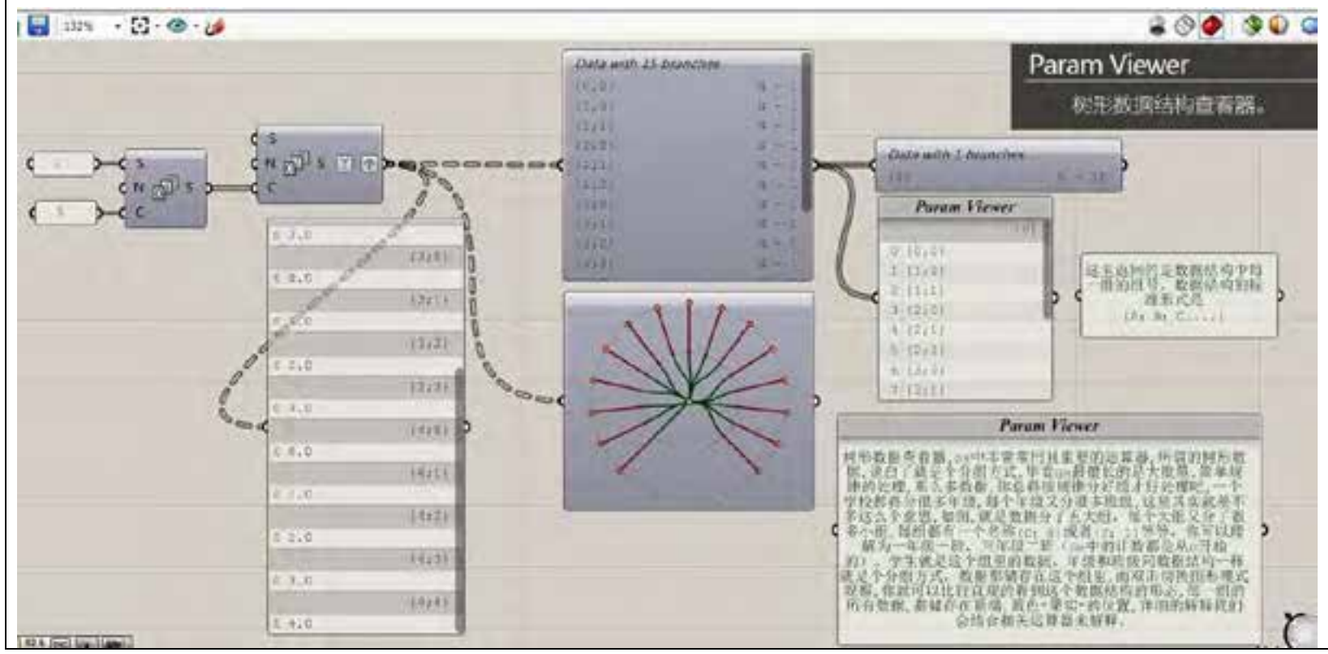


2、可以直接通过 rhino 中的 Grasshopper player 命令打开 GH 文件（等间距栏杆.gh），然后输入参数来生成目标物件，以便不会 gh 的人也可以大大降低使用门槛。（下方动图引自犀流堂）

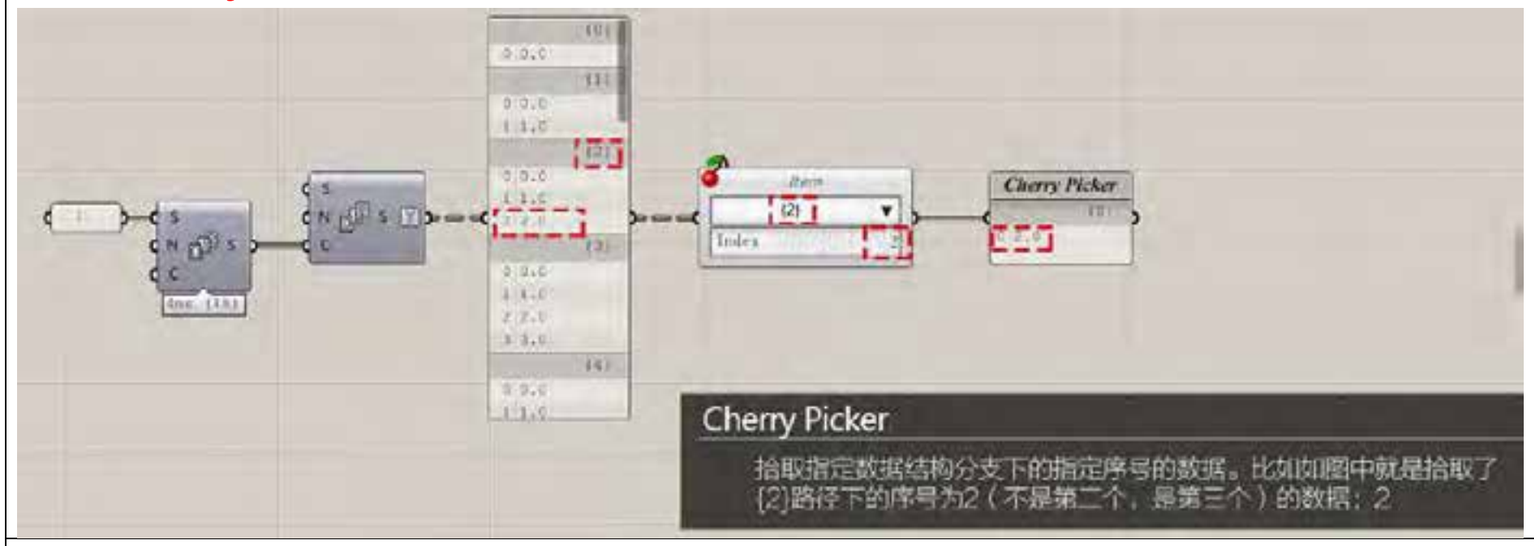
（这步我没有成功，输入命令就直接打开 GH 窗口了，待后续研究）



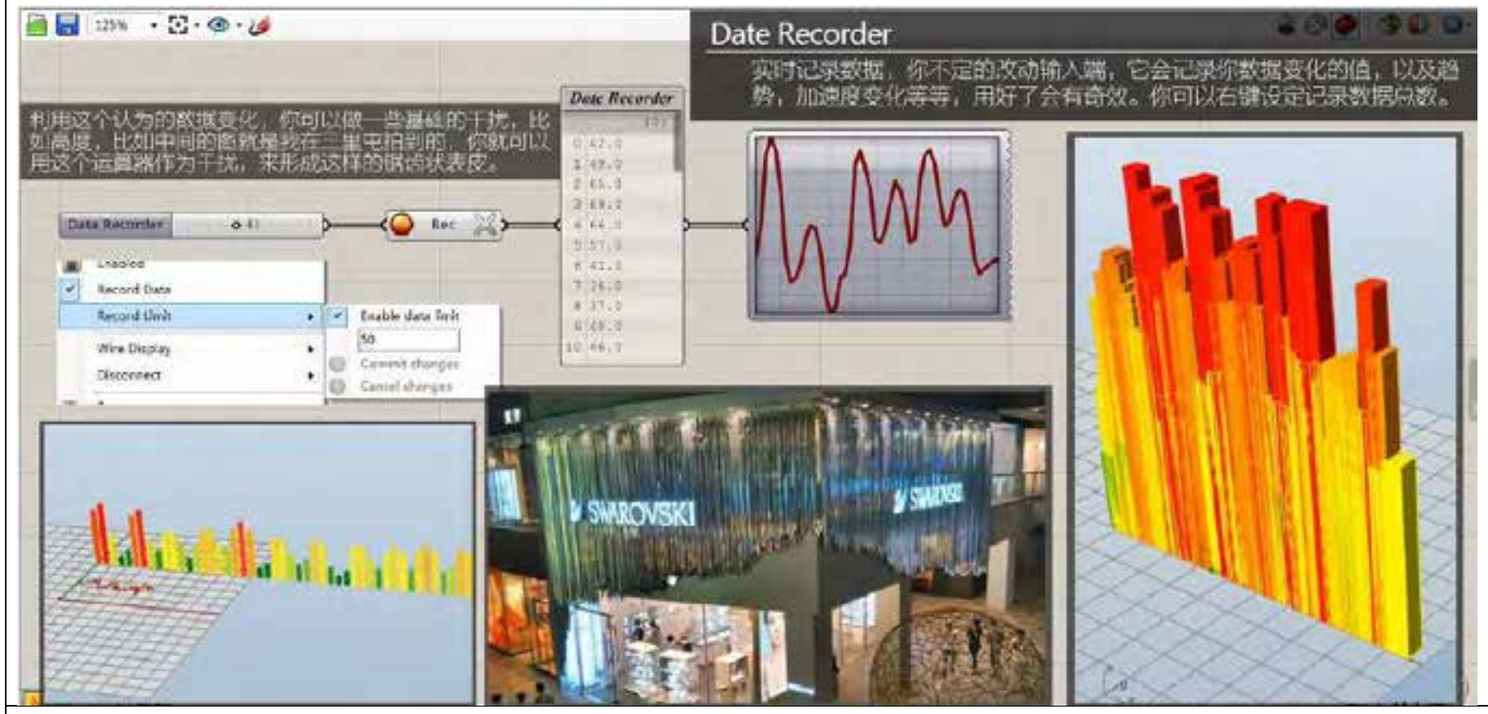
数据结构查看器 Param Viewer:

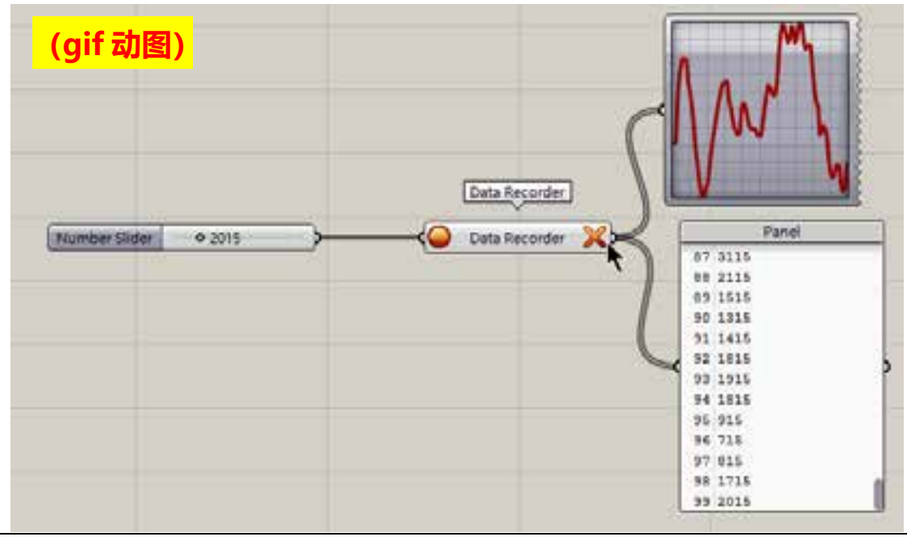


樱桃夹 Cherry Picker:



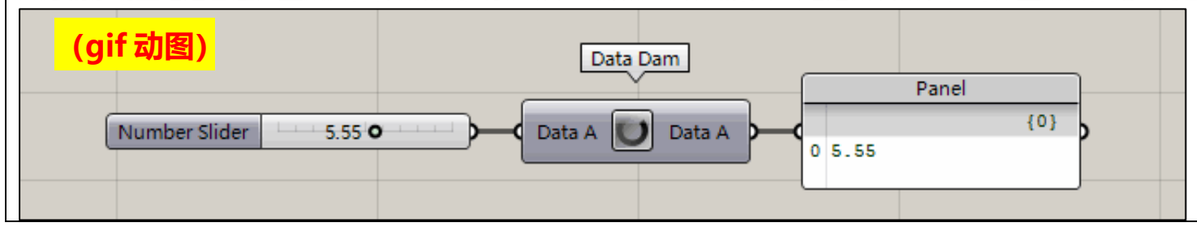
数据记录器 Data Recorder: 用来记录数据变化。左边的红球点击可以关闭或开启记录，右边的×可以删除现有记录。可以右键单击在菜单里修改记录数据的数量。

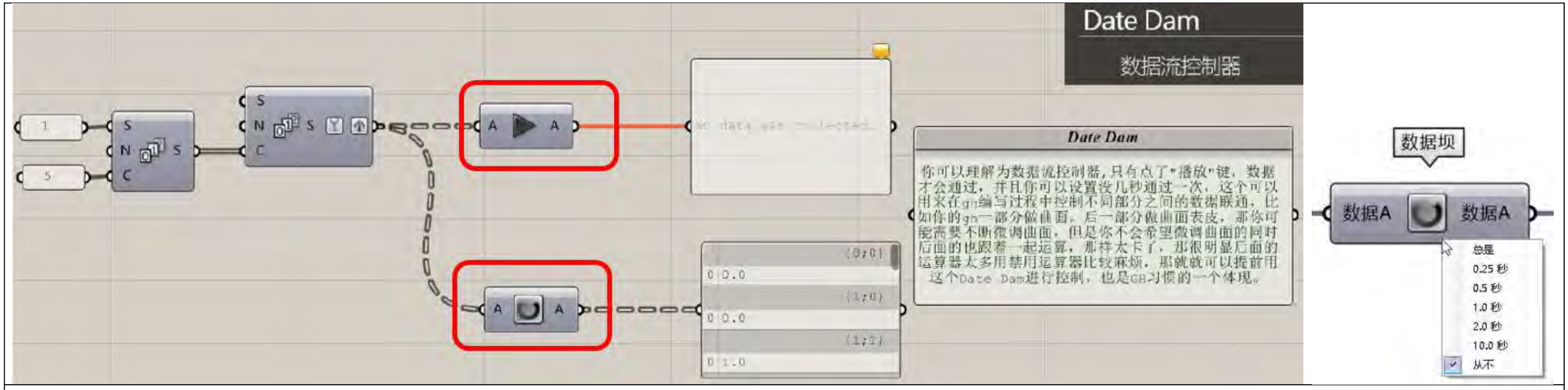




数据坝 Data Dam: 控制数据传输，一旦数据发生变化，就会停止传输，除非点击播放键；右键可以选择预设的时间间隔，控制数据传输间隔。

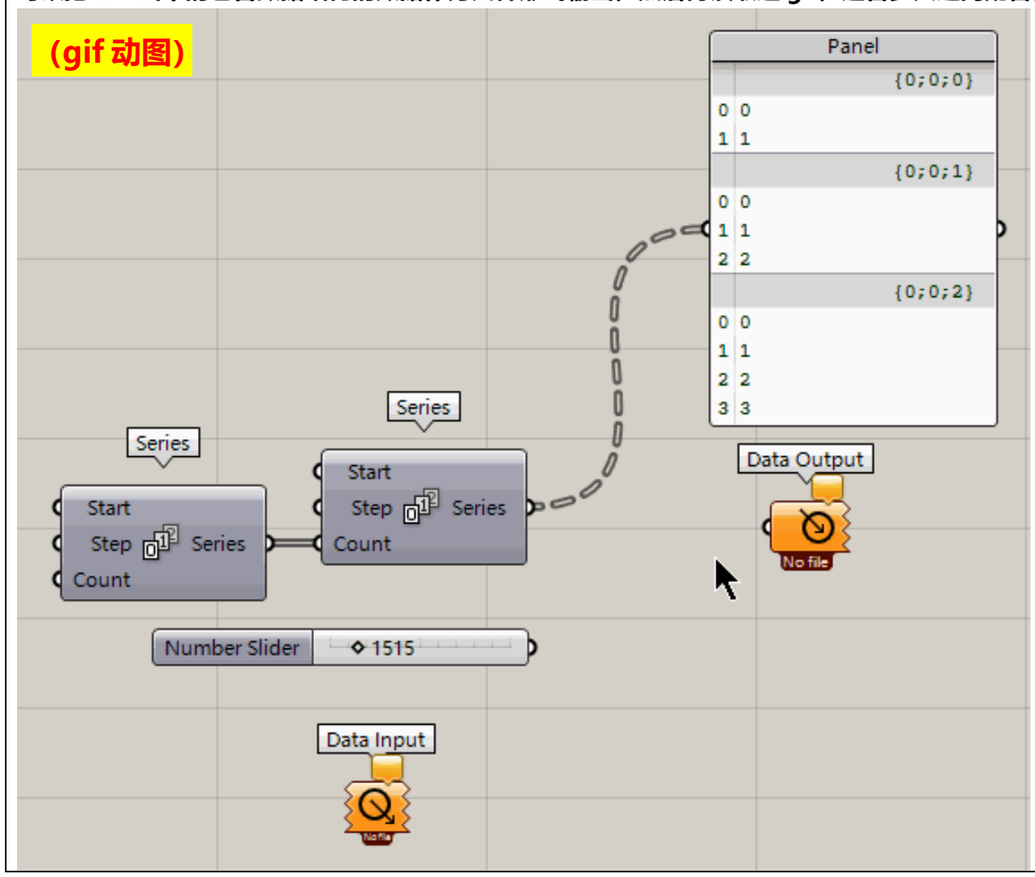
好处就是防止电池图运算量太大，修改数据导致的文件崩溃。所以如果你的电池太长太多，计算量太大，记得分区域设置几个 dam，只有点击播放键才能让数据通过，可以缓解计算压力。





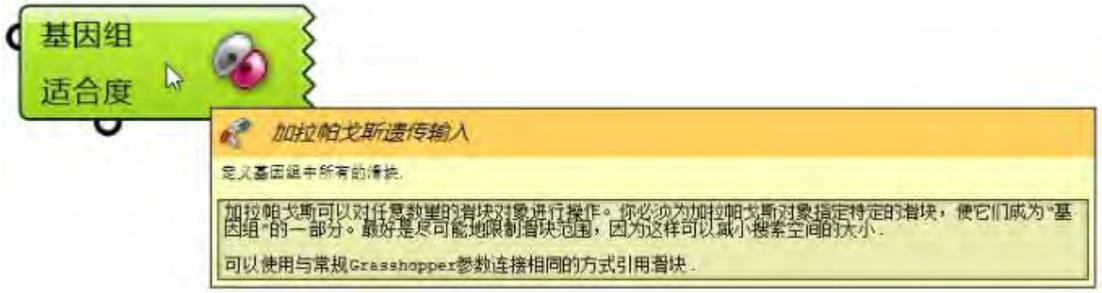
Data Input & Data Output 数据输入/数据输出

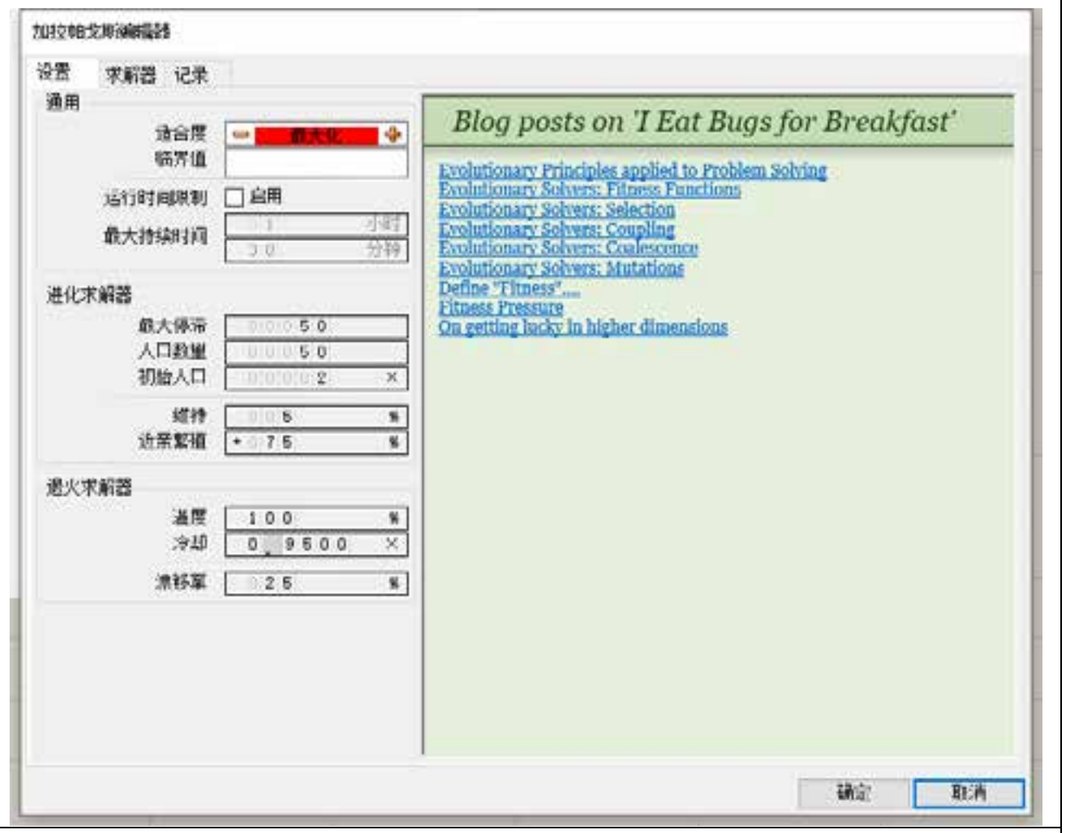
可以把 GH 当中的包含数据结构的数据作为文件形式输出，然后再读取进 gh，适合多人之间配合使用。比如把日照数据导出发给他人。（下方动图引用自犀流堂）



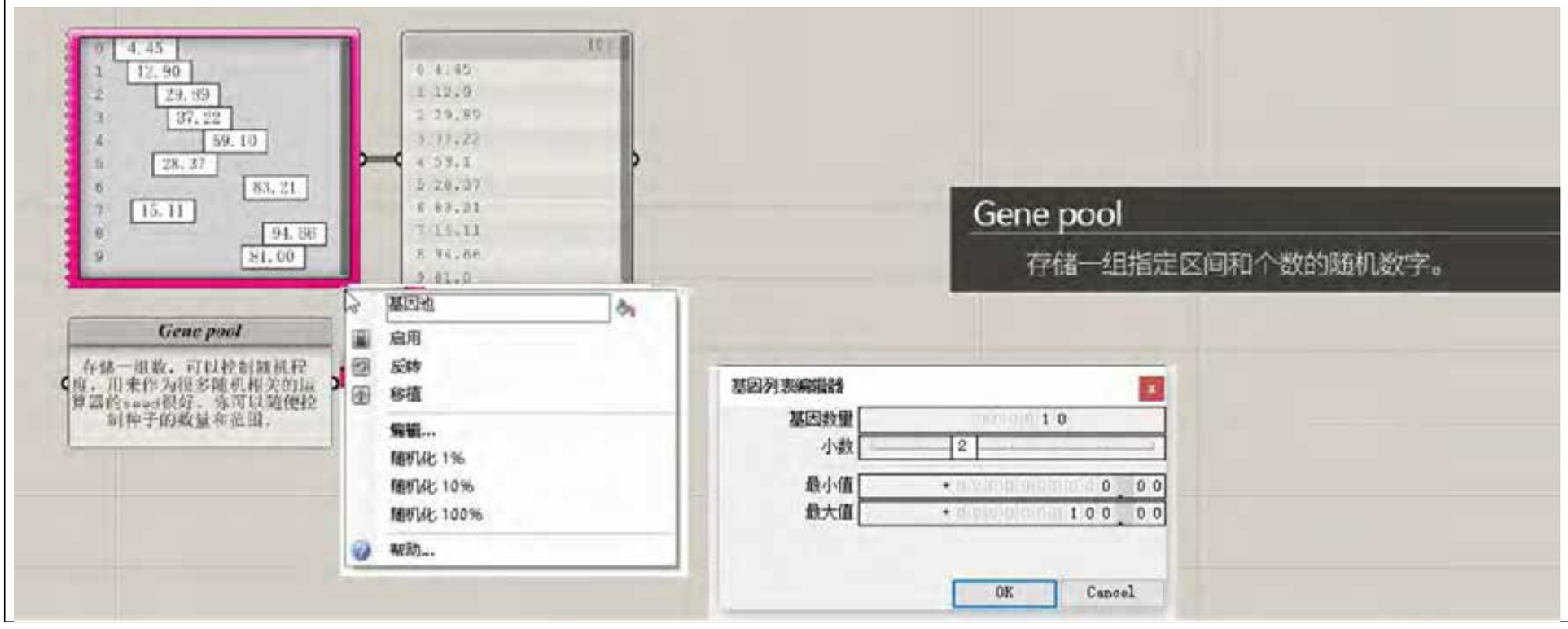
加拉帕戈斯 Galapagos（遗传算法）：

基因组（genome）端口用于接入变量，如 Slider（数字滑动条）、Gene Pool（基因池）运算器。其适合度（fitness）端用于接入所需求得最优解的变量，如面积，距离等等。



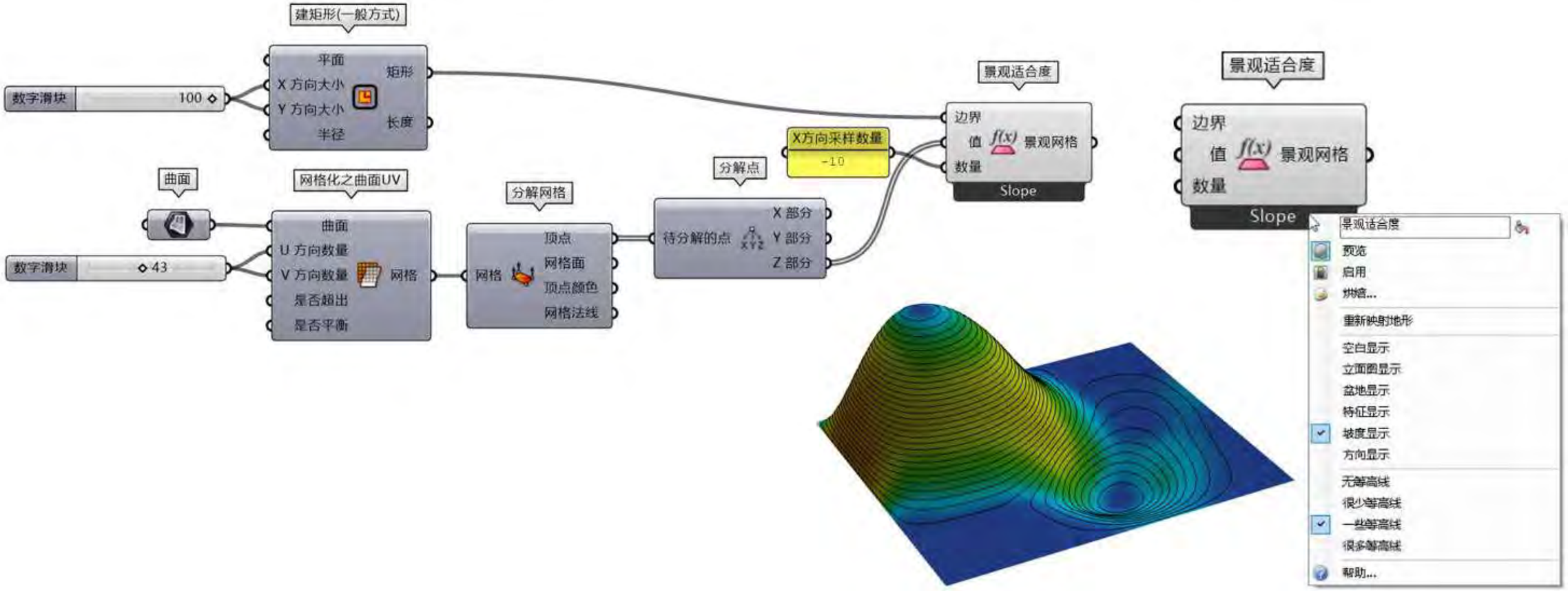


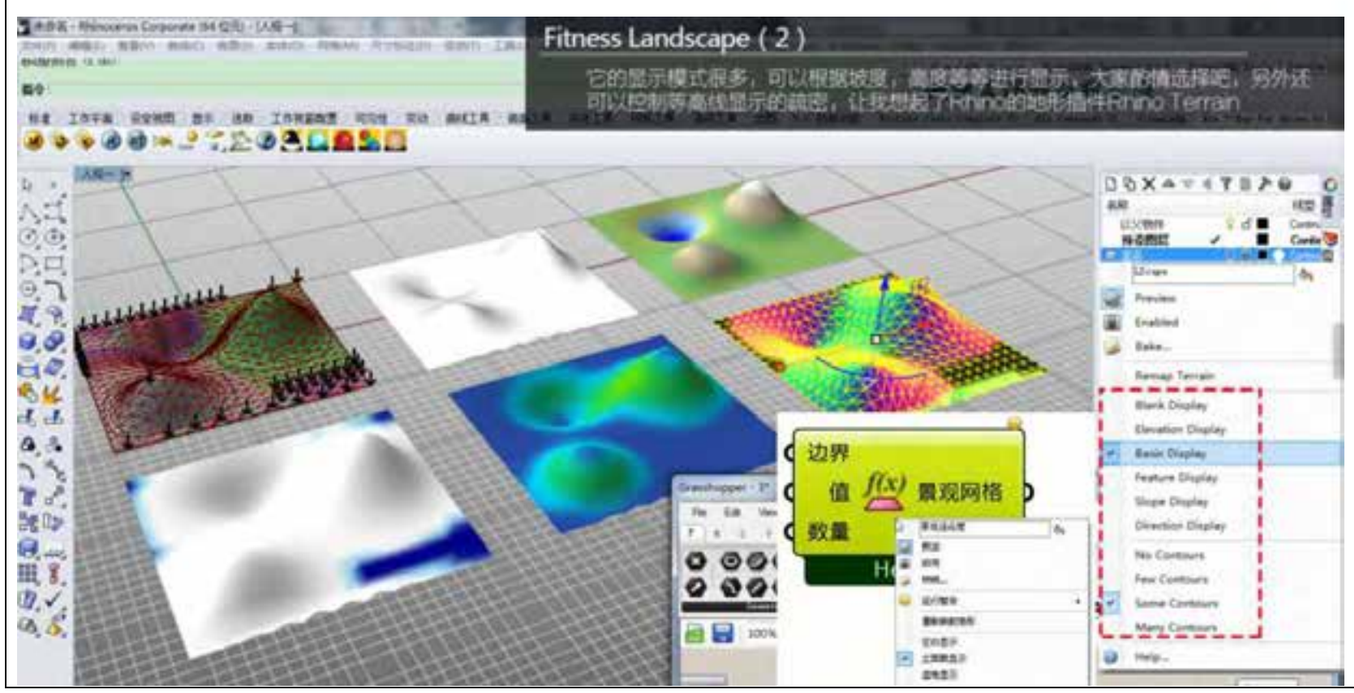
基因池 Gene Pool：



景观适合度 Fitness Landscape:

专门用来分析景观地形的运算器。高度，坡度，都可以分析，自动生成等高线，就是颜色比较难看不好改，所以我们还是习惯自己连，或者用插件



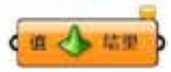


数学 Math

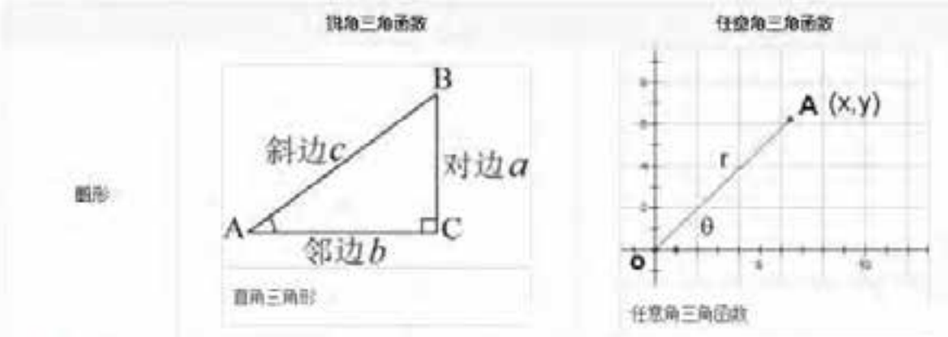
三角函数 Trig

数学 Math

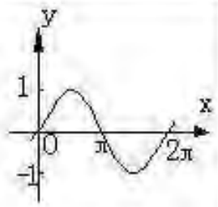
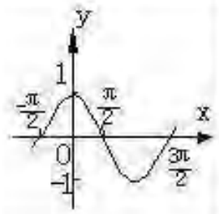
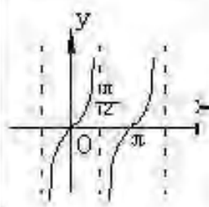
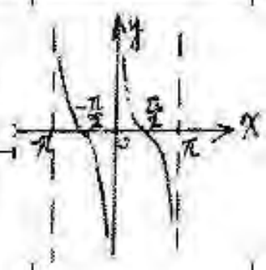
三角函数 Trig- (数学)

余弦 Cosine			正切 Tangent		
正弦 Sine			辛格函数 Sinc		归一化 sinc 函数 $Sinc(x)=\frac{\sin(\pi x)}{\pi x}$ (下方详解)
余切 CoTangent			余割 CoSecant		(下方详解)
反余弦 ArcCosine		(下方详解)	反正切 ArcTangent		(下方详解)
反正弦 ArcSine		(下方详解)	正割 Secant		(下方详解)
弧转角 Degrees		弧度单位 Pi=180 度	角转弧 Radians		弧度单位 Pi=180 度

定义式



正弦 (sin)	$\sin A=\frac{a}{c}$	$\sin\theta=\frac{y}{r}$
余弦 (cos)	$\cos A=\frac{b}{c}$	$\cos\theta=\frac{x}{r}$
正切 (tan或tg)	$\tan A=\frac{a}{b}$	$\tan\theta=\frac{y}{x}$
余切 (cot或ctg)	$\cot A=\frac{b}{a}$	$\cot\theta=\frac{x}{y}$
正割 (sec)	$\sec A=\frac{c}{b}$	$\sec\theta=\frac{r}{x}$
余割 (csc)	$\csc A=\frac{c}{a}$	$\csc\theta=\frac{r}{y}$

三角函数	$y=\sin x$	$y=\cos x$	$y=\mathrm{tg}x$	$y=\mathrm{ctg}x$
图　象				
定 义 域	$(-\infty,+\infty)$	$(-\infty,+\infty)$	$(n\pi-\frac{\pi}{2},\ n\pi+\frac{\pi}{2})$	$(n\pi,\ n\pi+\pi)$
值　域	$[-1,1]$	$[-1,1]$	$(-\infty,+\infty)$	$(-\infty,+\infty)$
最大(小)值 ($k\in Z$)	当 $x=2k\pi+\frac{\pi}{2}$ 时，$y_{max}=1$；	当 $x=2k\pi$时，$y_{max}=1$；当 $x=2k\pi+\pi$时，	无	无

辛格函数 Sinc：

一维sinc函数

函数定义

它们都是正弦函数和单调递减函数 1/x的乘积：

1.在数字信号处理和通信理论中，归一化sinc函数通常定义为：

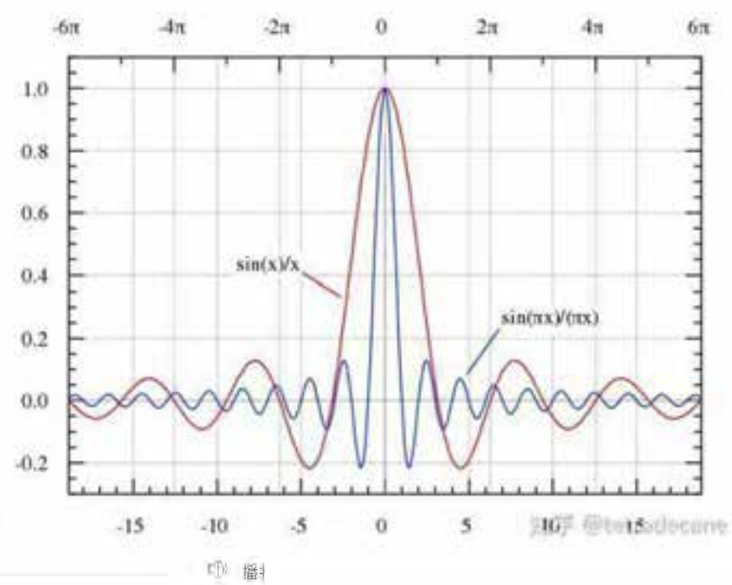
$$Sinc(x) = \frac{\sin(\pi x)}{\pi x}$$

sinc函数公式

2.在数学领域，非归一化sinc函数 (for sinus cardinalis)定义为：

$$Sinc(x) = \frac{\sin(x)}{x}$$

sinc函数公式



在这两种情况下，函数在 0 点的奇异点有时显式地定义为 1，sinc 函数处处可解析。非归一化 sinc 函数等同于归一化 sinc 函数，只是它的变量中没有放大系数 π。

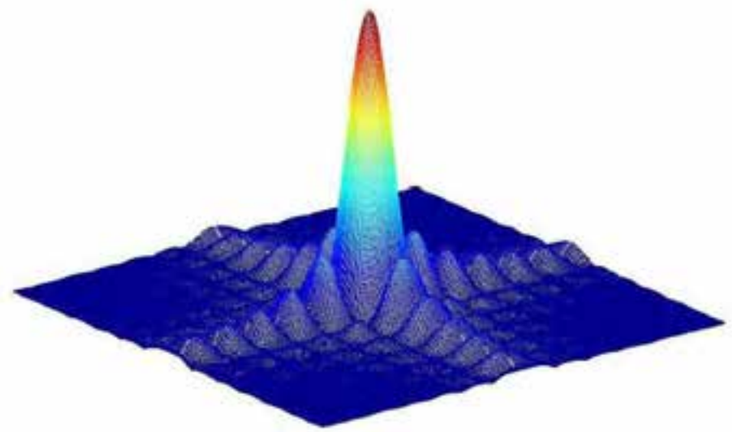
二维sinc函数

函数定义

二维的sinc()函数是一维sinc函数的扩展，其归一化定义为：

$$\sin c(x, y) = \frac{\sin(\pi x)}{\pi x} \times \frac{\sin(\pi y)}{\pi y}$$

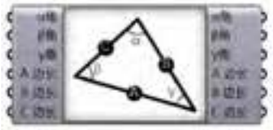
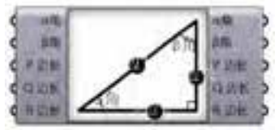
sinc



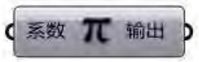
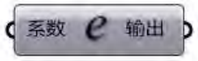
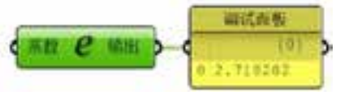
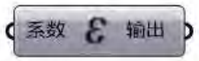
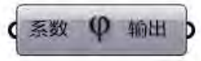
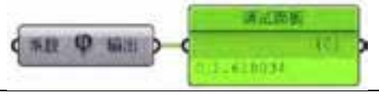
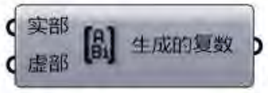
三角形 Triangle

公用 Util

三角形 Triangle- （数学）

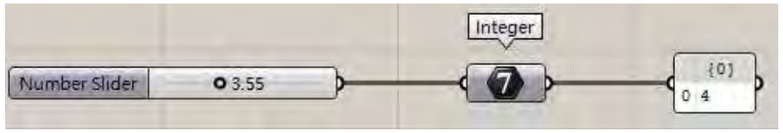
一般三角形		根据三角形的已知参数，求出三角形的其他参数	直角三角形		根据直角三角形的已知参数，求出三角形的其他参数
三角形内心		根据三个点构成的三角形，计算三角形内圆心（角平分线交点）	三角形垂心		根据三个点构成的三角形，计算三角形内垂心（顶点到对边垂直线的交点）
三角形外心		根据三个点构成的三角形，计算三角形外心（各边中点垂线的交点）	三角形重心		根据三个点构成的三角形，计算三角形重心（顶点到对边中点连线的交点）

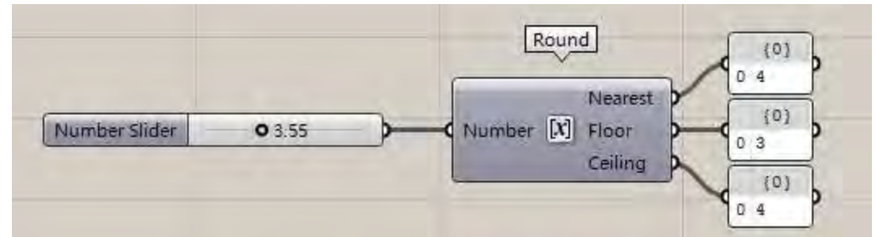
公用 Util- （数学）

派 Pi		圆周率，圆的周长与直径的比值	自然对数 Natural Logarithm		
艾普斯龙 Epsilon		一个无限趋近于 0 的数，却不等于 0（符号为希腊字母表的第 5 个字母）	黄金比例 Golden Ratio		
四舍五入 Round		四舍五入，输出端 floor 是不大于这个数值的最大值，Ceiling 是不小于这个数值的最小值（下方详解）	最大值 Maximum		对比两组或者多组数据，求对应数据最大值（下方详解）
最小值 Minimum		对比两组或者多组数据，求对应数据最小值（下方详解）	极值 Extremes		对比两组或者多组数据，求对应数据的极小值和极大值（下方详解）
加权平均 Weighted Average		（下方详解）	平滑数字 Smooth Number		柔滑数据过度，两个数字在设定时间内平滑变动，连续输出不断变动的数字（下方详解）
截断 Truncate		按设定 0-1 之间的数字（换算成百分比）删掉首尾的数值，只留下中间部分（下方详解）	插入数值 Interpolate data		自动补齐给定数列之间的数据，然后根据输入的参数（0-1），输出对应的数据。（下方详解）
模糊数字 Blur Numbers		给定数列根据相邻数据平均值来进行模糊处理，基本局势是小的变大，大的变小（下方详解）	算术平均 Average		计算数列的平均值，也可以用于计算三点构成的三角形的重心（顶点到对边中点连线的交点）
共轭复数 Complex Conjugate		对于复数 z=a+bi ，称复数 $\bar{z}$ =a-bi 为 z 的共轭复数。即两个实部相等，虚部互为相反数的复数互为共轭复数	创建复数		z=a+bi（a、b 均为实数）的数称为复数。其中，a 称为实部，b 称为虚部，i 称为虚数单位。
复数参数 Complex Argument		（下方详解）	复数的模		（下方详解）
提取复数		提取复数 z=a+bi 的实部 a 和虚部 b 数值			

四舍五入 Round：

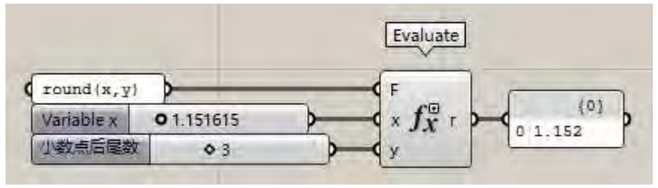
四舍五入，输出端 floor 是不大于这个数值的最大值，Ceiling 是不小于这个数值的最小值，用地板和天花板两个单词来表示还是比较形象的，不过呢，如果你单纯的想做四舍五入的话，直接接给 Integr 整数即可。

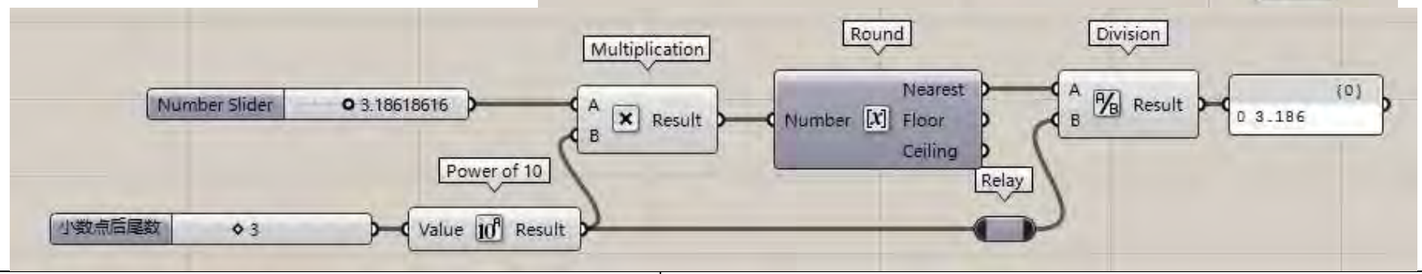




不过我们比较常见的用法是用来控制小数点后尾数：

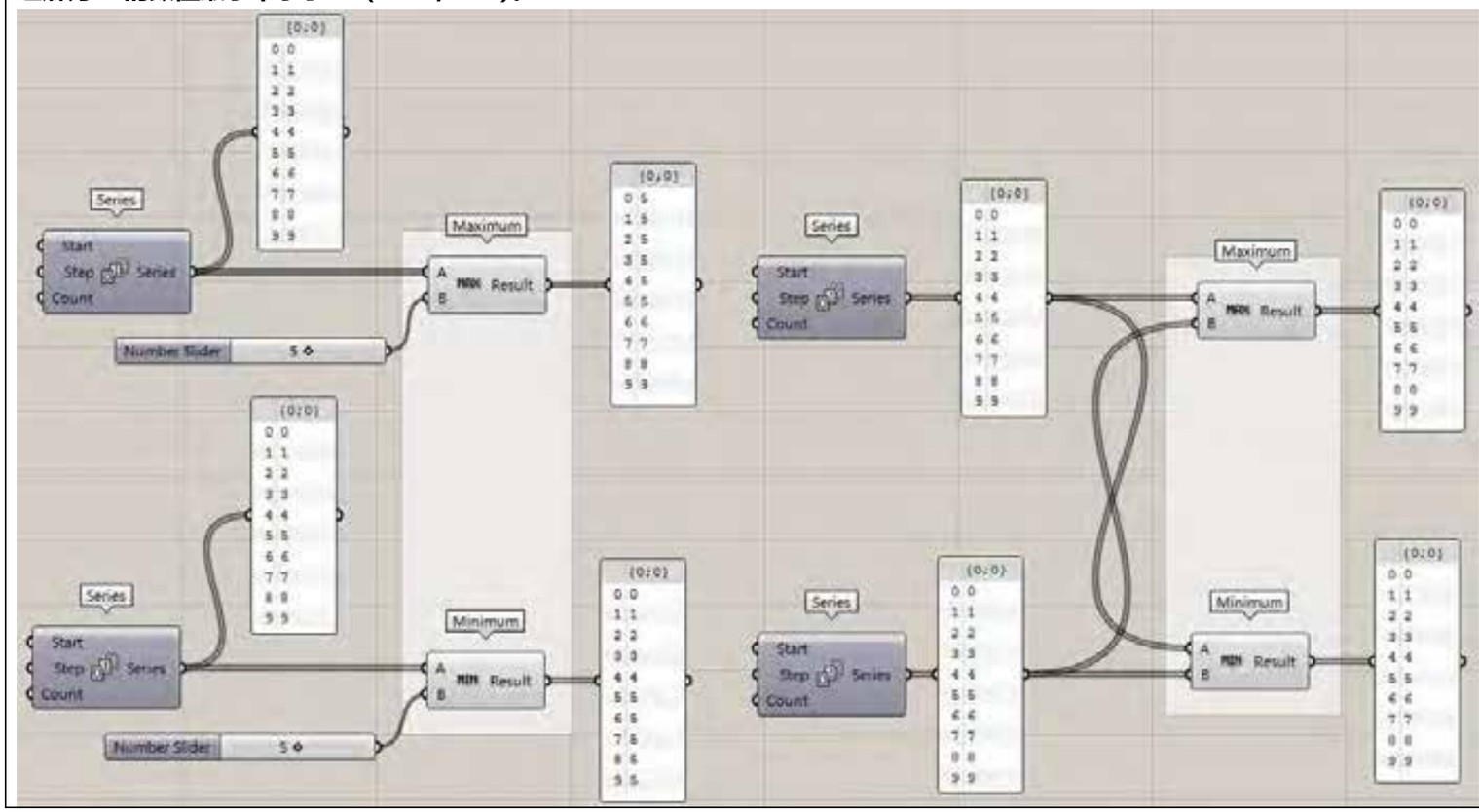
当然，我们在前面讲表达式的时候讲过一个更简单的用法：





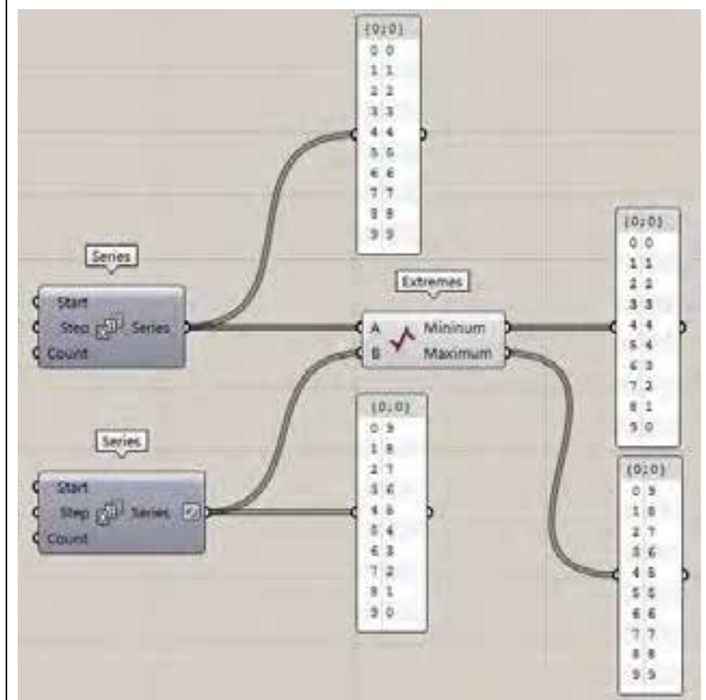
最小值和最大值 Maximum & Minimum：

取大值 和 取小值，对比 AB 两个输入端，然后返回较大或较小值。比较老的教程里经常用来在干扰的时候控制数值区间。比如 max 就可以理解为 A 的数值最小不小于 B（if AB,A=B）。



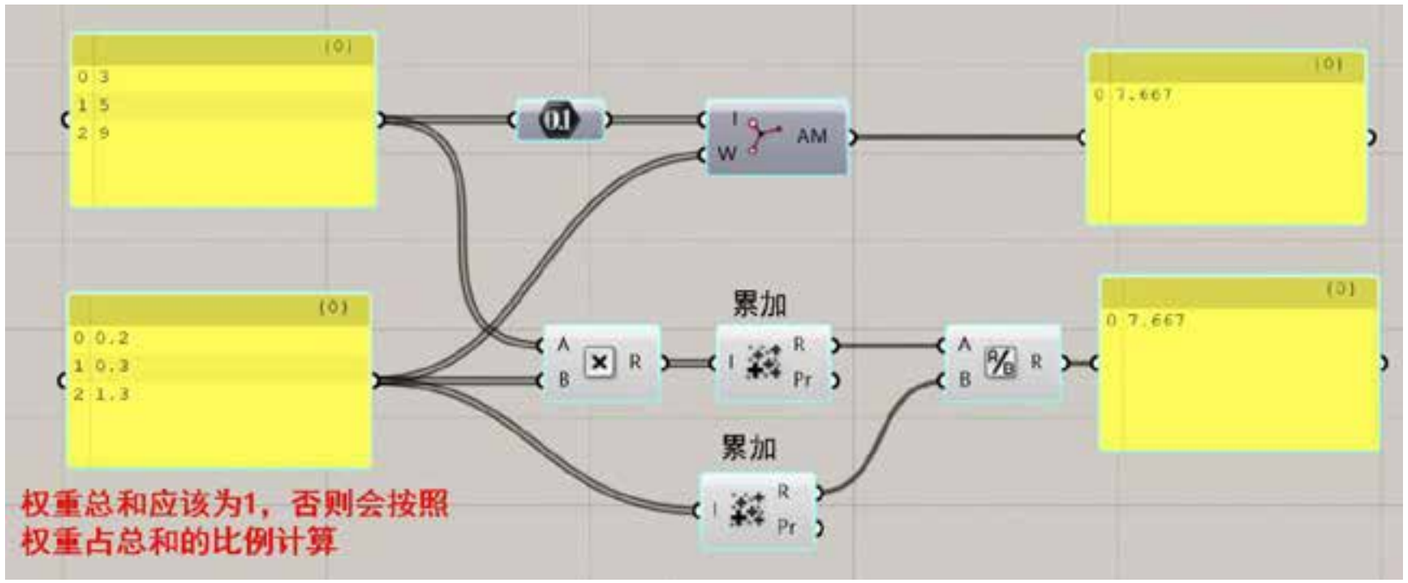
极值 Extremes：

对比两组或者多组数据，求对应数据的极小值和极大值



加权平均 Weighted Average:

加权平均数的算法应该是原数列的数值乘以所占比重（权重），并不需要除数列的长度 ，但是要求比重的总和必须为 1，如果不是 1 ，就会自动将其映射到 0-1 区间（即权重占权重累加数值的比例），等同与将其结果除比重的总和



权重未知（隐含）

现有以下两只股票：

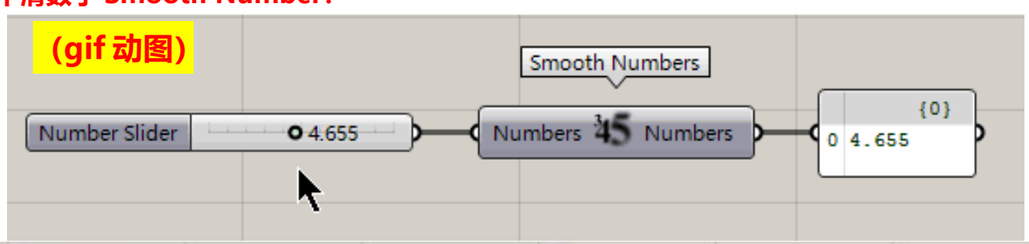
股票名	股数	价格
股票A	1000	10
股票B	2000	15

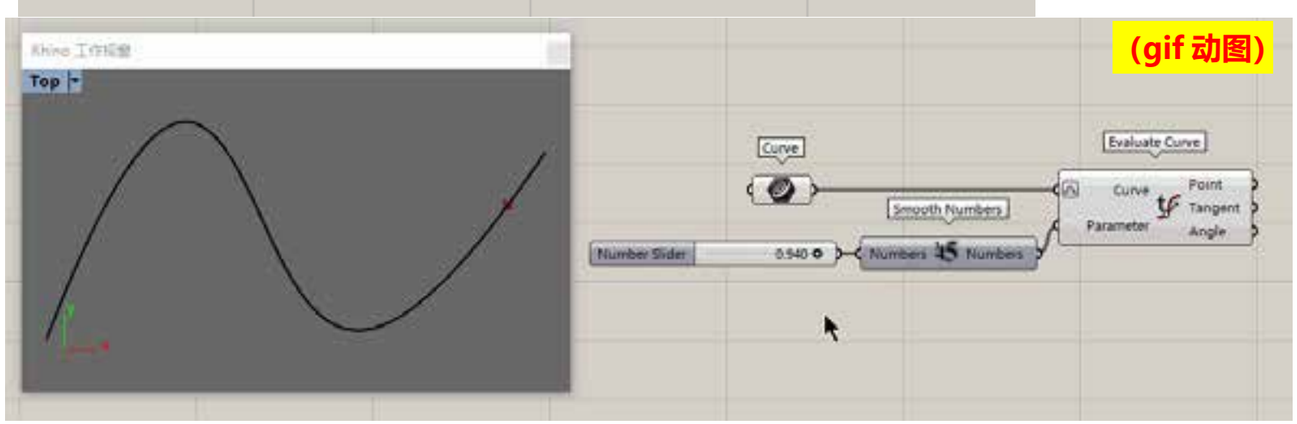
（注：在这里，股票占总股数的比重叫做权重）

那么，加权平均值（所有拥有股票的平均价格）

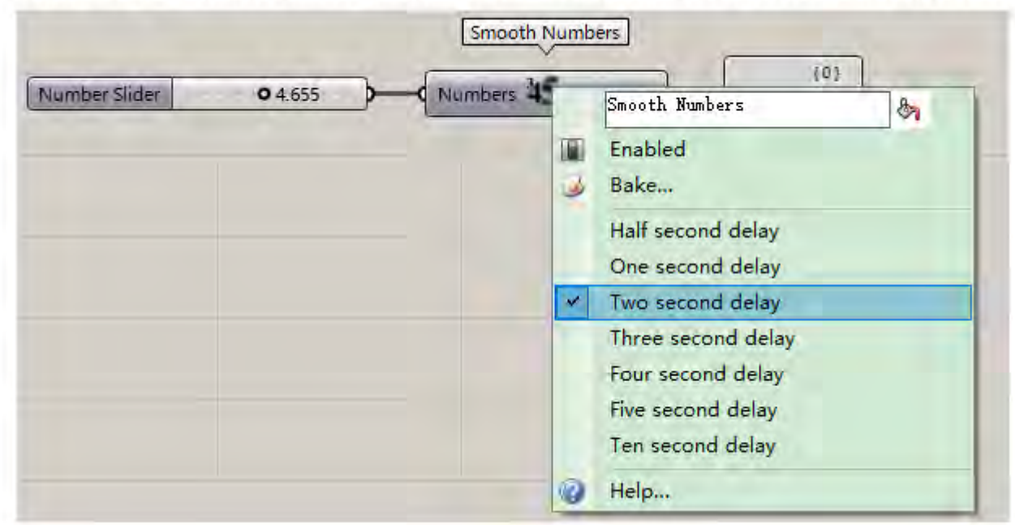
$$= \frac{10 \times 1000 + 15 \times 2000}{1000 + 2000} \approx 13.33$$

平滑数字 Smooth Number:

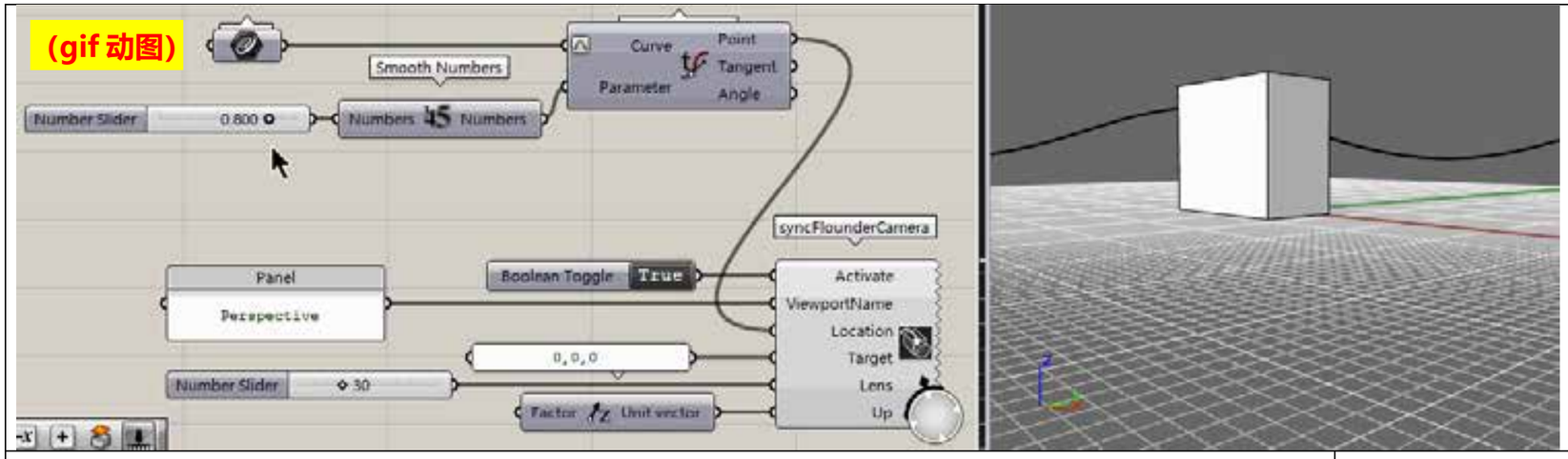




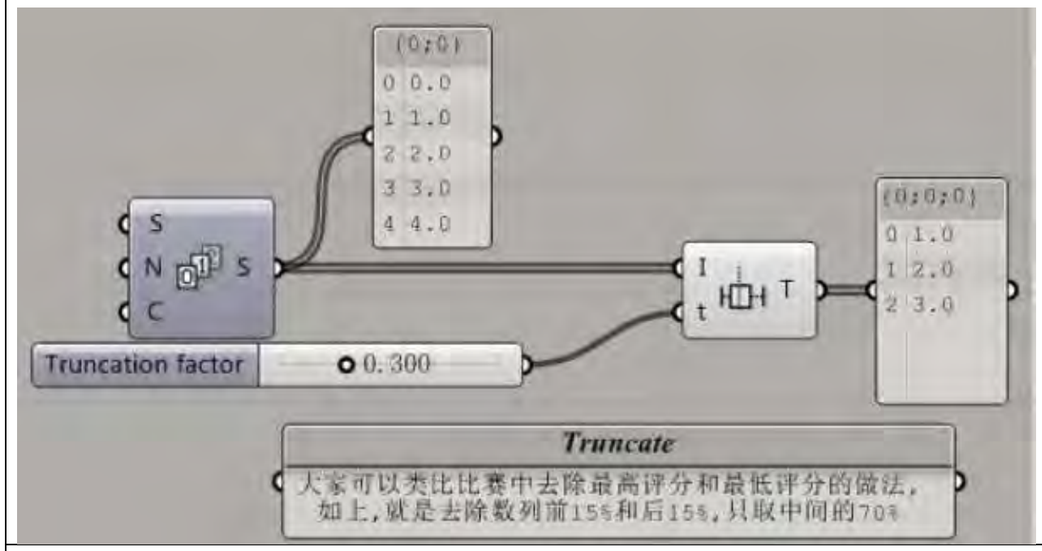
柔滑数据过度，当你修改数值时，这个运算器会在你指定的时间内将原有数值变化到你指定的数值。



再用个摄像机控制的例子来演示下，你会发现我们来回切换数值的时候，摄像机过度的很顺化，没有突变的感觉，有点云台的感觉。

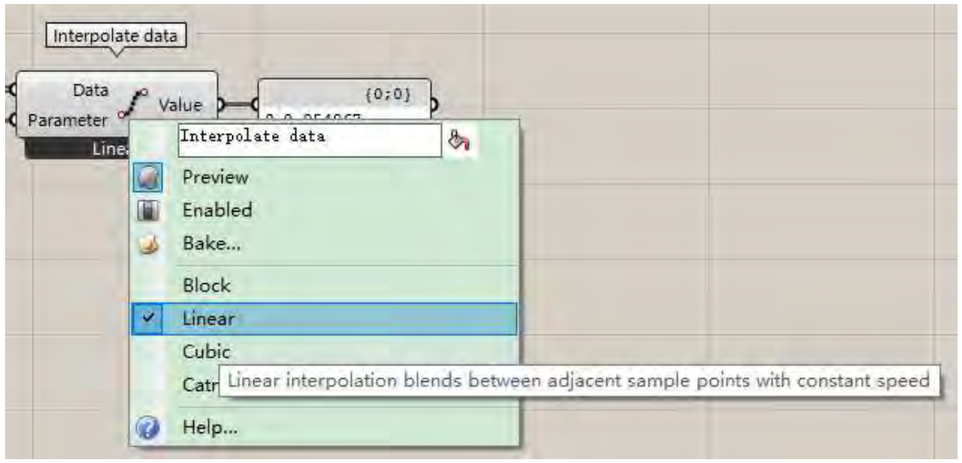


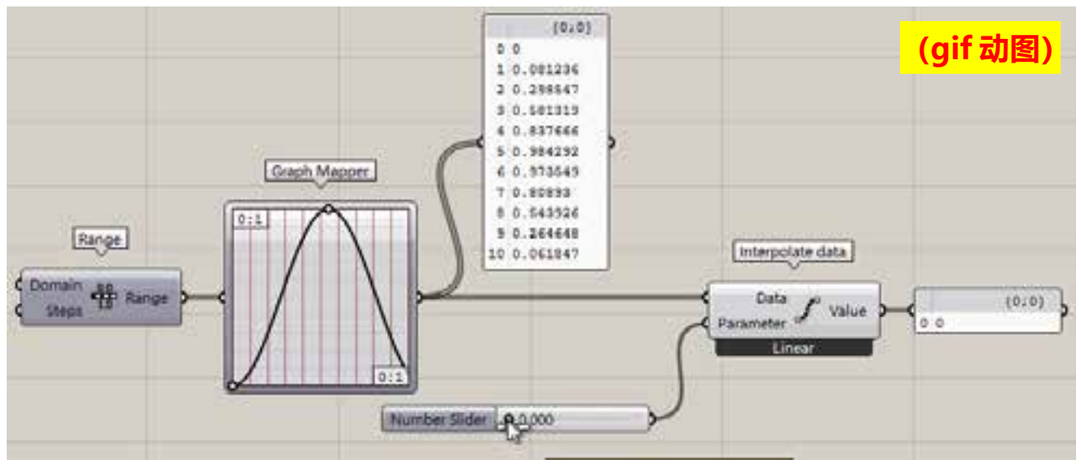
截断 Truncate：



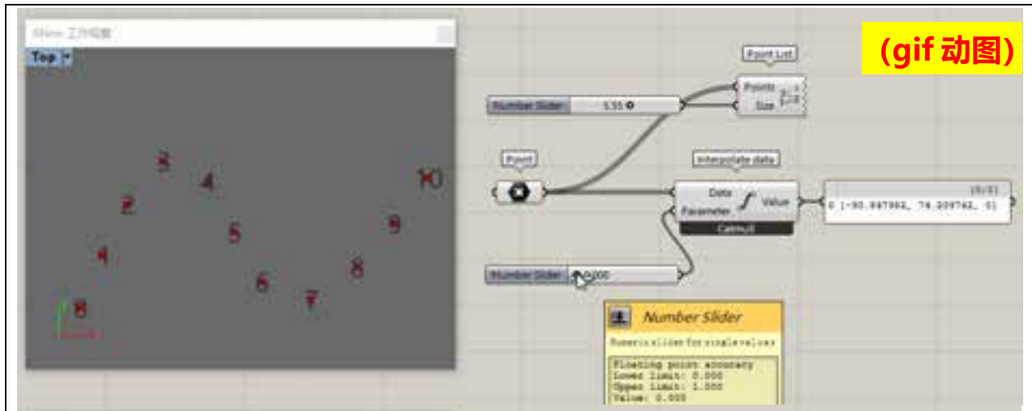
插入数值 Interpolate data： 自动补齐给定数列之间的数据，然后根据 Parameter（0-1）输入的数值，输出对应的数据。

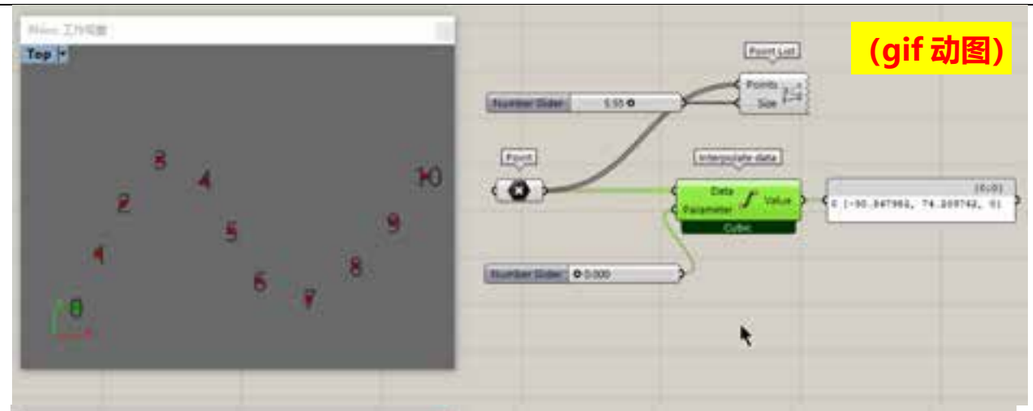
在多个数值变化对应的曲线上，在任意位置插入一个数值（可以理解成在多点生成的曲线上插入一个点）

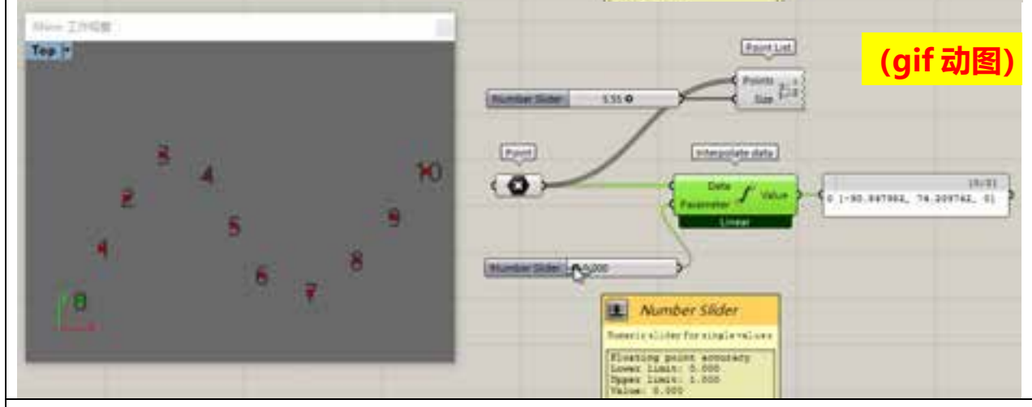


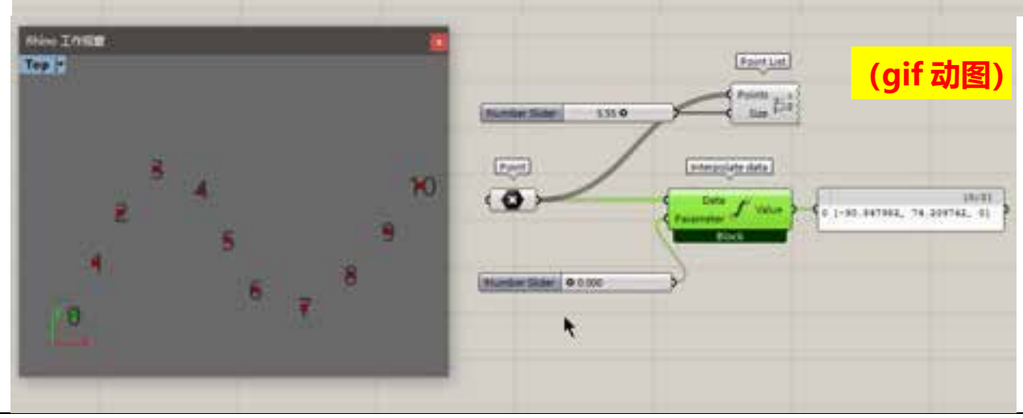


右键可以设置不同的类型：block 卡顿、阻塞，linear 线性，Cubic 三次曲线，Catmull 平滑（细分曲面）







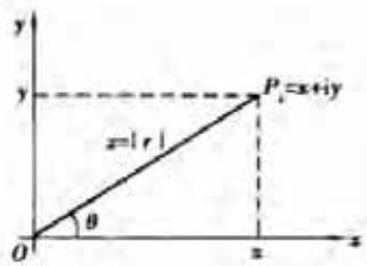


复数 Complex:

形如 z=a+bi（a、b 均为实数）的数称为复数。其中，a 称为实部，b 称为虚部，i 称为虚数单位。当 z 的虚部 b＝0 时，则 z 为实数；当 z 的虚部 b≠0 时，实部 a＝0 时，常称 z 为纯虚数。

对于复数 $z=a+bi$ ，称复数 $\bar{z}$ =a-bi为z的共轭复数。即两个实部相等，虚部互为相反数的复数互为共轭复数(conjugate complex number）。复数z的共轭复数记作 $\bar{z}$ 。

由于一个复数 $z=x+iy$ 可以由有序实数对 (x,y) 唯一确定，而有序实数对与平面直角坐标系 xOy 中的点一一对应，因此可以用坐标为 (x,y) 的点 P 来表示该复数，此时 x 轴上的点与实数对应，称 x 轴为实轴，y 轴上的点（除原点外）与纯虚数对应，称 y 轴为虚轴，像这样表示复数的平面称为复平面。



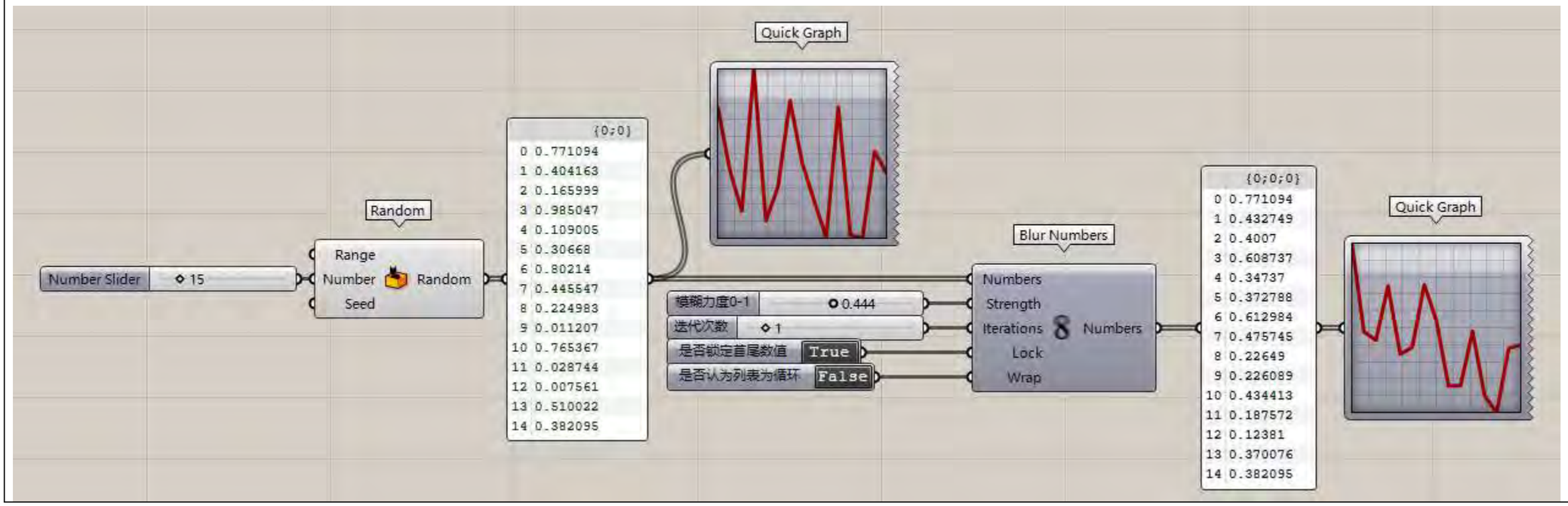
复数 $z=x+iy$ 还可以用向量 $\overrightarrow{OP}$ 来表示，x 与 y 分别是向量 $\overrightarrow{OP}$ 在 x 轴与 y 轴上的投影。这样，复数 z 就与平面上的向量 $\overrightarrow{OP}$ 建立了一一对应的关系。

$z=x+iy$ 向量 $\overrightarrow{OP}$ 的长度称为复数的模或绝对值，记作 $|z|$ ，于是

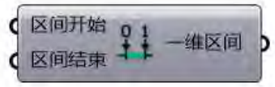
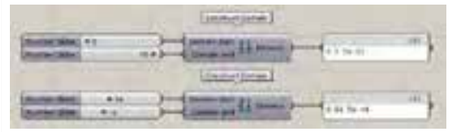
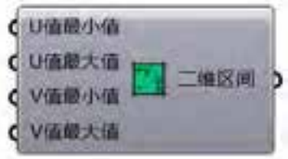
$$|z|=\sqrt{x^2+y^2}$$

当点 P 不是原点，即复数 $z\neq 0$ 时，向量 $\overrightarrow{OP}$ 与 x 轴正向的夹角称为复数 z 的辐角，记作 $Argz$ 。辐角的符号规定为：由正实轴依反时针方向转到 $\overrightarrow{OP}$ 为正，依顺时针方向转到 $\overrightarrow{OP}$ 为负。 [1]

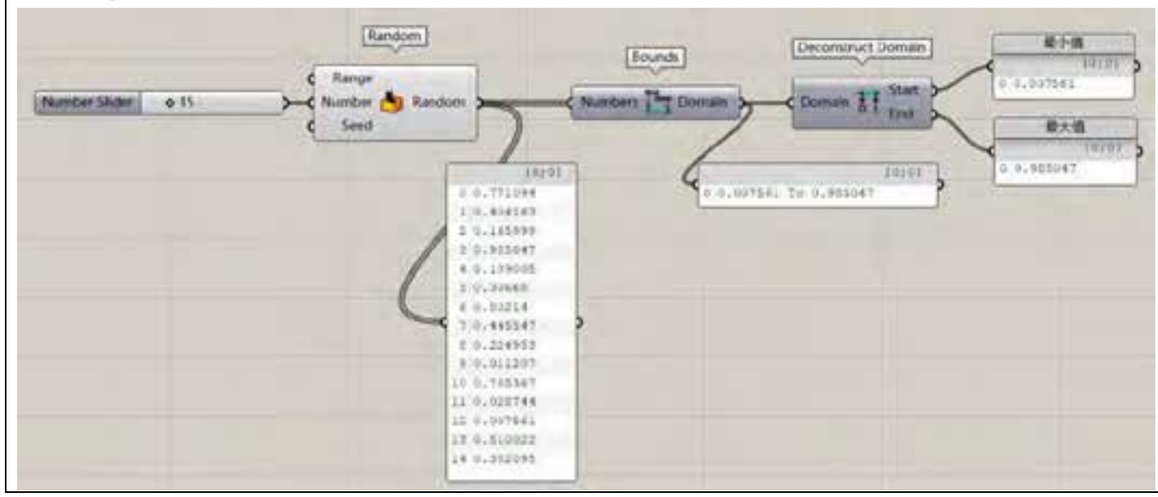
模糊数字 Blur Numbers：给定数列进行柔化处理，大的变小，小的变大，类似 SU 的柔化边线



区间 Domain-（数学）

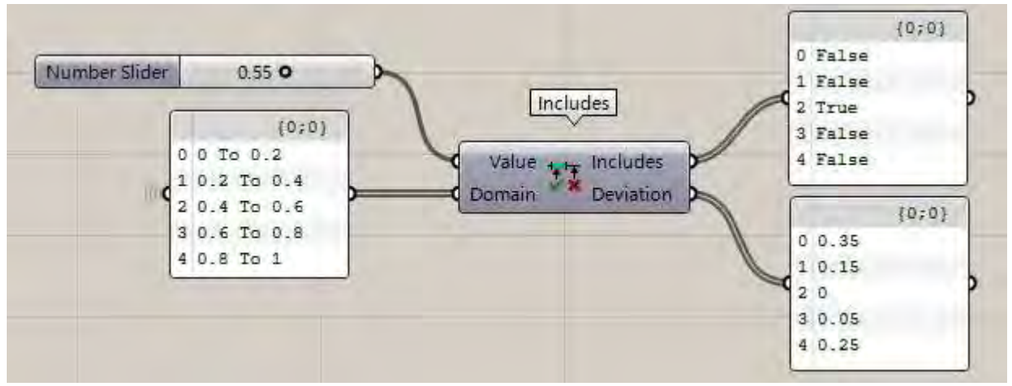
分解一维区间 Deconstruct Domain		拆分得到区间的大小值。	**构造一维区间 Construct Domain**		用两个值组成一个区间。 
一维边界 Bounds		获取一组数的区间范围，配合分解一维区间可以很容易的得到一组数的最大最小值。（下方详解）	**区间判断 Includes**		判断一个数值，是否在指定区间内，返回布尔值以及数值与区间的差值，也就是如果区间不在，那着这个数值距离和区间极值的差值（下方详解）
区间映射 Remap Numbers		将原区间的数值按照目标区间的范围等比缩放（下方详解）	**查找区间 Find Domain**		查找数值所在的区间并返回区间对应的序号。
等分一维区间 Divide Domain		按设定数字等分一维区间，输出多个等分区间	**连续区间 Consecutive Domains**		可以把一组数列组合成多个区间，Additive 端通过布尔值决定数值是否要依次相加。（下方详解）
分解二维区间到区间 Deconstruct Domain²		拆解二维区间成两个一维区间	**分解二维区间到数值 Deconstruct Domain²**		拆解二维区间成四个极值
构造二维区间到区间 Construct Domian²		根据两个一维区间构件二维区间	**构造二维区间到数值 Construct Domian²**		根据四个值构建二维区间
二维边界 Bounds 2D		点集所在的二维区间	**等分二维区间 Divide Domain²**		根据 UV 方向设定的等分数等分二维区间

一维边界 Bounds：获取一组数的区间范围，配合分解一维区间可以很容易的得到一组数的最大最小值。



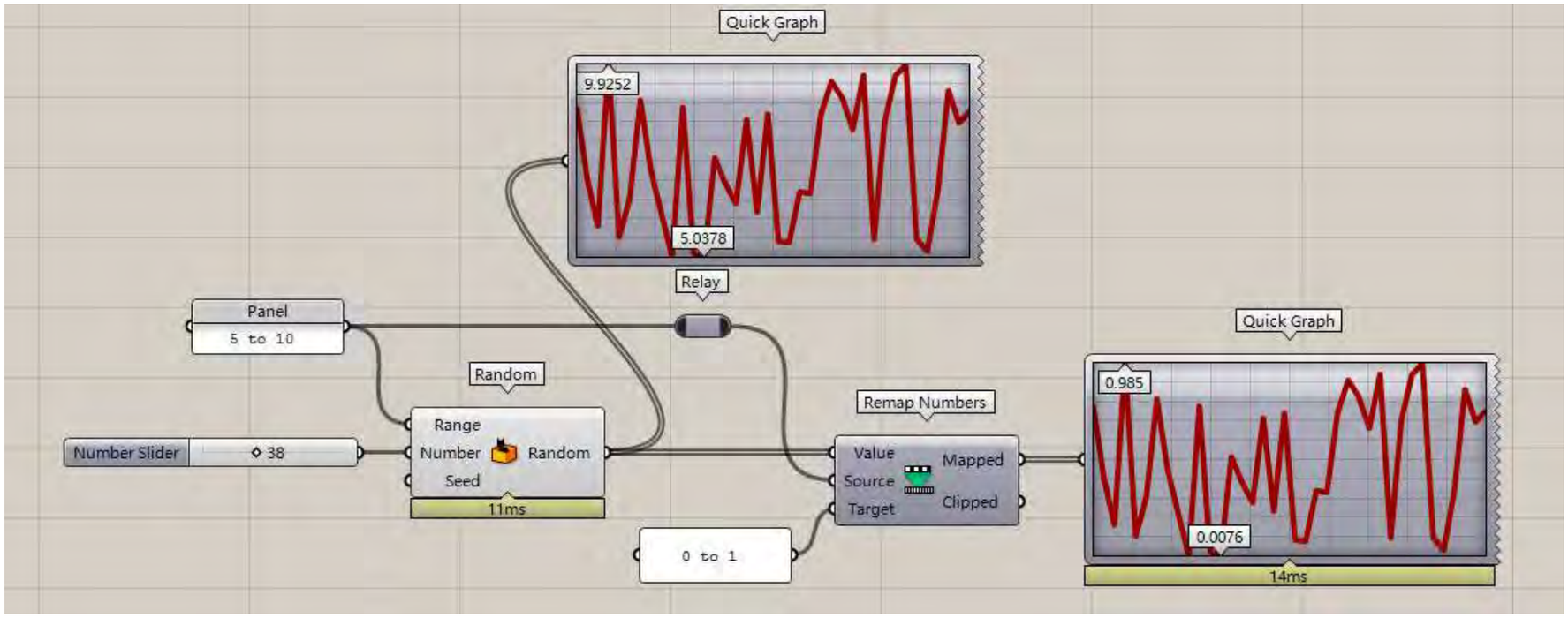
区间判断 Includes：

判断一个数值，是否在指定区间内，返回布尔值以及数值与区间的差值，也就是如果区间不在，那着这个数值距离和区间极值的差值

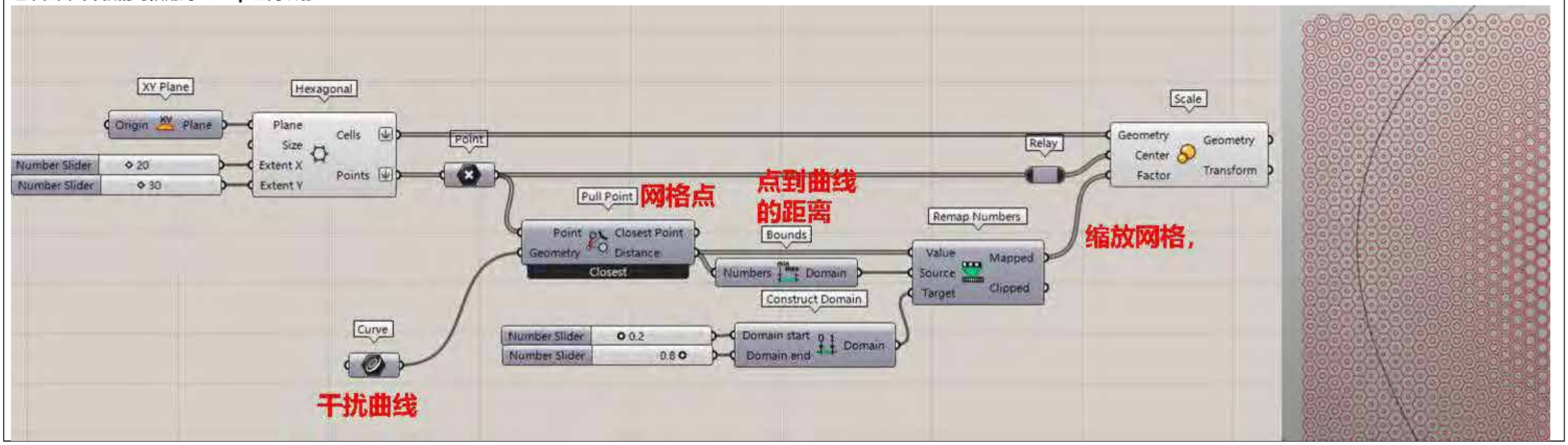


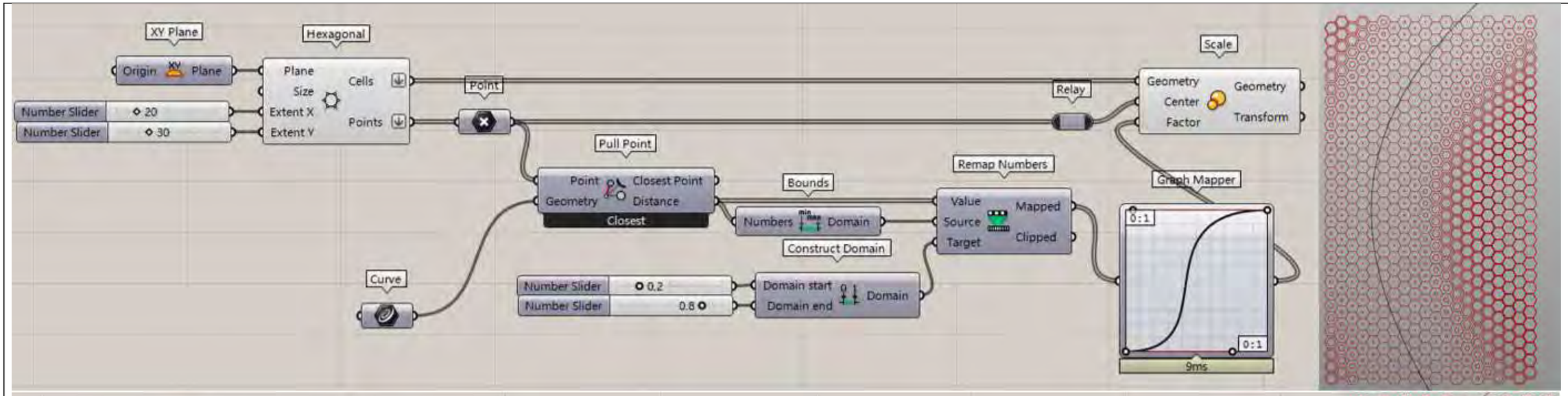
区间映射 Remap Numbers:

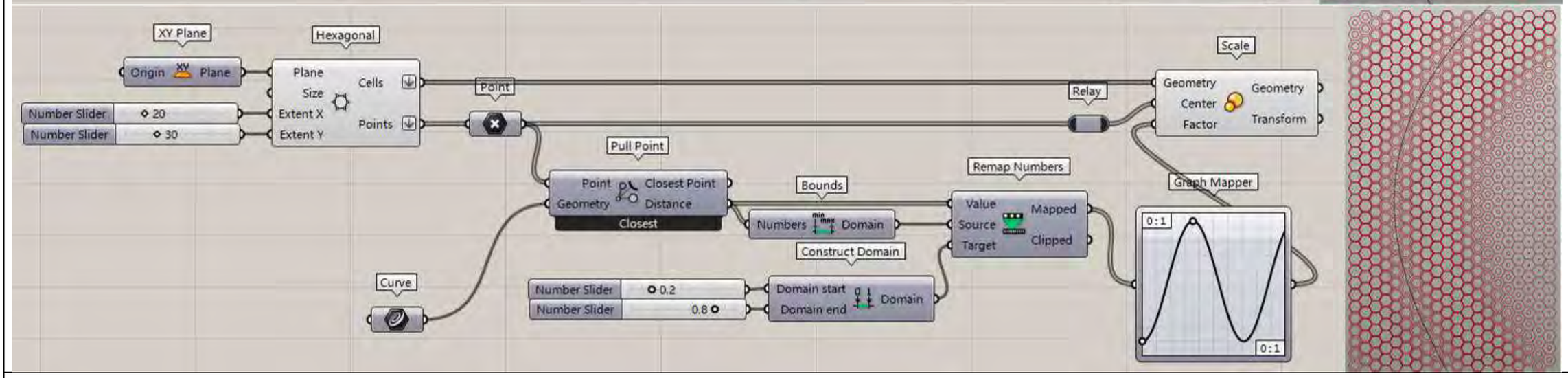
非常重要的一个运算器，经常被我们用来映射区间，映射听着你可能有点迷糊，虽然不太准确，但是说缩放估计你就能比较好的理解了。



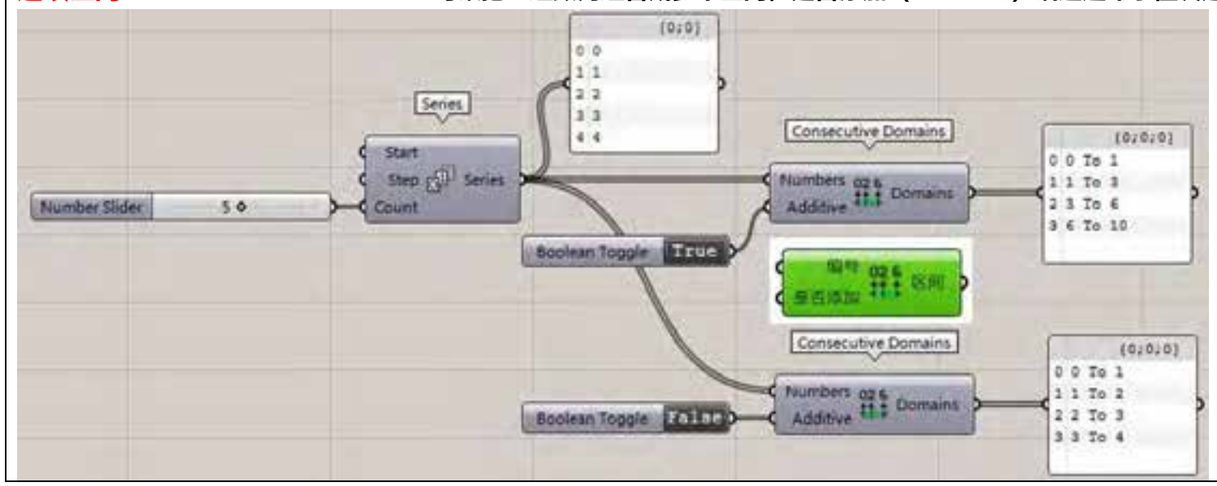
经常在曲线干扰的时候用到 remap 区间映射：







连续区间 Consecutive Domains：可以把一组数列组合成多个区间，是否添加（Additive）端通过布尔值决定数值是否要依次相加。



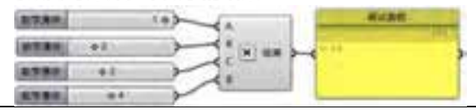
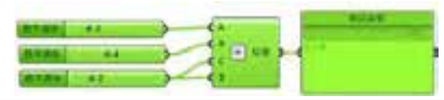
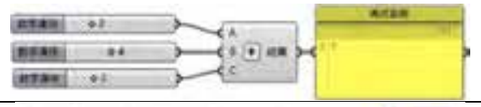
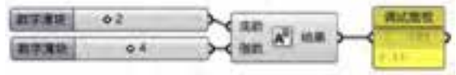
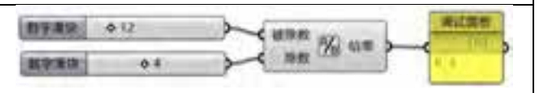
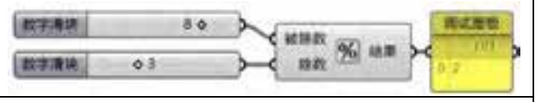
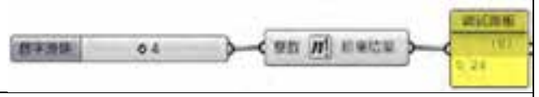
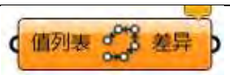
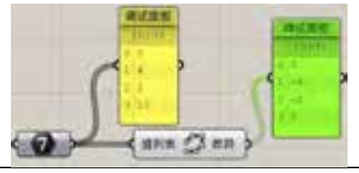
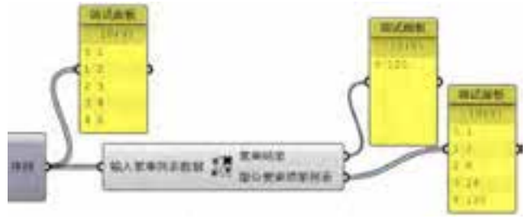
多项式 Polynomials

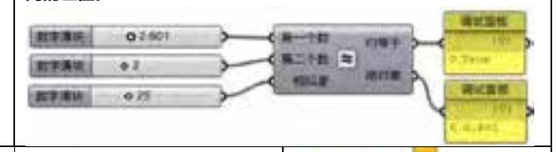
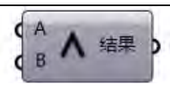
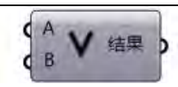
多项式 Polynomials - (数学)

平方 **Square**		**求平方值**	**平方根** **Square Root**		**求平方根**
立方 **Cube**		**求立方值**	**立方根** **Cube Root**		**求立方根**
10 的 N 次幂 **Power**		**10 的 N 次方**	**2 的 N 次幂** **Power of 2**		**2 的 N 次方**
E 的 N 次幂 **Power of e**		**e，自然常数，是数学中一个常数,是一个无限不循环小数，且为超越数,其值约为 2.71828.同时，e 也是一个成熟的细胞的平均分裂周期。**	**X 分之一** **One Over X**		**同时也是倒数。**
对数 10		**输入值** $N=10^x$**，输出 X 值**	**对数 N**		**根据公式** $N=a^x(a>0,a\neq 1)$**，输入 N 和 a 的值，输出 x 值**
自然对数		**以无理数 e（e=2.71828...）为底的对数称为自然对数（natural logarithm），并记为 lnN。** **输入值 N=** e^x**,输出 x 值**	**如果** $N=a^x(a>0,a\neq 1)$ **，即 a 的 x 次方等于 N（a>0，且 a≠1），那么数 x 叫做以 a 为底 N 的对数（logarithm），记作。其中，a 叫做对数的底数，N 叫做真数，x 叫做“以 a 为底 N 的对数”。**		

操作 Operators（运算符）

操作 Operators (运算符) - (数学)

乘法 **Multiplication**		**可以同时计算多个数字的结果** 	**减法** **Subtraction**		**可以计算多个减数的结果** 
加法 **Addition**		**可以同时计算多个数字的结果** 	**取反（负值）** **Negative**		**取负数**
幂 **Power**			**除法** **Division**		
整除 **Integer Division**		**计算整除结果，当不能被整除时，输出计算结果的整数部分（下方详解）**	**模除** **Modulus**		**计算除法的余数** 
绝对值 **Absolute**			**阶乘** **Factorial**		**阶乘，即 n!=1×2×3×...×n** 
相对差异 **Relative Differences**		**计算数据列表中后一位减去前一位数值的差值** 	**累乘** **Mass Multiplication**		**计算数列中所有数字相乘的结果** 
累加 **Mass Addition**		**计算数列中所有数字相加的结果**			

大于比较 Larger Than		判断一个数值是否大于另一个数值，输出布尔值。输出端分别为大于还是大于等于，		**小于比较 Smalller Than**		判断一个数值是否小于另一个数值，输出小于和小于等于的布尔值。	
相等比较 Equality		判断一个数值是否大于另一个数值，输出布尔值。输出端分别为大于还是大于等于		**约等于比较 Similarity**		判断两个数的差值是否在设定的相似度（百分比）范围内，在误差范围内则判断为相似，输出为 True。输出端绝对差输出的是数值之间的差值。 	
与门 Gate And		**或门 Gate Or**		**异或门 Gate Xor**		**非门 Gate Not**	
与非门 Gate Nand		**多门 Gate Majority**		**同或门 Gate Xnor**		**或非门 Gate Nor**	

与门、或门、非门、与非门、或非门、与或非门、异或门：

是常用的门电路逻辑功能。（1 代表 true,0 代表 false）

与门是两个或者两个以上输入端，一个输出端的器件。只要有一个输入端为 0 时，所有的组合结果输出就是 0。只有所有的输入端是 1 时，输出才是 1。

或门是两个或者两个以上输入端，一个输出端的器件。只要有一个输入端为 1 时，所有的组合结果输出就是 1。只有所有的输入端是 0 时，输出才是 0。

非门是一个输入端，一个输出端的器件，它的作用是使输入信号反向。例如输入 0，那么输出就是 1，输入 1，输出就是 0。

异或门是两个输入端输入相同的数据时（都是 1 或者都是 0），则输出 0；只有两个输入端输入的数据不同时，输出 1。

同或门则与异或门的结果正好相反，输入相同的数据时，输出 1，输入不同的数据时，输出 0.

首先我们知道 True 和 False 对应数字 1 和 0

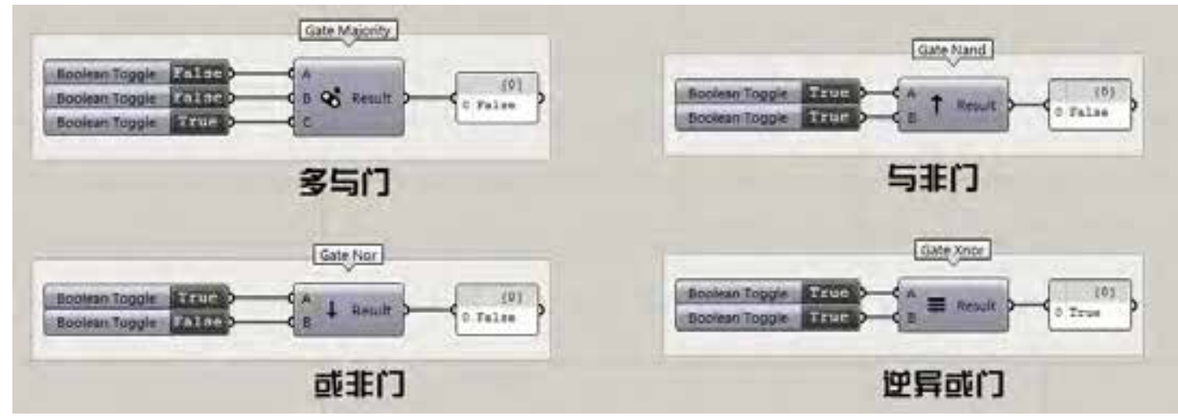
与门你可以把它看成乘法只要有 0 都为 0 （1 与 0 为 0，0 与 0 为 0，1 与 1 为 1）

或门你可以把它看成加法只要有 1 都为 1 （1 或 0 为 1，0 或 0 为 0，1 或 1 为 1）

非门你可以把它看成倒数（0 的时候为 1，1 的时候为 0 ）。

异或门可以看成减法求绝对值（都是 1 或者都是 0 的时候，相减绝对值等于 0）

同或门与异或门正好相反，也可以看做是输入的两个数字相加，再减去 1，求绝对值。



这四个我们也放到一起来，是前面四个的升级

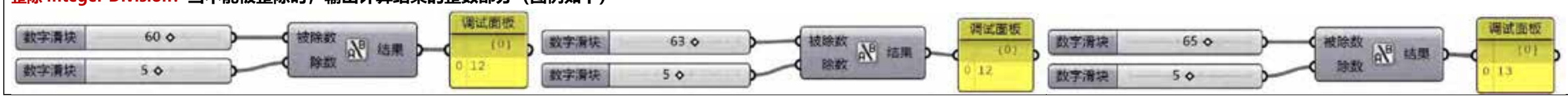
****多门：**三个里超过两个为 True 才输出 True，否则都输出 False**

****与非门：**其实就是与门+非门（与门结果反转得出的结果）**

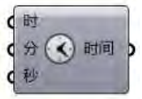
****或非门：**或门+非门（或门结果反转得出的结果）**

****同或门：**只有两个输入端相同才会输出 True**

整除 Integer Division：当不能被整除时，输出计算结果的整数部分（图例如下）

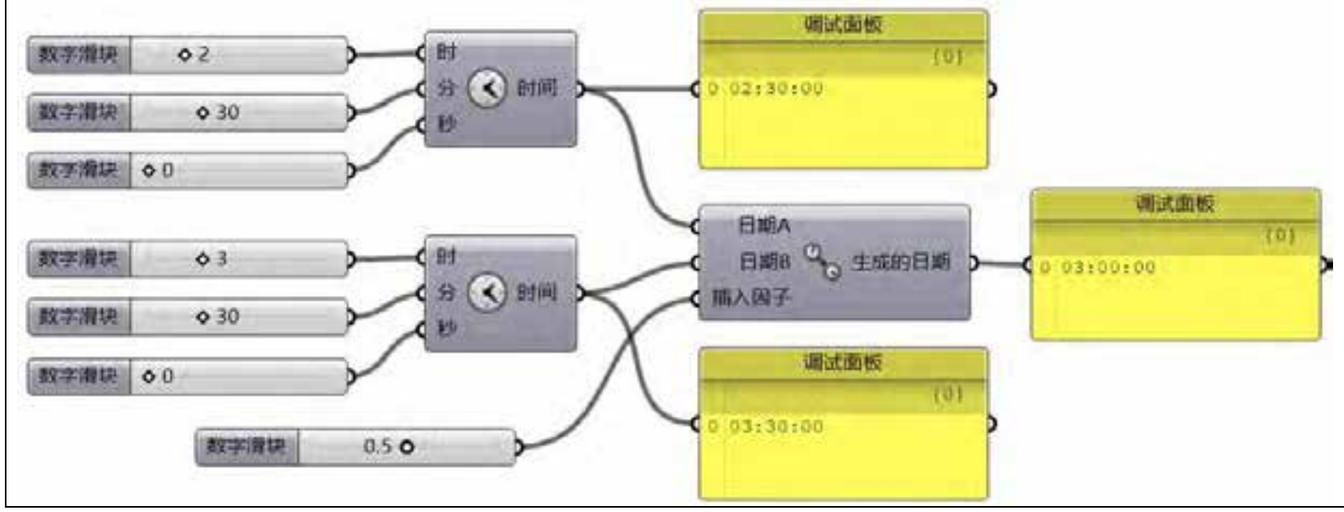


时间 Time-（数学）

分解时间 Deconstruct Date		**分解提取时间参数**	**构件平滑时间 Construct Smooth Time**		**构建一个时间段**
构造日期 Construct Date		**右键可以设定当前时间和日期**	**构造时间 Construct Time**		**右键可以设置当前时间**
构造特定日期 Construct Exotic Date		**设定公历日期，转换成各个国家（右键设置）的历法日期，比如中国的农历**			
插入时间 Interpolate Date		**在 A 和 B 之间，按照比例位置（0 和 1 之间）插入时间** **（下方详解）**	**日期范围 Date Range**		**等分时间区间，返回对应时间点。**
结合日期和时间		**把上面构建的日期和时间整合起来**			

插入时间 Interpolate Date:

在设定的日期或者时间区间，按比例插入时间点，输出对应的时间数据，



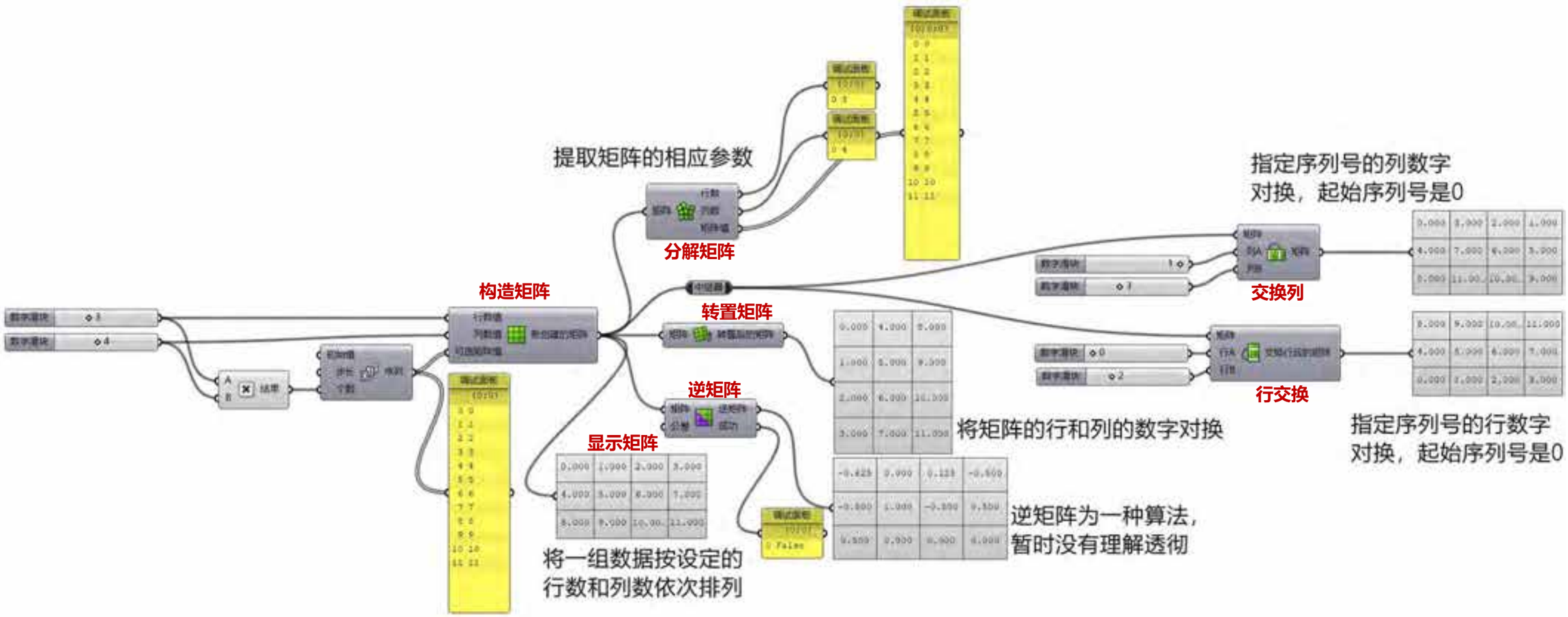
矩阵 Matrix

矩阵 Matrix- （数学）

分接矩阵 Deconstruct Matrix		（下方详解）	**显示矩阵 Display Matrix**		**显示矩阵内容**
构造矩阵 Construct Matrix		（下方详解）	**转置矩阵 Transpose Matrix**		（下方详解）
逆矩阵 Invert Matrix		（下方详解）			
交换列 Swap Columns		（下方详解）	**行交换 Swap Rows**		（下方详解）

矩阵 Matrix：是一个按照长方阵列排列的复数或实数集合（可以理解成在一个矩形表格中的数据，或者是一个矩形平面所包含的点）

三大基本功能——切换摄像机、控制球机、联动报警



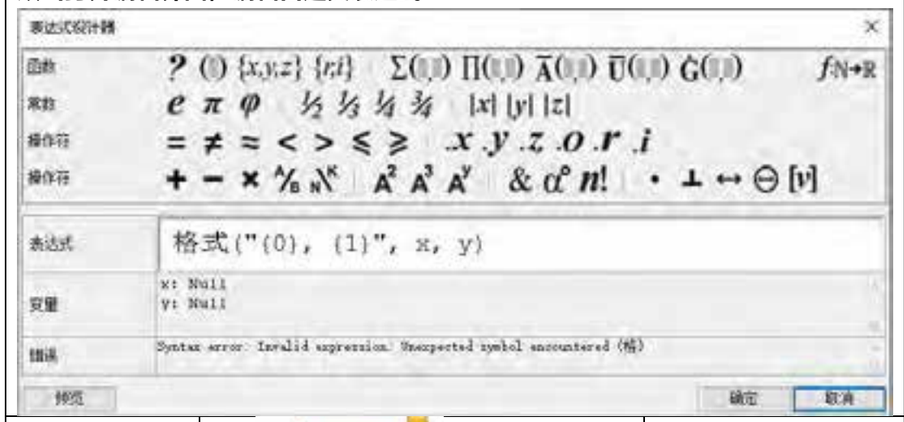
脚本 Script-（数学）

表达式
Expression

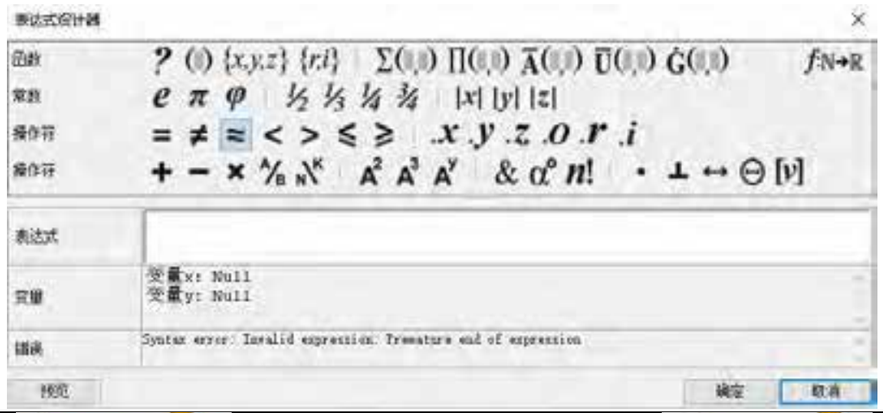
计算表达式
Evaluate

（下方详解）

双击打开编辑界面，编辑自定义表达式



可以输入表达式，也可以双击打开编辑界面，编辑自定义表达式



C#脚本编辑器

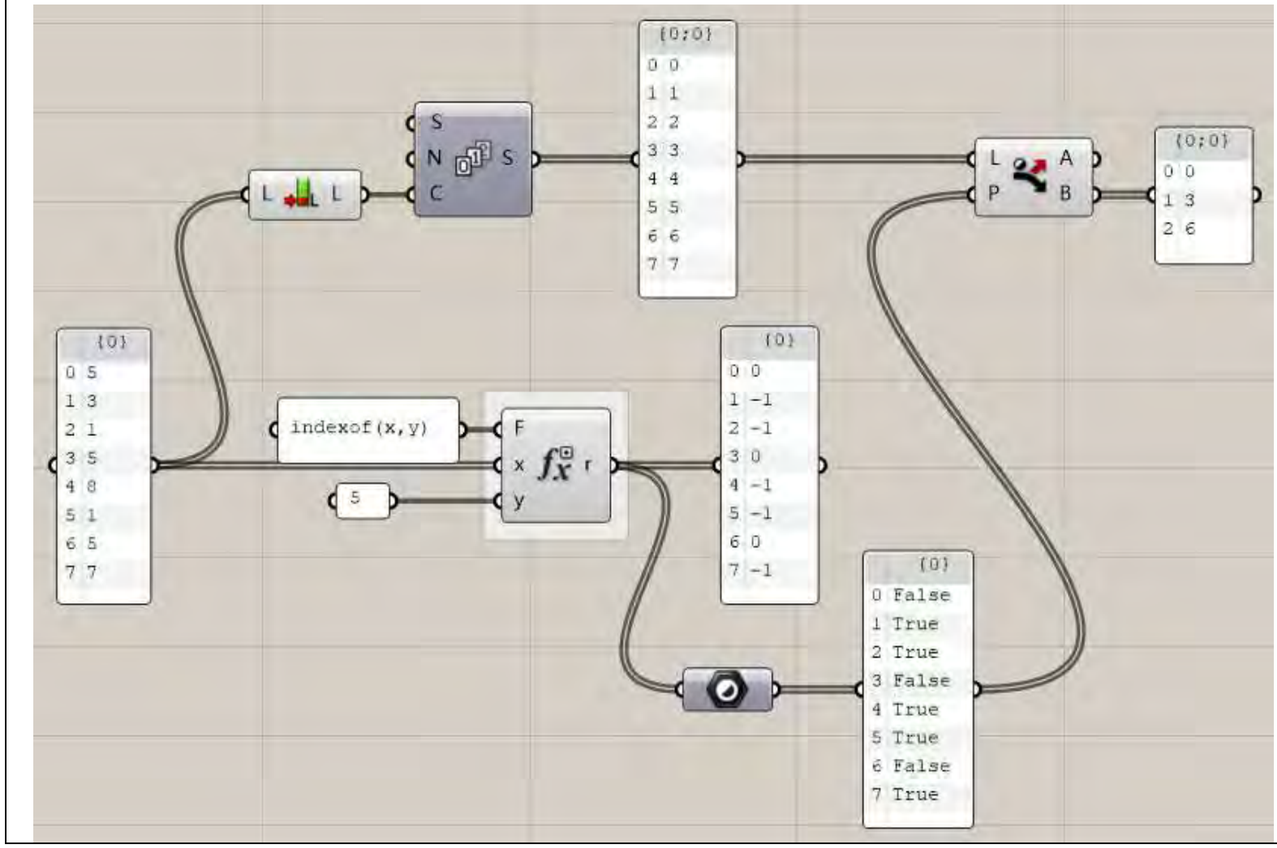
双击进入编辑

GhPython 脚本

双击进入编辑

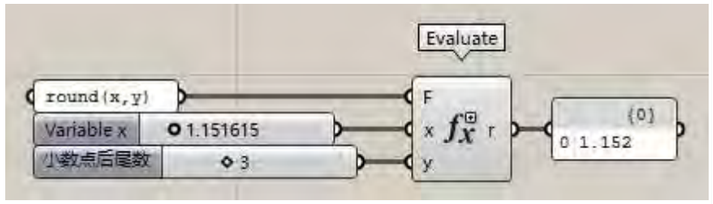
VB 脚本编辑器

双击进入编辑



计算表达式 Evaluate：

除了我们自己随便设置的运算法则之外，我们还可以调用很多内置的函数，比如经常会有同学问的一个问题：如何修改小数点后的尾数？

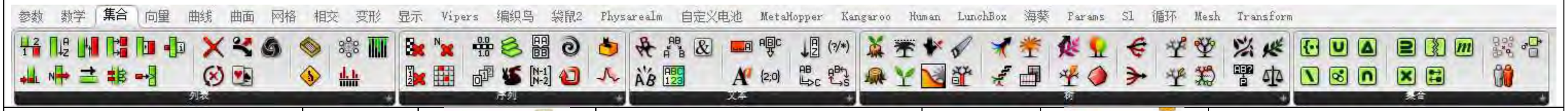


除此之外，Evaluate 还有很多用法， 比如再一个很多人会问的问题，如何得到指定数据的所有序号？需要注意的是，这个函数只认 text，直接输入数字是会报错的，必须先接给 panel。

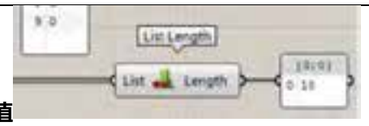
集合（sets）

列表 List

集合（sets） ——非常重要的数据处理运算器组

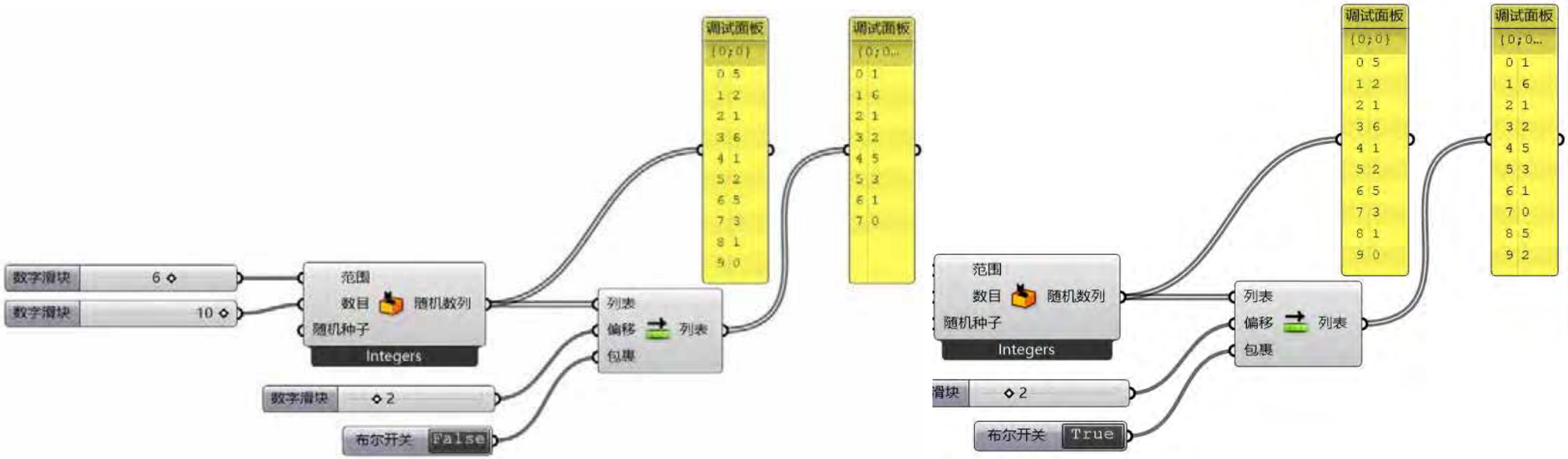


列表 List-（集合）

倒排列表 Reverse List		使列表中的数据反向排列，在列表的右键中的反转是同样的作用	**偏移列表 Shift List**		按照设定的偏移值偏移列表中的数值，包裹输入的布尔值决定是删除偏移的数据还是移动到最后（下方详解）
分割列表 Split list		在指定的序列号索引处，将列表分成两个（下方详解）	**列表分块 Partition List**		将列表按照指定的数量大小分组（下方详解）
列表替换 Replace Items		将列表中指定索引的数据替换为要替换的项输入的数据，包裹的布尔值决定输入的序列号（索引）大于列表的最大序列号时，是否从列表中循环取值（下方详解）	**列表长度 List Length**		输出列表长度值 
同步排序 Sort List		将列表的数值按从小到大的顺序重新排列，英文则按字母列表排序。很常用的运算器，比如我想将点按照高度排序，或者根据线长短进行排序等等。	**提取子集 Sub List**		从列表中抽取指定区间序号的数据，索引输出对应的序号（下方详解）
插入数据 Insert Items		在指定列表的指定序号位置插入指定的数据（可以同时插入多个）。包裹端为布尔值，决定如果序号打入列表总数量是否循环取值。（下方详解）	**查特定项 List Item**		取出列表中指定序列号（索引）的数据，包裹为布尔值，决定当输入的索引序列号大圩列表中最大序列号时，是否循环取值（下方详解）
索引查找 Item Index		查找列表中特定数据的序号，而且查找项输入的内容必须是原列表中提取的内容，这些内容包含了其在原始数列的序列号参数，所以即使有多个内容相同的数据，也只返回一个值。查找项输入端的数据如果不存在于列表中，则返回-1。（下方详解）			
判断空值 Null Item		判断列表中 Null 空值和 Invalid 无效值并返回布尔值，可以结合数据分流来筛选错误数据。（下方详解）	**填充空值 Replace Nulls**		替换数据列表中所有 null 空白值替换为输入的数据，并返回 null 值的数量
数据分流 Dispatch		列表按照分流模式输入的布尔值数列（默认 0 和 1 构成的数列）分成两组输出（下方详解）	**数据筛选 Pick'n'Choose**		模式端输入的是 0 和 1 组成的数列，数据流 0 和数据流 1 依次按照模式的规则依次抽取对应序列号的数据 （下方详解）
编织 Weave		根据编织模式设定的规则（0 和 1 构成的数列），依次将数据流 0 和数据流 1 合并成一组，可以增加多组自定义数据流接口（下方详解）	**布尔筛选（模式筛选） Sift Pattern**		类似数据分流，但是分组输出的数据依然放在对应原始数据的位置，没有数据的用 null 替换。输出端可以自由增加数量（下方详解）
组合数据 Combine Data		用于消除 null 空白值，输入 0 中的 null 空白值用输入 1 中对应序列号的数据替换，多出的数据依次排列在下面（下方详解）	**交叉引用 Cross Reference**		数据交叉处理：当有两组或多组数列时，如何处理不同列表之间的计算（下方详解）
短列表连接 Shortest List		短排法，数量多的列表 list，多处的数据直接删除。这个短排法算法比较少，就三个（下方详解）	**长列表连接 Longest List**		长排法，数量多的列表多出的数据全都和数据少的列表的最后一个数据运算。右键可以选择不同的算法，默认的是 repeat last，就是重复运算最后一个数据，而 GH 中的所有列表的运算方法都是默认 Longest list。（下方详解）

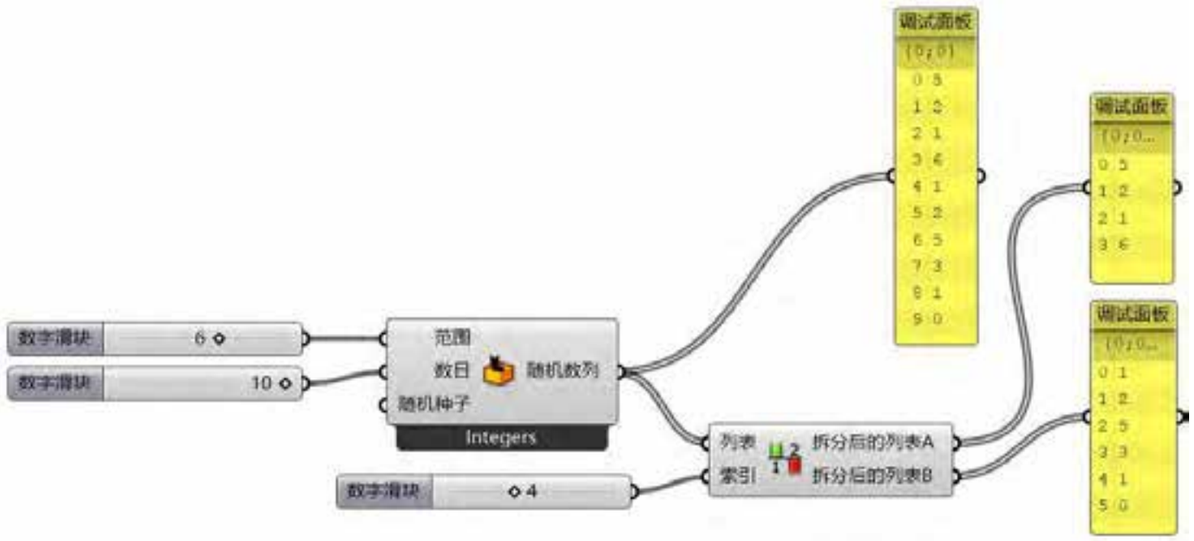
偏移列表 Shift List:

按照设定的偏移值偏移列表中的数值，包裹输入的布尔值决定偏移的数据是直接删除，还是通过循环列表插入最后。注意布尔值变化的结果



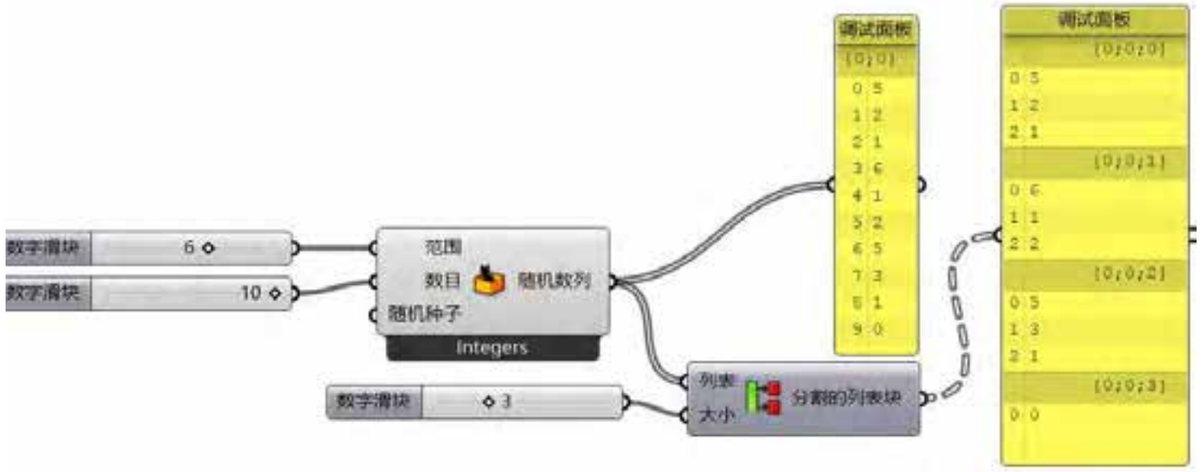
分割列表 Split list:

在指定的序列号索引处，将列表分成两个



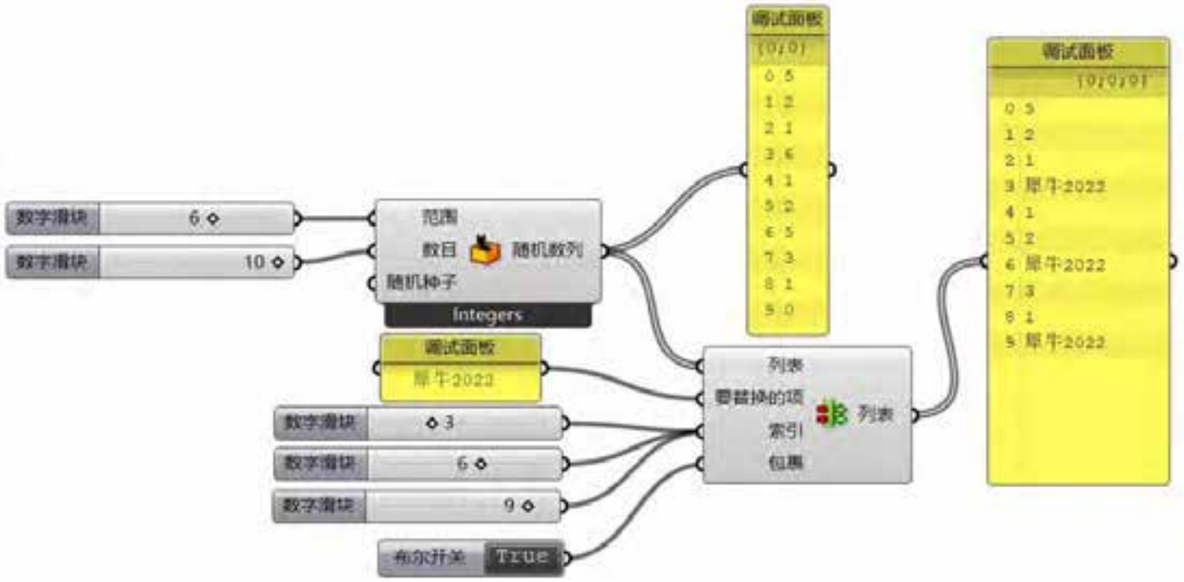
列表分块 Partition List:

将列表按照指定的数量大小分组



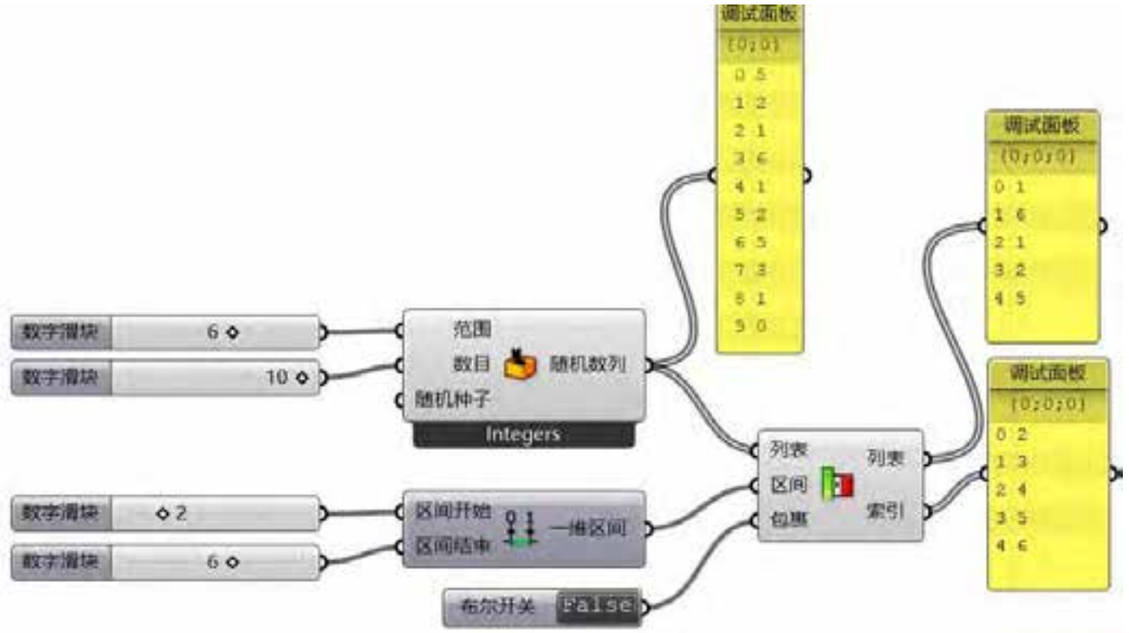
列表替换 Replace Items

用要替换的项输入的内容替换指定序列号索引的内容，可以同时指定多个序列号



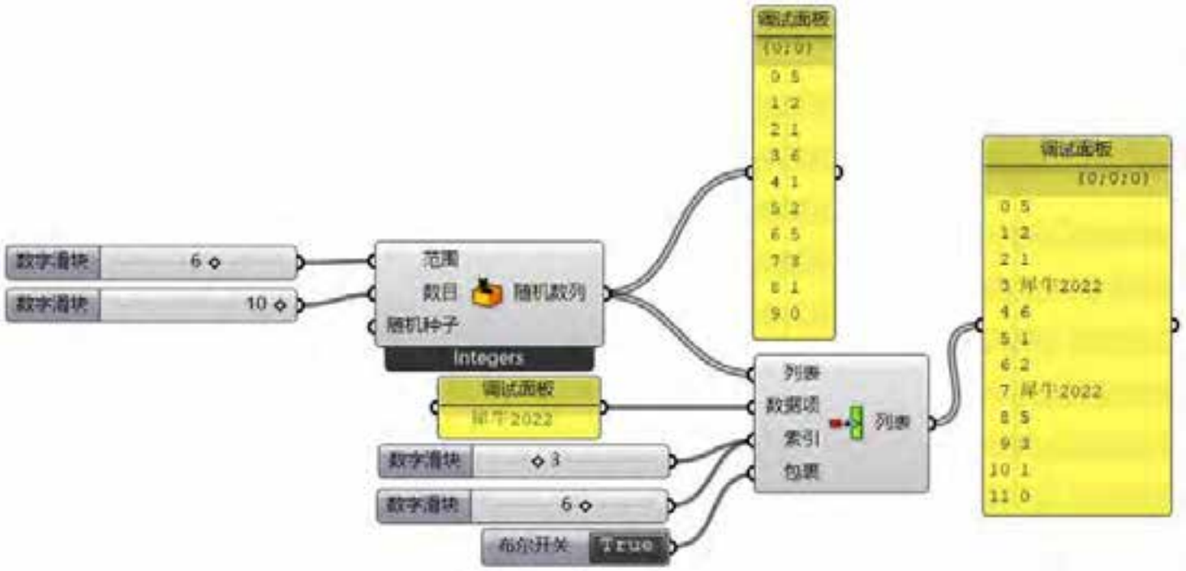
提取子集 Sub List:

提取指定区间序号的数据及序号



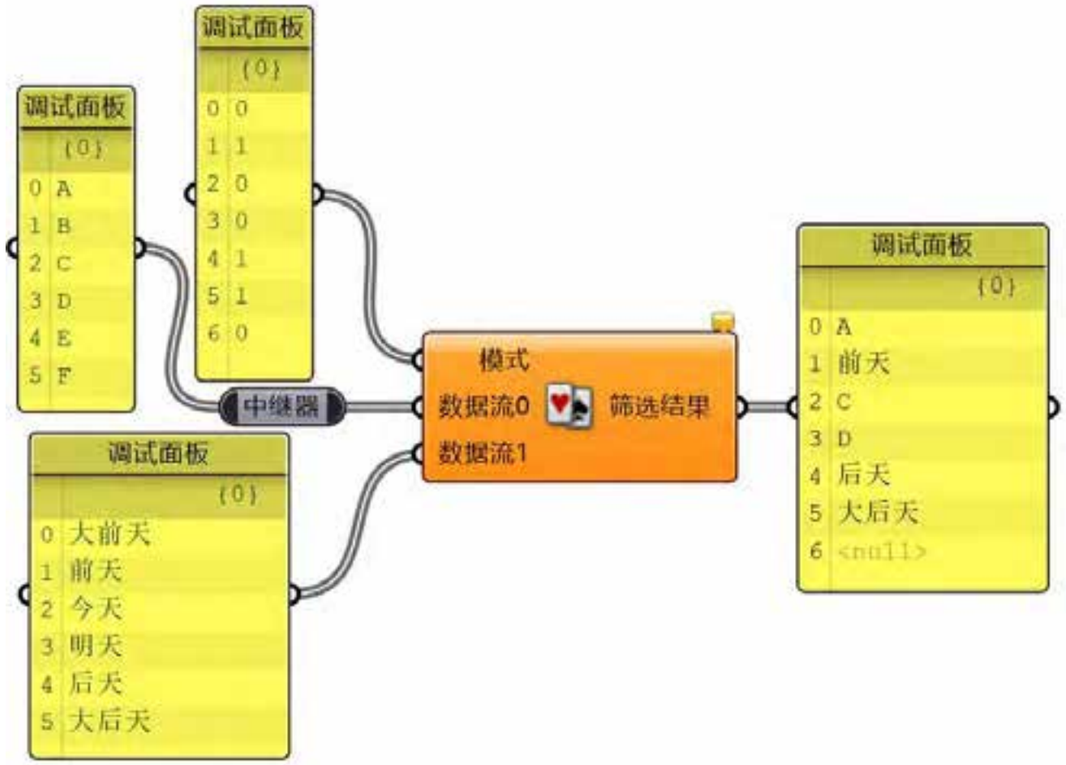
插入数据 Insert Items:

在指定列表的指定序号位置插入指定的数据（可以同时插入多个）。包裹端为布尔值，决定如果序号打入列表总数量是否循环取值。



数据筛选 Pick'n'Choose:

模式端输入的是 0 和 1 组成的数列，数据流 0 和数据流 1 依次按照模式的规则依次抽取对应序列号的数据

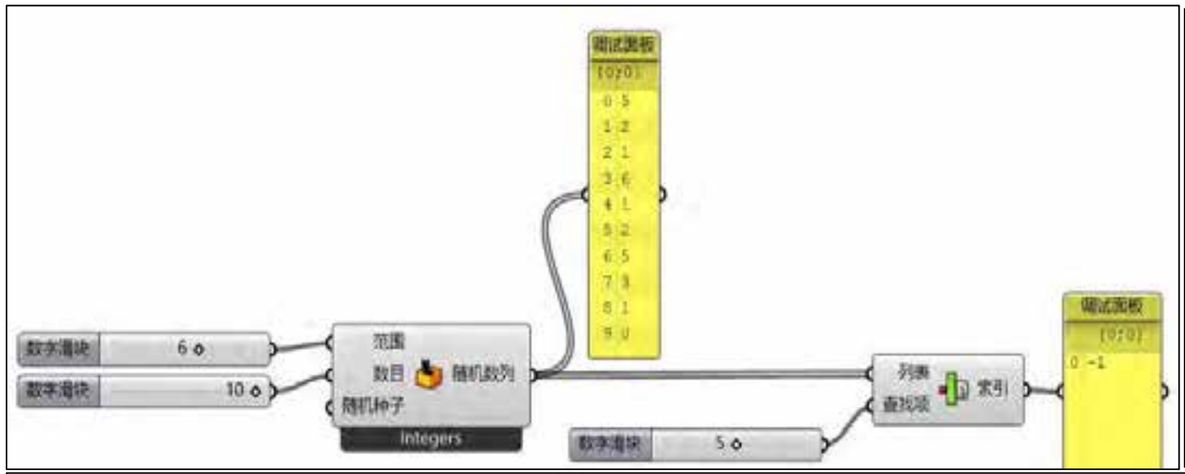


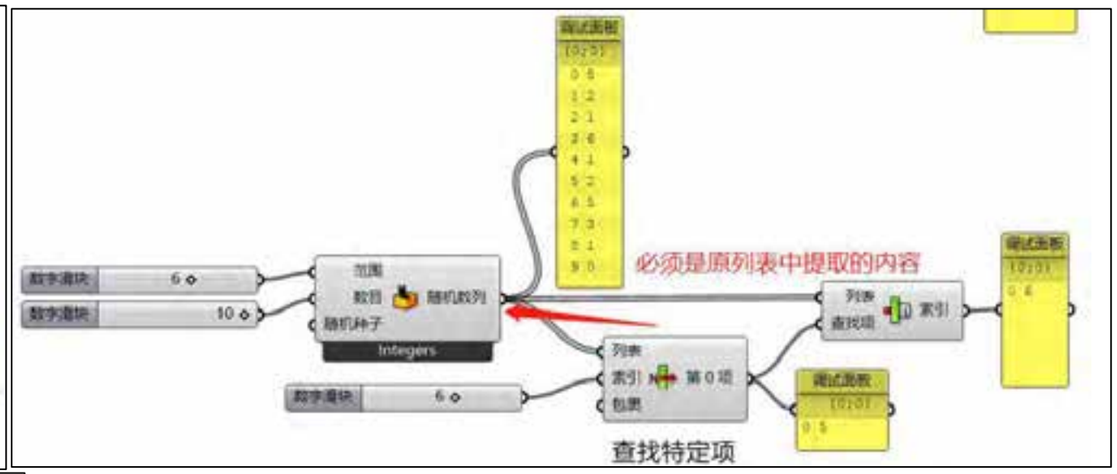
索引查找 Item Index ：

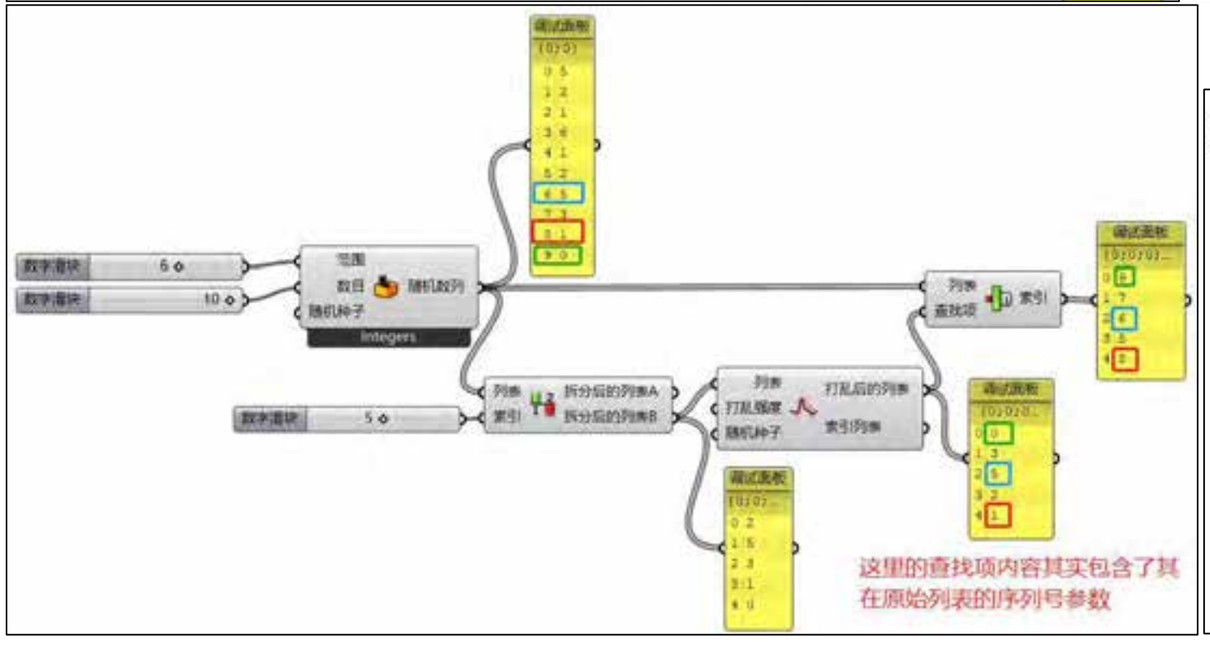
查找列表中特定数据的序号，而且查找项输入的内容必须是原列表中提取的内容，而且即使有多个重复内容，也只会返回一个序列号（提取的内容实际上包含了其在原始列表的序列号参数）

查特定项 List Item：

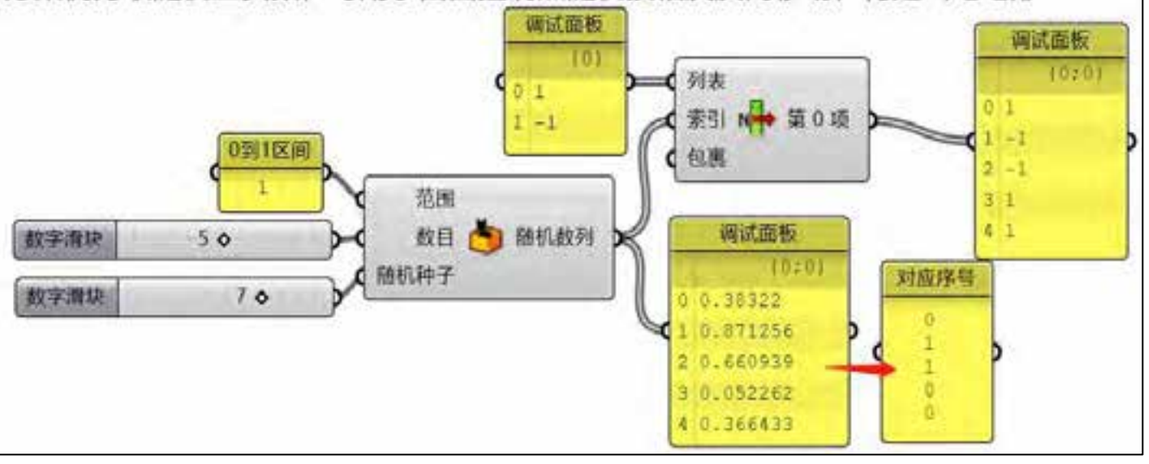
根据指定的序号（索引）查找列表中的内容，当输入的序号（索引）为带小数的数值时，会自动按照四舍五入进行取整。



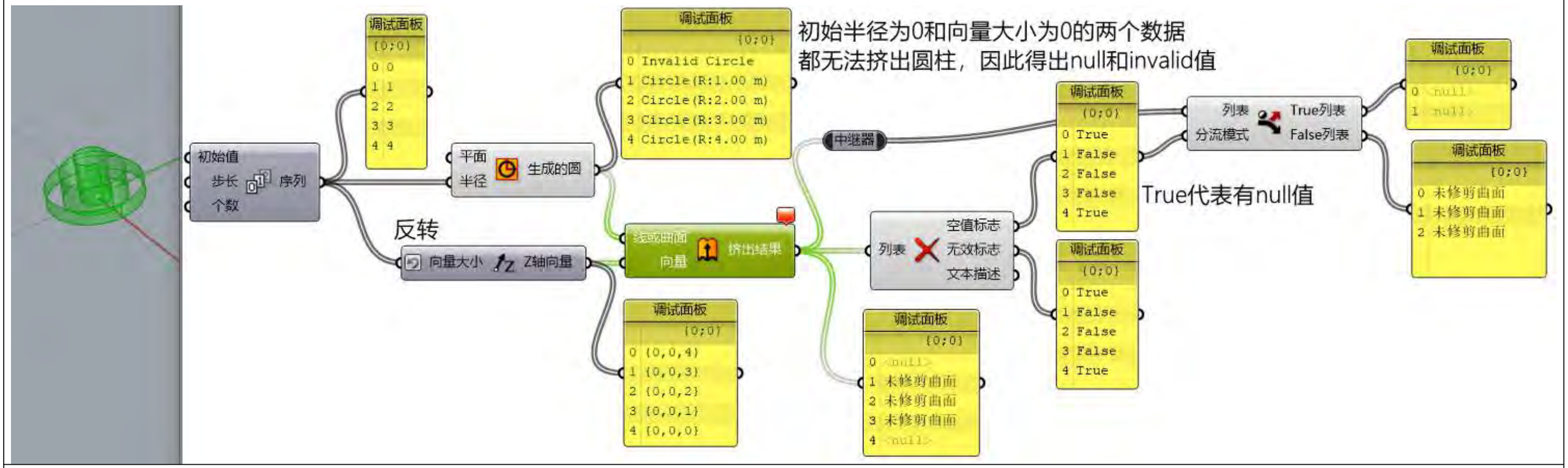




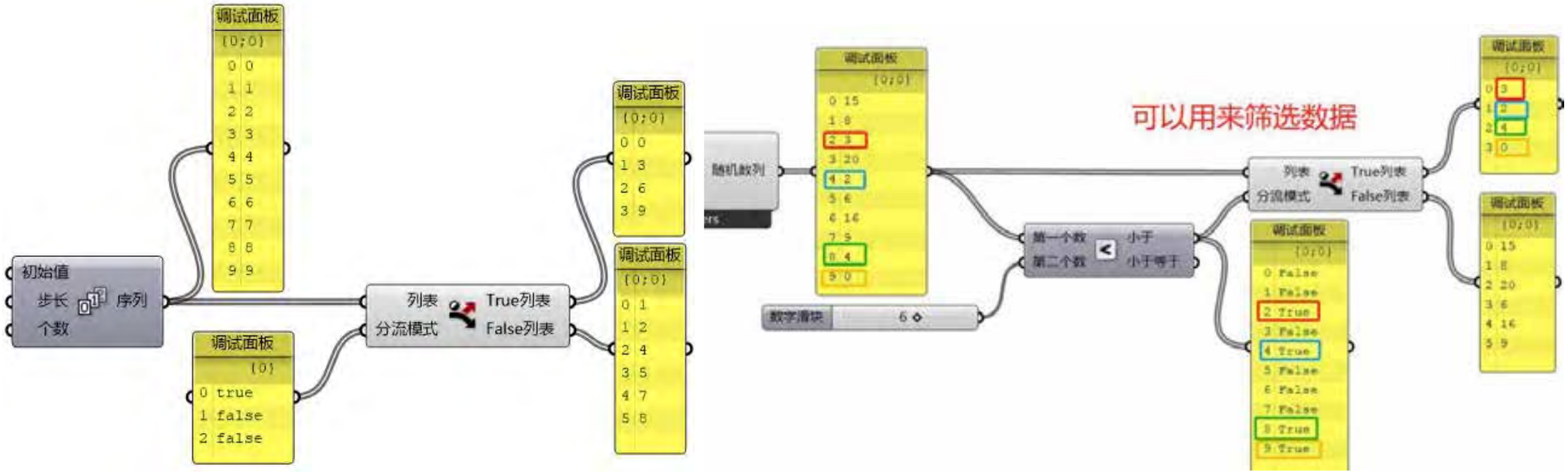
查特定项索引为小数时，四舍五入取整。
下方案例为取随机正负数，可用于曲面控制点随机按相反反向移动，构建山地地形



判断空值 Null Item： **判断列表中 Null 空值和 Invalid 无效（非法）值并分别输出布尔值，可以结合数据分流来筛选错误数据。**

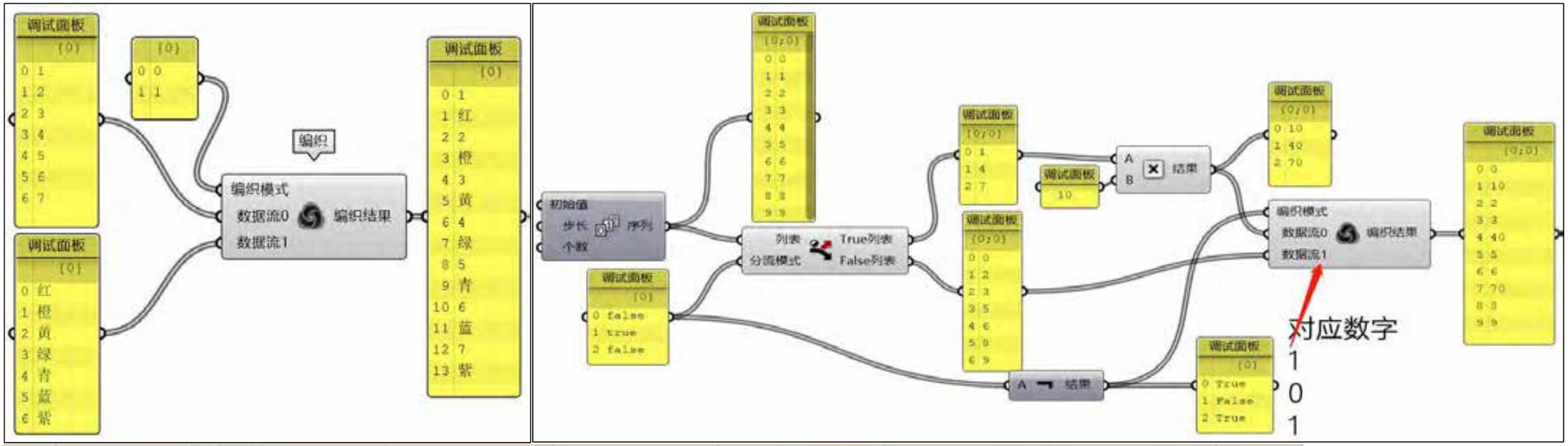


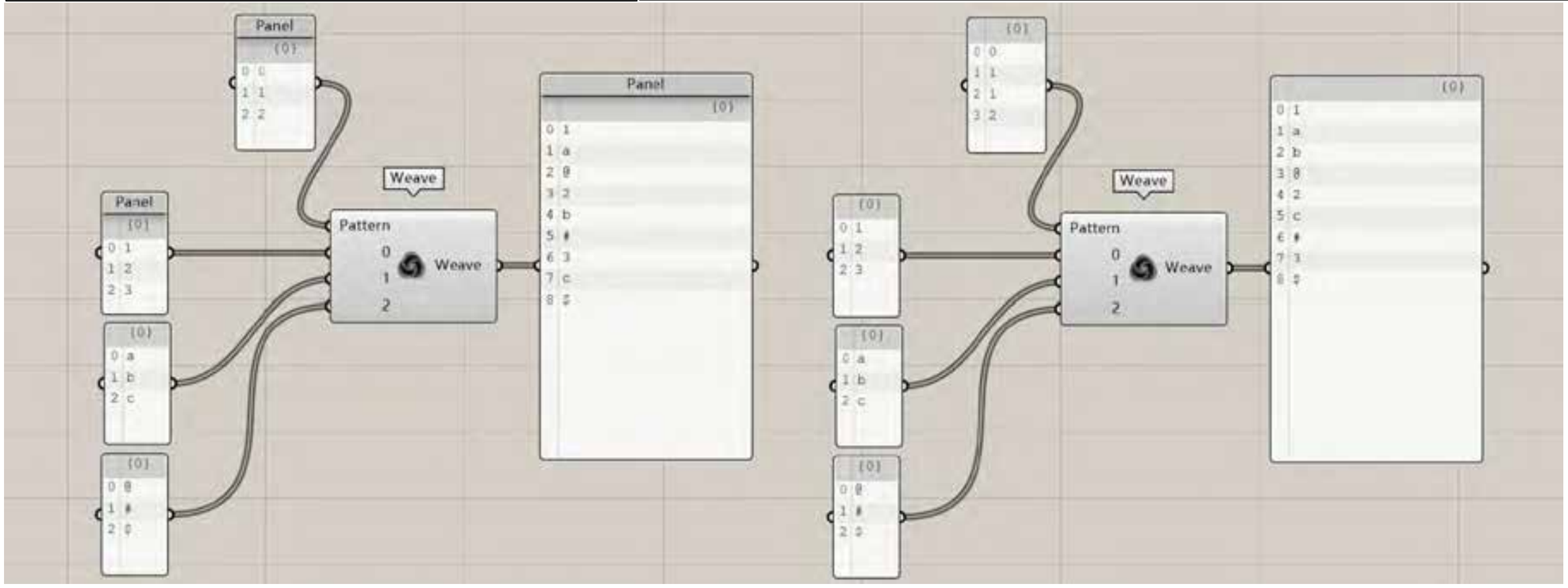
数据分流 Dispatch： **按照分流模型中的布尔值顺序取值，超出范围则循环取值，输出到对应的列表。可以用来筛选小于某个值的数据。**



编织 Weave：

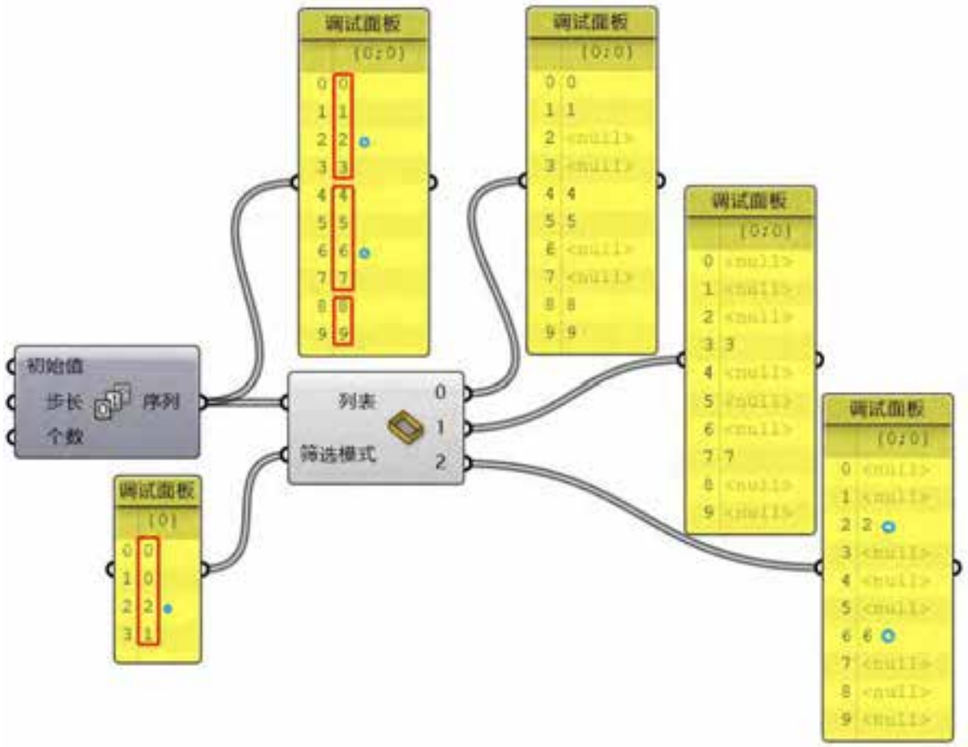
根据编织模式设定的规则（0 和 1 构成的数列），依次将数据流 0 和数据流 1 合并成一组，可以增加多组自定义数据流接口。（布尔值对应数字 0 和 1）





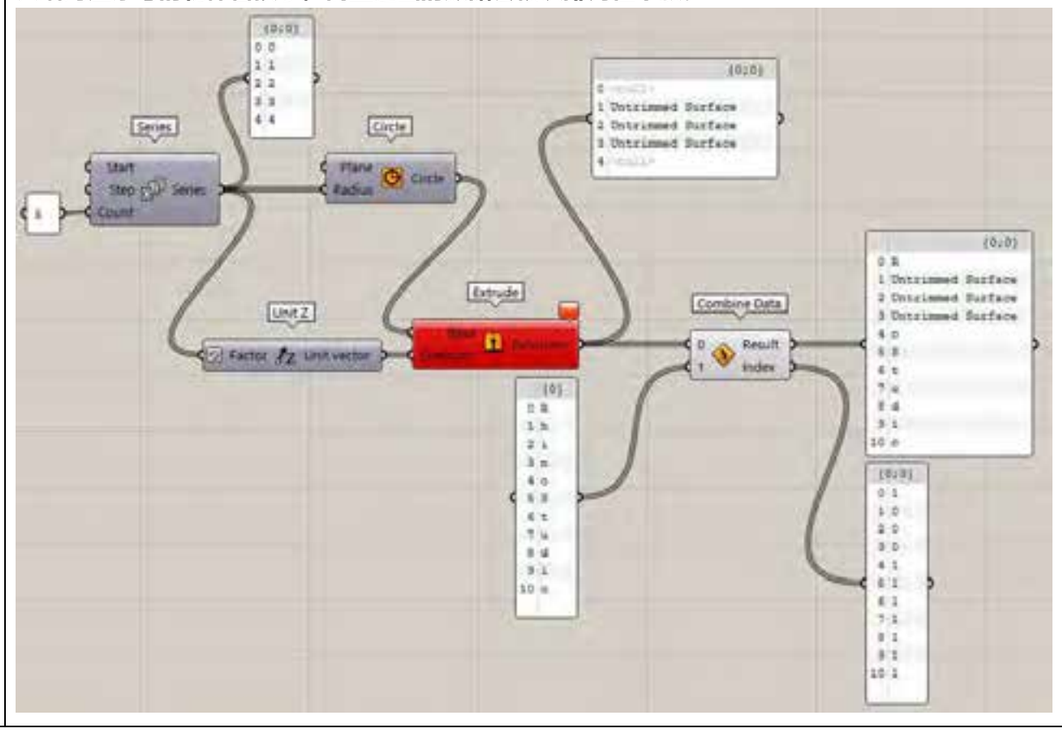
布尔筛选（模式筛选）Sift Pattern：

类似数据分流，但是分组输出的数据依然放在对应原始数据的位置，没有数据的用 null 替换。输出端可以自由增加数量



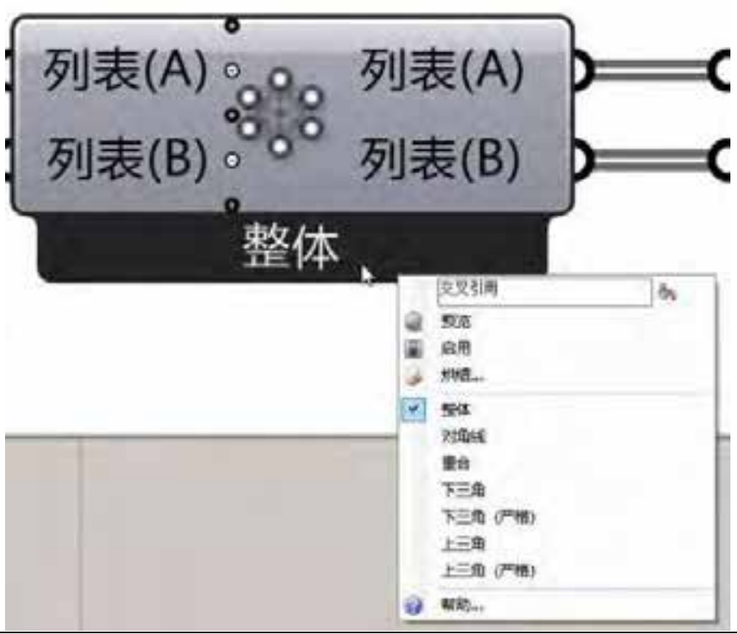
组合数据 Combine Data：

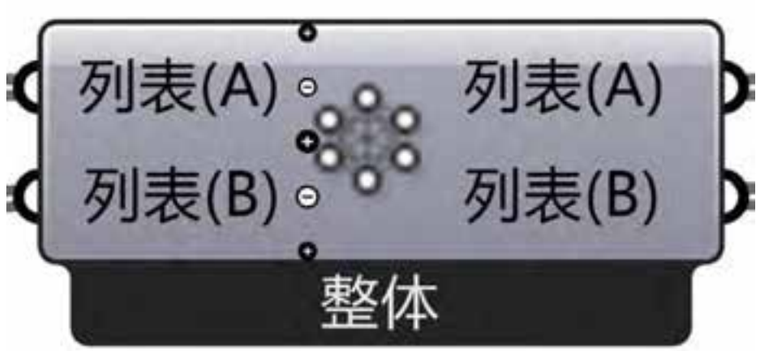
合并几个输入端的数据，目标就为了一个：消除 Null 值。如图，你可以看到 0 输入端列表中 null 值，都被 1 输入端对应序号的物体替换了，而且超出的数据会依次排列在下放。



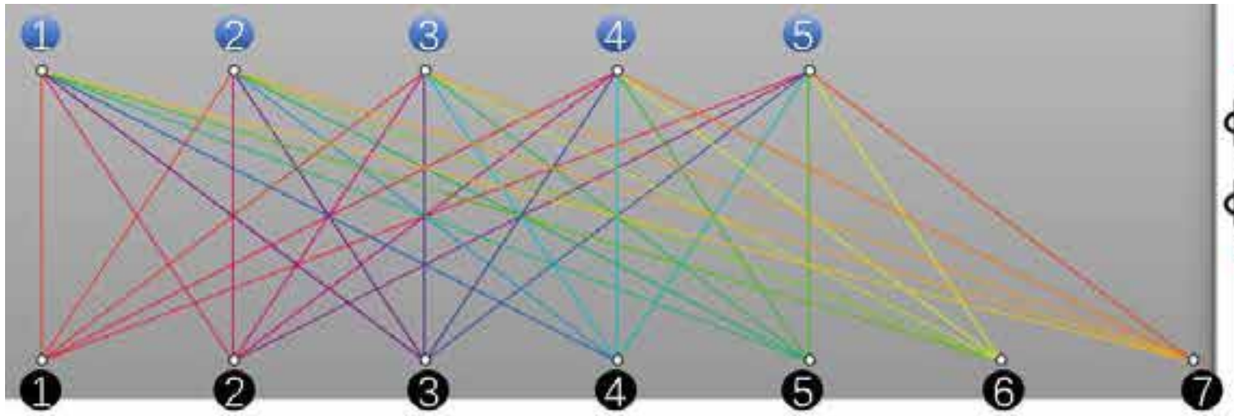
交叉引用 Cross Reference：

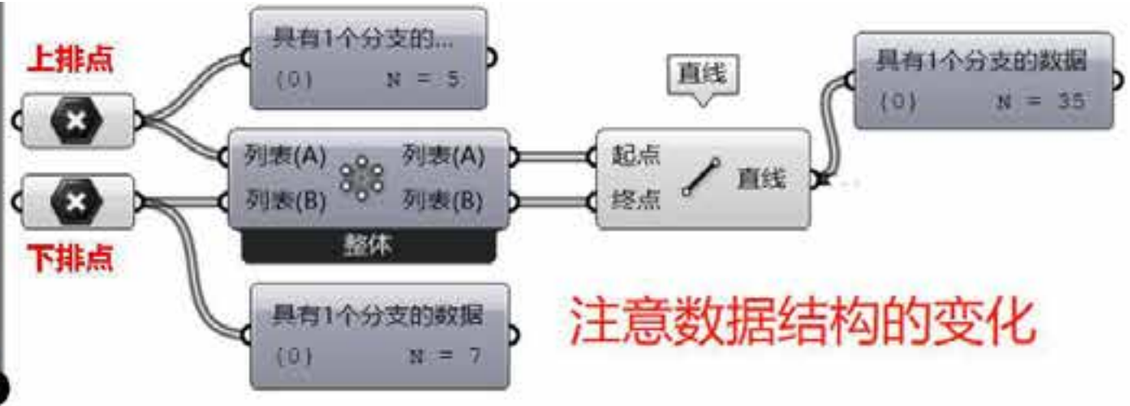
数据交叉处理：当有两组或多组数列时，如何处理不同列表之间的计算，可以增加任意输入端数量，右键可以选择不同的交叉计算方法：



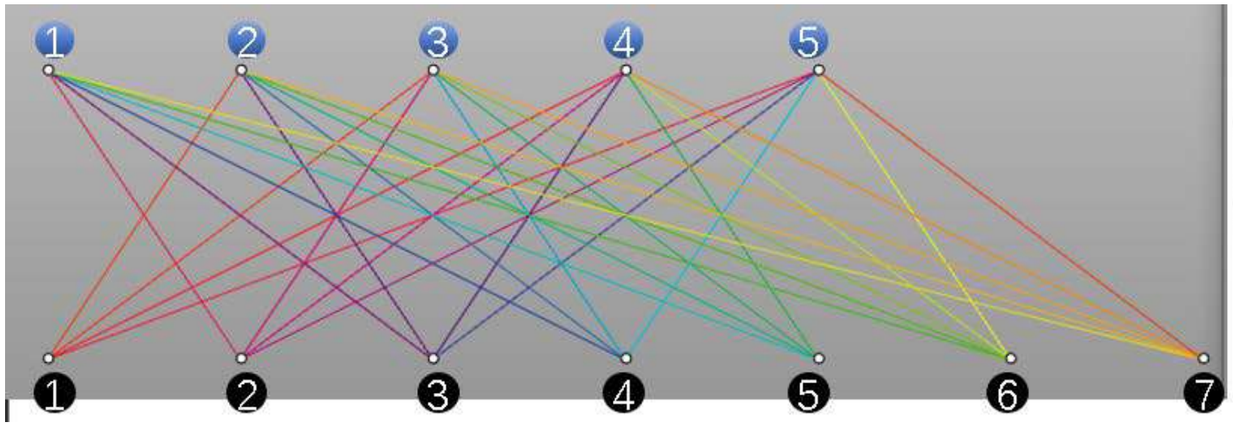


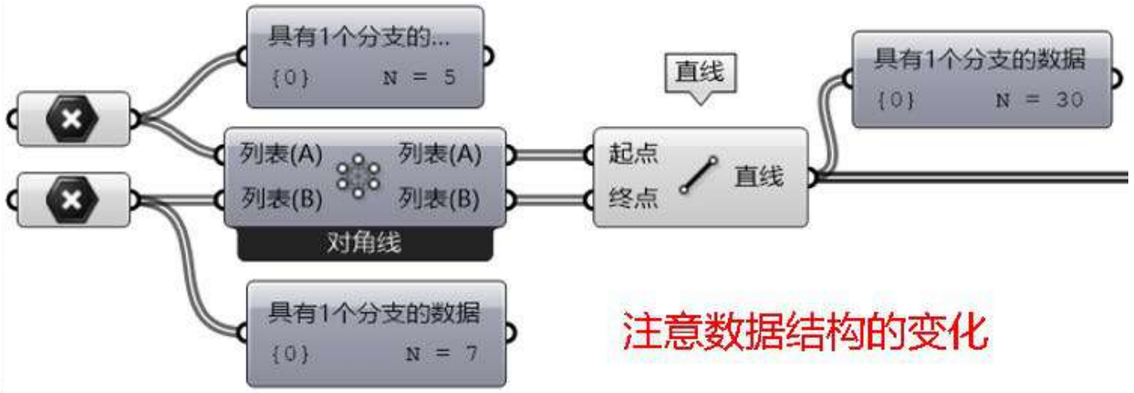
整体模式：使两组或多组数据之间，每个数据都互相进行计算。如下图每两点直接都有连线。





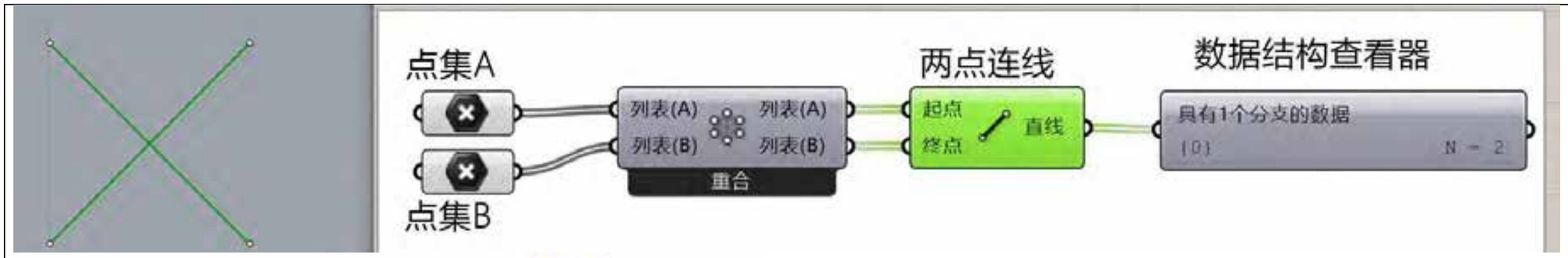
对角线模式：两组数据序号相同的数据不进行计算。如下图对面的两点不生成直线。

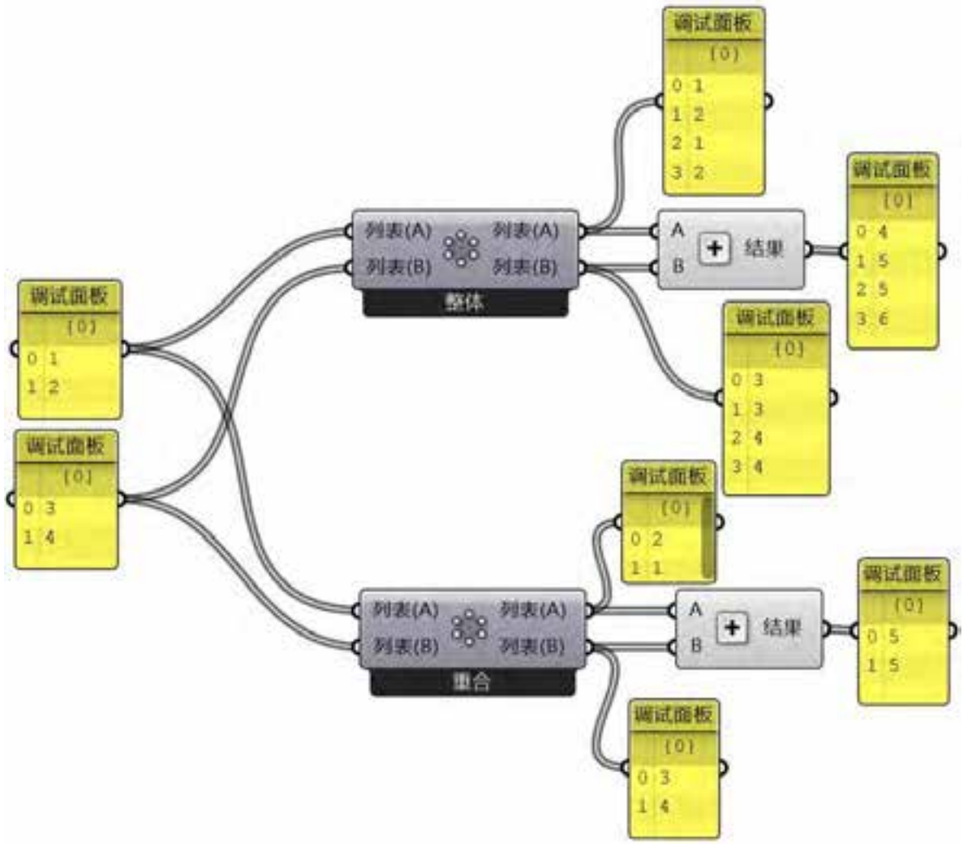


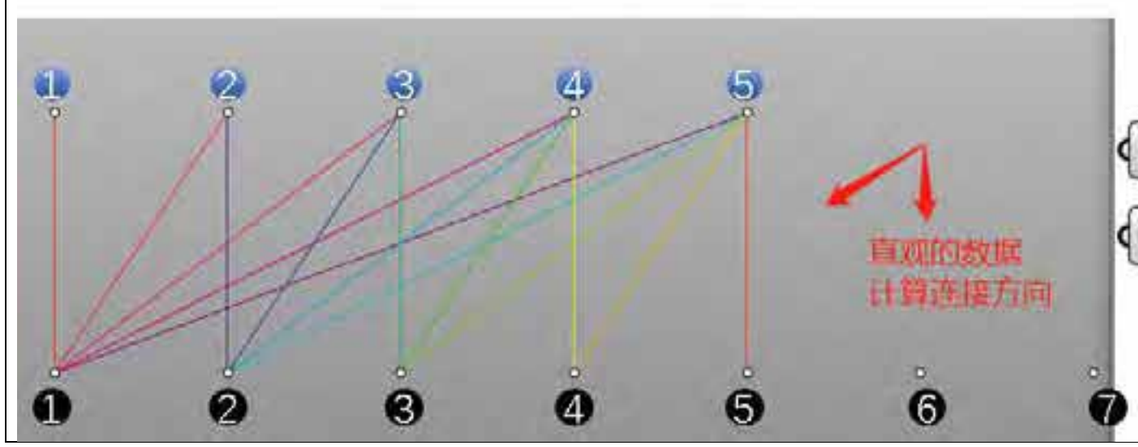


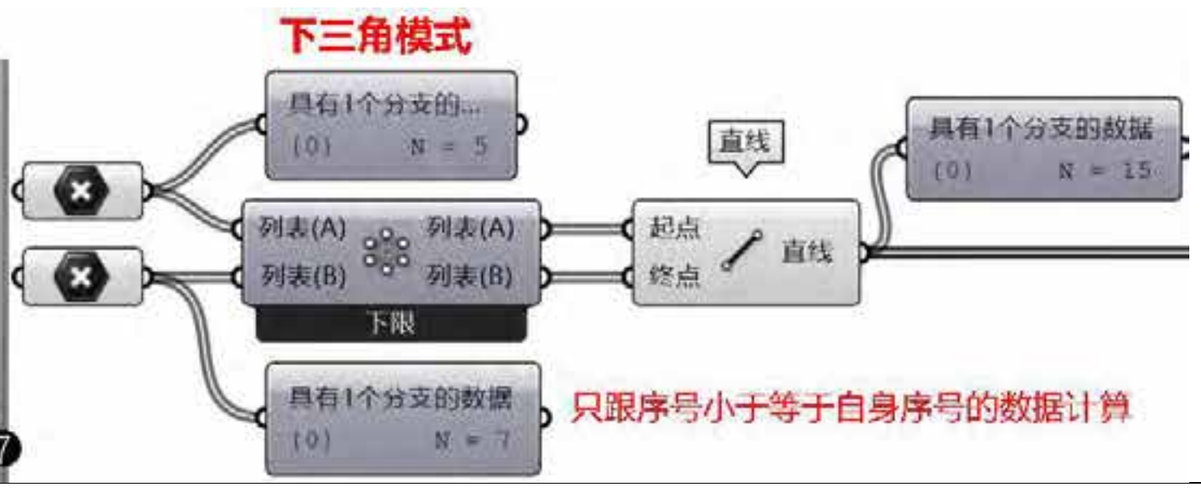
重合模式：消除重复数据。因为每个数据都会和其他数据进行计算，意味着肯定会有重复数据，这种交叉算法就是删除那些重复数据。

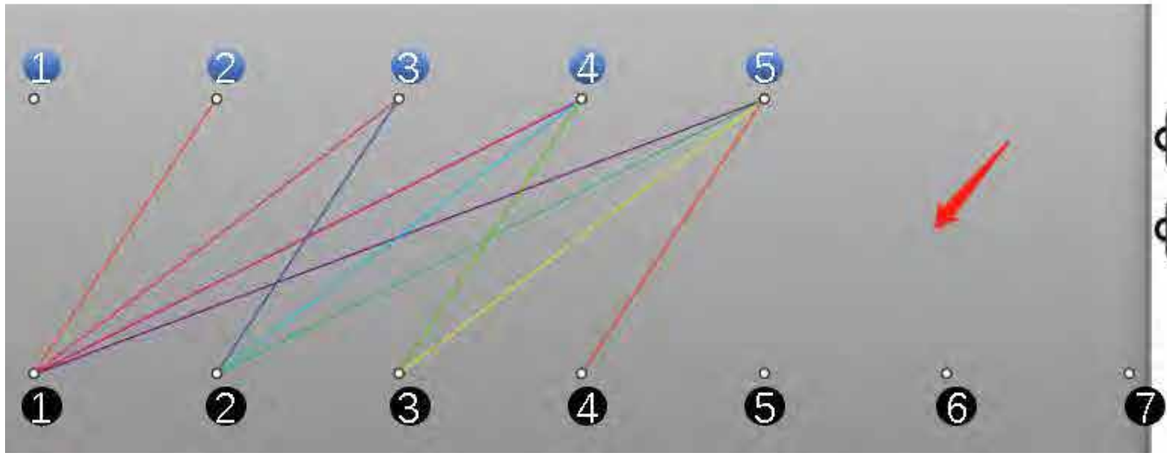
（注意看下面几组数据计算的对比，不过我暂时还没有理解）


点集A
点集B
列表(A)
列表(B)
重合
两点连线
起点
终点
直线
数据结构查看器
具有1个分支的数据
{0}
N = 2

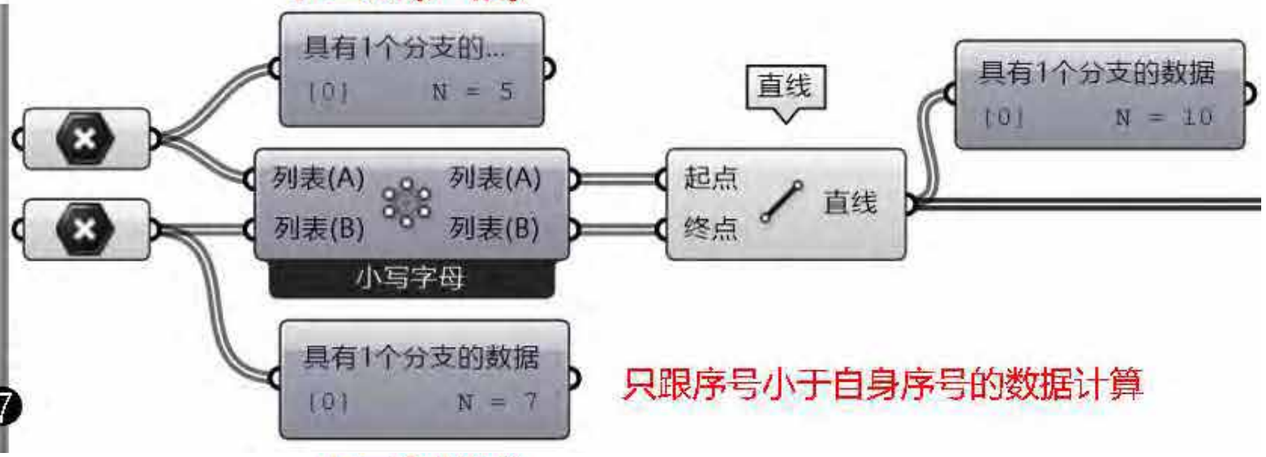

调试面板
列表(A)
列表(B)
整体
重合
结果
A
B


1
2
3
4
5
6
7
直观的数据
计算连接方向

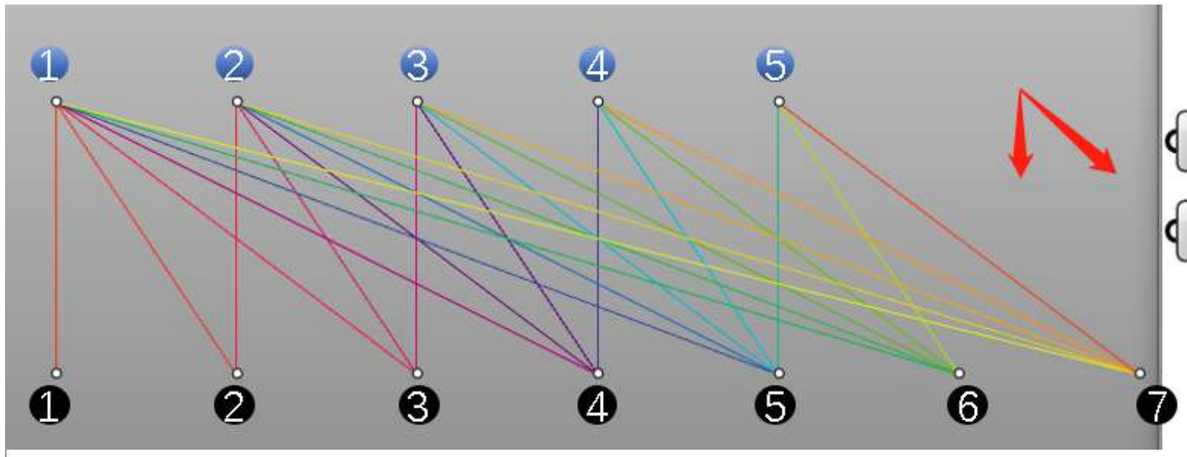

下三角模式
具有1个分支的...
{0}
N = 5
列表(A)
列表(B)
下限
直线
起点
终点
具有1个分支的数据
N = 15
具有1个分支的数据
N = 7
只跟序号小于等于自身序号的数据计算



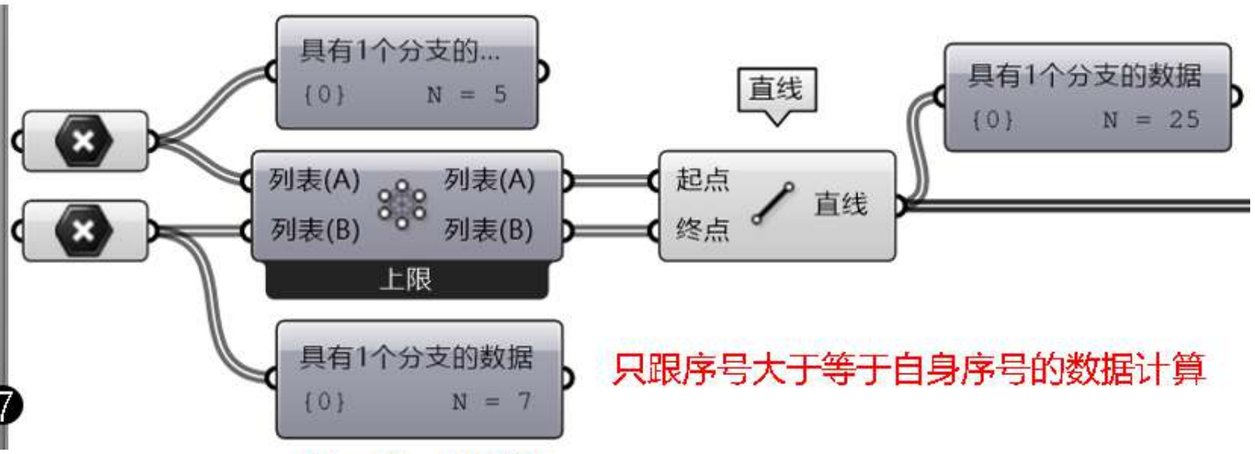
下三角(严格)



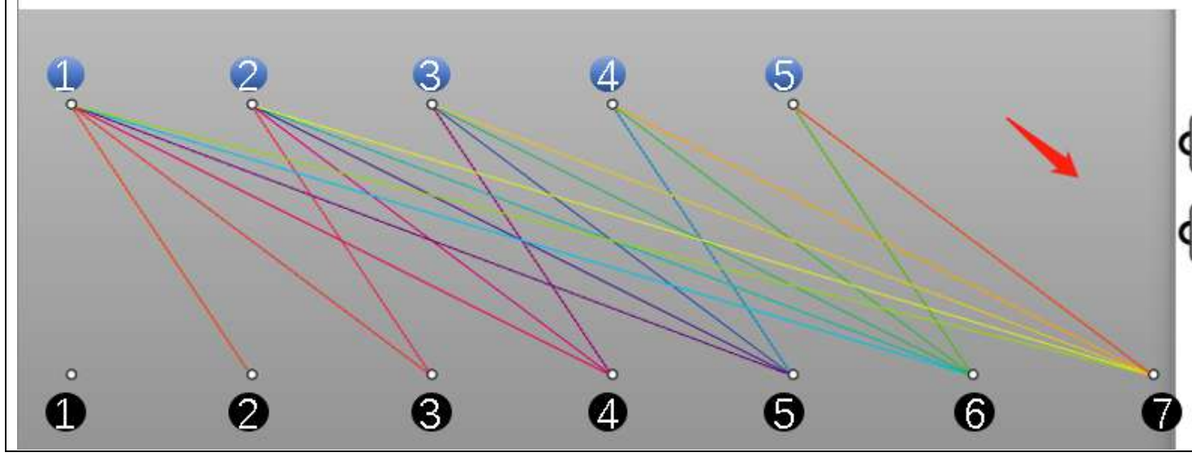
只跟序号小于自身序号的数据计算



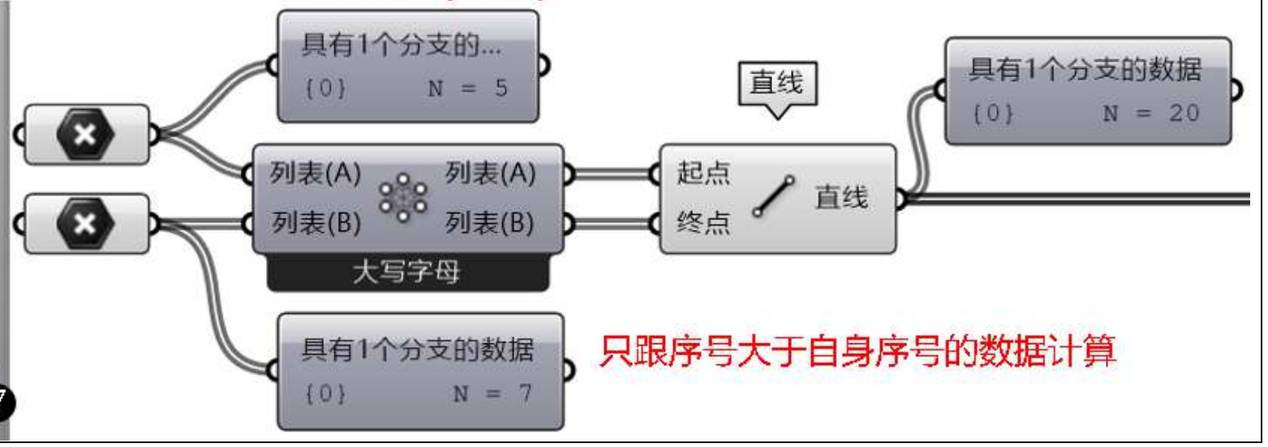
上三角模式



只跟序号大于等于自身序号的数据计算



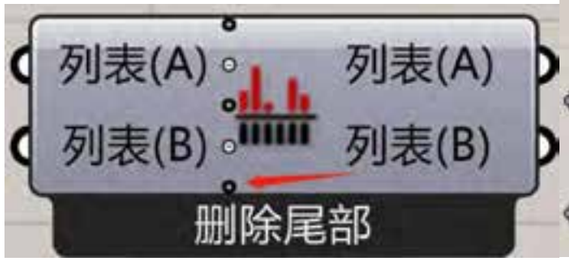
上三角（严格）

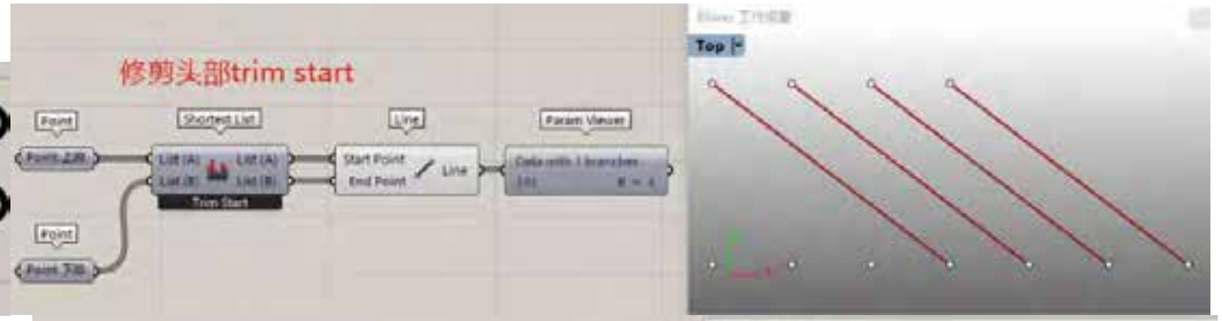


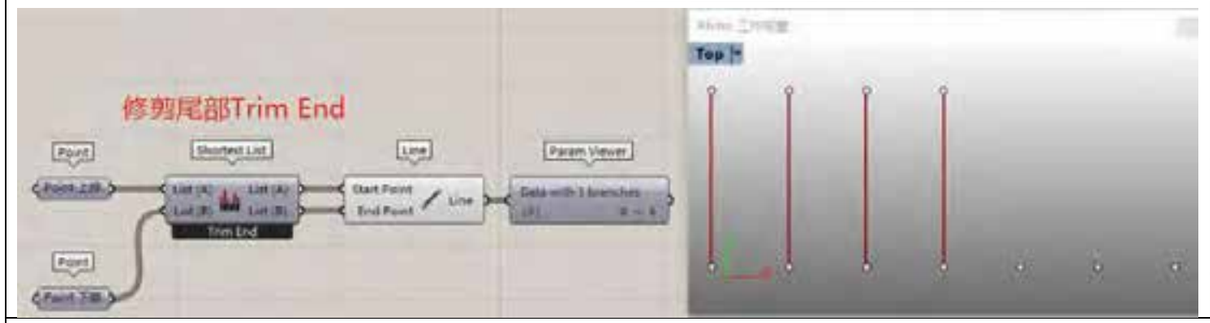
只跟序号大于自身序号的数据计算

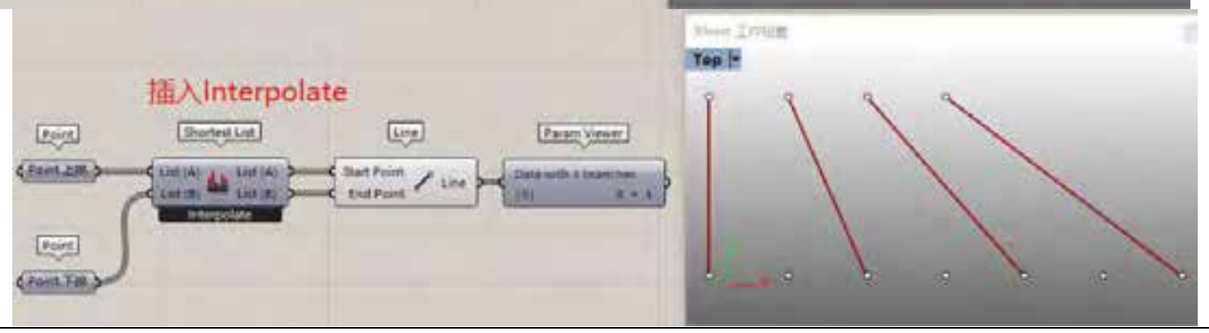
短列表连接 Shortest List：

短排法，数量多的列表 list 多出的数据直接删除。这个短排法算法比较少，就三个：修剪（删除）头部 Trim Start、修剪（删除）尾部 Trim End、插入 Interpolate。依然可以增减输入端数量。



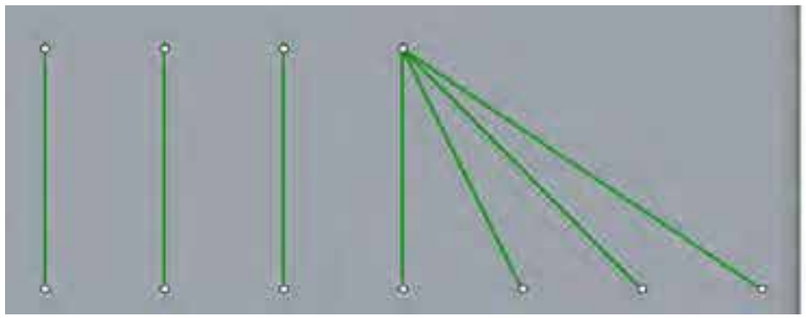


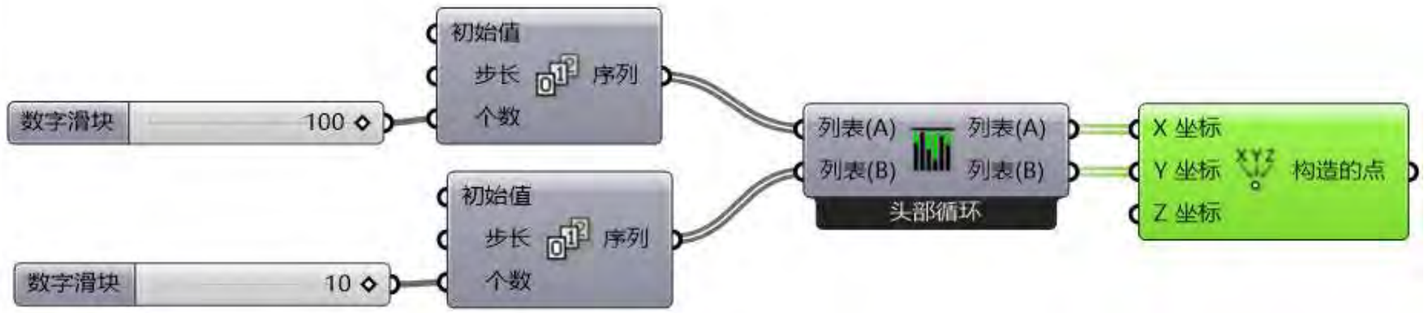


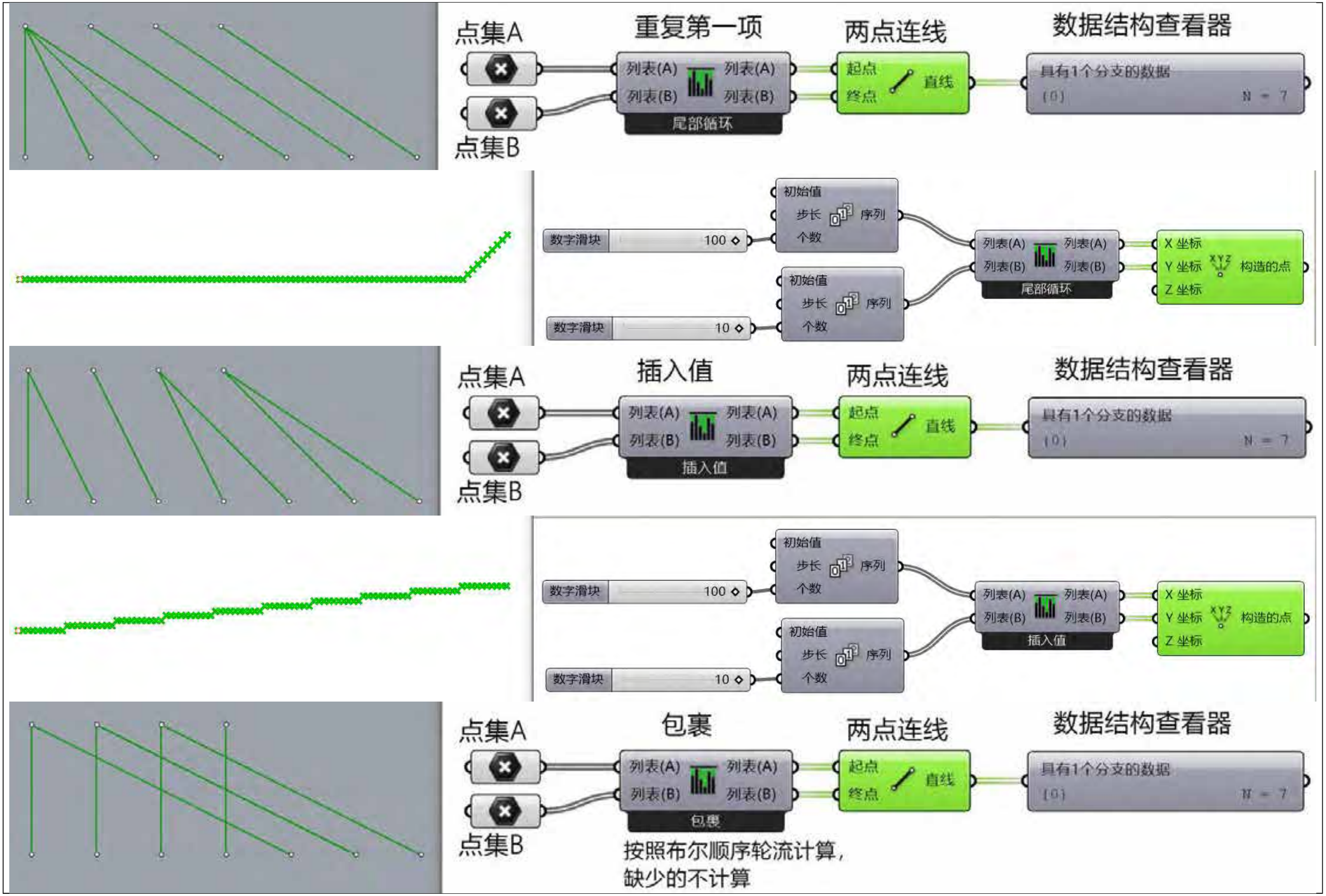


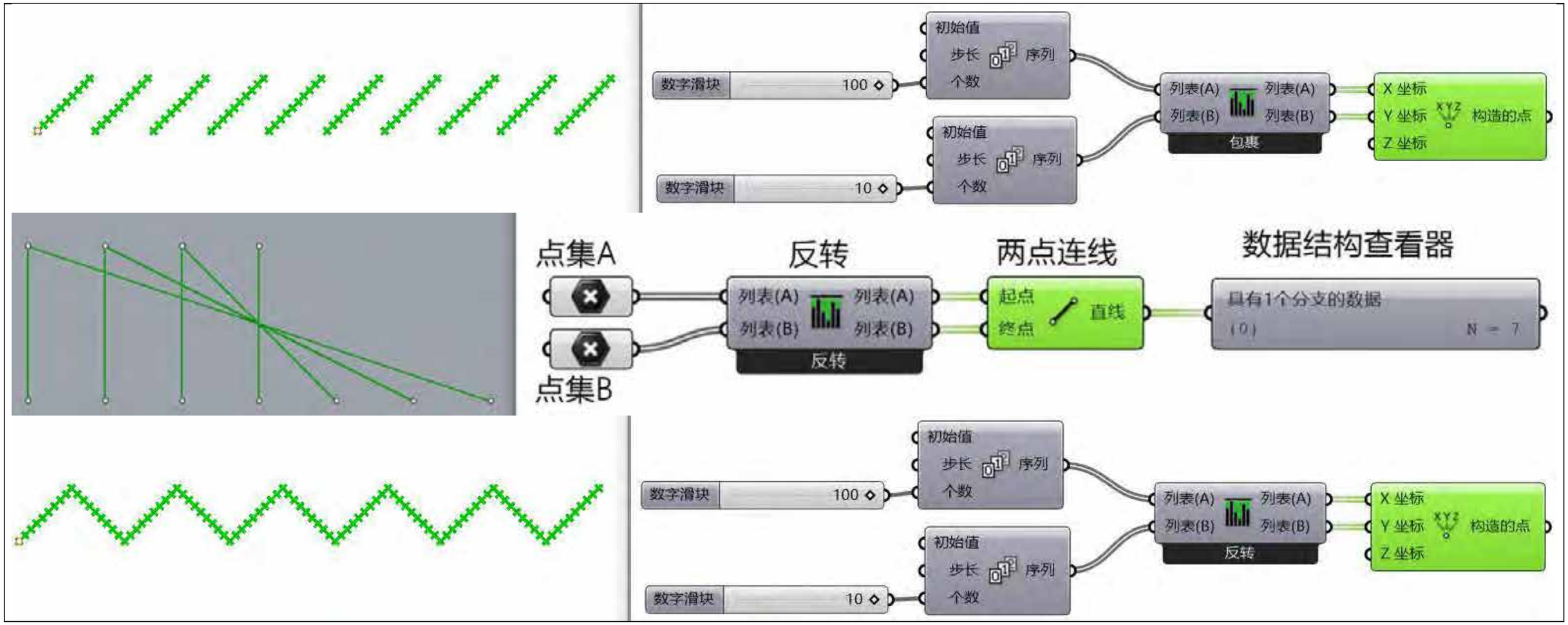
长列表连接 Longest List：

长排法，数量多的列表多出的数据全都和数据少的列表的最后一个数据运算。右键可以选择不同的算法，默认的是 repeat last，就是重复运算最后一个数据，而 GH 中的所有列表的运算方法都是默认 Longest list。

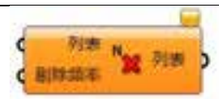




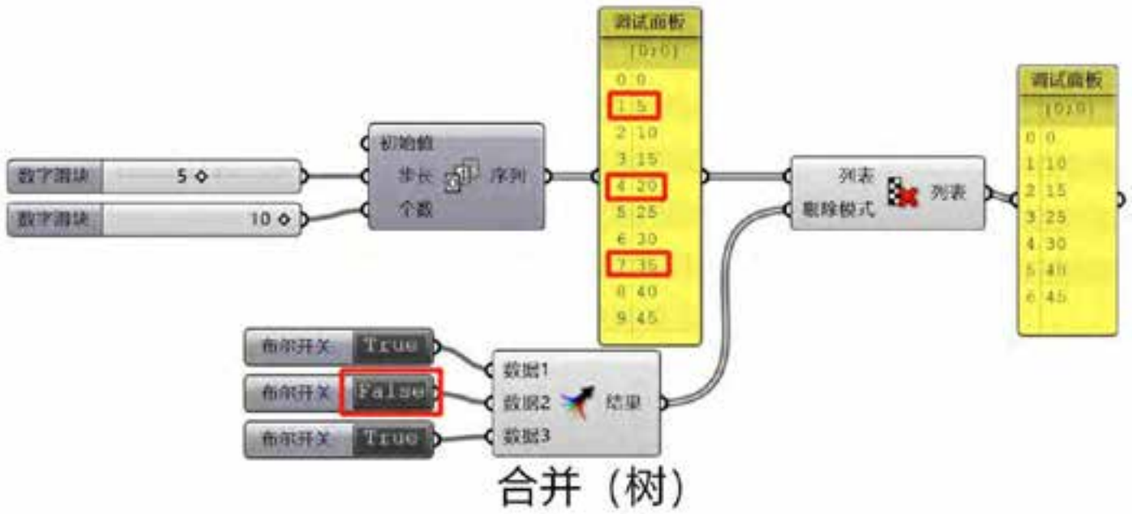

点集A
重复第一项
两点连线
数据结构查看器
列表(A)
列表(B)
列表(A)
列表(B)
尾部循环
起点
终点
直线
具有1个分支的数据
{0}
N = 7
点集B
初始值
步长
个数
序列
数字滑块
100
10
X 坐标
Y 坐标
Z 坐标
构造的点
点集A
插入值
两点连线
数据结构查看器
插入值
点集B
点集A
包裹
两点连线
数据结构查看器
包裹
点集B
按照布尔顺序轮流计算，
缺少的不计算


数字滑块
100
初始值
步长
个数
序列
数字滑块
10
初始值
步长
个数
序列
列表(A)
列表(B)
列表(A)
列表(B)
包裹
X 坐标
Y 坐标
Z 坐标
构造的点
点集A
点集B
反转
列表(A)
列表(B)
列表(A)
列表(B)
反转
两点连线
起点
终点
直线
数据结构查看器
具有1个分支的数据
{0}
N = 7
数字滑块
100
初始值
步长
个数
序列
数字滑块
10
初始值
步长
个数
序列
列表(A)
列表(B)
列表(A)
列表(B)
反转
X 坐标
Y 坐标
Z 坐标
构造的点

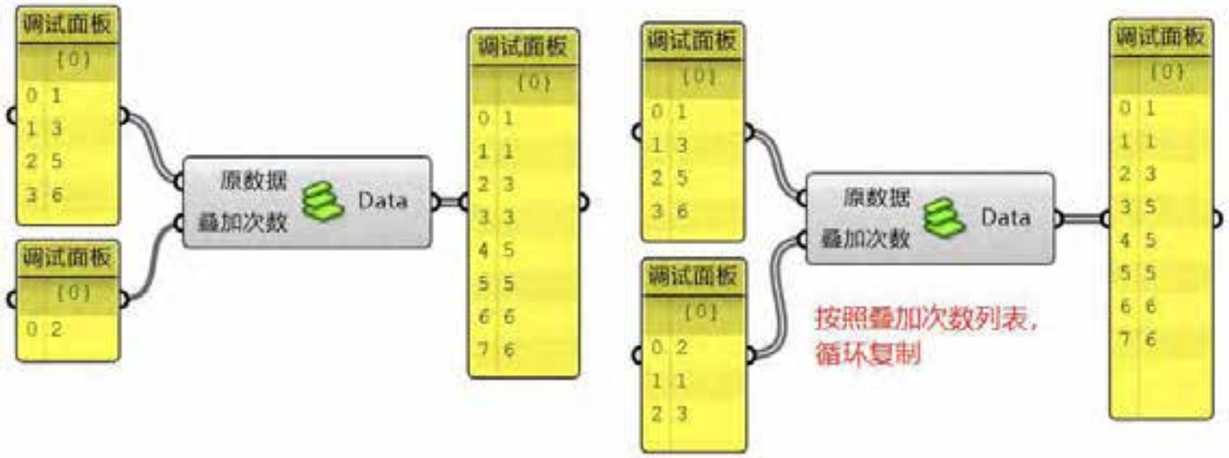
序列 Sequence-（集合）

剔除 N 倍 Cull Nth		循环删除设定剔除频率的数据，比如输入 2，则是删除第 2、4、6、8....个数字，对应的序列号是 1、3、5、7....，列表序号是从 0 开始	**剔除模式 Cull Pattern**		与数据分流类似，通过剔除模式输入的布尔值列表删除输入列表的数据，不同的是这里 false 对应的数据直接被删除（下方详解）
剔除索引 Cull Index		删除指定序号位置的数据，包裹输入布尔值，控制当序号大于列表总数时是否循环删除。（下方详解）	**随机删除 Random Reduce**		根据设定的数量随机删除数据，随机种子（数字）可以切换不同的随机删除方式。比如做一个随机删减的表皮（下方详解）
叠加数据 Stack Data		根据输入的叠加次数，对原数据对应序列号的数据逐个复制（下方详解）	**复制数据 Duplicate Data**		按设定的次数复制数据，排序为布尔值，控制列表内容是整体循环复制还是逐个按次数复制（下方详解）
字符序列 Char Sequence		字符池输入的字符依次排列组合，输出设定数量的组合（类似求密码组合）（下方详解）	**数字序列 Sequence**		根据符号端输入的自定义规则（默认是斐波那契数列 N=（N-1）+（N-2）），输出一组指定长度的数列（下方详解）
数字范围 Range		等分区间，输出等分点数值（下方详解）	**斐波那契 Fibonacci**		斐波那契数列（Fibonacci sequence）（下方详解）个数为计算次数 n，序列值 F（0）=A，F（1）=B，F(n+1)=F(n)+F(n-1)（n>=1，n∈N*），n+1 为序号
等差数列 Series		根据设定初始值（默认为 0）、步长（默认为 1）、个数（默认为 10），生成一组等差数列。默认生成的数列为 0 到 9 十个数字	**长度复制 Repeat Data**		循环复制列表数据，并根据设定的列表长度输出一组数据。比如数据输入 123，长度为 7，则输出 1231231 七个数字
伪随机数 Random		根据设定的数字范围，输出一组设定数量的随机数字，随机种子可以控制输出不同的数字组合。右键可以选择 Integers Numbers 输出整数	**随机打乱（震荡）Jitter**		将数列的数据打乱，打乱强度为 0-1 之间的数字（0 为不打乱，1 为完全打乱），随机种子为数值，控制不同变化。可用于生成材质颜色随机变化的表皮（下方详解）

剔除模式 Cull Pattern： 通过剔除模式输入的布尔值列表删除输入列表的数据

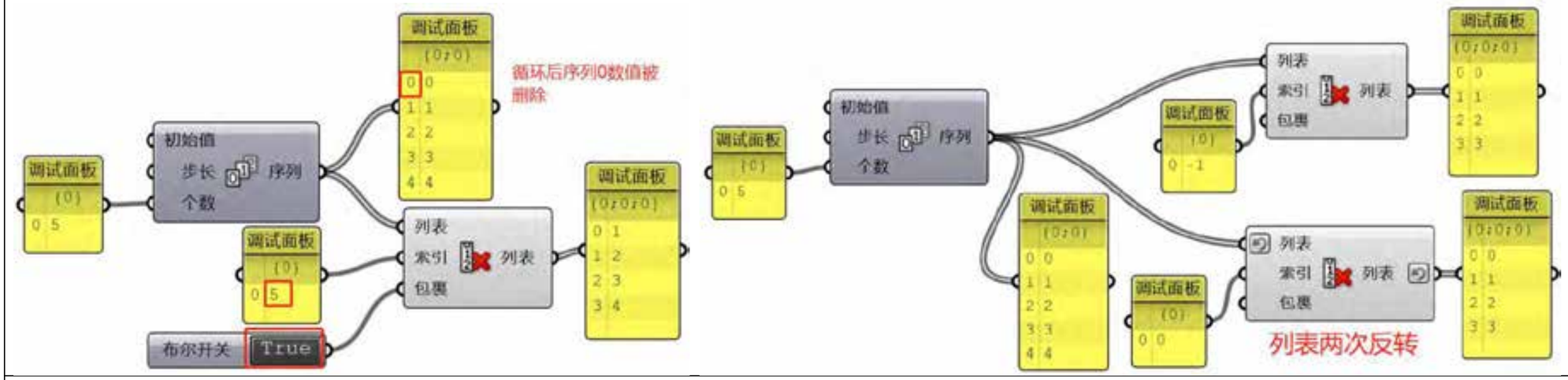


叠加数据 Stack Data：

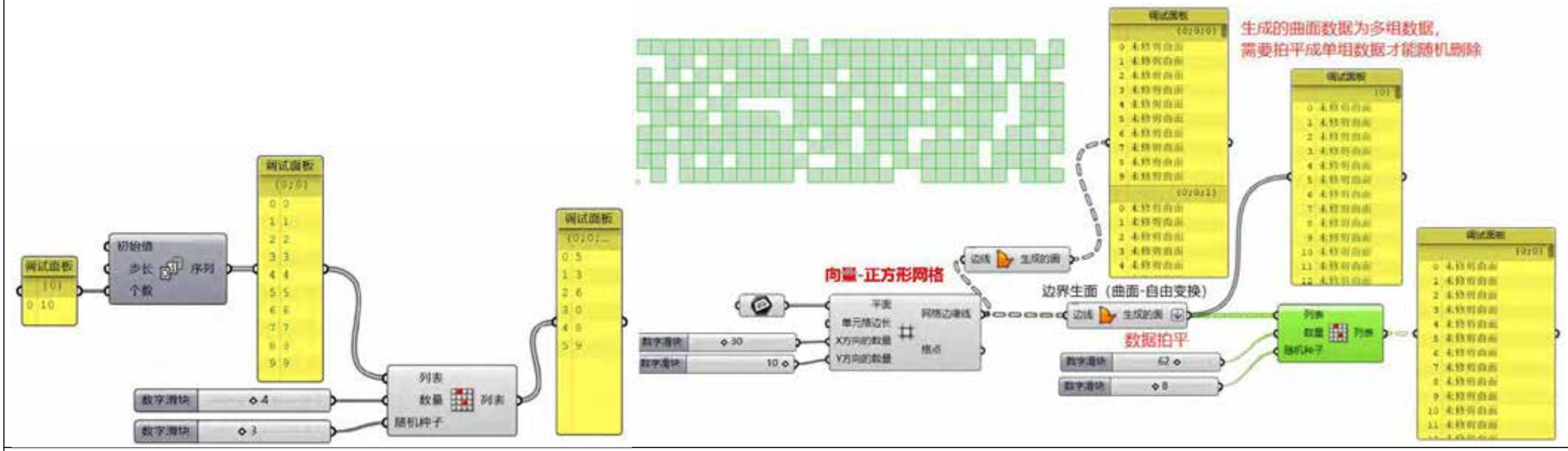


随机删除 Random Reduce
复制数据 Duplicate Data

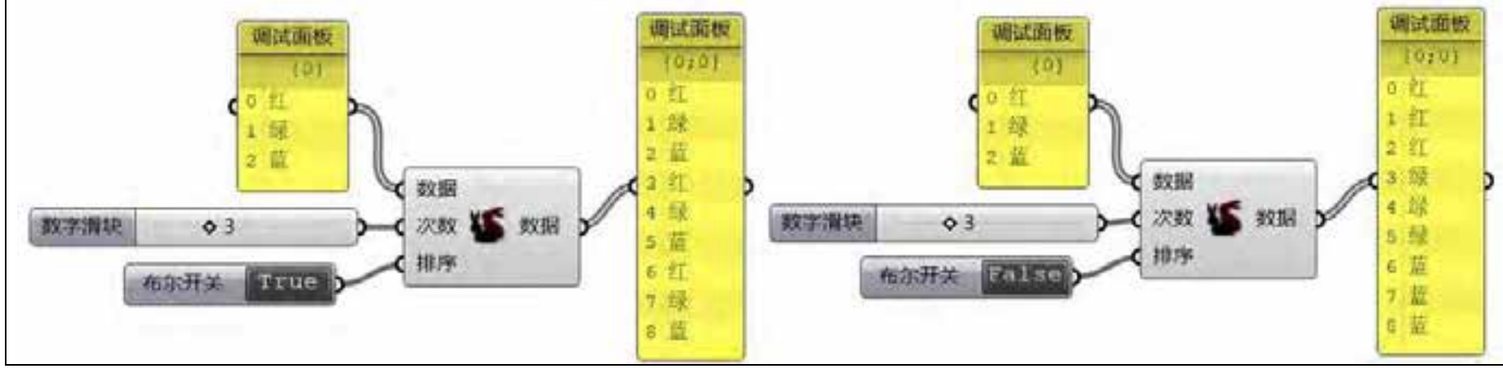
剔除索引 Cull Index： 删除指定序号位置的数据，包裹输入布尔值，控制当序号大于列表总数时是否循环删除。如果想删除数列根据•最后一个数的话有以下两种方法。



随机删除 Random Reduce： 根据设定的数量随机删除数据，随机种子（数字）可以切换不同的随机删除方式。（右侧为随机删减的表皮）

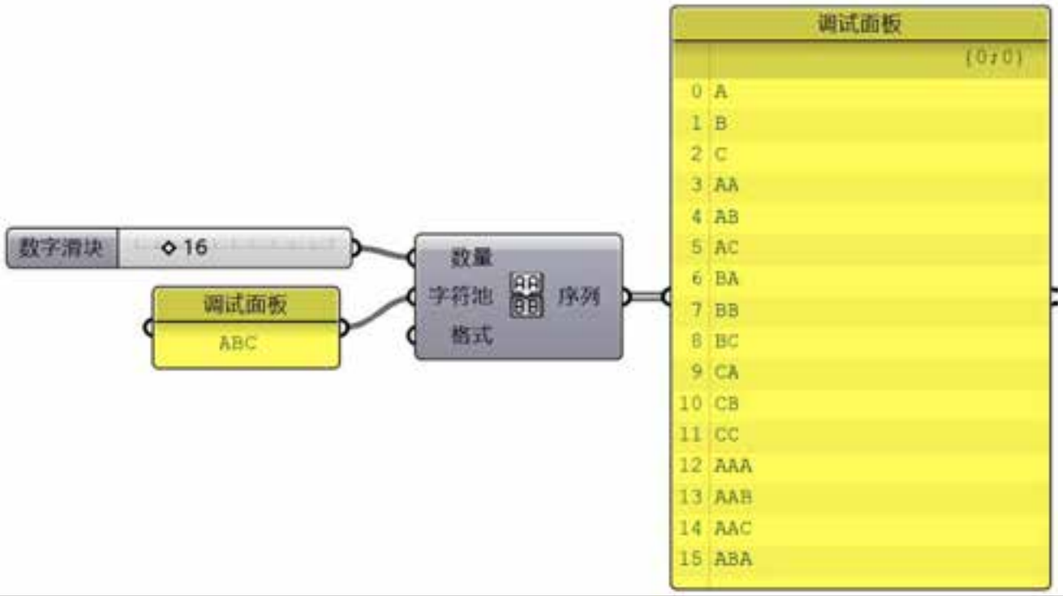


复制数据 Duplicate Data： 按设定的次数复制数据，排序为布尔值，控制列表内容是整体循环复制还是逐个按次数复制

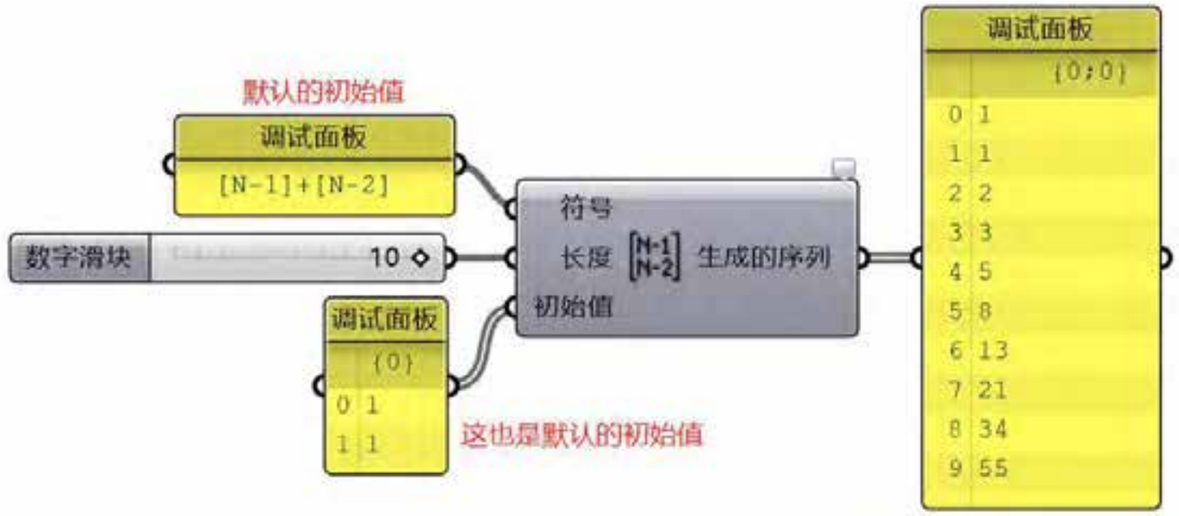


字符序列 Char Sequence
数字序列 Sequence
数字范围 Range
斐波那契 Fibonacci
随机打乱（震荡）Jitter

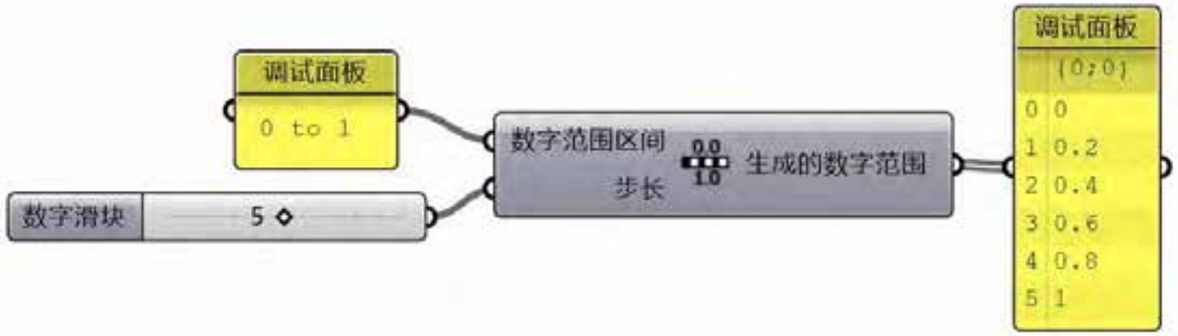
字符序列 Char Sequence：字符池输入的字符依次排列组合，输出设定数量的组合（类似求密码组合）



数字序列 Sequence：根据符号端输入的自定义规则（默认是斐波那契数列 N=（N-1）+（N-2）），输出一组指定长度的数列

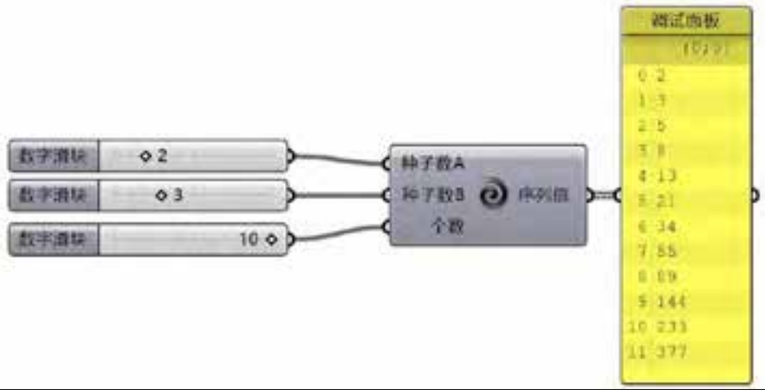


数字范围 Range：等分区间，输出等分点数值

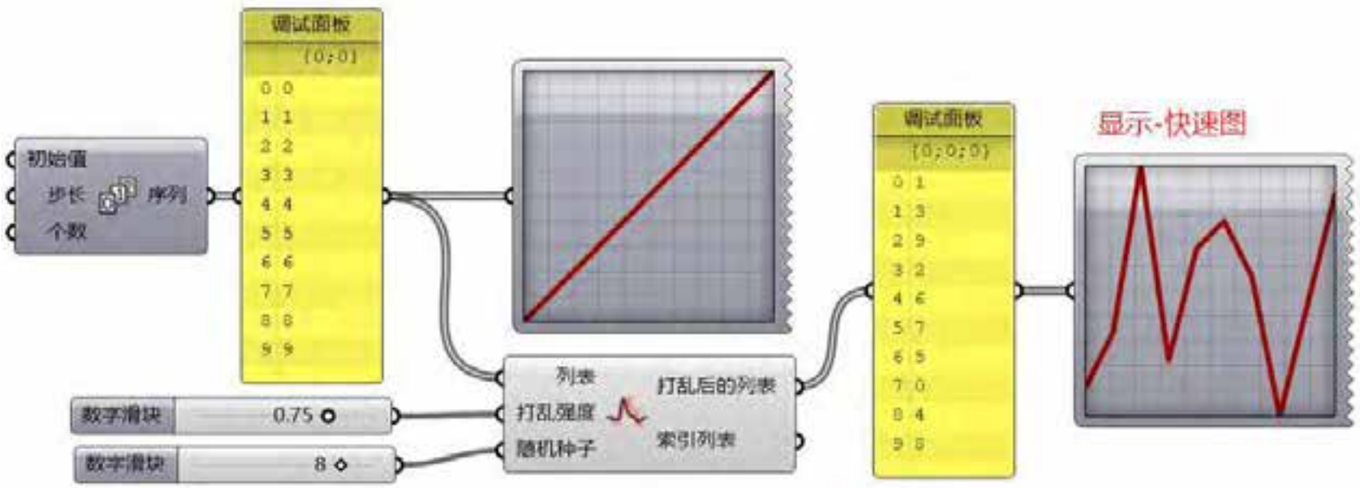


斐波那契 Fibonacci：

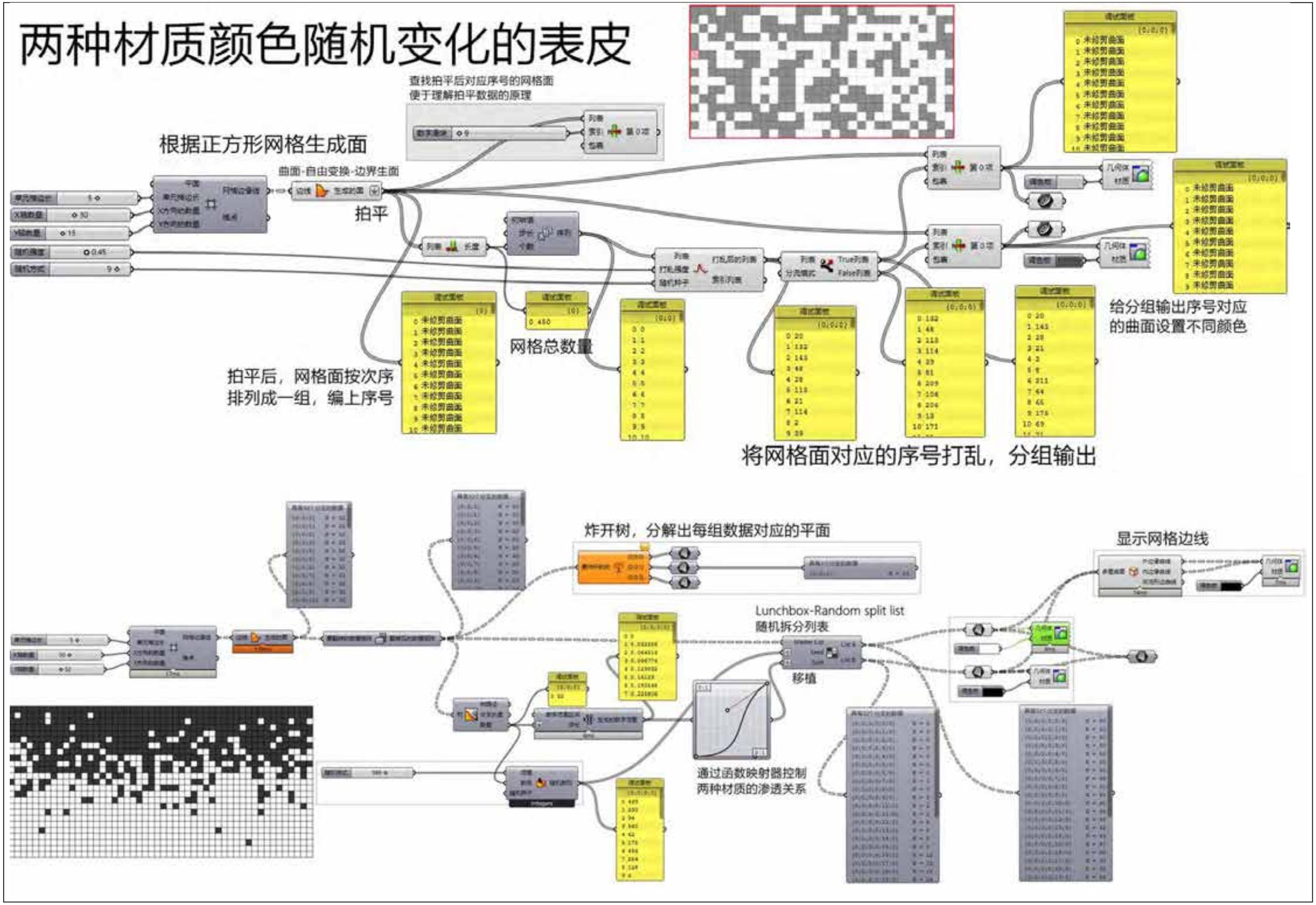
斐波纳契数列，又称黄金分割数列。在数学上，斐波纳契数列以如下被以递推的方法定义：F(1)=1，F(2)=1, F(n)=F(n-1)+F(n-2)（n>=3，n∈N*），当 n 值无限大时，F(n)/ F(n-1)就越接近黄金比 0.618。



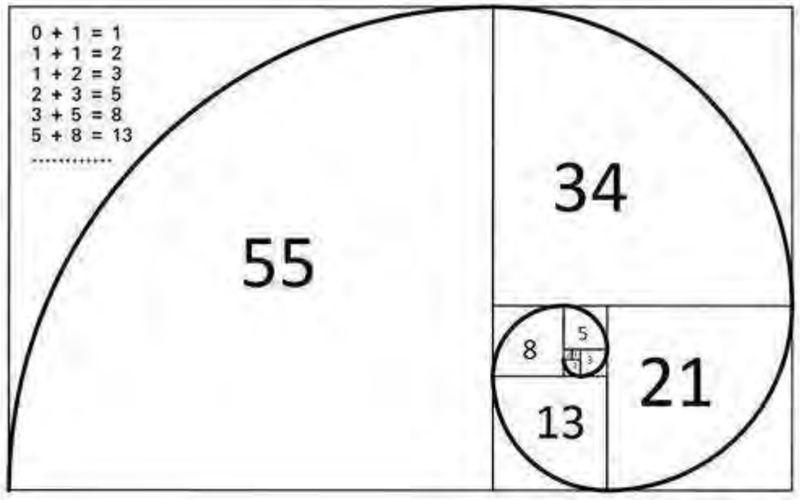
随机打乱（震荡）Jitter：将数列的数据打乱，打乱强度为 0-1 之间的数字（0 为不打乱，1 为完全打乱），随机种子为数值，控制不同变化



两种材质颜色随机变化的表皮



斐波那契数列（Fibonacci sequence），又称黄金分割数列，因数学家莱昂纳多·斐波那契（Leonardo Fibonacci）以兔子繁殖为例子而引入，故又称为“兔子数列”，指的是这样一个数列：1、1、2、3、5、8、13、21、34、……在数学上，斐波那契数列以如下被以递推的方法定义：$F(0)=0$，$F(1)=1$, $F(n)=F(n-1)+F(n-2)$（$n \geq 2$，$n \in N^*$）在现代物理、准晶体结构、化学等领域，斐波纳契数列都有直接的应用，为此，美国数学会从 1963 年起出版了以《斐波纳契数列季刊》为名的一份数学杂志，用于专门刊载这方面的研究成果。



文本 Text-（集合）

插入文本 Text Join

文字组合，讲多个字符或者文字组合成一个文字。可以设置组合的分隔符。

文本分割 Text Split

将文字在指定字符处断开。

文本打断 Characters

拆解文字为单个的字符，并返回每个字符的 Unicode 值。

Unicode(统一码、万国码、单一码)是计算机科学领域里的一项业界标准，包括字符集、编码方案等。

文本长度 Text Length

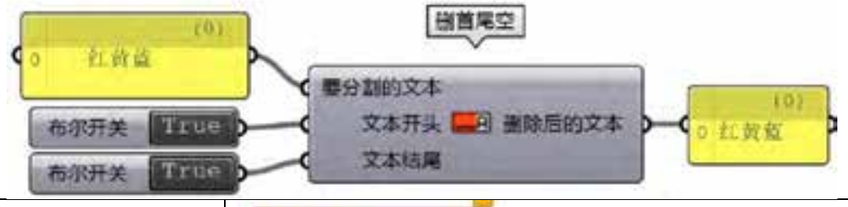
测量文字及字符的数量

连接文本 Concatenate

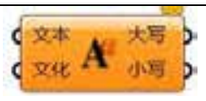
合并字符，输入端可以增加。

删首尾空 Text Trim

用来控制删除文字首尾的空格。



字大小写 Text Case



英文大小写转换

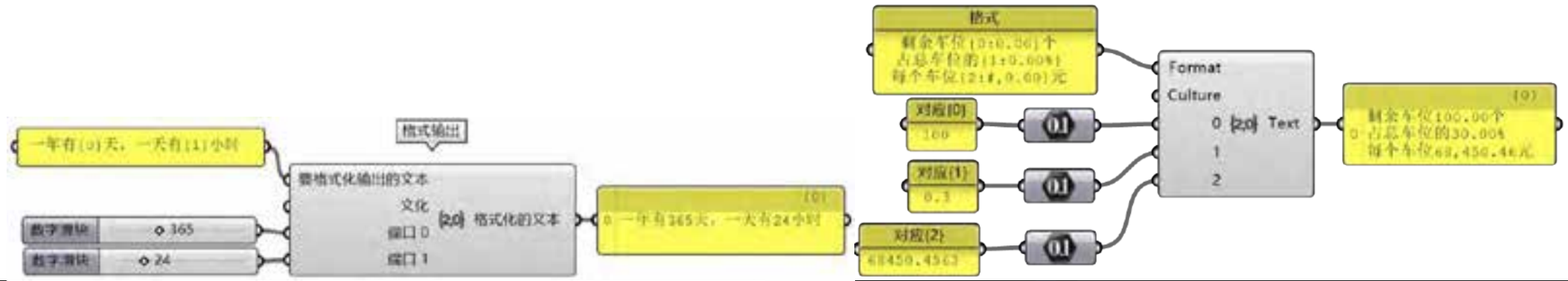
文本切片 Text Fragment

抽取文字中指定起始位置之指定长度的字符。

格式输出 Format

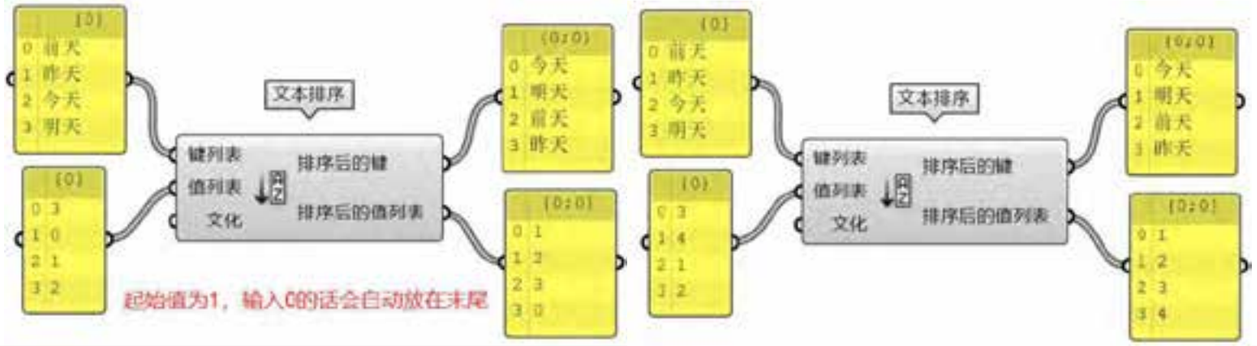
可以参考 python 中的函数用法，用输入端的数据代替掉 Format 端中｛｝内的内容。输入端数量　可以增加，

注意：Format 中的｛0｝一定要用英文输入法输入，不然会出错。

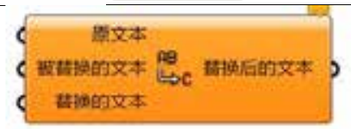


文本排序 Sort Text

根据值列表输入对应文字的编号，对输入的文字（键列表）重新排序。（经测试起始值为 1）



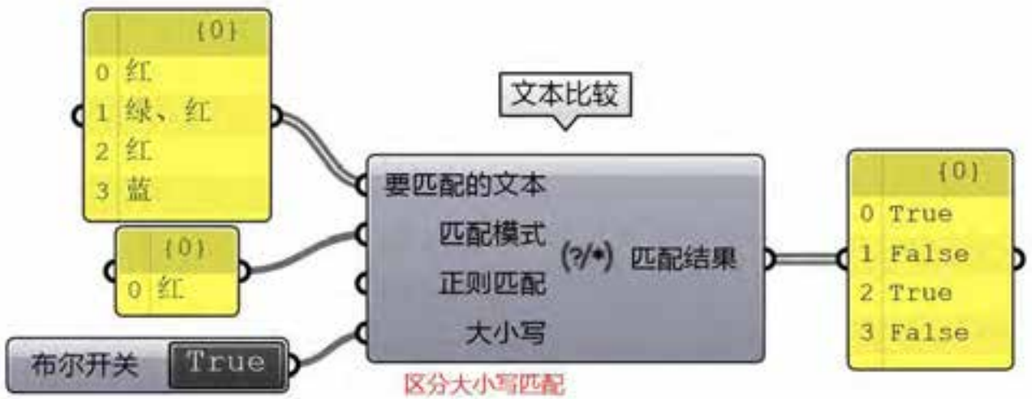
文本替换 Replace Text



替换指定文字或符号。

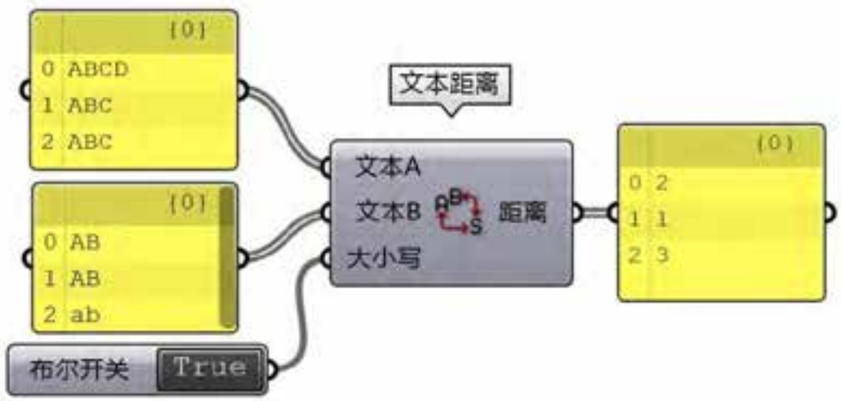
文本比较 Match Text

查找文字数据是否与指定的字符一致，输出布尔值

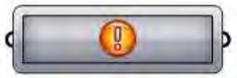


文本距离 Text Distance

计算两个字符串之间的 Levenshtein Distance，Levenshtein Distance 算法，又叫 Edit Distance 算法，是指两个字符串之间，由一个转成另一个所需的最少编辑操作次数。许可的编辑操作包括将一个字符替换成另一个字符，插入一个字符，删除一个字符。一般来说，编辑距离越小，两个串的相似度越大。

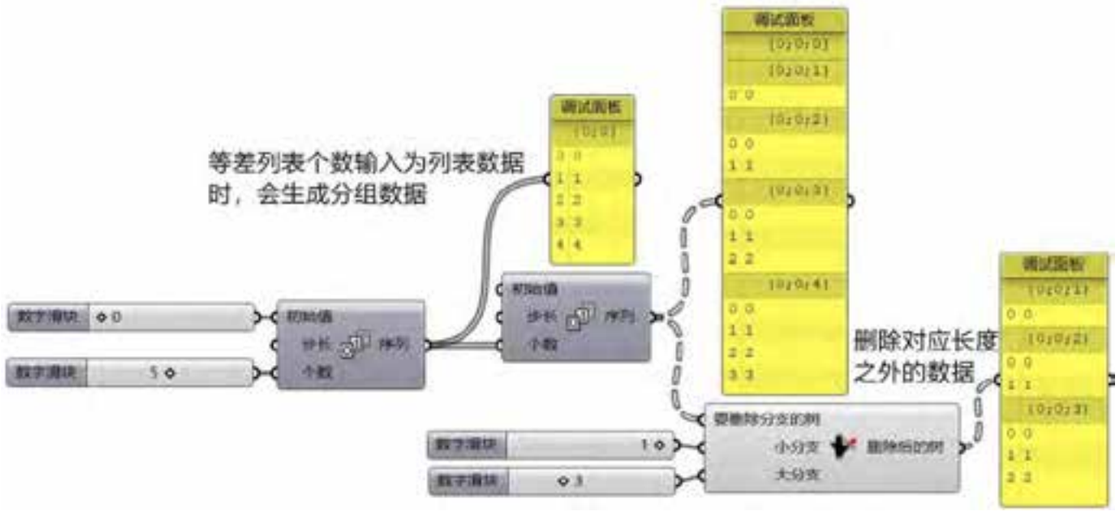


树 Tree-（集合）

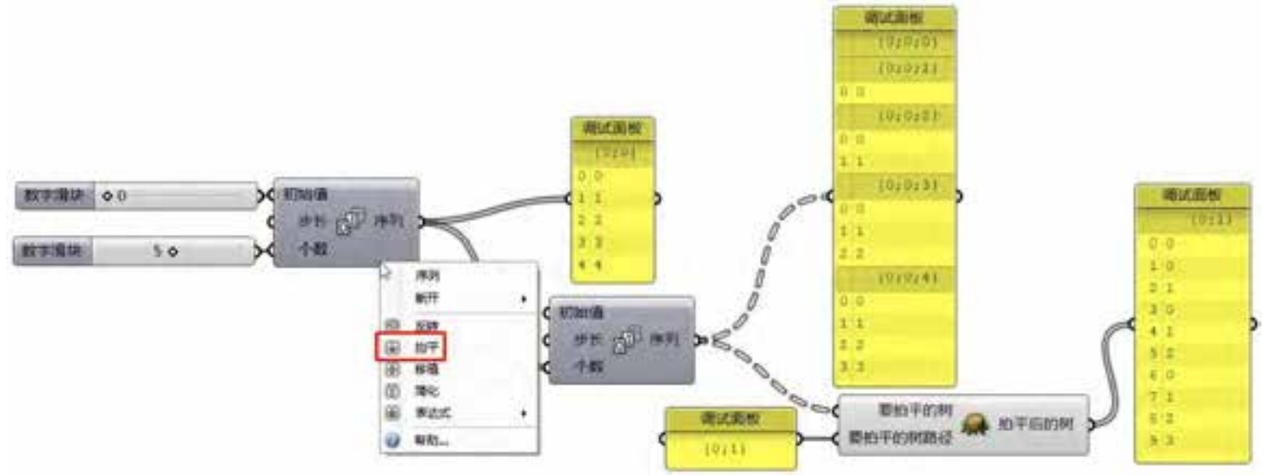
删除分支 Prune Tree		删除指定列表长度之外的数据，只保留设定长度部分（下方详解）	**反拍平 Unflatten Tree**		把拍平的数据，重新拍回去，不过要输入之前的数据结构作为拍平参数（下方详解）
拍平 Flatten Tree		拍平数据结构，去掉所有的数据结构，把数据放在一个组内。由于使用频繁，内置在很多运算器的输入和输出端（下方详解）	**末端拍平 Trim Tree**		修剪数据结构，根据指定的深度 Depth 来修剪数据结构（下方详解）
树统计 Tree Statistics		提取数据结构的数据路径名称（Paths），每一支数据路径内数据的分支长度（Length），以及总数据支数量（count）。（下方详解）	**清理树 Clean Tree**		根据设定的布尔值来决定是否删除树结构中的空项（null）、无效项（invalid）或空分支（下方详解）
移植 Graft Tree		将现有数据结构中的每个数据再单独成组，会增加一级数据结构。（下方详解）	**简化 Simplify tree**		简化数据结构，把冗余的数据结构路径名称简化，运算器输入或输出端右键中的简化作用一致。（下方详解）
合并 Merge		将两组或多组数据合并成一组，要注意数据的结构路径名称是否一致（下方详解）	**数据纠缠 Entwine**		拍平并按照输入端指定的路径名称组合输出数据。默认会拍平所有输入端的数据。（下方详解）
炸开树 Explode Tree		炸开数据结构，将树形数据分组输出到单组数据（下方详解）	**翻转矩阵 Flip Matrix**		翻转数据结构（下方详解）
偏移路径 Shift Paths		偏移数据结构（下方详解）	**分割树 Split Tree**		分割数据结构，将指定路径名称的数据支和其他数据支分离出来。（下方详解）
匹配 Match Tree		将一组数据结构的名称匹配成另一组。两组的数据结构数量应当一致，否则会报错。（下方详解）	**路径映射器 Path Mapper**		数据结构编辑器，可以自定义修改映射路径名称，或者右键选择预设的映射方式（下方详解）
定向输出 Stream Gate		数据流闸门，通过设定端口数值（0、1、2...），控制数据流分配到对应的输出端口（下方详解）	**数据流过滤 Stream Filter**		数据流过滤器，根据索引端输入的数值控制输出对应的输入端的数据流（下方详解）
查找树 Tree Item		根据指定的分支路径 path 和索引（序号）index 抽取输入端树形数据中指定序号的数据（下方详解）	**树分支 Tree Branch**		根据输入的分支路径名称抽取指定分支的数据。（下方详解）
相对差异 Relative Item		根据路径格式{A;B;C}(i)对应位置设定的偏移值，偏移数据结构（下方详解）	**相对差异（多项） Relative Items**		与相对差异的区别在于，相对差异多项是对两组数据进行处理（下方详解）
分解路径 Deconstruct Path		把路径名称拆解成数列（下方详解）	**替换路径 Replace Paths**		用替换关键字输入的自定义路径名称，替换掉查找关键字输入的路径名称，替换后如果存在两个相同路径名称，则两组数据会合并放在一起（下方详解）
构造分支路径 Construct Path		把分支路径下的数据组合成路径名称（下方详解）	**路径比较 Path Compare**		根据自定义路径名称（路径掩码），查找要比较的路径中有没有名称一致的，返回布尔值（下方详解）

删除分支 Prune Tree
拍平 Flatten Tree
反拍平 Unflatten Tree

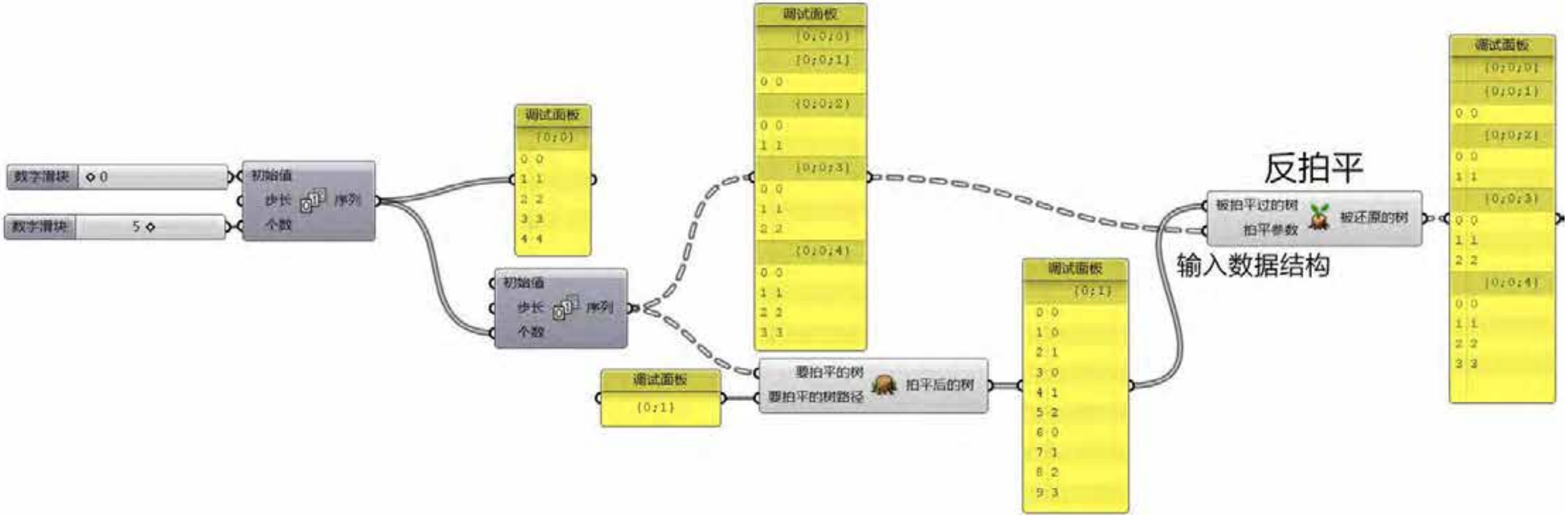
删除分支 Prune Tree：删除指定列表长度之外的数据，只保留设定长度部分



拍平 Flatten Tree：拍平数据结构，砍掉所有的所有的数据结构，放到一个组内，路径 path 端可以设定拍平后数据结构名称。打个比方有五个房间，每个房间人数从 0-4 都不同，拍平后就是把所有人放到一个房间内。由于这个运算器太过常用，所以被内置到几乎所有输出输入端的右键菜单里了。



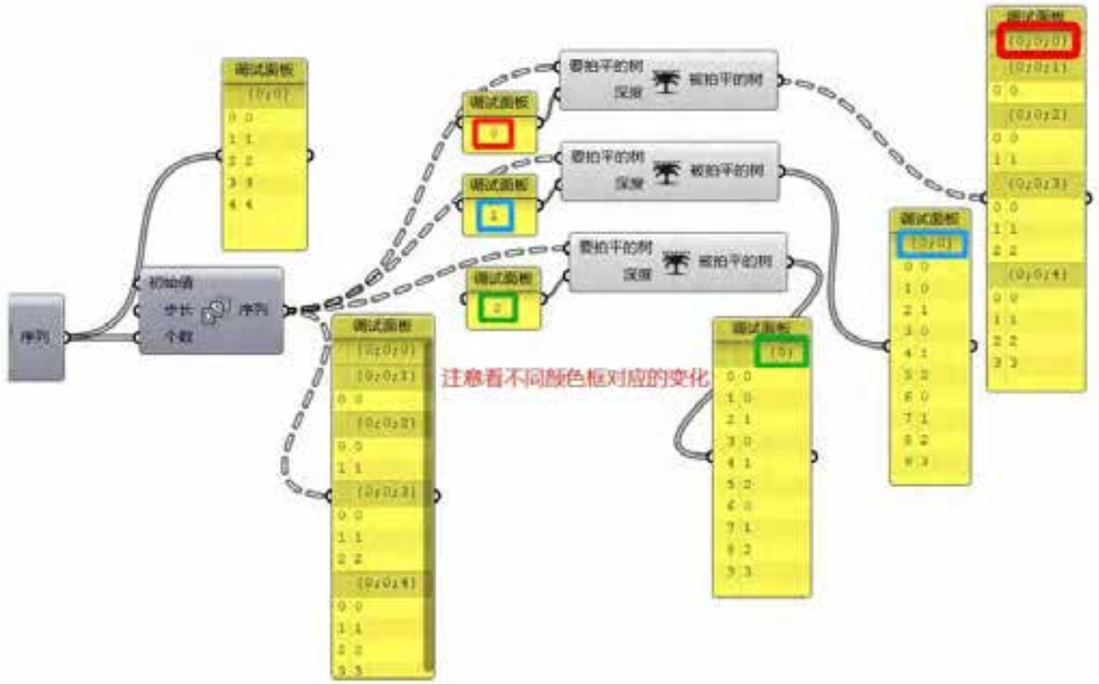
反拍平 Unflatten Tree：把拍平的数据，重新拍回去，不过要输入之前的数据结构作为拍平参数



末端拍平 Trim Tree
清理树 Clean Tree
树统计 Tree Statistics

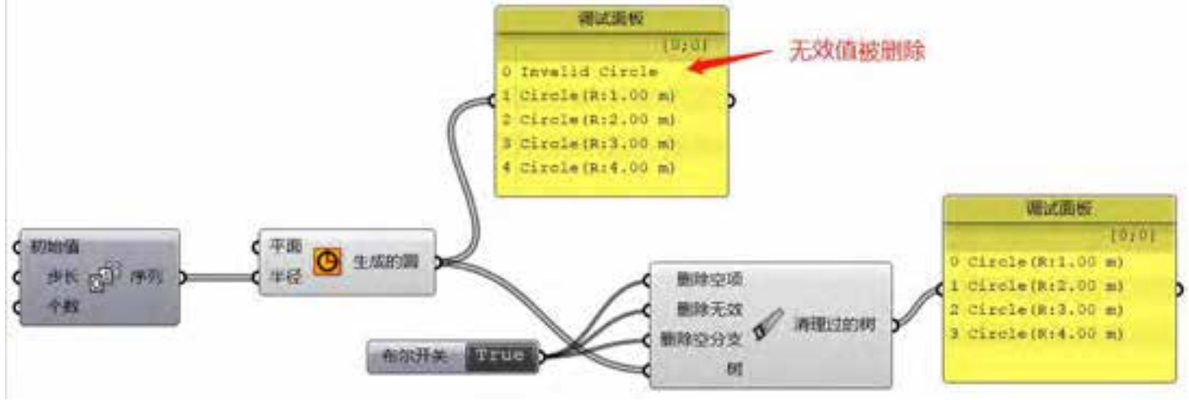
末端拍平 Trim Tree：

修剪数据结构，根据指定的深度 Depth 来修剪数据结构（从末端按照指定深度，减去对应的数据结构数）。



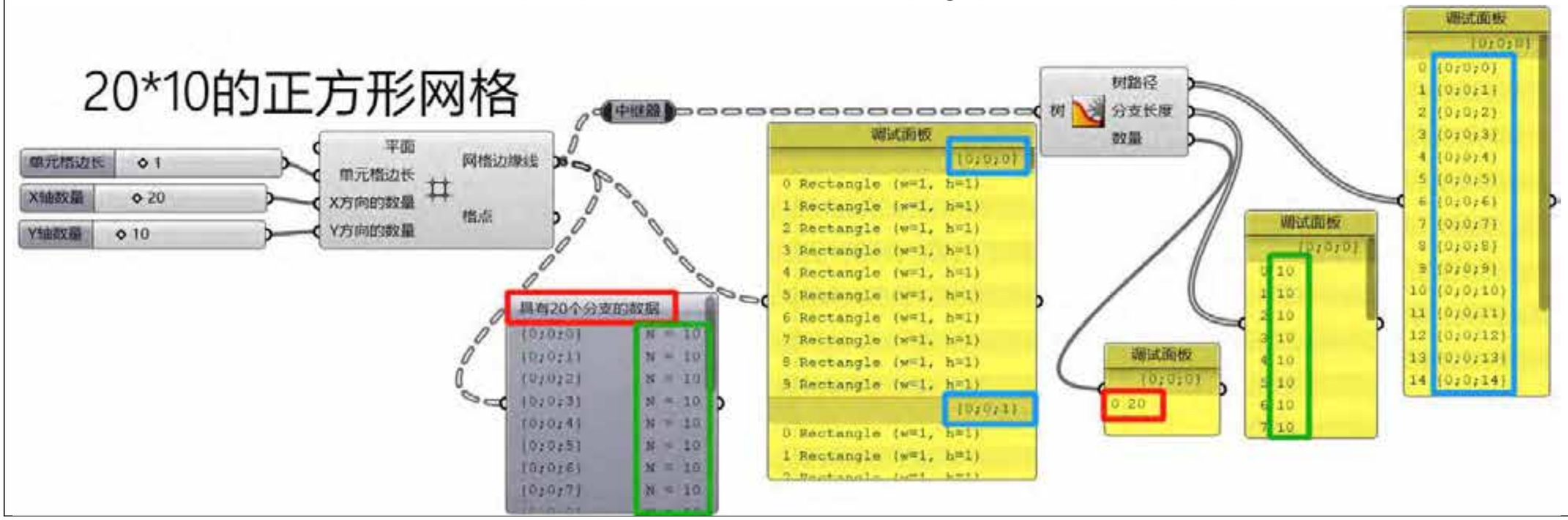
清理树 Clean Tree：

根据设定的布尔值来决定是否删除树结构中的空项（null）、无效项（invalid）或空分支

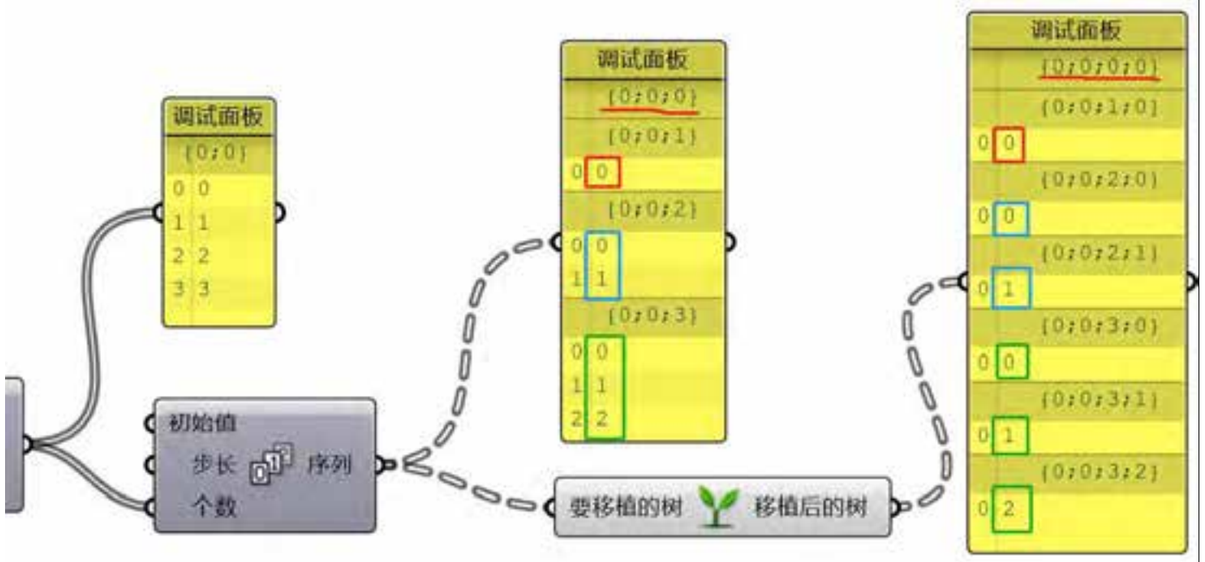


树统计 Tree Statistics：

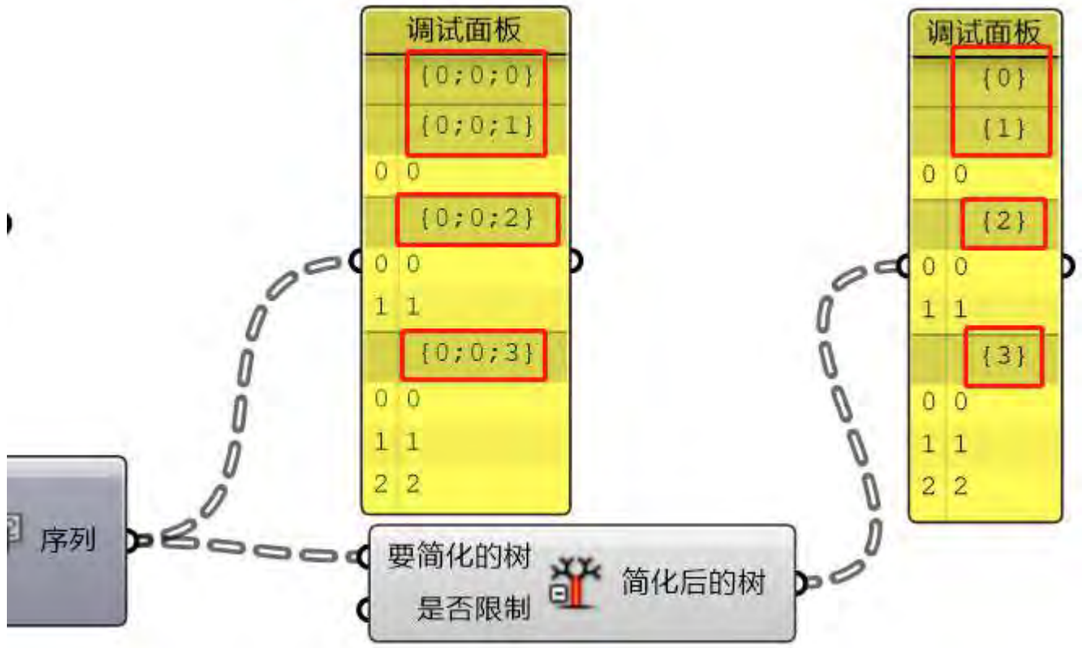
提取数据结构的数据路径名称（Paths），每一支数据路径内数据的分支长度（Length），以及总数据支数量（count）。



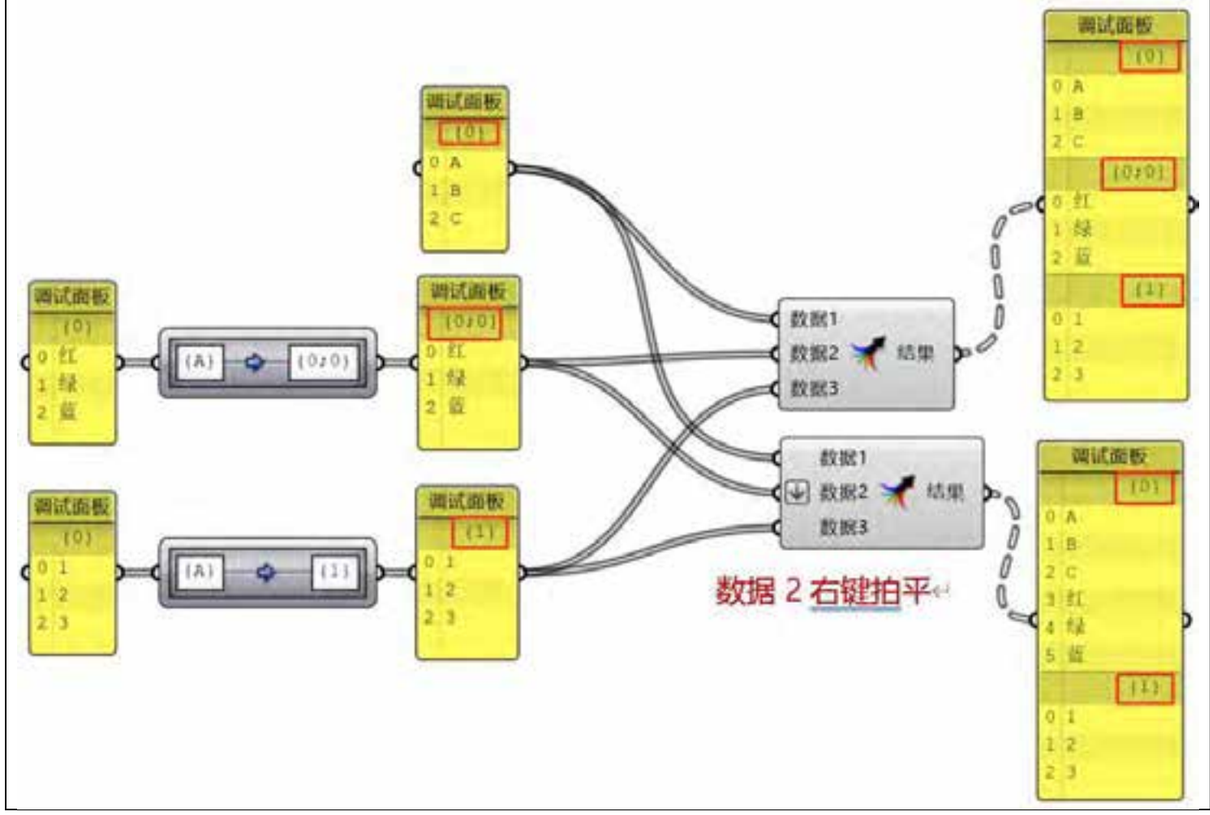
移植 Graft Tree： 将现有数据结构中的每个数据再单独成组，会增加一级数据结构。可以打个比方，原本有 4 个房间里住着不同数量的人，移植后，这四个房间的人依次每人分了一个单间。



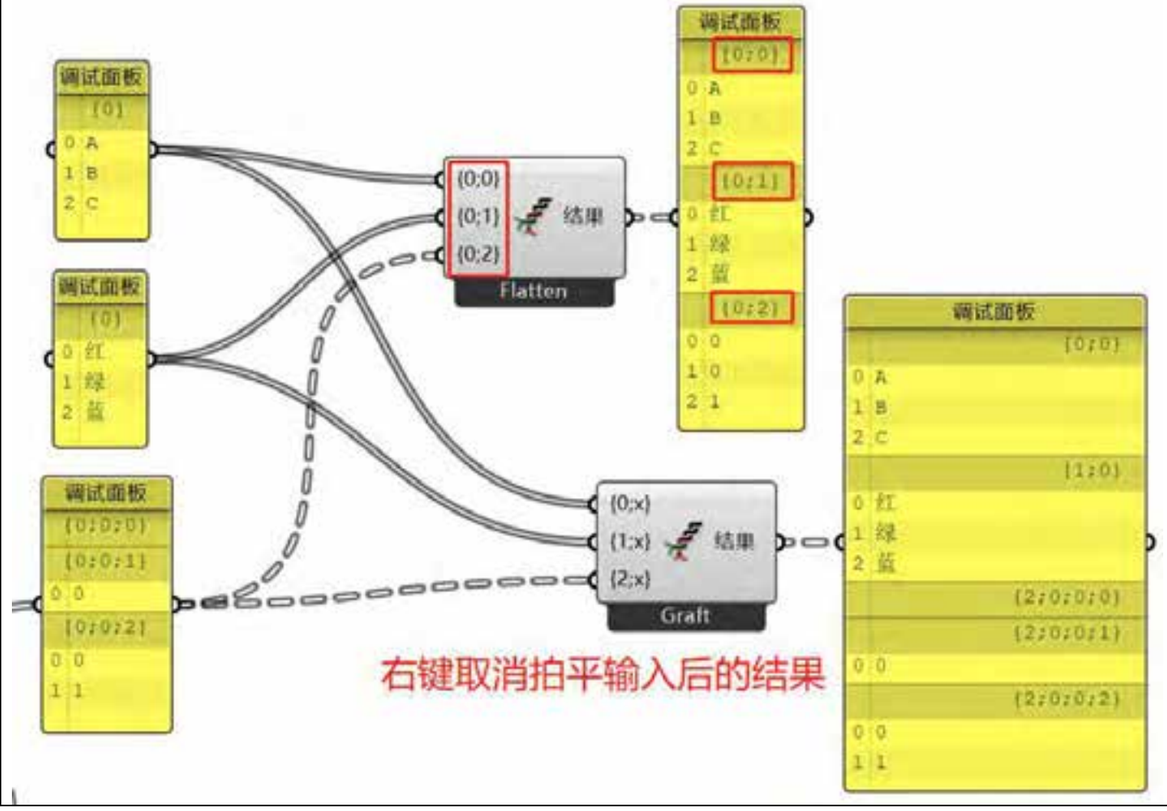
简化 Simplify tree： 简化数据结构，把冗余的数据结构路径名称简化。



合并 Merge： 将两组或多组数据合并成一组，要注意数据的结构路径名称是否一致。



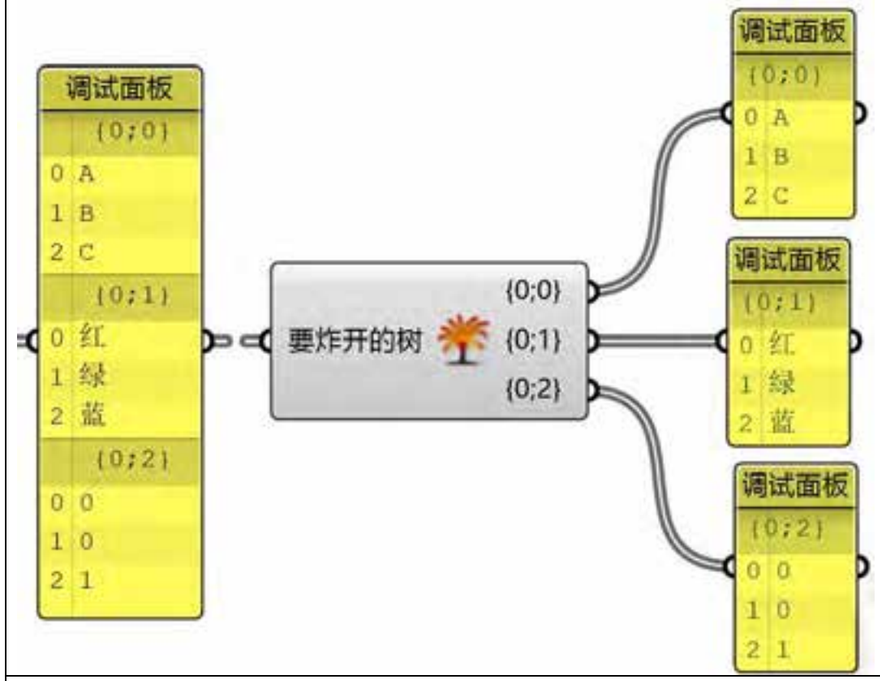
数据纠缠 Entwine： 将多组数据按照设定的路径组合在一起。默认会拍平所有输入端的数据。可以右键取消拍平。这样的话，数据纠缠运算器就会在输入端数据的基础上增加数据结构。



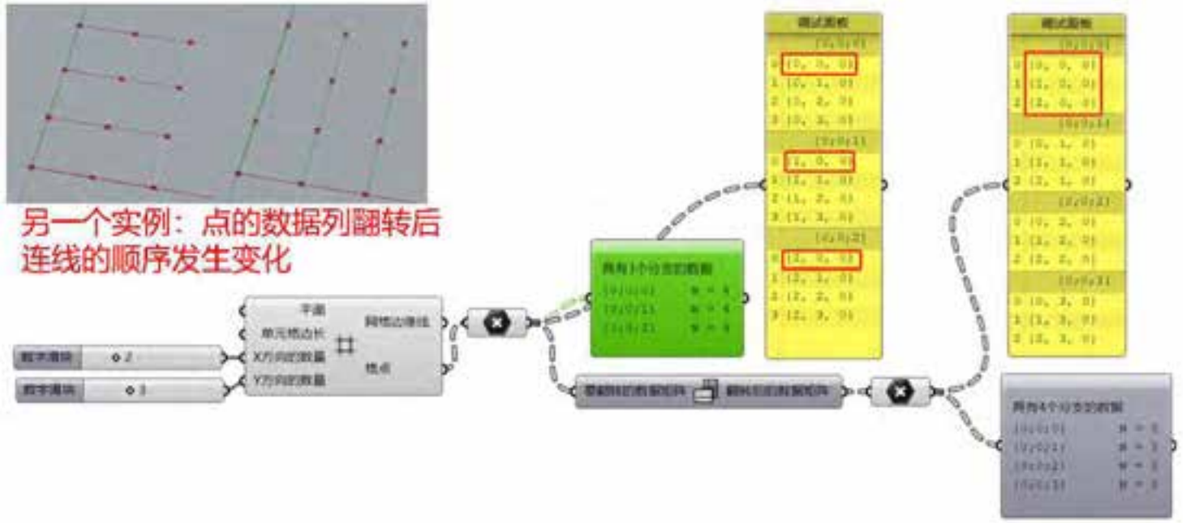
炸开树 Explode Tree
翻转矩阵 Flip Matrix
偏移路径 Shift Paths

炸开树 Explode Tree：将树形数据分组输出到单组数据。

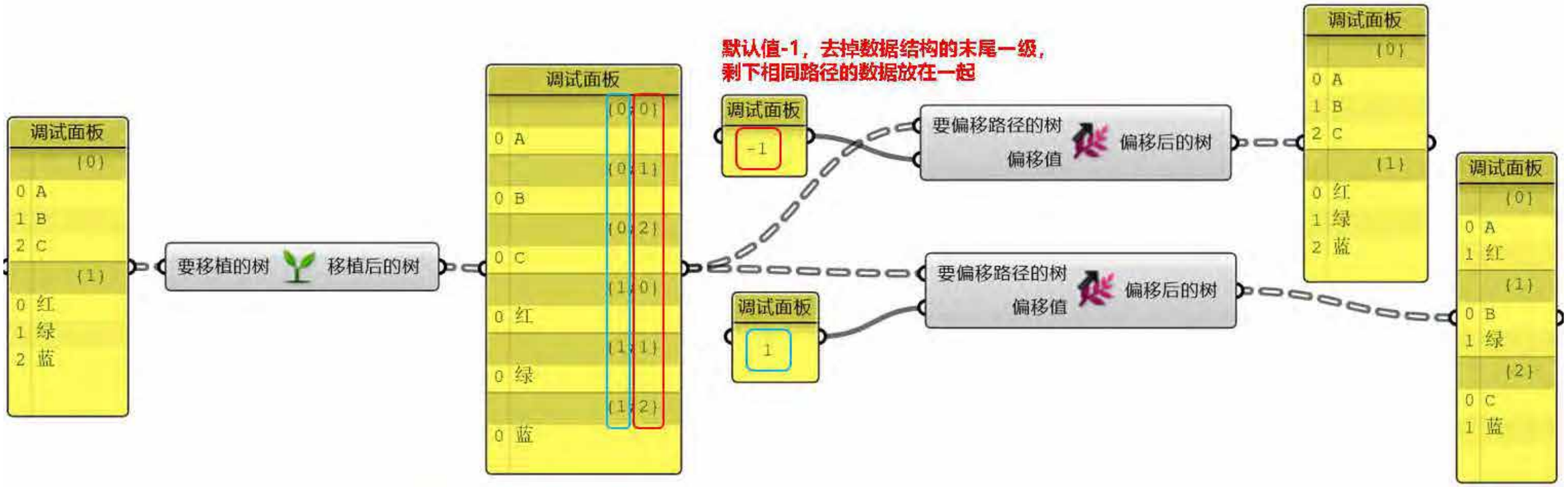
如果输出端数量大于输入数据的结构数量，则可以右键点击 match output 减去多出的输出端



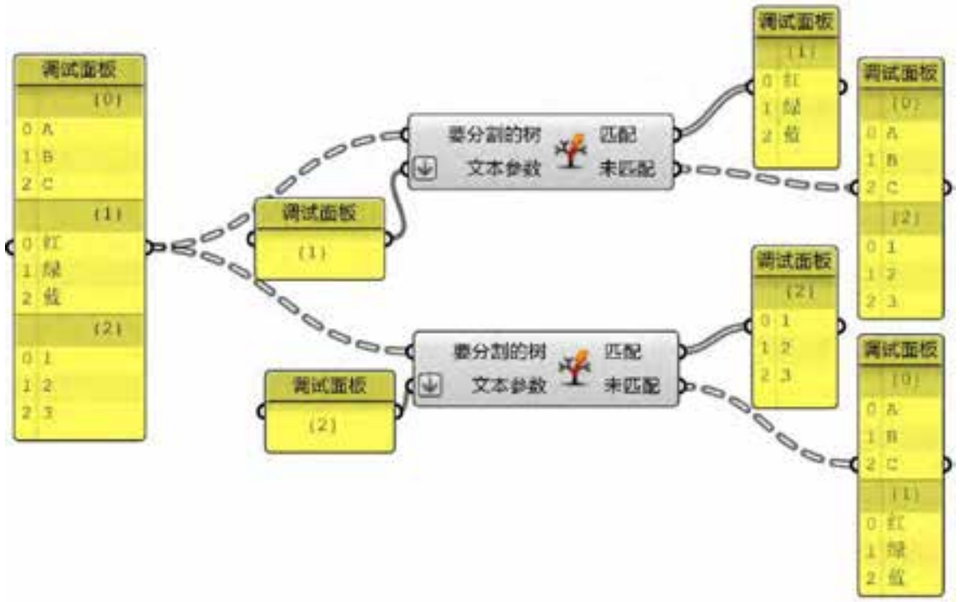
翻转矩阵 Flip Matrix：翻转数据结构。如下图，把原来的 3 大组，每组 4 个数据，翻转成 4 大组，每组 3 个数据。

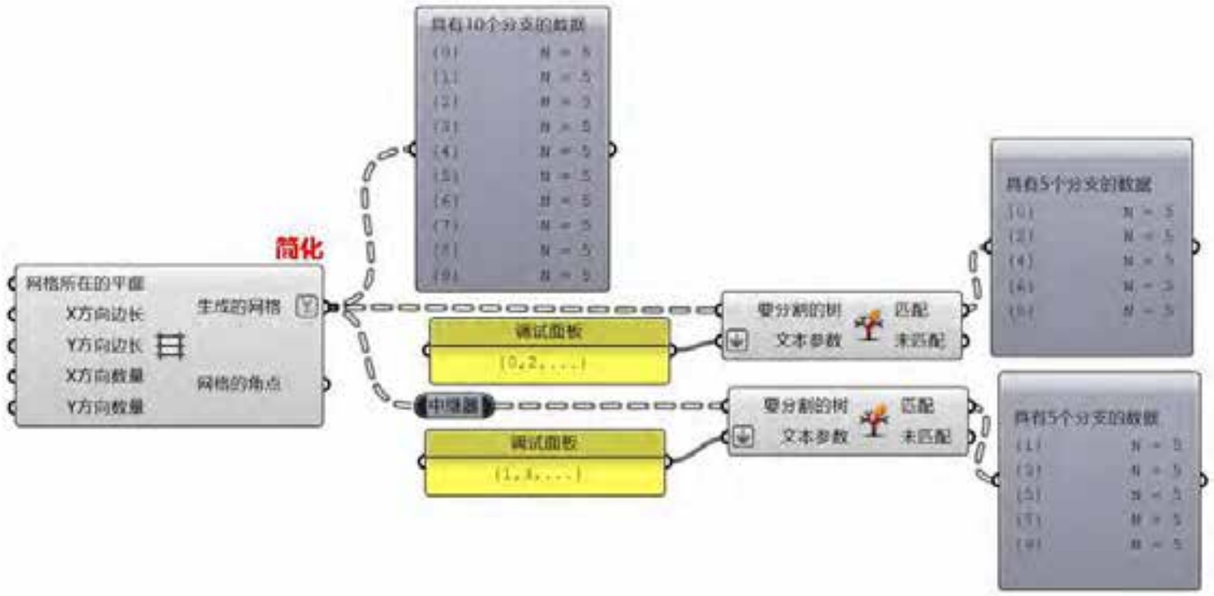


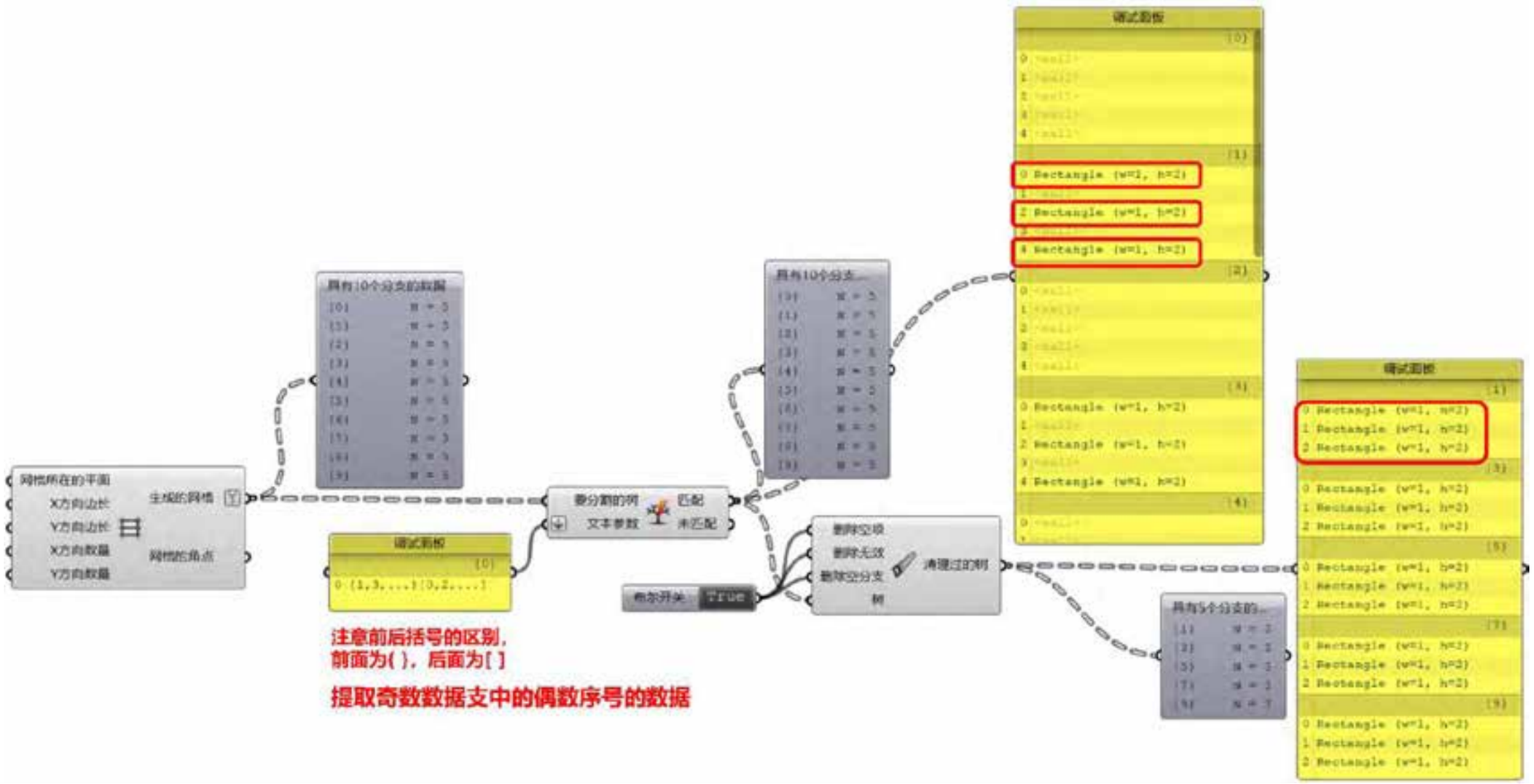
偏移路径 Shift Paths：偏移数据结构，偏移值为 1 时，去掉第一级数据结构，剩下所有相同路径的数据放到一起；偏移值设置为-1（根据数据结构级数循环原理）时，砍掉最后一级数据结构，。偏移路径与移植的作用正好相反，一个是减去数据结构，一个是增加数据结构。

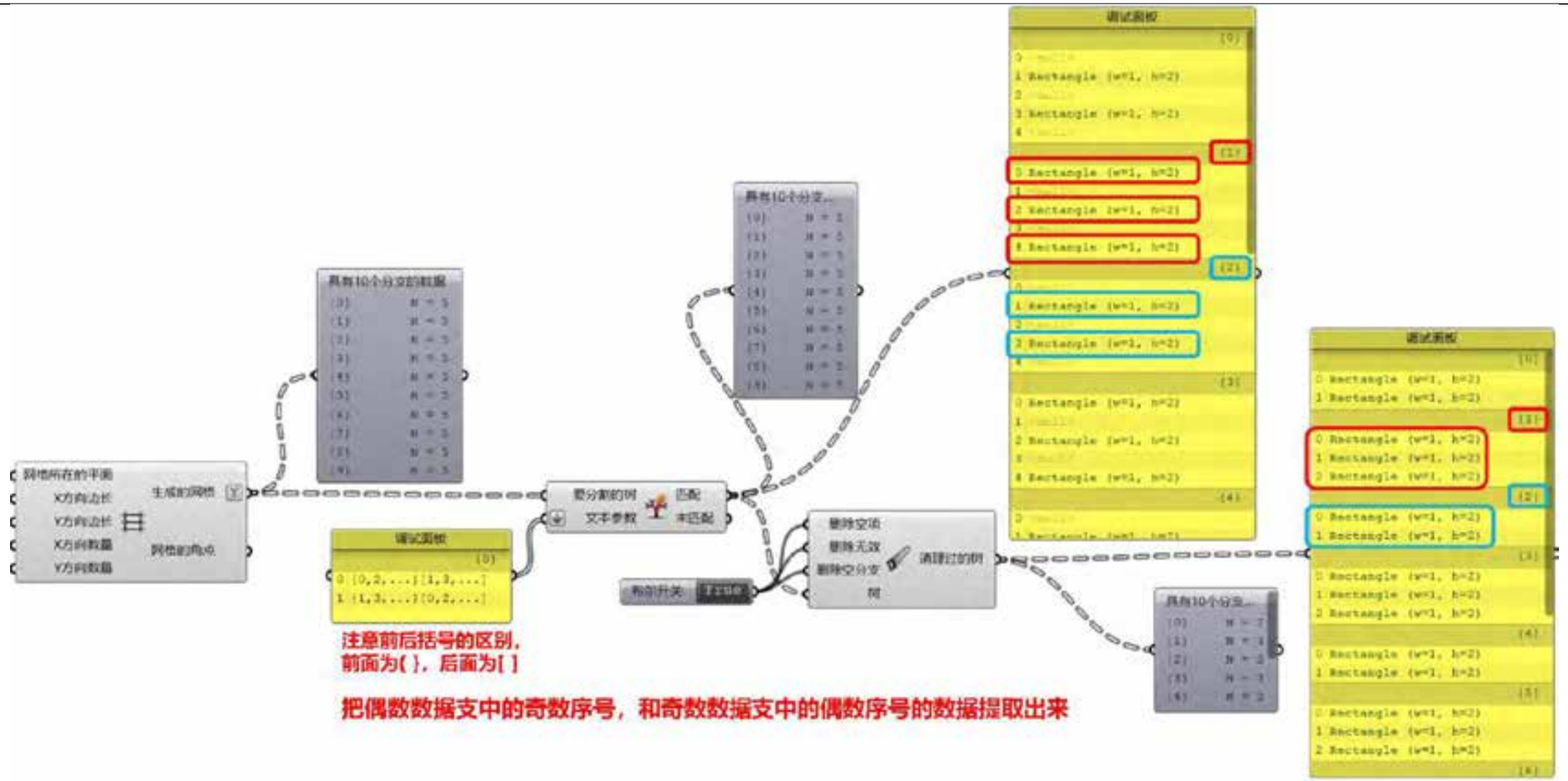


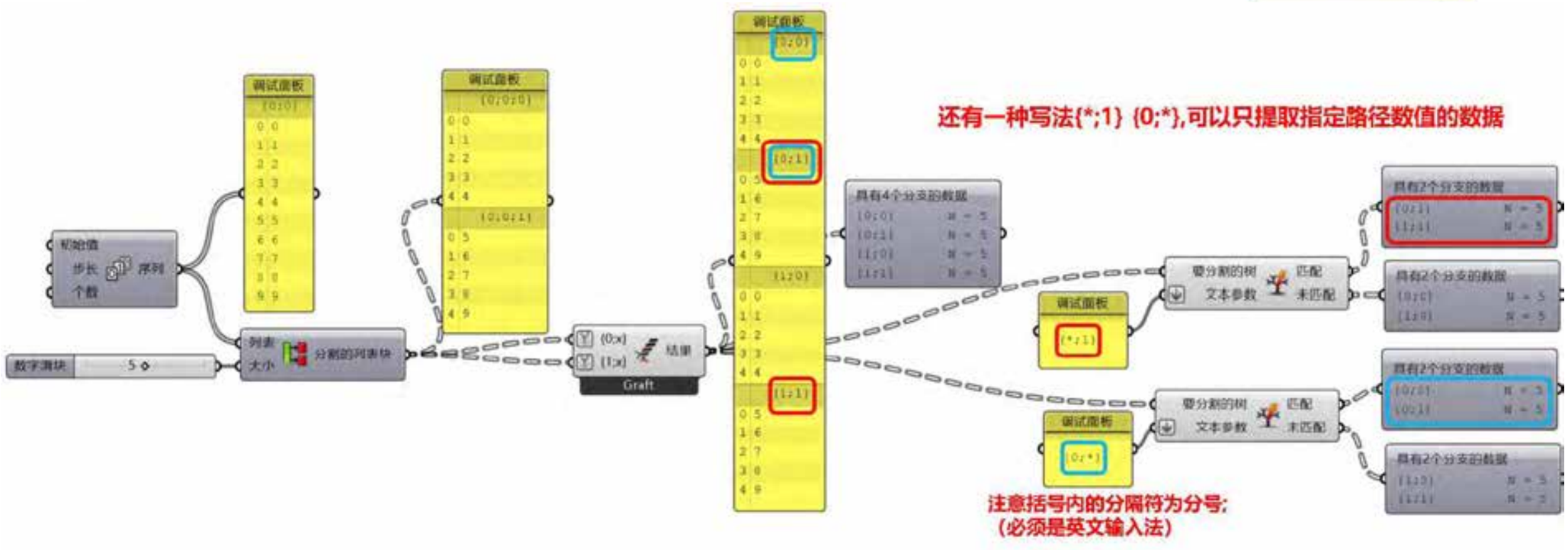
分割树 Split Tree：分割数据结构，将指定路径名称的数据支和其他数据支分离开来。除了直接输入路径名称之外，还有一些特殊操作，比如输入{0,2,...}、 {1,3,...}，就是单独摘出偶数路径和奇数路径。








注意前后括号的区别，
前面为{ }，后面为[ ]
把偶数数据支中的奇数序号，和奇数数据支中的偶数序号的数据提取出来


还有一种写法{*;1} {0;*},可以只提取指定路径数值的数据
注意括号内的分隔符为分号;
(必须是英文输入法)

巧用分割树（Split Tree）与树型数据选择规则（摘自知乎）

【Grasshopper 的奇技淫巧】（进阶版）—— 巧用 Split Tree 与树型数据选择规则

今天讲一下 **Split Tree** 这个神奇的电池，树型数据筛选的利器，以及其背后运作的“树型数据选择规则”逻辑。

在学 **Grasshopper** 的过程中对树型数据的理解可以算是 **gh** 入门的瓶颈，当对树型数据的原理比较透彻地理解之后，你的 **gh** 水平可以算入门了；而对树型数据的处理则是从跨入中高阶用户的门槛。

今天的更新应该算稍稍有难度的一集，在聊树型数据选择规则的部分，需要你对树型数据有一些理解。

（瞎聊两句）为什么要写进阶版？

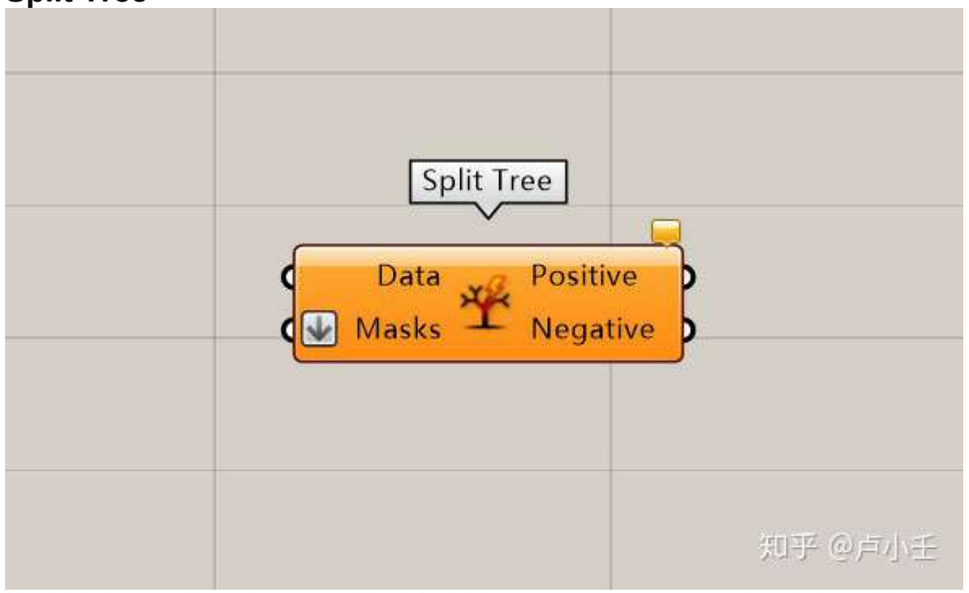
我其实也挺纠结的，进阶版相对受众或者感兴趣的人可能相对较少。但是在国内参数圈普遍都在吹嘘 Ladybug、Kangaroo、表皮干扰一键直通名校的浮躁的风气下，或许需要注入点新的东西 —— 触及参数化本质的东西 —— 数据。

因此想借着进阶版对奇技淫巧系列进行一个小拔高，看看能否带来些许改变。

来吧！

//////////////// 难度 1 星部分 ////////////////

Split Tree



这个是今天聊的主角，**GH** 的原生电池 —— **Split Tree**，主要用于 筛选/切分 树型数据

输入端：

**** Data** ： 需要被处理的树型数据

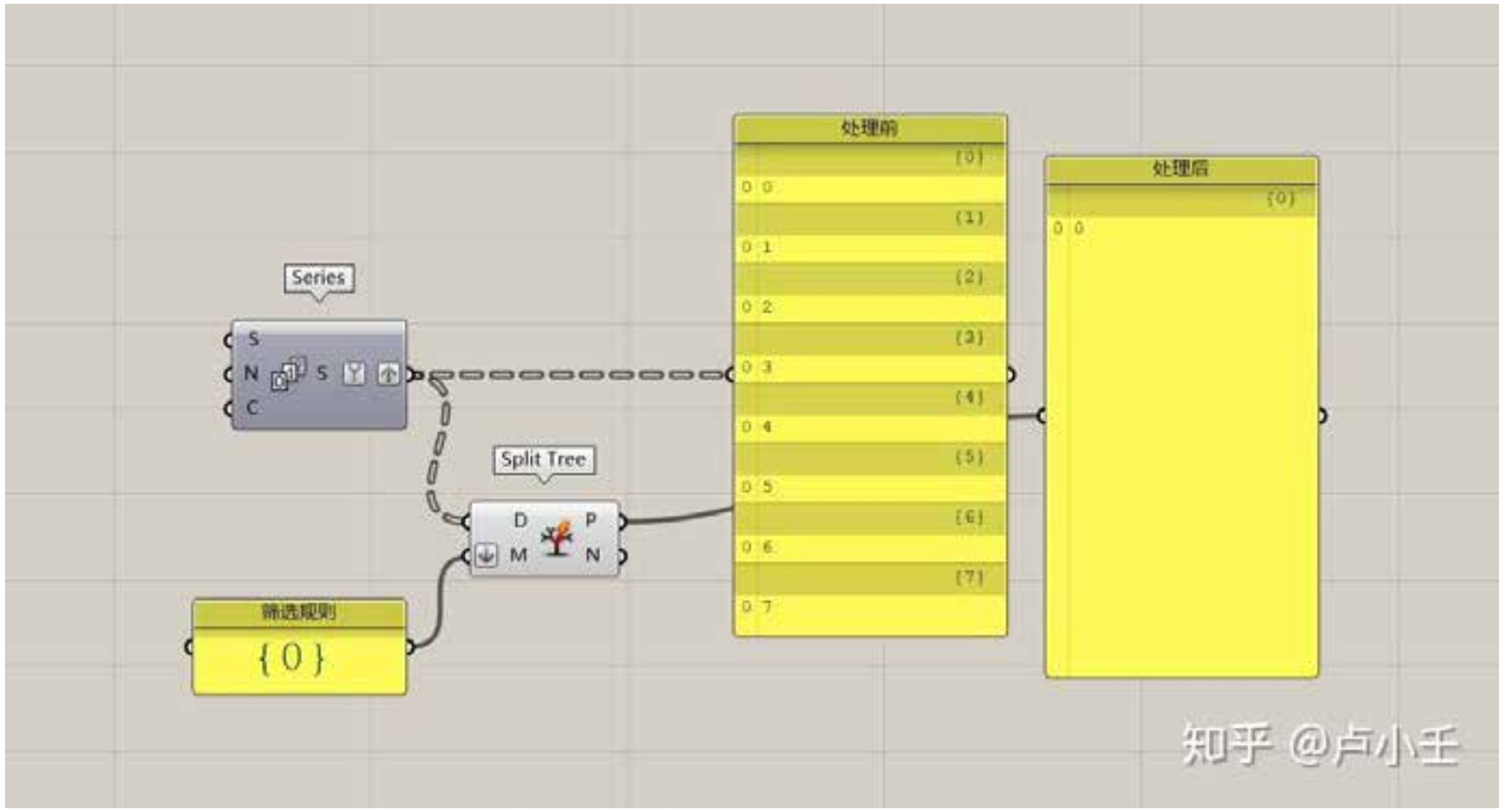
**** Masks** ：数据被 筛选/切分 的规则

输出端：

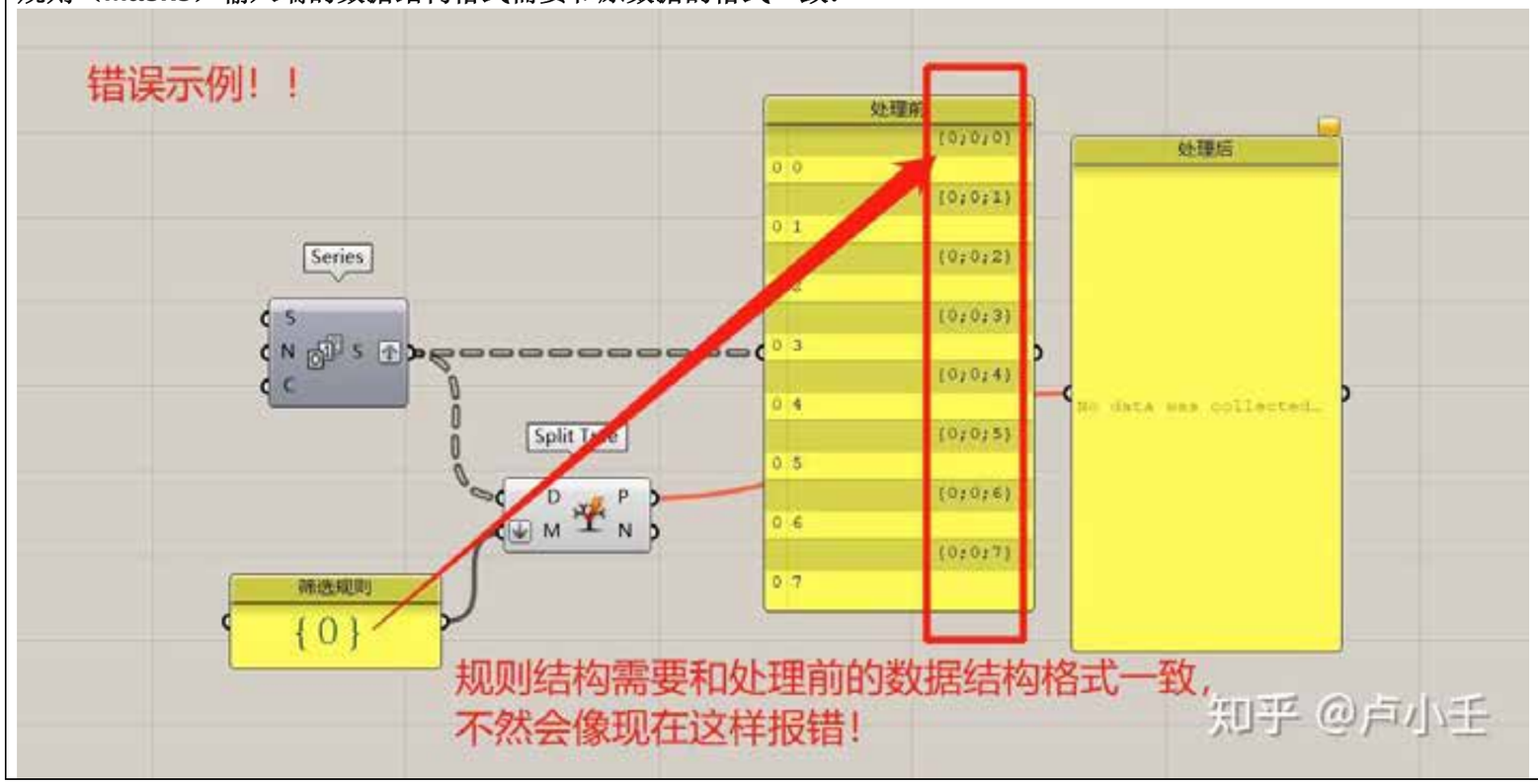
**** Positive** ：符合筛选规则的数据出口

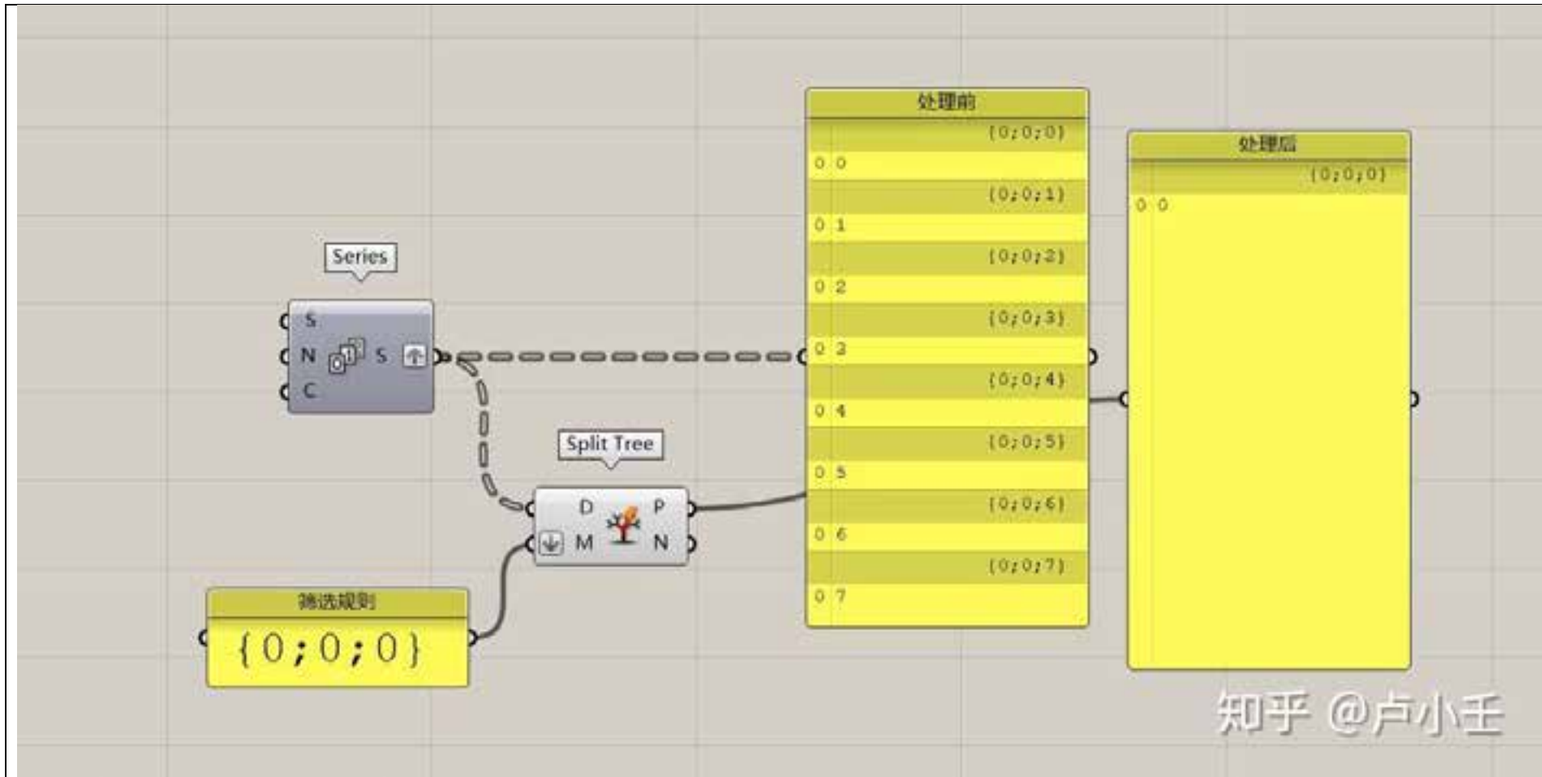
**** Negative**：不符合筛选规则的数据出口

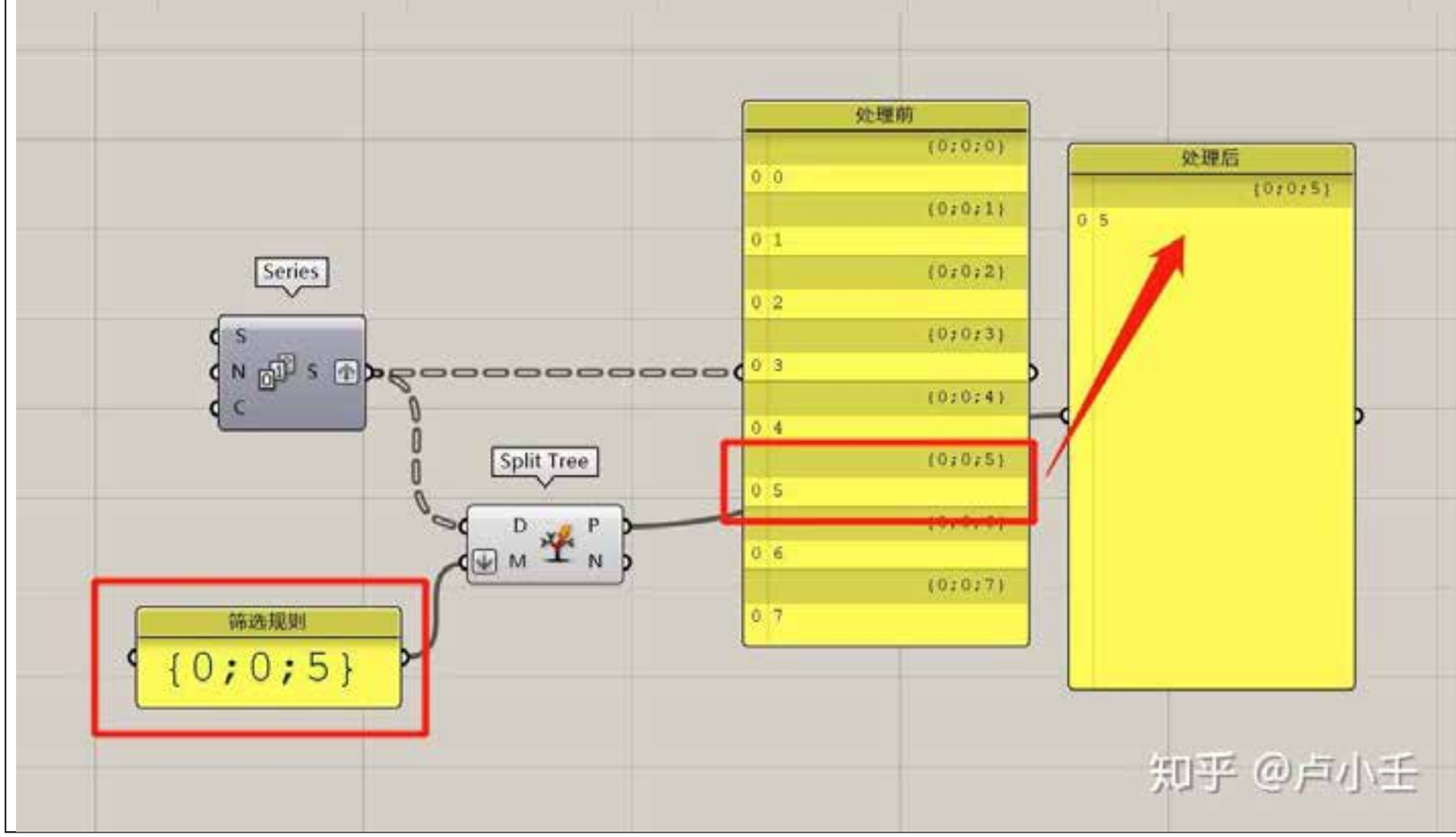
比如：



规则（**Masks**）输入端的数据结构格式需要和原数据的格式一致：



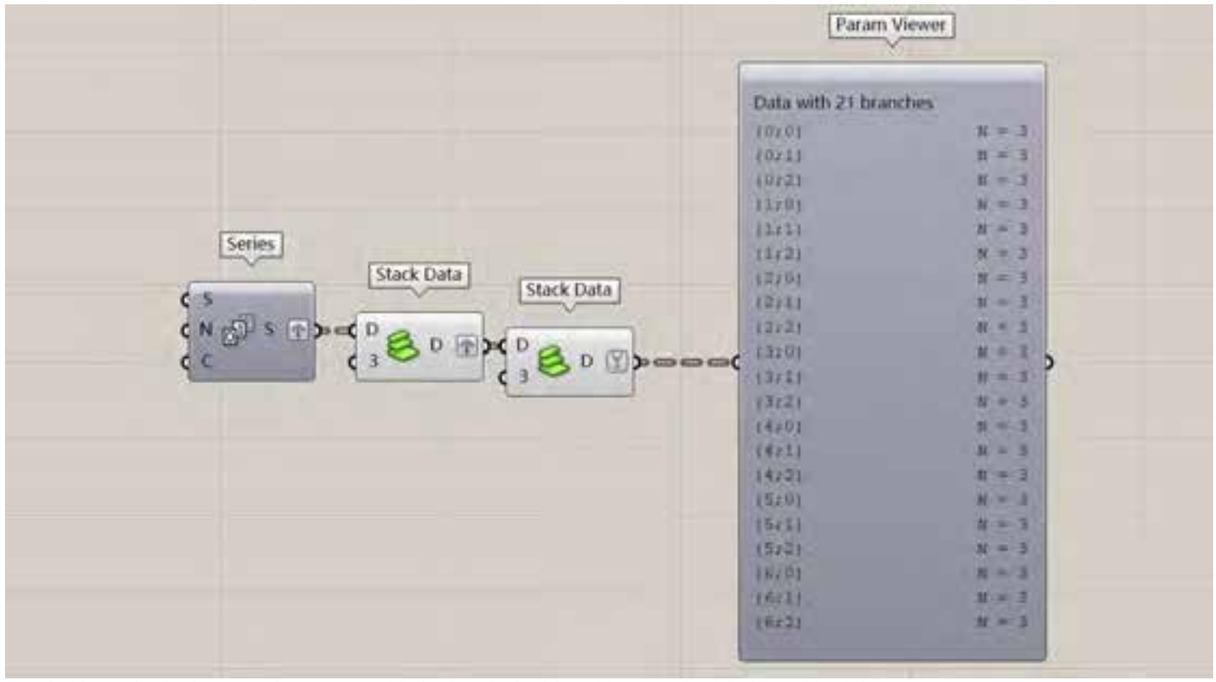

处理前
{0;0;0}
0 0
{0;0;1}
0 1
{0;0;2}
0 2
{0;0;3}
0 3
{0;0;4}
0 4
{0;0;5}
0 5
{0;0;6}
0 6
{0;0;7}
0 7
处理后
{0;0;0}
0 0
Series
S
N
S
C
Split Tree
D
P
M
N
筛选规则
{0;0;0}
知乎 @卢小壬


处理前
{0;0;0}
0 0
{0;0;1}
0 1
{0;0;2}
0 2
{0;0;3}
0 3
{0;0;4}
0 4
{0;0;5}
0 5
{0;0;6}
0 6
{0;0;7}
0 7
处理后
{0;0;5}
0 5
Series
S
N
S
C
Split Tree
D
P
M
N
筛选规则
{0;0;5}
知乎 @卢小壬

最基本的规则：

以大括号 **{ }** 构造，分号 ；区分不同层级的组号 ——////////////////// 难度 **2** 星部分 //////////////////

直接输入组号，筛选出组，所见即所得。那么，如果当数据更加复杂，变成了这样：

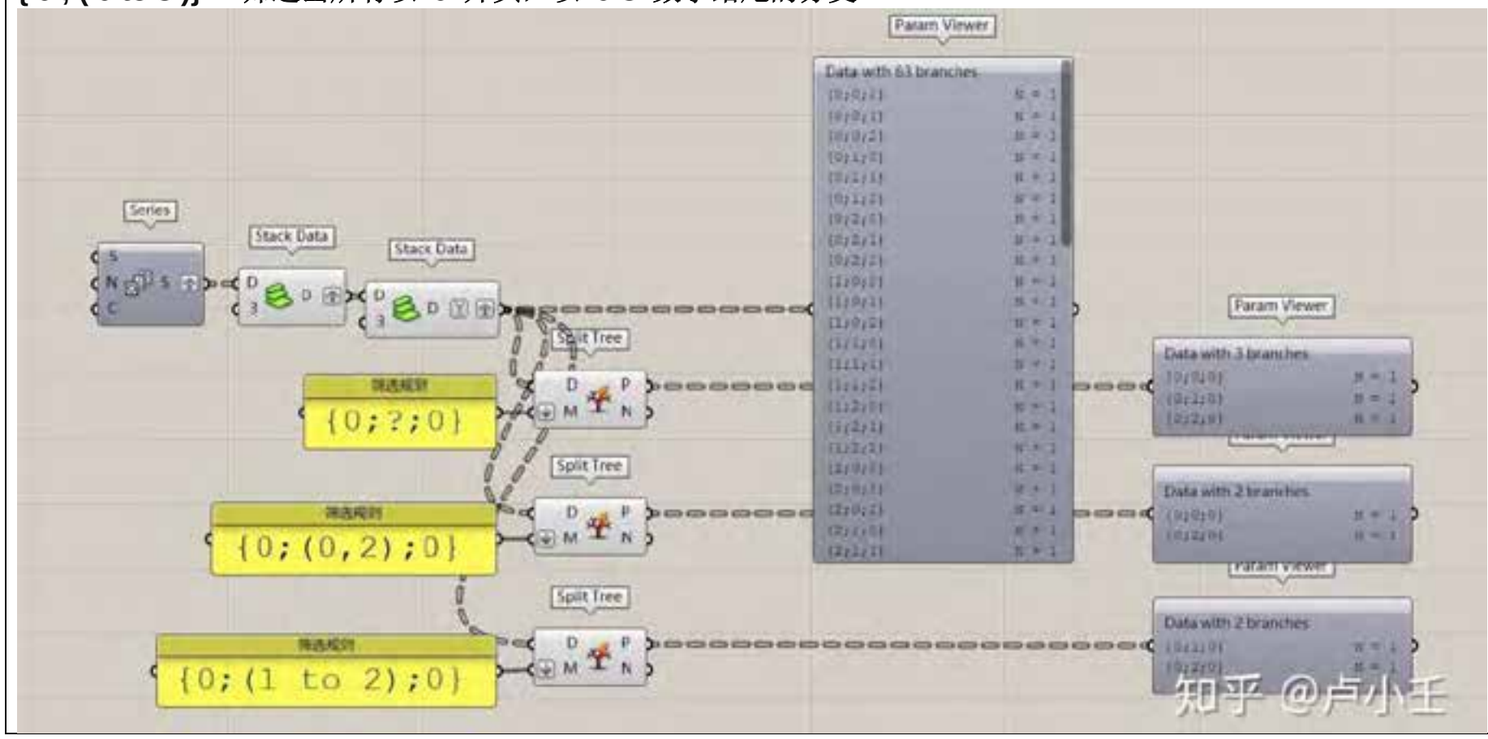


而我的要求也变得更加骚气，比如筛选出所有第二层组号为 **2** 的数据 ？？

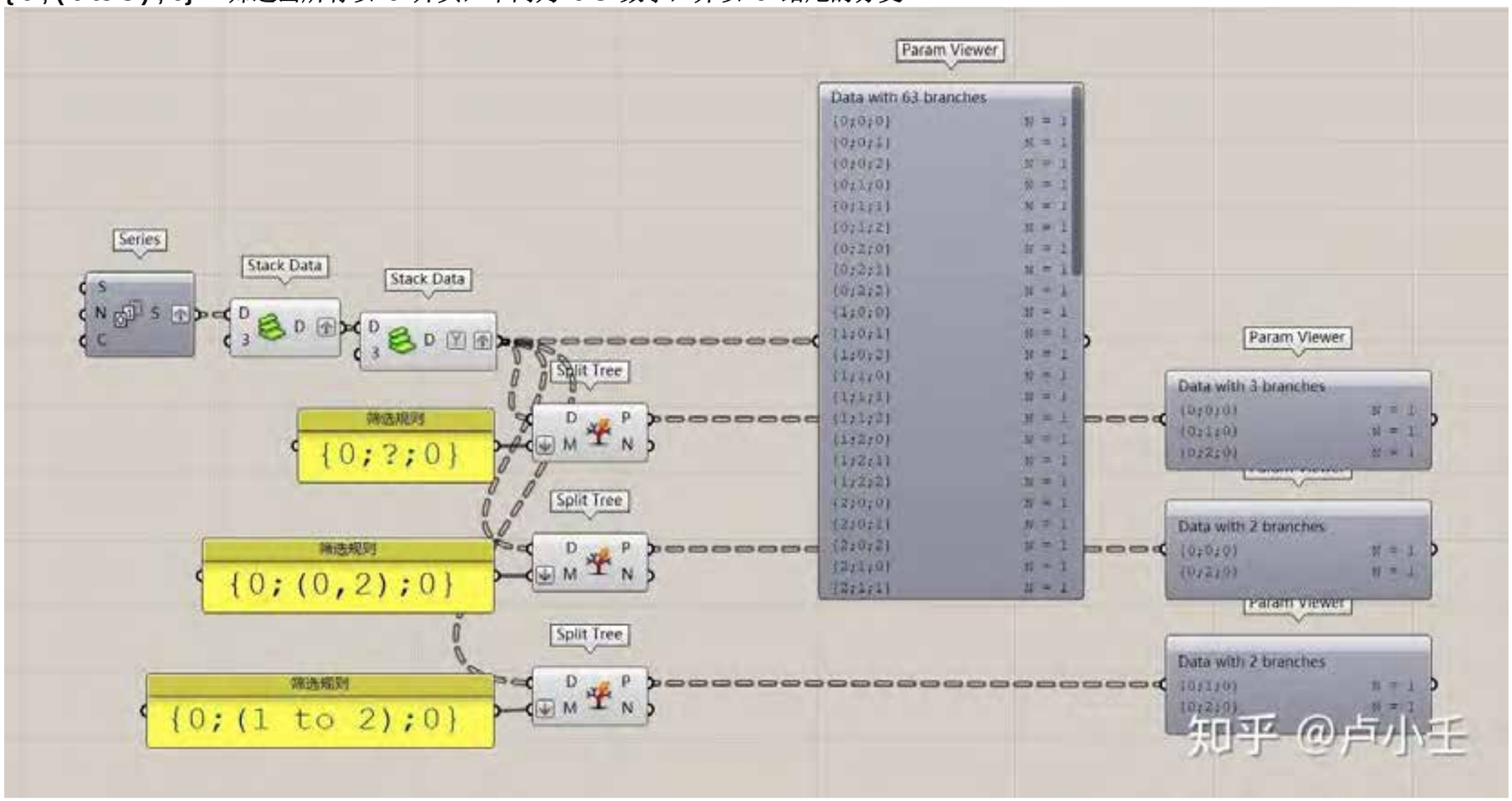
{ 0 ; ? } = 筛选出所有以 **0** 开头的分支

{ 0 ; (0,3)} = 筛选出所有以 **0** 开头，以 **0** 或 **3** 结尾的分支

{ 0 ; (0 to 3)} = 筛选出所有以 **0** 开头，以 **0-3** 数字结尾的分支



相应地，这些规则符号可以运用在任何一个分支层级：
{ 0 ; ? ; 0 } = 筛选出所有以 **0** 开头和结尾的分支
{ 0 ; (0,3) ; 0 } = 筛选出所有以 **0** 开头，中间 **0** 或 **3** ，并以 **0** 结尾的分支
{ 0 ; (0 to 3) ; 0} = 筛选出所有以 **0** 开头，中间为 **0-3** 数字，并以 **0** 结尾的分支

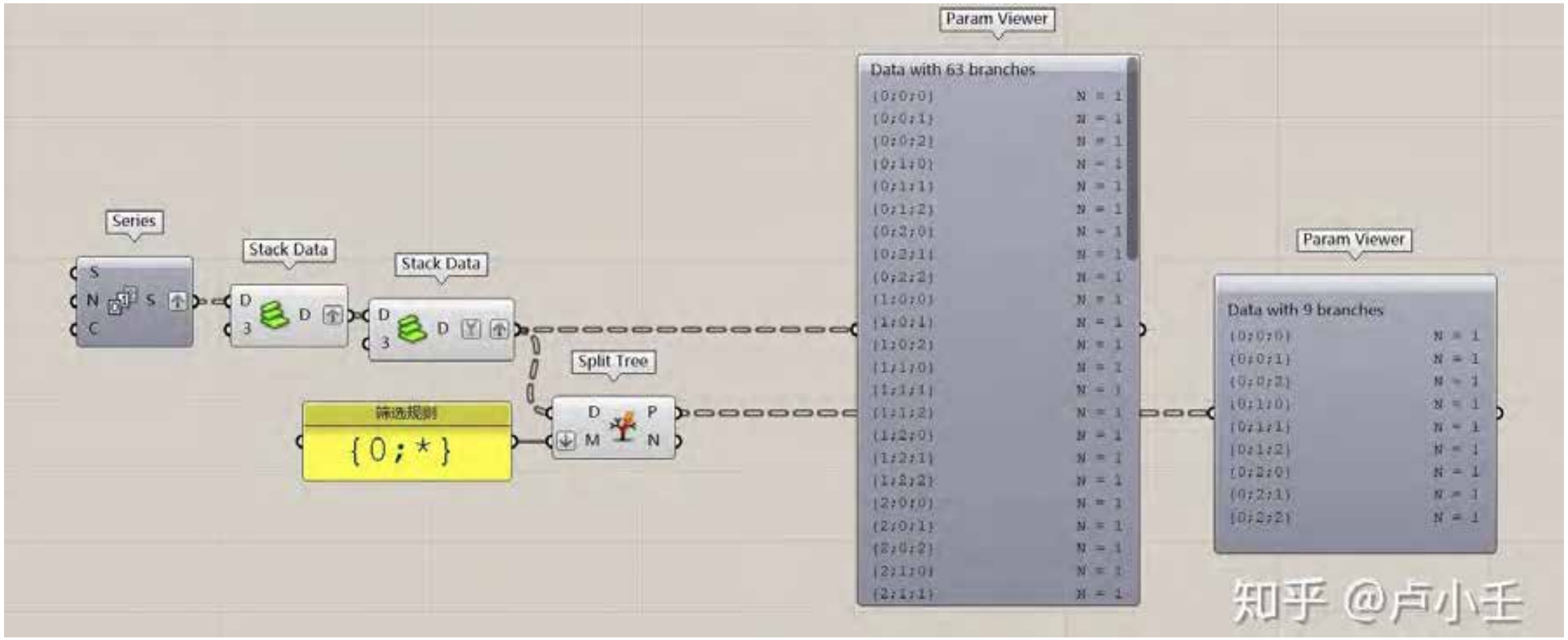


以上的操作方式，都是因为规则符号的引入

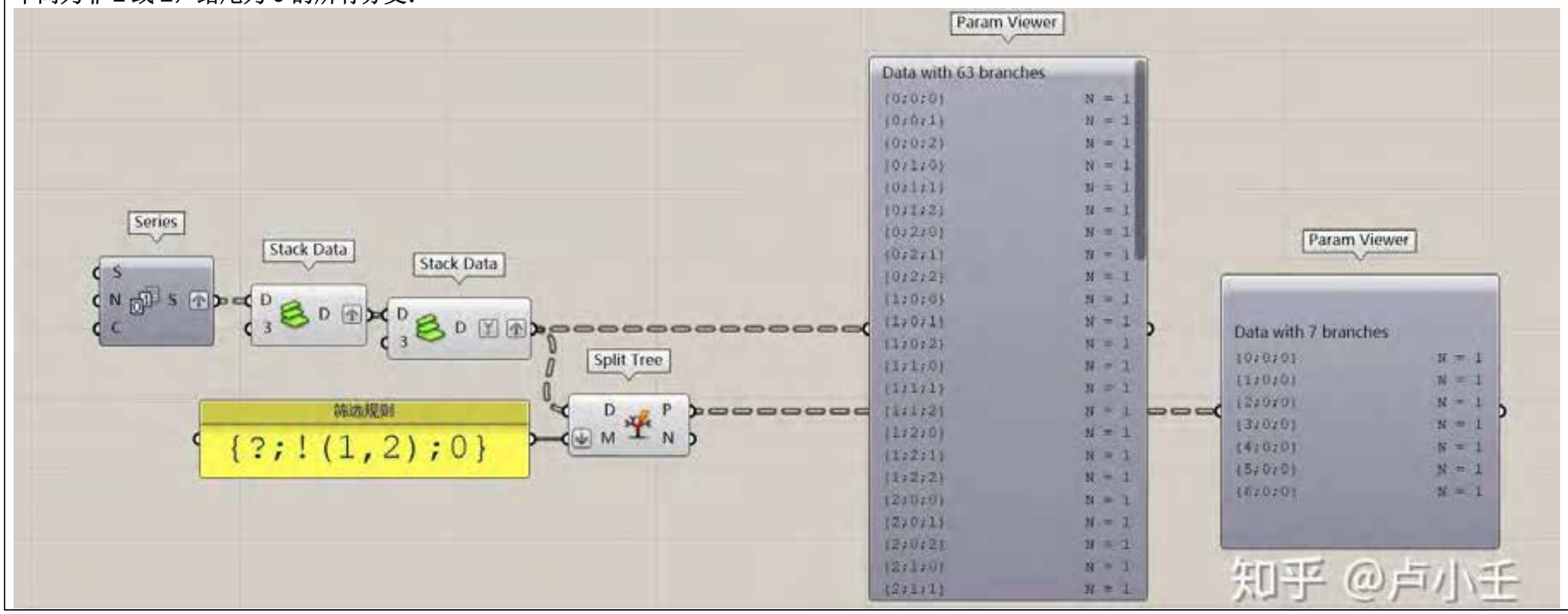
可用的规则符号：
？ = 任意的单个整数
*** =** 任意数量的整数
3 = 特定的整数（这里以 **3** 为例）
！ 3 = 除特定整数外的所有整数（这里以“非 **3**”的数为例）
(2, 4, 5) = 组内的任一整数（这里以 **2**、**4**、**5** 为例）
! (2, 4, 5) = 除组内任一整数的其他所有整数（这里以“非 **2**、**4**、**5**”的数为例）
(2 to 10) = 区间内的所有整数（这里以 **2** 到 **10** 区间为例）（包含 **2** 和 **10**）
! (2 to 10) = 区间外的所有整数（这里以 不在 **2** 到 **10** 区间 为例）（不包含 **2** 和 **10** ）
(0 , 2 , ...) = 无限序列中的任何整数（这里以 “偶数” 为例）
! (2 , 4 , ...) = 不属于无限序列中的任何整数（这里以 **“0** 以及其他奇数” 为例）
(0 , 2 , ... , 30) = 有限序列中的任何整数（这里以 **0** 到 **30** 中的所有偶数 为例）
! (2 , 4 , ... , 30) = 不属于有限序列中的任何整数（这里以 **"0** 和大于 **30** 的偶数，以及所有奇数“ 为例）

因此，玩法就变得多种多样了：

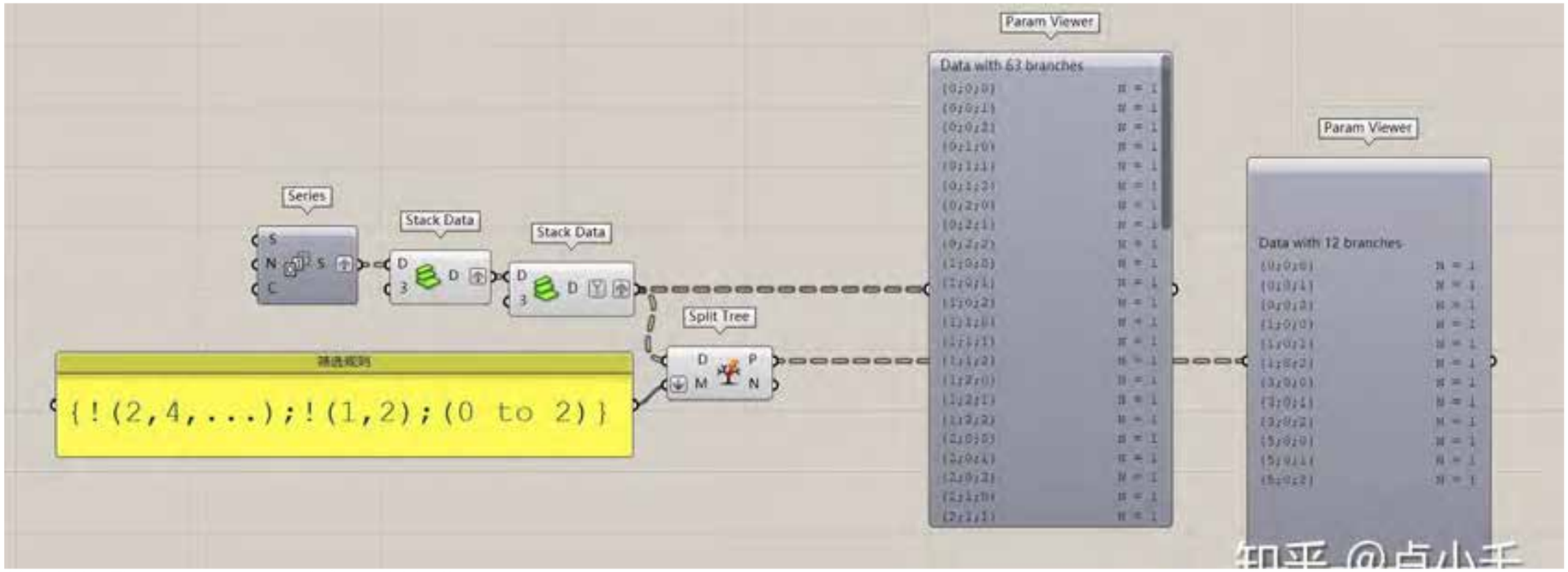
0 开头的所有分支：



中间为非 **1** 或 **2**，结尾为 **0** 的所有分支：



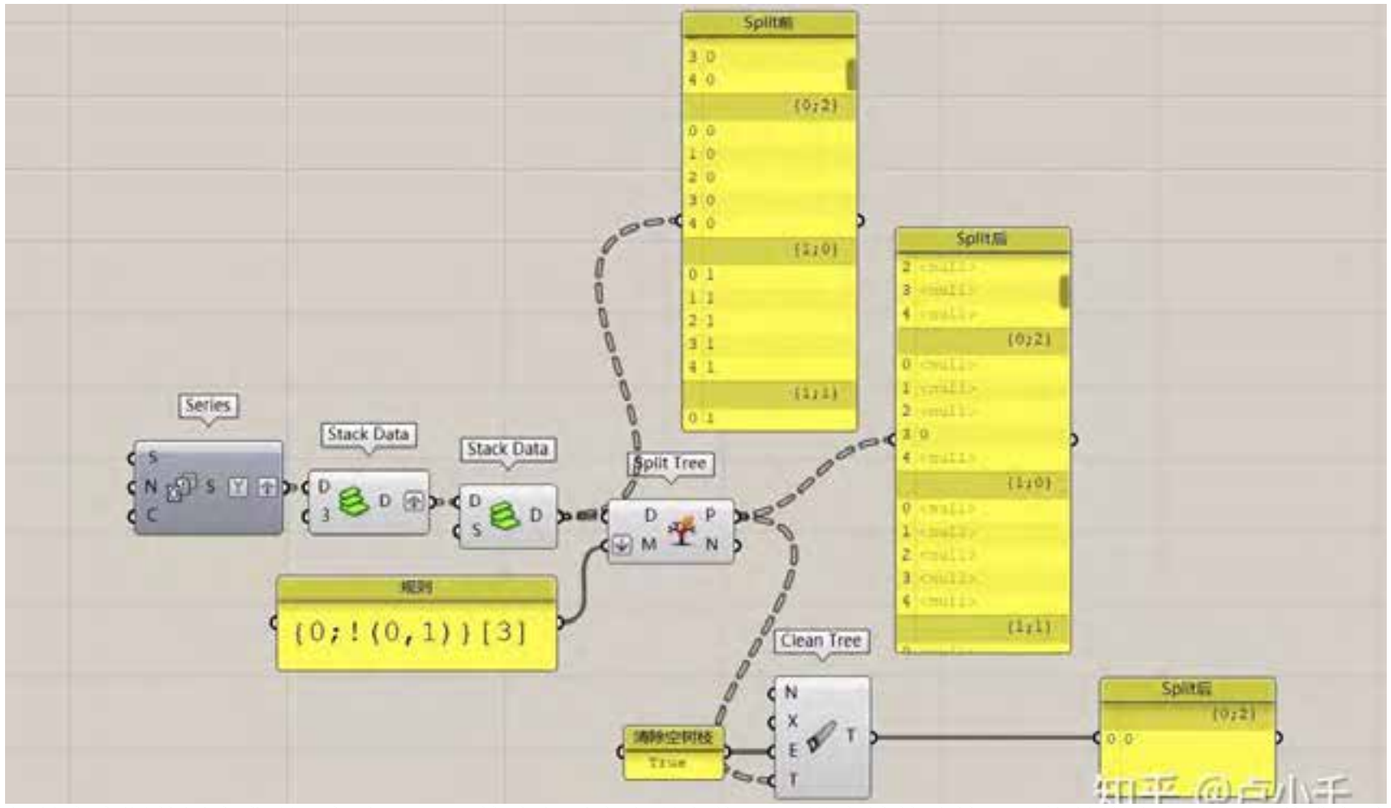
开头为 **0** 或奇数，中间为非 **1** 或 **2**，结尾为 **0** 到 **2** 区间的所有分支：



希望还没晕 **^^////////////////** 难度 **3** 星部分 **////////////////**

除上面的规则符号外，树型数据选择规则还可以通过中括号 **[ ]** 筛选出组中的项值（**item**）

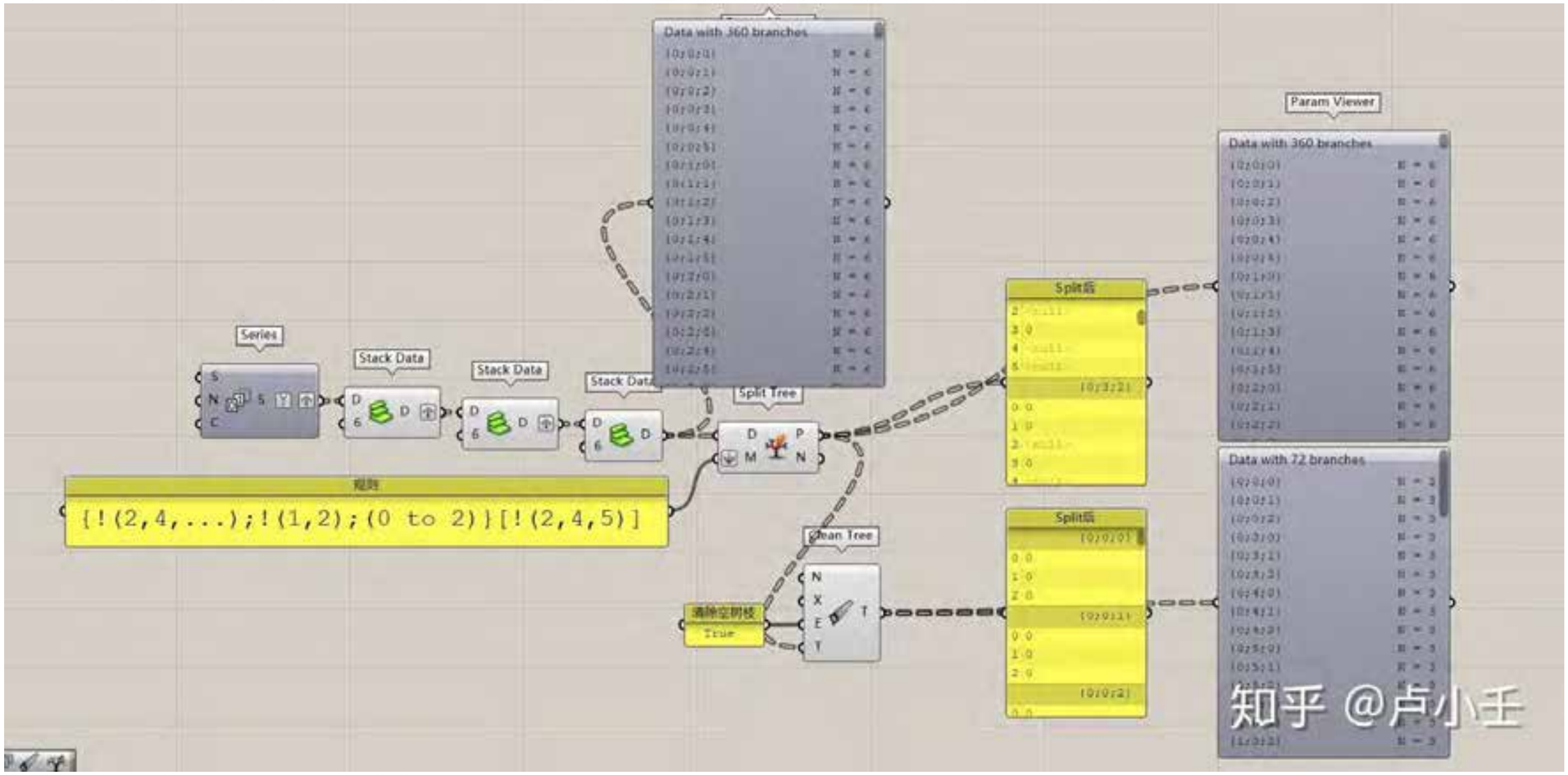
比如，末尾组号不为 **0** 或 **1** 的分支中 序号为 **3** 的 **item**：



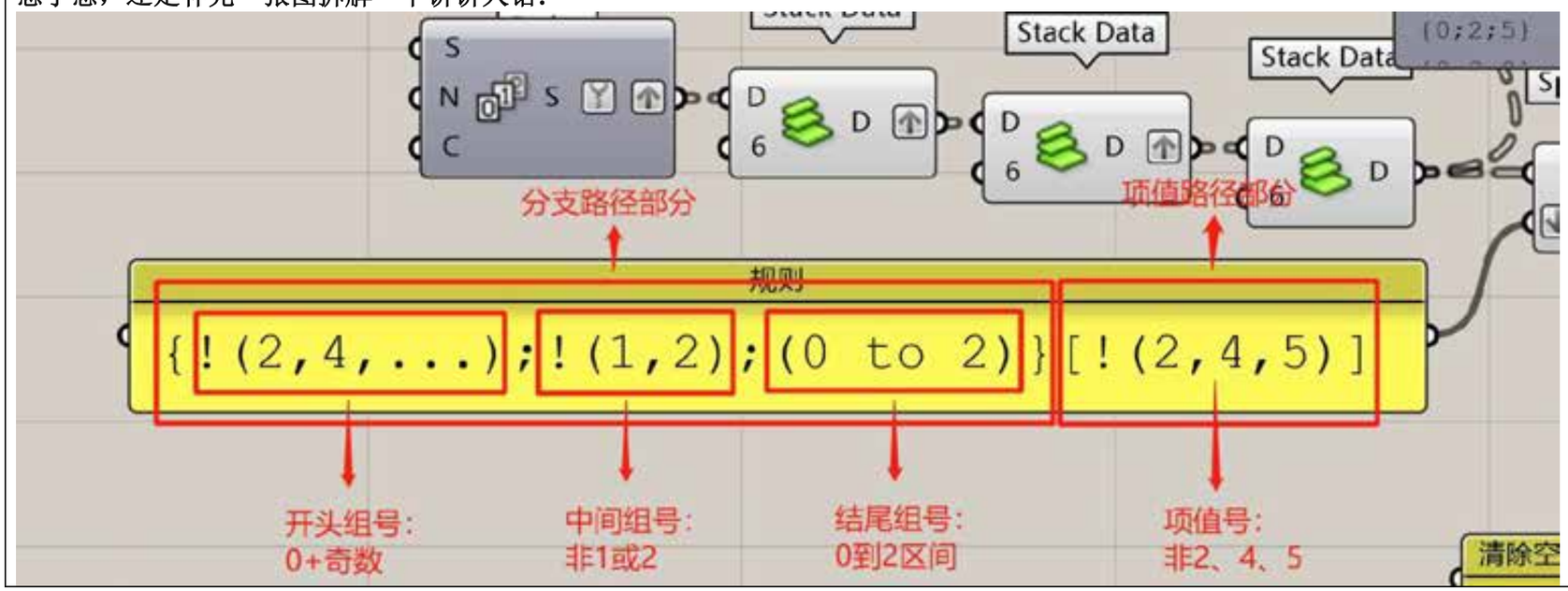
但是在实际使用中发现，加上 **item** 筛选之后会出现空值 **null** 没有被筛选掉，可能是个小 **bug**，因此我加上了一个 **Clean Tree** 将所有 **null** 空值 和所有 空分支清除

同时，**[ ]** 中括号内的 **item** 筛选也可以赋予规则符号，于是画风就会变得花里胡哨的。

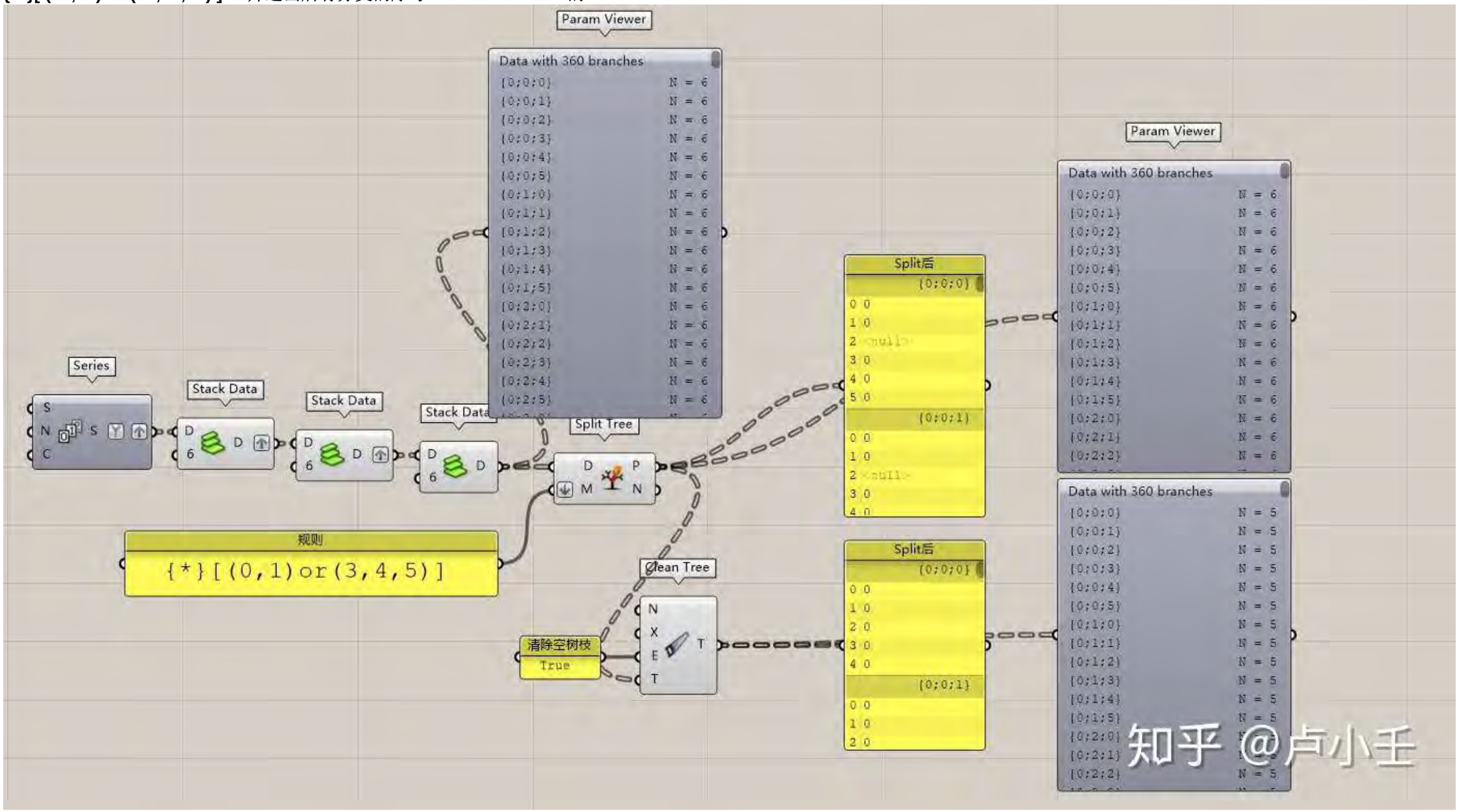
比如，筛选开头为 **0** 或奇数，中间为非 **1** 或 **2**，结尾为 **0** 到 **2** 区间的所有分支 中的 序号不为 **2**、**4**、**5** 的 **item** ..... **:**



想了想，还是补充一张图拆解一下讲讲人话：



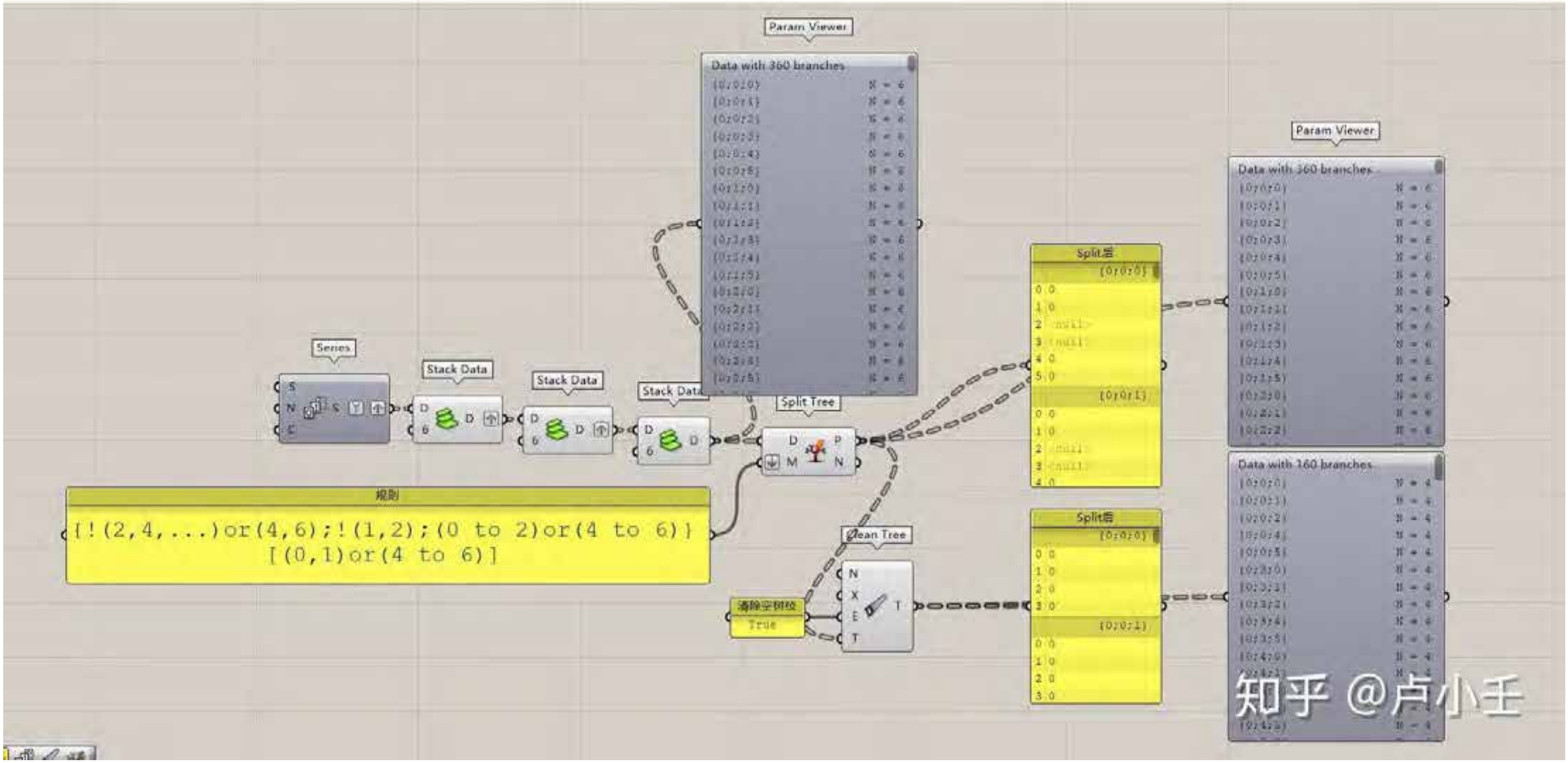
然而，上限更高的来了.....
可以用布尔运算组合多个规则，比如：
{ * }[(0 , 1) or (3 , 4 , 5)] = 筛选出所有分支的序号 **0、1、3、4、5** 的 **item**



所以也就可以根据自己的需要组合出千千万万种复杂到妈妈都认不得的筛选规则：

来，王炸！！

比如，筛选开头为 **0** 或 **4** 或 **6** 或奇数，中间为非 **1** 或 **2**，结尾为 **0** 到 **2** 区间或 **4** 到 **6** 区间的所有分支 中的；序号为 **0** 或 **1** 或 **4** 到 **6** 区间的所有 **item** ..... **:**

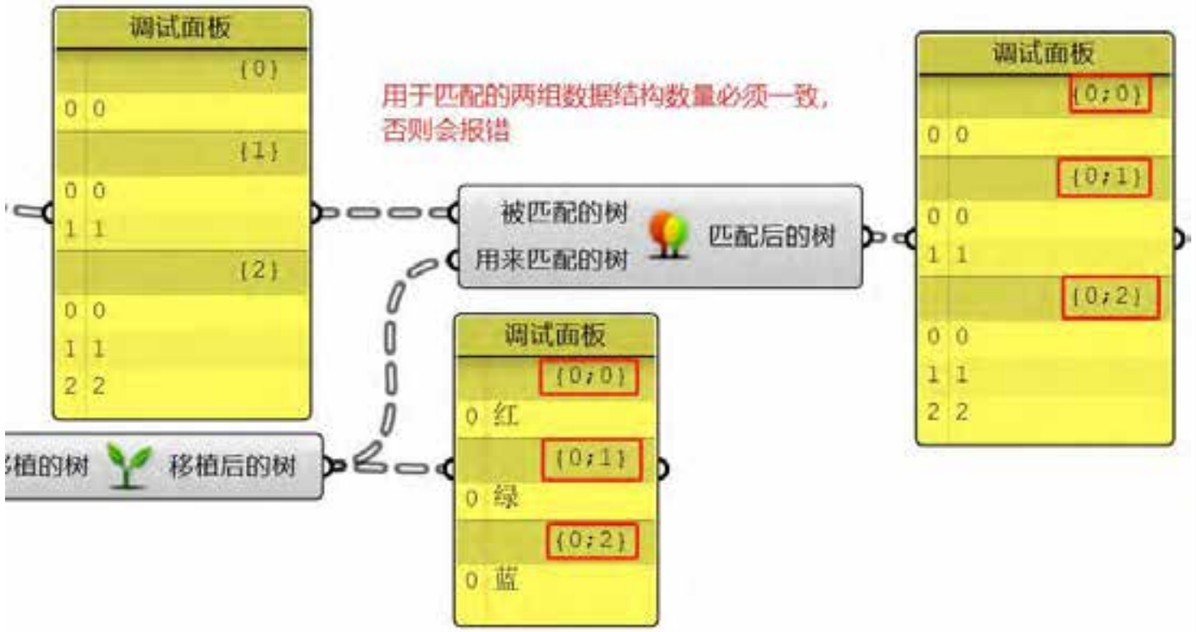


以上；

匹配 Match Tree
定向输出 Stream Gate
数据流过滤 Stream Filter
路径映射器 Path Mapper

匹配 Match Tree：

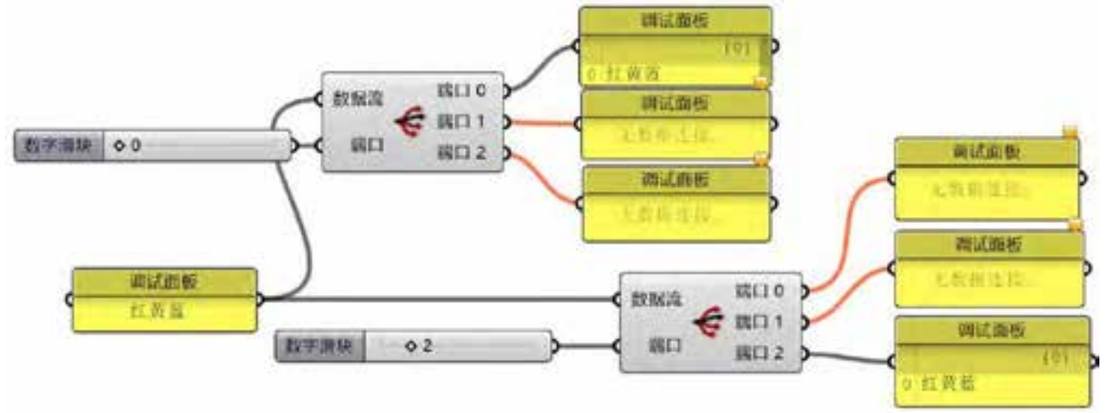
将一组数据结构的名称替换成另一组的数据结构名称。两组的数据结构数量应当一致，否则会报错



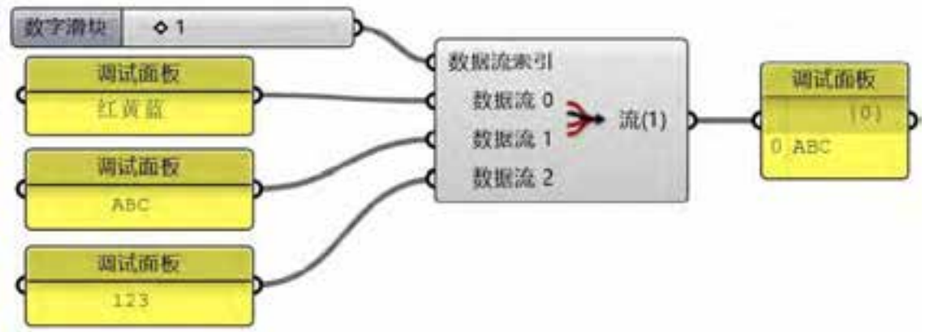
。

定向输出 Stream Gate：

数据流闸门，通过设定端口数值（0、1、2...），控制数据流分配到对应的输出端口

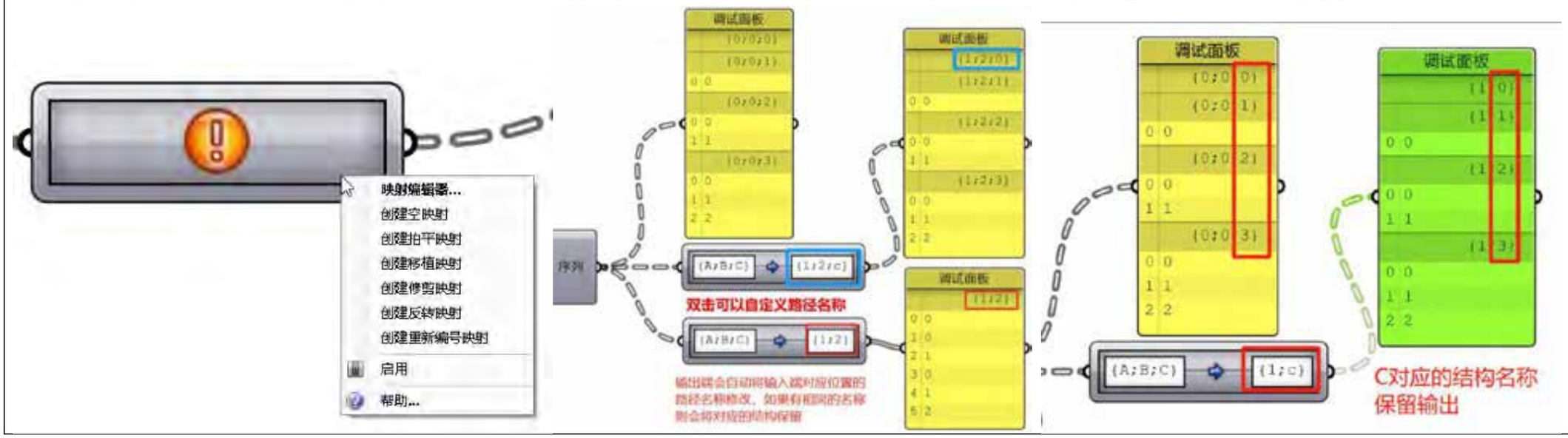


数据流过滤 Stream Filter: 数据流过滤器，根据索引端输入的数值控制输出对应的输入端的数据流

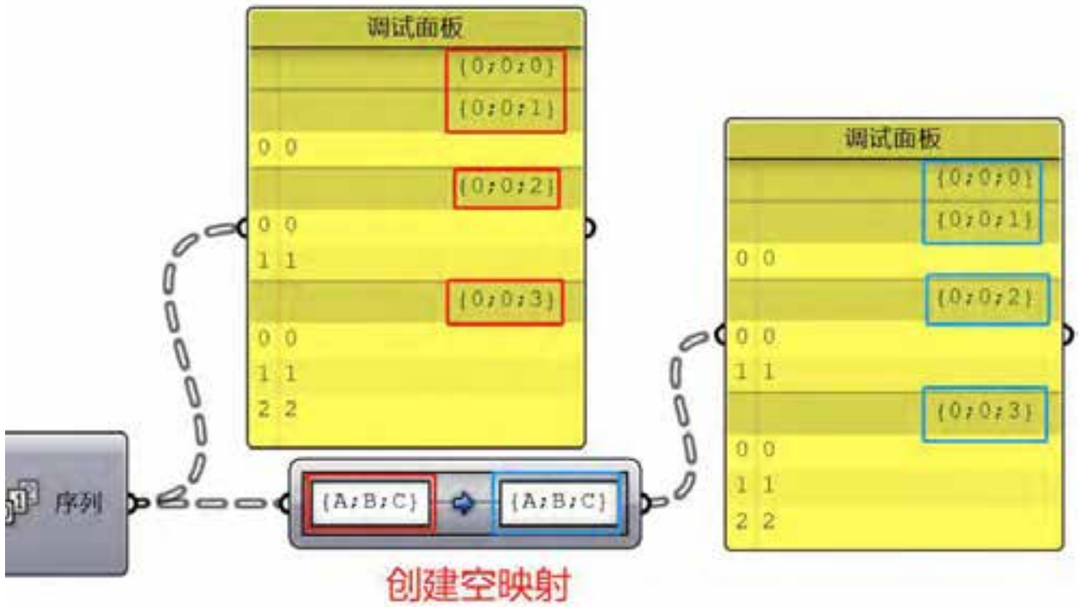


路径映射器 Path Mapper：数据结构编辑器。右键可以选择预设好的集中映射方式。（更多路径映射器的使用技巧可以参考（【犀流堂】100GH 运算器详解）https://www.kancloud.cn/rhinostudio/ghuse100/1665388）

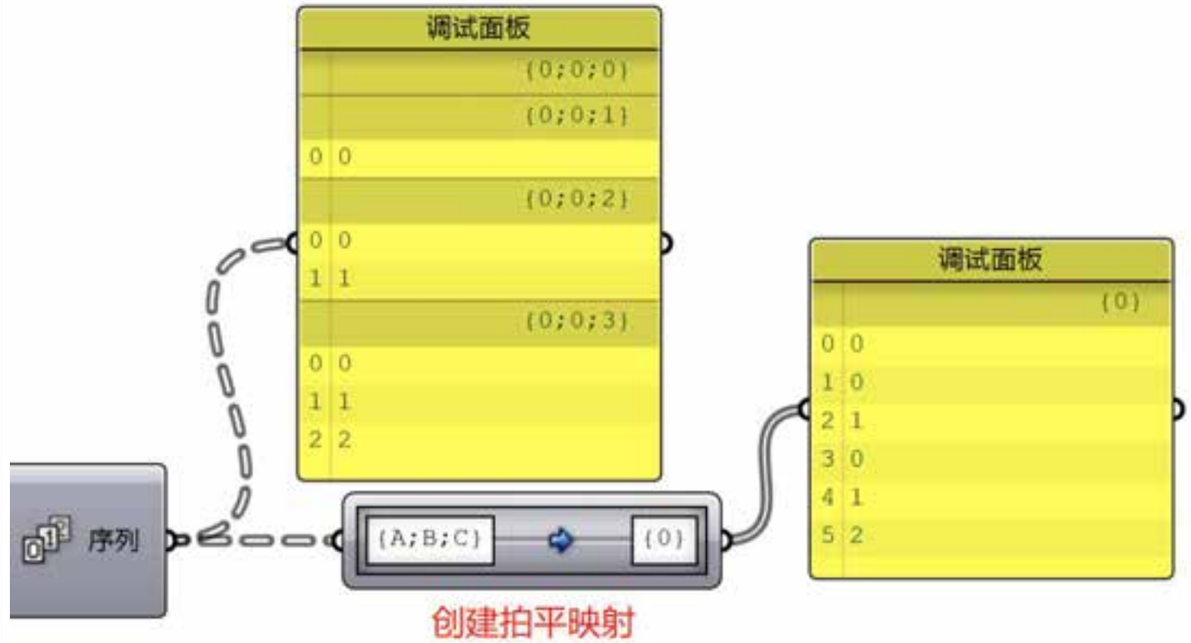
可以双击运算器打开编辑窗口，修改输出的路径名称。经测试，输出端格式字母只能用 ABC 这三个，大小写都可以，数字的话可以任意输入，也可以两者组合。



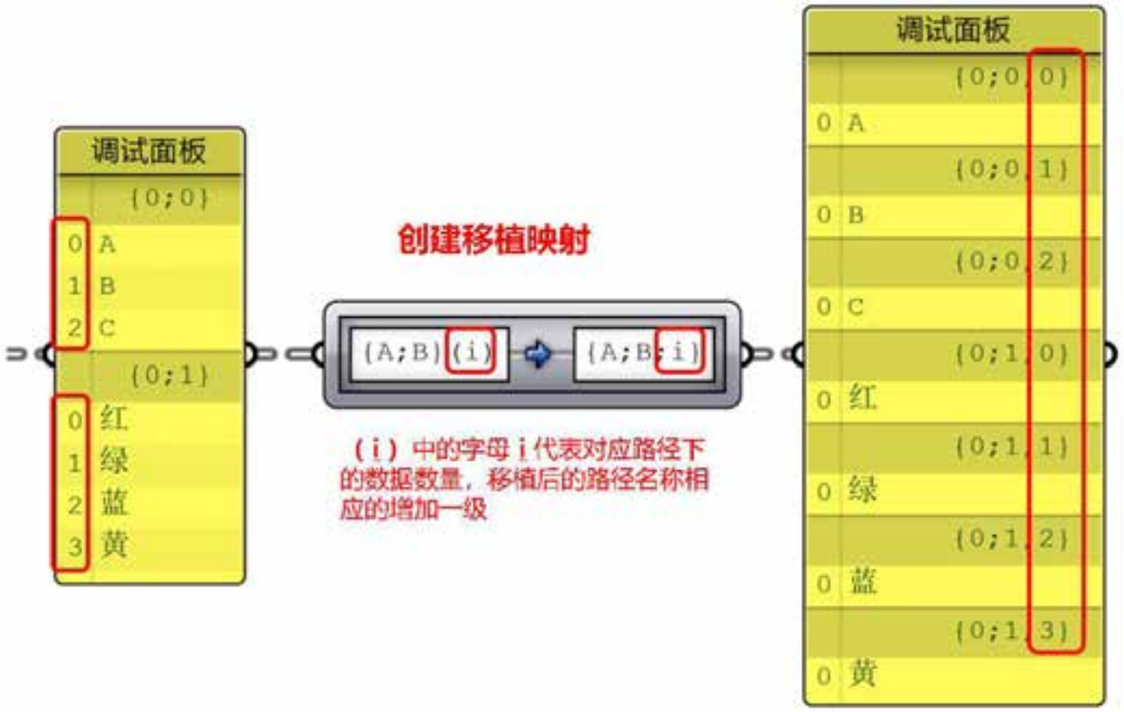
创建空映射 Create Null Mapping：不对数据结构发生改变。路径映射器上的文字{A;B;C}对应的就是输入端的数据结构格式。



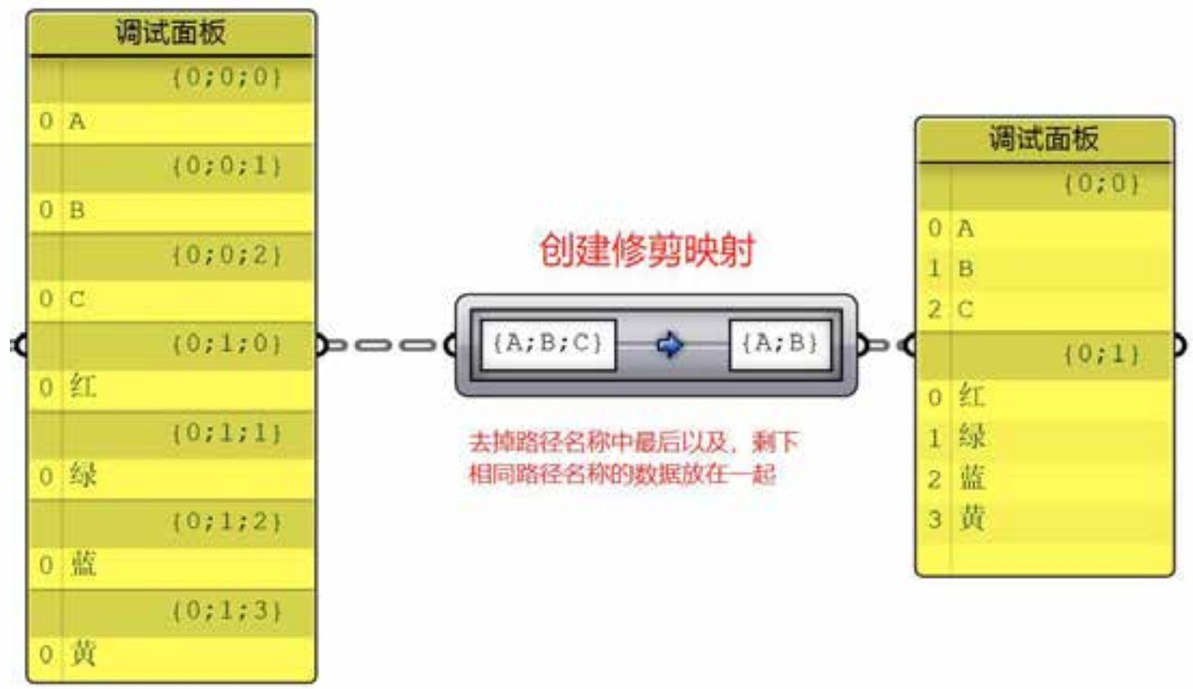
创建拍平映射 Create Flatten Mapping：跟前面的拍平运算器同理，将所有数据拍平放在同一个路径下。



创建移植映射 Create Graft Mapping：公式为 {A;B;C}(i)到{A;B;C;i}，i 对应数据结构中的序列号，移植映射就是把所有的数据单独放在一个分支路径下面，路径名称中自动增加一级数字为原来对应的序列号 i。

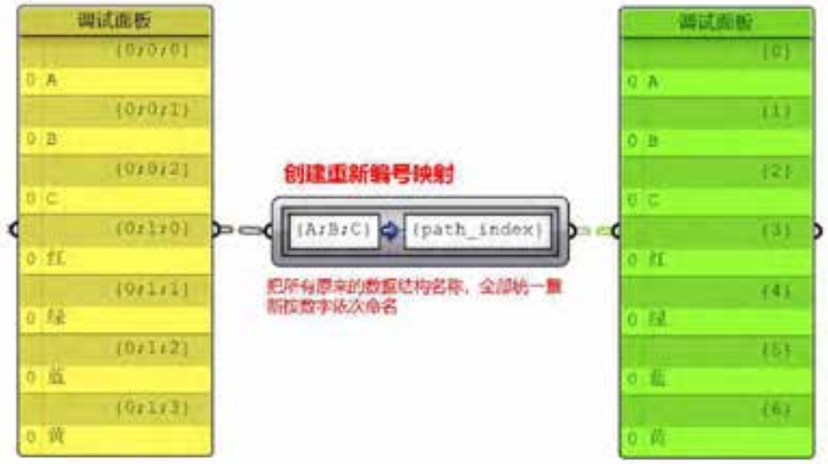


创建修剪映射 Create Trim Mapping：修剪数据结构，从数据结构名称的尾端砍掉一级。也可以双击自定义编辑公式砍掉两级结构。



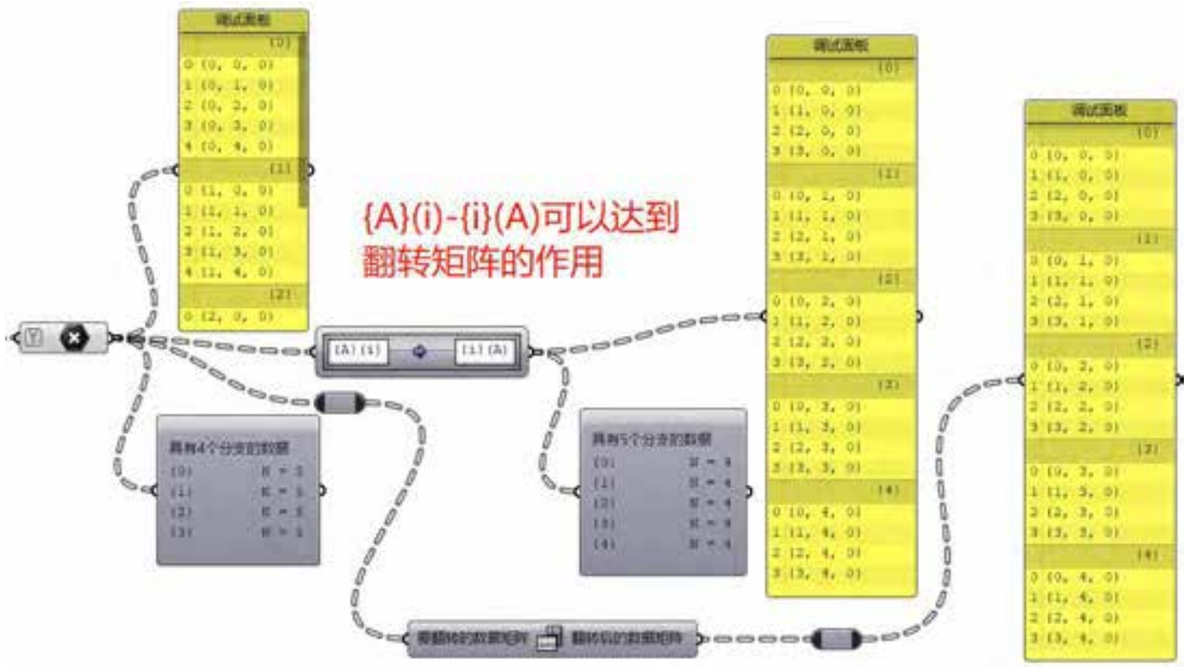
创建反转映射 Create Renumber Mapping：将对应路径结构下的数据排列顺序反转。

创建重新编号映射 Create Renumber Mapping：把所有原来的数据结构名称，全部统一重新按数字依次命名



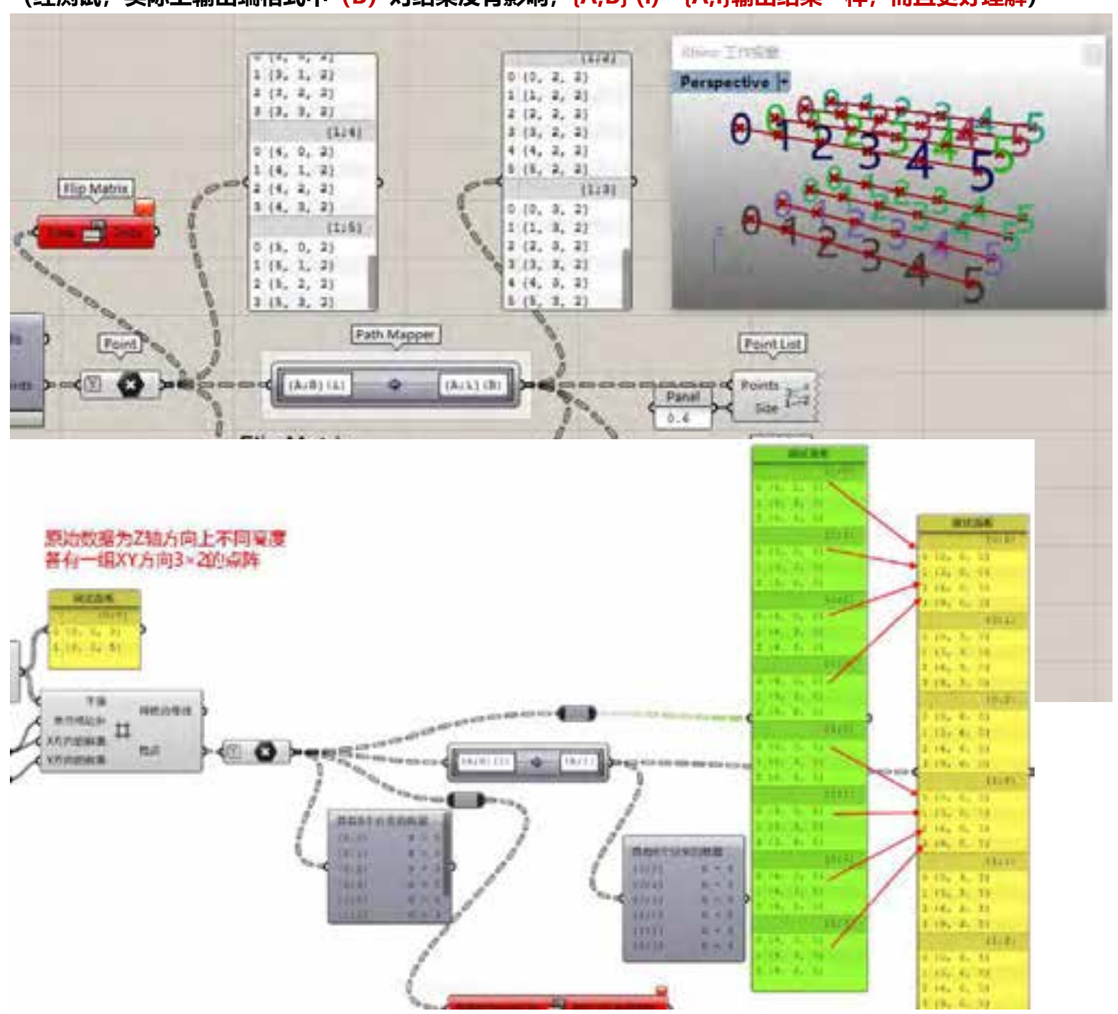
自定义路径映射器的几种技巧：

1、{A} (i)→{i}(A):与翻转矩阵同理。

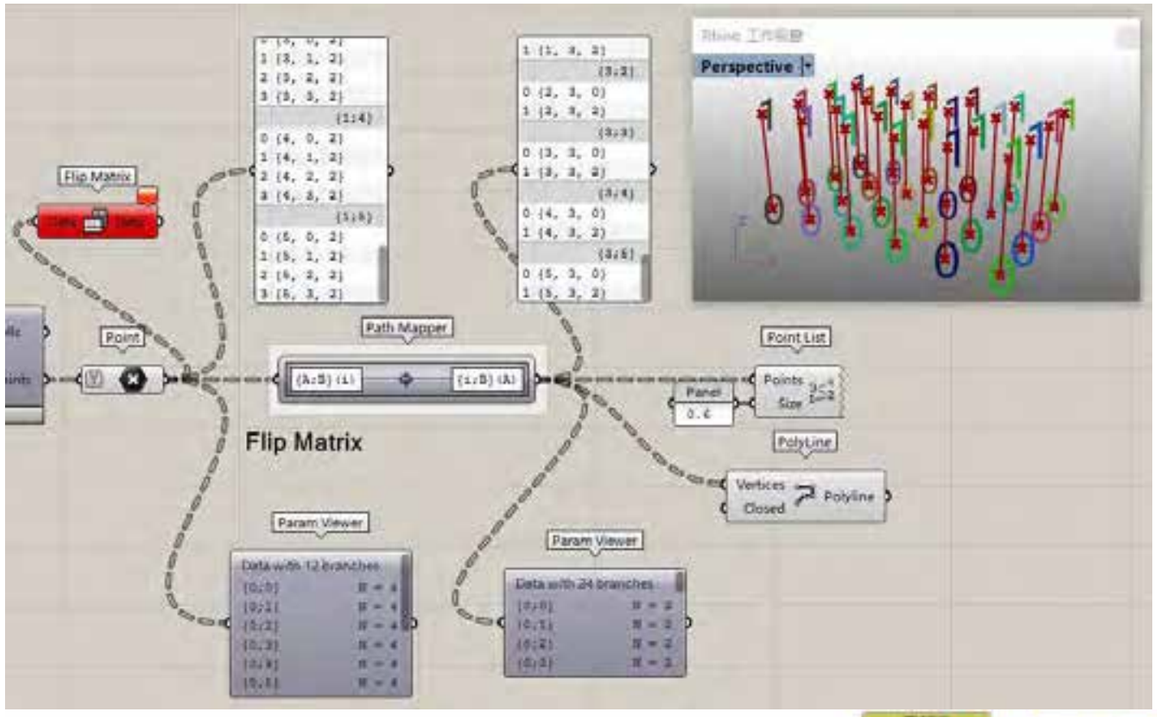


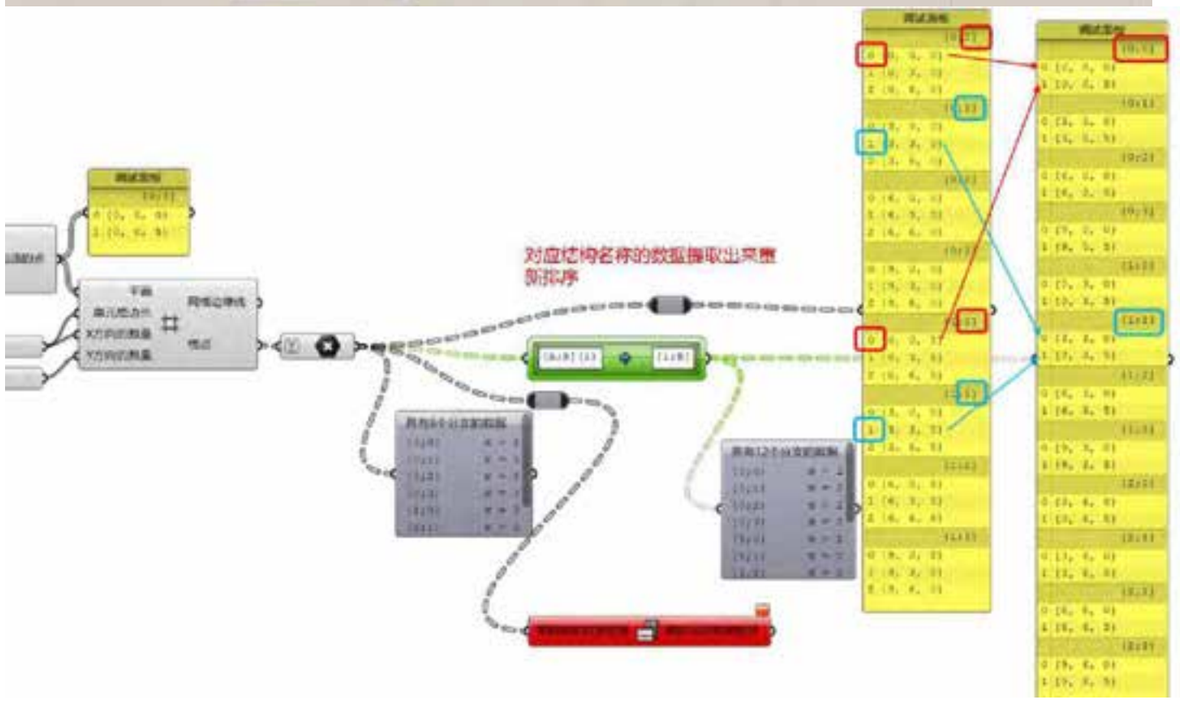
2、{A;B} (i)→{A;i}(B):当有 3 组数据结构时，翻转矩阵运算器就会失效。这时用路径映射器可以使两组点阵的连接顺序发生改变。下方实例为在 z 轴不同高度上，XY 方向有 6×4 的点阵。

（经测试，实际上输出端格式中（B）对结果没有影响，{A;B} (i)→{A;i}输出结果一样，而且更好理解）



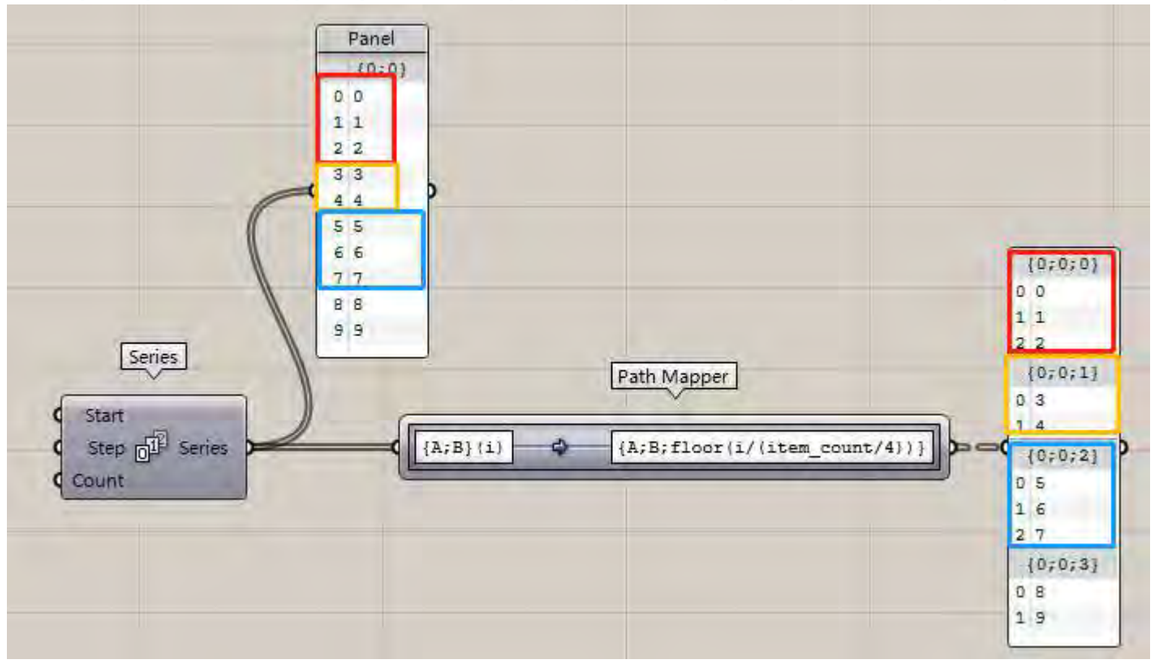
3、{A;B} (i)→{i;B}(A):相对应上一种路径映射，这种可以实现 z 轴方向的点互相连线。





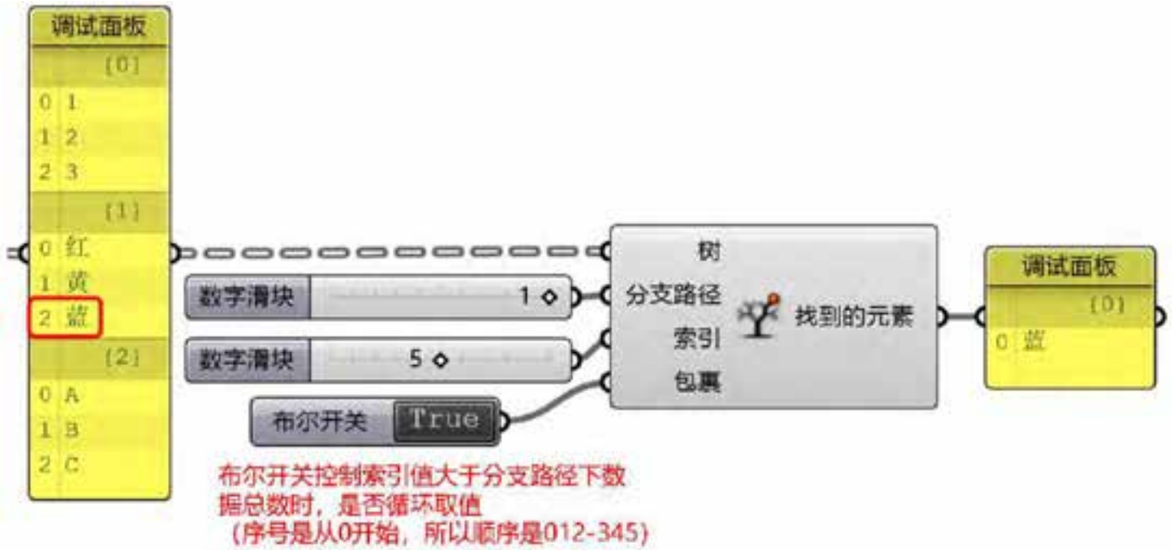
4、{A;B}(i)→{A;B;floor(i/(item_count/4))}：item_count 是当前数据分支内数据的数量（下面实例为 10），floor（x）函数就是取不大于 x 的最大整数，i 对应的是序号。

floor（0/2.5）=0；floor（1/2.5）=0；floor（2/2.5）=0；floor（3/2.5）=1；floor（4/2.5）=1；floor（5/2.5）=2；floor（6/2.5）=2；floor（7/2.5）=2；floor（8/2.5）=3；floor（9/2.5）=3；



查找树 Tree Item
树分支 Tree Branch
相对差异、相对差异（多项）Relative Item（s）

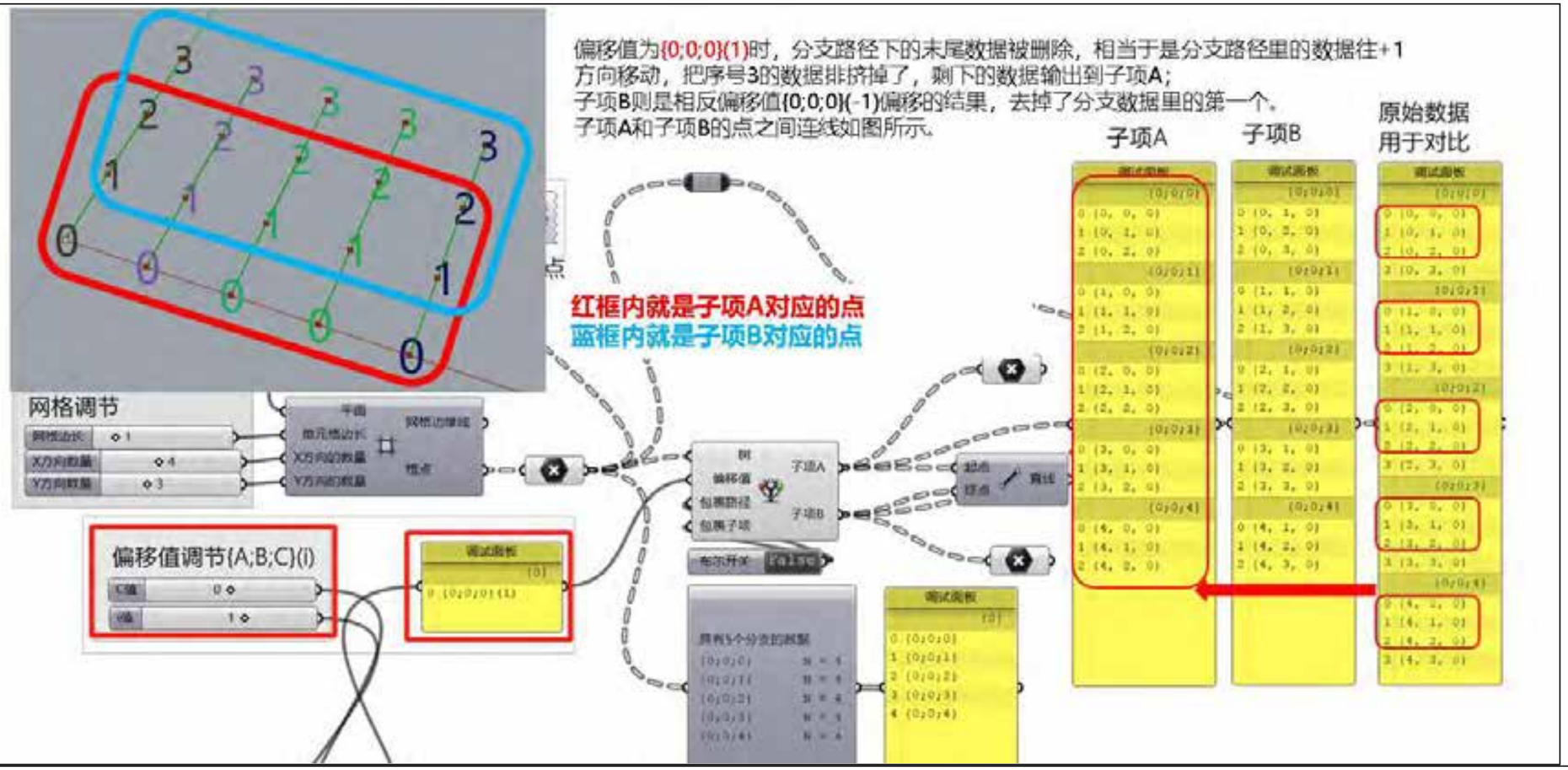
查找树 Tree Item：根据指定的分支路径 path 和索引（序号）index 抽取输入端树形数据中指定序号的数据



树分支 Tree Branch：根据输入的分支路径名称抽取指定分支的数据。

相对差异 Relative Item

首先要知道数据结构表达式为**{A;B;C}(i)**，i 为分支路径数据的序列号。设定偏移值为**{A;B;C}(i)**时，子项 A 为设定偏移值结果，子项 B 则是输出相反数偏移值**{-A;-B;-C}(-i)**的结果。



偏移值为{0;0;1}(0)时，整体数据最后一组被删除，相当于是整体数据按结构名称对应的C值+1的方向移动，把最后一组数据排挤掉了，剩下的数据输出到子项A；
子项B则是相反偏移值{0;0;-1}(0)偏移的结果，去掉了第一组数据。
子项A和子项B的点之间连线如图所示。

子项A　子项B　原始数据用于对比

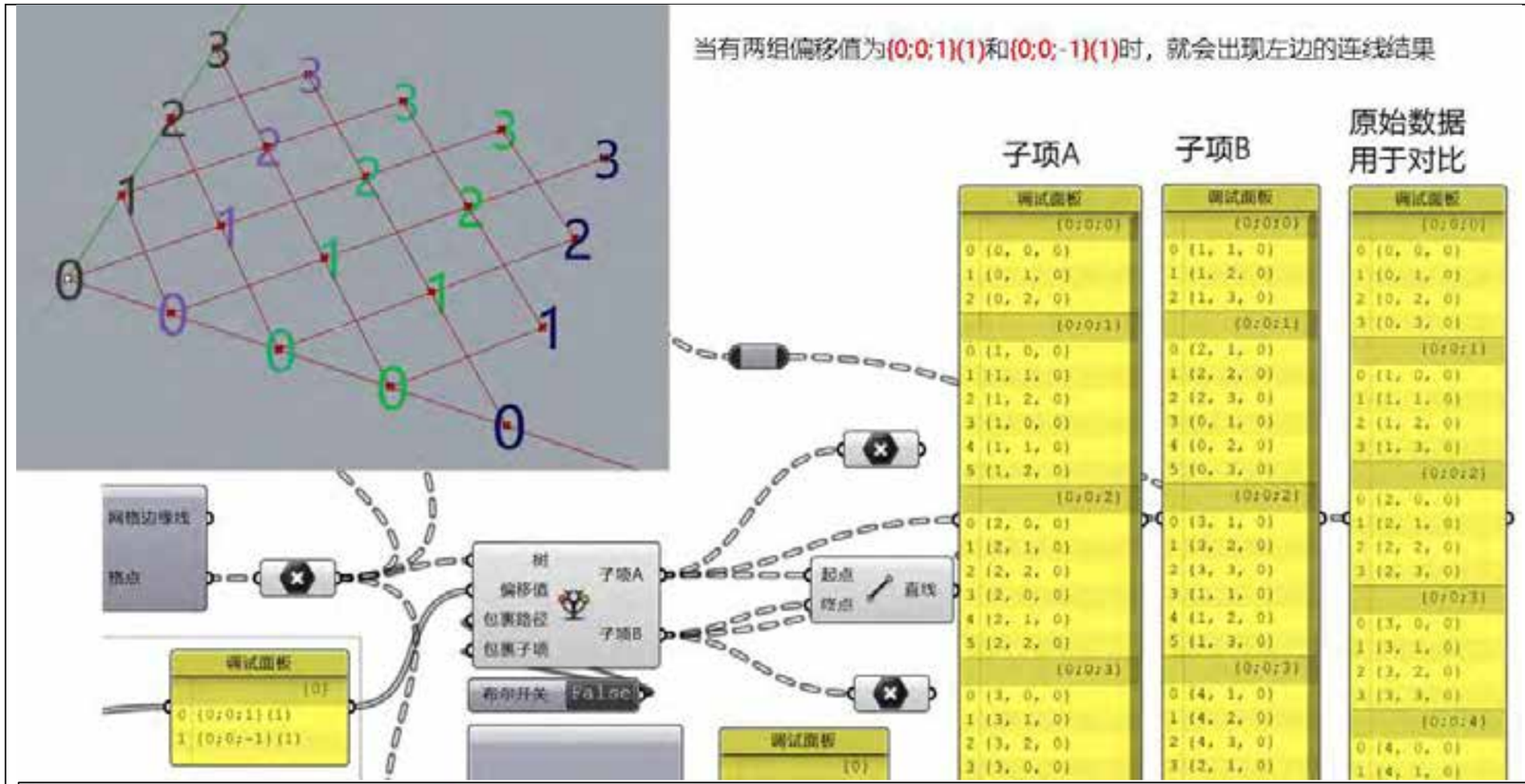
红框内就是子项A对应的点
蓝框内就是子项B对应的点

偏移值为{0;0;1}(1)时，就很好理解了，整体数据结构C对应的最后一组和分支路径 i 对应的末端序号都被删掉。
子项B则是相反偏移值{0;0;-1}(-1)偏移的结果。
子项A和子项B的点之间连线如图所示。

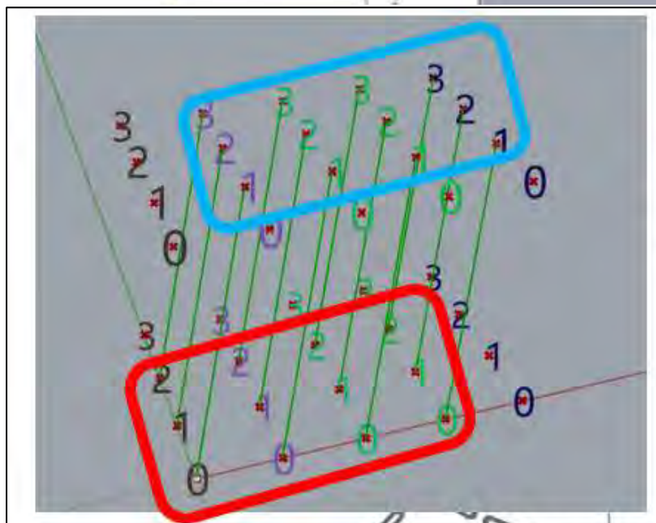
子项A　子项B　原始数据用于对比

红框内就是子项A对应的点
蓝框内就是子项B对应的点

当有两组偏移值为{0;0;1}(1)和{0;0;-1}(1)时，就会出现左边的连线结果



相对差异（多项）Relative Item



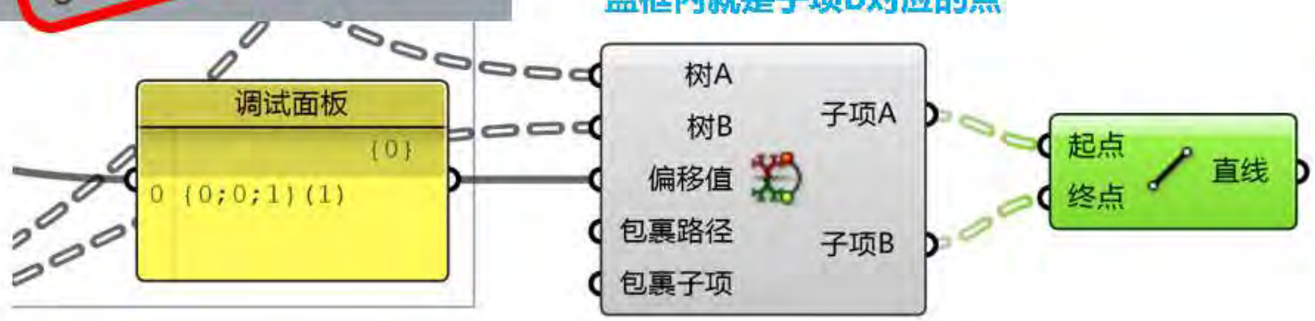
相对差异多项：

树A为下面的点阵，树B为上面的点阵

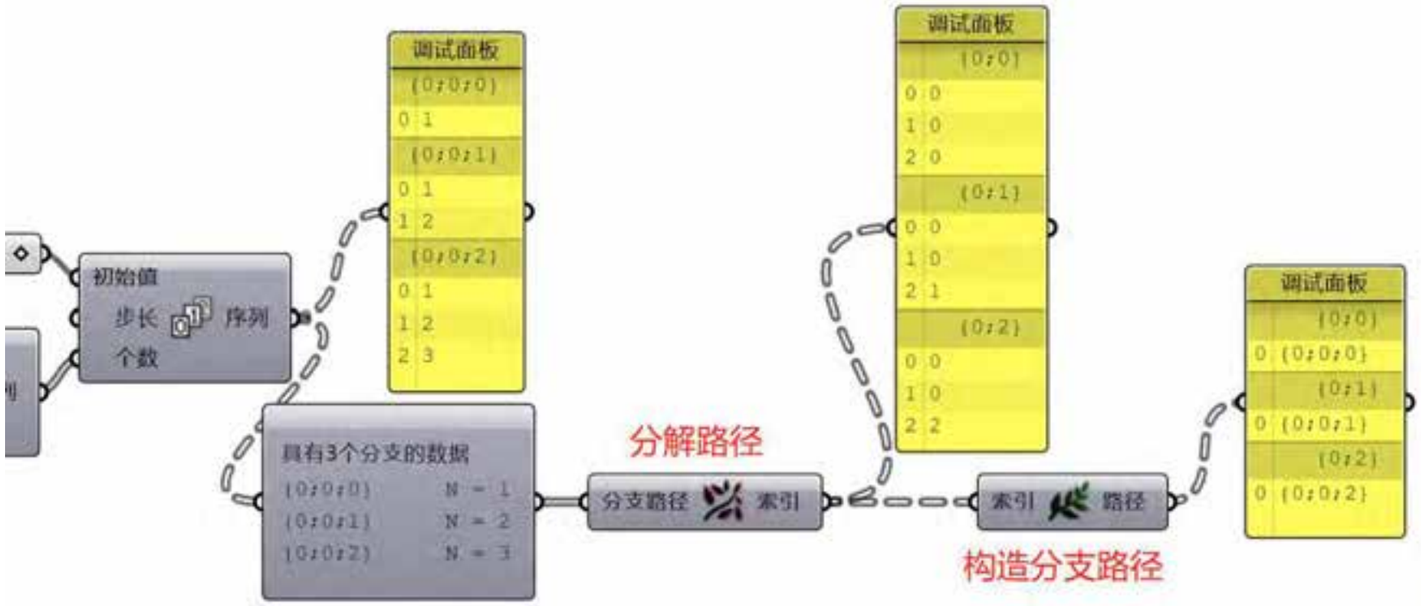
当偏移值为{0;0;1}(1)，子项A输出的是树A偏移后的点，子项B则是输出按偏移值{0;0;-1}(-1)偏移后的点，子项A与子项B之间的点连线如左图

红框内就是子项A对应的点

蓝框内就是子项B对应的点



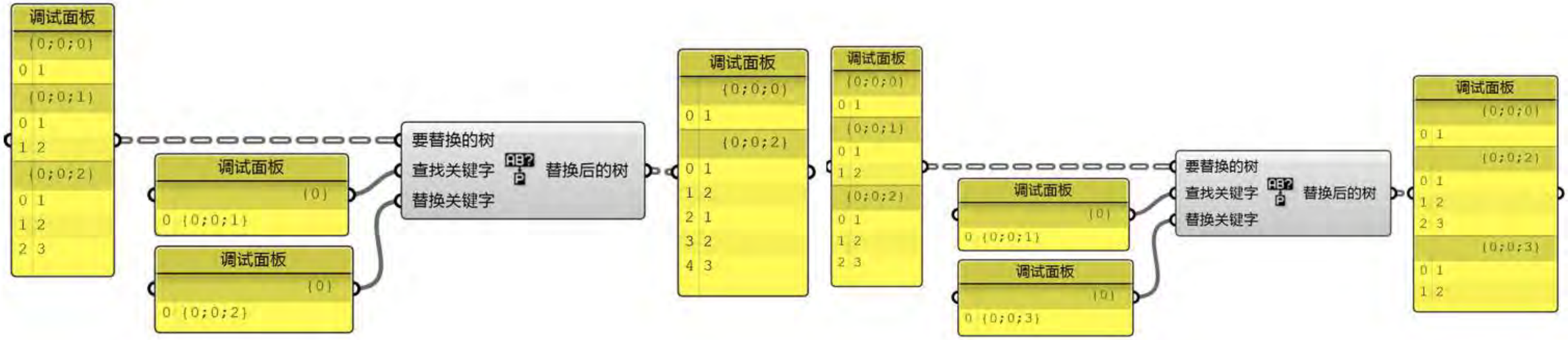
分解路径 Deconstruct Path 和 构造分支路径 Construct Path：前者把路径名称拆解成多个数字，后者把多个数字组合成数据路径名称。



路径比较 Path Compare：

根据自定义路径名称（路径掩码），查找要比较的路径中有没有名称一致的，返回布尔值

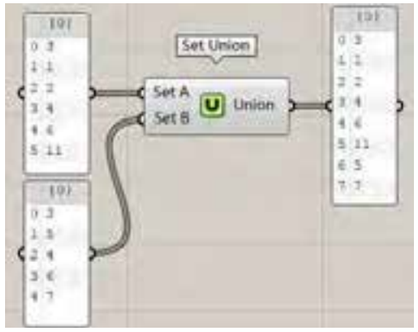
替换路径 Replace Paths：用替换关键字输入的自定义路径名称，替换掉查找关键字输入的路径名称，替换后的数据会按照新的路径名称重新排序，如果替换后存在两个相同路径名称，则两组数据会合并放在一起



集合 Sets-（集合）

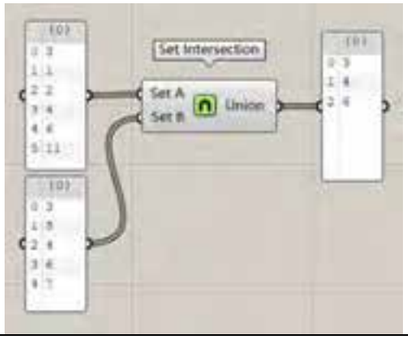
合并去重
Set Union

AB 两组数据合并，相同的数据合并成一个



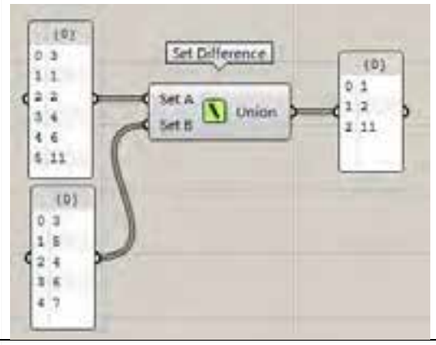
集合交集
Set Intersection

返回 AB 两组数据中都存在的数据。



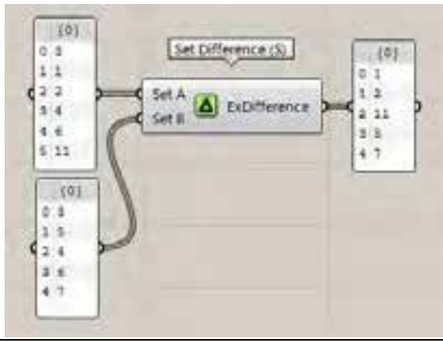
集合差集
Set Difference

返回 AB 两组数列中 A 有而 B 没有的数据。



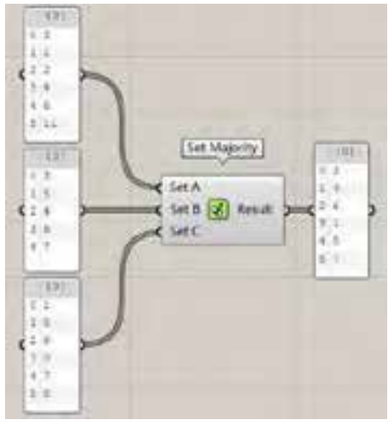
对称差集
Set Difference（s）

返回 AB 两组数据中只在 A 或只在 B 列表中存在的数据。



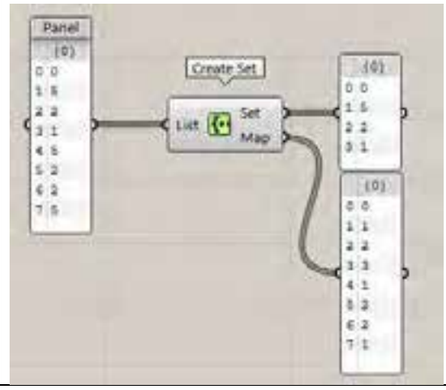
多次出现
Set Majority

返回三组数据中的出现过重复的数据



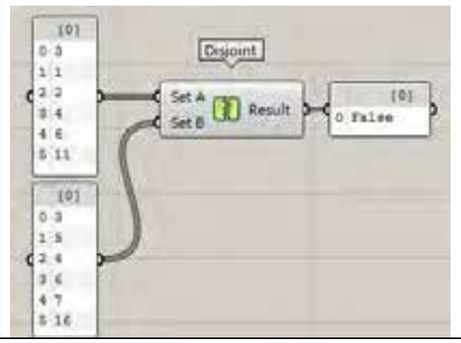
数据去重
Create Set

合并同类项，合并所有列表相同的数据，并返回相同数据的序号。注意列表端输入的数据类型可以是数字、矢量、布尔值等。



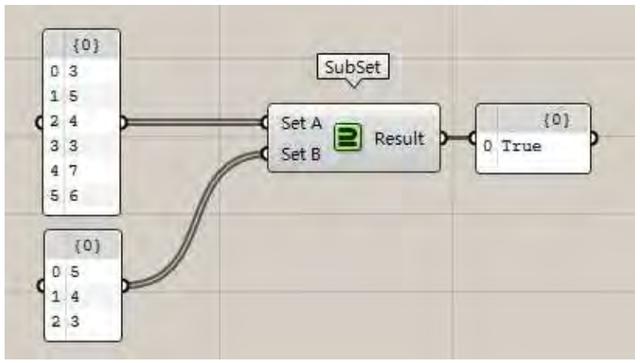
判断不相交
Disjoint

判断 AB 两组数据是不是不存在重复数据。如果没有相同数据，就返回 True；有相同数据，就返回 False。



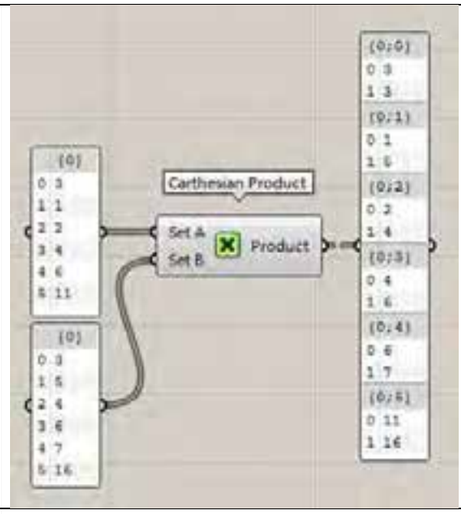
包含判断
SubSet

判断集合 B 的数据是否在集合 A 中存在，存在则输出结果 true



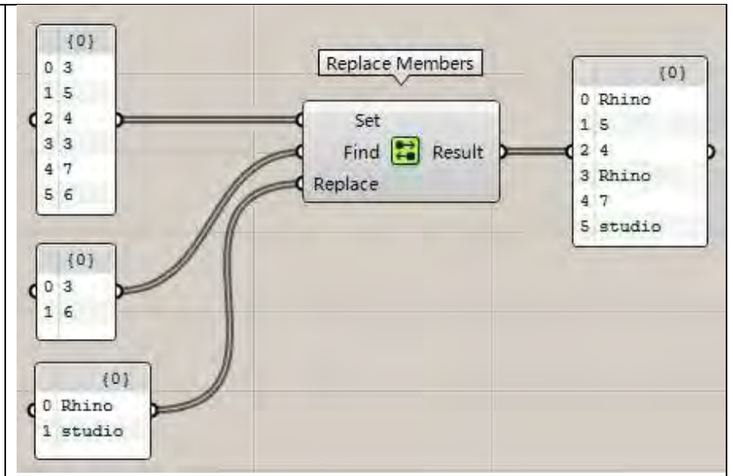
对应成组
Carthesian Product

将 AB 两组数据相同序号的数据放到一组内，要 AB 两组数据数量一致。



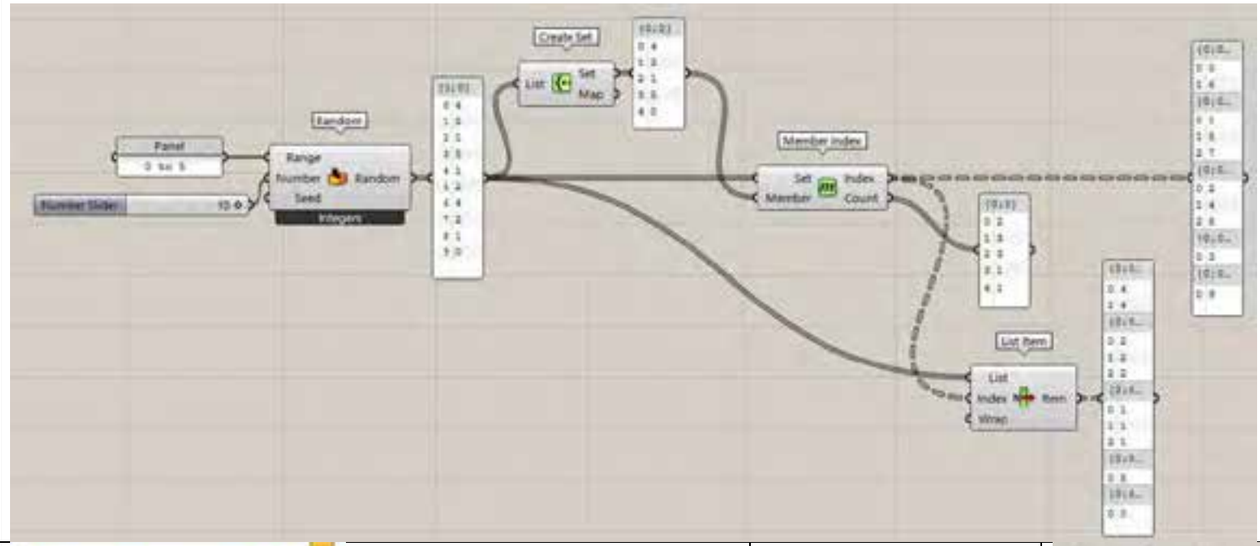
替换成员
Replace Members

查找要操作的集合中的要替换的项目并替换成给定的列表中的内容。要注意数据类型的一致性。



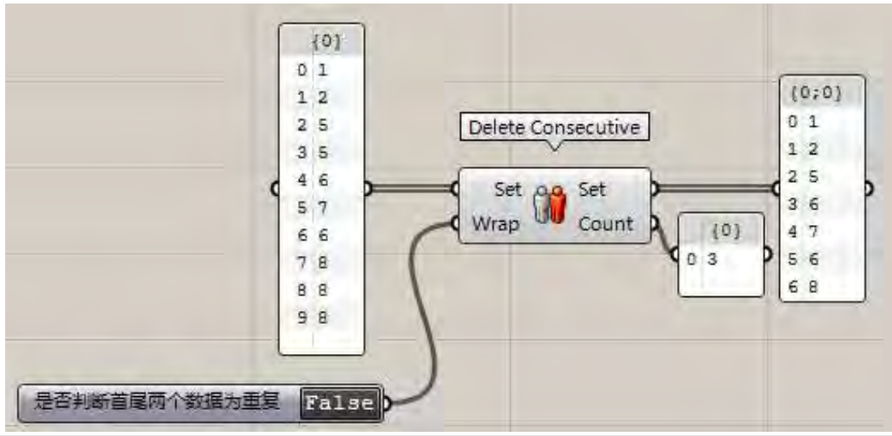
统计次数
Mermber Index

在要查找的集合中查找给定的成员（指定数据）所在的序号以及数量



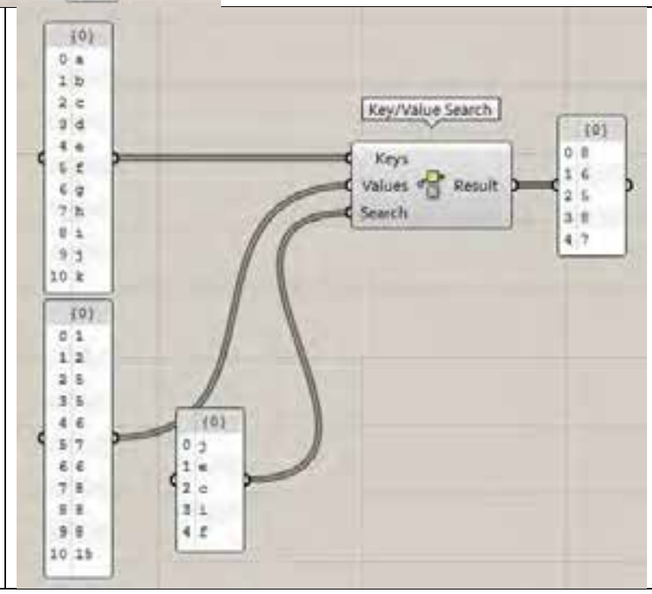
连续删除
Delete Consecutive

删除临近的重复的数据，并返回最大的重复值。



键/值匹配
Key/Value Search

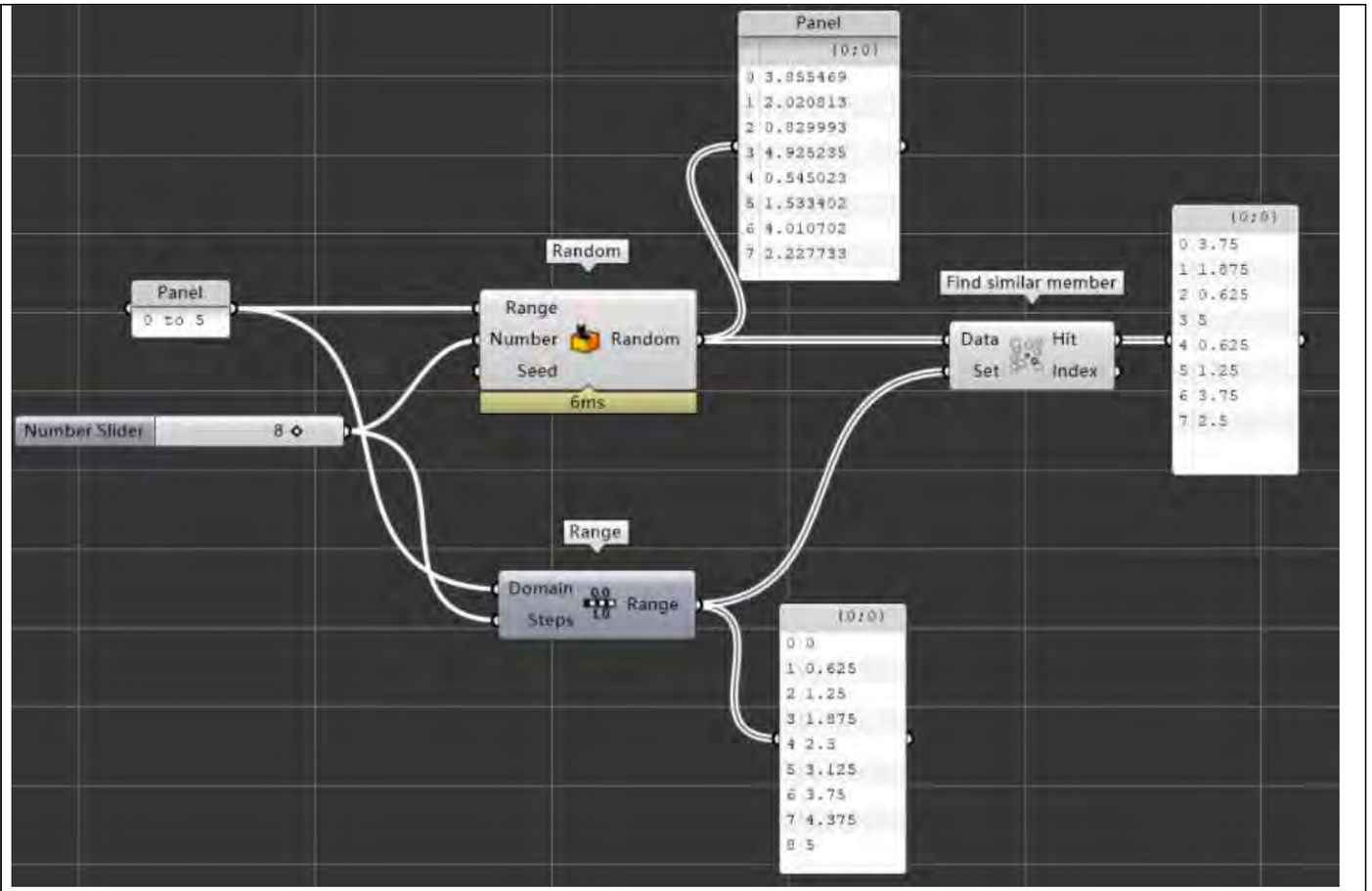
右图中，可以理解为将值（value）列表的序号替换为键（keys）中的字母，再根据对应的字母，查找提取值（value）中的数据



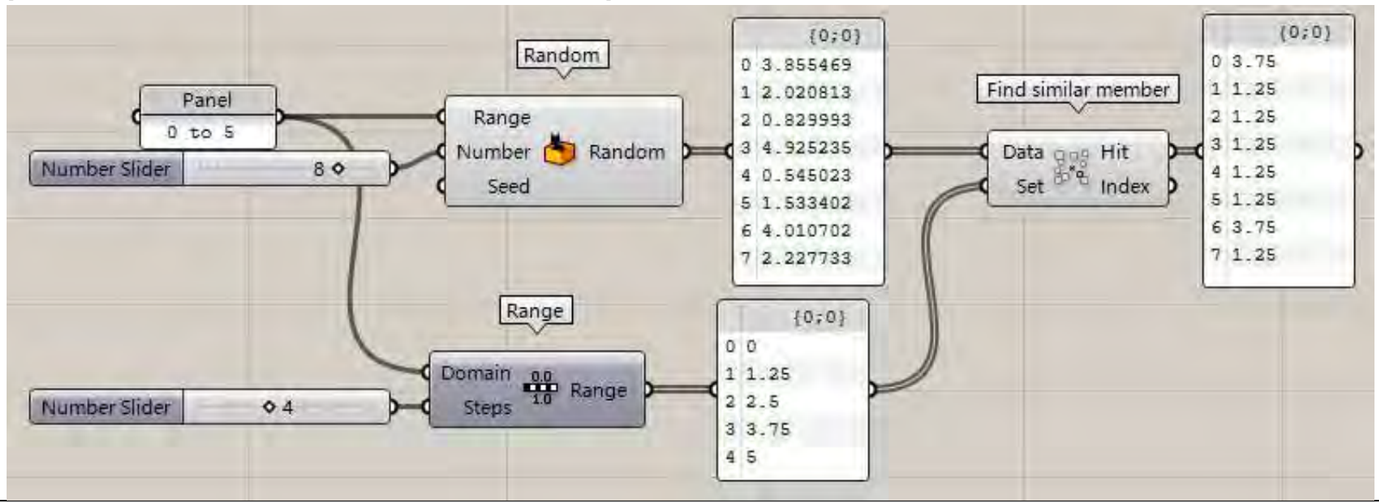
相似成员

Find similar member

在给定的数据集合(Set)中查找与要搜索的数据(Data)中最接近的数据，并用这个数据把要搜索的数据（Data）中的数据替换掉



在 Set 端中查找于 Data 端数据最接近的数据并替换，也可以用来做尺寸规格化哟~注意，data 端不能直接连接 panel，不然就会出现如下的错误，猜测可能是因为 panel 数据类型为 text 导致的错误。



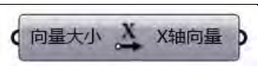
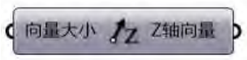
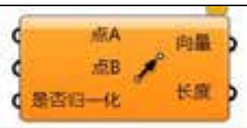
向量（Vector）

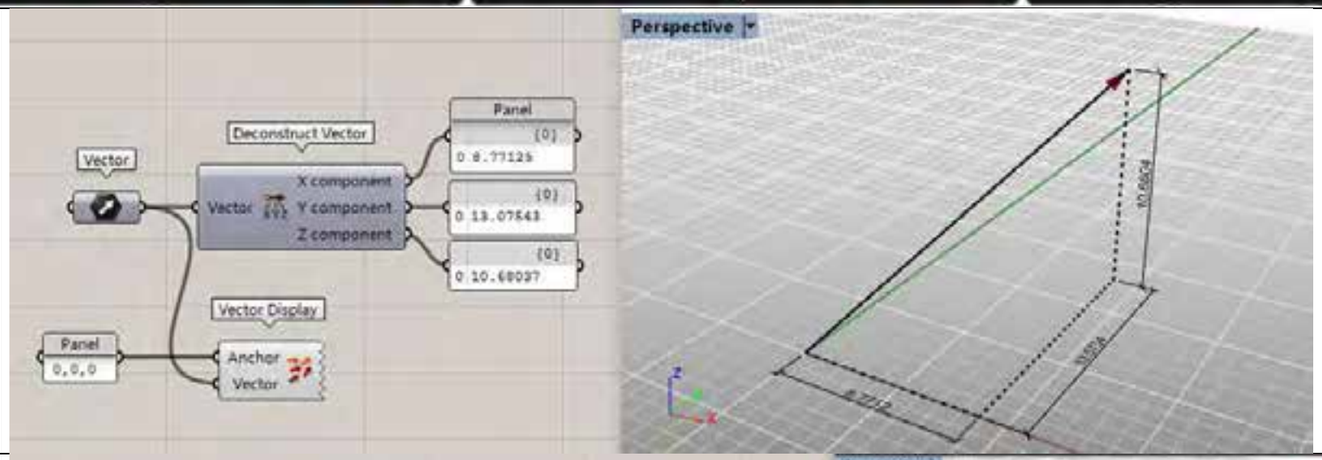
向量 Vector

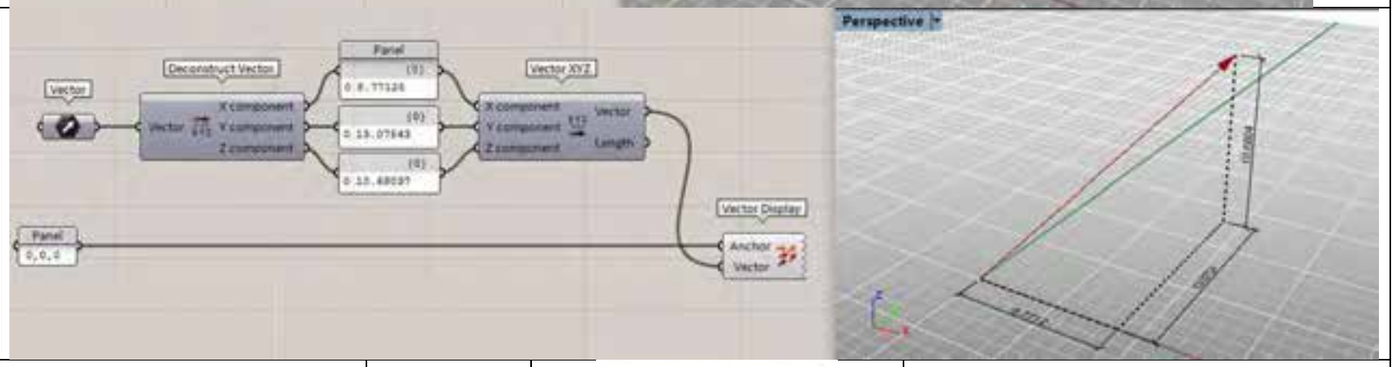
向量（Vector）

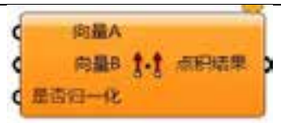
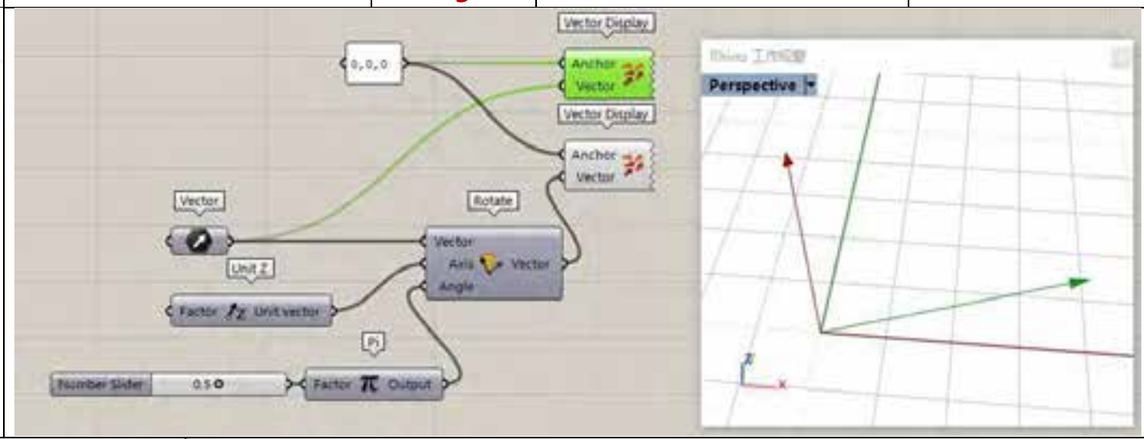
向量 Vector - （向量）

向量，是一种既有大小又有方向的量，又称为矢量。

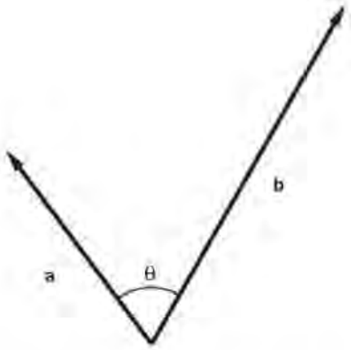
分解向量 Deconstruct Vector		**把向量拆解成 XYZ 轴上的投影长度**			
构造向量 Vector XYZ		**根据 XYZ 三轴长度来定义一个向量（Vector）。同时返回向量长度（Length）**			
X 轴向量 Unit X		**根据设定长度，创建一个以原点为起始点 X 轴方向的向量**	**Y 轴向量 Unit Y**		**根据设定长度，创建一个以原点为起始点 Y 轴方向的向量**
Z 轴向量 Unit Z		**根据设定长度，创建一个以原点为起始点 Z 轴方向的向量**	**向量归一 Unit Vector**		**将一个向量的长度改为 1，方向不变**
两点向量 Vector 2Pt		**生成点 A 到点 B 方向的向量（Vector），并返回向量长度（Length），是否归一化为布尔值**	**反转向量 Reverse**		**反转向量的方向**
向量叉积 Cross Product		**根据向量 A 和向量 B（也可以指定两条直线），求垂直于这两个向量所在平面的向量。（下方详解）** **向量积，数学中又称外积、叉积，物理中称矢积、叉乘，是一种在向量空间中向量的二元运算。与点积不同，它的运算结果是一个向量而不是一个标量。并且两个向量的叉积与这两个向量和垂直。**			





	向量振幅 Amplitude		**修改向量的长度（振幅大小）**	**向量点积 Dot Product**		**求向量点积，而向量点积其实就是两个矢量 xyz 值乘积的和，而向量有什么意义呢？事实上，向量的点积结果和两个向量之间的角度有关。（下方详解）**
	向量角度 Angle		**计算两个向量之间的夹角，反射角为 360 度减去夹角（下方详解）**	**向量长度 Vector Length**		**提取向量的长度信息**
	旋转向量 Rotate	**将向量按照指定的旋转轴和旋转角度进行旋转**				

向量点积 Dot Product： 求向量点积，而向量点积其实就是两个矢量 xyz 值乘积的和，而向量有什么意义呢？事实上，向量的点积结果和两个向量之间的角度有关。



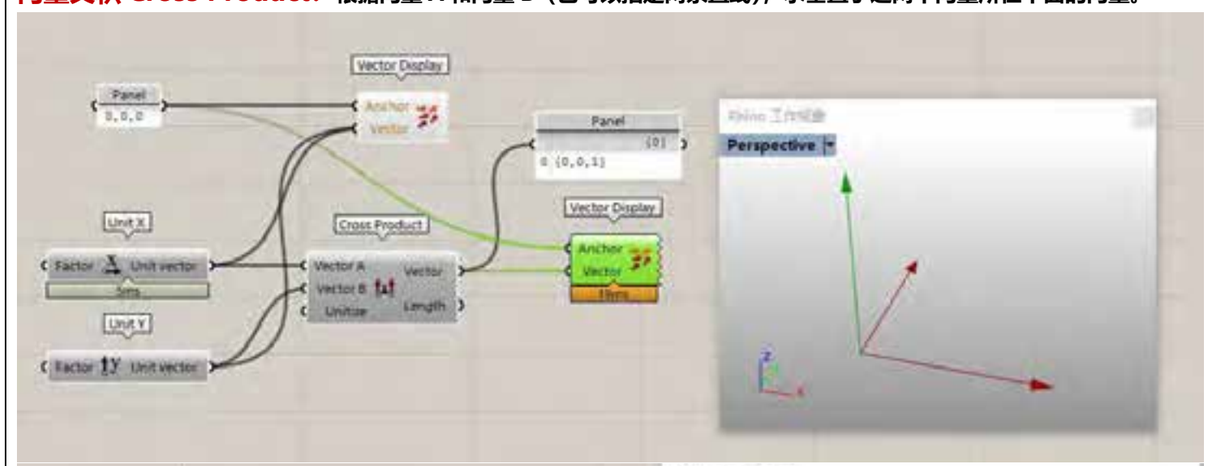
两个向量点积除了用这个公式外 $\mathbf{a}\cdot\mathbf{b}=\sum_{i=1}^{n}a_ib_i$，也可以用这个公式 $\mathbf{a}\cdot\mathbf{b}=\|\mathbf{a}\|\|\mathbf{b}\|\cos\theta$

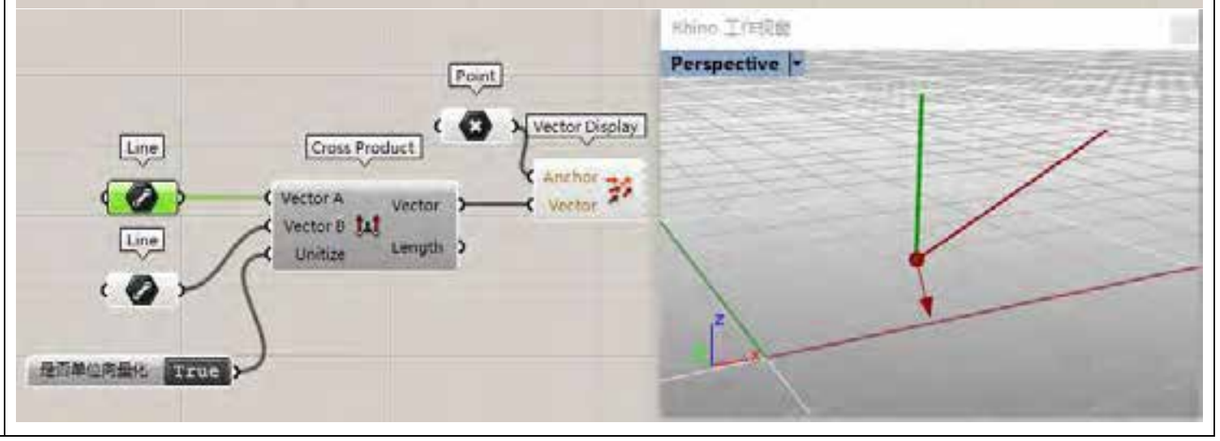
我们也很容易得到两个向量之间的角度计算公式 $\theta=\operatorname{acos}\left(\frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}\right)$

所以我们就知道了两个向量之间角度是锐角，那么点积大于0，垂直就等于0，钝角就小于0

a·b	θ	Angle is	a and b are
> 0	$0^\circ \quad \theta < 90^\circ$	锐角 (acute)	基本上是同个方向
0	$\theta = 90^\circ$	直角 (right)	perpendicular垂直
< 0	$90^\circ < \theta \quad 180^\circ$	钝角 (obtuse)	基本上是不同的方向

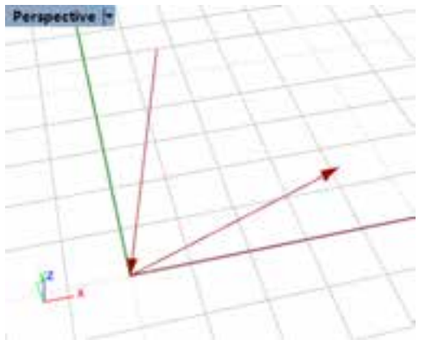
向量叉积 Cross Product：根据向量 A 和向量 B（也可以指定两条直线），求垂直于这两个向量所在平面的向量。



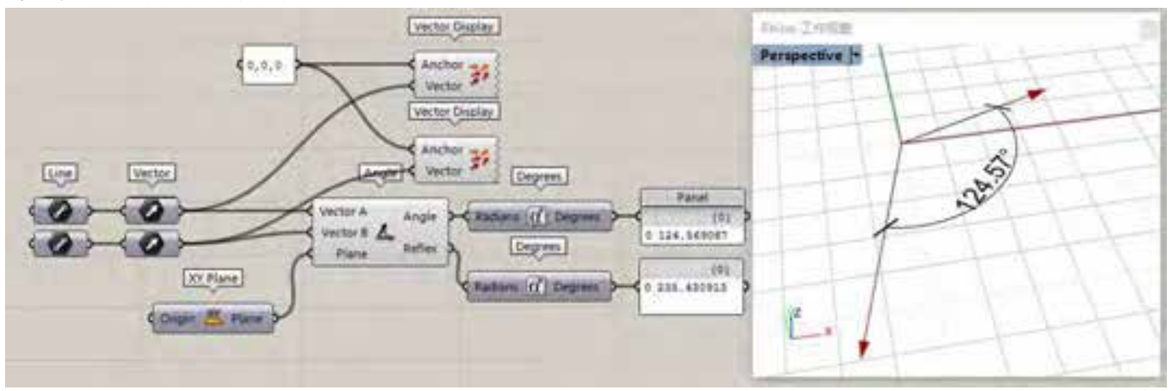


向量角度 Angle：

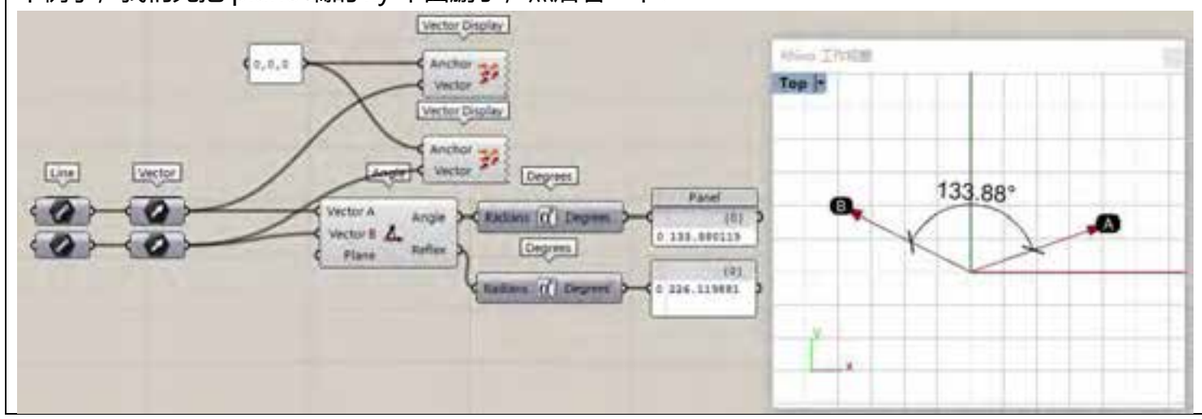
判断两个矢量之间的夹角，首先，矢量可以位移，所以如果两个矢量不在同一处，可以移动两个矢量到同一个端点处，其次，矢量是有方向的，所以如果两个矢量方向相反，比如下面的两个矢量：



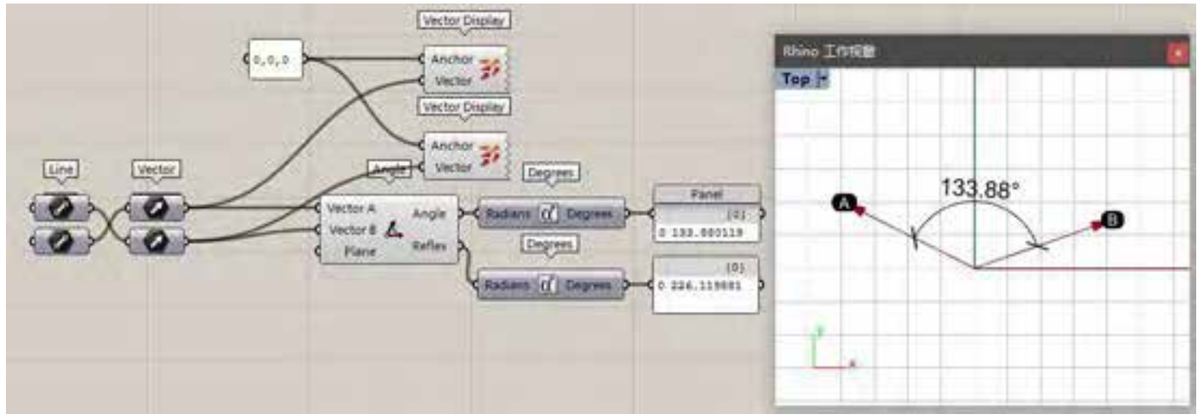
那么，这两个矢量的夹角就应该是：



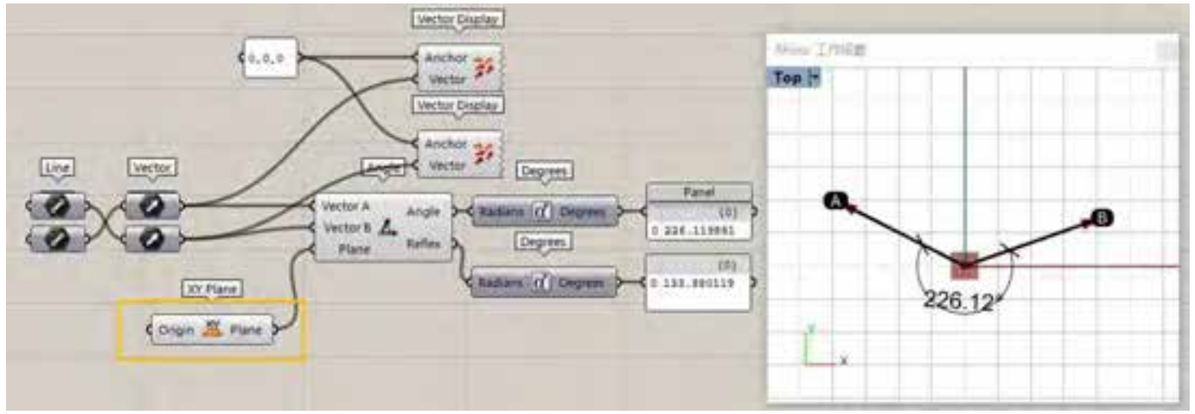
输出端 Angle 为夹角，输出端 Reflex 为反角。需要注意的是，两个矢量的顺序也会影响角度结果。举个例子，我们先把 plane 端的 xy 平面删了，然后看一下：



ok，这个时候我们调转一下 A 和 B 矢量，你会发现，角度依然是 133，并没有变，这是因为没有给定工作平面，所以依然按照两个之间的最小角来判断，也就是说，如果你不给定 plane 的话，两个矢量夹角永远不可能大于 180，这显然是不对的，因为有的时候我们就是需要判断超过 180 的角度。



所以这时候，我们加上限定平面 xy 平面，结果就显示正确了，运算器按照逆时针，判断 A 到 B 矢量的夹角。这样，才会出现超过 180 度的情况。

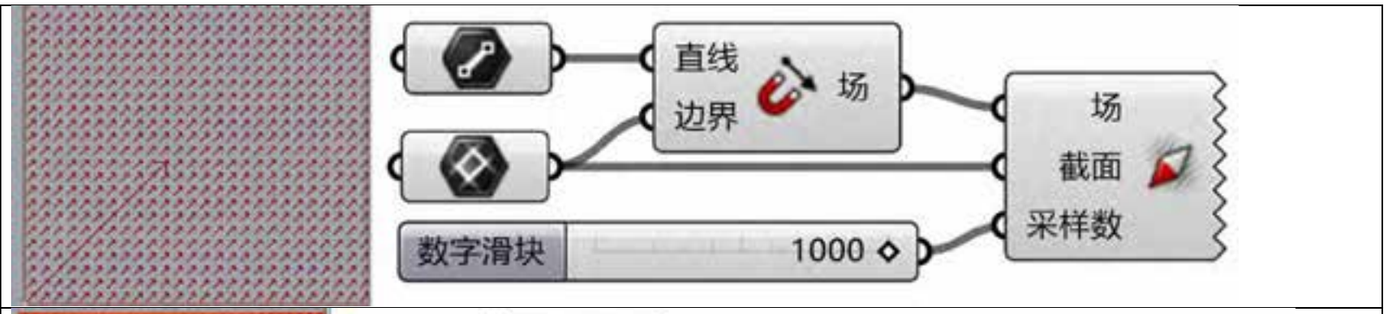


场 Field - (向量)

Field，即磁场，这个组里的运算器是用来模拟磁场效果的。

向量力场 Vector Force

根据一条直线的起始点作为向量，在设定的矩形范围内生成向量方向的磁场力，向量大小对应磁场力大小



旋转场 Spin Force

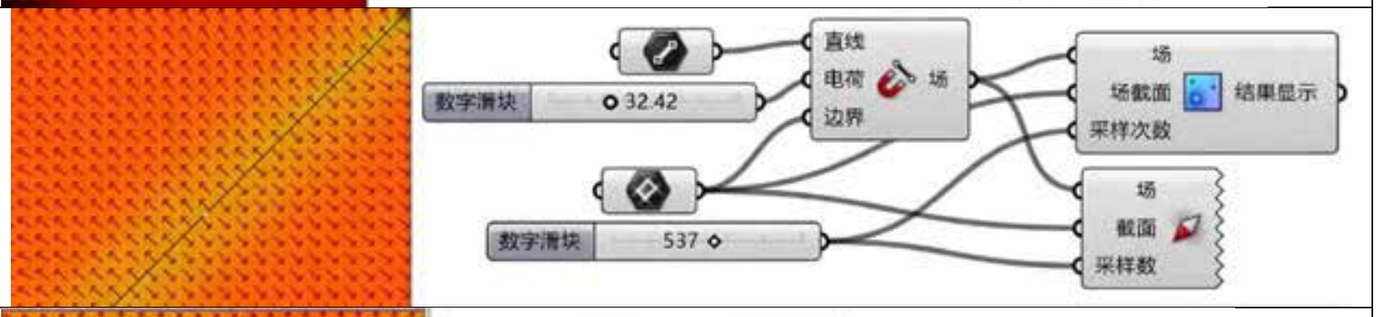
在设定的矩形范围内生成一个设定半径的螺旋磁场，可以调节磁场的强度、半径和衰减程度（可以通过张量显示和标量显示查看磁场的变化情况）

点电荷 Point Charge

根据指定点生成一个点磁场，矩形边界为磁场取样范围，可以分别调节电荷和衰变程度

线性电荷场 Line Charge

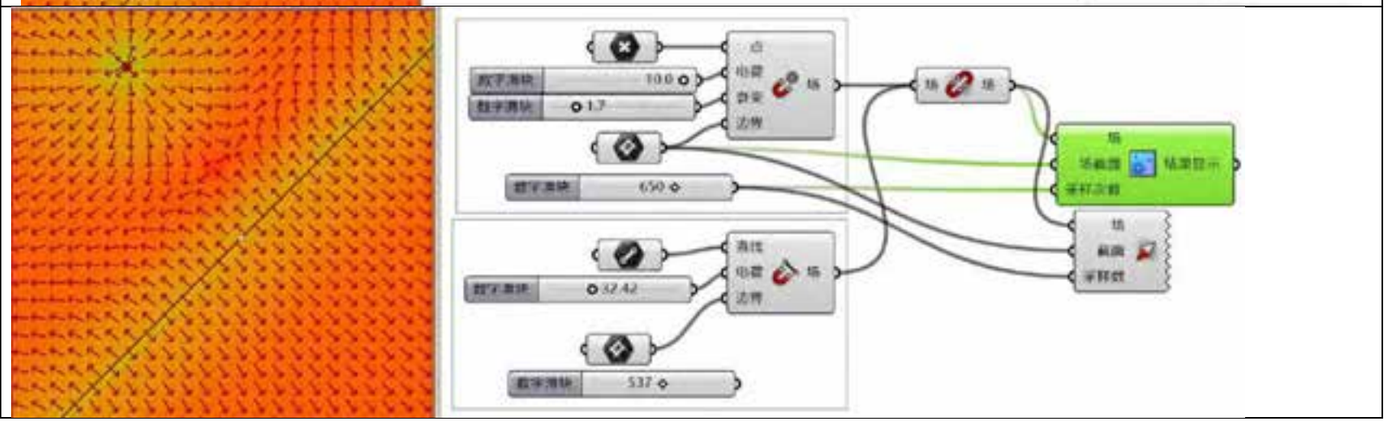
根据指定直线生成一个线性磁场，边界原理同上面一样，可以控制电荷强度

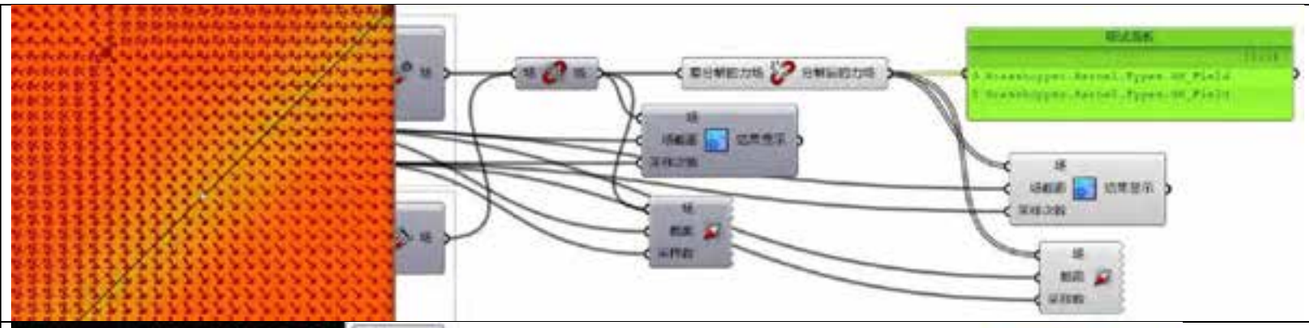
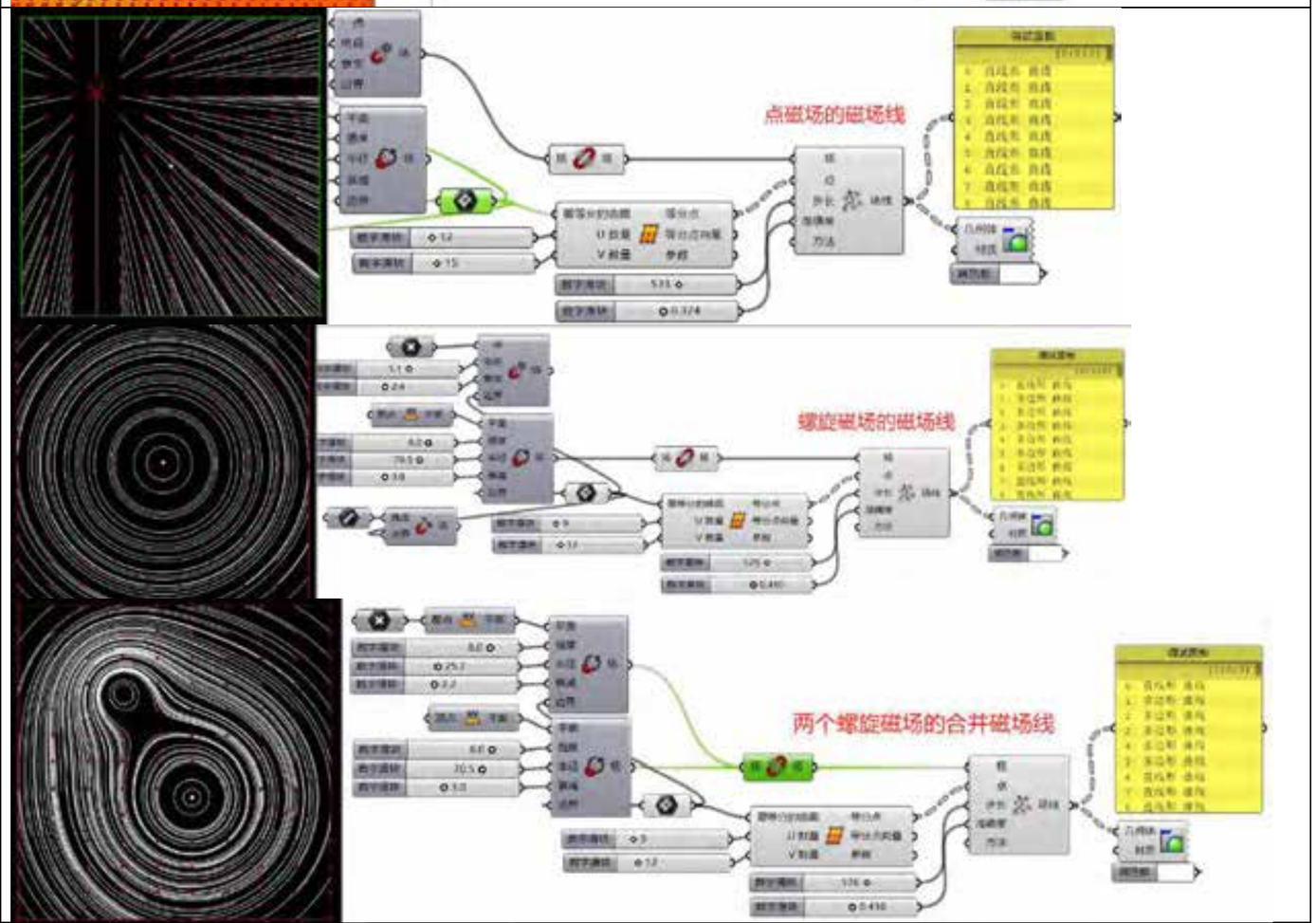
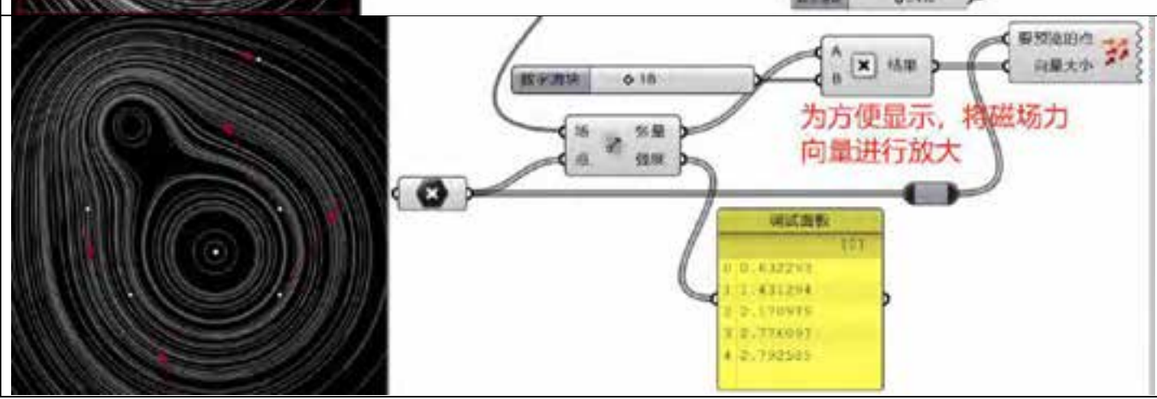


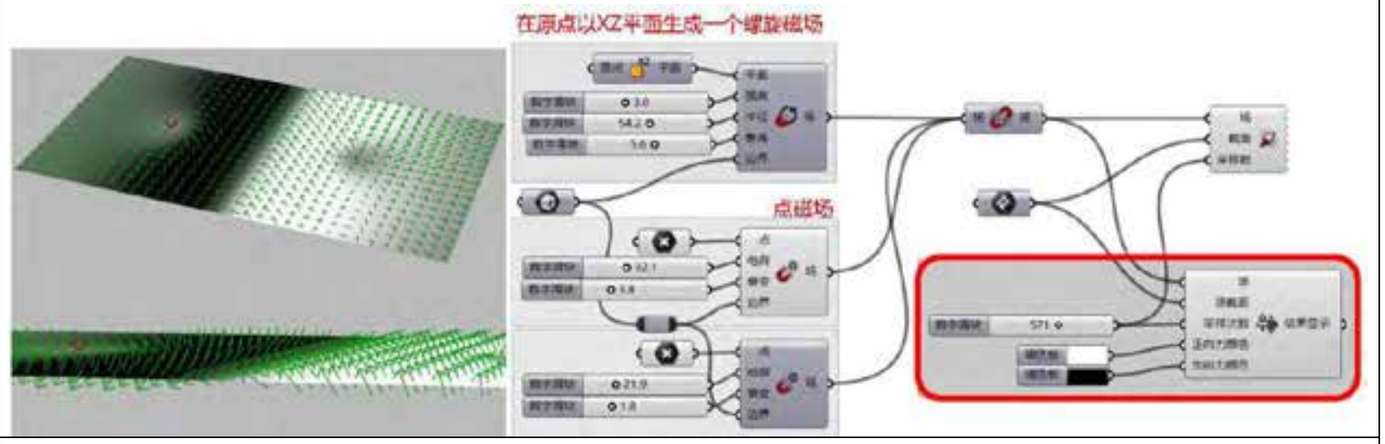
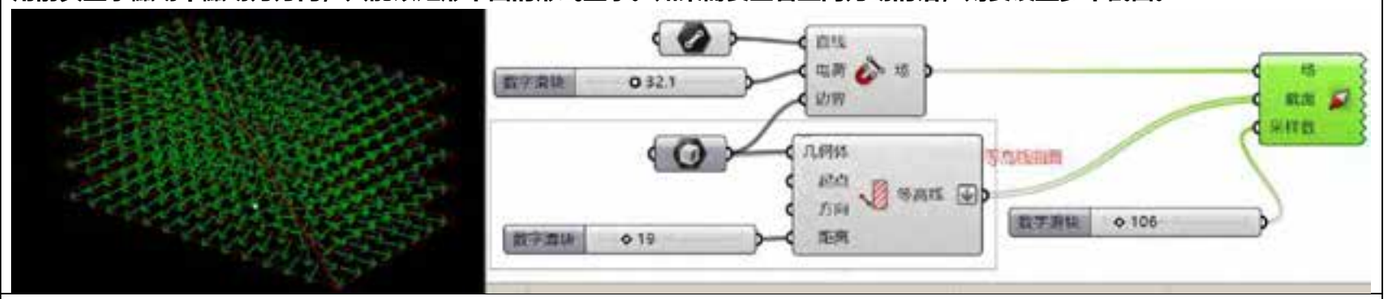
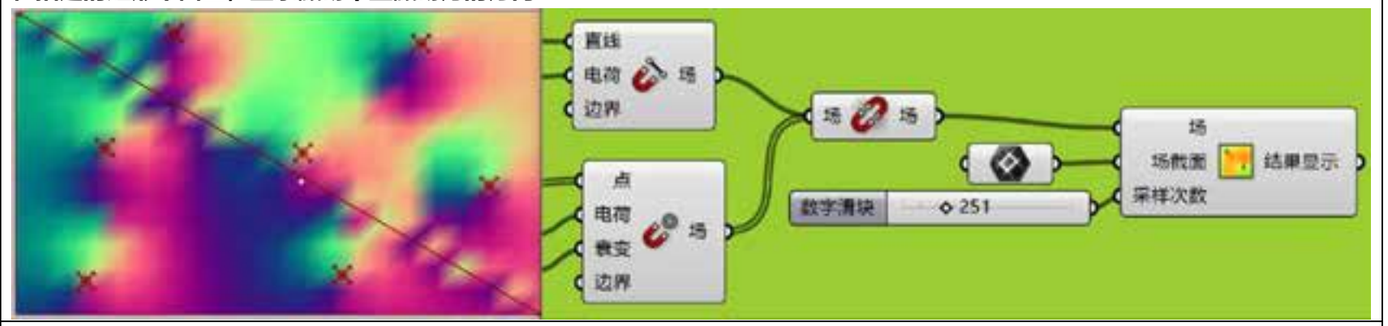
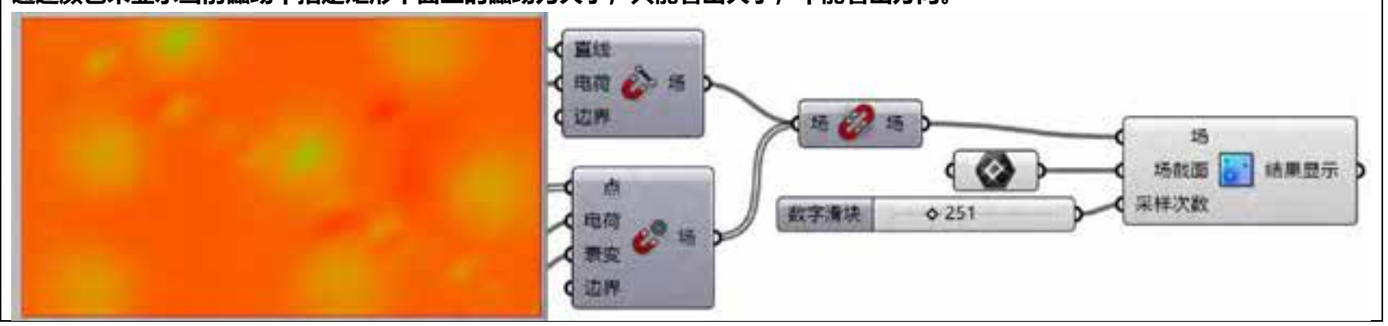
合并场 Merge Fields

将多个磁场合并，形成一个互相干扰的磁场

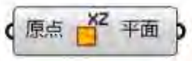
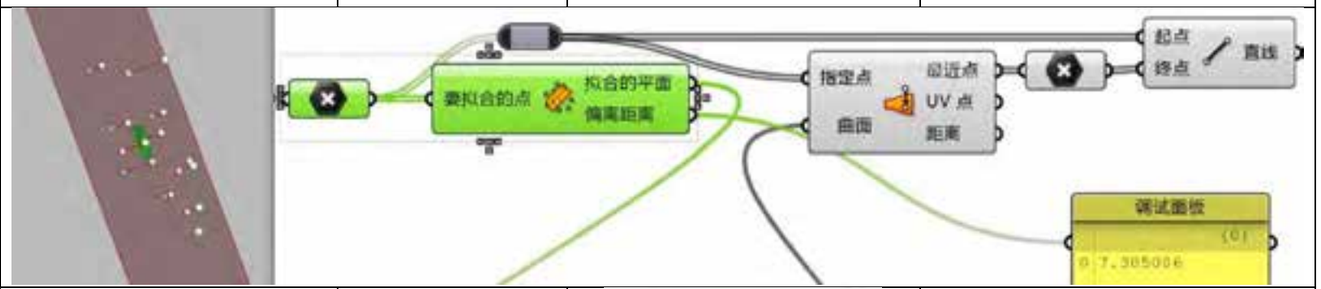
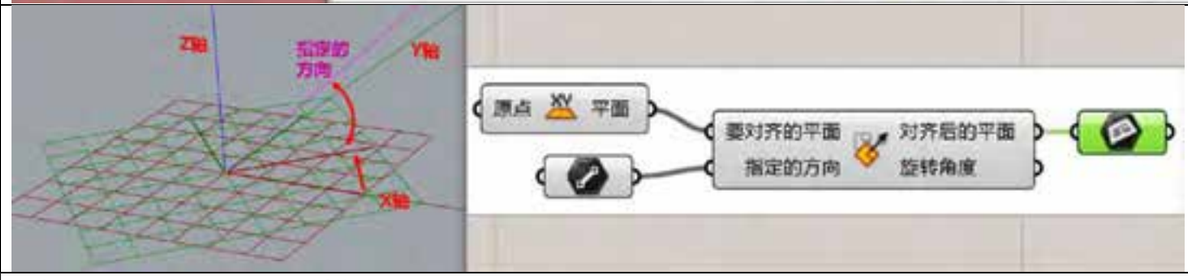
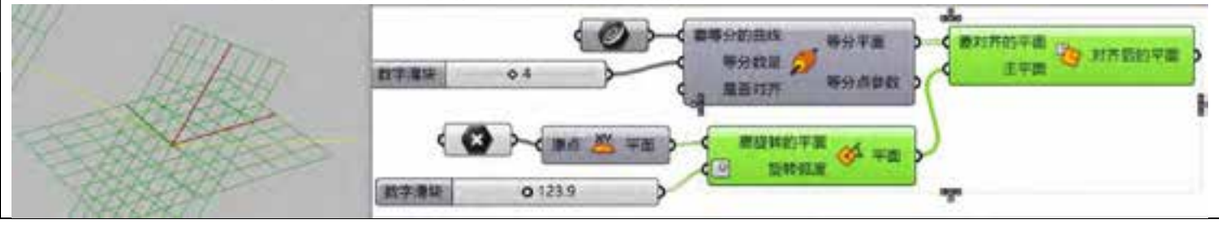
（右图为一个点磁场和线性磁场互相干扰的结果）

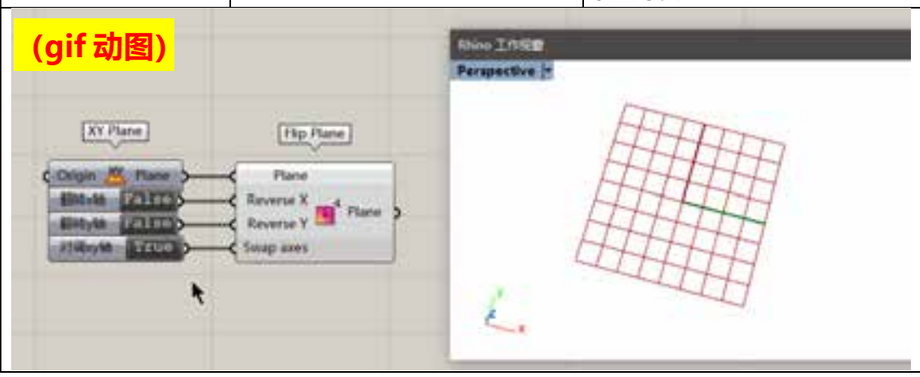


	分解力场 **Break Field**	 **将磁场分离，取消多个磁场之间的相互干扰，可以理解成把磁场放在不同次元空间，不再有互相作用**	
	场线 **Field Line**	 **显示磁场的磁场线，点对应的是磁场线的起始位置，步长（采样数）越高，精度越高，曲线准确度值越小，精度越高，方法端控制的是生成曲线阶数，默认就好，一般都不改。**	
	计算场 **Evaluate Field**	 **计算磁场中指定点的磁场力大小和方向，张量为磁场力向量，强度为磁场力大小**	

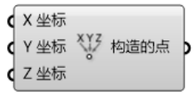
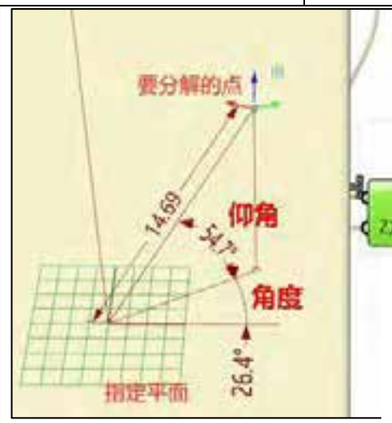
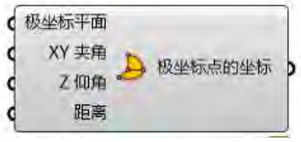
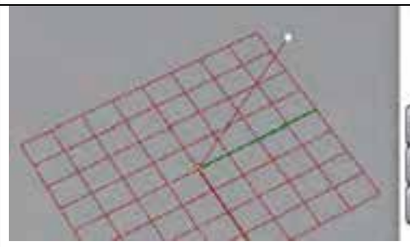
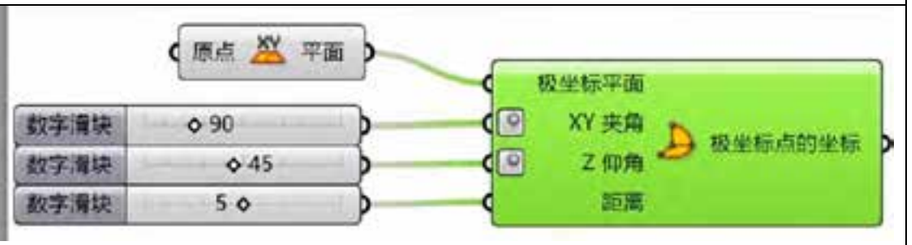
	垂直显示 **Perpendicular Display**		**用自定义颜色显示磁场中指定平面上的磁场力方向（默认向上为白色，向下为黑色）。磁场力是以空间的形式存在。** 下方实例为一个以 XZ 轴平面的螺旋磁场和两个点磁场的合并磁场中，点所在平面的磁场力方向，图中箭头可以看出磁场力在空间上的关系。 
	张量显示 **Tensor Display**		**用箭头显示磁场中磁场力方向，只能以矩形平面的形式显示。如果需要查看空间力场的话，则要设置多个截面。** 
	方向显示 **Direction Display**		**在指定的矩形平面上，显示磁场中里磁场力的方向** 
	标量显示 **Scalar Display**		**通过颜色来显示当前磁场中指定矩形平面上的磁场力大小，只能看出大小，不能看出方向。** 

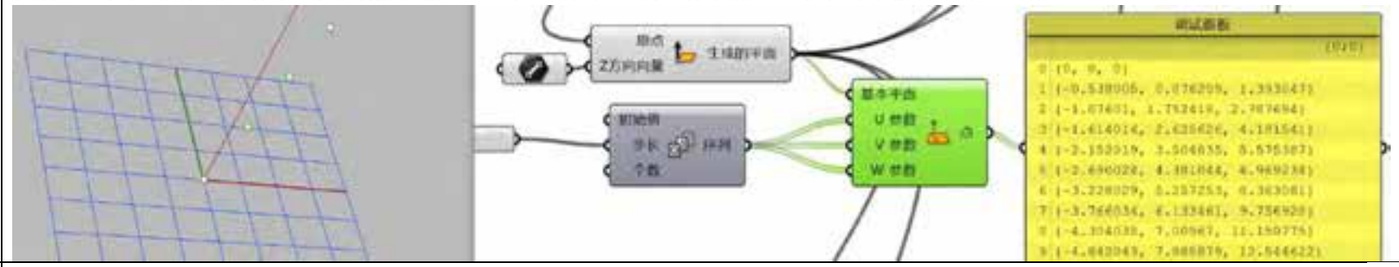
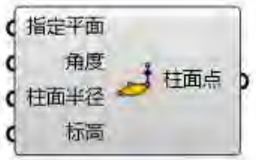
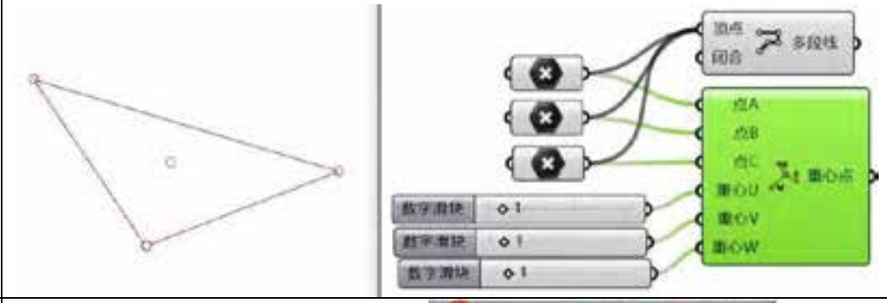
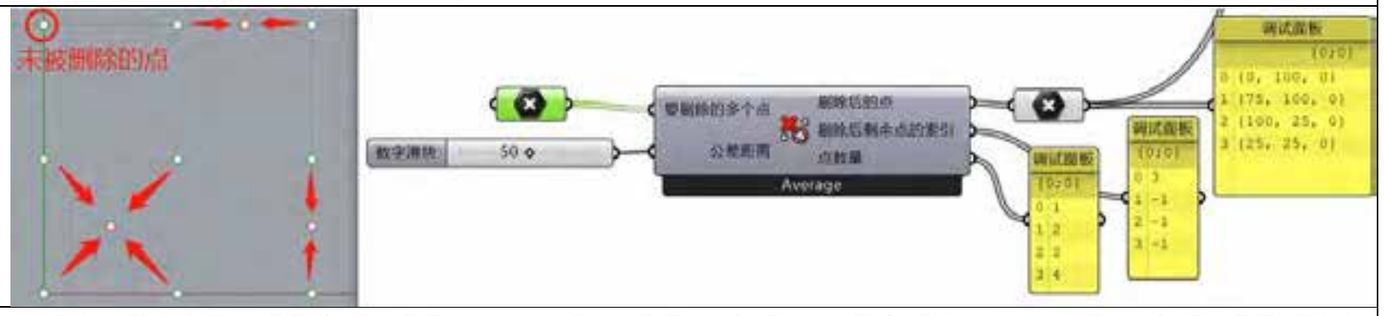
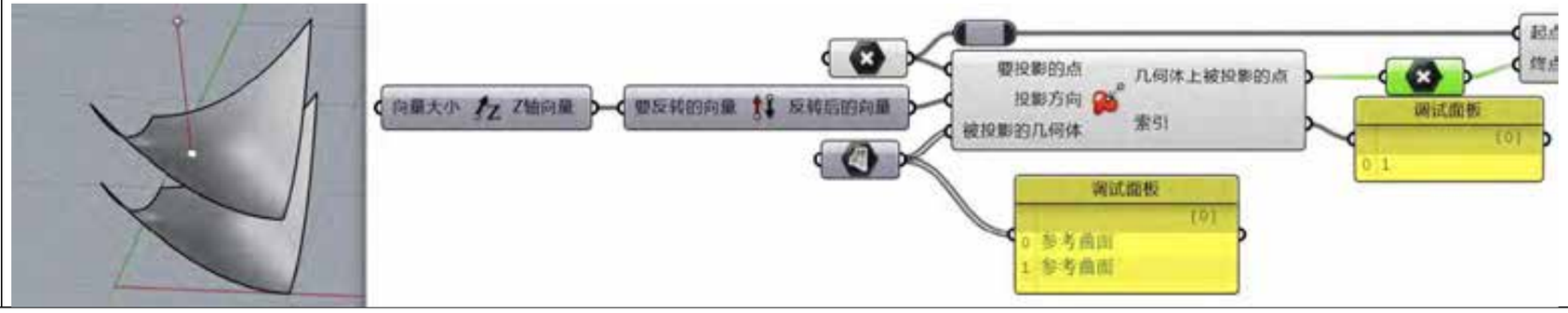
平面 Plane-（向量）

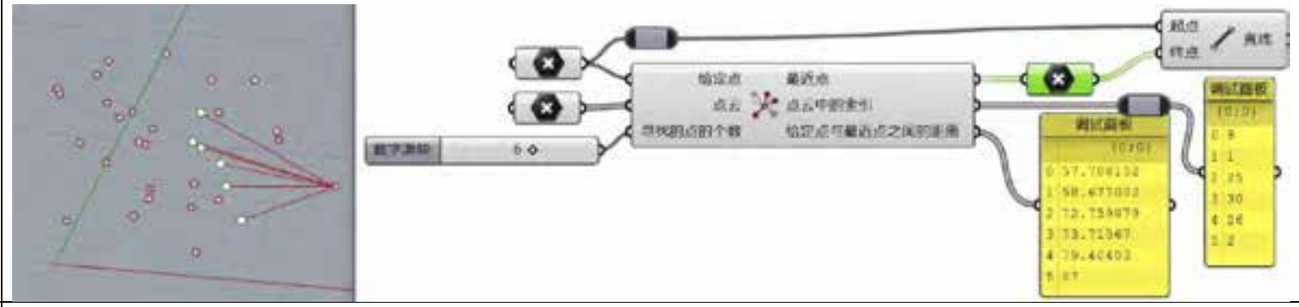
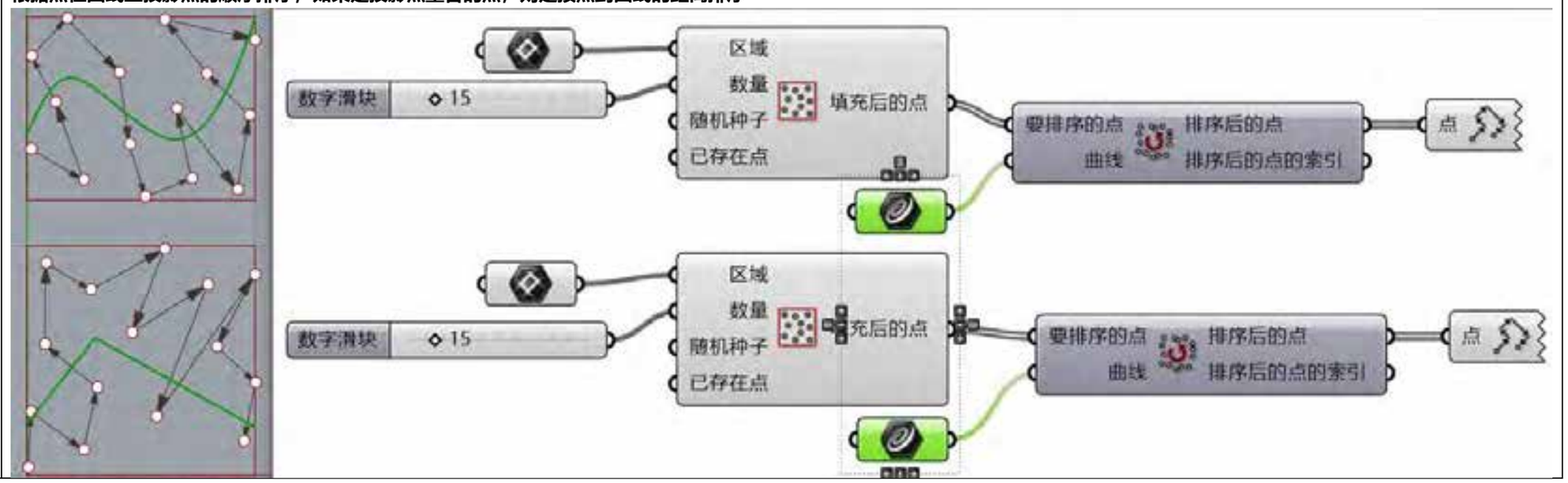
XY 平面 **XY plane**		**根据指定点（默认原点）创建 XY 轴平面**	**XZ 平面** **XZPlane**		**根据指定点（默认原点）创建 XZ 轴平面**
YZ 平面 **YZ Plane**		**根据指定点（默认原点）创建 YZ 轴平面**	**分解平面** **Deconstruct Plane**		**提取指定平面的原点和 XYZ 轴信息**
三点平面 **Plane 3Pt**		**根据三个点创建平面** **（三点共面原理）**	**垂直向量成面** **Plane Normal**		**根据指定点和 Z 轴向量，创建 XY 轴平面**
平面偏移 **Plane Offset**		**将平面按照设定偏移距离，沿平面的 Z 轴方向移动**	**直线+点平面** **Line + Pt**		**根据直线和点生成平面**
拟合平面 **Plane Fit**	 **根据一组随机点，生成一个最合适的平面**				
更改原点 **Plane Origin**		**将平面的原点修改到指定点** **（新原点）**	**构造平面** **Construct Plane**		**根据指定点和 XY 向量（也可以是直线）创建平面**
直线+直线平面 **Line + Line**	 **根据两条直线生成平面**	**当两条直线不共面时，则会生成一个以直线 A 为 X 轴，且与直线 B 平行的平面** 			
平面向量对齐 **Align Plane**	 **将平面沿 Z 轴旋转，使 X 轴与指定的方向之间夹角最小** （实际应用可以参考 Anemone 雨水路径模拟）				
批量对齐平面 **Align Planes**	 **将要对齐的平面（可以是多个面）绕自身的 Z 轴旋转，使自身 X 轴与主平面的 X 轴之间夹角最小**	**下方实例主平面为 XY 平面，通过旋转 XY 平面，观察要对齐的平面的变化** 			

	平面最近点 **Plane Closest Point**		求指定点（采样点）到投影平面的最近点及相关参数，投影点是在世界坐标里的坐标值，UV 点是投影点在投影平面上的坐标	**平面坐标** **Plane Coordinates**		求指定点在指定平面的坐标系里的坐标参数
	旋转平面 **Rotate Plane**		使指定平面（要旋转的平面）绕自身 Z 轴按照指定的角度（旋转弧度）旋转	**调整平面** **Adjust Plane**		将要调整的平面的 Z 轴与指定的 Z 轴向量方向调至一致（互相平行），原点位置不变
	翻转平面 **Flip Plane**		翻转平面对应的 X 轴和 Y 轴的方向，或者是 X 轴和 Y 轴互换位置，输入端下面三项为布尔值			

点 Point-（向量）

分解点 Deconstruct		**提取点的 XYZ 轴坐标值**	**构造点 Construct Point**		**根据指定的 XYZ 坐标值设定一个点，默认为原点**
数字转点 Numbers to Points	**将数字列表按设定的规则（默认 XYZ）转化成点的坐标值，数字列表必须是 3 的倍数，否则会报错**				
点转数字 Points to Numbers		**跟数字转点作用正好相反，将点的坐标值拆解成数字列表，默认规则 XYZ（见上图）**	**两点距离 Distance**		**计算两个点之间的距离**
分解点到极坐标 To Polar	**求指定点（要分解的点）到指定平面的相关参数，角度为点在平面上的投影点和平面原点之间的连线，与平面 X 轴的夹角，仰角为点和原点之间连线，与平面之间的夹角，距离为点到平面原点的距离。右键选择【pole down angles】，仰角输出的角度为仰角的余角**（如果两个角的和是一个直角，那么称这两个角互为余角，简称互余）	 **（跟下面的柱面点运算器原理相似，只不过采用的参数不一样。两者角度含义相同，其他两项分别对应由点及其投影点、原点构成的直角三角形的角和边长）**			
极坐标点 Point Polar	 **与分解点到极坐标正好相反，XY 夹角对应后者的角度，Z 仰角对应仰角，距离定义一致**	 			

定向点 **Point Oriented**		**在指定基本平面所在的坐标系里，根据指定的 UVW 值（相当于世界坐标系的 XYZ 值，可以输入数列）生成点** 
柱面点 **Point Cylindrical**	 **根据圆柱坐标系对应的参数生成点**	
重心 **Barycentric**		**计算三点对应的三角形的重心，重心 UVW 是建立在重心坐标上的值。(重心坐标下方有详解)** 
剔除重合点 **Cull Duplicates**		**按照设定的公差距离，把要剔除的多个点间距在公差距离范围内的合并成一个点，输出剩余的点和合并点；剔除后剩余点的索引为未被删除点的序列号（在原来的点数列中的序列），删除的点序号会合并成-1；点数量为合并点是由几个点合并。** 
投影点 **Project Point**		**要投影的点按照投影方向（向量，默认为 Z 轴的反向，即垂直向下）投影到被投影的几何体上，输出投影点（几何体上被投影的点）及被投影物体的序号（索引），被投影的几何体可以有多个，可以用来筛选出最接近指定点的物体。** 

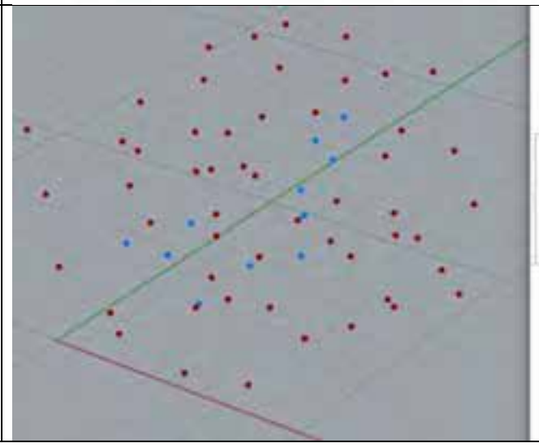
最近单点 **Closest Point**		**找出一堆点（点云）中距离指定点（给定点）的最近点，输出相应参数**
最近多点 **Closest Points**		**找出点云中离给定点最近的指定个数的点，并按距离远近将最近点的序号（点云中的索引）输出** 
点之坐标排序 **Sort Points**		**把要排序的点根据 XYZ 坐标值依次排序，先按照 X 坐标最接近的排序，X 坐标相同的点则按照 Y 坐标排序，XY 坐标都相同的则是按 Z 坐标排序**
点之曲线环绕排序 **Sort Along Curve**		**根据点在曲线上投影点的顺序排序，如果是投影点重合的点，则是按点到曲线的距离排序** 

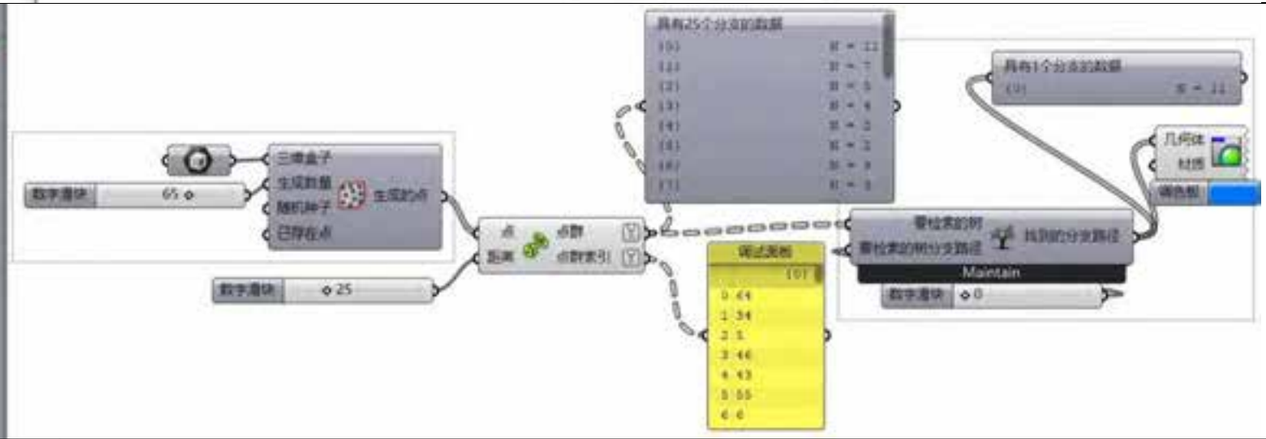
点拉回
Pull Point

计算指定点到几何体（曲线或者曲面）的最近点，并输出点到最近点之间的距离（多用于曲线干扰）

点群
Point Groups

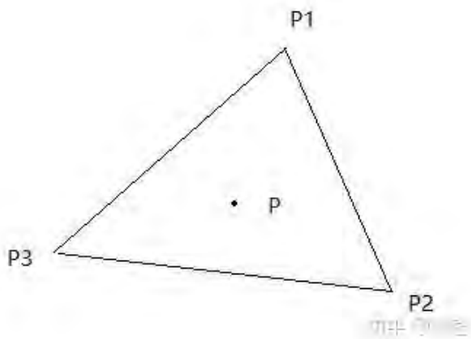
根据指定距离把点群对应的数据划分到不同的分支路径下面。输出对应点分组后在原数列里的序列号。
经实际操作，暂时没看明白距离跟数据分组规则是怎样一种关系，似乎也不是按点与点之间的距离。





重心坐标（Barycentric coordinates）：

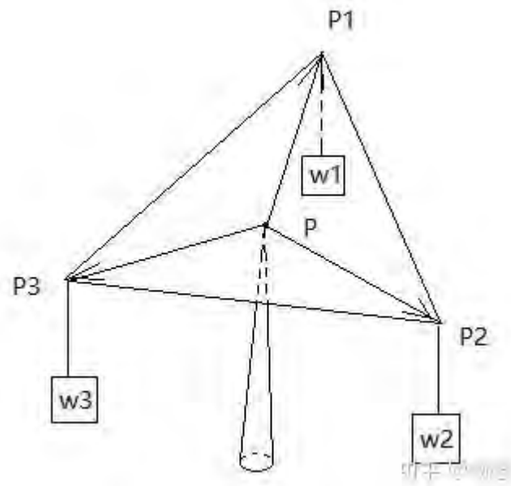
一，啥是重心坐标：



如图，对于空间三角形 P1P2P3 内任一点 P，必定唯一存在三个数 w1,w2,w3，满足：w1+w2+w3=1
P=w1*P1+w2*P2+w3*P3 （即 P 表示成 P1,P2,P3 的线性组合）
则(w1,w2,w3)就称为此三角形上 P 点的（归一化）重心坐标。
二，为啥叫重心坐标：
对 P=w1*P1+w2*P2+w3*P3 变形得 w1(P1-P)+w2(P2-P)+w3(P3-P)=0，即：

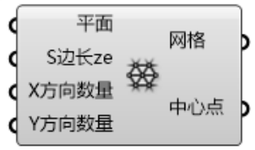
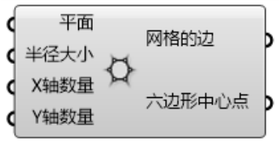
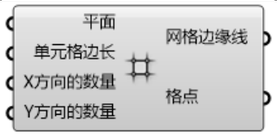
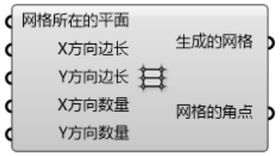
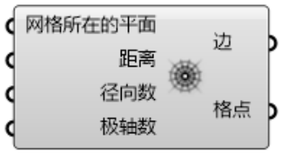
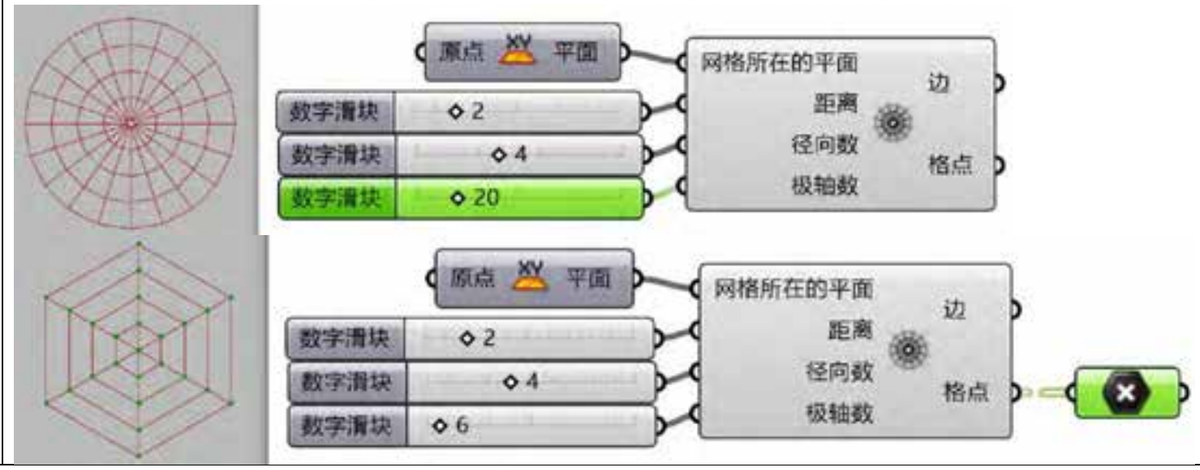
$$w_1\overrightarrow{PP_1}+w_2\overrightarrow{PP_2}+w_3\overrightarrow{PP_3}=0$$

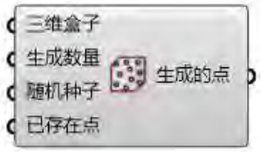
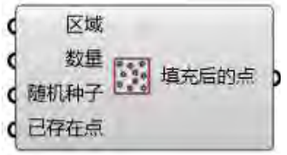
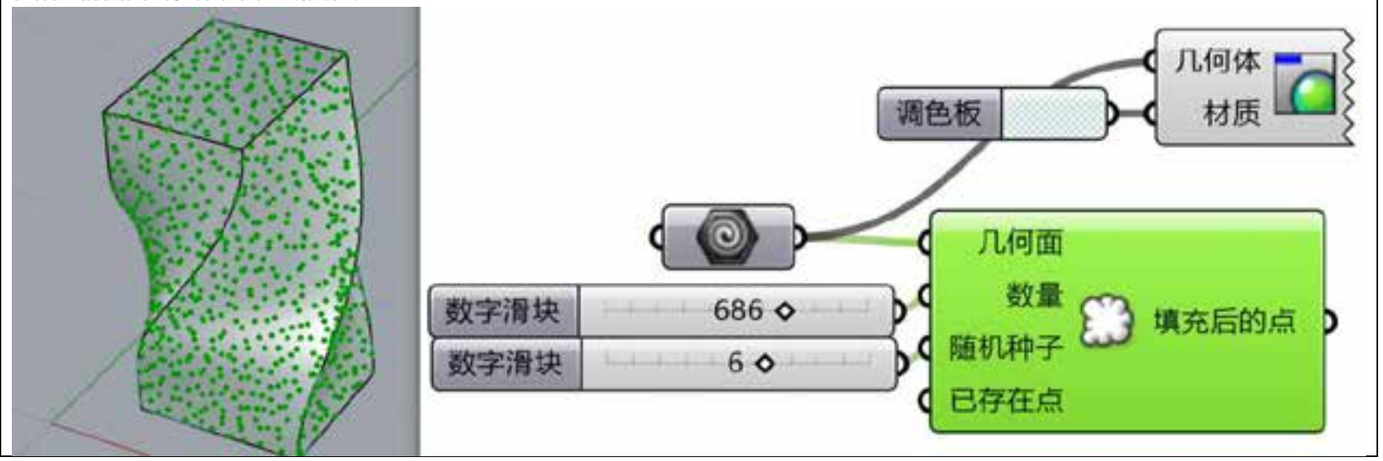
上式说明，如果 P1 点挂重物 w1，P2 点挂重物 w2，P3 点挂重物 w3，则 P 点正好是重心。如果把 P 点放避雷针上，能保持平衡，如图：



这就是“重心坐标”名称的由来。

网格 Grid-（向量）

三角形网格 **Triangular**		**生成等边三角形网格**
六边形网格 **Hexagonal**		**在指定平面上按照设定的单元格半径（中心店到顶点的距离）和 XY 轴方向数量，生成六边形网格**
正方形网格 **Square**		**生成正方形网格**
矩形网格 **Rectangular**		**生成矩形网格**
辐射网格 **Radial**		**在指定平面上，生成蜘蛛网阵列图案，距离为环状多边形间距，径向数为环状多边形的数量，极轴数为中点与环状多边形顶点连线的数量，也可以理解成是环状多边形的边数（数字越高，越趋近圆形），输出格点为环状多边形顶点及中心点。** 

	三维填充 **Populate 3D**		**在指定的矩形盒子里生成一系列点，如果有已存在点，则不会在同样的位置生成的随机点，随机种子可以调节随机点的排布方式。**
	二维填充 **Populate 2D**		**在指定矩形范围内生成随机点**
	填充几何面 **Populate Geometry**		**在指定的曲面或物体表面生成随机点** 

曲线（Curve）

公用 Util

曲线（Curve）

公用 Util -曲线

关于曲线运算器中的的参数：即 **t 值**，是曲线一维坐标系的值，类似曲面的 UV 区间。如果 t 值区间为 0 到 1，则 0 和 1 分别代表曲线的起点和终点；但 t=0.5 并不一定是曲线的中点，除非是直线。

我的理解是：曲线上的点根据曲线控制点，将曲线拉回成直线时，点在直线上的 t 值就是曲线上对应点的 t 值。

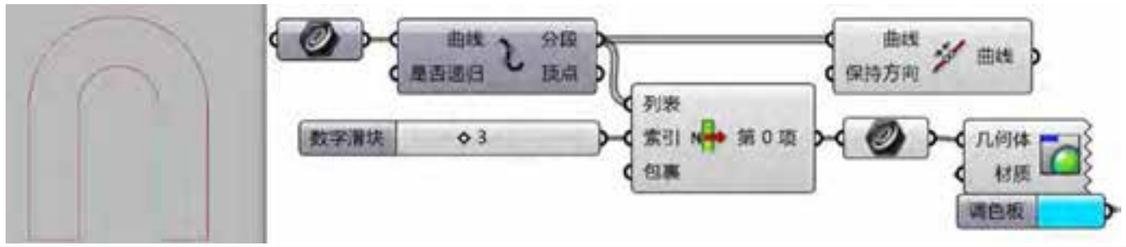
延长曲线 Extend Curve

按照设定长度将曲线向两端延长，**延长类型**有三个 0 则为直线，1 为圆弧，2 为圆滑曲线

炸开曲线 Explode

炸开多重曲线

是否递归为布尔值，默认为 true，控制连续的同类型线段是否炸开。

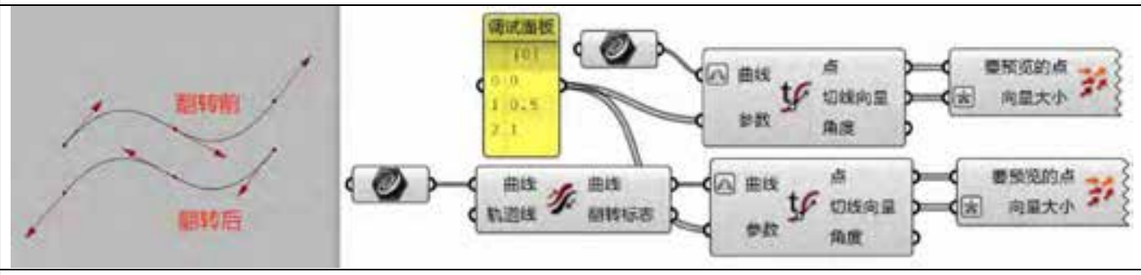


组合曲线 Join

将多个线段组合成多重曲线

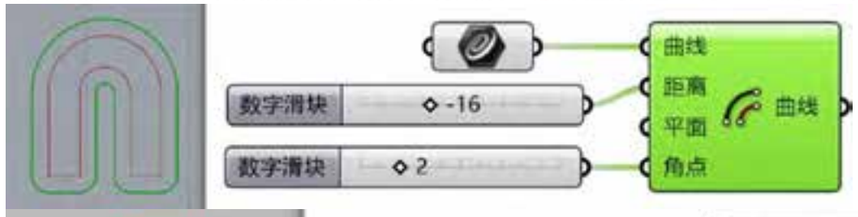
翻转曲线 Flip Curve

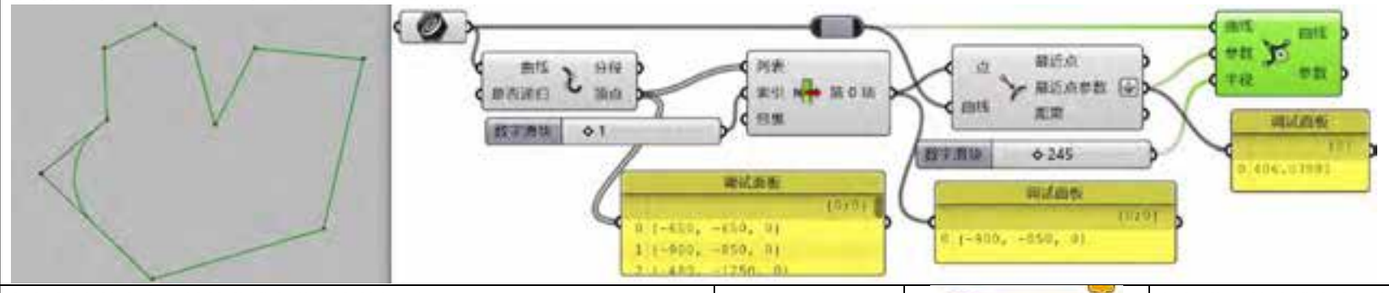
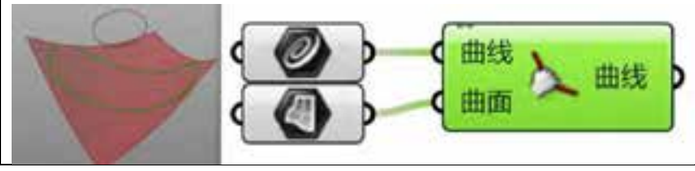
翻转曲线的方向，如果有参考的**轨道线**，则翻转成跟**轨道线**一致的方向，可以用于统一多个曲线的方向。

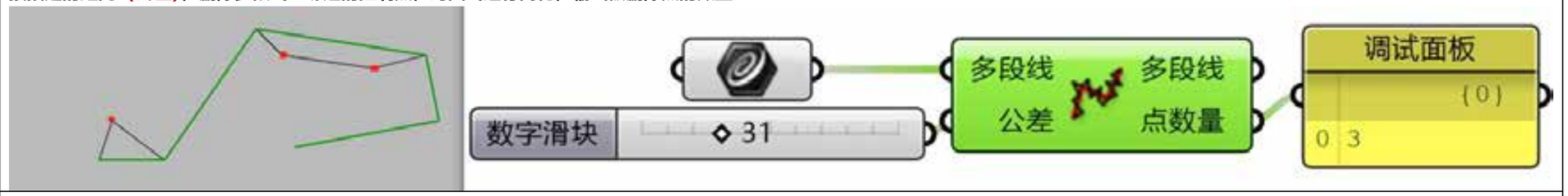
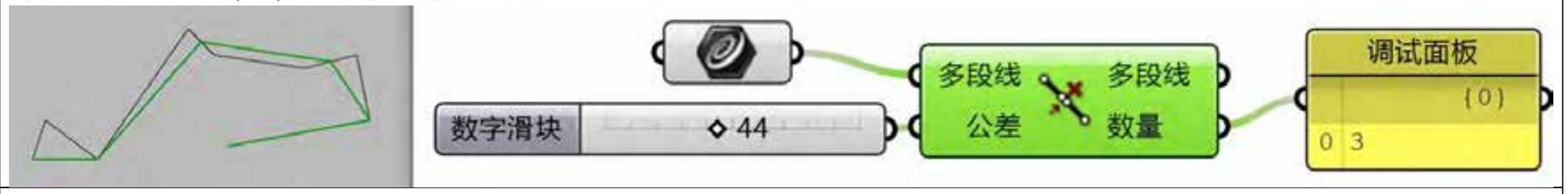
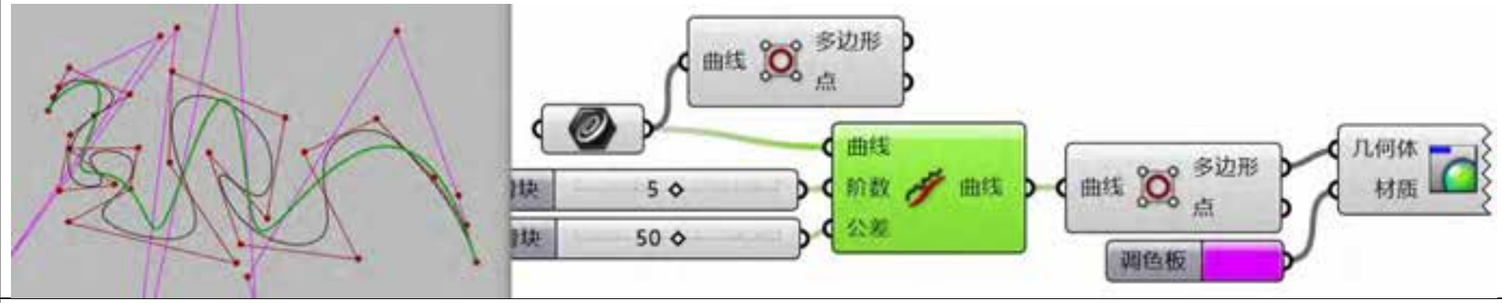
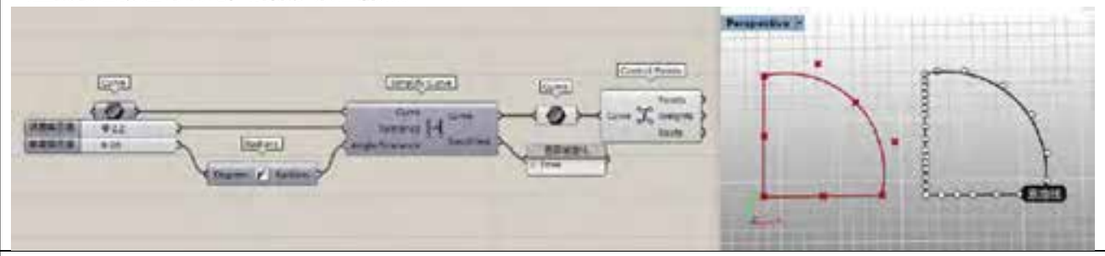
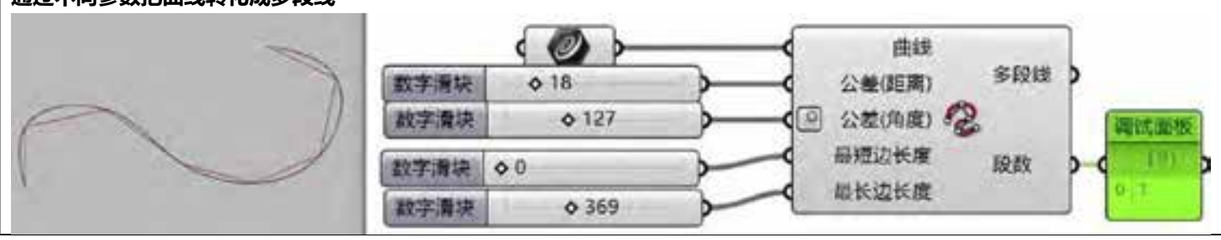


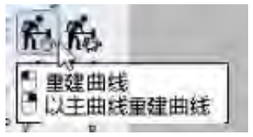
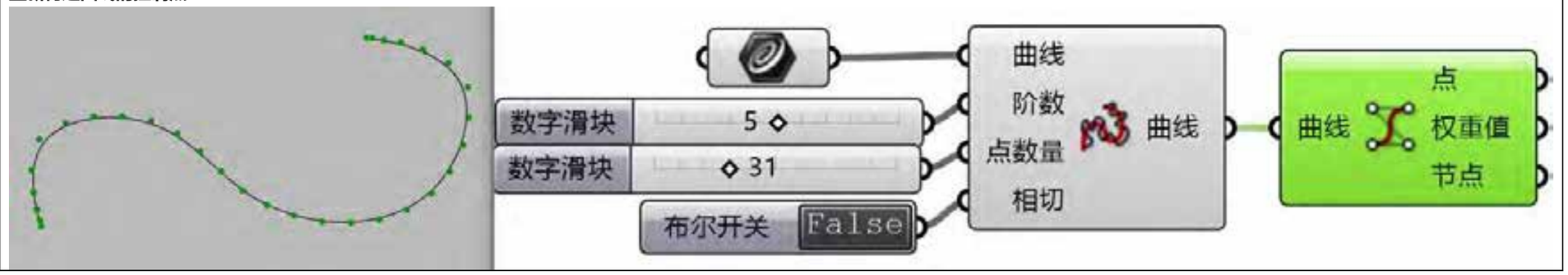
偏移曲线 Offset Curve

根据距离的正负值，将曲线在指定平面上向内或者向外偏移，**角点**类型为数值 0=无，1=尖角，2=圆角，3=光滑，4=斜角。**（最下方有详解）**



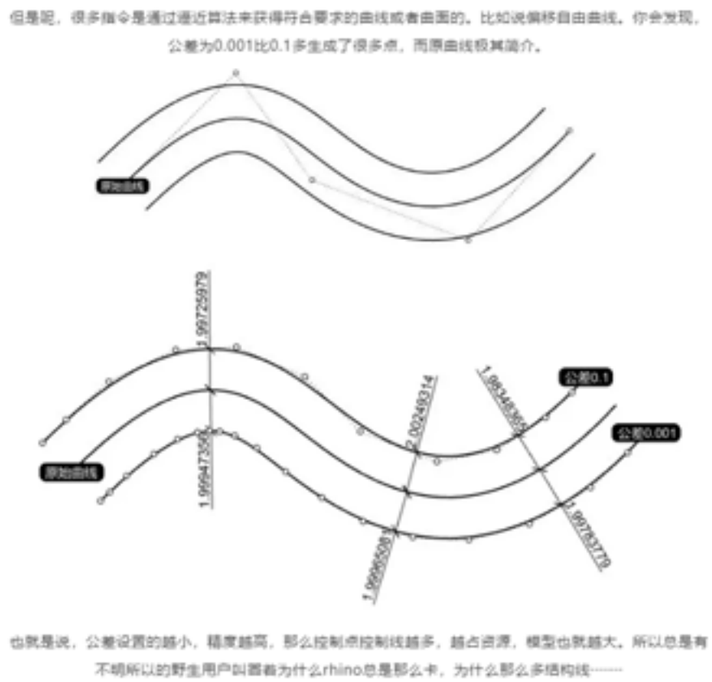
偏移曲线 (控制点) Offset Curve Loose		**偏移曲线，并使偏移后的曲线保持控制点一致（控制点在曲线对应位置的法线重合）** 		
偏移曲线（曲面） Offset on Srf		**偏移曲面上的曲线（偏移的曲线不会自动延伸到曲面边缘，如果要得到延伸到边缘的曲线，可以通过曲线沿着曲面法线方向挤出之后再偏移去相交）** 		
圆角 Fillet		**根据指定半径对曲线转角倒角** 	**圆角（距离） Fillet Distance**	 **根据距离对曲线进行倒角**	
圆角（参数） Fillet(t)		**根据曲线位置相关的参数，对曲线进行倒角** 			
投影曲线 Project		**根据指定的方向，计算曲线在曲面上的投影** 	**拉回曲线 Pull Curve**		**根据曲面的法线方向，把曲线拉回曲面。曲线上的点与被拉回后的点都在同一曲面法线上** 
调整接缝 Seam		**调整曲线的接缝位置。（下图曲线挤出成面后，蓝色箭头指的线就是接缝位置）** 			

多段线删点 **Reduce**		**按指定的距离（公差），删除多段线上邻近的控制点，对曲线进行简化，输出被删除点的数量** 
多段线简化 **Polyline** **Collapse**		**删除多段线上小于指定距离（公差）的线段，输出被删除线段的数量** 
平滑多段线 **Smooth Polyline**		**平滑多段线趋势，比较好理解**
拟合曲线 **Fit Curve**		**根据指定阶数和公差，来简化曲线控制点。** 
简化曲线 **Simplify Curve**		**对直线或圆弧进行简化，消除多余控制点。** 
简化曲线 **（公差）** **Curve To** **Polyline**		**通过不同参数把曲线转化成多段线** 

重建曲线 **Rebuild Curve**	 **与 Rhino 中这功能一致** 	**重新构建曲线的控制点** 

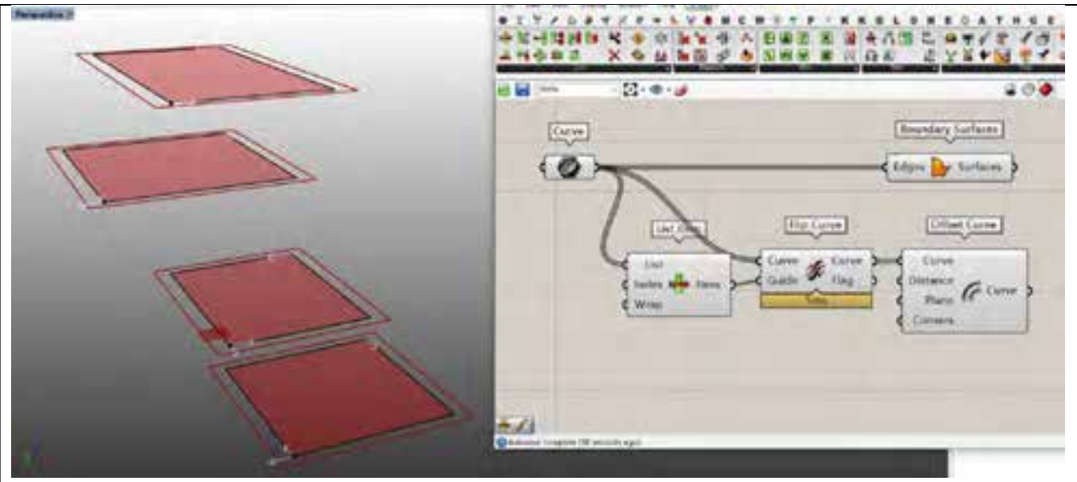
偏移曲线：向内还是向外偏移根据 D 的正负决定，而 C 拐角类型输入 0=无，1=圆角，2=圆角，3=柔滑曲线，4=斜角。再叨叨一遍，偏移曲线是逼近算法，控制点必定增多，所以不要去想什么偏移曲线之后还保持控制点一致了，保持控制点一致那距离必定不准。

不理解为啥的可以看一下我这篇文章里的公差部分：

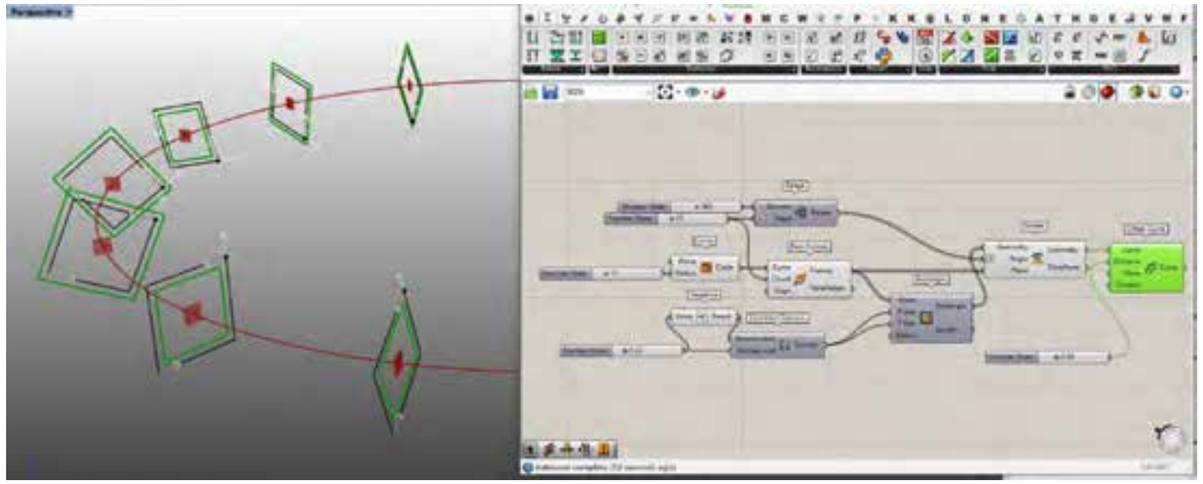


不过有的同学会遇到的一个问题就是，同样的（看起来）一堆线，去偏移，有的向内，有的向外，偏移方向不一致，那这可怎么办？比如明明都是平面的线，怎么还偏移方向不一致呢？因为曲线是有方向的，顺时针为负，逆时针为正，这时候你结合曲线方向再看你就知道了：

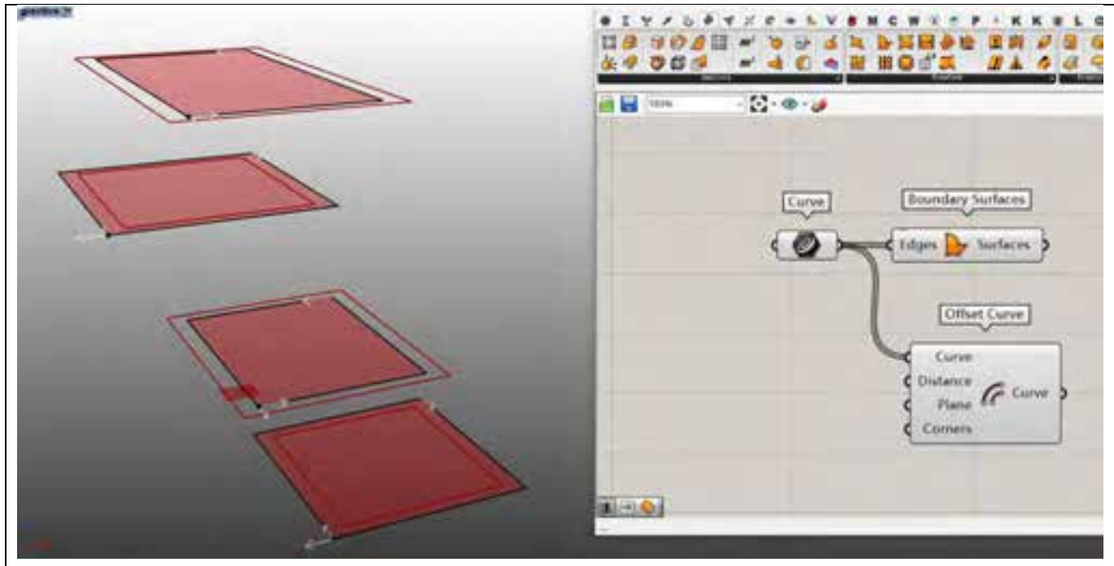
：



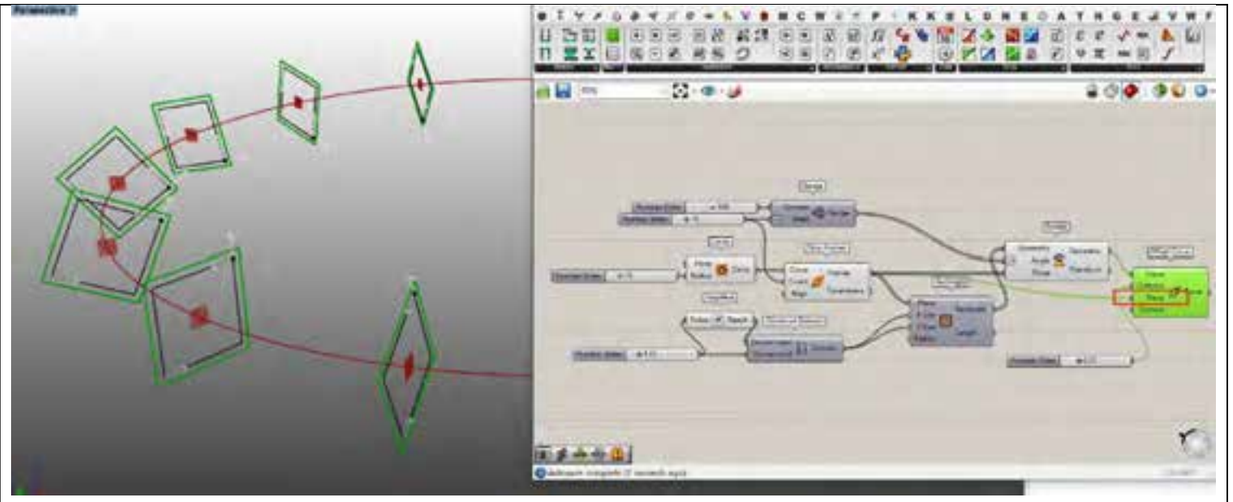
但是呢，又会有同学遇到另一种情况，那就是明明曲线都是一个方向的，咋偏移的还乱七八糟的：



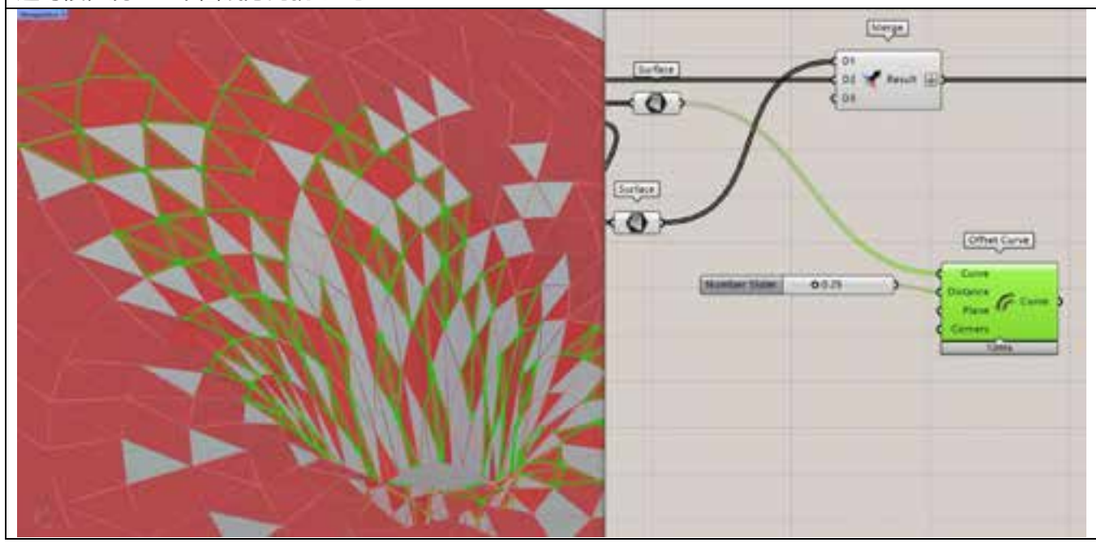
这是因为，曲线的逆时针还是顺时针，是根据工作平面来定的，工作平面不同，曲线可以被认定为不同的方向。所以这时候，最好上面的线，都是基于 xy 工作平面去判断曲线的顺还是逆时针，那由于曲线是空间曲线，不断旋转变化，结果就会不一样，就会出现这样乱七八糟的情况。解决办法也很简单，就是指定一下判断曲线方向用的工作平面：

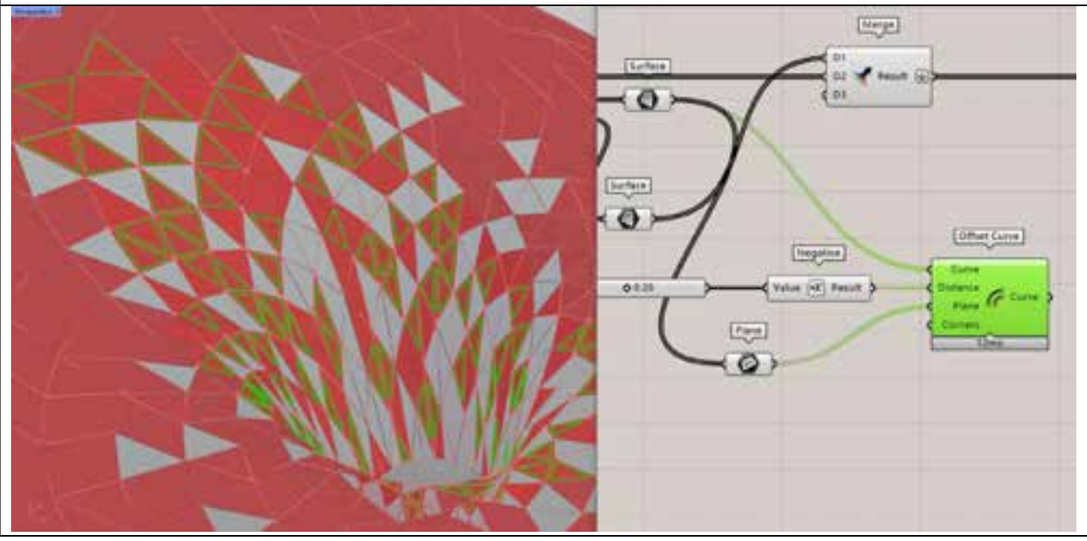


这时候，统一一下曲线方向就 ok 了



搞定收工，另外，如果你出现平面曲线偏移却出现错误的情况，那你需要给 offsetcrv 指定一下偏移平面才行：





分析 Analysis - 曲线

分解圆弧 Deconstruct Arc

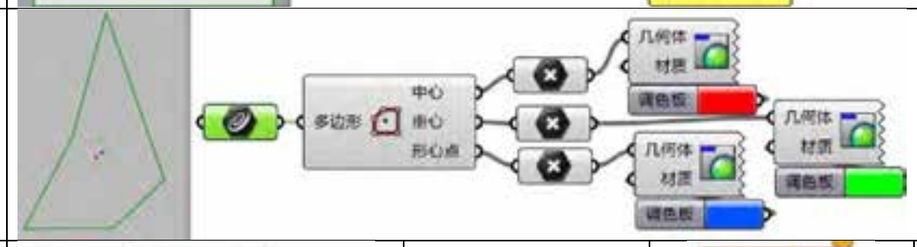
计算圆弧所在的平面，半径和角度

分解矩形 Deconstuct Rectangle

计算矩形所在的平面（以矩形中心为原点），及长宽所在的区间

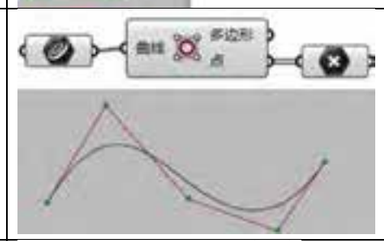
多边形中心 Polygon Center

计算多边形的中心、中心和形心点



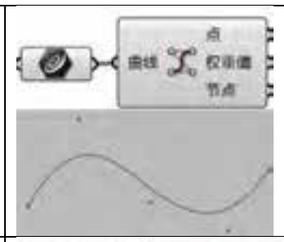
控制多边形 Control Polygon

获取曲线的控制点，并连接控制点以多段线形式显示



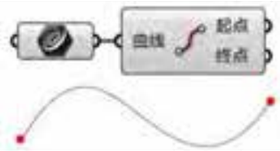
控制点 Control Points

获取曲线的控制点，以及曲线控制点权重（Weights），节点（Konts）为向量大小



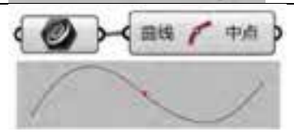
曲线之端点 End Points

获取曲线的端点



曲线之中点 Curve Middle

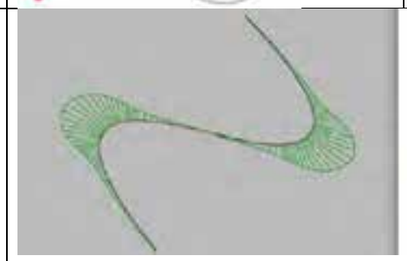
获取曲线的中点

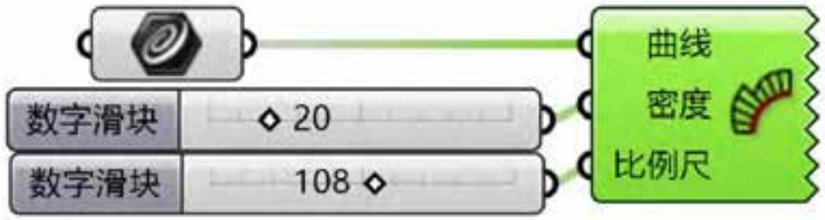


曲率图 Curvature Graph

查看曲线的曲率图形。

（曲线上的任何一点都有一条直线与该点正切的直线，我们也可以找出与该点正切的圆，这个圆的半径倒数就是曲线在该点的曲率）

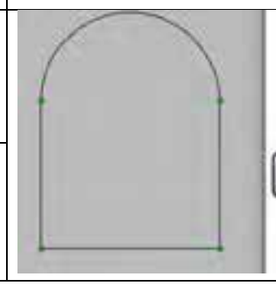




曲线之不连续性 Discontinuity

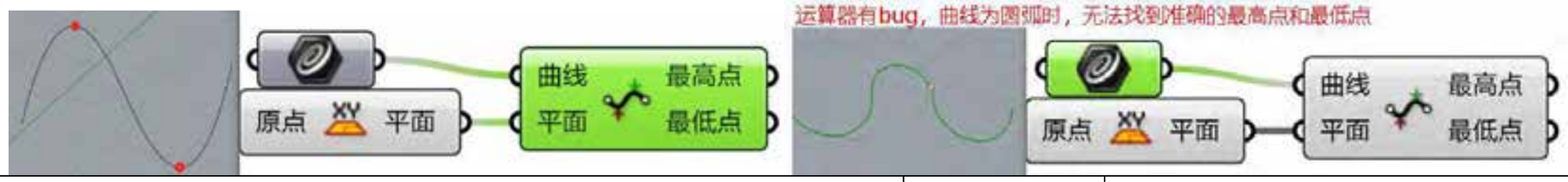
判断曲线不连续点，等级参数为 1 到 3，1 为不满足正且连续 G2，2 为不满足曲率连续 G3，3 =Cinfinite，即 G3 以上所有的连续性。

（连续性概述见下方）



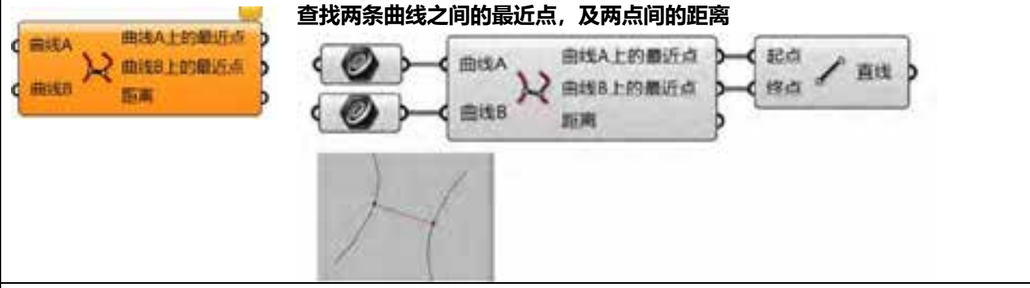
曲线极值 Extremes

判断曲线相对于指定平面的最高最低点



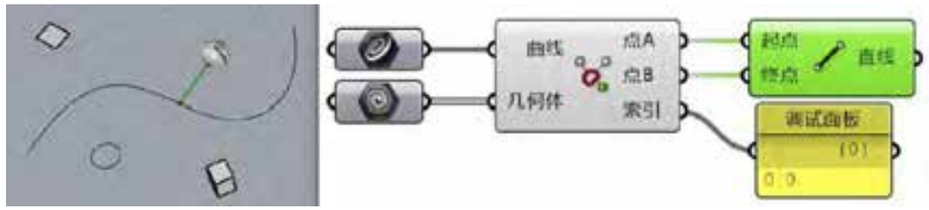
曲线之接近点 Curve Proximity

查找两条曲线之间的最近点，及两点间的距离



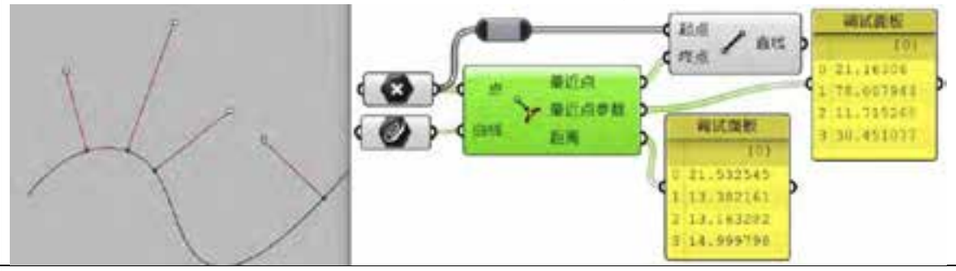
曲线之最近物体 Curve Nearest Object

查找离曲线最近的物体，点 A 为曲线上的点，点 B 为最近物体上的点，索引为最近物体的序号



曲线之最近点 Curve Closest Point

查找曲线上离指定点的最近点，及相关参数。（**最近点参数**为 t 值，区间在 0-1，t=0 则对应曲线起点，t=1 则对应曲线终点,但是 t=0.5 并不是曲线的中点，除非是直线）



检测曲线两侧的点 Curve Side

判断点在曲线的哪一侧

检测曲线的平面性 Planar

判断曲线是否为平面曲线，输出平面度为布尔值，曲线所在的平面和偏离值

检测闭合 Closed

判断曲线是否为闭合曲线或周期曲线（周期曲线详解见下方）

扭矩 Torsion

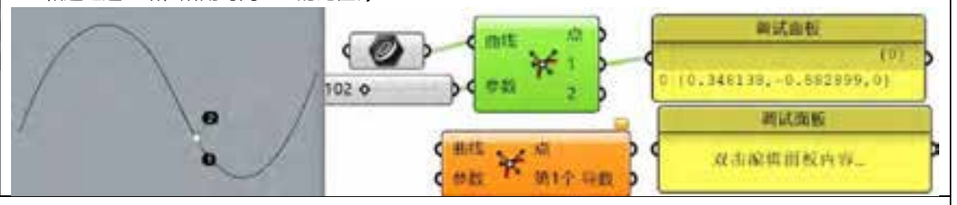
曲线在指定参数（曲线 t 值，不是曲线长度）处的点和扭力大小。

输入的**参数**为曲线上的距离位置，超过曲线长度时，会自动延伸出去

曲线之导数 Derivatives

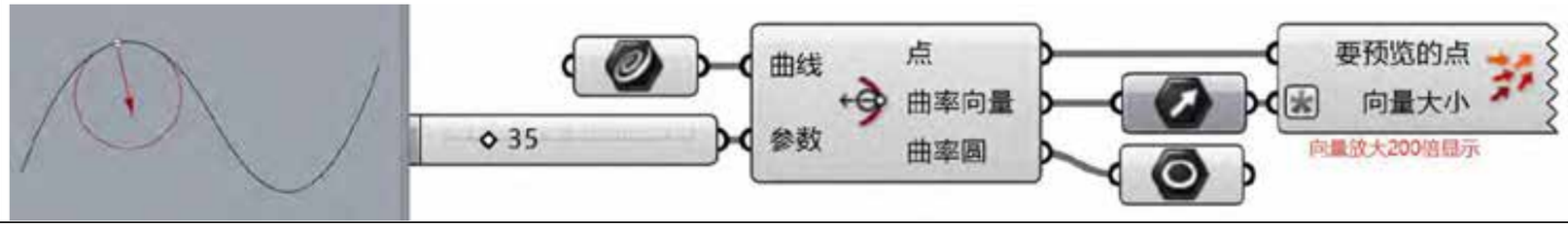
查找曲线上指定参数处的点，输出的导数为速率，输出端可以增加到六个。

（速率是标量，只有大小没有方向。速率是瞬时速度的大小或者等价于路程的变化率。速率是物体经过的路程 ▵S 和通过这一路程所用时间 ▵t 的比值。）



曲线之曲率 Curvature

查找曲线指定参数处的点，及该点的曲率，曲率圆为以曲率倒数为半径的圆

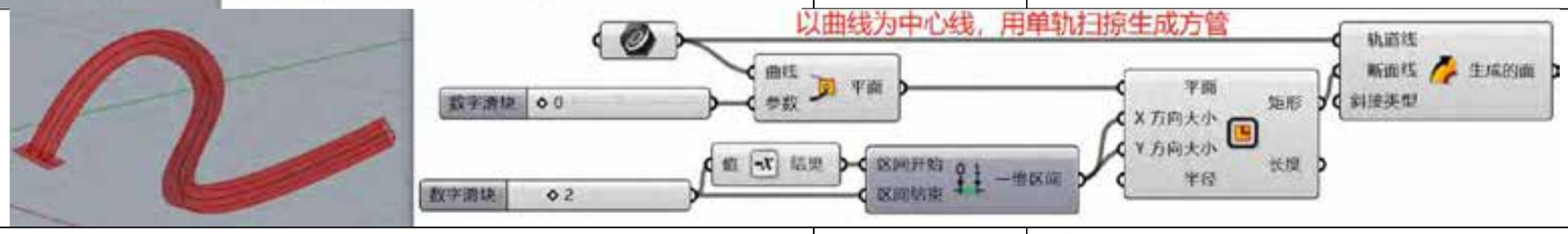


曲线之负法线平面
Curve Frame

查找曲线指定参数处的点所在的工作平面，平面的 X 轴为点在曲线上的切线方向，Y 轴为曲率向量的方向（参照上面的曲线之曲率运算器）。

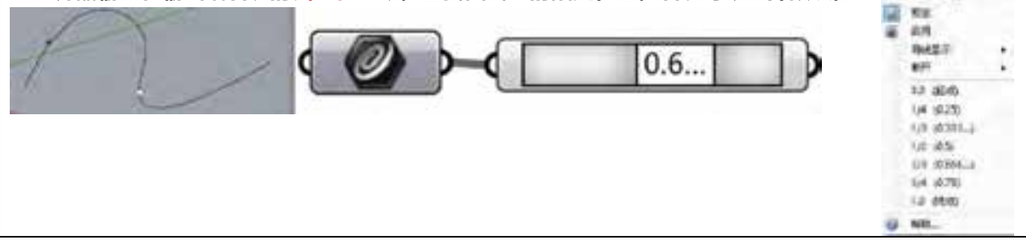
曲线垂直平面
Perp Frame

在曲线指定参数处的点，生成一个垂直与该点切线的平面



曲线归一化
Point On Curve

与运算器输入和输出端右键的归一化一致，查找曲线上的指定位置，右键可以选择预设值

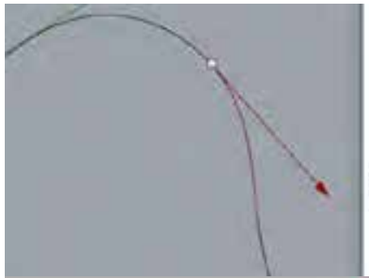


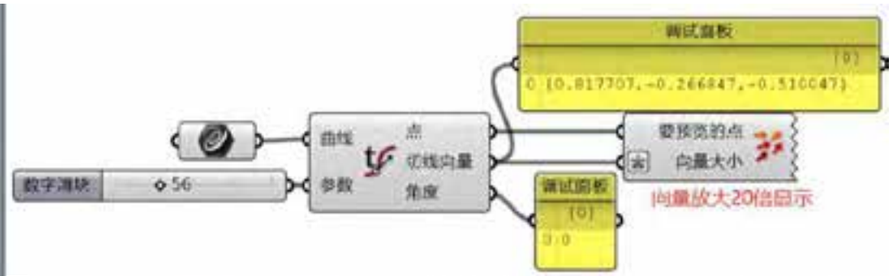
曲线水平平面
Horizontal Frame

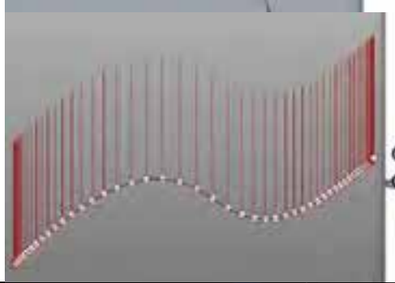
在曲线指定位置处，生成一个平行于 XY 平面的平面

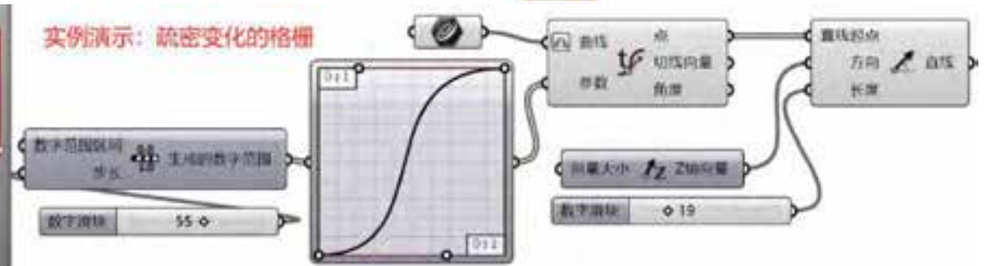
测量曲线
Evaluate Curve

查找曲线上指定位置处的点，并输出曲线上该点的切线向量和角度



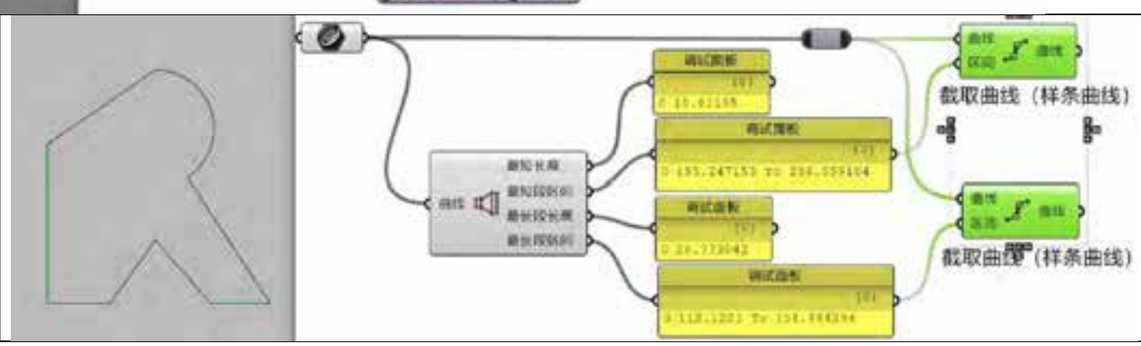






分段长度
Segment Lengths

提取多重曲线中最短和最长的一段线，及其在曲线上的 t 值区间



曲线区间
Curve Domain

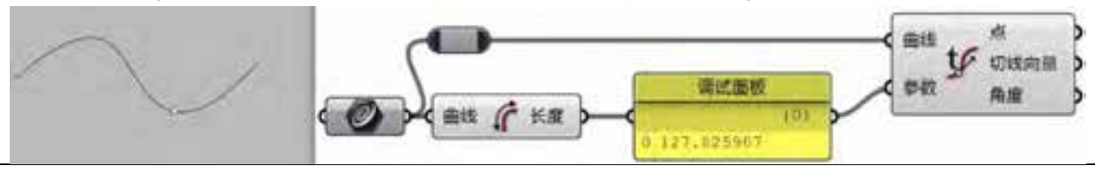
类似区间映射运算器，把曲线所在的区间，映射到自定义区间

求长度值
Evaluate Length

将曲线归一化后，通过指定长度系数（0-1 之间）定位曲线上的点，并输出该点的切线向量及参数（该点在没有归一化的曲线上对应的 t 值）

测量长度
Length

计算曲线长度（由此可以看出前面曲线有关运算器的参数，并不是曲线的长度）

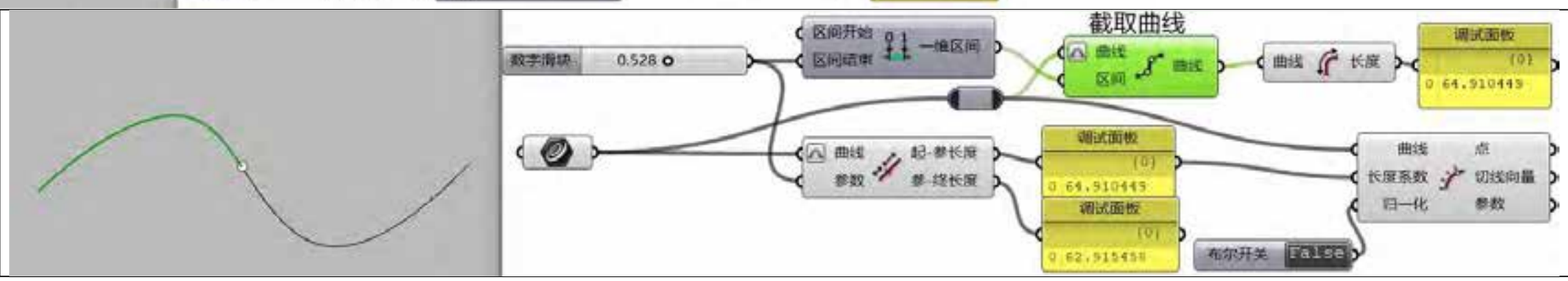


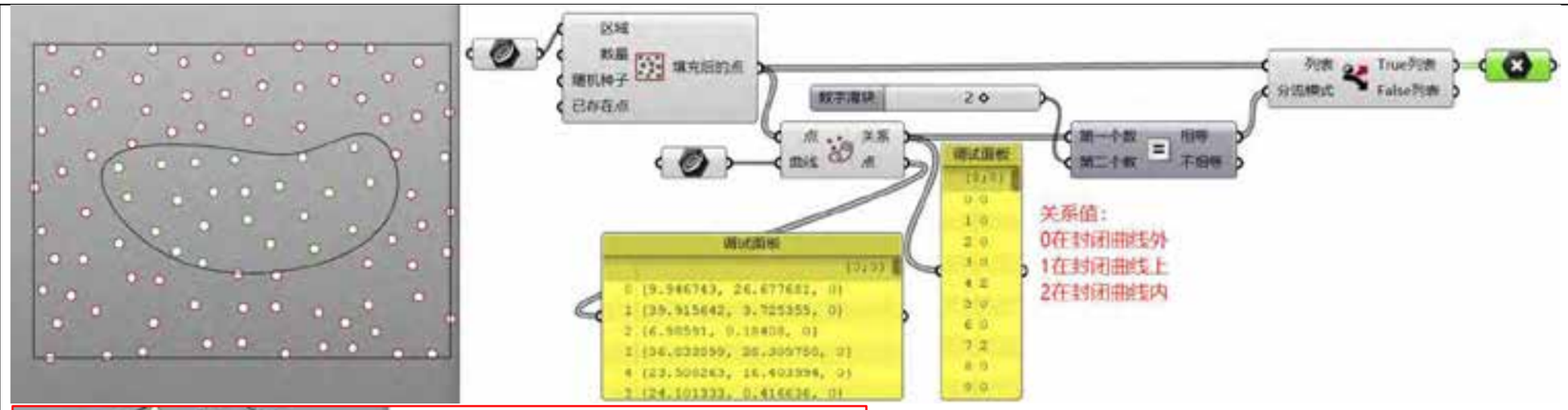
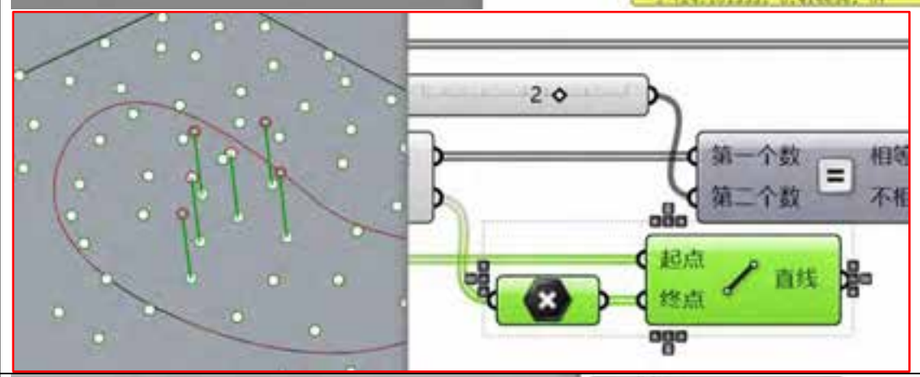
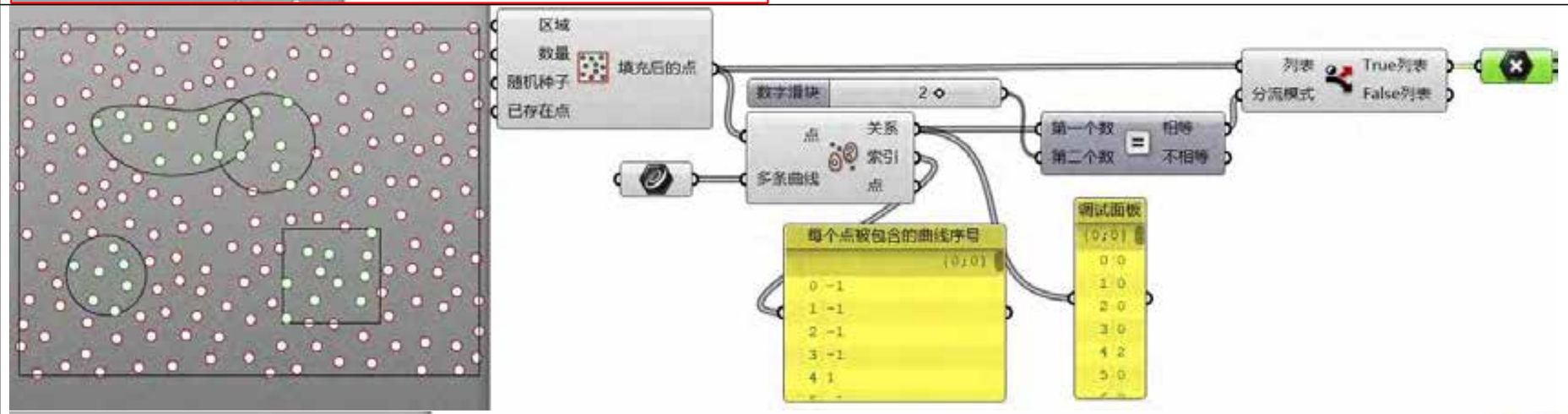
长度区间
Length Domain

查找曲线指定 t 值区间范围的曲线长度

长度参数
Length Parameter

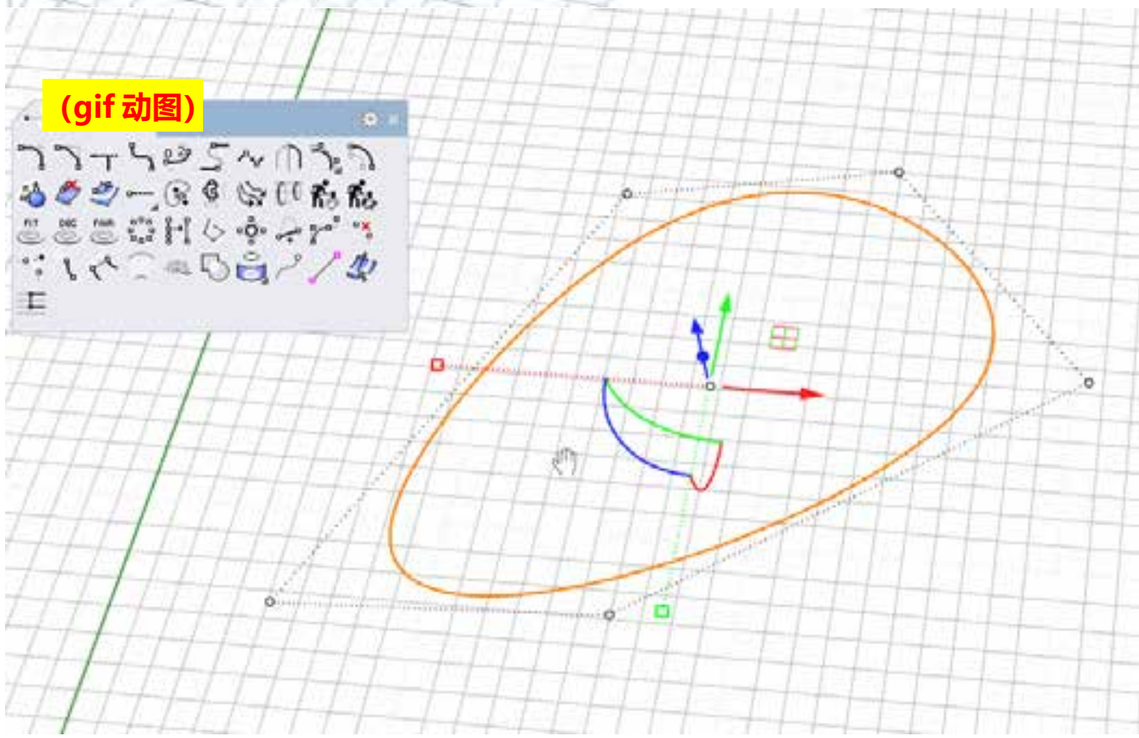
根据参数 (曲线 t 值) 定位点，把曲线分成前后两段长度



点在曲线内部 **Point In Curve**	 **判断点是否在指定的封闭曲线内，输出端关系列表中 0 为点在曲线外，1 为点在曲线上，2 为点在曲线内。** **如果有不在同一平面的点，则输出端的点会输出这些点的投影点。**	 
点在多条曲线内部 **Point in Curves**	 **判断点与多个曲线的包含关系，索引为点被包含曲线的序号**	

周期曲线：

Rhino 里有个命令，就是用来修改曲线是否周期化的。所以，即便是封闭曲线，也是分非周期和周期两种的。



连续性概述：

连续性概述

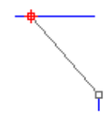
一个曲线或曲面可以被描述为具有 *Gn* 连续性，*n* 是表示光滑度的增量，即在曲线上取一点，然后分析该点与其两侧线段的关系。

- G0: 两侧曲线在这一点相遇(位置连续)。
- G1: 两侧曲线这一点的切线方向相同(相切连续)。
- G2: 两侧曲线在这一点的曲率相同(曲率连续)。

请参考: 维基百科: Geometric continuity

位置连续 (G0)

只测量两条曲线端点的位置是否相同，两条曲线的端点位于同一个位置时称为位置连续 (G0)，换句话说就是两条曲线的端点相接。

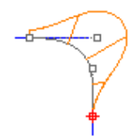


正切连续 (G1)

测量两条曲线端点的位置及方向是否相同，换句话说就是两条曲线的端点相接且方向一致。

曲线端点的方向是由第一、二个控制点决定，两条曲线相接点的前两个控制点 (共四个控制点) 位于同一直线上时称为正切。

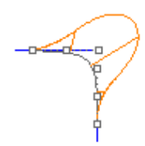
两条曲线在相接点的一阶导数相同。



曲率连续 (G2)

测量两条曲线端点的位置、方向及曲率是否相同，三者都相同时称为曲率连续 (G2)，换句话说就是两条曲线的端点不只相接，连方向与半径都一样。曲率连续无法以控制点的位置来判断。

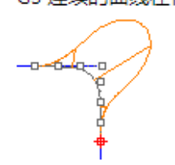
两条曲线在相接点的一阶、二阶导数相同。



G3

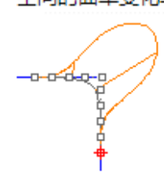
G3 连续比 G2 增加一个条件，两条曲线的端点除了位置、方向及半径一致以外，半径的变化率也必需相同。

G3 连续的曲线在相接点的一阶、二阶、三阶导数皆相同。



G4

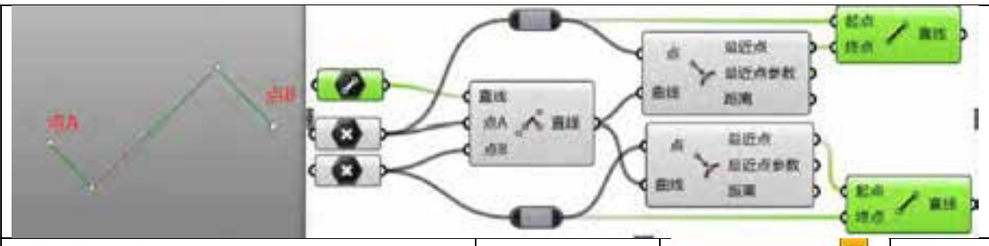
G4 连续极少用到，但可能在某些特殊案例很重要。G4 连续需要 G3 连续的所有条件以外，在 3D 空间的曲率变化率也必需相同。



初始 Primitive - 曲线

4 点生线
Line 4Pt

以直线为方向，两点在直线上的投影点间距为长度，生成直线

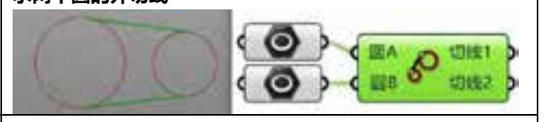


圆切线（内切）
Tangent Lines (In)

求两个圆的内切线

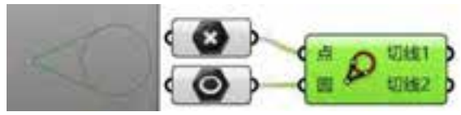
圆切线（外切）
Tangent Lines (Ex)

求两个圆的外切线



圆切线（点）
Tangent Lines

求点到圆的切线

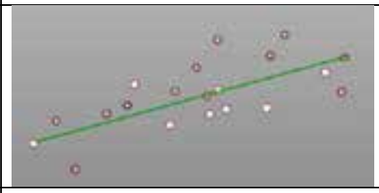


截取直线
Line 2Plane

以直线为方向，生成一段与两个平面相交的直线

拟合直线
Fit Line

通过多个点拟合生成一条直线



点生直线
Line SDL

以指定的起点、方向（可以是直线）、长度生成一段直线

直线
Line

通过两个点生成直线

三点生圆
Circle 3Pt

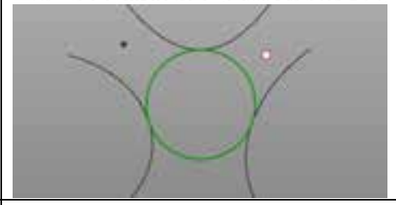
通过三个点生成圆，并输出圆半径和所在平面

双切生圆
Circle TanTan

根据一个指定的点（实际上不是圆心，某些特地情况下是在切点与圆心连线上），生成两条曲线的外切圆

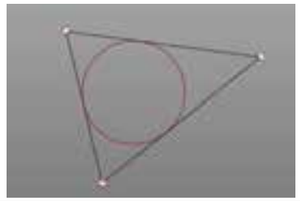
三切生圆
Circle TanTanTan

生成三条曲线的相切圆（圆点好像没有意义，不影响结果）



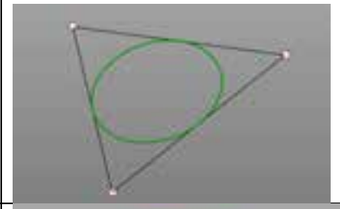
三角形内接圆
InCircle

通过三个点生成三角形的内街圆，及相应参数



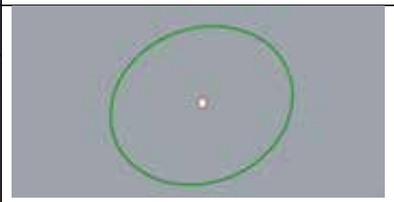
三角形椭圆
InEllipse

通过三个点生成三角形的内街椭圆，及其所在平面



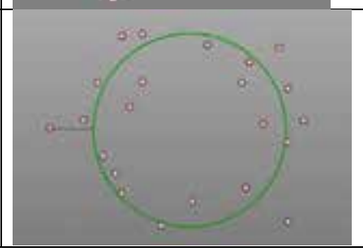
半径生圆
Circle

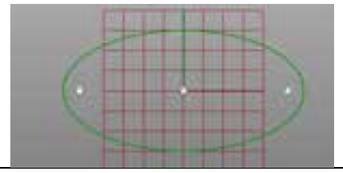
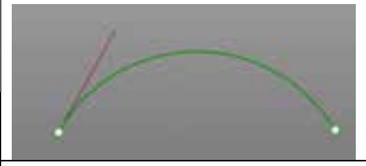
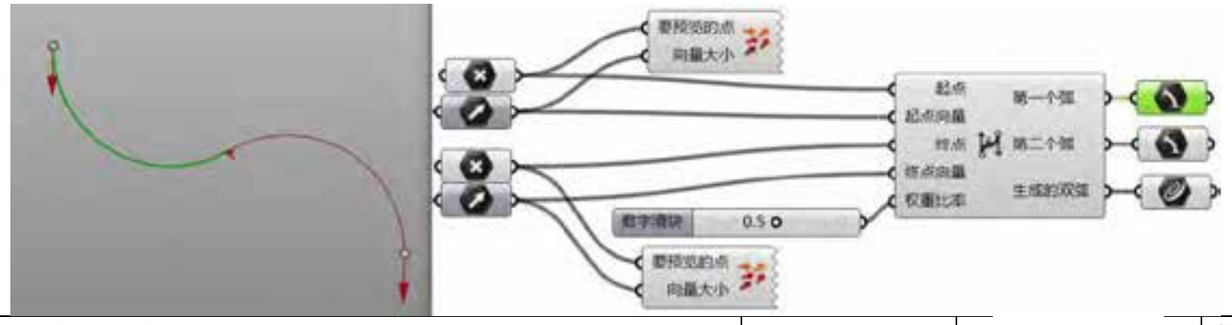
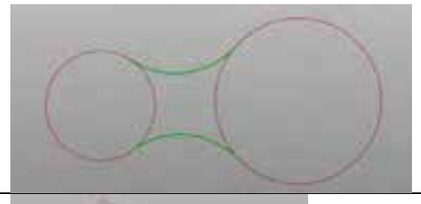
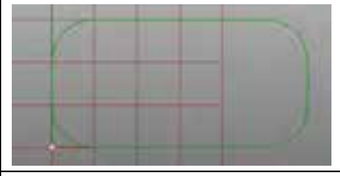
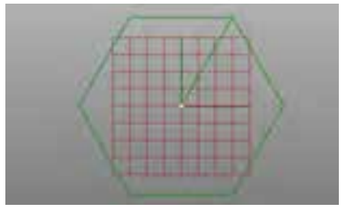
根据指定半径在指定平面（也可以是点，此时默认 xy 平面）上生成圆



拟合生圆
Circle Fit

通过点群生成一个拟合圆，输出圆半径和离圆最远的点到圆的距离



椭圆 **Ellipse**		通过指定椭圆的两个半径生成椭圆，并输出两个焦点 	**法线生圆** **Circle CNR**		根据圆心和圆的法线及半径，生成圆
三点生弧 **Arc 3Pt**		通过三点生成圆弧，输出圆弧半径及其所在平面 	**两点向量生弧** **Arc SED**		通过两个点及设定方向（其中一个点的切线方向）生成圆弧 
修正弧 **Modified Arc**		基于指定圆弧的圆心和平面，以设定的圆弧角和半径重新生成圆弧 	**半径角度生弧** **Arc**		在指定平面上，以设定半径和角度生成圆弧
生成双弧 **BiArc**		根据两个点及两点的向量方向，绘制两端圆弧组成的曲线 			
相切圆弧 **Tangent Arcs**		根据指定圆弧半径，生成与两个圆相切的圆弧 	**创建矩形（2 点）** **Rectangle 2Pt**		在指定平面上根据两点生成矩形，半径为倒角半径
创建矩形（3 点） **Rectangle 3Pt**	 根据三点创建矩形		**建矩形** **（一般方式）** **Rectangle**		根据指定的长和宽，及倒角半径，生成矩形。比较常用的生成矩形运算器 
多边形 **Polygon**		创建等边多边形，半径为中点到顶点的距离 	**多边形（单边）** **Polygon Edge**		以指定的边起点和边终点的距离为边长，在平面点所在的平面上，生成指定边数的等边多边形，输出相关参数

样条曲线 Spline - 曲线

内插点曲线
(向量)
Interpolate (t)

根据内插点， 和终点向量。及指定的节点类型（0 均匀，1 弦长，2 弦长平方根），生成一段曲线

内插点曲线
(角度)
Kinky Curve

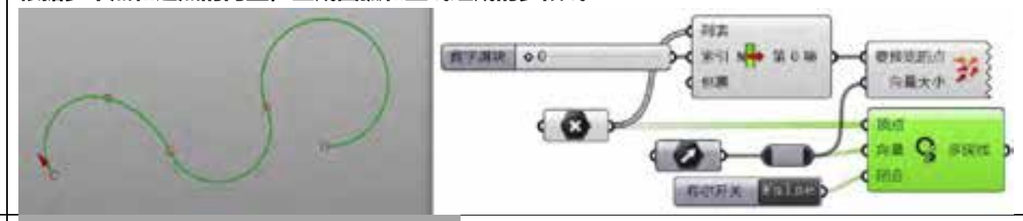
以阶数和角度作为参数生成的内插点曲线，当点之间的连线折角大于角度的两倍时才连接成曲线，否则为直线连接

切线方向生线
Tangent Curve

根据多个点，及每个点的向量绘制曲线

多段线
PolyArc

根据多个点和起点的向量，生成圆弧和直线组成的多段线



多点生多段线
PolyLine

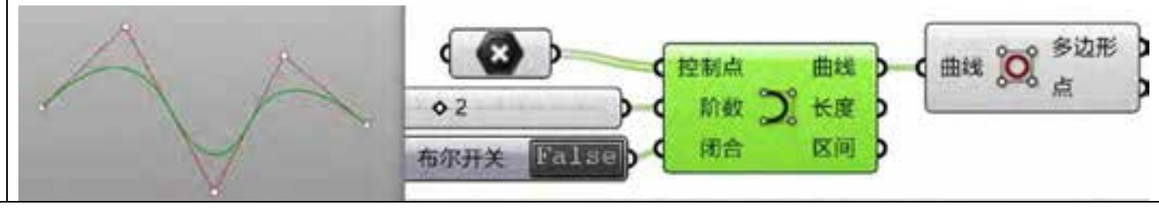
根据多点绘制多段线

插值曲线
Interpolate

多点绘制插值曲线（同 rhino 中的内插点曲线），节点类型（0 均匀，1 弦长，2 弦长平方根）

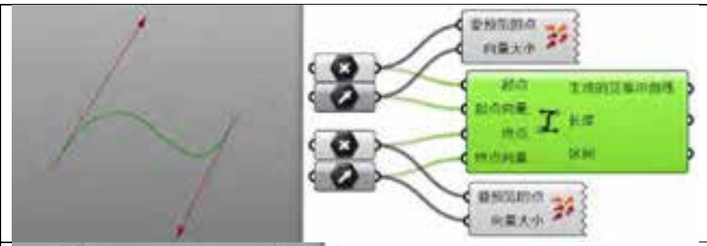
构造 Nurbs 曲线
Nurbs Curve

根据多点生成控制点曲线



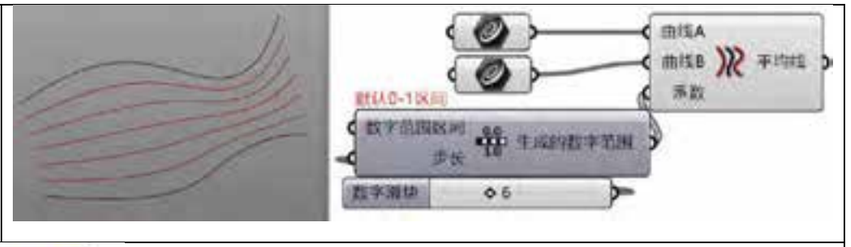
贝塞尔曲线
Bezier Span

根据起点和终点，及对应的向量生成贝塞尔曲线



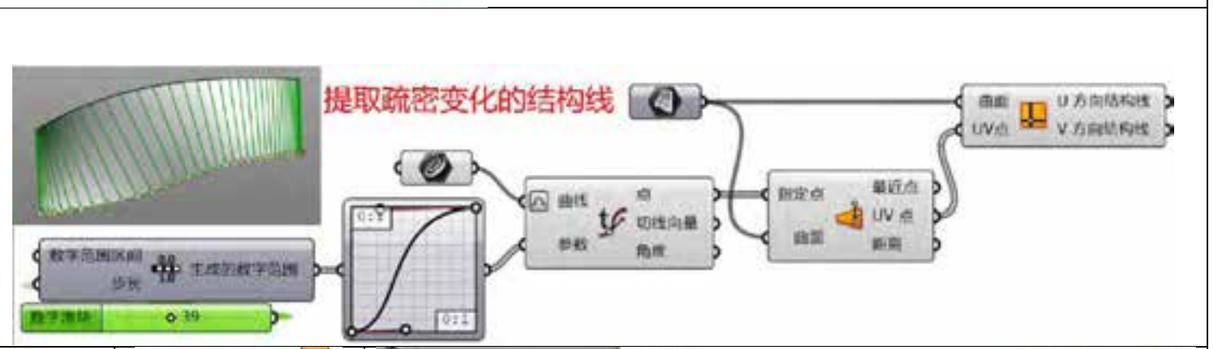
平均线
Tween Curve

在两条曲线之间建立均分曲线



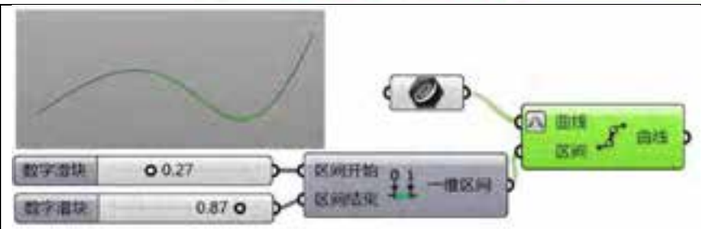
UV 结构线
Iso Curve

根据曲面上的点（UV 坐标）抽离曲面结构线



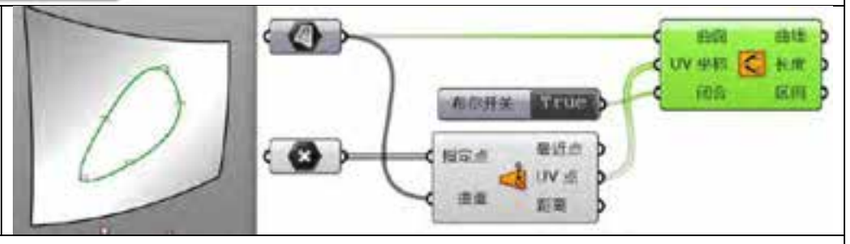
截取曲线
Sub Curve

根据指定区间截取曲线的一部分



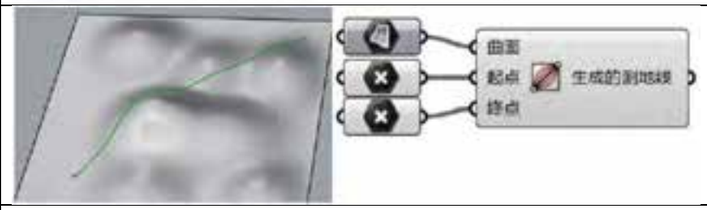
曲面生线
(UV 点)
Curve On Surface

根据曲面上的点(UV 坐标)绘制样条曲线



测地线
Geodesic

寻找曲面上两点之间最短路径

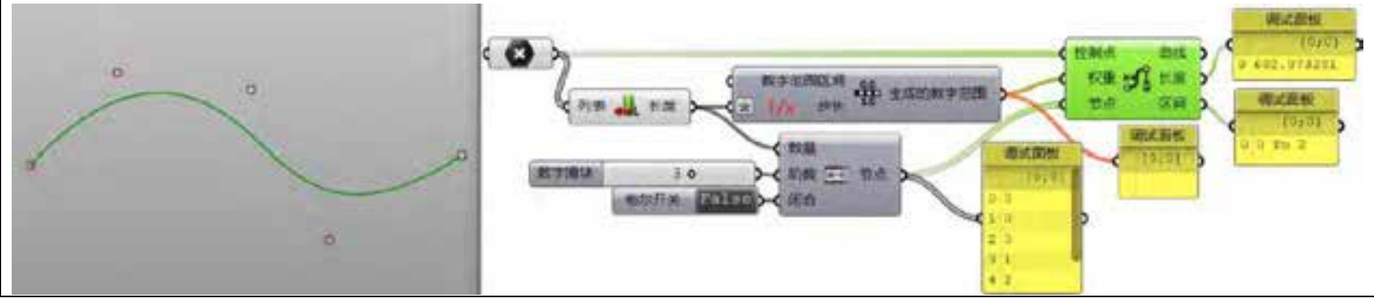


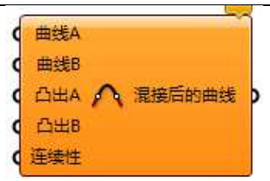
控制点生曲线
Nurbs Curve
PWK

节点向量
Knot Vector

控制点生曲线：根据控制点【Points】，点的权重【Weights】，以及节点矢量值【Knots】来生成一个自由曲线，点的权重即每个点对曲线的影响力，0 最小，1 最大。

节点向量：根据点的数量【Count】，曲线阶数【Degree】，以及曲线是否周期性【Periodic】来生成节点矢量值。

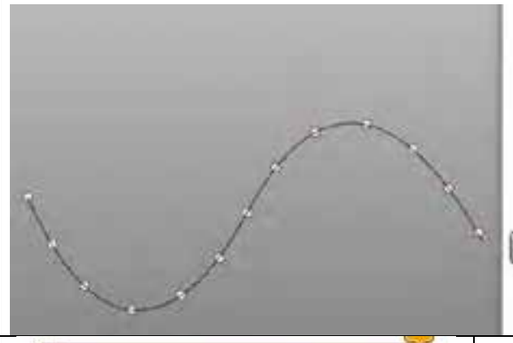


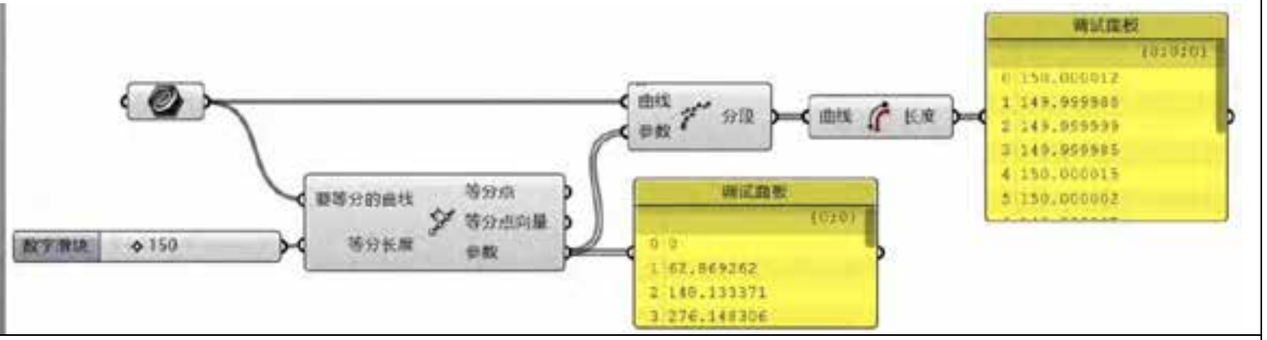
悬链线 **Catenary**		**模拟自由下垂的绳子，长度值要大于两点的间距。**
混接生线 **(混接曲线)** **Blend Curve**		**同 rhino 中的弧形混接，连续性有三种（0 位置，1 相切，2 曲率）**
混接生线（点） **Blend Curve** **Pt**		**同上方运算器类似，只不过是用指定点来控制衔接效果（点不一定是在曲线上）**
连接曲线 **Connect** **Curves**		**对多条曲线进行混接**

等分 Division -曲线

分割曲线 Shatter

在指定的 t 值位置处分割曲线，经常配合其他等分运算器对曲线分段。



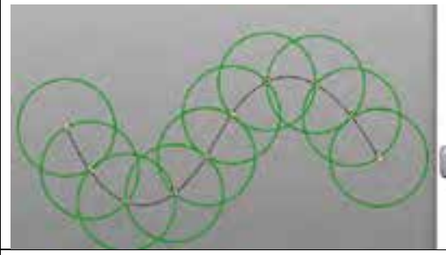


曲线之段数等分 Divide Curve

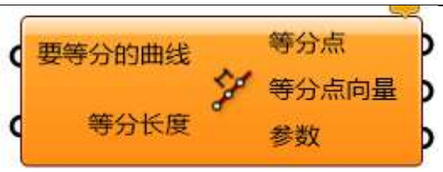
按段数等分曲线，输出等分点、等分点向量、参数。

曲线之距离等分 Divide Distance

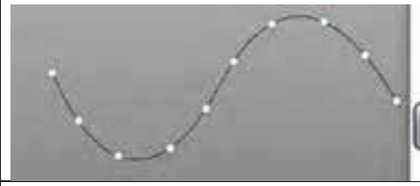
按等分点之间的距离等分曲线



曲线之长度等分 Divide Length

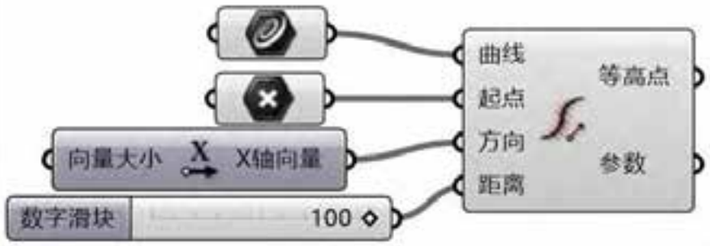


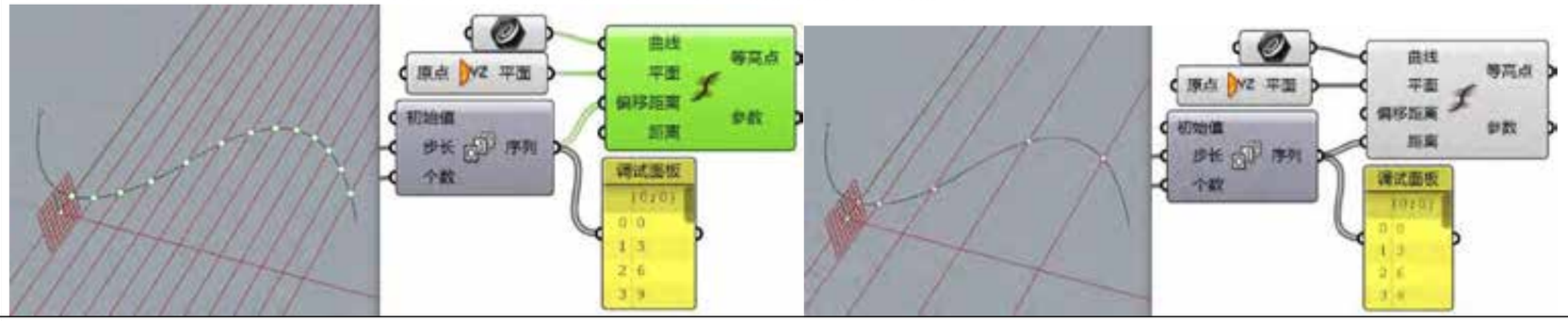
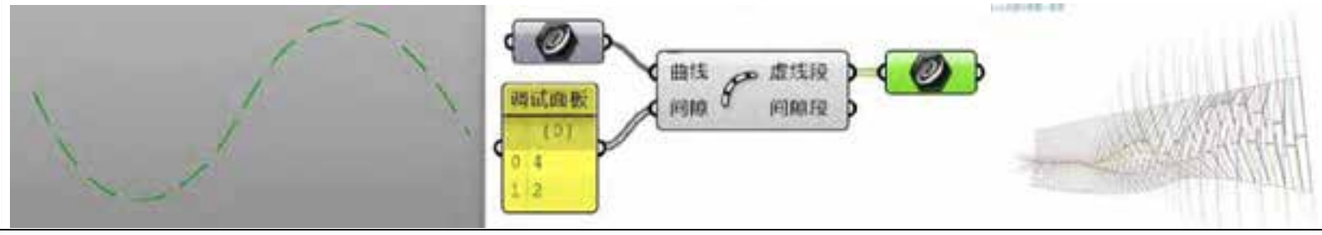
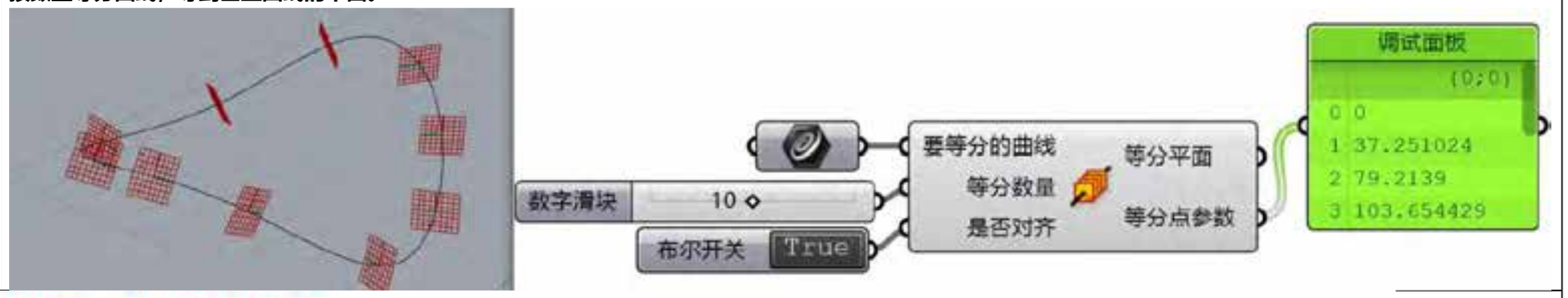
按长度等分曲线



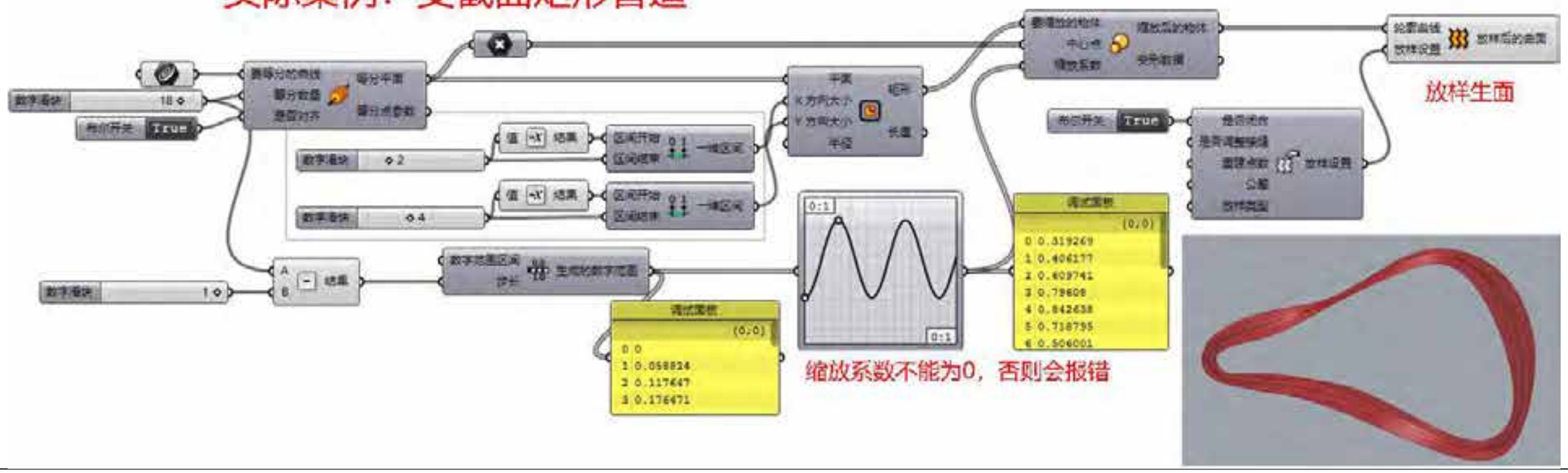
等高线 Contour

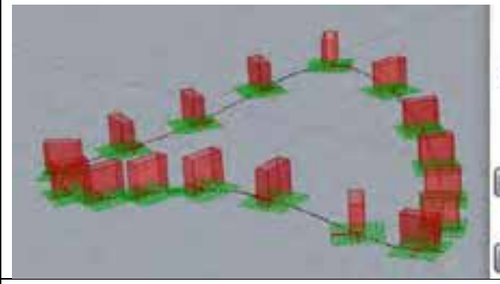
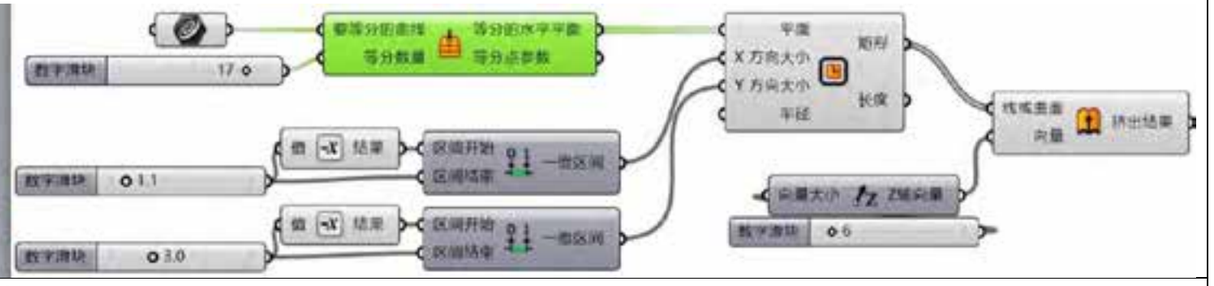
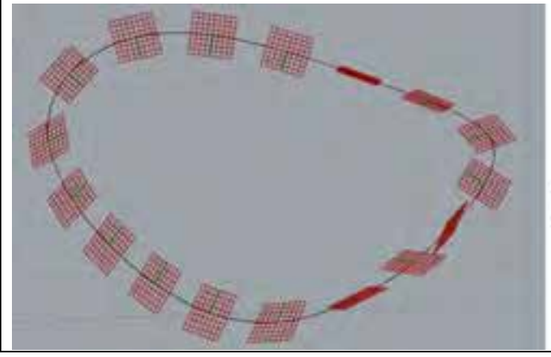
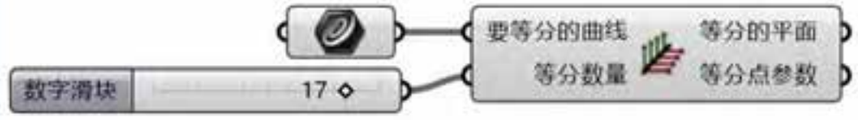
沿指定方向按距离等分曲线，等高线绘制方法



等高线（增强） **Contour (ex)**		**偏移与曲线相交的平面 ，输出偏移后的平面与曲线的交点。输入端距离为多次偏移后的平面之间的距离。** 
虚线等分 **Dash Pattern**		**按设定的间隙模式，把曲线转换成虚线（课程实例：Ace 咖啡 GH 建模 vray 动画分析图一条龙）** 
曲线等分之垂直平面 **Perp Frames**		**按数量等分曲线，等到垂直曲线的平面。** 

实际案例：变截面矩形管道



曲线之水平平面 **Horizontal Frames**		**生成等分点的水平平面。可以用来做幕墙竖挺。**  
曲线等分之负法线平面 **Curve Frames**		**生成以等分点为原点，等分点的切线方向为 X 轴的平面。**  

曲面 Surface

公用 Util

曲面 Surface

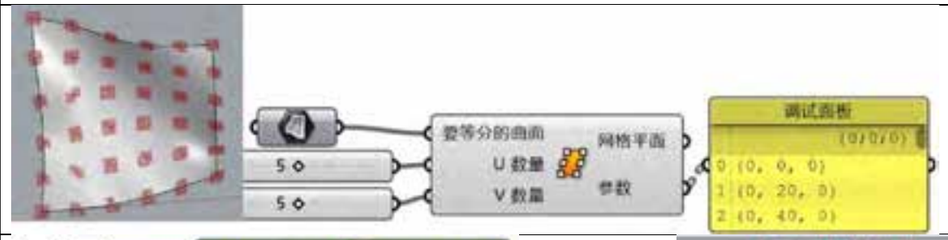
公用 Util - 曲面

曲面之 UV 等分 Divide Surface

基于曲面 UV 坐标，生成细分点，输出点在曲面上的法线方向，及对应 UV 坐标

曲面之网格等分 Surface Frames

生成曲面上包含等分点，且与曲面相切的平面



修剪还原 Untrim

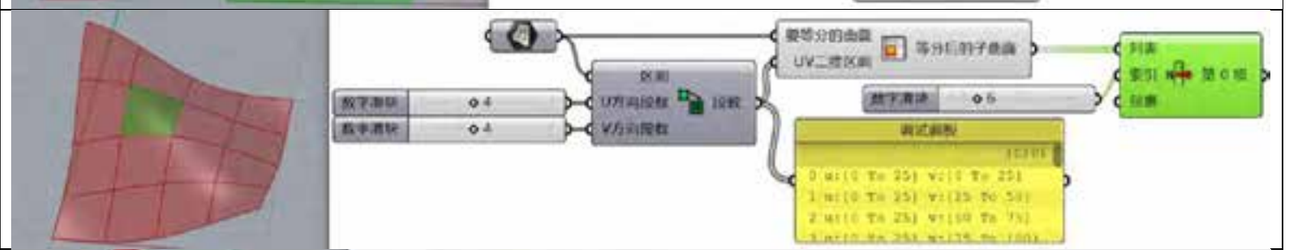
还原修剪过的曲面

曲面修剪流动 Copy Trim

类似曲面流动，把曲面投射到另一个曲面上。

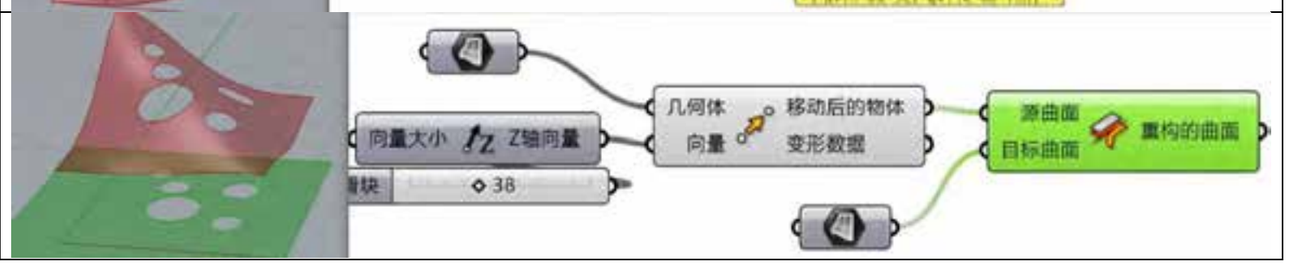
曲面等分 Isotrim

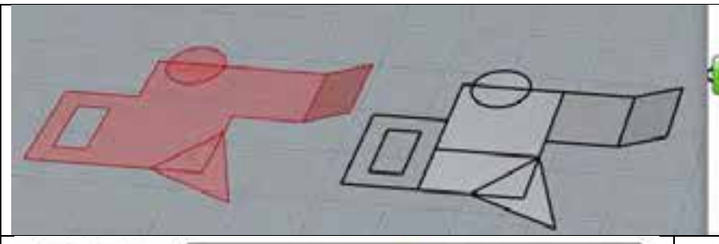
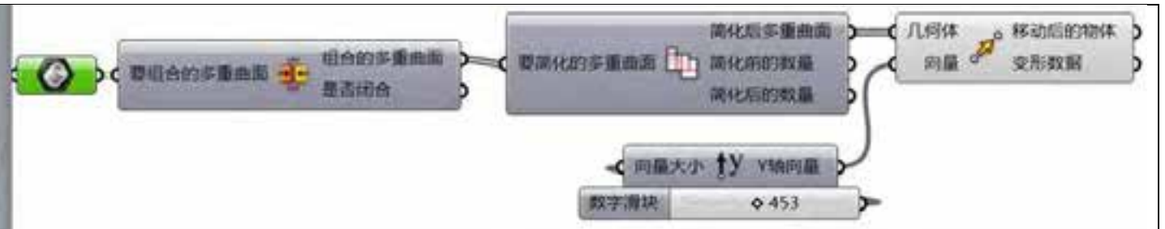
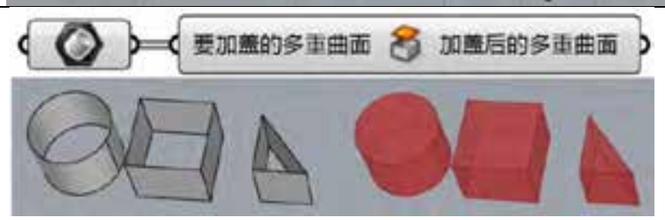
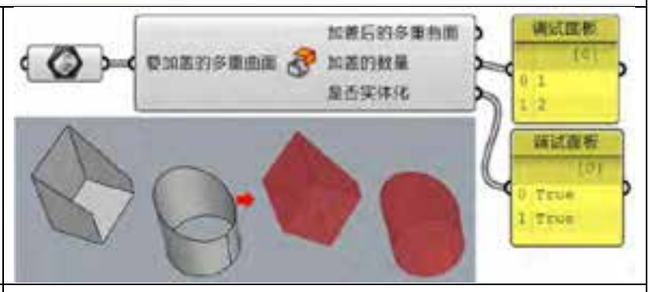
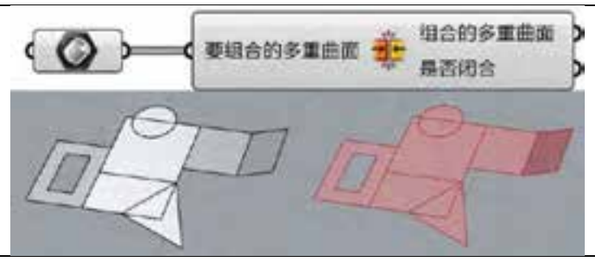
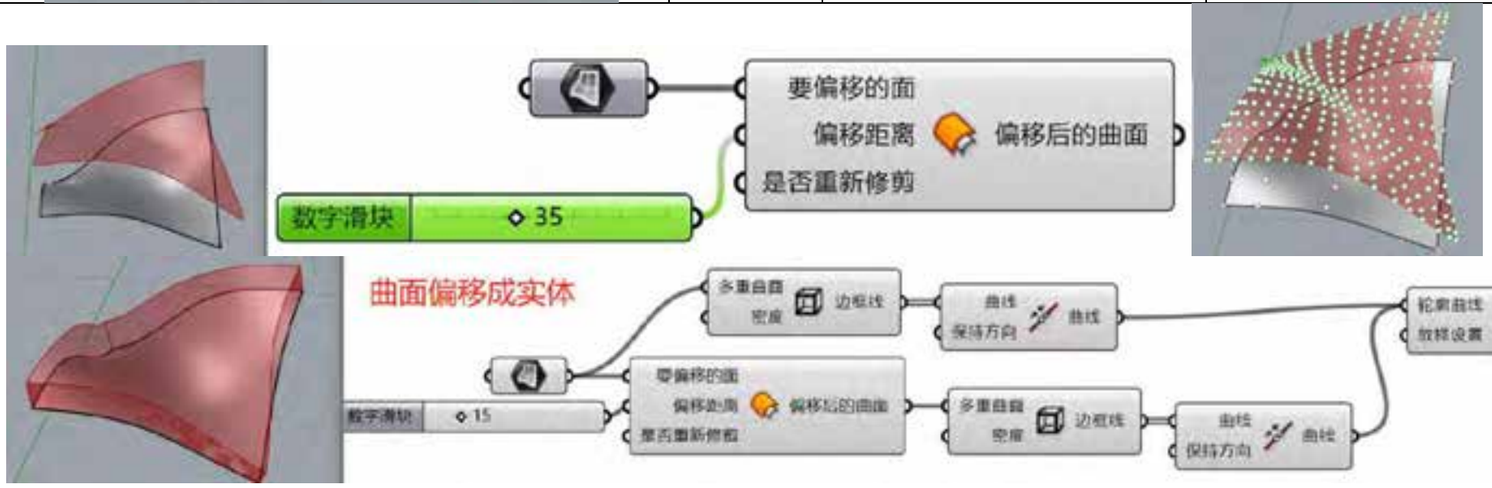
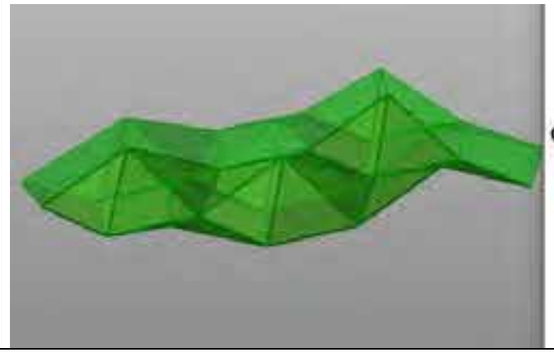
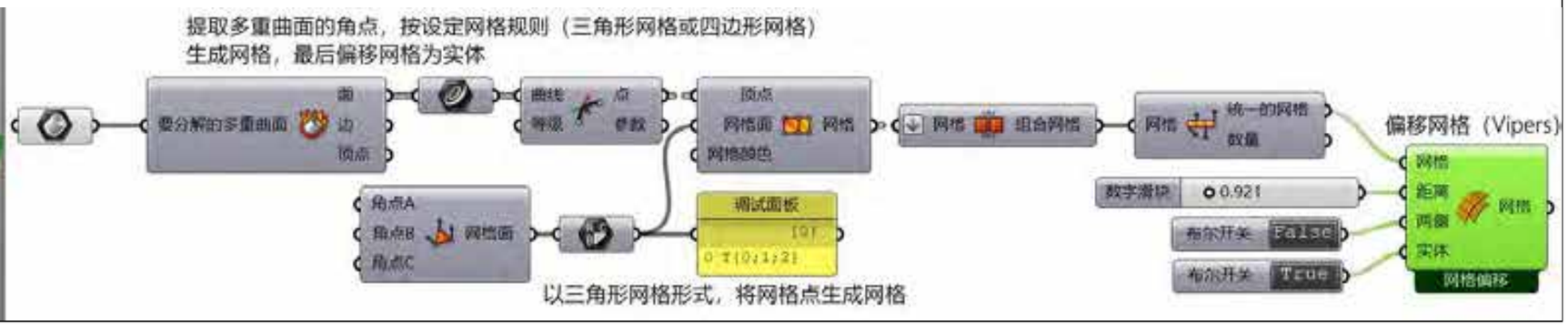
根据设定的 UV 二维区间，等分曲面

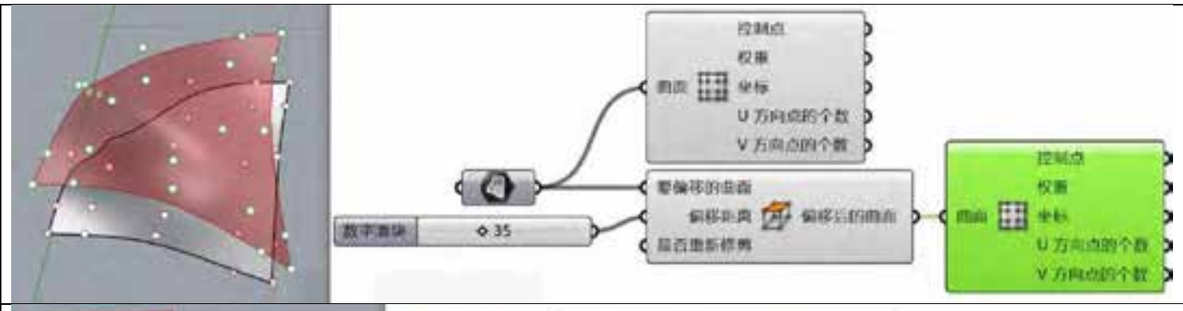
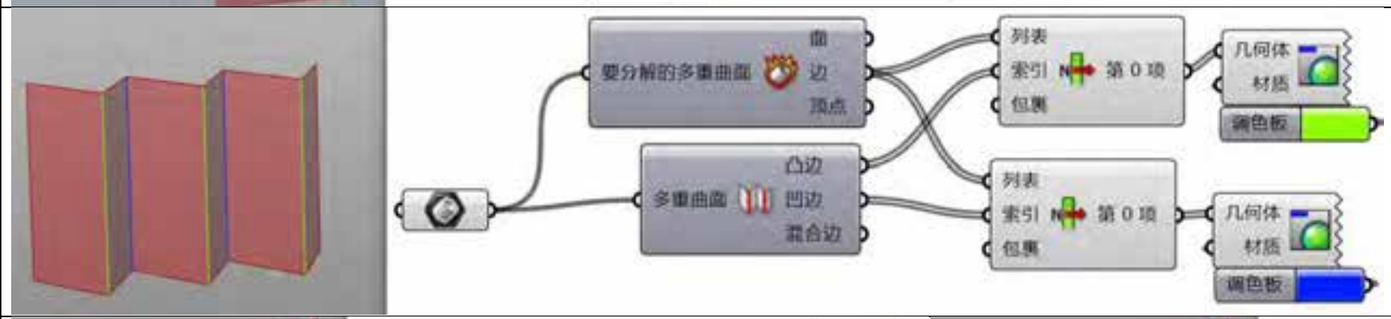
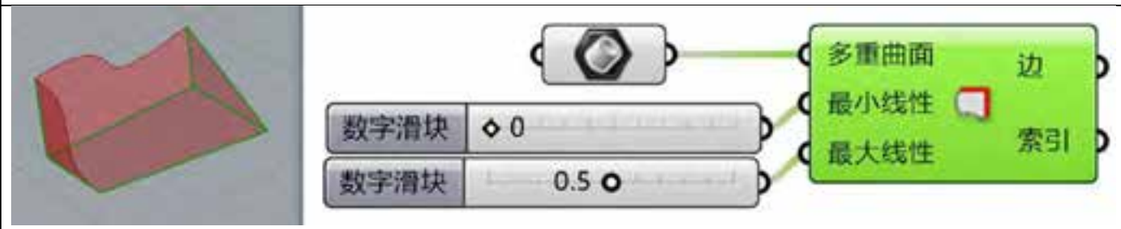
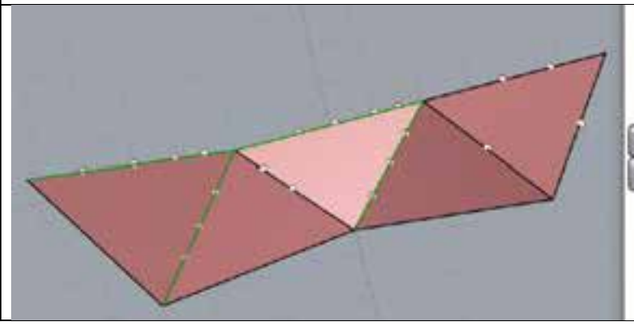
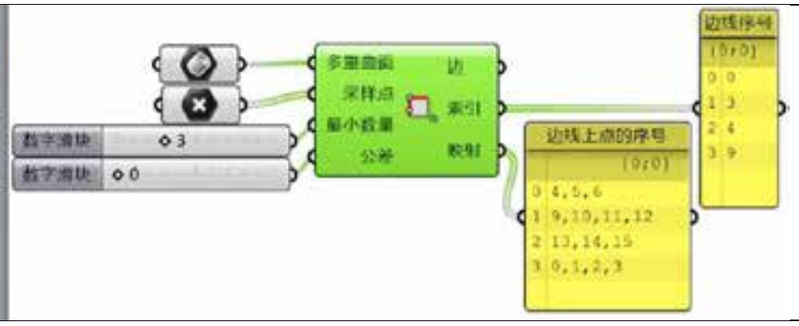


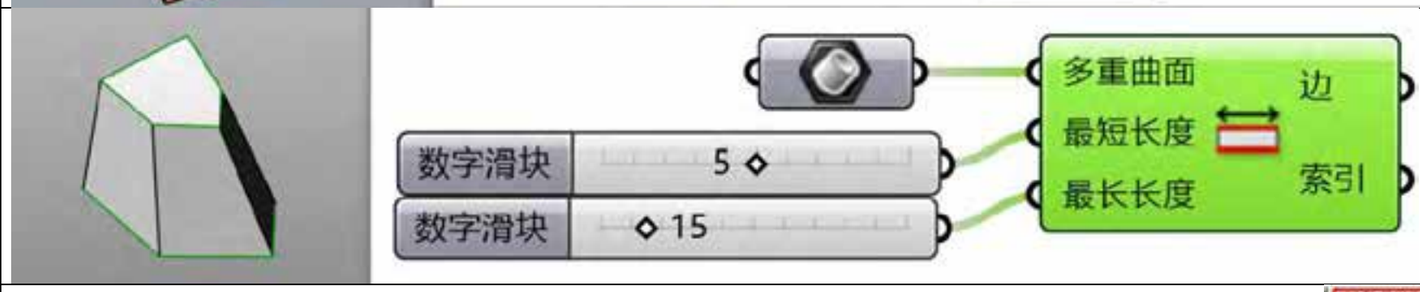
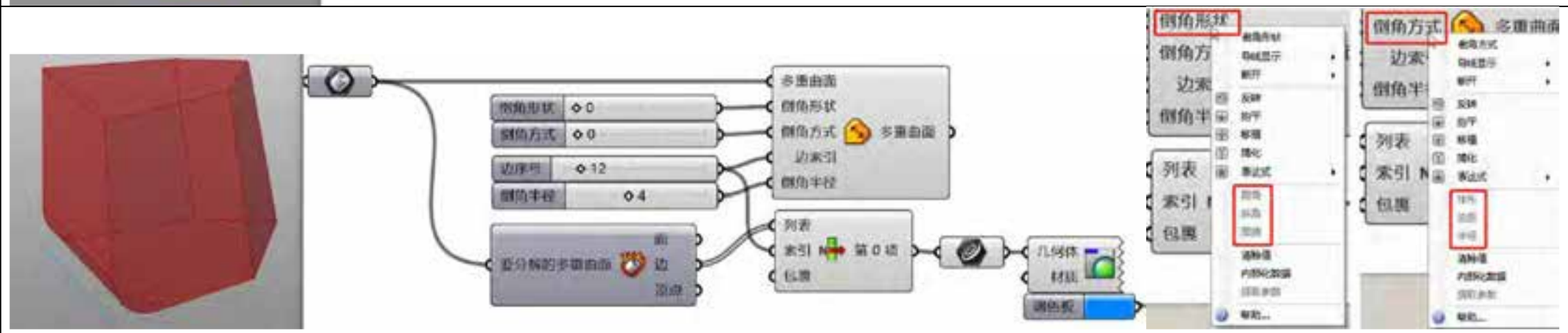
曲面重构 Retrim

用源曲面的修剪线，投射到目标曲面，去修剪目标曲面



合并曲面 **Merge Faces**	 **有共边的多个平面合并成一个平面，不共边的无法合并。**				
曲面加盖 **Cap Holes**	 **加盖，只能加平面的洞，异形曲面洞无法加盖。**		**曲面加盖** **(增强)** **Cap Holes Ex**	 **升级版曲面加盖**	
组合多重曲面 **Brep Join**	 **将多个曲面或者多重曲面，组合成一个多重曲面。** **但是不共边的曲面无法组合在一起，烘焙出的实体依然是独立曲面。**				
偏移曲面 **Offset Surface**	 **偏移曲面，使曲面上的点沿法线偏移，所以空间曲面偏移后会变形。** **偏移后的曲面控制点增多，下面的运算器则是保持偏移后的曲面控制点数量不变**	 			
	多重曲面偏移成实体的方法： 	 			

偏移曲面(控制点) Offset Surface Loose	 **偏移后控制点数量不变，因此偏移后曲面的精确度相比上面的运算器会降低。**	
翻转曲面 Flip	 **翻转曲面的方向。**	
凹凸边 Convex Edges	 **用于筛选多重曲面上凹凸关系的边，输出端为对应凹凸边的序号。** **凹凸关系由曲面方向决定。**	
边从方向 Edges from Directions	 **用于查找多重曲面上，跟指定方向（也可以是直线）相同，或者是在与方向夹角在指定角度范围内的边线**	
边从线性 Edges from Linearity	 **应该就是用于选取多重曲面上的直线。最小线性默认为 0，最大线性默认为 1，暂时没发现这个参数的作用。**	
边从点 Edges from Points	 **根据多重曲面各边的采样点的最小数量筛选边线，边线上采样点数量大于最小数量的会被选出，输出对应边线的序号（索引），及对应边线上采样点的序号（映射）**	 

边从面 **Edges from Faces**	 **筛选出多重曲面上，包含指定点的面的边缘，输出边的序号**	
边从长度 **Edges from Length**	 **根据指定的长度范围，筛选多重曲面上的边线**	
边倒圆角 **Fillet Edge**	 **根据指定序号的边，对多重曲面进行倒角。** **如果指定序号的边无法倒角，运算器会报错。** **倒角形状：0 圆角、1 斜角、2 混接** **倒角方式：0 球形、1 边距、2 半径** **倒角形状及倒角方式都可以右键选择预设的方式。**	

分析 Analysis - 曲面

分解盒子 Deconstruct Box

分解盒子得到 XYZ 轴边长区间，及盒子所在平面

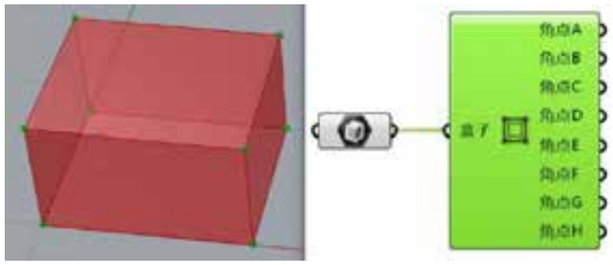
盒子属性 Box Properties

分解盒子得到相应属性参数，对角线向量的大小为对角线长度。简并性暂时无法理解。

简并（degeneracy，但英文 degeneracy 具有多种含义），在物理学、生物学等领域有不同解释。例如在量子力学中，原子中的电子，由其能量确定的同一能级状态，可以有两种不同自旋量子数的状态，该能级状态是两种不同的自旋状态的简并态。生物学上，简并是指遗传密码子的简并性，即同一种氨基酸具有两个或更多个密码子的现象。

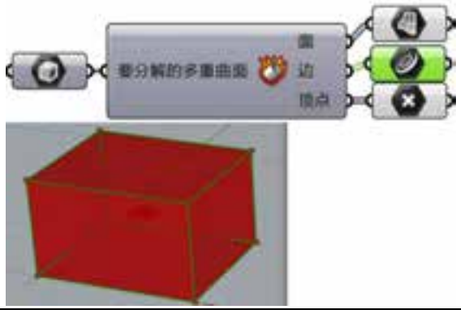
盒子顶点 Box Corners

分解盒子等到盒子的 8 个顶点



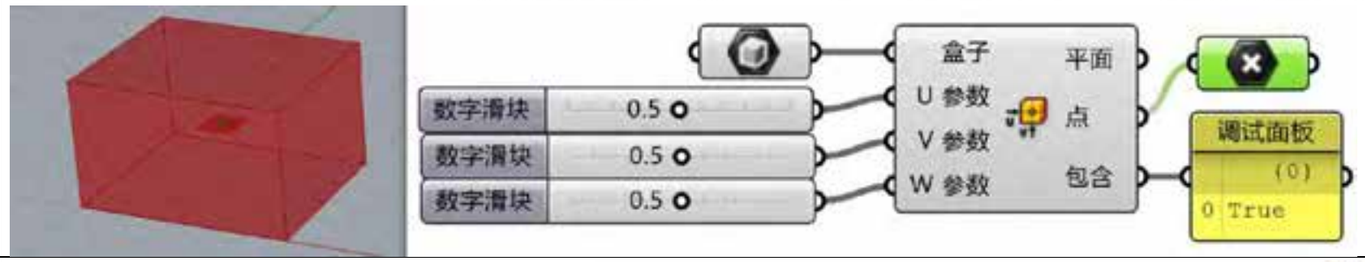
分解多重曲面 Deconstruct Brep

提取多重曲面的面、边、顶点，很常用



计算盒子 Evaluate Box

基于盒子的长宽高归一化之后的 UVW 坐标系，输入对应值定位盒子中的点，UVW 都是 0.5 时是在盒子的中心，在 0-1 范围以外则是盒子外部的点。



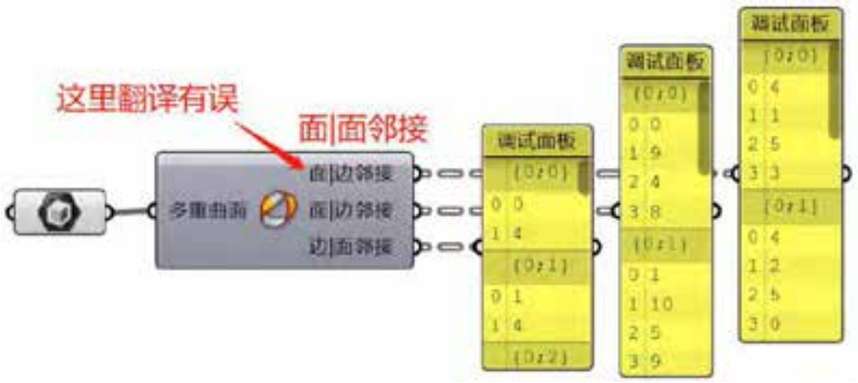
多重曲面拓扑结构 Brep Topology

显示一个多重曲面的拓扑关系。

面与面邻接：与一个面相接的面的序号都放在一组。

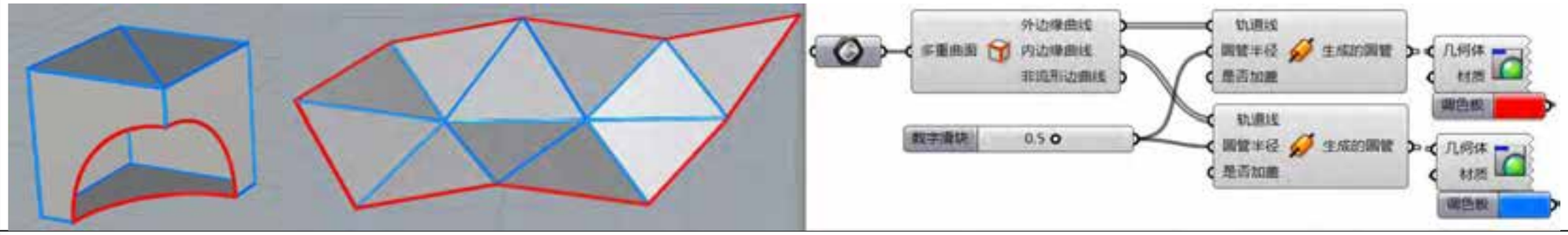
面与边邻接：每个面周围的边的序号放在一组。

边与面邻接：每条边相接的面的序号放在一组。



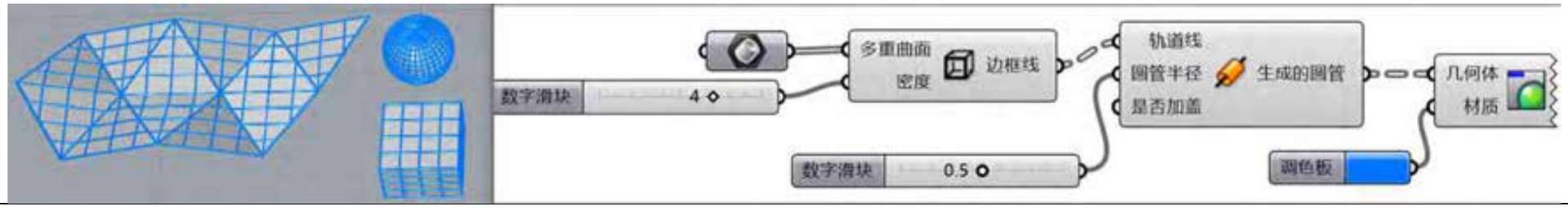
多重曲面边提取
Brep Edges

提取多重曲面的边缘。非流行边缘为被三个或三个以上网格面或曲面共用的边缘。这样的边缘是不合理的，出现即意味着多重曲面有错误。



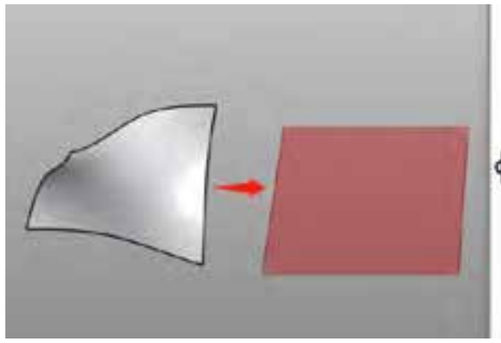
提取多重曲面边框
Brep Wireframe

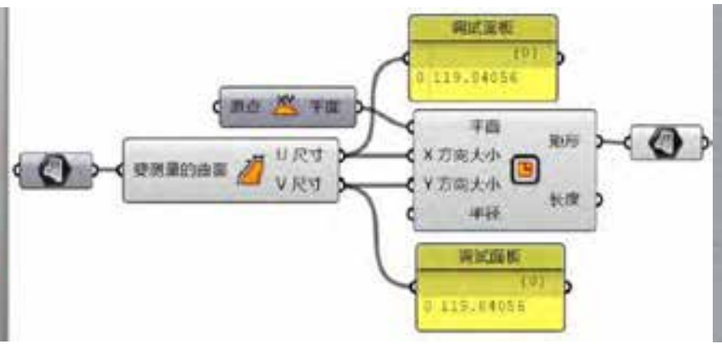
提取多重曲面的结构线，可以设置不同的结构线密度。

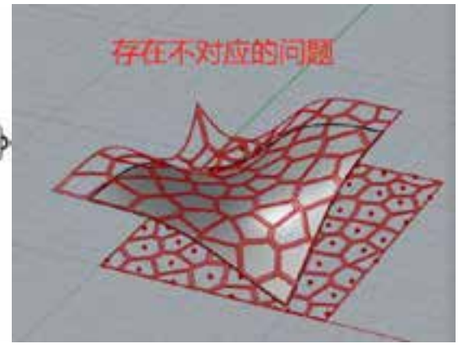


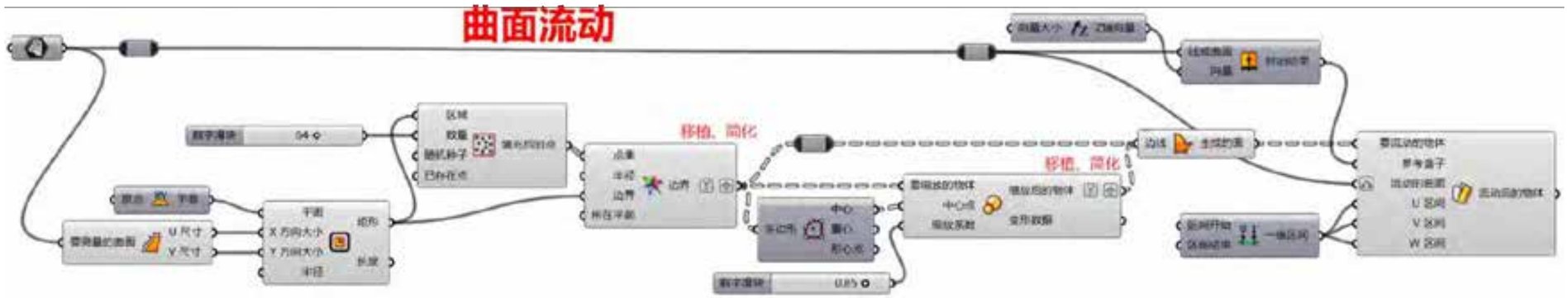
曲面 UV 尺寸
Dimensions

提取曲面的 UV 长度。





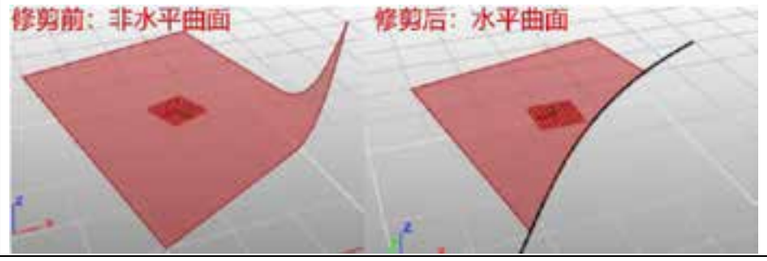


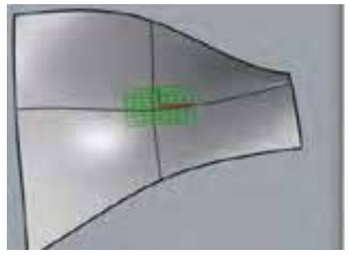


曲面平面性判断
Is Planar

判断一个曲面是不是水平曲面，返回布尔值和曲面中心点处与法线垂直的平面。

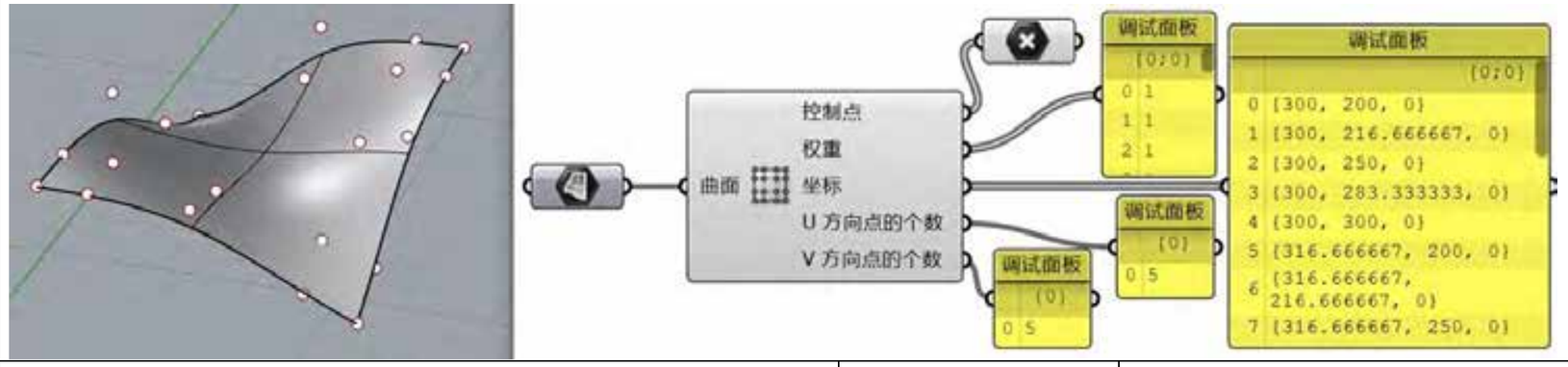
内部为布尔值，当为 true 时，如果一个修剪后的曲面，剩余部分是水平曲面，则判定为水平曲面。





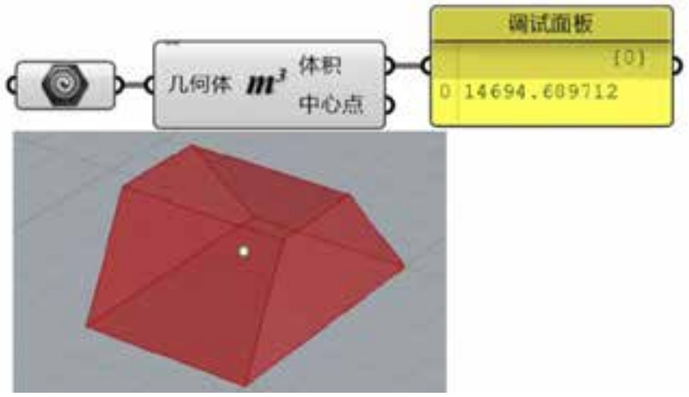
曲面控制点
Surface Points

提取曲面的控制点，及控制点的相关参数。



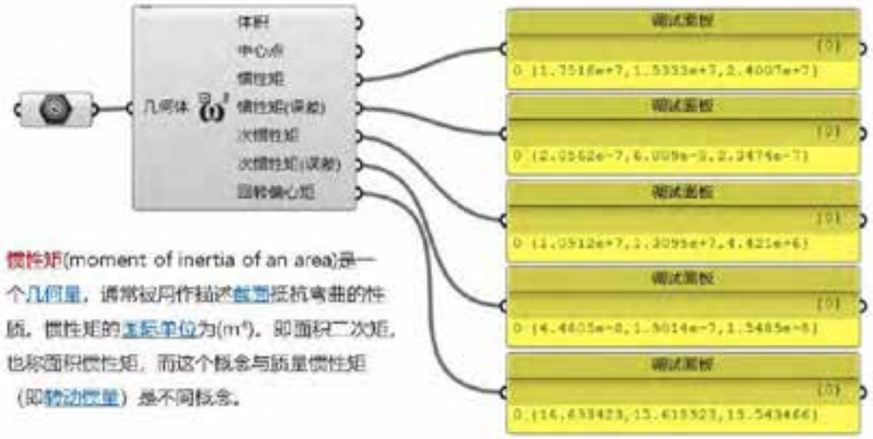
体积
Volume

计算几何体的体积，并输出几何体的中心点



体积矩
Volume Moments

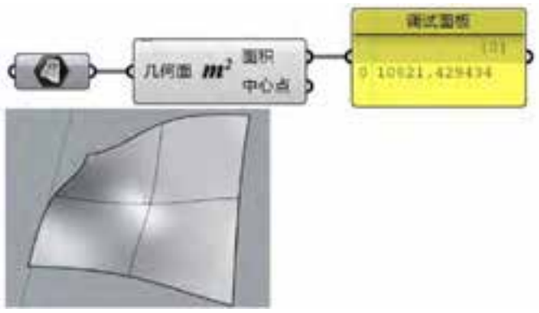
求几何体体积和中心点，及惯性矩相关参数



面积
Area

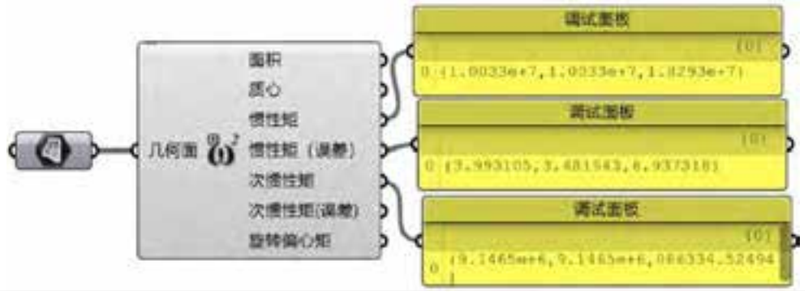
计算曲面的面积，及曲面中心点。

如果只是提取矩形、网格或多边形的中心点，用多边形中点 Polygon Center (曲线工具) 运算器会更节省运算时间。



面积矩
Area Moments

计算曲面的面积，中心点及惯性矩相关参数



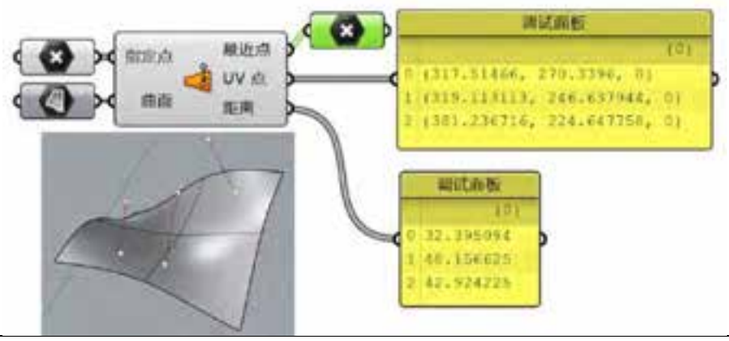
最近点之多重曲面
Brep Closest Point

计算多重曲面上与指定点距离最近的点，输出最近点的向量及两点的距离

最近点之曲面
Surface Closest Point

计算曲面上与指定点距离最近的点，输出最近点的 UV 坐标及两点的距离。

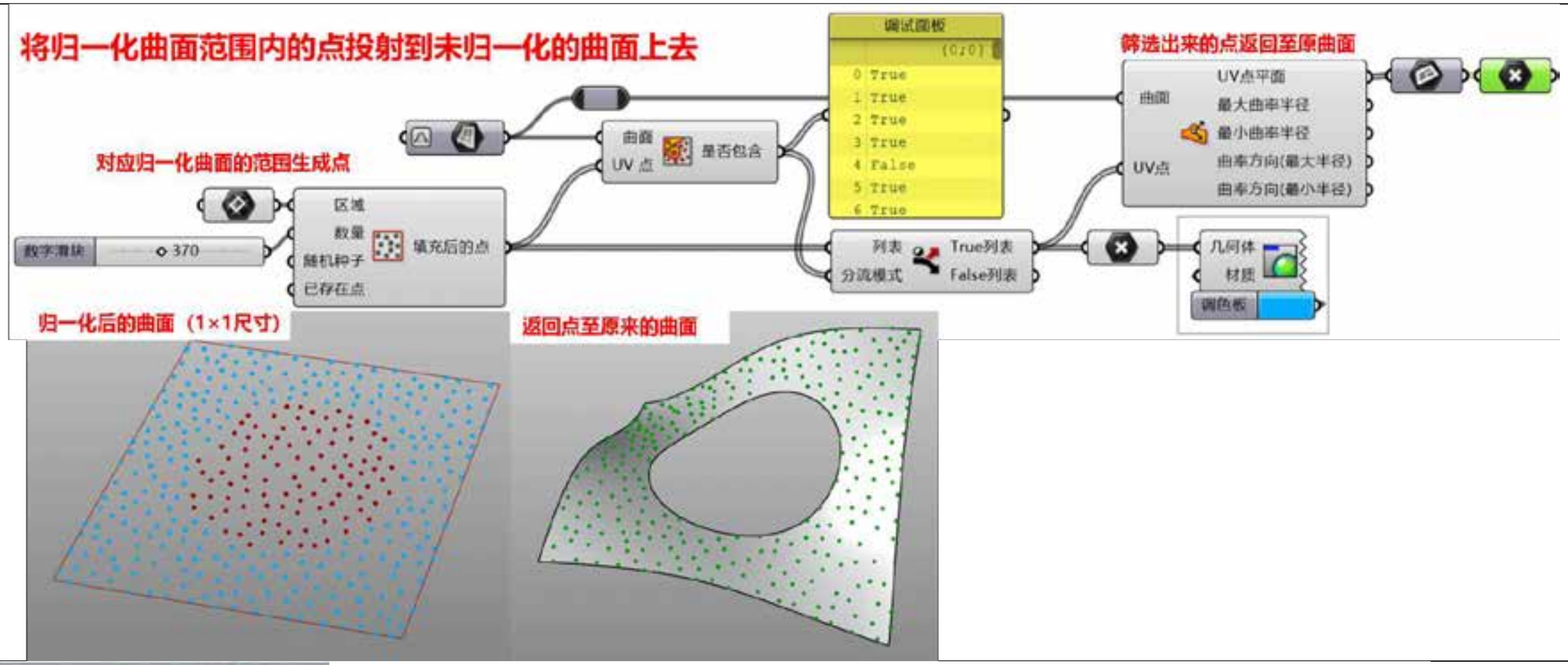
（可以用于把曲面上的点的世界坐标值转换成 UV 坐标值，比较常用的运算器）



修剪曲面内部点判断

Point In Trim

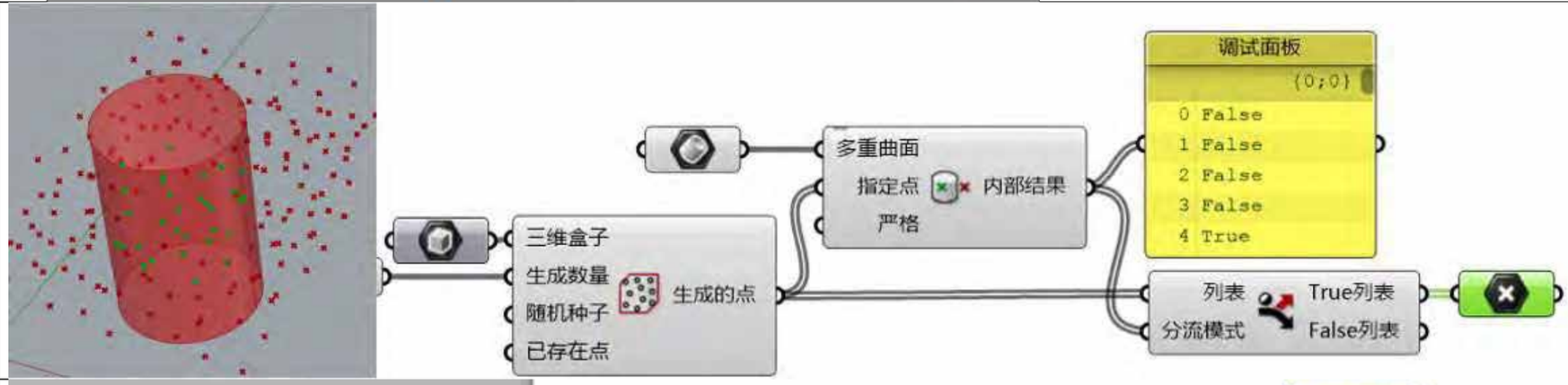
判断点是否在修剪过的曲面内。



单个多重曲面内部点

Point In Brep

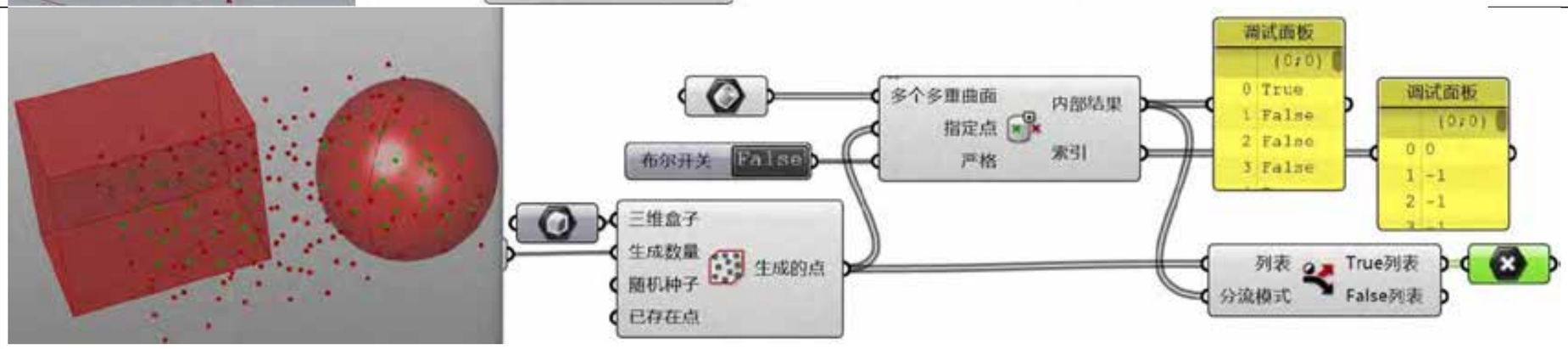
判断点是否在多重曲面内部。严格为布尔值，默认 false，如果为 true，则在几何体表面上的点不计入结果。

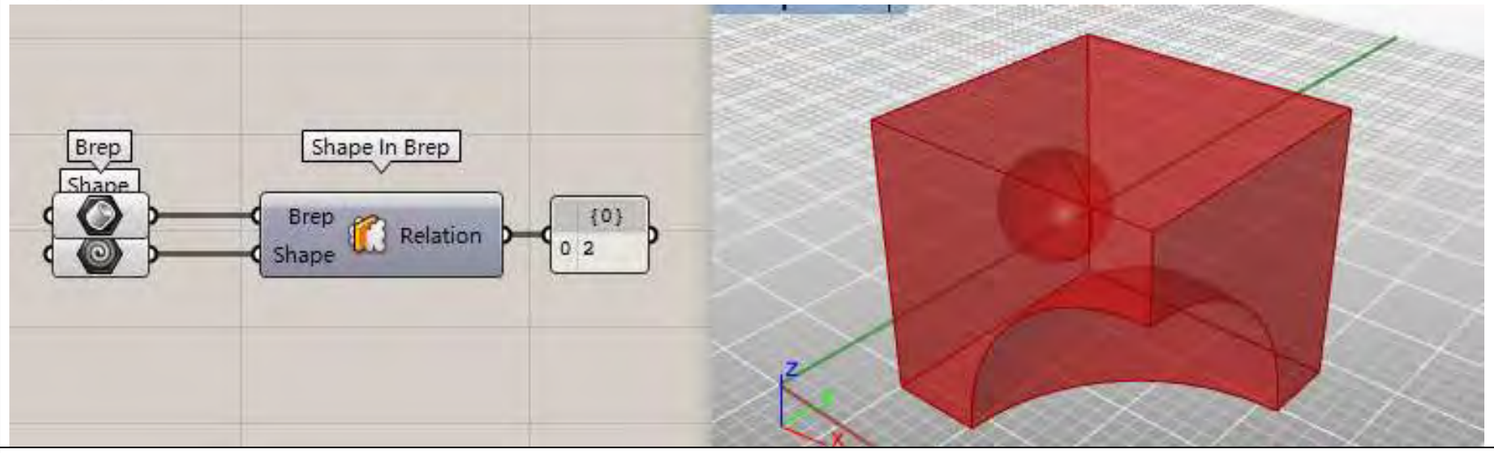
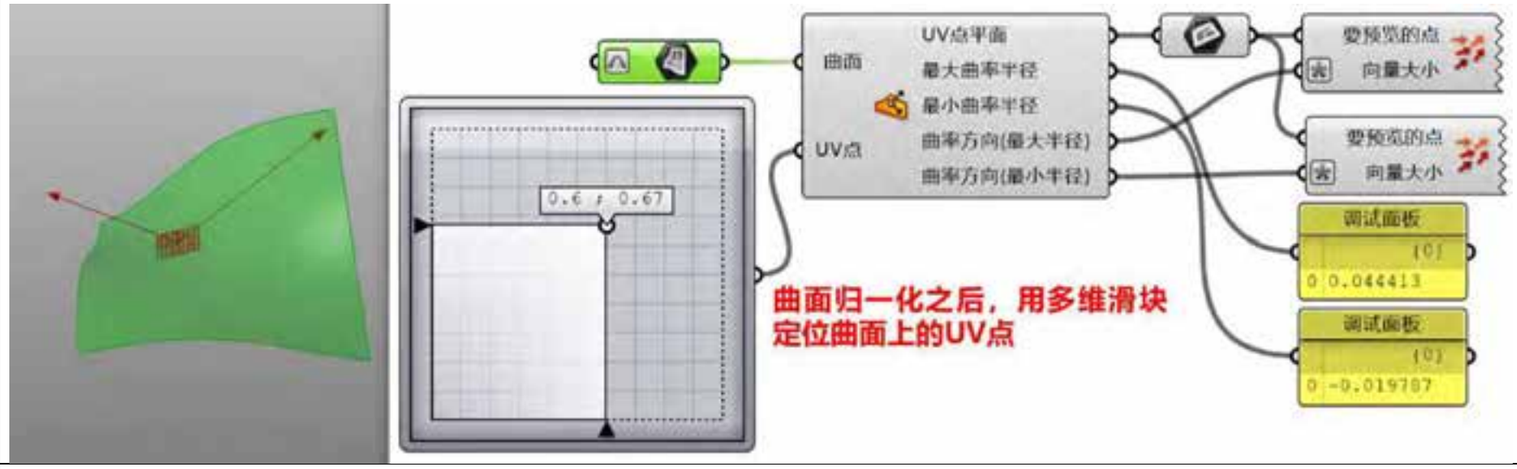
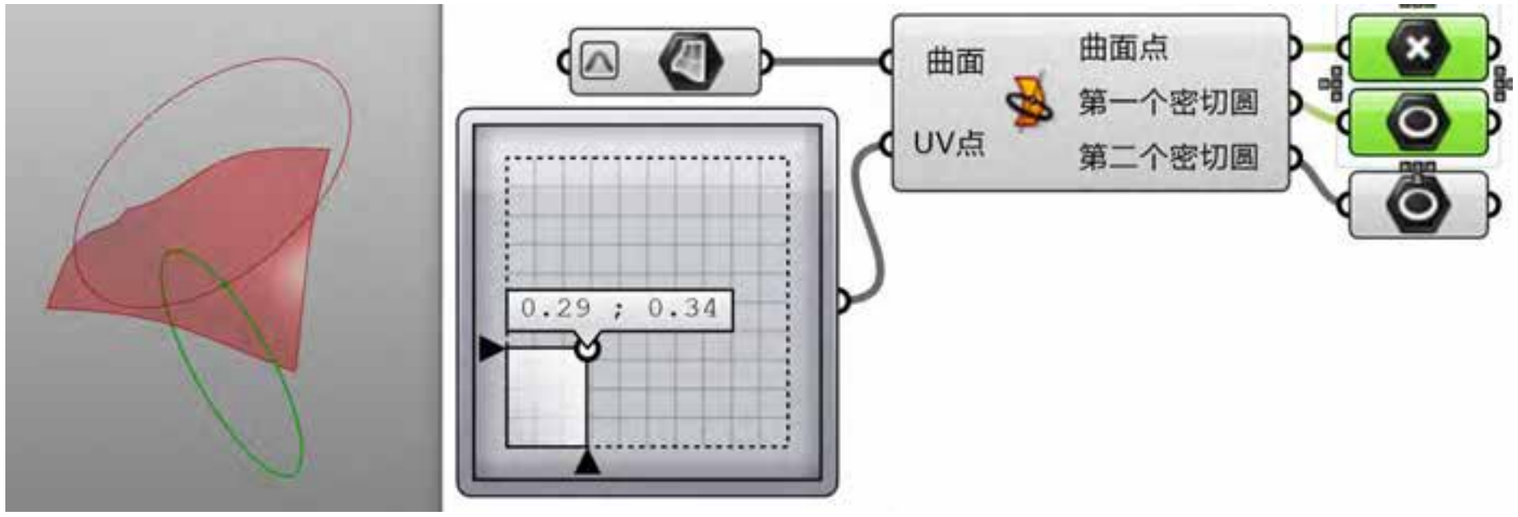


多个多重曲面内部点

Point In Breps

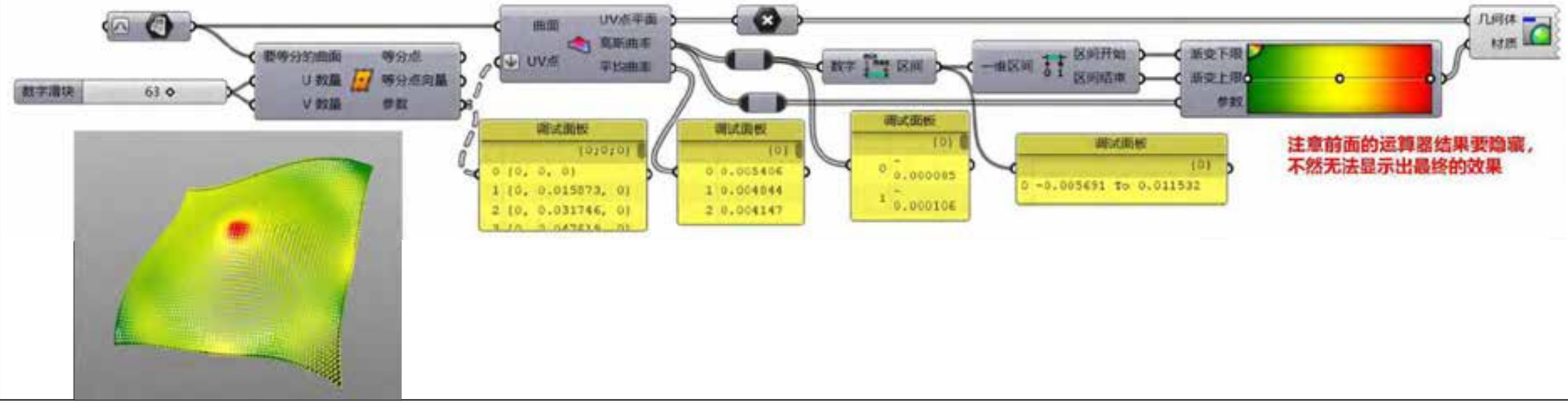
判断点是否在多个几何体内部，索引为对应点所在几何体的序号，都不包含的为-1。严格为布尔值，默认 false，如果为 true，则在几何体表面上的点不计入结果。



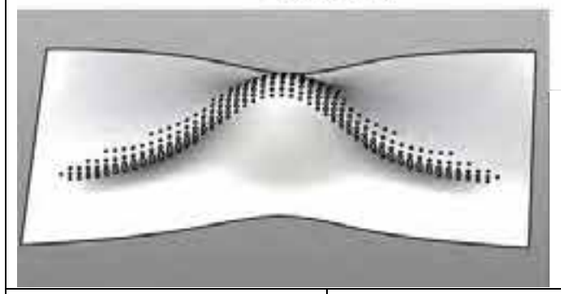
多重曲面内部判断 **Shape In Brep** 	**判断一个物体【Brep】是否在另一个物体【Shape】的内部，返回 0 则在内部，返回 1 则有是有相交部分，返回 2 则在物体外部，需要确保物体是包含关系的，如上图对调一下输入端的话，输出的也会是 2，因为小球无法包含大物体。** 
UV 点曲面曲率 **Principal Curvature** 	**求曲面上指定 UV 坐标的点所在的平面，及该点的对应的最大和最小曲率，曲率的向量方向。** 
密切圆 **Osculating Circles** 	**根据曲面指定 UV 点位置的曲率绘制密切圆，是显示曲面曲率变化的一种方式。** 

曲面曲率
Surface Curvature

计算曲面上指定点的曲率。下面为对曲面的曲率进行可视化。



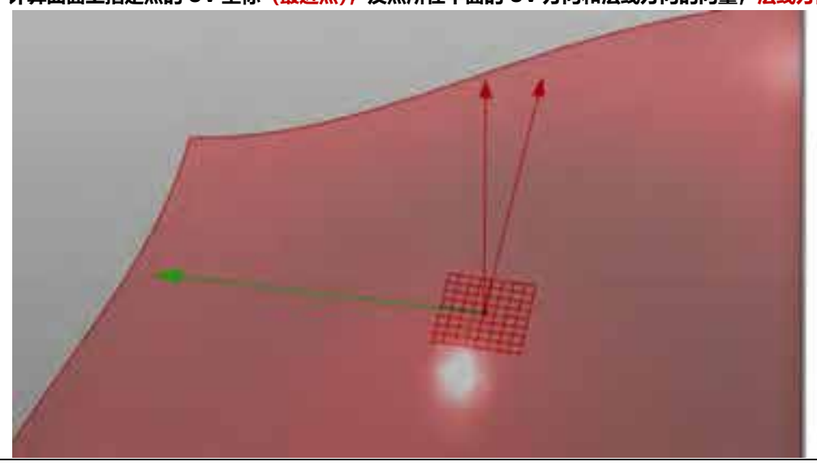
曲率干扰曲面开洞



建模思路：将曲面等分，取等分曲面的中心点生成圆，将圆心对应的曲率进行函数映射，用映射后的值缩放圆，挤出圆柱面与曲面相交，得出相交线，最后用相交线修剪曲面。

计算局部曲面
Evaluate Surface

计算曲面上指定点的 UV 坐标（最近点），及点所在平面的 UV 方向和法线方向的向量，法线方向类似世界坐标系的 Z 轴（曲面是二维坐标系，不存在 Z 轴），U 方向等同 X 轴，V 方向等同 Y 轴。

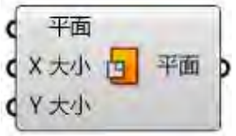


初始 Primitive- 曲面

形状生面
Plane Through Shape

在物体的指定截面位置（平面），生成一个把物体围住的矩形，右键可以选择矩形范围是基于物体的截面（section only）还是整个物体在平面上的投影面（total），膨胀度可以调节矩形大小。

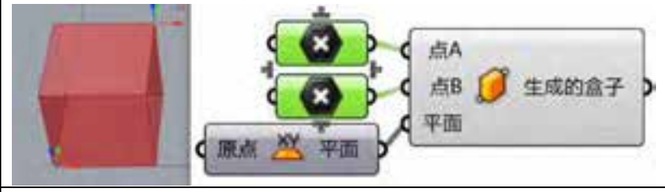
面生曲面
Plane Surface



在指定平面上，按设定尺寸生成矩形平面

2 点生盒
Box 2Pt

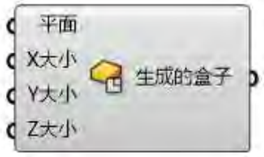
通过指定盒子的两个对角点生成盒子



区间生盒
Domain Box

以区间的长度为边长，生成盒子

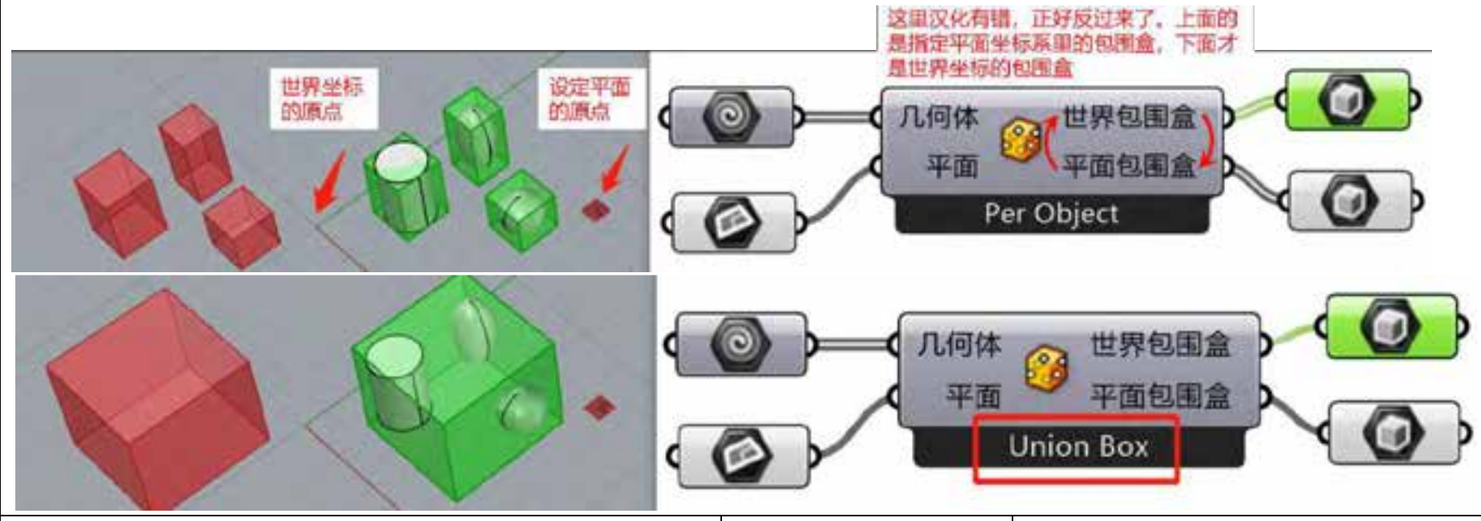
平面生盒
Center Box



以指定平面的原点为盒子的中心，按设定的各边长一半的数值，生成盒子。

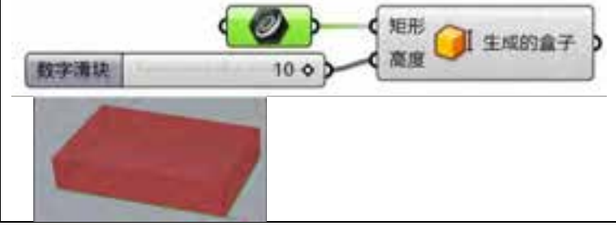
外包围盒
Bounding Box

生成包住指定物体的盒子，右键选择 Union Box，则生成一个包住所有物体的盒子。世界包围盒为在世界坐标里，根据指定的平面所在的坐标位置，也生成相应的包围盒，可以理解成 CAD 和 SU 里的指定基点复制。(输出端汉化版翻译有错，顺序要反过来)



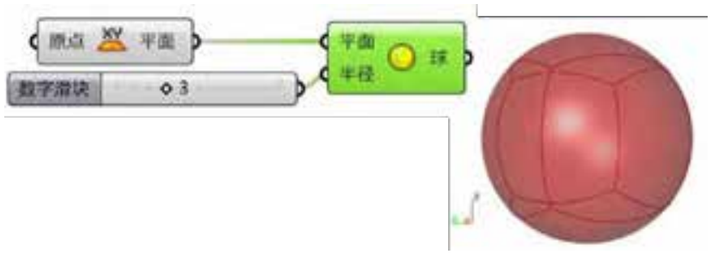
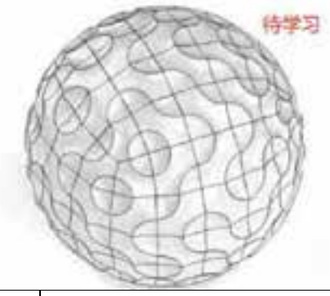
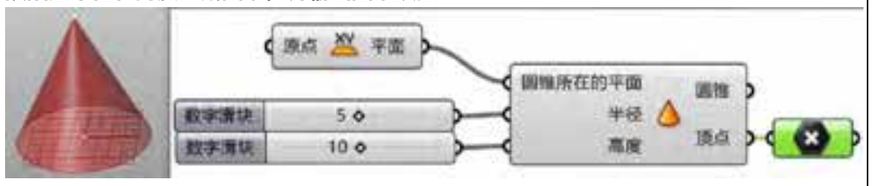
矩形生盒
Box Rectangle

根据指定的高度，把矩形挤出成盒



4 点生球
Sphere 4Pt

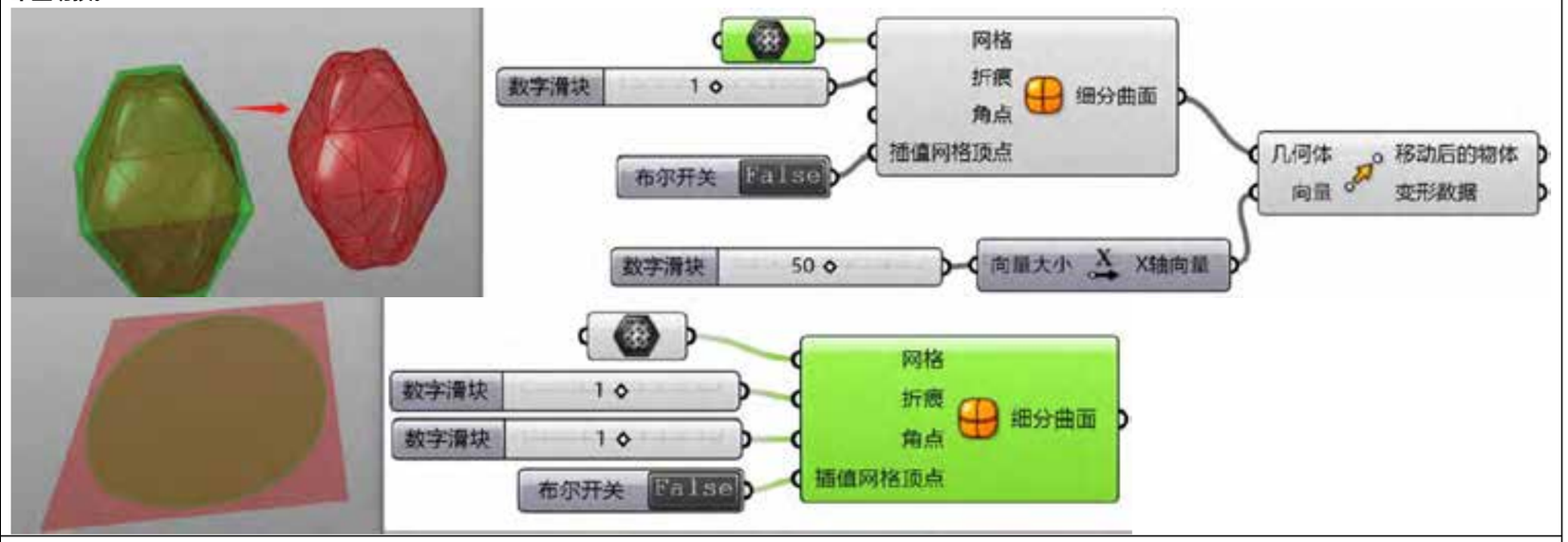
根据四个点生成球体，并输出球心及球半径

四边形生球（高质量） **Quad Sphere** 	**在指定平面上，按设定半径生成一个由六个四边曲面拟合的球体。**  		
圆柱 **Cylinder** 	**按指定的半径和高度生成圆柱** 	**圆锥** **Cone** 	**按指定半径和高度生成圆锥，并输出圆锥顶点** 
拟合球 **Sphere Fit** 	**根据点集生成拟合球体，输出球心及球半径** 	**球** **Sphere** 	**根据半径生成圆**

细分 Subd- 曲面

细分从网格
SubD from Mesh

根据网格创建细分曲面，折痕可以控制网格边缘处添加锐边。角点则是控制是够柔滑角点（针对平面的细分曲面），这两个参数都只能由 1 和 2 来输入控制，或者右键菜单里切换。

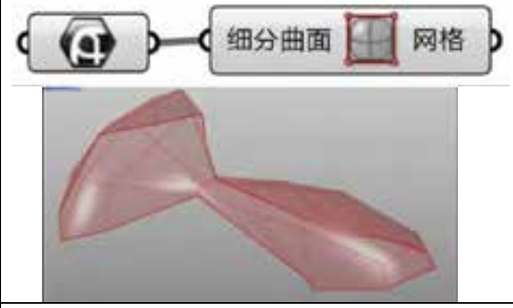


网格从细分曲面
Mesh from SubD

获取细分曲面的原始网格，密度可以控制提取的网格密度。

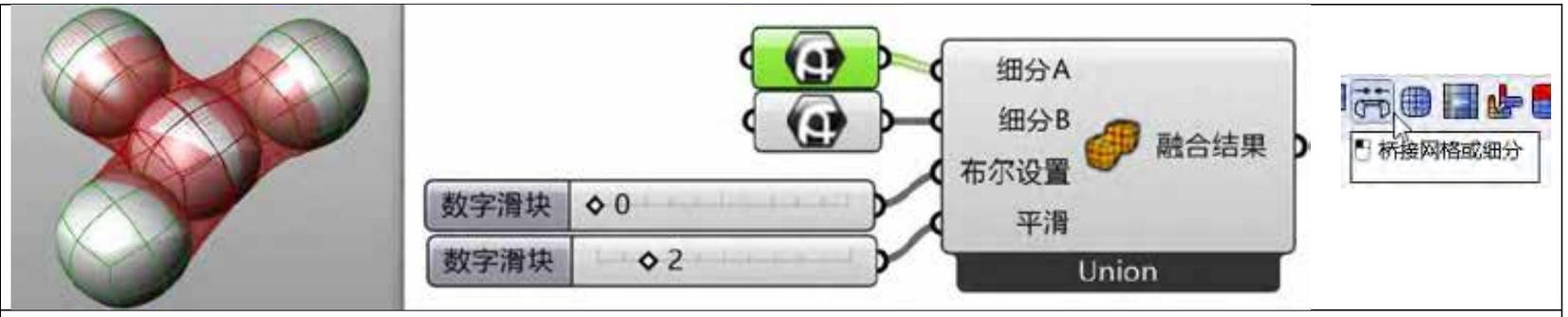
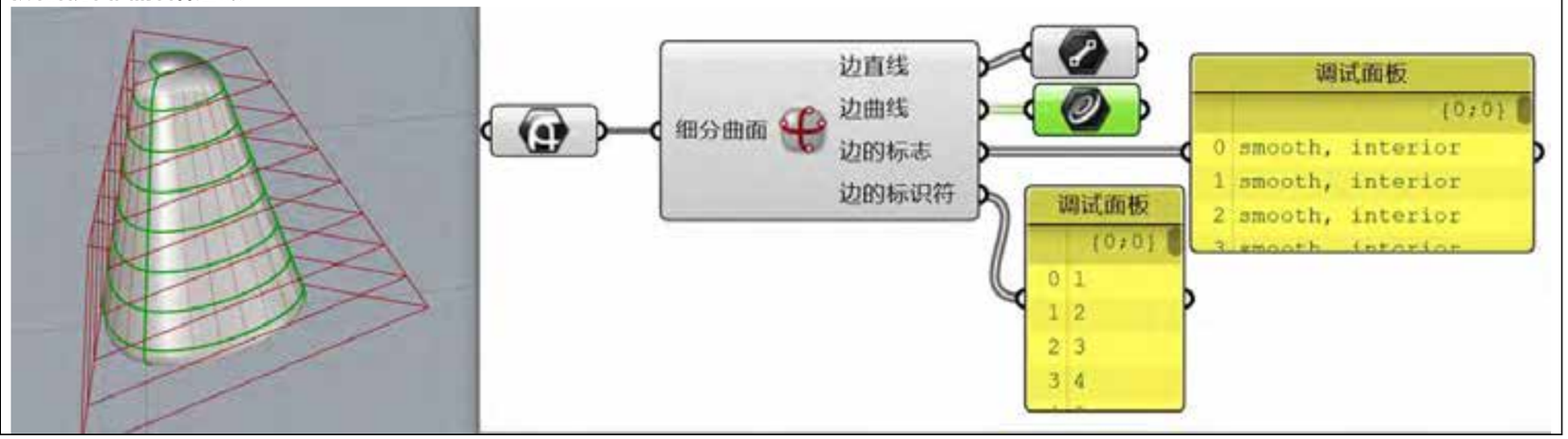
细分控制多边形
SubD Control Polygon

获取细分曲面的控制网格



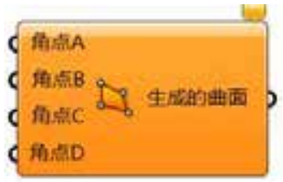
细分融合
SubD Fuse

对两组细分物件进行熔接，类似 rhino 中的细分曲面中的桥接网格和细分功能，但是这个运算器比较局限，不能对多个细分物件之间都进行熔接。

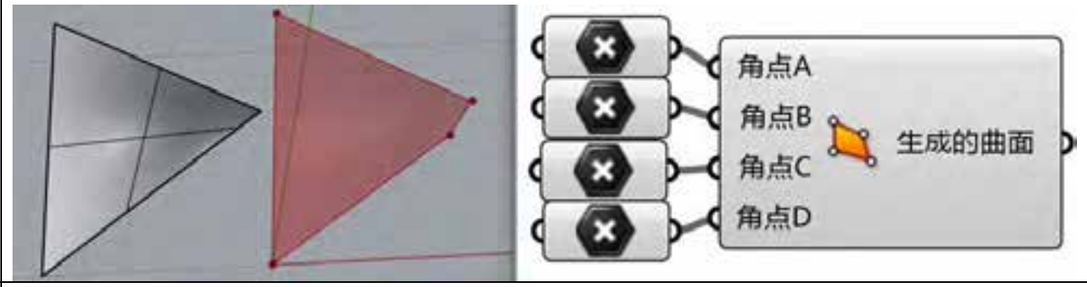
		
	细分边缘 **SubD Edges** 	**获取细分曲面的各种边线。** 

自由变换 Freeform- 曲面

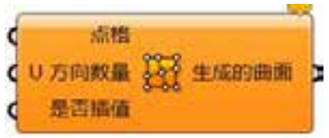
四点生面
4Point Surface



根据指定的四个点生成曲面，要注意点的顺序



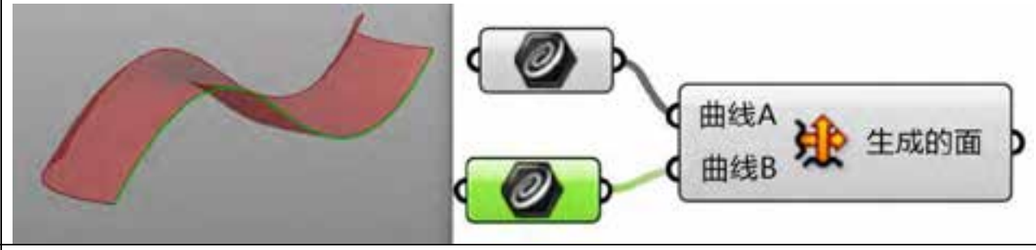
点格生面
Surface From Points



根据指定点集生成曲面，点的排序要符合 UV 方向的排布（暂时还没完全理解其中的原理）。
之所以只用输入 U count,是因为运算器会自动根据总数求出 V 方向的个数。

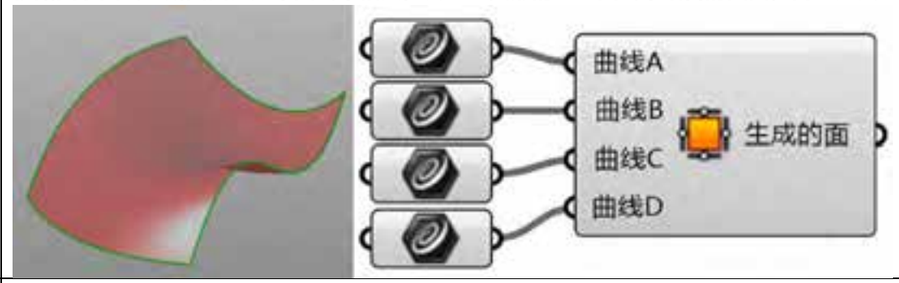
双边扫掠生面
Sum Surface

根据两个曲线扫掠形成曲面，属于单轨放样的阉割版。



四边生面
Edge Surface

根据四条曲线生成曲面。如果曲线不相交，会产生变形的且与曲线不结合的曲面，而且无法烘焙到 rhino 中。



拟合放样
Fit Loft

根据多次测试，应该是以多条曲线中的起始和末尾曲线放样成曲面，中间的曲线控制曲面舒张起伏趋势（类似控制点调节曲面）。

放样生面

Loft

放样设置

Loft Options

根据选择的曲线放样生成曲面，放样设置可以调节放样的相关参数，跟 rhino 的放样设置参数一致。

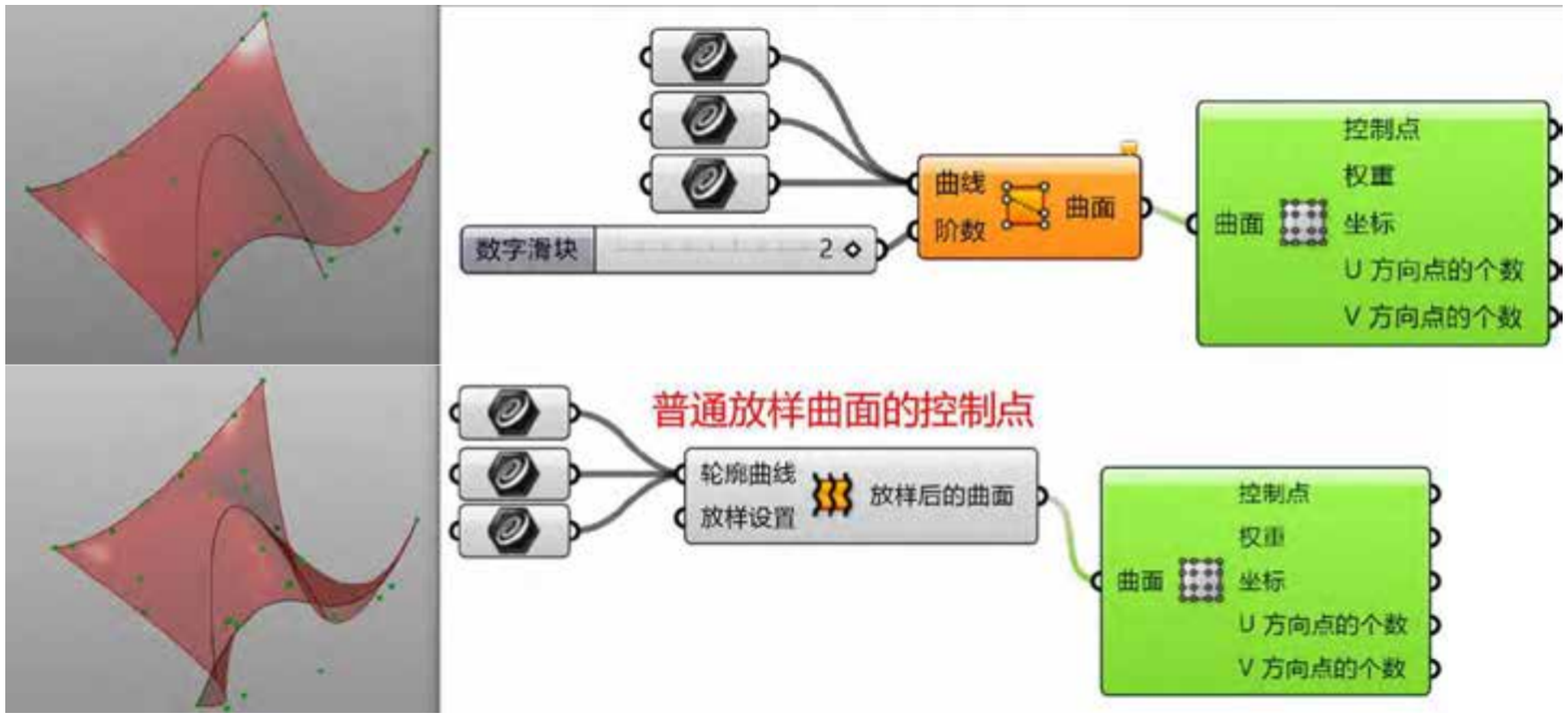
样类型可以参照 rhino 命令中的选项,0=普通放样;1=松弛,2=紧绷,3=平直区段, 5=均匀（没有 4）。

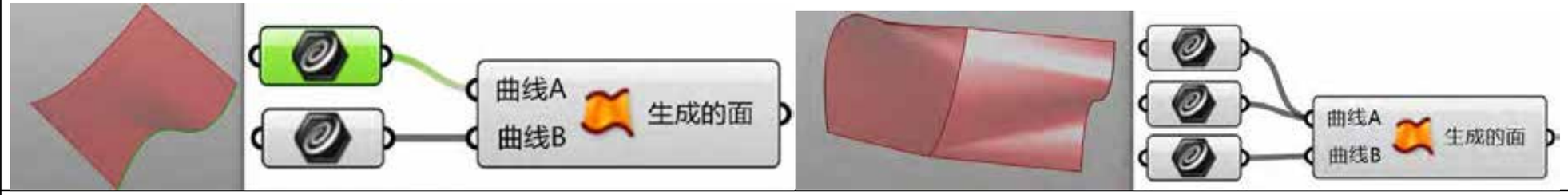
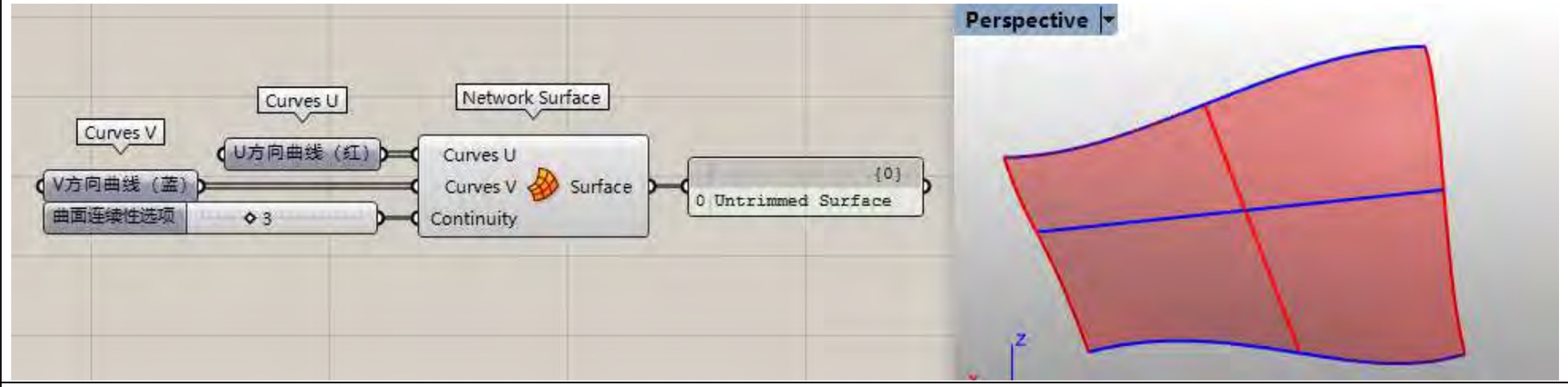
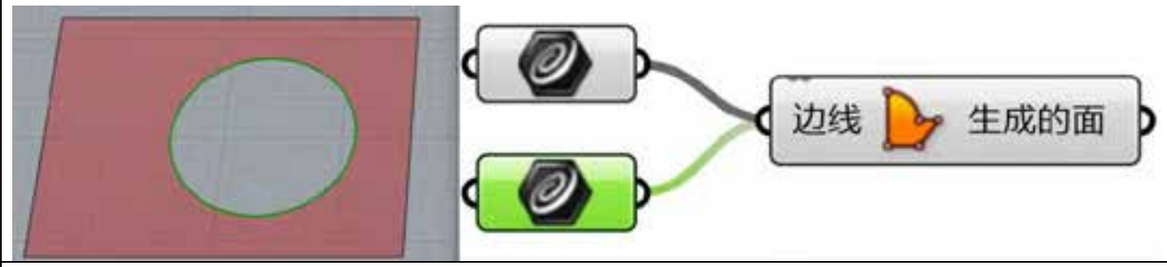
Grasshopper 不支持曲线到点的放样，因此可以使用以下方法。

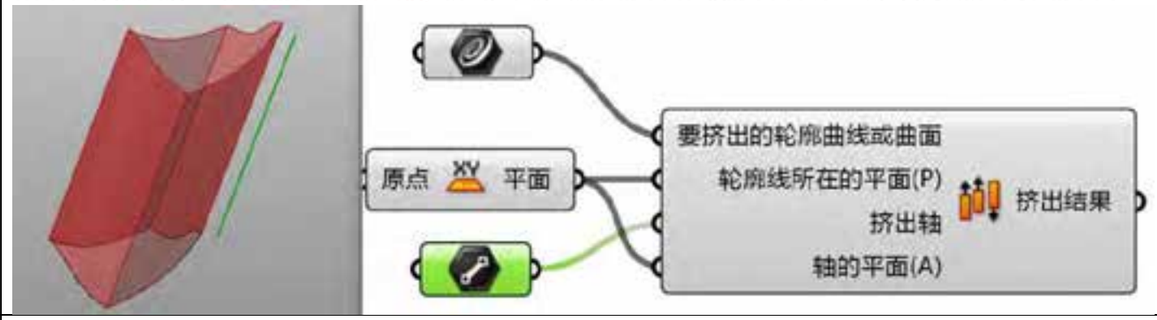
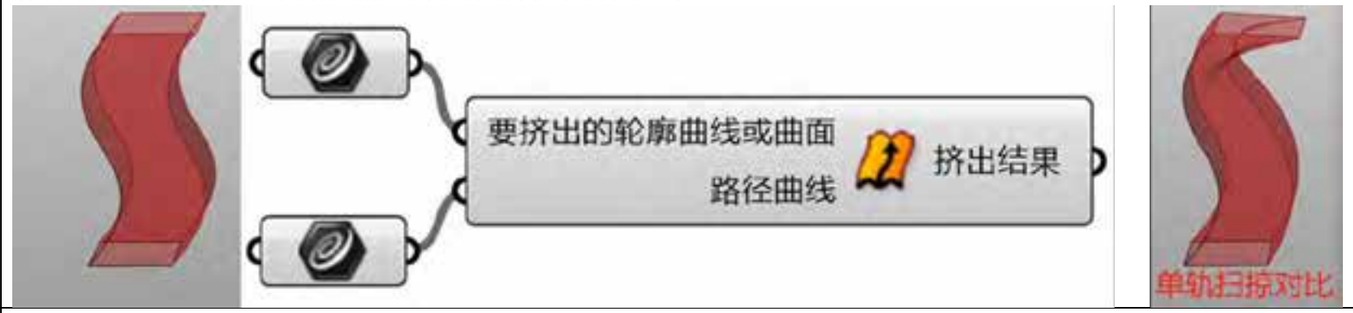
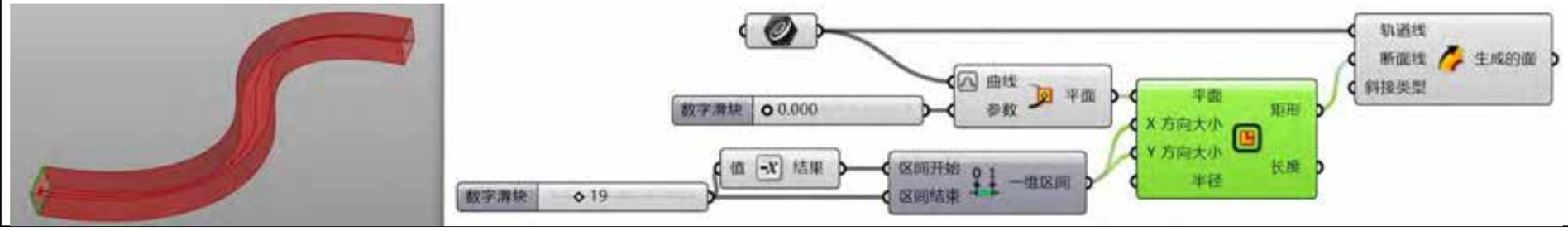
放样生面（控制点）

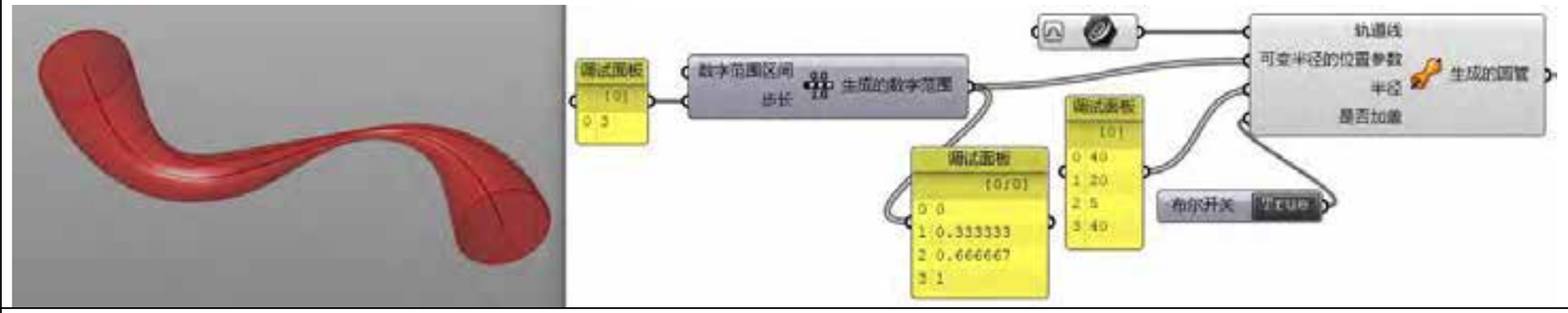
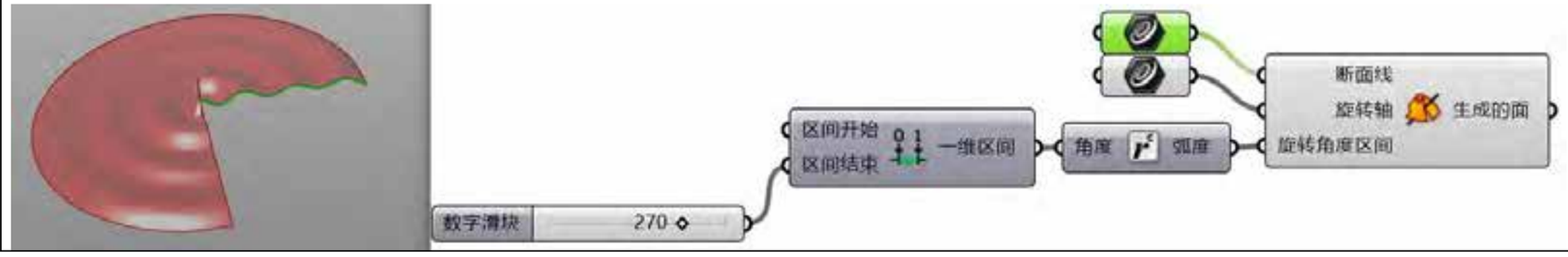
Control Point Loft

通过曲线放样生成的曲面，曲面的控制点与曲线的控制点一致。以下两种放样生成曲面的控制点可以看出区别。



直纹面生面 **Ruled Surface** 	**通过两条曲线放样生成曲面。（也可以接入多个曲线）** 
网格生面 **Network Surface** 	**以网线建立曲面。Contimuity 端本身是控制曲面连续性的，手工命令当中使用网线建立曲面的时候如果选择曲面边缘为成面曲线线，就可以激活,从而控制生成曲面和旁边曲面的连续性。（暂时没有测试成功，待研究）** 
边界生面 **Boundary Surfaces** 	**根据单个或多个平面曲线生成曲面，非平面曲线无法生成面。** 
挤出 **Extrude** 	**按照指定向量挤出线或者曲面，线挤出成面，曲面挤出成实体。**
挤出到点 **Extrude Point** 	**挤出曲线或曲面到点，形成椎体**

线性挤出 **Extrude Linear** 	**将曲线或者曲面沿直线方向的向量（挤出轴）挤出，轮廓线所在的平面和轴的平面最好一致，否则会挤出奇怪的东西** 
路径挤出 **Extrude Along** 	**沿路径挤出曲线或者曲面。下图右边的是单轨扫掠结果，可以看出区别。** 
多段线生面 **Fragment Patch** 	**根据多段线生成多重曲面。**
嵌面 **Patch** 	**根据曲线生成曲面，拖动控制点可以控制曲面局部的起伏变化（一般不需要增加控制点）。**
单轨扫掠 **Sweep1** 	**单轨放样。注意封闭轨道线单轨的时候需要满足两个条件，第一，断面线在曲线起点，第二，轨道线必须和断面线相交。** 

双轨扫掠 **Sweep2** 	**双轨放样。注意轨道线方向要一致，以及断面线是否在曲线的起点位置**
可变圆管 **Pipe Variable** 	**在不同轨道位置处，按照设定的半径大小生成一个大小变化的圆管** 
圆管 **Pipe** 	**曲线生成管道。是否加盖有三种参数：0 不加盖，1 平盖，2 圆盖。**
旋转扫掠生面 **Rail Revolution** 	**根据轨迹旋转放样。**
旋转放样生面 **Revolution** 	**旋转放样,可以控制旋转放样的角度。** 

网格 Mesh

三角关系 Triangulation（泰森多边形，德劳内网格等）

网格 Mesh

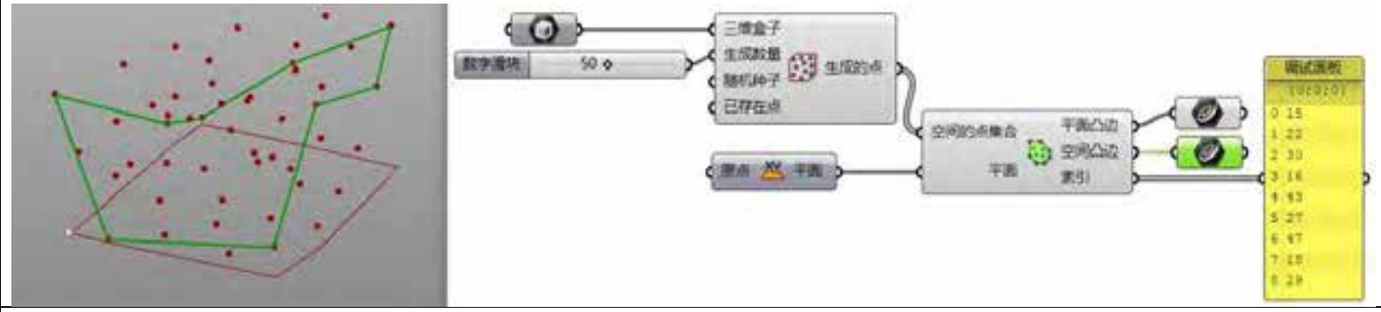
三角关系 Triangulation

— 网格

凸边

Convex Hull

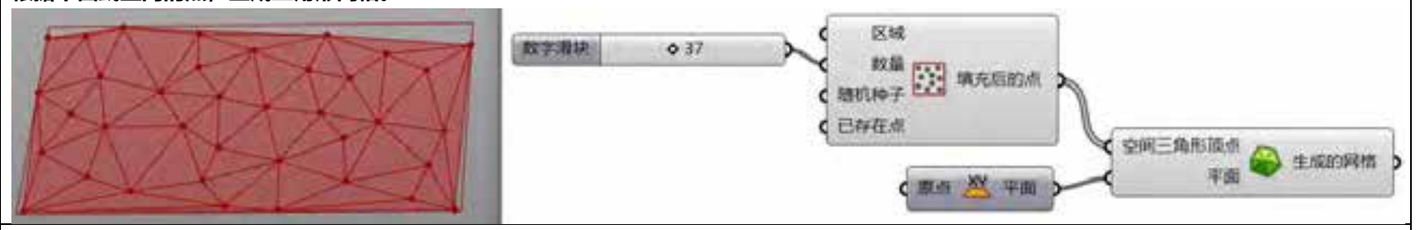
根据指定点在平面上投影后最外围的点，连成多段线，相应的投影点也连成多段线，并输出最外围点的序号。



德劳内网格

Delaunay Mesh

根据平面或空间的点，生成三角形网格。



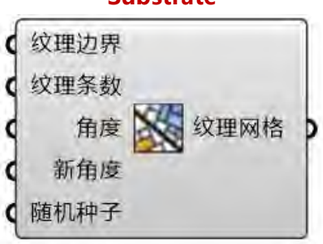
德劳内边

Delaunay Edges

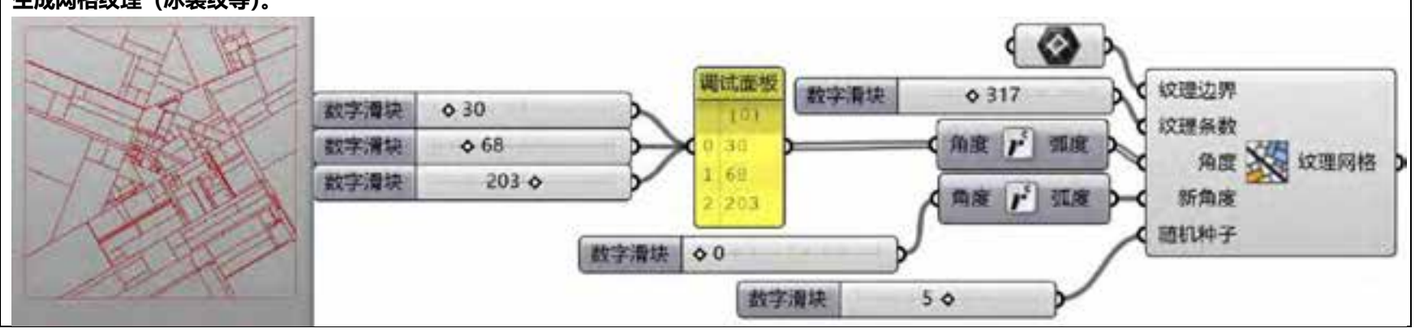
将选定的点（二维或三维的点）与附近最近的点相连，生成三角形连线。连接数为与每个点连接的其他各点的序号。

网格纹理

Substrate



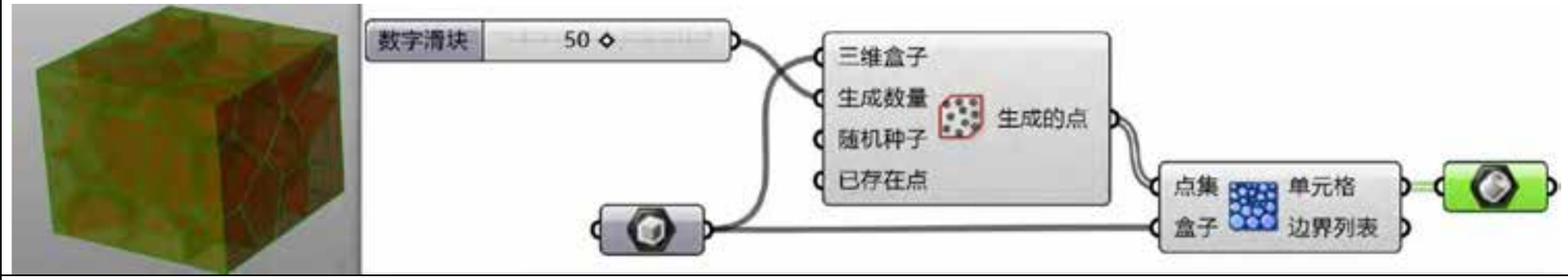
生成网格纹理（冰裂纹等）。



三维泰森多边形

Voronoi 3D

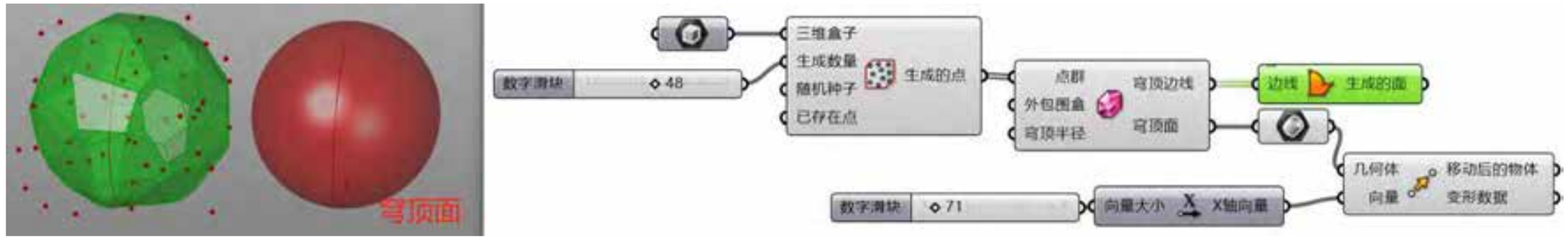
根据三维点集生成三维泰森多边形。



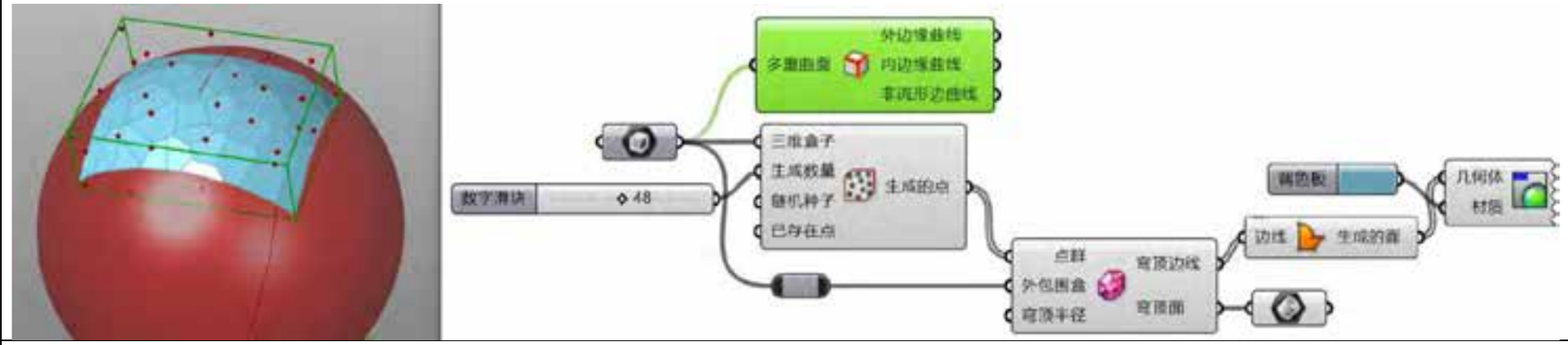
平面穹顶（泰森多边形）

Facet Dome

根据三维点生成一个泰森多边形，穹顶面为包含在泰森多边形内的球体。



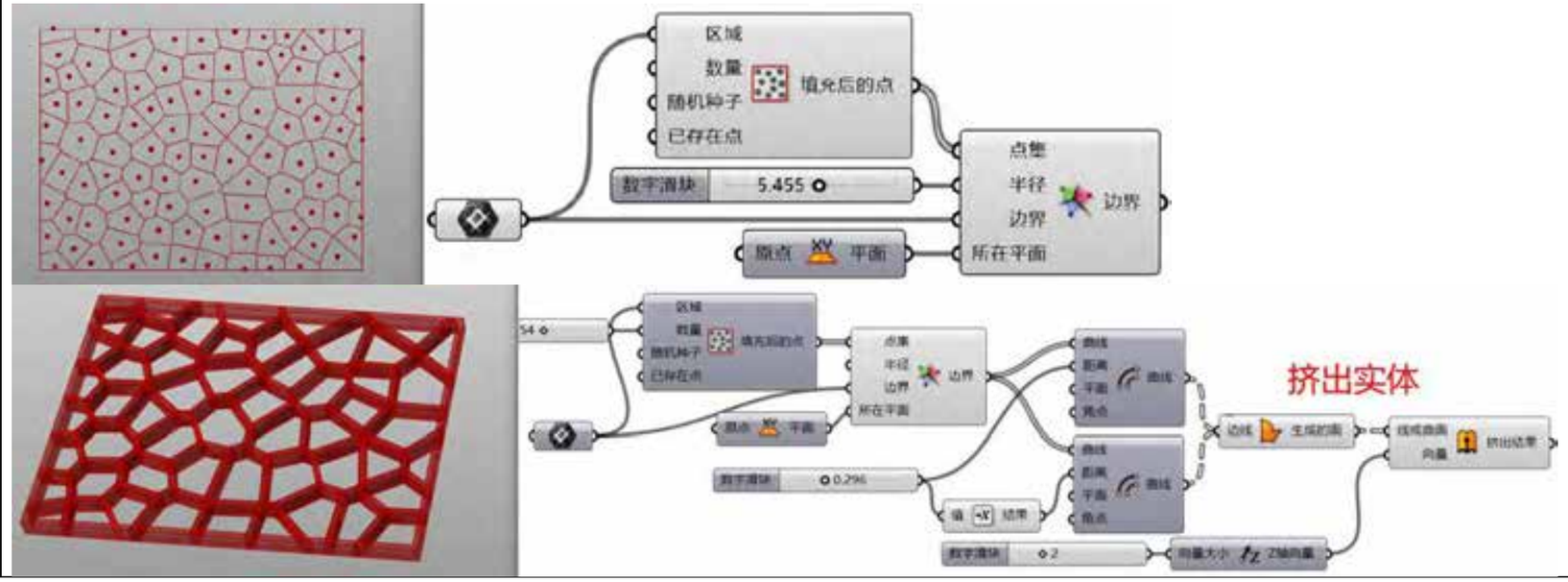
生成平板泰森多边形曲面的方法如下。



泰森多边形

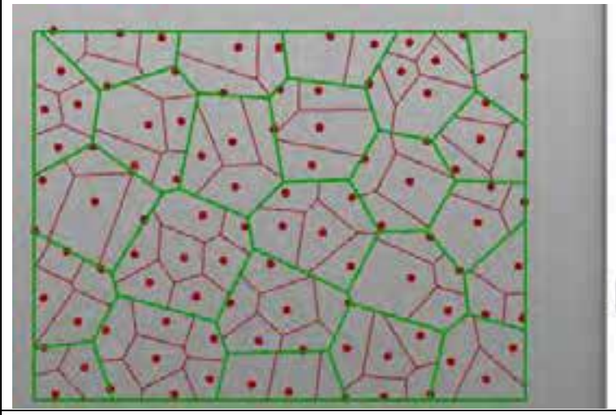
Voronoi

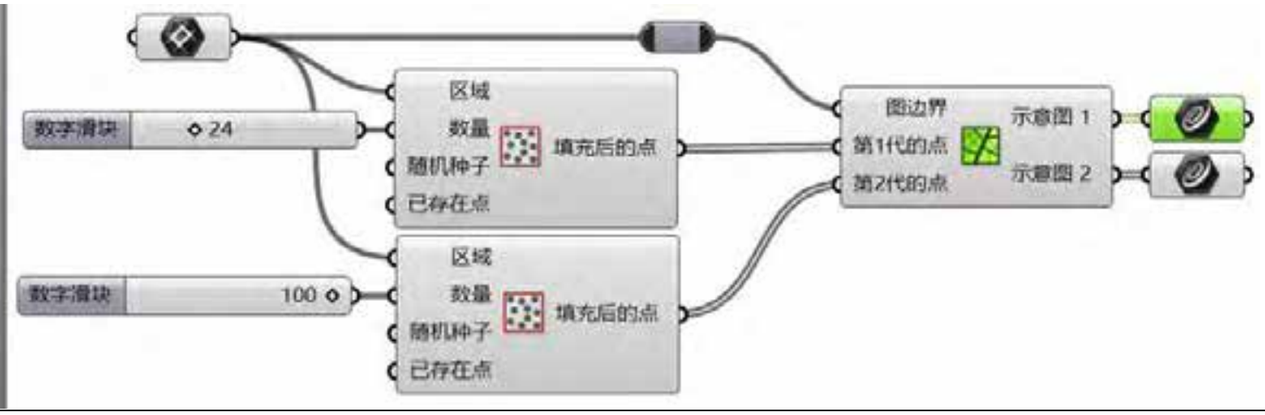
根据平面点集生成泰森多边形。



泰森多边形嵌套分组
Voronoi Groups

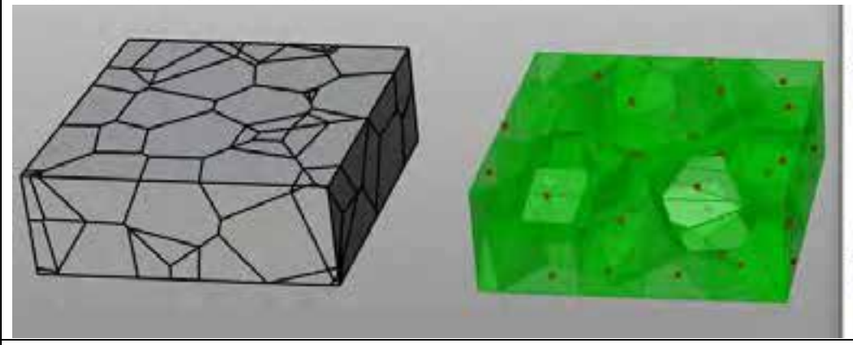
根据两组点集，生成嵌套的泰森多边形。示意图 1 对应第 1 代的点的网格，示意图 2 对应第 2 代的点的网格。





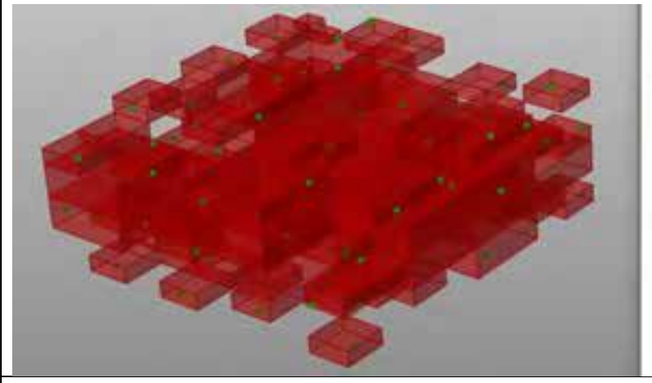
泰森多边形边界
Voronoi Cell

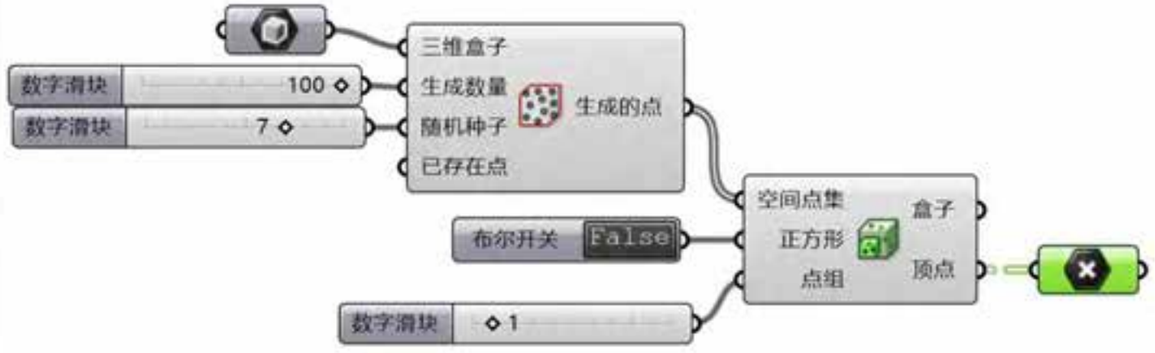
根据点集生成八叉树结构：根据第一组点为基点，在第二组点中寻找最近点，生成泰森多面体，以细胞状结合在一起。



八叉树网格
OcTree

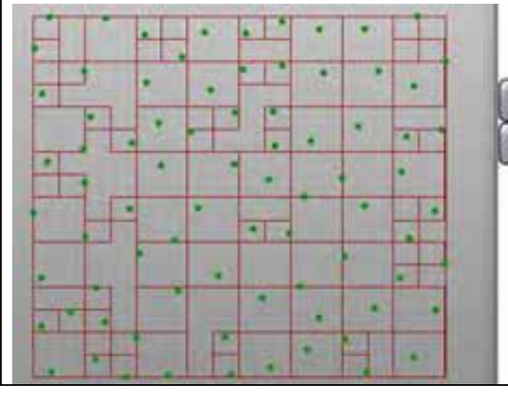
在三维点集中，按照设定的点组数量，将最近的点分组生成相交的矩形盒子（也可以通过布尔开关控制生成正方形盒子）。

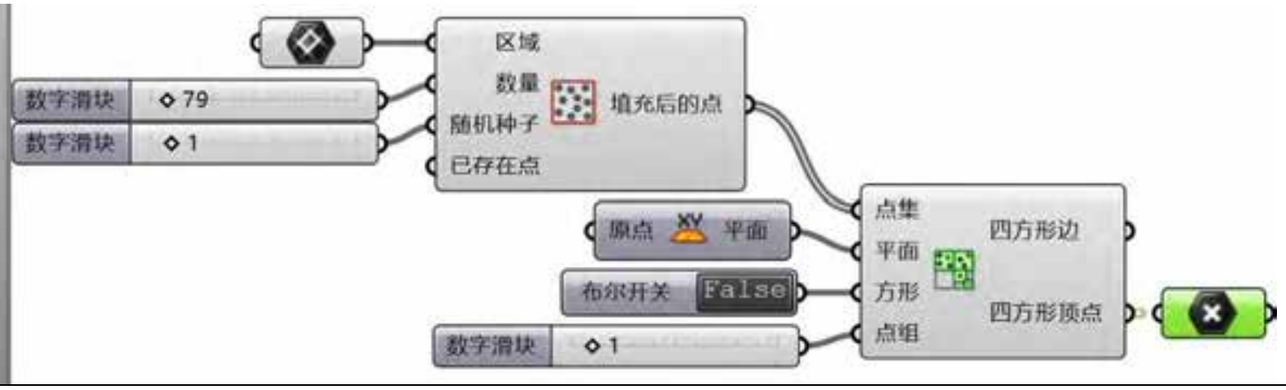




四叉树网格
QuadTree

按照设定的点组数量，将平面点中最近的点分组生成相交的矩形（也可以通过布尔开关控制生成正方形）。

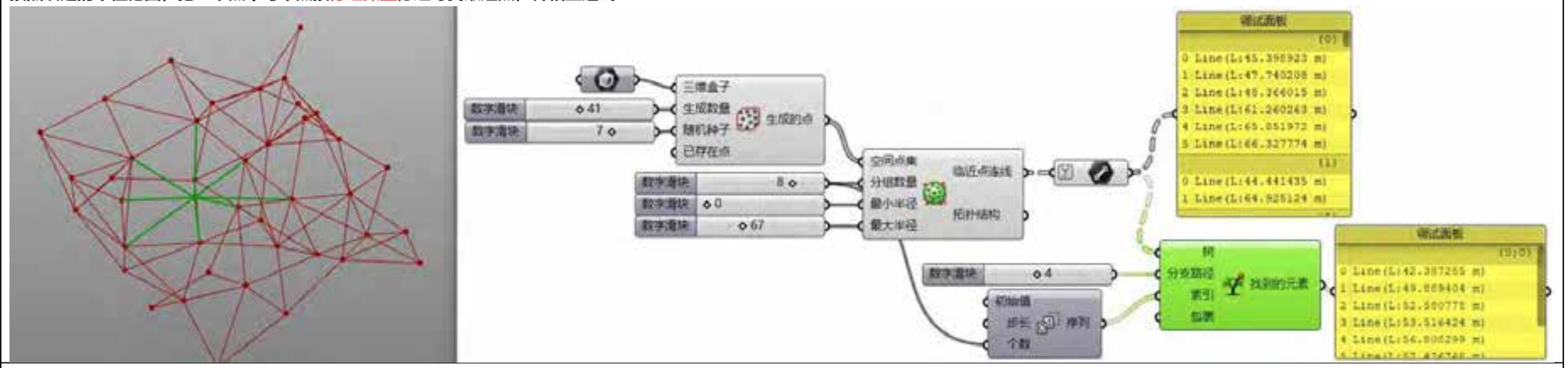




网格最近点三维分组

Proximity 3D

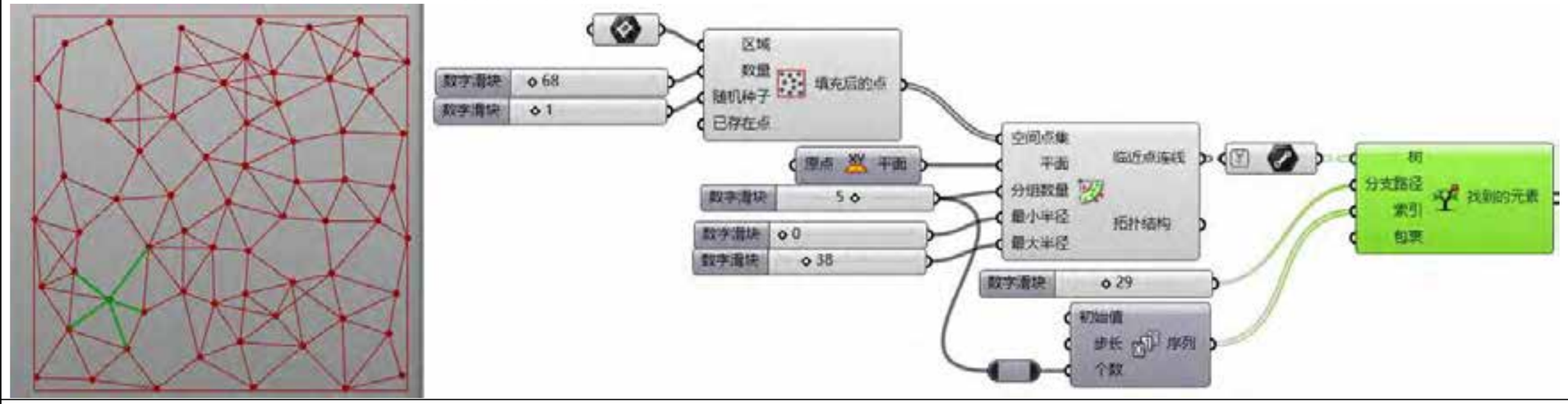
按照设定的半径范围，把三维点中每个点按分组数量分组寻找最近点，并相互连线



网格最近点二维分组

Proximity 2D

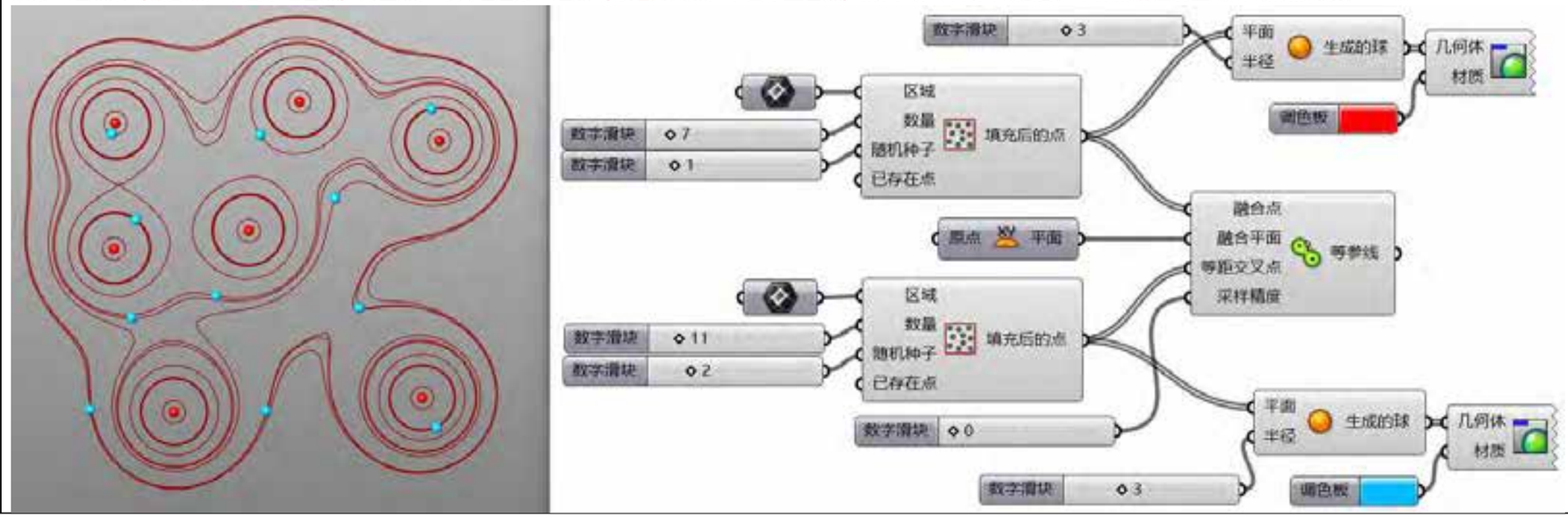
按照设定的半径范围，把平面点中每个点按分组数量分组寻找最近点，并相互连线

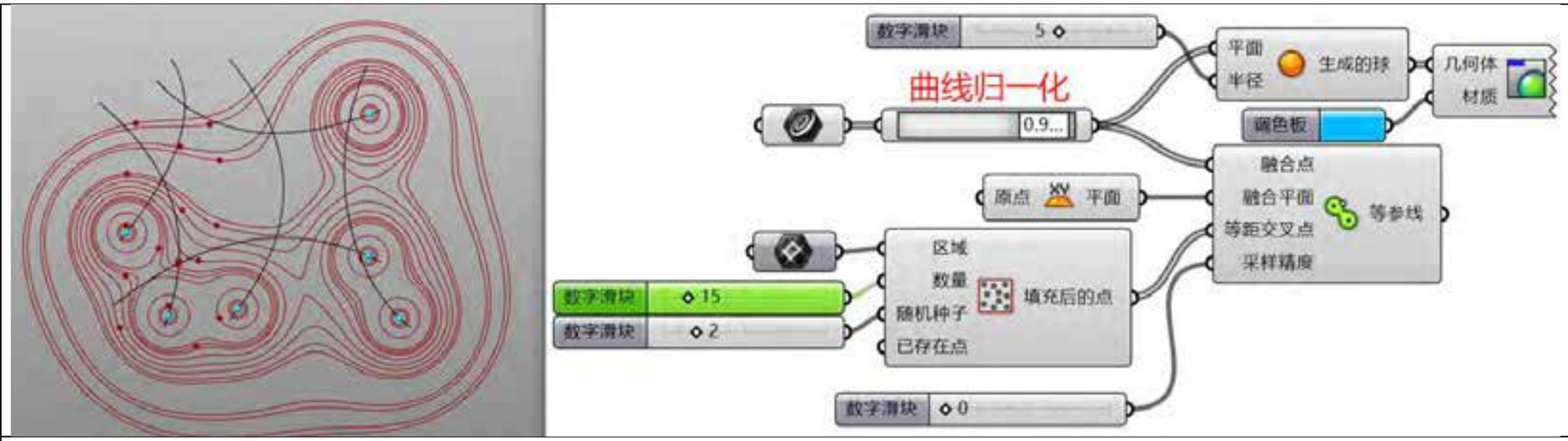


融合球

MetaBall

以第一组点集（融合点，下图中红色点）为基点，寻找经过第二组点（等距交叉点，下图中蓝色点）的曲线，并将曲线融合。采样精度控制生成曲线的精确度。

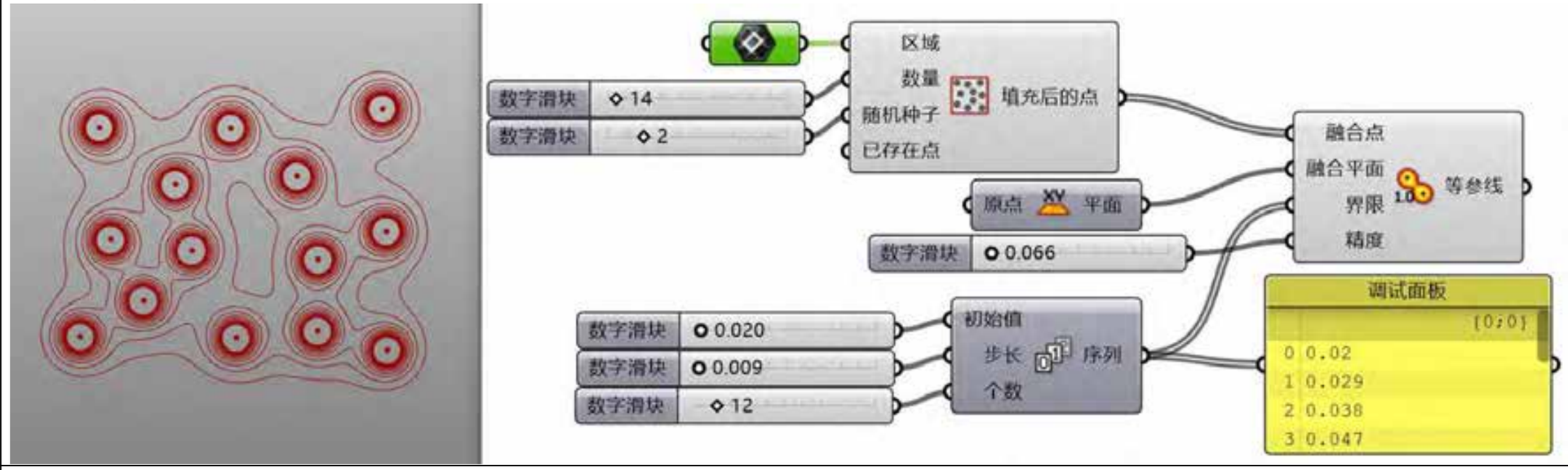




融合球（t）

MetaBall(t)

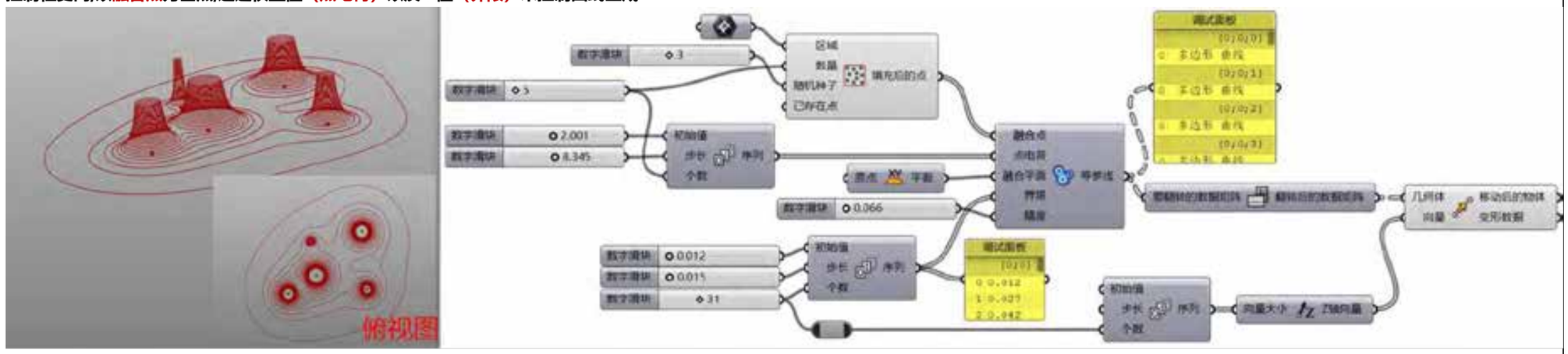
根据 t 值（界限）确定曲线的生成，t 值越小，生成的曲线范围越大。并且这里的曲线生成并不是等间距的,而是逐渐缩小的间距的,如果想做成类似等间距的线就得将 t 值进行处理。



自定义融合球（t）

MetaBall(t) Custom

控制性更高,以融合点为基点,通过权重值（点电荷）以及 t 值（界限）来控制曲线生成



四边形重构网格

Quad Remesh

四边形网格重构设置

Quad Remesh Setting

以四边形网格重构物体表面网格。

目标数量：生成的网格数量。

自适应的大小：控制网格四边形大小根据曲率变化。

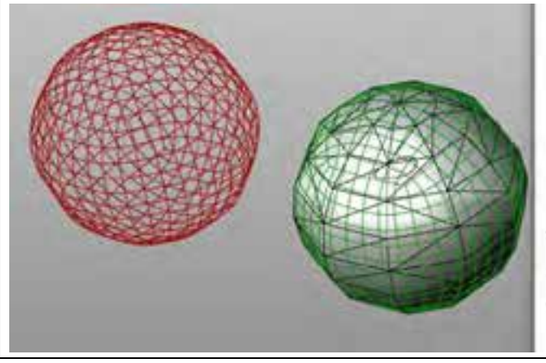
自适应数量：如果高曲率区域的四边形数量允许更多，则为 true.

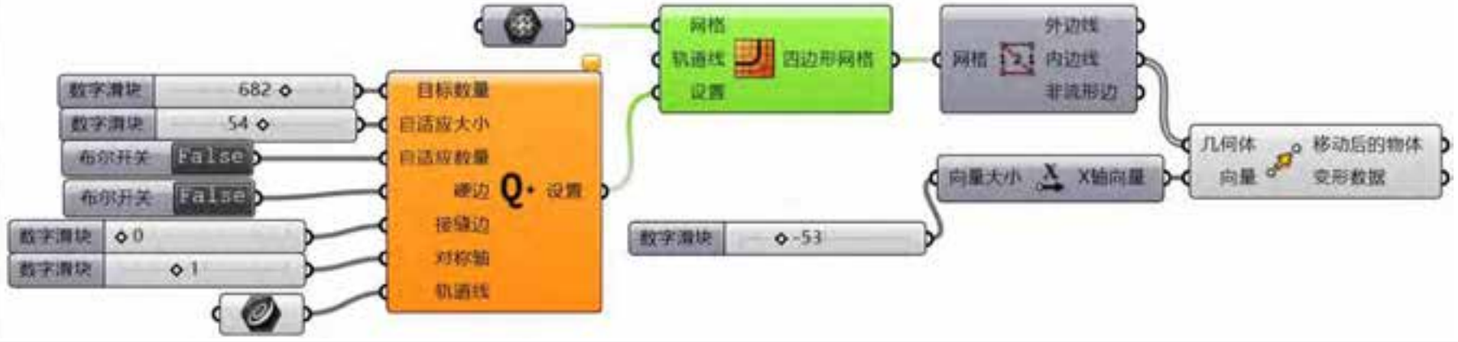
硬边：在输入的网格中检测并保留硬边。

接缝边有三种：0 关闭，1 智能，2 严格。检测并保留多重曲面的边缘。

对称轴有四种：0-无，1-X 轴，2-Y 轴，3-Z 轴。

轨道线可右键选择：0 近似值，1 环形边，2 循环边。轨道线可以用来干扰生成网格的走势。

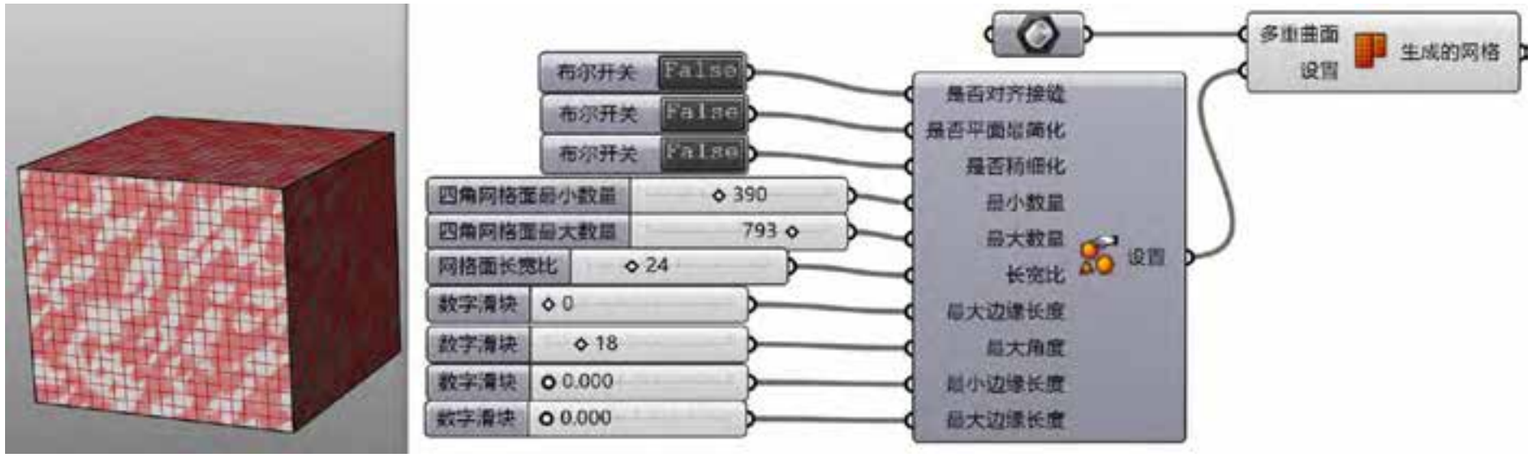




公用 Util - 网格

多重曲面生网格
Mesh Brep

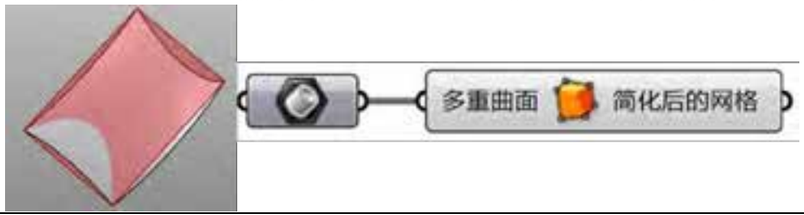
根据多重曲面生成网格。不连接设置，也会自动生成一个默认参数的网格。



设置（自定义）
Settings (Custom)

简化网格
Simple Mesh

根据多重曲面创建简化后的网格，其实就是提取曲面的四个交点直接生成网格。



网格化之曲面 UV
Mesh Surface

创建一个将曲面 UV 分割后的网格

设置（质量优先）
Settings（quality）

设置（速度优先）
Settings（Speed）

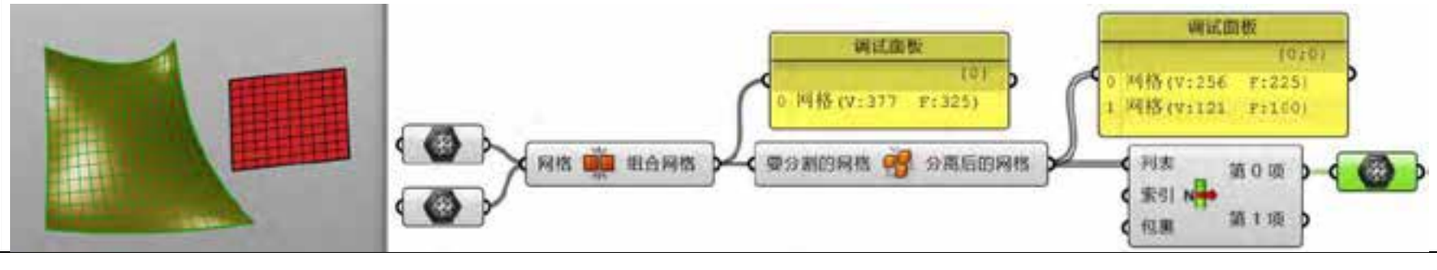
快速转化网格的两种设置。

网格组合
Mesh Join

分离网格
Disjoint Mesh

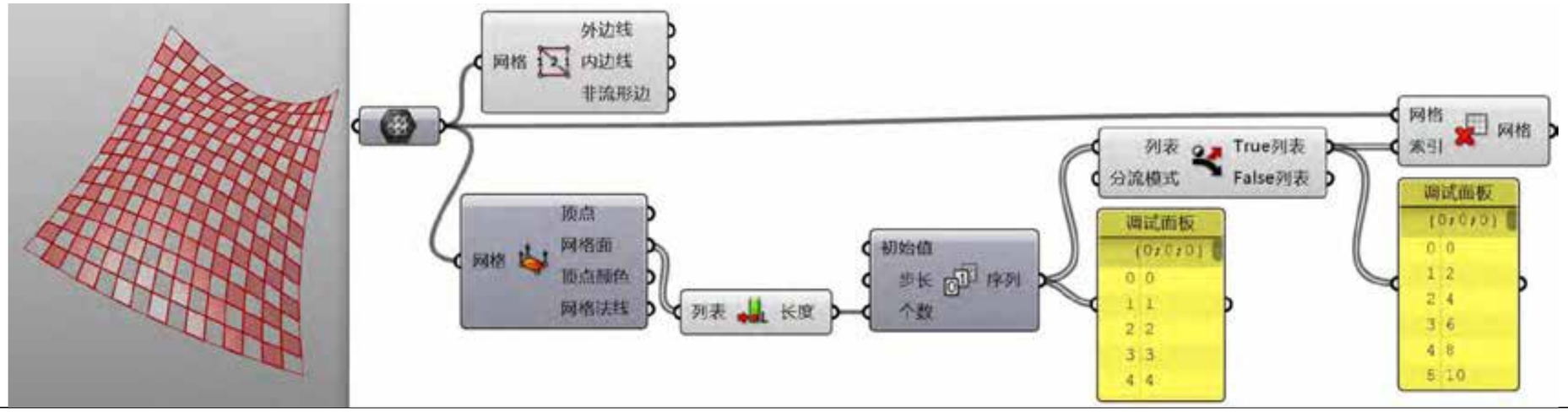
网格组合：将两个网格（无论是否相接）合并成一个网格，可以提升显示速度。

分离网格：如果组合后的网格相接的话，则无法分解，如果是彼此分离的部分组合而成的网格，则可以分解成单独部分的网格。



删除网格面
Delete Faces

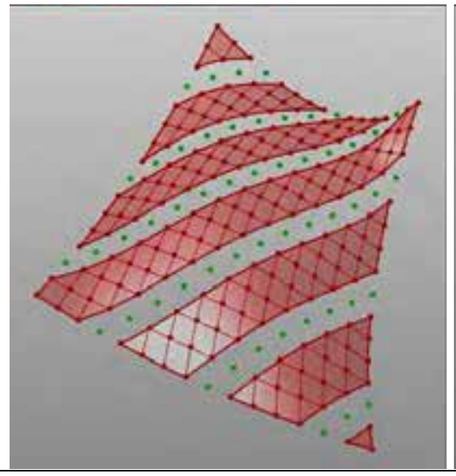
删除网格中指定序号的网格面。



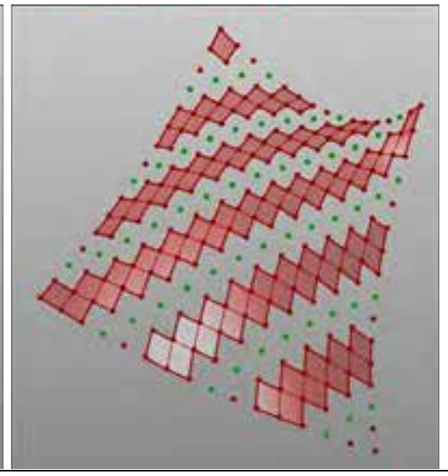
删除网格顶点
Delete Vertices

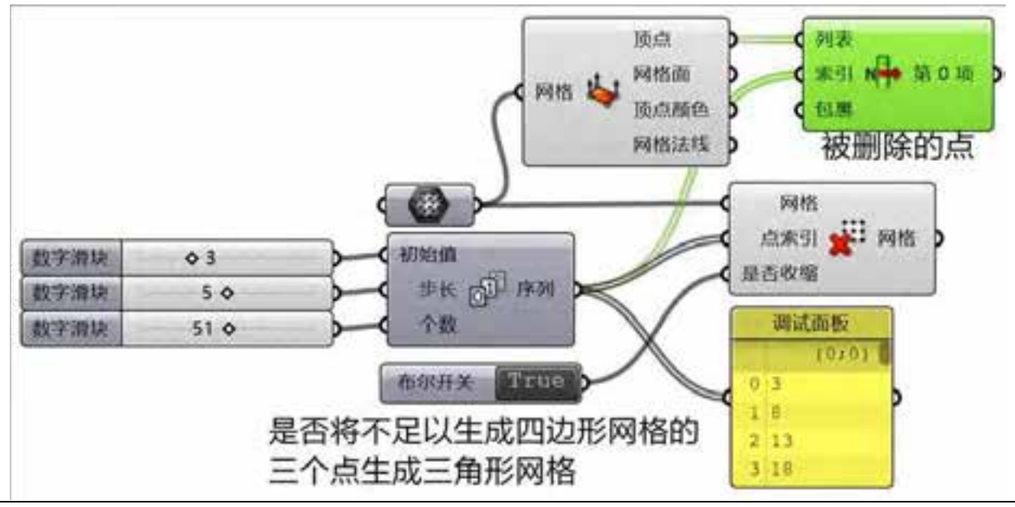
删除网格中指定序号的网格点。

是否收缩为true时



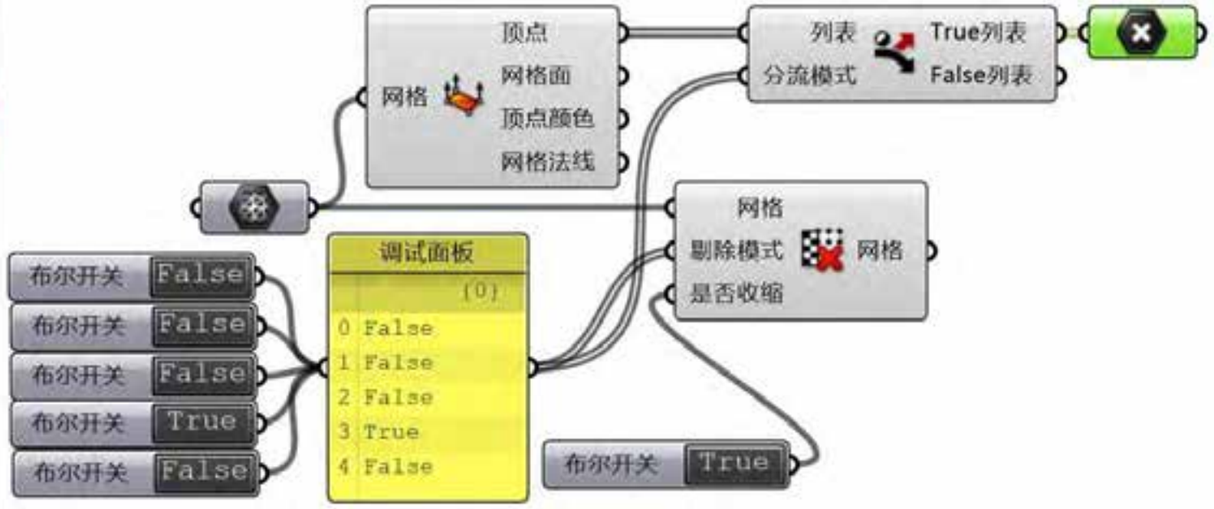
是否收缩为false时





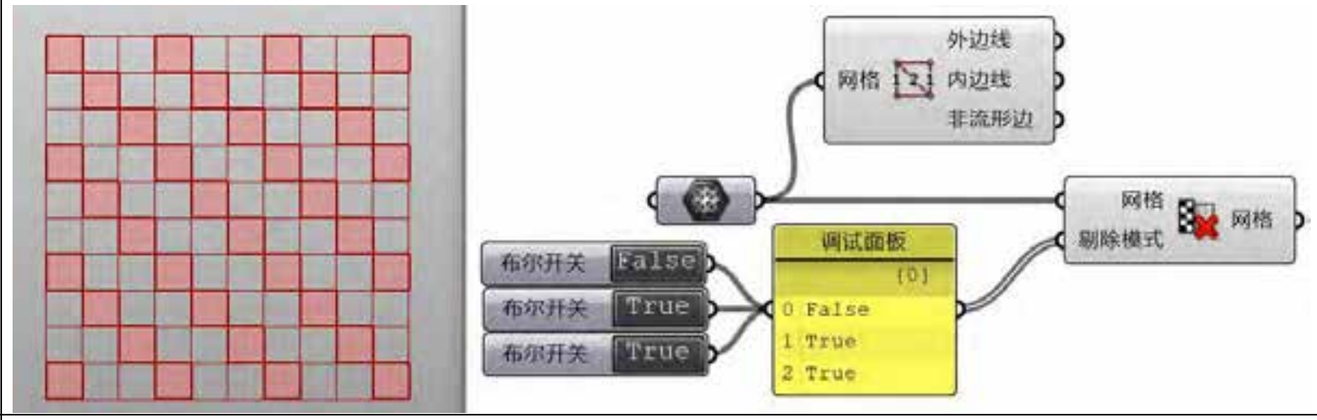
剔除网格点
Cull Vertices

根据设定的剔除模式删除网格点，true 为删除点，false 为保留点。



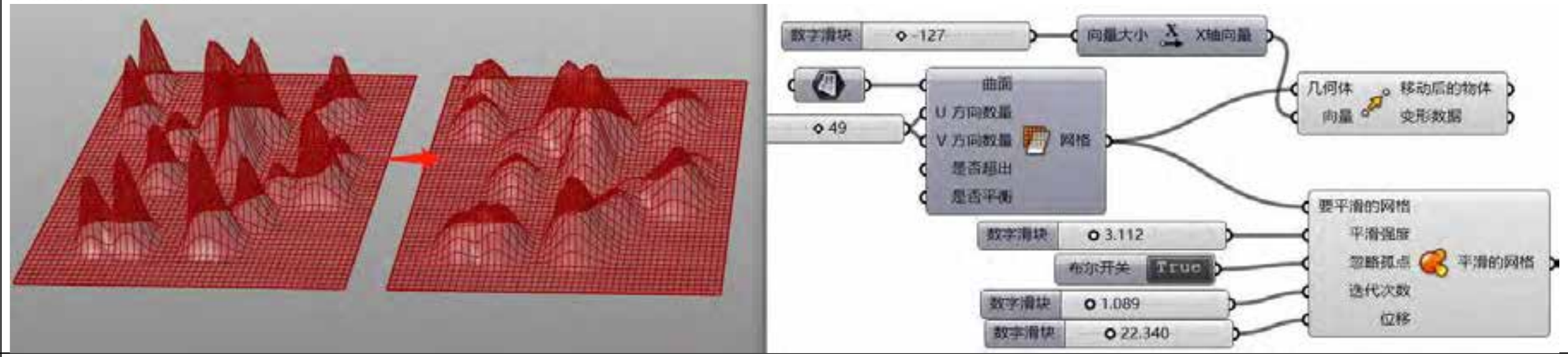
剔除网格面
Cull Faces

根据设定的剔除模式删除网格面。



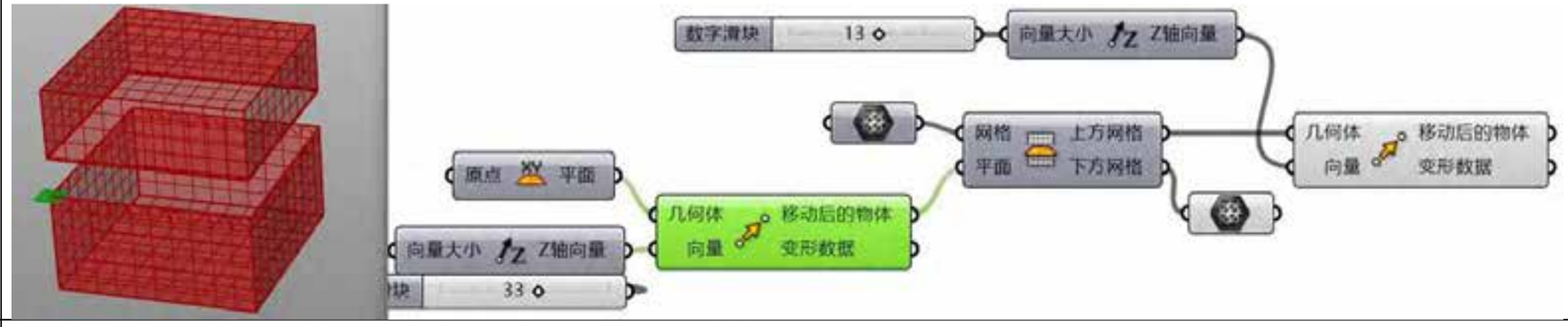
平滑网格
Smooth Mesh

平滑网格，降低网格起伏变化趋势。



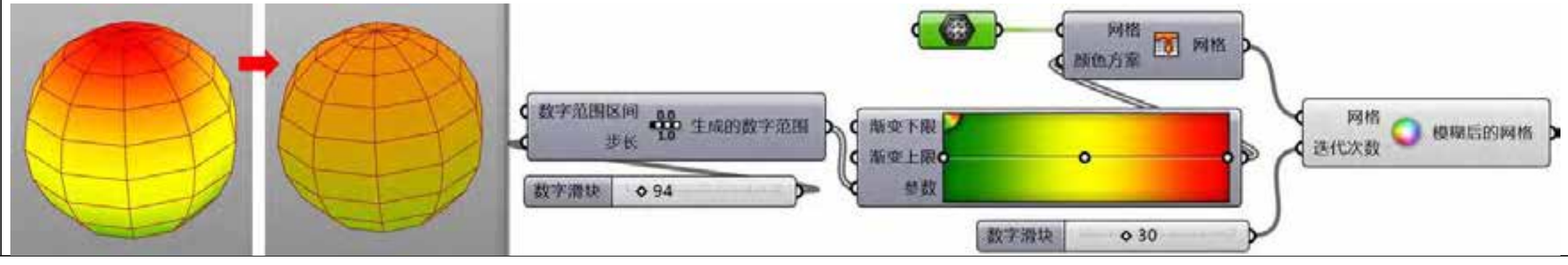
平面分割网格
Mesh Split Plane

根据指定平面，将网格一分为二。



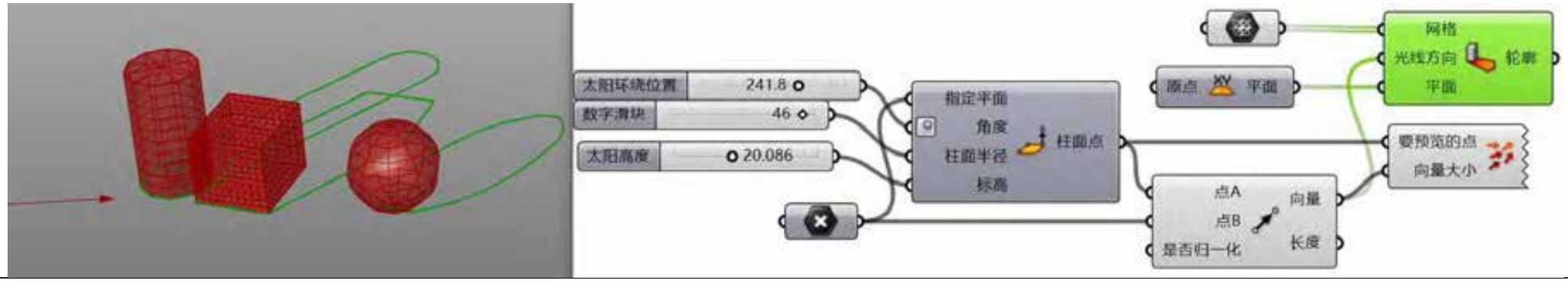
模糊网格颜色
Blur Mesh

对网格颜色进行模糊化处理，迭代次数是指模糊化的次数。可以让颜色之间过渡得更好。



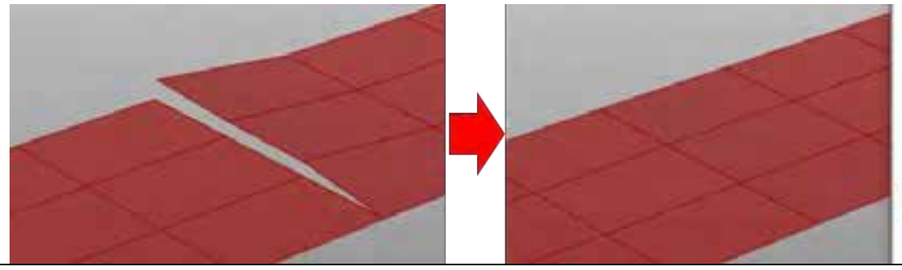
网格阴影
Mesh Shadow

计算网格在指定平面上的投影。这个运算器不能得到投射在物体上的阴影。



对齐顶点
Align Vertices

以指定公差对齐网格顶点。



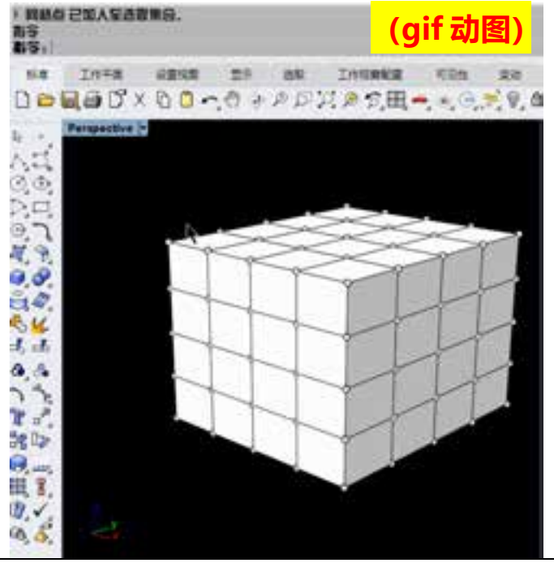
焊接网格
Weld Mesh

还原网格焊接
Unweld Mesh

焊接网格：网格边缘夹角小于设定角度的焊接在一起。

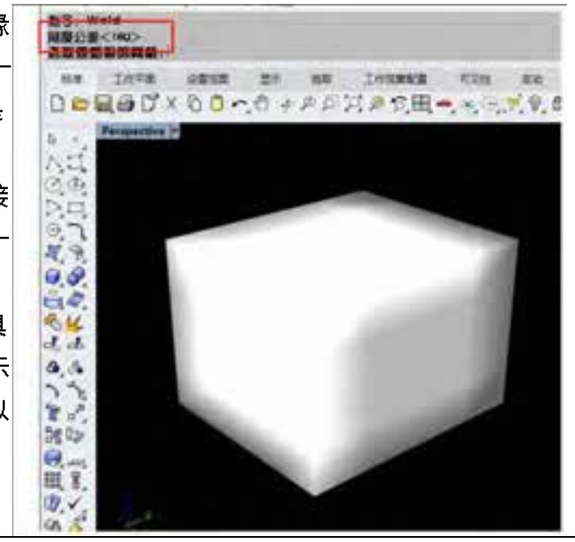
还原网格焊接：网格边缘夹角大于设定角度的取消焊接。

焊接，一来减少了网格的控制点，毕竟多个网格会共用相同的网格顶点。二来是有关显示的，显示模式中把网格的边缘显示设成 0，就可以让这些网格看起来和一般的 nurbs 物件毫无差异



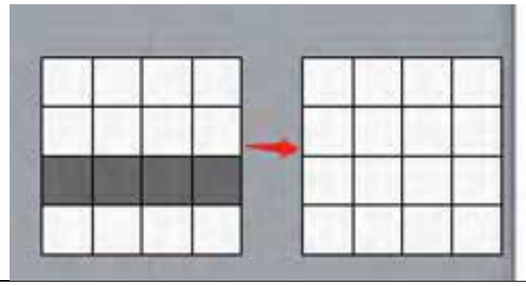
我们可以通过焊接来控制边缘是否显示,比如我给box焊接一下，角度设置为180，也就是所有网格夹角边缘小于180的都要被焊接，基本上就是焊接所有边缘，然后你就会得到一个很奇怪的box（右图）：

这是因为焊接后的顶点会具有材质过渡的属性，网格显示就会有这样的渐变效果，所以如果你看到这样的效果不爽，直接还原焊接就行。



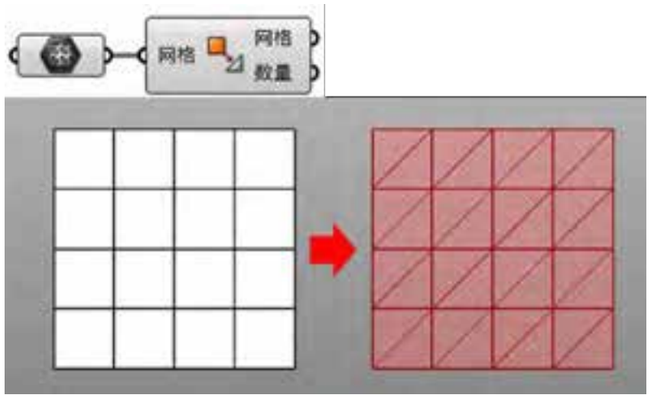
统一网格
Unify Mesh

统一网格的法线方向。



网格三角化
Triangulate

四边形网格转换成三角形网格。

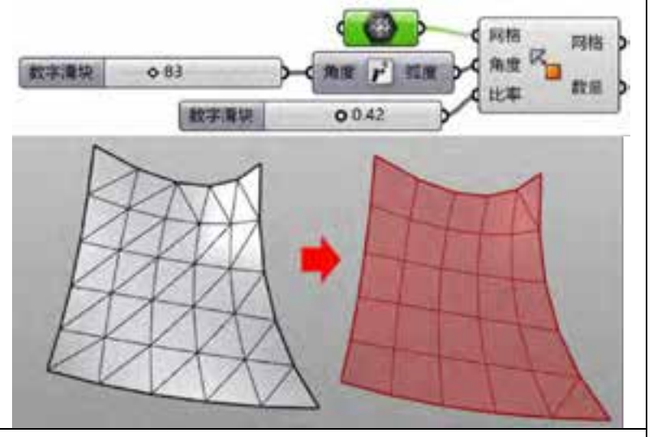


网格四边化
Quadrangulate

三角形网格转换成四边形网格。

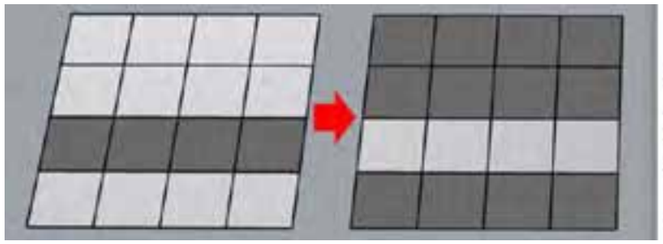
角度：超过设定角度的三角形不会合并。

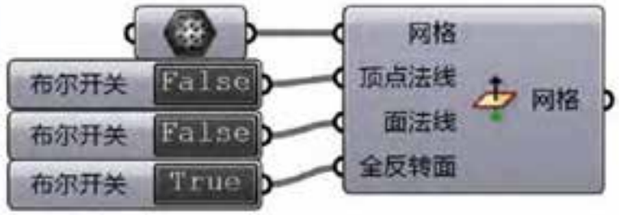
比率：如果四边形对角线的比率（最短对角线/最长对角线）超过设定比率，则不会考虑。



翻转网格
Flip Mesh

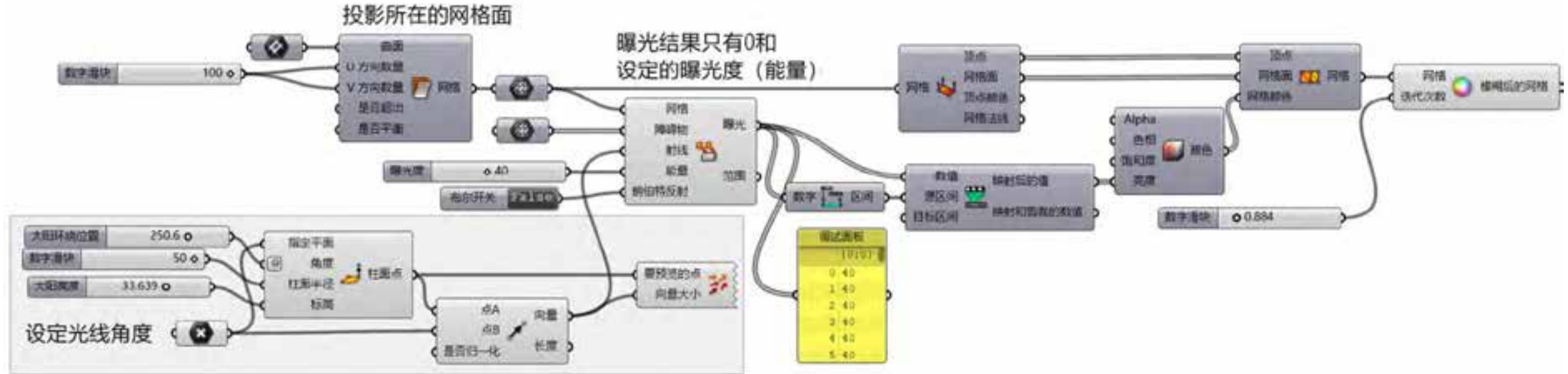
翻转网格方向。

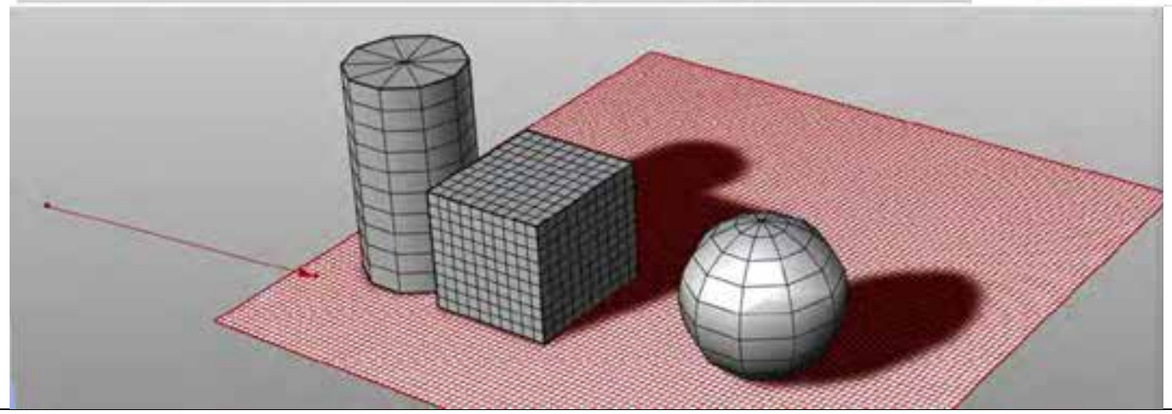




曝光
Exposure

计算光线照射下，物体在网格上的投影。**曝光**输出的是每个网格顶点的曝光值。

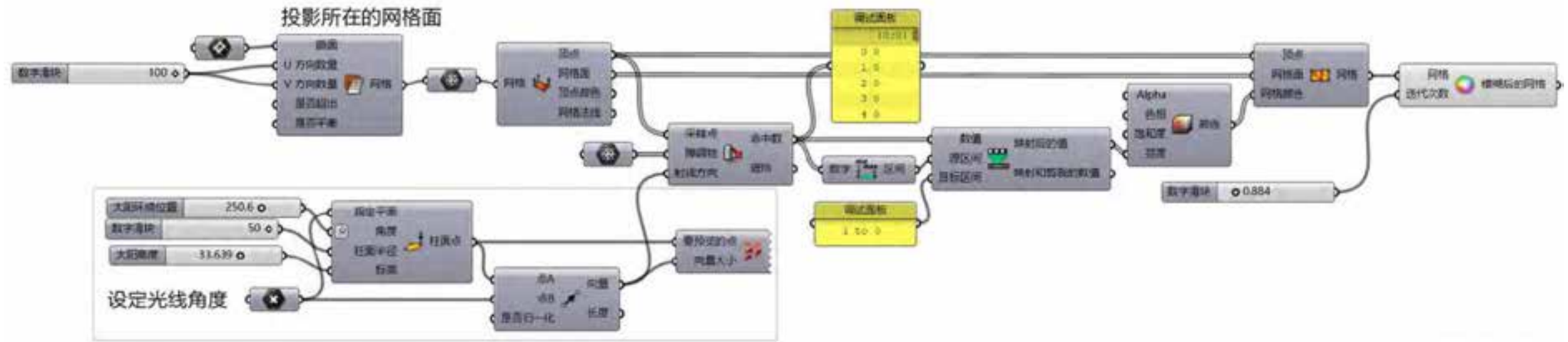


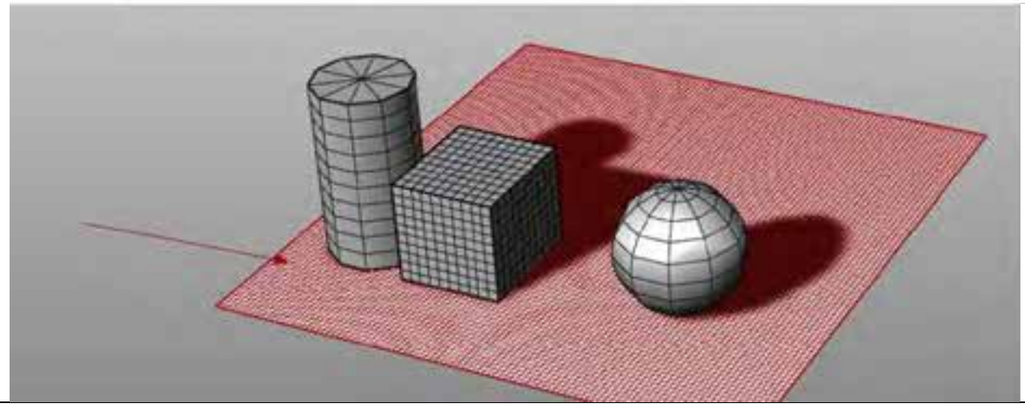


遮罩

Occlusion

同曝光运算器类似，计算光线照射下，物体在网格上的投影。命中数输出的是 0 和 1，1 代表被遮挡。





分析 Analysis

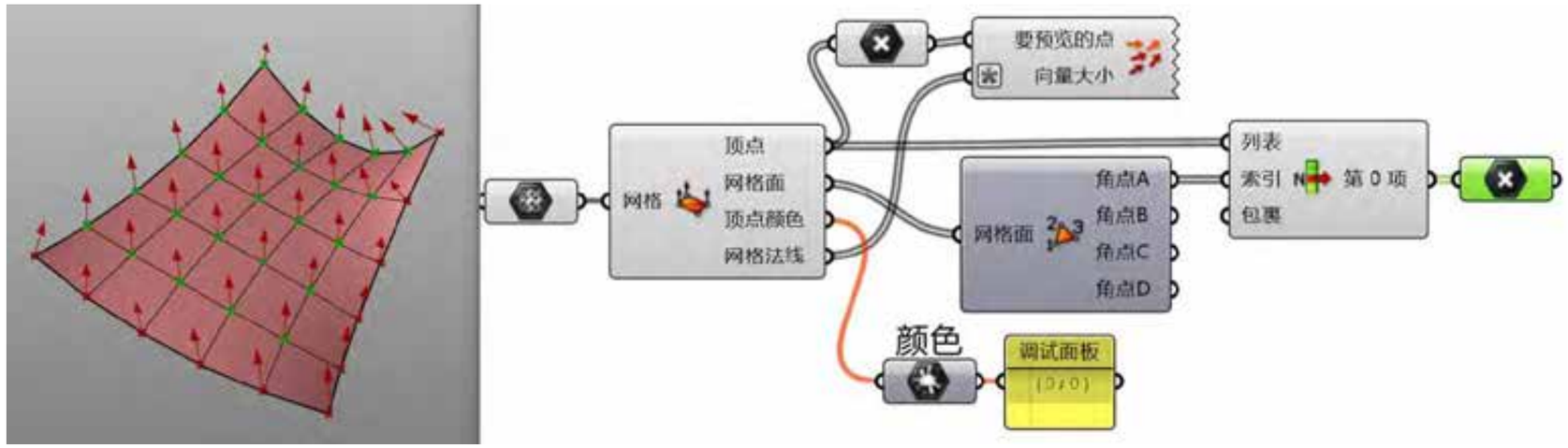
- 网格

分解网格
Deconstruct Mesh

分解网格面
Deconstruct Face

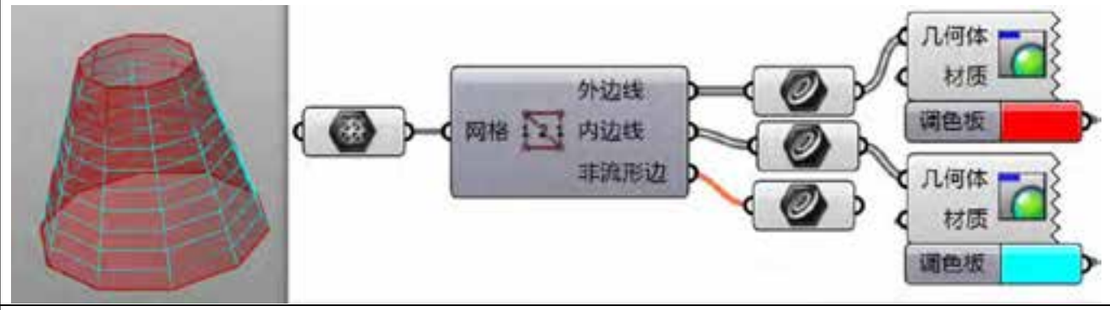
分解网格：将网格分解成点，面，颜色，以及其法线方向等基本元素。（快捷键 Ctrl + M 或者在显示菜单中，打开预览网格边缘）

分解网格面：提取网格面的四个角点的序号。



网格边缘
Mesh Edges

提取网格的各种边缘线。（非流形边缘，就是由三个或三个以上边缘相交的边缘，这种边缘是非法的会出问题，所以要尽量避免。）

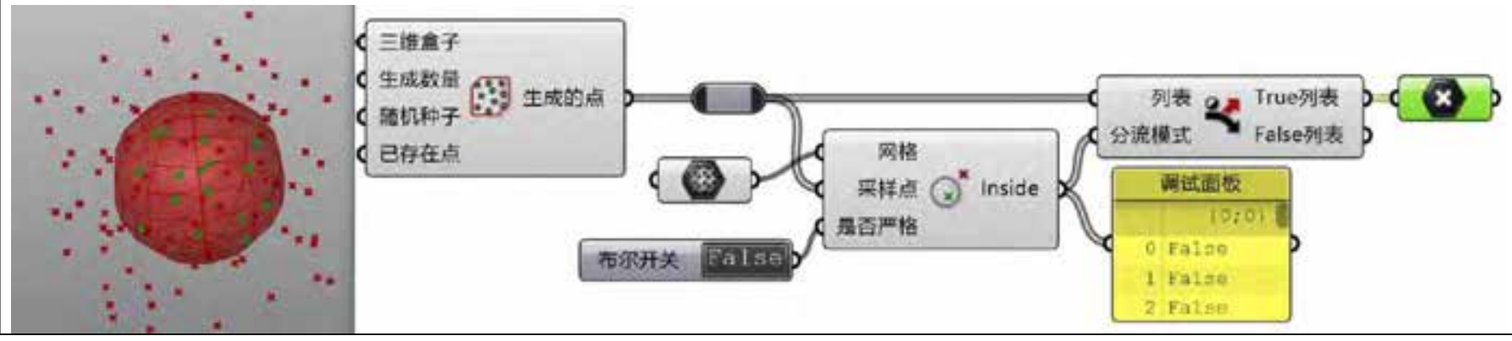


网格面法线方向
Face Normals

提取网格中每一个面的中心点以及这个中心点所在面的法线方向。

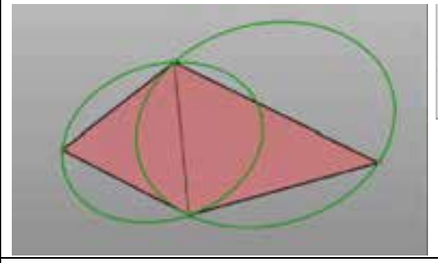
网格内部判断
Mesh Inclusion

判断点是否在网格的内部，并返回布尔值。



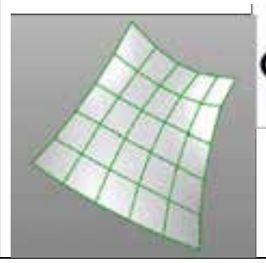
网格面外接圆
Face Circles

生成三角形网格的外切圆。



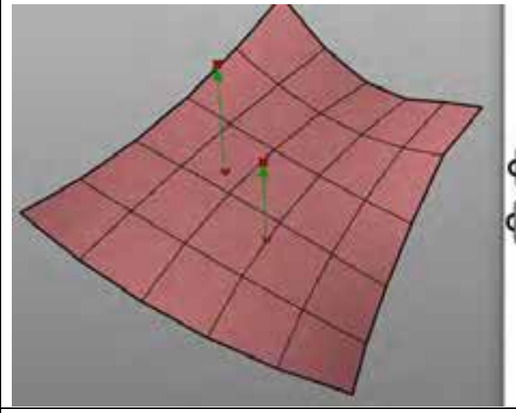
网格面边缘
Face Boundaries

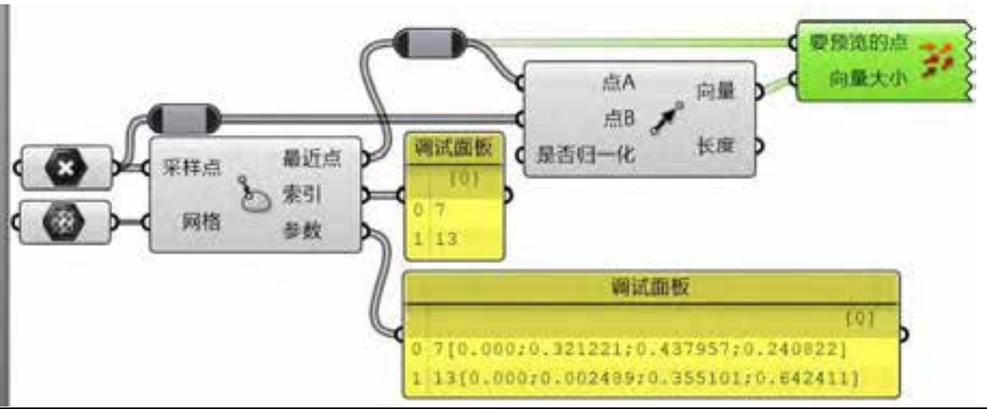
提取网格中每个网格面的边缘线。



最近点之网格
Mesh Closest Point

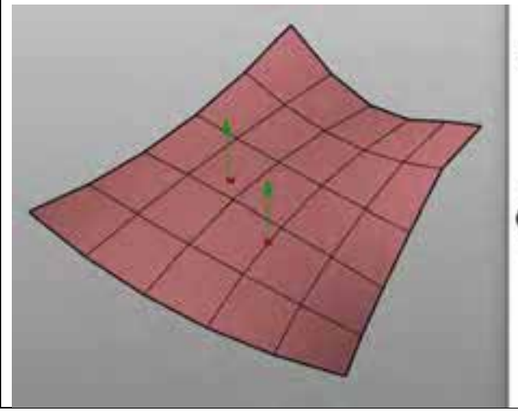
查找指定点到网格的最近点，输出最近点所在网格面的序号，及最近点在网格面的参数。

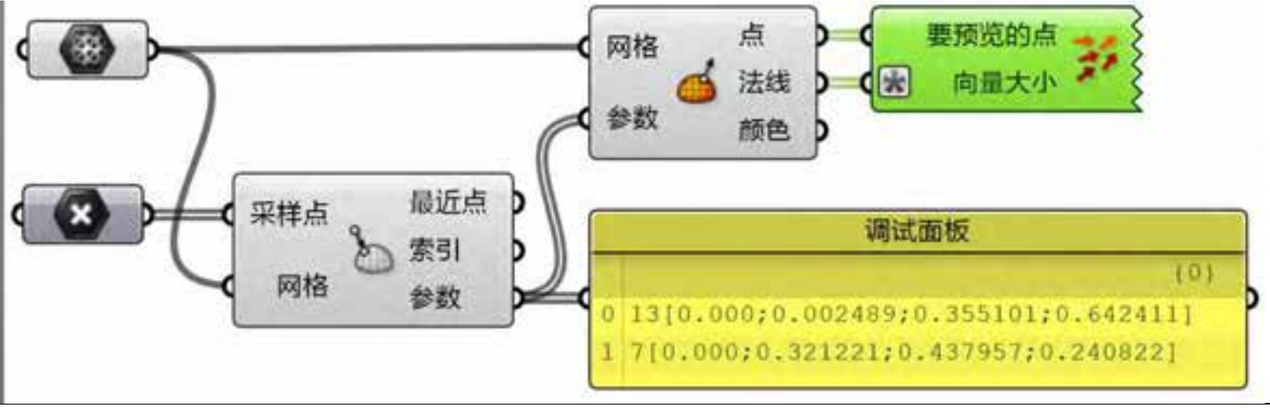




网格计算
Mesh Eval

根据指定的网格点参数，查找网格上的点，并输出该点的法线。

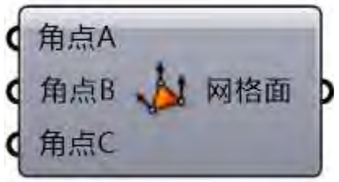




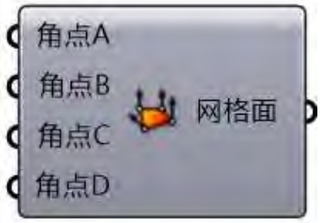
初始 Primitive

- 网格

三点生网格
Mesh Triangle



四点生网格
Mesh Quad



构造网格
Construct Mesh

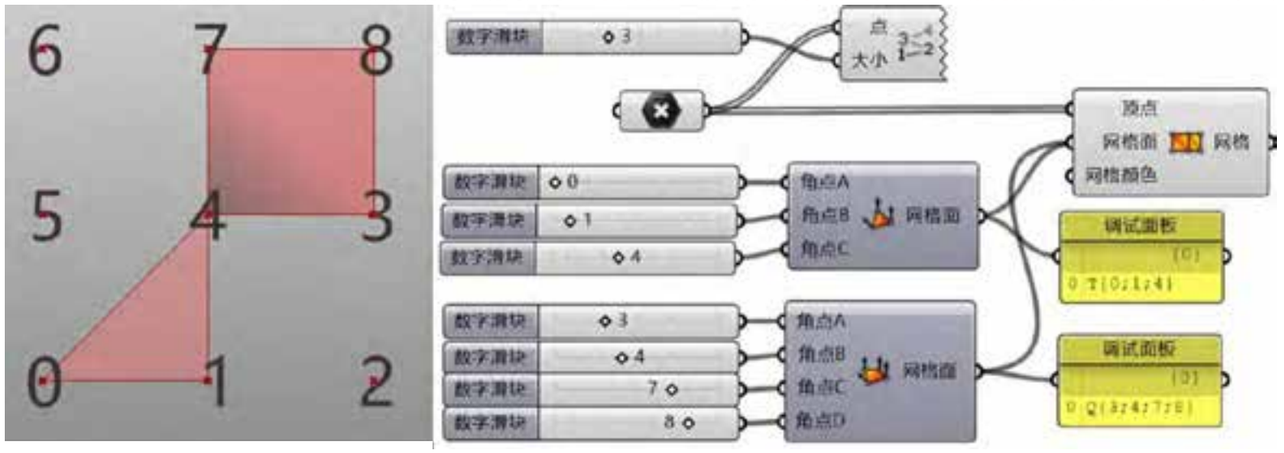
网格喷洒
Mesh Spray

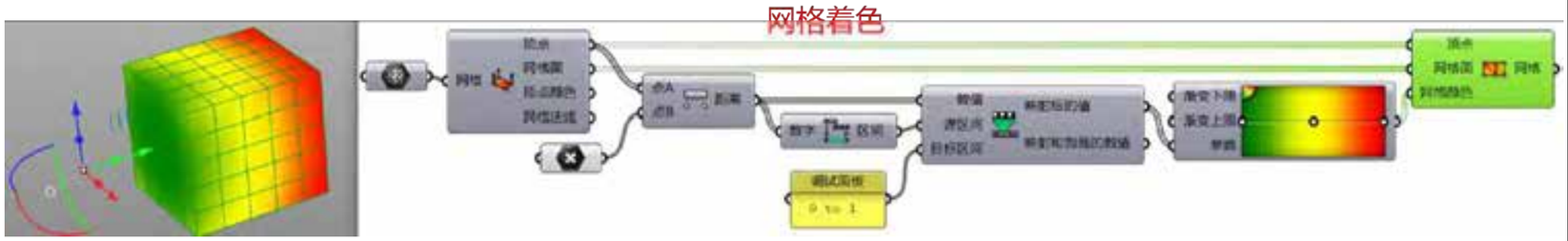
网格颜色
Mesh Colours

通过对点集进行编排序号，然后根据序号选择其中的三个点或四个点编成一组，通过构造网格生成网格。

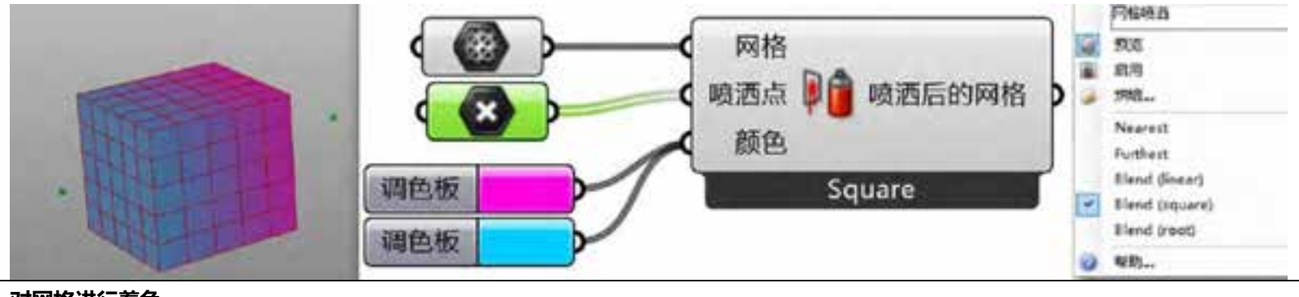
三点生网格/四点生网格：将指定点的序号编成一组。比如如果是 Q{1;2;3;4}意思就是依次将 1234 号点连接成面。不过需要注意的是，这个点序和网格正反是有关系的，顺时针为负，逆时针为正。

构造网格：指定的网格点数按照网格面成面规则（规则分为三角形网格（格式为 T{0；1；2}）和四边形网格（格式为 Q{0；1；2；3}）），连接网格点生成网格。

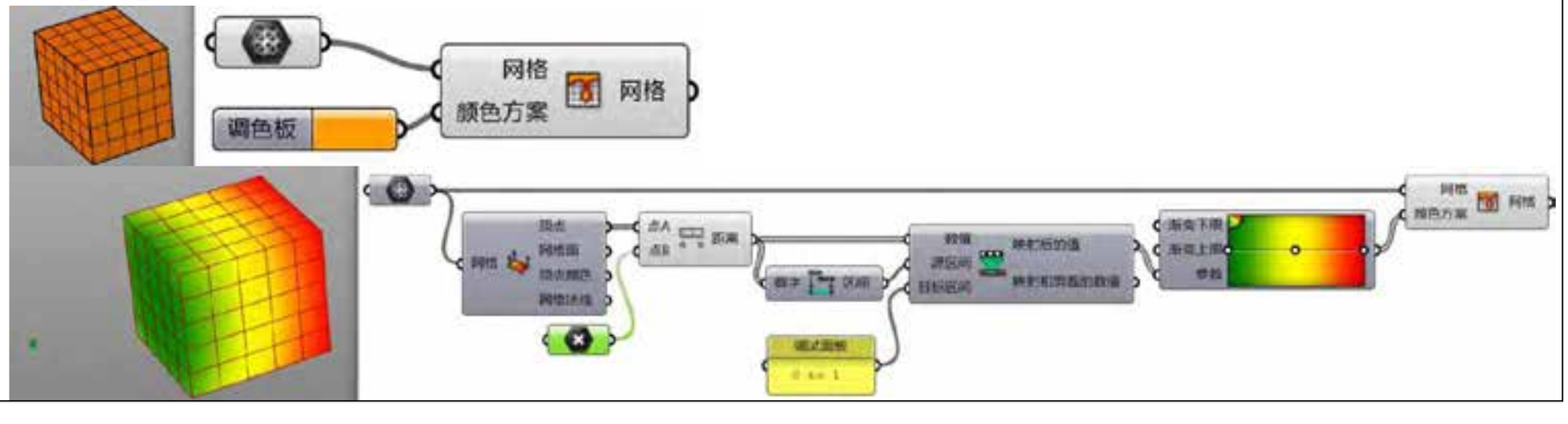


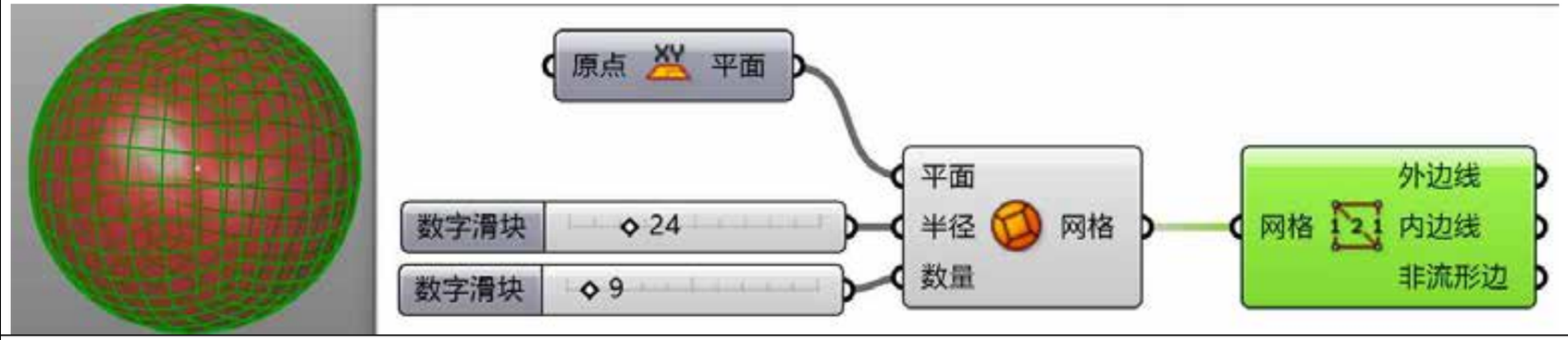
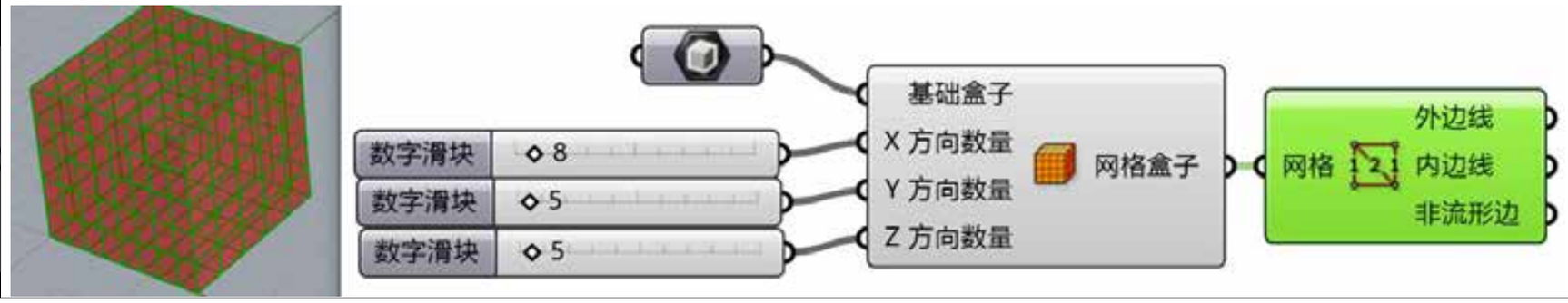


根据设定的喷洒点及颜色，对网格进行着色。右键可以选择不同的喷洒模式。



对网格进行着色。



球形网格 **Mesh Sphere** 	创建球形网格。
球形网格（增强） **Mesh Sphere Ex** 	创建拓扑球体网格，这个球体并不是一个标准球，而是六个面拼合的拟合球体。 
网格平面 **Mesh Plane** 	创建网格平面
网格生盒 **Mesh Box** 	创建网格盒子 

相交 Intersect

区域 Region

相交 Intersect

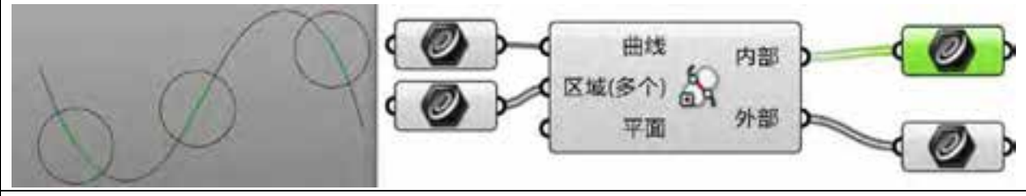
区域 Region - 相交

修剪（单区域）
Trim with Region

用封闭曲线（区域）对曲线进行修剪分割

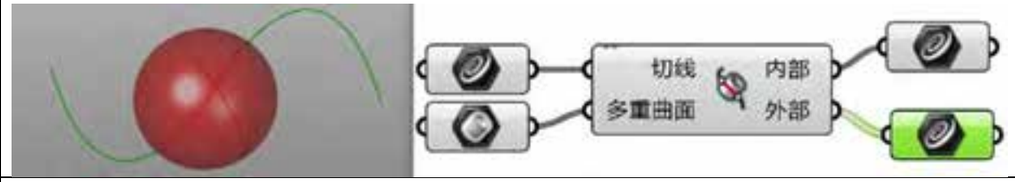
修剪（多区域）
Trim with Regions

用多个封闭曲线对曲线进行修剪分割



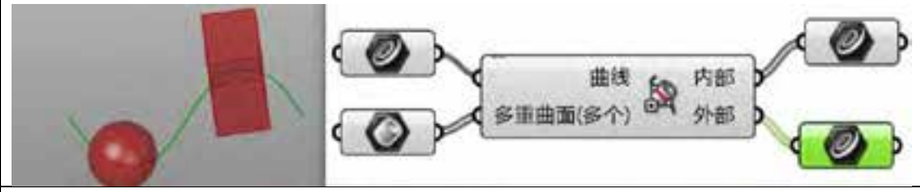
修剪（多重曲面）
Trim with Brep

用多重曲面对曲线进行修剪分割



修剪（多个多重曲面）
Trim with Breps

用多个多重曲面对曲线进行修剪分割



单个多重曲面分割曲线
Split with Brep

用一个多重曲面分割曲线，并得到分割点。

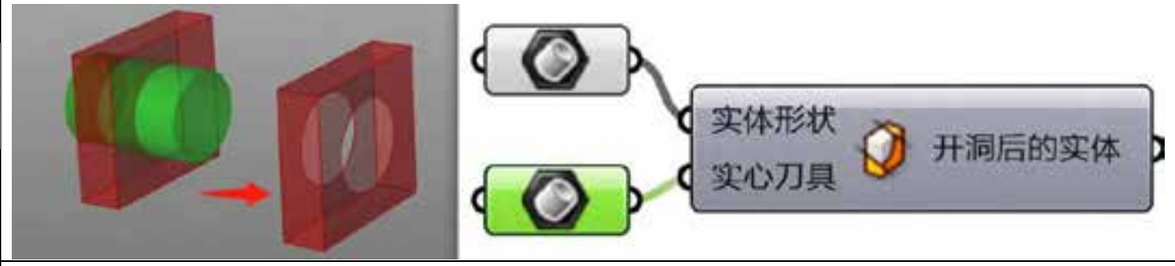
多个多重曲面分割曲线
Split with Breps

用多个多重曲面分割曲线，并得到分割点。

形状 shape - 相交

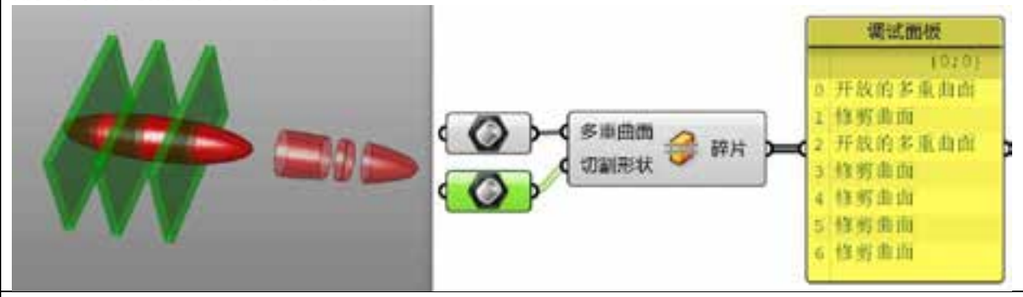
修剪实体（开洞）
Trim Solid

用一个几何体去修剪另一个几何体。



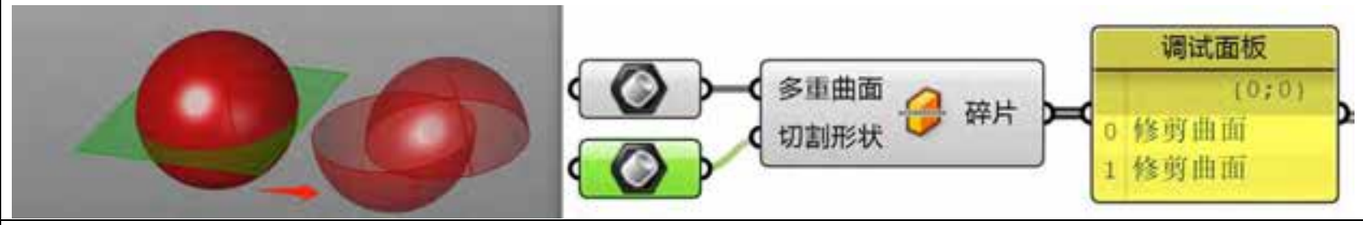
分割多个多重曲面
Split Brep Multiple

用多个物体对指定物体进行分割。



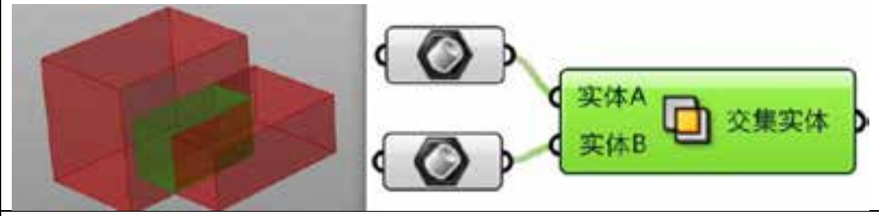
分割多重曲面
Split Brep

用曲面或几何体去分割物体。



实体交集
Solid Intersection

布尔交集运算

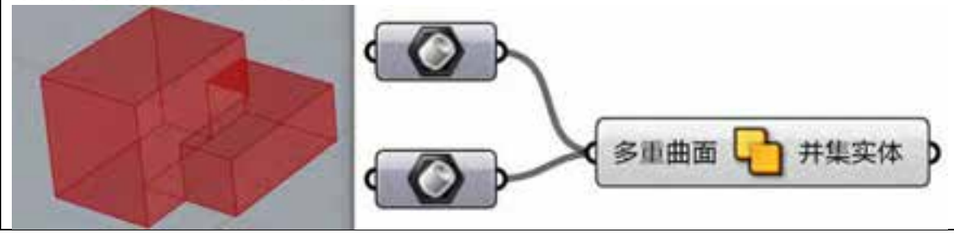


实体差集
Solid Difference

实体 A 减去跟实体 B 重合的部分。也可以用曲面裁剪，但是要注意曲面的方向。

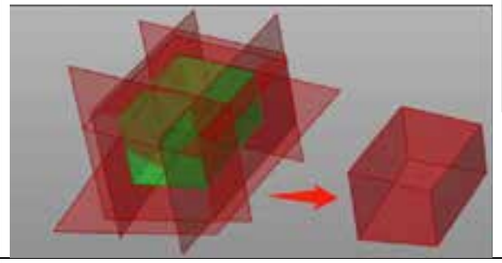
实体并集
Solid Union

布尔并集，将两个多重曲面合并。



边界实体
Boundary Volume

根据多个曲面包围的空间，创建一个实体（多重曲面）。



区域交集
Region Intersection

计算两条曲线的交集

区域差集
Region Difference

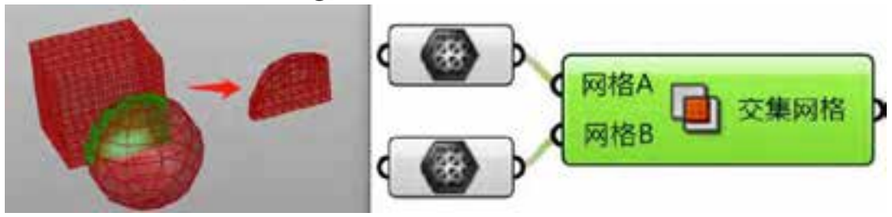
计算曲线 A 减去与曲线 B 的重合部分

区域并集
Region Union

计算两条曲线的并集

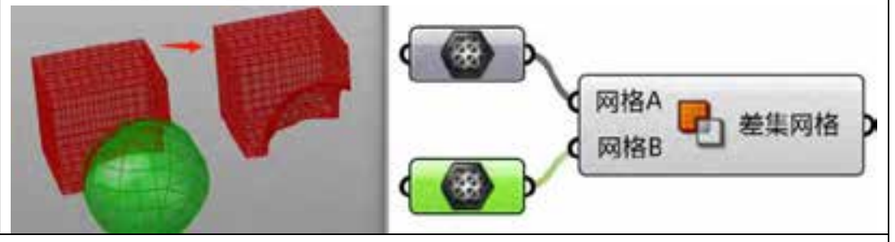
网格交集
Mesh Intersection

网格的交集运算（运算器有 bug）



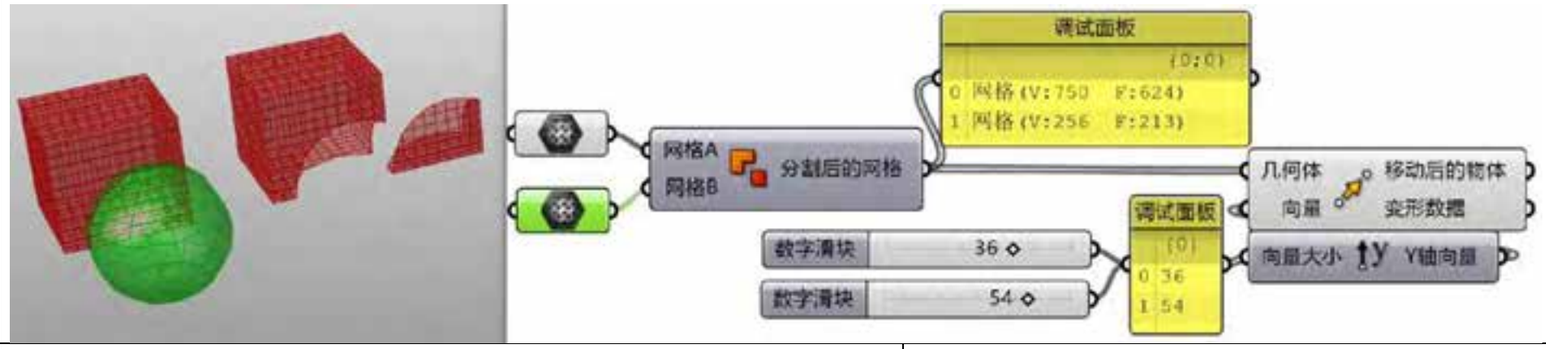
网格差集
Mesh Difference

网格 A 减去与网格 B 重合的部分



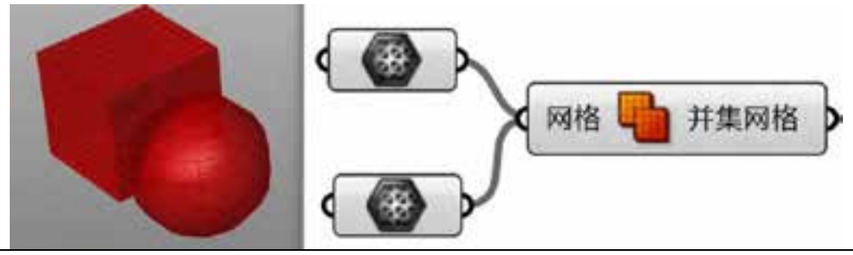
网格分割
Mesh Split

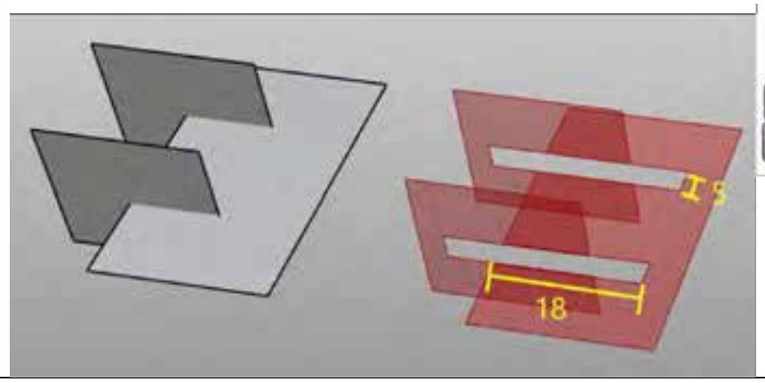
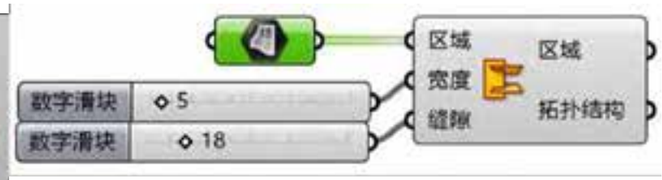
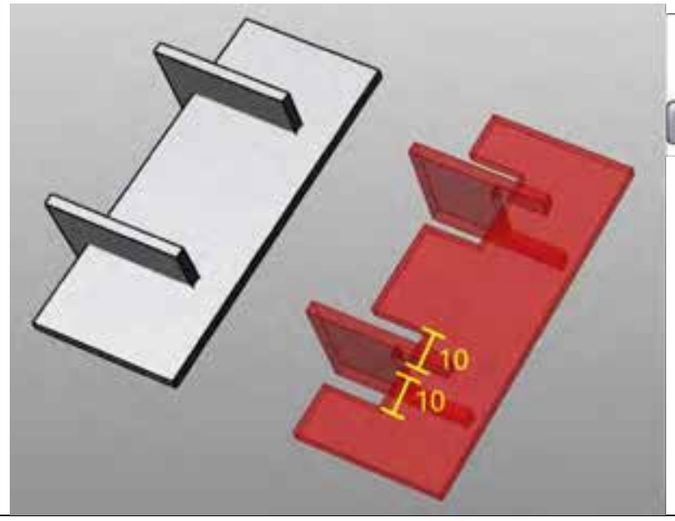
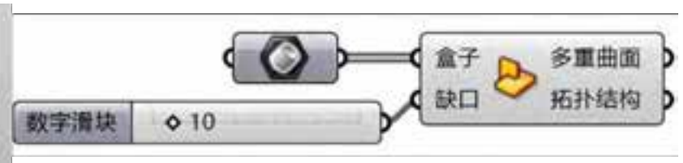
用网格 B 把网格 A 分割成多个部分



网格并集
Mesh Union

将两个网格合并成一个

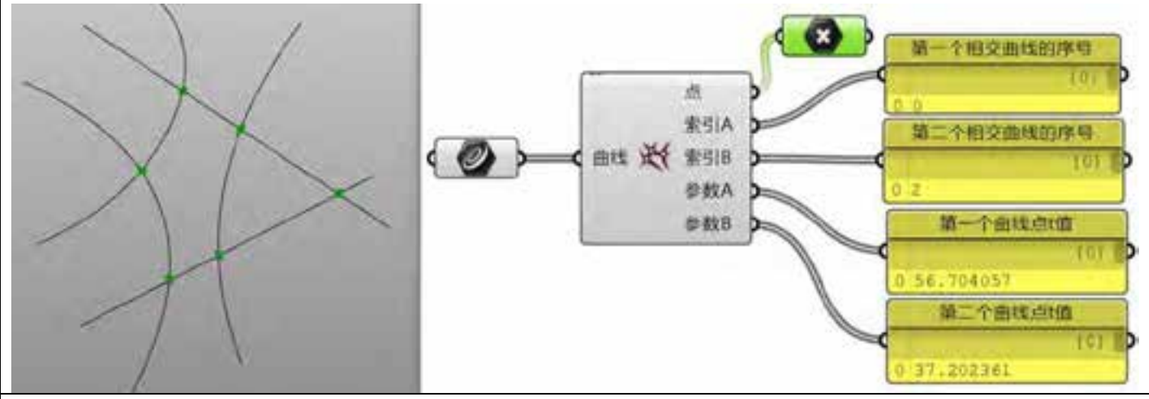


区域开槽 **Region Slits** 	**根据多个平面（或封闭的平面曲线）之间的接缝，对平面进行开槽。**  
盒子开槽 **Box Slits** 	**根据多个多重曲面进行开槽（不能以盒子数据输入，会报错）。**  

物理 Physical - 相交

多线相交
Multiple Curves

计算多个曲线的交点



曲线自相交
Curve | Self

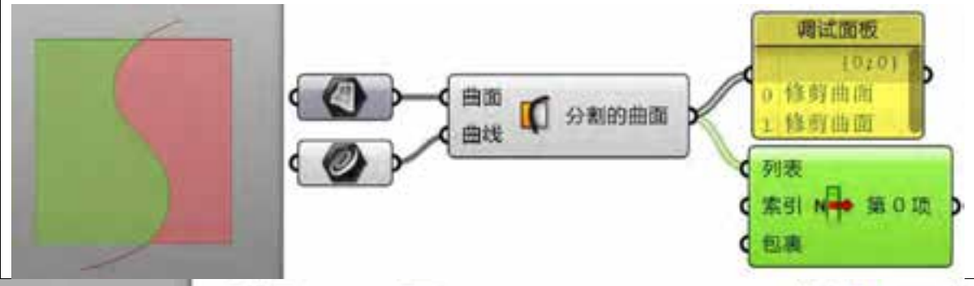
计算曲线的自交点

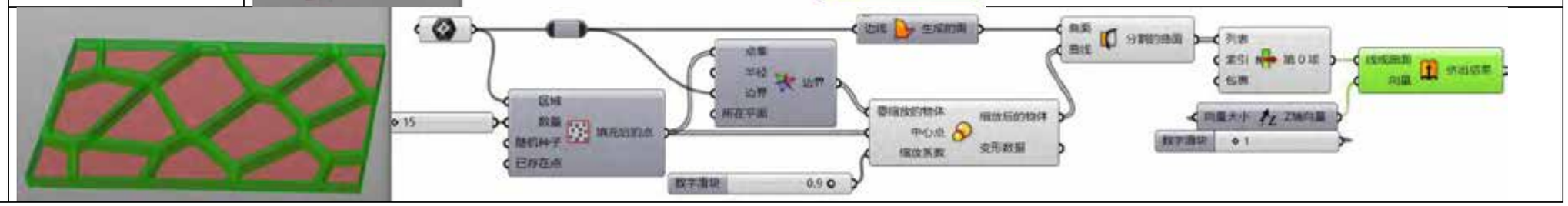
线|线相交
Curve | Curve

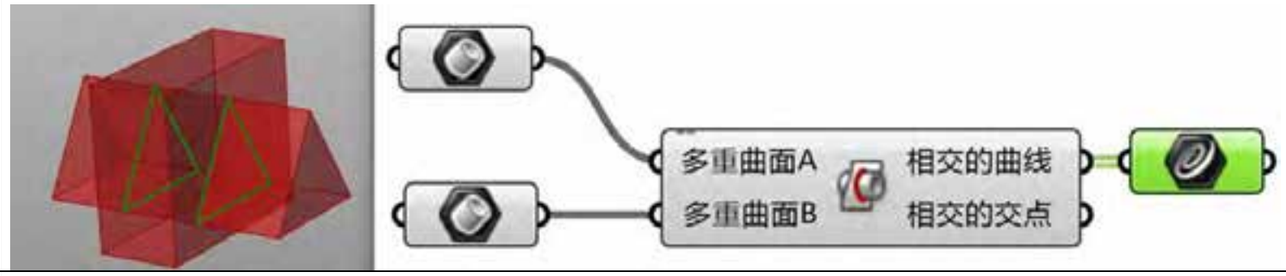
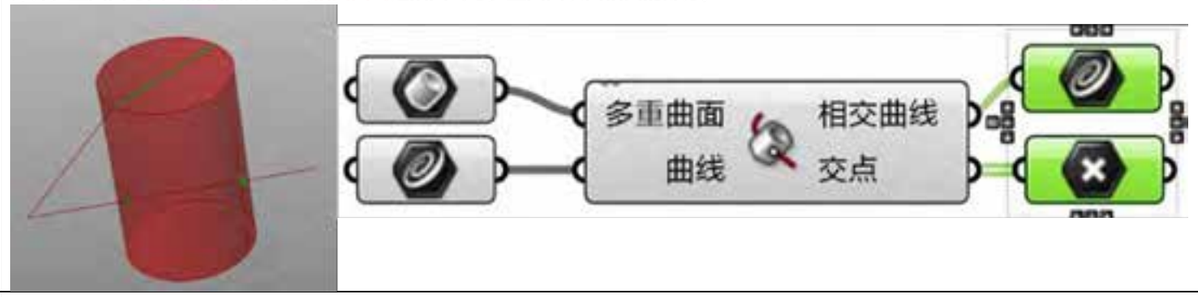
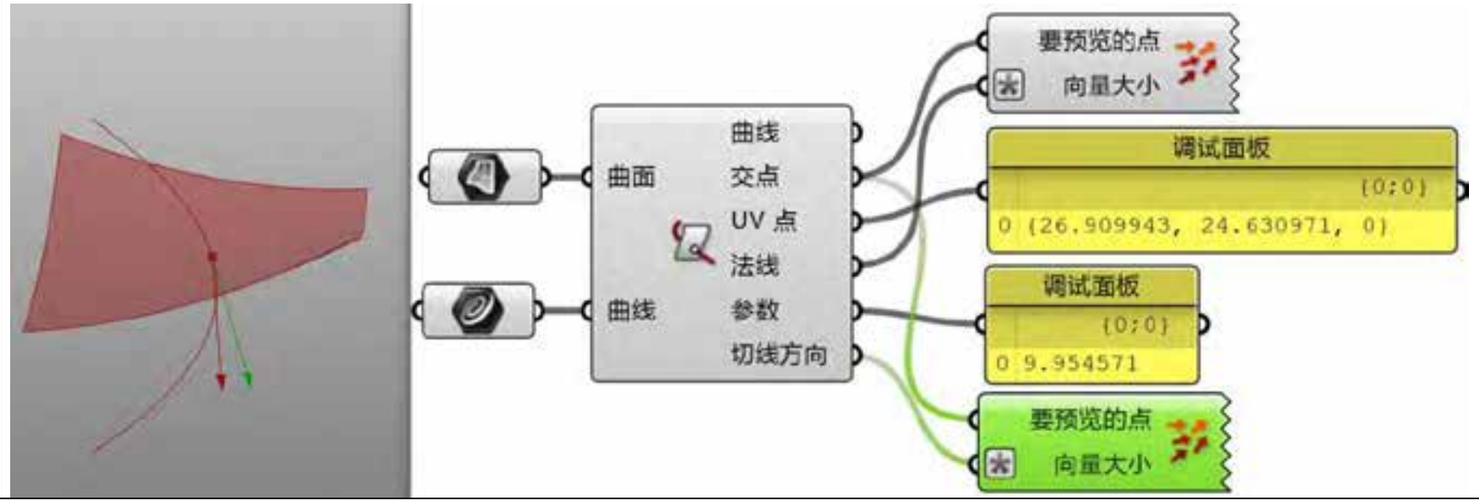
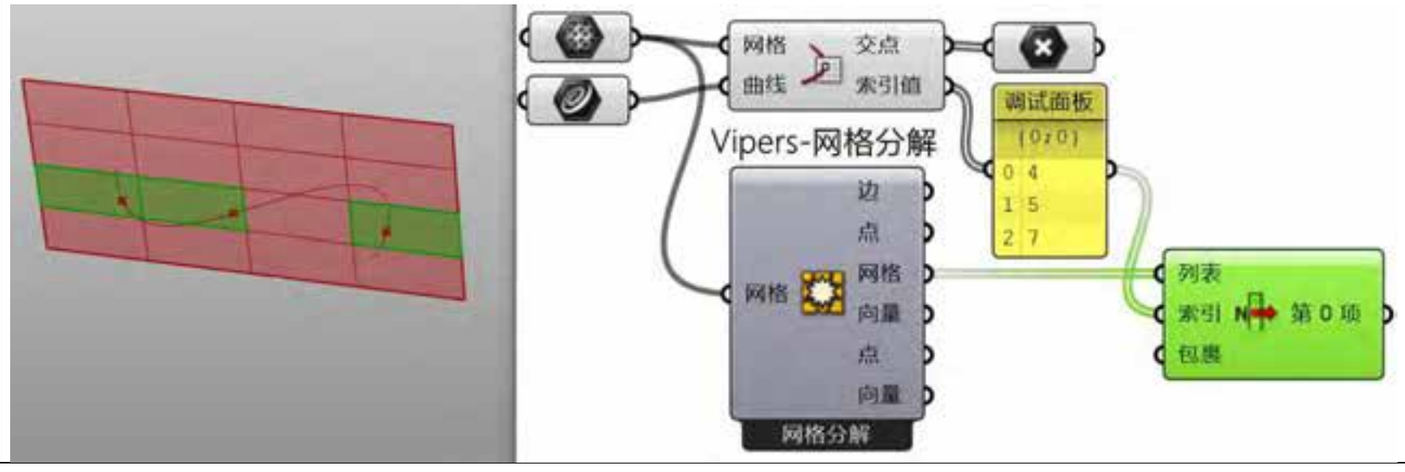
计算两条曲线的交点。参数 A 为交点在曲线 A 上的 t 值，参数 B 为交点在曲线 B 上的 t 值。

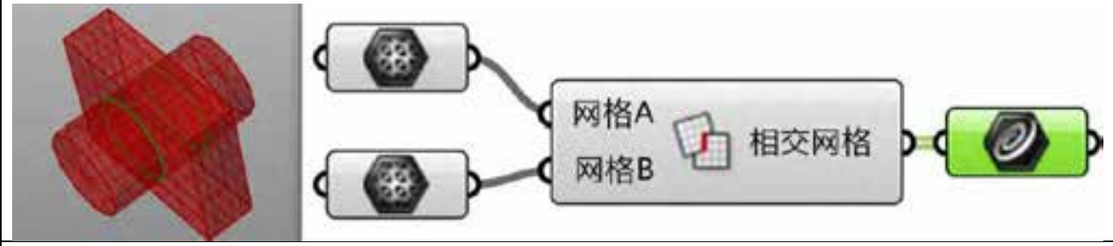
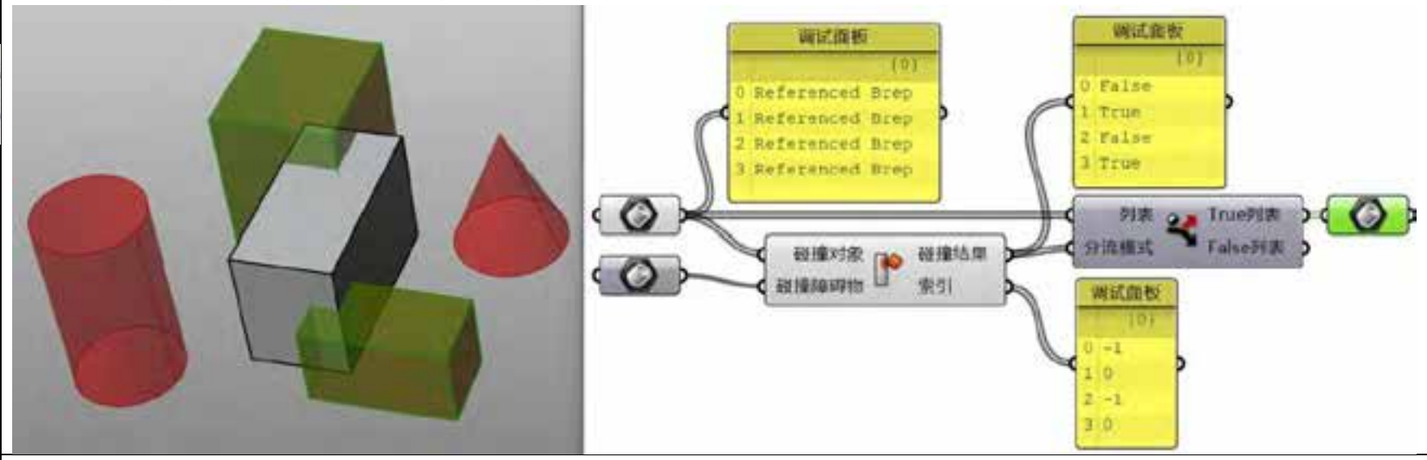
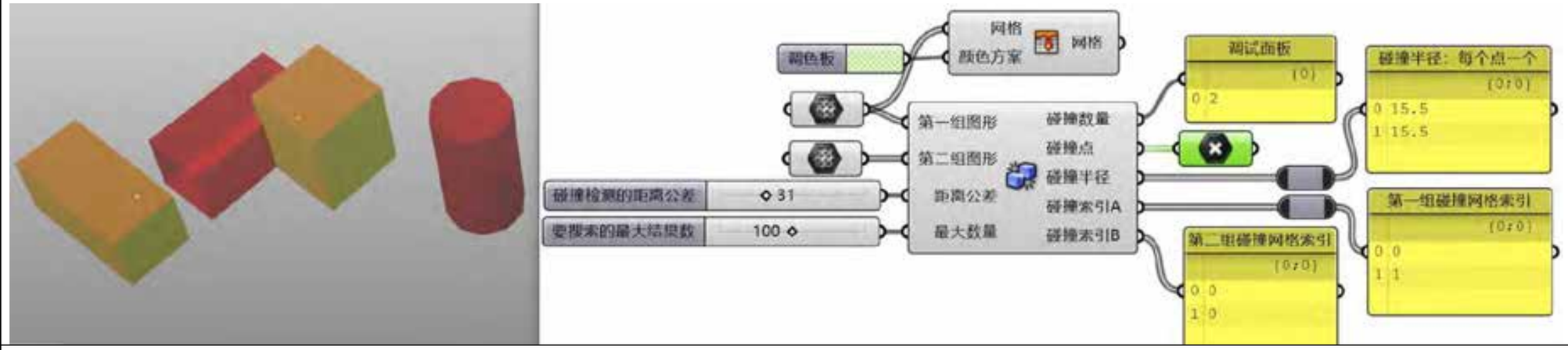
分割曲面
Surface Split

用曲线对曲面进行分割。





多重曲面|多重曲面相交 **Brep | Brep** 	计算多个多重曲面的相交线 
多重曲面|曲线相交 **Brep | Curve** 	**计算多重曲面与曲线的相交部分。（相交曲线是指曲线与多重曲面表面相交的部分）** 
曲面|曲线相交 **Surface | Curve** 	**计算曲面和曲线的相交线和相交点。UV 点为相交点在曲面上的 UV 坐标值，法线为点在曲面上的法线，参数为点在曲线上的参数，切线方向为点在曲线上的切线方向。** 
网格|曲线相交 **Mesh | Curve** 	**计算网格与曲线相交，索引值为相交的网格面序号。** 

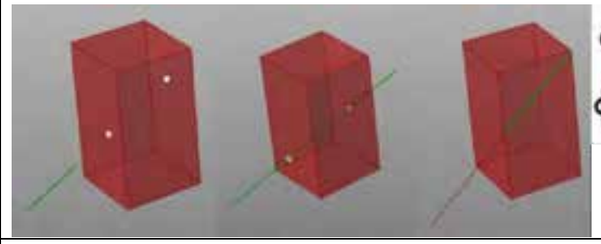
网格|网格相交 **Mesh | Mesh** 	**计算两个网格的相交。** 
碰撞一对多 **Collision One|Many** 	**检测单个物体（碰撞障碍物）与多个物体（碰撞对象）是否有碰撞。** 
碰撞分析 **Collision Analysis** 	**检测网格之间在指定的距离范围内（距离公差），是否会发生碰撞。** 
碰撞多对多 **Collision Many|Many** 	**判断多个物体之间是否相交。**

算术 Mathematical

- 相交

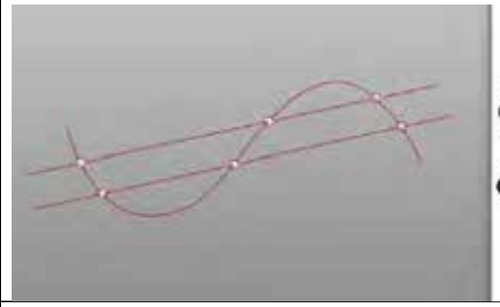
多重曲面|直线相交
Brep | Line

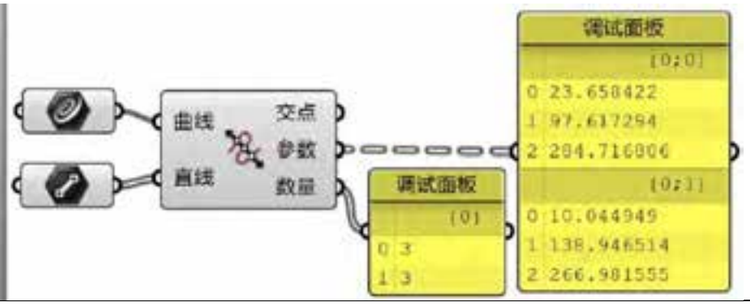
计算直线（无限延长）与多重曲面的交点。可以右键选择 Limit to First，则只保留直线和物体相交的第一个点。



曲线|直线相交
Curve | Line

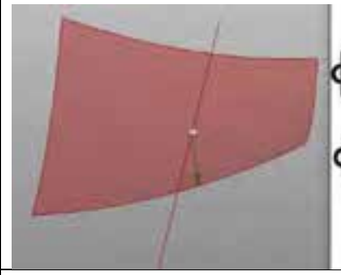
计算曲线和直线的交点，及交点在曲线上的 t 值和交点的数量。同上也可以右键选择 Limit to First。

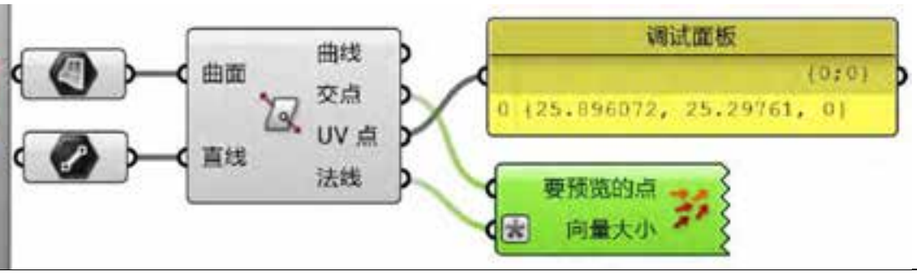




曲面|直线相交
Surface | Line

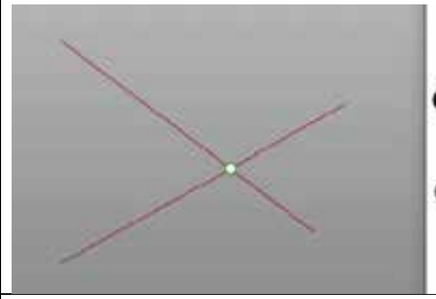
计算曲面与直线的相交。

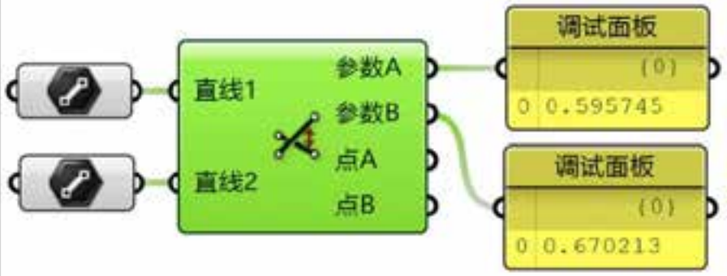




直线|直线相交
Line | Line

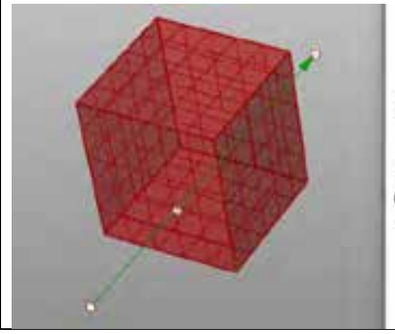
计算两条直线的相交，输出点在各自直线上的 t 值。

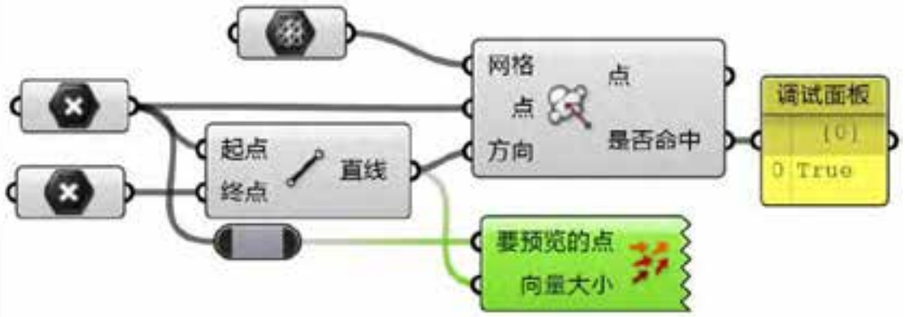




网格|射线相交
Mesh | Ray

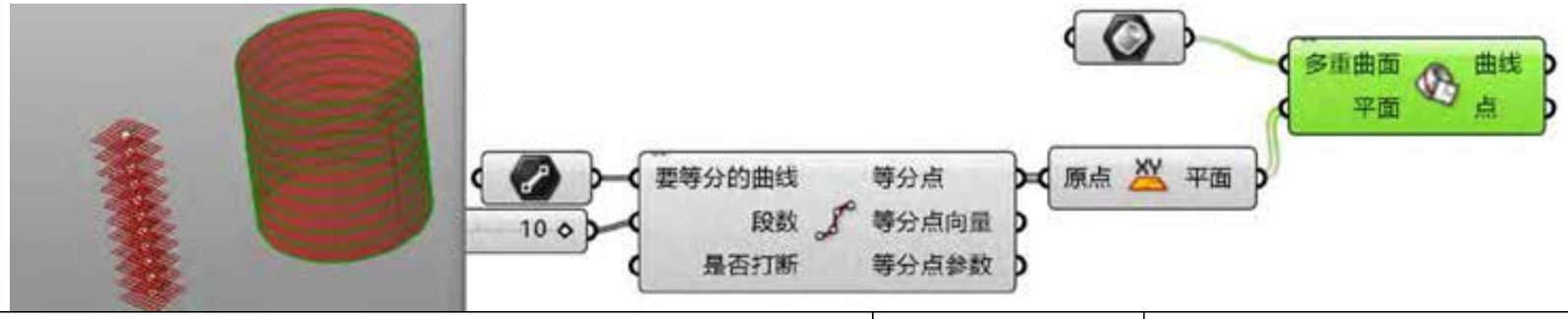
计算网格与射线的交点。射线是指从一个起点沿一个方向无限延长的线。





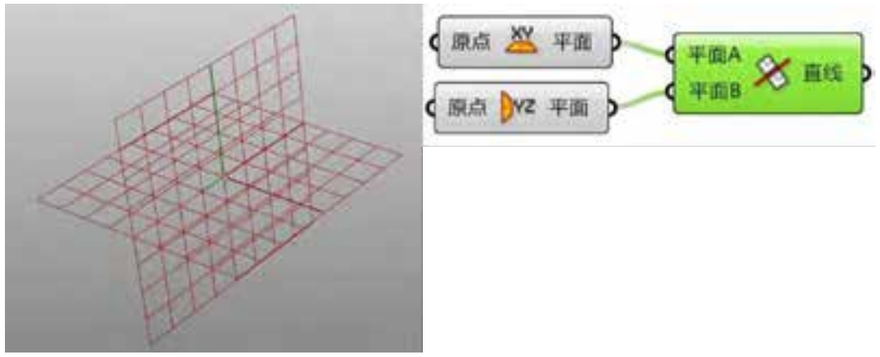
多重曲面|平面相交
Brep | Plane

计算多重曲面与平面的相交线及交点。



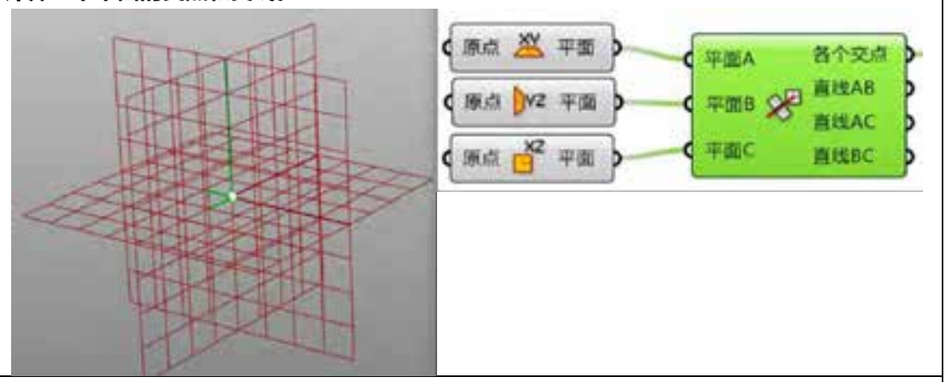
平面|平面相交
Plane | Plane

计算两个平面的相交线。由于两平面是无限平面，所以输出单位为 1 的长度。



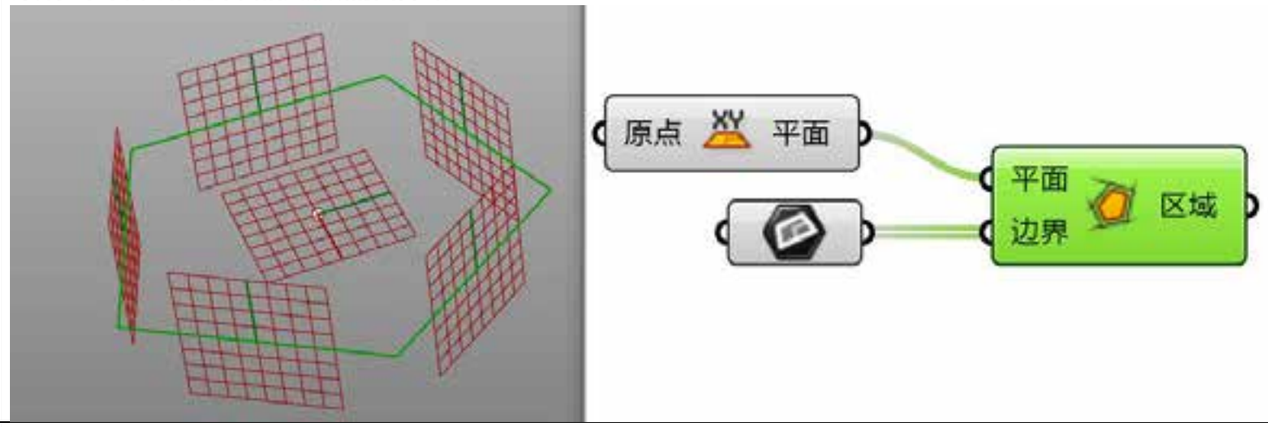
平面|平面|平面相交
Plane | Plane | Plane

计算三个平面的交点和交线。



平面边界
Plane Region

计算多个平面与指定平面相交形成的围合线。

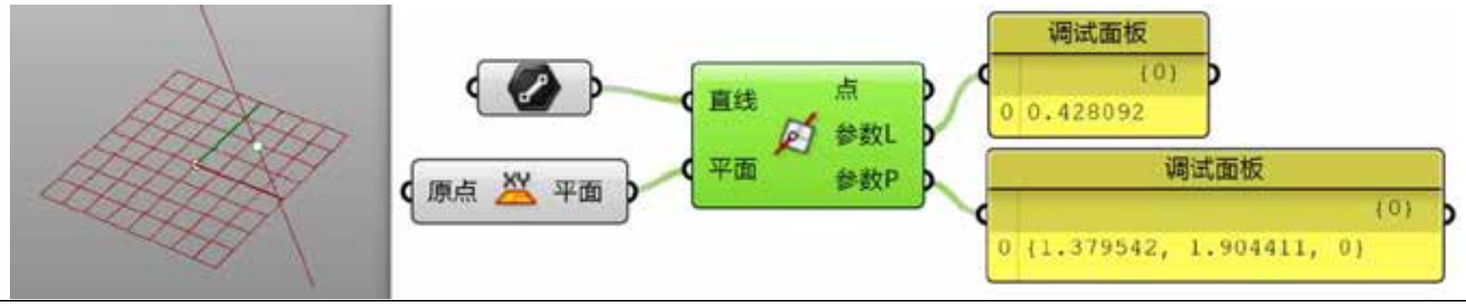


曲线|平面相交
Curve / Plane

计算曲线与平面的交点。参数 C 为交点在曲线上的 t 值，参数 P 为交点在平面坐标系中的坐标值。

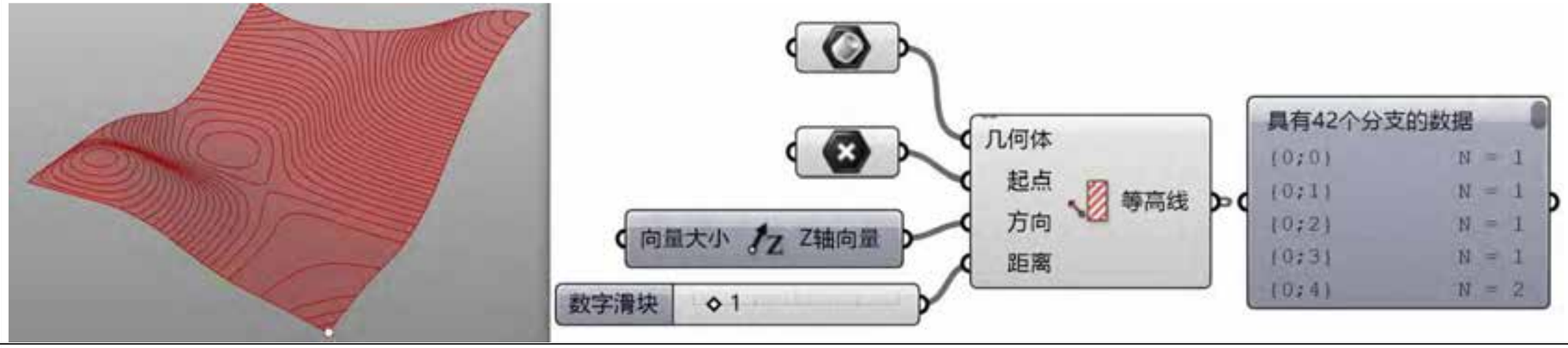
直线|平面相交
Line | Plane

计算直线与平面的交点。



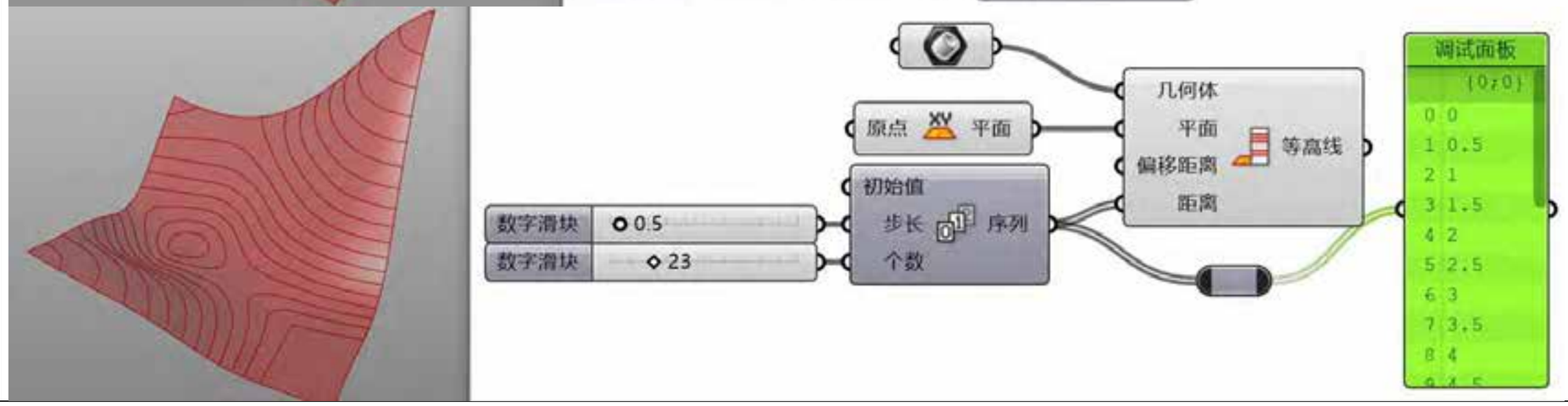
等高线曲面
Contour

等距断面线



等高线曲面（增强）
Contour (ex)

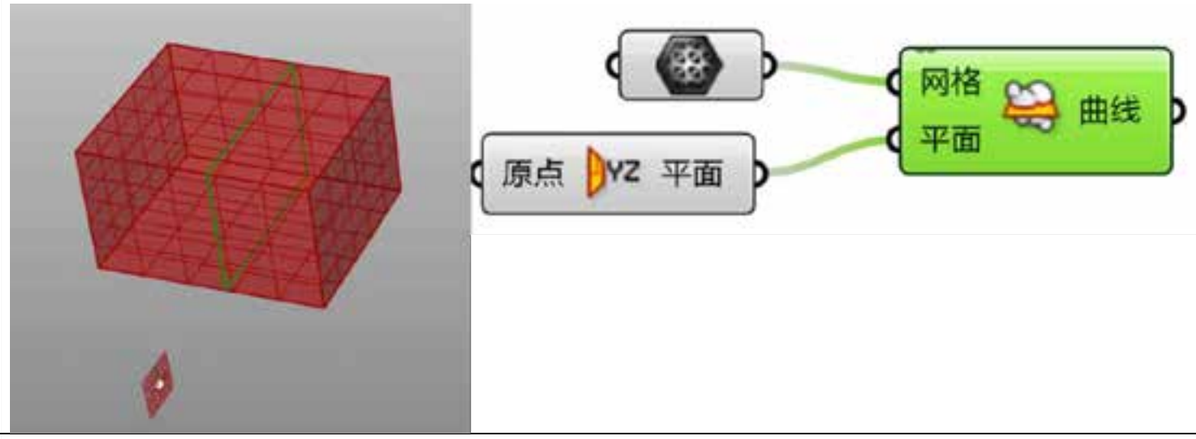
根据以指定平面为起始平面，离平面的距离来生成等高线。**偏移距离**是指每条等高线与平面之间的距离。**距离**则是等高线之间相对平面的垂直距离。



网格|平面相交

Mesh | Plane

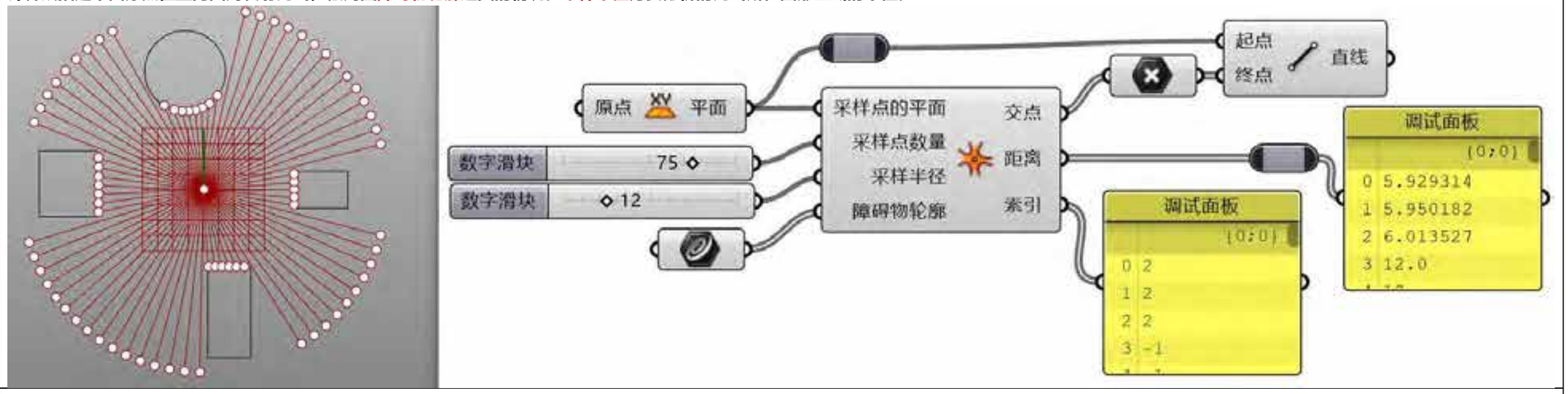
计算网格与平面的相交线。



等视

IsoVist

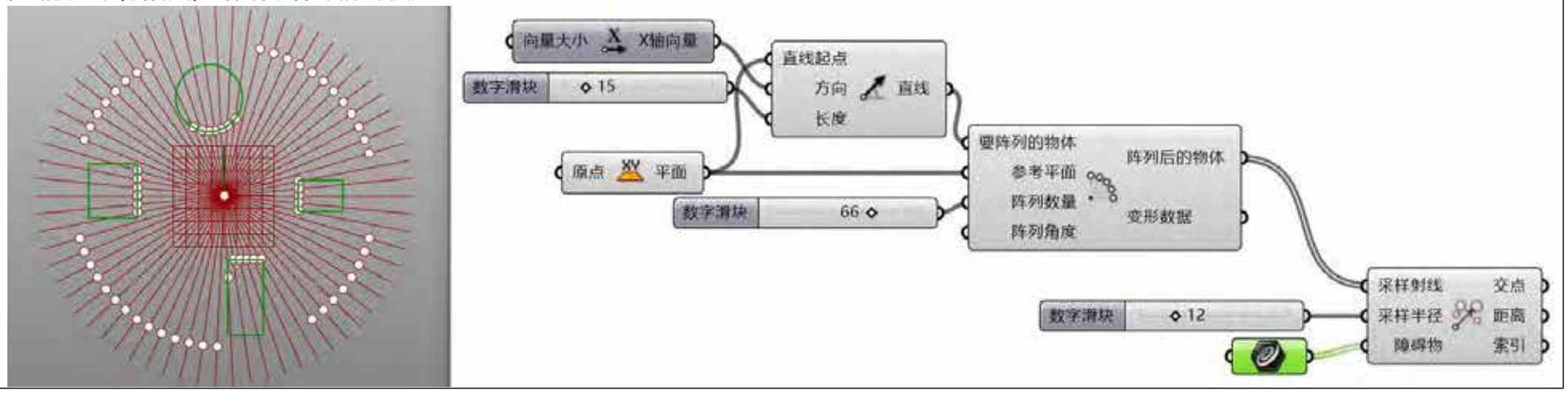
计算从指定平面原点位置向四周发射光线，被周围障碍物轮廓遮挡的情况。采样半径为要分析的光线所在圆形区域的半径。



等视射线

IsoVist Ray

检测指定直线（采样射线）与障碍物是否遮挡并返回交点



变形 Transform

仿射 Affine

变形 Transform

仿射 Affine – 变形

三角缩放
Scale NU

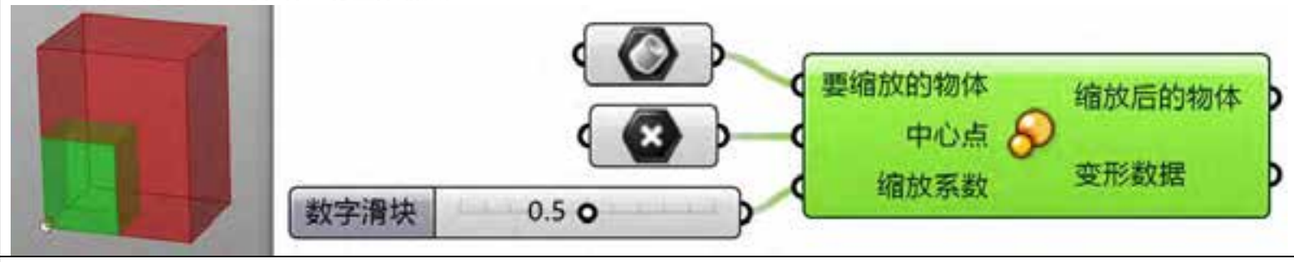
根据指定物体在参考平面坐标系的三轴缩放系数，对物体进行缩放。

相机缩放
Camera Obscura

根据小孔成像原理，对物体进行缩放。

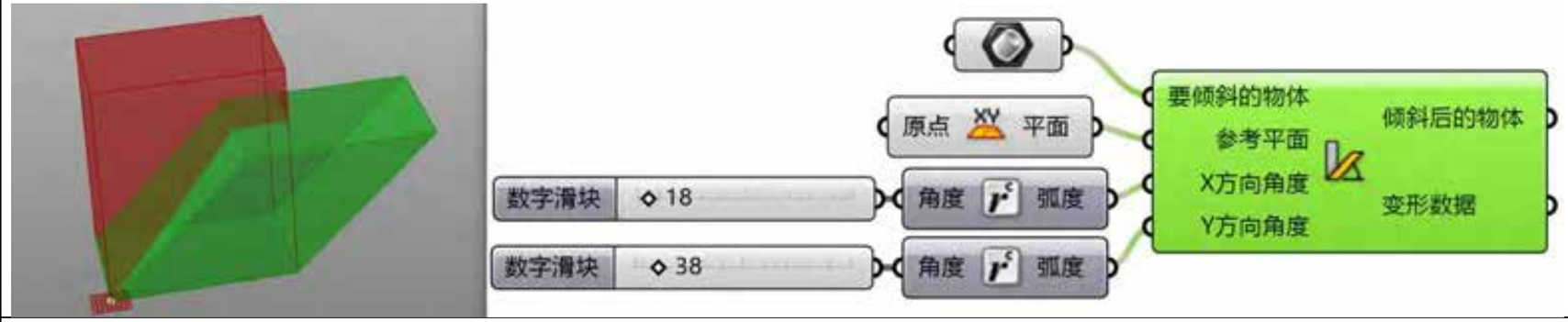
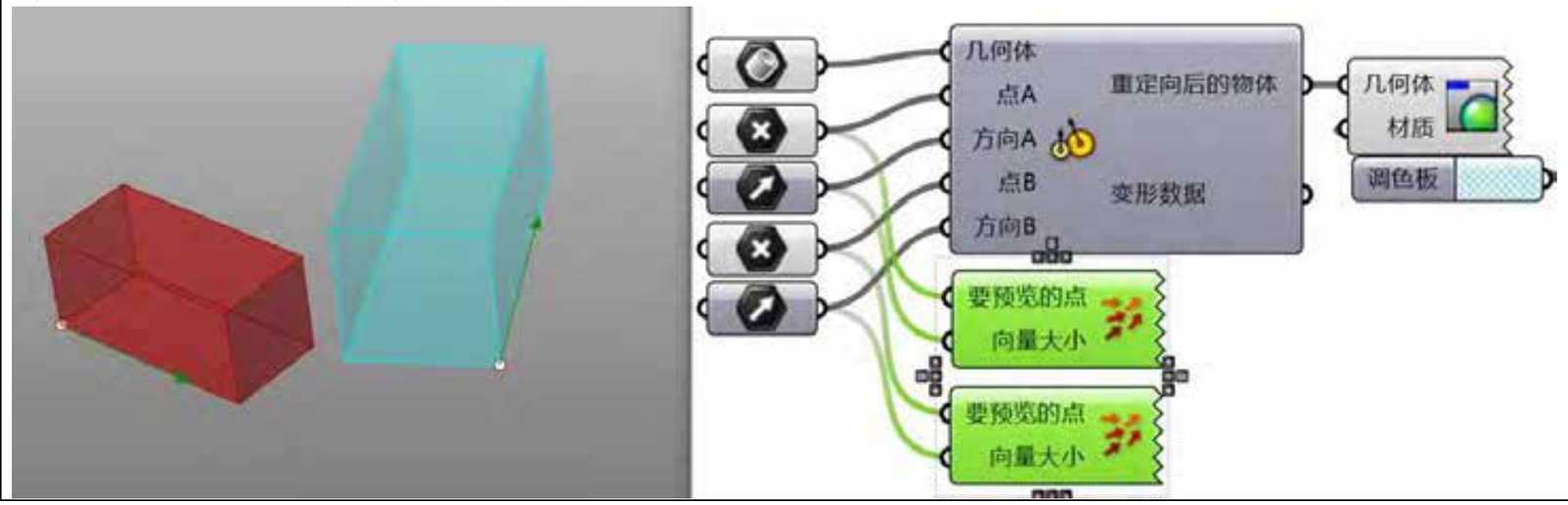
缩放
Scale

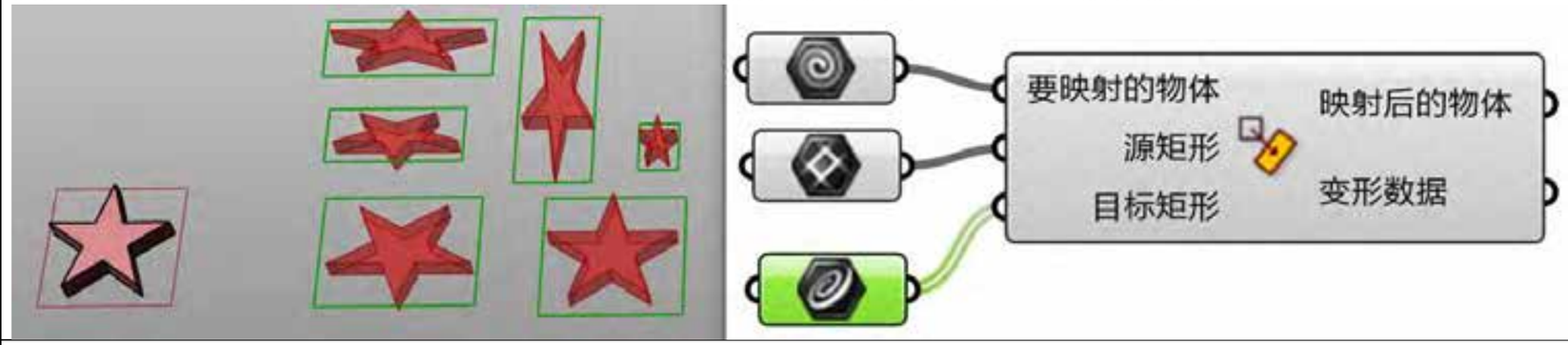
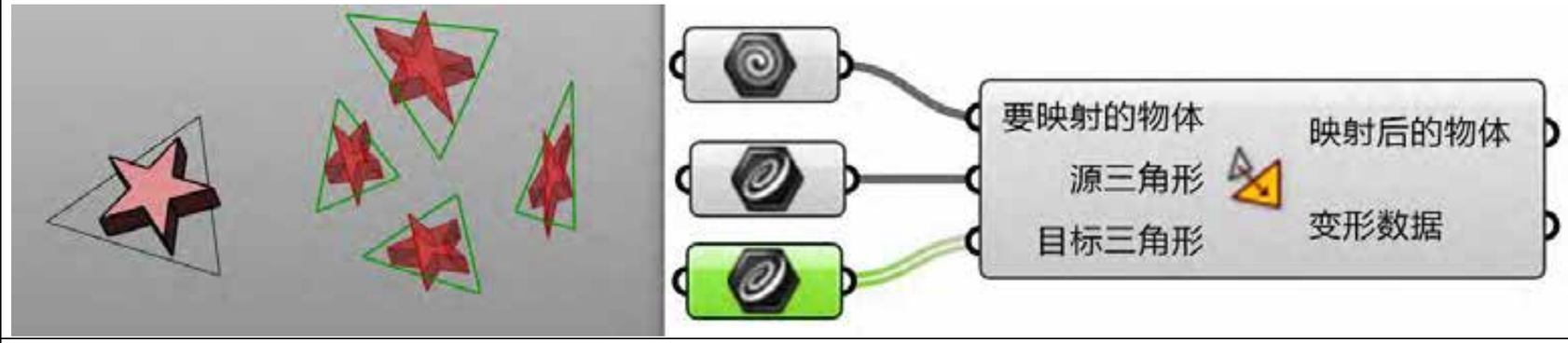
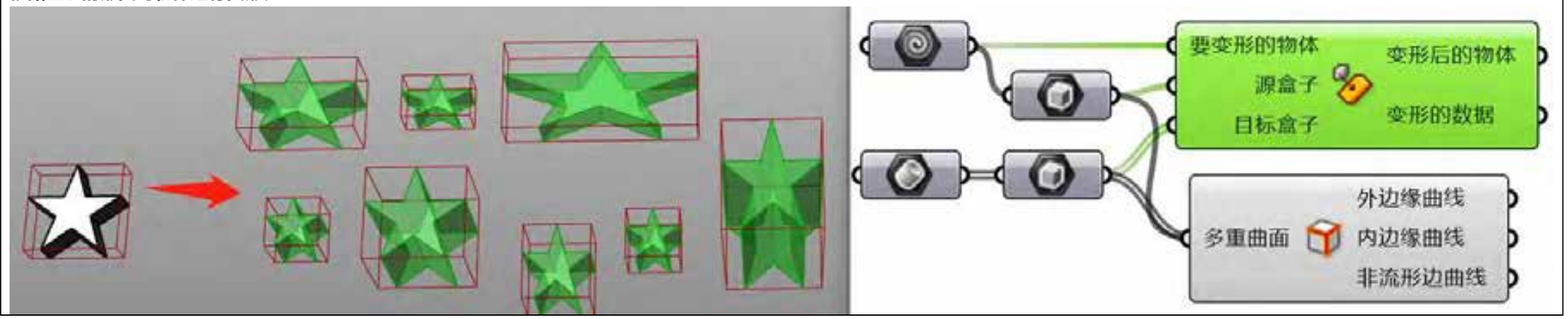
基于缩放点（中心点）对物体进行整体缩放。



定向倾斜
Shear

基于参考平面的原点为角点，与起始点和目标点之间连线的夹角角度，对物体进行倾斜变形。

角度倾斜 **Shear Angle** 	**以参考平面的原点为基点，按设定的角度对物体进行倾斜变形。X 方向角度为以 X 轴旋转的角度，Y 方向角度为以 Y 轴旋转的角度。** 
投影 **Project** 	**计算物体在指定平面上的垂直投影。**
定向投影 **Project Along** 	**按照设定的投影方向计算物体在平面上的投影。**
定向翻转 **Orient Direction** 	**参照设定的两个向量的方向及大小，对物体进行旋转缩放。** 

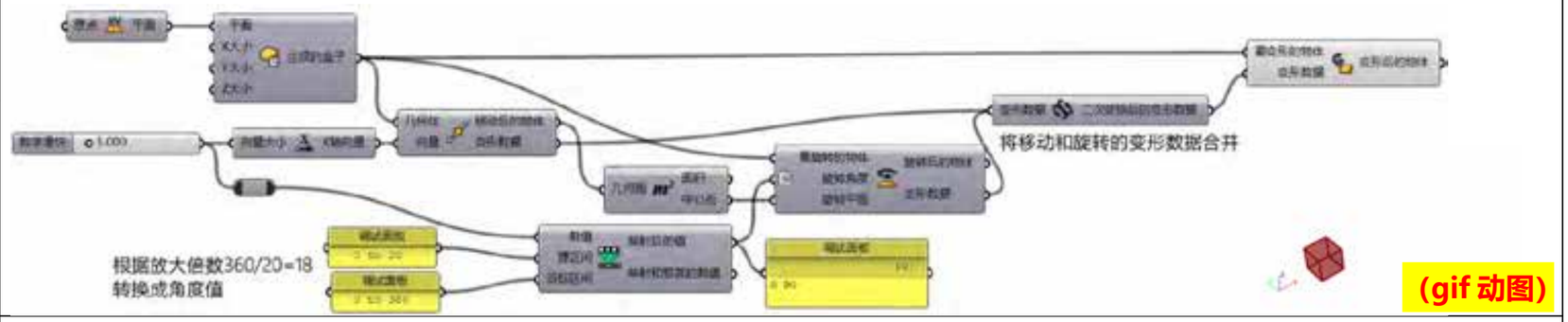
矩形映射 **Rectangle Mapping** 	**根据矩形之间的形变来对物体进行变形，但是只能映射一个物体（rhino 中的群组或图块都不可以），还要注意矩形角点的顺序。** 
三角映射 **Triangle Mapping** 	**根据三角形之间的形变来对物体进行变形。** 
盒子映射 **Box Mapping** 	**根据盒子的形变对物体进行变形。** 

公用 Util- 变形

复合二次变换 Compound

变形 Transform

复合二次变换：将多组变形数据合并，比如下图将移动和旋转变形数据合并后，可以使物体在移动的时候旋转。



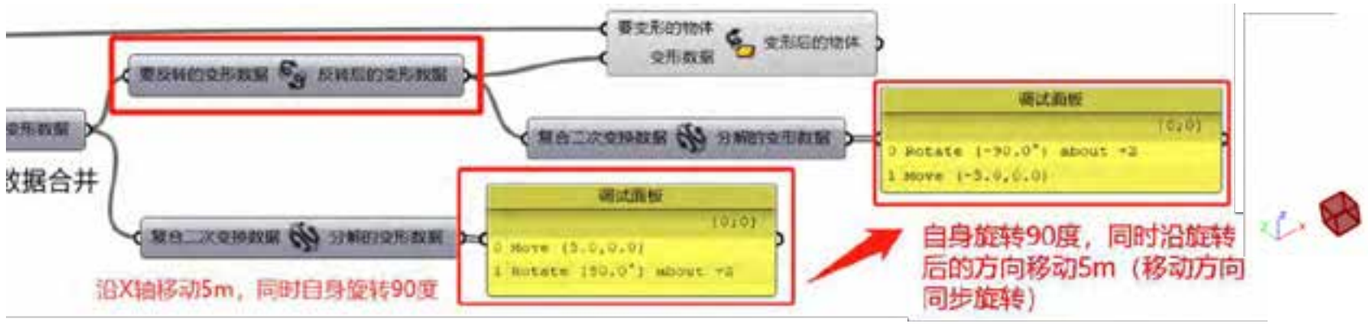
(gif 动图)

分解复合二次变换 Split

将合并的变形数据进行分解。

反转变换 Inverse Transform

将前面实例的两种变换反转。

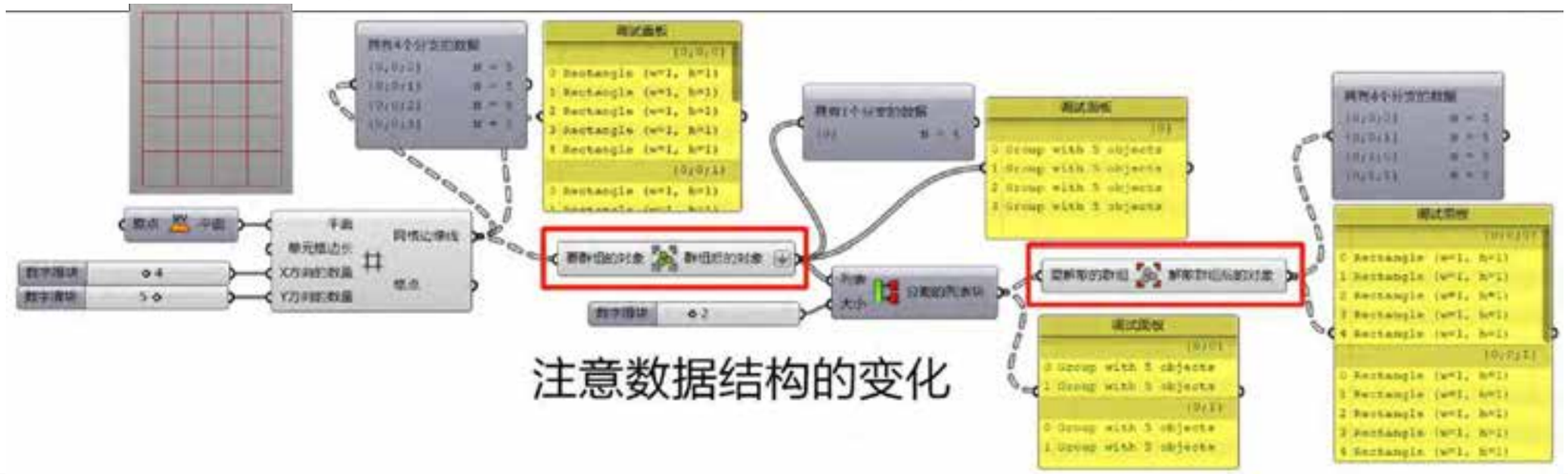


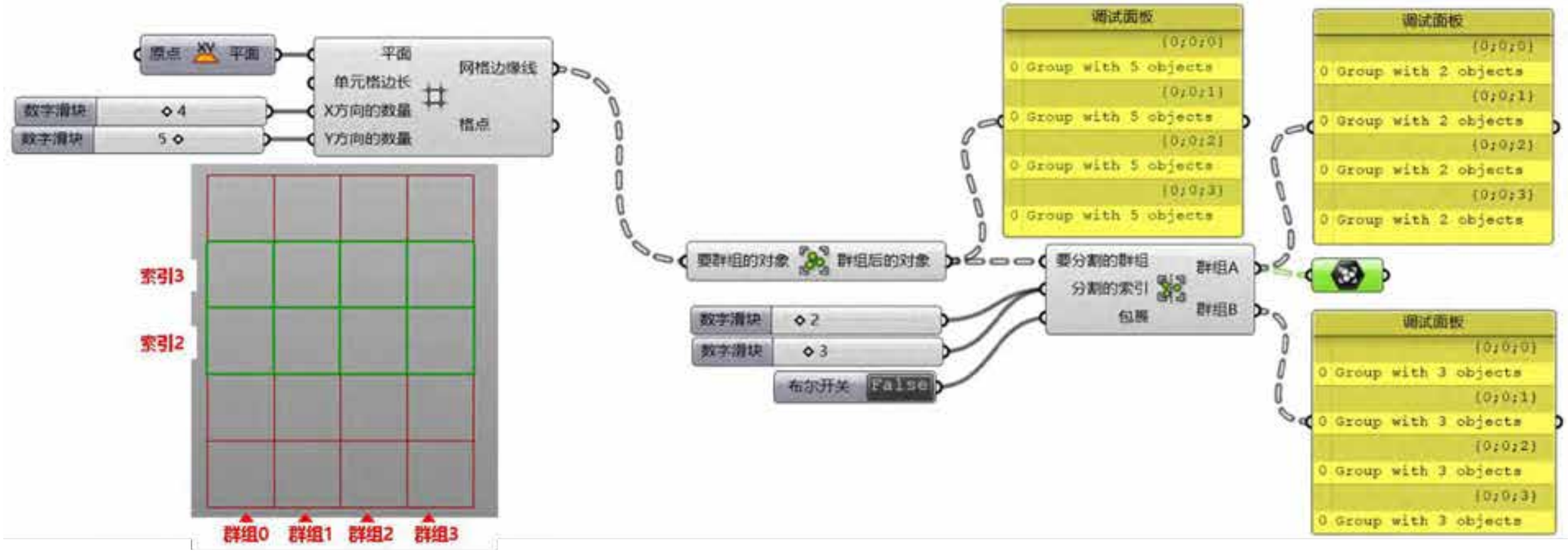
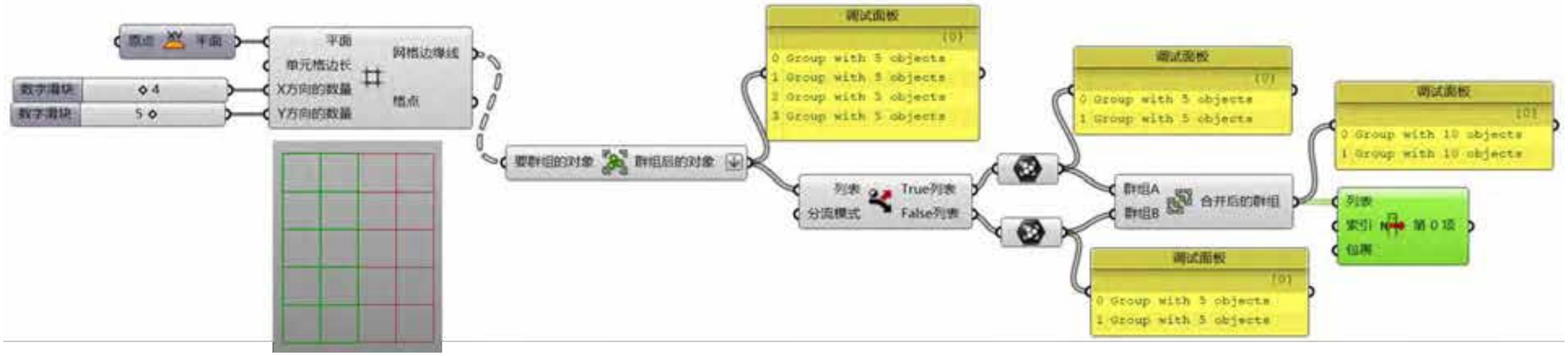
(gif 动图)

群组 Group

解散群组 Ungroup

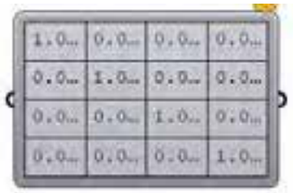
群组：将分组内的所有数据成群组（烘焙到 rhino 中自动成群组），作为一个物件来处理的运算器。（跟 rhino 中点群组功能一致）。解散群组功能相反。



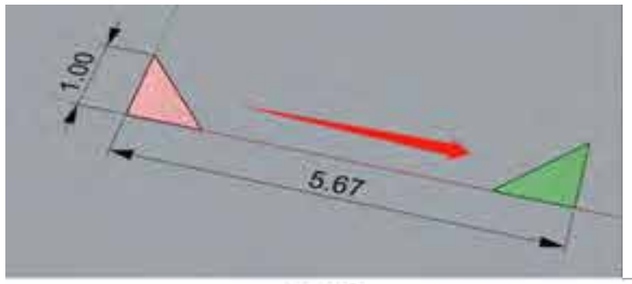
分割群组 **Split Group** 	**将群组中指定序号的物体分离出来组成群组。如下图中，将每个群组中的序号为 2、3 的网格分割出来成群组，剩下的序号 0、1、4 网格成群组。** 
合并群组 **Merge Group** 	**将两个列表中序号相同的群组提取出来，组成一个群组。** 

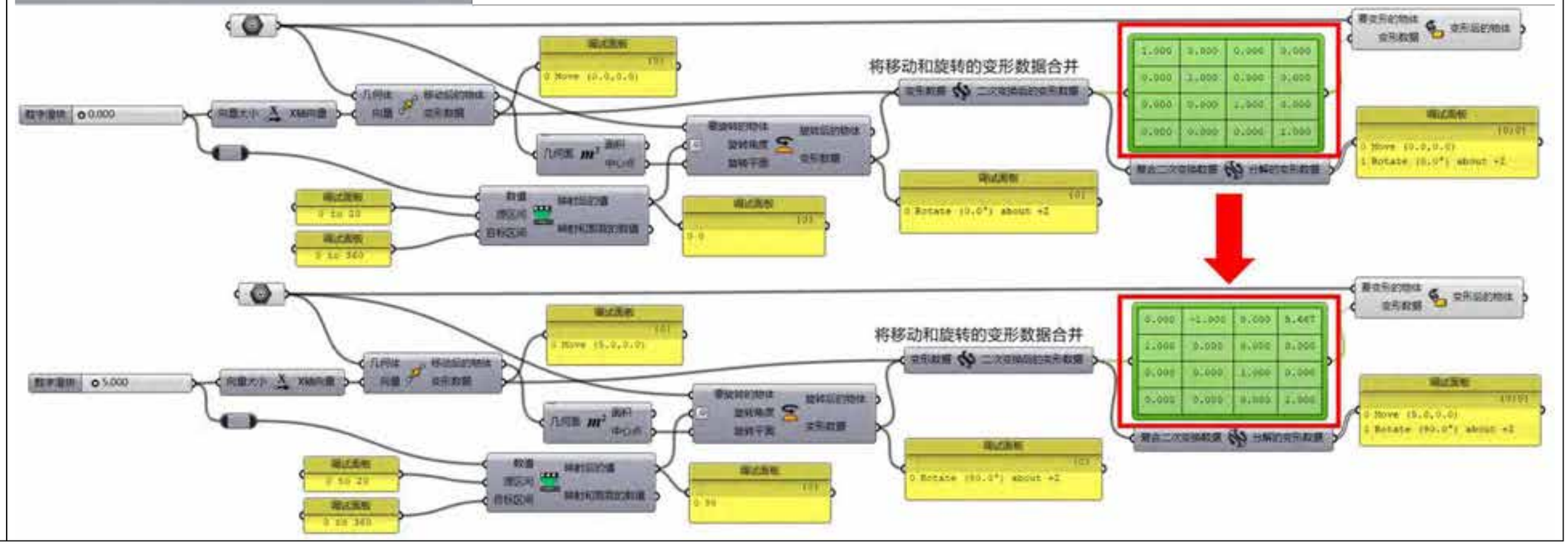
矩阵变换

Transform Matrix



以下实例可以查看矩阵的数据变换。（暂未理解透彻）

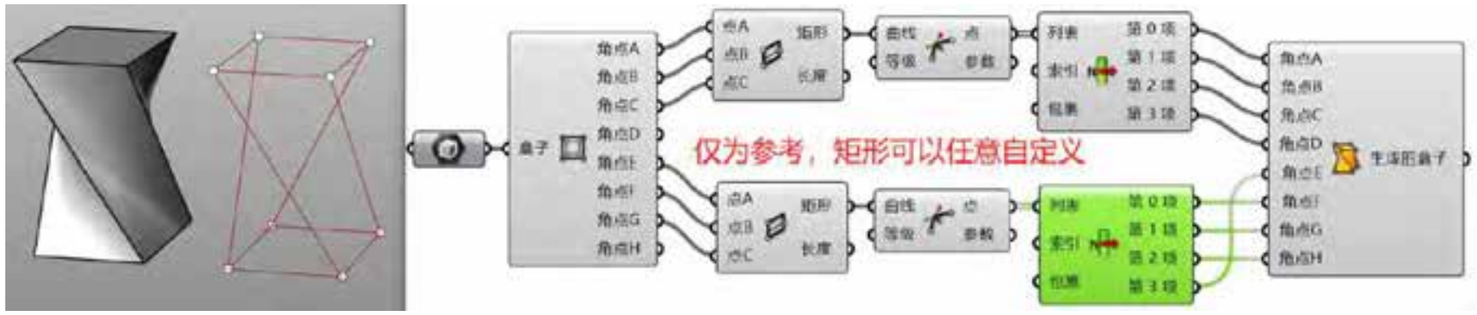




变形 Morph – 变形

8 点生盒
Twisted Box

根据八个点生成扭转的盒子。

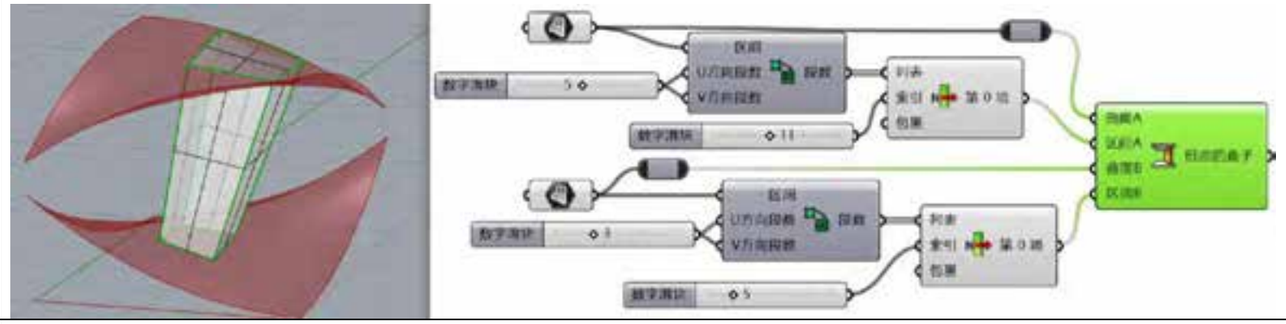


单面生盒
Surface Box

将曲面指定区间的区域，按照设定高度挤出成扭曲的盒子。

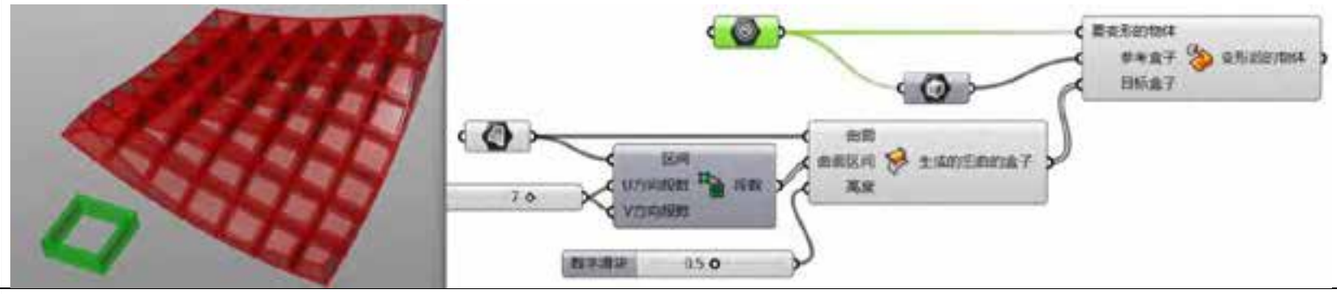
双面混接生盒
Blend Box

根据两个曲面各自指定区间的区域，连接生成一个扭曲的盒子。



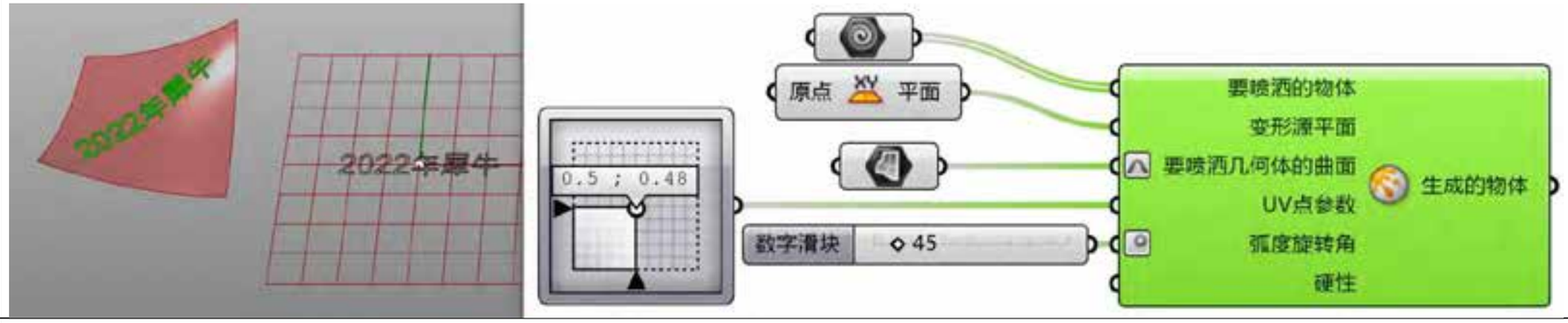
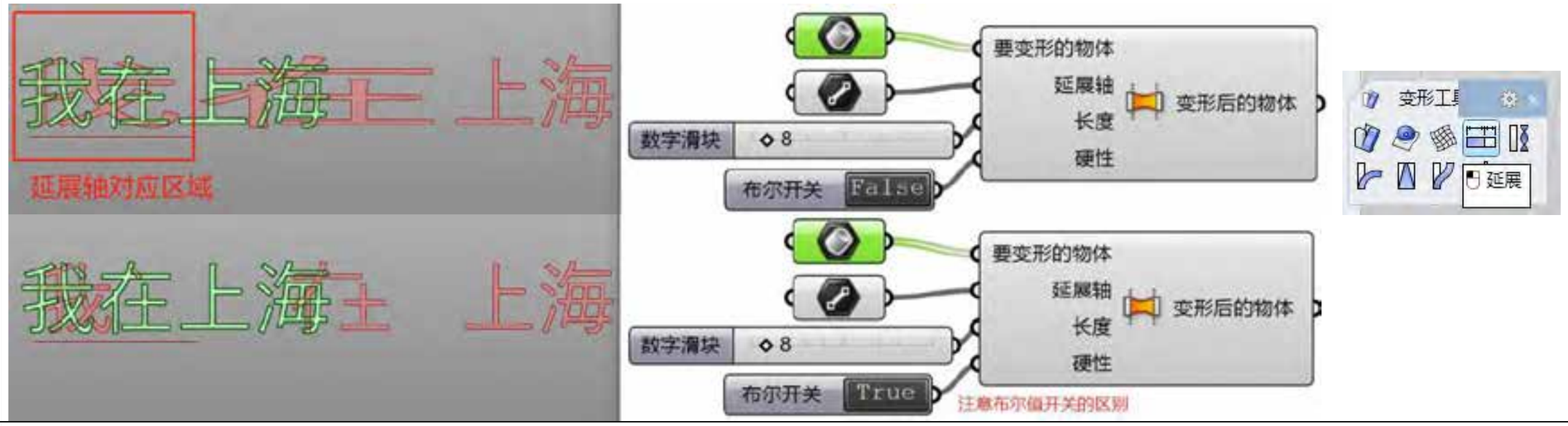
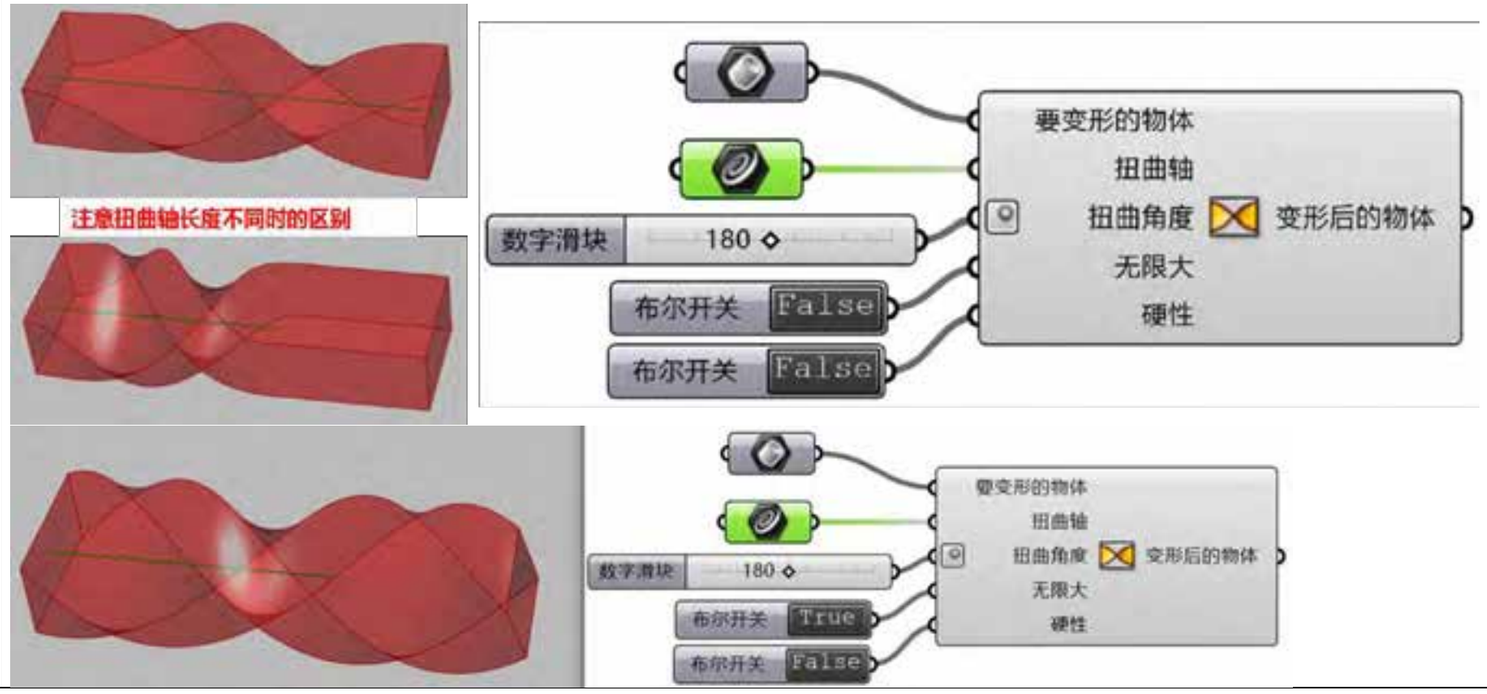
盒子变形
Box Morph

根据要变形的物体所在的盒子空间（参考盒子），将物体投射到目标盒子中去，跟前面的盒子映射类似。



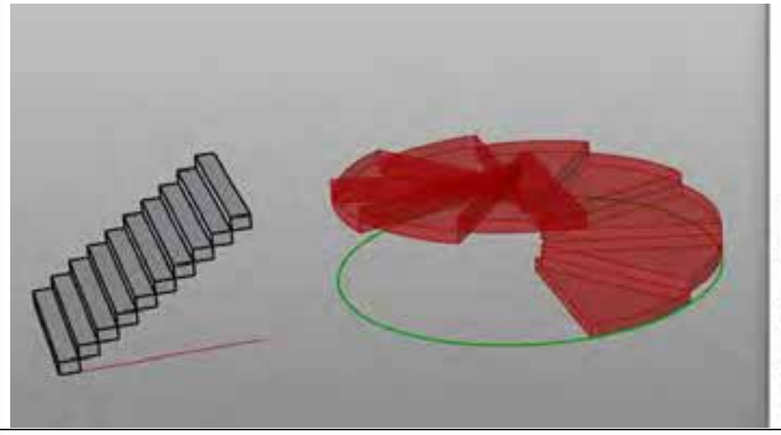
圆弧变形
Bend Deform

根据圆弧对物体进行弯曲变形，与 rhino 中的弯曲命令一直，注意物体起点与弧线起点对齐。

喷洒 **Splop** 	**将指定物体按照指定位置投射到曲面上，可以进行旋转。** 
延展 **Stretch** 	**将物体在直线（延展轴）对应长度范围的部分沿直线方向缩放。与 rhino 中的延展相同。** 
扭曲变形 **Twist** 	**将物体按照指定扭曲轴进行扭曲变形。无限大为布尔值，控制物体是严格在扭曲轴长度范围内扭曲，还是无限延长的范围扭曲。** 

曲线流动
Flow

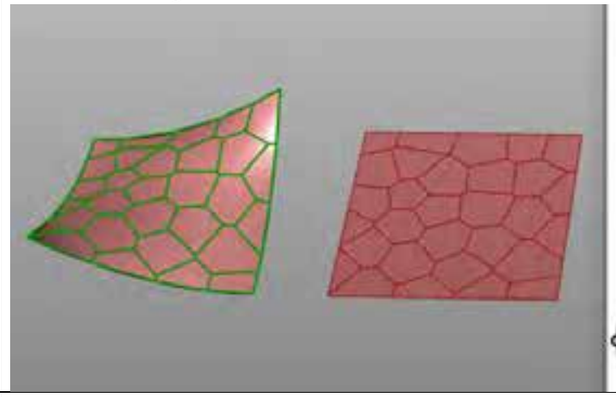
将物体按照指定曲线流动。同 rhino 中的曲线流动。

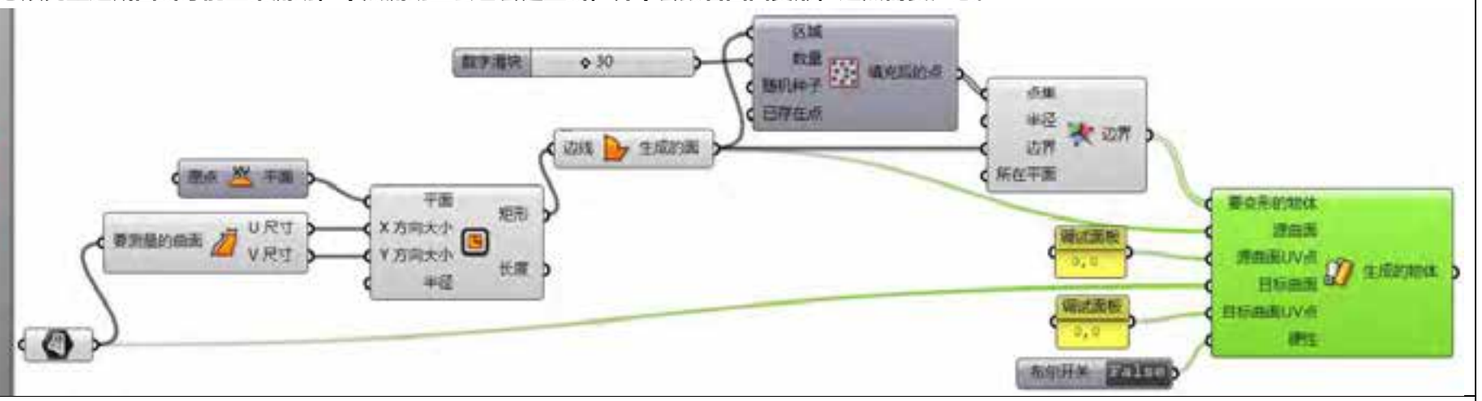


UV 点变形
Sporph

同 rhino 中的曲面流动，根据两个曲面指定的 UV 点位置对应，将**要变形的物体**投射到**目标曲面**上。如果要改变曲面流动方向，则要将曲面的方向上下翻转一下。

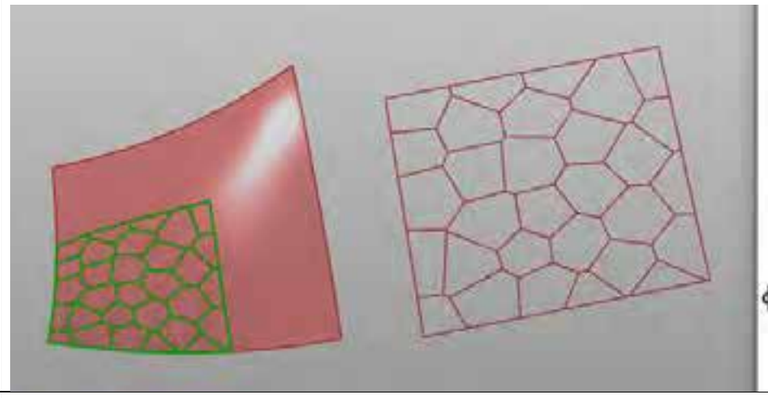
需要注意的是，gh 中如果流动物件是两点一阶的直线，那必须先重建成曲线才能正常流动，不然流动上去也会是直线，并不会跟着曲面变形，这点需要注意。

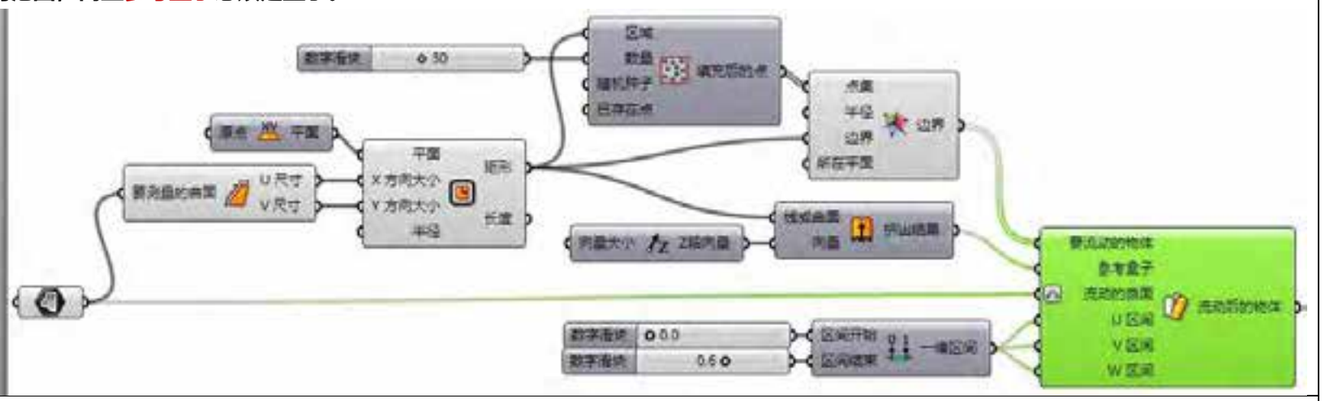




曲面流动
Surface Morph

与上方的 **UV 点变形**类似，只不过这个可以将物体流动到曲面上的指定区间范围，而且**参考盒子**必须是盒子。



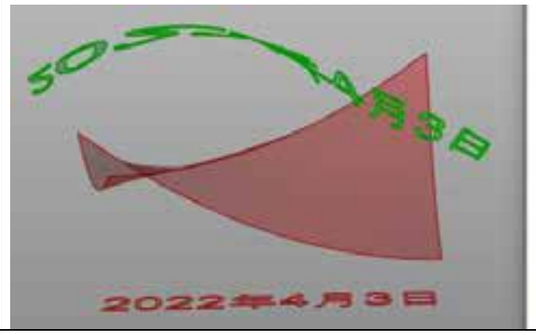


曲线镜像变形
Mirror Curve

使物体根据曲线镜像。**镜像切线**为布尔值，控制物体是严格按照曲线长度镜像，还是可以按照曲线无限延长的长度镜像（注意看下方两图的对比）。

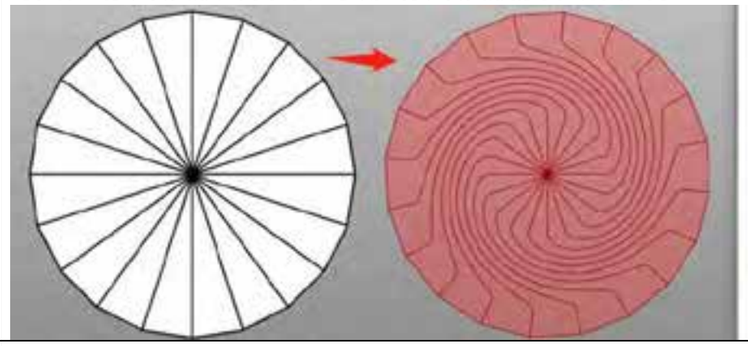
曲面镜像变形
Mirror Surface

使物体根据曲面镜像。镜像平面为布尔值，控制物体是严格按照曲面范围镜像，还是按照曲面无限延伸的范围镜像。



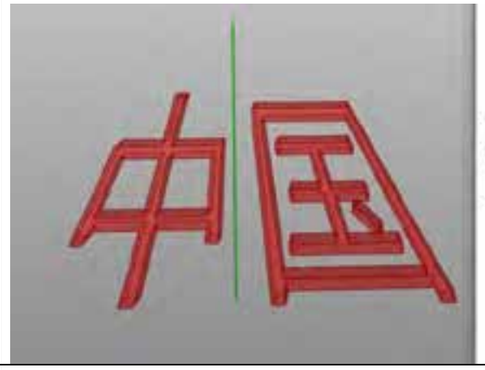
旋涡
Maelstrom

对物体进行绕转变形。两个半径可以控制绕转变形的半径范围。同 rhino 中的绕转。



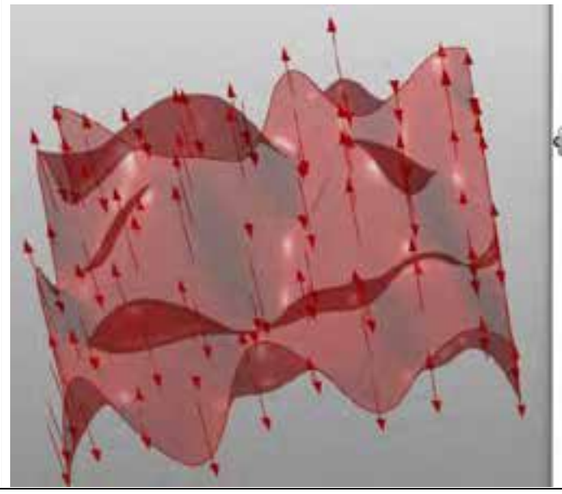
锥形变换
Taper

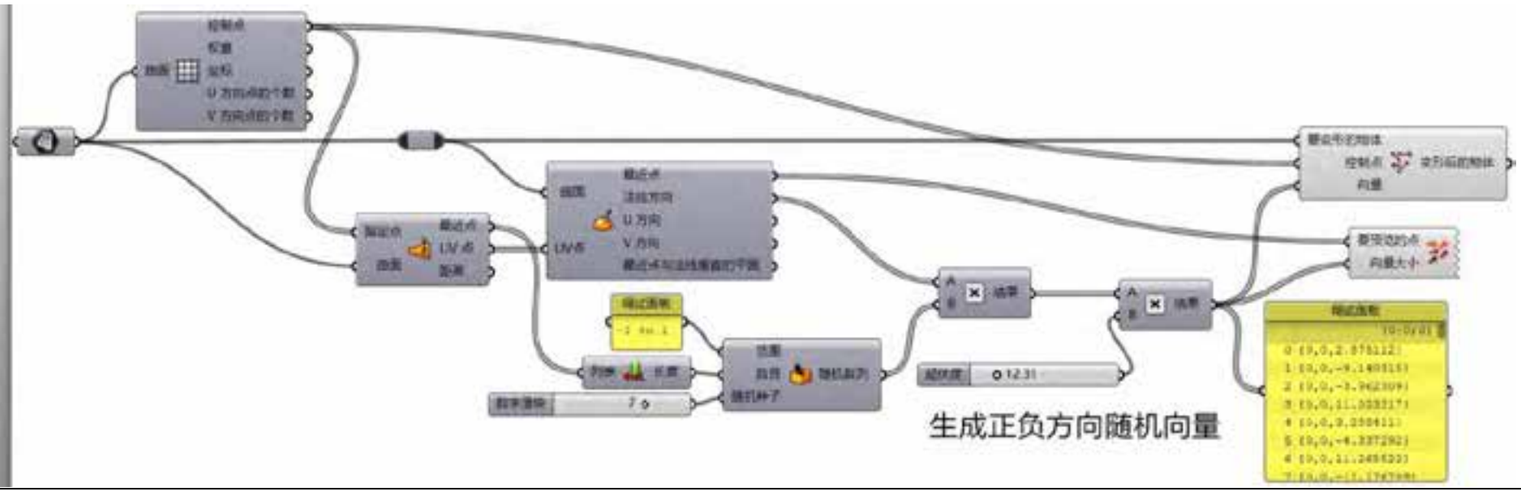
对物体进行锥状化变形，等同于 rhino 的锥状化。



控制点变形
Point Deform

通过移动物体控制点改变物体形状。（下图为获取矩形平面的控制点后，生成随机的正负方向向量，对矩形平面进行变形，可用于构建山体地形）

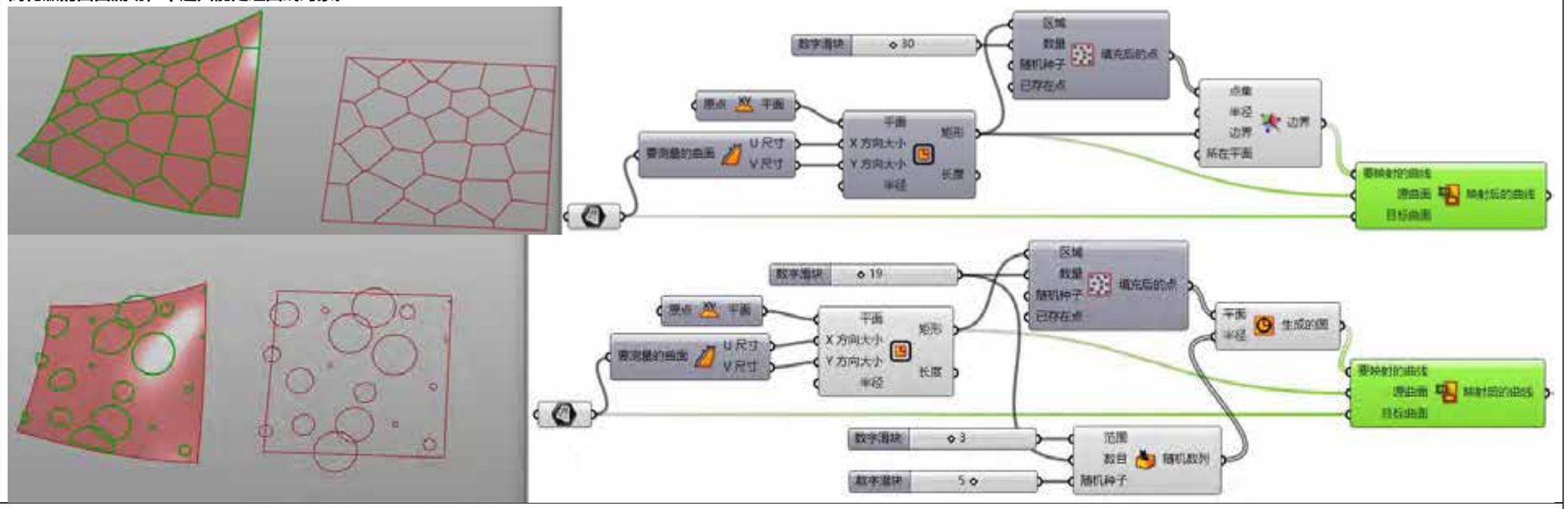




曲线映射到曲面

Map to Surface

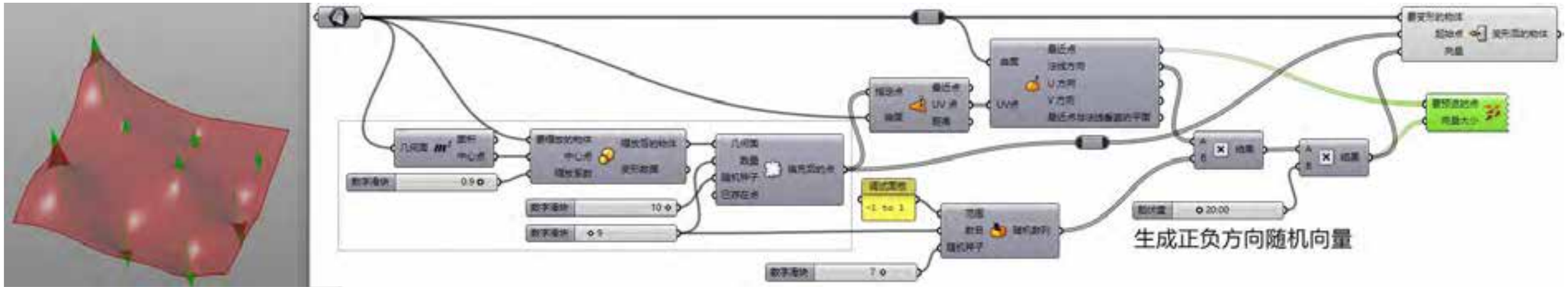
简化版的曲面流动，不过只能处理曲线对象。

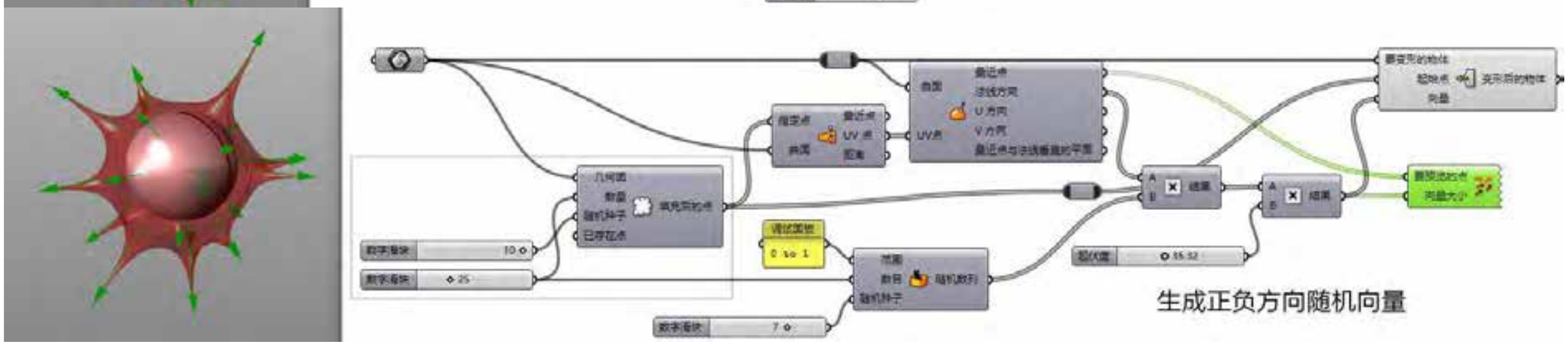


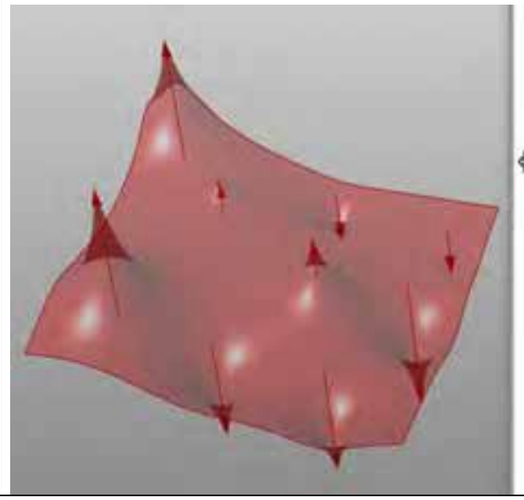
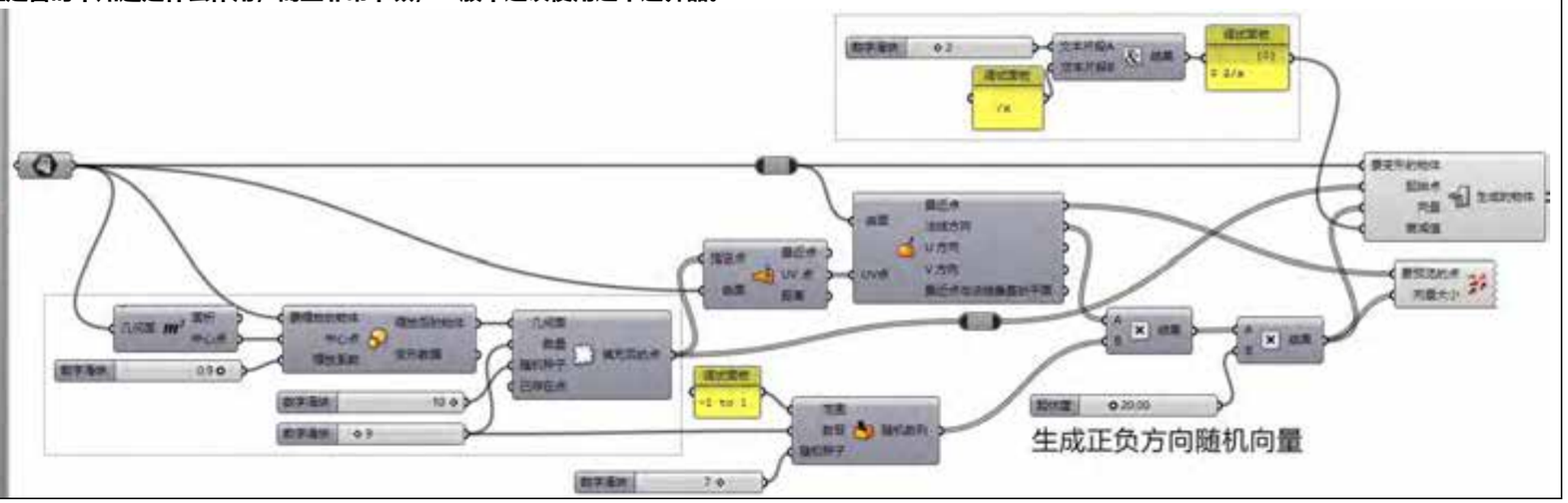
空间变形

Spatial Deform

与前面的控制点变形类似，只不过这个是根据物体上的点进行移动变形，所以变形后的曲面比较尖锐。





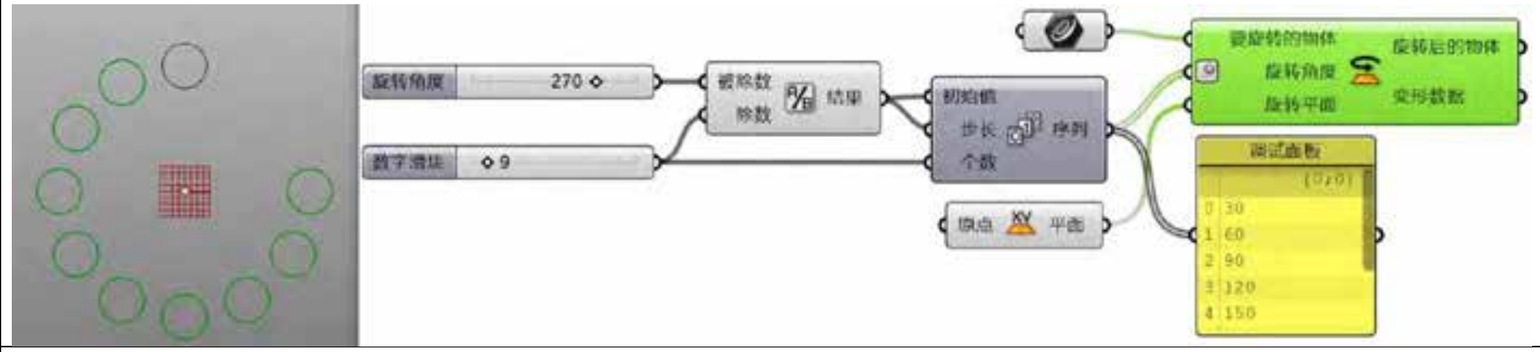
空间变形（自定义） **Spatial Deform (custom)** 	**与前面的空间变形运算器多了一个衰减值参数，但是暂时不知道是什么作用，而且非常卡顿，一般不建议使用这个运算器。**  

欧几里得 Euclidean

（变换）- 变形

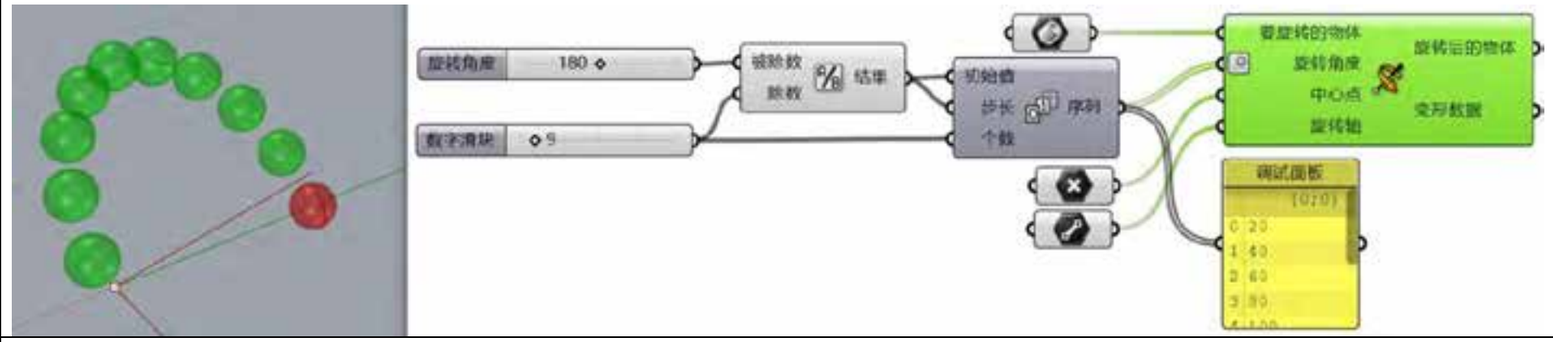
2D 旋转
Rotate

物体根据设定角度围绕指定平面的原点旋转。（习惯角度制的，要注意在旋转角度选项上右键选择 Degrees，否则默认是弧度制，下面的运算器同理）



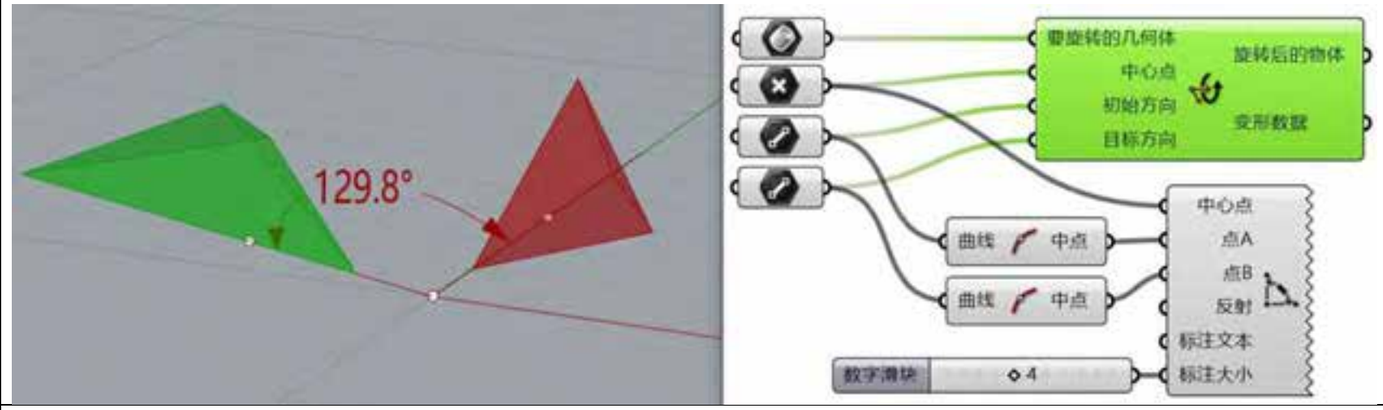
3D 旋转
Rotate 3D

将物体绕轴旋转。旋转轴为以指定的起点和直线方向的向量。



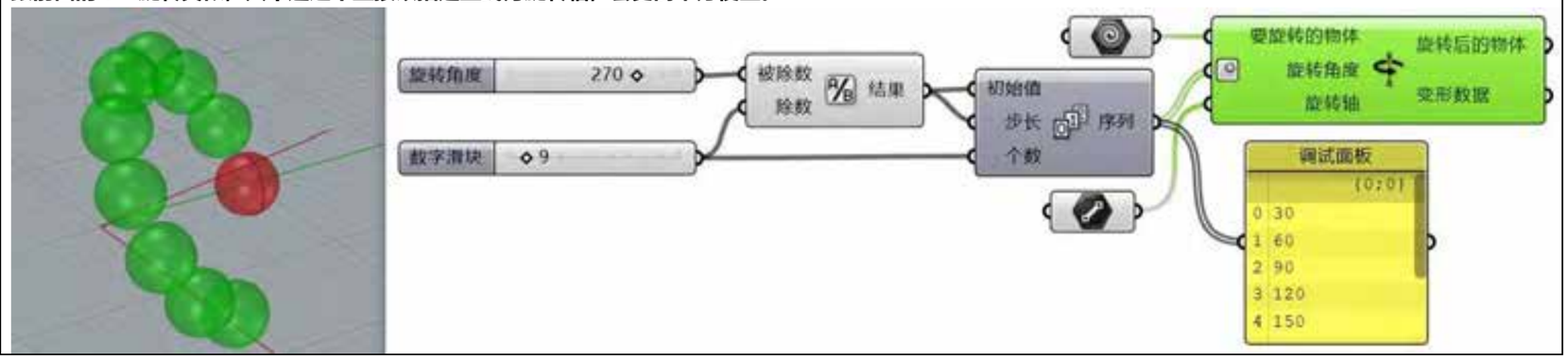
定向旋转
Rotate Direction

基于中心点及两个直线向量的夹角，对物体进行旋转。



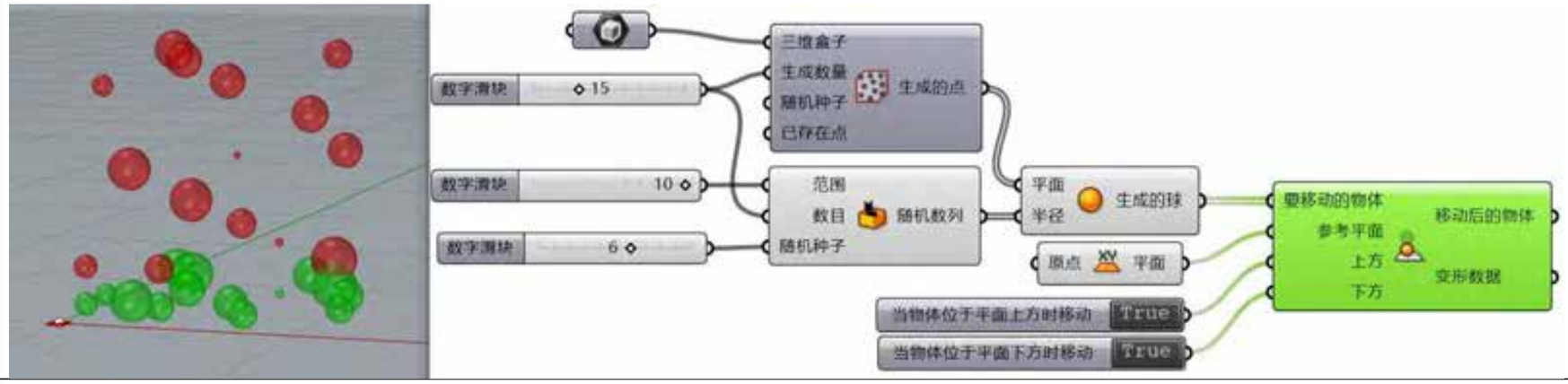
绕轴旋转
Rotate Axis

跟前面的 3D 旋转类似，只不过这个直接以指定直线为旋转轴，会更简单方便些。



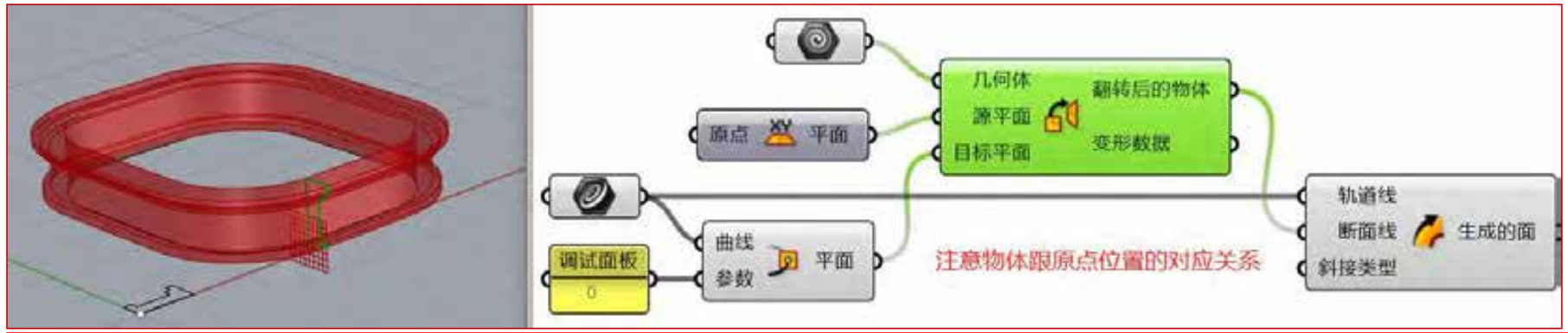
投影到平面
Move To Plane

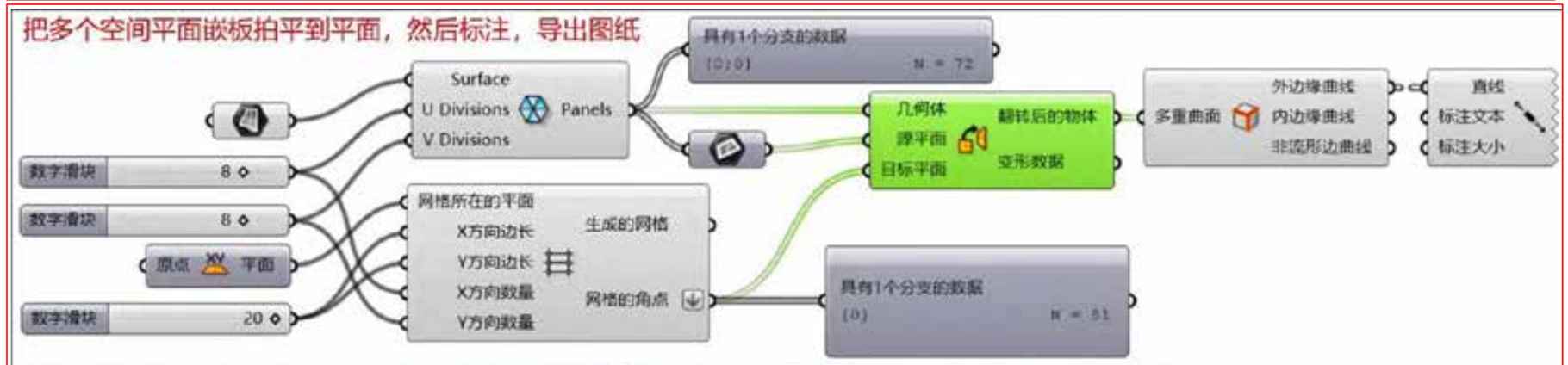
将物体移动到指定平面上。下方实例可以理解成大小不同的球体落在地面上。

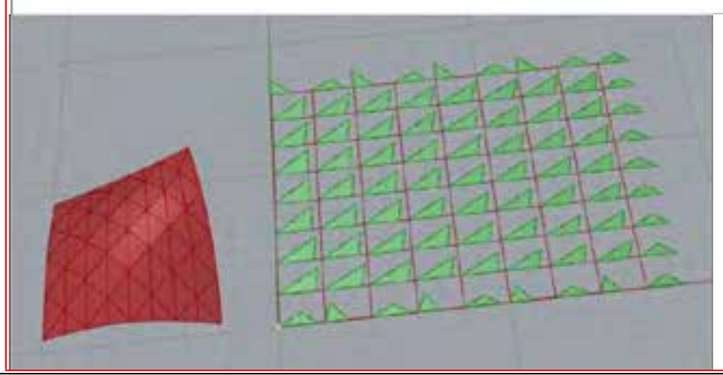


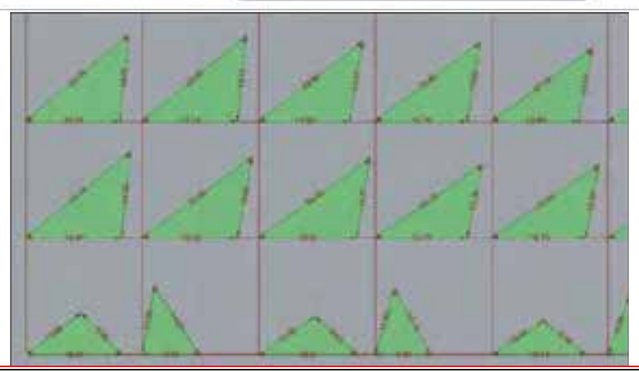
物体翻转
Orient

将物体根据两个平面的对应位置进行投射。结合单轨扫掠可用于做线脚、装饰槽等。



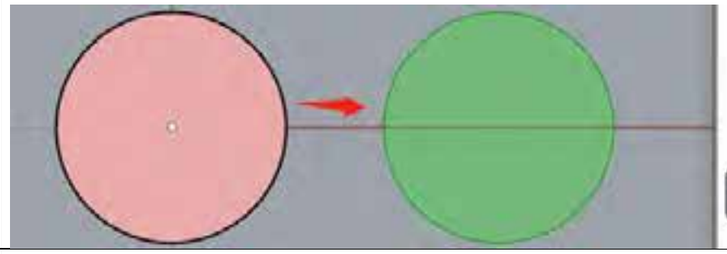






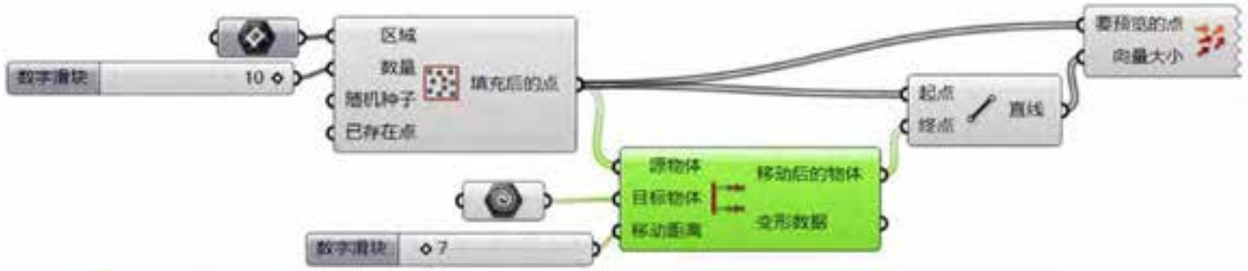
移动
Move

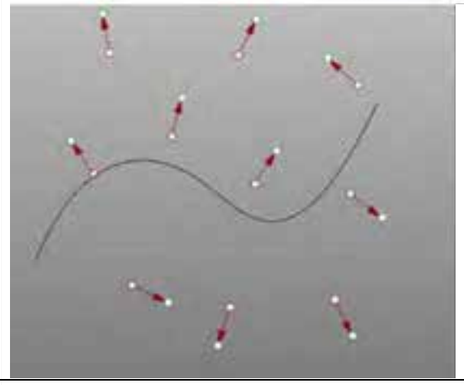
使物体沿指定大小和方向的向量移动。

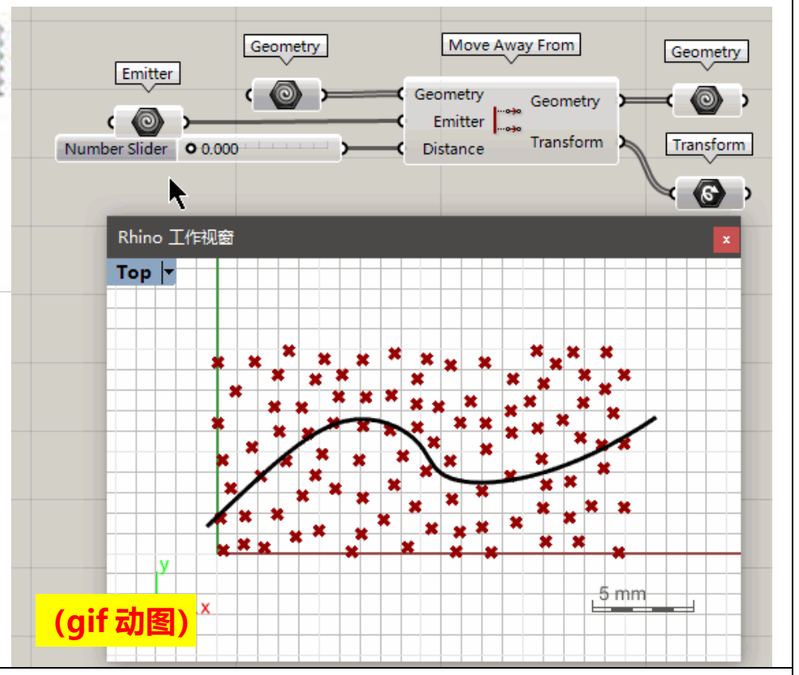


相对移动
Move Away From

使指定物体远离另一个物体。下图实例中，指定点沿这些点在曲线上最近点的法线方向远离曲线。

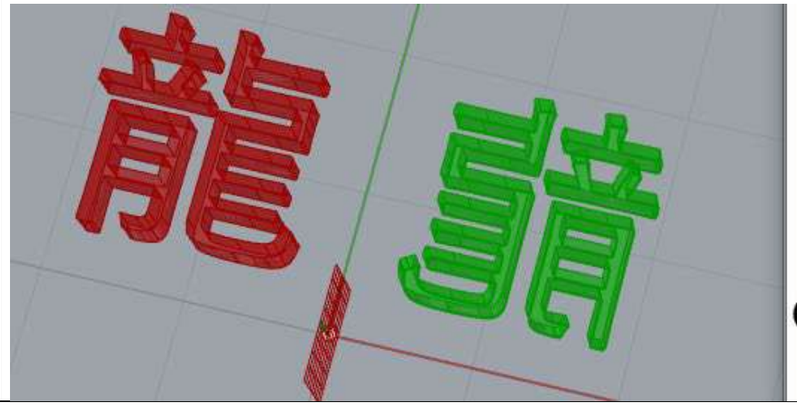






镜像
Mirror

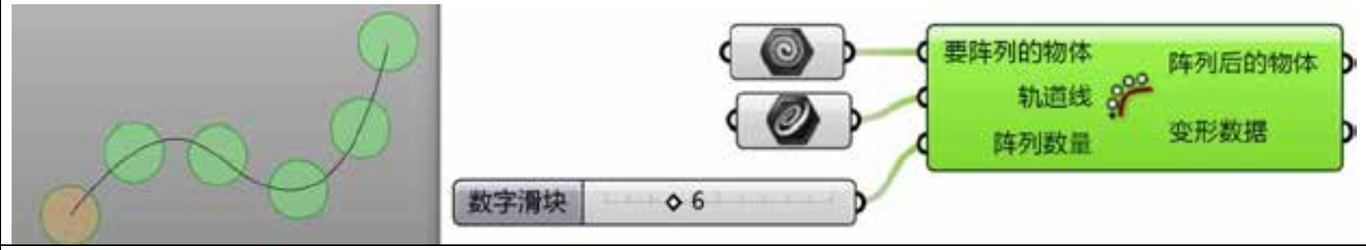
以指定平面对物体进行镜像。



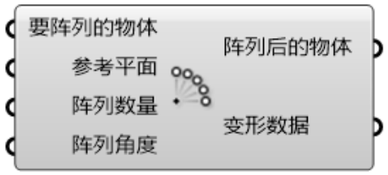
阵列 Array – 变形

曲线阵列
Curve Array

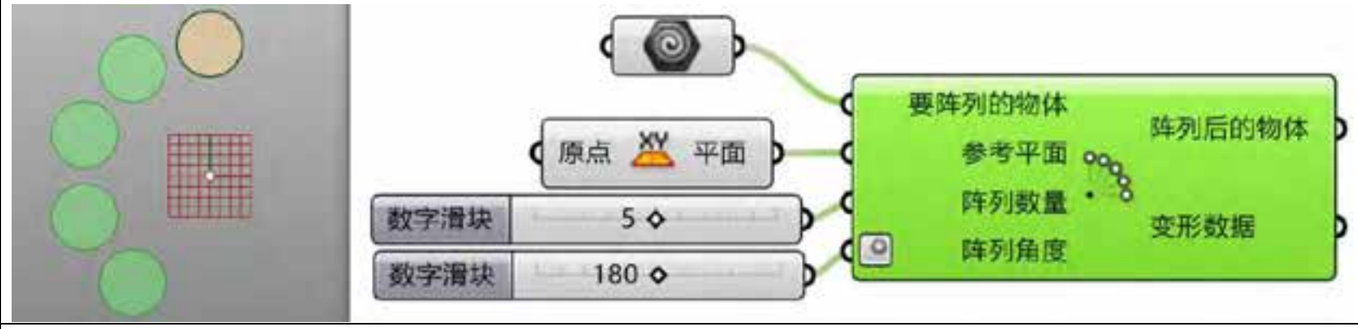
使物体沿曲线阵列。



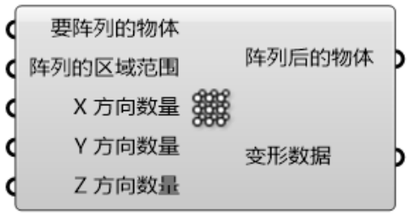
环形阵列
Polar Array



使物体围绕指定平面的原点为中心，进行环形阵列。

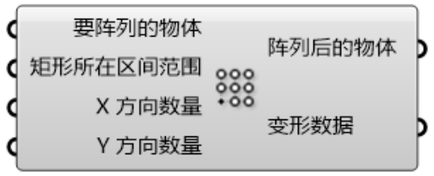


盒子阵列
Box Array



根据指定盒子的 XYZ 轴长度为对应 XYZ 轴方向的阵列间距，对物体进行阵列。

矩形阵列
Rectangular Array



以指定矩形的 XY 轴长度为对应 XY 轴方向阵列间距，对物体进行阵列。

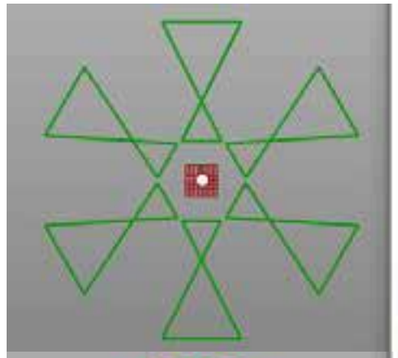
线性阵列
Linear Array

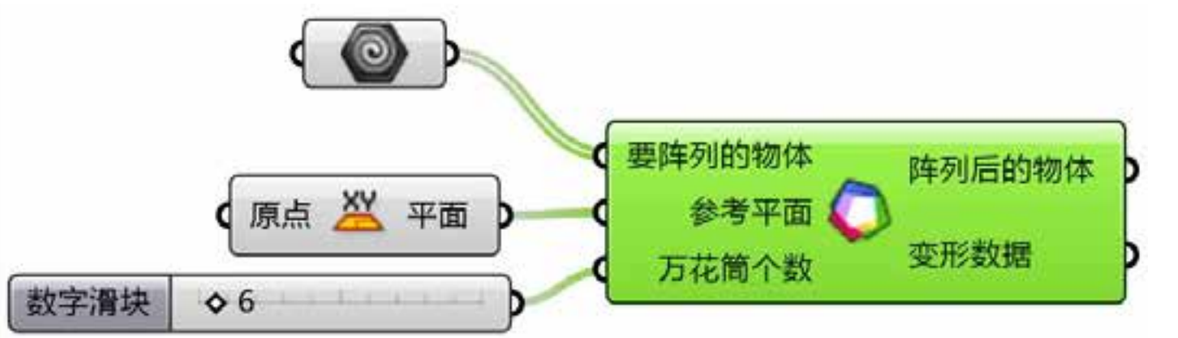
使物体以指定直线长度为阵列间距，沿直线方向阵列。

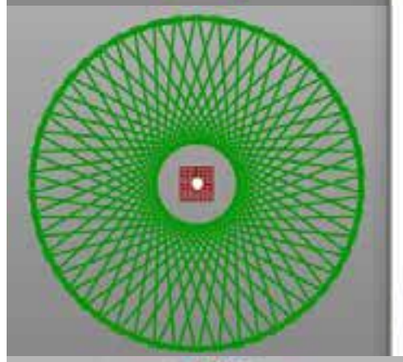
万花筒阵列

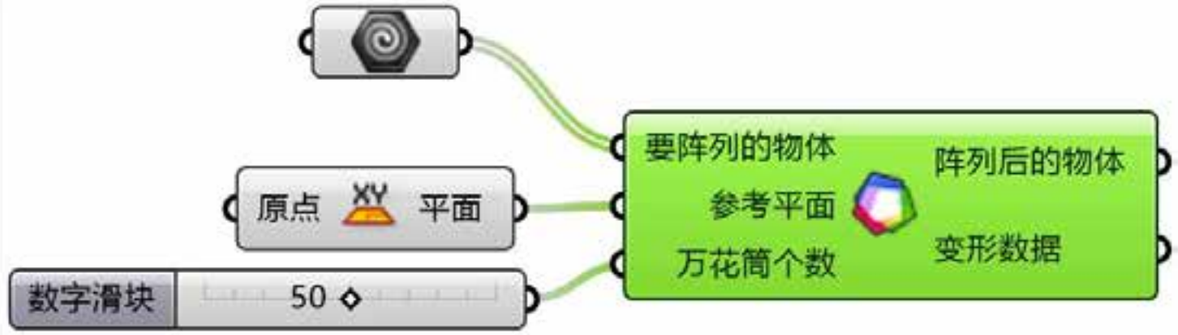
Kaleidoscope

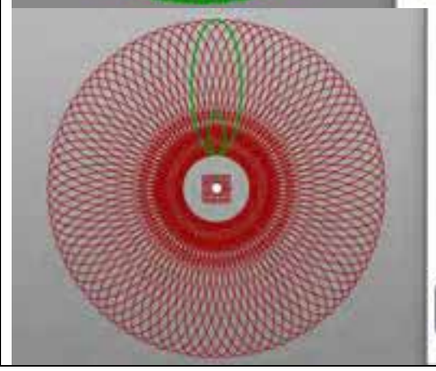
其实就是按照 360 度环形阵列，通过控制数量得到不同的图案。









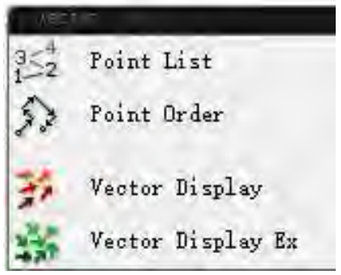


显示 Display

向量 Vector

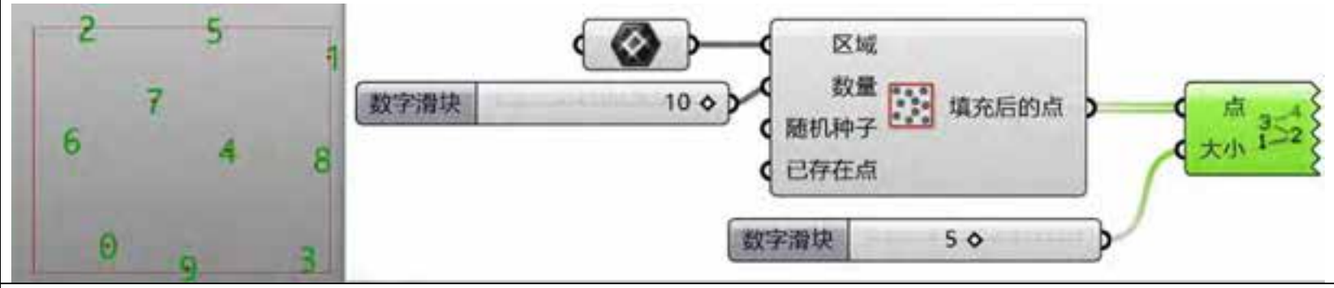
显示 Display

向量 Vector – 显示



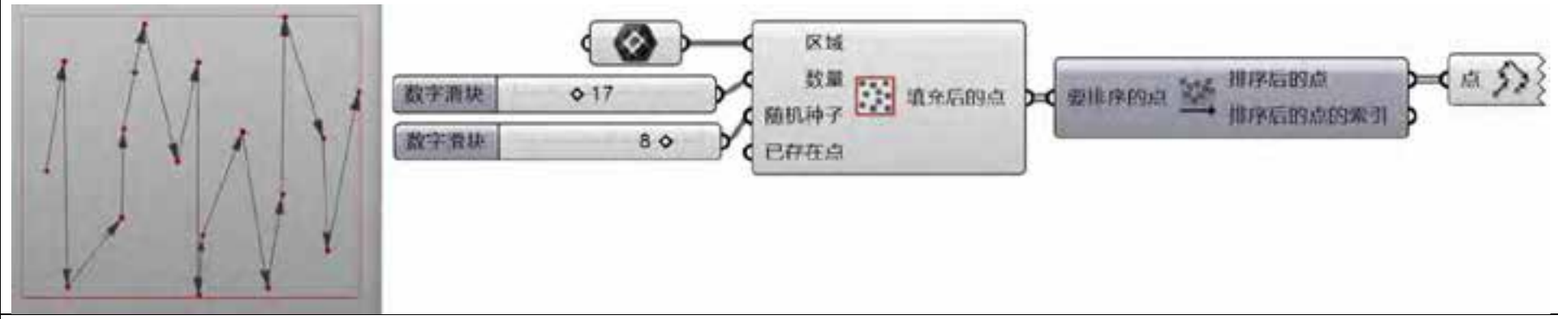
查看点
Point List

将点集按照列表顺序依次编号显示，可以调节显示字体的大小。



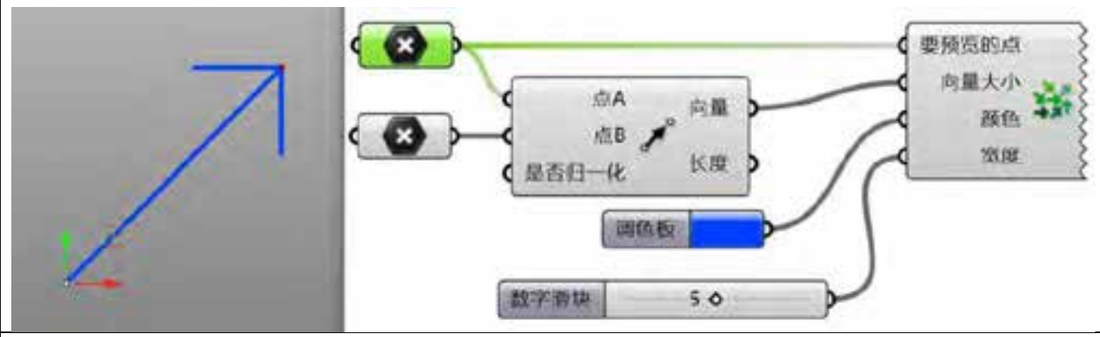
查看点排序
Point Order

通过箭头连接显示点集的顺序。

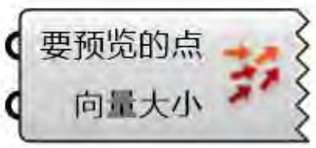


向量多彩显示
Vector Display Ex

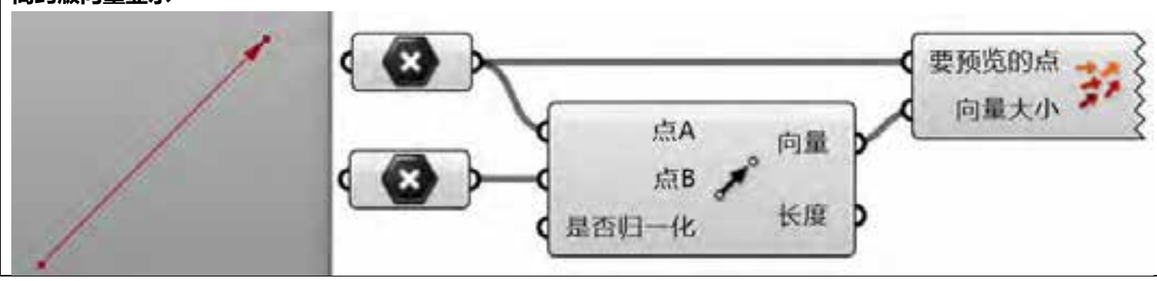
向量以箭头的形式显示，可以设定显示颜色和线型的宽度。



向量显示
Vector Display



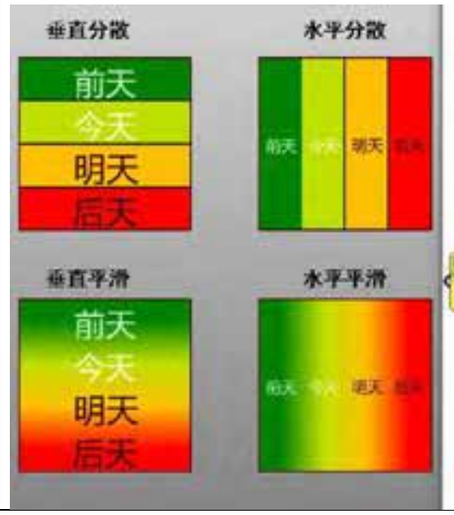
简约版向量显示

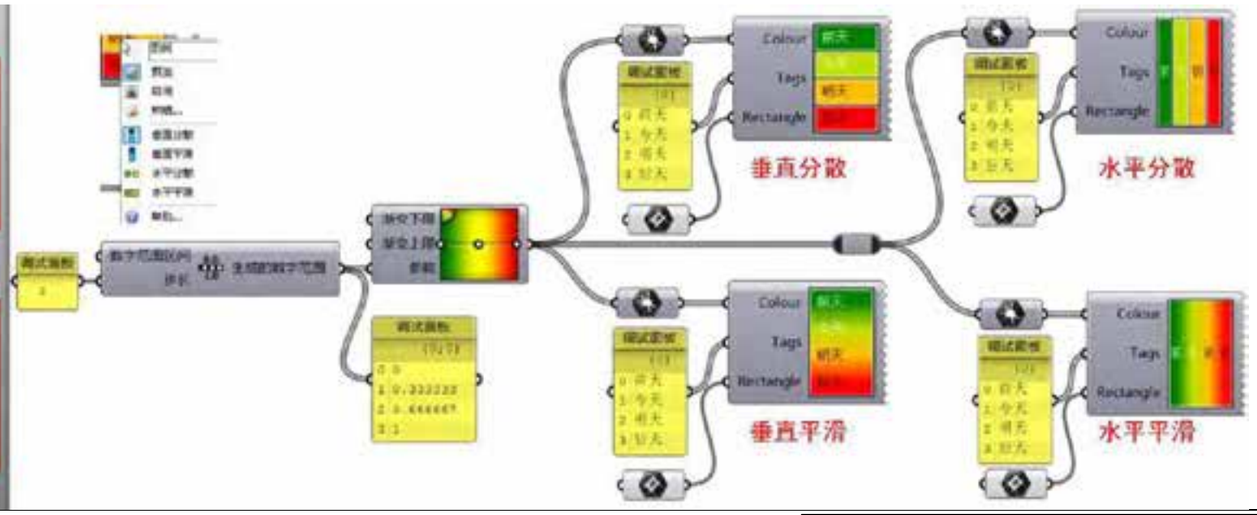


图像 Graphs - 显示

图例
Legend

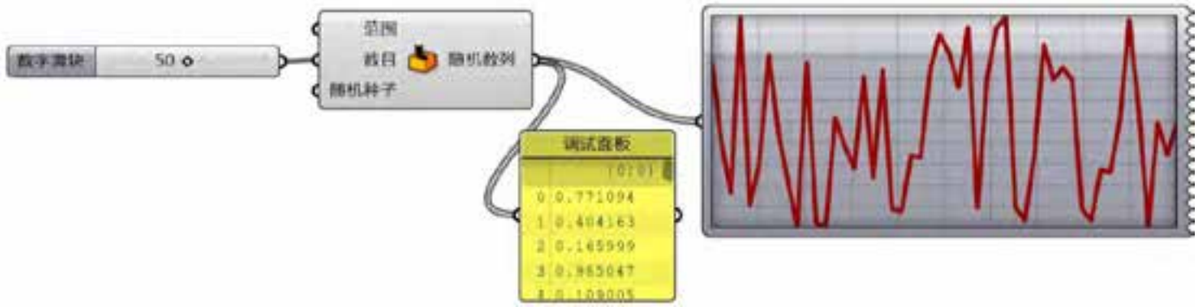
用于创建带颜色的文字标签图例。右键可以选择不同的图例和颜色的排列过度方式

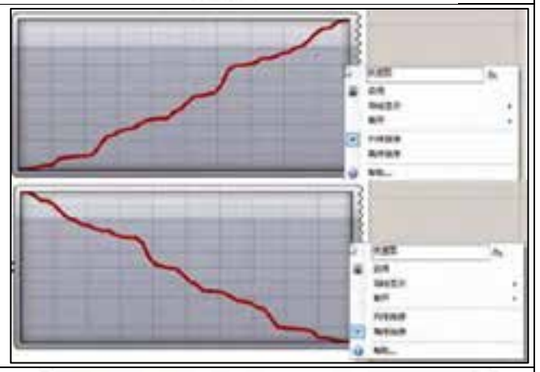




快速图
Quick Graph

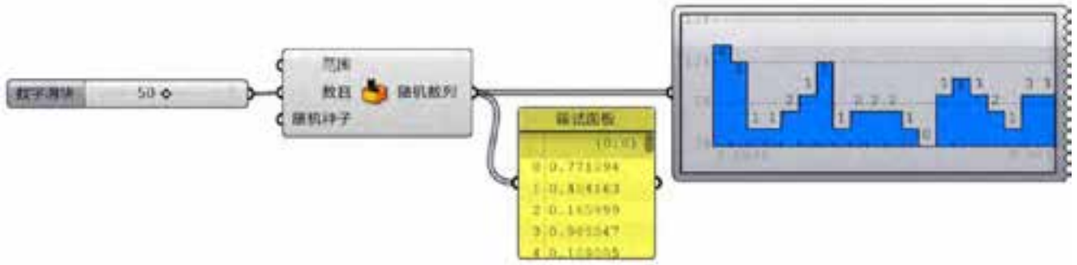
用于查看数据走势。可以右键选择数列中数据按升序或者降序重新排列显示。

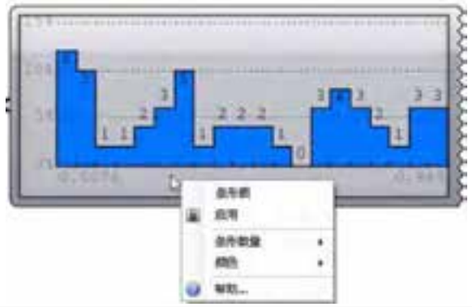




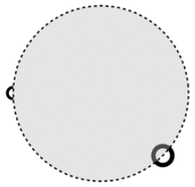
条形图
Bar Graph

柱状图，可以将数据分成指定数量的区间，然后显示每个区间内数据的数量。右键可以自定义条形数量及颜色。

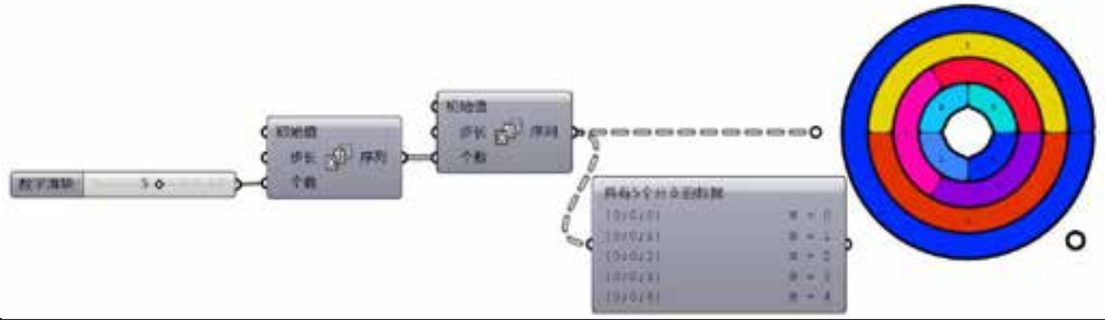


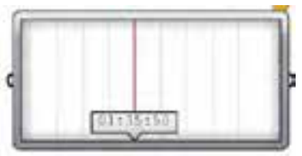
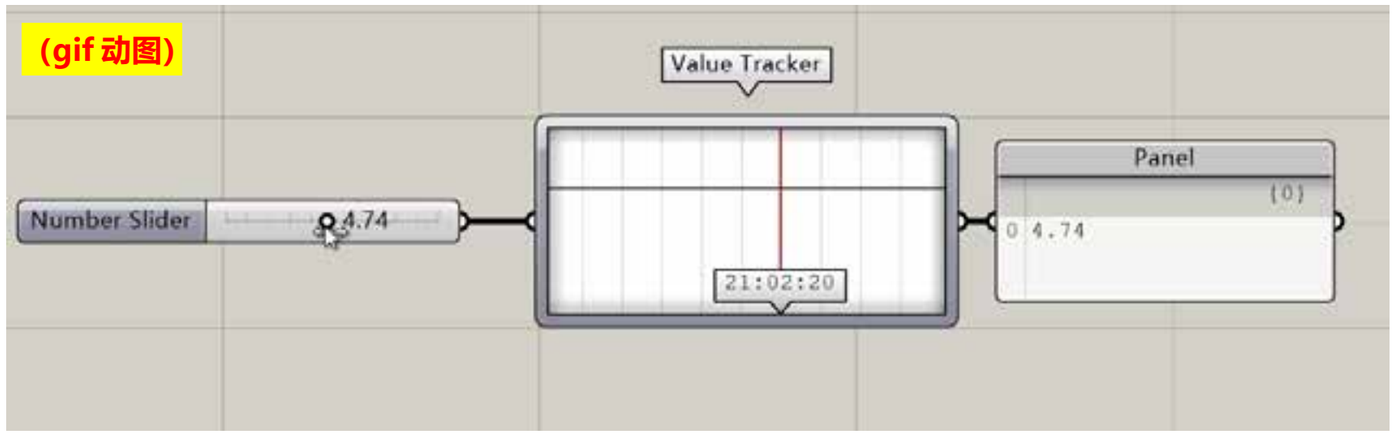
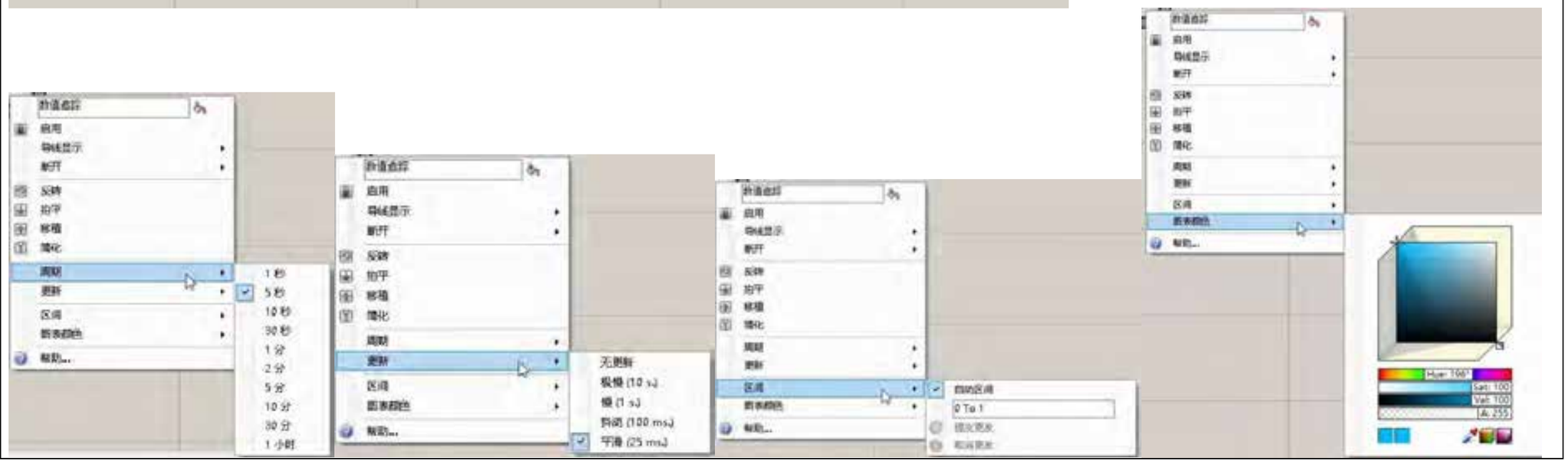


饼图
Pie Chart

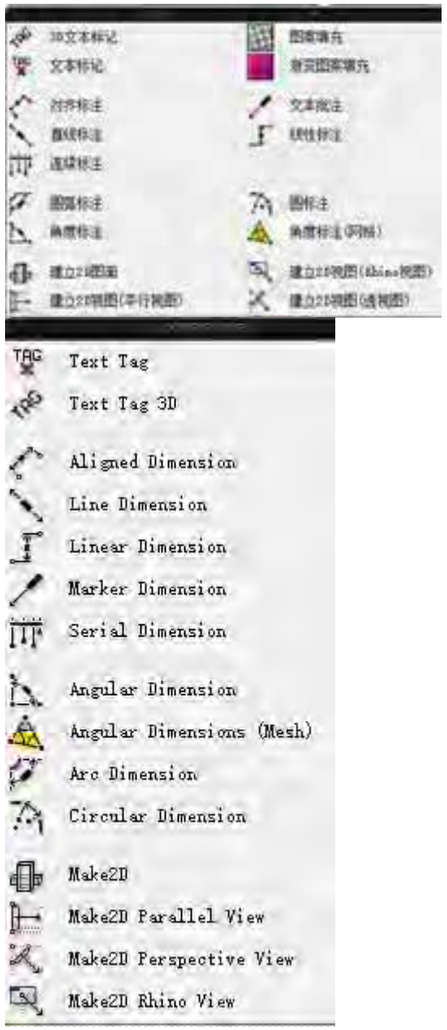


环形数据结构显示器，可以用这样比较花哨的形式来展现数据结构，一圈代表一个数据结构，实用性不大，看看就好。



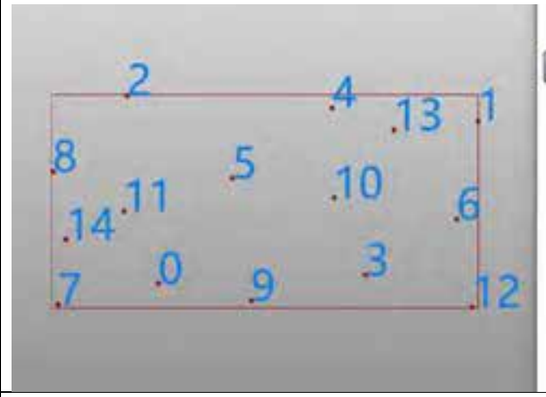
图片库 **Image Gallery** 	**图片自动播放器（可以理解为幻灯片）。双击运算器进入编辑界面，输入图片的网址或者直接把本地图片拖放到本地图像数据，设置图片的切换时间，运算器窗口会根据图片大小自动缩放。**
数值追踪 **Value Tracker** 	**记录数据变化的可视化运算器。右键运算器可以修改周期，即速度，数值越小，记录越快；设置数据刷新时间间隔；设置数据区间，一般都默认自动获取。设置图表颜色，暂时没发现实际作用是什么。**  

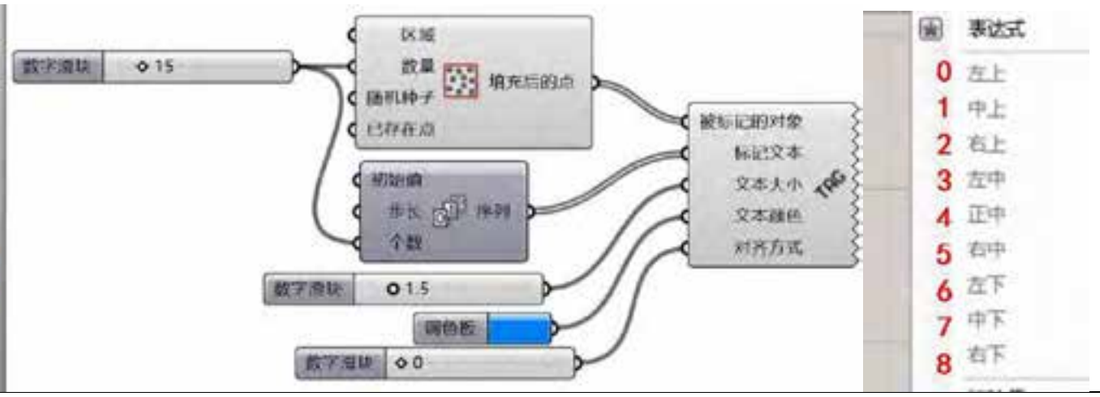
标注 Dimensions – 显示



3D 文本标记
Text Tag 3D

文字标注，可以设置文字大小及颜色，对齐方式有 9 种，分别对应数字 0-8。

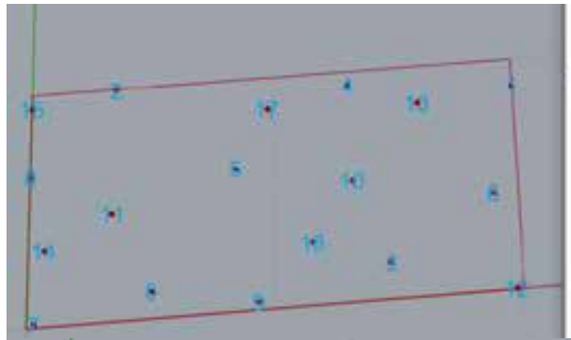


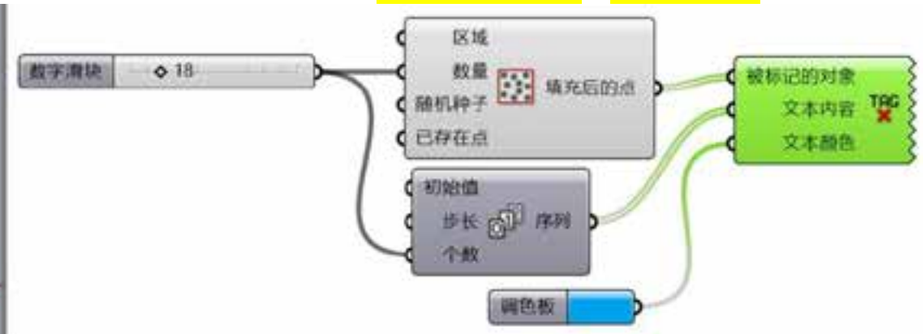


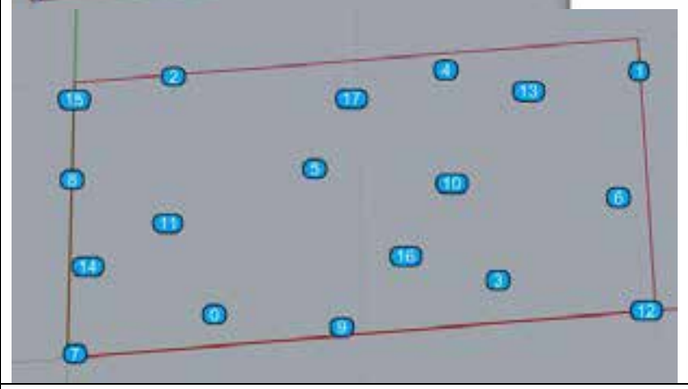
文本标记
Text Tag

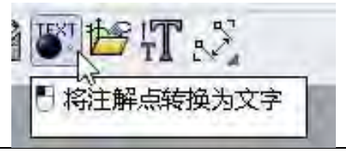
简化版文字标注，通过右键选择预设的文字大小。烘焙到 Rhino 中得到的是注释点。

注意注释点无法导出至 cad，注释点是 Rhino 特有的，想导出得使用 Rhino 出图工具栏中的将注解点转换为文字或 ConvertDots 命令将注释点转化成文字。



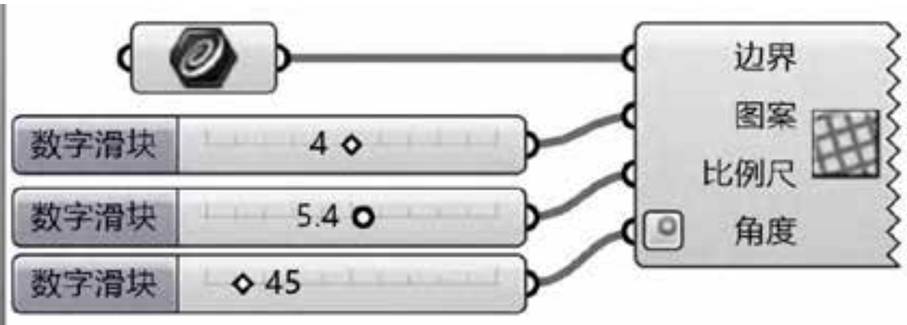


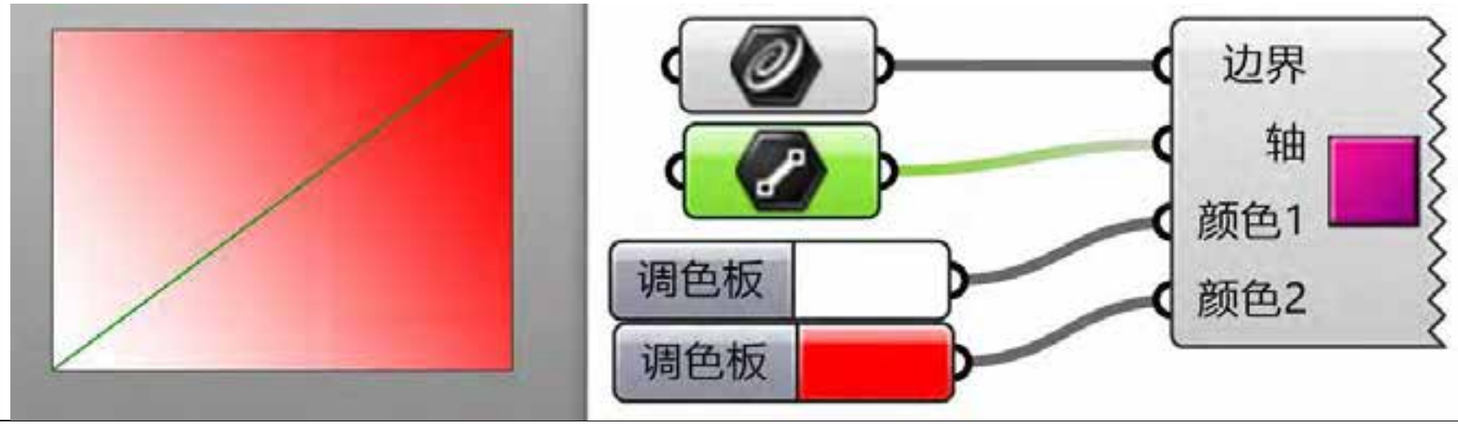
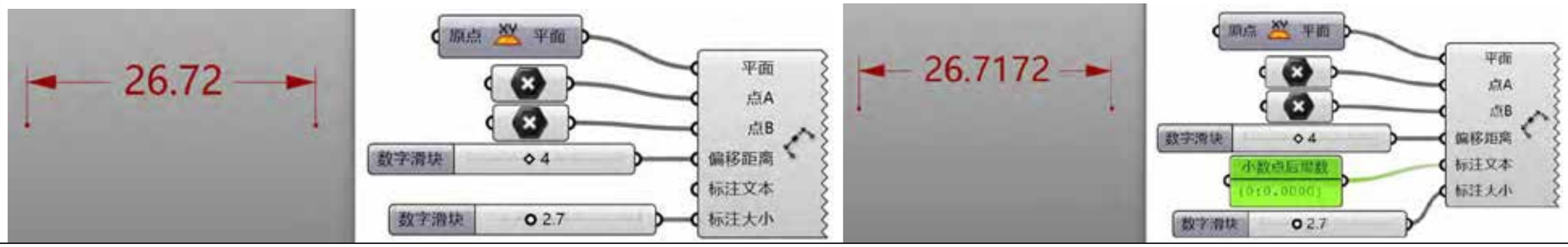


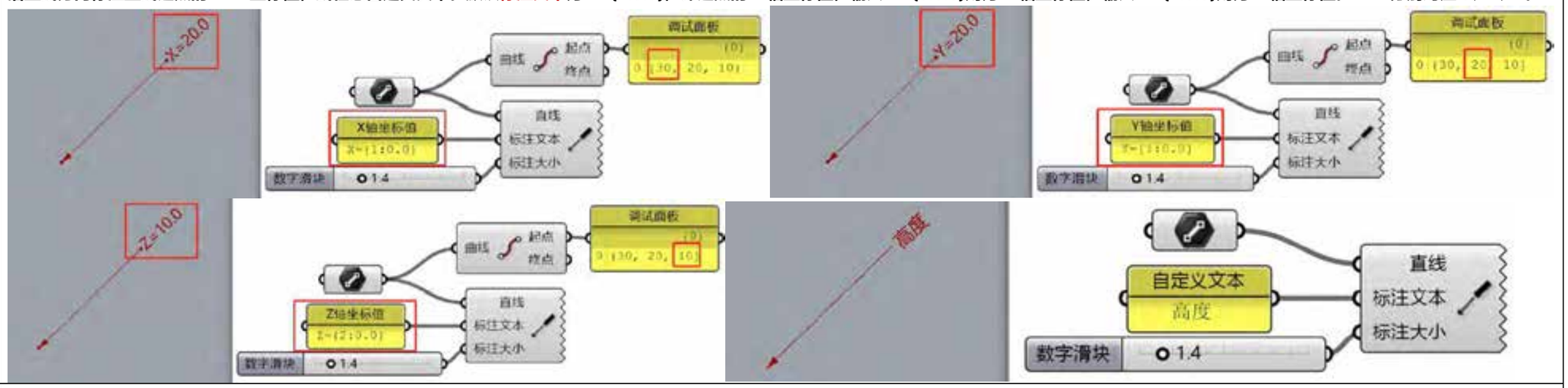
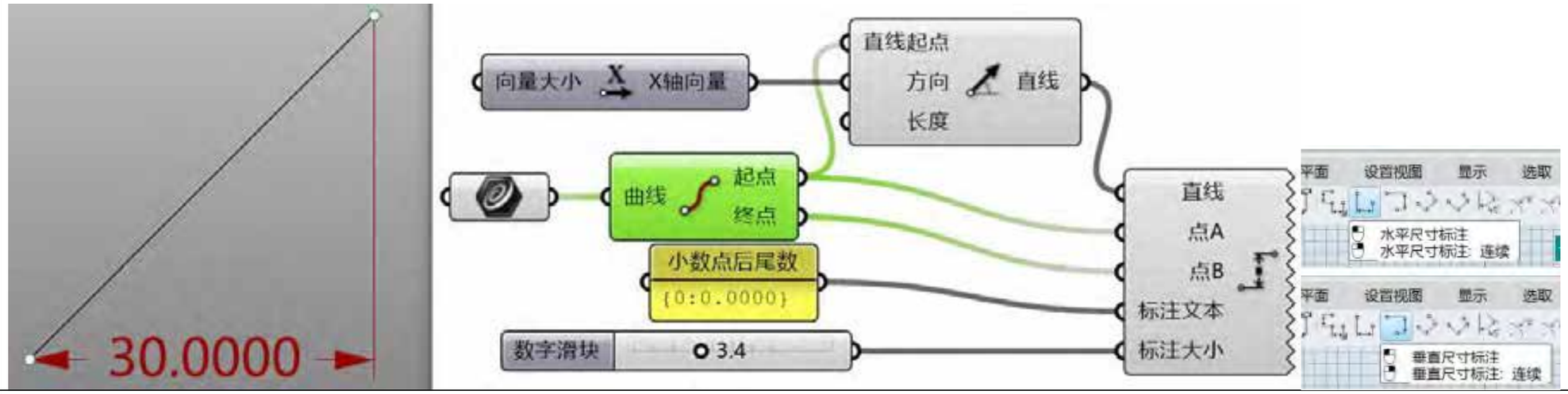
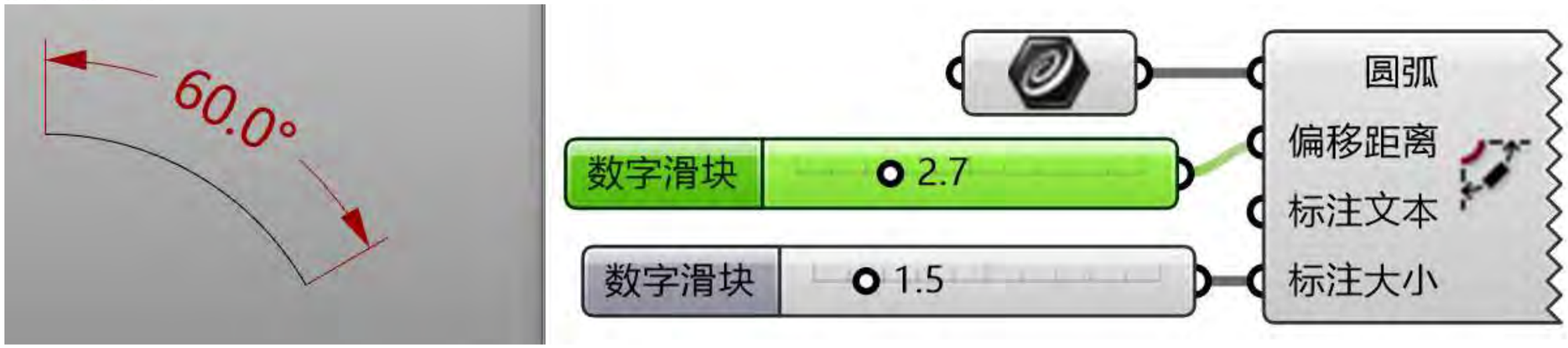


图案填充
Pattern Filling

对指定区域进行图案填充。图案右键可以设置预设的 9 种图案，对应数字 0-8。可以通过烘焙将结果返回到 Rhino 中。

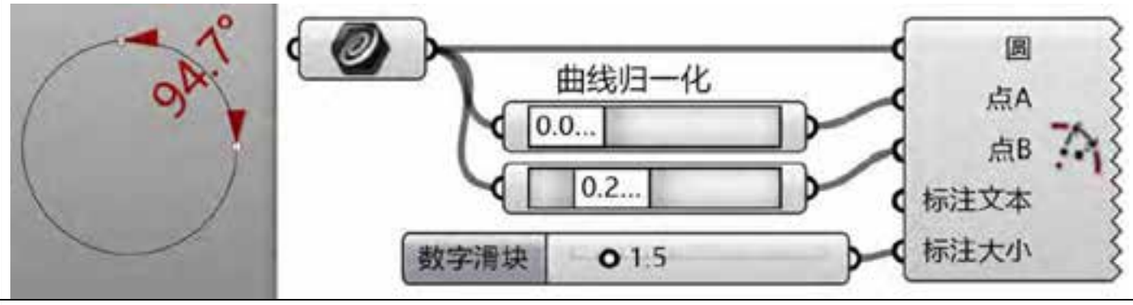


渐变图案填充 **Gradient hatch** 	**对指定区域按直线方向进行渐变色填充。可以通过烘焙将结果返回到 Rhino 中。** 
对齐标注 **Aligned Dimension** 	**两点距离标注。可以输入自定义标注文本，修改标注内容；也可以控制标注尺寸的小数点尾数。可以通过烘焙将标注返回到 Rhino 中，但是标注样式则由 rhino 当前标注样式决定。** 
直线标注 **Line Dimension** 	**直线尺寸标注。**
连续标注 **Serial Dimension** 	**点到直线距离的标注。**

文本标注 **Marker Dimension** 	**沿直线方向标注直线起点的 XYZ 坐标值，或任意自定义文本。默认标注文本为 Z={2:0.0}，即起点的 Z 轴坐标值；输入 X={0:0.0}则为 X 轴坐标值；输入 Y={1:0.0}则为 Y 轴坐标值。XYZ 分别对应 0、1、2。** 
线性标注 **Linear Dimension** 	**测量两点在指定直线上的投影点之间的距离。等同于 rhino 以及 cad 中的垂直标注和水平标注，但是这个运算器自由度更大。** 
圆弧标注 **Arc Dimension** 	**标注圆弧的圆心角。** 

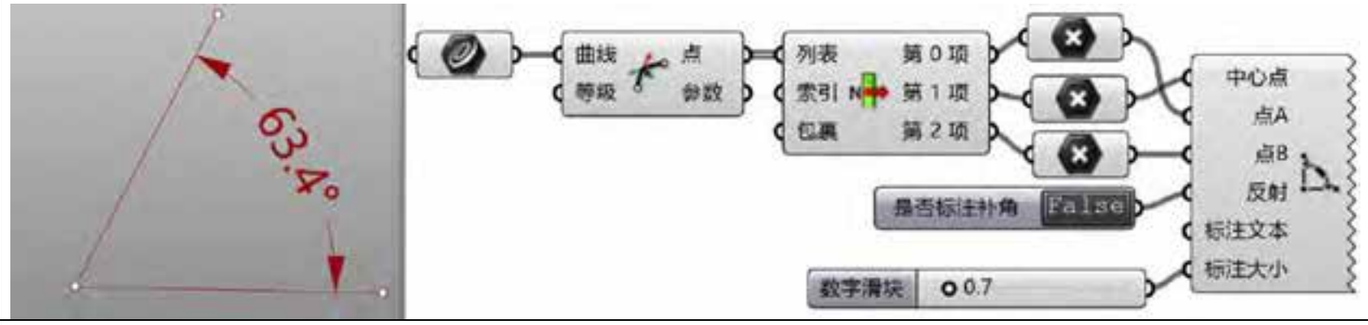
圆标注
Circular Dimension

标注圆上指定两个点之间圆弧对应的圆心角。注意点 A 到点 B 为逆时针方向。



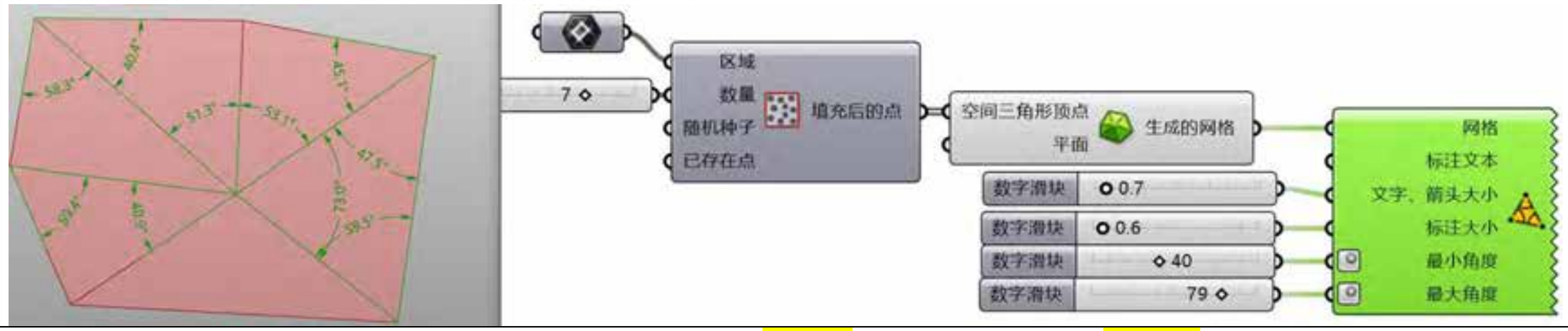
角度标注
Angular Dimension

标注三点连线的夹角。



角度标注（网格）
Angular Dimensions (Mesh)

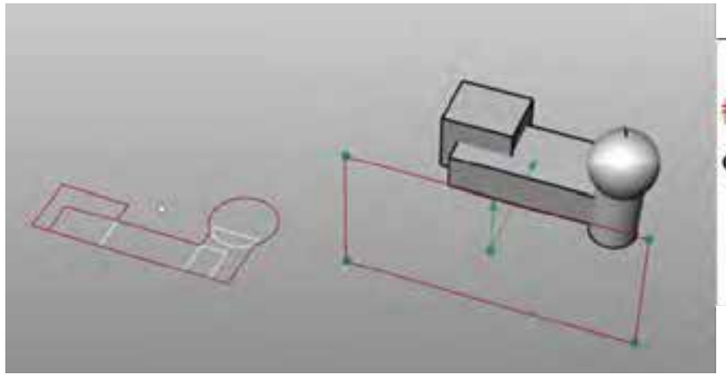
标注网格的各个角度，可以通过设定最小角度和最大角度筛选标注角度的取值范围。

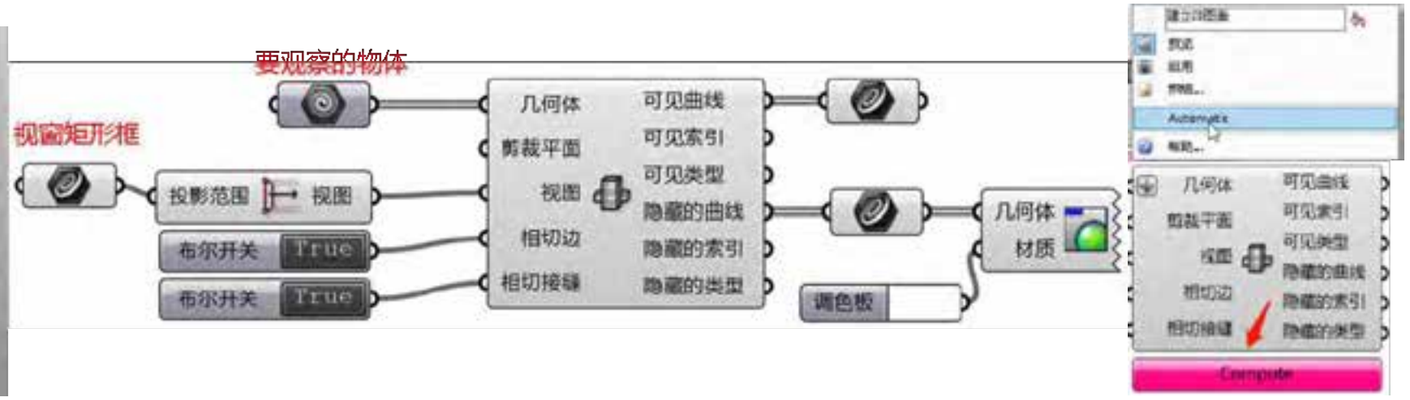


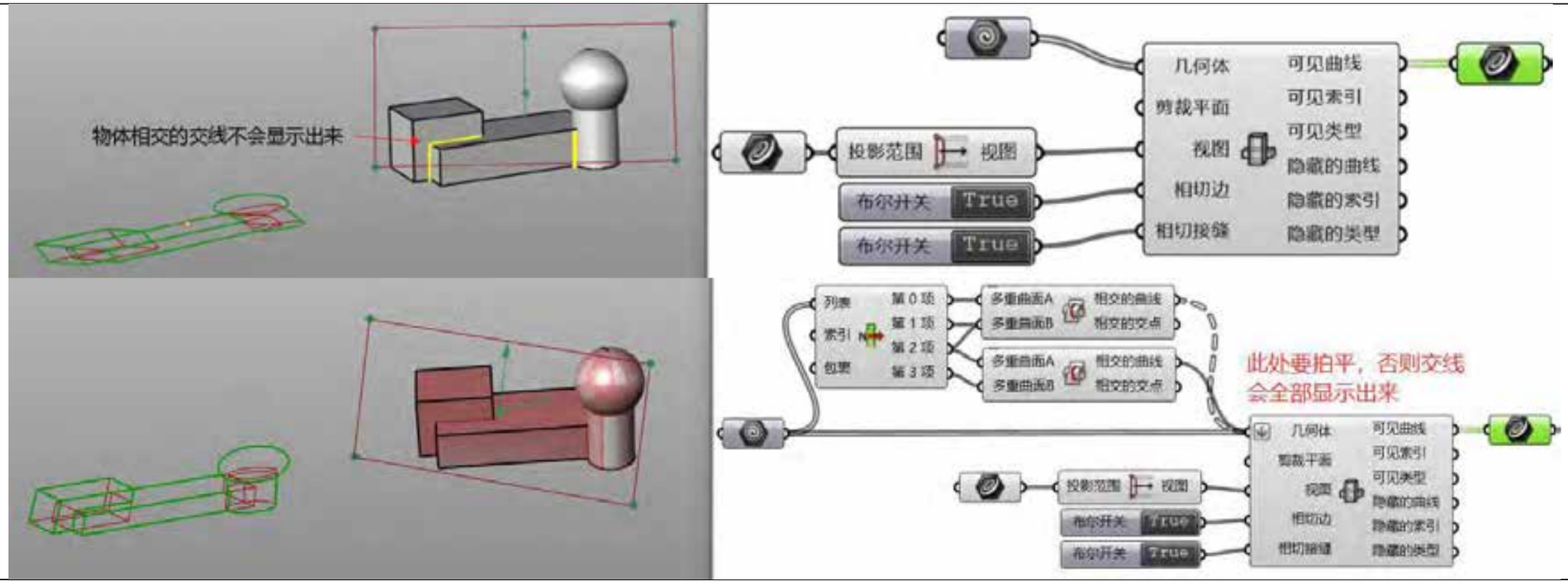
建立 2D 图面
Make2D

建立 2D 视图
(平行视图)
Make2D Parallel View

根据设定的矩形视窗（要注意矩形的方向），将视窗内的物体导出线性图纸。设置好后点击 compute 计算输出结果。或者右键选择 automatic，会自动更新结果。
但是这个运算器不会计算显示物体的交线，这时就需要求出相交线，

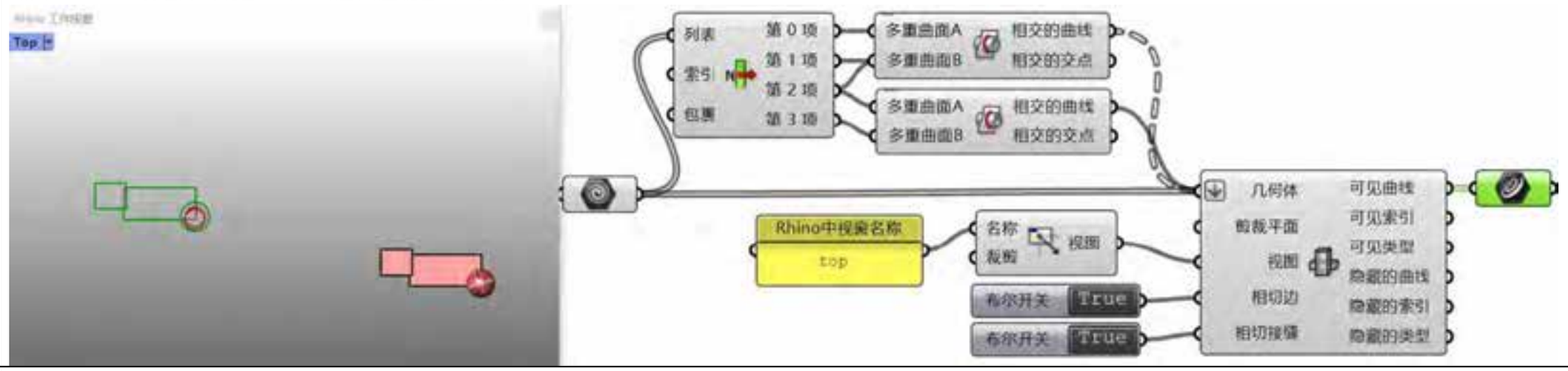






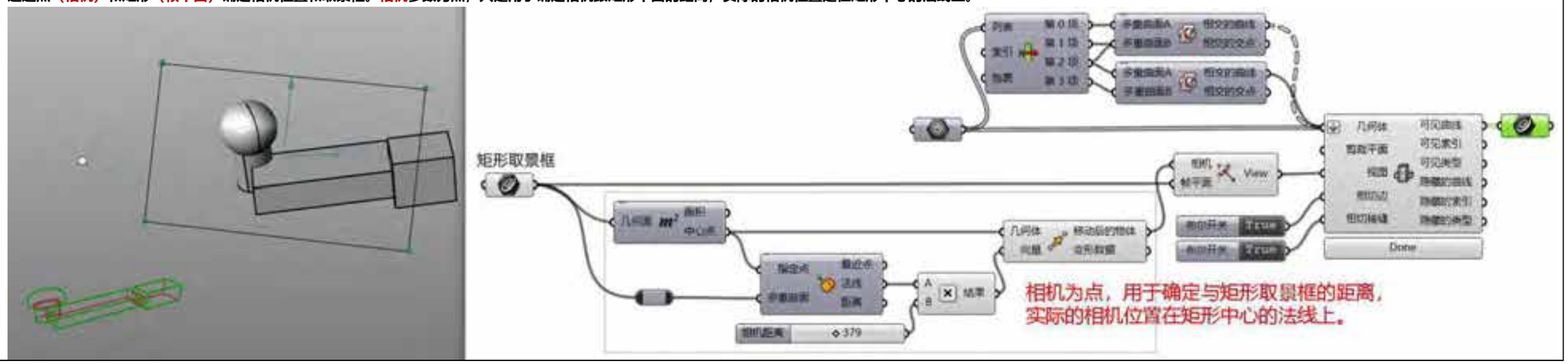
建立 2D 视图
(Rhino 视图)
Make2D Rhino View

输入 Rhino 中设定好的视图名称，输出对应视图的结果。



建立 2D 视图（透视图）
Make2D Perspective View

通过点（相机）和矩形（帧平面）确定相机位置和取景框。相机参数为点，只是用于确定相机跟矩形平面的距离，实际的相机位置是在矩形中心的法线上。



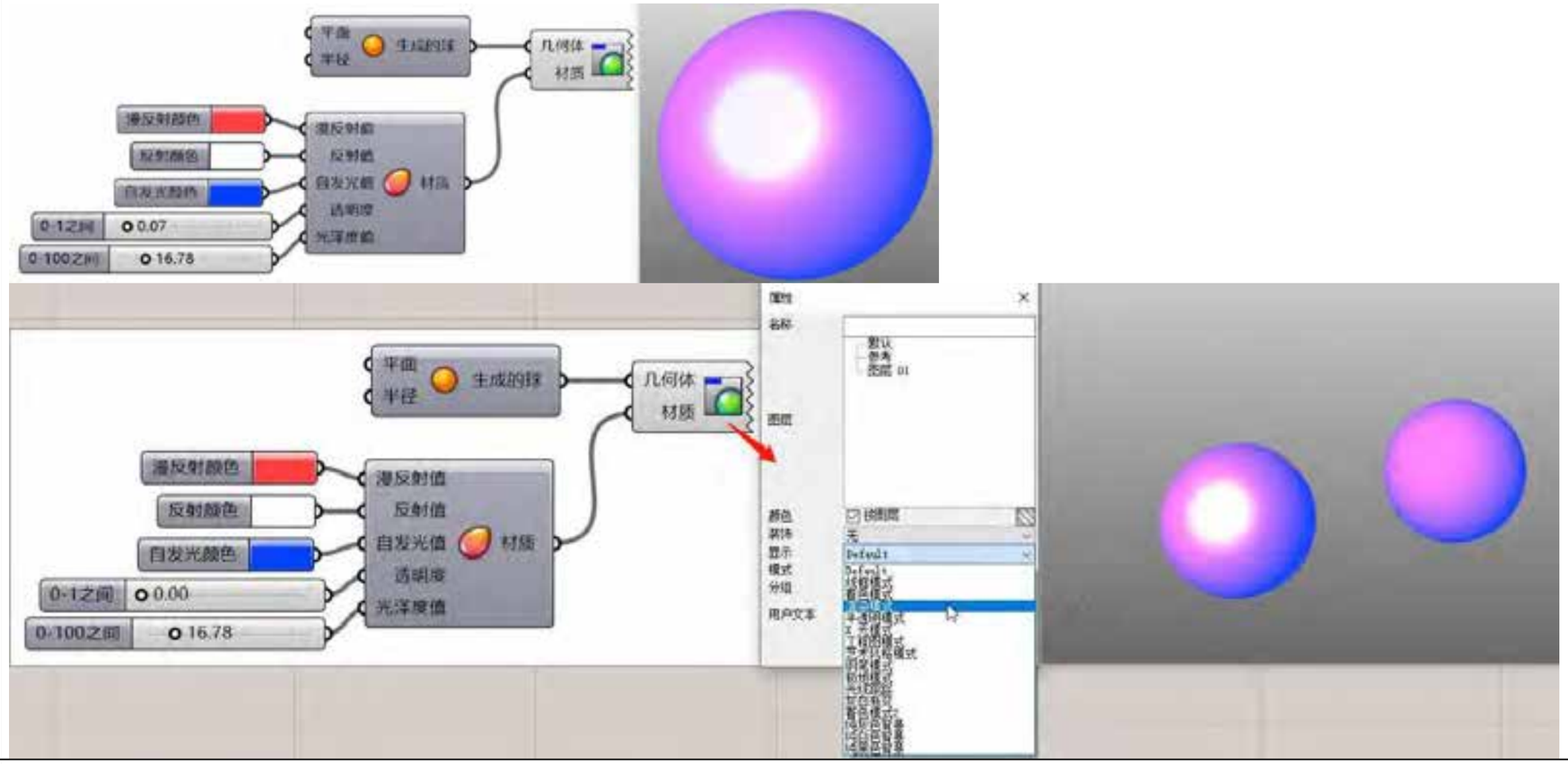
预览 Preview

– 显示

创建材质
Create Material

通过设定相关参数创建颜色材质，通过自定义预览显示效果。

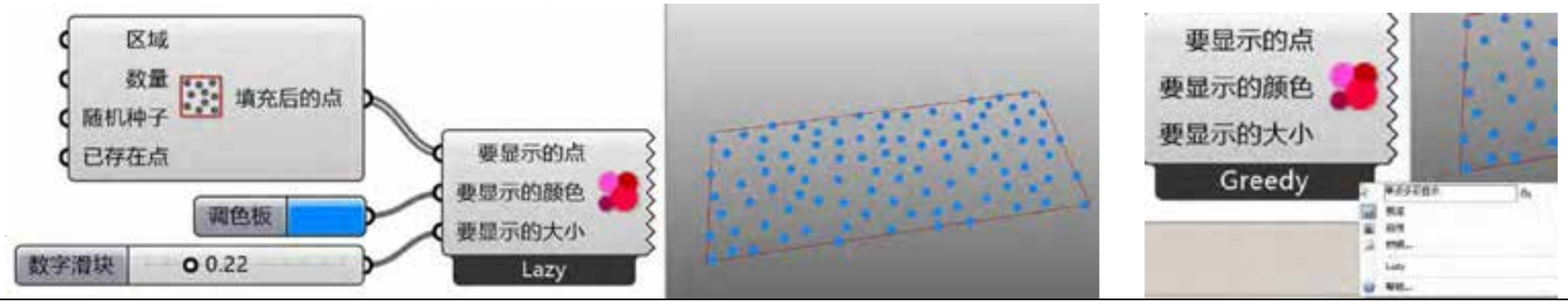
可以通过烘焙将材质输出到 Rhino 中，注意烘焙属性选择不同的显示模式会烘焙出对应模式的材质显示效果；默认 Default 模式，rhino 视窗要切换到渲染模式才能看到材质颜色。



自定义预览
Custom Preview

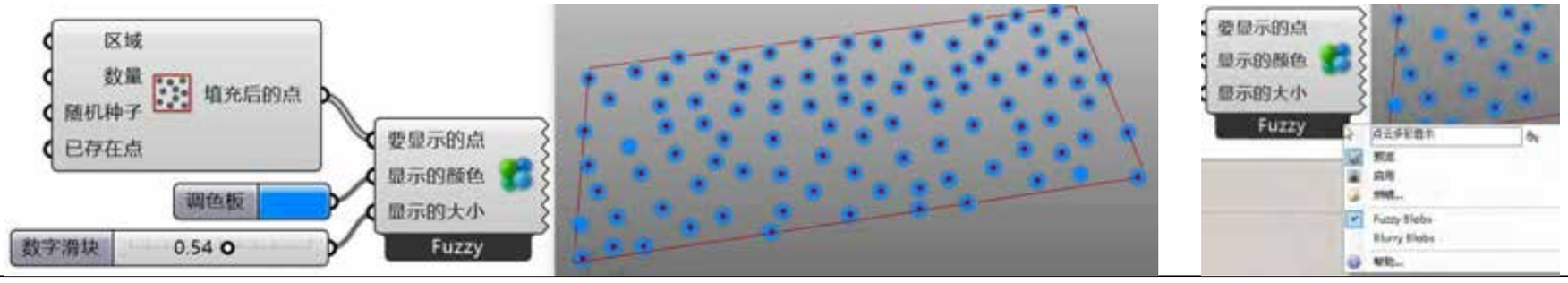
单点多彩显示
Dot Display

调整点的显示颜色和大小。右键可以切换 Lazy 或 Greedy，用来控制修改 Size 时候的显示速度，Greedy 的话修改大小数值的时候会比较卡，但是精度高，反之亦然 。



点云多彩显示
Cloud Display

点云显示样式，可以右键修改显示样式，有两种可供选择。



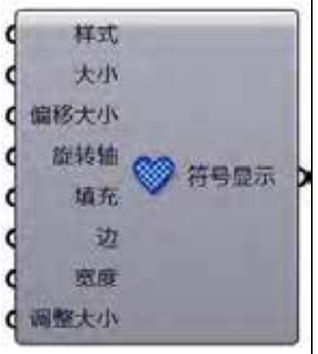
符号显示
Symbol Display

符号显示（简单）
Symbol (Simple)

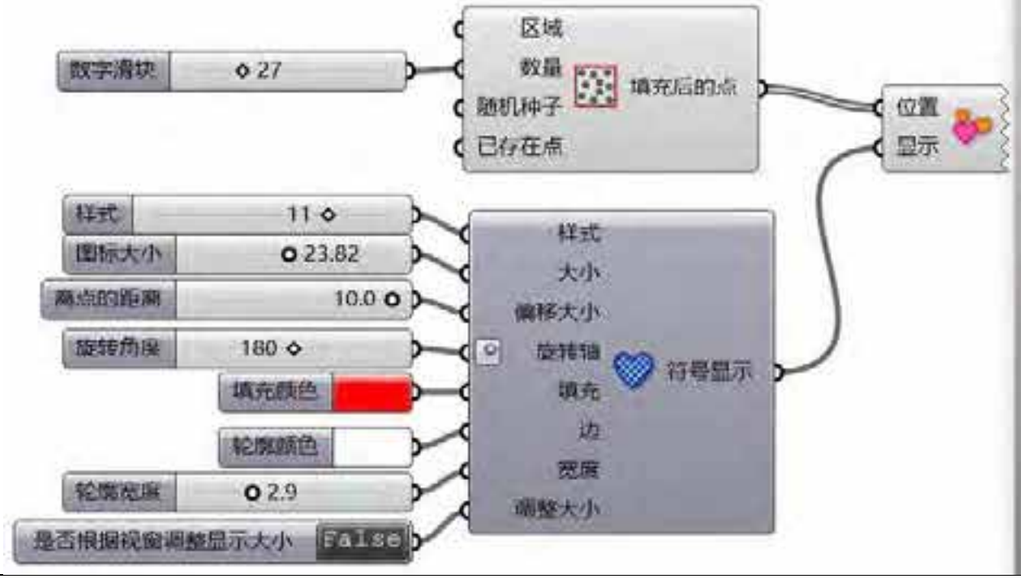
用图标符号标注点，共有 16 种预设样式，分别对应数字 0-15，可以调整符号的大小及旋转角度。

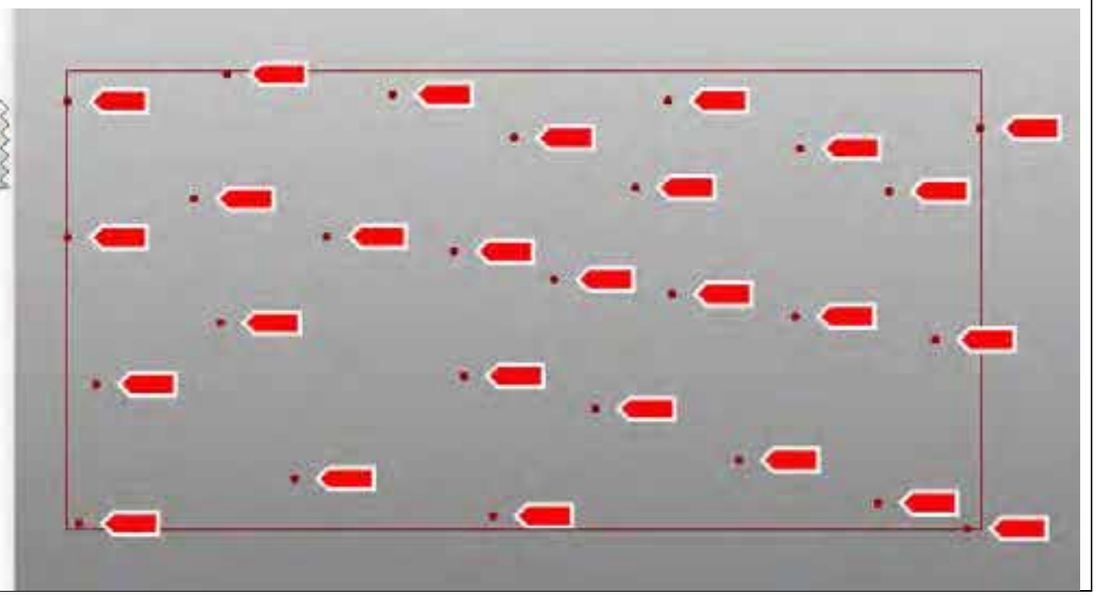
标注样式：

符号显示（高级）
Symbol (Advanced)



可调节图标符号的更多参数。图标离点的距离，轮廓颜色、大小等。

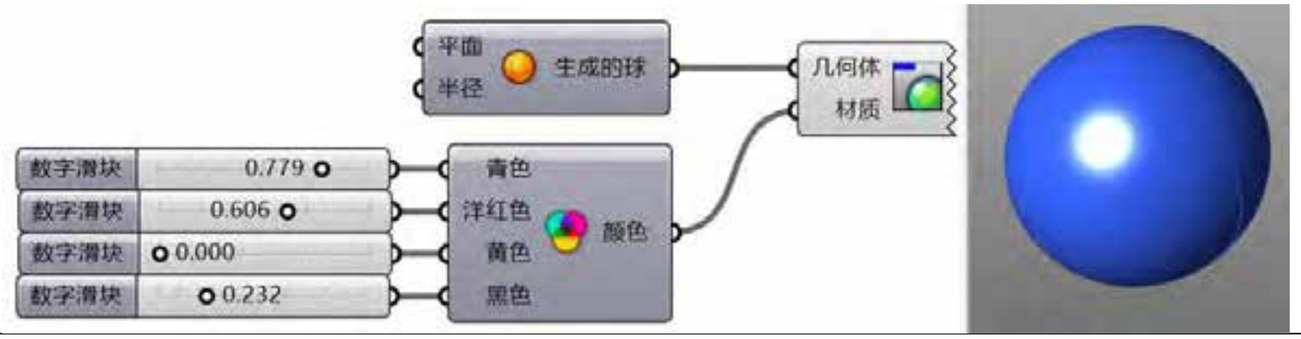




颜色 Colour – 显示

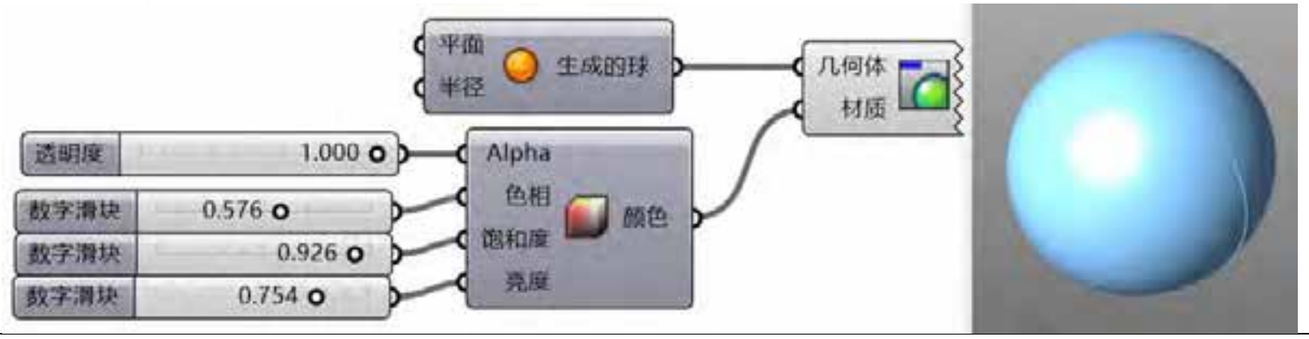
CMYK 颜色
Colour CMYK

CMYK 颜色，四个输入端都是在 0-1 之间取值。



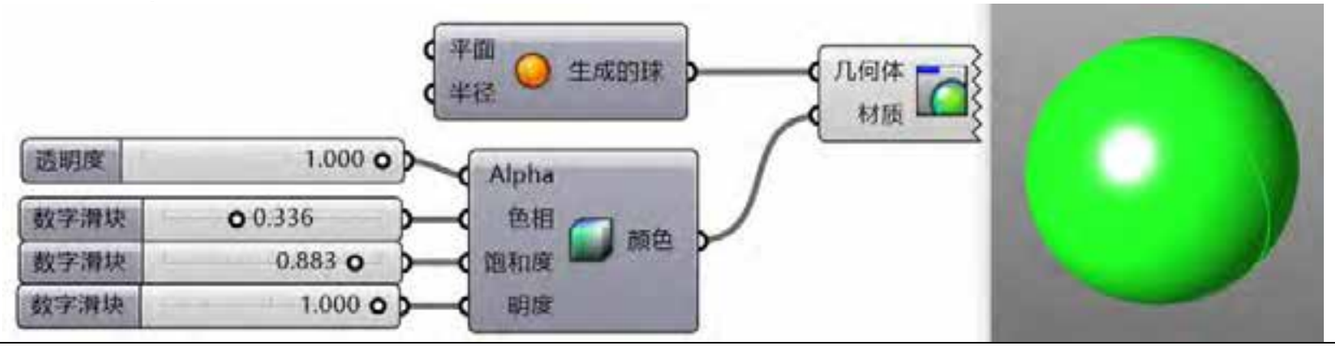
HSL 颜色
Colour HSL

HSL 颜色，Alpha 控制透明度。



HSV 颜色
Colour HSV

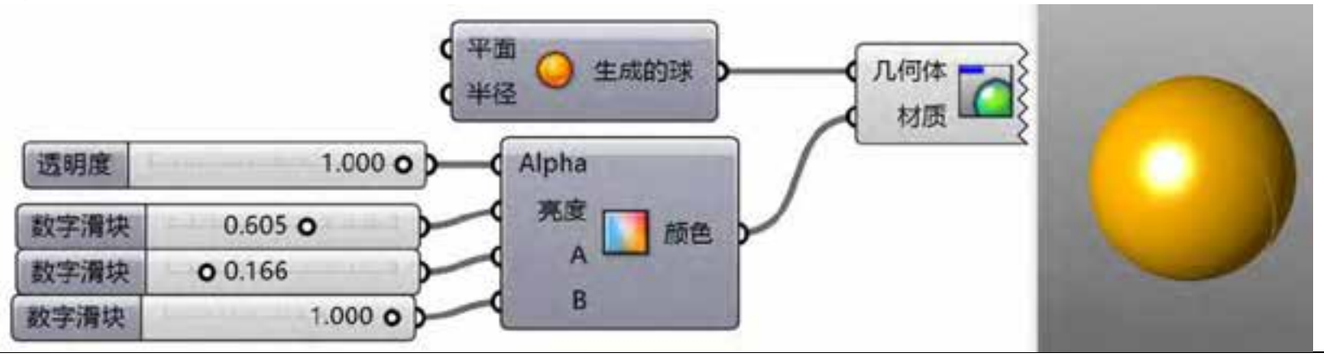
HSV 颜色，Alpha 控制透明度。

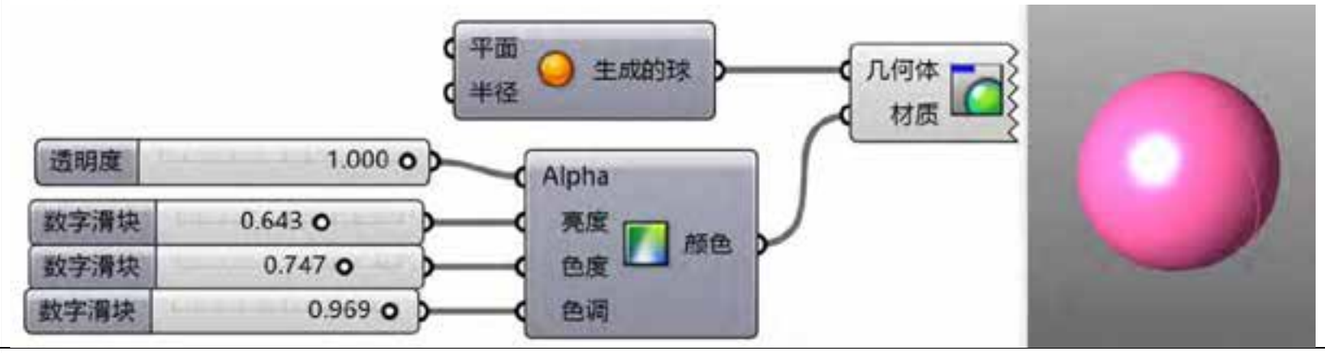
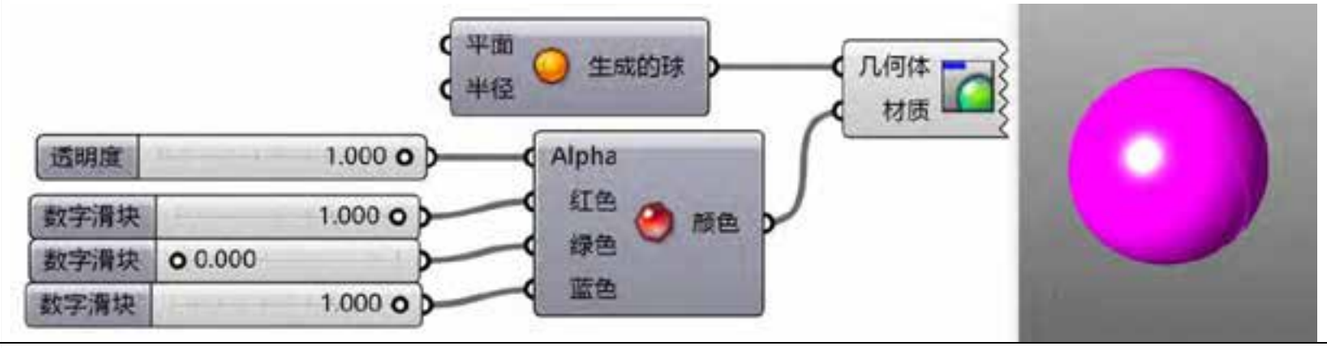
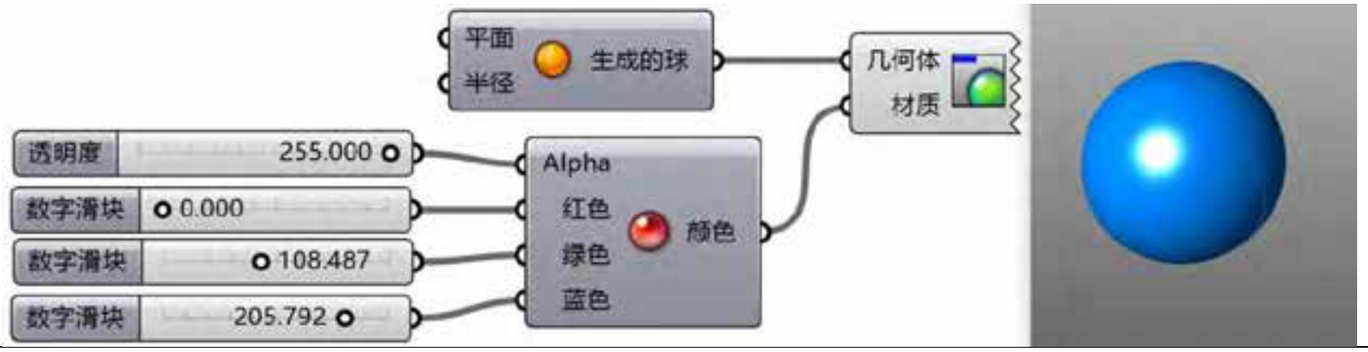
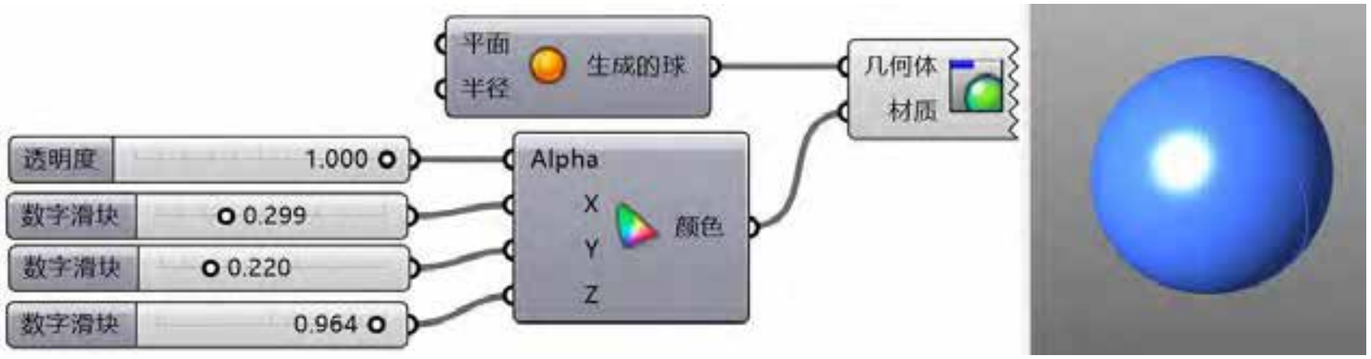


L*ab 颜色
Colour L*ab

不常用的话了解一下就好。

Lab 色彩空间（英语：Lab color space）是颜色-对立空间，带有维度 L 表示亮度，a 和 b 表示颜色对立维度，基于了非线性压缩的 CIE XYZ 色彩空间坐标。Hunter 1948 L, a, b 色彩空间的坐标是 L, a 和 b。[1][2]但是，Lab 经常用做 CIE 1976 (L*, a*, b*)色彩空间的非正式缩写（也叫做 CIELAB，它的坐标实际上是 L*, a*和 b*）。所以首字母 Lab 自身是有歧义的。这两个色彩空间在用途上有关联，但在实现上不同。



LCH 颜色 **Colour LCH** 	**LCH 颜色。** 
RGB(f)颜色 **Colour RGB (f)** 	**通过 0-1 直接的数值设置 RGB 颜色参数。** 
RGB 颜色 **Colour RGB** 	**通过 RGB 对应的颜色值（0-255）设置 RGB 颜色，透明度 alpha 也是 0-255 之间取值。** 
XYZ 颜色 **Colour XYZ** 	**用 CIE1931XYZ 色度系统来生成颜色。（下方详解）** 

分离 AHSL

Split AHSL

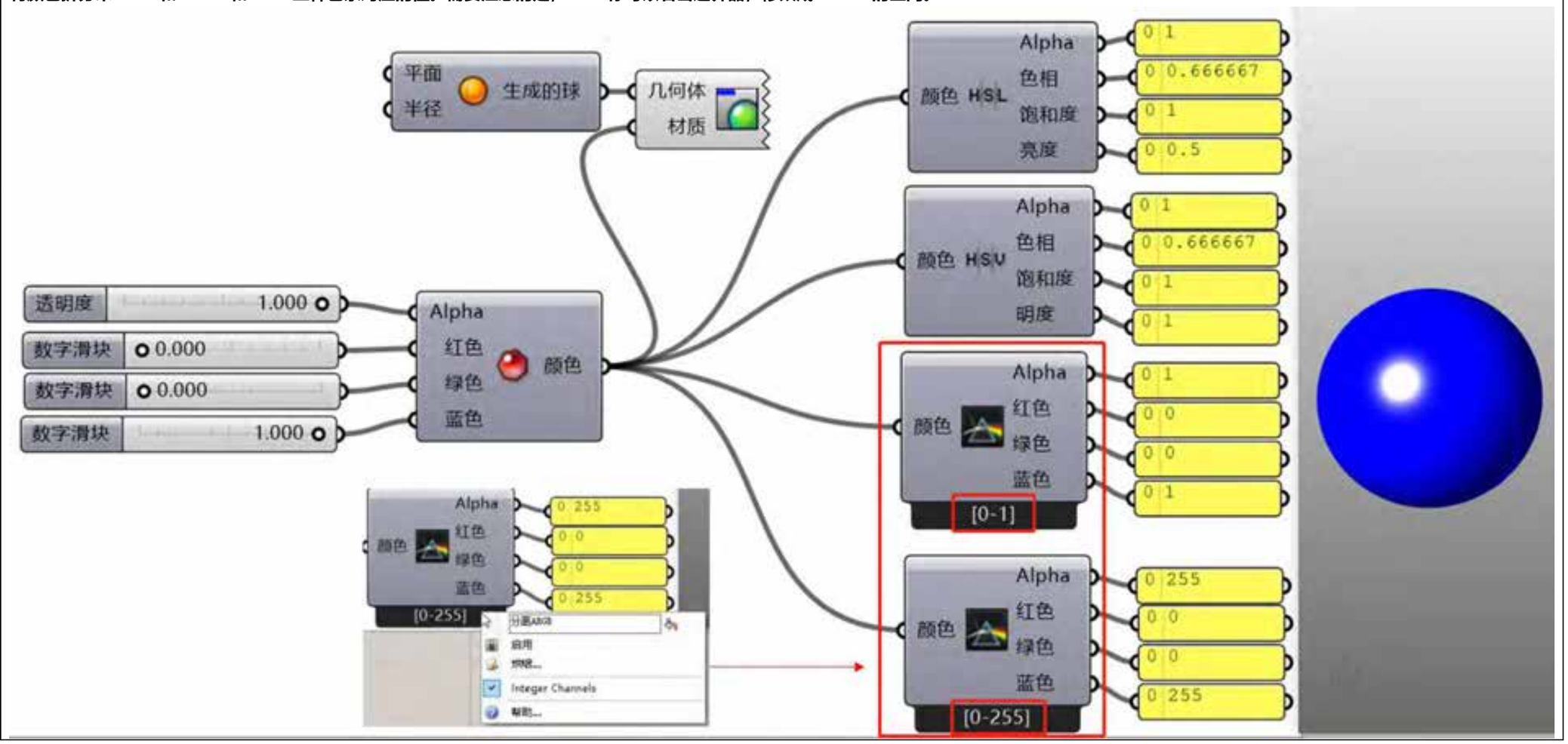
分离 AHSL

Split AHSV

分离 ARGB

Split ARGB

将颜色拆分未 HSL 和 HSV 和 RGB 三种色系对应的值。需要注意的是，RGB 你可以右击运算器，修改成 0-255 的区间。

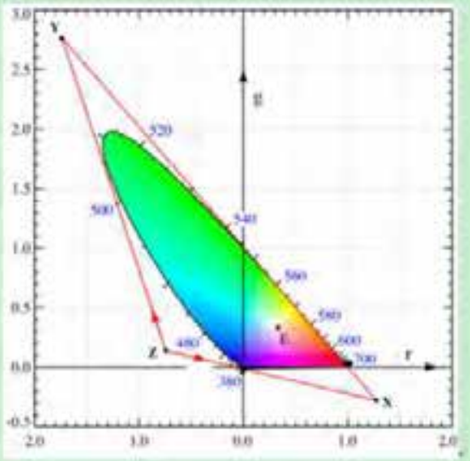


2.CIE1931XYZ色度系统

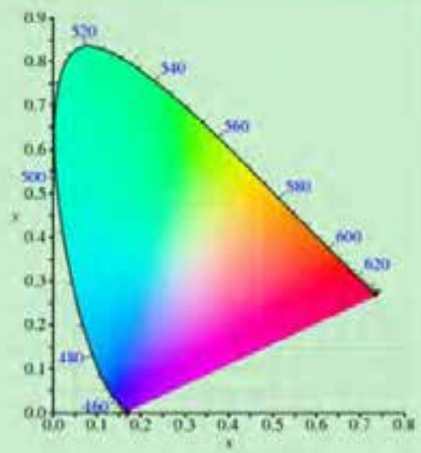
CIE1931RGB会存在负值，用起来不方便也不易于理解，因此进行了数学转换，用**假想的三个原色XYZ**代替了RGB系统的三原色并选择匹配等能白光确定三刺激值的单位。

如何确定假想三原色XYZ？需要考虑的因素如下：

1. 要想消除负坐标值，必须使XYZ三原色包围所有的颜色范围；
2. 规定XZ两个原色只代表色度，没有亮度，全部亮度都由Y承担；
3. 光谱轨迹曲线从**540nm附近到700nm**在r-g色坐标图上近似一条135度的直线，直线上所有颜色都可以有这两个单色光混合而成；
4. YZ边与光谱轨迹上**503nm的点的切线**重合；
5. 以相等数量的三刺激值匹配等能白光来确定三刺激值单位；



CIE1931 r-g 色度图

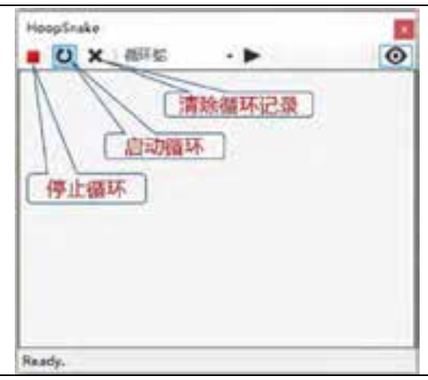


CIE1931 x-y 色度图

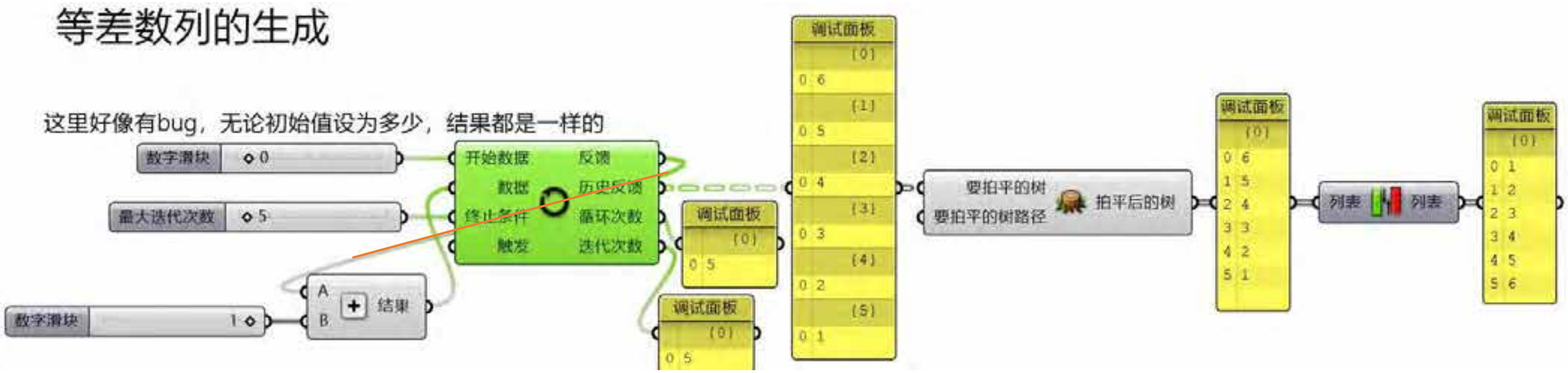
三条直线	
XZ：无亮度线	一个彩色的单位光亮度值 $L_{[C]} = rL_{[R]} + gL_{[G]} + bL_{[B]}$ $= r + 4.5907g + 0.0601b$ $L_{[C]} = 0$且$r + g + b = 1$ 得到$0.9399r + 4.5306g + 0.0601 = 0$
XY：穿过光谱曲线 540nm 和 700nm 两点的直线	$r + 0.99g - 1 = 0$
YZ:光谱曲线上 503nm 点的切线	$1.45r + 0.55g + 1 = 0$
三条直线的交点即为新的三原色 XYZ。	

循环 Hoopsnake

双击运算器进入编辑界面，控制循环的开启（右图），也可以在右键菜单控制循环。



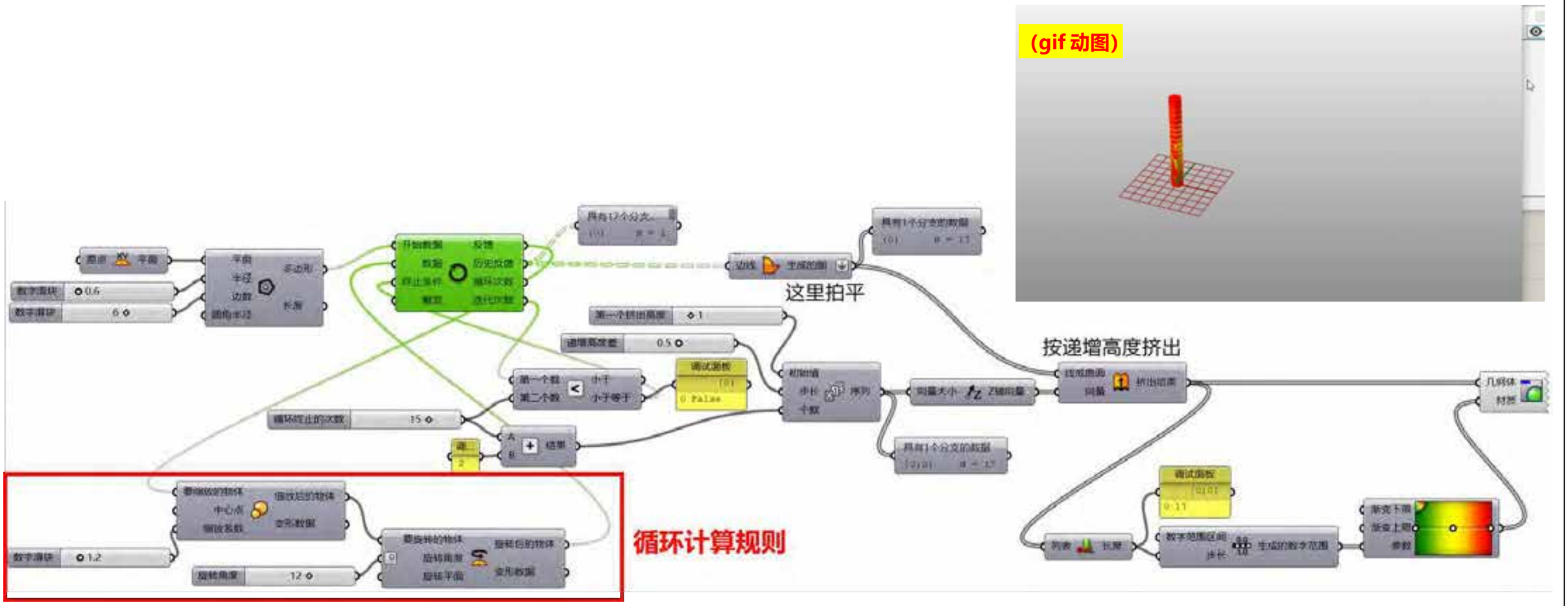
案例一：



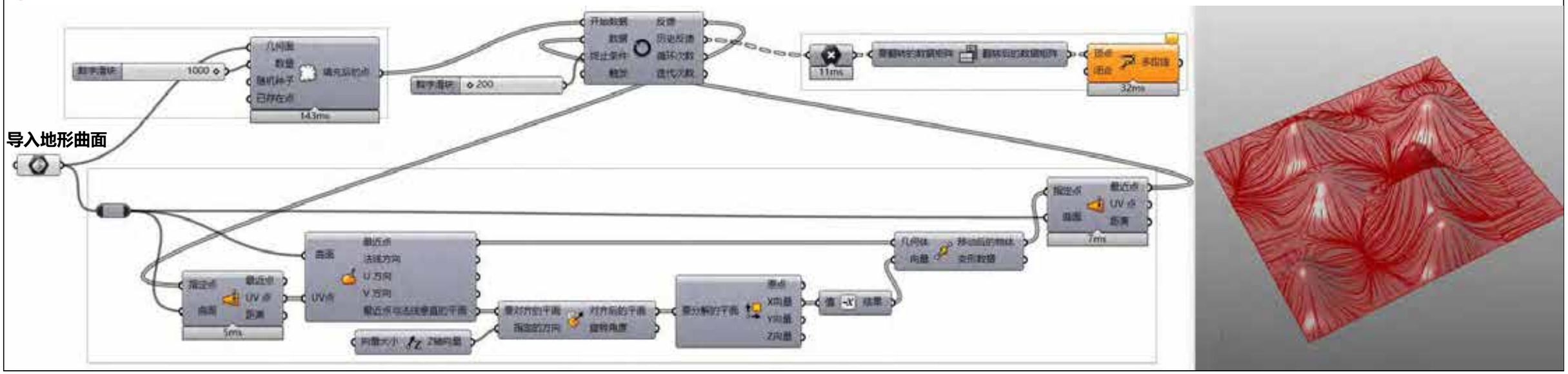
案例二：

1、下面案例中，最基础的运算过程是将一个六边形放大后旋转。

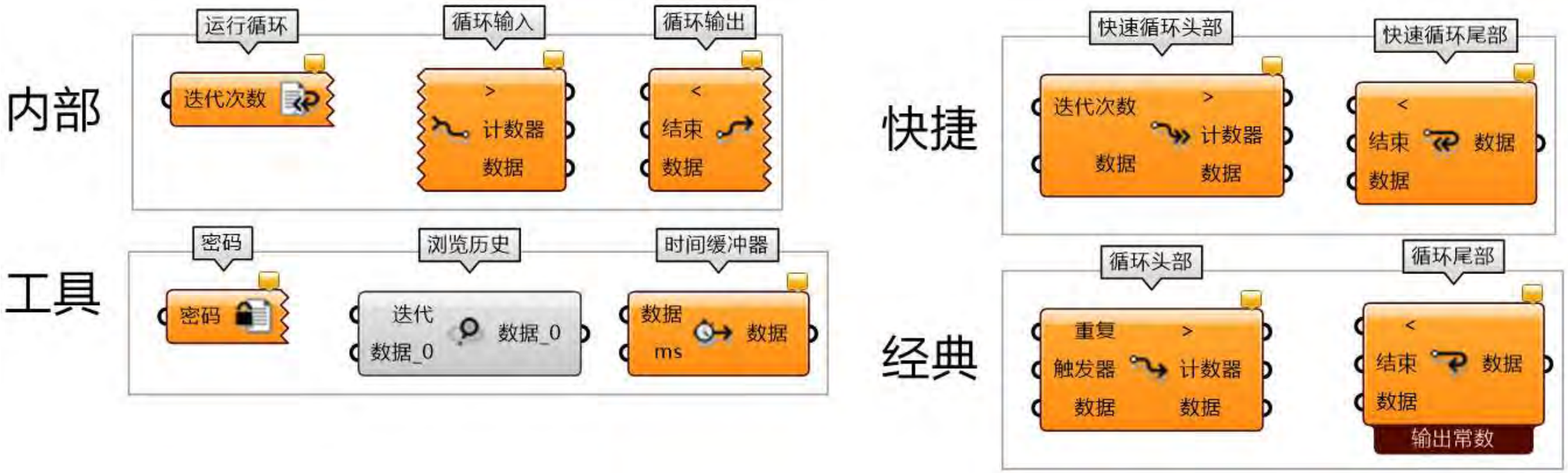
2、循环过程中，初始六边形接入开始数据，循环后的反馈即缩放旋转后的六边形，作为下次循环的初始数据接入循环计算过程的初始端（即要缩放的物体），循环计算后的结果接入循环运算器的数据端口，用小于比较运算器控制最大循环次数为设定值 15，超过最大次数时，小于比较运算器输出布尔值 False，结果输给终止条件控制循环结束。历史反馈包含了每次循环生成的六边形。然后按等差高度挤出，着色显示。



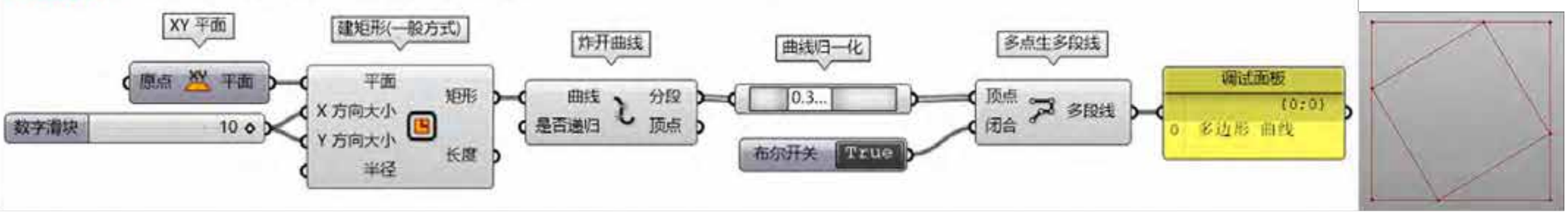
案例三：雨水路径模拟



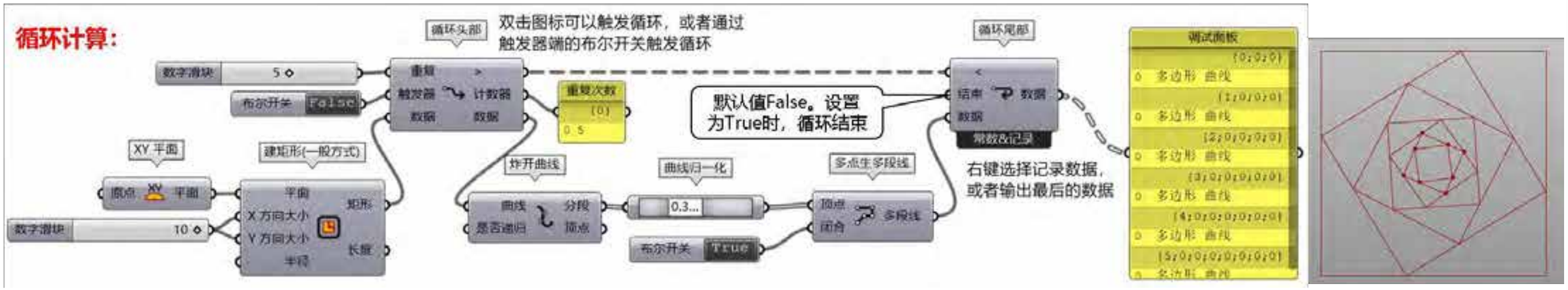
海葵 Anemone



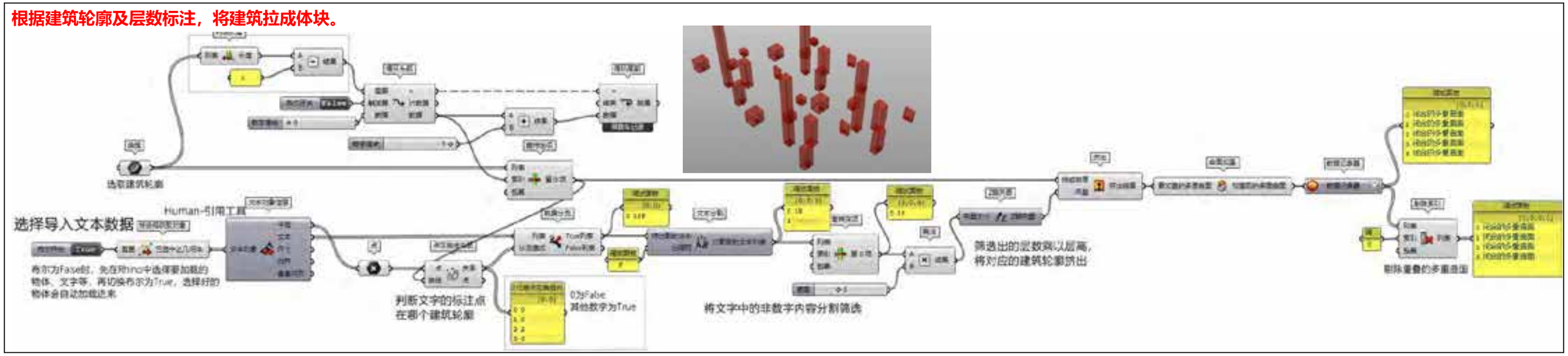
基础流程：正方形四边各取一点，连接各点生成正方形。



循环计算：



根据建筑轮廓及层数标注，将建筑拉成体块。

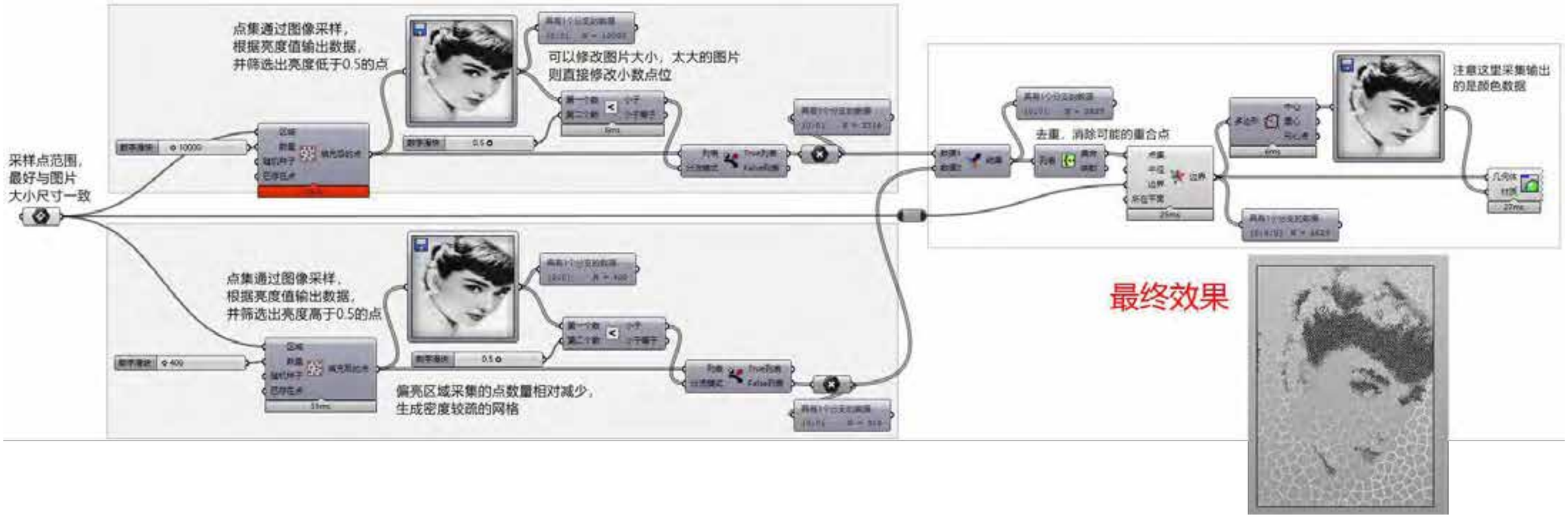
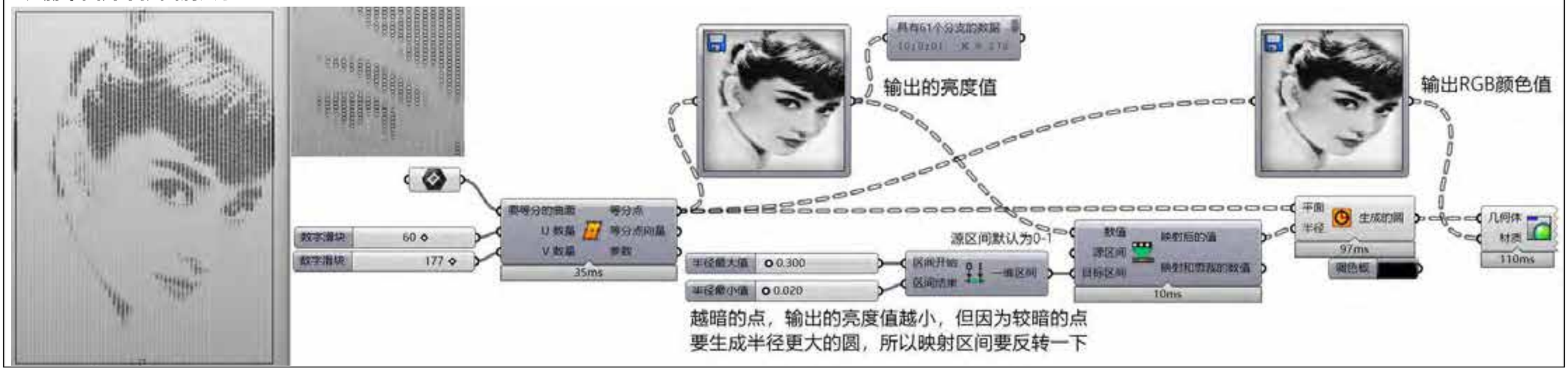


收集的一些案例

利用图像采集器 image samper 进行图片干扰

利用图像采集器 image samper 进行图片干扰

1、赫本图片干扰泰森多边形细分



2、赫本图片干扰圆洞大小