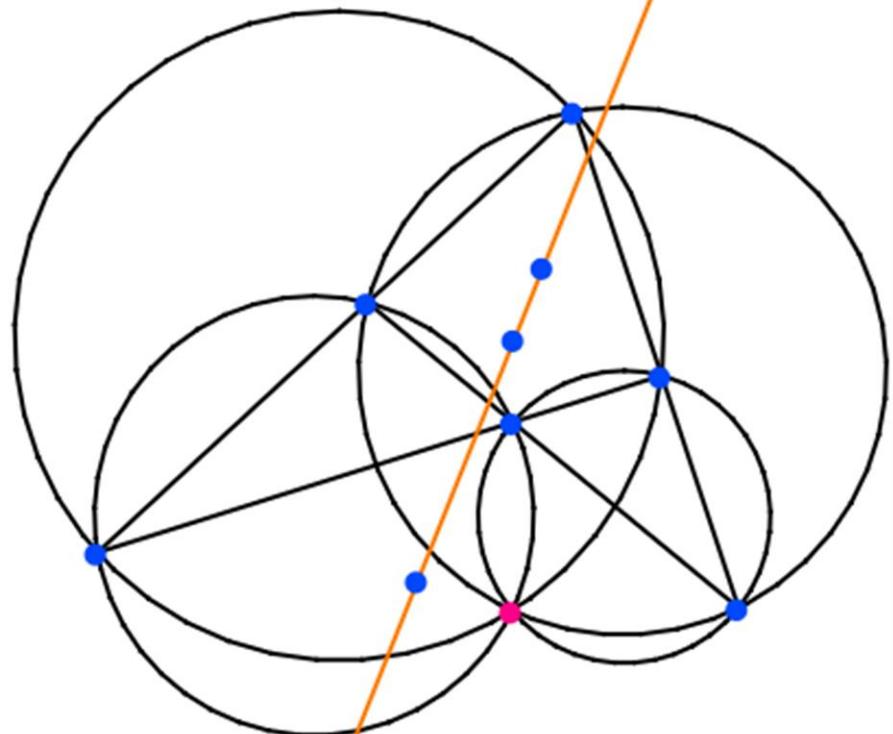


Version 4.8+



# 稽何计算器

用户说明

第 7 次修订

SQY

## 使用须知

感谢您对 SGC 的在线、离线软件感兴趣。本软件使您能够方便地探索数学。您可以使用 SGC 来绘制函数、创建动画、编辑程序、进行几何构造等。

**若要使用 SGC，您同意如下条款：**

- a. SGC 是开源软件。您不对 SGC 进行无意义的二次改编或者将 SGC 代码据为己有。
- b. 不对 SGC 官网发送、存储无意义的信息。
- c. 不得将 SGC 用于：
  - i. 违反中华人民共和国宪法、其他法律法规以及其他您所在地方的法律法规；
  - ii. 以计算器为媒介传播非法内容（包括但不限于暴力、色情、淫秽、辱骂、骚扰、侵权内容）；
  - iii. 使用此计算器进行考试作弊行为等(SQY 对因此产生的后果不承担责任)；
- d. 将 SGC 用于您个人的非商业用途，在为商业用途前请先申请。

这些条款可能会进行修订、增补。您在条款更新后继续使用 SGC，即表示您同意修订后的条款。如果不同意，您必须停止使用 SGC。SGC 保留在通知或不通知您的情况下修改或终止使用 SGC 的权利。

SQY 版权所有。本软件源代码开放，但复制、使用源代码需要标注原作者。

2025.7

# 简介

稽何计算器(SQY Graphic Calculator, SGC)是一个数学软件，始于 2022，主攻图形相关数学知识，在 scratch 以及能支持 scratch 3 的平台上运行。目前最新的版本是 4.8+.4.5 以上的版本能在 SGC 内获取更新。SGC 功能丰富，支持显函数、隐函数、参数方程、极坐标等多种类型。在 TurboWarp、html 环境下，运行速度与 desmos、GeoGebra 接近。您可以设置自己的参数，以获得比 desmos 更好的绘图效果。

SGC 兼容性高，手机、平板电脑、电脑上都可以运行。

敬请期待 SGC v5！

访问 <https://sqy419.axolotlpower.com/sgc> 获取更多信息。

## 致谢：

0. turbowarp.org; github pages; gtd232(美西螈力量)
1. Obdopqo-四万字超大画笔字库代码以及示例字库  
(使用教育部-通用规范汉字表,约减后收录 10000 余字符)
2. Alpha Math、Origin 图形计算器：同类优质作品，交流互鉴
3. -6-科学计算器中的公式渲染以及幂计算、傅里叶变换音乐生成器
4. 其他对 SGC 的开发做出贡献的开发者

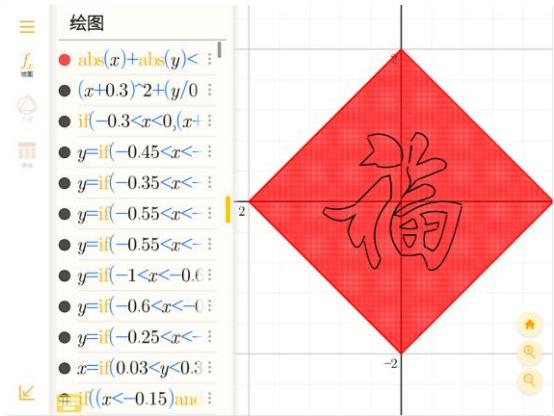
使用了如下非原创内容：

1. CMU Sans Serif、CMU Serif、Jetbrains Mono 等字体。
2. Obdopqo 多边形填充与点击判断

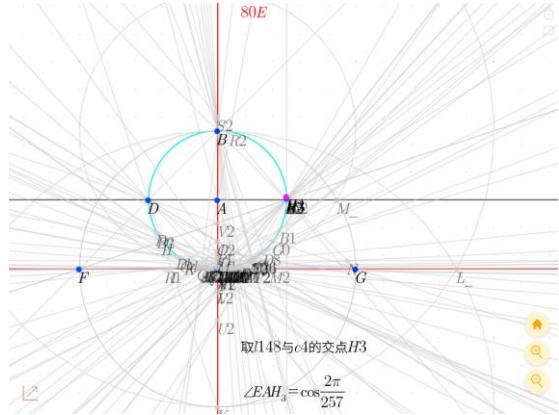
# 使用 SGC, 探究数学之美

Enjoy math with SGC

轻松创建创意图表，分享你的想法！

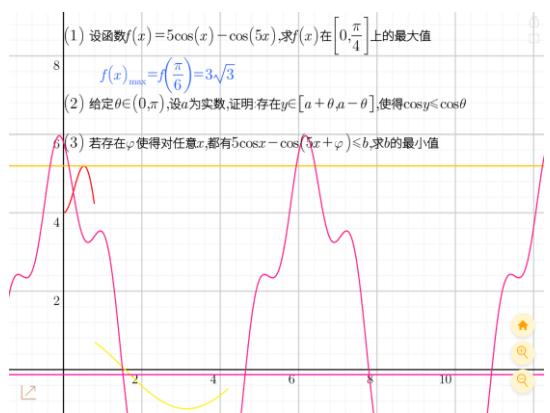


使用脚本，打破常规，创建高级演示动画，  
易如反掌。



不善言辞，如何表达情感？面对喜欢的人，  
总不能只用乏味的词语。SGC 帮助你使用数学之  
美，win his/her heart.

使用 SimpleSTeX，显示优雅公式与图像的结  
合。



## 常见问题

1. 为什么画不了图像?
  - a. 可能是你没有点 enter。只有将函数输入后,点击 enter 或者虚拟键盘上的箭头,才能解析输入的内容并绘制。目前 SGC 不支持预览功能.
  - b. 可能是你输入的函数计算比较复杂,需要解析较长时间,或者精度较高的隐函数.
  - c. 出现 bug.
2. 为什么我保存的东西没法导入 Arch,为什么 Arch 计算器里面的某些东西没法导入进来?

有些函数语法不一样。目前 SGC 只能导入自己的存档。
3. 为什么画出来的隐函数效果很差,丢失细节?

(参见章节 10.2)

这有可能是精度不足。在设置选择“绘图”,在当前模式中点击“隐式精度”, 输入合适的精度。精度过高可能会导致软件崩溃。精度很高的时候绘制中建议不要进行其他操作,否则可能会计算错误。拖动网格时可能会终止正在进行的计算。

# 目录

---

## 1 主界面

- 1.1 界面介绍
- 1.2 基本操作
- 1.3 键盘
  - 1.3.1 帮助
- 1.4 素材库

## 2 输入

- 2.1 输入方式
- 2.2 可以输入的内容
- 2.3 输入语法
- 2.4 常见函数构造指引

## 3 输入对象操作

- 3.1 主要界面操作
- 3.2 小窗口操作

## 4 绘制

- 4.1 平面绘制
- 4.2 3D 绘制

## 5 计算引擎

- 5.1 技术参数
- 5.2 计算
- 5.3 运算符与函数
- 5.4 高级运算符
  - 5.4.1-5.4.5 Sum, Prod, Int, Diff, fSolve

## 6 几何

- 6.1 基本操作
- 6.2 创建几何对象

## 7 文本

- 7.1 普通文本
- 7.2 TeX

## 8 脚本

- 8.1 脚本编辑器
- 8.2 语法
- 8.3 操作 SGC 底层
- 8.4 系统常数
- 8.5 图形库
- 8.6 动画播放

## 9 统计

- 9.1 统计表概述
- 9.2 数据的输入
- 9.3 数据的显示
- 9.4 列操作
- 9.5 列统计计算
- 9.6 列回归计算

## 10 设置

- 10.1 全局
- 10.2 坐标系与网格
- 10.3 绘图
- 10.4 性能
- 10.5 实验功能与其他
- 10.6 变量设置

## 11 多边形与逐帧动画

- 11.1 多边形
- 11.2 动画

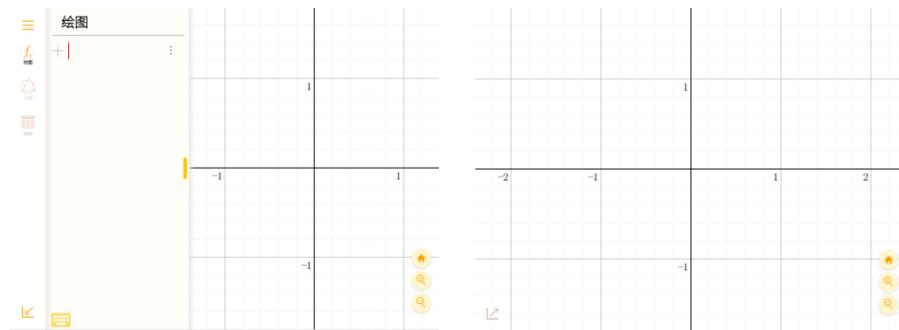
## 12 附录

## 约定

1. Infinity 必须大写, 小写的 infinity 不是 Infinity。
2. 在计算中, true=1, false=0。
3. 计算中, 常数 Pi 和 EulerGamma 也可以不大写。但是 Gamma 不指代 EulerGamma, 而是指代 Gamma 函数。
4. SGC 标准颜色代码: aabbccdd。aa 填入颜色, bb 填入饱和度, cc 填入亮度 dd 填入透明度。aa、bb、cc、dd 范围均为 0 至 99. 如 00999900 是相当于#ff0000 的红色。
5. SGC 的像素坐标系范围:  $x:[-240,240]$ ,  $y:[-180,180]$ 。
6. NaN 会显示为"?"。

# 1 主界面

## 1.1 界面介绍



点击小绿旗，启动计算器之后，会有如图的界面。这就是主界面。左侧浅黄色区域为输入区，右边有网格的区域是图形区。左边白色区域是模式区，点击汉堡按钮可以打开菜单。最下面的按钮可以隐藏输入区，全屏显示图形；再次按下此按钮可以显示输入区。按下小房子图标可以使坐标系回到初始位置，使用放大镜图标可以放大或者缩小坐标系。

## 1.2 基本操作



菜单栏打开后有几个选项。点击文件打开文件面板。

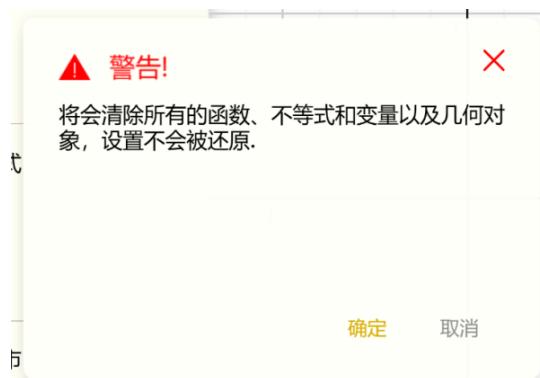


“文件”中可以保存或者打开 sgc 文件。“文件”中还可以保存设置 json。设置 json 可以保存当前 SGC 的全部设置，以及坐标系的位置和缩放因子。这样，打开文件时，就可以恢复保存时坐标系的位置。

SGC json 遵循标准 json 格式，规范如下：

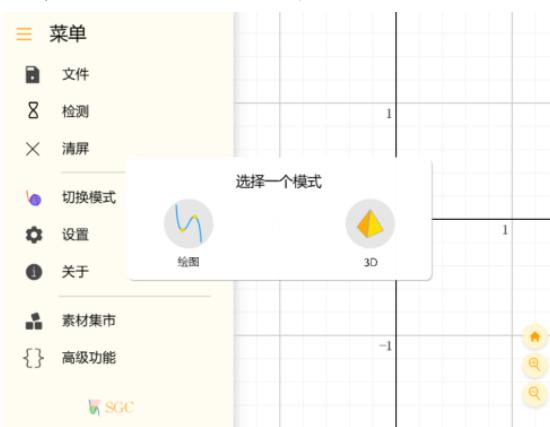
```
{  
  "SGCjson": "v0.2",  
  "fontsize" : 18, "highlight" : 1,  
  "sfsspeed" : 0.5,  
  "mode" : 1,  
  "axis" : 1, "NumOfAxis" : 1, "grid" : 1,  
  "FPS" : 0, "Temp" : 0, "MousePos" : 0, "loadIcon" : 1,  
  "SpecialPoints" : 0, "dtrecalc" : 0, "InputNum" : 0,  
  "pqmin" : 0, "pqmax" : 6.283185307179586,  
  "pqprec" : 400, "pqminexp" : 0, "pqmaxexp" : "2π",  
  "polmin" : 0, "polmax" : 6.283185307179586,  
  "polprec" : 400, "polminexp" : 0, "polmaxexp" : "2π",  
  "dx" : 0, "dy" : 0, "sf" : 0  
}
```

点击清屏将会弹出确认窗口,确认后将会清空屏幕,但是设置不会被还原。

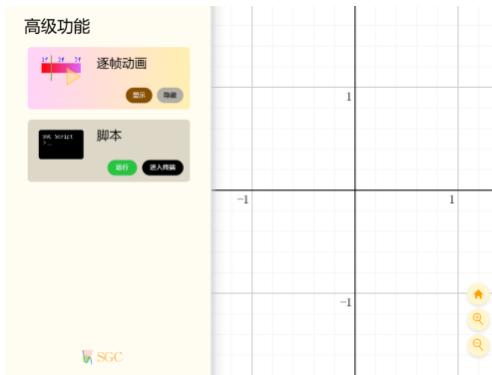


点击检测会运行检测模式,但这个功能并未完成。

点击切换模式可以切换到平面或者 3D 坐标系。

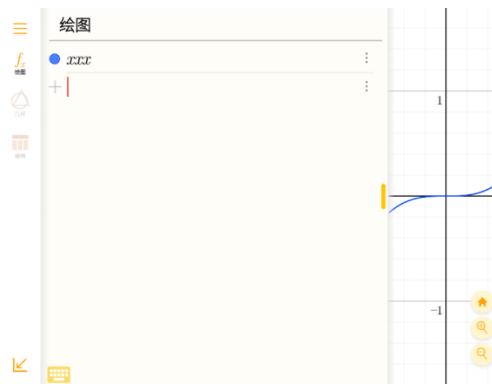


点击设置打开设置，点击关于打开关于页面。



(脚本的运行、更多信息参见章节 1.4)

点击高级功能可以选择打开动画模式以及脚本终端，点击相关的按钮即可。



在函数显示区,你可以拖动右侧黄色按钮来更改显示区的大小。下方有一个黄色的键盘图标，点击即可打开虚拟输入键盘。

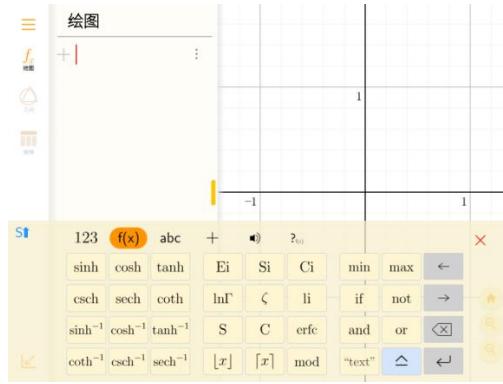
点击素材集市打开素材库。(参见章节 1.4)

### 1.3 键盘



打开键盘后,键盘上方会有一个橙色圆框,还有“123”(数字与符号)、“ $f(x)$ ”(函数)、“abc”(字母)、“+”(外部导入)、“喇叭”(函数音频)、“? $f(x)$ ”(帮助)几个按钮,点击前三个即可切换到对应的输入模式。

点击“+”从外部导入表达式,点击喇叭播放当前坐标系中函数的声音,点击问号打开帮助页面。在帮助页面可以搜索函数并查看说明。“数字与符号”模式可以输入0~9的数字、 $\pi$ 、 $e$ 、 $x$ 、 $y$ 、 $z$ 等常见量,也可以输入常用运算符,包括 $+$ ,  $-$ ,  $*$ ,  $/$ ,  $^$ 等。



函数模式可以输入函数，例如三角函数、特殊函数。另外，点击“text”按钮可以插入一对引号（参见章节 5.2）。字母模式可以输入字母和一些常数。蓝色向上箭头按钮是 shift，点击后键盘左上角会出现蓝色“ $S\uparrow$ ”字样，某些按钮会变化。

长按键盘上的键可以连续输入。键盘右上角有一个红色小叉，点击可以关闭键盘。输入区被隐藏后将无法打开键盘。

**Shift 键的高级功能：**按下 shift 再按下左右按钮，可以快速跳到当前表达式开头或者结尾。

点击“删除”可以删除光标前一个字符，长按可以快速删除。类似箭头的按键是 enter，可以将你输入的内容写入输入区并绘制。在实际的键盘下按 enter 作用等同。

注意：错误的表达式在按下 enter 键后也会被输入。空白的内容不会被输入。

### 1.3.1 帮助



点击“? $f(x)$ ”按钮打开帮助。上方搜索框操作与下面提到的函数输入方法一样。点击右上角红色叉关闭帮助。



例如搜索“integral”，出现了五个包含 integral 的函数。请注意，搜索也会显示相关结果，例如搜索“si”会出现 sin。

在帮助中点击某项，可以查看说明。例如点击“sinIntegral”，出现如图所示页面：

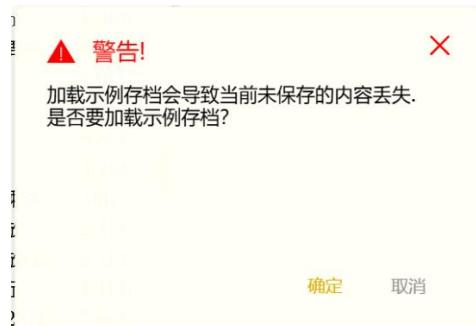


点击返回退回上一页。

## 1.4 素材库



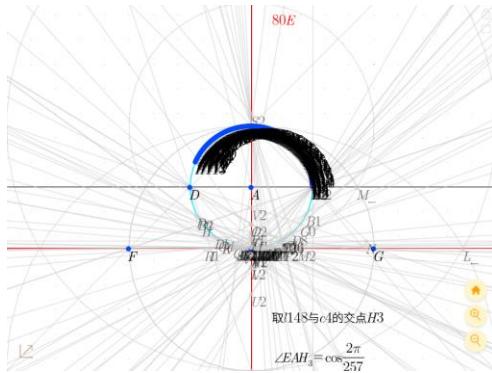
打开素材库后，出现如图所示界面。素材库分为几个部分：几何，函数，隐式，抽象，脚本等。其中，“脚本”中的素材，都需要运行脚本来实现效果。使用鼠标点击示例存档名称，即可打开这个存档。例如打开“80E 正 257 边形（脚本演示作图步骤）”后，出现弹窗：



点击“确定”即可打开这存档。



打开菜单栏，选择“高级功能”中“脚本”，点击“运行”，得到：



(脚本没有运行完。省略脚本运行过程)

(脚本的运行、更多信息参见章节 8)

## 2 输入

### 2.1 输入方式

SGC 提供了多种输入方式,你可以在虚拟键盘从外部导入表达式,也可以打开虚拟键盘进行输入,还可以直接按真正的键盘输入。

如果您使用原版 scratch 运行本软件,那么注意：由于 scratch 3 不支持 backspace,故删除键用“＼”代替。长按可以快速删除。



点击确认即可在光标前插入你输入的内容。

### 2.2 输入类型

在输入时,表达式左侧会有一个加号。点击加号能打开输入类型小窗口,可以选

择输入为表达式或者纯文本。纯文本可以作为注释，不会被计算。纯文本的语法错误会被忽略。目前 SGC 支持多种输入类型，包括但不限于显函数、参数方程、隐函数、常量（直接输出结果）。中文符号无法解析，但是中文可以作为变量。



## 2.3 输入语法

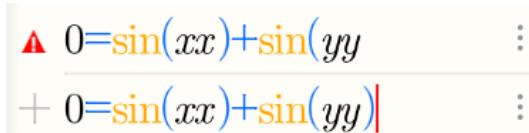
SGC 的输入默认使用 CMU Serif 系字体,带有高亮显示,这样可以清晰的显示你输入的内容。函数是黄色, 常量是绿色, 运算符是蓝色, 文本是灰色, 其余是黑色。其中, 自定义函数以斜体显示, 系统函数以正体显示。

$$\sin(ax+b)$$

这高亮可以在设置中关闭, 关闭后将显示

$$\sin(ax+b)$$

SGC 将会自动检测输入内容,这称作“错误检查”, 包括括号是否匹配、二元运算符是否合理、是否缺少输入对象、输入是否有效等。若你输入的内容有问题, 将会在输入区左侧显示一个红色“警告”图标, 否则是一个灰色的“加号”。比如输入“ $0=\sin(xx)+\sin(yy)$ ”,将会显示警告图标,将光标移动到最后并输入“)”补全即可。输入完成后,按下 enter 即可绘制或者输入。



### 2.3.1 自动补全与错误检查

你可以输入：

输入	输入解释	转义
$2\sin(2x)$	常量*函数,常量*变量	$2*\sin(2*x)$
$xx$	变量*变量	$x*x$
$ax\sin(x)$	变量*函数	$a*x*\sin(x)$
$x\sin x$	省略一个括号的函数	$x*\sin(x)$
$(x+1)(x-5)$	省略乘号的两个式子	
$2 < x < 5$	中间部分只有一个内容的不等式	$2 < x \text{ and } x < 5$

函数已经支持类似  $\sin x$  的简写。但是必要的圆括号不可以省略，例如  $\sin(x/y)$  不可以简写为  $\sin x/y$ 。

注意：遇到了可以解析成多种函数的内容，例如  $\operatorname{asinh}$ ，会按照最长的函数进行解析。例如：想要输入  $a*\sin(x)$ ,不可以输入  $\operatorname{asin}(x)$ 。否则会解析成  $\operatorname{arcsin}(x)$ 。

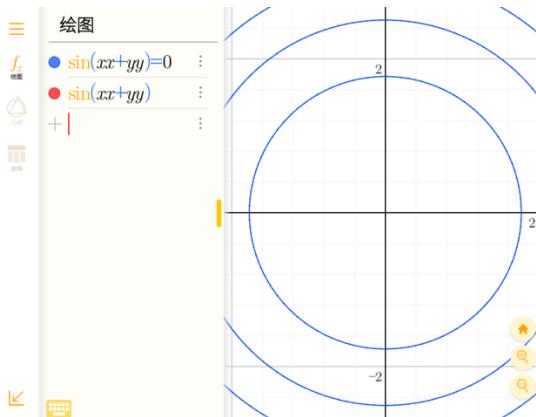
不可以输入：

1.  $2 < (x+1) < 5$ ,中间部分包含两个及以上内容的不等式
2.  $\sin\sin x$ ,两个不带括号的函数
3. 将  $\sin(x)^2$  输入为  $\sin^2(x)$

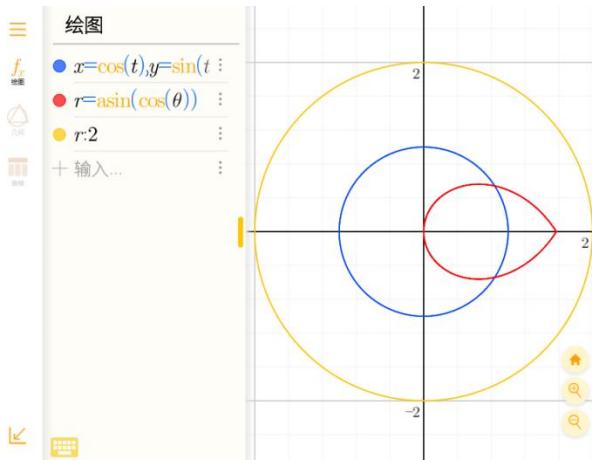
### 2.3.2 输入规范

若要输入隐函数,需要有隐式符号才能判定为隐函数。如“ $\sin(xx+yy)$ ”就会被识别为二元显函数,“ $\sin(xx+yy)=0$ ”才能被识别为隐函数。隐函数不支持形如  $1 < x < 2$  这样的输入。

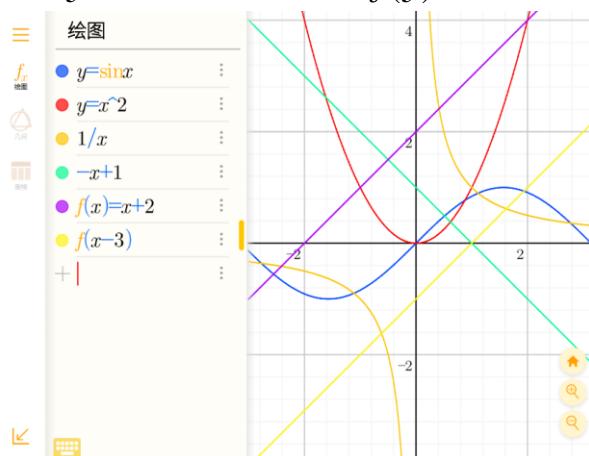
如果要绘制如上所述的图形,你需要输入：if( $1 < x, x < 2$ ).请注意：隐函数/不等式必须在最外层拥有一个判断符号。 $(xy=1)$ 这种输入不是隐函数，而是判断表达式。



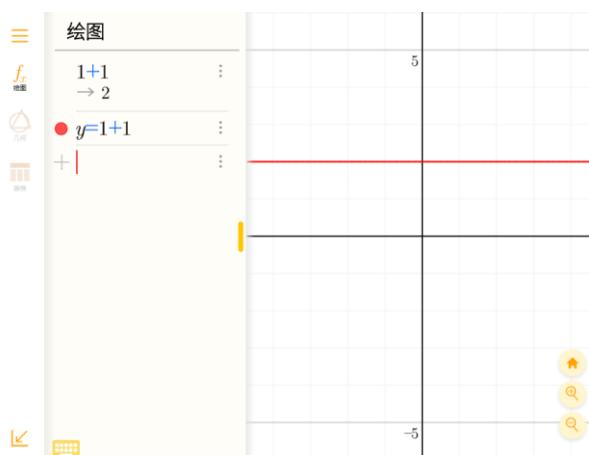
若要输入二元参数方程, 格式为： $x=f(t),y=g(t)$ 。变量均为  $t$ ，范围可以在设置中更改。如绘制半径为 1,圆心为(0,0)的圆,可以输入  $x=\cos(t),y=\sin(t)$ 。若要输入极坐标函数, 需要输入: $r=f(\theta)$ , 或者  $r: f(\theta)$ 强行定义极坐标函数, 避免和变量  $r$  冲突, 比如输入  $r=\operatorname{asin}(\cos(\theta))$ , 或者  $r:2$ , 强行绘制半径为 2 的圆。



若要输入一元显函数,可以直接输入函数的内容。比如函数  $y=1+x$ ,可以直接键入  $1+x$ ,也可以输入  $y=1+x$ ,或者  $f(x)=1+x$ 。以最后一种方式输入的函数,将可以在后面进行调用。SGC 同样也支持  $x$  关于  $y$  的函数。输入  $x=f(y)$ 或者仅包含  $y$  的表达式即可。



若要进行常量计算,系统将会自动识别。比如  $1+1$ ,会直接输出=2,而不是画一条  $y=2$  的直线。若想画出这种直线,你可以输入  $y=1+1$ 。这样就会把这个式子识别为显函数而不是常量。



变量分可播放变量和常量。创建变量可以输入  $a=0$ ,也可以输入  $a$  直接按下 enter。SGC 会自动创建未定义变量并显示滑动条,但是定义为可播放变量。常量不带有滑动条。记变量内容为  $u$ ,如果  $u$  仅为一个实数,那么是可播放变量,否则是常量。比如  $a=\sqrt{2}$  是常量.若要创建  $a=1$  这一常量,可以输入  $a=1+0$ .



SGC 支持定义你自己的函数。比如定义函数，名为  $f1$ ，参数为  $u, v$ 。

那么，输入如下内容：

$$f1(u,v)=u-v\sin u$$

未定义的函数以黑色展示，已定义的函数以黄色斜体展示。

请注意：SGC 支持定义拥有多个参数的函数。如果需要创建包含系统函数名的自定义函数，如 `mysin()`，需要在其中包含至少一个下划线。如 `my_sin(x)` 为合法命名。

例如定义  $\text{myfunc}(x,y)=x\sin y$ 。

并且输入  $1+\text{myfunc}(x,x-1)$ ，显示如图。



## 2.4 常见函数构造指引

注意：SGC 的函数还能以文本作为参数。

### 2.4.1 分段函数

在 SGC 中，有多种方式构造一个分段函数。

例如，要构造  $y=\begin{cases} -x, & x<0 \\ x^2, & x\geq 0 \end{cases}$ ，可以输入：`ifelse(x<0,-x,x^2)`，这是用 `ifelse` 判断。同样，也可以利用 SGC 的逻辑计算符： $(x<0)(-x)+(x\geq 0)x^2$ 。当  $x<0$ ，此表达式相当于  $(1)^*(-x)+(0)^*x^2$ ，反之同理。

### 2.4.2 距离

若要知道一个点  $(x,y)$  到原点的距离，可以定义函数： $\text{dis}(x,y)=\sqrt{xx+yy}$ 。

### 2.4.3 定义域的限制

若要限制函数定义域，可以使用 if 函数。例如，想要绘制 $-1 \leq x \leq 2$ 的  $y=x^2$ ，可以输入：  $y=\text{if}(-1 \leq x \leq 2, x^2)$ 。

这里如果使用  $y=(-1 \leq x \leq 2)x^2$ ，效果会有不同。因为在不满足条件时， $(-1 \leq x \leq 2)$  相当于 0，于是  $y=0$ ；而使用 if，不满足条件会返回 NaN，不会画出函数。

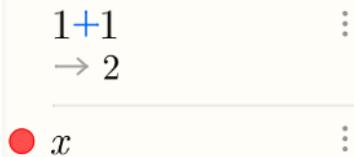
限制定义域也可以使用定义域有限的函数来实现。由于 SGC 没有 CAS 功能，表达式不会被简化，类似  $x+0$  的式子在内部依然使用  $x+0$  来计算。因此，上述式子可以用  $y=x^2+0/\sqrt{1+x}+0/\sqrt{2-x}$  来代替。

## 3 输入对象操作

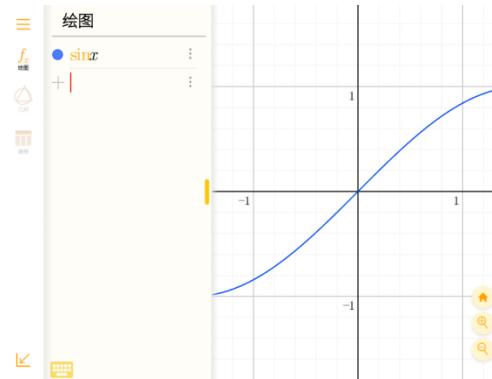
### 3.1 主要界面操作

输入一个非常量、变量对象后，左侧会出现一个彩色圆形。常量、变量对象将不会有此圆形，而是空的。

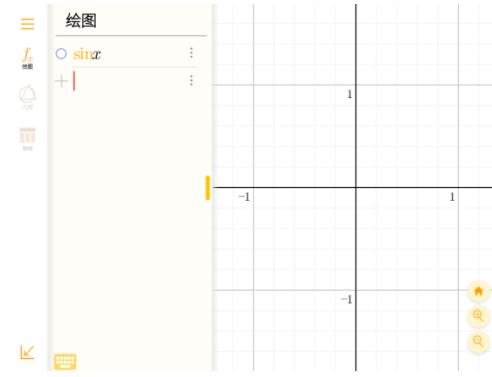
点击此圆形即可使当前对象显示或者隐藏。



1+1 的左侧没有圆形，但是  $x$  的左侧有。



输入了  $\sin x$ 。现在显示的是  $\sin x$  的图像。



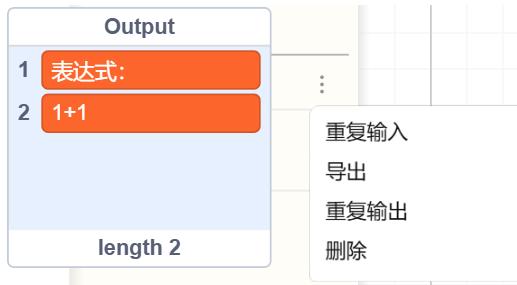
点击小圆形后， $\sin x$  这一对象被隐藏。

### 3.2 小窗口操作

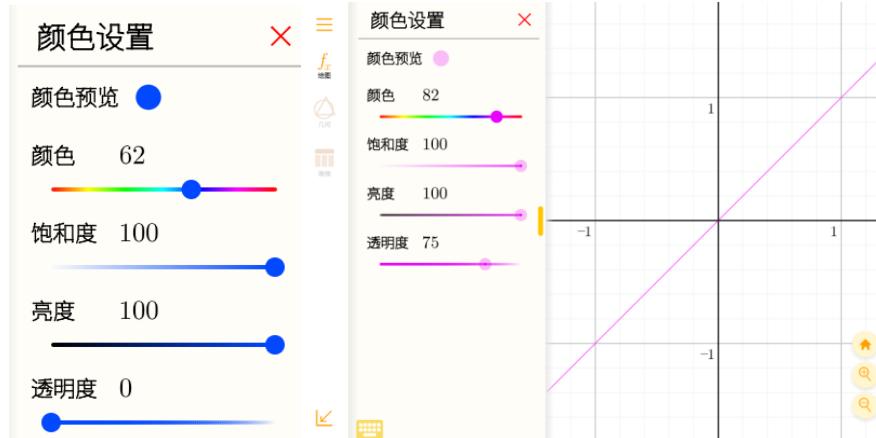
一个对象的最右侧有三个小点。点击后将会打开小窗口,有若干选项。鼠标所指示的选项会被高亮。



点击“重复输入”将会把选择的对象复制到当前光标的位置之前。点击“重复输出”将会把选择的对象所得到的结果复制到当前光标的位置之前。例如重复“1+1”的输出得到 2。点击“导出”将会导出选择的对象至一个列表,三击列表显示的那一项进行复制。



点击“颜色”将打开样式设置面板,你可以更改颜色、饱和度、亮度和透明度,下方可以预览修改后的颜色。点击“删除”将会删除选择的对象。



拖动滑动条即可修改参数. 四个参数最大值均为 100, 最小值均为 0。

## 4 绘制函数

### 4.1 平面绘制

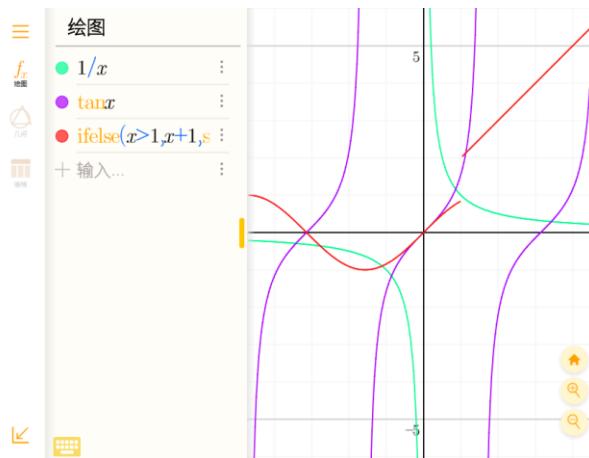
按住坐标系并拖动,可以改变坐标轴的位置。滚动鼠标滚轮,可以放大或缩小坐标系。滚动鼠标滚轮将会以鼠标为中心进行缩放,点击右侧小按钮将会以软件像素坐标系原点为中心缩放。

SGC 的显函数采用区块加载, 未加载到的地方在拖动时不会被绘制, 停止拖动后将会计算未加载部分并重新绘制。隐函数也加入了缓存功能, 这样可以节省计算时间。

缩放函数后会延时计算，避免卡顿。

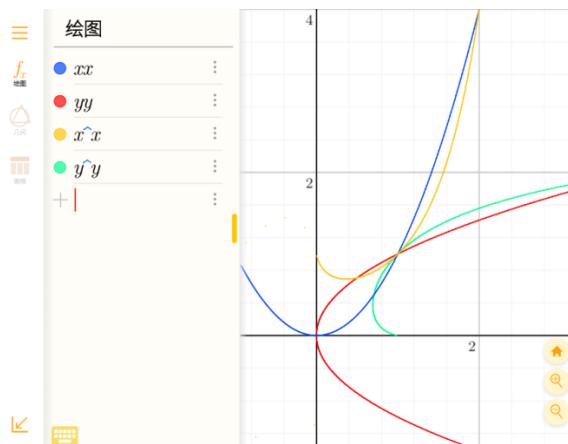
( 测试内容参见附录 11.1 )

隐函数推荐绘图精度 360，并推荐使用 TurboWarp 或者 html 端。



输入的函数会自动判断断点。( 参见设置 10.2 )

参数方程、极坐标若出现杂乱线条，将精度调到更高即可。

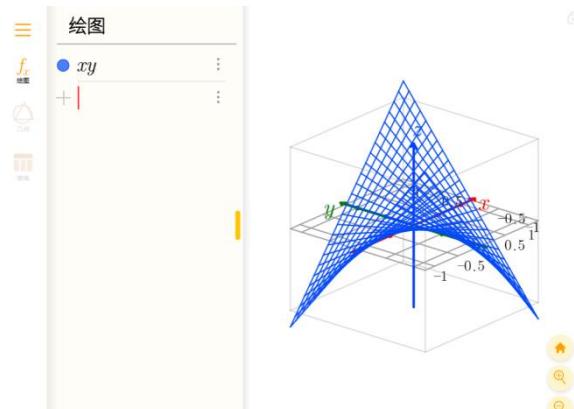


可见函数  $y=x^x$  和  $x=y^y$ , 函数  $y=x^2$  和  $x=y^2$  关于  $y=x$  对称。

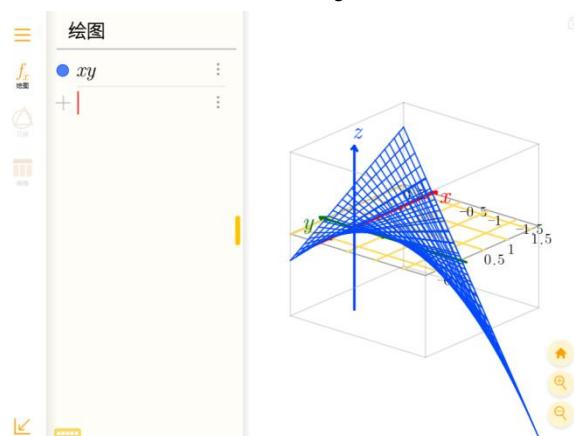
请注意：SGC 使用 marching squares 算法，因此如  $(x-y)(x+y)^2=0$  这样包含切面为  $z=0$  的隐函数可能无法绘制。

## 4.2 3D 绘制

3D 绘制支持绘制二元显函数和 3D 方程、不等式以及 3D 参数方程。3D 坐标系  $z$  轴是固定的,  $x$  轴和  $y$  轴可以转动。拖动 3D 网格时，网格会高亮。这代表正在平移  $x$ - $y$  平面。使用鼠标滚轮进行缩放时，缩放中心是鼠标所指代的  $x$ - $y$  平面上的一个点。



如图，这是默认  $x-y$  平面位置。



正在拖动  $x-y$  平面，网格被高亮。

( 测试内容参见附录 11.1 )

## 5 计算引擎

SGC 采用最新计算引擎 v4，支持下列几种对象：

数(number)	1
不区分 float 与 int	-2.5 0.333333333333333
字符串(String)	“hello SGC!” “0.3333333333333” “SQY!+/*/”
布尔值(Boolean)	true false

### 5.1 技术参数

对于三角函数，参数范围:  $-10^{10} < x < 10^{10}$ . 对于其他函数，保留 15 位有效数字。

实数最大为  $1.79769313486231e+308$  即  $2^{1024}$ ，最小为  $1e-323$ . 系统常数采用 15 位小数。

幂的计算：对于  $a^b$

若  $a>0$ ，返回  $e^{(\ln(a)*b)}$

若  $a < 0$ , 试着将  $b$  化为分数: 如果  $b$  的小数位数不足 7 位, 认为  $b$  是一个有理数。如果  $b$  化为分数  $x/y$  后,  $x$  或者  $y$  一者大于  $10^8$ , 认为  $b$  不是一个有理数。如果  $b$  是有理数, 那么进行幂的运算。

这样, SGC 能判断一些幂是否有意义:

$(-1)^{0.333}$	:
$\rightarrow ?$	
$(-1)^{0.3333333333333333}$	:
$\rightarrow -1$	
$(-1)^{(1/3)}$	:
$\rightarrow -1$	
$(-1)^{(-0.1145)}$	:
$\rightarrow ?$	
$(-1)^{(-0.114514)}$	:
$\rightarrow ?$	
$(-1)^{(-0.1145141)}$	:
$\rightarrow ?$	
$(-1)^{(-0.1145141919810)}$	:
$\rightarrow ?$	
$(-1)^{(\sin 1)}$	:
$\rightarrow ?$	
$(-2^{13})^{0.076923076923076923}$	:
$\rightarrow -2$	

另外, 认为  $0^0=1$ :

$0^0$	:
$\rightarrow 1$	

## 5.2 计算

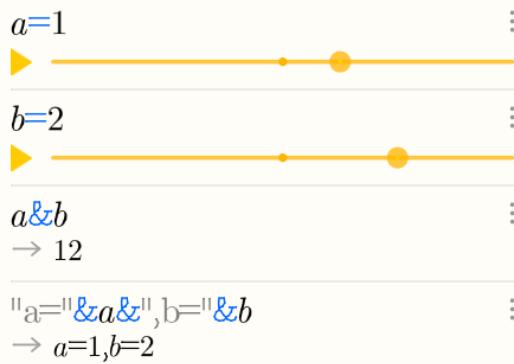
在计算中, 任何 NaN 去运算都得到 NaN。任何数除以 0 都得 NaN。

使用`&`来连接文本。如果 `a&b` 中, 有任何一者不是数, 那么会强行将这一者转换为一个字符串。

例如: 有一个变量 `a=1`, 另一个变量 `b=2`.

输入表达式 `a&b` 得到结果"12".

输入表达式"`a=&a&,b=&b,` 得到结果"`a=1,b=2`".



### 5.3 运算符与函数

SGC 支持的运算符如下：

数学运算: +,-,\*,/,^,! (阶乘)

字符串: &(字符串拼接. 如 "aa"&"bb"="aabb", "a"&l="a1")

逻辑运算: <,>,=,<=,>=,==

下表列举了 SGC 目前支持的函数。函数的输入不一定需要打括号,但是使用虚拟键盘输入的函数,都会带上括号。

函数表

(由于 scratch 不分大小写,这些函数也不需要区分大小写)

函数名	解释	注释
sin,cos,tan,csc,sec,cot	三角函数	弧度
asin,acos,atan acsc,asec,acot	反三角函数	弧度, arcsin 之类的也可以识别
sinh,cosh,tanh csch,sech,coth	双曲函数	
asinh,acosh,atanh acsch,asech,acoth	反双曲函数	
ln,log	对数函数	$\log(x)=\log_{10}(x)$
alog	反常用对数	$alog(x)=10^x$
abs	绝对值	
floor,ceil round	向下/上取整 四舍五入	
sqrt,cbrt	平方根,立方根	
min,max	最小值,最大值	$\min(a,b)$ $\max(a,b)$

if	如果	$\text{if}(x>0,x+1)$
ifelse	如果否则	不满足条件则返回 NaN $\text{ifelse}(x>0,x,1-x)$
case	选择	$\text{case}(u,x,v,y)$ 如果 $u$ 为真, 返回 $x$ 并跳过 $v$ 的计算, 如果 $v$ 为真, 返回 $y$ 。 但会先计算 $u$
gamma	伽马函数	
zeta	泽塔函数	
erf,erfc	高斯误差函数	
sgn	符号函数	参数 $>0$ 则返回 1, $<0$ 返回 -1, 否则返回 0
lambertW	朗博 W 函数	$f(x)=xe^x$ 的反函数, 返回主分支
expIntegral	指数积分, 正弦积分, 余弦积分, 对数积分	
sinIntegral		
cosIntegral		
Li		
beta	beta 函数	$\text{beta}(x,y)$
psi	digamma 函数	
FresnelS	菲涅尔正弦、余弦积分	
FresnelC		
EllipticK	第一类完全椭圆积分	
EllipticE	第二类完全椭圆积分	
sinc	$\frac{\sin x}{x}$	
SinhIntegral	双曲正弦、余弦积分	
CinhIntegral		
not	非	$\text{not}(x<0)$
and	与, 或	(a)and(b)
or		(c)or(b)

mod	取模	mod(a,b)返回 a 除以 b 的余数。 余数总是正的。
-----	----	-----------------------------------

## 5.4 高级运算符

SGC v4.4 及以上版本支持了高级运算符。目前支持 Sum, Prod, Int, Diff, fSolve。高级运算符支持嵌套。

例如

$$\sum_{y=0}^{100} \sum_{x=0}^{100} 2x+y$$

输入 sum(sum(2x+y,x,0,100),y,0,100)

$$\begin{aligned} &\text{sum(sum}(2x+y,x,0,100),y,0,100) \\ &\rightarrow 1530150 \end{aligned}$$

### 5.4.1 求和

求和的表达式为：Sum( $f(v),v,\min,\max$ )。这是对变量为  $v$  的  $f(v)$  从  $\min$  到  $\max$  求和。这里的变量可以是任意的。 $\min$ 、 $\max$  也可以是表达式。

例如：要计算

$$\sum_{x=1}^{100} \frac{1}{x^{10}}$$

输入 sum(1/x^10,x,1,100)。

$$\begin{aligned} &\text{sum}(1/x^10,x,1,100) \\ &\rightarrow 1.000994575128 \end{aligned}$$

这是  $\zeta(10)$  的一个很好的近似。

由于

$$\zeta(10) = \frac{\pi}{93555}$$

我们输入

$$\begin{aligned} &(93555\text{sum}(1/x^10,x,1,100))^{0.1} \\ &\rightarrow 3.14159265359 \end{aligned}$$

得到圆周率的前几位小数。

### 5.4.2 求乘积

求乘积的表达式为：Prod( $f(v),v,\min,\max$ )。这是对变量为  $v$  的  $f(v)$  从  $\min$  到  $\max$  求积。

例如：要计算

$$\prod_{y=1}^{10} y$$

输入  $\text{prod}(y,y,1,10)$ 。

$$\begin{aligned} \text{prod}(y,y,1,10) \\ \rightarrow 3628800 \end{aligned}$$

这计算  $10!=3628800$ 。

特别地，对于某个  $v$ ，如果遇到了  $f(v)=0$ ，会立即停止这个乘积的计算。利用此性质，可以做大于 2 的质数判断。

注意：这里的  $n$  必须大于 2。这里我们使用朴素算法判断质数。要确定  $n$  是否是一个质数，我们可以遍历从 2 至  $\lceil \sqrt{n} \rceil$  的整数来试除。如果某大于等于 2 的整数  $v$  使得  $\text{mod}(n,v)=0$ ，那么这样的  $n$  就不是质数。

综上所述，我们总结出以下表达式：

$$\text{isPrime}(n) = \prod_{v=2}^{\lceil \sqrt{n} \rceil} \text{not}(\text{mod}(n,v)==0)$$

输入  $\text{prod}(\text{not}(\text{mod}(n,v)==0),v,2,\text{ceil}(\text{sqrt}(n)))$ ，并令  $n=43$ ，得到

$$\begin{aligned} n=43 & \quad \vdots \\ \text{prod}(\text{not}(\text{mod}(n,k)==0),k,2,\text{ceil}(\text{sqrt}(n))) \\ \rightarrow 1 & \quad \vdots \end{aligned}$$

但是令  $n=20$ ，得到

$$\begin{aligned} n=20 & \quad \vdots \\ \text{prod}(\text{not}(\text{mod}(n,k)==0),k,2,\text{ceil}(\text{sqrt}(n))) \\ \rightarrow \text{false} & \quad \vdots \end{aligned}$$

不难得出，43 是质数，而 20 不是。

如果希望纳入 2 的判断，可以输入：

`ifelse(n==2,true,prod(not(mod(n,v)==0),v,2,ceil(sqrt(n))))`

### 5.4.3 求积分

SGC 中，积分使用 Gauss-Kronrod 自适应数值积分。SGC 默认会用数值方法判断积分的敛散性。这可以在设置中关闭。同样地，SGC 可以设置数值积分的递归上限。某些波动较大或者积分结果很大的函数，若得到了错误结果，可以适当调高递归上限。

求积分的表达式为： $\text{int}(f(v),v,\text{min},\text{max})$ 。这是对变量为  $v$  的  $f(v)$  从  $\text{min}$  到  $\text{max}$  求积分。

例如：求  $\int_0^3 t^2 dt = 9$ 。

输入  $\text{int}(t,t,0,3)$ ，得到

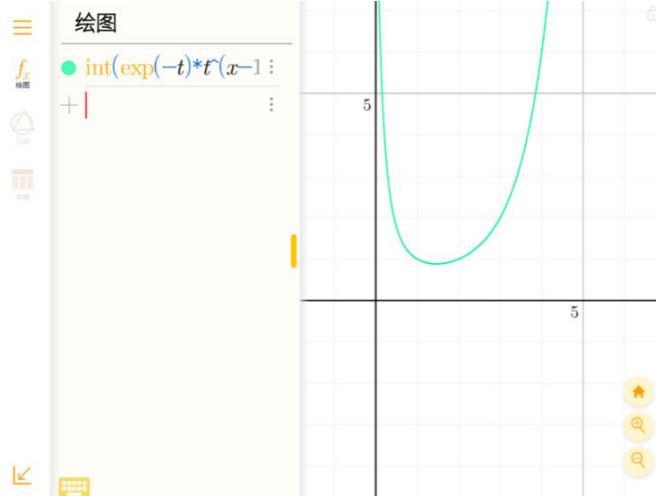
$$\begin{aligned} \text{int}(tt,t,0,3) \\ \rightarrow 9 \end{aligned}$$

例如：gamma 函数的定义式

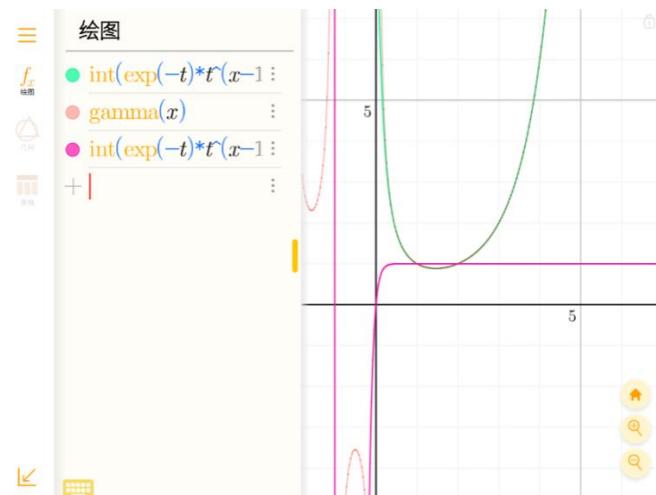
$$\text{gamma}(x) = \int_0^{+\infty} e^{-t} t^{x-1} dt$$

由于实际上取不到 $+\infty$ , 我们用 99 代替。

输入  $\text{int}(\exp(-t)*t^{(x-1)}, t, 0, 99)$ , 得到函数



对比实际的  $\text{gamma}(x)$ , 得到



下面的函数是  $y = \frac{1}{x!} \int_0^{99} e^{-t} t^{x-1} dt$  .

#### 5.4.4 求导

求导的表达式为：  $\text{Diff}(f(v), v, v_0)$ 。

这是对变量为  $v$  的  $f(v)$  在  $v=v_0$  处求导。

求导使用中心差分：

$$\text{diff}(f(x)) = \left. \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} \right|_{\Delta x=10^{-5}}$$

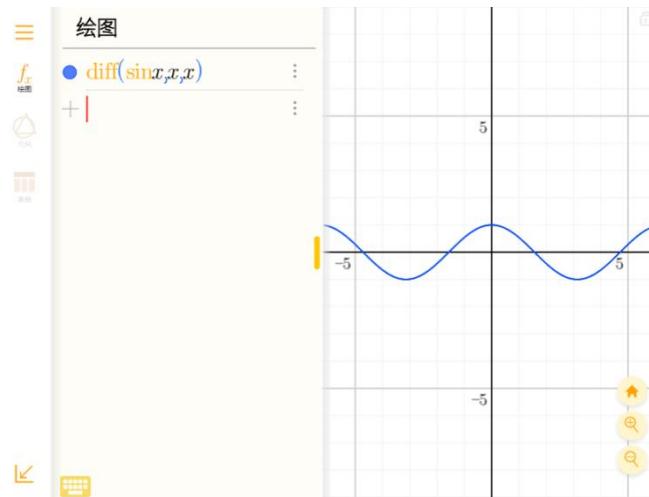
例如：要计算

$$\frac{\partial}{\partial x} \sin(x) \Big|_{x=1}$$

输入 `diff(sin(x),x,1)`。

`diff(sin(x),x,1)`  
→ 0.540302305863

若要计算  $\sin(x)$  的导数，输入 `diff(sin(x),x,x)`。



#### 5.4.5 fSolve

这函数是使用 Newton 方法求数值解的函数。

例如  $f(x)=x^2-2$  的零点为

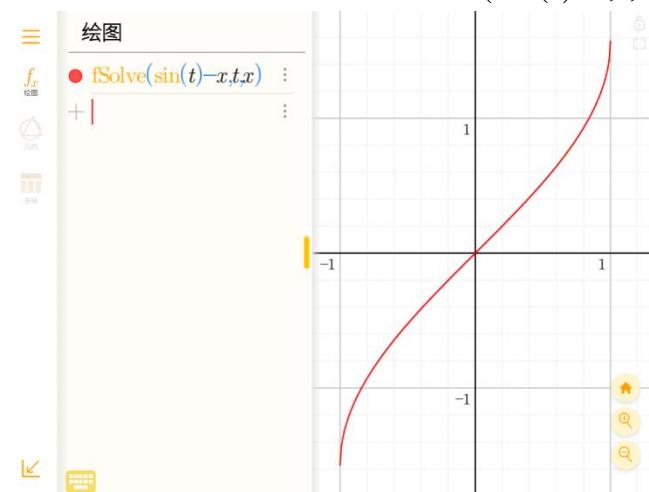
$x:=[1.414213562373095, -1.414213562373095]$ ,

我们想求这函数靠近 1 的零点，输入 `fSolve( $x^2-2,x,1$ )`，获得输出为 1.414213562373095.

`fSolve( $x^2-2,x,1$ )`  
→ 1.414213562373

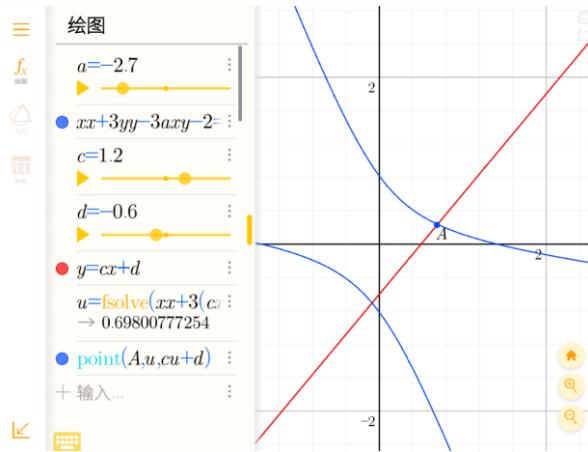
例如函数  $g=\arcsin x$  是  $f=\sin x$  的反函数，这也可以用 fSolve 描述。

这实际上在求解关于  $t$  方程  $\sin t=x$ 。输入 `fSolve( $\sin(t)-x,t,x$ )` 得到图像



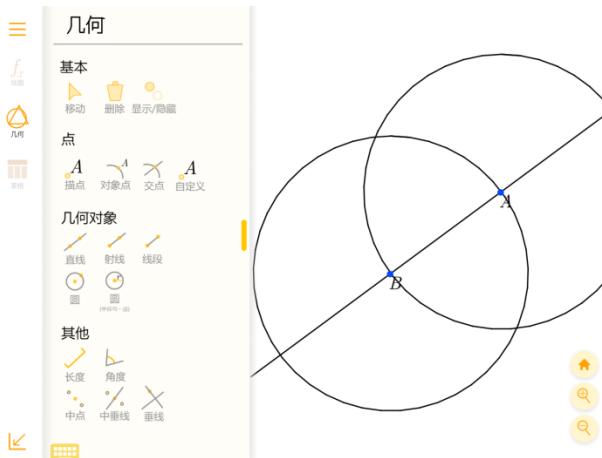
同样地，fSolve 可用于圆锥曲线交点以及任意函数交点的探究。

例如：想要探究圆锥曲线  $xx+3yy-3axy-2=0$  与  $y=cx+d$  的交点问题，若需要获取靠右的一个交点，可以联立两方程，用  $u=fSolve(xx+3(cx+d)^2-3ax(cx+d)-2,x,1)$  来确定交点横坐标。如图中 A 点即为待求交点。



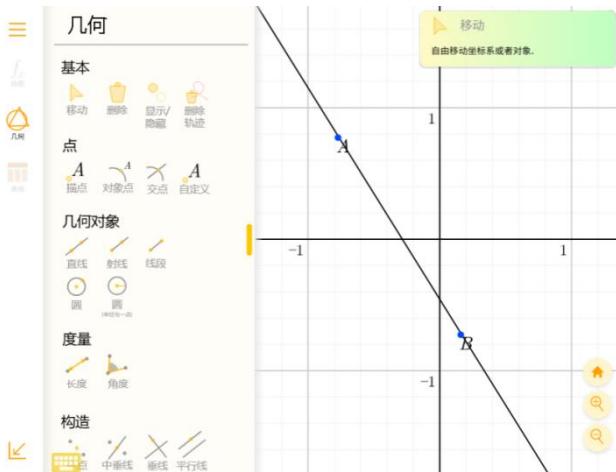
## 6 几何

SGC 提供了易用的几何功能,可以绘制点、线、圆、文本等对象。

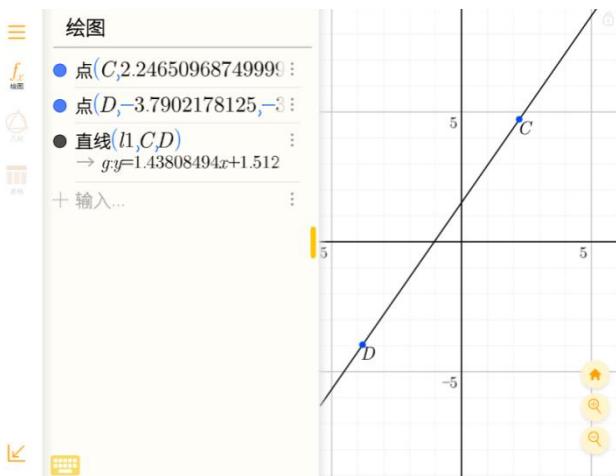


### 6.1 基本操作

左边模式栏选择“几何”，出现如图所示的几何页面。使用鼠标拖动或者鼠标滚轮可以移动几何功能页面。点击一个几何功能，右侧会弹出提示。您可以根据提示完成操作。



点击“绘图”，几何内容会在输入区显示。



在几何内容中，直线的解析式会显示出来，并且以 g:开头。

几何支持的功能如下：

(由于 scratch 不分大小写,这些函数也不需要区分大小写)

函数名	解释
点(名称,x,y)	在(x,y)处创建点
对象点(名称,所在对象[,因子])	在对象上创建一个点；如果填入了因子，那么对象点的位置由这个因子确定
交点(对象 1,对象 2,名称 1,名称 2)	确定两个对象的交点。如果有两个，那么两个交点均被创建；不需要的交点使用?填充
直线(名称,点 1,点 2)	创建过点 1 和点 2 的线
射线(名称,点 1,点 2)	
线段(名称,点 1,点 2)	
圆(名称,圆心,圆心外一点)	根据两个点创建一个圆
圆规(名称,圆心,点 1,点 2)	创建一个圆，但半径为点 1 到

	点 2 的距离
距离(变量名,点 1,点 2)	测量点 1 到点 2 的距离并赋值给变量
角度(变量名,点 1,点 2,点 3)	测量 点 1-点 2-点 3 所成的角度，并赋值给变量
中点(名称, 点 1, 点 2)	创建两点的中点
中垂线(名称, 点 1, 点 2)	创建过两点中点的垂线，使得该直线上任意一点到这两点的距离均相等
垂线(名称, 直线 1, 点 1)	过点 1 做直线 1 的垂线
平行线(名称, 直线 1, 点 1)	过点 1 做直线 1 的平行线
旋转(旋转中心,旋转对象,名称,旋转角)	将旋转对象绕着旋转中心旋转一定角度
对称(对称轴,对称对象,名称)	将对称对象沿着对称轴做对称
是否垂直(直线 1, 直线 2)	检验两直线是否垂直
是否平行(直线 1, 直线 2)	检验两直线是否平行
多边形(poly, 点 1,点 2, ...,点 n)	创建名称为 poly 的多边形
面积(变量名, 对象)	测量多边形或者圆的面积
角平分线(名称,点 1,点 2,点 3 [,等分数 n])	做角的(n)等分线, 点 2 是顶点

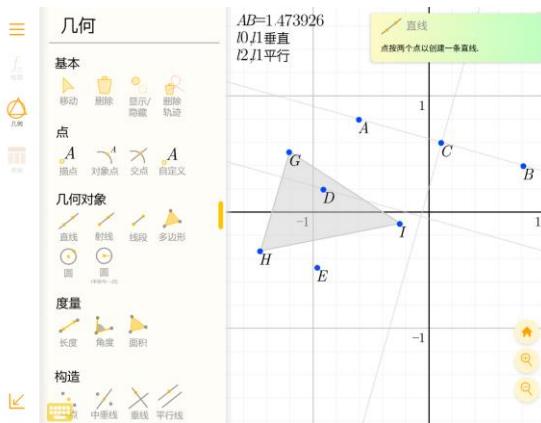
在 4.7.3 以上版本中，几何函数支持了英文：

函数名	英文名
点(名称,x,y)	point(名称,x,y)
对象点(名称,所在对象[,因子])	objpoint(名称,所在对象[,因子])
交点(对象 1,对象 2,名称 1,名称 2)	inter(对象 1,对象 2,名称 1,名称 2)
直线(名称,点 1,点 2)	line(名称,点 1,点 2)
射线(名称,点 1,点 2)	ray(名称,点 1,点 2)
线段(名称,点 1,点 2)	segment(名称,点 1,点 2)
圆(名称,圆心,圆心外一点)	circle(名称,圆心,圆心外一点)
圆规(名称,圆心,点 1,点 2)	compass(名称,圆心,点 1,点 2)
距离(变量名,点 1,点 2)	distance(变量名,点 1,点 2)
角度(变量名,点 1,点 2,点 3)	angle(变量名,点 1,点 2,点 3)
中点(名称, 点 1, 点 2)	midpoint(名称, 点 1, 点 2)
中垂线(名称, 点 1, 点 2)	perpbisector(名称, 点 1, 点 2)

垂线(名称, 直线 1, 点 1)	perpline(名称, 直线 1, 点 1)
平行线(名称, 直线 1, 点 1)	paraline(名称, 直线 1, 点 1)
旋转(旋转中心, 旋转对象, 名称, 旋转角)	rotate(旋转中心, 旋转对象, 名称, 旋转角)
对称(对称轴, 对称对象, 名称)	reflect(对称轴, 对称对象, 名称)
是否垂直(直线 1, 直线 2)	isperp(直线 1, 直线 2)
是否平行(直线 1, 直线 2)	isparallel(直线 1, 直线 2)
多边形(poly, 点 1, 点 2, ..., 点 n)	polygon_g(poly, 点 1, 点 2, ..., 点 n)
面积(变量名, 对象)	area(变量名, 对象)
角平分线(名称, 点 1, 点 2, 点 3 [, 等分数 n])	anglebisector(名称, 点 1, 点 2, 点 3 [, 等分数 n])

## 6.2 创建几何对象

创建对象会有几种提示。例如选择“直线”工具，需要选择两个点。这时，坐标系上所有不是点的几何对象都将变为灰色，便于提示选择的对象。



## 7 文本

### 7.1 普通文本

在坐标系中显示文本，使用以下格式：

text:<text>,x,y,size,type

其中, type 是 0 或者 1。0 表示字号大小和坐标系缩放相关,1 则表示不相关。

例如: 想要在坐标系(0,0)处以“1”号字显示文本: “你好 SGC! ”

输入: text: "你好 SGC! ",0,0,1,0

效果如图:



## 7.2 TeX

SGC 中有一个 STeX 终端. 部分遵循 TeX 语法规则,但是所有\关键字尽量用`代替。TeX 文本大小不会随坐标系缩放改变。

`frac{a}{b}	分数 a/b
`sqrt{a}	根号下 a
`nroot{u}{x}	u 次根号下 x
`overset{v}{u}	u 上的 v
`underset{v}{u}	u 下的 v
a^{b}	a 的 b 次幂
a_{b}	a 下标 b
`int	积分号
`infty	无穷大符号
`sum	求和符号
`prod	求积符号
`partial	偏导符号
`pi	圆周率
`longeq	长等号(==)
`+空格	空格
`mathrm	正体
`lfloor	左侧 floor
`rfloor	右侧 floor
`lceil	左侧 ceil
`rceil	右侧 ceil

括号如果匹配, 会自动变成合适的高度, 不需要输入`left, `right.

```
`left{  
`right{  
`left[  
`right[  
`left|  
`right|
```

```
{  
}  
[  
]
```

左侧绝对值符号  
右侧绝对值符号

支持一些连字：

```
=>  
->  
==
```

```
`Rightarrow  
`rightarrow  
`longeq
```

如果要在坐标系中显示 TeX 文本，输入：

latex:<latex>,x,y,size,0

或者仍然使用 text 标签

text:<latex>,x,y,size,t

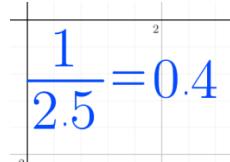
这将会以 TeX 解析此文本，

比如变量  $a=2.5$ ，在(0,0)处以 0.6 点 TeX 字号显示公式

$$\frac{1}{\text{<变量 } a>} = \text{<1/a 的值>}$$

输入：text: " $\frac{1}{\text{&a}\&}=\&1/\text{a},0,0,0.6,\text{t}$

这会先计算文本表达式 " $\frac{1}{\text{&a}\&}=\&$ " 获得输出，进而调用 TeX 解析结果。例如  $a=2.5$  时，" $\frac{1}{\text{&a}\&}=\&$ " 的结果为 " $\frac{1}{2.5}$ "，再使用 TeX 进行解析、渲染字符串 " $\frac{1}{2.5}$ "。



拖动滑动条能看到实时刷新的 TeX 公式。

在(0,0)处以 0.6 点 TeX 字号显示公式

$$\frac{e^{i\pi}+1}{e^{i\pi}+1}=0$$

下面是一些例子：

## 显示

$$\int_0^1 \frac{1}{1+x^3} dx = \frac{2\sqrt{3}\pi + \ln 64}{18}$$

## 代码

```
text:"`int`subsc`_`1`{0}`{1}`{0}`{1}`{1+x^3}` `mathrm{d}`  
`frac{1}{1+x^3}` `mathrm{d}`  
x=`frac{2`sqrt{3}`pi+ln`  
64}`{18}`,0,0,0.2,t
```

## LaTeX

```
\int^1_0 \frac{1}{1+x^3} dx  
\mathrm{d}`  
x=\frac{2\sqrt{3}\pi+\ln{64}}{18}
```

$$f(x) = a_0 + \sum_{n=1}^{\infty} \left( a_n \cos \frac{n\pi x}{L} + b_n \sin \frac{n\pi x}{L} \right)$$

```
text:"f(x)=a_0+`dwsc{`upsc{`sum}`{`infty}}{n=1}(a_n cos `frac{n`pi x}{L}+b_n sin `frac{n`pi x}{L})",0,0,0.2,t
```

```
f(x)=a_0+\sum^{`infty}_{n=1}(a_n \cos \frac{n \pi x}{L}+b_n \sin \frac{n \pi x}{L})
```

$$\sqrt{x_1^2 + x_2^2}$$

```
text:"`sqrt{x^2_{1}+x^2_{2}}",0,0,1,t
```

```
\sqrt{x^2_{1}+x^2_{2}}
```

$$\frac{\partial x^x}{\partial x} = x^x (1 + \ln x)$$

```
text:``frac{`partial x^x}{`partial x}=x^x{`x}(1+ln x)",0,0,0.2,t
```

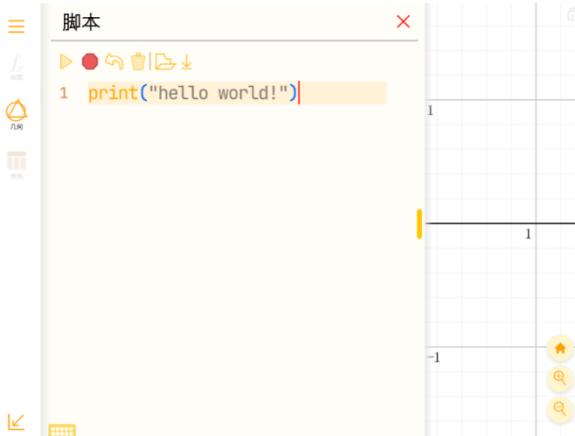
```
\frac{\partial x^x}{\partial x}=x^x{`x}(1+\ln x)
```

## 8 脚本

SGC 搭载简单的 SGC Script 语言。SGC Script 是一个基于栈机的编译语言，能操作 SGC 底层，修改 SGC 设置，做出简单易懂的演示动画。

### 8.1 脚本编辑器

在高级功能中选择进入编辑器，即可打开脚本编辑器。



(示例 1)

脚本编辑器采用 JetBrains Mono 字体，支持一些连字。例如

```
1 a=0
2 while a<=100{
3   if a!=15 || a!=26{
4     print(a)
5   }
6 }
```

您可以上下左右移动光标，按键盘或者虚拟键盘插入字符，按下 enter 新建一行。当前光标所在的那一行会以不同颜色标注。



上方为操作区按钮。按下第一个图标运行您的脚本，按下第二个图标强行停止您的脚本。按下第三个图标返回脚本编辑器，按下第四个图标清空脚本。按下第五个图标打开脚本，第六个保存脚本。

例如运行示例 1 代码，显示



在脚本编辑器中，文字会高亮。注释为灰色，其他和函数一样。SGC Script 关键字为紫色，标记关键字为蓝色。

在控制台中，Info(Information)类型以灰色显示，Normal 以默认颜色显示，Warning 以橙色显示，Error 以红色显示。

关键字表	
if	如果
else	否则
while	当
print	输出
break	打断
define	定义
list	列表
system	系统
graph	图形

system、graph 是系统库，不需要额外引用。

## 8.2 语法

### 8.2.1 数据类型与运算符

支持的数据类型为：

数(number)	1
不区分 float 与 int	-2.5 0.333333333333333
字符串(String)	“hello SGC!” “0.3333333333333” “SQY!+/*/”
布尔值(Boolean)	true false

支持的运算符为：

+,-,\*,/,^,&,<,>,==,<=,>=,!%,&&,||, and, or, not, if

数学函数为：

sin, cos, tan, ln, log, sqrt, abs, floor, ceil, round, mod, min, max, approx, random

其他函数为：

substr, length, sreplace, load, fetchfrom, str, num, getinput, (item, listlength), indexOf  
不区分大小写。部分函数说明如下：

函数名	解释
substr(string,a,b)	字符串 string 的第 a 到 b 个字符
length(a)	对于列表 a, 返回 a 的长度
	对于字符串 a, 返回 a 的字符数
	对于数 a, 返回数化为字符串之后的字符数
sreplace(string,a,b,c)	将字符串 string 的第 a 到 b 个字符替换为 c
find(string,sub_str)	寻找 string 中 sub_str 第一次出现的位置. 如果没有找到, 返回 0
cfind(string,sub_str)	寻找 string 中 sub_str 出现的次数
load()	从本地加载文件作为字符串
input()	从外部输入字符串
fetchfrom(url)	获取 url 的 html 代码或者内容
getinput(i)	获取 SGC 系统输入的第 i 项
approx(a,b)	将 a 保留 b 位小数

String 类型会有"标记。如 text 不是 string, 而"text"是 string.

空格不是关键字。因此下面两句话等价：

```
a=0+114514
a = 0 + 114514
```

## 8.2.2 变量与列表

SGC Script 目前只支持全局变量，列表的逻辑和 Scratch 相同，与一般的编程语言不同。只支持一维数组，强行创建多维数组会出现问题。

可以定义变量：如

```
a=1
b="text"
```

这之后，a, b 的值可以随意调用。

定义列表：

```
c=[0,1,2,3,4,5]
d=[]
```

d 是一个空列表。请注意：列表索引从 1 开始。

v4.3 之后的版本支持了负索引，例如 -1 代表倒数第一项：

```
list.replace(c, 2, "aaa")      //替换列表 c 第二项为"aaa"
list.insert(c, 2, "bbb")        //列表 c 第二前插入"bbb"
list.append(c, 114514)          //列表 c 末尾添加 114514
list.delete(c, 2)               //删除列表 c 第二项
list.sortb(c)                  //从小到大排序列表 c
list.sortl(c)                  //从大到小排序列表 c
```

```
c=[0,1,2,3,4,5]
print(getlist(c))              //以字符串形式获得列表 c
print(combine(c))              //以字符串形式拼接列表 c
print(item(c, 2))              //获取列表 c 的第 2 项
```

```
·> "[0,1,2,3,4,5]"
·> "012345"
·> 1
```

v4.3 之后的版本更新了语法糖，做出如下优化：

```
c=[0,1,2,3,4,5]
c[1]=114514
c.append(15)                  //c 末尾添加 15
c.delete(2)                    //删除列表 c 的第 2 项
print(c.list)                  //以字符串形式获得列表 c
print(c[2])                    //获取列表 c 的第 2 项
print(c.length)                //获取列表 c 的长度
```

```
·> "[114514,1,3,4,5,15]"
·> 1
·> 6
```

### 8.2.3 循环

SGC Script 目前只支持 while 循环。其语法如下：

```
a=0
while a<=100{
    a+=1
}
```

循环支持嵌套。使用 break 能打断当前循环，跳入下一循环。

```
a=0
while a<=100{
    a+=1
    if a==50{
        break
    }
}
print(a)
```

## 8.2.4 如果

如果语法如下：

```
k=input()
if k==0{
    print("zero")
}

j=input()
if j>0{
    print("greater than zero!")
}else{
    print("less than zero or equal to zero!")
}
```

## 8.2.5 定义函数

SGC Script 目前支持定义返回值。函数中定义的变量都是全局变量。函数支持递归。定义函数语法如下：

```
r=0 // y must be a integer.
define fastpow(x,y){
    if y==1{
        r=x
    }else{
        if mod(y,2)==1{
            fastpow(x,(y-1)/2)
            r=r*r*x
        }else{
            fastpow(x,y/2)
            r=r*r
        }
    }
}
fastpow(2,100)
print(r)
```

使用返回值的代码如下：

```
define fastpow(x,y){  
    if y==1{  
        return x  
    }else{  
        if mod(y,2)==1{  
            return fastpow(x,(y-1)/2)^2*x  
        }else{  
            return fastpow(x,y/2)^2  
        }  
    }  
}  
print(fastpow(2,100))
```

上述代码进行快速幂的计算，使用递归的方法。结果会输出  $2^{100}$ ：

```
> 1.2676506002282317e+30
```

## 8.3 操作 SGC 底层

### 8.3.1 输入操作

您可以使用 SGC Script 操作 SGC 的底层。这需要使用 system 库。这包括：新建输入、移除输入、修改输入颜色。

新建输入可以在当前输入区最后添加一个输入。例如：

```
system.addinput("y=x^2")
```

这将添加函数  $y=x^2$ 。注意：addinput 接收的参数必须是一个字符串，如果传入数，那么会将数转换为字符串。

如果想指定添加输入的颜色、是否显示，那么：

```
system.addinput("y=x^2", "00999900", 1)
```

这会指定添加的函数颜色为红色。

删除指定的输入，可以使用：

```
system.delinput(3)
```

这将删除第 3 个输入。如果不传入参数，那么将会清除所有的输入。

如果想设置输入某一项的颜色为  $a$ ，可以使用：

```
a="62999900"  
system.setinputcolor(2, a)
```

这段代码会把输入的第 2 项颜色设置为蓝色。

在运行代码时，控制台默认会刷新。使用 refresh 能控制控制台是否刷新，如果参数得到 true，那么刷新，反之不刷新。使用 hidewarning 能隐藏警告。

```
system.refresh(false)
system.hidewarning(false)
```

### 8.3.2 图形操作

SGC Script 能控制 SGC 的默认坐标系，包括控制位置、缩放大小。一般地，坐标系默认缩放大小为 100，位置为(0,0)。调整的是坐标系的偏移位置。

要移动坐标系，输入：

```
system.moveto(1.5, 1)
```

这会把坐标系原点向左移动 1.5 个单位，向上移动 1 个单位。

```
system.scale(sys.scale/2.6)
```

这会把单位长度缩小 2.6 倍。

system 中，相对于真正的底层，存储的是负向偏移量；而相对于脚本，存储的是正向偏移量。输入 sys.dx、sys.dy 获取当前 x、y 轴正向偏移量。输入 sys.scale 获取当前缩放值。

### 8.4 系统常数

使用 inputlen 获取当前输入的总项数。使用 dateto2000 获取当前时间到 2000 年的天数。这可以用来制作计时器。

例如：

```
time_stamp = dateto2000
define reset(){
    time_stamp = dateto2000
}
define getsec(){
    return (dateto2000 - time_stamp)*86400 //一天 86400 秒
}
```

使用 sys.time 获取 24 小时制时间。但是：这时间按照 $(时 + \frac{分}{60} + \frac{秒}{3600})$ 来计算。因此，获取当前时间的时、分、秒需要额外处理。一个获取当前“时”、“分”的函数如下：

```
define get_h(){
    return floor(sys.time)
}
define get_m(){
    return floor((sys.time-floor(sys.time))*60)
}
```

使用 mouse.x、mouse.y 获取当前鼠标在坐标系中的坐标。使用 mousedown 倾测鼠标是否按下。

## 8.5 图形库

SGC Script 有 graph 图形库。这个库的一些对象可以选择填入颜色，也可以不填。如果不填颜色，将使用默认颜色（红色）代替。语法如下：

使用 clean 清除脚本图层。

```
graph.clean()
```

若要绘制一个点，使用 point：

```
graph.point(1, 2)
```

这将在坐标系中(1,2)处画一个大小为 1px 的点。可以选择颜色：

```
graph.point(1, 2, "00999900")
```

这将在坐标系中(1,2)处画一个大小为 1px 的、红色的点。

若要绘制线段，使用 line：

```
graph.line(1, 2, 4, 5)
```

这将在坐标系中连接(1,2)、(4,5)。

graph 可以渲染文字。

使用 font 函数指定字体样式。font(1)是正体，font(2)是斜体。例如：

font 1 : regular font  
font 2 : italic font

使用 text 函数显示文字。（text 函数的语法参见章节 7.1）

使用 rect 填充矩形。

```
graph.rect(1, 2, 4, 5)
```

使用 pencolor 设置默认画笔颜色值，接受参数范围是 0..99。

## 8.6 动画播放

SGC Script 有 frame 动画库用于操作 SGC 的逐帧动画。

speed 设置动画速度，默认值为 1。

setmax 设置最大帧数，最大为 99999 帧。

start 开始动画，stop 停止动画。

```
frame.speed(1.14514)  
frame.setmax(6000)  
frame.start()  
frame.stop()
```

## 8.7 声音播放

SGC Script 有 sound 声音库用于操作 SGC 的音频播放。

loadfrom 可以从网络加载音频。

```
sound.loadfrom("https://sqy419.axolotlpower.com/download/assets/music/omr.mp3", "test") //从链接加载音频作为 test
```

loadsound 可以从本地加载音频。

```
sound.loadsound("test") //从本地加载音频命名为 test
```

playsound 函数用于播放。

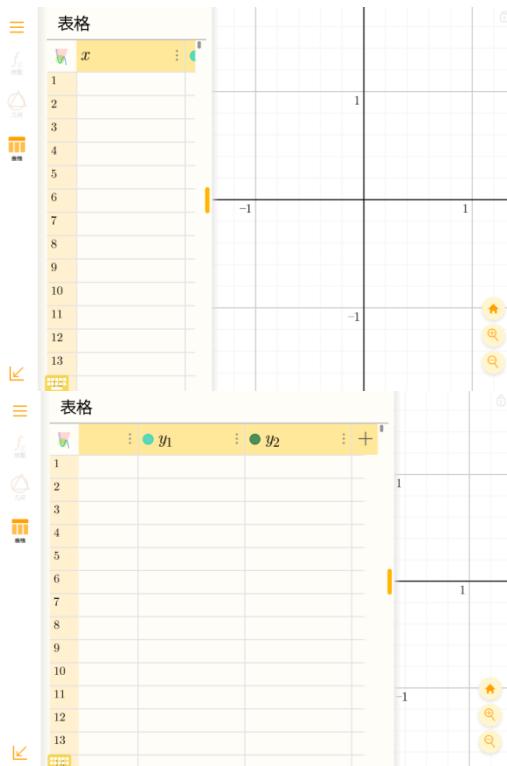
```
sound.playsound("test")
```

## 9 统计

SGC 支持简易的统计功能，其拥有 16 组双变量统计。

### 9.1 统计表概述

左边模式栏选择表格，出现如图所示的统计页面。您可以使用鼠标拖动统计表。统计表默认有 3 列。



如果在某一列右侧出现了“+”符号，点击它可以新建一列。

### 9.2 数据的输入

点击一个单元格即可进行输入。被点击的单元格会高亮显示。

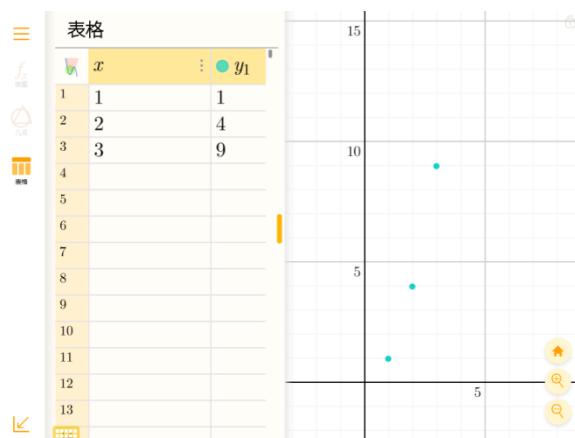
	x	y <sub>1</sub>
1	111+222	
2		
3		
4		

如果输入包含错误，会出现红色警告图标。

	x	y <sub>1</sub>	y <sub>2</sub>
1	111+222+▲		
2	1/▲		
3	1/3		
4			
5			

输入方式与输入函数同理，不再赘述。

### 9.3 数据的显示



(9.3-1)

在输入数据后，坐标系中会出现数据点。这些点的颜色是这一列左侧小圆片的颜色。

只有  $x$ 、 $y$  均有定义的点才会显示。有任何一个不完整的，这个点不会显示。



例如去掉了(3,9)的 9，这个点就没有显示。

## 9.4 列操作

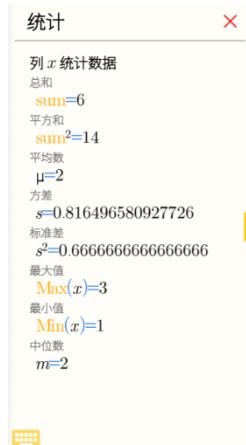
点击每一列旁边的三个小点，会出现小窗口。



点击“统计”进入列统计模式（参见 9.5）。点击“导入”可以导入数据，数据需要以空格间隔。例如某一列为[1,2,3,4,5]，导入输入 1 2 3 4 5 即可。点击“清除列数据”清除这一列的数据。在点击  $y_n$  列的时候才会显示“回归”功能。点击“回归”启动  $y_n$  列的回归计算。点击“样式”可以修改数据的显示颜色。

## 9.5 列统计计算

在点击“统计”后，会对这一列进行统计。出现如图(9.5-1)所示页面。这统计具有如下功能：求 $\sum x$ ,  $\sum x^2$ ,  $\bar{x}$ , 方差, 标准差,  $\text{Min}$ (最小值),  $\text{Max}$ (最大值),  $\text{Mid}$ (中位数)。如果统计数据有误（例如空列表），会给出如图(9.5-2)提示。



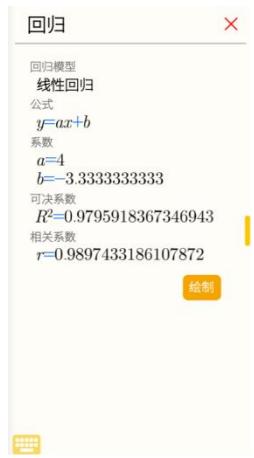
(9.5-1)



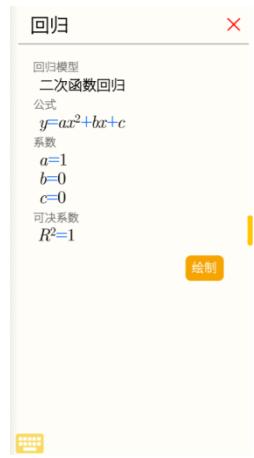
(9.5-2)

## 9.6 列回归计算

在点击“回归”后，会对这一列和  $x$  列进行回归计算，出现如图(9.6-1)所示页面。这界面会显示回归的参数，包括回归系数和可决系数。您可以自己选择回归模型。默认是线性回归，支持线性、二次、指数( $ae^{bx}$ )、对数回归( $a+b\ln x$ )。点击“回归模型”一栏就可以进行选择。如对(9.3-1)中的列进行二次回归计算，如图(9.6-2)。

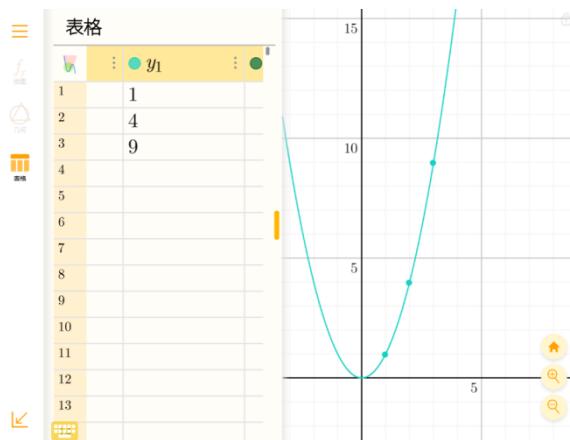


(9.6-1)



(9.6-2)

回归计算页面下方有一“绘制”按钮。点击即可将回归结果绘入坐标系。



## 10 设置

打开系统设置后，设置分为三个板块。

### 10.1 全局

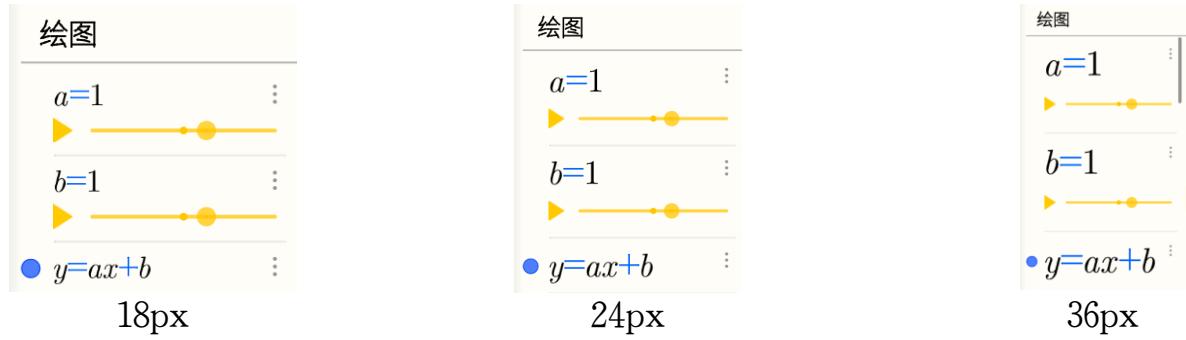


如图。全局设置的符号是一个类似地球的图案。全局里包含一些基本设置。

#### 10.1.1 输入显示

字号：点击字体大小后面的数字可以修改字体大小，最大是 36 点字体。下面是不

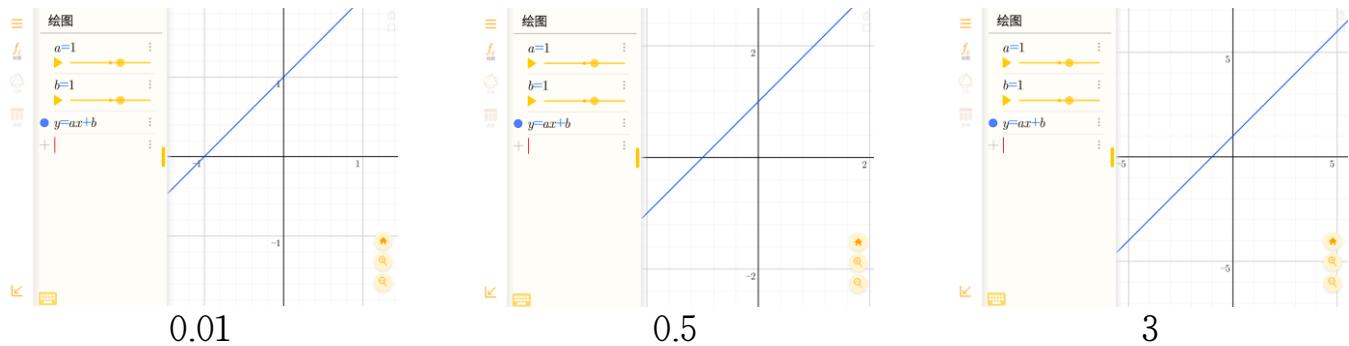
同字号的对比。



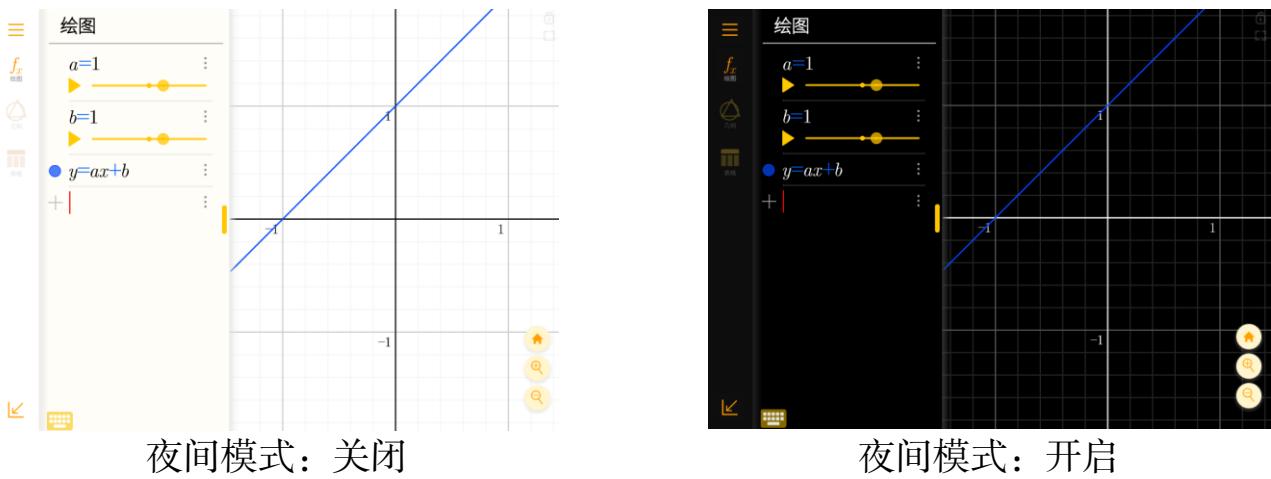
高亮显示前面已经提到过。(参见章节 2.3)

### 10.1.2 缩放

修改缩放速度后，鼠标滚轮的滚动与画布放大缩小按钮的效果将会改变。下面是只点击一次缩放按钮，使用不同缩放速度的效果（坐标系均回到初始位置）：

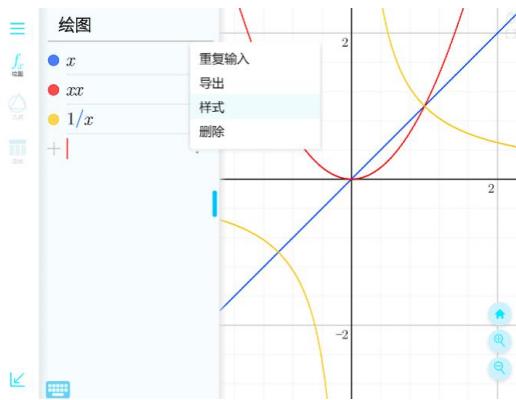


主题下可以打开夜间模式。打开后，软件以黑色为主题，但是绘图区的输入颜色不会取反。在输入区，黑色的字会变成白色的字。



修改坐标系参数后，右边的图形也会刷新。网格样式点击右侧文本即可修改。

您还可以在此修改颜色主题。SGC 默认颜色主题是黄色，对应颜色值为 13. 修改成 54 后效果如图。



## 10.2 坐标系与网格



如图。这个设置可以调整一些有关坐标系与网格的设置。

这里只给出“网格样式”的示例：



主要



主和次



网格点

如果不显示坐标轴，那么轴的编号会靠侧面显示。

如果打开“显示鼠标坐标”，那么鼠标位置会出现一个十字，这可以指示鼠标的位  
置。



### 10.3 绘图



如图。绘图里包含一些关于图形的设置。

点击断点判断的开关可以关闭或者打开断点判断。由于 SGC 采用描点法作图，因此断点判断对图像质量有很大影响。目前 SGC 能判断一般的显函数的跳跃间断点、无穷间断点和可去间断点。隐函数、参数方程、极坐标函数同样支持断点判断。

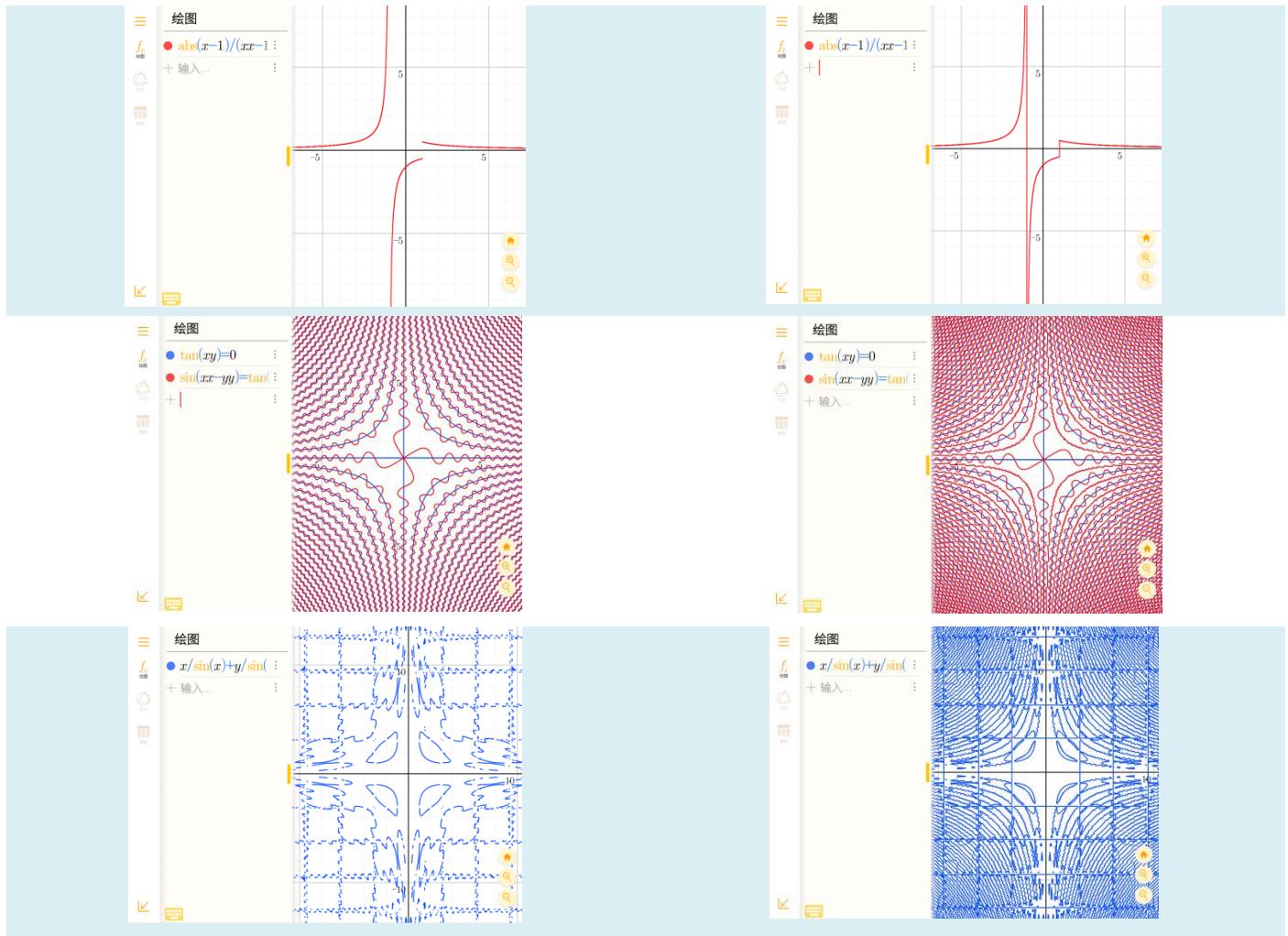
函数精度默认为 480，这表示在 480 个单位中以步长  $480/480=1$  采集数据点，并连成平滑的线显示在坐标系中。点击后面的数字即可修改函数精度，最大 900000。精度越高，描的点越多，计算就越慢，但图像更精确。

隐式精度为 80，实际是计算横 80 单位、竖  $80*0.75=60$  个单位、共  $80*60=4800$  个点。由于使用区块加载，区块大小为  $120*120$ ，所以隐式精度必须是 4 的倍数。推荐的精度为 240 至 360，使用 1440 为精度基本上所有图像画的都比 desmos 清楚，但这可能会造成计算缓慢。每次改变精度后，会先初始化隐式缓存单元。

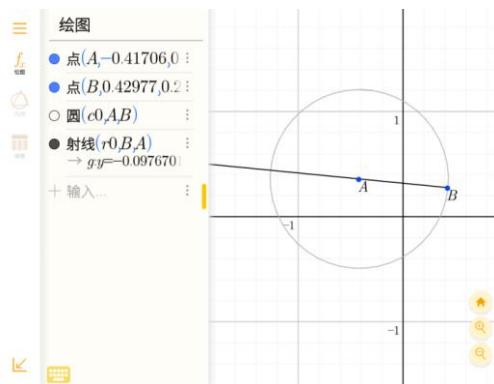
参数方程和极坐标的设置方法同上。（输入框使用参见 4.4）

下表列出了断点判断对图像质量的影响。

有断点判断	无断点判断



几何设置中,几何-显示隐藏对象打开后,原本隐藏的看不见的对象会以灰色展示,但是不能被点击到。您可以点击输入区左侧圆形再次显示该对象。如果打开吸附到网格,那么几何中的点如果距离附近某个网格小于4,将会吸附到最近的网格,同样可以吸附到格点。



如图,圆  $c0$  被隐藏,以灰色显示。

## 10.4 性能

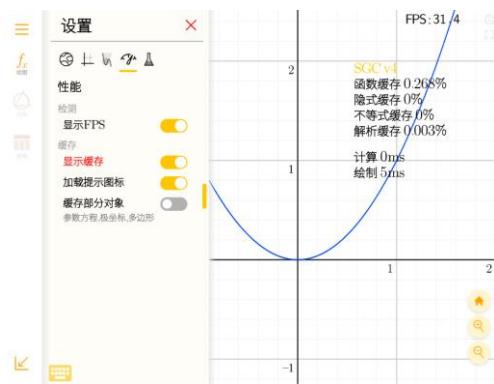


如图。性能设置里包含一些有关性能的设置。

FPS 显示打开后，FPS 数值会显示在右上角。

显示缓存能显示当前的缓存数据，包括“函数缓存”“隐式缓存”等，还会显示计算与绘制的时长。

提示图标是一个小沙漏，显示在右上角，如果出现了这个图标，表明正在计算。刷新全局存档时，最上方会出现黄色进度条。

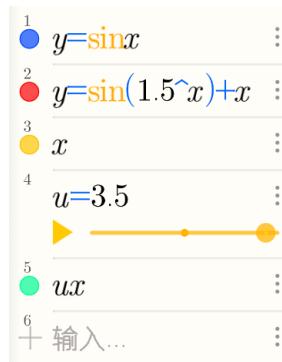


缓存部分对象打开后，除非进行了强制刷新全局存档，这些对象将不会在计算时被计算。这样可以提升图表计算速度，但可能导致图像错误。在参数方程包含正在实时调整的变量时，请不要开启此选项。

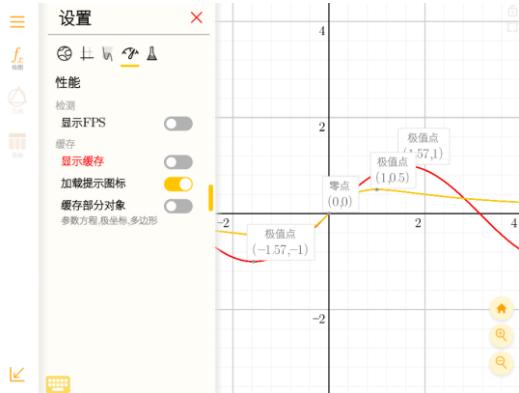
## 10.5 实验功能与其他

如图。其他里包含其他设置和实验性设置。

打开输入编号后，输入栏每一栏都会在左侧显示输入编号。



点击显示特殊点可以在平面模式下，点击函数的极值点或者零点，就能显示其具体位置。但是，极值点建议在函数精度>4800 时使用。每次点击输入区后，显示的特殊点都会被清除。



修改变量动画速度后，将会应用到所有变量。

## 10.6 变量设置

点击输入区您想修改的变量右边的三个点，选择设置。



之后进入了变量设置页面。

变量设置页面包含变量名、值等内容。灰色横线以及上面的文字称为文本框单元，如果下面的横线变成了黄色，说明当前正在修改这里的内容，文本框操作方法和输入函数一样。

点击变量名可以修改变量名。

### 10.6.1 变量设置的修改

点击变量值下文本框单元即可修改变量值，点击 enter 或者空白处和其他输入框，当前输入会自动保存并计算。

请注意：如果变量值超出了变量范围，那么会将超出的那个范围替换为变量值的表达式。

点击上限或者下限下文本框单元即可修改范围。请注意：如果范围超出变量值，会将变量值替换为正在修改的范围表达式。比如变量值为 1，却把范围修改成了-1，这会导致变量值变成-1. 如果上限比下限小，或者下限比上限大，那么上下限会交换位置。

## 10.6.2 变量增量

修改变量增量后，变量会从当前值开始，依次增加增量，直到大于上限。动画模式有“往返循环”：变量值从最小增加到最大，又从最大增加到最小，这样重复直到变量停止播放；“单向循环”：变量值从最小增加到最大，又返回最小，再增加到最大，这样重复直到变量停止播放；“单增一次”：从当前值增加到最大，并停止播放。新建的变量默认是往返循环。常值变量不支持动画。

# 11 多边形与逐帧动画

## 11.1 多边形

在 SGC 中，可以创建多边形对象。

输入 `polygon((x1,y1),..., (xn,yn))` 以创建多边形。如果  $(x_1,y_1)=(x_n,y_n)$ ，视为轮廓闭合，那么填充颜色。点的坐标可以包含表达式。

`polygon_p` 函数只支持点输入，点的坐标不可以包含表达式。`polygon_p` 函数可以大大加快解析速度。

注意：`polygon_g` 是几何对象，与这里的 `polygon` 不同。

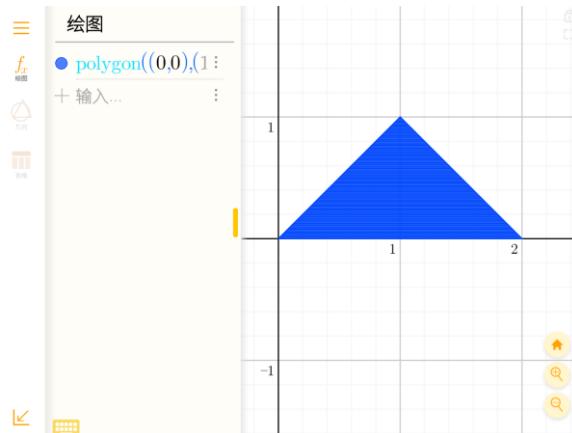
## 11.2 动画

SGC 支持按照帧播放动画。（打开动画控制台参照章节 1.2）

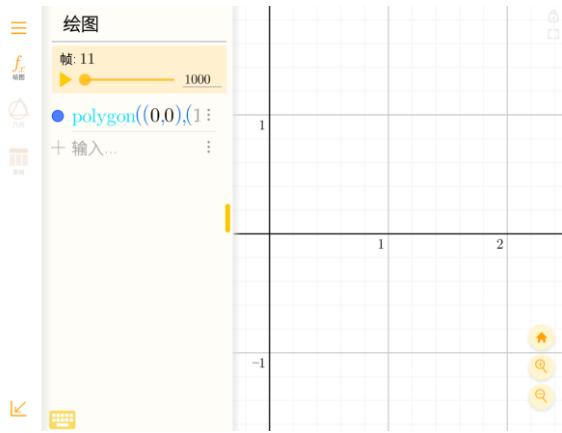
### 11.2.1 创建动画

目前，只有 `polygon`、`polygon_p` 支持动画。在表达式后面加上`#n`，就可以在第  $n$  帧显示这个对象。

例如：输入 `polygon((0,0),(1,1),(2,0),(0,0))`。



这会创建一个三角形，并且一直显示。修改为 `polygon((0,0),(1,1),(2,0),(0,0))#1` 后，三角形不显示了。这时，打开控制台，点击播放按钮，发现三角形一闪而过。



### 11.2.2 开始动画

点击播放按钮开始动画，再次点击可以暂停。

### 11.2.3 设置

点击“帧：\*\*\*”附近可以设置当前帧。点击滑动条右侧的数值可以设置最大帧数。最大为 99999。在设置-绘图-动画中，可以修改动画速度，默认为 1。

### 11.2.4 播放轮廓视频

使用 python 脚本，将视频转换为 polygon\_p 轮廓，搭配动画控制，就可以播放单色视频。

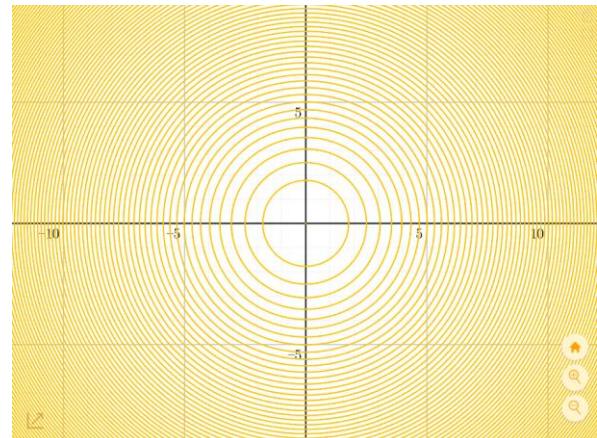
## 12 附录

### 12.1 示例表达式

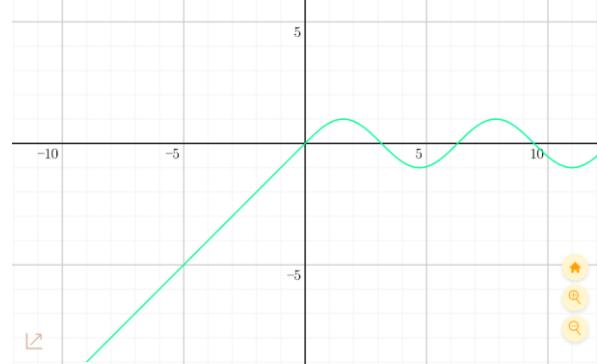
您可以复制以下表达式，使用 SGC 虚拟键盘粘贴到输入区。

二维：

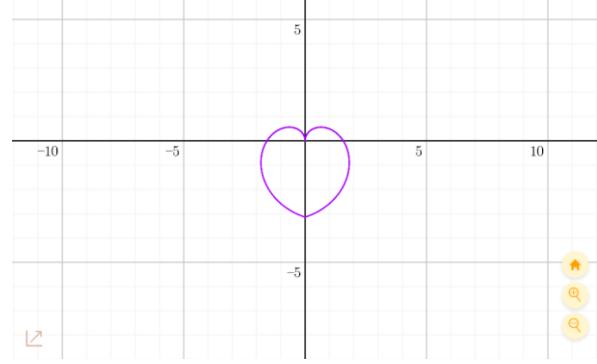
$$\sin(xx+yy)=0$$



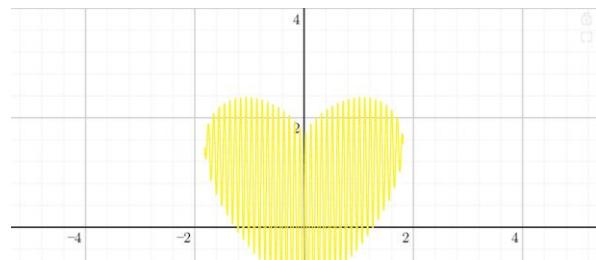
$$(x>0)(\sin(x))+x(x<0)$$



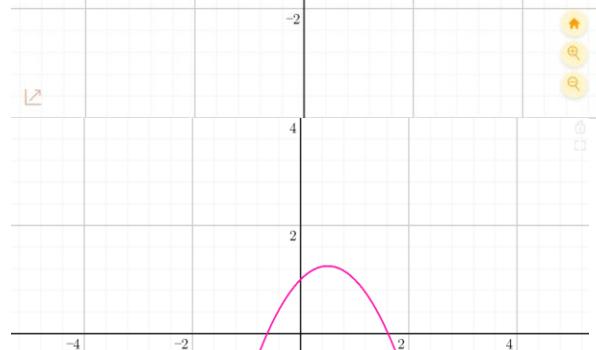
$$r=\cos(\sin(\theta))$$



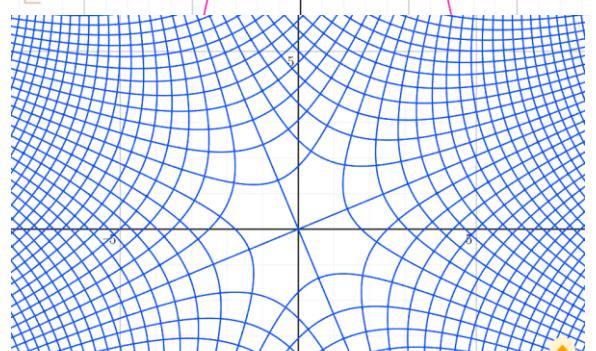
$a=21.2;$   
 $x^{(2/3)}+0.9\sqrt{3.3-xx}\sin(3ax)$



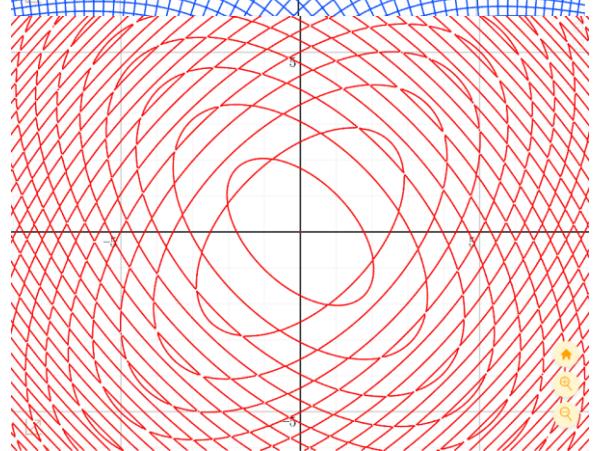
$a=-1;$   
 $b=1;$   
 $c=1;$   
 $y=ax^2+bx+c$



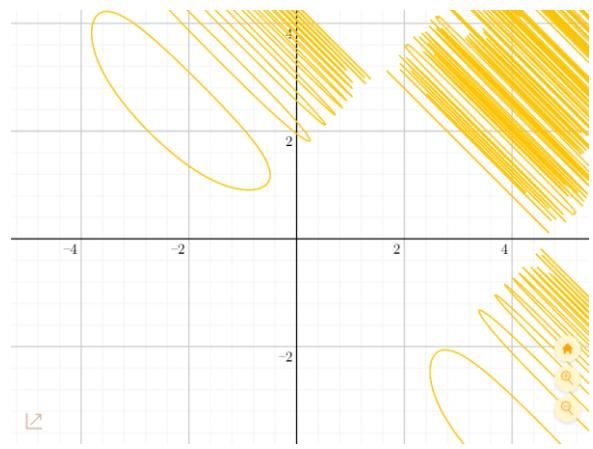
$\sin(xx-yy)=\sin(2xy)$



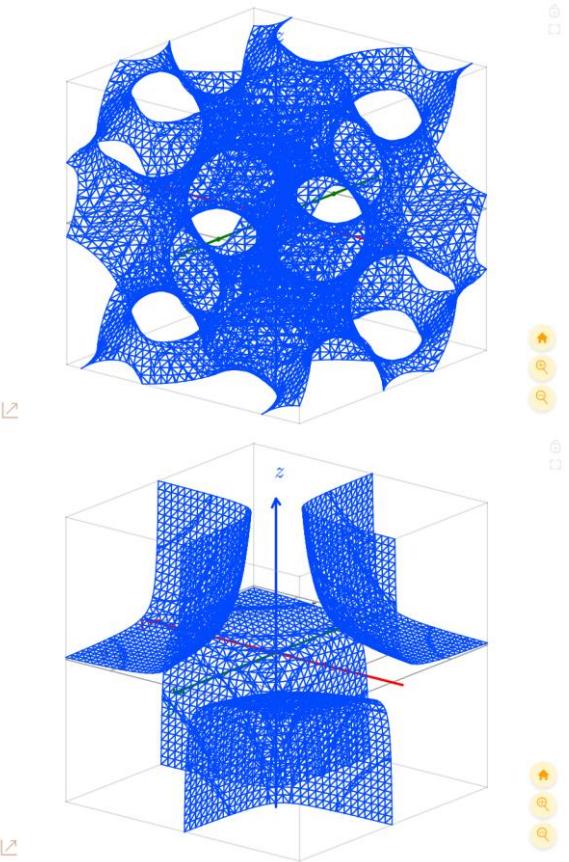
$\sin(xx+yy)=\sin(xy)$



$$\exp(\sin x + \cos y) = \sin(\exp(x+y))$$

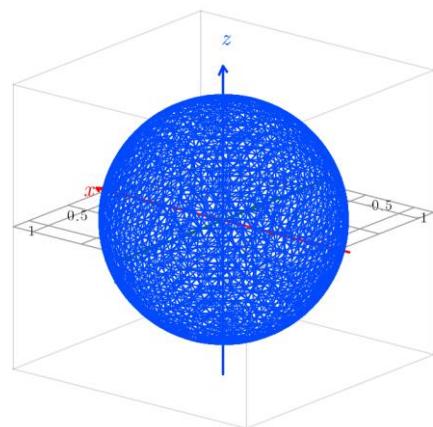


**3D:**  
$$\exp(\sin x + \cos y) = \sin(\exp(x+y))$$

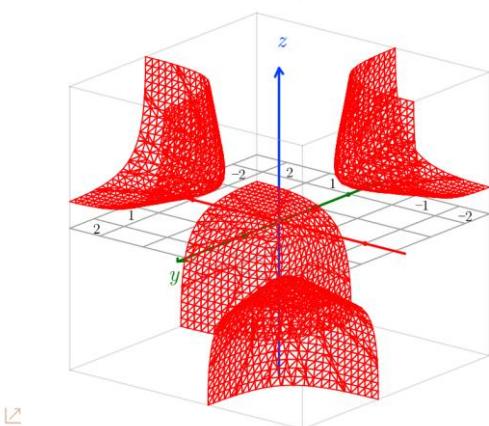


$$xyz=1$$

$$xx+yy+zz=1$$



$$xx+yy+zz-4xyz=0$$



## 13 关于内部实现的一些说明

SGC 是数值计算软件。在此，对一些数值计算方法做一定解释。

### 13.1 数值一阶导数的实现

这里假设以下的函数均具有二阶导数。

在高中，我们学过有关导数的知识。根据导数定义，可导的函数  $f(x)$  在  $x=x_0$  处导数的值，是  $f(x)$  在  $x=x_0$  处切线的斜率。其计算公式是：

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x)-f(x)}{\Delta x}$$

我们称这样的数值计算方法为向前差分。同样地，也有向后差分：

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x)-f(x-\Delta x)}{\Delta x}$$

这两种方法都可以用于计算数值导数。这里介绍一种相对来说精度更高的方法：中心差分。

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x)-f(x-\Delta x)}{2\Delta x}$$

结合导数定义与图像，这公式很好理解。在此我们不做证明。还有很多更高级、精度也更高的公式，由于表述复杂，在此不做解释。

它们的精度如下：

向前差分	$O(h^2)$
向后差分	$O(h^2)$
中心差分	$O(h^2) +$

对于任意的可导的函数  $f(x)$ ，我们可以使用导数加和、复合函数求导法则等来求导。但是，这对于计算机来说并不好实现（有一种叫做“自动微分”的算法可以实现符号求导，其基本原理是将一般表达式的一个算子  $u$  变为了  $\langle u, u' \rangle$  来存储，这样使用导数加和、复合函数求导法则等方法时，可以直接获取  $u'$ ）。因此，我们可以使用数值求导。使用中心差分就能快速获取导数值，而选取  $\Delta x$  的值尤为重要。

并不是越小的  $\Delta x$  就越好。计算机中，一般使用单精度或者双精度浮点数表示一个小数。这里，我们介绍双精度浮点数相关的算法。双精度浮点数能处理的最小的数是  $1e-320$  左右。但这并不意味着  $\Delta x$  能取  $1e-320$ 。

想象计算机使用双精度浮点数计算  $2^{100} + 1 - 2^{100}$ 。我们很容易得出答案是 1。计算机先计算  $2^{100}$ ，得到  $1.2676506002282294e+30$ ，容易发现， $2^{100}$  的个位、十位等，因为科学计数法而丢失了。这会导致  $2^{100} + 1 = (1.2676...82294e+30) + 1$ ，还是等于  $1.2676...82294e+30$ 。再做减法，导致结果为 0，这不是我们期待的。同理，如果  $\Delta x$  取得过小，会导致  $f(x+\Delta x)$ 、 $f(x-\Delta x)$  在双精度浮点数中的数值近乎相等，甚至完全一样，数值求导方法就失效了。考虑到  $O(h^2)$  的精度，我们可以选取  $1e-6$  至  $1e-5$  的  $\Delta x$ 。

例如：想要计算  $\sin x$  在  $x=1$  处的导数。我们容易求得，这数应该为  $\cos 1$ ，约为 0.5403023058681398。

编写 js 代码如下：

```
dx=5e-4;
console.log([Math.sin(1+dx),Math.sin(1-dx),
(Math.sin(1+dx)-Math.sin(1-dx))/dx/2]);
```

我们选取一些  $\Delta x$ ，用中心差分公式计算，得到如下表：

$f(x+\Delta x)$	$f(x-\Delta x)$	计算结果	$\Delta x$
0.8417410307657034	0.8412007284823478	0.5403022833555537	5e-4=0.0005
0.8415250108310384	0.8414169503700448	0.5403023049677103	1e-4=0.0001
0.8414979988713400	0.8414439686407756	0.5403023056438361	5e-5=0.00005
0.8414763877888816	0.8414655817427644	0.5403023058569989	1e-5=0.00001
0.8414736863089075	0.8414682832858488	0.5403023058736522	5e-6=0.000005
0.8414723355610315	0.8414696340495023	0.5403023058514478	2.5e-6=0.0000025
0.8414715251097816	0.8414704445051698	0.5403023058958567	1e-6=0.000001
0.8414712549589443	0.8414707146566384	0.5403023058958567	5e-7=0.0000005
0.8414710388381229	0.8414709307776618	0.5403023056738121	1e-7=0.0000005

其中蓝色为正确部分。不难发现，随着  $\Delta x$  的减小，精度在逐渐提高； $\Delta x$  小于  $1e-6$  后，精度反而出现了倒退的现象。

综上，我们可以选取  $2.5e-6$  左右的数作为  $\Delta x$ 。

顺便提一下计算二阶数值导数的方法。注意到  $f''(x)$  是给  $f(x)$  求导后再求导，因此我们可以计算

$$f''(x) = \lim_{\Delta x \rightarrow 0} \frac{f'(x+\Delta x) - f'(x-\Delta x)}{2\Delta x}$$

但这可能会导致非常大的误差，因为一阶导数本身具有误差，这样反而把误差放大了。我们可以考虑函数  $f(x)$  的泰勒展开：

$$\begin{aligned} f(x+\Delta x) &= f(x) + f'(x)(\Delta x) + \frac{f''(x)(\Delta x)^2}{2} \\ f(x-\Delta x) &= f(x) - f'(x)(\Delta x) + \frac{f''(x)(\Delta x)^2}{2} \end{aligned}$$

将上面两个式子相加，消去  $f'(x)$ ，便得到

$$f(x+\Delta x) + f(x-\Delta x) = 2f(x) + f''(x)(\Delta x)^2$$

移项并化简，得到

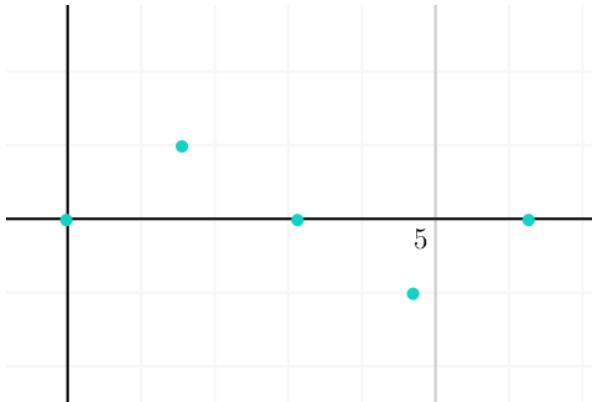
$$f''(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{\Delta x^2}$$

这样的二阶导数不依赖于  $f'(x)$  的计算，具有较高的精度和简单的计算过程。

## 13.2 作函数的图像

在初中，我们学过有关函数的知识。作函数的图像一般使用描点法。我们想要画出函数  $f(x)$  的图像，可以计算  $f(-2)$ 、 $f(-1.5)$ 、 $f(-1)$ ，等等。将这些函数的值和对应的  $x$  放在平面直角坐标系中，用平滑的曲线连接这些点，便得到这函数大致的图像。

在计算机中，一般地，也可以使用描点法作图。鉴于计算机计算的速度比手工计算快得多，我们可以取更多的  $x$  来计算，这样画出的图像也会更精确。例如，想要画出  $y=\sin x$  在区间  $[0, 2\pi]$  的图像，我们可以取  $(0,0)$ 、 $(\frac{\pi}{2}, 1)$ 、 $(\pi, 0)$ 、 $(\frac{3\pi}{2}, -1)$ 、 $(2\pi, 0)$  这五个点，基于我们学过的  $\sin x$  的性质，大致就能描绘  $y=\sin x$  的图像。



但是，我们只使用这五个点作图总会有误差。因此，我们可以编写程序，取更多的  $x$  值。例如这里选取  $[0, 2\pi]$  中 101 个  $x$ ，分别是  $0, \pi/50, \pi/25, \dots, 2\pi$ 。使用计算机算得如下结果（这里只展示一部分）：

$x$	0.8168	0.8482	0.8796	0.9111	0.9425	0.9738
$\sin x$	0.7289	0.7501	0.7705	0.7902	0.8090	0.8271

这样，我们便可以得到更精细的曲线。

以上便是描点法的基本思想。

对于一些变化率比较大的函数，这个方法就不能很好地描绘函数图像。例如  $y=\sin(\exp x)$ ，随着  $x$  的增大，这函数震荡的愈发剧烈。一般地，可以通过增加描点的数量（下称提高精度(precision)）来改善，但是效果仍然不理想。考虑到函数的导数能反映这函数的变化率，我们便可以根据导数来调整步长，这称为自适应绘图。

自适应绘图大致的思路如下：我们默认步长为  $j$ 。在计算  $f(x_i)$  ( $i > 2$ ) 时，总能根据导数公式，计算  $f'(x_i)$  的值。之后，我们便记  $|f'(x_i)| = k$ ，利用  $k$  的值来调整步长。比如可以让下一次计算的步长为  $1/k$ 。这样，对于变化率比较大的函数，步长能自动调整，便能较好地描绘图像。

特别地，我们还需要限制  $1/k$  的最大值与最小值，防止精度过低或者过高，导致计算过多数据。

这方法可以进一步优化。每当计算  $f'(x_i)$  时，总需要至少多计算一次，带来额外的

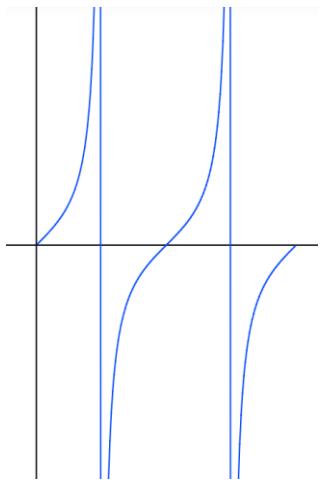
开销。因此，我们可以利用  $f(x_i)$  与  $f(x_{i-1})$  来近似计算  $f'(x_i)$ 。这样能提升计算速度。

SGC v5 便支持使用此方法绘图。但是，这个方法依然会带来一些问题。例如  $y=e^x$  不需要使用自适应绘图，因为其单调递增，但是其导数仍为  $e^x$ ，随着  $x$  增大， $e^x$  也增大，导致步长变小，计算了无意义的数据。

### 13.3 数值断点判断

下面介绍间断点(BreakPoints)。

上述的基本描点法看似没有问题，但计算机并不能很容易地获取函数的性质。例如： $y=\tan x$  在区间  $[0, 2\pi]$  上具有无穷间断点。我们可以很容易知道在  $x=\frac{\pi}{2}$  和  $x=\frac{3\pi}{2}$  处有间断点，可以在此处不连接。但是使用基本的描点法，我们得到如下“错误”图像：



计算机并不知道在  $x=\frac{\pi}{2}$  和  $x=\frac{3\pi}{2}$  处有间断点，因此在此处连接。这种情况应该如何处理呢？

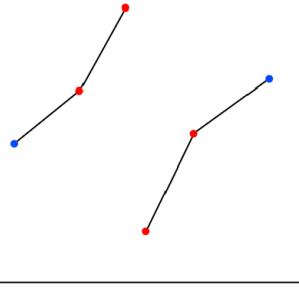
根据间断点的部分定义，我们知道：如果函数  $f(x)$  在  $x=x_0$  处具有不相等的左极限和右极限，这  $x_0$  便是函数  $f(x)$  的一个间断点。因此，我们可以在每一次计算第  $i$  个  $x$  时，考虑  $f(x_{i-1})$  和  $f(x_{i+1})$ 。如果  $f(x_i)$  未定义，那么  $x_0$  就是间断点。如果这两个数相差超过了  $\epsilon$ ，那么便认为这是一个间断点。 $\epsilon$  可以取  $1e-3$  左右，根据步长  $i$  而定。

第二种方法是计算  $f(x_i+\Delta x)$  与  $f(x_i-\Delta x)$  来替代  $f(x_{i-1})$  和  $f(x_{i+1})$ 。

第三种方法是取  $|f(x_{i-1})-f(x_i)|$  与  $\epsilon$  比较。SGC v1、v2 采用此方法。

上述三种方法并不具有普遍性，并且很容易误判。例如函数  $f(x)=114514x$ ，就算取了较小的步长， $f(x_{i-1})$  和  $f(x_{i+1})$  的差始终都会很大，导致所有点全部断开，这不是我们所期望的。这里介绍一种新的方法：四点法，也是 SGC v4 所采用的方法。

(1) 考虑无穷间断点

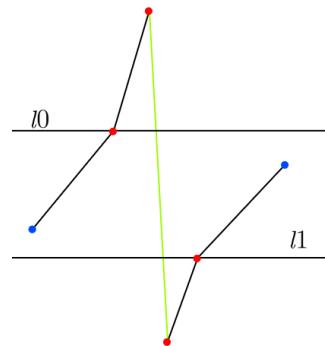


如图所示，如  $y=\tan x$  在  $x=\frac{\pi}{2}$  附近的行为。

取标红色的四个点，从左到右按顺序记为  $v_1$ 、 $v_2$ 、 $v_3$ 、 $v_4$ 。

如果  $v_1$ 、 $v_2$ 、 $v_3$ 、 $v_4$  中有未定义值，认为是间断点。

如图， $v_1-v_2$  与  $v_3-v_4$  的符号相同， $v_2-v_3$  与  $v_1-v_2$ 、 $v_3-v_4$  中任意一个符号相反，即  
 $(v_2-v_3)(v_1-v_2) \leq 0 \parallel (v_2-v_3)(v_3-v_4) \leq 0$

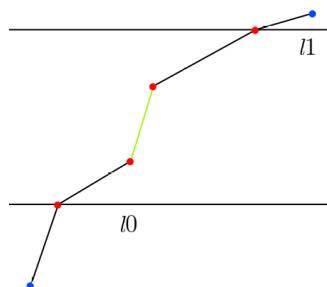


过  $v_1$ 、 $v_4$  做  $x$  轴平行线  $l_0$ 、 $l_1$ 。如果  $l_0$  在  $l_1$  的上方，认为  $v_2-v_3$  中有间断点。表达式为  $|v_1-v_2|+|v_3-v_4| < |v_2-v_3|$ 。保险起见，我们加上误差  $\epsilon$ ，即需要满足  $|v_1-v_2|+|v_3-v_4|+\epsilon < |v_2-v_3|$ 。这个式子无论是什么样的断点，都必须要满足。其中  $\epsilon$  视步长而定。SGC 中，取的是  $\epsilon=0.01/\text{当前缩放}$ 。

## (2) 考虑跳跃间断点

例如定义分段函数  $f(x)=\begin{cases} \sin x, & x<1 \\ x, & x \geq 1 \end{cases}$

显然，此函数在  $x=1$  处产生跳跃间断点，并且使用(1)中做差却产生了两侧同符号。



这里我们使用第二做差法： $|(v_1-v_2)(v_3-v_4)|<\epsilon|v_2-v_3|$ 。 $\epsilon$  视步长而定。SGC 中，取的是  $\epsilon=0.015$ /当前缩放。

### (3) 考虑震荡间断点

如  $y=\sin\frac{1}{x}$  在  $x=0$  附近的行为。一般情况下，震荡间断点不易被绘制出来，在此我们不做讨论。

使用(1)解决了  $y=\tan x$ 、 $y=\frac{1}{x}$  和  $y=[x]$  这类函数的间断点问题。使用(2)解决了分段函数的间断点问题。



极坐标、参数方程的间断点判断方法同理可得。

## 13.4 数值积分

数值积分已经有很多方法计算，例如矩形法、梯形法、抛物线法(Simpson 法)等。但是这些方法对于较复杂的被积函数，收敛速度就显得慢了。

这里介绍两种收敛较快、精度较高的方法。

### (1) 龙贝格法(Romberg 法)

将变步长（区间数加倍、步长减半）梯形法的积分近似值通过线性组合得到变步长 Simpson 法的积分近似值，再将变步长 Simpson 法的积分近似值通过线性组合得到变步长 Cotes 法的积分近似值，再将变步长 Cotes 法的积分近似值通过线性组合得到变步长 Romberg 法的积分近似值。这种逐次二分积分区间并利用线性组合的加速方法将一个粗糙的积分值  $T_n$  序列逐步加工成精度较高的积分值  $R_n$  序列的方法，称为 Romberg 积分法。

由于龙贝格法的步骤较多，且 Scratch 难以实现，在此不做过多介绍。

### (2) 高斯积分法与 Gauss-Kronrod 法

梯形法和抛物线法的表达式，会发现这两个方法的结果可以理解为一系列点的函数值，乘上某个系数后求和。这两种方法都是均匀分割区间来进行求和，至少具

有  $n-1$  的代数精度。而高斯积分法并不是均匀分割区间，而是计算不同高斯节点，乘以每个点的权重后再求和。它具有  $2n+1$  的代数精度，可精确处理  $2n+1$  次以下多项式积分。

高斯积分法处理  $[-1,1]$  上的  $f(x)$ 。我们可以用区间变换实现处理  $[a,b]$ 。这里不介绍高斯积分法具体的数学原理，只解释高斯积分法如何进行数值计算。高斯积分法可理解为以下公式：

$$\int_{-1}^1 f(x)dx = \sum_{j=0}^n w_j f(x_j)$$

$w_j$  就是权重。

高斯积分法节点使用勒让德多项式计算。勒让德(Legendre)多项式满足以下性质：当某个多项式  $P(x)$  的次数小于  $n$  时，有

$$\int_{-1}^1 P(x)P_n(x)dx = 0$$

并且满足递推关系：

$$P_{n+1}(x) = \frac{2n+1}{n+1} x P_n(x) - \frac{n}{n+1} P_{n-1}(x) \quad (n=1,2,\dots)$$

这样，我们便求得

$$\begin{aligned} P_1(x) &= x \\ P_2(x) &= \frac{3x^2 - 1}{2} \\ P_3(x) &= \frac{5x^3 - 3x}{2} \end{aligned}$$

以此类推。 $n$  阶高斯积分法的节点便是  $P_n(x)$  的根。注意到， $P_n(x)$  的根全部处于  $[-1,1]$  上，它具有  $n$  个实根。 $n$  阶高斯积分法的系数有如下公式：

$$w_i = \frac{2}{(1-x_i^2)(P'_n(x_i))^2}$$

例如：要计算

$$\int_{-1}^1 x^2 dx$$

这里我们选用 2 阶高斯积分公式。先求

$$P_2(x) = \frac{3x^2 - 1}{2}$$

的根，注意到是

$$\left\{-\frac{\sqrt{3}}{3}, \frac{\sqrt{3}}{3}\right\}$$

也就是  $\{-0.5773502691896257, 0.5773502691896257\}$ 。

再求出权重  $w=\{1,1\}$ 。代入得

$$\int_{-1}^1 x^2 dx \approx \left(-\frac{\sqrt{3}}{3}\right)^2 + \left(\frac{\sqrt{3}}{3}\right)^2 = \frac{2}{3}$$

这方法只计算了 2 次得到了准确结果。再例如使用 5 阶高斯积分公式计算

$$\int_{-1}^1 \cos x dx$$

其结果应该为  $2\sin 1=1.682941969615793$ 。这里我们计算节点为

$$\left\{0, \pm \frac{1}{3} \sqrt{5-2\sqrt{\frac{10}{7}}}, \pm \frac{1}{3} \sqrt{5+2\sqrt{\frac{10}{7}}}\right\}$$

其数值为  $\{0, \pm 0.538469310105683, \pm 0.906179845938664\}$ 。

计算权重为

$$\left\{\frac{128}{225}, \frac{322+13\sqrt{70}}{900}, \frac{322+13\sqrt{70}}{900}, \frac{322-13\sqrt{70}}{900}, \frac{322-13\sqrt{70}}{900}\right\}$$

其数值为  $\{0.5688888888888889, 0.47862867049936647,$

$0.47862867049936647, 0.23692688505618908, 0.23692688505618908\}$ 。

我们计算

$$\begin{aligned} & 0.5688888888888889 * \cos 0 \\ & + 2 * 0.47862867049936647 * \cos 0.538469310105683 \\ & + 2 * 0.23692688505618908 * \cos 0.906179845938664 \\ & = \textcolor{blue}{1.682941970407192}。 \end{aligned}$$

想要对函数  $f(x)$  在区间  $[a,b]$  积分，只需要做变换：

$$\int_a^b f(x) dx = \int_{-1}^1 \frac{b-a}{2} f\left(\frac{b-a}{2}t + \frac{a+b}{2}\right) dt$$

以上便是高斯积分法的基本思路。在实际应用时，我们可以提前计算好节点和权重，之后直接调用。SGC v3 就采用 300 节点的高斯积分进行概率计算。

不难发现，仅仅使用固定阶高斯积分法仍然存在问题。例如计算

$$\int_{-1}^1 |\sqrt{x}| dx$$

便遇到了收敛缓慢的问题。即便取 500 节点，精度也不是很高。这里，我们可以借鉴龙贝格法的思路：

(1) 计算某区间的值  $G_n$

在区间  $u$  计算  $n$  阶高斯积分法  $G_n$  的结果后，我们可以取更高的阶  $G_{n+1}$  进行第二次计算，然后比较  $|G_n - G_{n+1}|$  是否大于  $\epsilon$ 。

## (2) 误差判断

$|G_n - G_{n+1}|$ 如果大于  $\epsilon$ , 这说明这个区间是  $n$  阶不能处理的。这时, 便可以二分区间, 提高精度。在二分的两个区间中, 分别使用(1)中的方法再次进行判断。最后将各区间结果累加即可。

二分法一般采用递归的思路。重要的一点是, 我们还需要设置递归上限, 防止死循环。

注意到, 计算  $G_{n+1}$  又需要计算额外的节点, 造成计算量巨大。我们需要一种方法, 重复利用前面  $G_n$  的部分结果来减小计算量。这里介绍 Gauss-Kronrod 方法。Gauss-Kronrod 方法基于 Kronrod 点的误差估计, 可充分利用现有计算值, 从而达到有效控制精度的同时, 性能没有太大的损失。公式可理解为

$$\int_{-1}^1 f(x)dx = \sum_{j=0}^n w_j f(x_j) + \sum_{j=0}^{n+1} v_j f(u_j)$$

$u_j$  是 Stieltjes-Wigert 多项式的根。具体计算方法这里不给出。

数值积分重要的一点是大致判断积分的敛散性。这一点较为简单。在递归过程中, 如果连续多次出现  $|G_n - G_{n+1}| > \epsilon_1$ , 而且这个  $\epsilon_1$  较大, 那么认为此积分发散。

## 13.5 一元方程的数值解与 fSolve 函数

数值解有多种方法计算, 如高中所学的二分法等。但是二分法收敛可能较慢, 这里我们介绍几种方法:

### (1) 不动点迭代

不动点迭代便于理解, 适用于求方程  $x=f(x)$ , 但选择迭代方式非常重要。不恰当的迭代公式甚至会导致结果发散。例如要求  $x^2-x-1$  的根, 有几种不动点迭代方法。我们选用  $x=\sqrt{x+1}$  并选取  $x_0=1$  迭代。得到

$x_1$	1.4142135
$x_2$	1.5537739
$x_3$	1.5980531
$x_4$	1.6118477
$x_5$	1.6161212
$x_6$	1.6174427

但不动点迭代具有局限性。不动点迭代收敛较慢, 并且如求解  $\sin x = \cos x$  就不能使用这方法。

### (2) 牛顿迭代法(Newton-Raphson 迭代法)与割线法

要求解  $f(x)=0$ , 选取初始值  $x_0$ , 可以考虑  $f(x)$  在  $x=x_0$  泰勒展开的线性部分:

$$f(x) = f(x_0) + f'(x_0)(x - \Delta x)$$

因此令

$$0 = f(x_0) + f'(x_0)(x - x_0)$$

作为方程根的近似。解得每一个新的  $x$  是

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

用这公式解  $x^2 - x - 1 = 0$ , 令  $f = x^2 - x - 1$ , 并选取  $x_0 = 1$  迭代, 直到  $|x_n - x_{n+1}| < \epsilon$ 。那么

$$x_{n+1} = x_n - \frac{x_n^2 - x_n - 1}{2x_n - 1}$$

可得

$x_1$	2
$x_2$	1.6666666
$x_3$	1.6190476
$x_4$	1.6180344
$x_5$	1.6180339
$x_6$	1.6180339

可见这方法收敛速度快且易于实现。导数可以使用前面介绍过的方法进行计算。  
SGC v4 的 fSolve 函数采用此方法。

但是, 牛顿法的震荡问题仍没有很好的解决方法。

割线法是牛顿法的改进, 是利用  $f(x_{n-1})$  的值和  $f(x_n)$  进行近似的导数计算。

$$x_{n+1} = x_n - f(x_n) \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

### (3) 复合法

复合法并不需要计算  $f'(x_n)$  的值。公式如下:

$$x_{n+1} = x_n - \frac{f(x_n)^2}{f(f(x_n) + x_n) - f(x_n)}$$

用这公式解  $x^2 - x - 1 = 0$ , 并选取  $x_0 = 2$ , 可得

$x_1$	1.75
$x_2$	1.6388888
$x_3$	1.6186432
$x_4$	1.6180345
$x_5$	1.6180339
$x_6$	1.6180339

这方法并不是特别好处理, 比较依赖于初始值的选择。注意到若选取  $x_0 = 1$  进行迭代, 产生了未定义结果。