Three-Dimensional Coordinate System

6 三维坐标系

平面直角坐标系上升起一根竖轴

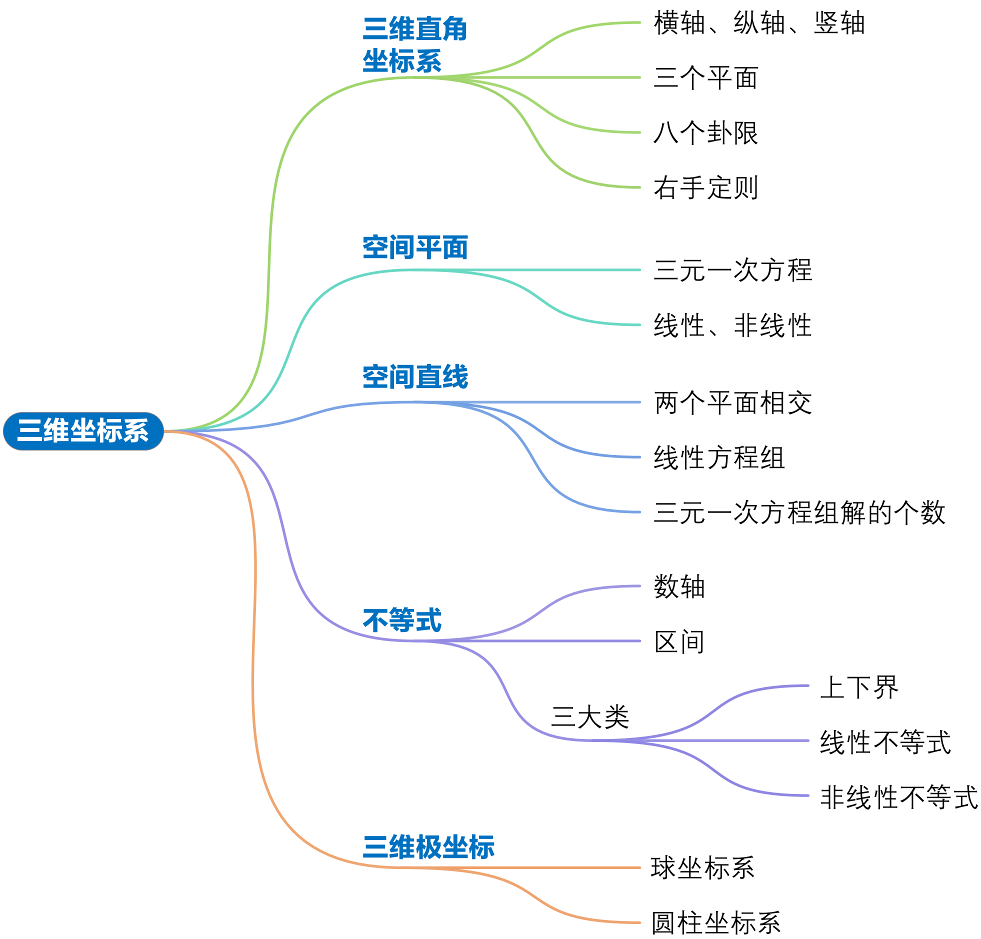
虚空无尽的蔚蓝，神秘深邃的苍穹，漫天飘舞的鸟虫，

时时刻刻在召唤，“腾空而起吧，人类！”

The blue distance, the mysterious Heavens, the example of birds and insects flying everywhere — are always beckoning Humanity to rise into the air.

—— 康斯坦丁·齐奥尔科夫斯基 (Konstantin Tsiolkovsky) | 俄罗斯火箭专家 | 1857 ~ 1935

- ax.plot_wireframe() 绘制线框图
- matplotlib.pyplot.contour()绘制平面等高线
- matplotlib.pyplot.contourf ()绘制平面填充等高线
- numpy.meshgrid() 产生网格化数据
- numpy.outer() 计算外积
- plot_parametric() 绘制二维参数方程
- plot3d_parametric_line() 绘制三维参数方程


三维坐标系
三维直角坐标系
横轴、纵轴、竖轴
三个平面
八个卦限
右手定则
空间平面
三元一次方程
线性、非线性
空间直线
两个平面相交
线性方程组
三元一次方程组解的个数
不等式
数轴
区间
三大类
上下界
线性不等式
非线性不等式
三维极坐标
球坐标系
圆柱坐标系

6.1 三维直角坐标系

费马 (Pierre de Fermat) 不但独立发明平面直角坐标系；他还在 xy 平面坐标系基础上插上 z 轴，发明了三维直角坐标系。

三维直角坐标系有三个坐标轴——x 轴或横轴 (x-axis)，y 轴或纵轴 (y-axis) 和 z 轴或竖轴 (z-axis)。本系列丛书也经常使用 x_1、x_2、x_3 来代表横轴、纵轴和竖轴。

图 1 所示三维直角坐标系有三个平面：xy 平面、yz 平面、xz 平面。x 轴和 y 轴构成 xy 平面，z 轴垂直于 xy 平面；y 轴和 z 轴构成 yz 平面，x 轴垂直于 yz 平面；x 轴和 z 轴构成 xz 平面，y 轴垂直于 xz 平面。这三个平面将三维空间分成了八个部分，称为**卦限** (octant)。

三维直角坐标系内坐标点可以写成 (a, b, c)。

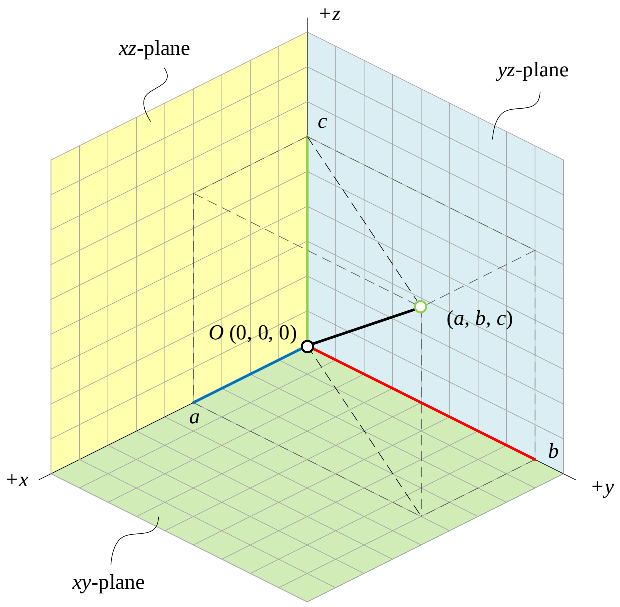


图 1. 三维直角坐标系和三个平面

图 2 所给出三种右手定则，用来确定三维直角坐标系 x、y 和 z 轴正方向。

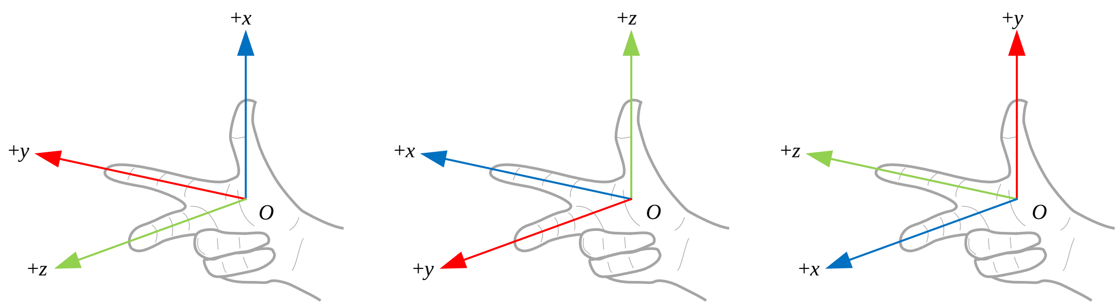


图 2. 右手定则确定三维直角坐标系 x、y 和 z 轴正方向

6.2 空间平面：三元一次方程

三维直角坐标系中，平面可以写成如下等式。

$$ax + by + cz + d = 0 \tag{1}$$

其中，x、y、z 为变量，a、b、c、d 为参数。实际上，这个等式就是代数中的三元一次方程。

举个例子，图 3 所示的平面对应的解析式为。

$$x + y - z = 0 \tag{2}$$

图 3 中网格面的颜色对应为 z 的数值，z 越大，越靠近暖色系；z 越小，越靠近冷色系。

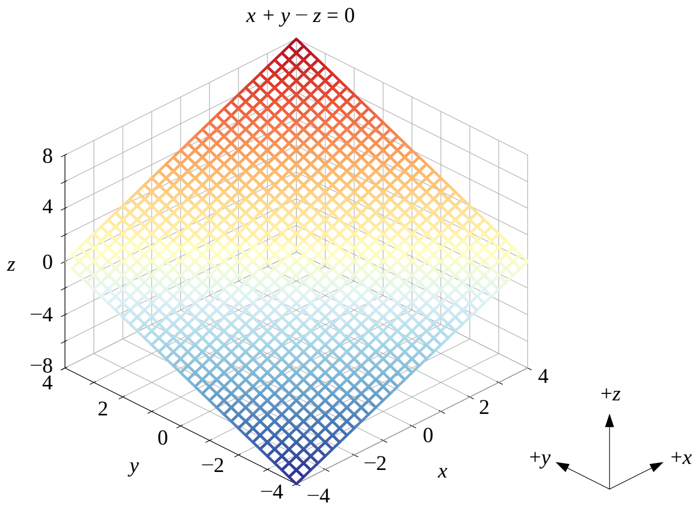


图 3. 等式 $x + y - z = 0$ 对应的平面

以 z 作为因变量、x 和 y 作为自变量的话，(2) 等价于如下二元函数。

$$z = f\left(x, y\right) = x + y \tag{3}$$

图 4 所示平面对应的解析式为。

$$y - z = 0 \tag{4}$$

图 4 中网格面平行于 x 轴；从等式上来看，不管 x 取任何值，y 和 z 都满足 $y - z = 0$。

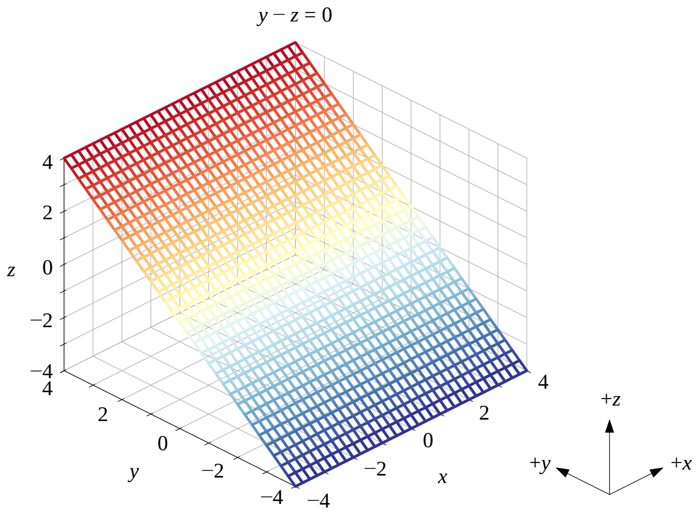


图 4. 等式 $y-z=0$ 对应的平面

图 5 所示的平面对应的解析式为。

$$x-z=0 \tag{5}$$

图 5 中网格面平行于 y 轴；不管 y 取任何值，y 和 z 的关系都满足 $x-z=0$。

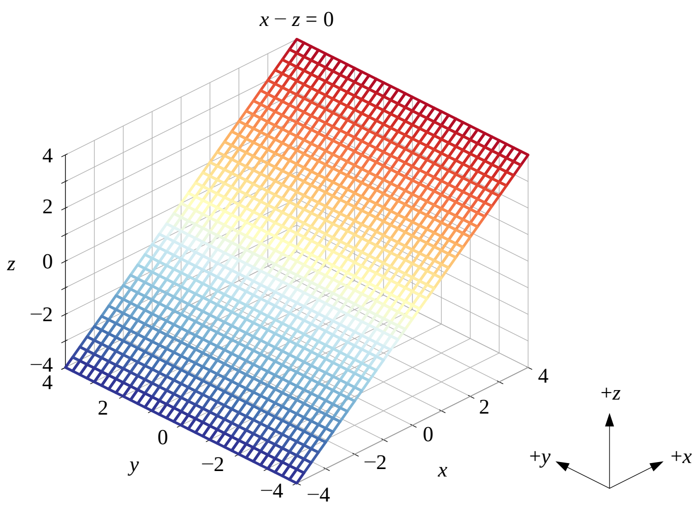


图 5. 等式 $x-z=0$ 对应的平面

图 6 所示平面对应的等式为 $z-2=0$，这个平面显然平行于 xy 平面。

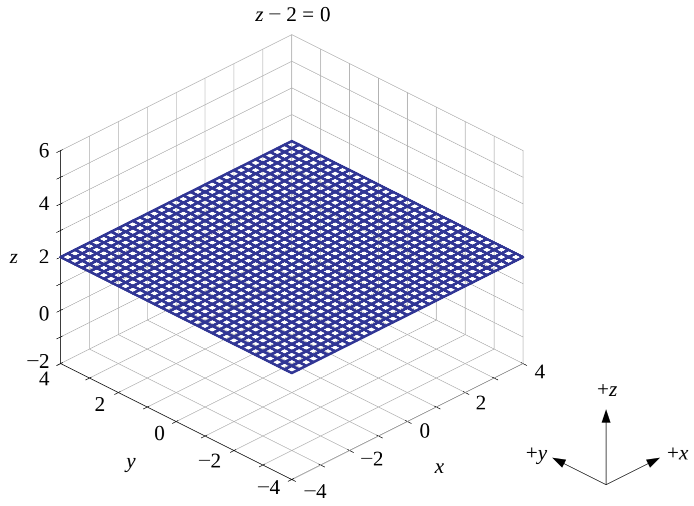


图 6. 等式 $z-2=0$ 对应的平面

图 7 ~ 图 9 三幅图中平面有一个共同特点，它们都垂直于 xy 平面。这三个平面，z 的取值都不影响平面和 xy 平面的相对位置。三个平面都相当于，xy 平面上一条直线沿 z 方向展开。反过来，图 7 ~ 图 9 三幅图中平面在 xy 平面上的投影为一条直线。

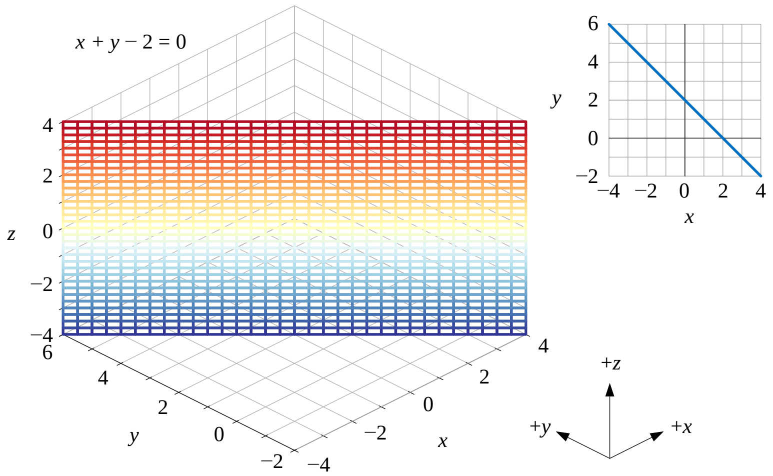


图 7. 等式 $x+y-2=0$ 对应的平面

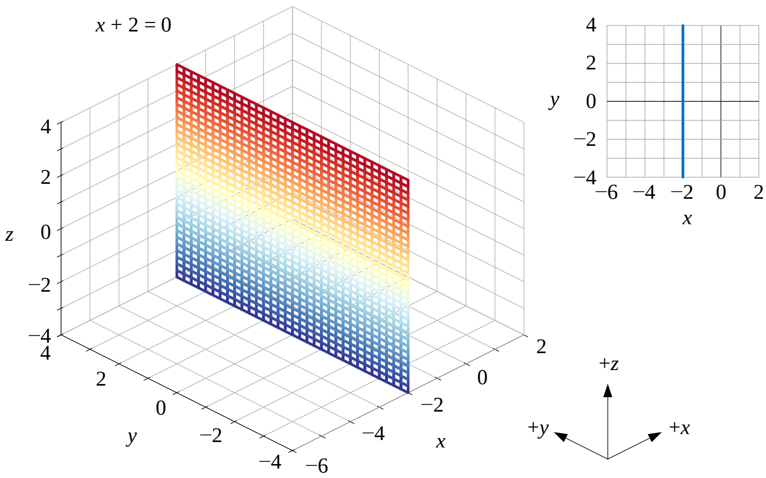


图 8. 等式 $x + 2 = 0$ 对应的平面

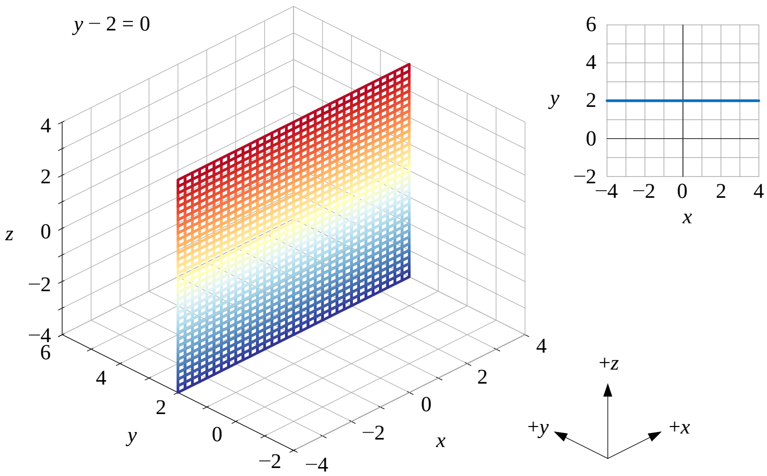


图 9. 等式 $y - 2 = 0$ 对应的平面

相信大家经常听到"线性"和"非线性"这两个词，这里我们简单区分两者。

在平面直角坐标系中，线性 (linearity) 是指量与量之间的关系用一条直线表示，比如 $y = ax + b$；平面上，线性函数即一次函数，对应图像为一条斜线。

在三维直角坐标系中，线性的几何形式是平面，也就是二元一次函数。

对于多元函数，线性的形式为 $y = b_1x_1 + b_2x_2 + b_3x_3 + ... + b_nx_n + b_0$；在多维空间中，其对应图像是超平面。

图 10 给出线性关系三个例子。

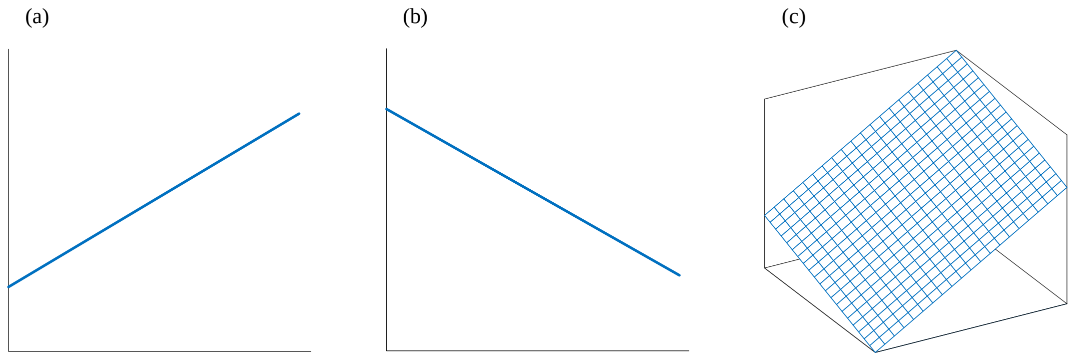


图 10. 线性关系

与线性相对的是非线性 (nonlinearity)，即图像不是直线、也不是平面、更不是超平面。变量之间的关系可以是曲线、折线，甚至不能用参数来描述。图 11 给出平面上非线性关系例子。

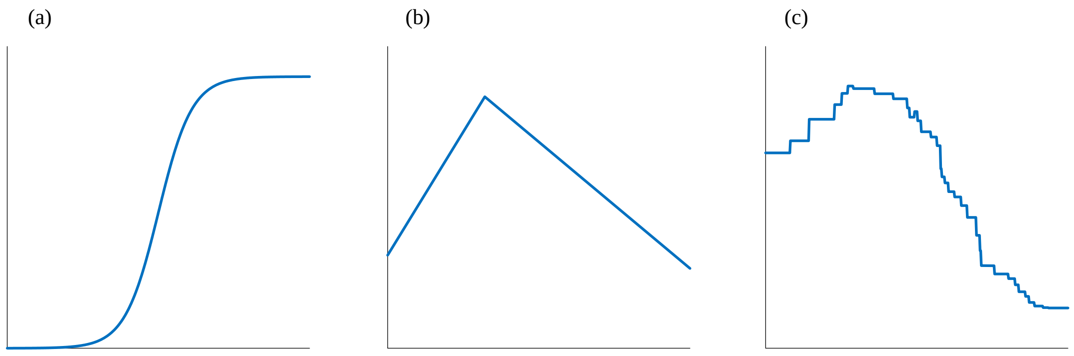


图 11. 非线性关系

机器学习中，回归模型是重要监督学习 (supervised learning)；回归模型研究变量和自变量之间关系，目的是分析预测。图 12 给出三类回归模型，图 12 (a) 是线性回归模型，图 12 (a) (b) 则是非线性回归模型。

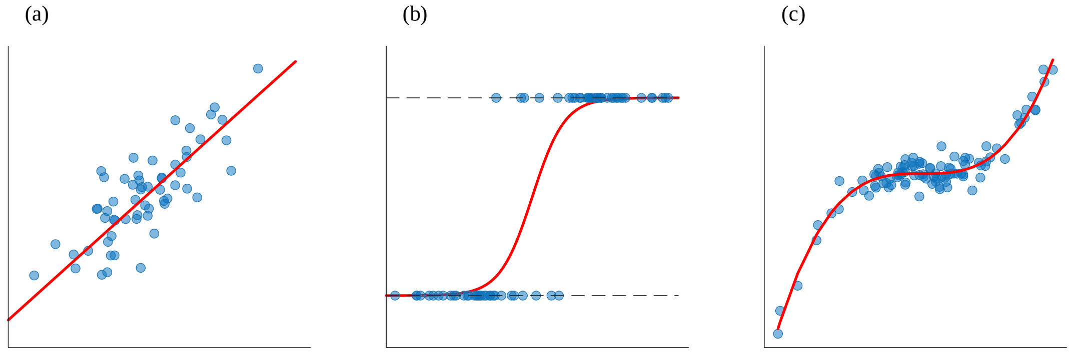


图 12. 机器学习中回归问题

监督学习中，二分类问题很常见；二分类输出标签一般为 0、1，比如图 13 中蓝色和红色数据点。图 13 (a) 所示为用线性 (一根直线) 决策边界 (decision boundary) 分类蓝色、红色数据点，图 13 (b) (c) 所示为非线性决策边界。

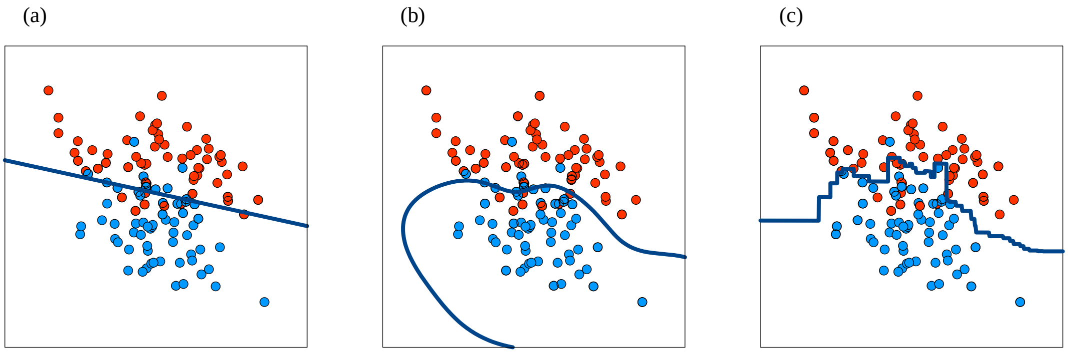


图 13. 机器学习中二分类问题

以下代码绘制本节几幅三维空间平面。

```
# Bk_Ch6_01

import math
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm

def plot_surf(xx,yy,zz,caption):

    norm_plt = plt.Normalize(zz.min(), zz.max())
    colors = cm.RdYlBu_r(norm_plt(zz))

    fig = plt.figure()
    ax = fig.gca(projection='3d')
    surf = ax.plot_surface(xx,yy,zz,
                           facecolors=colors, shade=False)
    surf.set_facecolor((0,0,0,0))

    plt.show()
    ax.set_proj_type('ortho')

    if xx.min() == xx.max():
        ax.set_xlim(xx.min() - 4,xx.min() + 4)
    else:
        ax.set_xlim(xx.min(),xx.max())

    if yy.min() == yy.max():
        ax.set_ylim(yy.min() - 4,yy.min() + 4)
    else:
        ax.set_ylim(yy.min(),yy.max())

    if zz.min() == zz.max():
        ax.set_zlim(zz.min() - 4,zz.min() + 4)
    else:
        ax.set_zlim(zz.min(),zz.max())

    plt.tight_layout()
```

```
    ax.set_xlabel('$\it{x}$')
    ax.set_ylabel('$\it{y}$')
    ax.set_zlabel('$\it{z}$')
    ax.set_title(caption)
    ax.view_init(azim=-135, elev=30)
    ax.xaxis._axinfo["grid"].update({"linewidth":0.25, "linestyle" : ":"})
    ax.yaxis._axinfo["grid"].update({"linewidth":0.25, "linestyle" : ":"})
    ax.zaxis._axinfo["grid"].update({"linewidth":0.25, "linestyle" : ":"})

num = 33
x = np.linspace(-4,4,num)
y = np.linspace(-4,4,num)
xx,yy = np.meshgrid(x,y);

plt.close('all')

#%% z - 2 = 0
zz = 2 + xx*0;
caption = '$z - 2 = 0$';
plot_surf (xx,yy,zz,caption)

#%% y - z = 0
zz = yy;
caption = '$z - y = 0$';
plot_surf (xx,yy,zz,caption)

#%% x - z = 0
zz = xx;
caption = '$x - z = 0$';
plot_surf (xx,yy,zz,caption)

#%% x + y - z = 0
zz = xx + yy;
caption = '$x + y - z = 0$';
plot_surf (xx,yy,zz,caption)

#%% vertical mesh plot

x = np.linspace(-4,4,num)
z = np.linspace(-4,4,num)
xx,zz = np.meshgrid(x,z);

#%% y - 2 = 0
yy = 2 - xx*0
caption = '$y - 2 = 0$';
plot_surf (xx,yy,zz,caption)

#%% x + y - 2 = 0
yy = 2 - xx
caption = '$x + y - 2 = 0$';
plot_surf (xx,yy,zz,caption)

#%% x + 2 = 0
y = np.linspace(-4,4,num)
z = np.linspace(-4,4,num)
yy,zz = np.meshgrid(y,z);

xx = -2 - yy*0
caption = '$x + 2 = 0$';
plot_surf (xx,yy,zz,caption)
```

6.3 空间直线：三元一次方程组

有了三维空间平面，确定一条空间直线则变得很简单——两个平面相交便确定一条空间直线；也就是说，一般情况下，两个三元一次方程确定一条三维空间直线。

比如，下例给出两个三元一次方程。

$$\begin{cases} x+y-z=0 \\ 2x-y-z=0 \end{cases} \tag{6}$$

每个方程代表三维空间的一个平面；如图 14 所示，这两个平面相交得到一条直线。

从代数角度，可以这样理解 (6)，这两个三元一次方程构成的方程组有无数组解，这些解都在图 14 所示黑色直线上。

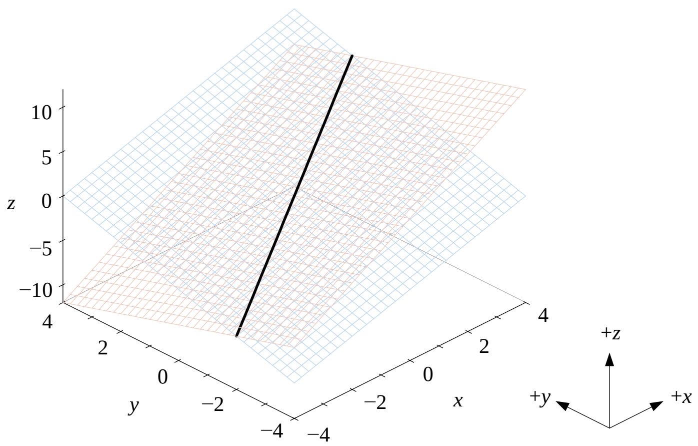


图 14. 两个相交平面确定一条直线

在 (6) 基础上，再加一个三元一次方程，得到如下方程组。

$$\begin{cases} x+y-z=0 \\ 2x-y-z=0 \\ -x+2y-z+2=0 \end{cases} \tag{7}$$

如图 15 所示，这三个平面相交于一点；也就是说，这个三元一次方程组有唯一解。

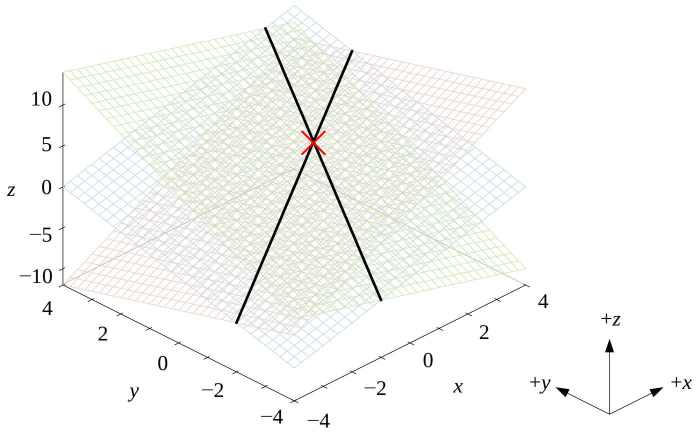


图 15. 三个平面平面相交于一点

(7) 一般写成如下矩阵运算形式。

$$\underbrace{\begin{bmatrix} 1 & 1 & -1 \\ 2 & -1 & -1 \\ -1 & 2 & -1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ -2 \end{bmatrix}}_{b} \tag{8}$$

这种形式叫做线性方程组 (system of linear equations)，(7) 一般写成 $\boldsymbol{Ax} = \boldsymbol{b}$。可以想见，当线性方程组的方程数不断增多，(7) 这种形式更规整，更便于计算；而且，对矩阵 $\boldsymbol{A}$ 的各种性质研究，可以判定线性方程组解的特点。

本书最后还会用“鸡兔同笼”问题再次讨论线性方程组。

三元一次方程组解的个数

图 16 所示为三元一次方程组解的个数几种可能性。

如图 16 (a) 所示，当三个平面相交于一点，方程组有且仅有一个解。

如图 16 (b) 所示，当三个平面相交于一条线，方程组有无数组解。无数组解还有其他情况，比如两个平面重合和第三个平面相交，再比如三个平面重合。

图 16 (c)、(d)、(e) 给出的是方程组无解的三种情况。图 16 (d) 中，三个平面平行；图 16 (e) 中，两个平面重合，与第三个平面平行。方程组还有其他无解的情况，比如三个平面两两相交，得到三条交线，而三条交线相互平行。

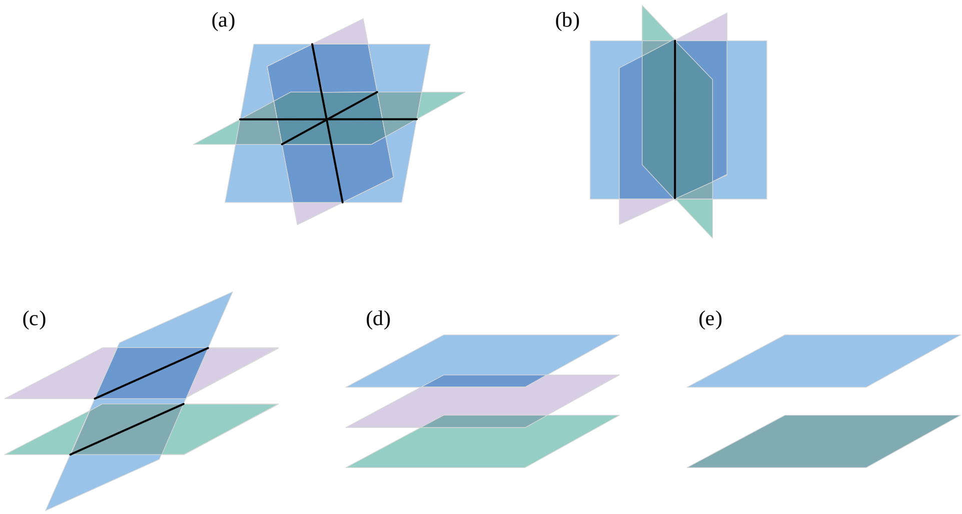


图 16. 三元一次方程组解的个数

以下代码绘制图 14；请大家自行修改代码绘制图 15。

```
# Bk_Ch6_02

import math
import numpy as np
import matplotlib.pyplot as plt

num = 33
x = np.linspace(-4,4,num)
y = np.linspace(-4,4,num)
xx,yy = np.meshgrid(x,y);

plt.close('all')

zz1 = xx + yy;
zz2 = 2*xx - yy;

fig = plt.figure()
ax = fig.gca(projection='3d')

CS = ax.contour(xx,yy, zz1 - zz2, levels = [0],
           colors = '#339933')
ax.cla()

ax.plot_wireframe(xx, yy, zz1, color = '#BDD6EE')
# , rstride=10, cstride=10
ax.plot_wireframe(xx, yy, zz2, color = '#ECCCC0')

# plot the intersection line

for i in range(0,len(CS.allsegs[0])):

    contour_points_x_y = CS.allsegs[0][i]

    contour_points_z = (contour_points_x_y[:,0] +
                        contour_points_x_y[:,1])

    ax.plot3D(contour_points_x_y[:,0],
```

```
                contour_points_x_y[:,1],
                contour_points_z,
                color = 'k',
                linewidth = 4)

ax.set_proj_type('ortho')

ax.set_xlim(xx.min(),xx.max())
ax.set_ylim(yy.min(),yy.max())

plt.tight_layout()
ax.set_xlabel('$\it{x}$')
ax.set_ylabel('$\it{y}$')
ax.set_zlabel('$\it{z}$')

ax.view_init(azim=-135, elev=30)
ax.grid(False)
```

6.4 不等式：划定区域

如图 17 所示，代数中，等式 (equality) 可以是确定的值 ($x = 1$)、确定的直线 ($x + y = 1$)、确定的曲线 ($x^2 + y^2 = 1$)、确定的平面 ($-x + y - z + 1 = 0$) 等等。

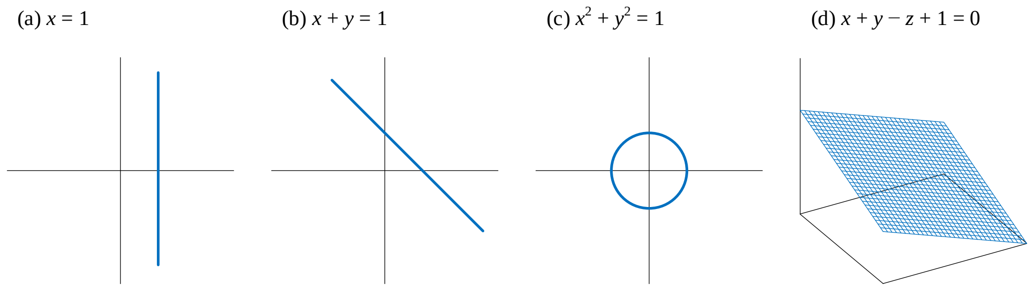


图 17. 等式的几何意义

然而，如图 18 所示，不等式 (inequality) 的几何意义则是划定区域，比如 x 的取值范围 ($x < 1$)、直线在平面上划定的区域 ($x + y \le 1$)、曲线在平面上划定的区域 ($x^2 + y^2 > 1$)、平面分割三维空间 ($-x + y - z + 1 < 0$)。图 18 中当边界为虚线时，意味着划定区域不包括蓝色线。

注意，图 18 中蓝色箭头指向满足不等式区域方向，它和梯度向量 (gradient vector) 有关。本系列丛书《矩阵力量》一册将介绍梯度向量相关内容。

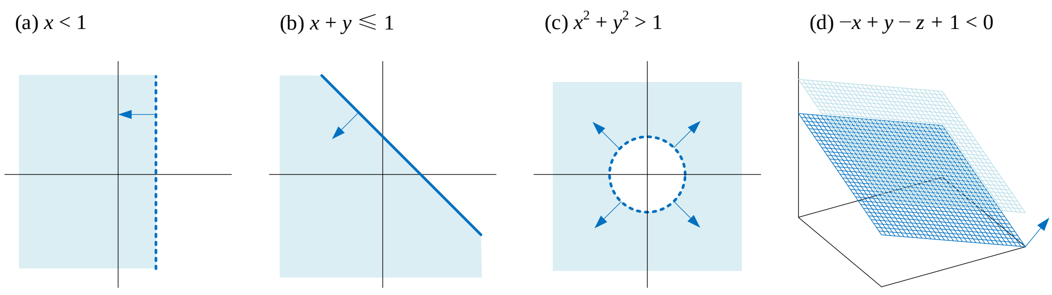


图 18. 不等式的几何意义

这两幅图告诉我们几何视角是理解代数式最直接的方式；本书在讲解每个数学工具式，都会给大家提供几何视角，以便加强理解，请大家格外留意。

数轴、绝对值、大小

为了理解不等式，让我们首先回顾数轴这个概念，数轴上的每一个点都对应一个实数，数轴上原点右侧的数为正数，原点左侧的数为负数。直角坐标系就是由数轴构造。

某个数的绝对值 (absolute value) 是指，数轴上该数与原点的距离；$|-5| = 5$ (读作 the absolute value of negative four equals four) 可以理解为−5 距离原点的距离为 5 个单位长度。x 的绝对值记做 $|x|$ (读作 absolute value of x)。而显然，实数绝对值为非负数，即 $|x| \geqslant 0$。

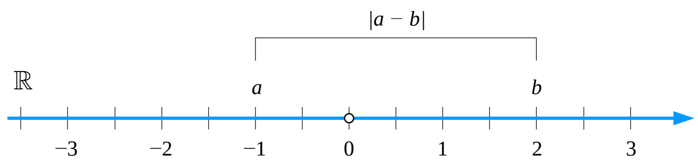


图 19. 实数轴上比较 a 和 b 大小

如果两个实数相等，这就意味着它们位于数轴同一点；当两个数不相等时，位于数轴右侧的数更大。如图 19 所示，实数 a 小于实数 b，可以表达为 $a < b$ (读作 a is less than b)；也可以说，在数轴上 a 在 b 的右侧 (a is to the left of b on the number line)。

这种用不等号 (inequality sign) 表达的式子被称作为不等式。表 1 总结六个不等式符号。

表 1. 六个不等式符号

数学表达	英文表达	汉语表达
$<$	less than	小于
$>$	greater than	大于
$\leq$	less than or equal to	小于等于
$\geq$	greater than or equal to	大于等于
$\ll$	much less than	远小于
$\gg$	much greater than	远大于

表 2. 不等式相关的英文表达

数学表达	英文表达
$4 > 3$	Four is greater than three. Three is less than four
$y \leq 9$	Small y is less than or equal to nine.
$x \geq -1$	Small x is greater than or equal minus one.
$-3 < x < 2$	Small x is greater than minus three and less than two.
$0 \leq x \leq 1$	x is greater than or equal to zero and less than or equal to one.
$a < b$	a is less than b.
$a > b$	a is greater than b.

$a \le b$	a is less than or equal to b. a is not greater than b.
$a \ge b$	a is greater than or equal to b. a is not less than b.
$a \ll b$	a is much less than b.
$a \gg b$	a is much greater than b.
$a \approx b$	a is approximately equal to b.
$a \ne b$	a is not equal to b.

区间

在数学上，某个变量的上下界可以写成区间。区间 (interval) 是指某个范围的数的集合，一般以集合形式表示。通用的区间记号中，圆括号表示“排除”，方括号表示“包括”。

如图 20 (a) 所示，开区间 (open interval) 不包括区间左右端点，可以记作 (a, b)，两端均为圆括号 (parentheses)。

如图 20 (b) 所示，闭区间 (closed interval) 包括区间两端端点，可以记作 $[a, b]$，两端均为方括号 (square brackets)。

如图 20 (c) 所示，左开右闭区间 (left-open and right-closed)，可以记做 $(a, b]$，不包括区间左端点、包括右端点。如图 20 (d) 所示，左闭右开区间 (right-open and left-closed)，可以记做 $[a, b)$，包括区间左端点、不包括右端点。

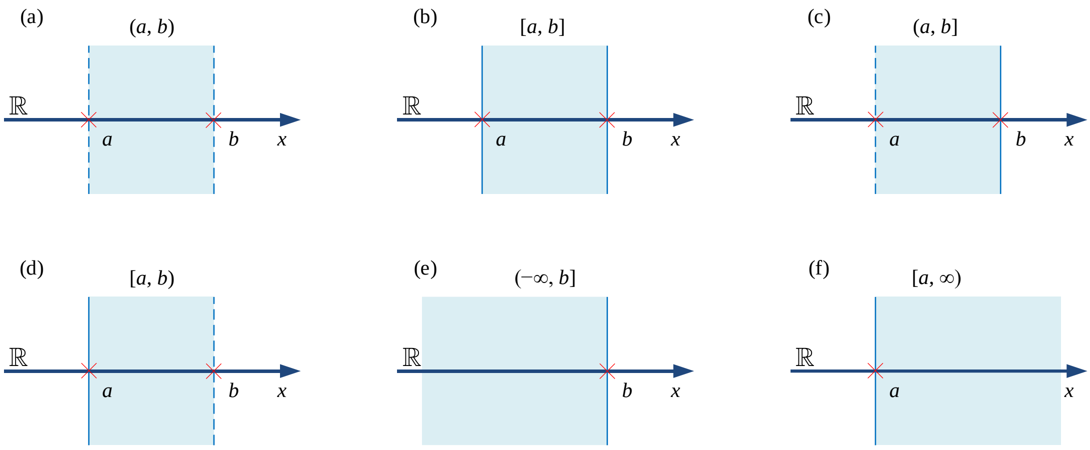


图 20. 六个区间

请大家特别注意，在优化问题求解中，如果变量两端均有界，一般只考虑闭区间，即可以取到端点数值。也就是，图 20 中前四个区间在优化问题中等价，a 叫做下界 (lower bound)，b 叫做上界 (upper bound)。

此外，构造优化问题时，一般都将各种不等式符号调整为小于等于号，即“≤”。本书后文将在优化入门一章专门讲解。

区间两端可能有界 (bounded) 或无界 (unbounded)，也就是区间某个可能没有端点，可以为无穷；正无穷 (infinity) 记作 ∞ 或 $+\infty$，负无限 (negative infinity) 记作 $-\infty$。

图 20 (e) 所示为左无界右有界 (left-unbounded and right-bounded)，比如 $(-\infty, b]$；图 20 (f) 所示为左有界右无界 (left-bounded and right-unbounded)，比如 $[a, \infty)$；左右均无界 (unbounded at both ends)，比如 $(-\infty, \infty)$，也就是整根实数轴。

表 3. 区间相关的英文表达

数学表达	英文表达
(a, b) $\{x \in \mathbb{R} \mid a < x < b\}$	The open interval from *a* to *b*. The interval from *a* to *b*, exclusive. The values between *a* and *b*, but not including the endpoints. *x* is greater than *a* and less than *b*. The set of all *x* such that *x* is in between *a* and *b*, exclusive.
$[a, b]$ $\{x \in \mathbb{R} \mid a \le x \le b\}$	The closed interval from *a* to *b*. The interval from *a* to *b*, inclusive. The values between *a* and *b*, including the endpoints. *x* is greater than or equal to *a* and less than or equal to *b*. The set of all *x* such that *x* is in between *a* and *b*, inclusive.
$(a, b]$ $\{x \in \mathbb{R} \mid a < x \le b\}$	The half-open interval from *a* to *b*, excluding *a* and including *b*. The values between *a* and *b*, excluding *a* and including *b*. The set of all *x* such that *x* is greater than *a* but less than or equal to *b*.
$[a, b)$ $\{x \in \mathbb{R} \mid a \le x < b\}$	The half-open interval from *a* to *b*, including *a* and excluding *b*. The values between *a* and *b*, including *a* and excluding *b*. The set of all *x* such that *x* is greater than or equal to *a* but less than *b*.

三大类不等式

本节介绍不等式的目的是服务优化问题求解，因此将不等式分为三大类。

◄ 上下界 (lower and upper bounds)，比如 $x > 2$

◄ 线性不等式 (linear inequalities)，比如 $x + y \le 1$

◄ 非线性不等式 (nonlinear inequalities)，比如 $x^2 + y^2 \geqslant 1$

在优化问题中，这些不等式统称为约束 (constraint)。本节后续将采用三种可视化方案呈现不等式划定的区域。

上下界

给定 x_1 的取值范围为。

$$x_1 + 1 > 0 \tag{9}$$

首先将上式大于号调整为小于号，(9) 改写成。

$$-x_1 - 1 < 0 \tag{10}$$

注意，本节后续不再区分 < 和 ≤。

根据 (10)，构造如下二元函数 $f(x_1, x_2)$。

$$f\left(x_1, x_2\right) = -x_1 - 1 \tag{11}$$

图 21 (a) 所示为三维直角坐标系中 $f(x_1, x_2)$ 的等高线图。对于一个二元函数 $f(x_1, x_2)$，等高线代表函数值相等的点连成曲线，即 $f(x_1, x_2) = c$。函数等高线类似地形图上海拔高度相同点连成曲线；等高线可以在三维空间展示，也可以在平面上绘制。

对于等高线这个概念陌生的读者不要怕，下一章我们会深入介绍等高线；此外，本书后续将专门讲解常用二元函数，本节内容相当于热身。

图 21 (a) 等高线采用“红黄蓝”色谱。暖色系颜色等高线对应 $f(x_1, x_2) > 0$，即不满足 (10)；冷色系颜色等高线对应 $f(x_1, x_2) < 0$，满足 (10)。图 21 (a) 中等高线相互平行。

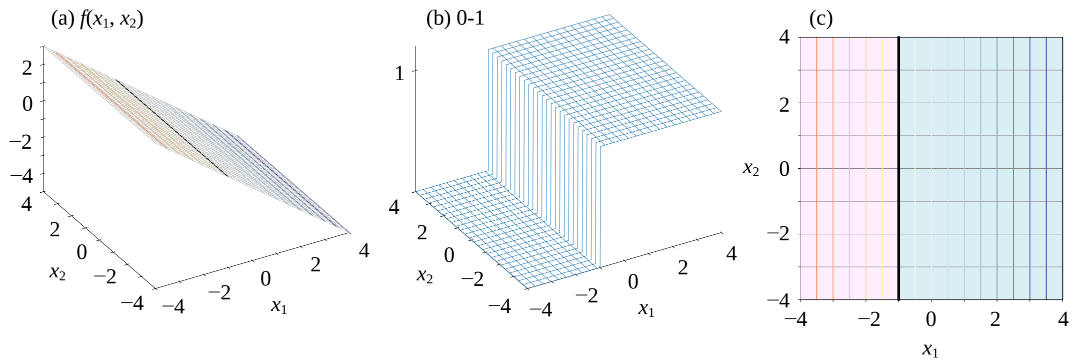


图 21. $x_1 + 1 > 0$ 三个可视化方案

然后，我们做一个“二分类”转换，满足 (10) 不等式的点 (x_1, x_2) 设为 1 (即 True)，不满足 (10) 的点设为 0 (即 False)，这样我们获得图 21 (b)。相当于把 $f(x_1, x_2)$ 变成一个 0-1 (False-True) 两值曲面。

再进一步，将图 21 (a) 等高线投影在 x_1x_2 平面上，获得图 21 (c) 平面等高线；图 21 (c) 中，蓝色阴影区域满足 (10)，对应图 21 (b) 中取值为 1 的区域。图 21 (c) 中黑色线就是决策边界，它将整个 x_1x_2 平面划分成两个区域，一个满足 (10)，一个不满足 (10)。

再举个例子，x_1 的取值范围给定为。

$$-1 < x_1 < 2 \tag{12}$$

利用绝对值运算，将 (12) 整理为。

$$\left|x_1 - 0.5\right| - 1.5 < 0 \tag{13}$$

根据 (13)，构造如下二元函数 $f(x_1, x_2)$。

$$f\left(x_1, x_2\right) = \left|x_1 - 0.5\right| - 1.5 \tag{14}$$

图 22 (a) 所示为 $f(x_1, x_2)$ 函数在三维直角坐标系中图像，整个曲面呈现 V 字形；同样，蓝色等高线处满足 (13)，而红色等高线处不满足 (13)。图 22 (b) 中取值 1 的区域满足 (13)。图 22 (c) 中背景色为蓝色区域满足 (13)。图 22 (c) 中两条黑色线为决策边界，两者相互平行。

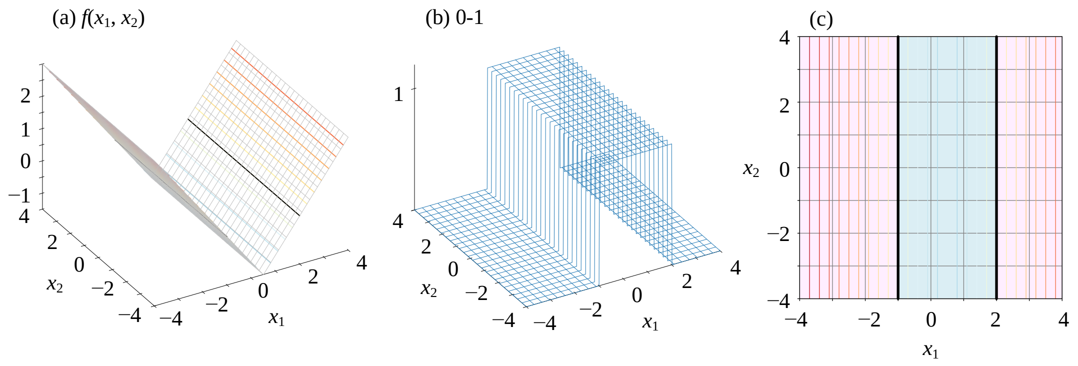


图 22. $-1 < x_1 < 2$ 三个可视化方案

再给定 x_2 的取值范围。

$$x_2 < 0 \text{ or } x_2 > 2 \tag{15}$$

将 (15) 整理为。

$$-|x_2 - 1| + 1 < 0 \tag{16}$$

根据 (16) 构造如下二元函数 $f(x_1, x_2)$。

$$f(x_1, x_2) = -|x_2 - 1| + 1 \tag{17}$$

图 23 (a) 所示为二元函数 $f(x_1, x_2)$ 在三维直角坐标系中图像。图 23 (b) 中 1 表示满足 (15)，0 表示不满足 (15)。图 23 (c) 中蓝色背景色区域满足 (15)。

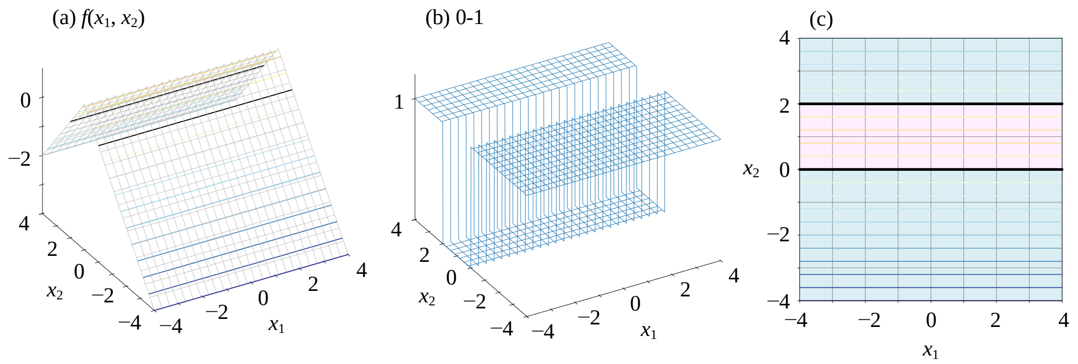


图 23. $x_2 < 0$ 或 $x_2 > 2$ 三个可视化方案

而几个不等式可以叠加构成不等式组。比如，(12) 和 (15) 两个不等式叠加得到。

$$\begin{cases} -1 < x_1 < 2 \\ x_2 < 0 \ \text{ or } \ x_2 > 2 \end{cases} \tag{18}$$

这相当于在 x_1x_2 平面上，同时限定了 x_1 和 x_2 的取值范围。图 24 所示为同时满足 (18) 两组不等式的区域。请大家根据本节文末代码，自行绘制这两幅图像。

此外，(15) 就相当于两个不等式叠加，请大家用不等式叠加的思路再重新分别探讨 (15)。

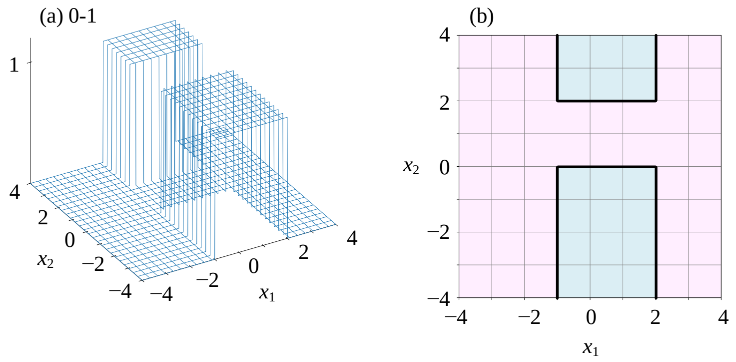


图 24. 同时满足$-1 < x_1 < 2$ 和 $x_2 < 0$ 或 $x_2 > 2$ 对应区域

线性不等式

线性不等式就是一次不等式，也就是不等式中单项式的次数最高为 1 次。线性不等式中可以含有若干未知量。虽然上下界也可以看做是线性不等式，但是在构造优化问题时我们还是将两类不等式分开处理。

给定如下线性不等式。

$$x_1 - x_2 < -1 \tag{19}$$

将 (19) 整理为。

$$x_1 - x_2 + 1 < 0 \tag{20}$$

构造如下二元函数 $f(x_1, x_2)$。

$$f(x_1, x_2) = x_1 - x_2 + 1 \tag{21}$$

图 25 (a) 所示为 $f(x_1, x_2)$ 在三维直角坐标系的图像。图 25 (b) 中取值为 1 的区域满足 (20)。图 25 (c) 中蓝色阴影的区域满足 (20)，黑色直线对应等式 $x_1 - x_2 + 1 = 0$。

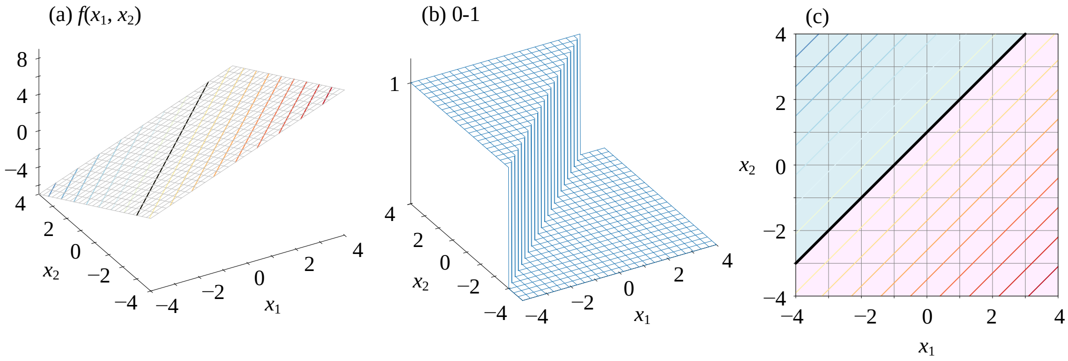


图 25. $x_1 - x_2 < -1$ 三个可视化方案

再给一个例子，给定如下线性不等式。

$$x_1 > 2x_2 \tag{22}$$

将 (22) 整理为。

$$-x_1 + 2x_2 < 0 \tag{23}$$

根据 (23)，构造如下二元函数 $f(x_1, x_2)$。

$$f(x_1, x_2) = -x_1 + 2x_2 \tag{24}$$

图 26 (a) 中蓝色等高线满足 (22)，而红色等高线不满足 (22)。图 26 (b) 中取值为 1 和图 26 (c) 中蓝色阴影区域满足 (22)。

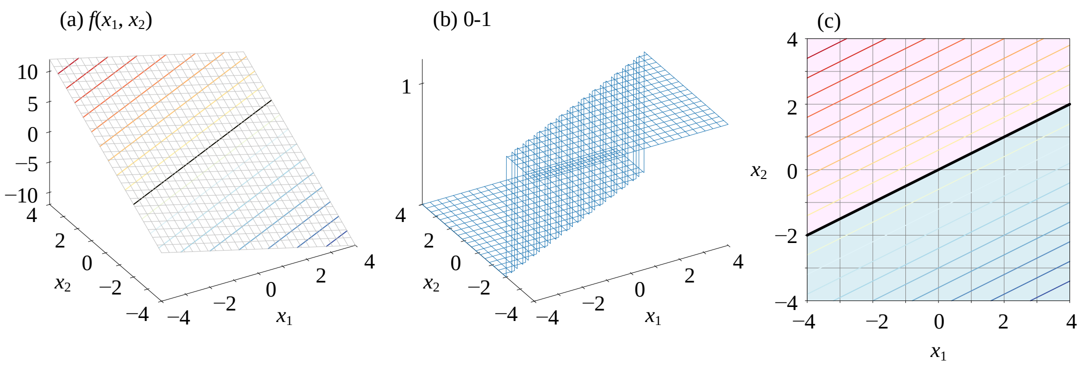


图 26. $x_1 > 2x_2$ 三个可视化方案

请大家将 (19) 和 (22) 两个不等式叠加构造一个不等式组，并绘制类似图 24 两图，可视化其划定的区域。

非线性不等式

除了线性不等式之外，其他各种形式的不等式都可以称作非线性不等式。下面举三个例子。

给定如下绝对值构造的不等式。

$$|x_1 + x_2| < 1 \tag{25}$$

(25) 整理为。

$$|x_1 + x_2| - 1 < 0 \tag{26}$$

构造如下二元函数 $f(x_1, x_2)$。

$$f(x_1, x_2) = |x_1 + x_2| - 1 \tag{27}$$

图 27 (a) 所示为 (27) 对应三维直角坐标系图像。图 27 (b) 中取值为 1 对应的区域和图 27 (c) 中蓝色阴影区域满足 (25)。

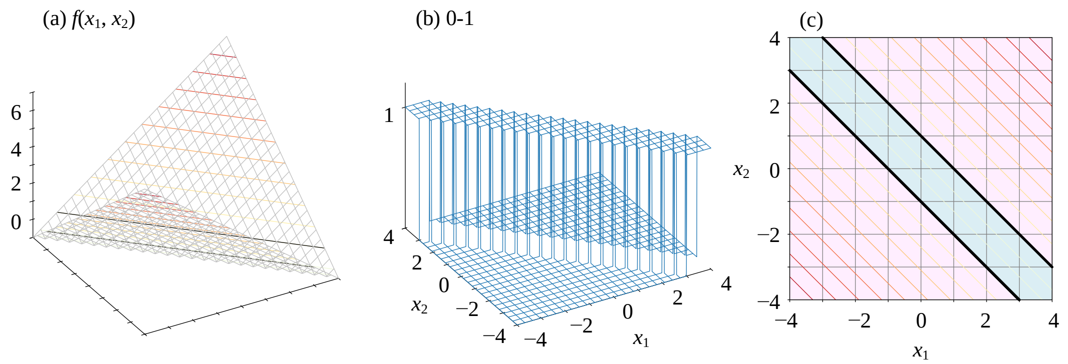


图 27. $|x_1 + x_2| < 1$ 三个可视化方案

此外，(25) 等价于

$$(x_1 + x_2)^2 < 1 \tag{28}$$

请大家自行绘制 (28) 对应的三幅图像。

给定第二个用绝对值构造的不等式。

$$|x_1| + |x_2| < 2 \tag{29}$$

将上式整理为。

$$|x_1| + |x_2| - 2 < 0 \tag{30}$$

构造如下二元函数 $f(x_1, x_2)$。

$$
f\left(x_1, x_2\right)=\left|x_1\right|+\left|x_2\right|-2 \tag{31}
$$

图 28 所示为 $f(x_1, x_2)$ 等高线，有意思的是等高线为一个个旋转 45°的正方形。大家还会在很多不同的应用场合看到类似图像。图 28 (b) 中取值为 1 对应的区域和图 28 (c) 中蓝色阴影区域满足 (29)

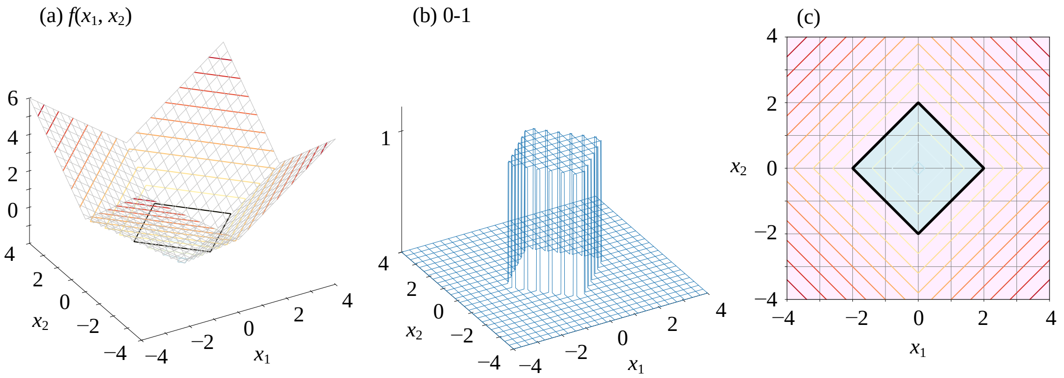


图 28. $\left|x_1\right|+\left|x_2\right|<2$ 三个可视化方案

再看个例子，给定如下非线性不等式。

$$
x_1^2+x_2^2<4 \tag{32}
$$

首先将整理为。

$$
x_1^2+x_2^2-4<0 \tag{33}
$$

在 x_1x_2 平面上，构造如下二元函数 $f(x_1, x_2)$。

$$
f\left(x_1, x_2\right)=x_1^2+x_2^2-4 \tag{34}
$$

图 29 (a) 所示为 (34) 中二元函数对应的曲面，曲面的等高线为同心圆；这种同心圆等高线还会在本书中反复出现，请大家留意。图 29 (b) 中取值为 1 对应的区域和图 28 (c) 中蓝色阴影区域满足 (29)。

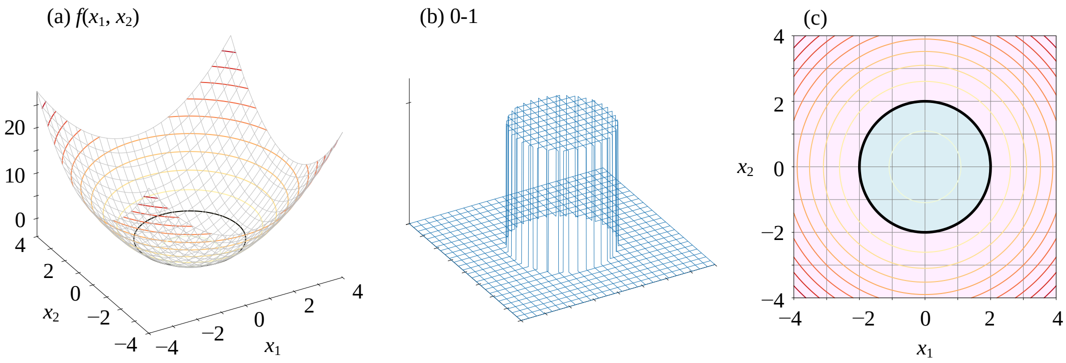


图 29. $x_1^2+x_2^2<4$ 三个可视化方案

此外，(32) 相当于。

$$\sqrt{x_1^2 + x_2^2} < 2 \tag{35}$$

请大家自行绘制 (35) 对应的三幅图像。另外，请将 (25) 和 (32) 两个不等式叠加构造不等式组，并绘制取值区域。

以下代码绘制本节大部分图像。

```python
# Bk_Ch6_03

import math
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

# define three visualization tools
# ========================
# 3D contour plot of zz
# ========================

def plot_3D_f_xy(xx,yy,zz):
    fig = plt.figure(figsize = (8,8))
    ax = fig.gca(projection='3d')

    ax.plot_wireframe(xx,yy, zz,
                      color = [0.75,0.75,0.75],
                      cmap='RdYlBu_r',
                      rstride=20, cstride=20,
                      linewidth = 0.25)

    l_max = max(np.max(zz),-np.min(zz))
    levels = np.linspace(-l_max,l_max,21)
    ax.contour(xx, yy, zz, levels = levels, cmap = 'RdYlBu_r')

    # plot decision boundary
    ax.contour(xx, yy, zz, levels = [0],
               colors=['k'])

    ax.set_proj_type('ortho')

    ax.set_xlim(xx.min(),xx.max())
    ax.set_ylim(yy.min(),yy.max())

    plt.tight_layout()
    ax.set_xlabel('$x_1$')
    ax.set_ylabel('$x_2$')
    ax.set_zlabel('f($x_1$,$x_2$)')
    ax.view_init(azim=-120, elev=30)
    ax.grid(False)

# ========================
# Wireframe plot of mask
# ========================

def plot_3D_mask(xx,yy,mask):
    fig = plt.figure(figsize = (8,8))
    ax = fig.gca(projection='3d')

    ax.plot_wireframe(xx,yy, mask,
                      cmap='RdYlBu_r',
                      rstride=20, cstride=20,
                      linewidth = 0.25)
```

```
    ax.set_proj_type('ortho')

    ax.set_xlim(xx.min(),xx.max())
    ax.set_ylim(yy.min(),yy.max())
    ax.set_zlim(0,1.2)

    plt.tight_layout()
    ax.set_xlabel('$x_1$')
    ax.set_ylabel('$x_2$')
    ax.set_zlabel('[0,1]')
    ax.set_zticks([0,1])
    ax.view_init(azim=-120, elev=30)
    ax.grid(False)

# ========================
# 2D contour plot
# ========================

def plot_2D_contour(xx,yy,zz,mask):

    # Create color maps
    rgb = [[255, 238, 255],  # red
           [219, 238, 244]]  # blue
    rgb = np.array(rgb)/255.

    cmap_light = ListedColormap(rgb)

    fig, ax = plt.subplots(figsize = (8,8))
    l_max = max(np.max(zz),-np.min(zz))
    levels = np.linspace(-l_max,l_max,21)
    plt.contourf(xx, yy, mask, cmap=cmap_light)
    plt.contour(xx, yy, zz, levels = levels, cmap = 'RdYlBu_r')

    # plot decision boundary
    plt.contour(xx, yy, zz, levels = [0],
                colors=['k'])

    # Figure decorations
    plt.xlim(xx.min(), xx.max())
    plt.ylim(yy.min(), yy.max())

    ax.grid(linestyle='--', linewidth=0.25, color=[0.5,0.5,0.5])
    # plt.axis('equal')
    plt.show()
    plt.xlabel('$x_1$')
    plt.ylabel('$x_2$')

#%%

num = 500
x = np.linspace(-4,4,num)
y = np.linspace(-4,4,num)
xx,yy = np.meshgrid(x,y);

plt.close('all')

#%% x1 + 1 > 0
zz = -xx - 1
# satisfy the inequality: 1
# otherwise: 0
mask_less_than_0 = (zz < 0) + 0

plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)

#%% -1 < x1 < 2
zz = np.abs(xx - 0.5) - 1.5
mask_less_than_0 = (zz < 0) + 0
```

```
plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)

#%% x2 < 0 or x2 > 2
zz = -np.abs(yy - 1) + 1
mask_less_than_0 = (zz < 0) + 0

plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)

#%% x1 - x2 + 1 < 0
zz = xx - yy + 1
mask_less_than_0 = (zz < 0) + 0

plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)

#%% x1 > 2*x2
zz = - xx + 2*yy
mask_less_than_0 = (zz < 0) + 0

plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)

#%% |x1 + x2| < 1
zz = np.abs(xx + yy) - 1
mask_less_than_0 = (zz < 0) + 0

plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)

#%% |x1| + |x2| < 2
zz = np.abs(xx) + np.abs(yy) - 2
mask_less_than_0 = (zz < 0) + 0

plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)

#%% x1**2 + x2**2 < 4
zz = xx**2 + yy**2 - 4
mask_less_than_0 = (zz < 0) + 0

plot_3D_f_xy(xx,yy,zz)
plot_3D_mask(xx,yy,mask_less_than_0)
plot_2D_contour(xx,yy,zz,mask_less_than_0)
```

6.5 三维极坐标

三维空间中也可以构造类似平面极坐标的坐标系统，如图 30 (a) 所示的**球坐标系** (spherical coordinate system)和图 30 (b) 所示的**圆柱坐标系** (cylindrical coordinate system)。

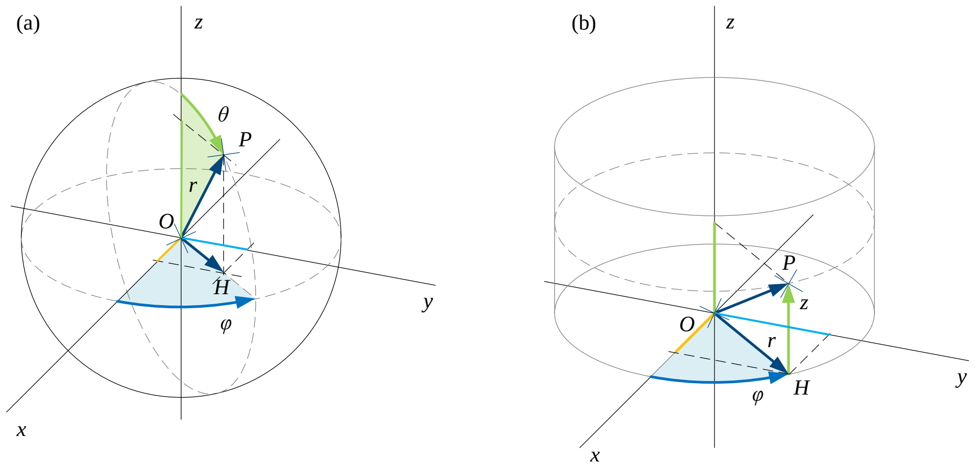


图 30. 球坐标系和圆柱坐标系

球坐标系

图 30 (a) 所示，球坐标相当于由两个平面极坐标系构造；球坐标系中定位点 P 用的是球坐标 (r, θ, φ)；其中，r 是 P 与原点 O 之间距离，也叫径向距离 (radial distance)；θ 是 OP 连线和 z 轴正方向夹角，叫做极角 (polar angle)；OP 连线在 xy 平面投影线为 OH，φ 是 OH 和 x 轴正方向向夹角，叫做方位角 (azimuth angle)。

球坐标到三维直角坐标系坐标的转化关系为。

$$\begin{cases} x = r\sin\theta\cdot\cos\varphi \\ y = r\sin\theta\cdot\sin\varphi \\ z = r\cos\theta \end{cases} \tag{36}$$

图 31 所示正圆球体对应解析式为。

$$x_1^2 + x_2^2 + x_3^2 = r^2 \tag{37}$$

其中，$r = 1$。在绘制这个正圆球体时，采用的就是球坐标。

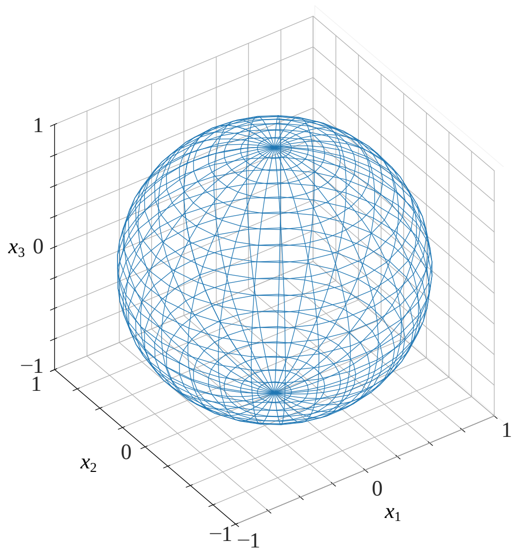


图 31. 球体网格面

以下代码绘制图 31。

```
# Bk_Ch6_04

import numpy as np
import matplotlib.pyplot as plt

u = np.linspace(0, np.pi, 30)
v = np.linspace(0, 2 * np.pi, 30)

x = np.outer(np.sin(u), np.sin(v))
y = np.outer(np.sin(u), np.cos(v))
z = np.outer(np.cos(u), np.ones_like(v))

fig = plt.figure()
ax = fig.gca(projection='3d')
# ax.set_aspect('equal')

ax.plot_wireframe(x, y, z)
ax.set_xlabel('$\it{x_1}$')
ax.set_ylabel('$\it{x_2}$')
ax.set_zlabel('$\it{x_3}$')
ax.view_init(azim=-125, elev=30)
```

圆柱坐标系

图 30 (b) 所示，圆柱坐标系相当于二维极坐标上升起一根 z 轴。在圆柱坐标系中，点 P 的坐标为 (r, φ, z)；这时，r 是 P 点与 z 轴的垂直距离；φ 还是 OP 在 xy 平面的投影线 OH 与正 x 轴之间的夹角。z 和三维直角坐标系的 z 一致。

从圆柱坐标到三维直角坐标系坐标转化关系为。

$$\begin{cases} x = r\cos\varphi \\ y = r\sin\varphi \\ z = z \end{cases} \tag{38}$$

上一章介绍的参数方程可以扩展到三维乃至多维。plot3d_parametric_line() 函数可以用来绘制参数方程构造的三维线图。

图 32 所示三维线图的参数方程就是采用圆柱坐标。

$$\begin{cases} x_1 = \cos(t) \\ x_2 = \sin(t) \\ x_3 = t \end{cases} \tag{39}$$

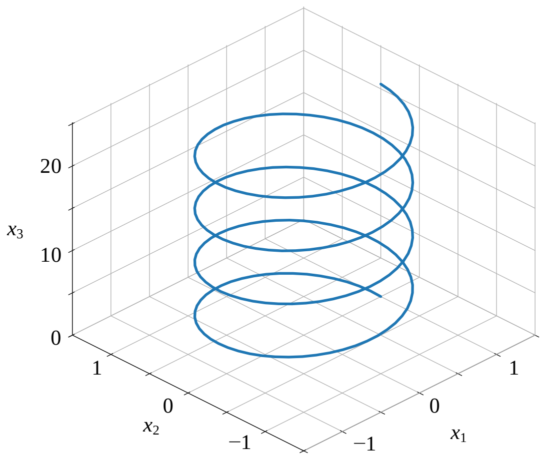


图 32. 三维参数方程线图

以下代码绘制图 32。

```
# Bk_Ch6_05

import numpy as np
import matplotlib.pyplot as plt

t = np.linspace(0,8*np.pi, 200)

# parametric equation of spiral
x1 = np.cos(t)
x2 = np.sin(t)
x3 = t

fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot(x1, x2, x3)

plt.show()
```

```
ax.set_proj_type('ortho')

ax.set_xlim(-1.5,1.5)
ax.set_ylim(-1.5,1.5)
ax.set_zlim(0,t.max())

plt.tight_layout()
ax.set_xlabel('$\it{x_1}$')
ax.set_ylabel('$\it{x_2}$')
ax.set_zlabel('$\it{x_3}$')

ax.view_init(azim=-135, elev=30)
ax.xaxis._axinfo["grid"].update({"linewidth":0.25, "linestyle" : ":"})
ax.yaxis._axinfo["grid"].update({"linewidth":0.25, "linestyle" : ":"})
ax.zaxis._axinfo["grid"].update({"linewidth":0.25, "linestyle" : ":"})
```

图 32 也可以用 `plot3d_parametric_line()` 函数绘制，具体代码如下。

```
from sympy import *
from sympy.plotting import plot3d_parametric_line
import math

t = symbols('t')

# parametric equation of spiral
x1 = cos(t)
x2 = sin(t)
x3 = t

plot3d_parametric_line(x1, x2, x3, (t, 0, 8*math.pi))
```

前文提过，坐标系让代数和几何紧密结合，坐标系让代数可视化，使几何参数化。

坐标系给一个个函数插上了翅膀，让它们能够在二维平面和三维空间自由翱翔。函数是本书接下来重点讲解的内容。