16 Partial Derivative 偏导数

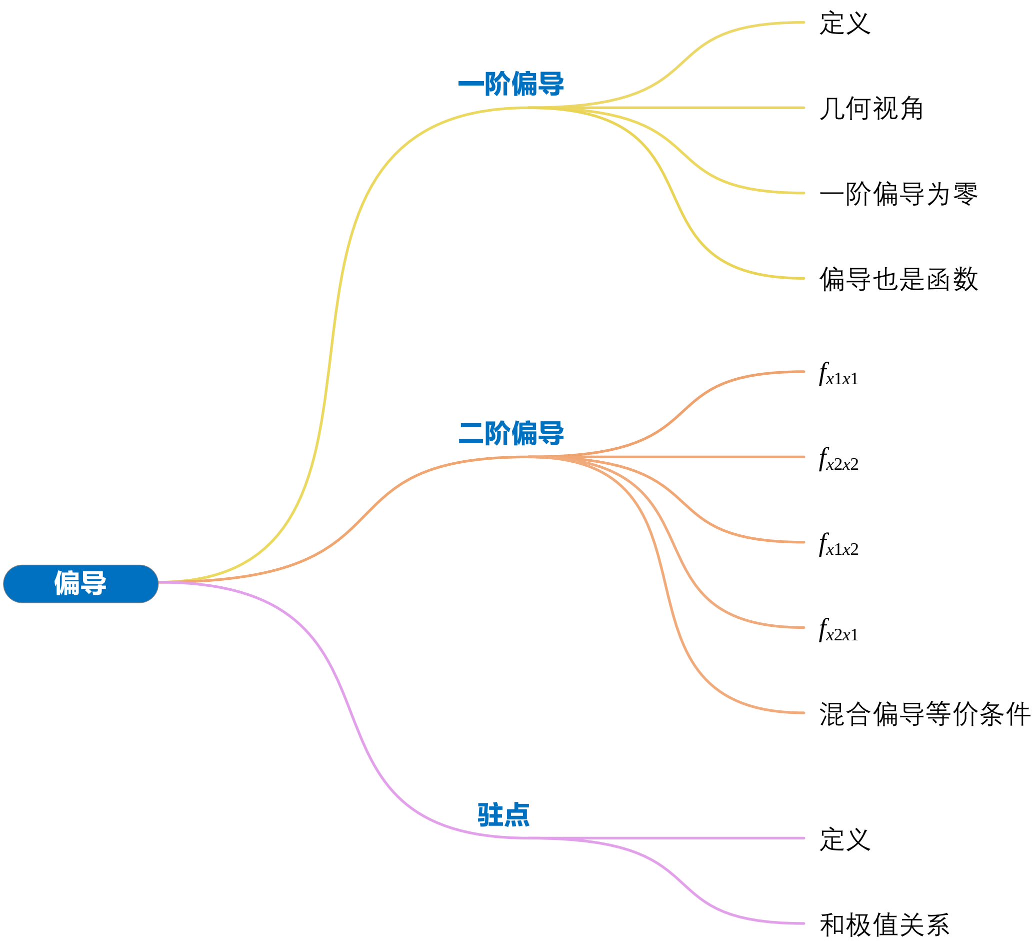
只对多元函数一个变量求导，其他变量保持定值

我不知道世人看我的眼光。依我看来，我不过是一个在海边玩耍的孩子，不时找到几个光滑卵石、漂亮贝壳，而惊喜万分；而展现在我面前的是，真理的浩瀚海洋，静候探索。

I do not know what I may appear to the world, but to myself I seem to have been only like a boy playing on the sea-shore, and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me.

——艾萨克·牛顿 (Isaac Newton) | 英国数学家、物理学家 | 1643 ~ 1727

- `ax.plot_surface()` 绘制三维曲面图
- `ax.plot_wireframe()` 绘制线框图
- `matplotlib.pyplot.contour()` 绘制等高线图
- `matplotlib.pyplot.contourf()` 绘制填充等高线图
- `sympy.abc` 引入符号变量
- `sympy.diff()` 求解符号导数和偏导解析式
- `sympy.exp()` 符号自然指数函数
- `sympy.lambdify()` 将符号表达式转化为函数
- `sympy.symbols()` 定义符号变量


偏导
一阶偏导
定义
几何视角
一阶偏导为零
偏导也是函数
二阶偏导
f_{x1x1}
f_{x2x2}
f_{x1x2}
f_{x2x1}
混合偏导等价条件
驻点
定义
和极值关系

16.1 几何角度看偏导数

本书前文介绍一元函数导数时，我们知道它是函数的变化率；从几何角度来看，导数就是一元函数曲线上某点切线的斜率。

之前，我们聊过一般情况二元函数 $f(x_1, x_2)$ 可以视作曲面。如图 1 所示，$f(x_1, x_2)$ 函数曲面上某一点 $(a, b, f(a, b))$ 如果光滑，该点处有无数条切线。

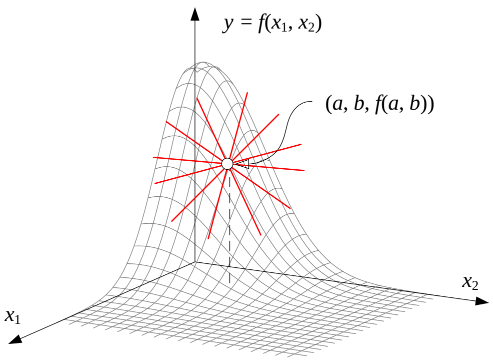


图 1. $f(x_1, x_2)$ 某点的切线

而我们特别关注的两条切线是图 2 (a) 和 (b) 红色直线对应的切线。图 2 (a) 中切线平行于 x_1y 平面，图 2 (b) 中切线平行于 x_2y 平面。这用到的就是本书前文介绍的剖面线思想。

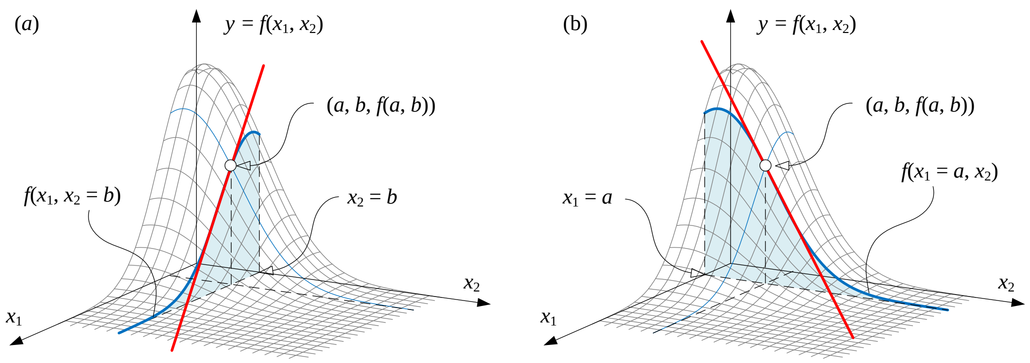


图 2. 几何视角看 $f(x_1, x_2)$ 偏导

对于这个二元函数，我们同样需要研究图 2 中两条切线斜率，也就是二元函数沿特定方向的变化率，这就需要**偏导数** (partial derivative) 这个数学工具。

对于多元函数 $f(x_1, x_2, \ldots, x_D)$ 来说，偏导数是关于函数的某一个特定变量 x_i 的导数，而保持其他变量恒定。

本节通过二元函数介绍偏导数的定义。

偏导数定义

设 $f(x_1, x_2)$ 是定义在 $\mathbb{R}^2$ 上的二元函数，$f(x_1, x_2)$ 在点 (a, b) 的某一邻域内有定义；将 x_2 固定在 $x_2 = b$，$f(x_1, b)$ 则变成一个关于 x_1 的一元函数，$f(x_1, b)$ 在 $x_1 = a$ 处关于 x_1 可导，则称 $f(x_1, x_2)$ 在点 (a, b) 处关于 x_1 **可偏微分** (partially differentiable)。

用极限，$f(x_1, x_2)$ 在点 (a, b) 处关于 x_1 的偏导定义为：

$$f_{x1}(a,b) = \left.\frac{\partial f}{\partial x_1}\right|_{\substack{x_1=a\\x_2=b}} = \lim_{\Delta x_1 \to 0} \frac{f\left(a+\Delta x_1, \overset{\text{Fixed}}{b}\right) - f\left(a, \overset{\text{Fixed}}{b}\right)}{\Delta x_1} \tag{1}$$

图 2 (a) 网格面为 $f(x_1, x_2)$ 函数曲面；从几何角度看偏导数，平行 x_1y 平面，在 $x_2 = b$ 切一刀得到浅蓝色的剖面线，偏导 $f_{x1}(a,b)$ 就是蓝色剖面线在 $(a, b, f(a, b))$ 一点的切线的斜率。注意，在三维直角坐标系中，该切线平行 x_1y 平面。

类似地，$f(x_1, x_2)$ 在 (a, b)点对于 x_2 的偏导可以定义为：

$$f_{x2}(a,b) = \left.\frac{\partial f}{\partial x_2}\right|_{\substack{x_1=a\\x_2=b}} = \lim_{\Delta x_2 \to 0} \frac{f(a, b+\Delta x_2) - f(a,b)}{\Delta x_2} \tag{2}$$

也从几何角度分析，如图 2 (b) 所示，偏导 $f_{x2}(a,b)$ 就是蓝色剖面线在 $(a, b, f(a, b))$ 一点的切线斜率。该切线平行 x_2y 平面。

一个多极值曲面

下面用如下这个较复杂二元函数 $f(x_1, x_2)$ 讲解偏导：

$$f(x_1,x_2) = 3(1-x_1)^2 \exp\left(-x_1^2 - (x_2+1)^2\right) - 10\left(\frac{x_1}{5} - x_1^3 - x_2^5\right)\exp\left(-x_1^2 - x_2^2\right) - \frac{1}{3}\exp\left(-(x_1+1)^2 - x_2^2\right) \tag{3}$$

对 x_1 偏导

图 3 给出 $f(x_1, x_2)$ 曲面上一系列散点。在每一个散点处，绘制平行于 x_1y 平面的切线，这些切线的斜率就是该点处 $f(x_1, x_2)$ 对 x_1 的偏导 $\partial f/\partial x_1 = f_{x1}$。

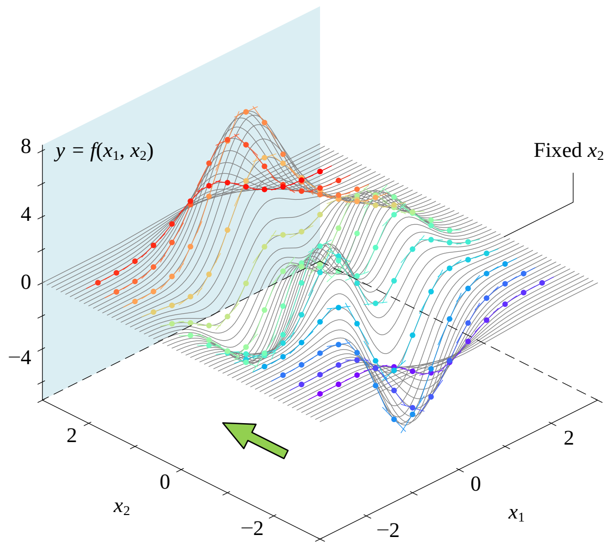


图 3. $f(x_1, x_2)$ 曲面上不同点处绘制 $f(x_1, x_2 = b)$ 切线

将这些切线投影到 x_1y 平面得到图 4。

如前文所述，固定 $x_2 = b$，$f(x_1, x_2)$ 这个二元函数变成了一个关于 x_1 的一元函数 $f(x_1, x_2 = b)$；不同 b 值对应不同的 $f(x_1, x_2 = b)$ 函数，对应图 4 中不同曲线。

在这些 $f(x_1, x_2 = b)$ 一元函数曲线上某点做切线，切线斜率就是二元函数 $f(x_1, x_2)$ 对 x_1 偏导。

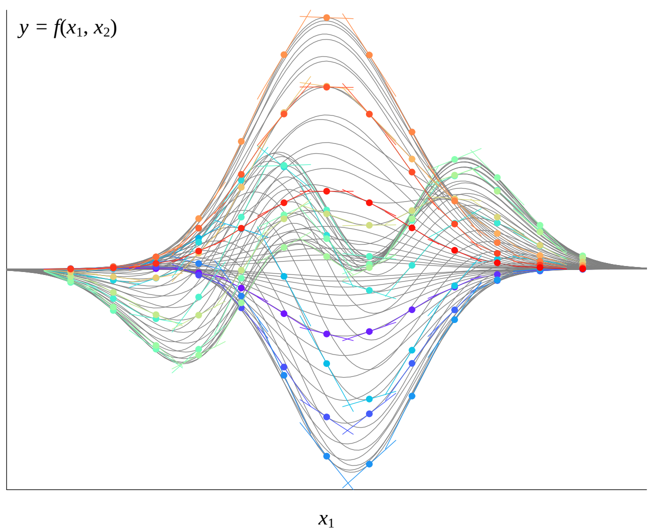


图 4. $f(x_1, x_2 = b)$ 函数和切线在 x_1y 平面投影

再次观察图 4，发现每一条曲线都能找到至少一条切线平行于 x_1 轴，也就是切线斜率为 0。将这些切线斜率为 0 的点连在一起可以得到图 5 中绿色曲线。不难看出，绿色曲线经过曲面的每个“山峰”和“山谷”，也就是二元函数极大值和极小值。

这一点观察对后续优化问题求解很重要。

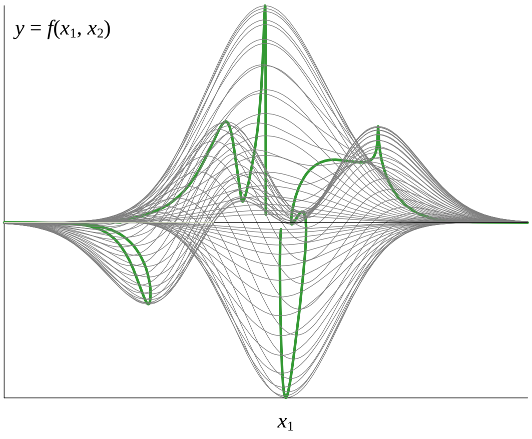


图 5. 将满足 $f_{x1}(x_1, x_2) = 0$ 的点连成线

对 x_2 偏导

下面，我们把同样的几何视角用在分析上 $f(x_1, x_2)$ 对 x_2 的偏导 $\partial f/\partial x_2 = f_{x2}$。

如图 6 所示，绘制 $f(x_1, x_2)$ 曲面上不同位置平行于 x_2y 平面的切线；而这些切线斜率就是不同点处 $f(x_1, x_2)$ 对 x_2 的偏导 $\partial f/\partial x_2 = f_{x2}$。

将这些切线投影到 x_2y 平面得到图 7。图中曲线都相当是一次函数，曲线上不同点切线斜率就是偏导；偏导用到的思维实际上也相当于“降维”，将三维曲面投影到平面上得到一系列曲线，然后再研究“变化率”。

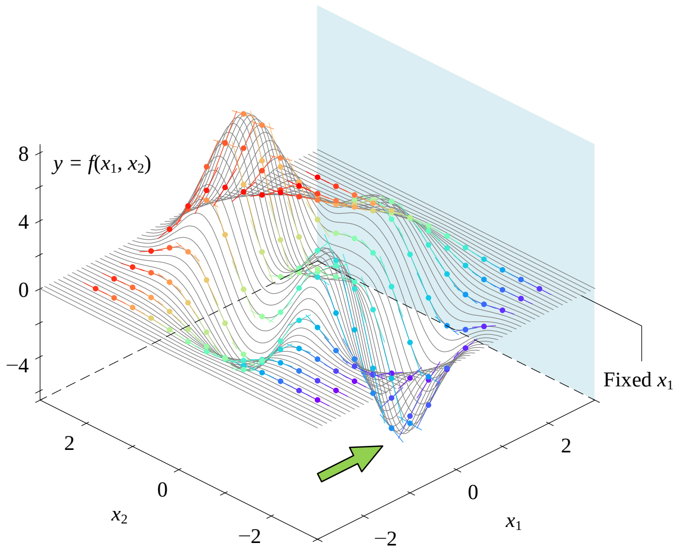


图 6. $f(x_1, x_2)$ 曲面上不同点处绘制 $f(x_1 = a, x_2)$ 切线

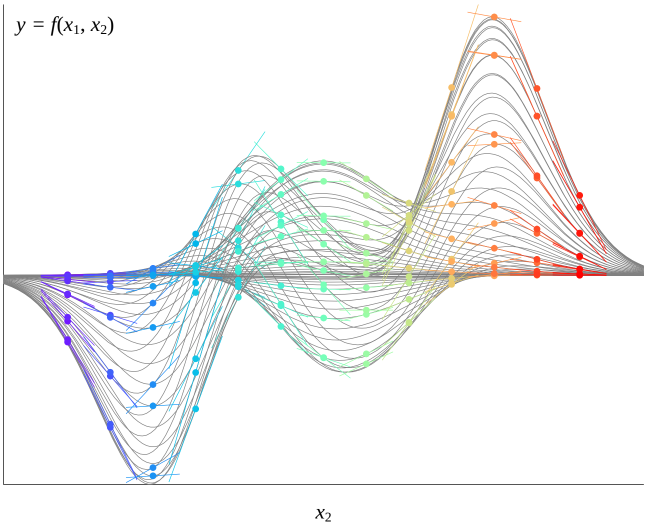


图 7. $f(x_1 = a, x_2)$ 函数和切线在 x_2y 平面投影

图 8 所示深蓝色曲线满足 $f_{x2}(x_1, x_2) = 0$；同样，我们发现这条深蓝色曲线经过曲面的"山峰"和"山谷"。本章后文会换一个视角来看图 5 中绿色曲线和图 8 深蓝色曲线。

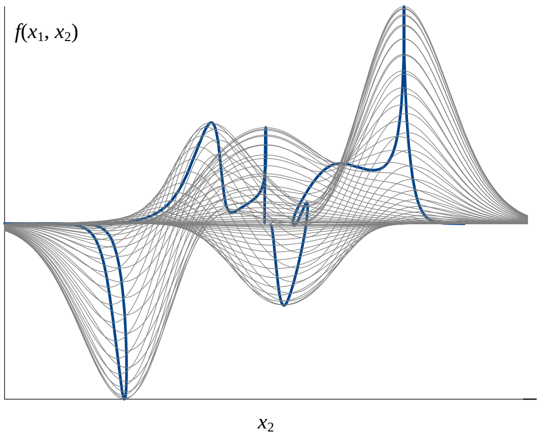


图 8. 将满足 $f_{x2}(x_1, x_2) = 0$ 的点连成线

本章开头说到，光滑曲面任意一点有无数条切线；也就是说，给定曲面一点 $(a, b, f(a, b))$ 从不同角度都可以获得曲面在该点处切线。

而对 x_1 偏导和对 x_2 偏导只能帮助我们定义两条切线。

大家可能会问，如何确定其他方向上切线斜率？这些"偏导数"又叫什么？

目前，我们已经掌握的数学工具尚不足以解决这个问题。我们把它留给本系列丛书《矩阵力量》一册。

表 1. 偏导数的英文表达

数学表达	英文表达
∂	Partial d, curly d, curved d, del
∂y	partial *y* the partial derivative of *y*
$\dfrac{\partial y}{\partial x}$	partial derivative of *y* with respect to *x* partial *y* over partial *x* partial derivative with respect to *x* of *y*
$\dfrac{\partial^2 y}{\partial x^2}$	Partial two *y* by partial *x* squared the second partial derivative of *y* with respect to *x*
$\dfrac{\partial^2 f}{\partial x \partial y}$	second partial derivative of *f*, first with respect to *x* and then with respect to *y*
$\dfrac{\partial f}{\partial x_1}$	the partial derivative of *f* with respect to *x* sub one partial d *f* over partial *x* sub one
$\dfrac{\partial^2 f}{\partial x_1^2}$	the second partial derivative of *f* with respect to *x* sub one Partial two *f* by partial *x* sub one squared

16.2 偏导也是函数

上一章说到导数也叫导函数，这是因为导数也是函数；同样，偏导也叫偏导函数，因为它也是函数。

对 x_1 偏导

计算 (3) 给出的二元函数 $f(x_1, x_2)$ 对于 x_1 的一阶偏导 $f_{x1}(x_1, x_2)$ 解析式：

$$
\begin{aligned}
f_{x1}(x_1, x_2) = & -6x_1(1-x_1)^2 \exp\left(-x_1^2-(x_2+1)^2\right) \\
& -2x_1\left(10x_1^3-2x_1+10x_2^5\right)\exp\left(-x_1^2-x_2^2\right) \\
& -\frac{1}{3}(-2x_1-2)\exp\left(-x_2^2-(x_1+1)^2\right) \\
& +(6x_1-6)\exp\left(-x_2^2-(x_1+1)^2\right) \\
& +\left(30x_1^2-2\right)\exp\left(-x_1^2-x_2^2\right)
\end{aligned}
\tag{4}
$$

可以发现，$f_{x1}(x_1, x_2)$ 也是一个二元函数。

图 9 所示为 $f_{x1}(x_1, x_2)$ 曲面，请大家格外注意图中绿色等高线，它们对应 $f_{x1}(x_1, x_2) = 0$；也就是图 5 中绿色曲线。

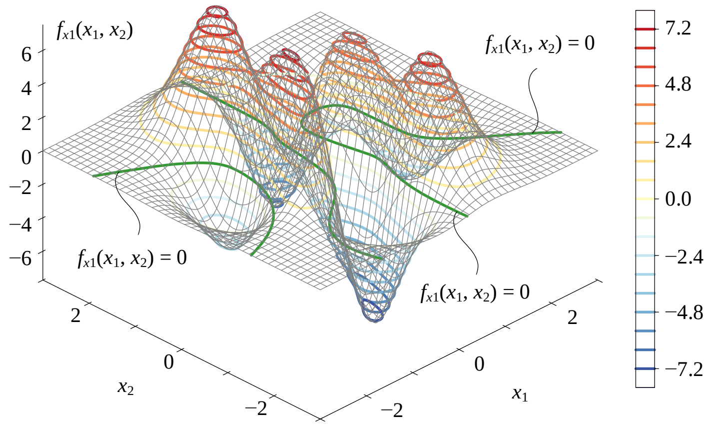


图 9. 二元函数 $f(x_1, x_2)$ 对 x_1 一阶偏导 $f_{x1}(x_1, x_2)$ 曲面

图 10 所示为 $f_{x1}(x_1, x_2)$ 平面填充等高线；从这个视角看 $f_{x1}(x_1, x_2) = 0$ 对应的绿色等高线更加方便。本章末将探讨绿色等高线和 $f(x_1, x_2)$ 曲面极值点的关系。

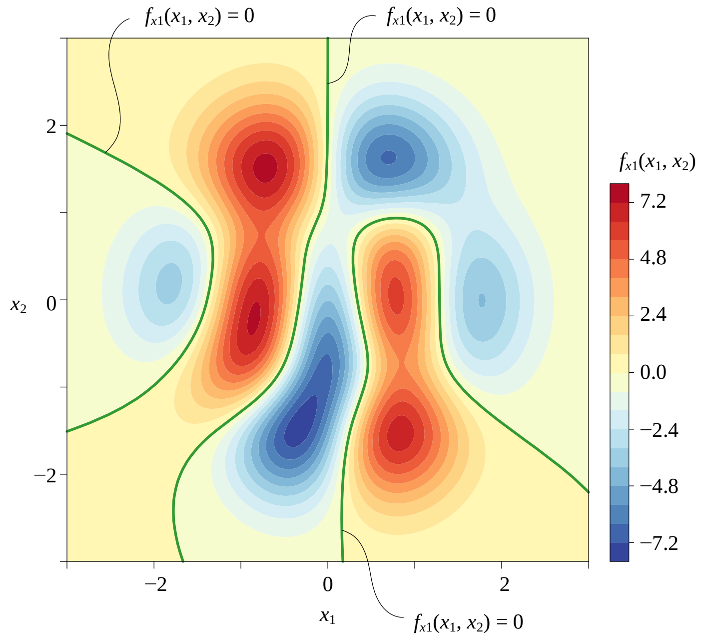


图 10. $f_{x1}(x_1, x_2)$ 平面填充等高线

代码文件 `Bk3_Ch16_01.py` 中 `Bk3_Ch16_01_A` 部分绘制图 9 和图 10。

```
# Bk3_Ch16_01_A

import numpy as np
from sympy import lambdify, diff, exp, latex
from sympy.abc import x, y
import numpy as np
from matplotlib import pyplot as plt

num = 301; # number of mesh grids
x_array = np.linspace(-3,3,num)
y_array = np.linspace(-3,3,num)
xx,yy = np.meshgrid(x_array,y_array)

plt.close('all')

# f_xy = x*exp(- x**2 - y**2);
f_xy =  3*(1-x)**2*exp(-(x**2) - (y+1)**2)\
    - 10*(x/5 - x**3 - y**5)*exp(-x**2-y**2)\
    - 1/3*exp(-(x+1)**2 - y**2)

f_xy_fcn = lambdify([x,y],f_xy)

f_xy_zz = f_xy_fcn(xx,yy)

#%% partial derivative with respect to x1

df_dx = f_xy.diff(x)
df_dx_fcn = lambdify([x,y],df_dx)
df_dx_zz = df_dx_fcn(xx,yy)

fig, ax = plt.subplots(subplot_kw={'projection': '3d'})

ax.plot_wireframe(xx,yy, df_dx_zz,
                  color = [0.5,0.5,0.5],
                  linewidth = 0.25)

colorbar = ax.contour(xx,yy, df_dx_zz,20,
             cmap = 'RdYlBu_r')

ax.contour(xx,yy, df_dx_zz, levels = [0],
           colors = '#339933',
           linestyles = '-')

fig.colorbar(colorbar, ax=ax)
ax.set_proj_type('ortho')
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$f_{x1}(x_1,x_2)$')
plt.tight_layout()
ax.set_xlim(xx.min(), xx.max())
ax.set_ylim(yy.min(), yy.max())
ax.view_init(azim=-135, elev=30)
ax.grid(False)
plt.show()

fig, ax = plt.subplots()

colorbar = ax.contourf(xx,yy, df_dx_zz, 20, cmap='RdYlBu_r')

ax.contour(xx,yy, df_dx_zz, levels = [0],
           colors = '#339933',
           linestyles = '-')

fig.colorbar(colorbar, ax=ax)
ax.set_xlim(xx.min(), xx.max())
ax.set_ylim(yy.min(), yy.max())
```

```
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
plt.gca().set_aspect('equal', adjustable='box')
plt.show()
```

对 x_2 偏导

配合前文代码，请自行计算 $f(x_1, x_2)$ 对于 x_2 的一阶偏导 $f_{x2}(x_1, x_2)$ 解析式。图 11 所示为 $f_{x2}(x_1, x_2)$ 曲面。

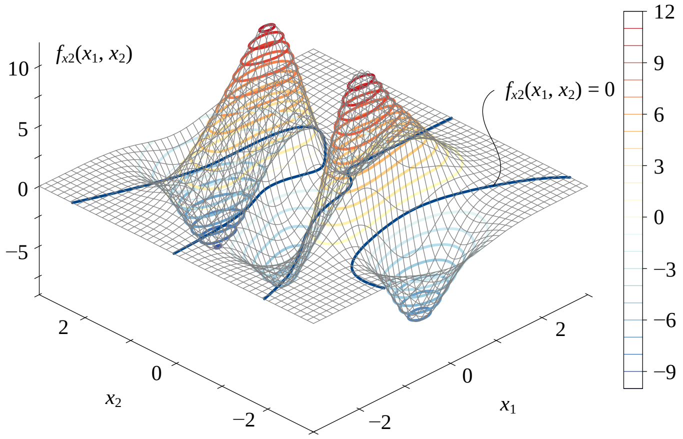


图 11. 二元函数 $f(x_1, x_2)$ 对 x_2 一阶偏导 $f_{x2}(x_1, x_2)$ 曲面

图 12 所示为 $f_{x2}(x_1, x_2)$ 曲面填充等高线，图中深蓝色等高线对应 $f_{x2}(x_1, x_2) = 0$。

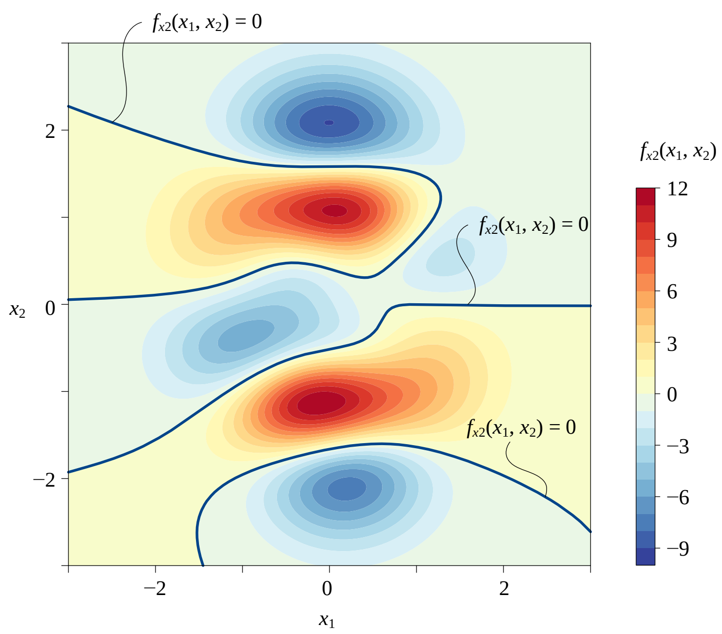


图 12. $f_{x2}(x_1, x_2)$ 平面填充等高线

代码文件 Bk3_Ch16_01.py 中 Bk3_Ch16_01_B 部分绘制图 11 和图 12。

```python
# Bk3_Ch16_01_B

#%% partial derivative with respect to x2

df_dy = f_xy.diff(y)
df_dy_fcn = lambdify([x,y],df_dy)
df_dy_zz = df_dy_fcn(xx,yy)

fig, ax = plt.subplots(subplot_kw={'projection': '3d'})

ax.plot_wireframe(xx,yy, df_dy_zz,
                  color = [0.5,0.5,0.5],
                  linewidth = 0.25)

colorbar = ax.contour(xx,yy, df_dy_zz,20,
             cmap = 'RdYlBu_r')

ax.contour(xx,yy, df_dy_zz, levels = [0],
           colors = '#00448A')

fig.colorbar(colorbar, ax=ax)

ax.set_proj_type('ortho')
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$f_{x2}(x_1,x_2)$')
plt.tight_layout()
ax.set_xlim(xx.min(), xx.max())
ax.set_ylim(yy.min(), yy.max())
ax.view_init(azim=-135, elev=30)
ax.grid(False)
plt.show()

fig, ax = plt.subplots()

colorbar = ax.contourf(xx,yy, df_dy_zz, 20, cmap='RdYlBu_r')

ax.contour(xx,yy, df_dy_zz, levels = [0],
           colors = '#00448A')

fig.colorbar(colorbar, ax=ax)
ax.set_xlim(xx.min(), xx.max()); ax.set_ylim(yy.min(), yy.max())
ax.set_xlabel('$x_1$'); ax.set_ylabel('$x_2$')
plt.gca().set_aspect('equal', adjustable='box')
plt.show()
```

16.3 二阶偏导：一阶偏导函数的一阶偏导

假设某个二元函数 $f(x_1, x_2)$ 对 x_1、x_2 分别具有偏导数 $f_{x1}(x_1, x_2)$、$f_{x2}(x_1, x_2)$；上一节内容告诉我们 $f_{x1}(x_1, x_2)$、$f_{x2}(x_1, x_2)$ 也是关于 x_1、x_2 的二元函数。

如果一阶偏导函数 $f_{x1}(x_1, x_2)$、$f_{x2}(x_1, x_2)$ 也有其各自一阶偏导数，则称它们是 $f(x_1, x_2)$ 的二阶偏导数。

对 x_1 二阶偏导

$f_{x1}(x_1, x_2)$ 对 x_1 求一阶偏导得到 $f(x_1, x_2)$ 对 x_1 的二阶偏导，记做。

$$\frac{\partial}{\partial x_1}\left(\frac{\partial f}{\partial x_1}\right)=\frac{\partial^2 f}{\partial x_1^2}=f_{x1x1}=\left(f_{x1}\right)_{x1} \tag{5}$$

图 13 所示为二阶偏导 f_{x1x1} 曲面和平面填充等高线。

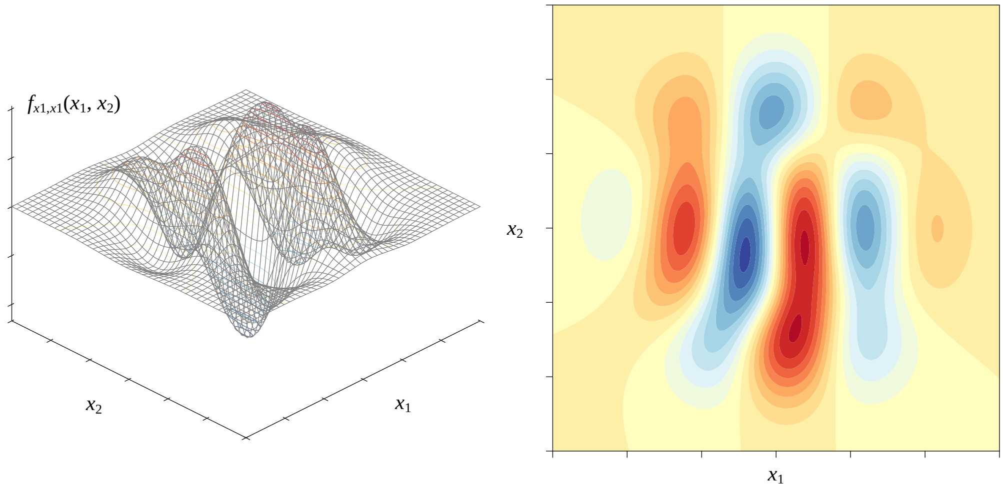


图 13. 二阶偏导 f_{x1x1} 曲面和平面填充等高线

对 x_2 二阶偏导

$f_{x2}(x_1, x_2)$ 对 x_2 求一阶偏导得到 $f(x_1, x_2)$ 对 x_2 的二阶偏导。

$$\frac{\partial}{\partial x_2}\left(\frac{\partial f}{\partial x_2}\right)=\frac{\partial^2 f}{\partial x_2^2}=f_{x2x2}=\left(f_{x2}\right)_{x2} \tag{6}$$

图 14 所示为二阶偏导 f_{x2x2} 曲面和平面填充等高线。

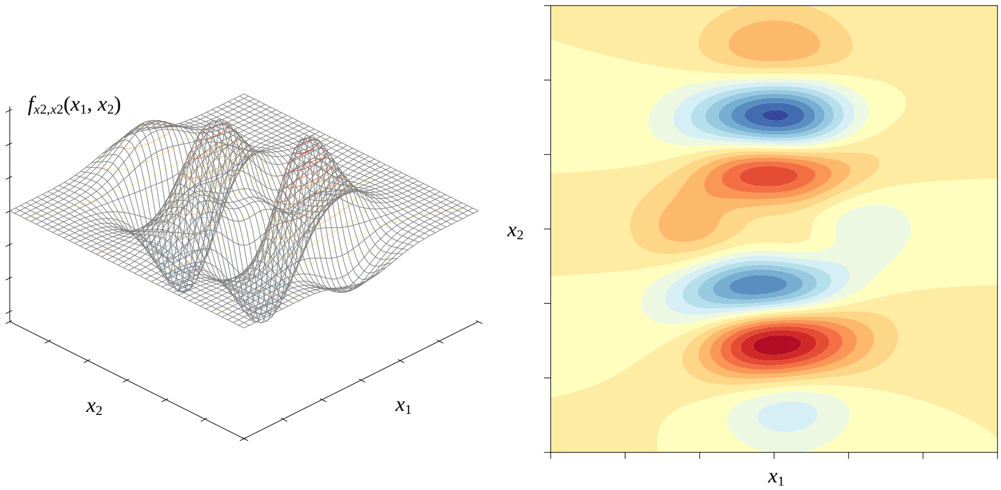


图 14. 二阶偏导 f_{x2x2} 曲面和平面填充等高线

二阶混合偏导

$f_{x1}(x_1, x_2)$ 对 x_2 求一阶偏导得到 $f(x_1, x_2)$ 先对 x_1、后对 x_2 二阶混合偏导，记做。

$$\frac{\partial}{\partial x_2}\left(\frac{\partial f}{\partial x_1}\right)=\underset{x_1 \to x_2}{\frac{\partial^2 f}{\partial x_1 \partial x_2}}=f_{x1x2}=\left(f_{x1}\right)_{x2} \tag{7}$$

请大家注意偏导先后顺序，先 x_1 后 x_2。

$f_{x2}(x_1, x_2)$ 对 x_1 求一阶偏导得到 $f(x_1, x_2)$ 先对 x_2、后对 x_1 二阶混合偏导。

$$\frac{\partial}{\partial x_1}\left(\frac{\partial f}{\partial x_2}\right)=\underset{x_2 \to x_1}{\frac{\partial^2 f}{\partial x_2 \partial x_1}}=f_{x2x1}=\left(f_{x2}\right)_{x1} \tag{8}$$

再次请大家注意混合偏导的先后顺序；不同教材的记法存在顺序颠倒。为了方便大家习惯，本章混合偏导记法采用同济大学编写的《高等数学》中记法规则。

如果函数 $f(x_1, x_2)$ 在某个特定区域内两个二阶混合偏导 f_{x2x1}、f_{x1x2} 连续，那么这两个混合偏导数相等，即。

$$\frac{\partial^2 f}{\partial x_2 \partial x_1}=\frac{\partial^2 f}{\partial x_1 \partial x_2} \tag{9}$$

对于 (3)，函数的二阶偏导连续；因此，f_{x2x1} 和 f_{x1x2} 等价。图 15 所示为二阶偏导 f_{x1x2} (= f_{x2x1}) 曲面和填充等高线。

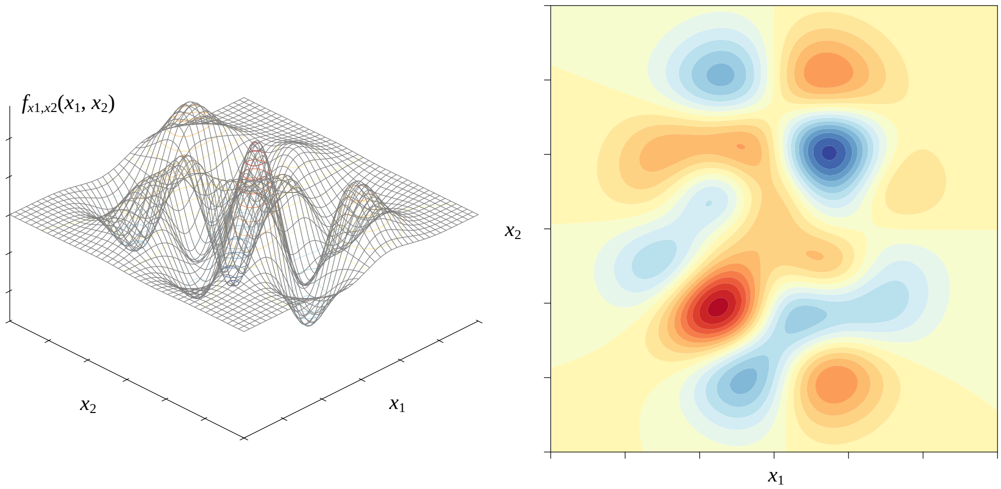


图 15. 二阶偏导 f_{x1x2} (= f_{x2x1}) 曲面和填充等高线

和杨辉三角的联系

图 16 所示为偏导数和杨辉三角的联系。

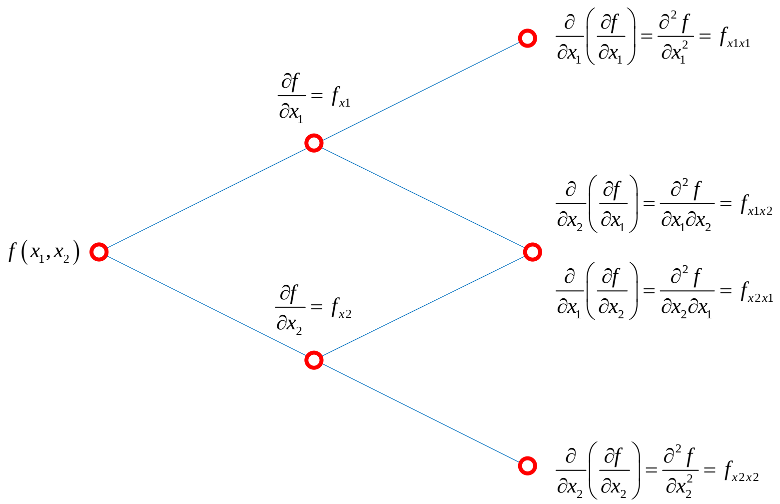


图 16. 杨辉三角在偏导数的应用

代码文件 Bk3_Ch16_01.py 中 Bk3_Ch16_01_C 部分绘制图 13、图 14、图 15。

```
# Bk3_Ch16_01_C

#%% second order partial derivatives

def plot_surface(xx, yy, surface, title_txt):
```

```
    fig = plt.figure(figsize=plt.figaspect(0.5))

    ax = fig.add_subplot(1, 2, 1, projection='3d')

    ax.plot_wireframe(xx,yy, surface,
                      color = [0.5,0.5,0.5],
                      linewidth = 0.25)

    colorbar = ax.contour(xx,yy, surface,20,
                 cmap = 'RdYlBu_r')

    # fig.colorbar(colorbar, ax=ax)
    ax.set_proj_type('ortho')
    ax.set_xlabel('$x_1$'); ax.set_ylabel('$x_2$')
    ax.set_zlabel(title_txt)
    ax.set_xlim(xx.min(), xx.max()); ax.set_ylim(yy.min(), yy.max())
    ax.view_init(azim=-135, elev=30)
    ax.grid(False)
    plt.show()

    ax = fig.add_subplot(1, 2, 2)

    colorbar = ax.contourf(xx,yy, surface, 20, cmap='RdYlBu_r')

    # fig.colorbar(colorbar, ax=ax)
    ax.set_xlim(xx.min(), xx.max()); ax.set_ylim(yy.min(), yy.max())
    ax.set_xlabel('$x_1$'); ax.set_ylabel('$x_2$')
    plt.gca().set_aspect('equal', adjustable='box')
    plt.show()

d2f_dxdy = f_xy.diff(x,y)
# d2f_dxdy = df_dy.diff(x)
# d2f_dxdy = df_dx.diff(y)
d2f_dxdy_fcn = lambdify([x,y],d2f_dxdy)
d2f_dxdy_zz = d2f_dxdy_fcn(xx,yy)

title_txt = '$f_{x1,x2}(x1,x2)$'
plot_surface(xx, yy, d2f_dxdy_zz, title_txt)

d2f_dxdx = f_xy.diff(x,2)
# d2f_dxdx = df_dx.diff(x)
d2f_dxdx_fcn = lambdify([x,y],d2f_dxdx)
d2f_dxdx_zz = d2f_dxdx_fcn(xx,yy)

title_txt = '$f_{x1,x1}(x1,x2)$'
plot_surface(xx, yy, d2f_dxdx_zz, title_txt)

d2f_dydy = f_xy.diff(y,2)
# d2f_dydy = df_dy.diff(y)
d2f_dydy_fcn = lambdify([x,y],d2f_dydy)
d2f_dydy_zz = d2f_dydy_fcn(xx,yy)

title_txt = '$f_{x2,x2}(x1,x2)$'
plot_surface(xx, yy, d2f_dydy_zz, title_txt)
```

16.4 二元曲面的驻点：一阶偏导为 0

上一章介绍过驻点这个概念。对于一次函数 $f(x)$，驻点处函数一阶导数为 0；从几何图像上来看，$f(x)$ 在驻点的切线平行于横轴。驻点可能对应一次函数的极小值、极大值或鞍点。

而对于二元函数，驻点对应一阶偏导为 0 的点。几何角度，驻点处切面平行于水平面。

对 x_1 一阶偏导为 0

图 9 和图 10 告诉我们 $f_{x1}(x_1, x_2) = 0$ 对应的坐标点 (x_1, x_2) 位置；如果将满足 $f_{x1}(x_1, x_2) = 0$ 等式的所有点映射到 $f(x_1, x_2)$ 曲面上，可以得到图 17 的绿色曲线。

仔细观察图 17 中绿色曲线，它们都经过 $f(x_1, x_2)$ 曲面上的极大值和极小值点。这一点，在图 18 填充等高线上看的更清楚。

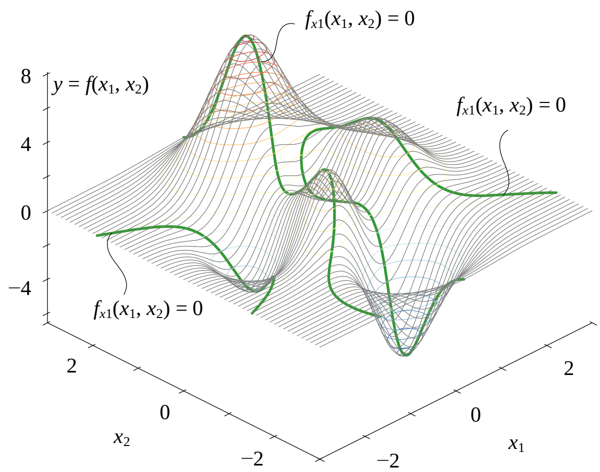


图 17. $f_{x1}(x_1, x_2) = 0$ 投影在 $f(x_1, x_2)$ 曲面上

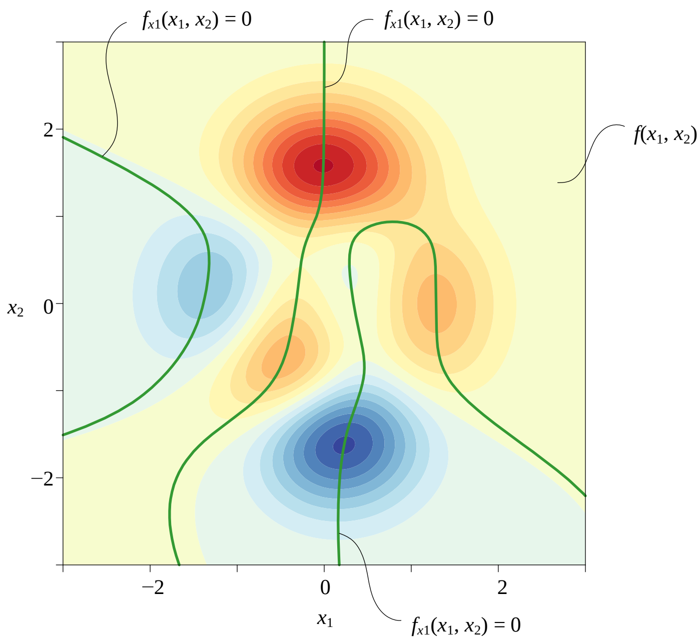


图 18. 将 $f_{x1}(x_1, x_2) = 0$ 投影在 $f(x_1, x_2)$ 曲面填充等高线上

对 x_2 一阶偏导为 0

同理，图 11 和图 12 给出 $f_{x2}(x_1, x_2) = 0$ 对应的坐标点 (x_1, x_2) 位置；将满足 $f_{x2}(x_1, x_2) = 0$ 等式的所有点映射到 $f(x_1, x_2)$ 曲面上，得到图 19 的蓝色曲线。图 19 中蓝色曲线也都经过 $f(x_1, x_2)$ 曲面上的极大值和极小值点。图 20 所示为平面填充等高线图。

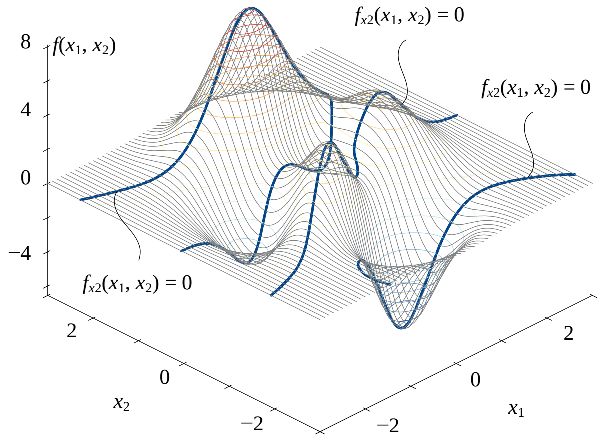


图 19. $f_{x2}(x_1, x_2) = 0$ 投影在 $f(x_1, x_2)$ 曲面上

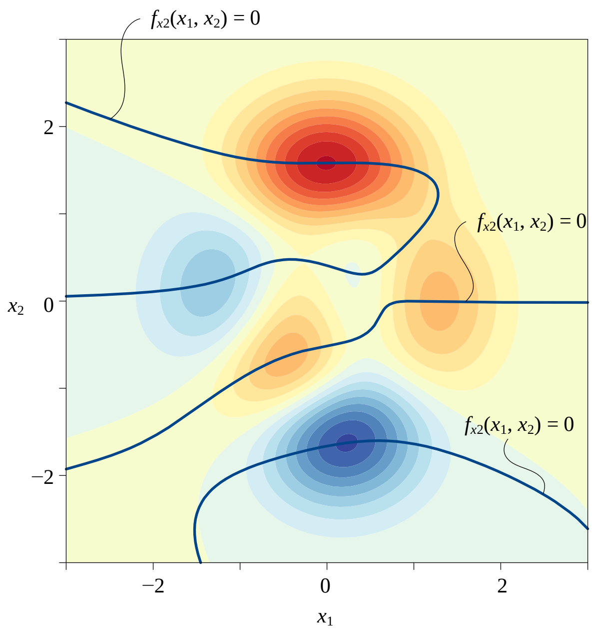


图 20. 将 $f_{x2}(x_1, x_2) = 0$ 投影在 $f(x_1, x_2)$ 曲面填充等高线上

二元函数驻点

将 $f_{x1}(x_1, x_2) = 0$ (绿色曲线) 和 $f_{x2}(x_1, x_2) = 0$ (蓝色曲线) 同时映射到 $f(x_1, x_2)$ 曲面，得到图 21。

$f(x_1, x_2)$ 曲面山峰和山谷，也就是极大和极小值点，正好都位于蓝色和绿色曲线的交点处。

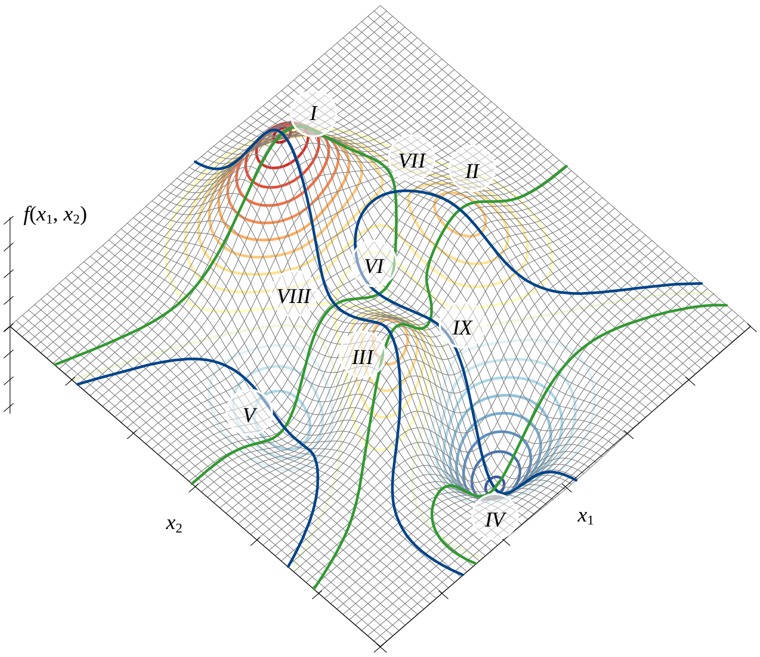


图 21. $f_{x1}(x_1, x_2) = 0$ 和 $f_{x2}(x_1, x_2) = 0$ 同时投影在 $f(x_1, x_2)$ 曲面上

图 22 给出的等高线更容易发现，I、II、III 点为极大值点，其中 I 为最大值点；IV、V、VI 为极小值点，其中 IV 为最小值点。

于此同时，我们也发现还有三个蓝绿曲线的交点 VII、VIII、IX，它们既不是极大值点，也不是极小值点；VII、VIII、IX 就是所谓的鞍点。

比如，在 IX 点，沿着绿色线向 IV 运动是下山，而沿着蓝色线向 III 运动是上山。

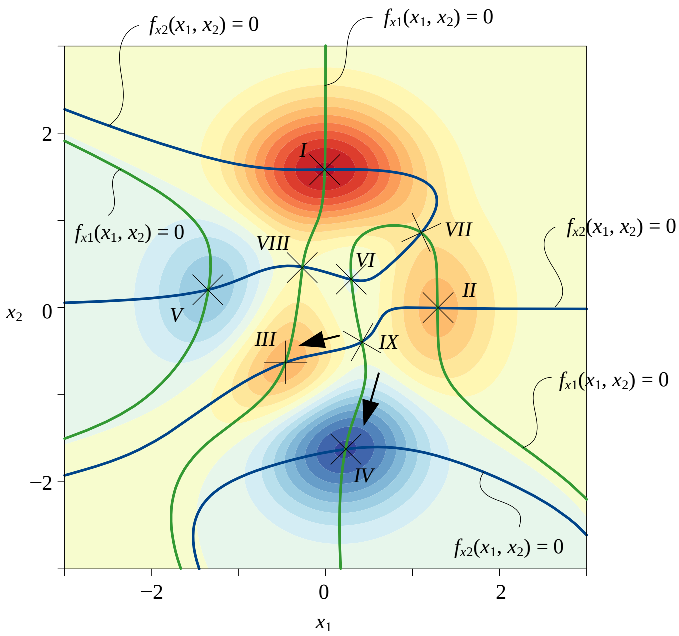


图 22. $f_{x1}(x_1, x_2) = 0$ 和 $f_{x2}(x_1, x_2) = 0$ 同时投影在 $f(x_1, x_2)$ 曲面填充等高线

对于具有多个“山峰”和“山谷”的曲面，利用一阶偏导为 0 来判断极值点显然不充分；本书后续将在优化问题一章，介绍如何判断二元函数的极值点。

代码文件 Bk3_Ch16_01.py 中 Bk3_Ch16_01_D 部分绘制图 18、图 20、图 21、图 22 四副图像。请读者自行绘制图 17 和图 19 两幅图像。

```
# Bk3_Ch16_01_D

fig, ax = plt.subplots()

colorbar = ax.contourf(xx,yy, f_xy_zz, 20, cmap='RdYlBu_r')
fig.colorbar(colorbar, ax=ax)
ax.contour(xx,yy, df_dx_zz, levels = [0],
           colors = '#339933')
ax.set_xlim(xx.min(), xx.max())
ax.set_ylim(yy.min(), yy.max())

ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
plt.gca().set_aspect('equal', adjustable='box')

plt.show()

fig, ax = plt.subplots()

colorbar = ax.contourf(xx,yy, f_xy_zz, 20, cmap='RdYlBu_r')
fig.colorbar(colorbar, ax=ax)
ax.contour(xx,yy, df_dy_zz, levels = [0],
           colors = '#00448A')
ax.set_xlim(xx.min(), xx.max())
ax.set_ylim(yy.min(), yy.max())

ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
plt.gca().set_aspect('equal', adjustable='box')

plt.show()


#%% stationary, intersections

fig, ax = plt.subplots(subplot_kw={'projection': '3d'})

CS_y = ax.contour(xx,yy, df_dy_zz, levels = [0],
           colors = '#339933')
CS_x = ax.contour(xx,yy, df_dx_zz, levels = [0],
           colors = '#339933')
ax.cla()

ax.plot_wireframe(xx,yy, f_xy_zz,
                  color = [0.5,0.5,0.5],
                  rstride=5, cstride=5,
                  linewidth = 0.25)

colorbar = ax.contour(xx,yy, f_xy_zz,20,
             cmap = 'RdYlBu_r')

fig.colorbar(colorbar, ax=ax)

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

    contour_points_x_y = CS_y.allsegs[0][i]
```

```python
    contour_points_z = f_xy_fcn(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 = '#339933',
              linewidth = 1)

for i in range(0,len(CS_x.allsegs[0])):
    contour_points_x_y = CS_x.allsegs[0][i]
    contour_points_z = f_xy_fcn(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 = '#00448A',
              linewidth = 1)

ax.set_proj_type('ortho')

ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$f(x_1,x_2)$')

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

ax.view_init(azim=-135, elev=30)
# ax.view_init(azim=-135, elev=60)
plt.tight_layout()
ax.grid(False)
plt.show()


fig, ax = plt.subplots()

colorbar = ax.contourf(xx,yy, f_xy_zz, 20, cmap='RdYlBu_r')
fig.colorbar(colorbar, ax=ax)

ax.contour(xx,yy, df_dx_zz, levels = [0],
           colors = '#339933',
           linestyles = '-')
ax.contour(xx,yy, df_dy_zz, levels = [0],
           colors = '#00448A',
           linestyles = '-')

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

ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')

plt.gca().set_aspect('equal', adjustable='box')

plt.show()
```

一元函数导数是函数变化率，几何角度是曲线切线斜率。本章利用"降维"这个思路，将一元函数导数这个数学工具拿来分析二元函数；对于二元函数或多元函数，我们给这个数学工具取了个名字叫做"偏导数"。"偏"字就是只考虑一个变量，或一个维度的意思。我们在介绍大西格玛 Σ 时，也创造了"偏求和"这个概念；在之后的积分内容中，我们还会见到"偏积分"。

本章还利用剖面线和等高线这两个可视化工具分析二元函数特征；请大家格外注意二元函数鞍点的性质。