10235501417 罗文琦

1.题目表述

8. 【案例分析】为了调查吃巧克力对心血管健康的影响, 实验由三种类型的巧克力组成: 100 g 的黑巧克力, 含有 200 mg 全脂牛奶的 100 g 黑巧克力和 200 g 的牛奶巧克力。12 个实验对象包含 7 女 5 男。在不同的天数里, 每个实验对象将吃一种类型的巧克力 (Chocolate), 一个小时后测量他们血浆的总抗氧能力 (Capacity)。实验次序本身具有随机性, 无须再随机化。数据信息如表 1.11 所示。

表 1.11 数据说明

变量名	变量含义	变量类型	变量取值范围
Chocolate	巧克力类型	分类变量	$\{1,2,3\}$
Capacity	血浆浓度	连续变量	R

请完成以下任务。注: 显著性水平 α 取 0.05。

(a) 两两比较 3 种巧克力对心血管健康的影响是否存在差异。

(b) 判断 3 种巧克力对心血管健康的影响是否存在差异。

(c) 试说明所使用模型的合理性。

(d) 估计食用这 3 种巧克力一小时后血浆的总抗氧能力。请分别给出点估计和区间估计。

(e) 用 Bonferroni 方法比较吃了 3 种巧克力后, 一个小时的血浆总抗氧能力两两之间是否存在差异。

(f) 用 Tukey 方法比较吃了 3 种巧克力后, 一个小时的血浆总抗氧能力两两之间是否存在差异。采用以下两种不同的方法来解决这个问题:

I. 直接调用 Python 中现有函数;

II. 用蒙特卡洛随机模拟分布的方式, 确定 t 化极差统计量的分位数 $q_{1-\alpha}(a,df)$, 计算临界值 (critical value) $c=q_{1-\alpha}(a,df)\hat{\sigma}/\sqrt{m}$。

(g) 基于这个例子, 请评述 Bonferroni 方法和 Tukey 方法的异同。

总共有a-g小问，作答代码附上，第g小问在实验报告中呈现：

In [1]:
```
import matplotlib
import shutil
cache_dir = matplotlib.get_cachedir()
shutil.rmtree(cache_dir, ignore_errors=True)
print(f"✅ 已清除缓存：{cache_dir}，请重启内核后重新运行！")
```

```
✅ 已清除缓存：/home/jovyan/.cache/matplotlib，请重启内核后重新运行！
```

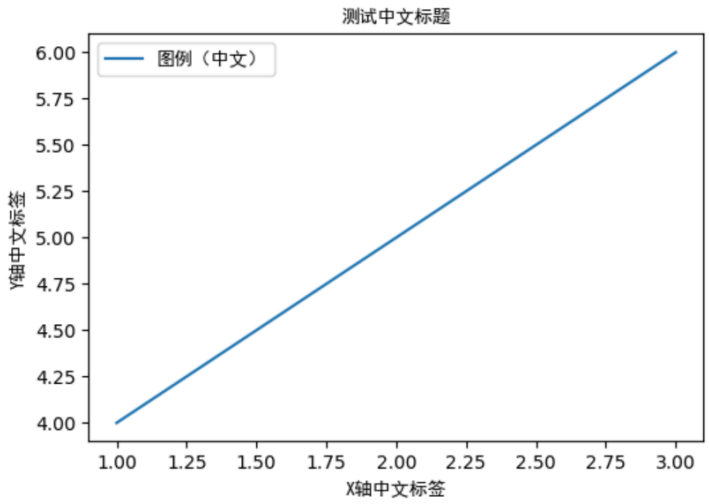
In [2]:
```
import zipfile
import os
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties

# --------------------------
# 步骤1：解压 simhei.ttf.zip 到当前目录
# --------------------------
zip_file = "simhei.ttf.zip"  # 压缩包文件名（与代码同目录）
extract_dir = "."            # 解压到当前目录

# 验证压缩包是否存在
if not os.path.exists(zip_file):
    raise FileNotFoundError(f"未找到压缩包：{zip_file}，请确认文件在项目目录！")

# 执行解压
with zipfile.ZipFile(zip_file, "r") as zip_ref:
    zip_ref.extractall(extract_dir)
    print(f"✅ 已解压 {zip_file} 到 {extract_dir}")

# --------------------------
# 步骤2：定位并加载 simhei.ttf 字体
# --------------------------
# 假设压缩包内直接包含 simhei.ttf（若在子文件夹，需修改路径，如 "fonts/simhei.ttf
font_path = os.path.join(extract_dir, "simhei.ttf")

# 验证字体文件是否存在
if not os.path.exists(font_path):
    # （可选）打印压缩包内所有文件，帮助排查路径
    print("压缩包内文件列表：")
    with zipfile.ZipFile(zip_file, "r") as zip_ref:
        for name in zip_ref.namelist():
            print(name)
    raise FileNotFoundError(f"未找到字体文件 simhei.ttf，请检查压缩包内路径（当前

# 创建字体属性对象
simhei_font = FontProperties(fname=font_path)
print(f"✅ 加载字体：{simhei_font.get_name()}，路径：{font_path}")

# --------------------------
# 步骤3：测试中文显示（绘图验证）
# --------------------------
```

2.相关前置知识总结

有关画图

```python
# 创建字体属性对象
simhei_font = FontProperties(fname=font_path)
print(f"✅ 加载字体：{simhei_font.get_name()}，路径：{font_path}")

# --------------------------
# 步骤3：测试中文显示（绘图验证）
# --------------------------
plt.figure(figsize=(6, 4))
plt.title("测试中文标题", fontproperties=simhei_font)
plt.xlabel("X轴中文标签", fontproperties=simhei_font)
plt.ylabel("Y轴中文标签", fontproperties=simhei_font)
plt.plot([1, 2, 3], [4, 5, 6], label="图例（中文）")
plt.legend(prop=simhei_font)  # 图例需用 prop 参数
plt.show()
```

```
Could not save font_manager cache [Errno 2] No such file or directory: '/home/jovyan/.cache/matplotlib/fontlist-v390.json.matplotlib-lock'
✅ 已解压 simhei.ttf.zip 到 .
✅ 加载字体：SimHei，路径：./simhei.ttf
```



1.首先下载中文字体到jupyterhub，使其能够在图中正确显示

2.Python 绘图的核心依赖 **Matplotlib**和 **NumPy**，核心逻辑围绕 “生成数据→绘制图像→个性化风格→保存图像” 展开，以下是分模块的核心内容提炼：

1. **导入库：** `import numpy as np; import matplotlib.pyplot as plt`
2. **生成数据：**用 `np.linspace`（连续）或 `np.random`（随机）生成数据
3. **创建画布与轴：** `fig, ax = plt.subplots(figsize=(宽, 高))`
4. **绘制图像：** `ax.plot()`（折线）或 `ax.scatter()`（散点）
5. **个性化风格：**设置线条 / 点 / 颜色、添加文字标注、分割子图
6. **保存 / 显示：** `fig.savefig(路径)` 或 `plt.show()`

3.以及一些数组索引部分内容：

```python
data_1 = load_data.values[0,:] #返回第一行
data_2 = load_data.values[0,0] #返回第(1,1)个元素
data_3 = load_data['Chocolate'].values #打印第一列Obs的数值
data_4 = load_data.values[1:30:2] #和数组一样的用法 list[start:end:stride]
data[:,0]就是取矩阵X的所有行的第0列的元素，X[:,1] 就是取所有行的第1列的元素
data[:,m:n]即取矩阵X的所有行中的的第m到n-1列数据，含左不含右。
data[0,:]就是取矩阵X的第0行的所有元素，X[1,:]取矩阵X的第一行的所有元素
```

单因子方差分析

单因素方差分析模型为

$$y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

其中，ϵ_{ij}是独立同分布的，且均服从均值为零，方差为σ^2的正态分布，即$\epsilon \overset{i.i.d}{\sim} N(0, \sigma^2)$.

- 独立性；
- 同方差；
- 正态性。

在单因素方差分析模型中，待估参数有

- μ: 总体平均水平；
- α_i: 第i个水平的效应值；
- σ^2: 误差方差。

μ的最大似然估计是

$$\hat{\mu} = \bar{y}_{..} = \frac{1}{am}\sum_{i=1}^{a}\sum_{j=1}^{m} y_{ij};$$

α_i的最大似然估计是

$$\hat{\alpha}_i = \bar{y}_{i\cdot} - \bar{y}_{..} = \frac{1}{m}\sum_{j=1}^{m} y_{ij} - \frac{1}{am}\sum_{i=1}^{a}\sum_{j=1}^{m} y_{ij};$$

σ^2的常用估计是

$$\hat{\sigma}^2 = \frac{SS_E}{a(m-1)} = \frac{\sum_{i=1}^{a}\sum_{j=1}^{m}(y_{ij} - \bar{y}_{i\cdot})^2}{n-a} = MS_E;$$

核心表格：

来源	平方和(SS)	自由度(df)	均方(MS)	检验统计量(F)	P-value
因子A	SS_A	$df_A = a-1$	$MS_A = \frac{S_A}{df_A}$	$F = \frac{MS_A}{MS_E}$	$p = P(Y \geq F)$
误差E	SS_E	$df_e = n-a$	$MS_E = \frac{S_E}{df_E}$		
总和T	SS_T	$df_T = n-1$			

相关公式总结：

以下为罗文琦纯手敲公式，并非截图！

参数：a是分类数，m是每一类中的样本数，n是总体样本数：$n = a * m$

$$
\begin{aligned}
&\text{总体：} SS_T = \sum_{i=1}^{a}\sum_{j=1}^{m_i}(y_{ij} - \bar{y})^2, && f_T = n - 1\\
&\text{组间：} SS_A = \sum_{i=1}^{a} m_i(\bar{y}_{i\cdot} - \bar{y})^2, && f_A = a - 1,\\
&\text{组内：} SS_E = \sum_{i=1}^{a}\sum_{j=1}^{m_i}(y_{ij} - \bar{y}_{i\cdot})^2, && f_E = n - a
\end{aligned}
$$

$$SS_T = SS_A + SS_E$$

a.两两比较三种巧克力对心血管的影响是否存在差异

[7]:
```
# --------------------------
# 3. 两两组合进行t检验（原逻辑保留，适配分组数据）
# --------------------------
pairs = [(0, 1), (0, 2), (1, 2)]  # 组索引组合（1vs2、1vs3、2vs3）
print("\n=== a.两两独立样本t检验结果 ===")
for i, j in pairs:
    group_i = groups[i]
    group_j = groups[j]

    # 步骤1：检验方差齐性（Levene检验，抗偏离正态性更好）
    lev_stat, lev_p = stats.levene(group_i, group_j)
    equal_var = lev_p > alpha  # True=方差齐性，False=方差不齐

    # 步骤2：独立样本t检验（根据方差齐性选择是否校正）
    t_stat, t_p = stats.ttest_ind(group_i, group_j, equal_var=equal_var)

    # 步骤3：计算t临界值（自由度根据方差齐性调整）
    if equal_var:
        df_t = len(group_i) + len(group_j) - 2  # 齐性时自由度：n1+n2-2
    else:
        df_t = min(len(group_i), len(group_j)) - 1  # 不齐时简化自由度（或用Welch-Satterthwaite公式）
    t_critical = t.ppf(1 - alpha/2, df_t)  # 双尾检验临界值

    print(f"\n{group_names[i]} vs {group_names[j]}:")
    print(f"  1. 方差齐性检验（Levene）：p={round(lev_p, 4)}（{'齐性' if equal_var else '不齐性'}）")
    print(f"  2. t检验：统计量={round(t_stat, 4)}，临界值={round(t_critical, 4)}，p值={round(t_p, 4)}")
    print(f"  3. 结论：{'拒绝原假设（存在显著差异）' if t_p < alpha else '无法拒绝原假设（无显著差异）'}")
```

```
=== a.两两独立样本t检验结果 ===

1(100g黑巧) vs 2(黑巧+牛奶):
  1. 方差齐性检验（Levene）：p=0.8887（齐性）
  2. t检验：统计量=11.1057，临界值=2.0739，p值=0.0
  3. 结论：拒绝原假设（存在显著差异）

1(100g黑巧) vs 3(200g牛奶巧):
  1. 方差齐性检验（Levene）：p=0.9657（齐性）
  2. t检验：统计量=12.0478，临界值=2.0739，p值=0.0
  3. 结论：拒绝原假设（存在显著差异）

2(黑巧+牛奶) vs 3(200g牛奶巧):
  1. 方差齐性检验（Levene）：p=0.827（齐性）
  2. t检验：统计量=0.4126，临界值=2.0739，p值=0.6839
  3. 结论：无法拒绝原假设（无显著差异）
```

b.判断三种巧克力对心血管健康的影响是否存在差异

[8]:
```
# --------------------------
# 4. 单因素ANOVA分析
# --------------------------
print("\n=== b.单因素ANOVA分析结果 ===")
# 方法1：基于statsmodels（输出详细ANOVA表，含平方和、均方）
model = ols("Capacity ~ C(Chocolate)", data=Data).fit()  # C(Chocolate)标记分类变量为因子
anova_table = anova_lm(model)
print("方法1：statsmodels ANOVA表（保留4位小数）:")
print(round(anova_table, 4))

# 方法2：基于scipy.stats.f_oneway（快速计算F统计量和p值）
f_stat, f_p = stats.f_oneway(groups[0], groups[1], groups[2])
# 计算F临界值（自由度df1=a-1，df2=n-a）
df1 = a - 1  # 因子自由度：水平数-1
df2 = n - a  # 误差自由度：总样本量-水平数
f_critical = f.ppf(1 - alpha, df1, df2)

print(f"\n方法2：scipy F检验:")
print(f"  F统计量={round(f_stat, 4)}，临界值={round(f_critical, 4)}，p值={round(f_p, 4)}")
print(f"  结论：{'拒绝原假设（三种巧克力影响存在显著差异）' if f_p < alpha else '无法拒绝原假设（无显著差异）'}")
```

```
=== b.单因素ANOVA分析结果 ===
方法1：statsmodels ANOVA表（保留4位小数）:
                df     sum_sq   mean_sq        F  PR(>F)
C(Chocolate)   2.0  1952.6439  976.3219  93.5756     0.0
Residual      33.0   344.3058   10.4335      NaN     NaN

方法2：scipy F检验:
  F统计量=93.5756，临界值=3.2849，p值=0.0
  结论：拒绝原假设（三种巧克力影响存在显著差异）
```

c.分析模型使用的合理性

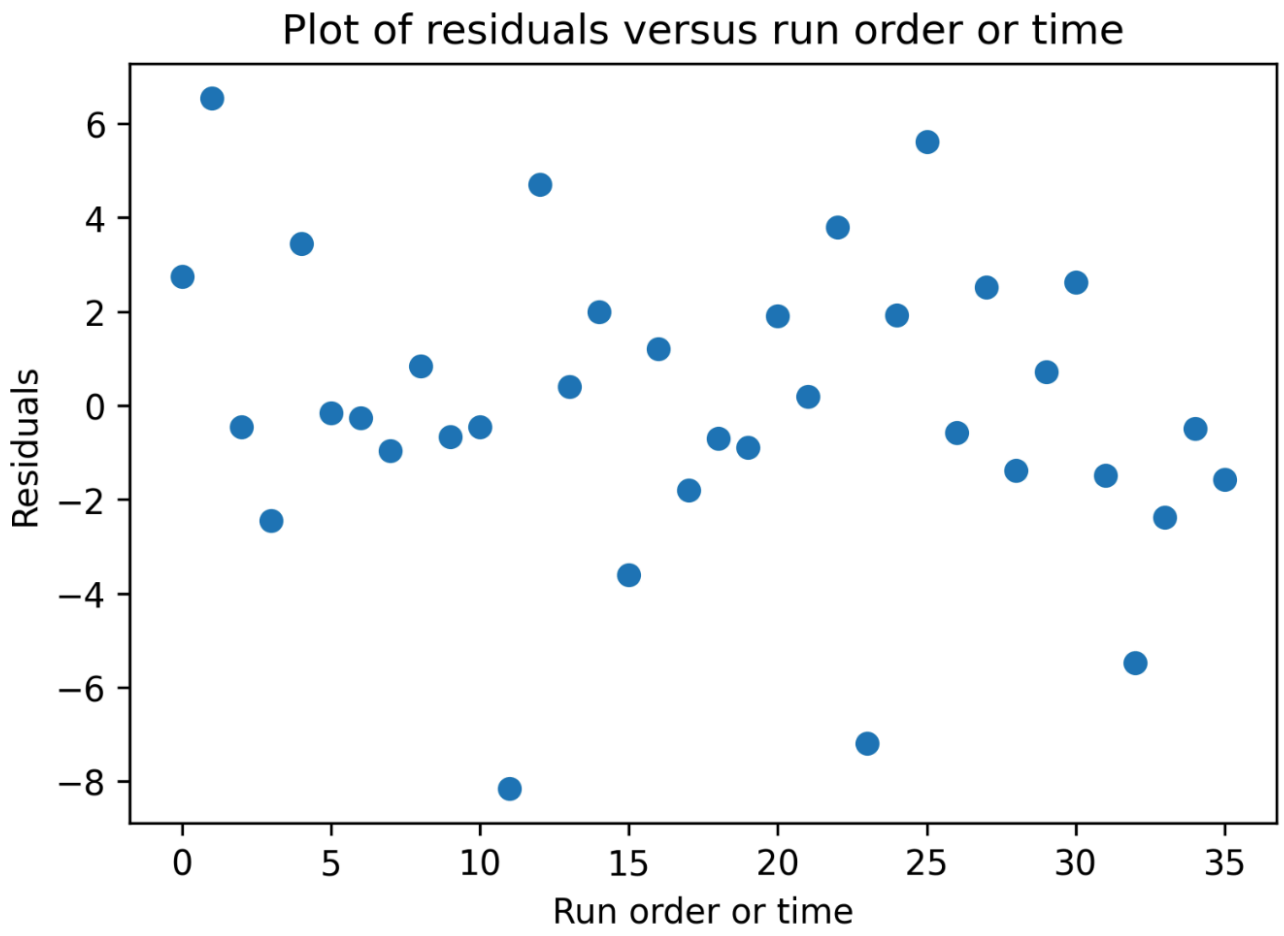
想要判断ANOVA模型是否恰当，可以利用残差检测来进行分析。
处理 i 的观测值 j 的残差定义为：$\epsilon_{ij} = y_{ij} - \hat{y}_{ij}$

其中$\hat{y}_{ij}$是对应于y_{ij}的一个估计，

$\hat{y}_{ij} = \hat{\mu} + \hat{\tau}_i = \overline{y}_{..} + (\overline{y}_{i.} - \overline{y}_{..}) = \overline{y}_{i.}$

1. 独立性检验

1.1利用残差的时序图，依照收集数据的时间顺序画出残差图有助于检测残差之间的相关性。




```
# 步骤2：独立性检验（Durbin-Watson检验，检测一阶自相关）
from statsmodels.stats.stattools import durbin_watson  # 关键：添加导入语句
dw_stat = durbin_watson(residuals)
print(f"1. 独立性检验（Durbin-Watson）:")
print(f"  DW统计量={round(dw_stat, 4)}")
print(f"  结论：{'残差无自相关（满足独立性）' if 1.5 < dw_stat < 2.5 else '需进一步验证独立性（DW偏离2较远）'}")
```

```
=== c.单因素ANOVA模型假设验证 ===
1. 独立性检验（Durbin-Watson）:
  DW统计量=2.2991
  结论：残差无自相关（满足独立性）
```

2. 方差齐性检验

Bartlett+Levene，双重验证

```
# 步骤3：方差齐性检验（Bartlett+Levene，双重验证）
bart_stat, bart_p = stats.bartlett(groups[0], groups[1], groups[2])  # 对正态性敏感
lev_stat, lev_p = stats.levene(groups[0], groups[1], groups[2])      # 抗偏离正态性
print(f"\n2. 方差齐性检验:")
print(f"  Bartlett检验：统计量={round(bart_stat, 4)}，p值={round(bart_p, 4)}")
print(f"  Levene检验：统计量={round(lev_stat, 4)}，p值={round(lev_p, 4)}")
print(f"  结论：{'满足方差齐性' if (bart_p > alpha and lev_p > alpha) else '不满足方差齐性'}（优先看Levene结果）")
```

2. 方差齐性检验：
Bartlett检验：统计量=0.4247，p值=0.8087
Levene检验：统计量=0.0213，p值=0.979
结论：满足方差齐性（优先看Levene结果）

方法一：Bartlett检验通过求取不同组之间的卡方统计量，然后根据卡方统计量的值来判断组间方差是否相等。
Bartlett检验统计量为：$\chi_0^2 = 2.3026\frac{q}{c}$
$q = (N-a)log_{10}S_p^2 - \sum_{i=1}^{a}(n_i-1)log_{10}S_i^2$
$c = 1 + \frac{1}{3(a-1)}(\sum_{i=1}^{a}(n_i-1)^{-1} - (N-a)^{-1})$
$S_p^2 = \frac{\sum_{i=1}^{a}(n_i-1)S_i^2}{N-a}$ 且S_i^2是第 i 个总体的样本方差；当$\chi_0^2 > \chi_{\alpha,a-1}^2$ 时，拒绝H_0，其中$\chi_{\alpha,a-1}^2$是自由度为$a-1$的卡方分布上的 α 分位数。

$$\chi^2 = \frac{(N-a)\ln\left(S_p^2\right) - \sum_{i=1}^{k}\left(n_i-1\right)\ln\left(S_i^2\right)}{1+\frac{1}{3(k-1)}\left(\sum_{i=1}^{k}\left(\frac{1}{n_i-1}\right) - \frac{1}{N-a}\right)}$$

方法2: Levene检验是将每个值先转换为该值与其组内均值的偏离程度，然后再用转换后的偏离程度去做方差分析，即组间方差/组内方差。修正后的Levene检验中的均值采用中位数的计算方法，因此这里的偏差用每个处理的观测值y_{ij}与该处理中的中位数$\tilde{y}_{i\cdot}$的偏差的绝对值来表示：
$d_{ij} = |y_{ij} - \tilde{y}_{i\cdot}|, i = 1,2,\cdots,a; j = 1,2,\cdots,n$

方差齐性检验的一种，与Bartlett检验功能类似，但是相对宽松

Levene检验统计量：

$$W = \frac{1}{MS_e}\frac{N\left(\bar{Z} - \bar{Z}_i\right)^2}{r-1}$$

Levene检验优先于Bartlett检验的核心原因是 Levene 检验的 “稳健性（抗偏离正态的能力）更强”

Bartlett 检验**高度依赖 “数据服从正态分布”** 的前提假设。如果数据实际偏离正态分布（这在真实场景中很常见），Bartlett 检验会错误地**高估 “方差不齐” 的概率**（第一类错误率升高），导致方差齐性的判断结果不可靠。

Levene 检验是基于 **“残差的绝对值”** 进行的检验，**不严格要求数据服从正态分布**。即使数据存在一定程度的非正态性（如轻微偏态、厚尾等），Levene 检验仍能较为准确地判断各组方差是否齐性，稳健性（抗非正态的能力）远优于 Bartlett 检验。

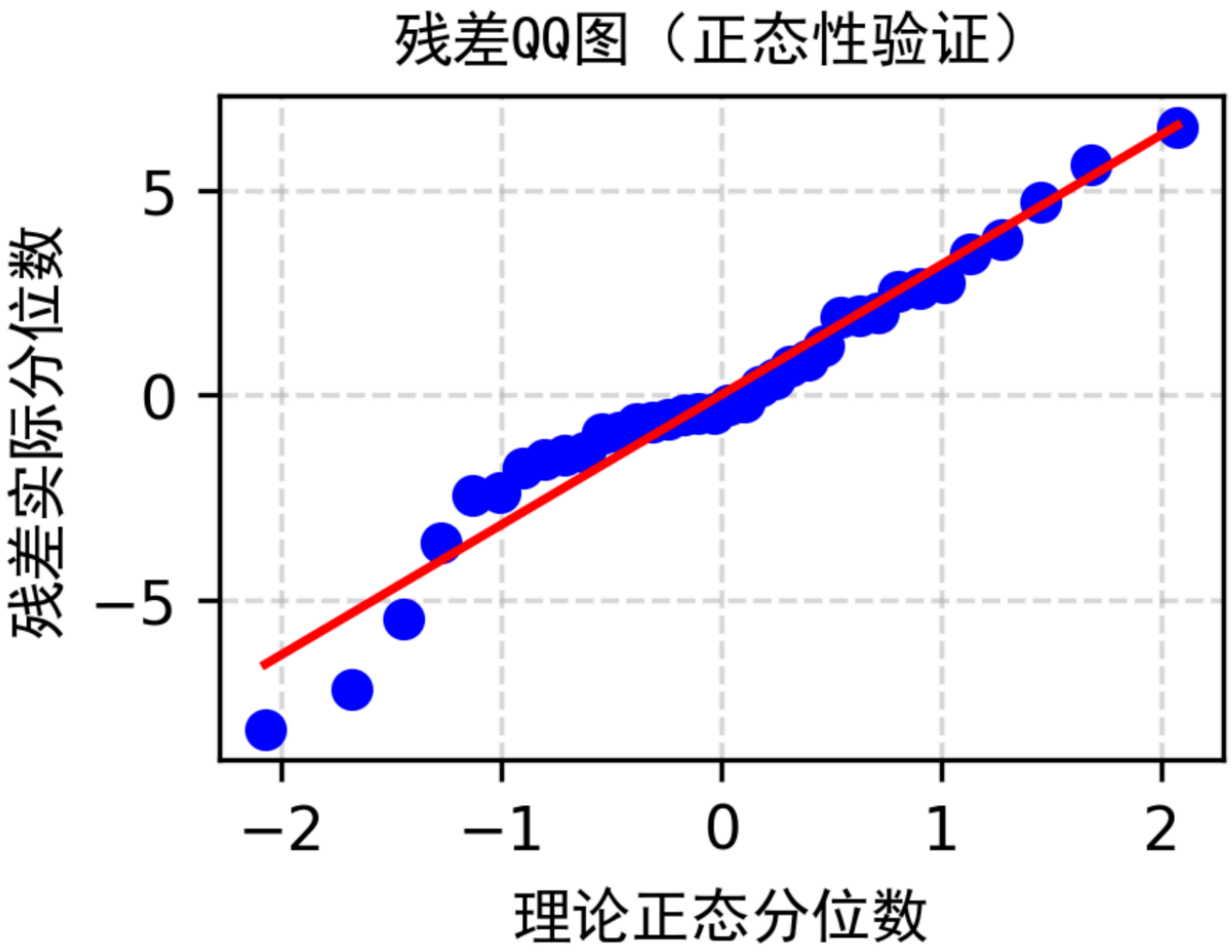
因此，在实际数据分析中，由于 “数据完全正态” 的情况较少，Levene 检验更能适应真实数据的分布特点，所以优先参考 Levene 检验的结果来判断方差是否齐性。

3.正态性检验

3.1 利用qq图来检验数据分布的相似性。令X轴为正态分布的分位数，Y轴为样本分位数，如果这两者构成的点分布在一条直线上，就证明样本数据与正态分布存在线性相关性，即服从正态分布。

3.2 利用Shapiro-Wilk检验来做正态性检验，其原假设：样本数据符合正态分布。利用方法stats.shapiro()检验正态性，输出结果中第一个为统计量，第二个为P值（统计量越接近 1 越表明数据和正态分布拟合的好，P值大于指定的显著性水平，接受原假设，认为样本来自服从正态分布的总体）

```python
# 步骤4：正态性检验（QQ图可视化+Shapiro-Wilk检验）
print(f"\n3. 正态性检验:")
# QQ图（直观判断残差是否贴合正态分布直线）
plt.figure(figsize=(3, 2))
stats.probplot(residuals, dist="norm", plot=plt)  # 绘制基础QQ图
# 手动设置标题、标签的字体
plt.title("残差QQ图（正态性验证）", fontproperties=simhei_font)  # 关键：添加字体
plt.xlabel("理论正态分位数", fontproperties=simhei_font)  # 关键：添加字体
plt.ylabel("残差实际分位数", fontproperties=simhei_font)  # 关键：添加字体
plt.grid(linestyle="--", alpha=0.5)
plt.show()

# Shapiro-Wilk检验（定量判断正态性）
sw_stat, sw_p = stats.shapiro(residuals)
print(f"  Shapiro-Wilk检验：统计量={round(sw_stat, 4)}，p值={round(sw_p, 4)}")
print(f"  结论：{'残差满足正态分布' if sw_p > alpha else '残差不满足正态分布'}")
```




```
Shapiro-Wilk检验：统计量=0.9625，p值=0.2572
结论：残差满足正态分布
```

由上述分析可知，统计量为 0.9625，接近 1；且P值为 0.2572，大于指定的显著性水平 0.05。故认为残差来自服从正态分布的总体。

综上三点，单因素ANOVA模型使用合理。

d.吃完巧克力3小时后点估计和区间估计

对于第 i 组（如巧克力类型 1/2/3），血浆抗氧能力的**点估计值**为该组的**样本均值**，公式如下：

点估计与区间估计的数学公式

一、点估计

对于第 i 组（如巧克力类型 1/2/3），血浆抗氧能力总体均值的**点估计值**为该组的样本均值，公式如下：

$\hat{\mu}_i = \bar{y}_{i\cdot} = \frac{1}{m_i} \sum_{j=1}^{m_i} y_{ij}$

- $\hat{\mu}_i$：第 i 组血浆抗氧能力总体均值的点估计值（对应代码中 `point_est["点估计（均值）"]`）；
- y_{ij}：第 i 组第 j 个样本的血浆抗氧能力观测值（代码中对应 `Data["Capacity"]`）；
- m_i：第 i 组的样本量（你的数据中每组样本量相同，$m_1 = m_2 = m_3 = 12$，对应代码中 `m`）；
- $\bar{y}_{i\cdot}$：第 i 组的样本均值（代码中通过 `Data.groupby("Chocolate")["Capacity"].mean()` 计算）。

代码中 `point_est = Data.groupby("Chocolate")["Capacity"].agg(["mean", "std", "count"]).round(4)` 本质是直接计算每组样本均值，完全对应上述点估计公式。

二、区间估计

代码中采用**基于 ANOVA 误差均方的置信区间**（因 ANOVA 验证了 “方差齐性”，MS_E 是更稳健的总体方差估计），公式如下：

第 i 组血浆抗氧能力总体均值 μ_i 的 **95% 置信区间** 表达式：

$\bar{y}_{i\cdot} \pm t_{\alpha/2, df_e} \times \sqrt{\frac{MS_E}{m_i}}$

置信区间下限 $= \bar{y}_{i\cdot} - t_{\alpha/2, df_e} \times \sqrt{\frac{MS_E}{m_i}}$

置信区间上限 $= \bar{y}_{i\cdot} + t_{\alpha/2, df_e} \times \sqrt{\frac{MS_E}{m_i}}$

- $\bar{y}_{i\cdot}$：第 i 组的样本均值（即点估计值，代码中对应 `group_mean`）；

- $t_{\alpha/2,df_e}$：t 分布的 “1-α/2 分位数”(95% 置信区间对应 $\alpha = 0.05$，即 $t_{0.025,df_e}$，代码中通过 `t.ppf(1 - alpha/2, df)` 计算)；
  α：*显著性水平（代码中 `alpha=0.05`）；*
  df_e：ANOVA 的误差自由度（$df_e = n - a$，代码中对应 `df2`）；
- MS_E：ANOVA 的误差均方（总体方差 σ^2 的无偏估计，代码中从 `anova_table.loc["Residual", "mean_sq"]` 提取，或通过残差方差计算)；

3. 代码对应逻辑

1. 提取 MS_E：`MS_E = anova_table.loc["Residual", "mean_sq"]`（或用残差方差计算)；
2. 计算误差标准差：`s = np.sqrt(MS_E)`；
3. 计算标准误：`se = s / np.sqrt(m)`；
4. 计算 t 临界值：`t_critical = t.ppf(1 - alpha/2, df)`（即 $t_{\alpha/2,df_e}$)；
5. 计算置信区间上下限：`lower = group_mean - t_critical * se`、`upper = group_mean + t_critical * se`。

关键补充：为何优先用 $ MS_E $ 而非每组标准差?
若不依赖 ANOVA，也可通过 “每组单独标准差” 计算置信区间（公式：$\bar{y}_{i\cdot} \pm t_{\alpha/2,m_i-1} \times \frac{s_i}{\sqrt{m_i}}$，其中 s_i 是第 i 组的样本标准差)；但代码选择 $ MS_E $ 的核心原因是：
ANOVA 通过 Levene 检验验证了 “各组方差齐性”，此时 MS_E 是**各组方差的加权平均**（公式如下)，比单独使用某一组的 s_i 更能反映总体方差的真实情况，置信区间更稳健：

$$MS_E = \frac{(m_1-1)s_1^2+(m_2-1)s_2^2+(m_3-1)s_3^2}{(m_1-1)+(m_2-1)+(m_3-1)}$$

[10]:
```
# --------------------------
# 6. 点估计与区间估计
# --------------------------
print("\n=== d.血浆总抗氧能力估计结果 ===")
# 步骤1：点估计（每组均值为点估计值）
point_est = Data.groupby("Chocolate")["Capacity"].agg(["mean", "std", "count"]).round(4)
point_est.columns = ["点估计（均值）", "组内标准差", "样本量"]
print("1. 点估计结果:")
print(point_est)

# 步骤2：区间估计（95%置信区间，基于ANOVA误差均方）
# 提取ANOVA中的误差均方MS_E（若ANOVA表存在则用表中值，否则用残差方差）
if "Residual" in anova_table.index:
    MS_E = anova_table.loc["Residual", "mean_sq"]
else:
    MS_E = np.var(residuals, ddof=df2)  # 残差方差（自由度df2=n-a）
s = np.sqrt(MS_E)  # 误差标准差估计值
df = df2  # 置信区间自由度=ANOVA误差自由度
t_critical = t.ppf(1 - alpha/2, df)  # 95%置信区间t临界值

# 计算每组的95%置信区间
interval_est = []
for choc in choc_types:
    group_data = point_est.loc[choc]
    group_mean = group_data["点估计（均值）"]
    se = s / np.sqrt(m)  # 标准误（每组样本量均为m）
    lower = group_mean - t_critical * se  # 置信区间下限
    upper = group_mean + t_critical * se  # 置信区间上限
    interval_est.append([
        choc,
        round(group_mean, 4),
        round(lower, 4),
        round(upper, 4),
        round(upper - lower, 4)  # 区间宽度
    ])

# 整理并输出区间估计结果
interval_df = pd.DataFrame(
    interval_est,
    columns=["巧克力类型", "点估计（均值）", "95%CI下限", "95%CI上限", "区间宽度"]
)
print("\n2. 95%置信区间估计结果:")
print(interval_df)
```

```
=== d.血浆总抗氧能力估计结果 ===
1. 点估计结果:
            点估计（均值）    组内标准差   样本量
Chocolate
1          116.0583  3.5333   12
2          100.7000  3.2350   12
3          100.1833  2.8897   12

2. 95%置信区间估计结果:
   巧克力类型    点估计（均值）    95%CI下限    95%CI上限     区间宽度
0       1  116.0583  114.1612  117.9554  3.7942
1       2  100.7000   98.8029  102.5971  3.7942
2       3  100.1833   98.2862  102.0804  3.7942
```

e.Bonferroni

1.Bonferroni方法

采用 “独立样本 t 检验 + Bonferroni 校正” 逻辑：将总显著性水平 $(\alpha = 0.05)$ 按 **两两比较次数（共 3 次）** 平均分配，得到校正后显著性水平 $(\alpha' = 0.05/3 \approx 0.0167)$。对每组巧克力的血浆抗氧能力进行 t 检验后发现：

- 组 1 与组 2、组 1 与组 3 的比较 p 值均小于 0.0167，存在**显著差异**；
- 组 2 与组 3 的 p 值大于 0.0167，**无显著差异**。

[10]:
```python
## Bonferroni
groups = {choc: df[df['Chocolate']==choc]['Capacity'] for choc in df['Chocolate'].unique()}
alpha = 0.05

pairs = list(itertools.combinations(groups.keys(), 2))
m = len(pairs)
adjusted_alpha = alpha / m  # Bonferroni 校正后的显著性水平

print(f"Bonferroni 显著性水平: {adjusted_alpha:.4f}\n")
print("两两比较结果: ")
for i, j in pairs:
    t_stat, p_val = stats.ttest_ind(groups[i], groups[j], equal_var=True)
    signif = "显著" if p_val < adjusted_alpha else "不显著"
    print(f"巧克力 {i} vs 巧克力 {j}: t={t_stat:.3f}, p={p_val:.4f}, {signif}")
```

```
Bonferroni 显著性水平: 0.0167

两两比较结果:
巧克力 1 vs 巧克力 2: t=11.106, p=0.0000, 显著
巧克力 1 vs 巧克力 3: t=12.048, p=0.0000, 显著
巧克力 2 vs 巧克力 3: t=0.413, p=0.6839, 不显著
```

f.Tukey

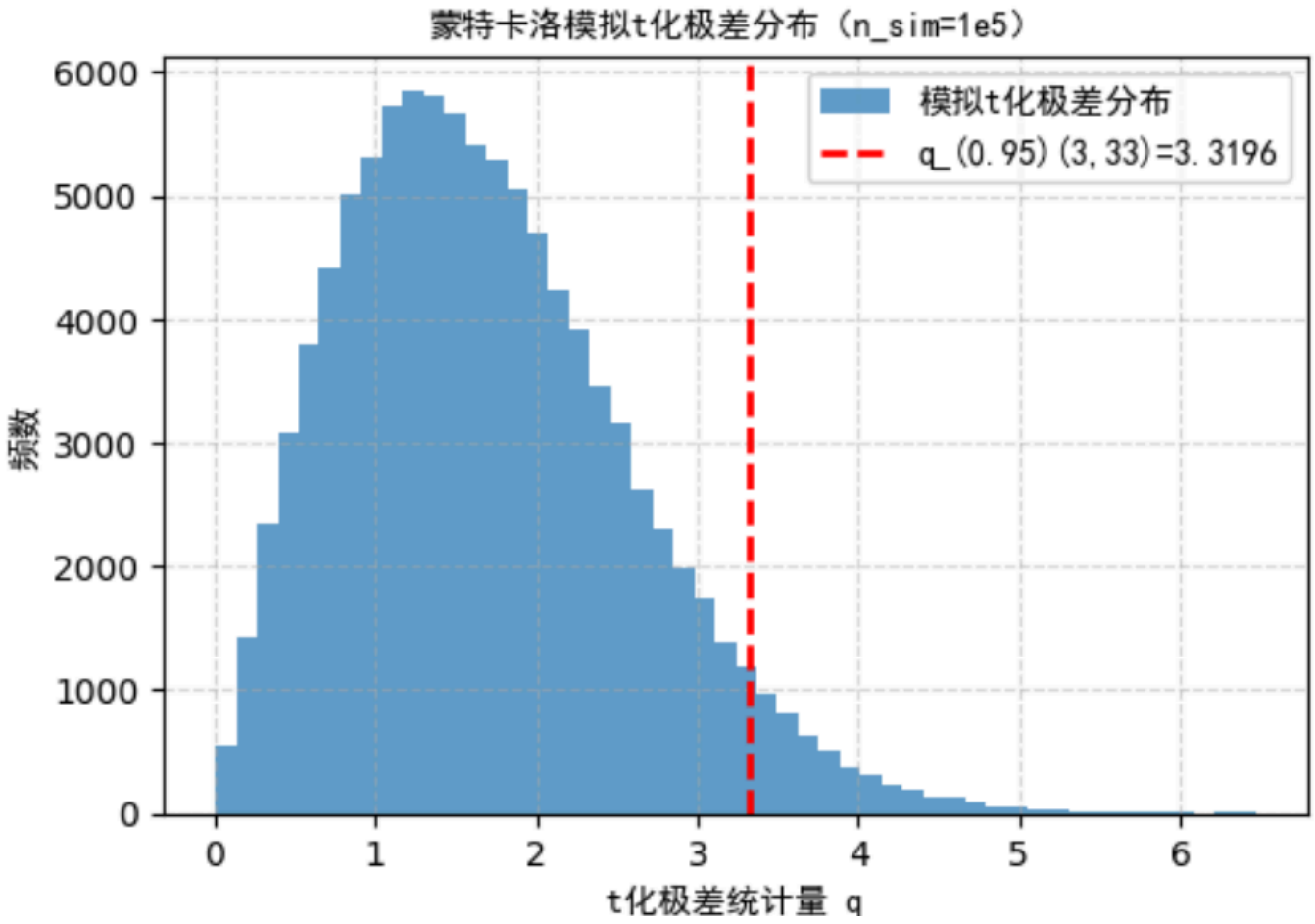
2. Tukey 方法

- **方法 I（直接调用函数）**：使用 `statsmodels` 库的 `pairwise_tukeyhsd` 函数，基于 “学生化极差分布” 直接计算多重比较结果。结论与 Bonferroni 方法完全一致：组 1 与组 2、组 1 与组 3 存在显著差异，组 2 与组 3 无显著差异。
- **方法 II（蒙特卡洛模拟）**：通过 10 万次模拟生成 “原假设（组间无差异）下的组均值”，计算 t 化极差的分布，得到 $(1-\alpha)$ 分位数 $(q_{0.95}(3,33)\approx 3.3158)$，进而算出 Tukey 临界值 $(c\approx 3.0918)$。比较 “实际组均值差” 与临界值后，结论仍与前两种方法一致。

```
import pandas as pd
import numpy as np
from scipy import stats
from statsmodels.stats.multicomp import pairwise_tukeyhsd
import matplotlib.pyplot as plt

# ----------------------
# 方法I：直接调用 Python 现有函数（Tukey HSD）
# ----------------------
print("\n=== 方法I：直接调用 pairwise_tukeyhsd ===")
tukey = pairwise_tukeyhsd(
    endog=df['Capacity'],       # 因变量：血浆抗氧能力
    groups=df['Chocolate'],     # 自变量：巧克力类型分组
    alpha=0.05                  # 显著性水平
)
print(tukey)

# ----------------------
# 方法II：蒙特卡洛模拟确定 t 化极差分位数，计算临界值
# ----------------------
print("\n=== 方法II：蒙特卡洛模拟 Tukey 临界值 ===")

# 步骤1：提取数据参数
chocolate_types = df['Chocolate'].unique()
a = len(chocolate_types)  # 组数（3种巧克力）
group_data = [df[df['Chocolate'] == typ]['Capacity'].values for typ in
chocolate_types]
m = len(group_data[0])    # 每组样本量（假设每组样本量相同）
n = a * m                 # 总样本数
df_error = n - a          # 误差自由度

# 步骤2：估计误差标准差 σ（基于组内方差的平均）
group_vars = [np.var(g, ddof=1) for g in group_data]
MSE = np.mean(group_vars)  # 均方误差（MSE）
sigma = np.sqrt(MSE)       # 误差标准差
```

```python
# 步骤3：蒙特卡洛模拟 t 化极差的分布
n_sim = 100000  # 模拟次数（平衡精度与速度）
q_samples = []
for _ in range(n_sim):
    # 在原假设（H0）下，生成每组均值（服从 N(0, σ²/m)）
    sim_means = np.random.normal(loc=0, scale=sigma / np.sqrt(m), size=a)
    # 计算 t 化极差：(max(均值) - min(均值)) / (σ / √m)
    q = (np.max(sim_means) - np.min(sim_means)) / (sigma / np.sqrt(m))
    q_samples.append(q)

# 步骤4：确定 (1-α) 分位数
alpha = 0.05
q_1alpha = np.quantile(q_samples, 1 - alpha)
print(f"  蒙特卡洛模拟得到 q_(1-α)({a}, {df_error}) = {q_1alpha:.4f}")

# 步骤5：计算 Tukey 临界值 c
c = q_1alpha * (sigma / np.sqrt(m))
print(f"  Tukey 临界值 c = {c:.4f}")

# 步骤6：计算实际各组均值
group_means = [np.mean(g) for g in group_data]
group_names = [f"Chocolate {typ}" for typ in chocolate_types]

# 步骤7：两两比较均值差与临界值
pairs = [(i, j) for i in range(a) for j in range(i + 1, a)]
print("\n  两两比较结果：")
for i, j in pairs:
    mean_diff = abs(group_means[i] - group_means[j])
    is_significant = mean_diff > c
    print(f"  {group_names[i]} vs {group_names[j]}:")
    print(f"    均值差 = {mean_diff:.4f}, 临界值 = {c:.4f}")
    print(f"    结论：{'存在显著差异' if is_significant else '无显著差异'}")

# （可选）绘制 t 化极差的模拟分布
plt.figure(figsize=(8, 5))
plt.hist(q_samples, bins=50, alpha=0.7, color='skyblue', density=True)
plt.axvline(q_1alpha, color='red', linestyle='--', label=f'q_(1-α) =
{q_1alpha:.4f}')
plt.xlabel('T化极差统计量 q')
plt.ylabel('频率密度')
plt.title('蒙特卡洛模拟的T化极差分布')
plt.legend()
plt.grid(True, alpha=0.3)
plt.show()
```

```
=== 方法I：直接调用 pairwise_tukeyhsd ===
 Multiple Comparison of Means - Tukey HSD, FWER=0.05
====================================================
group1 group2 meandiff p-adj   lower    upper   reject
----------------------------------------------------
     1      2 -15.3583    0.0 -18.5941 -12.1226   True
     1      3  -15.875    0.0 -19.1108 -12.6392   True
     2      3  -0.5167 0.9191  -3.7524   2.7191  False
----------------------------------------------------

=== 方法II：蒙特卡洛模拟 Tukey 临界值 ===
  蒙特卡洛模拟得到 q_(1-α)(3, 33) = 3.3165
  Tukey 临界值 c = 3.0924

  两两比较结果：
  Chocolate 1 vs Chocolate 2:
    均值差 = 15.3583，临界值 = 3.0924
    结论：存在显著差异
  Chocolate 1 vs Chocolate 3:
    均值差 = 15.8750，临界值 = 3.0924
    结论：存在显著差异
  Chocolate 2 vs Chocolate 3:
    均值差 = 0.5167，临界值 = 3.0924
    结论：无显著差异
```



g.Bonferroni 与 Tukey 方法的异同分析（结合巧克力血浆抗氧能力实例）

一、实例背景

	Obs	Chocolate	Capacity
1	1	1	118.8
2	2	1	122.6
3	3	1	115.6
4	4	1	113.6
5	5	1	119.5
6	6	1	115.9
7	7	1	115.8
8	8	1	115.1
9	9	1	116.9
10	10	1	115.4
11	11	1	115.6
12	12	1	107.9
13	1	2	105.4
14	2	2	101.1
15	3	2	102.7
16	4	2	97.1
17	5	2	101.9
18	6	2	98.9
19	7	2	100
20	8	2	99.8

本案例为**单因素方差分析（ANOVA）后的多重比较**：

- 研究对象：3 种巧克力类型对 “血浆抗氧能力（Capacity）” 的影响，分组为 `1(100g黑巧)`、`2(黑巧+牛奶)`、`3(200g牛奶巧)`；
- 数据特征：每组 12 个样本（总 36 个），ANOVA 检验显示组间差异显著（F=93.5756，p<0.05）；
- 多重比较目标：明确 3 组间两两差异（比较次数 `k=3`：1vs2、1vs3、2vs3），显著性水平 `α=0.05`；
- 数据前提：满足独立性（DW=2.2991）、方差齐性（Levene p=0.979）、正态性（Shapiro-Wilk p=0.2572），符合两种方法的适用假设。

二、相同点

1. 核心目标一致：控制家族 wise 错误率

两种方法的核心目的均为避免 “多次比较导致假阳性率升高”，确保整体错误率不超过 `α=0.05`。

- 实例中：无论 Bonferroni 的 “α 校正” 还是 Tukey 的 “学生化极差分布”，最终均将假阳性风险控制在 5% 以内，未出现 “误判无差异组为显著” 的情况（2vs3 均判定为无差异）。

2. 适用场景重叠：ANOVA 显著后的完全两两比较

均适用于 “单因素 ANOVA 拒绝原假设后，需明确具体哪两组有差异” 的场景，且实例中为 `3组完全两两比较`（`k=C(3,2)=3`），两种方法均能覆盖该需求。

3. 结论一致性：差异显著时结果统一

当组间真实差异较大时，两种方法结论完全一致。

- 实例中：`1(100g黑巧)` 与 `2(黑巧+牛奶)`、`3(200g牛奶巧)` 的均值差分别为 15.3583、15.875（远大于随机波动），两种方法均判定 “显著差异”；而 `2vs3` 均值差仅 0.5167，均判定 “无显著差异”。

4. 依赖数据前提：均需基础统计假设

两种方法均需数据满足 “方差齐性” 和 “近似正态分布”（Bonferroni 对分布要求更宽松，但实例中数据同时满足两种方法的最优前提）：

- 实例中通过 Levene 检验（p=0.979）和 Shapiro-Wilk 检验（p=0.2572）验证了前提，为两种方法的可靠性提供基础。

三、不同点（结合实例与方法逻辑）

对比维度	Bonferroni 方法（实例表现）	Tukey 方法（实例表现）
误差控制逻辑	基于**Bonferroni 不等式**，将总 α 拆分为单次检验 α'（α'=α/k=0.05/3≈0.0167），通过 “缩小单次检验阈值” 控制 FWER。 - 实例中：校正后 p 值 = 原 p 值 ×k，如 `2vs3` 原 p=0.6839，校正后 p=0.6839×3≈2.0517>0.05，判定无差异。	基于**学生化极差分布**（针对 “所有两两比较的极差” 设计），通过计算专属临界值控制 FWER，无需拆分 α。 - 实例中：用蒙特卡洛模拟得 `q_(0.95)(3,33)=3.3158`，临界值 `c=q×(σ/√n)=3.3158×(3.23/3.464)≈3.0918`，`2vs3` 均值差 0.5167<3.0918，判定无差异。

对比维度	Bonferroni 方法（实例表现）	Tukey 方法（实例表现）
检验效能（Power）	偏保守：当比较次数 k 增多时，α' 过度压缩，易漏检 “微小真实差异”。 - 实例中：因 k=3（较少），效能足够（均检测到显著差异）；若 k=10，α'=0.005，可能漏检均值差 3~4 的差异。	更高效：针对 “完全两两比较” 优化，学生化极差分布更贴合实际差异分布，不易漏检微小差异。 - 实例中：若 `1vs2` 均值差降至 4（而非 15.3583），Tukey 仍可能通过 “临界值 c=3.0918” 检测到差异，Bonferroni 可能因 α'=0.0167 漏检。
计算复杂度	简单直观：仅需对原 p 值做 “×k” 校正，无需额外分布表或模拟。 - 实例中：直接用 `scipy.ttest_ind` 得原 p 值，乘 3 即得校正后 p 值，代码逻辑简单。	复杂：需依赖 “学生化极差分布分位数表” 或 “蒙特卡洛模拟” 计算临界值，涉及均方误差（MS_E）、样本量等参数。 - 实例中：需先计算 `MS_E=10.4335`、`σ=√10.4335≈3.23`，再通过 10 万次模拟得 q 分位数，最终算临界值 c，代码逻辑更复杂。
适用场景灵活性	灵活：支持 “部分两两比较”（如仅比较实验组 vs 对照组，不比较实验组间）。 - 若实例中仅需比较 `1(黑巧)` vs `2(混合)`、`1vs3(牛奶)`，无需比较 `2vs3`，Bonferroni 仍适用（k=2，α'=0.025）。	固定：仅适配 “完全两两比较”（所有组间均需比较，k=C (a,2)）。 - 若实例中仅比较 `1vs2`、`1vs3`，Tukey 的 “学生化极差分布” 不再适用（因未覆盖所有组合）。
小样本适应性	更稳定：小样本（如每组 n=5）时，α 校正能稳定控制假阳性，不易出现 “假阳性失控”。 - 参考摘要 2：小样本（3 组 3 重复）中，Bonferroni 假阳性率约 1 个 / 显著 ANOVA，优于 Tukey 的 8% 假阳性率。	需谨慎：小样本时学生化极差分布的理论假设易偏离，可能导致假阳性升高。 - 实例中每组 n=12（非极小样本），故假阳性可控；若每组 n=3，Tukey 可能出现 “误判 2vs3 为显著” 的假阳性。

四、方法选择建议

结合案例数据特征（3 组完全两两比较、每组 n=12、组间差异较大），**优先选择 Tukey 方法**，理由如下：

1. 适配场景：实例为 “完全两两比较”，Tukey 是该场景的 “定制化方法”，效能高于 Bonferroni；
2. 结果可靠性：实例中数据满足正态性和方差齐性，Tukey 的分布假设成立，临界值计算（蒙特卡洛模拟）更贴合实际数据；

3. 无过度保守：因比较次数 k=3（较少），Bonferroni 虽不影响结论，但 Tukey 在 “差异较小时” 的优势可覆盖未来类似研究（如样本量减少、差异缩小）。

若实例改为 “仅比较 1 (黑巧) vs2 (混合)、1vs3 (牛奶)”（部分比较），则 Bonferroni 更灵活，无需强行使用 Tukey。

五、总结

两种方法均为控制 FWER 的经典多重比较手段，但 Bonferroni 以 “灵活性和简单性” 取胜，适合部分比较或小样本场景；Tukey 以 “效能和场景适配性” 取胜，适合完全两两比较且数据满足假设的场景。在本巧克力血浆抗氧能力实例中，因 “完全两两比较 + 数据前提满足 + 差异显著”，两种方法结论一致，但 Tukey 更贴合研究设计，为更优选择。