

# 衍生工具模型

## 作业-No.4

姓名：叶云鹏

学号：2501210025

2025-10-05

1. 证明 BS 解在离散对冲最小二乘损失函数下的最优性。分析累积对冲误差的时间序列性质。

证明. (a) **BS 解最优性证明**

期权在  $t = 0$  卖出，收入期权费  $c_0$ ，到期支付  $Z = \max(0, S_T - K)$ 。对冲组合在  $t = kh$  ( $h = T/n, k = 0, 1, \dots, n$ ) 定义为  $\Pi(k) = \delta_k S_{kh} - B_k$ ，其中  $\delta_k$  是标的资产持仓， $B_k = (\delta_k - \delta_{k-1})S_{kh} + e^{rh} B_{k-1}$ 。对冲误差为：

$$\text{PL}(k) = \Pi(k_-) - c_k = \delta_{k-1} S_{kh} - e^{rh} B_{k-1} - c_k, \quad k = 1, \dots, n-1$$

到期误差为  $\text{PL}(n) = \Pi(n_-) - Z = \delta_{n-1} S_T - e^{rh} B_{n-1} - \max(0, S_T - K)$ 。目标是证明 BS delta  $\delta_k^{BS} = \frac{\partial c_k}{\partial S_{kh}}$  最小化单步均方对冲误差  $\mathbb{E}[(\text{PL}(k))^2]$ 。

在离散时间，单步对冲误差为：

$$\text{PL}(k) = \delta_{k-1} S_{kh} - e^{rh} B_{k-1} - c_k$$

其中  $\Pi(k_-) = \delta_{k-1} S_{kh} - e^{rh} B_{k-1}$ ， $c_k$  是  $t = kh$  的期权价值。目标是最小化：

$$\mathbb{E}[(\text{PL}(k))^2 | \mathcal{F}_{(k-1)h}] = \mathbb{E}[(\delta_{k-1} S_{kh} - e^{rh} B_{k-1} - c_k)^2 | \mathcal{F}_{(k-1)h}]$$

令  $\Delta S_k = S_{kh} - S_{(k-1)h}$ ，则：

$$\text{PL}(k) = \delta_{k-1} (S_{(k-1)h} + \Delta S_k) - e^{rh} B_{k-1} - c_k$$

因为  $S_{(k-1)h}$ 、 $B_{k-1}$  在  $\mathcal{F}_{(k-1)h}$  中已知，优化  $\delta_{k-1}$  等价于最小化：

$$\mathbb{E}[(\delta_{k-1} \Delta S_k - (c_k - e^{rh} B_{k-1} - \delta_{k-1} S_{(k-1)h}))^2 | \mathcal{F}_{(k-1)h}]$$

令  $Y = c_k - e^{rh} B_{k-1} - \delta_{k-1} S_{(k-1)h}$ ，这是一个线性回归问题，最优  $\delta_{k-1}$  为：

$$\delta_{k-1}^* = \frac{\text{Cov}(\Delta S_k, Y)}{\text{Var}(\Delta S_k)} = \frac{\mathbb{E}[\Delta S_k (c_k - e^{rh} B_{k-1} - \delta_{k-1} S_{(k-1)h})]}{\mathbb{E}[(\Delta S_k)^2]}$$

在 BS 模型下， $S_t$  遵循几何布朗运动，期权价值  $c_k$  满足 BS PDE：

$$\frac{\partial c}{\partial t} + rS \frac{\partial c}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 c}{\partial S^2} = rc$$

对于小  $h$ ，由泰勒展开：

$$c_k \approx c_{k-1} + \frac{\partial c_{k-1}}{\partial S} \Delta S_k + \frac{\partial c_{k-1}}{\partial t} h + \frac{1}{2} \frac{\partial^2 c_{k-1}}{\partial S^2} (\Delta S_k)^2$$

取条件期望，忽略高阶项（因为  $\mathbb{E}[(\Delta S_k)^2] \approx \sigma^2 S_{(k-1)h}^2 h$ ）：

$$\mathbb{E}[c_k | \mathcal{F}_{(k-1)h}] \approx c_{k-1} + \left( \frac{\partial c_{k-1}}{\partial t} + \frac{1}{2} \sigma^2 S_{(k-1)h}^2 \frac{\partial^2 c_{k-1}}{\partial S^2} \right) h$$

由 BS PDE,  $\frac{\partial c_{k-1}}{\partial t} + \frac{1}{2} \sigma^2 S_{(k-1)h}^2 \frac{\partial^2 c_{k-1}}{\partial S^2} = r c_{k-1} - r S_{(k-1)h} \frac{\partial c_{k-1}}{\partial S}$ ，代入：

$$\mathbb{E}[c_k - c_{k-1} | \mathcal{F}_{(k-1)h}] \approx \frac{\partial c_{k-1}}{\partial S} \mathbb{E}[\Delta S_k] + \left( r c_{k-1} - r S_{(k-1)h} \frac{\partial c_{k-1}}{\partial S} \right) h$$

在风险中性测度下， $\mathbb{E}[\Delta S_k] \approx r S_{(k-1)h} h$ ，所以：

$$\mathbb{E}[c_k - c_{k-1}] \approx \frac{\partial c_{k-1}}{\partial S} \cdot r S_{(k-1)h} h + \left( r c_{k-1} - r S_{(k-1)h} \frac{\partial c_{k-1}}{\partial S} \right) h = r c_{k-1} h$$

因此， $\text{Cov}(\Delta S_k, c_k) \approx \frac{\partial c_{k-1}}{\partial S} \text{Var}(\Delta S_k)$ ，代入：

$$\delta_{k-1}^* \approx \frac{\partial c_{k-1}}{\partial S} = \delta_{k-1}^{BS}$$

证明完成

### (b) 累计对冲误差性质

对于累积对冲误差，定义  $L_t^n = \sum_{k=1}^{\lfloor t/h \rfloor} \text{PL}(k)$ 。我们有：

$$\text{PL}(k) = \delta_{k-1} S_{kh} - \left( e^{rkh} B_0 + \sum_{j=1}^{k-1} e^{r(k-j)h} (\delta_j - \delta_{j-1}) S_{jh} \right) - c_k$$

时间序列性质：

- **期望**：在风险中性测度下， $\mathbb{E}[\text{PL}(k)] \approx 0$ ，因为  $\delta_{k-1}^{BS}$  使  $\mathbb{E}[\Pi(k) - c_k] \approx 0$ （由 BS 模型的期望匹配）。累积误差  $\mathbb{E}[L_t^n] \approx 0$ 。
- **方差**： $\text{Var}(\text{PL}(k)) \approx \mathbb{E}[(\delta_{k-1}^{BS} \Delta S_k - (c_k - c_{k-1}))^2]$ ，主要由  $\Gamma = \frac{\partial^2 c}{\partial S^2}$  驱动， $\text{Var}(L_t^n) \propto \frac{t}{n} \sigma^2 \int_0^t S_u^2 \Gamma_u^2 du$ 。
- **分布**：单步误差近似正态（由  $\Delta S_k$  的正态性），累积误差为和的分布，近似正态。
- **自相关**：因为  $\Delta S_k$  在不同  $k$  独立， $\text{PL}(k)$  之间近似无相关。

□

2. 期权有效期内的某个时间点（不是初始时刻和到期时刻），你已经知道标的资产价格的历史走势，这种数据不一定来自风险中性概率分布，试给出利用标的资产价格的历史观测数据进行对冲的方法，比较与 BS 方法的对冲效果的差异。

证明. 在  $t = kh$  ( $0 < t < T$ ,  $h = T/n$ )，给定历史价格  $S_0, S_h, \dots, S_{kh}$ ，可使用最小方差对冲方法，最小化组合价值变化方差。，对冲组合为  $\Pi(k) = \delta_k S_{kh} - B_k$ ，误差为  $\text{PL}(k) = \delta_{k-1} S_{kh} - e^{rh} B_{k-1} - c_k$ 。

**方法步骤：**

- (a) 从历史数据计算波动率  $\sigma_{hist}$ ，如滚动窗口标准差： $\sigma_{hist} = \sqrt{\frac{1}{m} \sum_{j=k-m}^{k-1} \left( \frac{S_{jh} - S_{(j-1)h}}{S_{(j-1)h}} \right)^2} / h$ 。
- (b) 计算 BS delta： $\delta_k^{BS} = \Phi(d_1)$ ，其中  $d_1 = \frac{\ln(S_{kh}/K) + (r + \sigma_{hist}^2/2)(T - kh)}{\sigma_{hist} \sqrt{T - kh}}$ 。
- (c) 估计价格-波动率相关  $\rho = \text{Corr} \left( \frac{\Delta S_{jh}}{S_{(j-1)h}}, \Delta \sigma_{jh} \right)$ ，其中  $\Delta \sigma_{jh}$  从历史波动率序列计算。

(d) 调整 delta 为最小方差 delta:  $\delta_k^{MV} = \delta_k^{BS} + v_k^{BS} \cdot \frac{\rho \sigma_{\Delta\sigma}}{\sigma_{S_{kh}}}$ , 其中  $v_k^{BS} = \frac{\partial c_k}{\partial \sigma}$ ,  $\sigma_{\Delta\sigma}$  和  $\sigma_{S_{kh}}$  分别为波动率变化和价格变化的标准差。

(e) 构造组合:  $\Pi(k) = \delta_k^{MV} S_{kh} - B_k$ , 其中  $B_k = (\delta_k^{MV} - \delta_{k-1}^{MV}) S_{kh} + e^{rh} B_{k-1}$ 。

与 BS 方法的比较:

- **相似性**: 两者均基于 delta 对冲,  $\delta_k^{MV} \approx \delta_k^{BS}$  当  $\rho \approx 0$  或波动率近似常数。
- **差异**: BS 方法假设风险中性测度下的常数波动率, 忽略  $\Delta S$  和  $\Delta\sigma$  的相关性, 导致  $PL(k)$  方差较高, 尤其在杠杆效应 (负相关) 市场。最小方差方法利用历史数据捕捉  $\rho$ , 调整 delta, 减少  $\text{Var}(PL(k))$

□

3. 分析在 BS 框架下考虑对冲成本 (交易量的固定百分比) 的离散对冲误差的敏感性分析: 1) 对冲频率; 2) IV; 3) 未到期期限。

证明.

1) 对冲频率: 高频率 (小  $h = T/n$ ) 减少  $PL(k)$  的波动 (因  $\delta_{k-1} \approx \frac{\partial c_k}{\partial S}$  更精确), 但增加交易次数, 成本累积为  $k \sum |\delta_k - \delta_{k-1}| S_{kh}$ 。

2) IV (隐含波动率): 高 IV 增加  $\Gamma = \frac{\partial^2 c}{\partial S^2}$ , 使  $\delta_k$  变化剧烈, 放大  $PL(k)$  和成本  $k \cdot |\delta_k - \delta_{k-1}| S_{kh}$ 。Leland 方法调整  $\sigma_m = \sigma \sqrt{1 + \sqrt{2/\pi} k / (\sigma \sqrt{h})}$ , 减少高 IV 下误差。

3) 未到期期限: 近到期 ( $T - kh$  小),  $\frac{\partial \delta}{\partial t}$  (charm) 大,  $\delta_k$  变化快, 需高频调整, 成本和误差均增

□

```

In [89]: import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
from scipy.stats import norm

# 设置随机种子以确保可重复性
np.random.seed(233)

# 参数设置
S0 = 49 # 初始标的资产价格
K = 50 # 期权执行价 (平价期权)
T = 0.3846 # 期权期限 (年)
r = 0.05 # 无风险利率
sigma = 0.2 # 波动率
miu = 0.13
c = 0.000 # 交易成本率 (每次交易的比例)
N_paths = 1000000 # 模拟路径数
weeks_per_year = 52 # 一年52周
n_values = [5, 4, 2, 1, 0.5, 0.25] # 对冲间隔周数

# 计算理论期权价格 (Black-Scholes公式)
def bs_call_price(S, K, T, r, sigma):
    d1 = (np.log(S / K) + (r + 0.5 * sigma**2) * T) / (sigma * np.s
    d2 = d1 - sigma * np.sqrt(T)
    call_price = S * norm.cdf(d1) - K * np.exp(-r * T) * norm.cdf(d
    return call_price

# 生成几何布朗运动 (GBM) 路径
def generate_gbm_paths(S0, T, miu, sigma, n_steps, n_paths):
    dt = T / n_steps
    paths = np.zeros((n_paths, n_steps + 1))
    paths[:, 0] = S0
    for i in range(n_steps):
        z = np.random.standard_normal(n_paths)
        paths[:, i + 1] = paths[:, i] * np.exp((miu - 0.5 * sigma**
    return paths

# 生成二叉树模型路径
def generate_binomial_paths(S0, T, miu, sigma, n_steps, n_paths):
    dt = T / n_steps
    u = np.exp(sigma * np.sqrt(dt))
    d = 1 / u
    p = (np.exp(miu * dt) - d) / (u - d)
    paths = np.zeros((n_paths, n_steps + 1))
    paths[:, 0] = S0
    for i in range(n_steps):
        random_values = np.random.rand(n_paths)
        up_move = random_values < p
        paths[up_move, i + 1] = paths[up_move, i] * u
        paths[~up_move, i + 1] = paths[~up_move, i] * d
    return paths

```

# 止损策略实现

```
def stop_loss_hedge(paths, K, cost_rate, n_steps):
    n_paths = paths.shape[0]
    PL_list = []
    # 计算理论期权价格
    BS_price = bs_call_price(S0, K, T, r, sigma)
    for i in range(n_paths):
        S_path = paths[i, :]
        cash = BS_price # 初始现金为期权价格
        stock_holding = 0 # 股票头寸, 0表示无持仓, 1表示持有1股
        # 初始头寸设置
        if S_path[0] >= K:
            cash -= S_path[0] * (1 + cost_rate) # 买入股票, 支付交易成本
            stock_holding = 1
        # 遍历每个时间点 (到期前)
        for j in range(1, n_steps):
            S_curr = S_path[j]
            if stock_holding == 0 and S_curr >= K:
                # 买入股票
                cash -= S_curr * (1 + cost_rate)
                stock_holding = 1
            elif stock_holding == 1 and S_curr < K:
                # 卖出股票
                cash += S_curr * (1 - cost_rate)
                stock_holding = 0
        # 到期日处理
        S_T = S_path[-1]
        if stock_holding == 1:
            cash += S_T * (1 - cost_rate) # 卖出股票
        option_payoff = max(S_T - K, 0) # 期权支付
        cash -= option_payoff
        PL = cash # 损益为最终现金
        PL_list.append(PL)
    return np.array(PL_list)
```

# 计算止损策略的表现

```
def analyze_stop_loss():
    BS_price = bs_call_price(S0, K, T, r, sigma)
    results_gbm = {}
    results_binomial = {}
    for n in n_values:
        steps = int(20 / n) # 总步数
        if steps == 0:
            continue
        # print(f"\n对冲间隔: {n}周, 步数: {steps}")
        # GBM路径
        paths_gbm = generate_gbm_paths(S0, T, mu, sigma, steps, N)
        PL_gbm = stop_loss_hedge(paths_gbm, K, c, steps)
        std_ratio_gbm = np.std(PL_gbm) / BS_price
        results_gbm[n] = std_ratio_gbm
        # print(f"GBM - 标准差比率: {std_ratio_gbm:.4f}")
        # 二叉树路径
        paths_binomial = generate_binomial_paths(S0, T, mu, sigma,
        PL_binomial = stop_loss_hedge(paths_binomial, K, c, steps)
        std_ratio_binomial = np.std(PL_binomial) / BS_price
        results_binomial[n] = std_ratio_binomial
```

```

        # print(f"二叉树 - 标准差比率: {std_ratio_binomial:.4f}")
    return results_gbm, results_binomial

plt.show()

```

```

In [91]: # 主程序
if __name__ == "__main__":
    BS_price = bs_call_price(S0, K, T, r, sigma)
    print(f"理论期权价格: {BS_price:.4f}")
    results_gbm, results_binomial = analyze_stop_loss()

    # 绘制结果
    n_list = list(results_gbm.keys())
    gbm_ratios = [results_gbm[n] for n in n_list]
    bin_ratios = [results_binomial[n] for n in n_list]

    r_df = pd.DataFrame({
        'gbm 模拟': gbm_ratios,
        '二叉树模拟': bin_ratios
    }).T

    r_df.columns = [f'{n}周' for n in n_values]
    display(r_df.round(2))

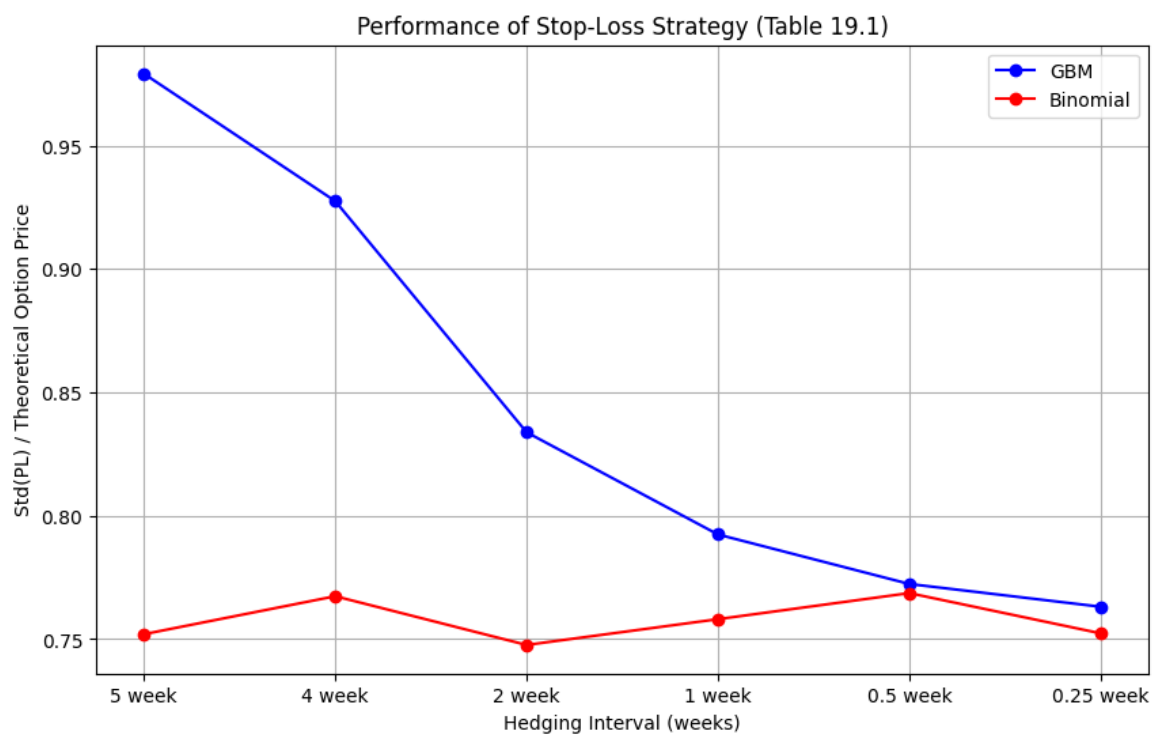
    plt.figure(figsize=(10, 6))

    plt.plot([f'{n} week' for n in n_values], gbm_ratios, 'bo-', label='gbm 模拟')
    plt.plot([f'{n} week' for n in n_values], bin_ratios, 'ro-', label='二叉树模拟')
    plt.xlabel('Hedging Interval (weeks)')
    plt.ylabel('Std(PL) / Theoretical Option Price')
    plt.title('Performance of Stop-Loss Strategy (Table 19.1)')
    plt.legend()
    plt.grid(True)

```

理论期权价格: 2.4005

	5周	4周	2周	1周	0.5周	0.25周
gbm 模拟	0.98	0.93	0.83	0.79	0.77	0.76
二叉树模拟	0.75	0.77	0.75	0.76	0.77	0.75



In [ ]: