黄之豪

20110980005

更详细的相关内容会在作业截止提交后上传github：https://github.com/HUANGZHIHAO1994/Financial-risk-management.git

一、涉及内容

第二次作业涉及：

- **蒙特卡洛计算亚式期权以及希腊字母计算（第一题）：** 样本路径生成、亚式期权定价、置信区间估计、delta、vega使用pathwise method、gamma使用likelihood ratio method计算
- **Longstaff-Schwartz method 计算美式看跌（第二题）：** 样本路径生成、Tsitsiklis et al.(1999)方法、动态规划

二、项目结构及相关说明

1. 结构及说明

```
├── config.py
├── hw2_1.py
├── hw2_1.txt
├── hw2_2_compare.txt
├── hw2_2.py
├── hw2_2.txt
├── LSM.py
├── project_structure.txt
├── __pycache__
│   ├── config.cpython-36.pyc
│   └── hw2_1.cpython-36.pyc
├── references
│   ├── LongstaffSchwartzAmericanOptionsLeastSquareMonteCarlo.pdf
│   ├── Tsitsiklis.pdf
│   └── 随机波动率模型下基于精确模拟算法的期权计算理论.pdf
├── Report.pdf
└── requirements.txt

2 directories, 15 files
```

其中，以下是主要文件：

config.py 是两道题都使用的全局变量文件，第一题使用 **hw2_1.py**，第二题使用 **hw2_1.py** 和 **hw2_2.py** 以及与 **LSM.py** 对比

2. 环境

```
pip install -r requirements.txt
```

三、第一题（hw2_1.py）

为了便于复现结果，本次作业统一使用 `np.random.seed(1234)` 作为随机种子！

相关数值计算结果可见于 **5. 模拟结果**

1. 生成样本路径

给定条件：$S(t_0)$=30，K=30，r=3%，σ=35%，T=1

使用以下公式生成样本路径：

$$S(t_{i+1}) = S(t_i)exp\{(r - \frac{\sigma^2}{2})(t_{i+1} - t_i) + \sigma\sqrt{t_{i+1} - t_i}Z_{i+1}\} \quad i = 0, 1, \cdots, 251$$

$$Z_{i+1} \sim N(0, 1)$$

$$Sample\ path_{252\times10000} = \begin{pmatrix} s(t_1)_1 & s(t_1)_2 & \cdots & s(t_1)_{10000} \\ s(t_2)_1 & s(t_2)_2 & \cdots & s(t_2)_{10000} \\ \vdots & \vdots & \ddots & \vdots \\ s(t_{252})_1 & s(t_{252})_2 & \cdots & s(t_{252})_{10000} \end{pmatrix}$$

本次作业中， $\Delta t = t_{i+1} - t_i = \frac{1}{252}$ ，实现样本路径可通过生成 (252, 10000) 维的标准正态随机数，之后逐行计算 S(t) ，具体可见 `generate_samples` 方法：

```
def generate_samples():
```

2. 亚式期权定价

生成样本路径后，亚式期权可根据下式定价：

$$\bar{S}_i = \frac{1}{252}\sum_{m=1}^{252} S(t_m)_i$$

$$\bar{c} = \frac{1}{10000}\sum_{i=1}^{10000} e^{-rT}(\bar{S}_i - K)^+$$

详情可见 `calculate` 方法：

```
def calculate(stock_matrix, d_st_d_sigma_matrix, standard_normal_matrix):
```

3. 置信区间计算

按下式计算置信区间:

$$c_i = e^{-rT}(\bar{S}_i - K)^+$$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(c_i - \bar{c})^2$$

$$(\bar{c} - t_{1-\frac{\alpha}{2}}(n-1)\frac{s}{\sqrt{n}},\ \bar{c} + t_{1-\frac{\alpha}{2}}(n-1)\frac{s}{\sqrt{n}})$$

详情可见 `calculate` 方法:

```
def calculate(stock_matrix, d_st_d_sigma_matrix, standard_normal_matrix):
```

4. 希腊字母计算

4.1 delta计算

亚式期权 delta 计算在 lecturenote6 中已经给出公式，此处不再重复推导，公式如下:

$$\alpha^{'}(S_0) = E[e^{-rT}1_{\{\bar{S}>K\}}\frac{\bar{S}}{S_0}] \approx \frac{1}{n}\sum_{i=1}^{n}e^{-rT}1_{\{\bar{S}_i>K\}}\frac{\bar{S}_i}{S_0}$$

delta 计算相对简单，并不需要新增计算，详情可见 `calculate` 方法:

```
def calculate(stock_matrix, d_st_d_sigma_matrix, standard_normal_matrix):
```

4.2 vega计算

亚式期权 vega 计算与 lecturenote6 中 delta 计算推导相似，但要比 delta 复杂，具体推导如下:

$$\alpha(\sigma) = E[e^{-rT}(\bar{S} - K)^+] = E[Y(\sigma)]$$

$$\frac{\mathrm{d}}{\mathrm{d}\sigma}Y(\sigma) = e^{-rT}1_{\{\bar{S}>K\}}\frac{\mathrm{d}}{\mathrm{d}\sigma}\bar{S}$$

$$\frac{\mathrm{d}}{\mathrm{d}\sigma}\bar{S} = \frac{1}{m}\sum_{i=1}^{m}\frac{\mathrm{d}}{\mathrm{d}\sigma}S(t_i)$$

根据 lecturenote6 中 $S(t_i)$ 计算 $\frac{\mathrm{d}}{\mathrm{d}\sigma}S(t_i)$ 得:

$$S(t_i) = S(t_0)exp\{(r - \frac{\sigma^2}{2})t_i + \sigma\sqrt{\Delta t}(Z_1 + \cdots + Z_i)\}$$

$$\frac{\mathrm{d}}{\mathrm{d}\sigma}S(t_i) = S(t_i)(-\sigma t_i + \sqrt{\Delta t}(Z_1 + \cdots + Z_i))$$

综上得到 vega 计算式：

$$\alpha^{'}(\sigma) = E[e^{-rT}1_{\{\bar{S}>K\}}\frac{1}{m}\sum_{i=1}^{m}S(t_i)(-\sigma t_i + \sqrt{\Delta t}(Z_1 + \cdots + Z_i))]$$

可以看到， vega 与 delta 计算差别主要在 $\frac{\mathrm{d}}{\mathrm{d}\sigma}S(t_i)$ 上，此处采用在生成样本路径时一起计算 $\frac{\mathrm{d}}{\mathrm{d}\sigma}S(t_i)$ 的方法，详情可见 `generate_samples` 方法中 `d_st_d_sigma_matrix` 和 `calculate` 方法：

```
d_st_d_sigma_matrix[i] = stock_matrix[i] * (-VOLATILITY * DELTA_T * (i + 1) +
np.sqrt(DELTA_T) * sum_of_standard_normal_matrix[i])

def calculate(stock_matrix, d_st_d_sigma_matrix, standard_normal_matrix):
```

4.3 gamma计算

亚式期权 gamma 计算使用了 likelihood ratio method ，是希腊字母计算中最复杂的，也是最容易出错的，此处使用符号与 lecturenote6 中一致，具体推导如下：

$$\begin{aligned}\alpha^{''}(\theta) &= \int_{\mathbb{R}^m} g(\vec{x})\frac{\partial^2 f(\vec{x};\theta)}{\partial\theta^2}\mathrm{d}\vec{x} \\ &= \int_{\mathbb{R}^m} g(\vec{x})\frac{\frac{\partial^2 f(\vec{x};\theta)}{\partial\theta^2}}{f(\vec{x};\theta)}f(\vec{x};\theta)\mathrm{d}\vec{x} \\ &= E[g(\vec{x})\frac{\frac{\partial^2 f(\vec{x};\theta)}{\partial\theta^2}}{f(\vec{x};\theta)}]\end{aligned}$$

4.3.1 错误的做法

值得注意的是，必须从上式开始推导，<u>不能使用</u>如下方法推导：

$$\alpha^{''}(\theta) = \frac{\mathrm{d}}{\mathrm{d}\theta}\alpha^{'}(\theta) = \frac{\partial}{\partial\theta}E[g(\vec{x})\frac{\partial}{\partial\theta}log[f(\vec{x};\theta)]]$$

原因在于上式第二个等号不成立

$$\frac{\partial}{\partial\theta}E[g(\vec{x})\frac{\partial}{\partial\theta}log[f(\vec{x};\theta)]] = E[g(\vec{x})\frac{\partial}{\partial\theta}(\frac{\frac{\partial f(\vec{x};\theta)}{\partial\theta}}{f(\vec{x};\theta)})] \neq E[g(\vec{x})\frac{\frac{\partial^2 f(\vec{x};\theta)}{\partial\theta^2}}{f(\vec{x};\theta)}]$$

$$\frac{f^{''}(x)}{f(x)} \neq (\frac{f^{'}(x)}{f(x)})^{'}$$

因此，求 gamma <u>不能通过</u>求得 delta 的结果：$\frac{\partial log(g(S_1,\cdots,S_m))}{\partial S_0} = \frac{Z_1}{S_0\sigma\sqrt{\Delta t}}$ 基础上继续求导得到，如此做法在最终 gamma 结果中会少了 Z_1^2 项。

4.3.2 正确的做法

只能利用 $\alpha''(\theta) = E[g(\vec{x})\frac{\frac{\partial^2 f(\vec{x};\theta)}{\partial\theta^2}}{f(\vec{x};\theta)}]$ 求 gamma，可利用中间结果 $\frac{\partial log(g(S_1,\cdots,S_m))}{\partial S_0} = \frac{logS_1 - logS_0 - (r-\frac{\sigma^2}{2})\Delta t}{S_0\sigma^2\Delta t}$ ，具体使用如下：

$$\frac{\partial g(S_1,\cdots,S_m)}{\partial S_0} = \frac{\partial log(g(S_1,\cdots,S_m))}{\partial S_0} g(S_1,\cdots,S_m)$$

$$\begin{aligned}
\frac{\partial^2 g(S_1,\cdots,S_m)}{\partial S_0^2} &= \frac{\partial}{\partial S_0}(\frac{logS_1 - logS_0 - (r-\frac{\sigma^2}{2})\Delta t}{S_0\sigma^2\Delta t} g(S_1,\cdots,S_m)) \\
&= (\frac{-1}{S_0^2\sigma^2\Delta t} - \frac{Z_1}{S_0^2\sigma\sqrt{\Delta t}}) g(S_1,\cdots,S_m) + \frac{logS_1 - logS_0 - (r-\frac{\sigma^2}{2})\Delta t}{S_0\sigma^2\Delta t} \frac{\partial g(S_1,\cdots,S_m)}{\partial S_0} \\
&= (\frac{-1 - Z_1\sigma\sqrt{\Delta t}}{S_0^2\sigma^2\Delta t}) g(S_1,\cdots,S_m) + (\frac{logS_1 - logS_0 - (r-\frac{\sigma^2}{2})\Delta t}{S_0\sigma^2\Delta t})^2 g(S_1,\cdots,S_m) \\
&= \frac{Z_1^2 - 1 - Z_1\sigma\sqrt{\Delta t}}{S_0^2\sigma^2\Delta t} g(S_1,\cdots,S_m)
\end{aligned}$$

进而得到

$$\alpha''(\theta) = E[e^{-rT}(\bar{S} - K)^+ \frac{Z_1^2 - 1 - Z_1\sigma\sqrt{\Delta t}}{S_0^2\sigma^2\Delta t}]$$

上述推导也可参考下面文献的 P1546-1547 (文章中分母 S_0 还是少了个平方，但总体推导思路正确)

> [1]马俊美,杨宇婷,顾桂定,徐承龙.随机波动率模型下基于精确模拟算法的期权计算理论[J].同济大学学报(自然科学版),2017,45(10):1539-1548.

具体代码实现可见 `calculate` 方法中：

```
pathwise_gamma = np.mean(np.exp(-RISK_FREE_RATE_OPTION * EXPIRES_ANNUALIZE) * np.maximum(0, s_i_bar_vector - K) * (np.power(standard_normal_matrix[0], 2) - standard_normal_matrix[0] * VOLATILITY * np.sqrt(DELTA_T) - 1) / (np.power(S_0 * VOLATILITY, 2) * DELTA_T))
```

5. 模拟结果

计算结果也可见于： **hw2_1.txt**

注意：gamma 由于是二阶导，本次作业也只取了10000个 sample path ，因此随机种子设置不同会导致结果相差非常大。因此再次强调：以下是设定 `np.random.seed(1234)` 下计算出的结果：

Asion option price	2.633973843392969
confidence interval	[2.5490384687955756, 2.7189092179903622]
delta	0.5529336723428989
vega	6.873542661601932
gamma	0.1917685559177979

四、第二题（hw2_2.py）

由于洪教授上课 lecturenote5 中 Longstaff-Schwartz 方法与 Longstaff-Schwartz (2001)：Valuing American Options by Simulation: A Simple Least-Squares Approach 原文有所不同，故此处使用了三种方法并比较结果和运行速度。

数值计算结果见 **5.计算结果**

1. 生成样本路径

参考第一题生成样本路径方法，完全一样

2. Tsitsiklis et al.(1999)方法

> Roy J N T B V . Optimal Stopping of Markov Processes: Hilbert Space Theory, Approximation Algorithms, and an Application to Pricing High-Dimensional Financial Derivatives[J]. Automatic Control IEEE Transactions on, 1999, 44(10):1840-1851.
>
> 格拉瑟曼, PaulGlasserman, Glasserman, et al. 金融工程中的蒙特卡罗方法[M]. 高等教育出版社, 2013.

其实，洪教授 lecturenote5 中的 Longstaff-Schwartz 方法是Tsitsiklis et al. (1999)方法，通过多项式回归拟合Continue Value 之后，进而通过下式使用从后向前逐步递推方法获取期权价值（具体算法由于讲义中有不在此处详述）：

$$\widetilde{V_{m-1,j}} = \max\{(K - S_{m-1,j})^+, C_{m-1}(S_{m-1,j})\}$$

2.1 优点

优点是计算简单、理解容易，相比 Longstaff-Schwartz (2001) 原文中的方法少了判断 in the money 以及动态规划部分，理解也相对容易是一种非常直觉的 **拟合—继续价值与立即行权价值取最大—继续向前计算** 流程。

2.2 缺点

1. 容易高估期权价值。从上式也可以看到，如果 Continue Value 拟合值 $C_{m-1}(S_{m-1,j})$ 十分大（高估许多），是会影响到后续许多步的拟合值，也可能最终直接影响期权价值 V_0 。那为什么说这个方法容易高估呢，因为拟合值 $C_{m-1}(S_{m-1,j})$ 十分小的时候会选取 $(K - S_{m-1,j})^+$ 作为下一次拟合依据，影响并不是那么大。
2. 由1，自然地，该方法对拟合精度要求高。
3. 该方法与常规美式期权（二叉树计算）思路并不同，拟合出 Continue Value，不仅用作判断是否 exercise，还可能用到下一次拟合和以及影响期权定价。

具体实现可见 `tsitsiklis` 方法：

```python
def tsitsiklis(stock_matrix):
```

3. Longstaff-Schwartz

> Longstaff F A , Schwartz E S . Valuing American Options by Simulation: A Simple Least-Squares Approach[J]. Review of Financial Studies, 2001(1):113-147.
>
> 格拉瑟曼, PaulGlasserman, Glasserman, et al. 金融工程中的蒙特卡罗方法[M]. 高等教育出版社, 2013.

Longstaff-Schwartz(2001) 原文P117："We use only in-the-money paths since it allows us to better estimate the conditional expectation function in the region where exercise is relevant and significantly improves the efficiency of the algorithm."

上述说明Longstaff-Schwartz(2001)只选择那些 in-the-money paths 进行多项式回归拟合，因为与二叉树计算美式看跌期权思路一致：当期如果立即执行期权收益为0，那么不管Continue Value是多少，总是会选择不执行。

通过查看 Longstaff-Schwartz(2001) 原文中 P115-120例子可以发现（文章相对易懂可看）：

1. 拟合的Continue Value仅用作判断是否当期执行，如果确定当期执行会把后续现金流全设为0（已经执行自然没有现金流），这一点也符合二叉树计算美式看跌期权的思路
2. 拟合用的Y是 in-the-money paths 中向后追述第一个也是唯一一个现金流大于0的值。因为 Longstaff-Schwartz(2001) 原文方法考虑整体现金流，因此最终期权价值是整个现金流矩阵折现得到，相关计算也使用了动态规划思想

《金融工程中的蒙特卡罗方法》中 P434 也提及了两种方法差别。

3.1 优点

计算更精准，并且计算思路与二叉树方法计算美式看跌期权价格一致。

3.2 缺点

动态规划耗时较长，且程序相对不易实现

本次作业针对其耗时较长的缺点，通过程序设计可以优化大大缩短时长

3.3 优化思路

首先要深入理解 Longstaff-Schwartz 方法，该方法其实每一步都分为 out-of-money 和 in-the-money ，再通过对 in-the-money 细分，最终形成三部分：out-of-money、continue、exercise，其中后两部分是 in-the-money 部分的细分。

优化方法： 还是利用 Tsitsiklis et al. (1999) 方法逐步向前推进的思路，具体地，每一步：把那些 out-of-money 的 以及 判断要 continue 的 sample path 的上一期现金流折现到当前步，其余判断要 exercise 的 sample path 自然地选择 $K - S_t$ 作为当期现金流，如此逐步向前迭代下去。

这么做的好处是：实质上每次计算只需用当前步和上一期的现金流数据，而动态规划需要用整个现金流矩阵，并且动态规划判断要 continue 时向后追述现金流以及判断要 exercise 时向后将现金流都设置为0都是十分耗时的。

详细实现可见 `longstaff_schwartz` 方法：

```
def longstaff_schwartz(stock_matrix):
```

4. 结合Longstaff-Schwartz和Tsitsiklis

优缺点也介于两种方法之间。

与 Longstaff-Schwartz 方法相同的是，把那些 out-of-money 的现金流折现；但是，不同的是，根据判断结果确定要 continue 时，现金流使用多项式回归拟合的 Continue Value ，具体可见 `longstaff_schwartz_combine_tsitsiklis` 方法

```
def longstaff_schwartz_combine_tsitsiklis(stock_matrix):
```

5. 计算结果

5.1 说明以及结果展示

其实 Longstaff-Schwartz 方法已有学者在github上实现：https://github.com/aminjellali/Longstaff-and-Schwartz.git

该作者完全使用原文动态规划思路进行编程，我对他的程序稍作修改之后使不同程序能在结果上一致，进而进行性能比较：

1. 该作者期权参数与题目给定不同
2. 该作者使用了Laguerre多项式，与Longstaff_Schwartz(2001)和洪教授讲义使用一般多项式不同

修改完形成 **LSM.py** 文件。

我自己写的程序是 **hw2_2.py** 程序中有上述提及的三种方法实现，比较结果如下（结果也保存在了 **hw2_2.txt** 和 **hw2_2_compare.txt** 文件中）：

hw2_2.py	
Longstaff_Schwartz (2001)计算用时	2.655221536755562
Longstaff_Schwartz (2001) price	3.8151357853081094
Combine of (Longstaff_Schwartz, 2001) and Tsitsiklis et al, 1999) price	3.8306045815398466
Tsitsiklis et al. (1999) or the lecturenote5 from Prof. L. Jeff Hong price	4.4527336640111255
LSM.py	
Longstaff_Schwartz (2001) price	3.815135785308124
Longstaff_Schwartz (2001)计算用时	172.07279300689697

5.2 总结

1. 可以看到计算结果一致情况下，**hw2_2.py** 中 Longstaff_Schwartz(2001) 方法用时快了65倍（其实有一种情况是必须使用动态规划的，那就是想要知道执行期权的具体时刻，即 Longstaff Schwartz (2001) P119 stopping rule 表格。如果只关心定价则可使用 **hw2_2.py** ）
2. 可以看到，Tsitsiklis 方法确实会高估期权价值，两者结合也确实介于两者之间
3. **注**： 同样的参数，第一次作业美式看跌期权计算结果为：3.7557436745895885，显然 Longstaff Schwartz(2001) 是与该结果最接近的，这与他们思路相同也有关系