量子计算
——量子金融

Quantum Finance

网址：www.qubits.top

作者：Calvin Tang

邮箱：179209347@qq.com

Qubits
qubits.top

介绍

本教程基于 IBM 的 **Qiskit，Qiskit[finance]** 编写。
https://qiskit.org/documentation/finance/tutorials/05_bull_spread_pricing.html

本教程包含：
1. 牛市套利
2. 量子算法 - 通过QAE求解问题
3. 代码实例
 * TODO：完善算法的详细解读

Qiskit：
https://qiskit.org/documentation/getting_started.html
Qiskit finance：
https://qiskit.org/documentation/finance/tutorials/index.html

Github & Gitee 代码地址：
https://github.com/mymagicpower/qubits/tree/main/quantum_qiskit_finance/05_bull_spread_pricing.py
https://gitee.com/mymagicpower/qubits/tree/main/quantum_qiskit_finance/05_bull_spread_pricing.py

Qiskit，Qiskit[finance] 配置和安装

虚拟环境

```
# 创建虚拟环境
conda create -n ENV_NAME python=3.8.0
# 切换虚拟环境
conda activate ENV_NAME
# 退出虚拟环境
conda deactivate ENV_NAME
# 查看现有虚拟环境
conda env list
# 删除现有虚拟环境
conda remove -n ENV_NAME --all
```

安装 Qiskit

```
pip install qiskit
```

```
# install extra visualization support
# For zsh user (newer versions of macOS)
# pip install 'qiskit[visualization]'

pip install qiskit[visualization]
```

安装 Qiskit[finance]

```
# For zsh user (newer versions of macOS)
# pip install 'qiskit[finance]’

pip install qiskit[finance]
```

Calvin，QQ: 179209347 Mail: 179209347@qq.com

期权的概念

期权是一种“选择交易与否的权利”。
如果此权利为“买进”标的物，则称为买权，也称为看涨期权；
如果此权利为“卖出”标的物，则称为卖权，也称为看跌期权。

期权交易的4种策略

- 买入“买权（看涨期权）”(看涨期权多头)
- 卖出“买权（看涨期权）”
- 买入“卖权（看跌期权）”
- 卖出“卖权（看跌期权）”

期权的分类

- 欧式期权与美式期权
- 实值期权、平值期权、虚值期权
- 现货期权、期货期权

期权合约要素

- 标的物
- 单位合约数量
- 执行日期（到期日）
- 执行价格（敲定价格）
- 期权买方（期权多头方）
- 期权卖方（期权空头方）
- 权利金（期权费、期权价格）

Calvin，QQ: 179209347 Mail: 179209347@qq.com

几种期权交易策略

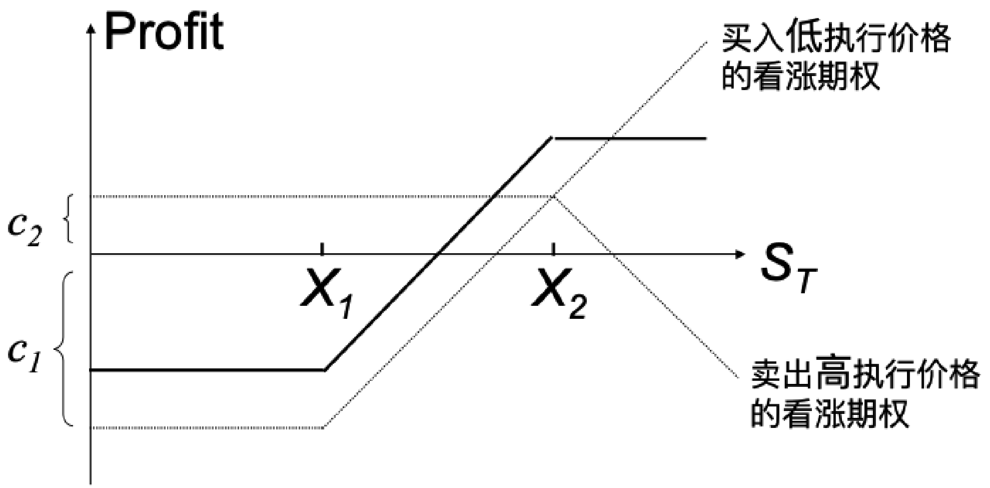
- 单个期权交易
- 用一个期权和基础资产组合
- 期权组合 Combination：同时用看涨和看跌构造的交易策 略
  – 跨式期权 Straddle
  – 偏跨式期权 Strips and Straps – 宽跨式期权 Strangles
- 期权价差 Spread：用同一类型的两个或更多期权构造的交 易策略
- – 牛市价差 Bull Spreads
  – 熊市价差 Bear Spreads
  – 蝶式价差 Butterfly Spreads
  – 日历价差 Calendar Spreads
  – 对角价差 Diagonal Spreads

牛市价差 — 用看涨期权构造

- 有两份看涨期权
  相同的到期日： $T_1 = T_2$
  不同的执行价格： $X_1 < X_2$
  期权价格也不同： $c_1 > c_2$

策略:
买入一份较低执行价格的看涨期权
卖出一份较高执行价格的看涨期权

初始支出 : c_1
初始收入: c_2
初始净支出: $c_1 - c_2$



- 初始净支出: $c_1 - c_2$
- 可能从牛市中获得的最大收入: $(X_2 - X_1) - (c_1 - c_2)$
- 盈亏平衡点: $X_1 + (c_1 - c_2)$

牛市价差 — 用看涨期权构造（简化模型）

假定 $K_1<K_2$，损益函数定义为： $\min\{\max\{S_T-K_1,0\},K_2-K_1\}$

- T：期权到期日
- S：现货市价
- K：成交价

在后面的例子里，量子计算算法基于振幅估计实现，估算到期损益：

$$\mathbb{E}\left[\min\{\max\{S_T-K_1,0\},K_2-K_1\}\right]$$

金融衍生品期权定价的现货市价约束条件：

$$\Delta=\mathbb{P}\left[K_1\leq S\leq K_2\right]$$

Uncertainty Model

在后面的例子里，量子计算算法基于振幅估计实现，估算到期损益：

我们构造一个量子线路，加载对数正态分布(Log-Normal Distribution)数据，初始化量子态。
数据分布区间[low,high]，使用 2^n 个网格点离散化， n 表示使用的量子位数。幺正变换算子如下：

$$|0\rangle_n \mapsto |\psi\rangle_n = \sum_{i=0}^{2^n-1} \sqrt{p_i}|i\rangle_n,$$

P_i 表示离散分布概率，i为对应正确区间的仿射映射：

$$\{0, \dots, 2^n - 1\} \ni i \mapsto \frac{\text{high} - \text{low}}{2^n - 1} * i + \text{low} \in [\text{low}, \text{high}].$$

Calvin，QQ: 179209347 Mail: 179209347@qq.com

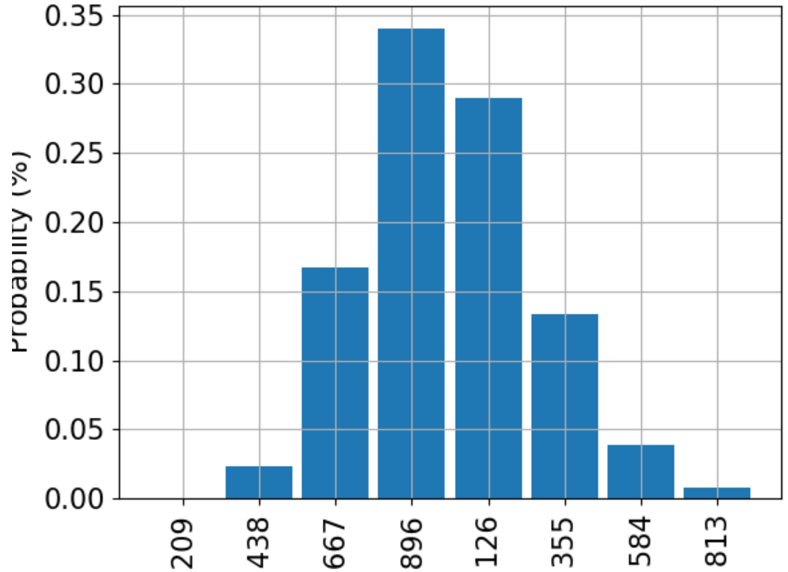
Uncertainty Model

```
# parameters for considered random distribution
S = 2.0  # initial spot price
vol = 0.4  # volatility of 40%
r = 0.05  # annual interest rate of 4%
T = 40 / 365  # 40 days to maturity

# resulting parameters for log-normal distribution
mu = (r - 0.5 * vol**2) * T + np.log(S)
sigma = vol * np.sqrt(T)
mean = np.exp(mu + sigma**2 / 2)
variance = (np.exp(sigma**2) - 1) * np.exp(2 * mu + sigma**2)
stddev = np.sqrt(variance)

# lowest and highest value considered for the spot price; in
between, an equidistant discretization is considered.
low = np.maximum(0, mean - 3 * stddev)
high = mean + 3 * stddev
```

```
# construct A operator for QAE for the payoff function by
# composing the uncertainty model and the objective
uncertainty_model = LogNormalDistribution(
    num_uncertainty_qubits, mu=mu, sigma=sigma**2,
bounds=(low, high)
)
```



https://qiskit.org/documentation/finance/tutorials/03_european_call_option_pricing.html

损益函数 （Payoff Function）

S_T<K_1时损益函数等于0，然后损益函数线性增加，边界为K_2。
使用2个比较器实现，如果S_T>=K_1且S_T<=K_2时，会交换辅助量子比特: |0> → |1>，
辅助量子比特用于控制损益函数的线性部分。
当 |y| 足够小时：

$$\sin^2(y + \pi/4) \approx y + 1/2$$

因此，对于一个给定的近似缩放因子： $c_{\text{approx}} \in [0, 1]$ and $x \in [0, 1]$

有：

$$\sin^2(\pi/2 * c_{\text{approx}} * (x - 1/2) + \pi/4) \approx \pi/2 * c_{\text{approx}} * (x - 1/2) + 1/2$$

我们使用受控绕Y轴旋转，很容易构造一个算子：

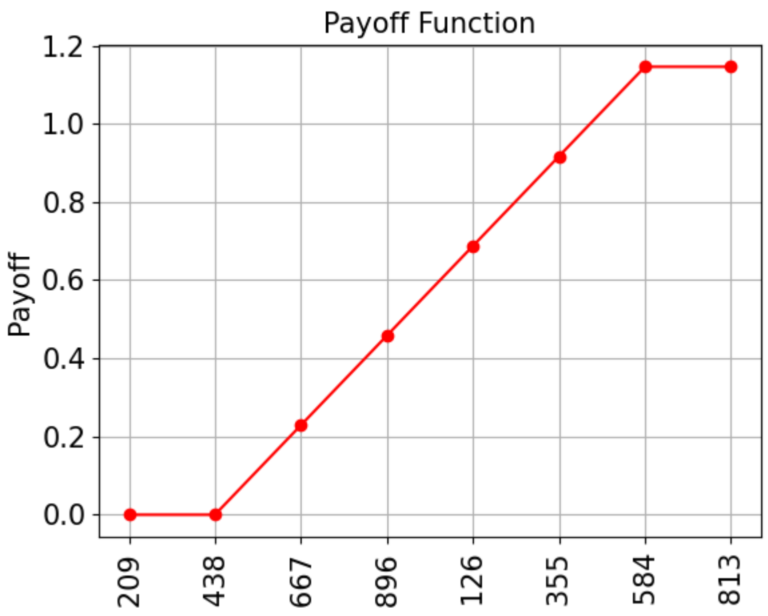
$$|x\rangle|0\rangle \mapsto |x\rangle\left(\cos(a * x + b)|0\rangle + \sin(a * x + b)|1\rangle\right)$$

最后，我们需要做的是测量|1>的概率振幅： $\sin^2(a * x + b)$

c_{approx} 越小越准，但是量子位m也需要相应调整。

损益函数

```
bull_spread_objective = LinearAmplitudeFunction(
    num_uncertainty_qubits,
    slopes,
    offsets,
    domain=(low, high),
    image=(f_min, f_max),
    breakpoints=breakpoints,
    rescaling_factor=rescaling_factor,
)
# construct A operator for QAE for the payoff function by
# composing the uncertainty model and the objective
bull_spread = bull_spread_objective.compose(uncertainty_model, front=True)
# plot exact payoff function (evaluated on the grid of the uncertainty model)
x = uncertainty_model.values
y = np.minimum(np.maximum(0, x - strike_price_1), strike_price_2 - strike_price_1)
…
plt.show()
```



结果：

```
exact expected value:   0.5695
exact delta value:      0.9291
```

振幅估计

```python
# Evaluate Expected Payoff
# set target precision and confidence level
epsilon = 0.01
alpha = 0.05

qi = QuantumInstance(Aer.get_backend("aer_simulator"), shots=100)
problem = EstimationProblem(
    state_preparation=bull_spread,
    objective_qubits=[num_uncertainty_qubits],
    post_processing=bull_spread_objective.post_processing,
)
# construct amplitude estimation
ae = IterativeAmplitudeEstimation(epsilon, alpha=alpha,
quantum_instance=qi)
result = ae.estimate(problem)
conf_int = np.array(result.confidence_interval_processed)
print("Exact value:    \t%.4f" % exact_value)
print("Estimated value:\t%.4f" % result.estimation_processed)
print("Confidence interval: \t[%.4f, %.4f]" % tuple(conf_int))
```

结果：

```
Exact value:            0.5695
Estimated value:        0.5630
Confidence interval:    [0.5454, 0.5805]
```

参考

[1] Quantum Risk Analysis. Woerner, Egger. 2018.
https://www.nature.com/articles/s41534-019-0130-6

[2] Option Pricing using Quantum Computers. Stamatopoulos et al. 2019.
https://quantum-journal.org/papers/q-2020-07-06-291/

Qubits
qubits.top

Thank

You