量子计算
—量子金融

Quantum Finance

网址：www.qubits.top

作者：Calvin Tang

邮箱：179209347@qq.com

介绍

本教程基于 IBM 的 **Qiskit，Qiskit[finance]** 编写。
https://qiskit.org/documentation/finance/tutorials/03_european_call_option_pricing.html

本教程包含：
1. 欧式看涨期权定价
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/03_european_call_option_pricing.py
https://gitee.com/mymagicpower/qubits/tree/main/quantum_qiskit_finance/03_european_call_option_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]
```

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）

期权的损益分析

- 执行价值：期权买方执行期权权利时能获得的利润。

(以欧式期权为例)

- T：期权到期日。
- S：现货市价
- K：成交价
- c：欧式买权的期权费
- p：欧式卖权的期权费

期权种类	到期损益
欧式看涨 期权多头	Max { $S_T - K - c$, - c }
欧式看涨 期权空头	Min { $K - S_T + c$, + c }
欧式看跌 期权多头	Max { $K - S_T - p$, - p }
欧式看跌 期权空头	Min { $S_T - K + p$, + p }

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

期权的损益分析 – 看涨期权

- 执行价值：期权买方执行期权权利时能获得的利润。

(以欧式期权为例)

- T：期权到期日。
- S：现货市价
- K：成交价
- c：欧式买权的期权费（假定为 0）

期权种类	到期损益
欧式看涨 期权多头	Max { S_T – K, 0 }
欧式看涨 期权空头	Min { K – S_T, 0}

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：

$$\mathbb{E}\left[\max\{S_T - K, 0\}\right]$$

金融衍生品期权定价的现货市价约束条件：

$$\Delta = \mathbb{P}\left[S_T \geq K\right]$$

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：

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

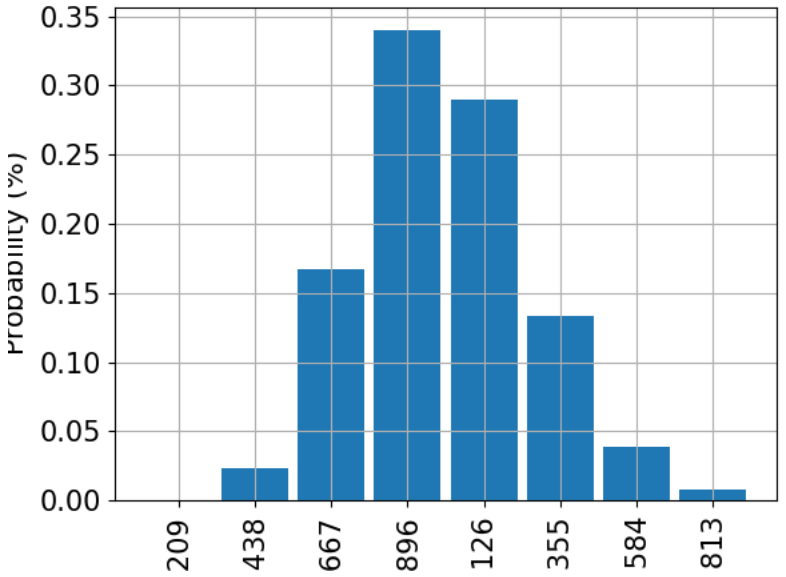
Uncertainty Model

```
# 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

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

损益函数 （Payoff Function）

S_T小于等于K时损益函数等于0，然后线性增加。
使用比较器实现，它会交换辅助量子比特: |0> → |1>, 如果S_T>=K, 该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：　$\sin^2(a * x + b)$

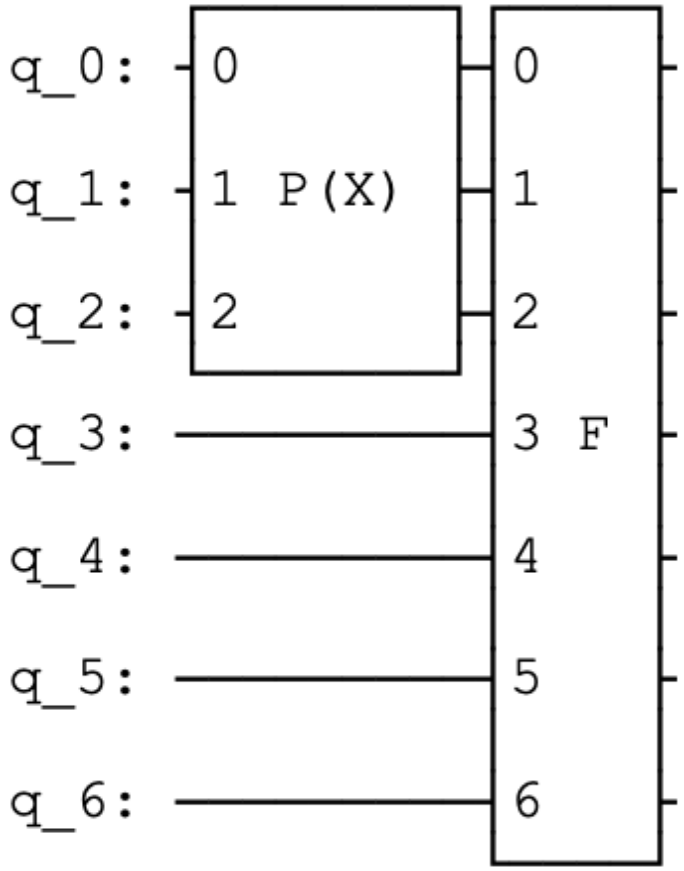
c_{approx} 越小越准，但是量子位m也需要相应调整。

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

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

量子线路 – 初始化

```
# construct A operator for QAE for the payoff function by
# composing the uncertainty model and the objective
num_qubits = european_call_objective.num_qubits
european_call = QuantumCircuit(num_qubits)
european_call.append(uncertainty_model,
range(num_uncertainty_qubits))
european_call.append(european_call_objective,
range(num_qubits))
```



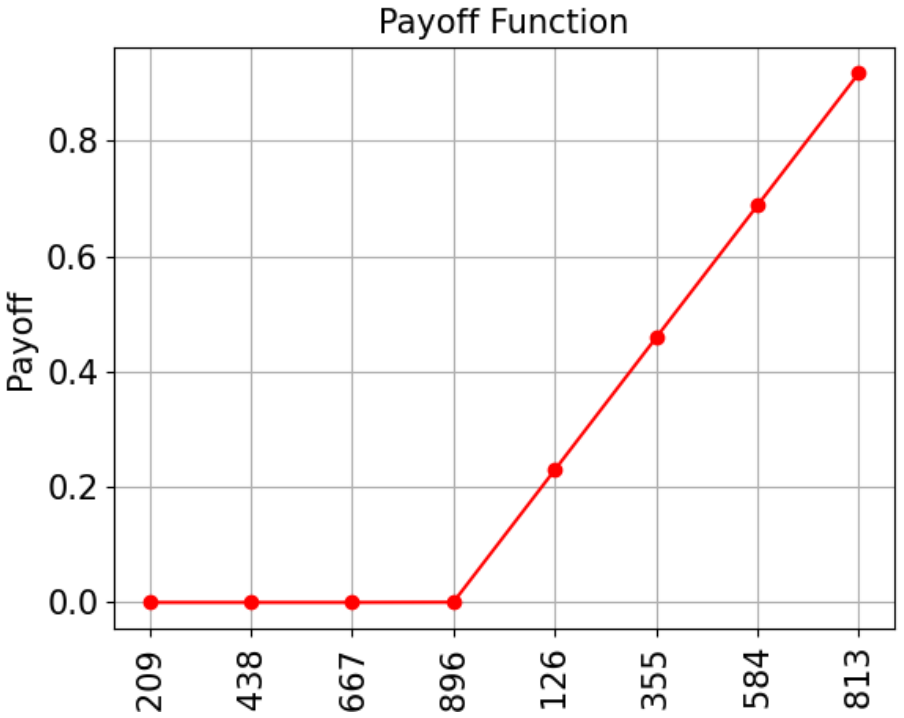
https://qiskit.org/documentation/finance/tutorials/03_european_call_option_pricing.html

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

损益函数

```
# plot exact payoff function (evaluated on the grid of the uncertainty model)
x = uncertainty_model.values
y = np.maximum(0, x - strike_price)
plt.plot(x, y, "ro-")
plt.grid()
plt.title("Payoff Function", size=15)
plt.xlabel("Spot Price", size=15)
plt.ylabel("Payoff", size=15)
plt.xticks(x, size=15, rotation=90)
plt.yticks(size=15)
plt.show()

# evaluate exact expected value (normalized to the [0, 1] interval)
exact_value = np.dot(uncertainty_model.probabilities, y)
exact_delta = sum(uncertainty_model.probabilities[x >= strike_price])
print("exact expected value:\t%.4f" % exact_value)
print("exact delta value:   \t%.4f" % exact_delta)
```



结果：

```
exact expected value:   0.1623
exact delta value:      0.8098
```

振幅估计

```
# 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=european_call,
    objective_qubits=[3],
    post_processing=european_call_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:        0.1623
Estimated value:    0.1665
Confidence interval:    [0.1616, 0.1715]
```

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/

Thank You