UROPS Project Presentation 3

Wang Zexin

Variance Reduction Techniques
of Monte Carlo methods in Financial Engineering

Quantitative Finance
National University of Singapore

February 2, 2017

Today's Agenda

1. Variance Reduction Techniques
   - Control Variates
   - Stratified Sampling
   - Importance Sampling

Changes due to different Python version

We are using Python 3.6 while the version in the book is Python 2.7
So here is a list of items to change

- print x now becomes print(x)
- dict.iteritems() now becomes dict.items()
- xrange now becomes range
- lambda (k, v) : (v, k) is no longer available
- instead we can only use: lambda x : (x[1], x[0])
- x / 2 is float division, while x // 2 is integer division

Control Variates - Mathematics

Our aim is to estimate $\mathrm{E}[Y_i]$
Under simulation, we use :

$$\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$$

(unbiased estimate)
Now we introduce a new variable X, with realizations X_i, for i from 1 to n
Define $Y_i(\lambda) = Y_i - \lambda(X_i - \mathrm{E}(X))$

$$\bar{Y}(\lambda) = \bar{Y} - \lambda(\bar{X} - \mathrm{E}(X)) = \frac{1}{n}\sum_{i=1}^{n}[Y_i - \lambda(X_i - \mathrm{E}(X))]$$

Control Variates - Unbiasedness and Consistency

The new estimate $\bar{Y}(\lambda)$ is unbiased and consistent.

Unbiasedness:

$$\mathrm{E}(\bar{Y}(\lambda)) = \mathrm{E}[\bar{Y} - \lambda(\bar{X} - \mathrm{E}(X))] = \mathrm{E}(\bar{Y}) - \lambda(\mathrm{E}(\bar{X}) - \mathrm{E}(X)) = \mathrm{E}(Y)$$

$$\begin{aligned} \textit{Consistency}: \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} Y_i(\lambda) &= \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n}[Y_i - \lambda(X_i - \mathrm{E}(X))] \\ &= \mathrm{E}[Y - \lambda(X - \mathrm{E}(X))] \\ &= \mathrm{E}(Y) \end{aligned}$$

Control Variates - Controlling Variance of estimate

$$\begin{aligned}\mathrm{Var}[Y_i(\lambda)] &= \mathrm{Var}[Y_i - \lambda(X_i - \mathrm{E}(X))] \\ &= \mathrm{Var}[Y_i - \lambda X_i] \\ &= \mathrm{Var}(Y_i) + \lambda^2 \mathrm{Var}(X_i) - 2\lambda \mathrm{Cov}(X_i, Y_i) \\ &= \sigma_Y^2 + \lambda^2 \sigma_X^2 - 2\lambda \sigma_X \sigma_Y \rho_{XY}\end{aligned}$$

In order to find the minimum variance by varying λ

Set $\frac{\partial \mathrm{Var}[Y_i(\lambda)]}{\partial \lambda} = 2\lambda\sigma_X^2 - 2\sigma_X\sigma_Y\rho_{XY}$ to 0:

$$\lambda^* = \frac{2\sigma_X\sigma_Y\rho_{XY}}{2\sigma_X^2} = \frac{\sigma_X\sigma_Y\rho_{XY}}{\sigma_X^2} = \frac{\mathrm{Cov}(X, Y)}{\mathrm{Var}(X)}$$

Control Variates - Controlling Variance of estimate

Compare the new variance with the old:

$$\begin{aligned}
\frac{\mathrm{Var}[Y_i - \lambda^*(X_i - \mathrm{E}(X))]}{\mathrm{Var}(Y)} &= \frac{\sigma_Y^2 + \lambda^{*2}\sigma_X^2 - 2\lambda^*\sigma_X\sigma_Y\rho_{XY}}{\sigma_Y^2} \\
&= 1 + \frac{\frac{\mathrm{Var}(X)(\mathrm{Cov}(X,Y))^2}{(\mathrm{Var}(X))^2} - \frac{2(\mathrm{Cov}(X,Y))^2}{\mathrm{Var}(X)}}{\sigma_Y^2} \\
&= 1 + \frac{\frac{\sigma_X^4\sigma_Y^2\rho_{XY}^2}{\sigma_X^4} - \frac{2\sigma_X^2\sigma_Y^2\rho_{XY}^2}{\sigma_X^2}}{\sigma_Y^2} \\
&= 1 + \frac{\sigma_Y^2\rho_{XY}^2 - 2\sigma_Y^2\rho_{XY}^2}{\sigma_Y^2} \\
&= 1 - \rho_{XY}^2
\end{aligned}$$

Conclusion from the theoretical results:
The stronger the correlation, the better the reduction in variance.

Choices of Control Variates

We can use several different random variables as control variates.

- Underlying asset prices
- Tractable options
- Bond prices
- Tractable dynamics

Control Variates - estimate for λ^*

In the case that it is not feasible to calculate using the probability distribution of X and Y, we should work λ^* out as an estimate.

$$\hat{\lambda}^* = \frac{\mathrm{Cov}(X, Y)}{\mathrm{Var}(X)} \approx \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

Implementation of Algorithm in pseudocode

- // pilot simulation to obtain correlation
- for i = 1 to n
- $generate(X_i, Y_i)$
- end for
- computer $\lambda^* = \frac{\mathrm{Cov}(X,Y)}{\mathrm{Var}(X)}$
- for i = 1 to n
- $generate(X_i, Y_i)$
- set $Y_i(\lambda) = Y_i + \lambda^*(X_i - \mathrm{E}[X])$
- end for
- Estimate : $\bar{Y}(\lambda) = \frac{1}{n}\sum_{i=1}^{n} Y_i(\lambda)$
- $\hat{\sigma}^2_{n,Y_\lambda} = \frac{1}{n-1}\sum(Y_i(\lambda) - \bar{Y}(\lambda))^2$
- $100(1-\alpha)\%C.I. = [\bar{Y}(\lambda) - \mathcal{Z}_{1-\alpha/2}\frac{\hat{\sigma}_{n,Y_\lambda}}{\sqrt{n}}, \bar{Y}(\lambda) + \mathcal{Z}_{1-\alpha/2}\frac{\hat{\sigma}_{n,Y_\lambda}}{\sqrt{n}}]$

Underlying asset prices for european call payoff

The underlying asset prices provide a source of control variates for its natural correlation with the derivative payoff.
The control variate estimator is formed like this:

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \lambda[S_i(T) - e^{rT}S(0)])$$

For European Call option, $Y = e^{-rT}(S(T) - K)^+$. However, one problem arises: as the strike price goes higher, the correlation of the underlying asset prices and the payoff decreases, this diminishes the effect of control variates upon variance reduction.

Underlying asset prices for european call payoff

```
def control_variate_underlying_assets_european_call(strikePrice):
    # parameters
    sigma = 0.25;   T = 1.0;   I = 50000;   S0 = 100;   r = 0.05
    # pilot simulation
    standardNormals = npr.standard_normal(I)
    stockPrices = S0*np.exp((r-sigma**2/2)*T+sigma*np.sqrt(T)*standardNormals)
    optionPayoffs = np.maximum(stockPrices - strikePrice, 0)
    covarianceMatrix = np.cov(stockPrices, optionPayoffs)
    lambdaStar = covarianceMatrix[0][1] / covarianceMatrix[0][0]
    # main simulation
    standardNormals = npr.standard_normal(I)
    stockPrices = S0*np.exp((r-sigma**2/2)*T+sigma*np.sqrt(T)*standardNormals)
    optionPayoffs = np.maximum(stockPrices - strikePrice, 0)
    newPayoffs = optionPayoffs - lambdaStar * (stockPrices - np.exp(r*T-sigma**2/2)*S0)
    optionPayoffs *= np.exp(-r*T)
    newPayoffs *= np.exp(-r*T)
    oldEstimate = np.average(optionPayoffs)
    oldVariance = np.var(optionPayoffs)
    newEstimate = np.average(newPayoffs)
    estimateVariance = np.var(newPayoffs)
    return (oldEstimate, oldVariance, newEstimate, estimateVariance)
```

Underlying asset prices for european call payoff

```
>>> control_variate_underlying_assets_european_call(110)
(7.9526125237289351, 235.57933182537033, 6.3854083266098609, 59.341699626324072)
>>> control_variate_underlying_assets_european_call(120)
(4.9382602410114398, 150.74744099925451, 3.8167887763856152, 56.071709217421791)
>>> control_variate_underlying_assets_european_call(130)
(3.0531363826324109, 96.72415721337812, 2.1864815097461197, 47.519775800219293)
>>> control_variate_underlying_assets_european_call(140)
(1.7885787488126001, 56.252990015991479, 1.2199308291753748, 34.33210462229821)
>>> control_variate_underlying_assets_european_call(150)
(1.0358230908781245, 32.671615929639692, 0.65592152857003061, 23.232528158440577)
>>> control_variate_underlying_assets_european_call(160)
(0.60697754065805898, 18.934380780940483, 0.36803106172068806, 14.937284997492556)
>>> control_variate_underlying_assets_european_call(180)
(0.18028884053911629, 5.3281139036213094, 0.10005118060612941, 4.8118277289914406)
>>> control_variate_underlying_assets_european_call(200)
(0.050316015313149949, 1.356498550346926, 0.013441171113046274, 1.3073205972274462)
```

From the output, we can see that as time goes, the proportion of variance reduced from including underlying asset prices as control variates decreases. However, as the proportion of variance reduction drops, the initial variance is also decreasing. Hence we can still comment that underlying asset price can serve as a good choice of control variate.

Tractable options for european call payoff

The payoff of other tractable options may also provide a source of control variates for its natural correlation with the derivative payoff.
The control variate estimator is formed like this:

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \lambda[(K' - S_i(T))^+ - \mathrm{E}((K' - S(T))^+)])$$

For European Call option, $Y = e^{-rT}(S(T) - K)^+$.
Here we are using European Put option payoff as the control variates, with K' being the put option strike price and $(K' - S(T))^+$ being the put option payoff.
The problem is that we do not know which strike price may give the strongest correlation between these two option payoffs, so I shall start with experimenting on some of the potential candidates.

Tractable options for european call payoff - programme

```
def control_variate_tractable_option_european_call(strikePrice, tractableOptionStrike):
    # parameters
    sigma = 0.25;   T = 1.0;   I = 50000;   S0 = 100;   r = 0.05
    #tractableOptionStrike = 2 * S0 - strikePrice
    # pilot simulation
    standardNormals = npr.standard_normal(I)
    stockPrices = S0*np.exp((r-sigma**2/2)*T+sigma*np.sqrt(T)*standardNormals)
    putOptionPayoffs = np.maximum(tractableOptionStrike - stockPrices, 0)
    optionPayoffs = np.maximum(stockPrices - strikePrice, 0)
    covarianceMatrix = np.cov(putOptionPayoffs, optionPayoffs)
    lambdaStar = covarianceMatrix[0][1] / covarianceMatrix[0][0]
    # main simulation
    standardNormals = npr.standard_normal(I)
    stockPrices = S0*np.exp((r-sigma**2/2)*T+sigma*np.sqrt(T)*standardNormals)
    putOptionPayoffs = np.maximum(tractableOptionStrike - stockPrices, 0)
    optionPayoffs = np.maximum(stockPrices - strikePrice, 0)
    averagePutOptionPayoffs = np.average(putOptionPayoffs) # use average as an estimate
    newPayoffs = optionPayoffs - lambdaStar * (putOptionPayoffs - averagePutOptionPayoffs)
    optionPayoffs *= np.exp(-r*T)
    newPayoffs *= np.exp(-r*T)
    oldEstimate = np.average(optionPayoffs)
    oldVariance = np.var(optionPayoffs)
    newEstimate = np.average(newPayoffs)
    estimateVariance = np.var(newPayoffs)
    return (oldEstimate, oldVariance, newEstimate, estimateVariance)
```

Tractable options for european call payoff - test

```
>>> control_variate_tractable_option_european_call(110, 100)
(8.0152852317087309, 239.47761839573417, 8.0152852317087291, 209.18361685580228)
>>> control_variate_tractable_option_european_call(110, 110)
(8.0123489595178619, 238.43231362018861, 8.0123489595178636, 188.00563587315739)
>>> control_variate_tractable_option_european_call(110, 120)
(7.947010058064901, 238.28768643443632, 7.9470100580648975, 160.67287120246266)
>>> control_variate_tractable_option_european_call(110, 130)
(8.0739346840195747, 238.41044499212592, 8.0739346840195729, 132.26033629286306)
>>> control_variate_tractable_option_european_call(110, 140)
(8.0747461764303647, 239.33744712772824, 8.0747461764303647, 110.82578158858529)
>>> control_variate_tractable_option_european_call(110, 150)
(7.9354475705588179, 232.9443571542966, 7.9354475705588188, 90.542181157034761)
>>> control_variate_tractable_option_european_call(110, 160)
(8.1098489196050707, 242.60516526050534, 8.1098489196050689, 82.462773649524365)
>>> control_variate_tractable_option_european_call(110, 170)
(8.0178945531485066, 240.15781627076689, 8.017894553148512, 73.462500473668783)
>>> control_variate_tractable_option_european_call(110, 180)
(8.0727220564922337, 245.06596160295058, 8.0727220564922408, 69.355579745932303)
>>> control_variate_tractable_option_european_call(110, 200)
(8.1132583702030576, 241.8884291326724, 8.1132583702030718, 62.418296218306928)
```

When strike price is low, the increase in the strike price of put option will cause a larger proportion of reduction in variance. However, as the strike price increases, the payoffs is becoming more similar to stock prices, and eventually it can only achieve the same reduction in variance as compared to using underlying asset prices when the strike is infinitely large.

Tractable options for european call payoff - test

```
>>> control_variate_tractable_option_european_call(150, 100)
(1.0640146781327136, 35.697354449138452, 1.0640146781327136, 35.165606234827358)
>>> control_variate_tractable_option_european_call(150, 110)
(1.0053614591036153, 31.584974815190524, 1.0053614591036151, 30.776534500636153)
>>> control_variate_tractable_option_european_call(150, 120)
(1.0606036718768947, 34.491430622469025, 1.0606036718768952, 33.096157309125253)
>>> control_variate_tractable_option_european_call(150, 130)
(1.0218853613098864, 31.70992152617659, 1.0218853613098862, 29.78462584652026)
>>> control_variate_tractable_option_european_call(150, 140)
(1.0196441318450431, 32.132016932181742, 1.0196441318450431, 29.370379085219721)
>>> control_variate_tractable_option_european_call(150, 150)
(1.0636643834985118, 34.454626030525063, 1.0636643834985118, 30.315447985642312)
>>> control_variate_tractable_option_european_call(150, 160)
(1.0764996368794038, 34.713562816476504, 1.076499636879404, 29.06780228792714)
>>> control_variate_tractable_option_european_call(150, 170)
(1.0255544717670815, 31.690719041250027, 1.0255544717670821, 25.449840668450985)
>>> control_variate_tractable_option_european_call(150, 180)
(1.011748793605298, 32.25751356353684, 1.0117487936052998, 25.177450984370932)
>>> control_variate_tractable_option_european_call(150, 200)
(1.0702178836059906, 34.385247393768424, 1.0702178836059901, 25.12168141370045)
```

When the strike price of call option is higher, the increase in the strike price of put option will cause a larger proportion of reduction in variance generally. Still there is one point which is the strike price of the call, where the reduction in variance seems to achieve its minimum.

Tractable options for european call payoff - test

Hence, this method of underlying variates using Tractable options can only work at most as good as the method using underlying asset prices as control variates.

Stratified Sampling - Mathematics

Stratified sampling refers broadly to any sampling mechanism that constrains the fraction of observations drawn from specific subsets (or strata) of the sample space.
Goal: estimate $\mathrm{E}[Y]$, $Y \in \mathbb{R}$
Dividing the sample space into K parts, with $A_1, \ldots, A_K$ being disjoint subsets of the real line for which $\mathrm{P}(Y \in \cup_i A_i) = 1$.
Then by the Bayes′ Theorem,

$$\mathrm{E}[Y] = \sum_{i=1}^{K} \mathrm{P}(Y \in A_i)\mathrm{E}[Y|Y \in A_i] = \sum_{i=1}^{K} p_i \mathrm{E}[Y|Y \in A_i]$$

Stratified Sampling - Mathematics

Simplest case: proportional sampling, ensuring $p_i = \mathrm{P}(Y \in A_i)$, the fraction of observations drawn from stratum A_i exactly matches theoretical probability.
Unbiased estimator of $\mathrm{E}(Y)$ is provided by:

$$\hat{Y} = \sum_{i=1}^{K} p_i \frac{1}{n_i} \sum_{j=1}^{n_i} Y_{ij} = \frac{1}{n} \sum_{i=1}^{K} \sum_{j=1}^{n_i} Y_{ij}$$

$$\mathrm{E}[\hat{Y}] = \mathrm{E}[\sum_{i=1}^{K} p_i \frac{1}{n_i} \sum_{j=1}^{n_i} Y_{ij}] = \sum_{i=1}^{K} \mathrm{P}(Y \in A_i)\mathrm{E}[Y|Y \in A_i] = \mathrm{E}[Y]$$

Stratified Sampling - Mathematics

Simplest case: proportional sampling, ensuring $p_i = \mathrm{P}(Y \in A_i)$, the fraction of observations drawn from stratum A_i exactly matches theoretical probability.
We shall generate a stratification variable X which take values in the union of the disjoint sets A_is. As supposedly the values of X are high correlated to the values of Y, in many cases, Y is a function of X. For example, if Y is the discounted payoff of European call option, X can be the discrete path of underlying asset prices.

Stratified Sampling - Algorithm

Let X be the discrete path of underlying asset prices, with Y being the European call payoff discounted to time 0. If X has k discrete prices on one path, $\Omega \in \mathbb{R}^k$ is the sample space for the X_is, where $\Omega = \cup_i A_i$ is the union of disjoint sets.
Here for the purpose of stratified sampling we should satisfy two conditions:

- $\mathbb{P}(X \in A_i)$ can be easily computed.
- $(Y|X \in A_i)$ Given $X \in A_i$, Y can be easily generated.

In order to further simplify the trouble of dividing Ω into disjoint subsets, we are going to use the stock prices half way as a condition to fully determine the distributions of A_i. The possible stock prices at time $\frac{T}{2}$ will be generated based on the Geometric Brownian Motion of the underlying stock prices. Given we are going to produce n equal probable subsets of Ω, the standard normal distribution can be sliced into n pieces to provide sources of generation.

Stratified Sampling - Algorithm

If the standard normal distribution is going to be splitted into 50 parts, with each starting from quantile 0% to quantile 2%, quantile 2% to quantile 4%, ..., quantile 98% to quantile 100%, we shall use their middle point in quantile to generate expectations of stock prices half way, namely quantile 1%, quantile 3%, ..., quantile 99%.
The next part of algorithm is to use the GBM formula to calculate the 50 expectations of the stock prices at time $\frac{T}{2}$.

$$S_T = S_0 \exp\{(r - \frac{1}{2}\sigma^2)T + \sigma\sqrt{T}z_{\alpha_i}\}$$

where z_{α_i} is the quantile of standard normal distribution at $(2i-1)\%$
With the expectations at these points, we can go on to generate for every group $\frac{\mathrm{I}}{50}$ number of stock prices at time T, and calculate the corresponding European call payoffs.

Stratified Sampling - Implementation

```
def stratified_sampling_european_call(strikePrice):
    # parameters
    sigma = 0.25;   T = 1.0;   I = 50000;   S0 = 100;   r = 0.05
    n = 50;   K = I / n;   totalEstimate = 0; pi = 1 / n; totalPayoffs = np.array([])
    # generate expectations at time T/2
    quantiles = [(2*x-1)/100 for x in range(1, n+1, 1)]
    standardNormals = scs.norm.ppf(quantiles, 0, 1)
    halfPrices = S0*np.exp((r-sigma**2/2)*T/2 + sigma * np.sqrt(T/2)*standardNormals)
    # generate I / n prices starting from the halfPrices
    for i in range(n):
        rand = np.random.standard_normal(I//n)
        finalPrices = halfPrices[i]*np.exp((r-sigma**2/2)*T/2+sigma*np.sqrt(T/2)*rand)
        payoffs = np.maximum(finalPrices - strikePrice, 0)
        totalPayoffs = np.append(totalPayoffs, payoffs)
        averagePayoffs = np.average(payoffs)
        totalEstimate += averagePayoffs * pi;
    totalEstimate *= np.exp(-r * T)
    totalPayoffsVariance = np.var(totalPayoffs)
    return (totalEstimate, totalPayoffsVariance)
```

Stratified Sampling - Test

```
>>> mcs_european_call(105)
(10.17348596976716, 321.36706018947638)
>>> mcs_european_call(105)
(9.950038084924852 2, 317.17163970797588)
>>> mcs_european_call(105)
(10.07555680171568 8, 326.29835611601544)
>>> mcs_european_call(105)
(10.02944689115126, 323.24111129999949)
>>> stratified_sampling_european_call(105)
(9.955572367387157 2, 312.29845152187380)
>>> stratified_sampling_european_call(105)
(9.986126949744395 6, 311.28731897417157)
>>> stratified_sampling_european_call(105)
(10.00876184768549, 311.67914756016950)
>>> stratified_sampling_european_call(105)
(9.935448289606890 5, 311.57455940891941)
```

Although not much, it is clear that there is variance reduction effect.

Importance Sampling - Mathematics

Thank You

E0007424@u.nus.edu