IEOR E4703: Monte-Carlo Simulation

Other Miscellaneous Topics and Applications of Monte-Carlo

Martin Haugh

Department of Industrial Engineering and Operations Research
Columbia University
Email: martin.b.haugh@gmail.com

Outline

Capital Allocation in Risk Management

An Application

Quasi-Monte-Carlo

Pricing Bermudan Options

Capital Allocation in Risk Management

Total loss given by $L = \sum_{i=1}^{n} L_i$.

Suppose we have determined the risk, $\varrho(L)$, of this loss.

The capital allocation problem seeks a decomposition, $AC_1, \ldots, AC_n$, such that

$$\varrho(L) \ = \ \sum_{i=1}^{n} AC_i \tag{1}$$

- AC_i is interpreted as the risk capital allocated to the i^{th} loss, L_i.

This problem is important in the setting of performance evaluation where we want to compute a risk-adjusted return on capital (RAROC).

e.g. We might set RAROC$_i$ $=$ Expected Profit$_i$ / Risk Capital$_i$

- must determine risk capital of each L_i in order to compute RAROC$_i$.

Capital Allocation

More formally, let $L(\boldsymbol{\lambda}) := \sum_{i=1}^{n} \lambda_i L_i$ be the loss associated with the portfolio consisting of λ_i units of the loss, L_i, for $i = 1, \ldots, n$.

Loss on actual portfolio under consideration then given by $L(\mathbf{1})$.

Let $\varrho(\cdot)$ be a risk measure on a space $\mathcal{M}$ that contains $L(\boldsymbol{\lambda})$ for all $\boldsymbol{\lambda} \in \Lambda$, an open set containing $\mathbf{1}$.

Then the associated risk measure function, $r_\varrho : \Lambda \to \mathbb{R}$, is defined by

$$r_\varrho(\boldsymbol{\lambda}) = \varrho(L(\boldsymbol{\lambda})).$$

We have the following definition ...

Capital Allocation Principles

Definition: Let r_ϱ be a risk measure function on some set $\Lambda \subset \mathbb{R}^n \setminus \mathbf{0}$ such that $\mathbf{1} \in \Lambda$.

Then a mapping, $f^{r_\varrho} : \Lambda \to \mathbb{R}^n$, is called a per-unit capital allocation principle associated with r_ϱ if, for all $\boldsymbol{\lambda} \in \Lambda$, we have

$$\sum_{i=1}^{n} \lambda_i f_i^{r_\varrho}(\boldsymbol{\lambda}) \;=\; r_\varrho(\boldsymbol{\lambda}). \tag{2}$$

- We then interpret $f_i^{r_\varrho}$ as the amount of capital allocated to one unit of L_i when the overall portfolio loss is $L(\boldsymbol{\lambda})$.
- The amount of capital allocated to a position of $\lambda_i L_i$ is therefore $\lambda_i f_i^{r_\varrho}$ and so by (2), the total risk capital is fully allocated.

The Euler Allocation Principle

Definition: If r_ϱ is a positive-homogeneous risk-measure function which is differentiable on the set Λ, then the per-unit Euler capital allocation principle associated with r_ϱ is the mapping

$$f^{r_\varrho} : \Lambda \to \mathbb{R}^n \ : \ f_i^{r_\varrho}(\boldsymbol{\lambda}) \ = \ \frac{\partial r_\varrho}{\partial \lambda_i}(\boldsymbol{\lambda}).$$

- The Euler allocation principle is a full allocation principle since a well-known property of any positive homogeneous and differentiable function, $r(\cdot)$ is that it satisfies
$$r(\boldsymbol{\lambda}) = \sum_{i=1}^{n} \lambda_i \frac{\partial r}{\partial \lambda_i}(\boldsymbol{\lambda}).$$
- The Euler allocation principle therefore gives us different risk allocations for different positive homogeneous risk measures.
- There are good economic reasons for employing the Euler principle when computing capital allocations.

Value-at-Risk and Value-at-Risk Contributions

Let $r_{VaR}^{\alpha}(\boldsymbol{\lambda}) = \mathsf{VaR}_{\alpha}(L(\boldsymbol{\lambda}))$ be our risk measure function.

Then subject to technical conditions can be shown that

$$
\begin{aligned}
f_i^{r_{VaR}^{\alpha}}(\boldsymbol{\lambda}) &= \frac{\partial r_{VaR}^{\alpha}}{\partial \lambda_i}(\boldsymbol{\lambda}) \\
&= \mathsf{E}\left[L_i \mid L(\boldsymbol{\lambda}) = \mathsf{VaR}_{\alpha}(L(\boldsymbol{\lambda}))\right], \quad \text{for } \; i = 1, \ldots, n. \qquad (3)
\end{aligned}
$$

Capital allocation, AC_i, for L_i is then obtained by setting $\boldsymbol{\lambda} = \mathbf{1}$ in (3).

Will now use (3) and Monte-Carlo to estimate the VaR contributions from each security in a portfolio.

- Monte-Carlo is a general approach that can be used for complex portfolios where (3) cannot be calculated analytically.

An Application: Estimating Value-at-Risk Contributions

Recall total portfolio loss is $L = \sum_{i=1}^{n} L_i$.

According to (3) with $\lambda = 1$ we know that

$$\begin{aligned} AC_i &= \mathsf{E}\left[L_i \mid L = \mathsf{VaR}_\alpha(L)\right] & (4) \\ &= \left.\frac{\partial\, \mathsf{VaR}_\alpha(\boldsymbol{\lambda})}{\partial \lambda_i}\right|_{\lambda=1} \\ &= w_i \frac{\partial\, \mathsf{VaR}_\alpha}{\partial w_i} & (5) \end{aligned}$$

for $i = 1, \ldots, n$ and where w_i is the number of units of the i^{th} security held in the portfolio.

Question: How might we use Monte-Carlo to estimate the VaR contribution, AC_i, of the i^{th} asset?

Solution: There are three approaches we might take:

First Approach: Monte-Carlo and Finite Differences

As AC_i is a (mathematical) derivative we could estimate it numerically using a finite-difference estimator.

Such an estimator based on (5) would take the form

$$\widehat{AC}_i := \frac{\mathsf{VaR}_\alpha^{i,+} - \mathsf{VaR}_\alpha^{i,-}}{2\delta_i} \tag{6}$$

where $\mathsf{VaR}_\alpha^{i,+}$ ($\mathsf{VaR}_\alpha^{i,-}$) is the portfolio VaR when number of units of the i^{th} security is increased (decreased) by $\delta_i w_i$ units.

Each term in numerator of (6) can be estimated via Monte-Carlo

- same set of random returns should be used to estimate each term.

What value of δ_i should we use? There is a bias-variance tradeoff but a value of $\delta_i = .1$ seems to work well.

This estimator will not satisfy the additivity property so that $\sum_i^n \widehat{AC}_i \neq \mathsf{VaR}_\alpha$

- but easy to re-scale estimated $\widehat{AC}_i$'s so that the property will be satisfied.

Second Approach: Naive Monte-Carlo

Another approach is to estimate (4) directly. Could do this by simulating N portfolio losses $L^{(1)}, \ldots, L^{(N)}$ with $L^{(j)} = \sum_{i=1}^{n} L_i^{(j)}$

- $L_i^{(j)}$ is the loss on the i^{th} security in the j^{th} simulation trial.

Could then set (why?) $AC_i = L_i^{(m)}$ where m denotes the VaR_α scenario, i.e. $L^{(m)}$ is the $\lceil N(1-\alpha) \rceil^{th}$ largest of the N simulated portfolio losses.

Question: Will this estimator satisfy the additivity property, i.e. will $\sum_i^n AC_i = \mathrm{VaR}_\alpha$?

Question: What is the problem with this approach? Will this problem disappear if we let $N \to \infty$?

A Third Approach: Kernel Smoothing Monte-Carlo

An alternative approach that resolves the problem with the second approach is to take a weighted average of the losses in the i^{th} security around the VaR_α scenario.

A convenient way to do this is via a kernel function.

In particular, say $K(x;h) := K\left(\frac{x}{h}\right)$ is a kernel function if it is:

1. Symmetric about zero
2. Takes a maximum at $x = 0$
3. And is non-negative for all x.

A simple choice is to take the triangle kernel so that

$$K(x;h) := \max\left(1 - \left|\frac{x}{h}\right|,\, 0\right).$$

A Third Approach: Kernel Smoothing Monte-Carlo

The kernel estimate of AC_i is then given by

$$\widehat{AC}_i^{ker} := \frac{\sum_{j=1}^N K\left(L^{(j)} - \hat{\mathsf{VaR}}_\alpha; h\right) L_i^{(j)}}{\sum_{j=1}^N K\left(L^{(j)} - \hat{\mathsf{VaR}}_\alpha; h\right)} \tag{7}$$

where $\widehat{\mathsf{VaR}}_\alpha := L^{(m)}$ with m as defined above.

One minor problem with (7) is that the additivity property doesn't hold. Can easily correct this by instead setting

$$\widehat{AC}_i^{ker} := \widehat{\mathsf{VaR}}_\alpha \frac{\sum_{j=1}^N K\left(L^{(j)} - \hat{\mathsf{VaR}}_\alpha; h\right) L_i^{(j)}}{\sum_{j=1}^N K\left(L^{(j)} - \hat{\mathsf{VaR}}_\alpha; h\right) L^{(j)}}. \tag{8}$$

Must choose an appropriate value of smoothing parameter, h.

Can be shown that an optimal choice is to set

$$h = 2.575\,\sigma\,N^{-1/5}$$

where $\sigma = \mathsf{std}(L)$, a quantity that we can easily estimate.

When Losses Are Elliptically Distributed

If $L_1, \ldots, L_N$ have an elliptical distribution then it may be shown that

$$AC_i \ = \ \mathsf{E}[L_i] + \frac{\mathsf{Cov}(L, L_i)}{\mathsf{Var}(L)} \left(\mathsf{VaR}_\alpha(L) - \mathsf{E}[L]\right). \tag{9}$$

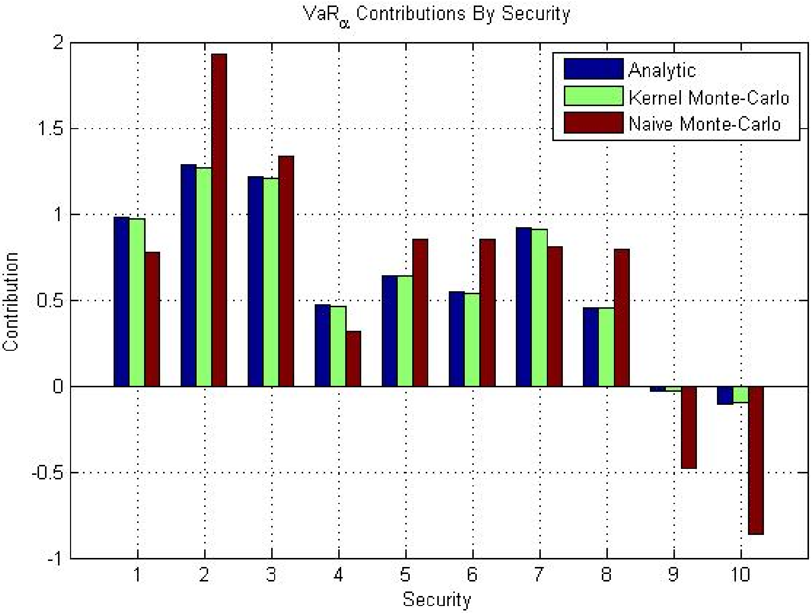
In numerical example below, we assume 10 security returns are elliptically distributed. In particular, losses satisfy $(L_1, \ldots, L_n) \sim \mathsf{MN}_n(\mathbf{0}, \mathbf{\Sigma})$.

Other details include:

1. First eight securities were all positively correlated with one another.
2. Second-to-last security uncorrelated with all other securities.
3. Last security had a correlation of -0.2 with the remaining securities.
4. Long position held on each security.

Estimated $VaR_{\alpha=.99}$ contributions of the securities displayed in figure below

- last two securities have a negative contribution to total portfolio VaR
- also note how inaccurate the “naive” Monte-Carlo estimator is
- but kernel Monte-Carlo is very accurate!


VaR_{α} Contributions By Security
Analytic
Kernel Monte-Carlo
Naive Monte-Carlo
Contribution
Security
2
1.5
1
0.5
0
-0.5
-1
1
2
3
4
5
6
7
8
9
10

Quasi Monte-Carlo Methods

Consider problem of computing an integral over the d-dimensional unit cube.

A principle advantage of Monte Carlo is the order $1/\sqrt{n}$ convergence rate

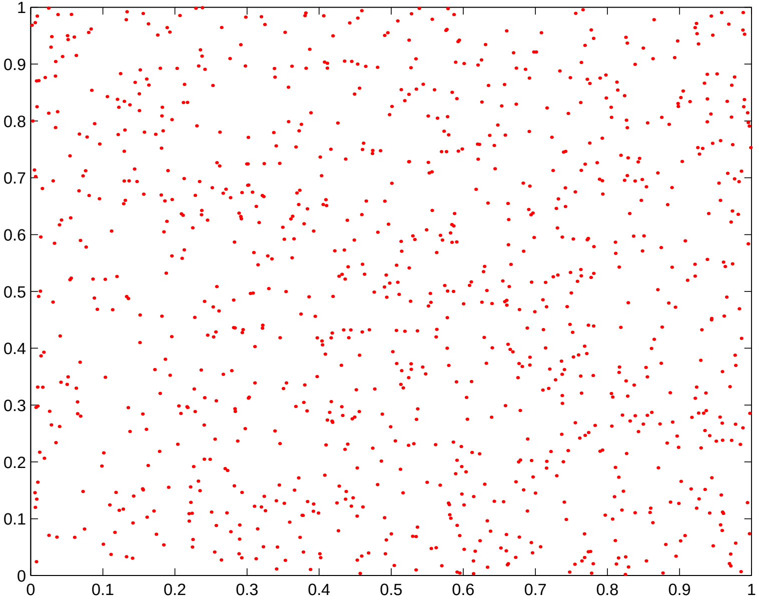
- which is independent of d.

In contrast, standard numerical integration schemes based on a rectangular grid of points converge as $1/(n^{2/d})$.

But many interesting problems are high-dimensional so Monte Carlo simulation can provide a significant computational advantage.

But ... a sample of uniformly distributed points in the d-dimensional unit cube covers the cube **inefficiently**

- see figure on next slide for example



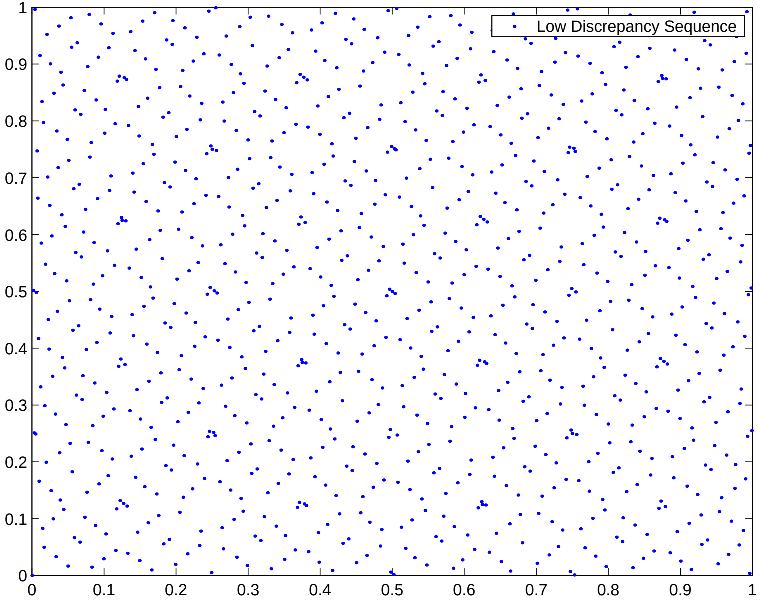
Uniform Random Samples in $[0,1]^2$

Low Discrepancy Sequences

A d-dimensional low discrepancy sequence (LDS) is a deterministic sequence of points in the d-dimensional unit cube that fills the cube efficiently, i.e. it has a **low discrepancy**.

This low discrepancy property results in a convergence rate of $(\log n)^d/n$, implying in particular that they can often be much more effective than Monte Carlo methods.

An example of a 2-dimensional LDS is plotted in figure on next slide.



Dimensional Low Discrepancy Points in $[0, 1]^2$

Low Discrepancy Sequences

It is clear there is nothing random about low discrepancy points.

Hence the term Quasi Monte Carlo often used to refer to approaches that use LDS as an alternative to standard Monte Carlo methods.

If the objective is to calculate

$$\theta \ := \ \mathsf{E}[f(U_1, \ldots, U_d)] \ = \ \int_{[0,1)^d} f(x) \ dx$$

then we can estimate it with

$$\hat{\theta} \ := \ \frac{1}{n} \sum_{i=1}^{n} f(x_i)$$

where $x_1, \ldots, x_n$ are a d-dimensional sequence of low-discrepancy points from the unit hypercube, $[0,1)^d$.

Discrepancy

Definition: Given a collection, $\mathcal{A}$, of subsets of $[0,1)^d$, we define the discrepancy of a set of points $\{x_1, \ldots, x_n\}$ *relative* to $\mathcal{A}$ as

$$D(x_1, \ldots, x_n;\ \mathcal{A}) \ := \ \sup_{A \in \mathcal{A}} \left| \frac{\#\{x_i \in A\}}{n} \ - \ \text{vol}(A) \right|$$

where vol(A) denotes the volume of A.

- The discrepancy of a sequence, $D(x_1, \ldots, x_n)$, is then obtained by taking $\mathcal{A}$ to be the collection of rectangles of the form

$$\prod_{j=1}^{d} [u_j, v_j), \quad 0 \le u_j < v_j \le 1. \tag{10}$$

- The star discrepancy, $D^*(x_1, \ldots, x_n)$ is obtained by taking $u_j = 0$ in (10).
- Low discrepancy sequences are sequences of points for which the star discrepancy is small (in a sense we do not define here).

Low Discrepancy Sequences

Recall that if we wish to generate IID d-dimensional vectors of $U(0,1)$ random variables then we can simply:

1. Generate a sequence of $U_1, \ldots, U_d, U_{d+1}, \ldots, U_{2d}, \ldots$ of IID uniforms
2. Take $(U_1, \ldots, U_d),\ (U_{d+1}, \ldots, U_{2d}), \ldots$ as our sequence of IID vectors.

This is **not** the case with low discrepancy sequences: the dimensionality, d, is a key component in constructing the low discrepancy points.

This dependence on the dimensionality of the problem is clear when we realize the need to specify the dimensionality of the problem before tackling any given problem.

Question: How might you evaluate an expectation

$$\theta := \mathsf{E}[f(\mathbf{X})]$$

where $\mathbf{X}$ is a d-dimensional multivariate normal random vector? Consider first the case where the d normal random variables are independent.

Advantages of Low Discrepancy Sequences

1. Their asymptotic convergence properties are superior to those of Monte Carlo simulation and their performance is often dramatically superior in practice.

2. The number of points, n, need not be known in advance. This is a property shared with Monte Carlo but not with numerical integration techniques that are based on regular grids.

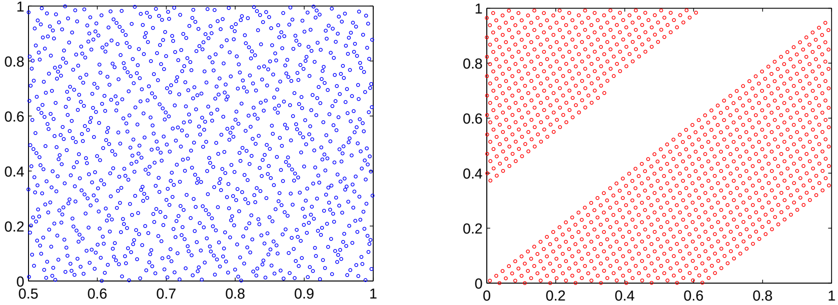
Disadvantages of Low Discrepancy Sequences

1. For a fixed sample size, n, there is no guarantee that low discrepancy sequences will outperform Monte Carlo simulation.

e.g. Many popular LDS cover the initial coordinates, (x_1, x_2), more or less uniformly.

But they do not cover the final coordinates, (x_{d-1}, x_d), in a sufficiently uniform manner (until n is sufficiently large).

Figures (a) and (b) display 2-dimensional projections of the first 2 and final 2 coordinates, respectively, of the first $1,000$ points of the 32-dimensional Halton sequence.



(a) 1^{st} 2 dimensions of 32-dimensional Halton sequence

(b) Last 2 dimensions of the sequence.

Disadvantages of Low Discrepancy Sequences

e.g. ctd. Clear the $1,000$ points fill first 2 dimensions much more uniformly than the final two dimensions.

This behavior is not atypical and can lead to very inaccurate results if:

1. An insufficient number of points is used
2. Function being integrated depends to a significant extent on arguments of the higher dimensions
   - often the case when pricing derivative securities, for example
   - might then be necessary to raise n to an unsatisfactorily high level.

There are various methods available for counteracting this problem, including the use of leaped and scrambled sequences.

Also possible to use the **Brownian bridge** construction and / or **stratified sampling** techniques to overcome some of these problems.

Disadvantages of Low Discrepancy Sequences

2. Since LDS are deterministic, confidence intervals (CIs) are not readily available
   - so difficult to tell whether or not an estimate is sufficiently accurate.

There are now methods available to **randomize** LDS, however, and so approximate CIIs can be constructed.

One method is to generate a uniform random vector, $U_1 \in R^d$, and then set

$$\hat{\theta}_1 \ := \ \frac{1}{n} \sum_{i=1}^{n} f\left((x_i + U_1) \bmod 1\right)$$

where the mod 1 operation is applied separately to each of the d coordinates.

Note that $\hat{\theta}_1$ is now (why?) an unbiased estimator of θ.

Can repeat this m times to obtain IID sample $\hat{\theta}_1, \ldots, \hat{\theta}_m$ which can then be used to construct CIs for θ.

Quasi Monte-Carlo

More care needed when applying LDS than when applying Monte Carlo.

But LDS often produce significantly better estimates

- therefore worth considering for applications where computational requirements are very demanding.

In practice LDS are often applied in very high-dimensional problems when traditional Monte-Carlo might be too slow

e.g. pricing of mortgage-backed securities.

But precisely in high-dimensional applications where most care is needed when using LDS.

Also worth noting that careful use of **variance reduction** techniques can often narrow the gap significantly between the performance of LDS and Monte-Carlo.

Theory underlying LDS is based on number theory and abstract algebra and is **not probabilistic**.

An Application: Pricing MBS

We consider the pricing of a principal-only (PO) and interest-only (IO) MBS.

Underlying mortgage pool has the following characteristics:

- Initial balance of the pool is $\$10m$
- Each underlying mortgage has $T = 30$ years to maturity
- Each mortgage makes monthly payments
- Average coupon rate is 10%
- But service and guaranty fees of $.5\%$ yield a pass-through rate of

$$10\% - .5\% \;=\; 9.5\%$$

.

We need a prepayment model and a **term-structure model.**

A Prepayment Model (Richard and Roll 1989)

We assume

$$CPR_k = RI_k \times AGE_k \times MM_k \times BM_k \tag{11}$$

where:

- RI_k is the refinancing incentive with

  $$RI_k := .28 \; + \; .14 \; \tan^{-1}\left(-8.57 \; + \; 430\left(WAC - r_k(10)\right)\right) \tag{12}$$

  where $r_k(10)$ is the prevailing 10-year spot rate at time k.

- $AGE_k = \min\left(1, \; t/30\right)$ is the seasoning multiplier.

- MM_k is the monthly multiplier with, for example,

  $$x := [.94 \; .76 \; .74 \; .95 \; .98 \; .92 \; .98 \; 1.1 \; 1.18 \; 1.22 \; 1.23 \; .98].$$

  Then $MM_k = x(5)$ if k falls in May or $MM_k = x(2)$ if k falls in February etc.

- $BM_k = .3 + .7M_{k-1}/M_0$ is the burnout multiplier where $M_k =$ remaining principal balance at time k.

Choosing a Term Structure Model

Also need to specify a term-structure model in order to fully specify the model.

The term structure model will be used to:

(i) discount all of the MBS cash-flows in the usual martingale pricing framework

(ii) to compute the refinancing incentive according to (11) and (12).

Will assume a Vasicek model for the term structure so that

$$dr_t \;=\; \alpha(\mu - r_t)\; dt + \sigma\; dW_t$$

where $r_0 = .08$, $\alpha = 0.2$, $\mu = 0.1$, $\sigma = .05$ and W_t is a Q-Brownian motion.

With this choice we can compute $r_t(10)$ analytically.

Monte-Carlo Prices of IO and PO MBS

Used $N = 20,000$ paths.

Approximate 95% CI for the IO MBS was

$$[\$4.009m,\ \$4.019m].$$

Approximate 95% CI for the PO MBS was

$$[\$6.225m,\ \$6.279m].$$

Question: Can you give any intuition for why the approximate 95% confidence interval for the PO is much wider than the corresponding interval for the IO?

Quasi Monte-Carlo Prices of IO and PO MBS

Pricing IO and PO securities using $20,000$ points of a 360-dimensional LDS we obtain price estimates of $\$4.011m$ and $\$6.257m$, respectively.

Both of these estimates are inside the 95% Monte-Carlo CIs

- thereby suggesting that the $20,000$ points is probably sufficient.

If instead we use 10 blocks of $10,000$ low discrepancy points where we randomize each block, then we obtain

$$\begin{aligned} 95\% \text{ CI for IO Price} &= [\$4.013m,\ \$4.016m] \\ 95\% \text{ CI for PO Price} &= [\$6.252m,\ \$6.256m] \end{aligned}$$

Note that these CIs are **inside** the CIs that were obtained using Monte-Carlo.

Of course five times as many points were used to obtain these LDS-based CIs but they are narrower than the Monte-Carlo based CIs by a factor greater than $\sqrt{5}$.

Pricing Bermudan Options

The general Bermudan option pricing problem at time $t = 0$ is to compute

$$V_0 \;:=\; \sup_{\tau \in \mathcal{T}} \mathsf{E}_0^{\mathcal{Q}} \left[\frac{h_\tau}{B_\tau} \right] \tag{13}$$

- $\mathcal{T} = \{0 \leq t_1, \ldots, t_n = T\}$ is the set of possible exercise dates
- B_t is the value of the cash account at time t
- $h_t = h(X_t)$ is the payoff function if the option is exercised at time t
- X_t represents the time t (vector) value of the state variables in the model.

e.g. In the case of a Bermudan swaption in the LIBOR market model X_t would represent the time t value of the various forward LIBOR rates.

Value Iteration and Q-Value Iteration

In theory (13) is easily solved using value iteration:

$$\begin{aligned} V_T &= h(X_T) \quad \text{and} \\ V_t &= \max\left(h(X_t),\ \mathsf{E}_t^{\mathcal{Q}}\left[\frac{B_t}{B_{t+1}} V_{t+1}(X_{t+1})\right]\right). \end{aligned}$$

- option price then given by $V_0(X_0)$.

Value iteration is one of the main approaches for solving DPs.

An alternative to value iteration is Q-value iteration.

The Q-value function is the value of the option conditional on it **not** being exercised today, i.e. the Q-value is the continuation value of the option

$$Q_t(X_t) = \mathsf{E}_t^{\mathcal{Q}}\left[\frac{B_t}{B_{t+1}} V_{t+1}(X_{t+1})\right]. \tag{14}$$

Q-Value Iteration

Option value at time $t+1$ then given by

$$V_{t+1}(X_{t+1}) \;=\; \max(h(X_{t+1}), Q_{t+1}(X_{t+1})) \tag{15}$$

so that if we substitute (15) into (14) we obtain

$$Q_t(X_t) = \mathsf{E}_t^{\mathcal{Q}}\left[\frac{B_t}{B_{t+1}}\max\left(h(X_{t+1}), Q_{t+1}(X_{t+1})\right)\right]. \tag{16}$$

- this is Q-value iteration.

If X_t is high dimensional, then both value iteration and Q-value iteration are impractical

- this is the so-called curse of dimensionality.

But we can perform an approximate and efficient version of Q-value iteration using cross-path regressions.

Cross-Path Regressions

First step is to choose a set of basis functions, $\phi_1(X), \ldots, \phi_m(X)$.

These basis functions define a linear architecture that will be used to approximate the Q-value functions.

In particular, will approximate $Q_t(X_t)$ with

$$\widetilde{Q}_t(X_t) := r_1^{(t)}\phi_1(X_t) \ + \ \ldots \ + \ r_m^{(t)}\phi_m(X_t)$$

where $r_t := (r_1^{(t)}, \ldots, r_m^{(t)})$ is a vector of time t parameters.

Cross-Path Regression for Approximate Q-Value Iteration

```
generate $N$ independent paths of $X_t$ for $t = 1, ..., T$
set $\widetilde{Q}_T(X_T^i) = 0$ for all $i = 1$ to $N$
for $t = T - 1$ Down to $1$
    Estimate $r_t = (r_1^{(t)}, \ldots, r_m^{(t)})$
    set $\widetilde{Q}_t(X_t^i) \ = \ \sum_k r_k^{(t)} \phi_k(X_t^i)$   for all $i$
end for
set $\widetilde{V}_0(X_0) = \max\Big(h(X_0),\ \widetilde{Q}_0(X_0)\Big)$
```

Cross-Path Regression for Approximate Q-Value Iteration

Two steps require further explanation:

1. Estimate r_t by regressing $\alpha \max\left(h(X_{t+1}),\ \widetilde{Q}(X_{t+1})\right)$ on $(\phi_1(X_t), \ldots, \phi_m(X_t))$ where $\alpha := B_t/B_{t+1}$ is the discount factor for period $[t,\ t+1]$.

   Have N observations for this regression with N typically ≈ 10k to 50k.

2. Since all N paths have the same starting point, X_0, can estimate $\widetilde{Q}_0(X_0)$ by averaging and discounting $\widetilde{Q}_1(\cdot)$ evaluated at the N successor points of X_0.

Obviously more details are required to fully specify the algorithm.

Constructing a Lower Bound on the True Option Price

Quite common in practice to use an alternative estimate, $\underline{V}_0$, of V_0.

$\underline{V}_0$ obtained by simulating the exercise strategy that is defined implicity by the sequence of Q-value function approximations.

That is, define

$$\widetilde{\tau} \ := \ \min\{t \in \mathcal{T} : \widetilde{Q}_t \leq h_t\}$$

and

$$\underline{V}_0 \ := \ \mathsf{E}_0^{\mathcal{Q}}\left[\frac{h_{\widetilde{\tau}}}{B_{\widetilde{\tau}}}\right].$$

Question: Why is $\underline{V}_0$ is an unbiased lower bound on V_0?

Question: Can you guess why we prefer to do an approximate Q-value iteration instead of an approximate value-iteration?

Constructing a Lower Bound on the True Option Price

These algorithms perform extremely well on realistic high-dimensional problems.

There has also been considerable theoretical work explaining why this is so.

Quality of $\underline{V}_0$ can be explained in part by noting that exercise errors are never made as long as $Q_t(\cdot)$ and $\widetilde{Q}_t(\cdot)$ lie on the **same** side of the optimal exercise boundary.

This means that it is possible to have large errors in $\widetilde{Q}_t(\cdot)$ that do not impact the quality of $\underline{V}_0$!

Computing Upper Bounds on Bermudan Option Prices

For an arbitrary super-martingale, π_t, value of the Bermudan option satisfies

$$
\begin{aligned}
V_0 = \sup_{\tau \in \mathcal{T}} \mathsf{E}_0^{\mathcal{Q}} \left[\frac{h_\tau}{B_\tau} \right] &= \sup_{\tau \in \mathcal{T}} \mathsf{E}_0^{\mathcal{Q}} \left[\frac{h_\tau}{B_\tau} - \pi_\tau + \pi_\tau \right] \\
&\leq \sup_{\tau \in \mathcal{T}} \mathsf{E}_0^{\mathcal{Q}} \left[\frac{h_\tau}{B_\tau} - \pi_\tau \right] + \sup_{\tau \in \mathcal{T}} \mathsf{E}_0^{\mathcal{Q}} \left[\pi_\tau \right] \\
&\leq \sup_{\tau \in \mathcal{T}} \mathsf{E}_0^{\mathcal{Q}} \left[\frac{h_\tau}{B_\tau} - \pi_\tau \right] + \pi_0 \\
&\leq \mathsf{E}_0^{\mathcal{Q}} \left[\max_{t \in \mathcal{T}} \left(\frac{h_t}{B_t} - \pi_t \right) \right] + \pi_0 \qquad (17)
\end{aligned}
$$

where the second inequality follows from the optional sampling theorem.

Taking the infimum over π_t on rhs of (17) implies

$$
V_0 \;\leq\; U_0 \;:=\; \inf_{\pi} \; \mathsf{E}_0^{\mathcal{Q}} \left[\max_{t \in \mathcal{T}} \left(\frac{h_t}{B_t} - \pi_t \right) \right] + \pi_0. \qquad (18)
$$

Computing Upper Bounds on Bermudan Option Prices

But it is known(!) that V_t/B_t is itself a super-martingale so

$$U_0 \leq \mathsf{E}_0^{\mathcal{Q}} \left[\max_{t \in \mathcal{T}} \left(h_t/B_t - V_t/B_t \right) \right] + V_0.$$

Since $V_t \geq h_t$ for all t, can conclude that $U_0 \leq V_0$.

Therefore, $V_0 = U_0$, and equality is attained when $\pi_t = V_t/B_t$.

Therefore an upper bound on V_0 can be constructed simply by evaluating rhs of (17) for any super-martingale, π_t.

And if super-martingale satisfies $\pi_t \geq h_t/B_t$, then V_0 bounded above by π_0.

Computing Upper Bounds on Bermudan Option Prices

If $\pi_t = V_t/B_t$ then upper bound on the rhs of (17) equals the true price, V_0.

Suggests a tight upper bound can be obtained by using an accurate approximation, $\widetilde{V}_t$, to define π_t.

One possibility is to define π_t as a martingale:

$$\begin{aligned} \pi_0 &= \widetilde{V}_0 && (19) \\ \pi_{t+1} &= \pi_t + \frac{\widetilde{V}_{t+1}}{B_{t+1}} - \frac{\widetilde{V}_t}{B_t} - \mathsf{E}_t\left[\frac{\widetilde{V}_{t+1}}{B_{t+1}} - \frac{\widetilde{V}_t}{B_t}\right]. && (20) \end{aligned}$$

Let $\overline{V}_0$ denote the upper bound from (17) corresponding to choice of super-martingale in (19) and (20).

Then easy to see the upper bound explicitly given by

$$\overline{V}_0 = \widetilde{V}_0 + \mathsf{E}_0^{\mathcal{Q}}\left[\max_{t\in\mathcal{T}}\left(\frac{h_t}{B_t} - \frac{\widetilde{V}_t}{B_t} + \sum_{j=1}^{t}\mathsf{E}_{j-1}^{\mathcal{Q}}\left[\frac{\widetilde{V}_j}{B_j} - \frac{\widetilde{V}_{j-1}}{B_{j-1}}\right]\right)\right]. \qquad (21)$$