Quasi-Monte Carlo

David Fengyu Ke, Santiago Mok, and
John Lee

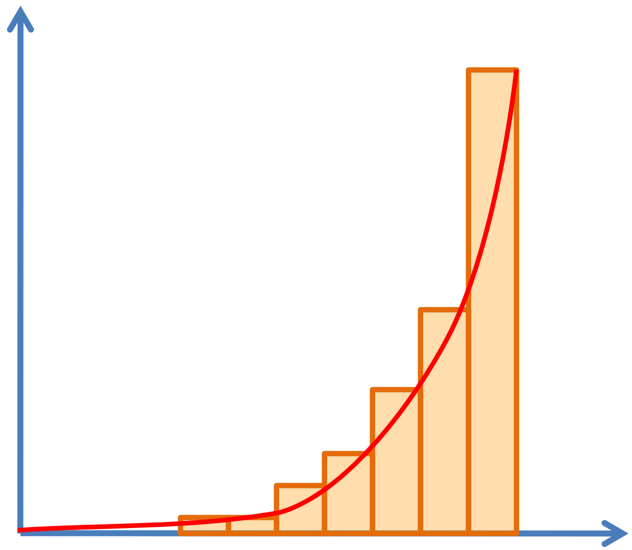
Overview

- Why Monte Carlo anyways?
- Quasi-Monte Carlo and low discrepancy
- When to use Quasi-Monte Carlo
- Example
- Generating Halton sequences

Why Monte Carlo?

- Traditional numerical integration uses regular grids to maximize accuracy:
  - Example:

$$\int_{x \in [0,1]^n} e^{Ax} dx$$



Why Monte Carlo?

- Traditional numerical integration has trouble in high dimensions – the grids do not scale well:
  - Example:

$$\int_{x\in[0,1]^n} e^{Ax}\,dx$$

- Picking uniform points in the hypercube:
  - In 1 dimension (n=1), pick 5 uniform points
  - In 2 dimensions (n=2), pick 5^2= 25 points
  - In 5 dimensions (n=5), pick 5^5= 3125 points?!

Integration is difficult to do for n > 4 !

Why Monte Carlo?

- Monte Carlo estimates the integral using samples:
  - Example:

$$\int_{x\in[0,1]^n} e^{Ax}dx \approx \sum_{i=1}^{N} e^{Ax_i}$$

    - Use random samples x_i to estimate the integral

Accuracy is independent of the dimension!

$$\text{Accuracy} \propto \frac{1}{\sqrt{N}}$$

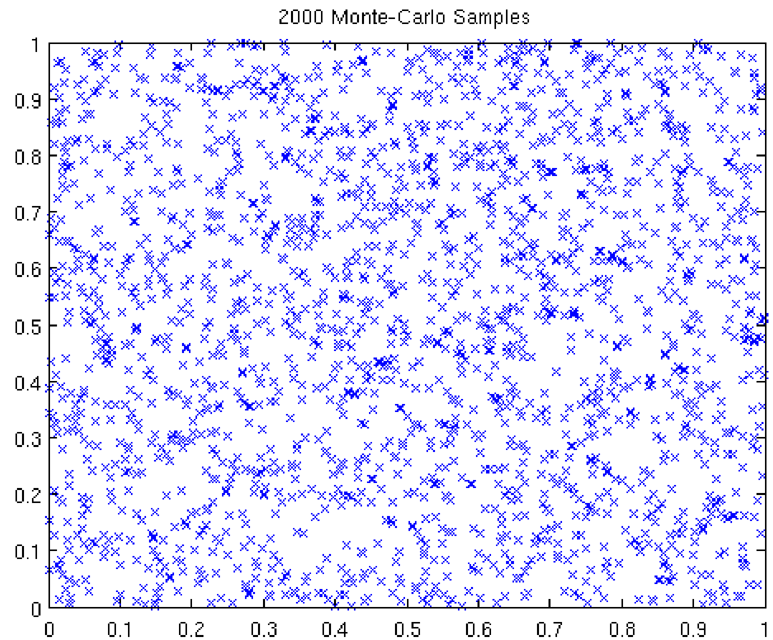
Quasi-Monte Carlo

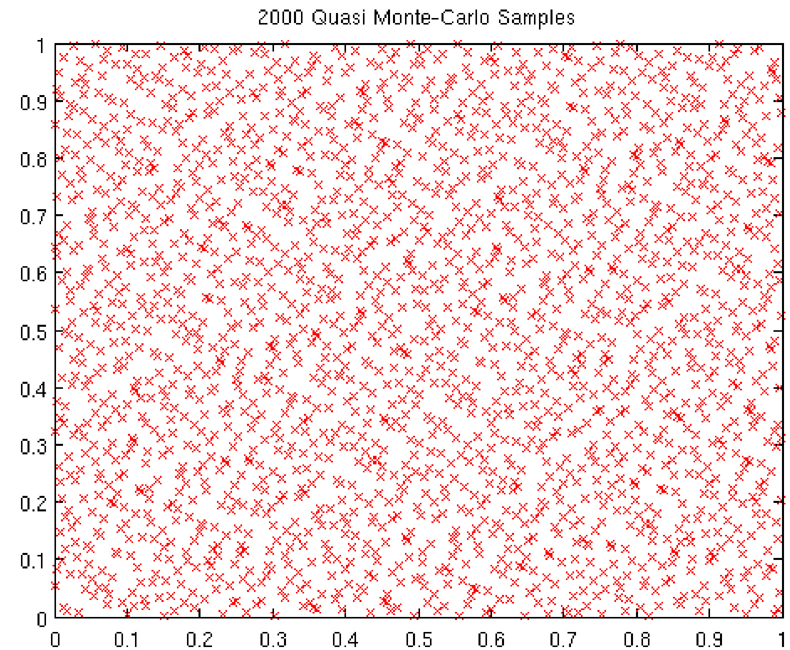
- Combines the best parts of traditional numerical integration with best parts of Monte-Carlo
  - Regularity of traditional numerical integration
    - Points are more regular than monte-carlo points
  - Accuracy that is independent of dimension
    - Don't need a ton of grid points to get accurate results*
- Accuracy is enhanced:

$$\text{Accuracy} \propto \frac{1}{N}$$

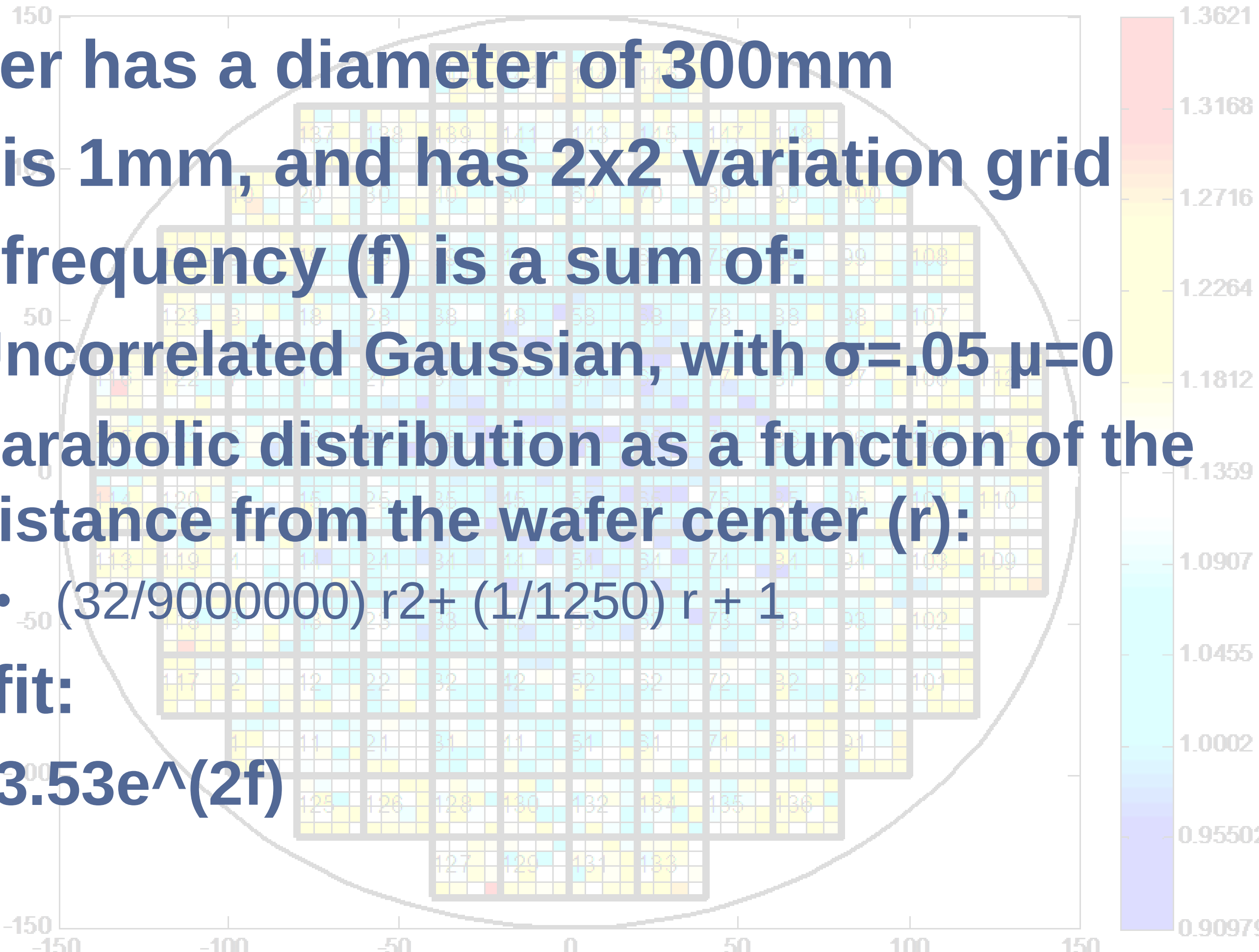
Quasi-Monte Carlo: How?

- Uses low discrepancy:
  - Increases the regularity of monte-carlo
  - Less ‘clumping’, ‘whitespaces’ -> less **discrepancy**





Example: Estimating Wafer Profit



- **Wafer has a diameter of 300mm**
- **Die is 1mm, and has 2x2 variation grid**
- **Die frequency (f) is a sum of:**
  - **Uncorrelated Gaussian, with σ=.05 μ=0**
  - **Parabolic distribution as a function of the distance from the wafer center (r):**
    - (32/9000000) r2+ (1/1250) r + 1
- **Profit:**
  - **13.53e^(2f)**

Example: Estimating Wafer Profit

Calculated Average Profit (tens of thousands of dollars) Dimensions: 2368

	5	10	15	20	25	30	35	40	45	50	10000
MC	1.5430	1.5385	1.5375	1.5420	1.5419	1.5400	1.5411	1.5402	1.5403	1.5405	1.5405
QMC	1.5360	1.5361	1.5368	1.5373	1.5376	1.5381	1.5387	1.5391	1.5396	1.5401	1.7170

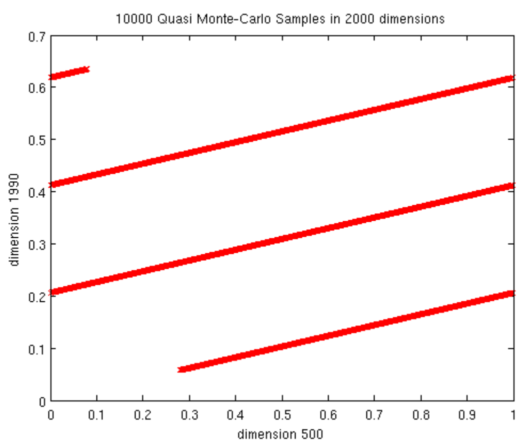
Percentage Error (%) Dimensions: 2368

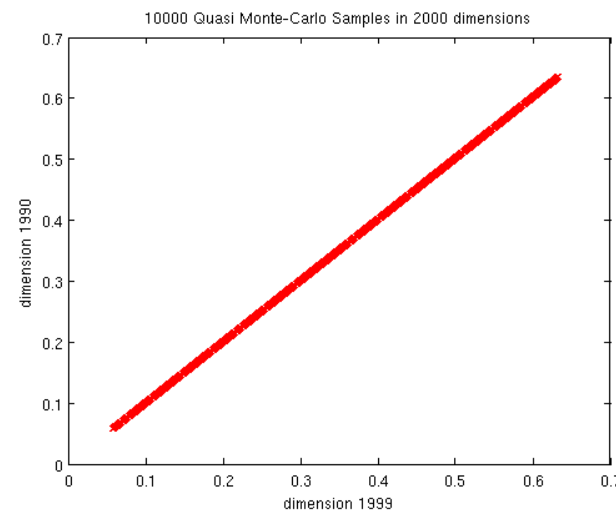
	5	10	15	20	25	30	35	40	45	50	10000
MC	0.1623	0.1298	0.1947	0.0974	0.0909	0.0325	0.0389	0.0195	0.0130	0	0
QMC	13.844	9.9555	8.0301	6.0237	5.2881	4.0893	3.1108	2.4045	2.1099	1.5655	0

QMC performs worse than MC!

Why does QMC perform worse?

- Quasi-Monte Carlo suffers from the same problem as conventional integration: a minimum number of samples is needed to fill the space
  - The number of samples is proportional to 2^n
    - For 1 dimension, approx 2 points
    - For 5 dimensions, approx 32 points
    - For 2000 dimensions, approx ∞ points

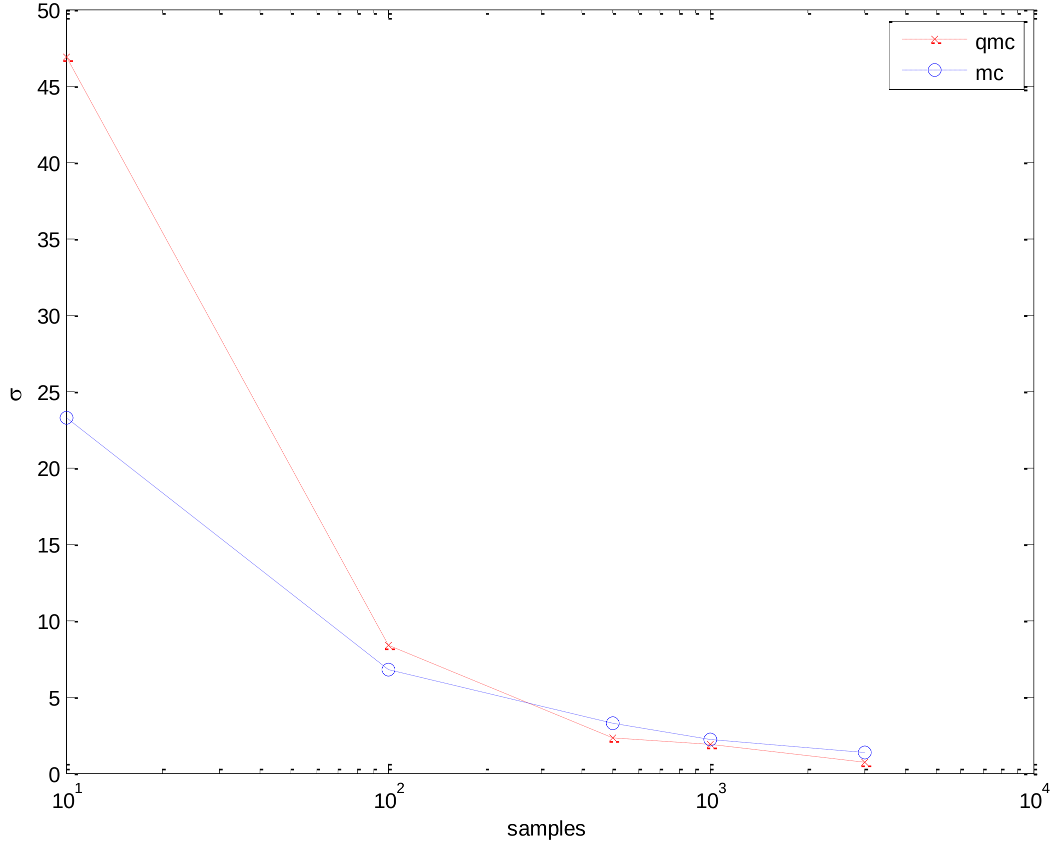




Example QMC vs. MC

Dimensions: 35

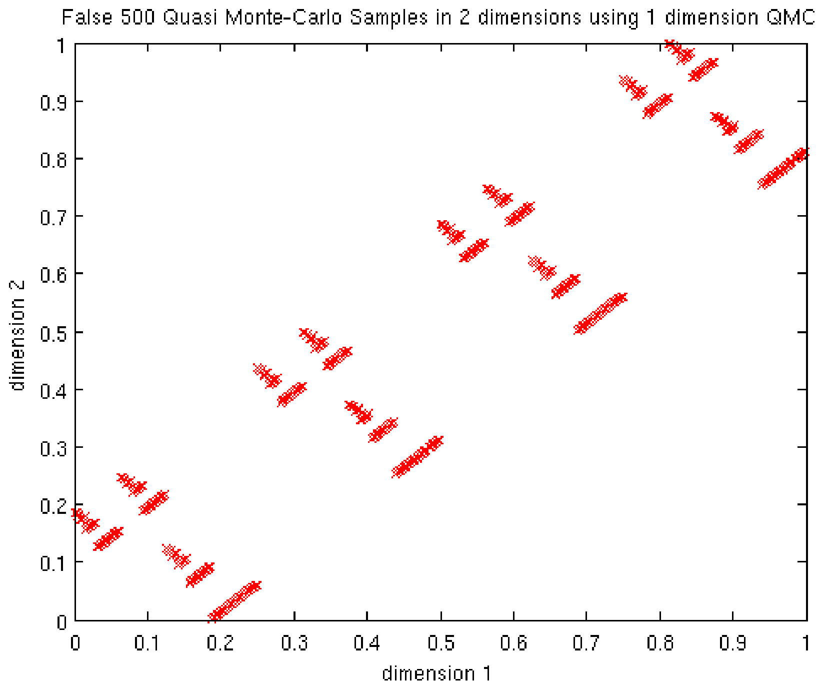
	Sigma 10	Sigma 100	Sigma 500	Sigma 1000	Sigma 3000
MC	23.2779	6.7813	3.2406	2.2104	1.3634
QMC	46.8910	8.2996	2.3315	1.9111	0.7155



In 25 dimensions, QMC is more accurate for 300+ samples

Another pitfall – generating the correct dimension

- Cannot create higher dimension QMC sequences using lower dimension QMC sequences
  - Unlike pseudo-random
- What happens:



Generating Halton Sequences

- Halton sequences are a type of low discrepancy sequence used for QMC
- Every Halton sequence, regardless of dimension, use prime number(s) for its base(s) (Halton1958)

Generating Halton Sequences

- Express any integer n as the sum of the successive powers of radix R. That is,
  - $n = n_{-M} n_{M-1} \ldots n_2 n_1 n_0 = n_0 + n_1 R + n_2 R^2 + \ldots + n_M R^M$, where $M = [\log_R n]$
- a new fraction, f , between 0 and 1, is formed when all the powers of the radix are changed with their respective inverses. That is,
  - $f = f_R(n) = 0.n_0\, n_1\, n_2 \ldots n_{-M}\, n_{M-1} = n_0 R^{-1} + n_1 R^{-2} + n_2 R^{-3} + \ldots + n_M R^{-M-1}$
- To generate N numbers in K dimensions, use the k-dimensional space
  - $(n/N,\ f_{R1}(n),\ f_{R2}(n),\ \ldots,\ f_{RK-1}(n))$,
  - where n = 1, 2, …, N and $R_1, R_2, \ldots, R_{k-1}$ are the first k-1 primes.