Adaptive Quasi-Monte Carlo Integration

June 4\textsuperscript{th}, 2002

Rudolf Schrer
The Problem

Given a function

\[ f : C_s \to \mathbb{R}, \]

with \( C_s = [0, 1]^s \subset \mathbb{R}^s \) denoting the \( s \)-dimensional unit cube.

Calculate an approximation \( Q_f \) for the multi-variate integral

\[ I_f := \int_{C_s} f(x) \, dx. \]

\( Q_f \) has to be based on \( f \)-evaluations at \( n \) points \( x_i \in C_s \) which can be chosen arbitrarily by the integration routine.

Therefore, \( Q_f \) will be of the form

\[ Q_nf = \sum_{i=1}^{n} w_i f(x_i). \]
Monte Carlo Integration

\[ Q_n f := \frac{1}{n} \sum_{i=1}^{n} f(x_i) \]

with \( x_i \) random samples uniformly distributed in \( C_s \).

- \(|f - Q_n f| \approx \frac{\sigma(f)}{\sqrt{n}} = \sqrt{\frac{\text{Var}(f)}{n}} = O(n^{-1/2})\)

- Independent of dimension \( s \)

- \( \sigma(f) \) behaves well for a huge class of integrands

- We can even estimate the accuracy:

\[ |f - Q_n f| \approx \sqrt{\frac{\text{Var}(f)}{n}} \approx \sqrt{\frac{\sum f^2(x_i) - \frac{1}{n} (\sum f(x_i))^2}{n(n-1)}} \]

- \( \Rightarrow \) MC integration is a pretty foolproof way to estimate an integral
Can we do better?

- Random points are evenly distributed in any dimension

- However, random clusters and gaps appear

- Are there high-dimensional, evenly distributed, but regular point sets?
Quasi-Monte Carlo

- Instead of drawing random samples, use low discrepancy point-sets like \((t, m, s)\)-nets!

The first 2048 points from the Sobol sequence 
\(x_1x_3\)-projection (left), \(x_{37}x_{40}\)-projection (right)
Performance of Quasi-Monte Carlo

- Koksma-Hlawka inequality:
  \[ |If - Q_nf| \leq V(f) \cdot D^*_n \leq c \frac{\log^s n}{n} \]
  - Only an upper bound, no estimator
  - \( V(f) = \infty \) even for simple integrands
  - No general method for estimating \( V(f) \)
  - \( \log^s n \) is huge for affordable \( n \)

- However, it works quite well in practice:
  \[ |If - Q_nf| \approx O(n^{-1}) \] is usually obtained!
Adaptive Integration

**Algorithm 1 Adaptive Integration**

Put $C_s$ into region collection

**while** estimated error too large **do**

Choose subregion with large error

Split region

Apply basic rule

Store new regions in region collection

**end while**
Stratified Sampling

Taking $n/2$ samples from two halves of $C_s$ is always better than sampling $C_s$ with $n$ points!

$$\bar{Q}_{nf} = \frac{1}{2} \left( Q_{n/2}f_\alpha + Q_{n/2}f_\beta \right)$$

with $f_\alpha$ and $f_\beta$ denoting $f$ restricted to the left and right subcube.

Variance of this estimator:

$$\text{Var}(\bar{Q}_{nf}) = \frac{1}{4} \left( \text{Var}(Q_{n/2}f_\alpha) + \text{Var}(Q_{n/2}f_\beta) \right)$$

$$\approx \frac{1}{4} \left( \frac{\text{Var}(f_\alpha)}{n/2} + \frac{\text{Var}(f_\beta)}{n/2} \right)$$

$$= \frac{1}{2n} \left( \text{Var}(f_\alpha) + \text{Var}(f_\beta) \right)$$

$$= \frac{1}{n} \cdot \frac{\text{Var}(f_\alpha) + \text{Var}(f_\beta)}{2}$$

$$\leq \frac{1}{n} \cdot \text{Var}(f) = \text{Var}(Q_nf)$$
Recursive Stratified Sampling

Stratification can be done recursively, leading to $n$ subcubes with one random point in each of them.

This comes close to a grid. However, randomization performs much better!
MISER – Adaptive Stratification

Stratification improves performance whenever

\[ \sigma(f_\alpha) \neq \sigma(f_\beta). \]

The optimal performance can be achieved by allocating points such that

\[ n_\alpha/n_\beta = \sigma(f_\alpha)/\sigma(f_\beta). \]

This lead directly to the following adaptive algorithm:

**Algorithm 2 MISER**

1: Allocate points for presampling
2: Estimate \( \sigma_{\alpha_i} \) and \( \sigma_{\beta_i} \) for all \( i = 1, \ldots, s \) halves
3: Choose split dimension
4: Assign point budgets \( N_\alpha \) and \( N_\beta \)
5: Apply MISER to both subcubes
6: Calculate final estimate
Importance Sampling

- Integration error depends on $\text{Var}(f)$

- What if
  
  - We have positive-valued function $p$ with
    $$\int_{C_s} p(x) \, dx = 1,$$
    i.e. $p$ is a probability density function
  
  - $p$ mimics $f$ such that $p \sim |f|$

- Then
  
  - $f/p$ has a very low variance

  -
    $$\int_{C_s} f(x) \, dx = \int_{C_s} \frac{f(x)}{p(x)} \, dP(x),$$
    i.e. the sample mean of $f/p$ with density $p$
    equals sample mean of $f$ with density 1.
Recipe:

Find a pdf $p$ with

- $p \sim |f|$
- We can generate $p$-distributed random numbers

Adaptive Importance Sampling

Algorithm 3 Adaptive Importance Sampling

Start with $p \equiv 1 / \text{vol } C_s$

for $i = 1, \ldots, m$ do

Sample $f / p$ to refine $p$

end for

Use remaining points to sample $f / p$ with density $p$

Algorithms differ by the available functions $p$ and by the way they are estimated.

VEGAS

VEGAS uses a product of piecewise constant, one-dimensional functions.
Control Variates

Break $f$ into two parts $\varphi$ and $(f - \varphi)$ such that

- $I_\varphi$ can be calculated analytically
- $\text{Var}(f - \varphi)$ is small

\[
Q_n f = I_\varphi + \frac{1}{n} \sum_{i=1}^{n} (f - \varphi)(x_i) \\
= I_\varphi + \frac{1}{n} \sum_{i=1}^{n} f(x_i) + \frac{1}{n} \sum_{i=1}^{n} \varphi(x_i) \\
= I_\varphi + \bar{f} + \bar{\varphi}
\]

\[
\text{Var}(Q_n f) = \text{Var}(\bar{f}) + \text{Var}(\bar{\varphi}) - 2 \text{Cov}(\bar{f}, \bar{\varphi})
\]

How can be find $\varphi$?

- “Parallel Simulation”
- Adaptive?
Results

Corner Peak, s=5

# Correct Digits vs. # Integrand Evaluations

Corner Peak, s=15

# Correct Digits vs. # Integrand Evaluations

Methods compared:
- MISER
- VEGAS
- Monte Carlo
- Stratified
- Sobol
Can we extend these results to QMC?

- **Key Concept “Variance Reduction”**
  - \( \text{Var}(f) \) can be estimated easily. However, there is no direct correlation between \( \text{Var}(f) \) and the integration error.
  - Knowing \( V(f) \) would give an upper bound. However, it can neither be estimated nor is the inequality sharp.
  - There are empirical results suggesting that \(|Q_n f - I f| \approx \frac{\text{Var}(f)}{n}\) for many integrands
  - Integration error can be estimated using randomized QMC (e.g.: Owen Scrambling)

- Generating arbitrary distributions is possible, at least if an explicit transformation function is available.

- Applying two nets of size \( n/2 \) to two halves of \( C_s \) definitely decreases performance

- QMC integration has a high rate of convergence by itself. Therefore, improving on it will be harder.