Woudschoten Conference 2016

Introduction to quasi-Monte Carlo methods, with application to PDEs with random coefficients

Part 1

High-dimensional integration: the quasi-Monte Carlo way

Frances Kuo

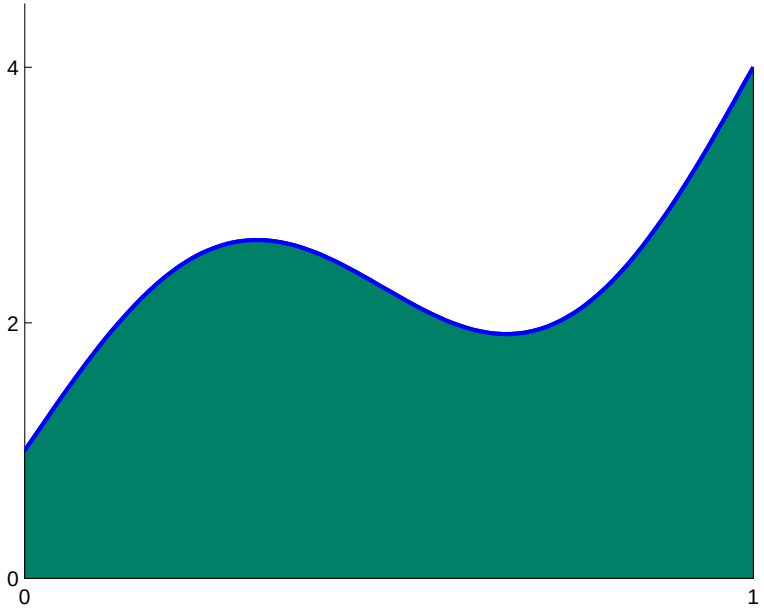
f.kuo@unsw.edu.au

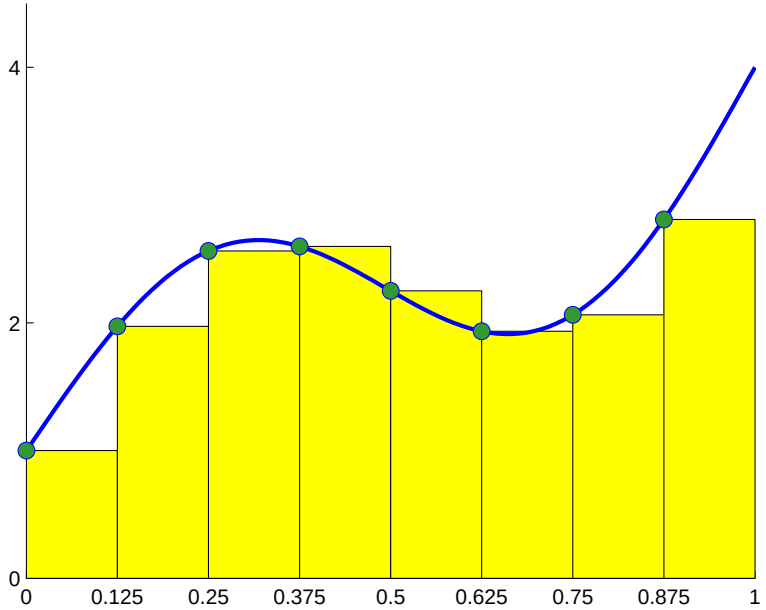
University of New South Wales, Sydney, Australia

∼ There is an *Acta Numerica 2013* survey of the same title, written jointly with Josef Dick and Ian Sloan (UNSW, Australia) ∼

Integration in one dimension

What is the area under the curve between 0 and 1?





Approximation:

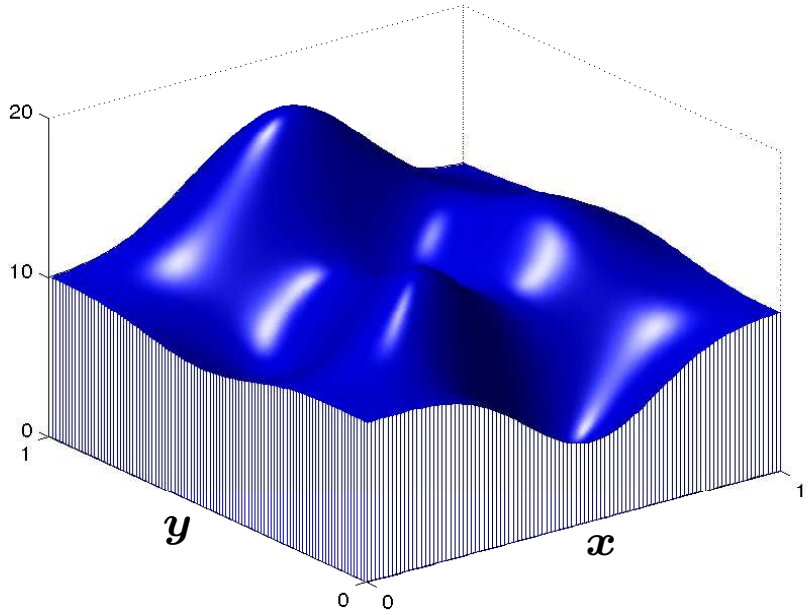
AREA $\approx$ sum of the areas of the 8 rectangles

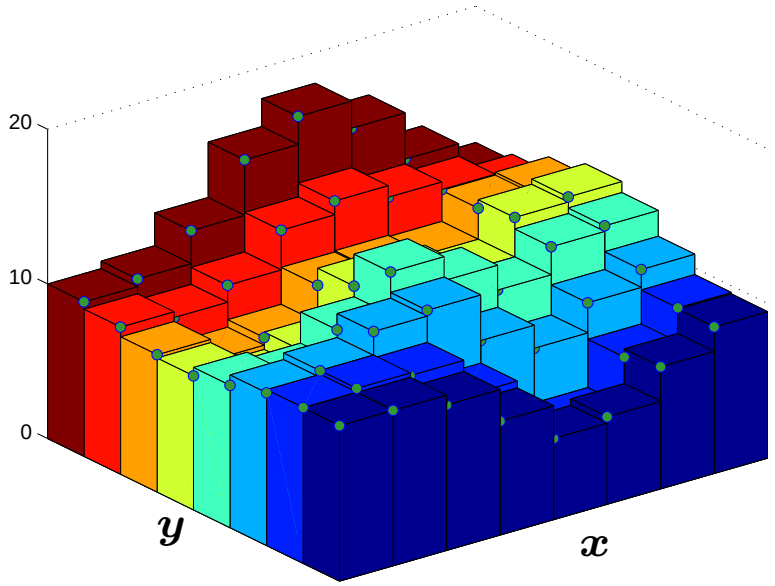
- height $=$ curve values at the dots
- width $= \dfrac{1}{8}$

We get a better approximation with more rectangles

Integration in two dimensions

What is the volume under the surface for x and y between 0 and 1?





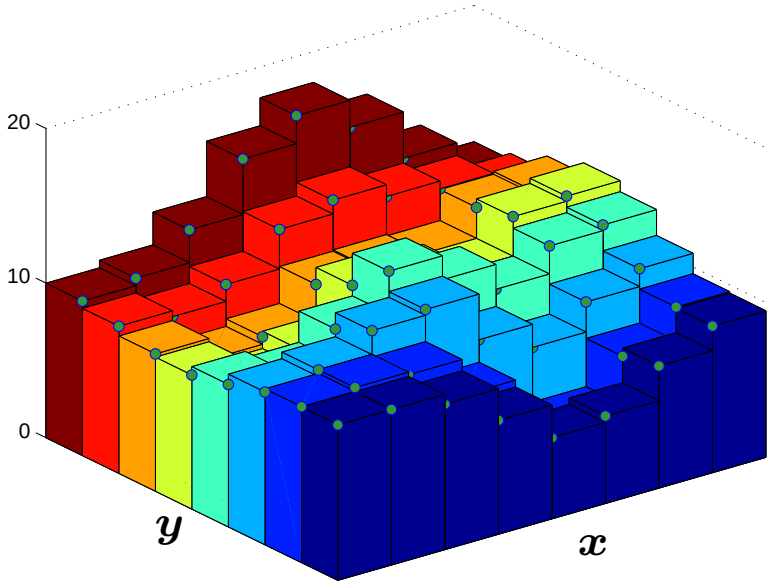
Approximation:

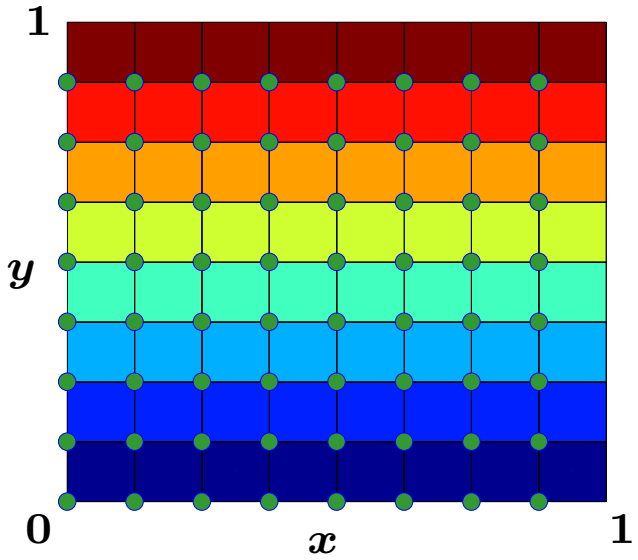
VOLUME $\approx$ sum of the volumes of the $8 \times 8 = 64$ rectangular prisms

- height $=$ surface values at the dots
- base area $= \frac{1}{8} \times \frac{1}{8} = \frac{1}{64}$

Integration in two dimensions

Looking straight down at the xy-plane... we see a grid structure:





Approximation:

VOLUME $\approx$ average of the surface values at the $8 \times 8 = 64$ dots

We get a better approximation with finer grids...

8^2 dots $\rightarrow$ k^2 dots

Integration in s dimensions

Integral in high dimensions $\Rightarrow$ "hyper volume" *plot ???*

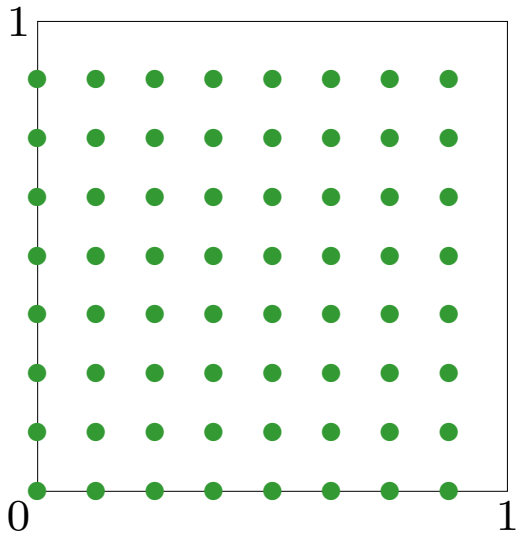
Example: **Collateralized Mortgage Obligations**

– a 30-year mortgage with monthly repayment calculations

$$s = 30 \times 12 = 360$$

Approximation by product grids:

- k^2 points in the unit square
- k^3 points in the unit cube
- $\vdots$
- k^s points in the "hyper cube"



A product rule with 64 points

⟨2D illustration⟩

$s = 360,\ k = 2 \ \Rightarrow\ 2^{360}$ *is astronomical* $(2^{20} \approx$ *one million*$)$

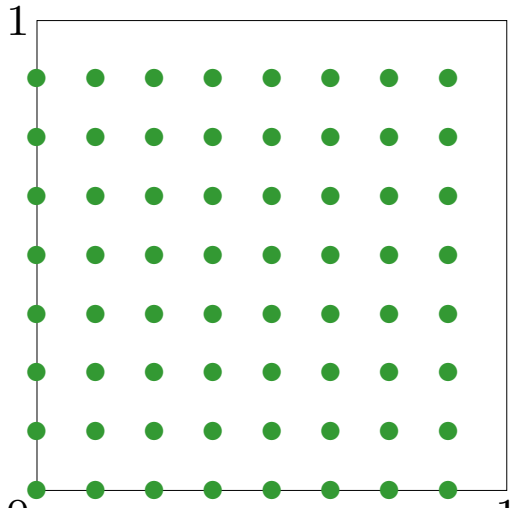
We need to stay away from product grids in high dimensions

Monte Carlo methods

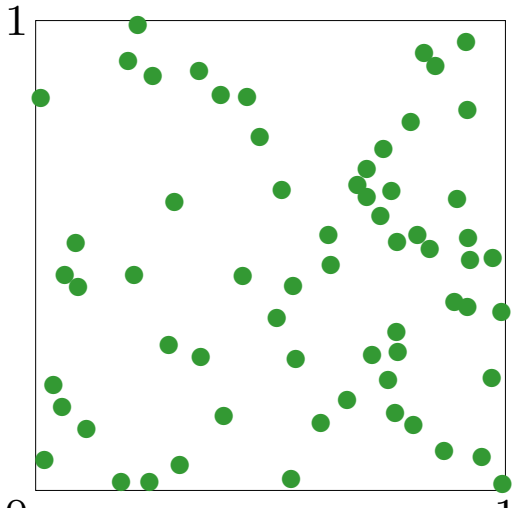
Integral

$\approx$ average of function values at **random** points in the hyper cube

drawbacks: big gaps, clusters, slow convergence



A product rule with 64 points



64 random points

⟨2D illustrations⟩

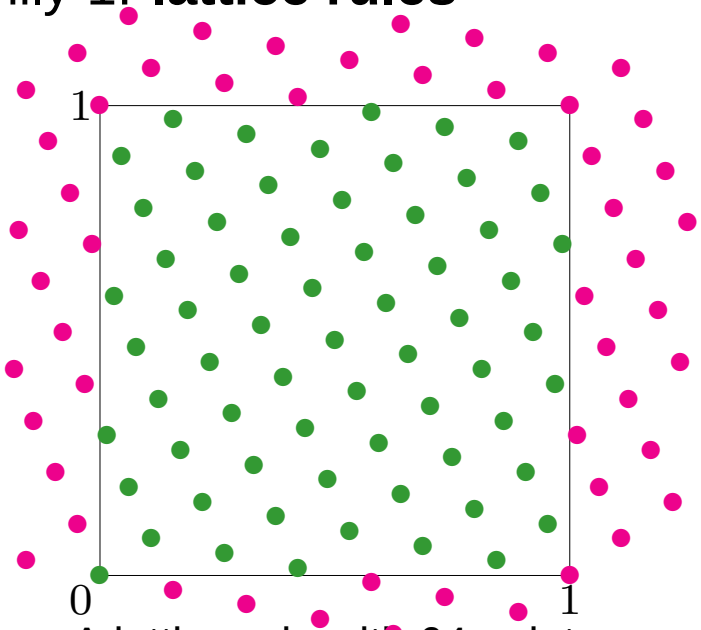
Quasi-Monte Carlo methods

Integral

≈ average of function values at **deterministically chosen** points in the hyper cube

advantages: more uniform than random, faster convergence

Family 1: **lattice rules**



A lattice rule with 64 points

⟨2D illustration⟩

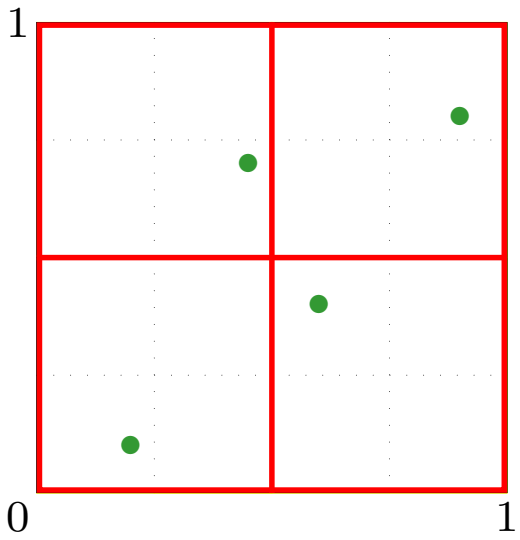
- very regular
- rotated and stretched product grid
- nice projections
- “group” structure

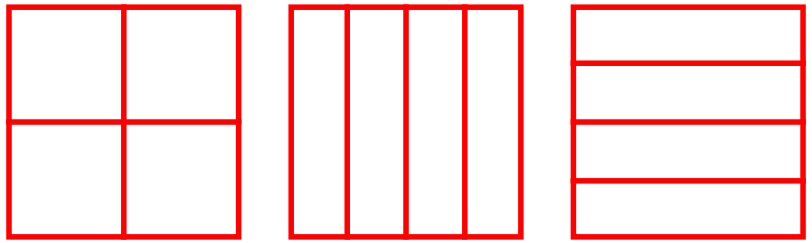
Quasi-Monte Carlo methods

Family 2: **digital nets**

It is all about having the right number of points in various subdivisions.

2D Example: we want to place 4 points in the unit square so that there is exactly one point in each of the 4 rectangles of the same shape and size, given by the three possible subdivisions:



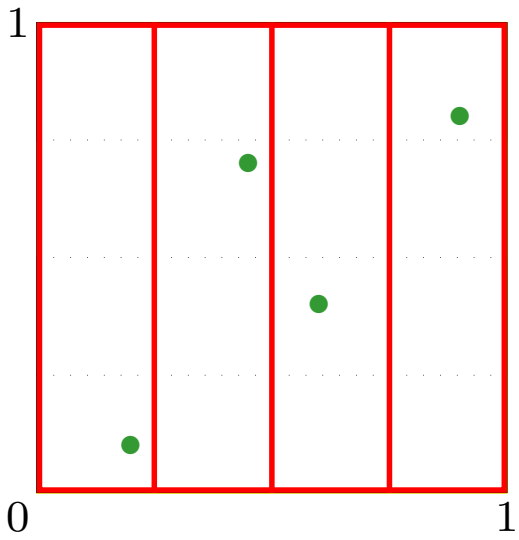


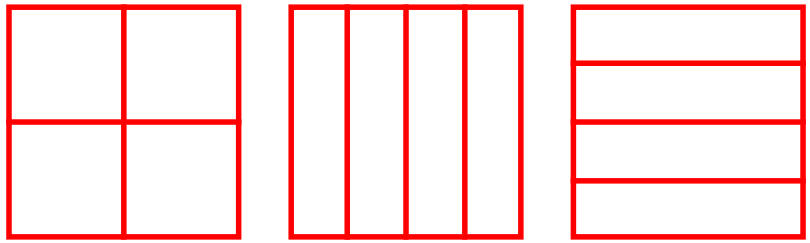
Quasi-Monte Carlo methods

Family 2: **digital nets**

It is all about having the right number of points in various subdivisions.

2D Example: we want to place 4 points in the unit square so that there is exactly one point in each of the 4 rectangles of the same shape and size, given by the three possible subdivisions:



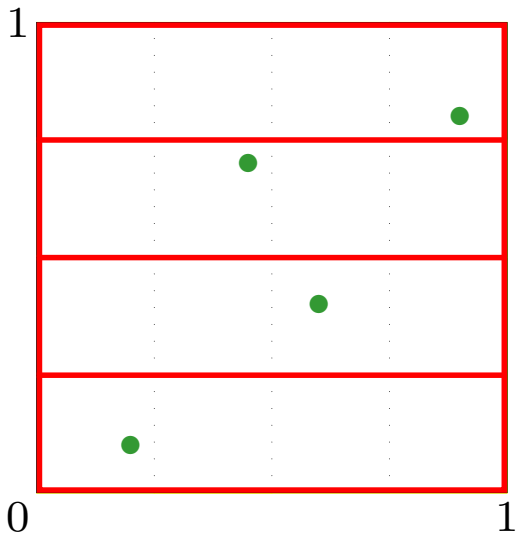


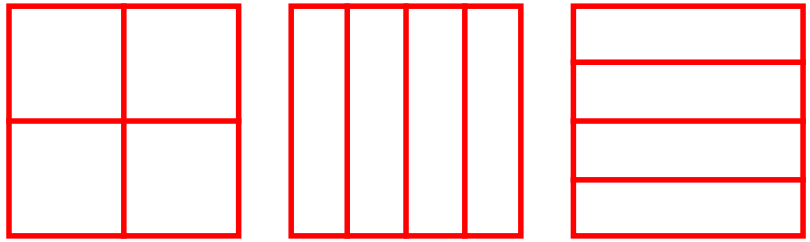
Quasi-Monte Carlo methods

Family 2: **digital nets**

It is all about having the right number of points in various subdivisions.

2D Example: we want to place 4 points in the unit square so that there is exactly one point in each of the 4 rectangles of the same shape and size, given by the three possible subdivisions:



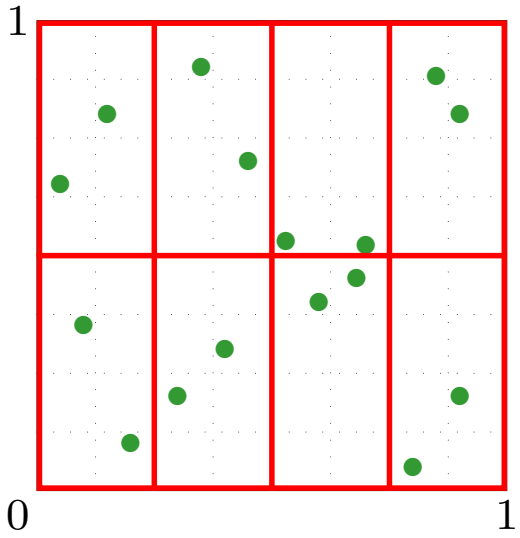


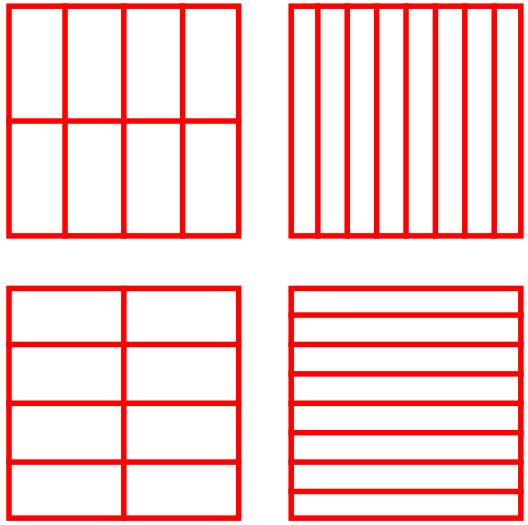
Quasi-Monte Carlo methods

Family 2: **digital nets**

It is all about having the right number of points in various subdivisions.

2D Exercise: draw a "$(1, 4, 2)$-net in base 2", that is, we want to place 16 points in the unit square so that there are exactly 2 points in each of the 8 rectangles of the same shape and size, given by the four possible subdivisions:



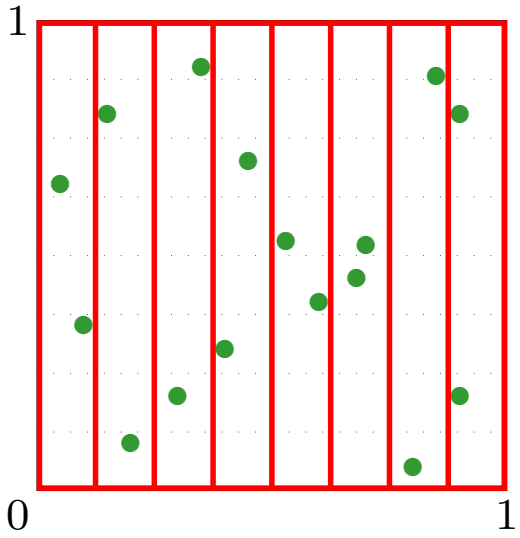


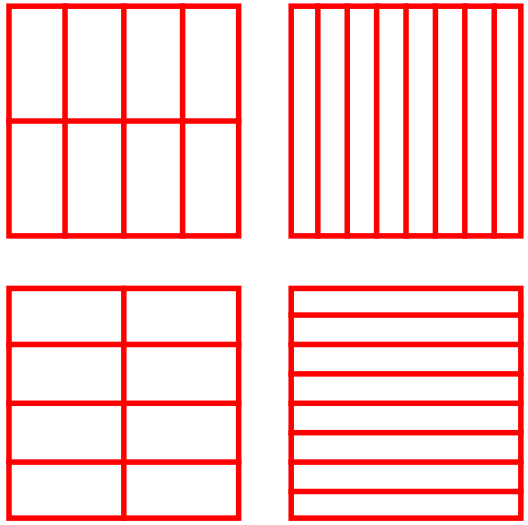
Quasi-Monte Carlo methods

Family 2: **digital nets**

It is all about having the right number of points in various subdivisions.

2D Exercise: draw a "$(1, 4, 2)$-net in base 2", that is, we want to place 16 points in the unit square so that there are exactly 2 points in each of the 8 rectangles of the same shape and size, given by the four possible subdivisions:



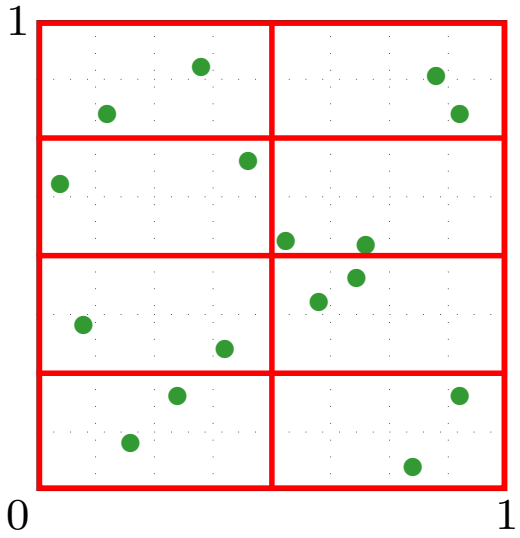


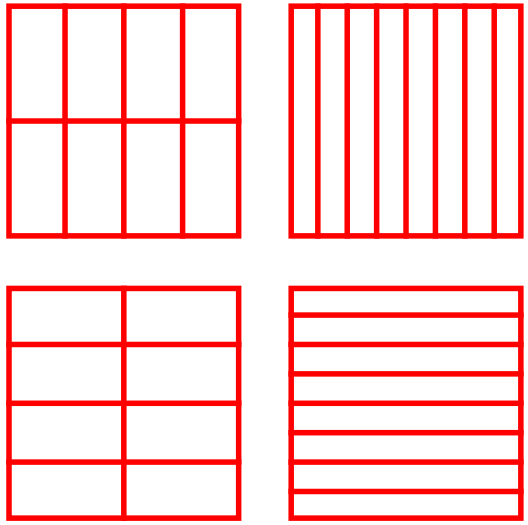
Quasi-Monte Carlo methods

Family 2: **digital nets**

It is all about having the right number of points in various subdivisions.

2D Exercise: draw a "$(1, 4, 2)$-net in base 2", that is, we want to place 16 points in the unit square so that there are exactly 2 points in each of the 8 rectangles of the same shape and size, given by the four possible subdivisions:



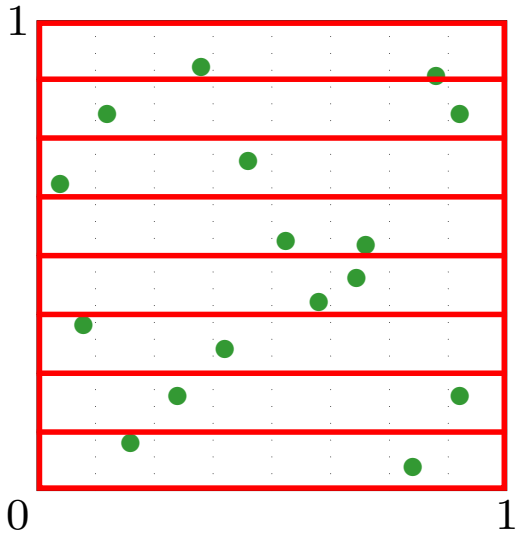


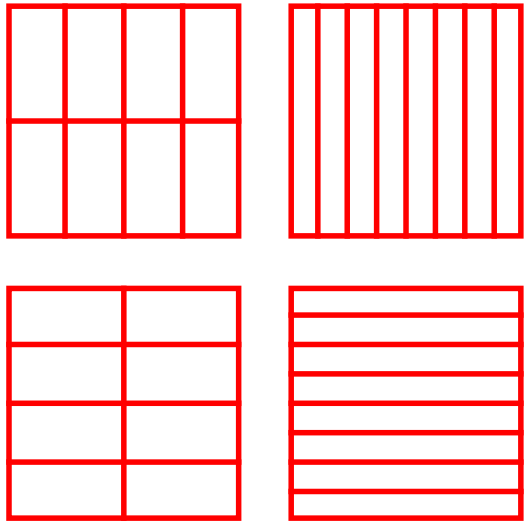
Quasi-Monte Carlo methods

Family 2: **digital nets**

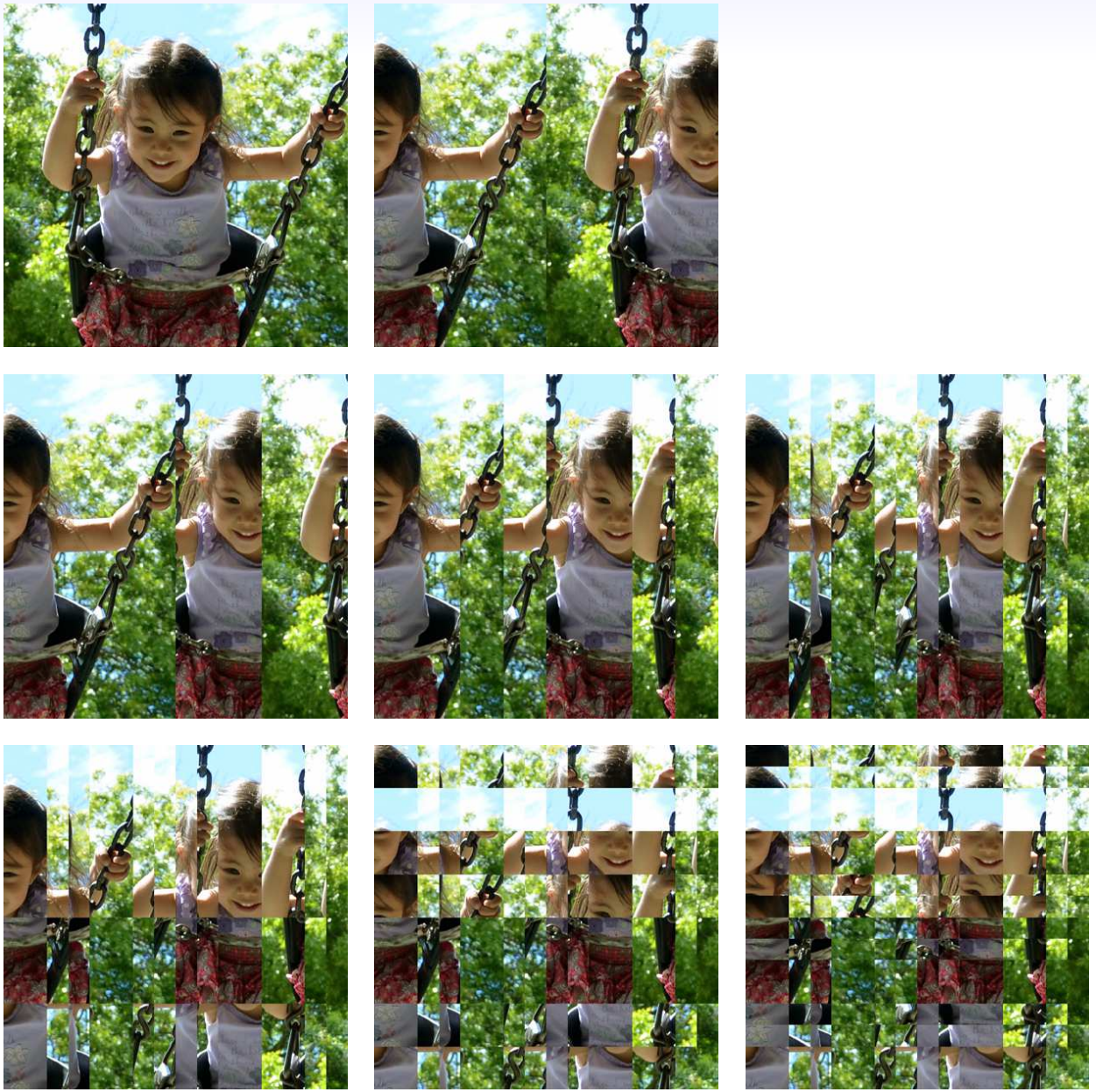
It is all about having the right number of points in various subdivisions.

2D Exercise: draw a "$(1, 4, 2)$-net in base 2", that is, we want to place 16 points in the unit square so that there are exactly 2 points in each of the 8 rectangles of the same shape and size, given by the four possible subdivisions:





Scrambling

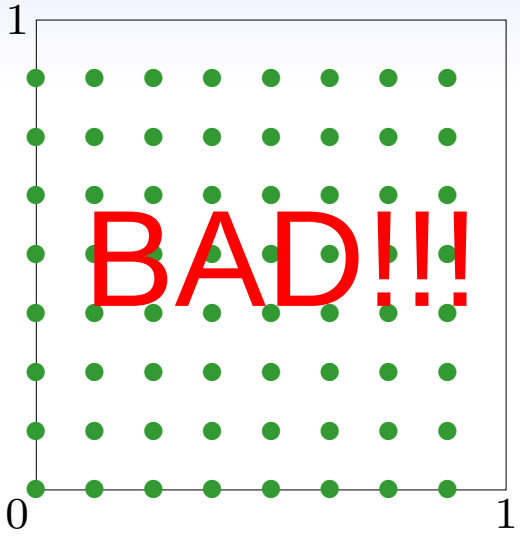


High dimensional numerical integration

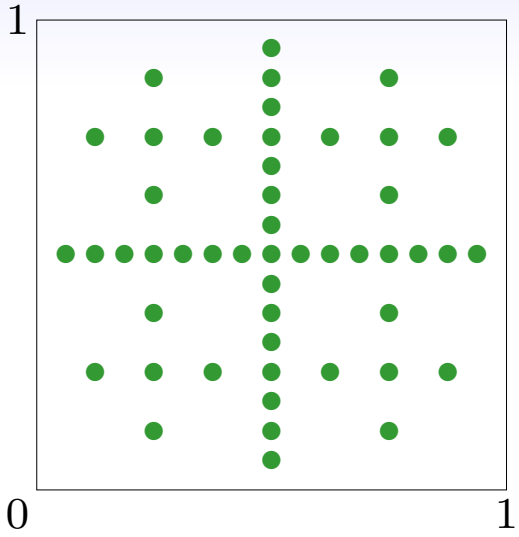
$$\int_{[0,1]^s} f(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y}$$

$$\approx \sum_{i=1}^{n} w_i f(\boldsymbol{t}_i)$$

$$n = k^s$$

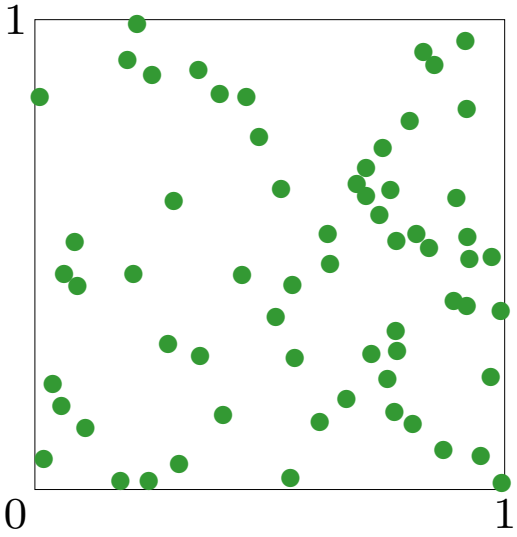


A product rule with 64 points

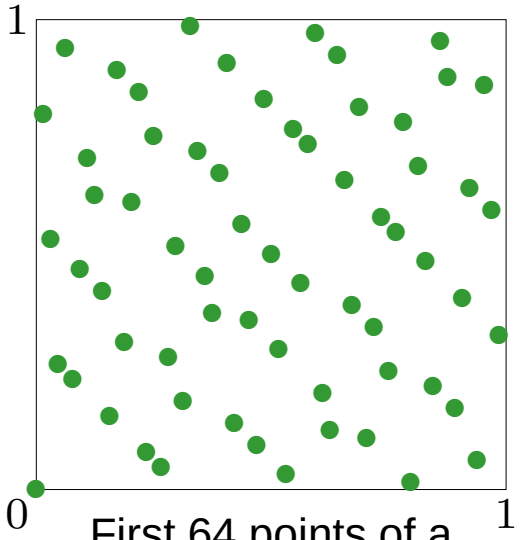


A sparse grid with 49 points

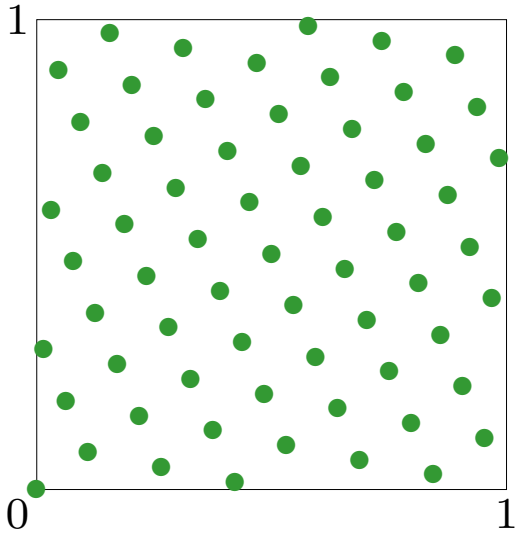
If $f(\boldsymbol{y}) = y_1$ then the error of product rule is $\mathcal{O}(k^{-1}) = \mathcal{O}(n^{-1/s})$



64 random points



First 64 points of a
2D Sobol′ sequence



A lattice rule with 64 points

MC v.s. QMC in the unit cube

$$\int_{[0,1]^s} f(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} \approx \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{t}_i)$$

Monte Carlo method

$\boldsymbol{t}_i$ random uniform

$n^{-1/2}$ convergence

order of variables irrelevant

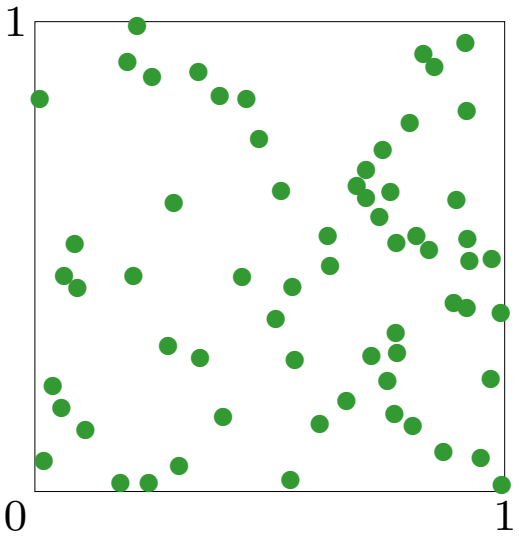
Quasi-Monte Carlo methods

$\boldsymbol{t}_i$ deterministic

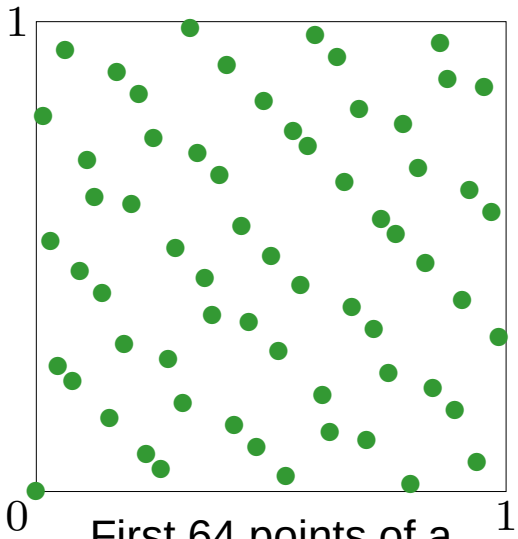
close to n^{-1} convergence or better

more effective for earlier variables and lower-order projections

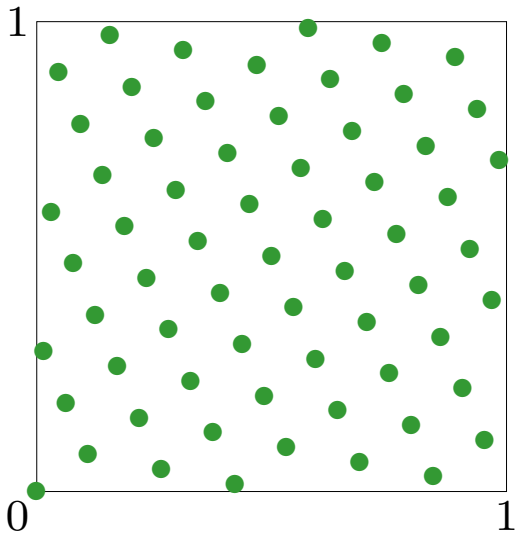
order of variables very important



64 random points



First 64 points of a 2D Sobol′ sequence



A lattice rule with 64 points

use *randomized* QMC methods for error estimation

QMC

Two main families of QMC methods:

- **(t,m,s)-nets** and **(t,s)-sequences**
- **lattice rules**

Niederreiter book (1992)

Sloan and Joe book (1994)

Dick and Pillichshammer book (2010)

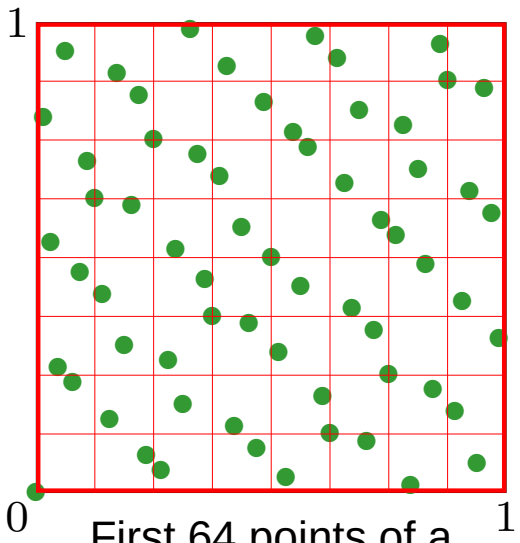
Dick, K., Sloan Acta Numerica (2013)

Important developments:

- ***component-by-component (CBC) construction, "fast" CBC***
- ***higher order digital nets***

Nuyens and Cools (2006)

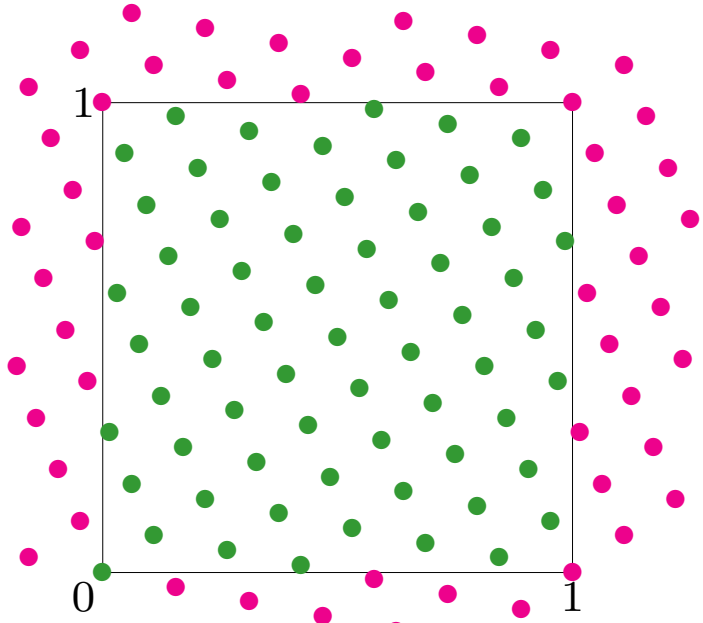
Dick (2008)



First 64 points of a 2D Sobol′ sequence

(0,6,2)-net

Having the right number of points in various sub-cubes



A lattice rule with 64 points

A group under addition modulo $\mathbb{Z}$ and includes the integer points

Lattice rules

Rank-1 lattice rules have points

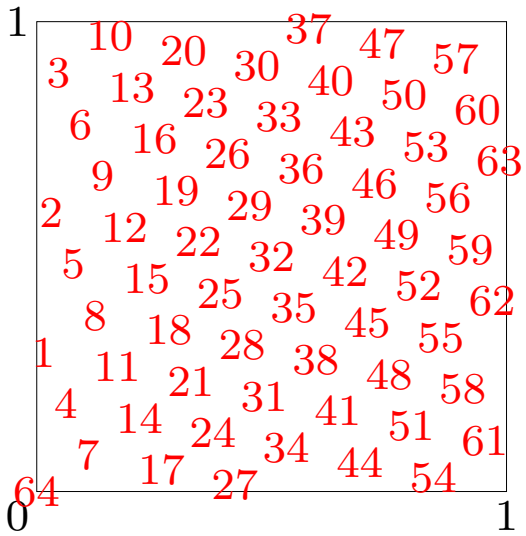
$$\boldsymbol{t}_i = \mathbf{frac}\left(\frac{i}{n}\,\boldsymbol{z}\right), \qquad i = 1, 2, \ldots, n$$

$\boldsymbol{z} \in \mathbb{Z}^s$ – the generating vector, with all components *coprime* to n

$\mathbf{frac}(\cdot)$ – means to take the fractional part of all components

$\sim$ quality determined by the choice of $\boldsymbol{z}$ $\sim$

$$n = 64 \quad \boldsymbol{z} = (1, 19) \quad \boldsymbol{t}_i = \mathbf{frac}\left(\frac{i}{64}(1, 19)\right)$$



A lattice rule with 64 points

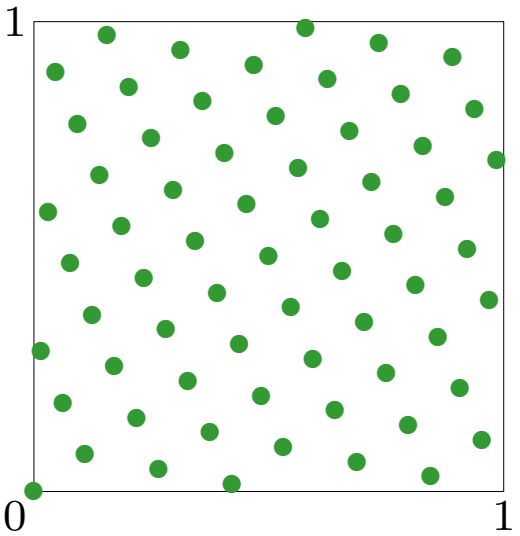
Randomly shifted lattice rules

Shifted rank-1 lattice rules have points

$$\boldsymbol{t}_i = \mathrm{frac}\left(\frac{i}{n}\boldsymbol{z} + \boldsymbol{\Delta}\right), \qquad i = 1, 2, \ldots, n$$

$\boldsymbol{\Delta} \in [0,1)^s$ – the shift

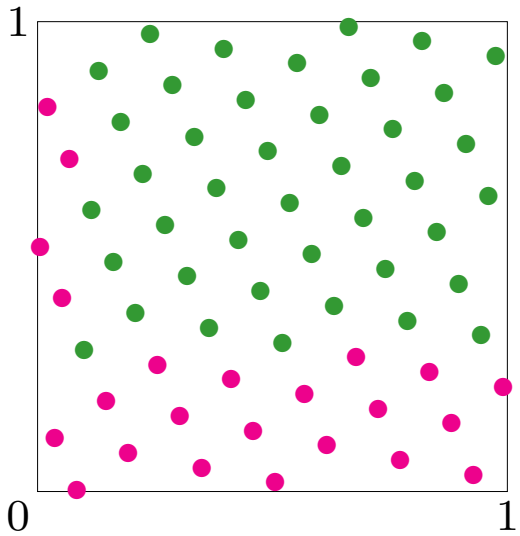
$\sim$ use a number of random shifts for error estimation $\sim$



A lattice rule with 64 points

shifted by

$\boldsymbol{\Delta} = (0.1, 0.3)$



A shifted lattice rule with 64 points

Component-by-component construction

- Want to find $\boldsymbol{z}$ for which some error criterion is as small as possible

  $\sim$ Exhaustive search is practically impossible - too many choices! $\sim$

- **CBC algorithm** [Korobov (1959); Sloan, Reztsov (2002); Sloan, K., Joe (2002),. . .]

  1. Set $z_1 = 1$.
  2. With z_1 fixed, choose z_2 to minimize the error criterion in 2D.
  3. With z_1, z_2 fixed, choose z_3 to minimize the error criterion in 3D.
  4. etc.

  [K. (2003); Dick (2004)]

- Optimal rate of convergence $\mathcal{O}(n^{-1+\delta})$ in "weighted Sobolev space", independently of s under an appropriate condition on the weights

  $\sim$ Averaging argument: there is always one choice as good as average! $\sim$

  [Nuyens, Cools (2006)]

- Cost of algorithm for "product weights" is $\mathcal{O}(n \log n\, s)$ using FFT

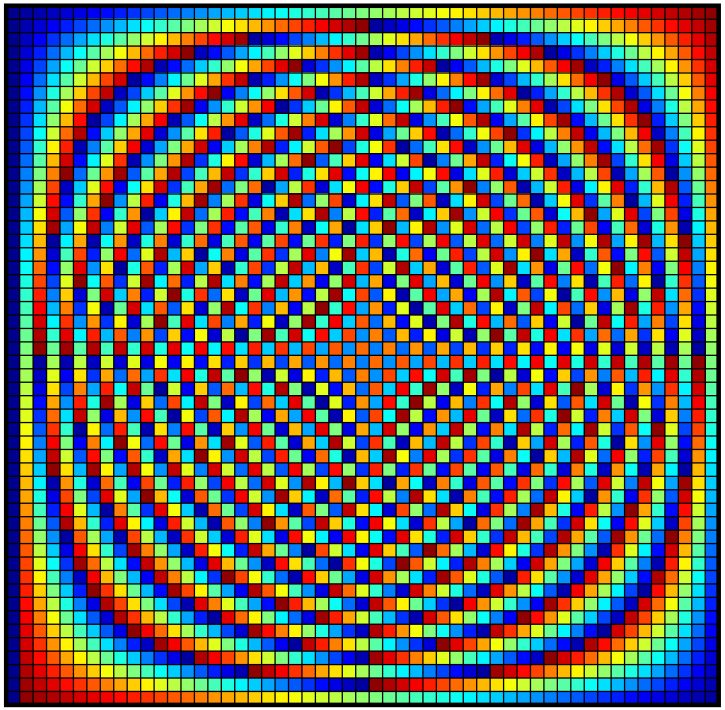
  [Hickernell, Hong, L'Ecuyer, Lemieux (2000); Hickernell, Niederreiter (2003)]

- Extensible/embedded variants [Cools, K., Nuyens (2006)] [Dick, Pillichshammer, Waterhouse (2007)]

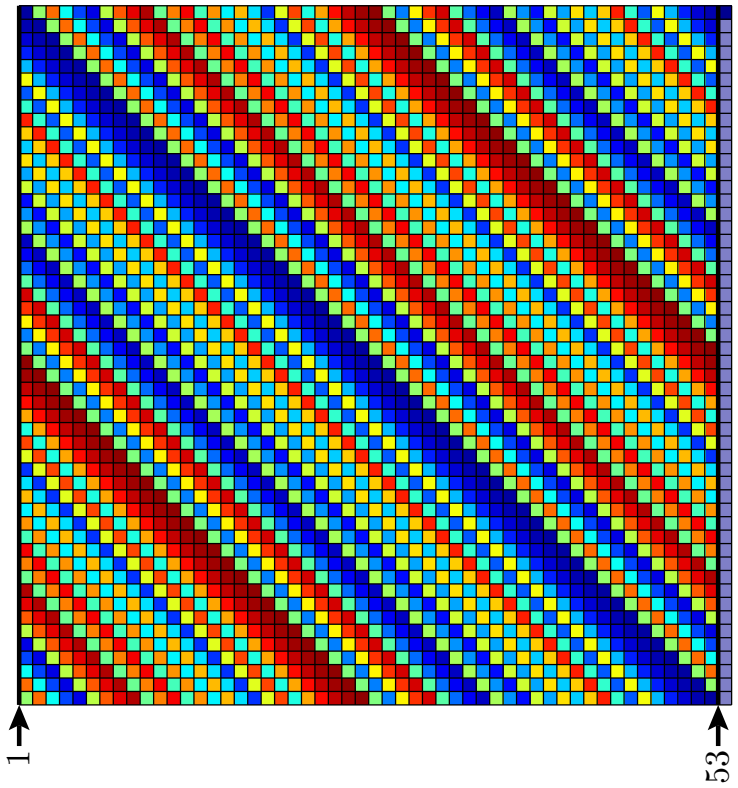
`http://people.cs.kuleuven.be/~dirk.nuyens/fast-cbc/`
`http://people.cs.kuleuven.be/~dirk.nuyens/qmc-generators/`

Fast CBC construction

$n = 53$



Natural ordering of the indices



Generator ordering of the indices

Matrix-vector multiplication with a circulant matrix can be done using FFT

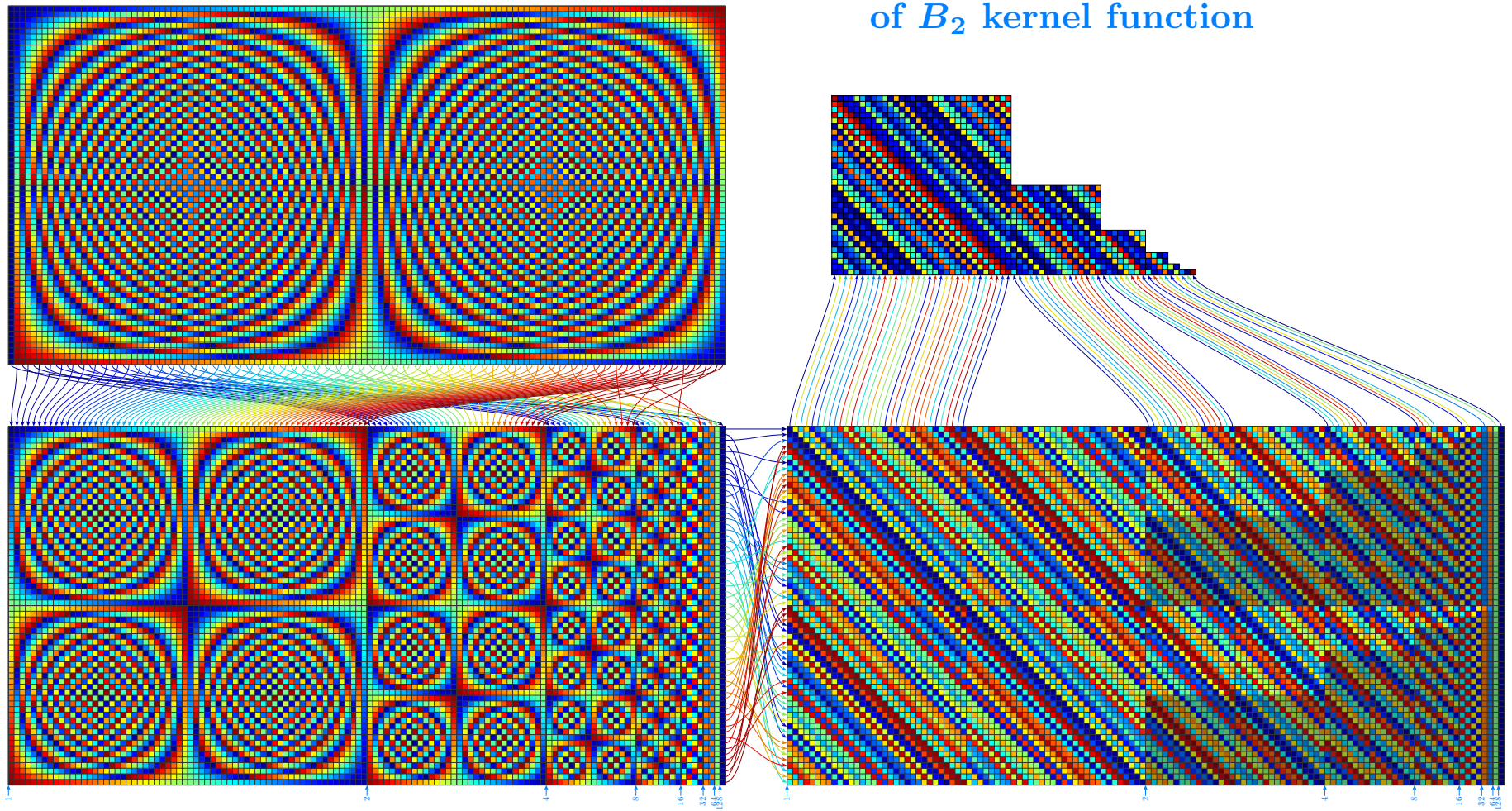
Images by Dirk Nuyens, KU Leuven

Fast CBC construction

$$n = 128 = 2^7$$

① Natural ordering of the indices

④ Symmetric reduction after application of B_2 kernel function



② Grouping on divisors

③ Generator ordering of the indices

Images by Dirk Nuyens, KU Leuven

Setting 1: standard QMC for unit cube

- **Worst case error bound**

$$\left| \int_{[0,1]^s} f(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} - \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{t}_i) \right| \le e^{\mathrm{wor}}_{\boldsymbol{\gamma}}(\boldsymbol{t}_1,\ldots,\boldsymbol{t}_n)\,\|f\|_{\boldsymbol{\gamma}}$$

- **Weighted Sobolev space** [Sloan, Woźniakowski (1998)]

$$\|f\|^2_{\boldsymbol{\gamma}} = \sum_{\mathfrak{u}\subseteq\{1:s\}} \frac{1}{\gamma_{\mathfrak{u}}} \int_{[0,1]^{|\mathfrak{u}|}} \left| \frac{\partial^{|\mathfrak{u}|} f}{\partial \boldsymbol{y}_{\mathfrak{u}}}(\boldsymbol{y}_{\mathfrak{u}}; 0) \right|^2 \mathrm{d}\boldsymbol{y}_{\mathfrak{u}}$$

2^s subsets

"weights"

"anchor" at 0 (also "unanchored")

Mixed first derivatives are square integrable

Small weight $\gamma_{\mathfrak{u}}$ means that f depends weakly on the variables $\boldsymbol{y}_{\mathfrak{u}}$

- **Pair with randomly shifted lattice rules**

- **Choose weights to minimize the error bound** [K., Schwab, Sloan (2012)]

$$\underbrace{\left(\frac{2}{n}\sum_{\mathfrak{u}\subseteq\{1:s\}} \gamma_{\mathfrak{u}}^{\lambda} a_{\mathfrak{u}}\right)^{1/(2\lambda)}}_{\text{bound on worst case error (CBC)}} \underbrace{\left(\sum_{\mathfrak{u}\subseteq\{1:s\}} \frac{b_{\mathfrak{u}}}{\gamma_{\mathfrak{u}}}\right)^{1/2}}_{\text{bound on norm}} \Rightarrow \gamma_{\mathfrak{u}} = \left(\frac{b_{\mathfrak{u}}}{a_{\mathfrak{u}}}\right)^{1/(1+\lambda)}$$

- **Construct points (CBC) to minimize the worst case error**

Setting 2: QMC integration over $\mathbb{R}^s$

- **Change of variables**

$$\int_{\mathbb{R}^s} f(\boldsymbol{y}) \prod_{j=1}^{s} \phi(y_j)\,\mathrm{d}\boldsymbol{y} = \int_{[0,1]^s} f(\Phi^{-1}(\boldsymbol{w}))\,\mathrm{d}\boldsymbol{w}$$

- **Norm with weight function** [Wasilkowski, Woźniakowski (2000)]

[K., Sloan, Wasilkowski, Waterhouse (2010); Nichols, K. (2014)]

- **Pair with randomly shifted lattice rules**

Setting 3: smooth integrands in the cube

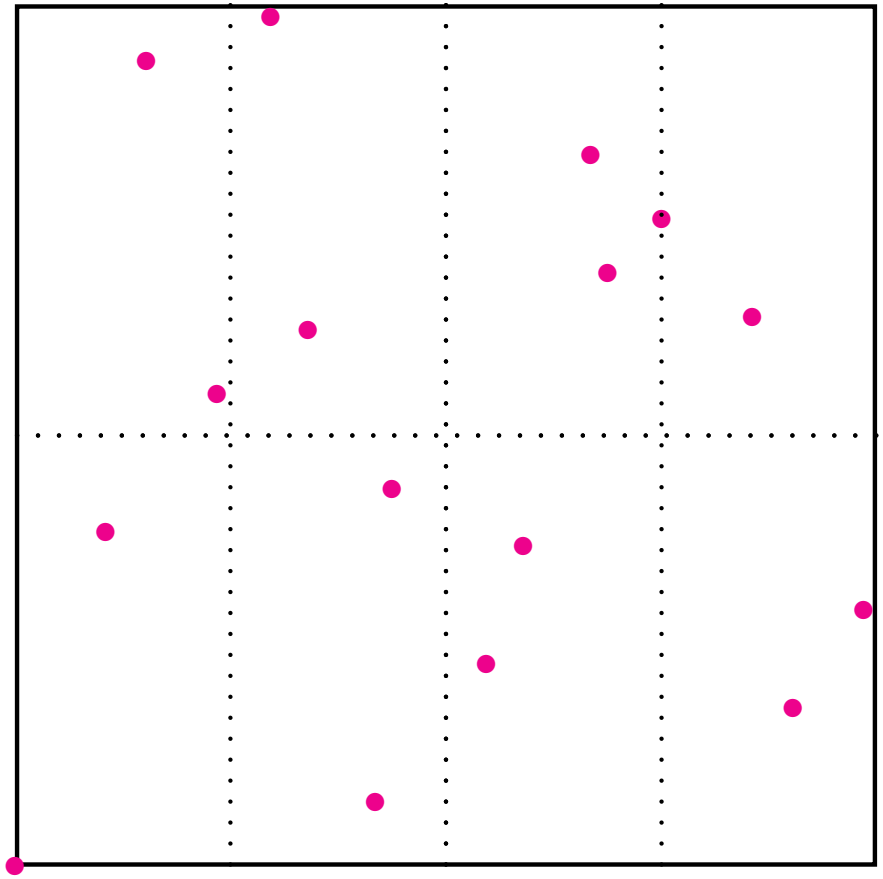
- **Norm involving higher order mixed derivatives**
- **Pair with higher order digital nets** [Dick (2008)]
  - Classical polynomial lattice rule [Niederreiter (1992)]
    - $n = b^m$ points with prime b
    - An irreducible polynomial with degree m
    - A generating vector of s polynomials with degree $< m$
  - Interlaced polynomial lattice rule [Goda, Dick (2012)]
    - Interlacing factor α
    - An irreducible polynomial with degree m
    - A generating vector of αs polynomials with degree $< m$
    - Digit interlacing function $\mathscr{D}_\alpha : [0,1)^\alpha \to [0,1)$

      $(0.x_{11}x_{12}x_{13}\cdots)_b, (0.x_{21}x_{22}x_{23}\cdots)_b, \ \ldots, \ (0.x_{\alpha 1}x_{\alpha 2}x_{\alpha 3}\cdots)_b$

      becomes $(0.x_{11}x_{21}\cdots x_{\alpha 1}x_{12}x_{22}\cdots x_{\alpha 2}x_{13}x_{23}\cdots x_{\alpha 3}\cdots)_b$

Properties of higher order digital nets

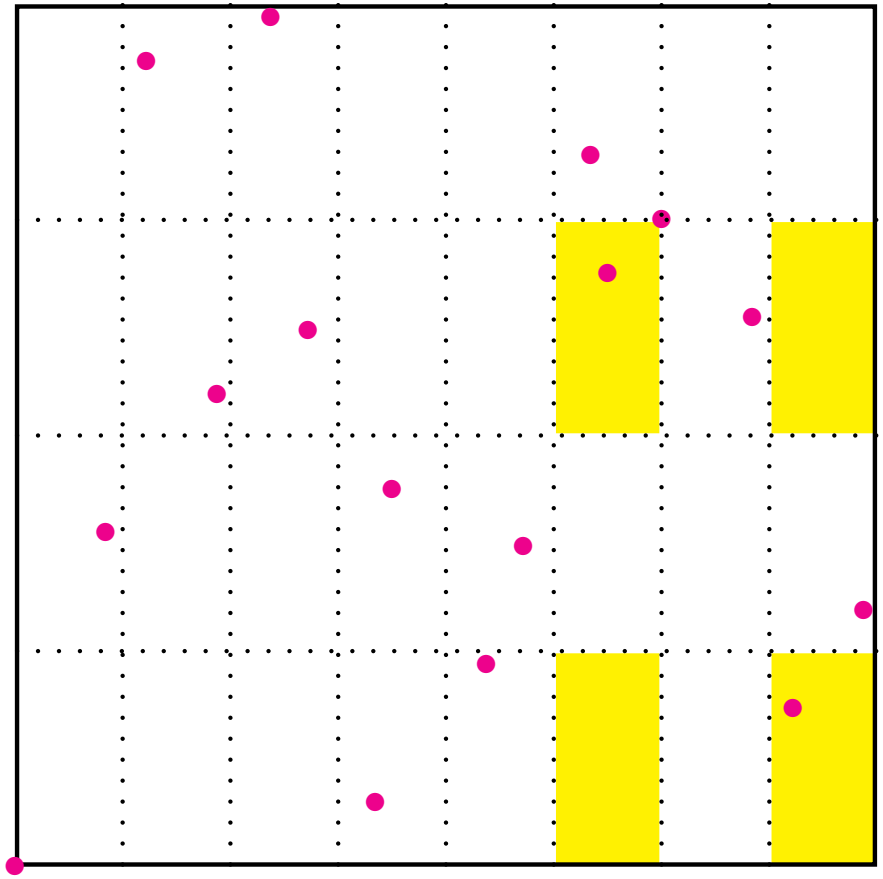
16 points obtained from a 4D Sobol′ sequence with interlacing factor 2:



Each rectangle contains exactly 2 points.

Properties of higher order digital nets

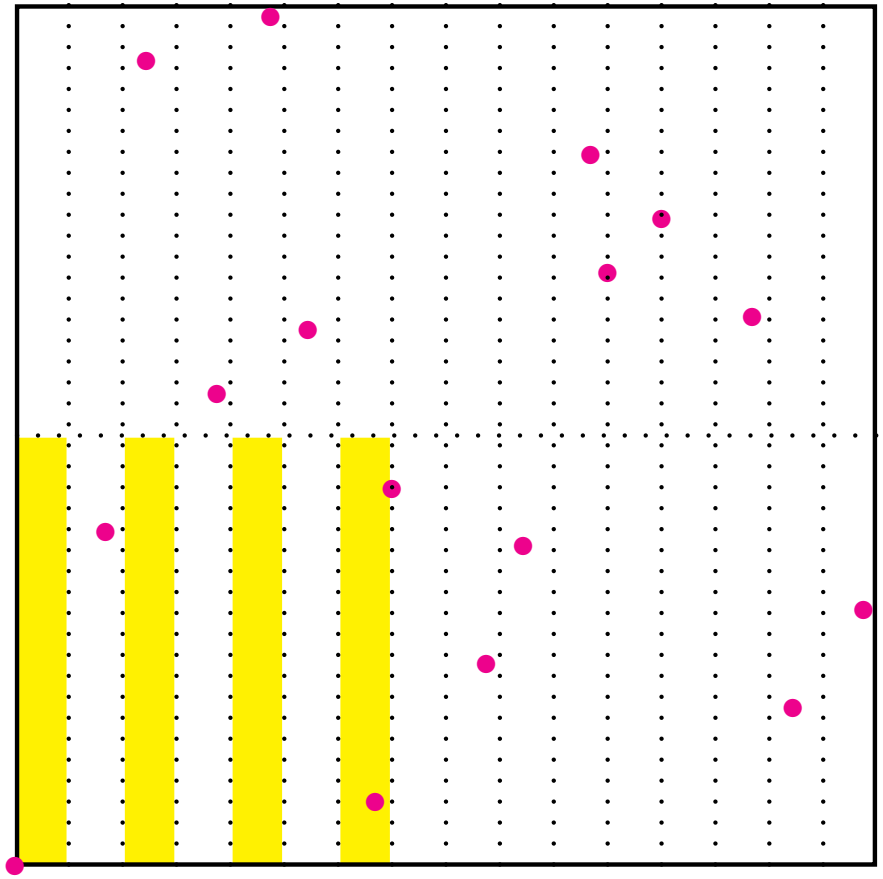
16 points obtained from a 4D Sobol′ sequence with interlacing factor 2:



The shaded area contains exactly 2 points.

Properties of higher order digital nets

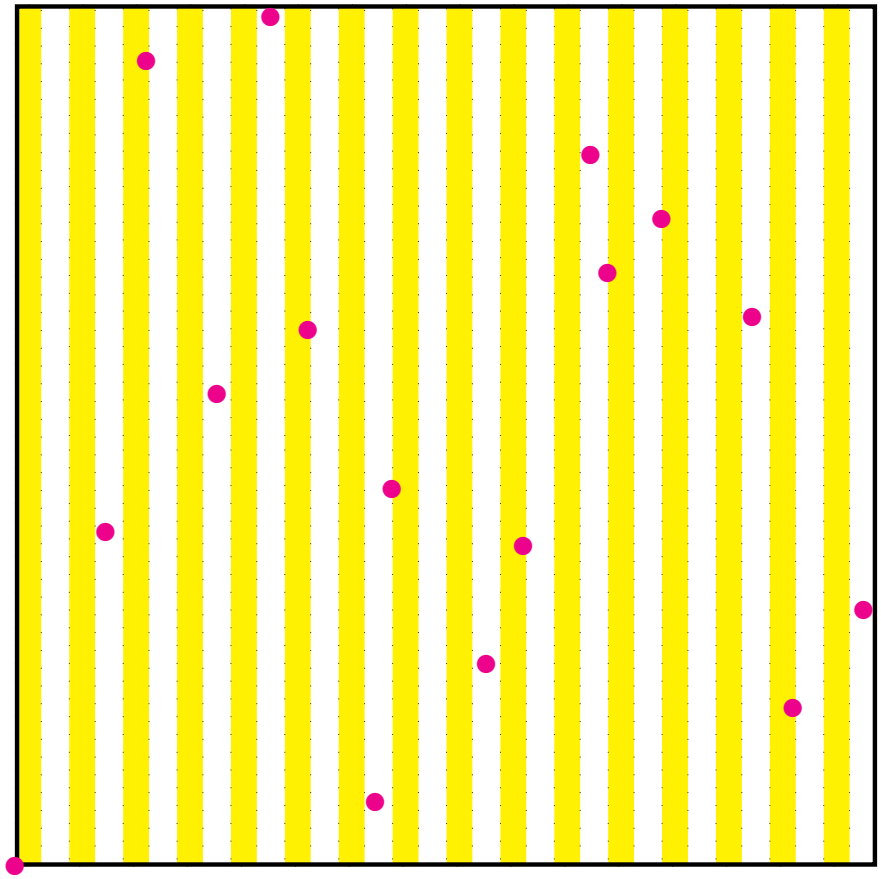
16 points obtained from a 4D Sobol′ sequence with interlacing factor 2:



The shaded area contains exactly 2 points.

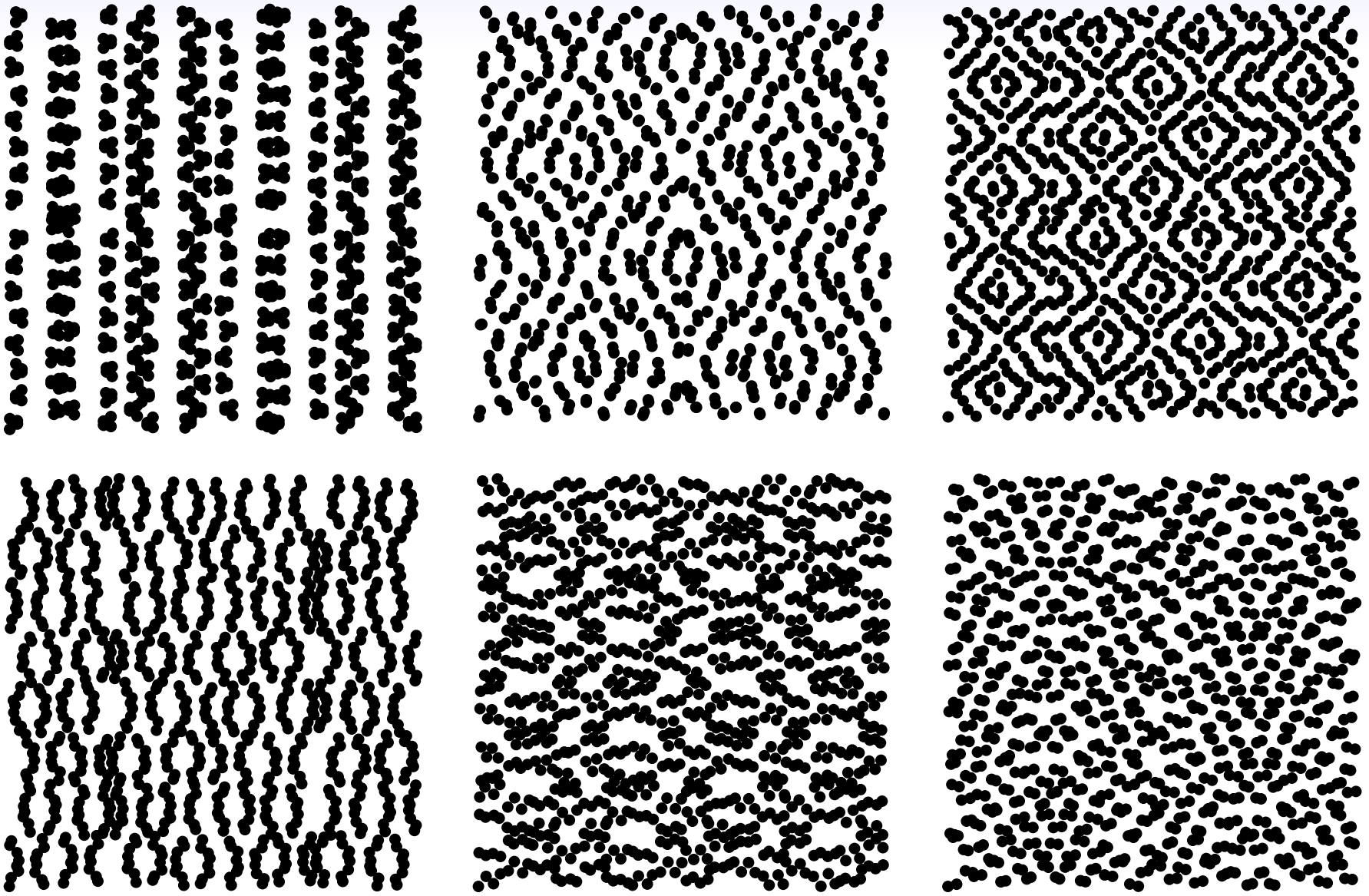
Properties of higher order digital nets

16 points obtained from a 4D Sobol′ sequence with interlacing factor 2:



The shaded area contains exactly half of the points.

Projections of higher order digital nets



Images by Dirk Nuyens, KU Leuven

Application 1: QMC for option pricing

Path-dependent option

[Giles, K., Sloan, Waterhouse (2008)]

$$\int_{\mathbb{R}^s} \max\left(\frac{1}{s}\sum_{j=1}^{s} S_j(\boldsymbol{z}) - K, 0\right) \frac{\exp(-\frac{1}{2}\boldsymbol{z}^{\mathrm{T}}\Sigma^{-1}\boldsymbol{z})}{\sqrt{(2\pi)^s \det(\Sigma)}}\,\mathrm{d}\boldsymbol{z}$$

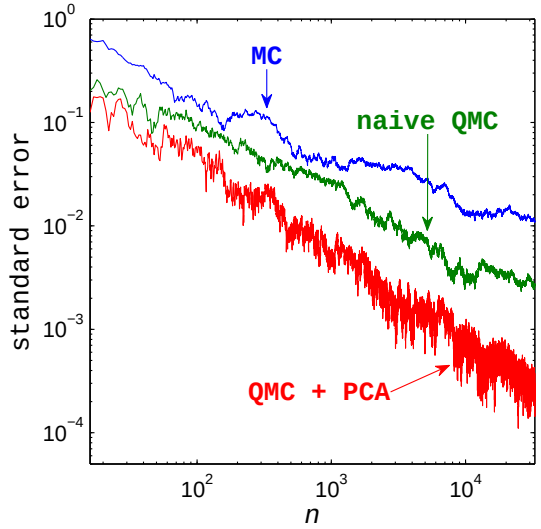
$$= \int_{\mathbb{R}^s} f(\boldsymbol{y}) \prod_{j=1}^{s} \phi_{\mathrm{nor}}(y_j)\,\mathrm{d}\boldsymbol{y}$$

$$= \int_{[0,1]^s} g(\boldsymbol{w})\,\mathrm{d}\boldsymbol{w}$$

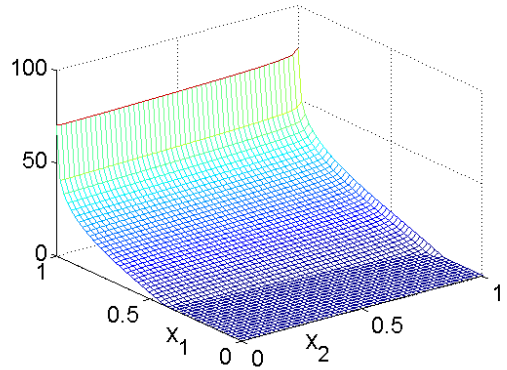
Write $\Sigma = AA^{\mathrm{T}}$

Sub. $\boldsymbol{z} = A\boldsymbol{y}$

Sub. $\boldsymbol{y} = \Phi_{\mathrm{nor}}^{-1}(\boldsymbol{w})$



- In the cube, g is unbounded and has a kink



- ANOVA decomposition: $f(\boldsymbol{y}) = \sum_{\mathfrak{u}\subseteq\{1:s\}} f_{\mathfrak{u}}(\boldsymbol{y}_{\mathfrak{u}})$
- All $f_{\mathfrak{u}}$ with $\mathfrak{u} \neq \{1:s\}$ are smooth under BB

[Griebel, K., Sloan (2010, 2013, 2016)]

- Smoothing by preintegration

[Griewank, Leövey, K., Sloan (in progress)]

$(P_k f)(\boldsymbol{y}) = \int_{\mathbb{R}} f(\boldsymbol{y})\,\phi_{\mathrm{nor}}(y_k)\,\mathrm{d}y_k$ is smooth for some choice of k

Application 2: QMC for maximum likelihood

Generalized linear mixed model [K., Dunsmuir, Sloan, Wand, Womersley (2008)]

$$\int_{\mathbb{R}^s} \left(\prod_{j=1}^{s} \frac{\exp(\tau_j(\beta + z_j) - e^{\beta+z_j})}{\tau_j!} \right) \frac{\exp(-\frac{1}{2}\boldsymbol{z}^{\mathrm{T}}\Sigma^{-1}\boldsymbol{z})}{\sqrt{(2\pi)^s \det(\Sigma)}} \, \mathrm{d}\boldsymbol{z}$$

$$= \int_{\mathbb{R}^s} f(\boldsymbol{y}) \prod_{j=1}^{s} \phi(y_j) \, \mathrm{d}\boldsymbol{y}$$

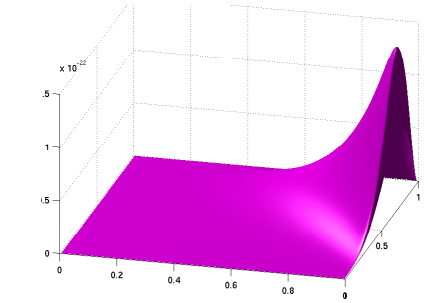
$$= \int_{[0,1]^s} g(\boldsymbol{w}) \, \mathrm{d}\boldsymbol{w}$$

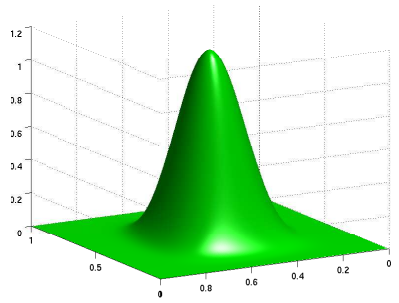
Re-center and re-scale, and multiply and divide by ϕ

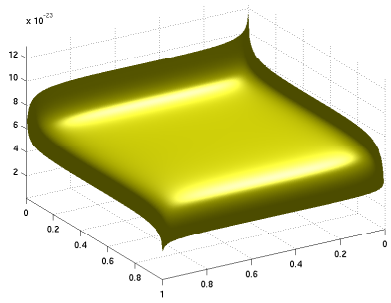
Sub. $\boldsymbol{z} = A^*\boldsymbol{y} + \boldsymbol{z}^*$

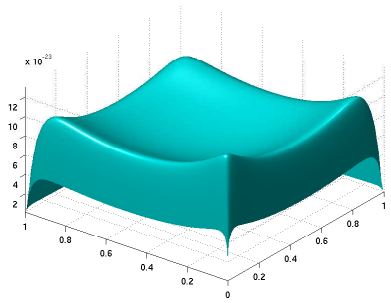
Sub. $\boldsymbol{y} = \Phi^{-1}(\boldsymbol{w})$

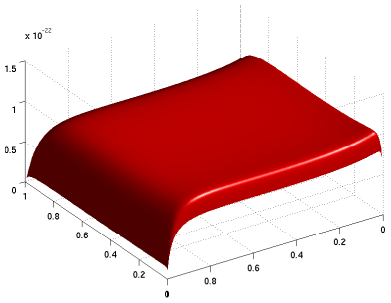
cf. importance sampling


no re-centering (worst)


no re-scaling (bad)


ϕ normal (good)


ϕ logistic (better)


ϕ Student-t (best)

- In the cube, g and/or its derivatives are unbounded
- New weighted space setting over $\mathbb{R}^s$ [K., Sloan, Wasilkowski, Waterhouse (2010)] [Nichols, K. (2014)]
- Differentiate f to estimate the norm [Sinescu, K., Sloan (2013)]

Application 3: QMC for PDEs with random coeff.

Uncertainty in groundwater flow

eg. risk analysis of radwaste disposal or CO_2 sequestration

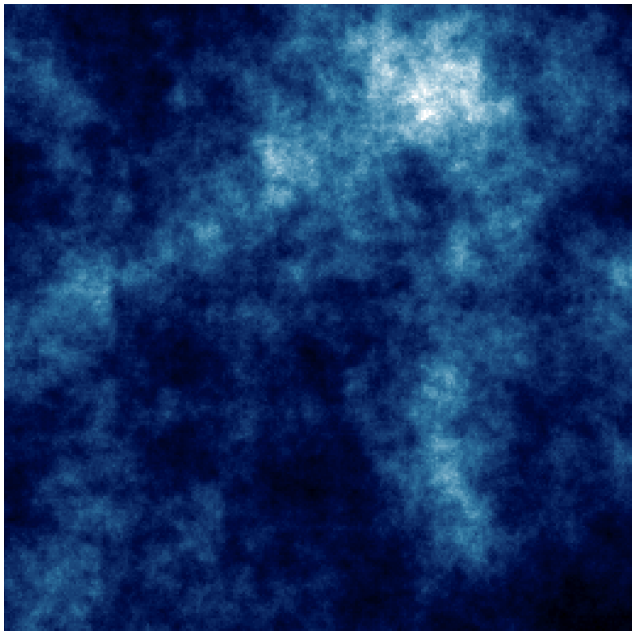
Darcy's law

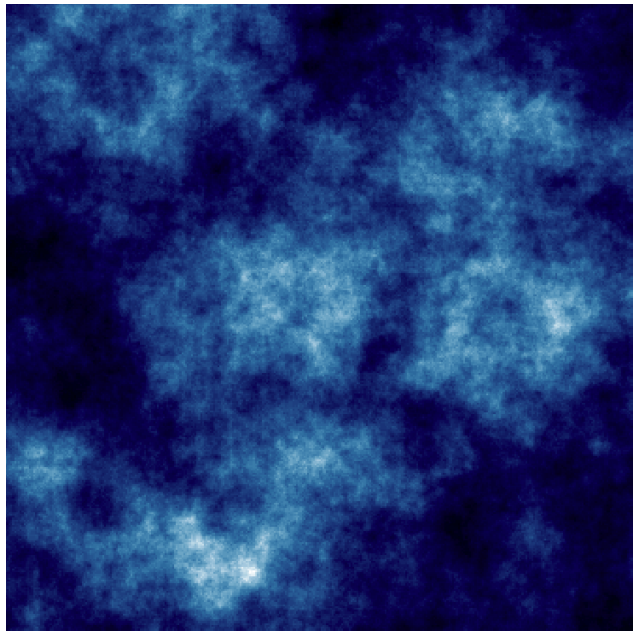
mass conservation law

$$q + a\,\vec{\nabla} p = \kappa$$
$$\nabla \cdot q = 0$$

in $D \subset \mathbb{R}^d$, $d = 1, 2, 3$

together with boundary conditions





Uncertainty in $a(\boldsymbol{x}, \omega)$ leads to uncertainty in $q(\boldsymbol{x}, \omega)$ and $p(\boldsymbol{x}, \omega)$

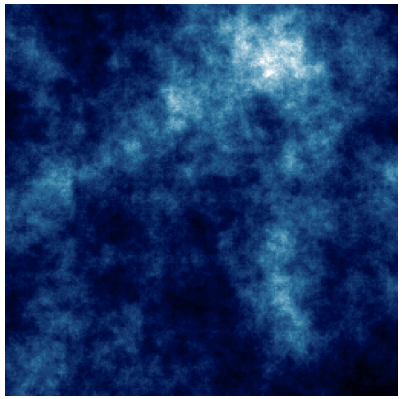
Application 3: QMC for PDEs with random coeff.

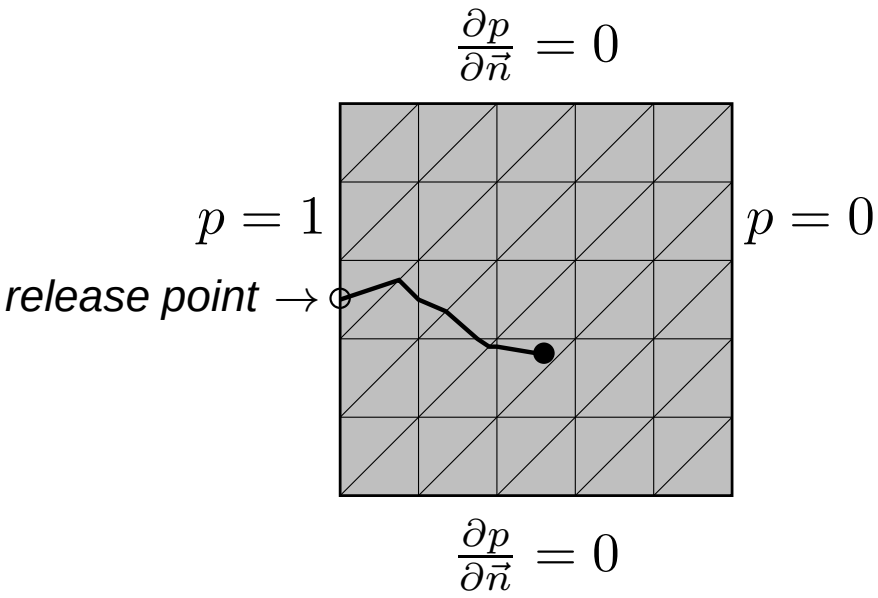
To compute the expected value of some quantity of interest:

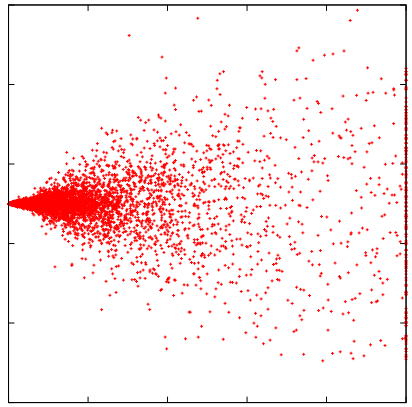
1. Generate a number of realizations of the random field
   (Some approximation may be required)
2. For each realization, solve the PDE using e.g. FEM / FVM / mFEM
3. Take the average of all solutions from different realizations

This describes Monte Carlo simulation.

Example: particle dispersion







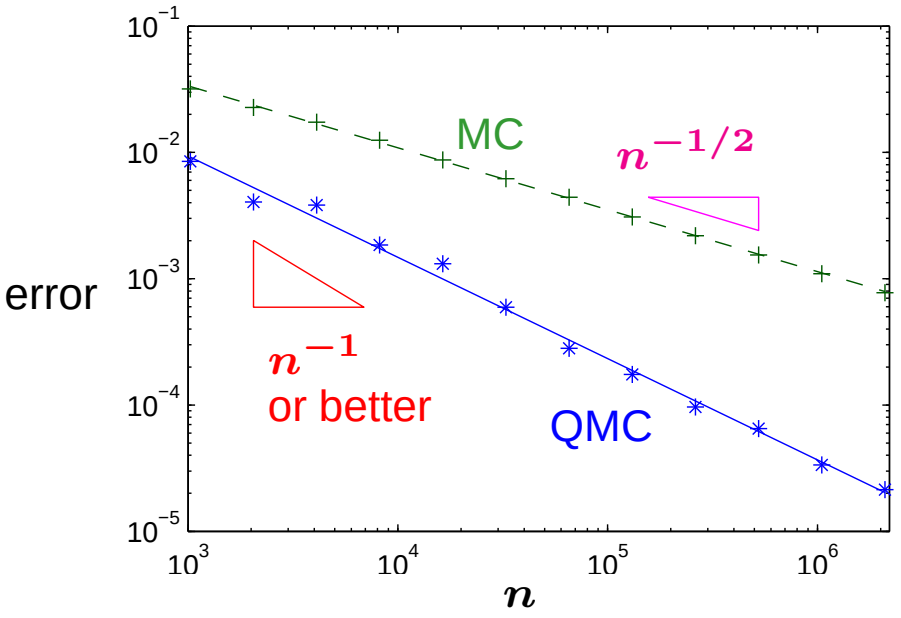
Application 3: QMC for PDEs with random coeff.

To compute the expected value of some quantity of interest:

1. Generate a number of realizations of the random field
   (Some approximation may be required)
2. For each realization, solve the PDE using e.g. FEM / FVM / mFEM
3. Take the average of all solutions from different realizations

This describes Monte Carlo simulation.

NOTE: **expected value** $=$ (high dimensional) **integral**



$\rightarrow$ use **quasi-Monte Carlo methods**

$s =$ stochastic dimension

SEE PART 2

Summary of Part 1

- QMC methods are equal-weight quadrature rules over the unit cube
  - Transformation to the unit cube plays a crucial role (also for MC and SG)
  - Better convergence rates than MC (also higher order)
  - Good for earlier variables and lower-order projections
  - Ordering the variables is very important
  - Randomized QMC:

    unbiased, simple error estimation, good convergence rate

- QMC error bound can be independent of dimension

- QMC analysis: weighted spaces, fast CBC construction