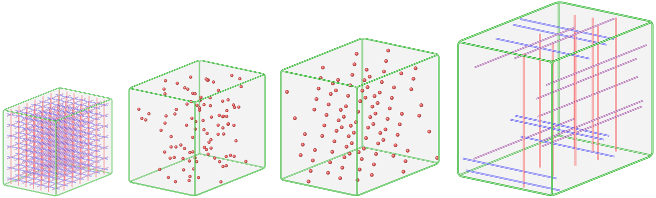
EPSRC

University of Brighton

Parallel cross interpolation for accurate evaluation of high–dimensional integrals

Dmitry Savostyanov

University of Brighton



Algorithms and applications of high dimensional approximation
Bath, 23 November 2018

Why do we need high–dimensional integrals?

EPSRC

University of Brighton

Why do we need high–dimensional integrals?

- Generalised Gaussian integrals in mathematical finance

$$\int_{\mathbb{R}^d} f(x) e^{-\frac{1}{2} x^T A x} \, dx$$

Why do we need high–dimensional integrals?

- Generalised Gaussian integrals in mathematical finance

$$\int_{\mathbb{R}^d} f(x) e^{-\frac{1}{2} x^T A x} \, dx$$

- Ising integrals in mathematical physics

$$\int_{[0,1]^{d-1}} \prod_{j \leqslant k} \left(\frac{t_k t_{k-1} \cdots t_{j+1} - 1}{t_k t_{k-1} \cdots t_{j+1} + 1} \right)^2 \frac{dt}{\left(1 + \sum_{k=2}^{d} t_2 \cdots t_k\right)\left(1 + \sum_{k=2}^{d} t_k \cdots t_d\right)}$$

Why do we need high–dimensional integrals?

- Generalised Gaussian integrals in mathematical finance

$$\int_{\mathbb{R}^d} f(x) e^{-\frac{1}{2}x^T A x}\, dx$$

- Ising integrals in mathematical physics

$$\int_{[0,1]^{d-1}} \prod_{j \leqslant k} \left(\frac{t_k t_{k-1} \cdots t_{j+1} - 1}{t_k t_{k-1} \cdots t_{j+1} + 1} \right)^2 \frac{dt}{\left(1 + \sum_{k=2}^{d} t_2 \cdots t_k\right)\left(1 + \sum_{k=2}^{d} t_k \cdots t_d\right)}$$

- Weyl's integrals in representation theory / random matrix theory

$$\int_{\mathbb{U}_d} f(U)\, dU = \frac{1}{(2\pi)^d d!} \int_{[0,2\pi]^d} f(\mathrm{diag}(e^{i\theta_1}, \ldots, e^{i\theta_d})) \prod_{j \leqslant k} \left| e^{i\theta_j} - e^{i\theta_k} \right|^2 d\theta$$

High–dimensional integrals

Why do we need high–dimensional integrals?

- Generalised Gaussian integrals in mathematical finance

$$\int_{\mathbb{R}^d} f(x) e^{-\frac{1}{2}x^T A x}\, dx$$

- Ising integrals in mathematical physics

$$\int_{[0,1]^{d-1}} \prod_{j\leqslant k} \left(\frac{t_k t_{k-1}\cdots t_{j+1}-1}{t_k t_{k-1}\cdots t_{j+1}+1}\right)^2 \frac{dt}{\left(1+\sum_{k=2}^{d} t_2\cdots t_k\right)\left(1+\sum_{k=2}^{d} t_k\cdots t_d\right)}$$

- Weyl's integrals in representation theory / random matrix theory

$$\int_{\mathbb{U}_d} f(U)\, dU = \frac{1}{(2\pi)^d d!}\int_{[0,2\pi]^d} f(\mathrm{diag}(e^{i\theta_1},\ldots,e^{i\theta_d}))\prod_{j\leqslant k}\left|e^{i\theta_j}-e^{i\theta_k}\right|^2 d\theta$$

- Stochastic and parametric PDEs $\quad \mathcal{D}_x(\xi)u(x,\xi) = f(x)$

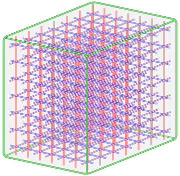
EPSRC

University of Brighton

University of Brighton

How can we compute high–dimensional integrals?

- Tensor product grid

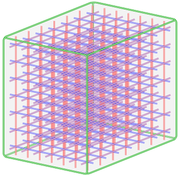


bad

Main message

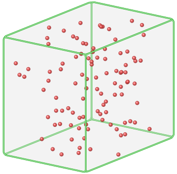
How can we compute high–dimensional integrals?

- Tensor product grid



bad

- Monte Carlo



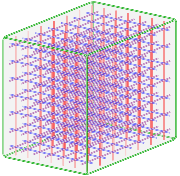
ok but slow

EPSRC

University of Brighton

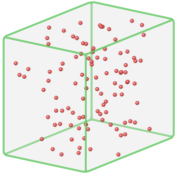
How can we compute high–dimensional integrals?

- Tensor product grid



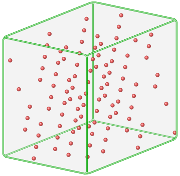
bad

- Monte Carlo



ok but slow

- quasi Monte Carlo



better

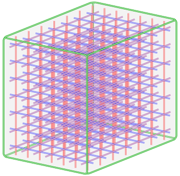
EPSRC

University of Brighton

Main message

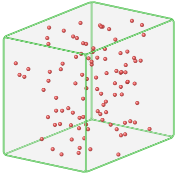
How can we compute high–dimensional integrals?

- Tensor product grid



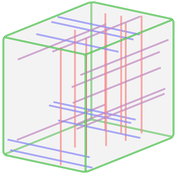
bad

- Monte Carlo



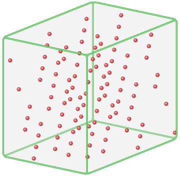
ok but slow

- Cross interpolation



even better

- quasi Monte Carlo



better

EPSRC

University of Brighton

Cross interpolation for functions

$$\int_{\mathbb{R}^3} e^{-\frac{1}{2}\|x\|^2}\,dx = \int_{\mathbb{R}}\int_{\mathbb{R}}\int_{\mathbb{R}} e^{-\frac{1}{2}(x_1^2+x_2^2+x_3^3)}\,dx_1\,dx_2\,dx_3$$

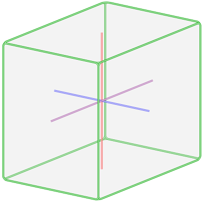
$$= \left(\int_{\mathbb{R}} e^{-\frac{1}{2}x_1^2}\,dx_1\right)\left(\int_{\mathbb{R}} e^{-\frac{1}{2}x_2^2}\,dx_2\right)\left(\int_{\mathbb{R}} e^{-\frac{1}{2}x_3^2}\,dx_3\right)$$

$$= \left(\sqrt{2\pi}\right)^3$$

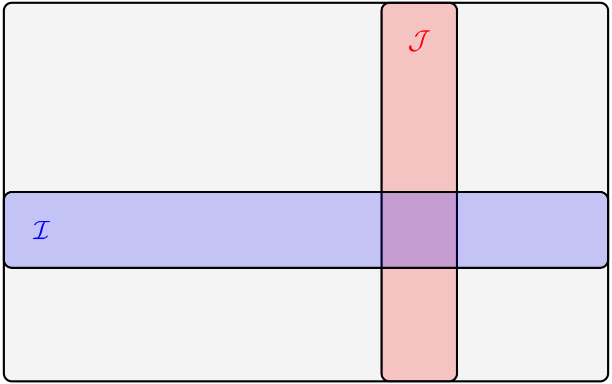
Separation of variables:

$$e^{-\frac{1}{2}(x_1^2+x_2^2+x_3^2)} = e^{-\frac{1}{2}x_1^2}\,e^{-\frac{1}{2}x_2^2}\,e^{-\frac{1}{2}x_3^2}$$

$$f(x_1,x_2,x_3) = f(x_1,0,0)f(0,x_2,0)f(0,0,x_3)$$



Cross interpolation for matrices



$$A(i,j) \approx \tilde{A}(i,j) = A(i,\mathcal{J})\,[A(\mathcal{I},\mathcal{J})]^{-1}\,A(\mathcal{I},j).$$

$mr + rn - r^2$ parameters (SVD), $mr + rn - r^2$ interpolation points

EPSRC

University of Brighton

Cross interpolation for matrices (good)

$$A = \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \end{pmatrix}$$

$$= \begin{pmatrix} 1/2 & 1/3 \\ 1/3 & 1/4 \\ 1/4 & 1/5 \\ 1/5 & 1/6 \\ 1/6 & 1/7 \end{pmatrix} \cdot \begin{pmatrix} 1/2 & 1/3 \\ 1/3 & 1/4 \end{pmatrix}^{-1} \cdot \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \end{pmatrix}$$

$$+ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.67 & 2.86 & 3.57 \\ 0 & 0 & 2.86 & 5.00 & 6.35 \\ 0 & 0 & 3.57 & 6.35 & \mathbf{8.17} \end{pmatrix} \times \mathbf{10^{-3}}$$

EPSRC

University of Brighton

Cross interpolation for matrices (not so good)

$$A = \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \end{pmatrix}$$

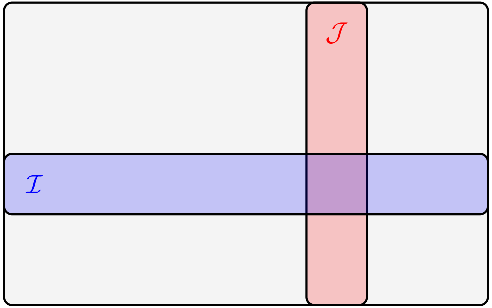
$$= \begin{pmatrix} 1/5 & 1/6 \\ 1/6 & 1/7 \\ 1/7 & 1/8 \\ 1/8 & 1/9 \\ 1/9 & 1/10 \end{pmatrix} \cdot \begin{pmatrix} 1/8 & 1/9 \\ 1/9 & 1/10 \end{pmatrix}^{-1} \cdot \begin{pmatrix} 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \end{pmatrix}$$

$$+ \begin{pmatrix} \mathbf{8.00} & 1.90 & 0.36 & 0 & 0 \\ 1.90 & 0.51 & 0.10 & 0 & 0 \\ 0.36 & 0.10 & 0.02 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \times \mathbf{10^{-2}}$$

EPSRC

University of Brighton

Cross interpolation for matrices (maxvol)



Theorem (Quasioptimality of maximum–volume cross)

$$[\mathcal{I}, \mathcal{J}] = \text{maxvol}[A(i,j)], \qquad \tilde{A}(i,j) = A(i,\mathcal{J})\,[A(\mathcal{I},\mathcal{J})]^{-1}\,A(\mathcal{I},j).$$

$$|A - \tilde{A}| \leqslant (r+1) \;\; \min_{\text{rank}\, X = r} \|A - X\|$$

$$|A - \tilde{A}| \leqslant (r+1)^2 \min_{\text{rank}\, X = r} |A - X|$$

[Goreinov, Tyrtyshnikov, 2001 & 2011]

EPSRC
University of Brighton

Cross interpolation for matrices (maxvol)

$$A = \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \end{pmatrix}$$

$$= \begin{pmatrix} 1/2 & 1/5 \\ 1/3 & 1/6 \\ 1/4 & 1/7 \\ 1/5 & 1/8 \\ 1/6 & 1/9 \end{pmatrix} \cdot \begin{pmatrix} 1/2 & 1/5 \\ 1/5 & 1/8 \end{pmatrix}^{-1} \cdot \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \end{pmatrix}$$

$$+ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{3.09} & 1.59 & 0 & -1.18 \\ 0 & 1.59 & 0.85 & 0 & -0.66 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & -1.18 & -0.66 & 0 & 0.55 \end{pmatrix} \times \mathbf{10^{-3}}$$

EPSRC
University of Brighton

Cross interpolation algorithm (step 1)

$$A = \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \end{pmatrix}$$

$$= \begin{pmatrix} 1/2 \\ 1/3 \\ 1/4 \\ 1/5 \\ 1/6 \end{pmatrix} \cdot \left(1/2\right)^{-1} \cdot \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \end{pmatrix}$$

$$- \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2.78 & 3.33 & 3.33 & 3.17 \\ 0 & 3.33 & 4.17 & 4.28 & 4.17 \\ 0 & 3.33 & 4.28 & \mathbf{4.50} & 4.44 \\ 0 & 3.17 & 4.17 & 4.44 & 4.44 \end{pmatrix} \times \mathbf{10^{-2}}$$

Cross interpolation algorithm (step 2)

$$A = \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \\ 1/6 & 1/7 & 1/8 & 1/9 & 1/10 \end{pmatrix}$$

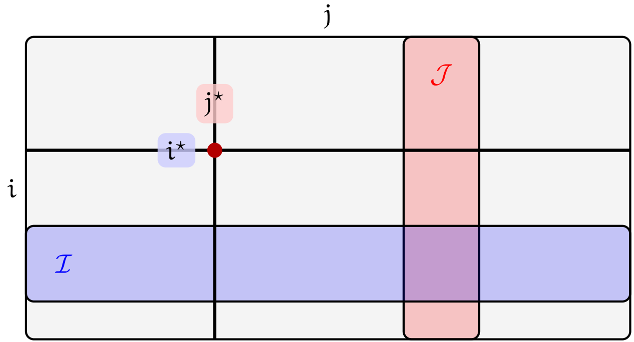
$$= \begin{pmatrix} 1/2 & 1/5 \\ 1/3 & 1/6 \\ 1/4 & 1/7 \\ 1/5 & 1/8 \\ 1/6 & 1/9 \end{pmatrix} \cdot \begin{pmatrix} 1/2 & 1/5 \\ 1/5 & 1/8 \end{pmatrix}^{-1} \cdot \begin{pmatrix} 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \end{pmatrix}$$

$$+ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{3.09} & 1.59 & 0 & -1.18 \\ 0 & 1.59 & 0.85 & 0 & -0.66 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & -1.18 & -0.66 & 0 & 0.55 \end{pmatrix} \times \mathbf{10^{-3}}$$

EPSRC

University of Brighton

Cross interpolation algorithm



Algorithm (Gaussian elimination with partial pivoting)

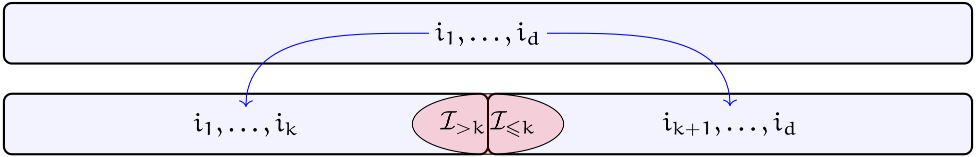
- Find a pivot $(i^\star, j^\star)$ such that $\left|A(i^\star, j^\star) - A(i^\star, \mathcal{J})[A(\mathcal{I}, \mathcal{J})]^{-1} A(\mathcal{I}, j^\star)\right|$ is large
- Add $i^\star$ to $\mathcal{I}$ and $j^\star$ to $\mathcal{J}$
- Update columns $[A(i, \mathcal{J})]$, submatrix $[A(\mathcal{I}, \mathcal{J})]^{-1}$, and rows $[A(\mathcal{I}, j)]$

EPSRC

University of Brighton

Matrix interpolation for tensors

$$\begin{aligned} A(i_1,\ldots,i_k;i_{k+1},\ldots,i_d) &= A(i_{\leqslant k}, i_{>k}) \approx \tilde{A}(i_{\leqslant k}, i_{>k}) \\ &= A(i_{\leqslant k}, \mathcal{I}_{>k}) \left[A(\mathcal{I}_{\leqslant k}, \mathcal{I}_{>k})\right]^{-1} A(\mathcal{I}_{\leqslant k}, i_{>k}) \\ &= A(i_1 \ldots i_k, \mathcal{I}_{>k}) \left[A(\mathcal{I}_{\leqslant k}, \mathcal{I}_{>k})\right]^{-1} A(\mathcal{I}_{\leqslant k}, i_{k+1} \ldots i_d) \end{aligned}$$

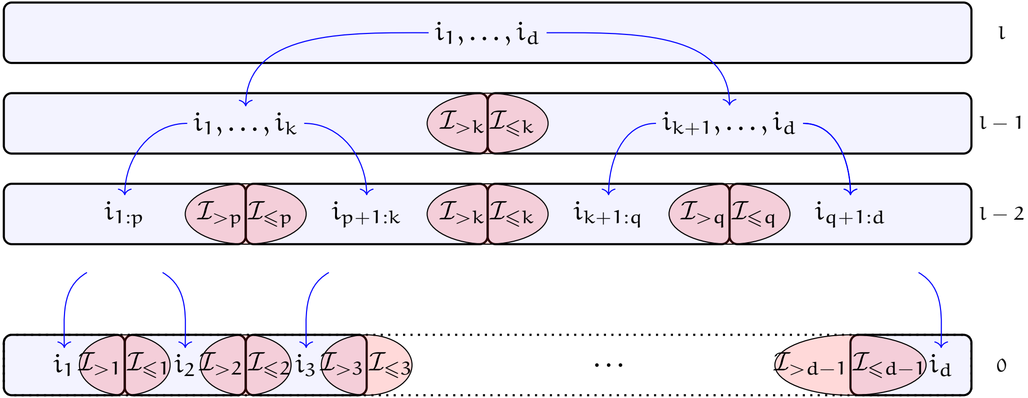


Error estimate:

$$|A - \tilde{A}| \leqslant (r_k + 1)^2 \min |A - X|$$

EPSRC
University of Brighton

Cross interpolation for tensors (theory)

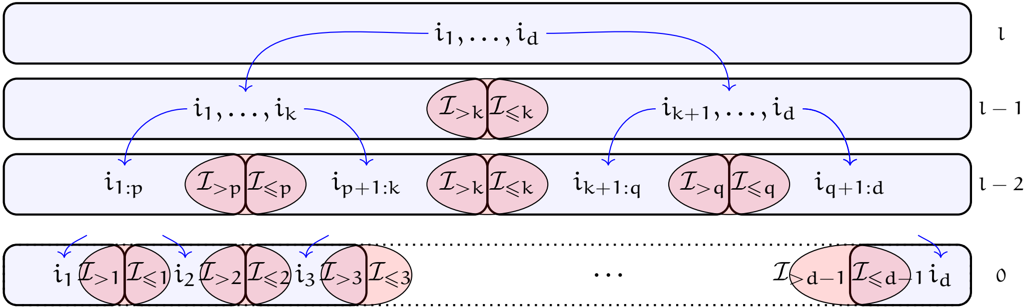


Theorem (Cross–aproximation formula for tensors)

$$\tilde{A}(i_1, i_2 \dots, i_d) = A(i_1, \mathcal{I}_{>1})\,[A(\mathcal{I}_{\leqslant 1}, \mathcal{I}_{>1})]^{-1}\,A(\mathcal{I}_{\leqslant 1}, i_2, \mathcal{I}_{>2}) \dots A(\mathcal{I}_{\leqslant d-1}, i_d)$$

[Oseledets, Tyrtyshnikov, 2010] [Savostyanov, Oseledets, 2011]

Cross interpolation for tensors (theory)



Theorem (Quasioptimality of maxvol cross)

If $[\mathcal{I}_{\leqslant k}, \mathcal{I}_{>k}] = \text{maxvol}[A(i_{\leqslant k}, i_{>k})]$ then

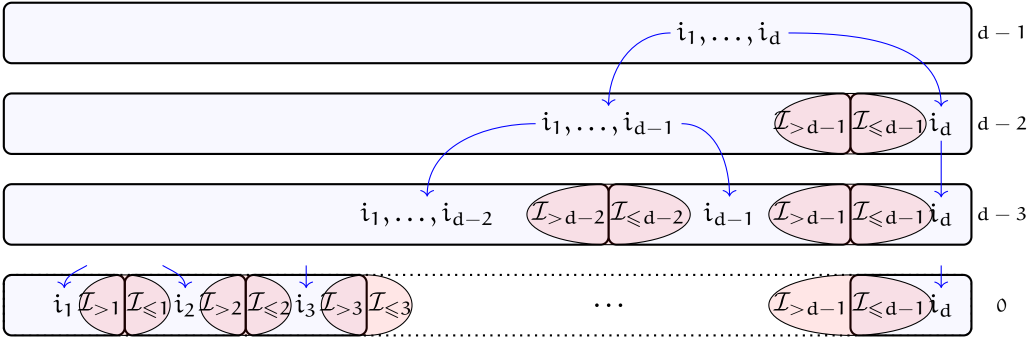
$$|A - \tilde{A}| \leqslant (2r + \kappa r + 1)^{\lceil \log_2 d \rceil} (r+1)^2 \min |A - X|$$

where $r = \max r_k, \quad \kappa = \max \kappa_k, \quad \kappa_k = r_k |A| |A_k^{-1}|$

[Savostyanov, 2014]

EPSRC
University of Brighton

Cross interpolation for tensors (theory)

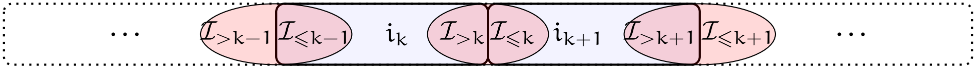


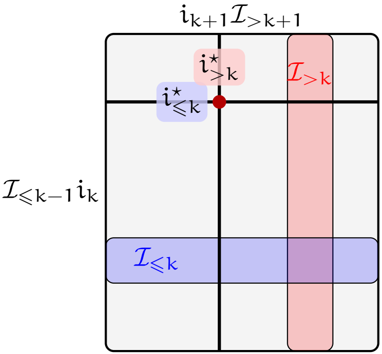
Theorem (Interpolation property for tensors)

If indices are nested $i_{>k} \in \mathcal{I}_{>k} \Rightarrow i_{>k+1} \in \mathcal{I}_{>k+1}$, $\quad i_{\leqslant k} \in \mathcal{I}_{\leqslant k} \Rightarrow i_{\leqslant k-1} \in \mathcal{I}_{\leqslant k-1}$,
then interpolation formula interpolates in all points $A(\mathcal{I}_{\leqslant k-1}, i_k, \mathcal{I}_{>k}) = \tilde{A}(\mathcal{I}_{\leqslant k-1}, i_k, \mathcal{I}_{>k})$

[Savostyanov, 2014]

Cross interpolation for tensors (practice)





Algorithm (DMRG–like tensor cross interpolation)

do $k = 1, \dots, d-1$ (half–sweep)

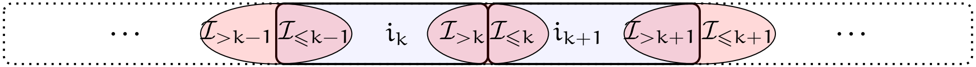
- Find a pivot $i^\star = (i^\star_{\leqslant k}, i^\star_{>k})$ in the superblock $\left[A(\mathcal{I}^{\leqslant k-1} i_k, i_{k+1}\mathcal{I}^{>k+1})\right]$
- Add $i^\star_{\leqslant k}$ to $\mathcal{I}_{\leqslant k}$, and $i^\star_{>k}$ to $\mathcal{I}_{>k}$
- Update the TT–cores $\left[A(\mathcal{I}^{\leqslant k-1}, i_k, \mathcal{I}^{>k})\right]$ and $\left[A(\mathcal{I}^{\leqslant k}, i_{k+1}, \mathcal{I}^{>k+1})\right]$ of the interpolation $\tilde{A}$

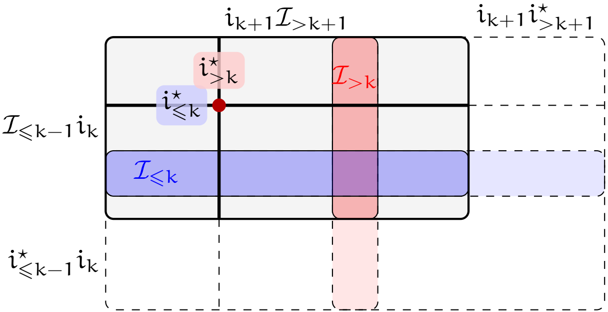
Complexity $\sum_{k=1}^{d} r_{k-1} n_k r_k - \sum_{k=1}^{d-1} r_k$ tensor entries plus $\mathcal{O}(dnr^3)$ operations

[Savostyanov, 2014]

EPSRC
University of Brighton

Cross interpolation for tensors (practice)





Parallel algorithm

- Along cores: (MPI)
  update all crosses $[\mathcal{I}^{\leqslant k}, \mathcal{I}^{>k}]$ simultaneously for $k = 1, \dots, d-1$
- Along mode: (OMP)
  compute all entries $A[\mathcal{I}^{\leqslant k}, i_k, \mathcal{I}^{>k}]$ simultaneously for $i_k = 1, \dots, n_k$

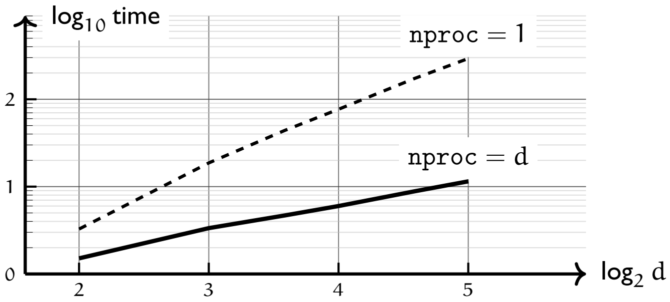
Related work

- [Grasedyck et al, Comput. Visual. Sci. 2015] Parallel random sampling in HT
- [Etter, SISC 2016] Parallel ALS for solving linear systems

EPSRC

University of Brighton

Parallel cross interpolation algorithm



- implemented in Fortran+MPI (+OpenMP), works better than Matlab+SPD
- sequential code scales quadratically with d
- parallel code scales linearly with d (each element costs $\sim d$)

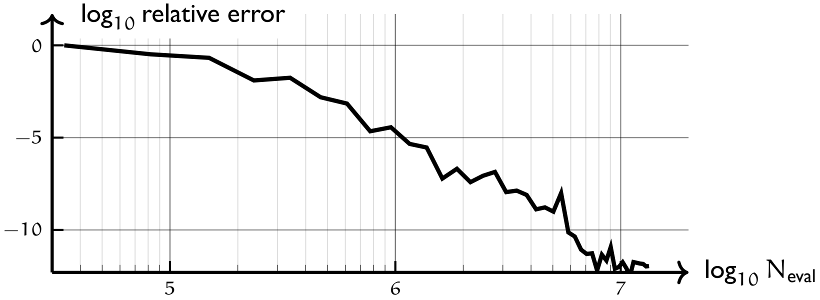
EPSRC

University of Brighton

Generalised Gaussian

$$\int_{\mathbb{R}^d} e^{-\frac{1}{2}x^T A x}\, dx = \frac{(2\pi)^{d/2}}{\sqrt{\det(A)}}$$

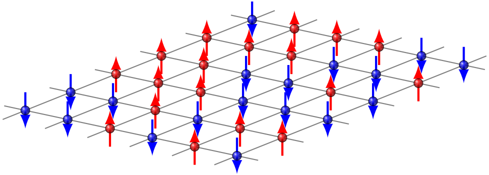
- $A = \left[e^{-|i-j|}\right]_{i,j=1}^{d}$, $d = 100$, $n = 50$ nodes in each mode
- $n^d = 50^{100}$ quadrature nodes overall — **curse of dimensionality**



EPSRC

University of Brighton

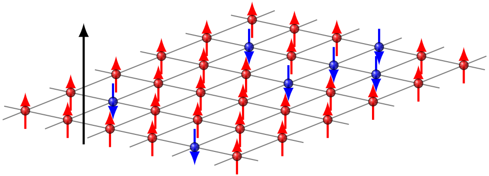
2D Ising model



- electrons have spins: $|\uparrow\rangle$ or $|\downarrow\rangle$

EPSRC
University of Brighton

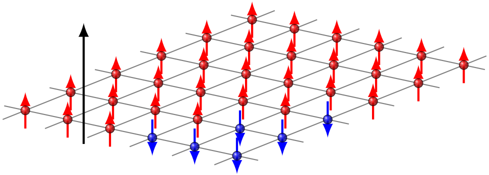
2D Ising model



- electrons have spins: $|\uparrow\rangle$ or $|\downarrow\rangle$
- external magnetic field aligns the spins

Ising integrals

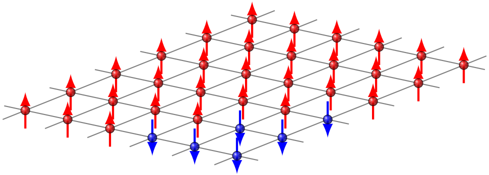
2D Ising model



- electrons have spins: $|\uparrow\rangle$ or $|\downarrow\rangle$
- external magnetic field aligns the spins
- in ferromagnetics, aligned spins form domains

Ising integrals

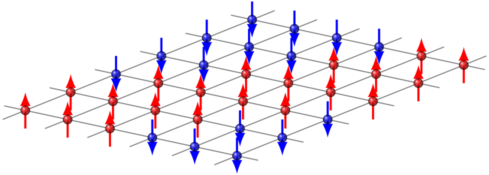
2D Ising model



- electrons have spins: $|\uparrow\rangle$ or $|\downarrow\rangle$
- external magnetic field aligns the spins
- in ferromagnetics, aligned spins form domains
- domains persist even when the external field is zero

Ising integrals

2D Ising model

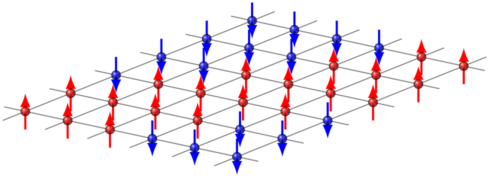


- electrons have spins: $|\uparrow\rangle$ or $|\downarrow\rangle$
- external magnetic field aligns the spins
- in ferromagnetics, aligned spins form domains
- domains persist even when the external field is zero
- heating and cooling removes the total magnetisation, but domains still form *spontaneously*

EPSRC
University of Brighton

Ising integrals

2D Ising model



- electrons have spins: $|\uparrow\rangle$ or $|\downarrow\rangle$
- external magnetic field aligns the spins
- in ferromagnetics, aligned spins form domains
- domains persist even when the external field is zero
- heating and cooling removes the total magnetisation, but domains still form *spontaneously*

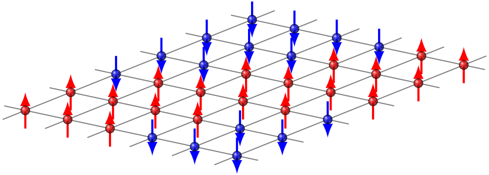
- Systems with next–neighbour interaction exhibit co-operative behavior
  (the same effect in gas–liquid transition, binary alloys, biology, genetics, economics, etc)

EPSRC

University of Brighton

Ising integrals

2D Ising model

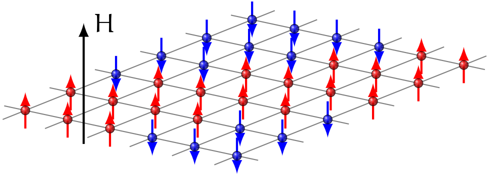


- electrons have spins: $|\uparrow\rangle$ or $|\downarrow\rangle$
- external magnetic field aligns the spins
- in ferromagnetics, aligned spins form domains
- domains persist even when the external field is zero
- heating and cooling removes the total magnetisation, but domains still form *spontaneously*

- Systems with next–neighbour interaction exhibit co-operative behavior (the same effect in gas–liquid transition, binary alloys, biology, genetics, economics, etc)
- Phase transition effect: $\begin{cases} \text{spontaneous magnetisation}, & \text{when } T < T_c \\ \text{demagnetisation}, & \text{when } T > T_c \end{cases}$

Ising integrals

2D Ising model



- spin configuration $\sigma_{i,j} \in \{\uparrow, \downarrow\} = \{+1, -1\}$
- energy of configuration

$$E(\sigma) = \underbrace{-\sum_{i,j} \sigma_{i,j}\sigma_{i,j+1} - \sum_{i,j} \sigma_{i,j}\sigma_{i+1,j}}_{\text{next neighbour interaction}} - \underbrace{H \sum_{i,j} \sigma_{i,j}}_{\text{response to magnetic field}}$$

- Gibbs measure — probability of configuration $P(\sigma) = \exp(-E(\sigma)/kT)/Z$
- Partition function — sum of all measures $Z(T, H) = \sum_\sigma \exp(-E(\sigma)/kT)$
- Helmholtz free energy $F = -kT \log Z(T, H)$, per particle $f(T, h) = \lim_{N\to\infty} F(T, H)/N$
- Spontaneous magnetisation $m_0(T) = \left.\frac{\partial f}{\partial H}\right|_{H=0}$
- Susceptibility $\chi_0(T) = \left.\frac{\partial^2 f}{\partial H^2}\right|_{H=0}$ related to spin–spin correlation $\langle \sigma_{0,0}\sigma_{m,n}\rangle$

EPSRC
University of Brighton

Ising integrals

Ising susceptibility integrals

- Phase transition (Curie temperature)

$$\chi_{\pm}(T) \sim C_{0,\pm} |1 - T/T_c|^{-7/4}$$

- Susceptibility amplitudes

$$C_{0,+} \sim \sum_{n\,\text{odd}} \frac{\pi D_n}{(2\pi)^n}, \qquad C_{0,-} \sim \sum_{n\,\text{even}} \frac{\pi D_n}{(2\pi)^n},$$

$$D_n = \frac{4}{n!} \int_{\mathbb{R}^n_+} \frac{\prod_{j<k} \left(\frac{u_j - u_k}{u_j + u_k}\right)^2}{\left(\sum_{k=1}^n (u_k + 1/u_k)\right)^2} \frac{du}{u_1 \cdots u_n}$$

[Wu, McCoy, Tracy, Barouch, 1976]

EPSRC

University of Brighton

Ising susceptibility integrals

$$A = \prod_{j \leqslant k} \left(\frac{t_k t_{k-1} \cdots t_{j+1} - 1}{t_k t_{k-1} \cdots t_{j+1} + 1} \right)^2$$

$$1/B = \left(1 + \sum_{k=2}^{d} t_2 \cdots t_k \right) \left(1 + \sum_{k=2}^{d} t_k \cdots t_d \right)$$

$$C_n = 2 \int_{[0,1]^{d-1}} B(t_2, \cdots, t_n)\, \mathrm{d}t$$

$$D_n = 2 \int_{[0,1]^{d-1}} (AB)(t_2, \cdots, t_n)\, \mathrm{d}t$$

$$E_n = 2 \int_{[0,1]^{d-1}} A(t_2, \cdots, t_n)\, \mathrm{d}t$$

Ising integrals

Ising susceptibility integrals

$$A = \prod_{j \leqslant k} \left(\frac{t_k t_{k-1} \cdots t_{j+1} - 1}{t_k t_{k-1} \cdots t_{j+1} + 1} \right)^2$$

$$1/B = \left(1 + \sum_{k=2}^{d} t_2 \cdots t_k \right) \left(1 + \sum_{k=2}^{d} t_k \cdots t_d \right)$$

$$C_n = 2 \int_{[0,1]^{d-1}} B(t_2, \cdots, t_n)\, \mathrm{d}t$$

$$D_n = 2 \int_{[0,1]^{d-1}} (AB)(t_2, \cdots, t_n)\, \mathrm{d}t$$

$$E_n = 2 \int_{[0,1]^{d-1}} A(t_2, \cdots, t_n)\, \mathrm{d}t$$

- C_n can be reduced to 2–dimensional integral and computed to high precision. Extrapolation for $n \to \infty$ gives

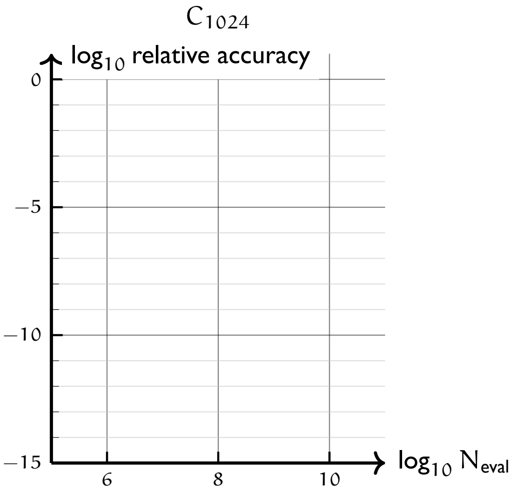
$$C_\infty = 0.63047350337438679611204\ldots$$
$$= 2e^{-2\gamma}$$

This result was then proven analytically — experiment leading to theory.

- Similarly, high–precision numerical evaluation of D_n can lead to analytic formulas and physical insights

[Bailey, Borwein, Crandall, 2006]

Verification



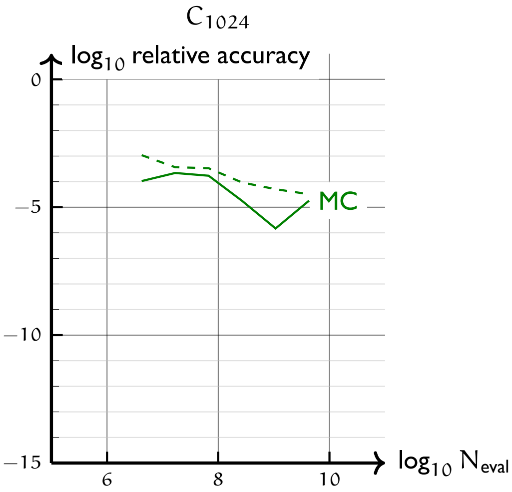
- C_{1024} is reduced to two–dimensional integral and computed to 500 digits [Bailey, Borwein, Crandall, 2006]
- We compute C_{1024} as 1023–dimensional integral to verify cross interpolation

EPSRC

University of Brighton

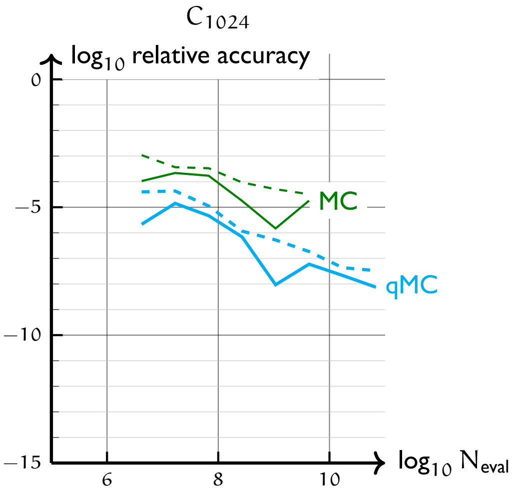
Ising integrals

Verification



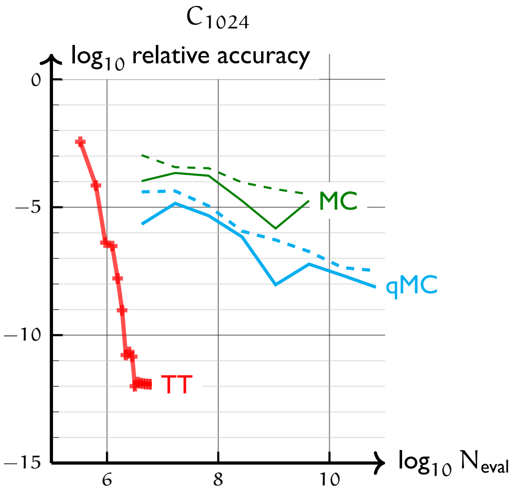
- C_{1024} is reduced to two–dimensional integral and computed to 500 digits [Bailey, Borwein, Crandall, 2006]
- We compute C_{1024} as 1023–dimensional integral to verify cross interpolation
- MC converges as $\sim N^{-1/2}$ (dashed line — estimate, solid — error)

Verification



- C_{1024} is reduced to two–dimensional integral and computed to 500 digits [Bailey, Borwein, Crandall, 2006]
- We compute C_{1024} as 1023–dimensional integral to verify cross interpolation
- MC converges as $\sim N^{-1/2}$ (dashed line — estimate, solid — error)
- qMC converges as $\sim N^{-0.76}$ (optimised lattice [F. Y. Kuo, 2003])

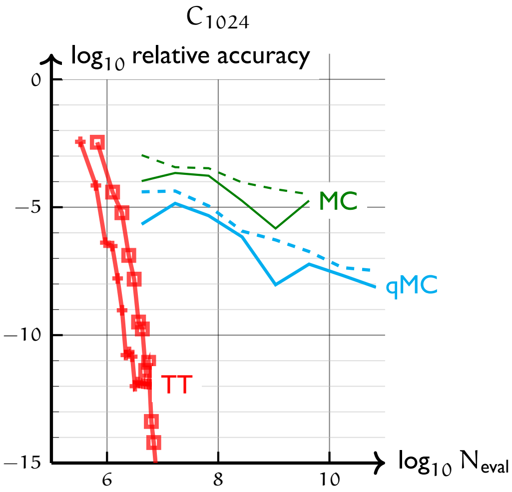
Verification



- C_{1024} is reduced to two–dimensional integral and computed to 500 digits [Bailey, Borwein, Crandall, 2006]
- We compute C_{1024} as 1023–dimensional integral to verify cross interpolation
- MC converges as $\sim N^{-1/2}$ (dashed line — estimate, solid — error)
- qMC converges as $\sim N^{-0.76}$ (optimised lattice [F. Y. Kuo, 2003])
- TT cross interpolation

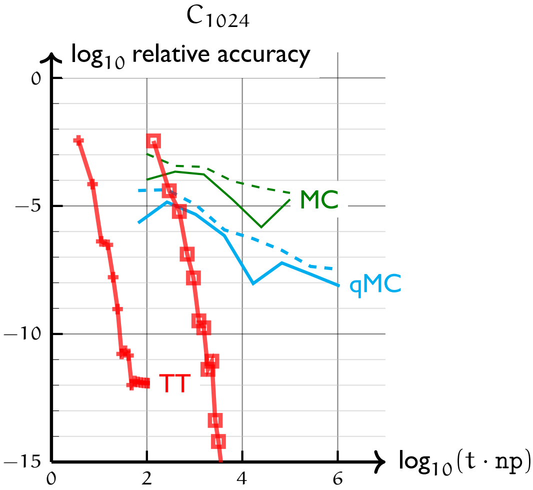
EPSRC
University of Brighton

Verification



- C_{1024} is reduced to two–dimensional integral and computed to 500 digits [Bailey, Borwein, Crandall, 2006]
- We compute C_{1024} as 1023–dimensional integral to verify cross interpolation
- MC converges as $\sim N^{-1/2}$ (dashed line — estimate, solid — error)
- qMC converges as $\sim N^{-0.76}$ (optimised lattice [F. Y. Kuo, 2003])
- TT cross interpolation (+ — double precision, □ — quad precision)

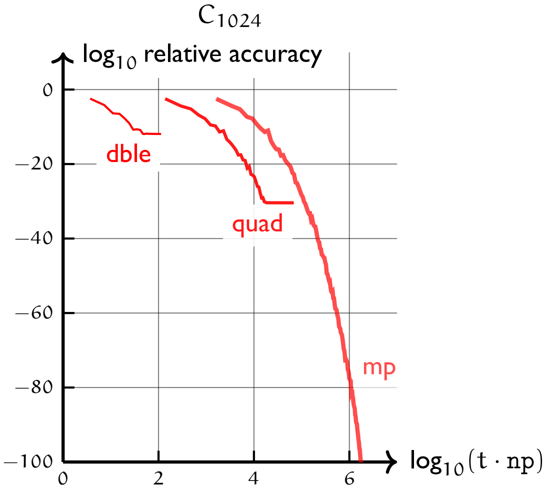
Benchmarking



- TT cross interpolation uses $\mathcal{O}(dnr^3)$ linear algebra operations on top of dnr^2 function evaluations — it's fair to compare CPU time
- Quadruple precision is obtained using `gfortran`'s option `default-real-8`; BLAS/Lapack compiled from the reference source (i.e. not optimised for speed)

Ising integrals

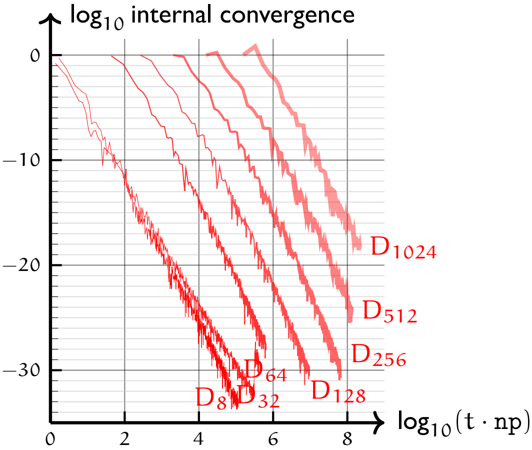
Benchmarking



- TT cross interpolation uses $\mathcal{O}(dnr^3)$ linear algebra operations on top of dnr^2 function evaluations — it's fair to compare CPU time
- Quadruple precision is obtained using `gfortran`'s option `default-real-8`; BLAS/Lapack compiled from the reference source (i.e. not optimised for speed)
- Multiple precision is obtained using `MPFUN2015` package [D. H. Bailey]; BLAS/Lapack functions written from scratch (i.e. not optimised at all)
- Is it exponential convergence we see?

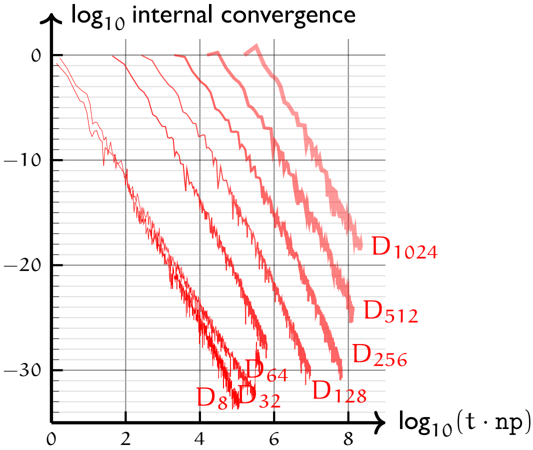
Ising integrals

Exploration



- Each element of D_n cost $\mathcal{O}(n^2)$ which is n times more expensive than C_n
- For D_{1024} can get 18 digits using 4 days on 512 nodes of Balena (U Bath)

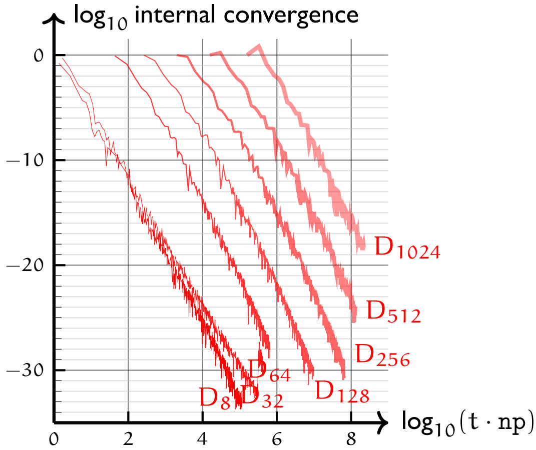
Exploration



- Each element of D_n cost $\mathcal{O}(n^2)$ which is n times more expensive than C_n
- For D_{1024} can get 18 digits using 4 days on 512 nodes of Balena (U Bath)
- Good news: the observed convergence of TT cross interpolation is $O(N^{-7.5})$

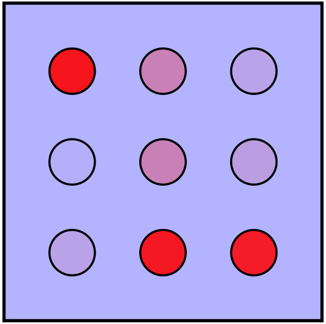
Ising integrals

Exploration



- Each element of D_n cost $\mathcal{O}(n^2)$ which is n times more expensive than C_n
- For D_{1024} can get 18 digits using 4 days on 512 nodes of Balena (U Bath)
- Good news: the observed convergence of TT cross interpolation is $O(N^{-7.5})$
- Other news: we are still far from the 100 digit target that enables us to use inverse symbolic calculators

Cookies problem



$$-\nabla_x(a(x,p)\nabla_x u(x,p)) = 1 \quad x \in \Omega$$
$$u(x,p) = 0 \quad x \in \partial\Omega$$

$$a(x,p) = \begin{cases} p_{s,t} & x \in \text{cookie}_{s,t} \\ 1 & \text{otherwise} \end{cases}$$

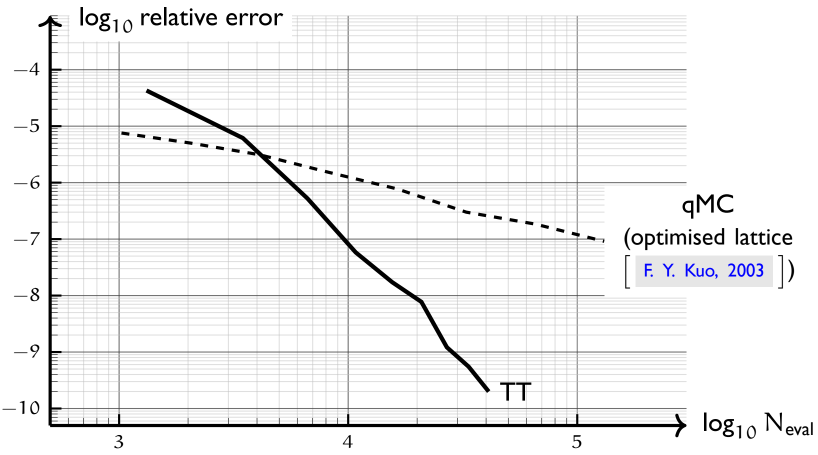
[Ballani, Grasedyck, 2014]

- We have 'only' 3×3 parameters $p_{s,t} \in [\frac{1}{2}, 2]$
- Take 'only' $n = 10$ Chebyshev points to discretise each $p_{s,t}$
- Face $n^d = 10^9$ PDEs to solve — **curse of dimensionality**
- Apply tensor formats to remedy the situation

EPSRC

University of Brighton

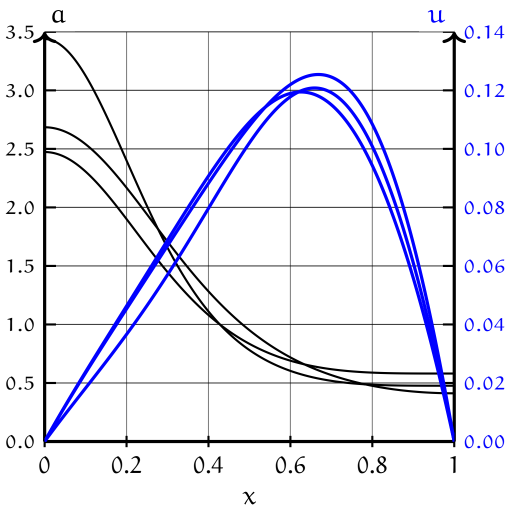
Parametric problems



- value of interest $f(p) = \int_\Omega u(x,p)\,dx$ (method is now limited to scalar tensors)
- tensor $f(p)$ of size $10 \times \cdots \times 10$ stored in full (in TT format)
- any satistics (mean, moments, convolutions) can be computed instantly

Stochastic problems

Karhunen-Loéve expansion

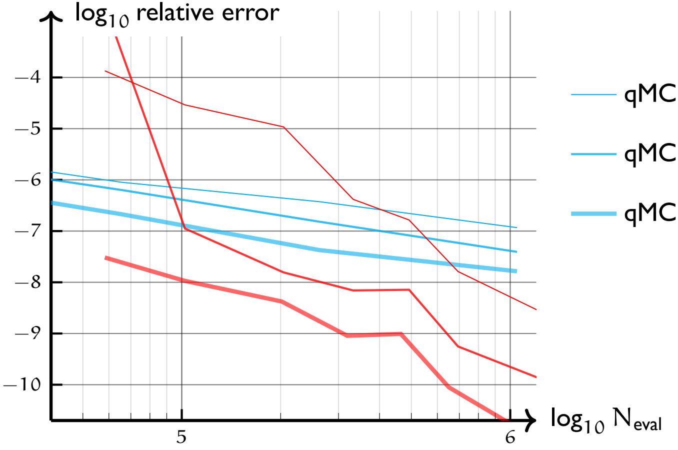


$$-\nabla_x(a(x,\xi)\nabla_x u(x,\xi)) = 1 \quad x\in[0,1]$$
$$u(x,\xi) = 0 \quad x\in\{0,1\}$$

$$a(x,\xi) = \exp\left(\sum_{k=1}^{d}\frac{1}{k^\gamma}\cos(k\pi x)\xi_k\right)$$

- $d \sim 10$ parameters $\xi_k \in [-1,1]$
- $n = 15$ LG quadrature points per ξ_k
- $n^d = 15^{10}$ PDEs to solve
- **curse of dimensionality**
- apply tensor formats

Stochastic problems



- TT-cross interpolation converges faster than quasi Monte Carlo
- TT-cross interpolation likes smooth coefficients
- Quasi Monte Carlo is preferable when low accuracy is required

EPSRC
University of Brighton

Conclusions

- tensor interpolation is accurate (for maxvol sets)
- tensor interpolation formula interpolates (for nested sets)
- tensor interpolation algorithm is fast, scalable and reliable

References

- Quasioptimality of maximum-volume cross interpolation of tensors
  Linear Algebra and Applications 458:217–244, 2014.