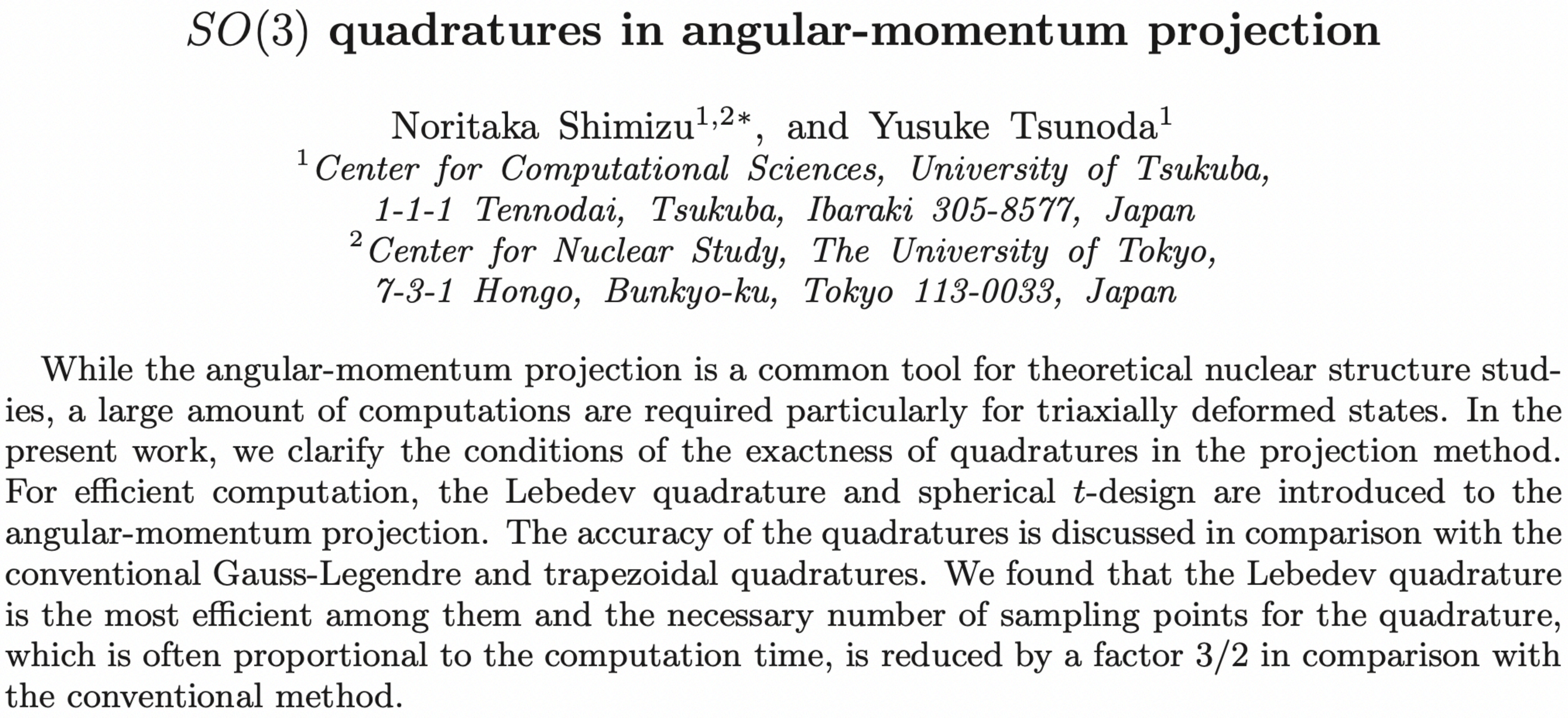
SO(3) quadratures in angular-momentum projection

Jiangming Yao (尧江明)
中山大学物理与天文学院
School of Physics and Astronomy
Sun Yat-sen University

Journal Club, Nuclear Physics Group

Nov. 07, 2022

Abstract



$SO(3)$ quadratures in angular-momentum projection

Noritaka Shimizu[1,2*], and Yusuke Tsunoda[1]
[1] *Center for Computational Sciences, University of Tsukuba, 1-1-1 Tennodai, Tsukuba, Ibaraki 305-8577, Japan*
[2] *Center for Nuclear Study, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan*

While the angular-momentum projection is a common tool for theoretical nuclear structure studies, a large amount of computations are required particularly for triaxially deformed states. In the present work, we clarify the conditions of the exactness of quadratures in the projection method. For efficient computation, the Lebedev quadrature and spherical t-design are introduced to the angular-momentum projection. The accuracy of the quadratures is discussed in comparison with the conventional Gauss-Legendre and trapezoidal quadratures. We found that the Lebedev quadrature is the most efficient among them and the necessary number of sampling points for the quadrature, which is often proportional to the computation time, is reduced by a factor 3/2 in comparison with the conventional method.

https://arxiv.org/pdf/2205.04119.pdf

Computer Physics Communication, in press (2022)

Introduction

The angular-momentum projector is defined as

$$\hat{P}^J_{MK} = \frac{2J+1}{8\pi^2}\int d\Omega D^{J*}_{MK}(\Omega)\hat{R}(\Omega)$$

$\hat{R}(\Omega)$ is the rotation operator as

$$\hat{R}(\Omega) = e^{i\hat{J}_z\alpha}e^{i\hat{J}_y\beta}e^{i\hat{J}_z\gamma}.$$

D^J_{MK} is the Wigner D-function and is defined as

$$D^J_{MK}(\Omega) = e^{iM\alpha}d^J_{MK}(\beta)e^{iK\gamma}$$

Introduction

The wave function and its energy E are evaluated by solving the generalized eigenvalue problem

$$\sum_{jK} H^{(J)}_{iM,jK} g_{jK} = E \sum_{jK} N^{(J)}_{iM,jK} g_{jK}$$

$$\begin{aligned}
H^{(J)}_{iM,jK} &= \langle \phi_i | \hat{H} \hat{P}^J_{MK} | \phi_j \rangle \\
&= \frac{2J+1}{8\pi^2} \int d\Omega D^{J*}_{MK}(\Omega) \langle \phi_i | \hat{H} \hat{R}(\Omega) | \phi_j \rangle \\
N^{(J)}_{iM,jK} &= \langle \phi_i | \hat{P}^J_{MK} | \phi_j \rangle \\
&= \frac{2J+1}{8\pi^2} \int d\Omega D^{J*}_{MK}(\Omega) \langle \phi_i | \hat{R}(\Omega) | \phi_j \rangle
\end{aligned}$$

Introduction

Expansion the deformed wave function in terms of angular–momentum eigenfunctions

$$|\phi\rangle = \sum_{I=0\,\mathrm{or}\,\frac{1}{2}}^{I_{\max}} \sum_{K=-I}^{I} v_{IK} |I, K\rangle$$

With the relation

$$\hat{R}(\Omega)|I, K\rangle = \sum_{M=-I}^{I} D^{I}_{MK}(\Omega)|I, M\rangle.$$

Then, the angular-momentum projected state is

$$\hat{P}^{J}_{MK}|\phi\rangle = \frac{2J+1}{8\pi^2} \int d\Omega D^{J*}_{MK}(\Omega) \times \sum_{I=0\,\mathrm{or}\,\frac{1}{2}}^{I_{\max}} \sum_{M',K'=-I}^{I} v_{IK'} D^{I}_{M'K'}(\Omega)|I, M'\rangle,$$

Introduction

Orthogonality condition:

$$\begin{aligned} & D^{J*}_{MK}(\Omega) D^{I}_{M'K'}(\Omega) \\ = & \sum_{J'=|J-I|}^{J+I} (-1)^{M-K} \langle J, -M, I, M' | J', M'-M \rangle \\ \times & \langle J, -K, I, K' | J', K'-K \rangle D^{J'}_{M'-M, K'-K}(\Omega) \quad (\end{aligned}$$

Finally,

$$\begin{aligned} \hat{P}^{J}_{MK} |\phi\rangle = & \sum_{I, M', K', J'} c^{J,I,J'}_{M,M',K,K'} |I, M'\rangle \\ & \times \int d\Omega D^{J'}_{M'-M, K'-K}(\Omega), \end{aligned}$$

where $c^{J,I,J'}_{M,M',K,K'}$ are coefficients depending on their indices and $v_{IK'}$, and independent of Ω. As a consequence, the integral in Eq. (12) can be performed exactly by using a quadrature rule with a degree of exactness $t = J + I_{\max}$.

Introduction

- The computational resource for the projection procedure is approximately proportional to the number of the sampling points in the quadrature to evaluate the integral of Euler angles in the projection operator.

- It is greatly valuable to find an efficient way to reduce the number of the points as much as possible and to save the computation time in various theoretical frameworks containing the angular-momentum projection.

Introduction

- The trapezoidal (T) the Gauss-Legendre (GL) quadratures

$$\begin{aligned}\hat{P}^J_{MK}|\phi\rangle &= \sum_{I,M',K',J'} c^{J,I,J'}_{M,M',K,K'}|I,M'\rangle \\ &\times \int_0^{2\pi} d\alpha\, e^{i(M'-M)\alpha} \\ &\times \int_0^{\pi} \sin\beta d\beta\, d^{J'}_{M'-M,K'-K}(\beta) \\ &\times \int_0^{2\pi} d\gamma\, e^{i(K'-K)\gamma}.\end{aligned}$$

trapezoidal $$\int_0^{2\pi} d\gamma\, e^{i(K'-K)\gamma} = \frac{2\pi}{N_z}\sum_{m=1}^{N_z} e^{i(K'-K)\frac{2\pi m}{N_z}}$$

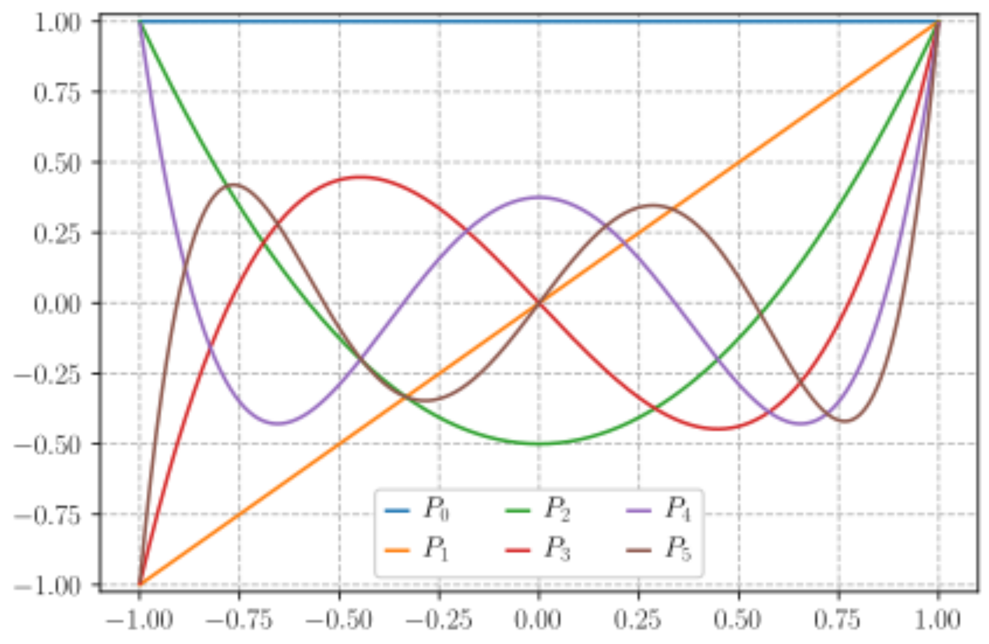
The minimum number of the Nz for exact quadrature

$$N_z = J + I_{\max} + 1 = t + 1.$$

I_max is the maximum angular momentum contained in the wave function.

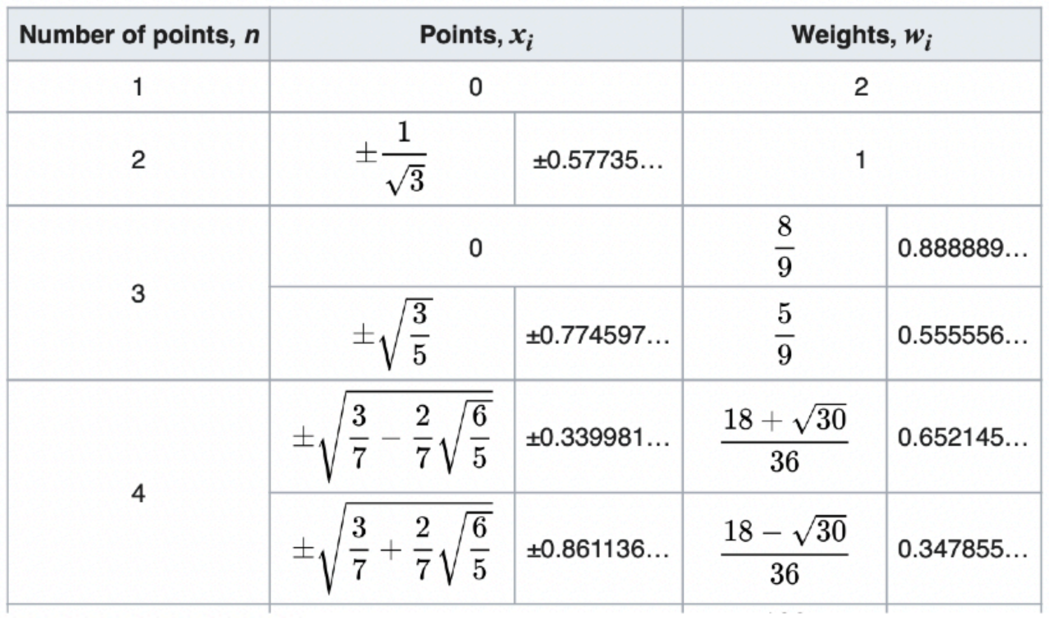
Introduction

Gauss–Legendre quadrature



$$\int_{-1}^{1} f(x)\,dx \approx \sum_{i=1}^{n} w_i f(x_i)$$

$$w_i = \frac{2}{\left(1 - x_i^2\right)\left[P_n'(x_i)\right]^2}.$$



Number of points, n	Points, x_i		Weights, w_i	
1	0		2	
2	$\pm\frac{1}{\sqrt{3}}$	±0.57735…	1	
3	0		$\frac{8}{9}$	0.888889…
	$\pm\sqrt{\frac{3}{5}}$	±0.774597…	$\frac{5}{9}$	0.555556…
4	$\pm\sqrt{\frac{3}{7} - \frac{2}{7}\sqrt{\frac{6}{5}}}$	±0.339981…	$\frac{18+\sqrt{30}}{36}$	0.652145…
	$\pm\sqrt{\frac{3}{7} + \frac{2}{7}\sqrt{\frac{6}{5}}}$	±0.861136…	$\frac{18-\sqrt{30}}{36}$	0.347855…

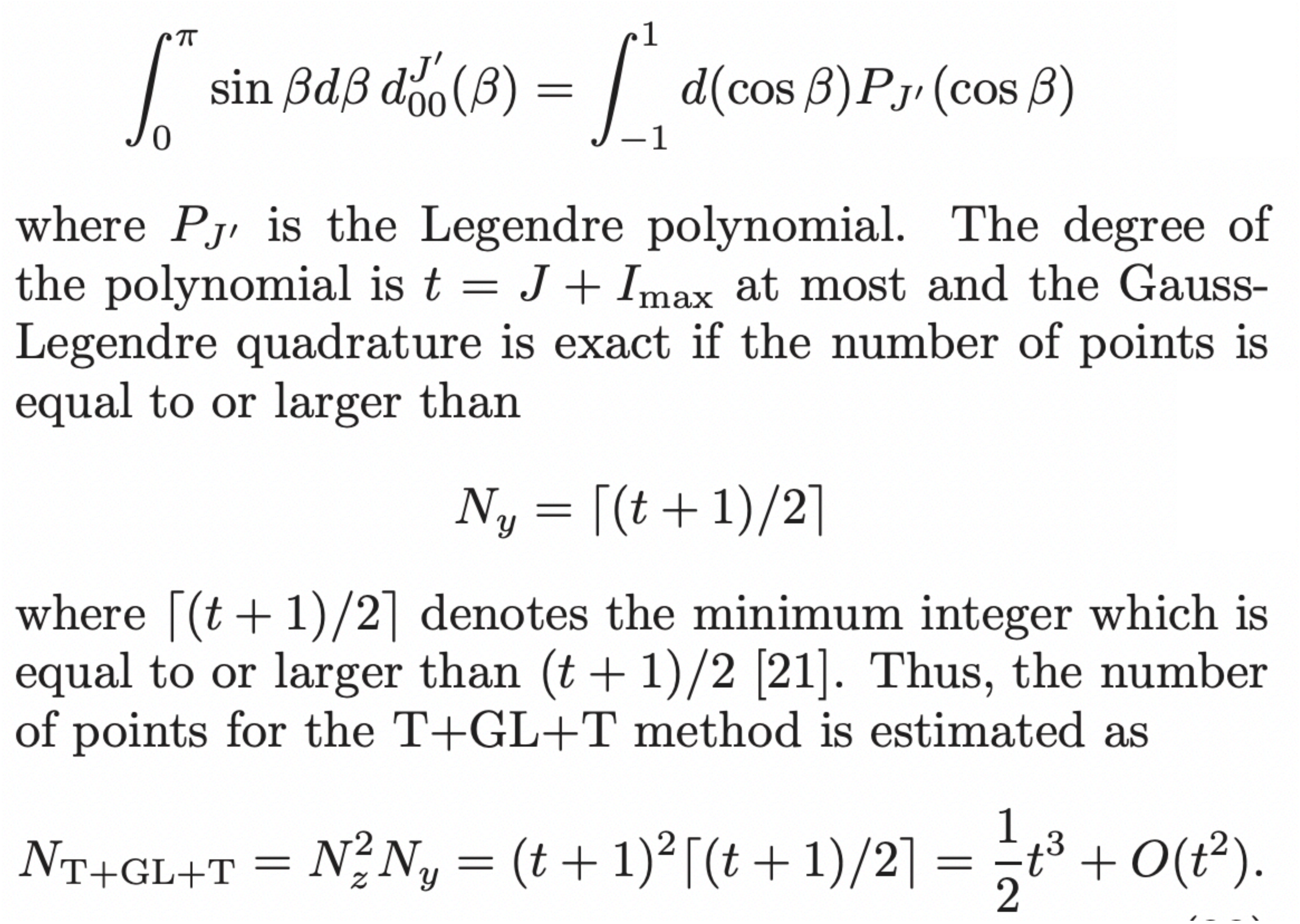
where

- *n* is the number of sample points used,
- w_i are quadrature weights, and
- x_i are the roots of the *n*th Legendre polynomial.

This choice of quadrature weights w_i and quadrature nodes x_i is the unique choice that allows the quadrature rule to integrate degree $2n-1$ polynomials exactly.

Introduction

Gauss–Legendre quadrature



$$\int_0^\pi \sin\beta d\beta\, d^{J'}_{00}(\beta) = \int_{-1}^1 d(\cos\beta) P_{J'}(\cos\beta)$$

where $P_{J'}$ is the Legendre polynomial. The degree of the polynomial is $t = J + I_{\max}$ at most and the Gauss-Legendre quadrature is exact if the number of points is equal to or larger than

$$N_y = \lceil (t+1)/2 \rceil$$

where $\lceil (t+1)/2 \rceil$ denotes the minimum integer which is equal to or larger than $(t+1)/2$ [21]. Thus, the number of points for the T+GL+T method is estimated as

$$N_{\mathrm{T+GL+T}} = N_z^2 N_y = (t+1)^2 \lceil (t+1)/2 \rceil = \frac{1}{2} t^3 + O(t^2).$$

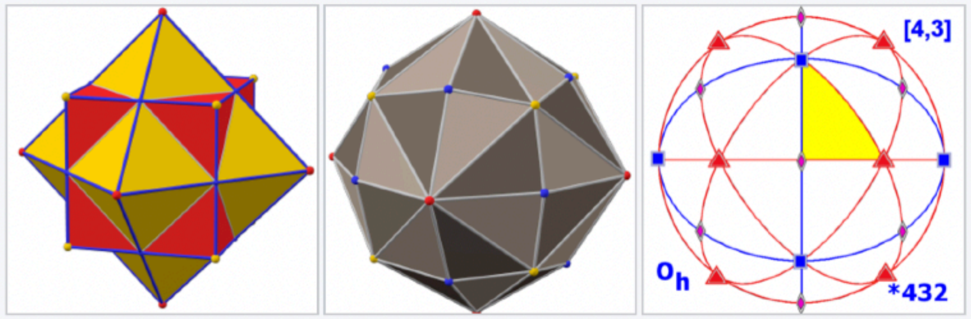
Introduction

- A projection method using linear algebra was proposed for efficient computation.

C. W. Johnson and K. D. O'Mara, Phys. Rev. C 96, 064304 (2017);
C. W. Johnson and C. Jiao, J. Phys. G 46, 015101 (2018)

Introduction

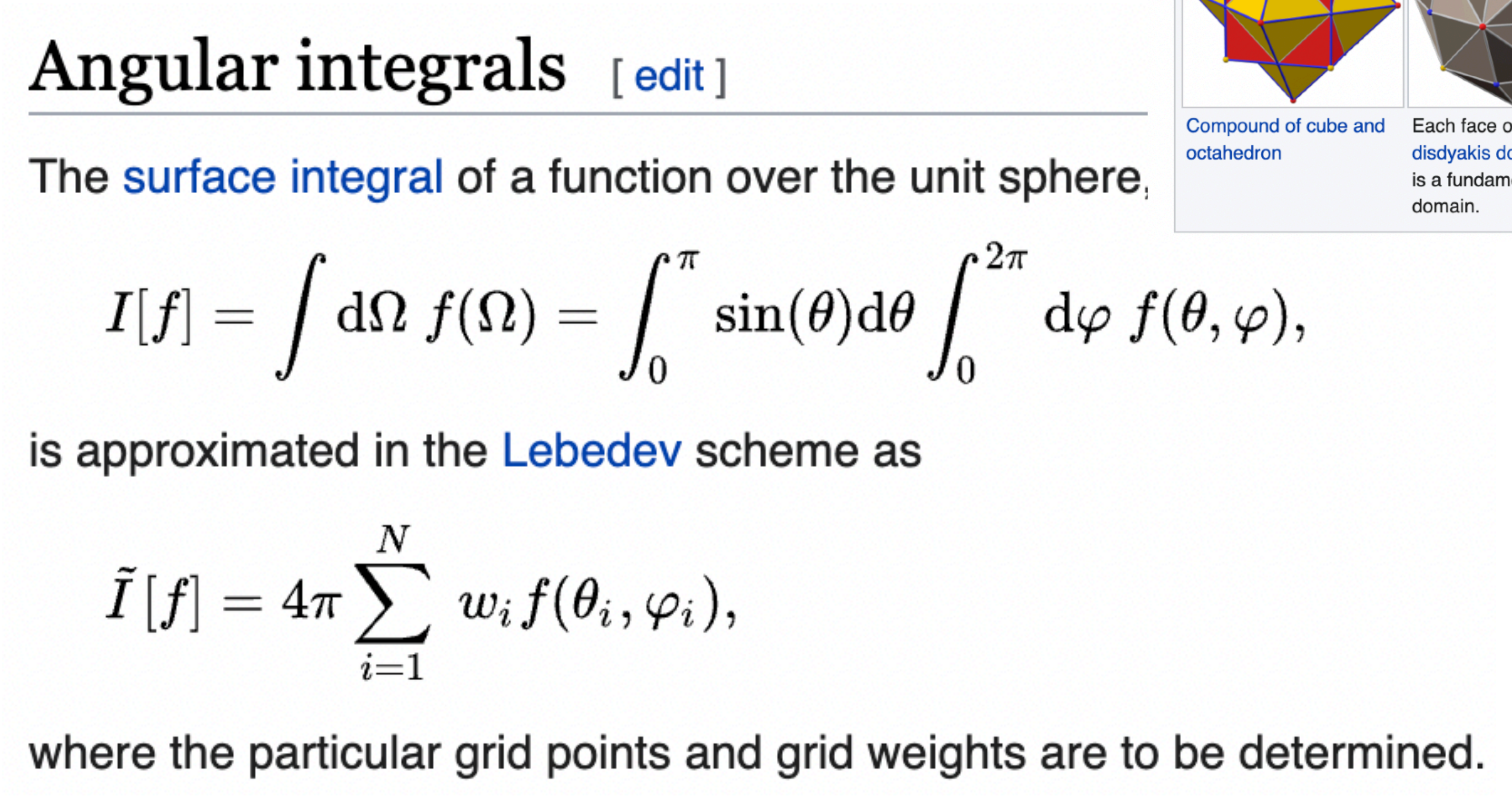
- **The Lebedev quadrature**: The points of the Lebedev quadrature keep the symmetry of octahedral （八面体） rotation and inversion(反演), and are distributed almost equally on the spherical surface.



Compound of cube and octahedron

Each face of the disdyakis dodecahedron is a fundamental domain.

The octahedral group O_h with fundamental domain



Angular integrals [edit]

The surface integral of a function over the unit sphere,

$$I[f] = \int d\Omega\ f(\Omega) = \int_0^{\pi} \sin(\theta) d\theta \int_0^{2\pi} d\varphi\ f(\theta,\varphi),$$

is approximated in the Lebedev scheme as

$$\tilde{I}[f] = 4\pi \sum_{i=1}^{N} w_i f(\theta_i, \varphi_i),$$

where the particular grid points and grid weights are to be determined.

Introduction

SUN YAT-SEN UNIVERSITY

- For any point on the sphere, there are either 5, 7, 11, 23, or 47 equivalent points with respect to the octahedral group, all are included in the grid.

- All points equivalent under the rotational and inversion group share the same weights.

- The smallest such set of points is constructed from all six permutations of (±1, 0, 0) (collectively denoted as a1), leading to an integration scheme

$$\tilde{I}_6[f] = A_1 \sum_{i=1}^{6} f(a_i^1),$$

Distinct classes of grid points

	Typical element	Constraint	Number of points
a^1	$(1,0,0)$		6
a^2	$\frac{1}{\sqrt{2}}(1,1,0)$		12
a^3	$\frac{1}{\sqrt{3}}(1,1,1)$		8

- Grids with two more sets of points, corresponding to the centers and vertices of the octahedron

$$\tilde{I}_{26}[f] = A_1 \sum_{i=1}^{6} f(a_i^1) + A_2 \sum_{i=1}^{12} f(a_i^2) + A_3 \sum_{i=1}^{8} f(a_i^3),$$

where A_1, A_2, and A_3 are the weight functions that still need to be determined.

Introduction

- In general, the Lebedev scheme is

$$\tilde{I}_N[f] = A_1 \sum_{i=1}^{6} f(a_i^1) + A_2 \sum_{i=1}^{12} f(a_i^2) + A_3 \sum_{i=1}^{8} f(a_i^3)$$

$$+ \sum_{k=1}^{N_1} B_k \sum_{i=1}^{24} f(b_i^k) + \sum_{k=1}^{N_2} C_k \sum_{i=1}^{24} f(c_i^k) + \sum_{k=1}^{N_3} D_k \sum_{i=1}^{48} f(d_i^k),$$

where the total number of points, *N*, is

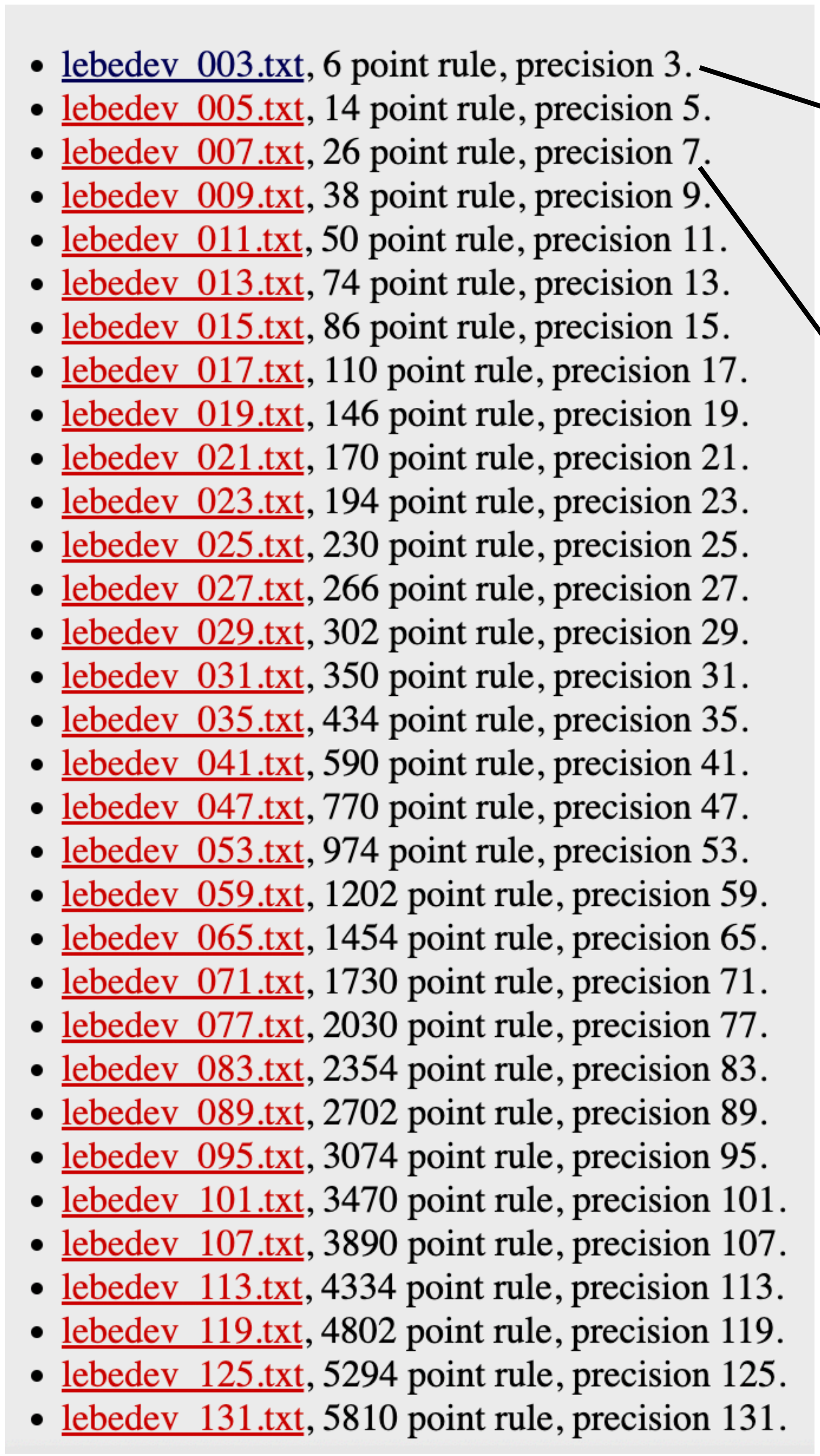
$$N = 26 + 24(N_1 + N_2) + 48N_3.$$

Distinct classes of grid points

	Typical element	Constraint	Number of points
a^1	$(1,0,0)$		6
a^2	$\frac{1}{\sqrt{2}}(1,1,0)$		12
a^3	$\frac{1}{\sqrt{3}}(1,1,1)$		8
b^k	(l_k, l_k, m_k)	$2l_k^2 + m_k^2 = 1$	24
c^k	$(p_k, q_k, 0)$	$p_k^2 + q_k^2 = 1$	24
d^k	(r_k, S_k, W_k)	$r_k^2 + S_k^2 + W_k^2 = 1$	48

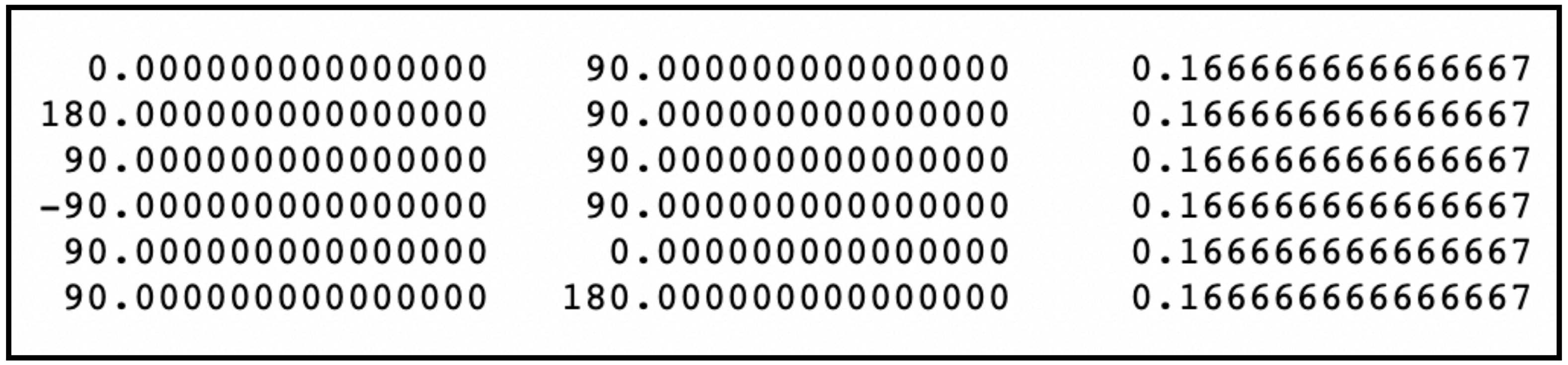
- The points and their weights are determined so that the quadrature is exact for any spherical harmonics with degree up to t.

Introduction

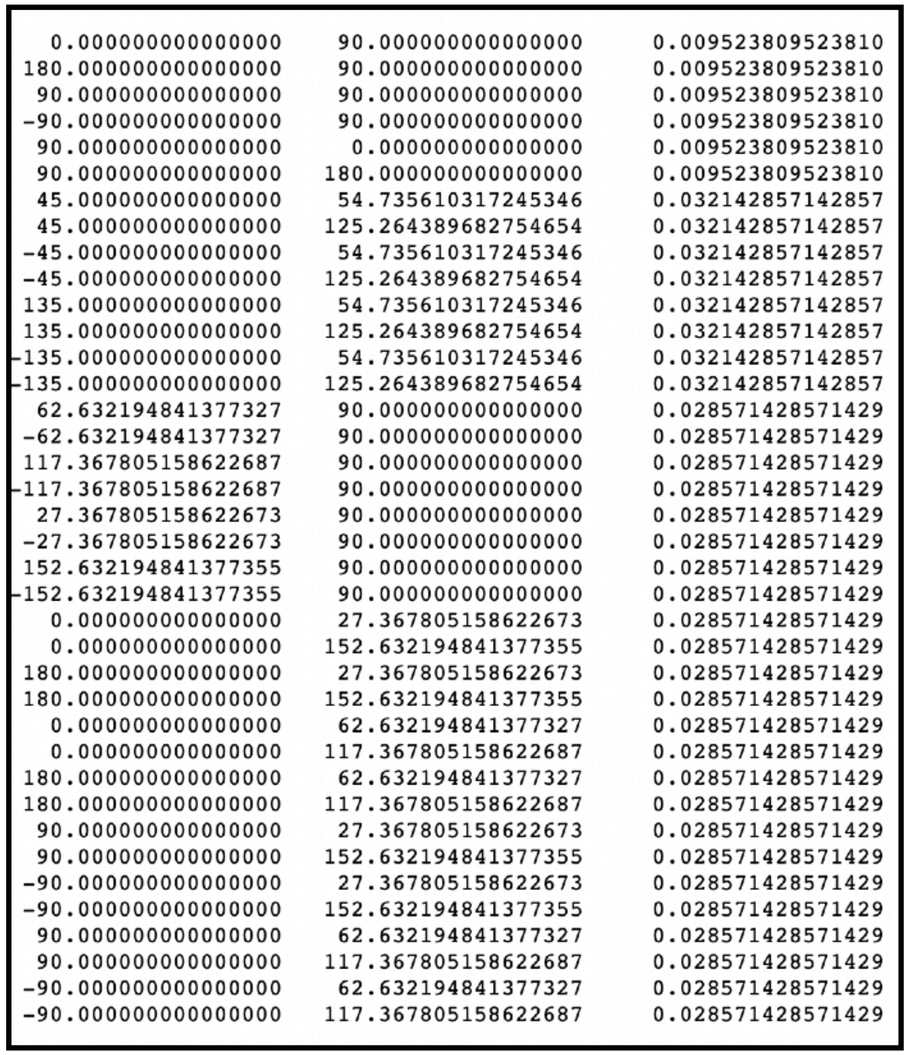


- lebedev_003.txt, 6 point rule, precision 3.
- lebedev_005.txt, 14 point rule, precision 5.
- lebedev_007.txt, 26 point rule, precision 7.
- lebedev_009.txt, 38 point rule, precision 9.
- lebedev_011.txt, 50 point rule, precision 11.
- lebedev_013.txt, 74 point rule, precision 13.
- lebedev_015.txt, 86 point rule, precision 15.
- lebedev_017.txt, 110 point rule, precision 17.
- lebedev_019.txt, 146 point rule, precision 19.
- lebedev_021.txt, 170 point rule, precision 21.
- lebedev_023.txt, 194 point rule, precision 23.
- lebedev_025.txt, 230 point rule, precision 25.
- lebedev_027.txt, 266 point rule, precision 27.
- lebedev_029.txt, 302 point rule, precision 29.
- lebedev_031.txt, 350 point rule, precision 31.
- lebedev_035.txt, 434 point rule, precision 35.
- lebedev_041.txt, 590 point rule, precision 41.
- lebedev_047.txt, 770 point rule, precision 47.
- lebedev_053.txt, 974 point rule, precision 53.
- lebedev_059.txt, 1202 point rule, precision 59.
- lebedev_065.txt, 1454 point rule, precision 65.
- lebedev_071.txt, 1730 point rule, precision 71.
- lebedev_077.txt, 2030 point rule, precision 77.
- lebedev_083.txt, 2354 point rule, precision 83.
- lebedev_089.txt, 2702 point rule, precision 89.
- lebedev_095.txt, 3074 point rule, precision 95.
- lebedev_101.txt, 3470 point rule, precision 101.
- lebedev_107.txt, 3890 point rule, precision 107.
- lebedev_113.txt, 4334 point rule, precision 113.
- lebedev_119.txt, 4802 point rule, precision 119.
- lebedev_125.txt, 5294 point rule, precision 125.
- lebedev_131.txt, 5810 point rule, precision 131.

theta_i   phi_ i   w_i



```
  0.000000000000000    90.000000000000000    0.166666666666667
180.000000000000000    90.000000000000000    0.166666666666667
 90.000000000000000    90.000000000000000    0.166666666666667
-90.000000000000000    90.000000000000000    0.166666666666667
 90.000000000000000     0.000000000000000    0.166666666666667
 90.000000000000000   180.000000000000000    0.166666666666667
```

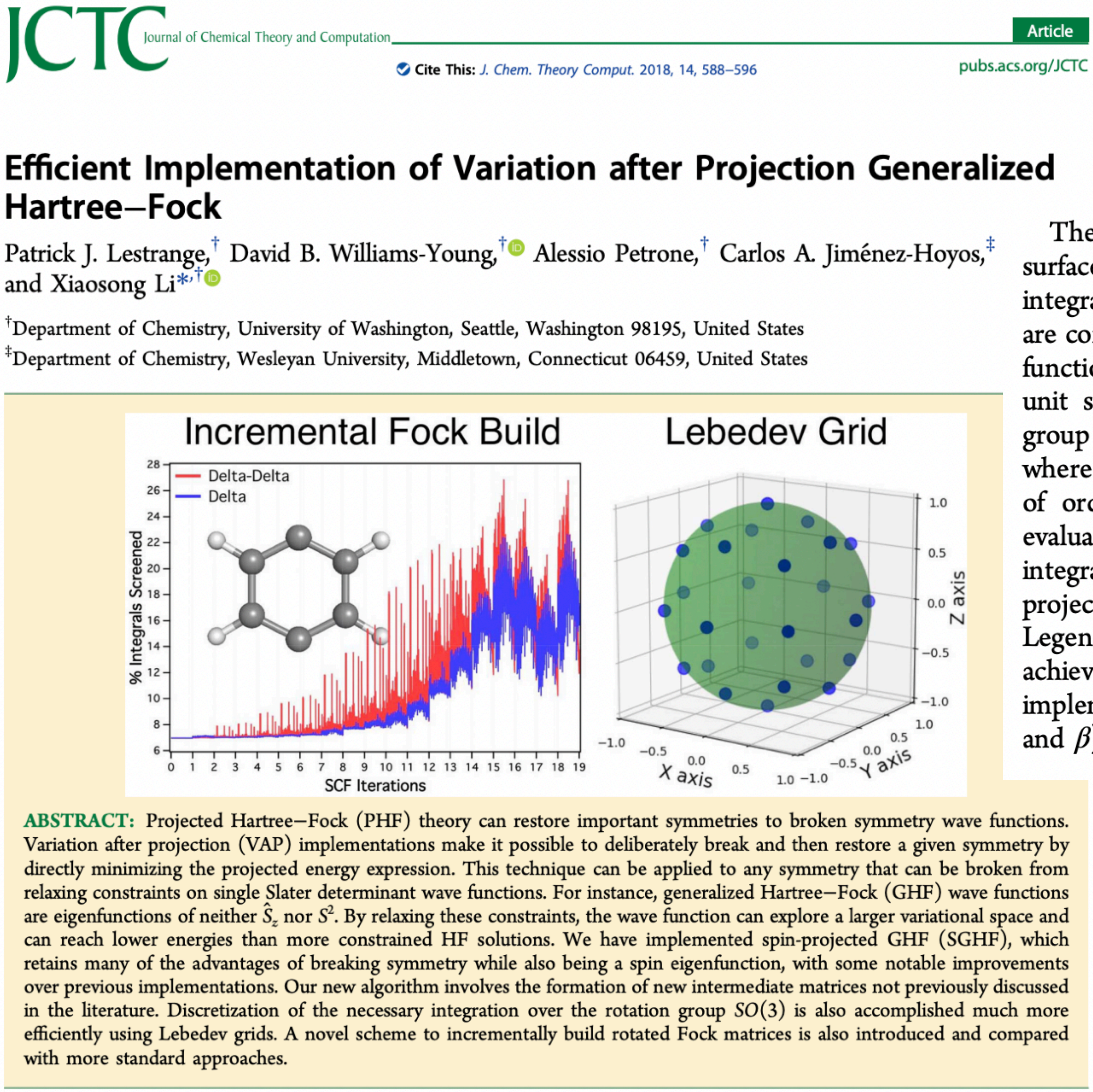


```
   0.000000000000000    90.000000000000000    0.009523809523810
 180.000000000000000    90.000000000000000    0.009523809523810
  90.000000000000000    90.000000000000000    0.009523809523810
 -90.000000000000000    90.000000000000000    0.009523809523810
  90.000000000000000     0.000000000000000    0.009523809523810
  90.000000000000000   180.000000000000000    0.009523809523810
  45.000000000000000    54.735610317245346    0.032142857142857
  45.000000000000000   125.264389682754654    0.032142857142857
 -45.000000000000000    54.735610317245346    0.032142857142857
 -45.000000000000000   125.264389682754654    0.032142857142857
 135.000000000000000    54.735610317245346    0.032142857142857
 135.000000000000000   125.264389682754654    0.032142857142857
-135.000000000000000    54.735610317245346    0.032142857142857
-135.000000000000000   125.264389682754654    0.032142857142857
  62.632194841377327    90.000000000000000    0.028571428571429
 -62.632194841377327    90.000000000000000    0.028571428571429
 117.367805158622687    90.000000000000000    0.028571428571429
-117.367805158622687    90.000000000000000    0.028571428571429
  27.367805158622673    90.000000000000000    0.028571428571429
 -27.367805158622673    90.000000000000000    0.028571428571429
 152.632194841377355    90.000000000000000    0.028571428571429
-152.632194841377355    90.000000000000000    0.028571428571429
   0.000000000000000    27.367805158622673    0.028571428571429
   0.000000000000000   152.632194841377355    0.028571428571429
 180.000000000000000    27.367805158622673    0.028571428571429
 180.000000000000000   152.632194841377355    0.028571428571429
   0.000000000000000    62.632194841377327    0.028571428571429
   0.000000000000000   117.367805158622687    0.028571428571429
 180.000000000000000    62.632194841377327    0.028571428571429
 180.000000000000000   117.367805158622687    0.028571428571429
  90.000000000000000    27.367805158622673    0.028571428571429
  90.000000000000000   152.632194841377355    0.028571428571429
 -90.000000000000000    27.367805158622673    0.028571428571429
 -90.000000000000000   152.632194841377355    0.028571428571429
  90.000000000000000    62.632194841377327    0.028571428571429
  90.000000000000000   117.367805158622687    0.028571428571429
 -90.000000000000000    62.632194841377327    0.028571428571429
 -90.000000000000000   117.367805158622687    0.028571428571429
```

https://people.sc.fsu.edu/~jburkardt/datasets/sphere_lebedev_rule/sphere_lebedev_rule.html

The model

SUN YAT-SEN UNIVERSITY



JCTC Journal of Chemical Theory and Computation

Article

Cite This: J. Chem. Theory Comput. 2018, 14, 588–596

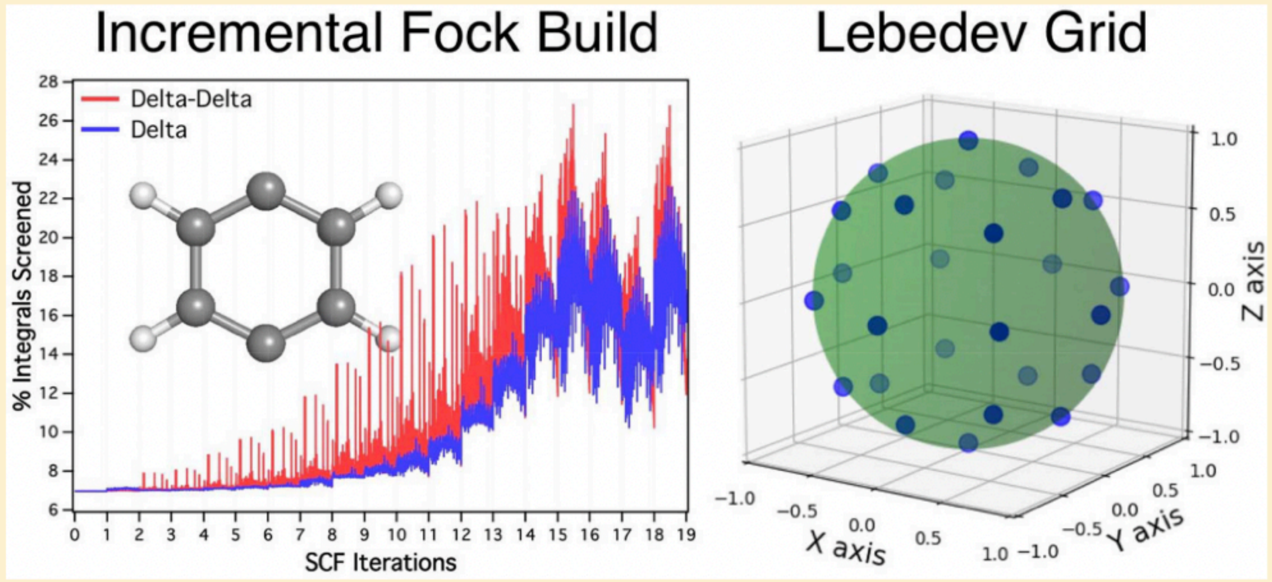
pubs.acs.org/JCTC

Efficient Implementation of Variation after Projection Generalized Hartree–Fock

Patrick J. Lestrange,[†] David B. Williams-Young,[†] Alessio Petrone,[†] Carlos A. Jiménez-Hoyos,[‡] and Xiaosong Li*,[†]

[†]Department of Chemistry, University of Washington, Seattle, Washington 98195, United States

[‡]Department of Chemistry, Wesleyan University, Middletown, Connecticut 06459, United States



ABSTRACT: Projected Hartree–Fock (PHF) theory can restore important symmetries to broken symmetry wave functions. Variation after projection (VAP) implementations make it possible to deliberately break and then restore a given symmetry by directly minimizing the projected energy expression. This technique can be applied to any symmetry that can be broken from relaxing constraints on single Slater determinant wave functions. For instance, generalized Hartree–Fock (GHF) wave functions are eigenfunctions of neither $\hat{S}_z$ nor S^2. By relaxing these constraints, the wave function can explore a larger variational space and can reach lower energies than more constrained HF solutions. We have implemented spin-projected GHF (SGHF), which retains many of the advantages of breaking symmetry while also being a spin eigenfunction, with some notable improvements over previous implementations. Our new algorithm involves the formation of new intermediate matrices not previously discussed in the literature. Discretization of the necessary integration over the rotation group $SO(3)$ is also accomplished much more efficiently using Lebedev grids. A novel scheme to incrementally build rotated Fock matrices is also introduced and compared with more standard approaches.

The integration over $SO(3)$ can be broken down to two surface integrals over a 2-sphere S^2 and one over S^1.[26] Lebedev integration grids discretize the surface integral of a 2-sphere and are commonly used when evaluating DFT exchange correlation functionals.[27,28] All integration points lie on the surface of a unit sphere and are invariant under the octahedral rotation group with inversion. They are classified into different orders, where the order n grid integrates exactly all spherical harmonics of order n or less. Lebedev grids are efficient schemes to evaluate the surface integral of a unit sphere and can be used to integrate over the α (or γ) and β rotation angles for spin-projected GHF. Compared to the mixed Trapezoid and Gauss-Legendre grid, far fewer integration points are required to achieve the same accuracy in spin symmetry restoration. Our implementation uses a Lebedev grid for integration over S^2 (α and β) and a Trapezoid grid for integration over S^1 (γ).

https://pubs.acs.org/doi/abs/10.1021/acs.jctc.7b00832

Introduction

SUN YAT-SEN UNIVERSITY

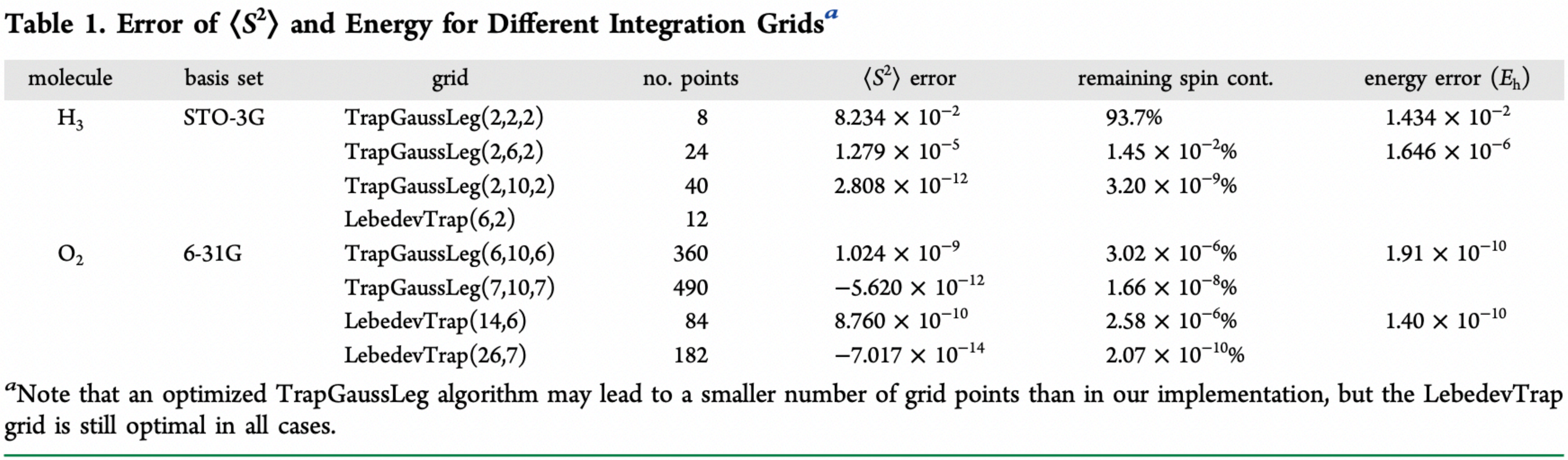


Table 1. Error of $\langle S^2 \rangle$ and Energy for Different Integration Grids[a]

molecule	basis set	grid	no. points	$\langle S^2 \rangle$ error	remaining spin cont.	energy error (E_h)
H_3	STO-3G	TrapGaussLeg(2,2,2)	8	8.234×10^{-2}	93.7%	1.434×10^{-2}
		TrapGaussLeg(2,6,2)	24	1.279×10^{-5}	1.45×10^{-2}%	1.646×10^{-6}
		TrapGaussLeg(2,10,2)	40	2.808×10^{-12}	3.20×10^{-9}%	
		LebedevTrap(6,2)	12			
O_2	6-31G	TrapGaussLeg(6,10,6)	360	1.024×10^{-9}	3.02×10^{-6}%	1.91×10^{-10}
		TrapGaussLeg(7,10,7)	490	-5.620×10^{-12}	1.66×10^{-8}%	
		LebedevTrap(14,6)	84	8.760×10^{-10}	2.58×10^{-6}%	1.40×10^{-10}
		LebedevTrap(26,7)	182	-7.017×10^{-14}	2.07×10^{-10}%	

[a]Note that an optimized TrapGaussLeg algorithm may lead to a smaller number of grid points than in our implementation, but the LebedevTrap grid is still optimal in all cases.

https://pubs.acs.org/doi/abs/10.1021/acs.jctc.7b00832

The model

- Number of points of the Lebedev quadrature for spherical surface against the degree t

$$N_{\text{Lebedev}} \simeq \frac{1}{3}(t+1)^2 + 2$$

The sampling points and weights for up to 131st-degree polynomial are available.

Lebedev, V. I.; D. N. Laikov (1999). "A quadrature formula for the sphere of the 131st algebraic order of accuracy". Doklady Mathematics. 59 (3): 477–481.

- Spherical design (SD) on S2: the sampling points are equally distributed on the sphere surface S2 as far as possible

$$N_{\text{SD2}} = \frac{1}{2}t^2 + \frac{1}{2}t + O(1)$$

competing performance with the Lebedev method for a certain type of integrand functions

C. H. L. Beentjes, Quadrature on a spherical surface, Technical Report, Oxford University (2015).

Numerical test

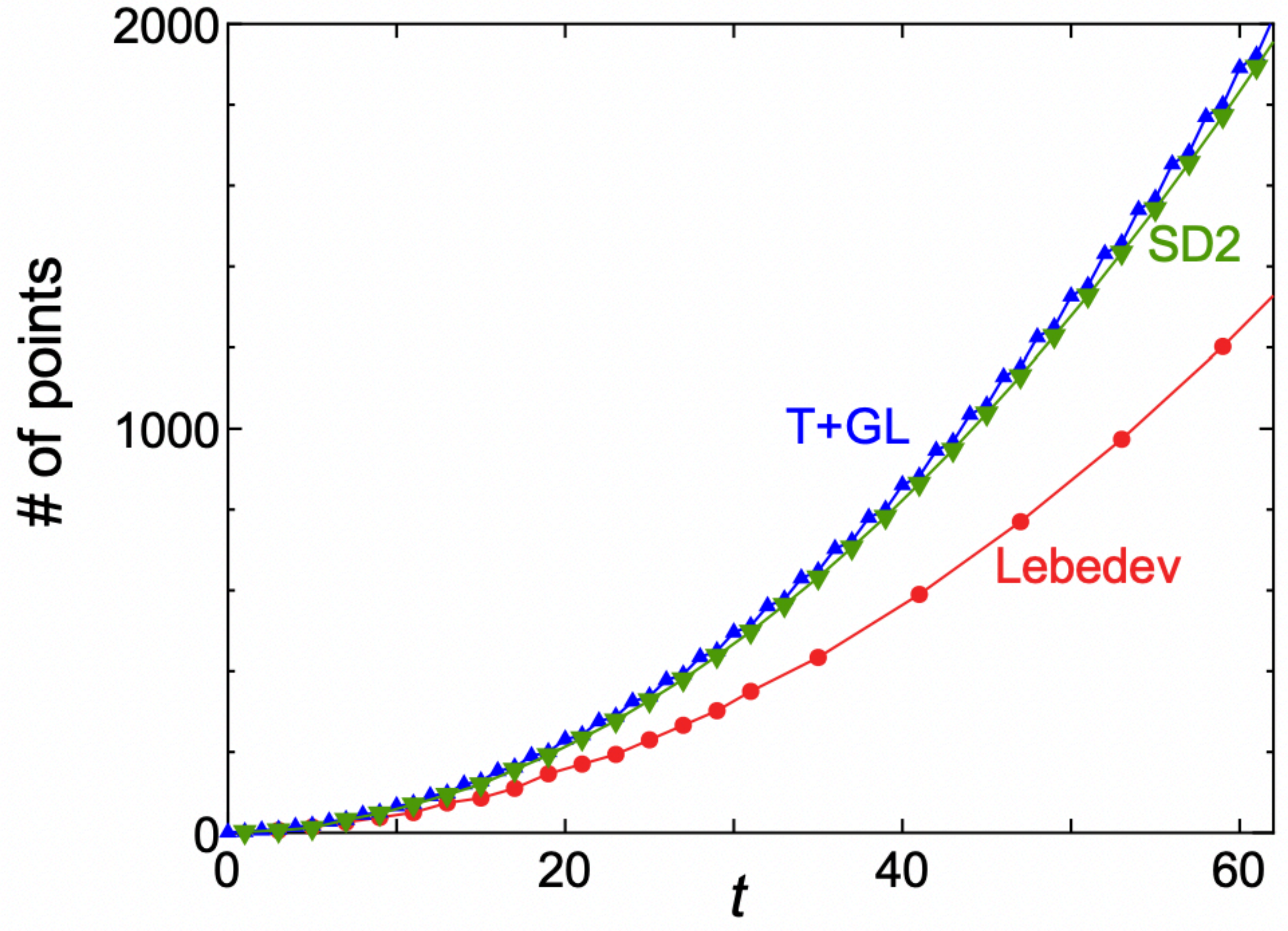


FIG. 1: Number of points for the Lebedev quadrature (red circles), trapezoidal+Gauss-Legendre (T+GL) quadrature (blue triangles), and the spherical design (SD2, green inverse triangles) to calculate the integral of a polynomial of at most t degree exactly for the unit sphere surface $\mathbb{S}^2$ (α and β). For the T+GL method, $(t+1)\lceil (t+1)/2 \rceil$ is shown.

Numerical test

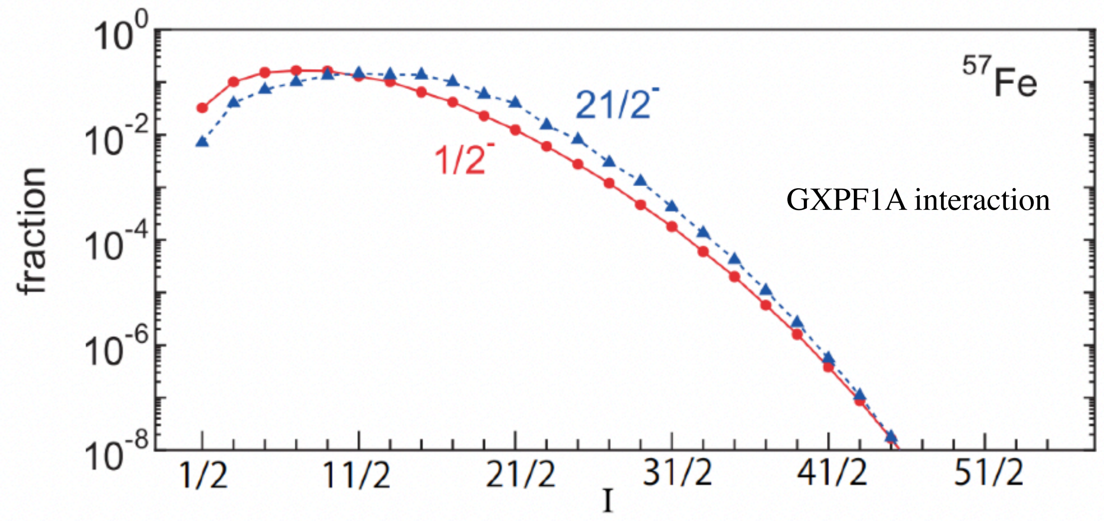


FIG. 3: I-projected components, f_I, of the JHF wave functions of ^{57}Fe. The wave functions are given by the JHF calculations with $J^\pi = 1/2^-$ and $21/2^-$.

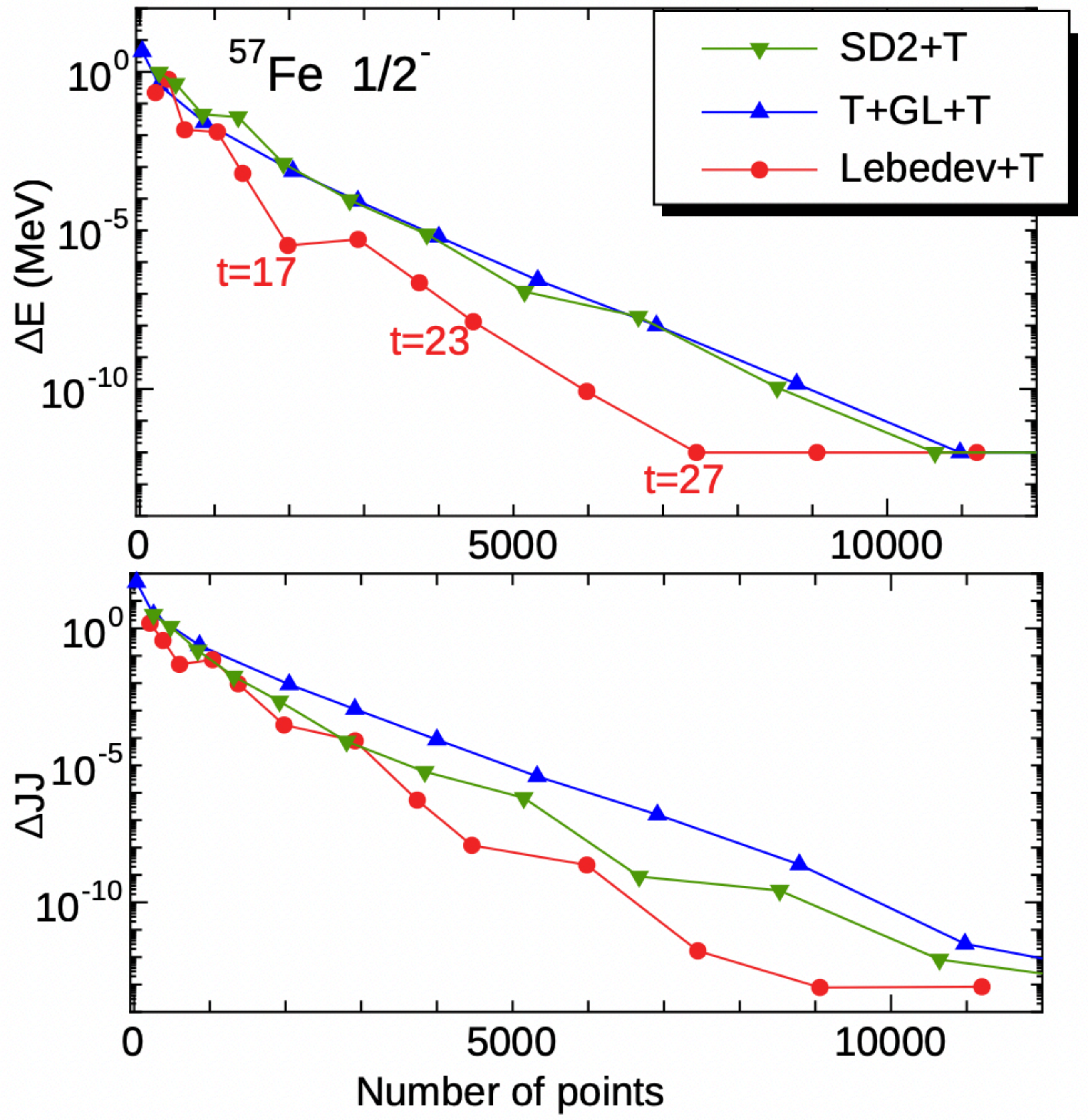


FIG. 4: Error of the Hamiltonian and J^2 expectation values, $\Delta E = |E - E_{\text{exact}}|$ and $\Delta J^2 = |\langle \hat{J}^2 \rangle - J(J+1)|$, of the $1/2^-$ state of ^{57}Fe against the number of points. The wave function is given by the JHF calculation.

Numerical test

SUN YAT-SEN UNIVERSITY

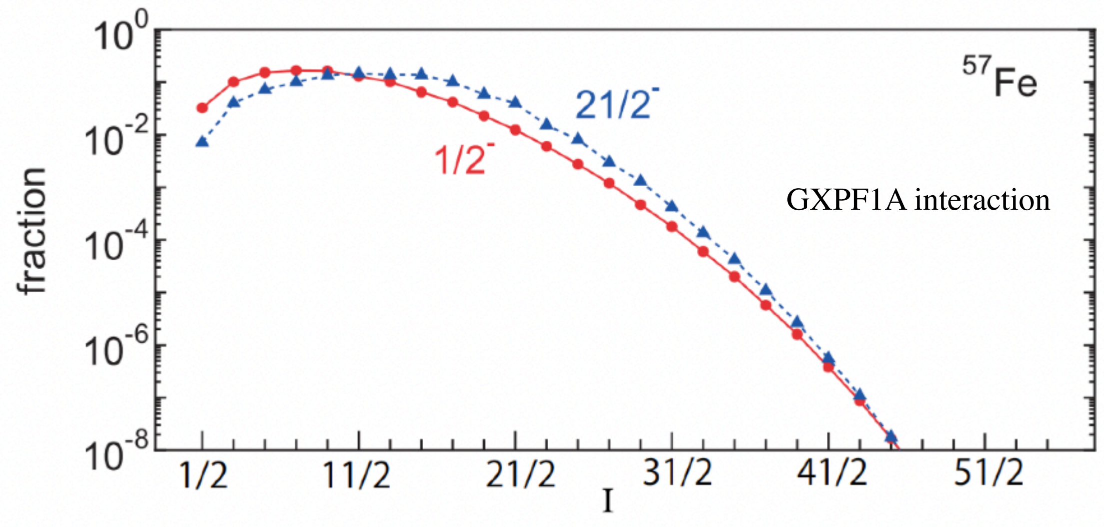


FIG. 3: I-projected components, f_I, of the JHF wave functions of ^{57}Fe. The wave functions are given by the JHF calculations with $J^\pi = 1/2^-$ and $21/2^-$.

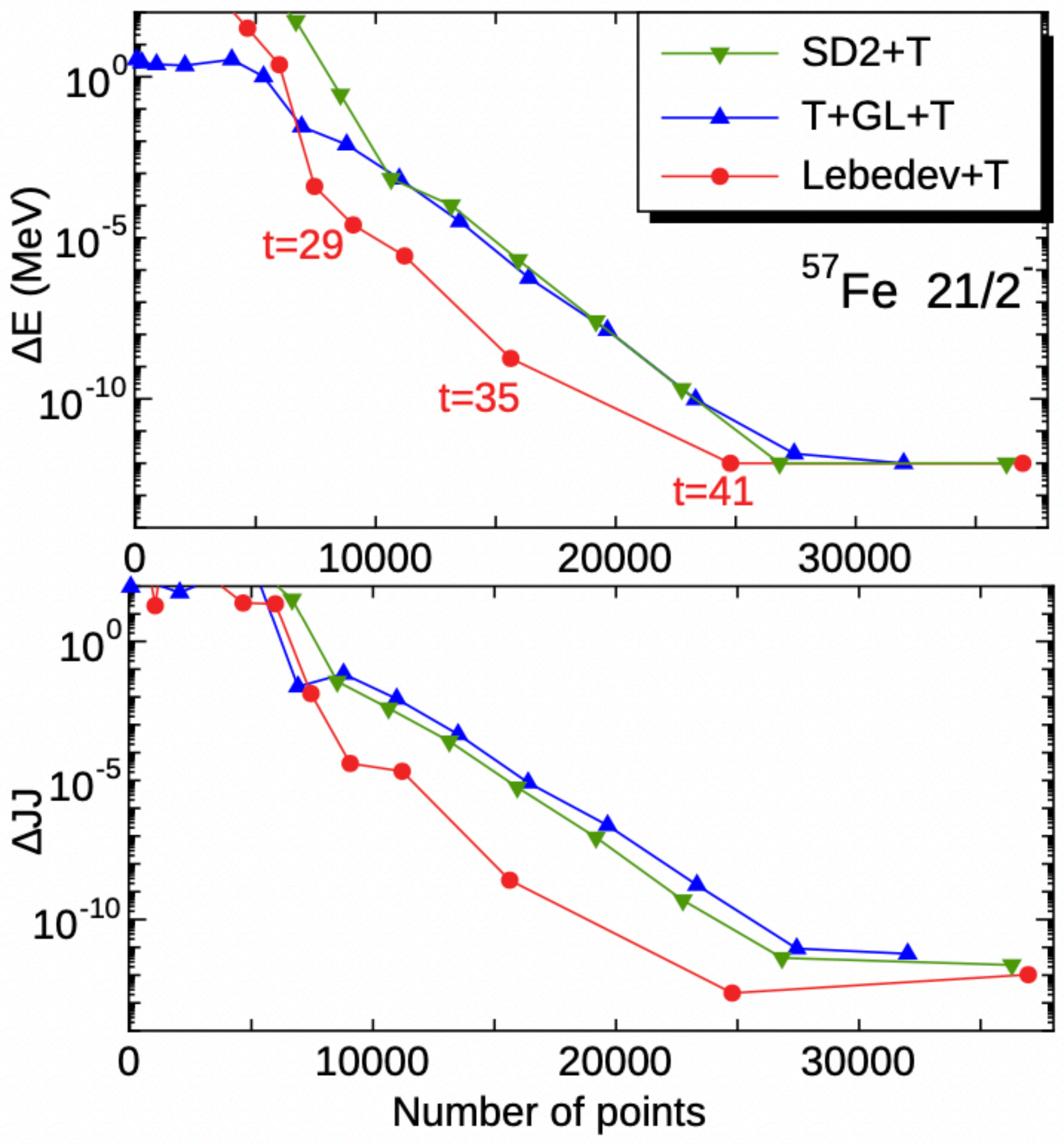
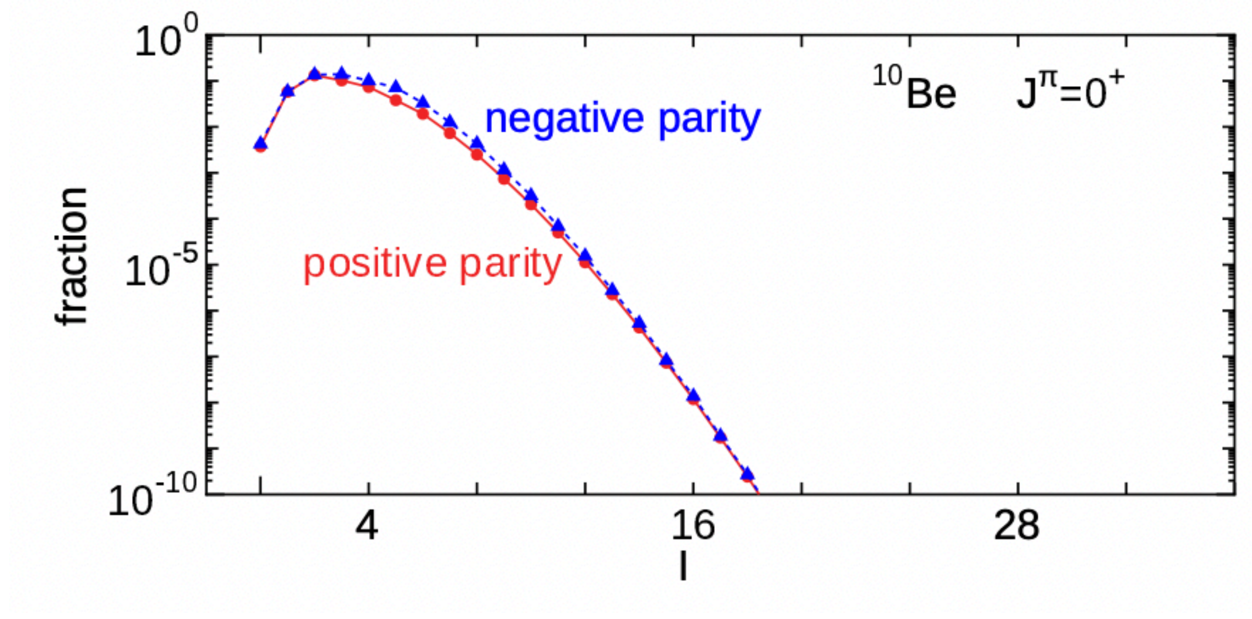
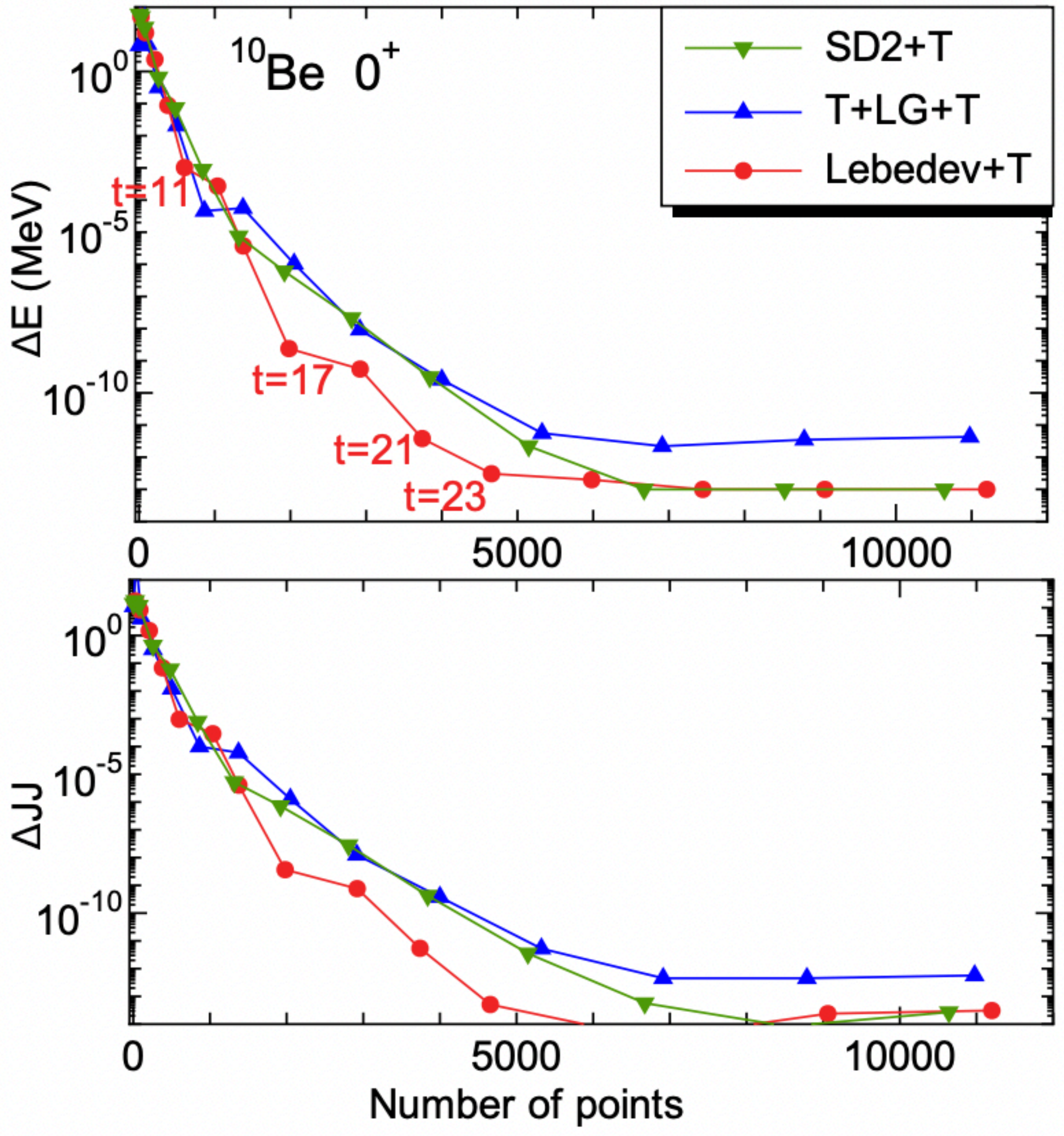
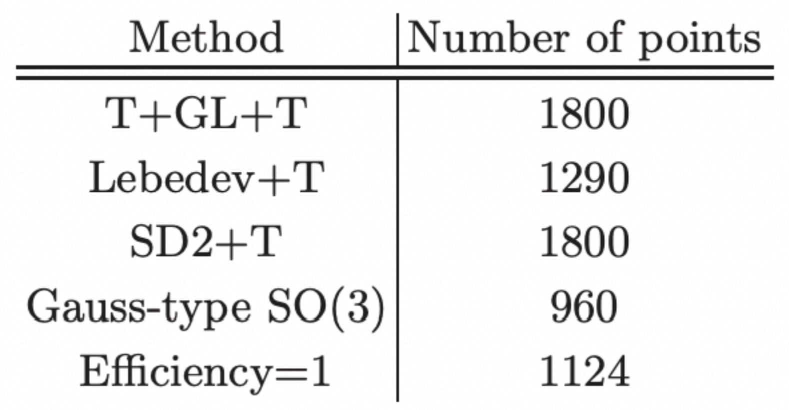


FIG. 5: Errors of the Hamiltonian and J^2 expectation values of the $21/2^-$ state of ^{57}Fe against the number of the quadrature points. See caption of Fig. 4 for details.

Numerical test



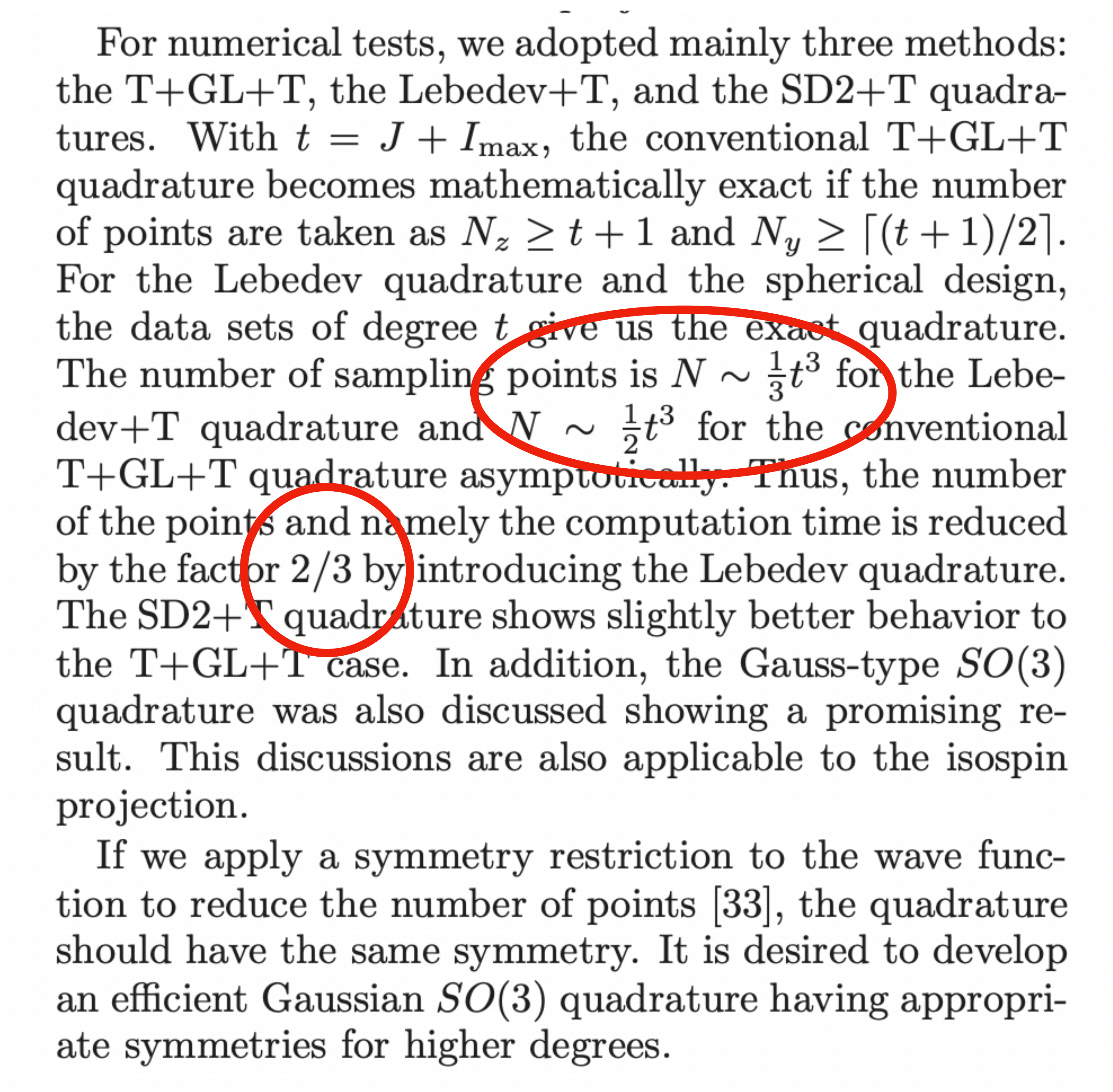




Method	Number of points
T+GL+T	1800
Lebedev+T	1290
SD2+T	1800
Gauss-type SO(3)	960
Efficiency=1	1124

TABLE I: Number of points of quadratures for a degree of exactness $t = 14$.

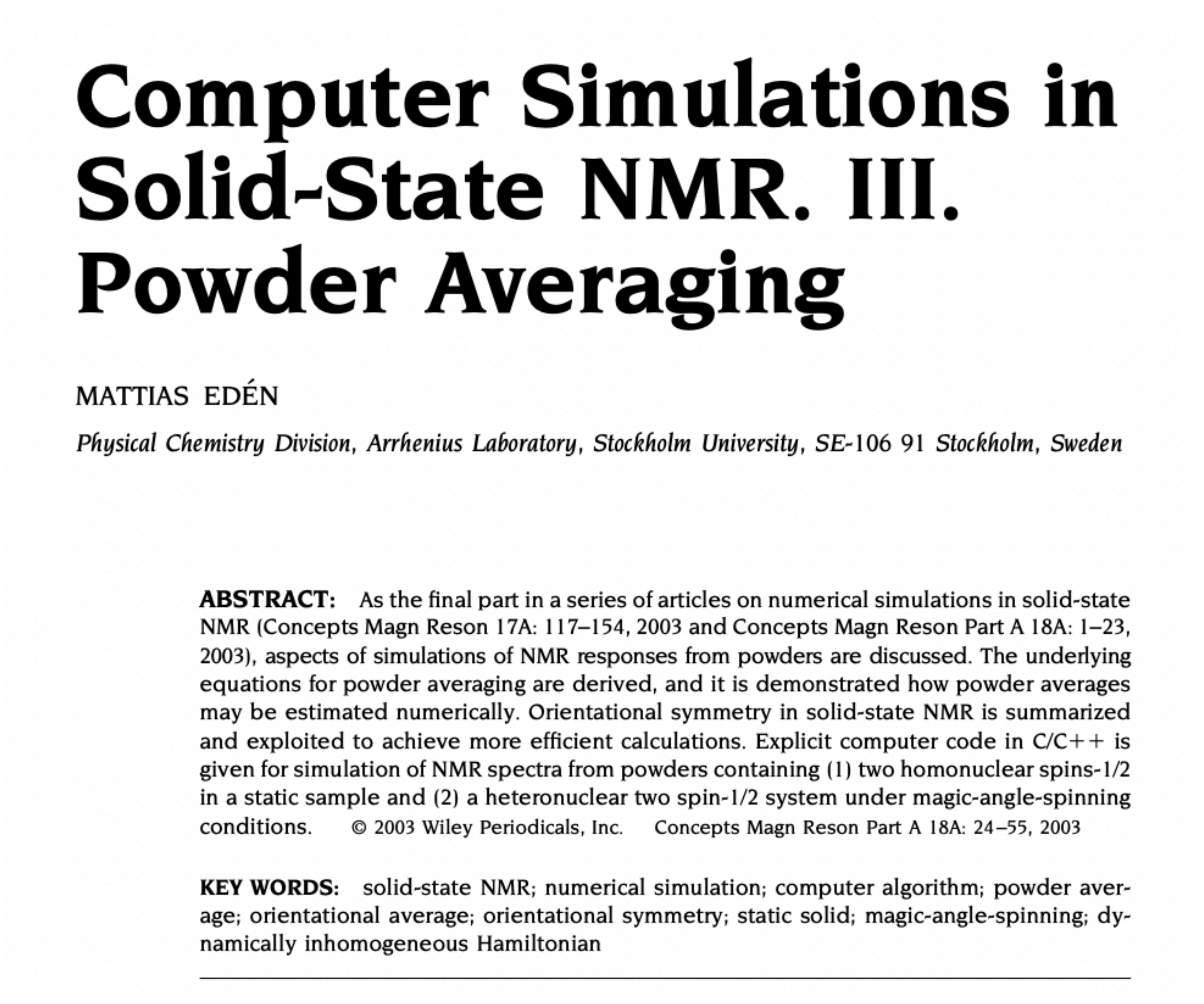
Summary



For numerical tests, we adopted mainly three methods: the T+GL+T, the Lebedev+T, and the SD2+T quadratures. With $t = J + I_{\max}$, the conventional T+GL+T quadrature becomes mathematically exact if the number of points are taken as $N_z \geq t+1$ and $N_y \geq \lceil (t+1)/2 \rceil$. For the Lebedev quadrature and the spherical design, the data sets of degree t give us the exact quadrature. The number of sampling points is $N \sim \frac{1}{3}t^3$ for the Lebedev+T quadrature and $N \sim \frac{1}{2}t^3$ for the conventional T+GL+T quadrature asymptotically. Thus, the number of the points and namely the computation time is reduced by the factor $2/3$ by introducing the Lebedev quadrature. The SD2+T quadrature shows slightly better behavior to the T+GL+T case. In addition, the Gauss-type $SO(3)$ quadrature was also discussed showing a promising result. This discussions are also applicable to the isospin projection.

If we apply a symmetry restriction to the wave function to reduce the number of points [33], the quadrature should have the same symmetry. It is desired to develop an efficient Gaussian $SO(3)$ quadrature having appropriate symmetries for higher degrees.

The ZCW method



Computer Simulations in Solid-State NMR. III. Powder Averaging

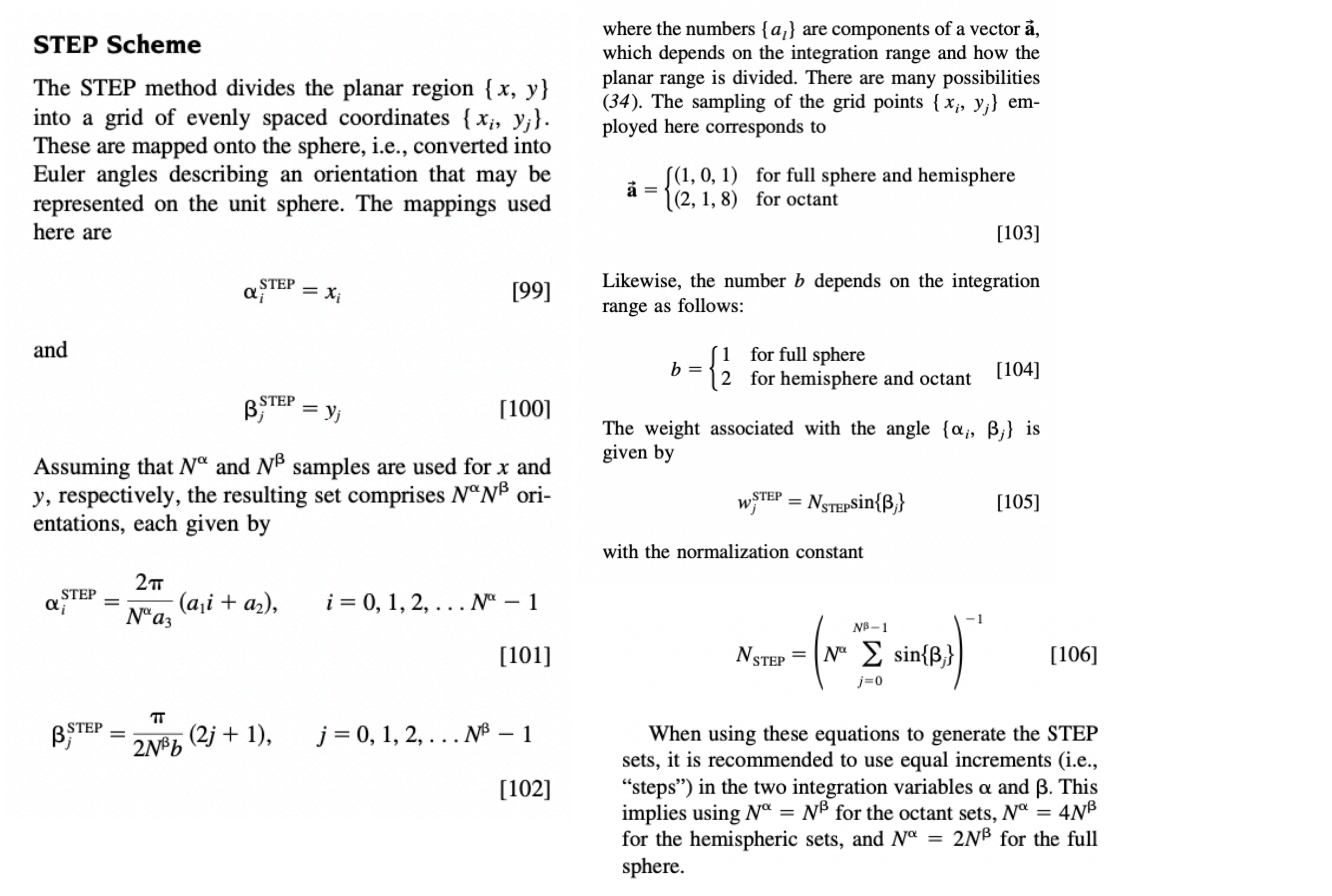
MATTIAS EDÉN

Physical Chemistry Division, Arrhenius Laboratory, Stockholm University, SE-106 91 Stockholm, Sweden

ABSTRACT: As the final part in a series of articles on numerical simulations in solid-state NMR (Concepts Magn Reson 17A: 117–154, 2003 and Concepts Magn Reson Part A 18A: 1–23, 2003), aspects of simulations of NMR responses from powders are discussed. The underlying equations for powder averaging are derived, and it is demonstrated how powder averages may be estimated numerically. Orientational symmetry in solid-state NMR is summarized and exploited to achieve more efficient calculations. Explicit computer code in C/C++ is given for simulation of NMR spectra from powders containing (1) two homonuclear spins-1/2 in a static sample and (2) a heteronuclear two spin-1/2 system under magic-angle-spinning conditions. © 2003 Wiley Periodicals, Inc. Concepts Magn Reson Part A 18A: 24–55, 2003

KEY WORDS: solid-state NMR; numerical simulation; computer algorithm; powder average; orientational average; orientational symmetry; static solid; magic-angle-spinning; dynamically inhomogeneous Hamiltonian

The ZCW method



STEP Scheme

The STEP method divides the planar region $\{x, y\}$ into a grid of evenly spaced coordinates $\{x_i, y_j\}$. These are mapped onto the sphere, i.e., converted into Euler angles describing an orientation that may be represented on the unit sphere. The mappings used here are

$$\alpha_i^{\mathrm{STEP}} = x_i \qquad [99]$$

and

$$\beta_j^{\mathrm{STEP}} = y_j \qquad [100]$$

Assuming that N^α and N^β samples are used for x and y, respectively, the resulting set comprises $N^\alpha N^\beta$ orientations, each given by

$$\alpha_i^{\mathrm{STEP}} = \frac{2\pi}{N^\alpha a_3}(a_1 i + a_2), \qquad i = 0, 1, 2, \ldots N^\alpha - 1 \qquad [101]$$

$$\beta_j^{\mathrm{STEP}} = \frac{\pi}{2N^\beta b}(2j + 1), \qquad j = 0, 1, 2, \ldots N^\beta - 1 \qquad [102]$$

where the numbers $\{a_l\}$ are components of a vector $\vec{\mathbf{a}}$, which depends on the integration range and how the planar range is divided. There are many possibilities (*34*). The sampling of the grid points $\{x_i, y_j\}$ employed here corresponds to

$$\vec{\mathbf{a}} = \begin{cases} (1, 0, 1) & \text{for full sphere and hemisphere} \\ (2, 1, 8) & \text{for octant} \end{cases} \qquad [103]$$

Likewise, the number b depends on the integration range as follows:

$$b = \begin{cases} 1 & \text{for full sphere} \\ 2 & \text{for hemisphere and octant} \end{cases} \qquad [104]$$

The weight associated with the angle $\{\alpha_i, \beta_j\}$ is given by

$$w_j^{\mathrm{STEP}} = N_{\mathrm{STEP}} \sin\{\beta_j\} \qquad [105]$$

with the normalization constant

$$N_{\mathrm{STEP}} = \left(N^\alpha \sum_{j=0}^{N^\beta - 1} \sin\{\beta_j\} \right)^{-1} \qquad [106]$$

When using these equations to generate the STEP sets, it is recommended to use equal increments (i.e., "steps") in the two integration variables α and β. This implies using $N^\alpha = N^\beta$ for the octant sets, $N^\alpha = 4N^\beta$ for the hemispheric sets, and $N^\alpha = 2N^\beta$ for the full sphere.

The ZCW method

SUN YAT-SEN UNIVERSITY

ZCW Scheme

It is out of the scope of this article to justify the generic equations for the ZCW sets. Detailed explanations may be found in Refs. (*28–30, 32*). The ZCW partitions are generated from numbers F_M of the Fibonacci series (*15*). These are given by the recursion

$$F_M = F_{M-1} + F_{M-2}, \qquad M = 0, 1, 2, \ldots \tag{107}$$

with $F_0 = 8$ and $F_1 = 13$. For a given integer M, the corresponding ZCW set comprises

$$N_M = F_{M+2} \tag{108}$$

samples over the planar range $\{x, y\}$. Next, the mappings

$$\alpha_i^{ZCW} = x_i \tag{109}$$

and

$$\beta_j^{ZCW} = \arccos\{y_j\} \tag{110}$$

are used, giving the Euler angles

$$\alpha_j^{ZCW} = \frac{2\pi}{c_3}\,\mathrm{mod}\{jF_M/N_M, 1\}, \qquad j = 0, 1, 2, \ldots, N_M - 1 \tag{111}$$

$$\beta_j^{ZCW} = \arccos[c_1(c_2\mathrm{mod}\{j/N_M, 1\} - 1)], \qquad j = 0, 1, 2, \ldots, N_M - 1 \tag{112}$$

As for the STEP implementation, the numbers $\{c_l\}$ are components of a vector $\vec{c}$, given by

$$\vec{c} = \begin{cases} (1, 2, 1) & \text{for full sphere} \\ (-1, 1, 1) & \text{for hemisphere} \\ (2, 1, 8) & \text{for octant} \end{cases} \tag{113}$$

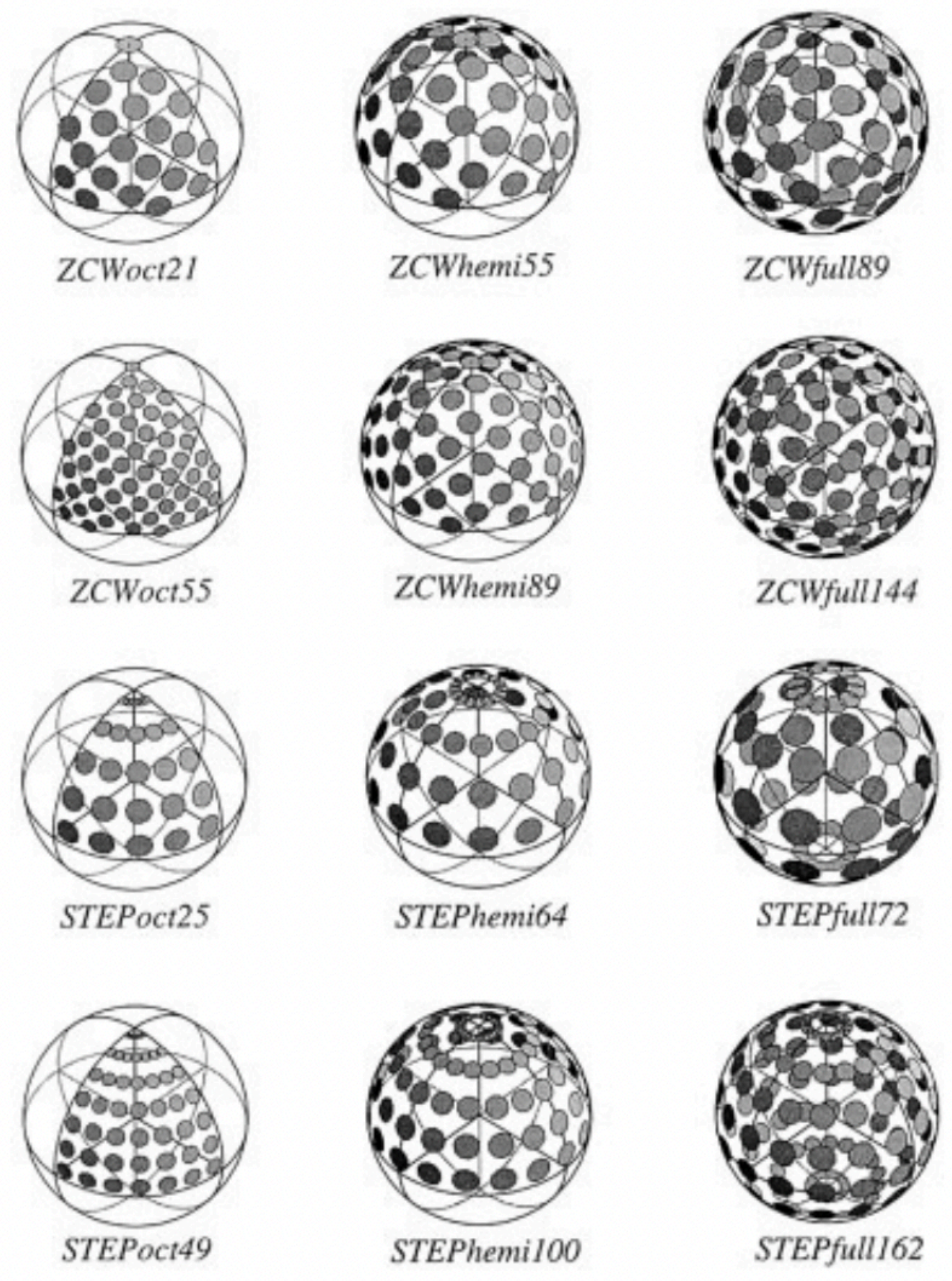
The weights are equal for all angles in a ZCW set; they are calculated as the inverse of the number of orientations:

$$w_j^{ZCW} = N_M^{-1} \tag{114}$$

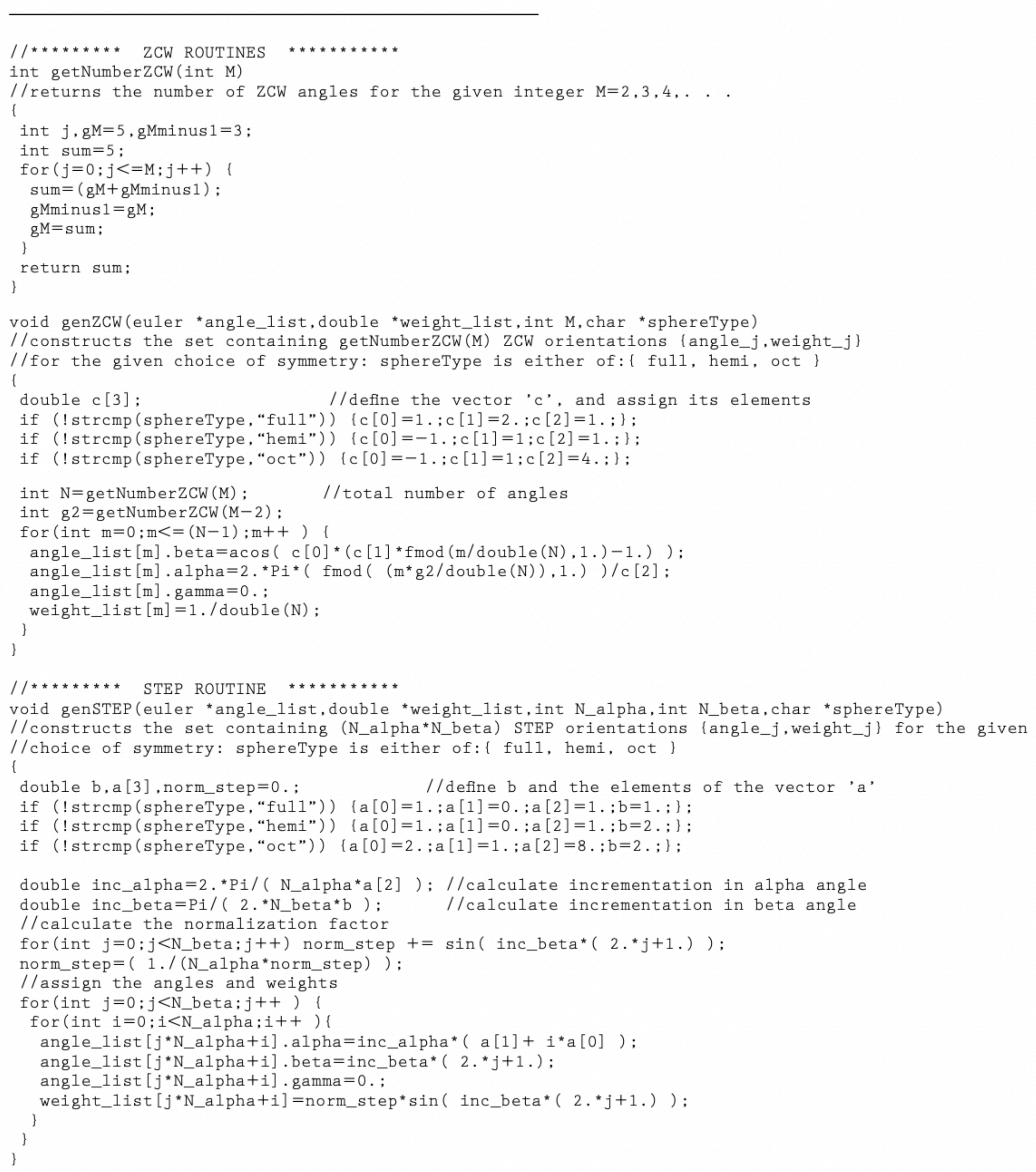
28. Zaremba SK, Good lattice points, discrepancy, and numerical integration. Ann Mat Pura Appl 1966; 4:73: 293–317.
29. Conroy H. Molecular Schrödinger equation. VIII. A new method for the evaluation of multidimensional integrals. J Chem Phys 1967; 47:5307–5318.
30. Cheng VB, Suzukawa HH, Wolfsberg M. Investigations of a nonrandom numerical method for multidimensional integration. J Chem Phys 1973; 59:3992–3999.

32. Koons JM, Hughes E, Cho HM, Ellis PD. Extracting multitensor solid-state NMR parameters from lineshapes. J Magn Reson A 1995; 114:12–23.

The ZCW method



The ZCW method



```
//*********  ZCW ROUTINES  ***********
int getNumberZCW(int M)
//returns the number of ZCW angles for the given integer M=2,3,4,. . .
{
 int j,gM=5,gMminus1=3;
 int sum=5;
 for(j=0;j<=M;j++) {
  sum=(gM+gMminus1);
  gMminus1=gM;
  gM=sum;
 }
 return sum;
}

void genZCW(euler *angle_list,double *weight_list,int M,char *sphereType)
//constructs the set containing getNumberZCW(M) ZCW orientations {angle_j,weight_j}
//for the given choice of symmetry: sphereType is either of:{ full, hemi, oct }
{
 double c[3];                  //define the vector 'c', and assign its elements
 if (!strcmp(sphereType,“full”)) {c[0]=1.;c[1]=2.;c[2]=1.;};
 if (!strcmp(sphereType,“hemi”)) {c[0]=-1.;c[1]=1;c[2]=1.;};
 if (!strcmp(sphereType,“oct”)) {c[0]=-1.;c[1]=1;c[2]=4.;};

 int N=getNumberZCW(M);       //total number of angles
 int g2=getNumberZCW(M-2);
 for(int m=0;m<=(N-1);m++ ) {
  angle_list[m].beta=acos( c[0]*(c[1]*fmod(m/double(N),1.)-1.) );
  angle_list[m].alpha=2.*Pi*( fmod( (m*g2/double(N)),1.) )/c[2];
  angle_list[m].gamma=0.;
  weight_list[m]=1./double(N);
 }
}

//*********  STEP ROUTINE  ***********
void genSTEP(euler *angle_list,double *weight_list,int N_alpha,int N_beta,char *sphereType)
//constructs the set containing (N_alpha*N_beta) STEP orientations {angle_j,weight_j} for the given
//choice of symmetry: sphereType is either of:{ full, hemi, oct }
{
 double b,a[3],norm_step=0.;            //define b and the elements of the vector 'a'
 if (!strcmp(sphereType,“full”)) {a[0]=1.;a[1]=0.;a[2]=1.;b=1.;};
 if (!strcmp(sphereType,“hemi”)) {a[0]=1.;a[1]=0.;a[2]=1.;b=2.;};
 if (!strcmp(sphereType,“oct”)) {a[0]=2.;a[1]=1.;a[2]=8.;b=2.;};

 double inc_alpha=2.*Pi/( N_alpha*a[2] ); //calculate incrementation in alpha angle
 double inc_beta=Pi/( 2.*N_beta*b );      //calculate incrementation in beta angle
 //calculate the normalization factor
 for(int j=0;j<N_beta;j++) norm_step += sin( inc_beta*( 2.*j+1.) );
 norm_step=( 1./(N_alpha*norm_step) );
 //assign the angles and weights
 for(int j=0;j<N_beta;j++ ) {
  for(int i=0;i<N_alpha;i++ ){
   angle_list[j*N_alpha+i].alpha=inc_alpha*( a[1]+ i*a[0] );
   angle_list[j*N_alpha+i].beta=inc_beta*( 2.*j+1.);
   angle_list[j*N_alpha+i].gamma=0.;
   weight_list[j*N_alpha+i]=norm_step*sin( inc_beta*( 2.*j+1.) );
  }
 }
}
```