Numerical Quadrature over the Surface of a Sphere

Bengt Fornberg

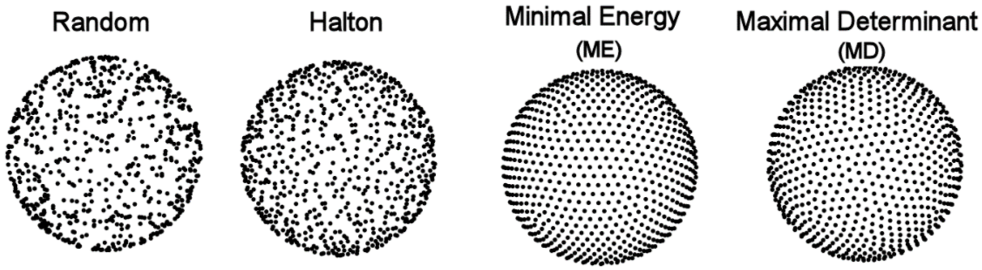
University of Colorado at Boulder
Department of Applied Mathematics

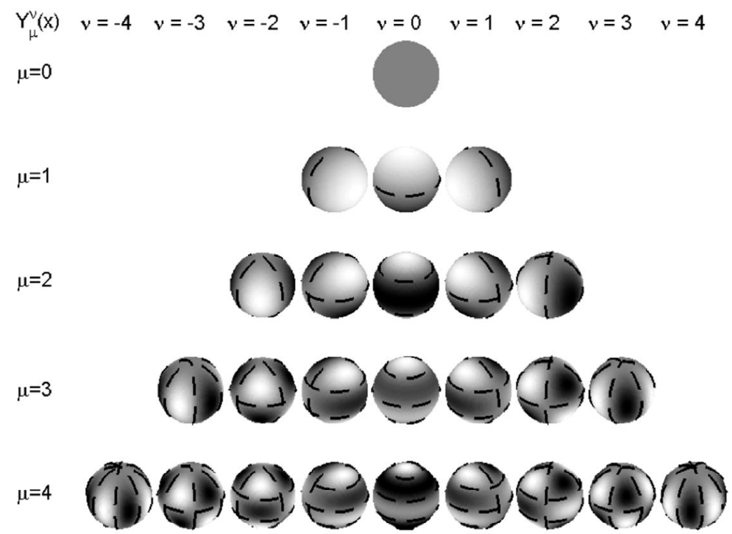
in collaboration with:

Jonah Reeger

Air Force Institute of Technology,
Wright Patterson Air Force Base
Department of Mathematics and Statistics

Spherical Harmonics (SPH)-based algorithms for quasi-uniform node sets:





Weights are readily obtained via SPH interpolation:

$$\begin{bmatrix} Y_0^0(\underline{x}_1) & \cdots & Y_0^0(\underline{x}_N) \\ \vdots & A & \vdots \\ \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_N \end{bmatrix} = \begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_N \end{bmatrix} = \begin{bmatrix} & A^{-1} & \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix}$$

λ_1 = surface integral => top row of A^{-1} contains the quadrature weights

Key issue: The *A*-matrix often becomes extremely ill conditioned. Two remedies:

1. **Interpolation and cubature on the sphere**

   (R.S. Womersley and I.H. Sloan), http://web.maths.unsw.edu.au/~rsw/Sphere/ (2003,2007)

   Introduces MD nodes: These are designed to optimize the *A*-matrix condition number

2. **On spherical harmonics based numerical quadrature over the surface of a sphere**

   (B. Fornberg and J. Martel), Adv. Comp. Math. 40 (2014), 1169-1184.

   Notes that the ill-conditioning is caused by very few singular values – issue avoided by (in Matlab) replacing 'inv' by 'pinv'. Then ME becomes excellent (even better than MD) and Random & Halton useable

In both cases:
- Spectral accuracy
- Cost: $O(N^3)$ operations and $O(N^2)$ memory for N nodes,
- No opportunity for local node refinement.

Radial Basis Function (RBF)-based algorithms for variable density node sets:

1. Kernel based quadrature on spheres and other homogeneous spaces

(E. Fuselier, T. Hangelbroek, F.J. Narcowich, J.D. Ward and G.B. Wright), Numer. Math. 127 (2014), 57-92.

Main algorithmic steps:

- Fit the data f_i at $\underline{x}_i$ by a linear combination of radial basis functions (RBFs)

$$s(\underline{x}) = \sum \lambda_i \phi(\| \underline{x} - \underline{x}_i \|); \quad \phi(r) = r^{2k+1}$$

- Create local Lagrangian basis functions for the RBF interpolant
- Use these as preconditioners in GMRES iterations for obtaining quadrature weights

Features:

- As implemented, order of accuracy $O(h^4)$,
- Cost $O(N^2)$ operations and $O(N^2)$ memory for N nodes,
- Results published only for quasi-uniform node sets

2. Numerical quadrature over the surface of a sphere

(J.A. Reeger and B. Fornberg), in preparation.

Topic of this presentation

Features:

- Also RBF based, but non-iterative algorithm,
- As implemented, order of accuracy $O(h^7)$,
- Cost $O(N \log N)$ operations and $O(N)$ memory for N nodes,
- Algorithm 'embarrassingly parallel'

Algorithm steps:

1. Given nodes on the sphere, create a spherical Delaunay triangularization

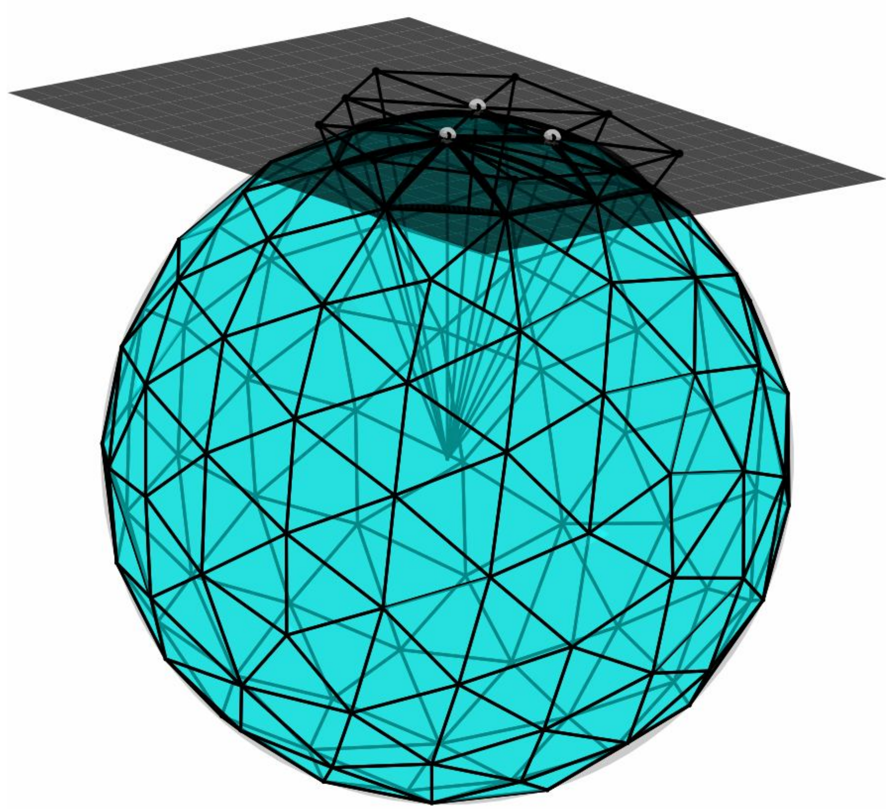
2. For each surface triangle, project it together with some nearby nodes to a tangent plane

3. Find quadrature weights over the local tangent plane node set for the central planar triangle

4. Convert weights from the tangent plane case to corresponding weights on sphere surface

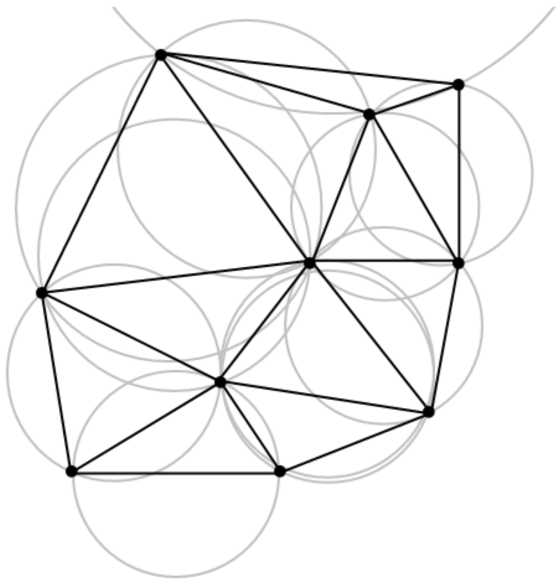
5. Add together the weights for the individual triangles to obtain the full weight set for the sphere

Concept illustration:



Next 4 slides explain these 5 steps in more detail

Step 1: Given nodes on the sphere, create a Delaunay triangularization



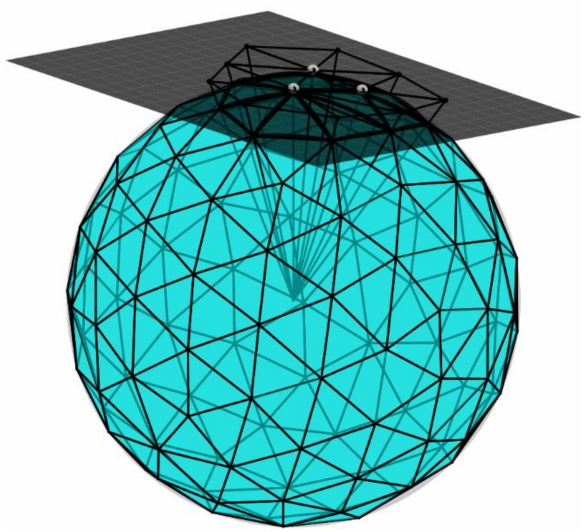
Delaunay triangularization in a 2-D planar case:

- Forms triangles so that no point ever falls inside the circumcircle to any triangle
- Provides guarantee against inside 'skinny' triangles
- Cost: $O(N \log N)$ operations for N nodes.

Generalization to surface of a sphere maintains all key features

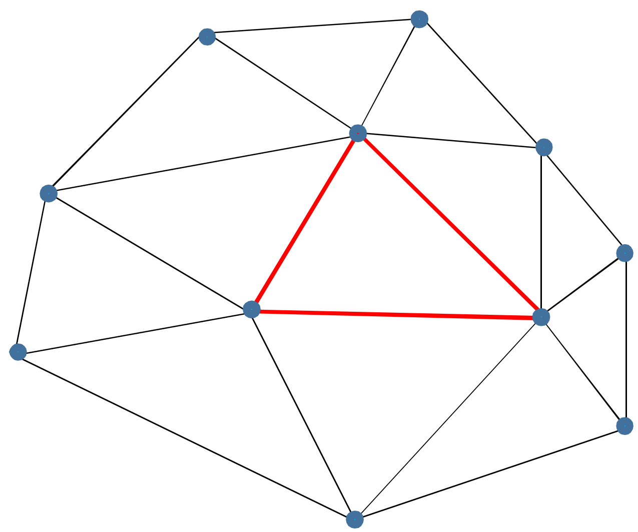
Further generalizations to 3-D (and beyond) fail to assure well-balanced tetrahedral elements based on 3-D scattered nodes.

Step 2: For each surface triangle, project it with some nearby nodes to a tangent plane



Note: The projection is from the sphere center, so it is not a stereographic (conformal) mapping. However, all spherical triangles map to straight-line triangles in tangent plane.

Step 3: Find quadrature weights over the tangent plane node set for the target triangle



Over each 2-D node set surrounding the central triangle Δ, find an RBF interpolant

$$s(\underline{x}) = \sum_{i=1}^{n} \lambda_i \phi(\| \underline{x} - \underline{x}_i \|) + \{a + bx + cy + \ldots\}$$

with matching constraints $\sum_{i=1}^{n} \lambda_i = \sum_{i=1}^{n} \lambda_i x_i = \sum_{i=1}^{n} \lambda_i y_i = \ldots = 0$

Then $\iint_{\Delta} s(\underline{x}) d\underline{x}$ will evaluate to the form $\sum_{i=1}^{n} w_i f_i$,
Where w_i are quadrature weights.

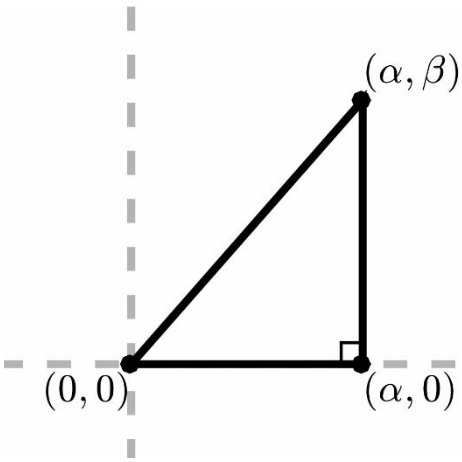
In order to determine the weights w_i, one needs to evaluate

$\iint_{\Delta} \{\text{bivariate polynomial terms}\}\, d\underline{x}$ Elementary

$\iint_{\Delta} \phi(\| \underline{x} - \underline{x}_i \|)\, d\underline{x},\ i = 1, 2, \ldots, n$ Less elementary, but closed forms available (next slide)

Step 3: Find quadrature weights over the tangent plane node set for the target triangle ... (continued)

Case when an RBF is centered at an acute corner (here at the origin) of a right-angle triangle:



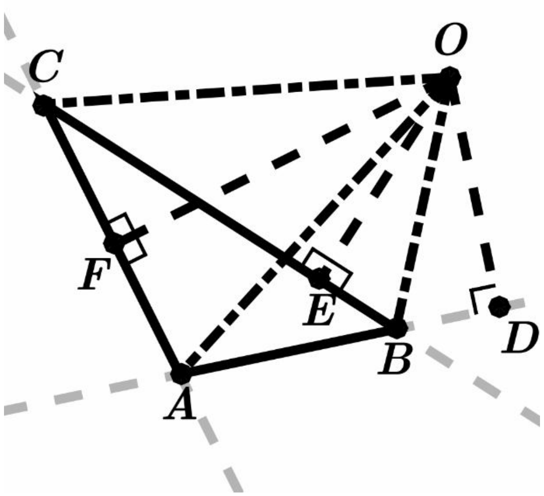
Explicit formulas available for $\phi(r) = r^{2k+1}, \phi(r) = r^{2k} \log r,$ etc.
For example:

$$\iint_\Delta r^3 \, d\underline{x} = \frac{\alpha}{40}\left(3\alpha^4 \operatorname{arcsinh}\left(\frac{\beta}{\alpha}\right) + \beta\sqrt{\alpha^2+\beta^2}\left(5\alpha^2+2\beta^2\right)\right)$$

$$\iint_\Delta r^5 \, d\underline{x} = \frac{\alpha}{336}\left(15\alpha^6 \operatorname{arcsinh}\left(\frac{\beta}{\alpha}\right) + \beta\sqrt{\alpha^2+\beta^2}\left(33\alpha^4+26\alpha^2\beta^2+8\beta^4\right)\right)$$

etc.

Case when an RBF is centered at location "*O*" outside an arbitrarily shaped triangle "Δ*ABC*":



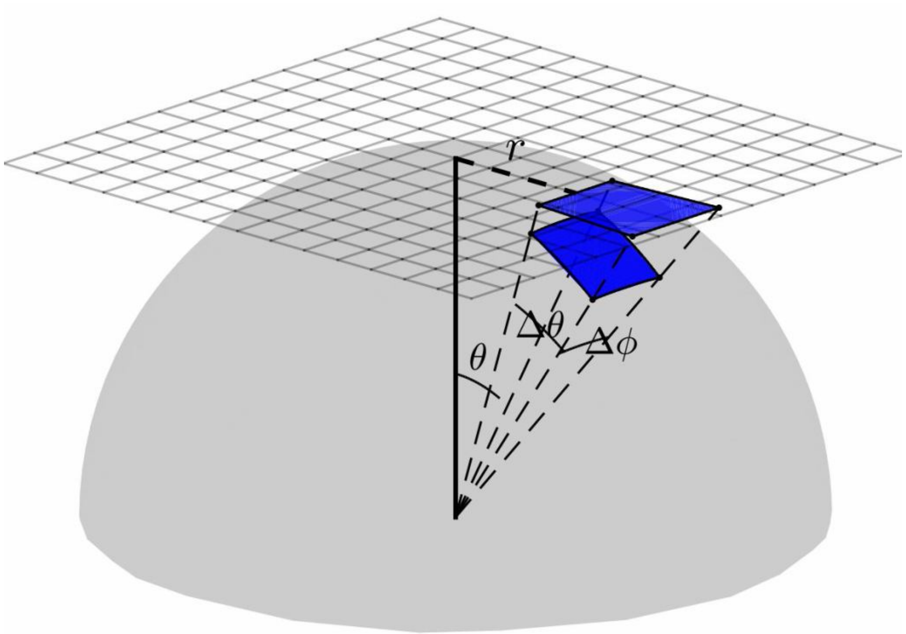
Task reduces to a combination of six cases as described above.

For example, with the nodes placed as shown to the left:

$$\iint_{\Delta ABC} \phi \, d\underline{x} = \iint_{\Delta OAD} \phi \, d\underline{x} - \iint_{\Delta OBD} \phi \, d\underline{x} + \iint_{\Delta OAF} \phi \, d\underline{x} + \iint_{\Delta OCF} \phi \, d\underline{x} - \iint_{\Delta OBE} \phi \, d\underline{x} - \iint_{\Delta OCE} \phi \, d\underline{x}$$

A few lines of code suffice to find *C*, *D*, *E*, and the signs (+ or -) for the six integrals.

Step 4: Convert weights from the tangent plane case to the corresponding weights on the surface of the sphere



Elementary trigonometry gives for infinitesimal area elements:

$$\frac{\text{Area}_{\text{element in plane}}}{\text{Area}_{\text{element on sphere}}} = \left(1 + r^2\right)^{3/2}$$

In order to convert weights from a quadrature formula in the tangent plane to one on the surface of the sphere, one simply needs to multiply weights a distance r from the tangent point by the factor $1/\left(1+r^2\right)^{3/2}$.

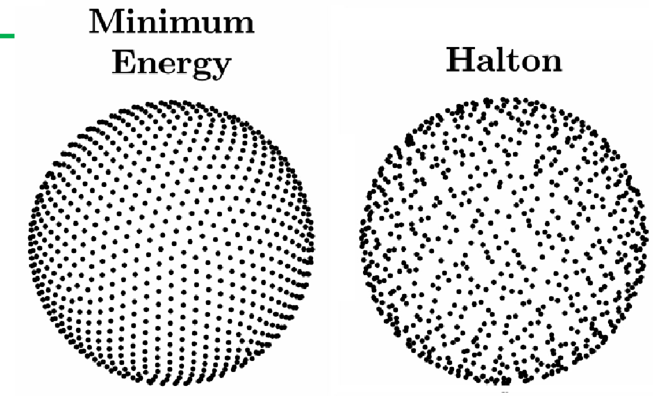
Step 5: Add together the weights for the individual triangles to obtain the full weight set for the sphere

Test problems and results

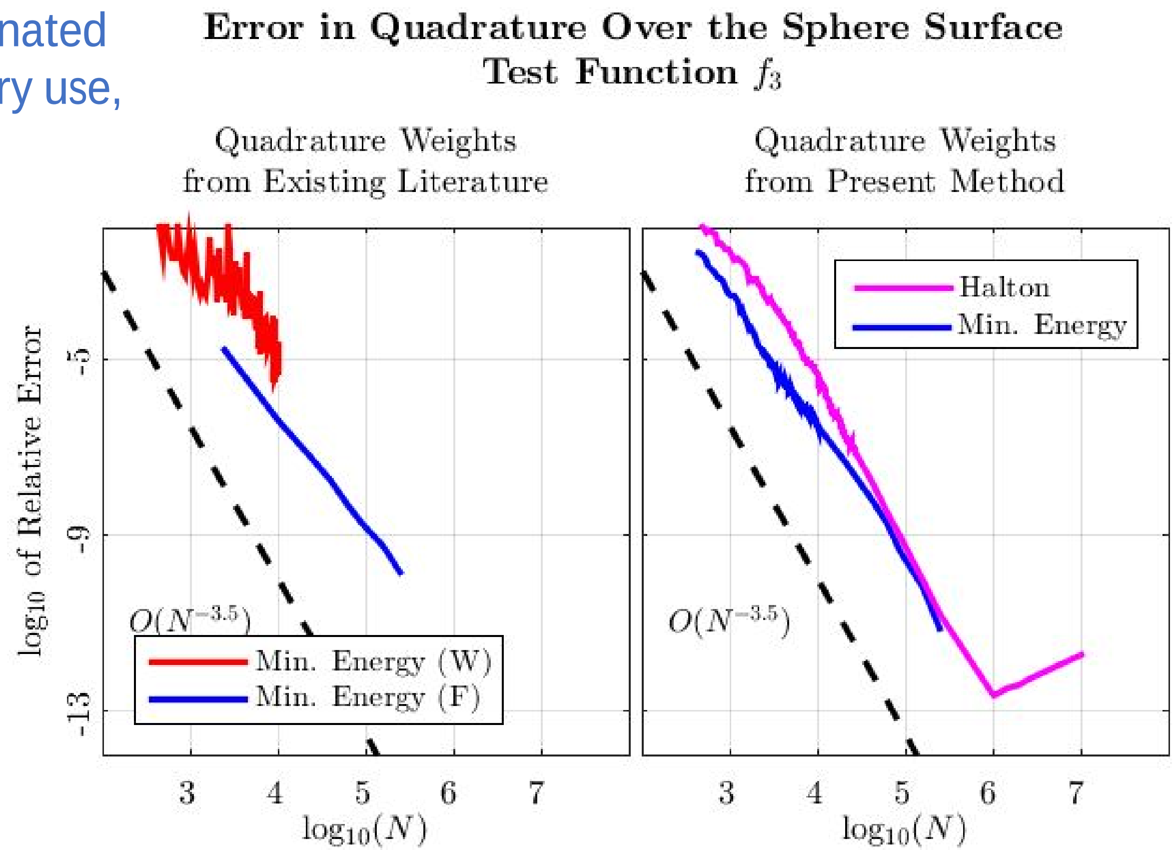
Present method uses default settings when computing quadrature weights for each spherical triangle:

- 80 nearest neighbors
- Bivariate polynomial terms up to degree 7

All results show worst error case when the test function has been randomly rotated 1,000 times

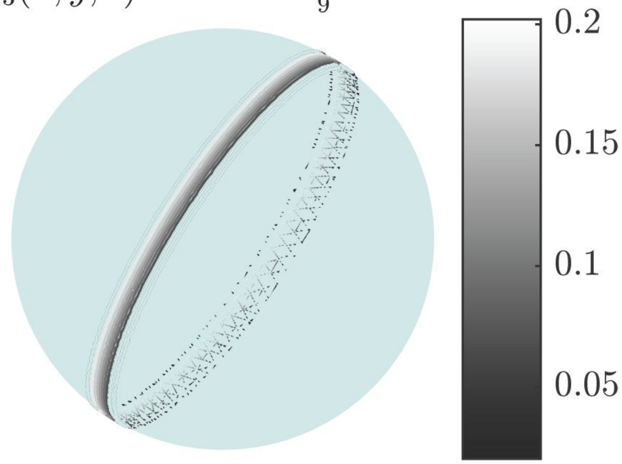


In left error subplot, curves (Womersley & Sloan and Fuselier et.al., resp.) terminated by their $O(N^3)$ cost and $O(N^2)$ memory use, respectively.

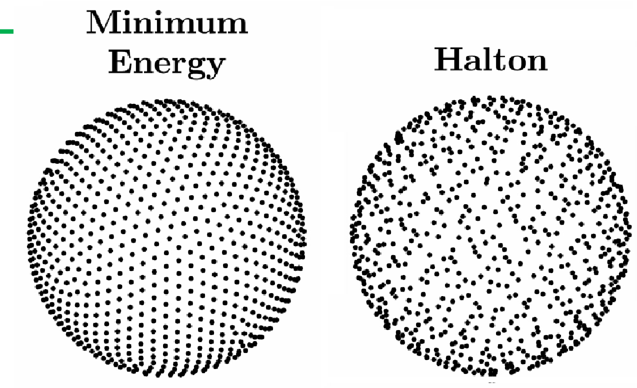


Test function: Infinitely smooth, but sharp gradient only along narrow band around sphere

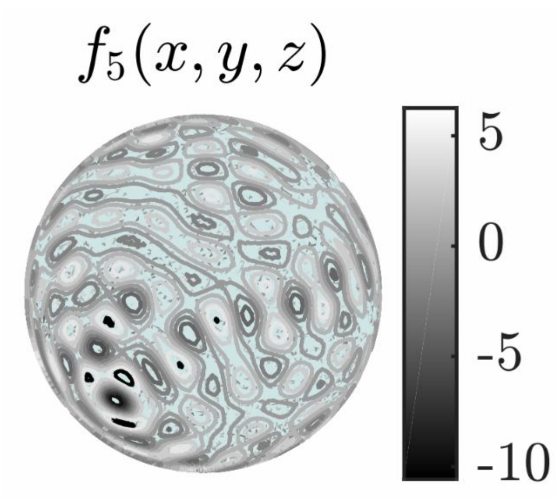
$$f_3(x,y,z) = \frac{1+\tanh(-9x-9y+9z)}{9}$$

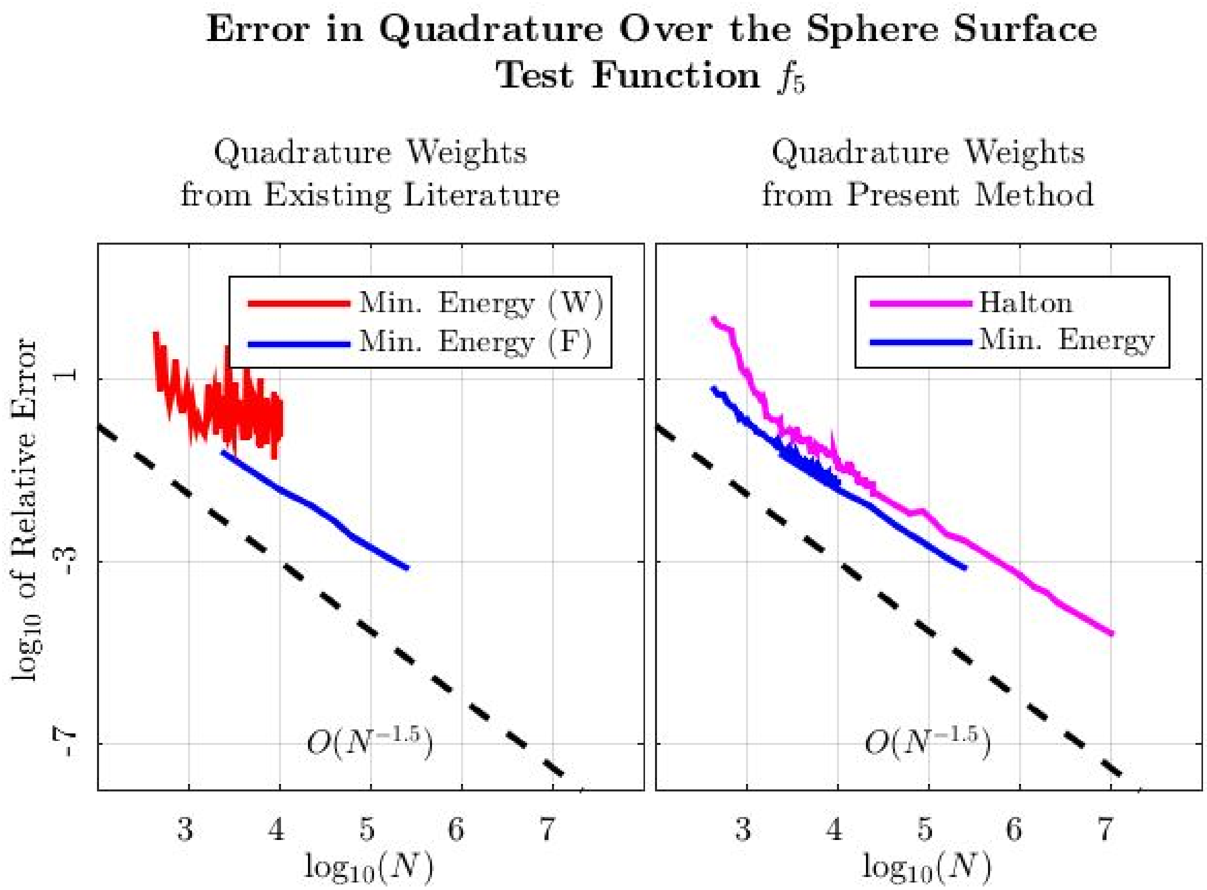


Test problems and results

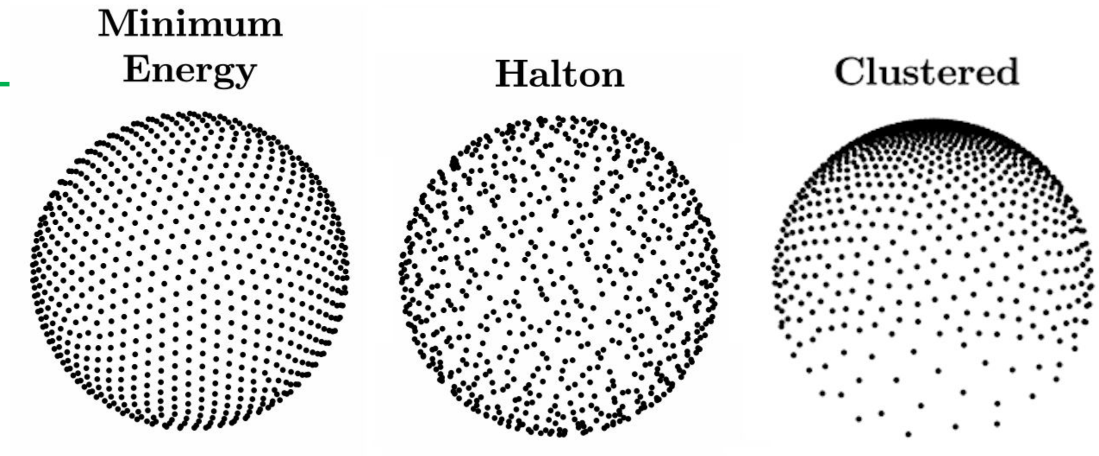


Test function: Highly oscillatory over whole sphere, with singularity at tip of sharp spike (lowering the convergence rate for all methods)



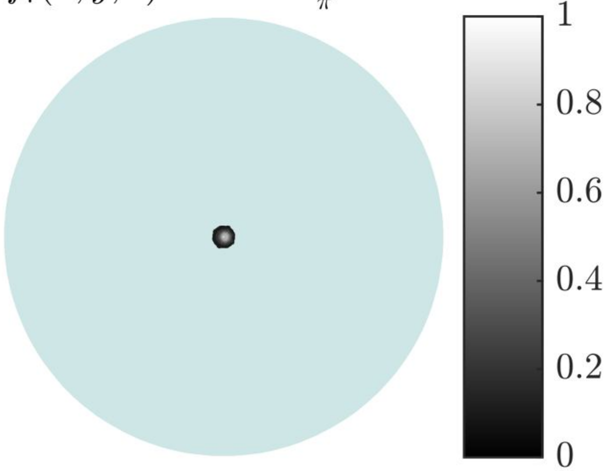


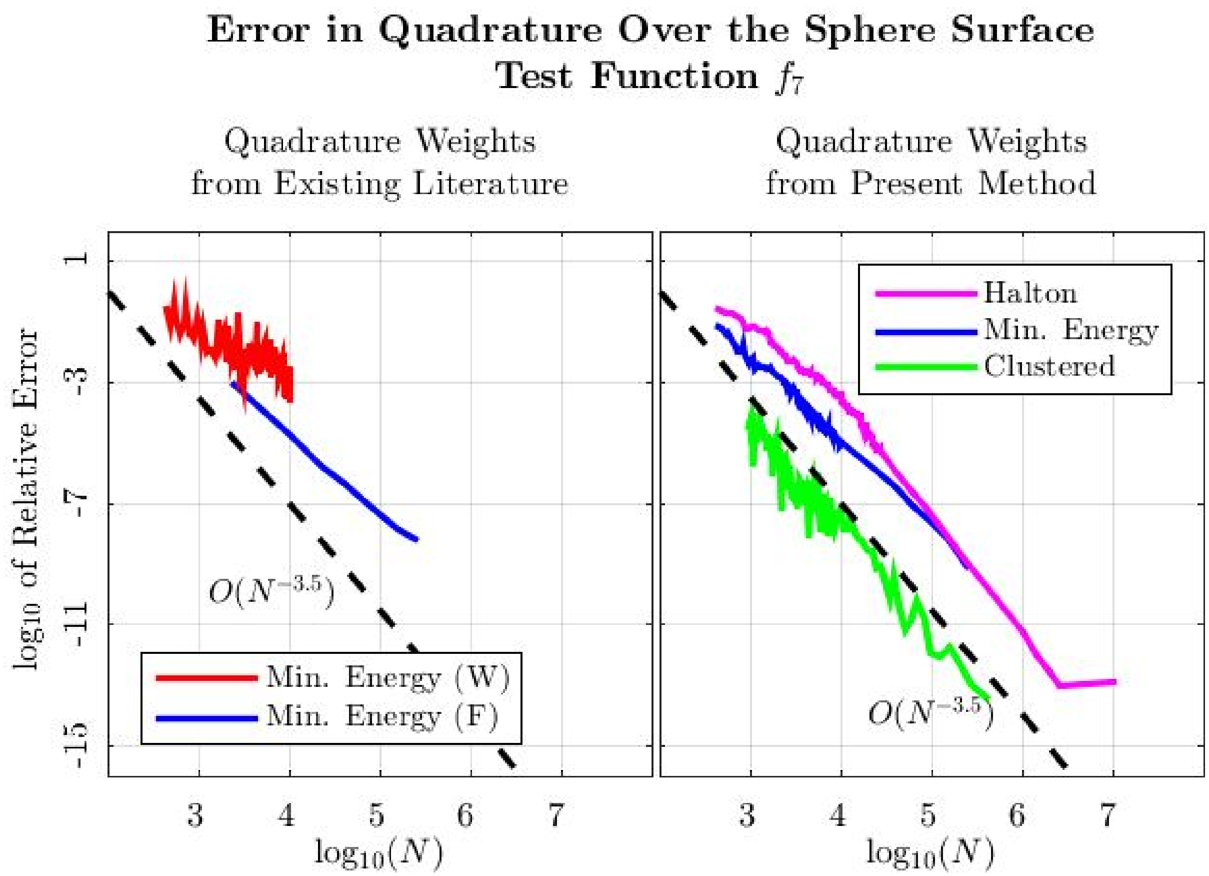
Test problems and results



Test function: Infinitely smooth, but extremely spiked at one location

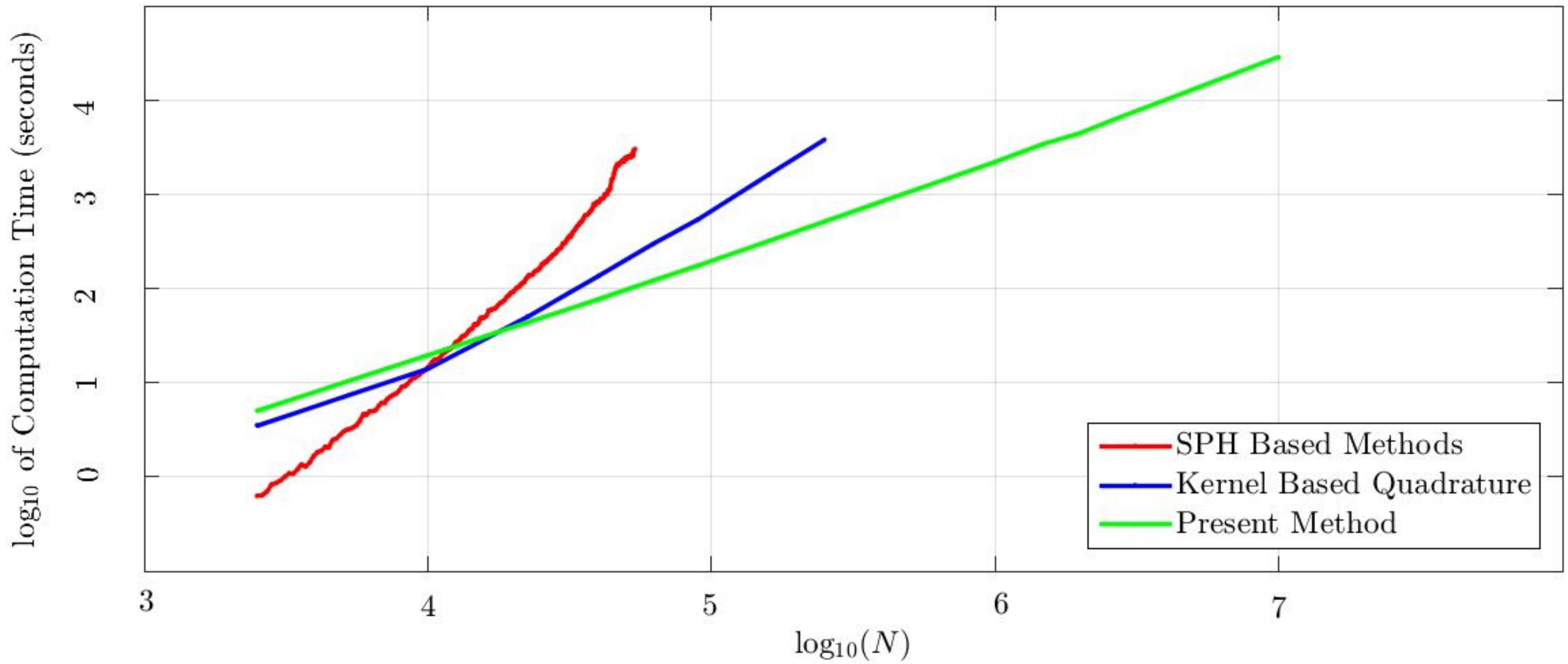
$$f_7(x,y,z) = \frac{\frac{\pi}{2}+\text{atan}(\sigma(z-z_0))}{\pi}$$





Timing results

- Computations for ‘Present’ and ‘SPH based’ methods performed on a Pentium i7-2600, 3.40 GHz, 16.0 GB RAM in MATLAB R2013a
- Timing for ‘Kernel Based Quadrature’ converted from a different system



- Both the SPH and the Kernel Based Quadrature size limited by their $O(N^2)$ memory requirements

- Using parfor in Matlab, the times for the present method can be reduced in proportion to the number of available cores

Conclusions

- A high order accurate algorithm has been developed for quadrature over the surface of a sphere

- The node sets can feature any types of density variations (e.g. local refinement in certain areas, etc.)

- The total cost is $O(N \log N)$ operations and $O(N)$ memory for finding weights for N nodes. The algorithm is 'embarrassingly parallel' , making it trivial to use any number of available processors.

- Even on a standard PC, it can be run for N-values in the millions. This eliminates the need for tabulating weights for specific node distributions.

Manuscript in preparation:

Numerical quadrature over the surface of a sphere (J.A. Reeger and B. Fornberg).