MTB Challenge: Optimizing Lebedev Quadrature Points Sorting on a Sphere

MATLAB RFspin Challenge — Winter Term 2023/24

November 12, 2023

1 Motivation

When performing numerical integration over a spherical shell, the Lebedev quadrature points represent a fundamental tool for achieving high-precision results. This challenge invites participants to apply their algorithmic thinking to sort these points optimally – a problem with practical implications yet underpinned by rich theoretical considerations. It is an excellent opportunity for both novices and experts in computational mathematics. We are not looking for groundbreaking revelations but rather clever, practical solutions that improve current methodologies.

2 Task

Develop an algorithm that optimizes the sorting of Lebedev quadrature points on a sphere, aiming to maximize the ratio of the original distance to the optimized distance after sorting

$$a = \frac{d_{\text{orig}}}{d_{\text{optimal}}}. \tag{1}$$

For clarification, both original and optimal distances must be measured using the Great-circle theorem. Numerically, this distance can be obtained using a dot and cross-product with the following set of equations

$$\Delta\sigma = \arctan\left(\frac{\mathbf{n}_1 \times \mathbf{n}_2}{\mathbf{n}_1 \cdot \mathbf{n}_2}\right), \tag{2}$$

where the $\Delta\sigma$ represents the central angle between two points and vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ refer to the normals to the ellipsoid at the two positions. The final arc distance between two points is $d = r\Delta\sigma$, where r represents the sphere radius. However, considering that computing the great-circle distance is an insignificant part of the problem, participants will receive a MATLAB function that calculates the total distance of the quadrature.

Figure 1 shows a result representation. In the left part of the Figure, you can see the original order of the points and the initial travel route. The centre pane shows an optimal route. The acceleration a equals 4.27 using the metric from (1). The right pane shows a graphical representation of the great-circle routes between the points on the sphere.

As a hint, we suggest participants study the Traveling salesperson problem.

3 Resources Provided

1. Three sets of Lebedev quadrature points. Test Set—Points with $L_{\max} = 3$ (see Figure 2), Competitive Set—Points with $L_{\max} = 11$, Advanced Set—Points with $L_{\max} = 13$. Notice that $L_{\max}$ denotes the degree of quadrature and predetermines number of points used (26, 194, and 266 points, respectively).

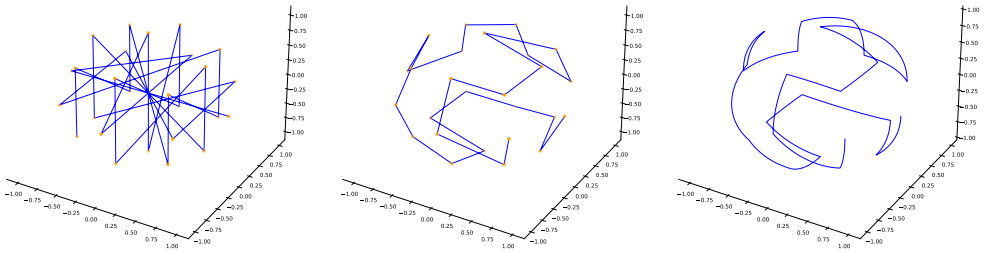

Figure 1: Comparison between non-optimal (left) and optimal (centre) route through all points given by the Lebedev quadrature of degree $L_{\max} = 5$. The right pane shows the optimally sorted points and the final travel route (following the spherical surface).

2. Comparative diagrams showcasing the path differences between the original and sorted lists for the test set ($L_{\max} = 3$).
3. A guide and code snippet for calculating the distance between points on a sphere.

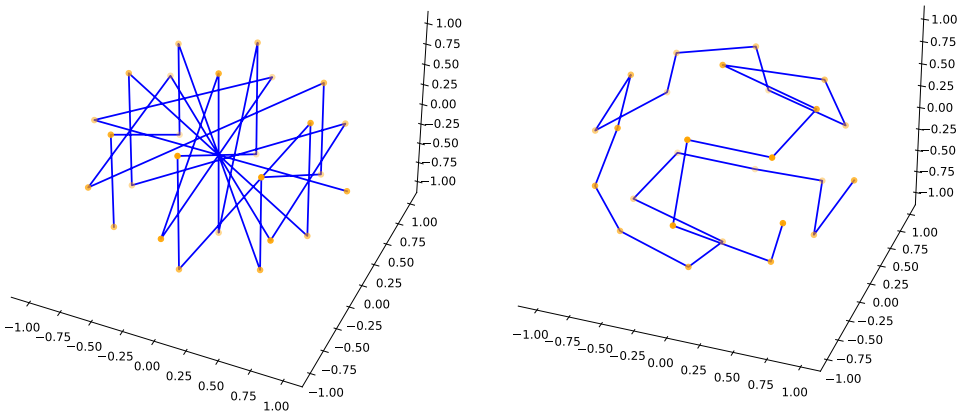

Figure 2: Comparison between optimal and non-optimal Lebedev quadrature for $L_{\max} = 3$ with acceleration 2.97.

4 Task requirements

1. Optimize the sorting of the points for each set provided.
2. Calculate and report the acceleration ratio a (1).
3. For the "Advanced" set, the algorithm must not exceed a run-time of 4 hours on our designated machine.
4. Document the acceleration or performance improvement of the sorting algorithm. Notice that it took approx. 340s to calculate $L_{\max} = 11$, CPU: i7-1185G7, RAM: 32 GB.
5. Output the sorted points into a .MAT or .TXT file.
6. Create an animation illustrating the sorting process (animation is not a part of the patch evaluation and its computational time is not measured). Use your creativity and surprise us.
7. As a bonus, draw the Great-circle arches between the points as shown in Fig. 1 on the right side.

5 Prize

The winning code will be compared with the commercial solution provided by RFSpin company for near-field measurement system. A bottle of the finest Cyrodilic brandy will be presented to the winner.

Participating in the competition will automatically reward you with some of the unique MATLAB merchandise.

The three best solutions will also be rewarded with Humusoft vouchers for MATLAB courses, where you can learn some of the advanced functionalities of MATLAB language, such as Simulink, parallel computations, and embedded coding.

6 Submission Guidelines

1. The final submission should include the optimized code, the resulting sorted points files, and the accompanying animation.
2. Participants are encouraged to comment on their code for clarity and briefly explain their algorithm's approach.
3. Presentation of the algorithm and a quick showcase.
4. An unlimited number of students can select this project. However, no collaboration between students is expected.
5. Like regular projects, a short presentation (a few minutes) is expected.
6. No external MATLAB toolboxes or third-party libraries are allowed.
7. It is possible to always withdraw from the competition and select one of the regular projects. This decision should be discussed with the teachers; their approval is required.

7 Evaluation Criteria

1. The efficiency of the sorting algorithm according to a.
2. Compliance with the run-time constraint for the 'Advanced' set.

8 Deadline

Complete the challenge till **January 5, 2024, 23:59**.

9 Submission Process

Submit the project files via BRUTE.

10 Contact Information

For any queries or clarifications, please contact:

- Štěpán Bosák, bosak@rfspin.com,
- matlab@fel.cvut.cz

We encourage your participation and look forward to your innovative solutions. Good luck to all contestants!