The Follower algorithm and a program using it to exploring spaces

Lawrence C. Andrews[a]* and Herbert J. Bernstein[b]

[a]Ronin Institute, 9515 NE 137th St, Kirkland, WA, 98034-1820 USA, and [b]Ronin Institute, c/o NSLS-II, Brookhaven National Laboratory, Upton, NY, 11973 USA.

E-mail: lawrence.andrews@ronininstitute.org

Follower; **Delaunay**; **Delone**; **Niggli**; **Selling**

Abstract

The Follower algorithm was developed to examine the properties of continuous functions and to verify their computer implementations. In particular, we studied the distance measures for distances between unit cells.

1. Introduction

We have created several methods for determining "distances" between unit cells (or lattices): $\mathbf{V^7}$ (Andrews *et al.*, 1980), $\mathbf{G^6}$ (Andrews & Bernstein, 1988), $\mathbf{D^7}$ (Andrews & Bernstein, 2014), and $\mathbf{S^6}$ (Andrews *et al.*, 2019*b*). They have been used for the creation of databases (Andrews *et al.* (1980) and Andrews *et al.* (2019*a*)), for identification of Bravais lattice types (Andrews & Bernstein, 1988), and for implementing clustering for serial crystallography (Bernstein *et al.*, 2020) (Nguyen *et al.*, 2020).

1.1. Issues to be solved

Implementation of computer methods for distance calculations in spaces $\mathbf{V^7}$, $\mathbf{G^6}$, $\mathbf{D^7}$, and $\mathbf{S^6}$ begins by reduction to standard form (Niggli, 1928) (Delaunay, 1932). The implementations are complicated by the need for iterative examination of many alternative unit cells that may be close to matching the reduced cell (Andrews *et al.*, 1980), (Andrews & Bernstein, 1988), (Andrews & Bernstein, 2014). There is no presently-known method to evaluate how deeply these iterations through alternatives should be taken, resulting in complex computer code that is difficult to verify.

These being studies where the correct answers cannot be foreseen, a method to at least test for consistency of the results was needed. Two desirable properties of a valid measure would be that the distance between two points would vary continuously with displacement and that two non-coincident points would not have a distance measure of zero.

Because we are working with maps on manifolds (locally flat Euclidean tangents to curved surfaces) rather than simple global Euclidean spaces, we cannot extrapolate linearly from what we see in the small, which is sufficient for databases and clustering, to the longer distances needed for a clear understanding of the topology of these spaces.

2. Creating Follower

Our solution to the above issues was to develop the Follower algorithm. In its simplest form, the algorithm is initialized with two points. One is chosen as the fixed point and the other becomes the mobile point. Then (in the chosen space) the mobile point is moved along a line to the fixed point, computing the distance between them at each step. Figure 1 shows one kind of result that might be expected. In the figure, a mobile point has been chosen, and the Niggli-reduced cell will be the fixed point. The graph displays the $\mathbf{G^6}$ (Andrews & Bernstein, 2014) distance from the mobile point to the fixed point.

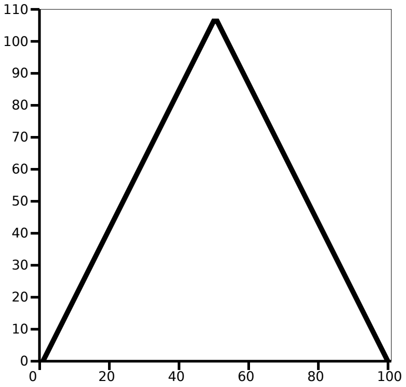

Fig. 1. Simple Follower plot where Euclidean distance is adequate. Here the initial mobile point is the unreduced unit cell 10.45274 10.21606 10.34577 89.89371 91.52875 92.22760 $(a, b.c.\alpha, \beta, \gamma)$, and the fixed point is the corresponding Niggli-reduced cell 10.21607 10.34577 10.45275 88.47129 87.77250 89.89375. 101 points are sampled; the abscissa is the $\mathbf{G^6}$ distance between them in units of Å^2. The graph ends with zero distance because the mobile point taken the same value as the fixed point. This initial point has zero distance because it is a sample from the same lattice as the reduced cell (the final point).

3. Analogy

To anticipate the complexity that Follower might show us, consider a sodium chloride crystal. Inject a proton into the crystal and determine the distance to the nearest sodium ion. Then move the proton through the crystal in a straight line. Barring a head-on collision, as the proton approaches the crystal the distance to the nearest sodium will first decrease and then increase, but eventually another sodium ion will be closer, and the distance to the newly nearest sodium ion will decrease and then later increase. This fall, rise, fall, and rise of the distance will continue as the proton movement continues through the crystal.

In the same fashion, in Follower the mobile point moves through the multitude of representations of the crystal lattice. We will expect that sometimes the shortest distance will increase and sometimes decrease. This might happen several times.

4. Alternate path choices

In our implementation of Follower, there are five different choices of measurements.

POINT: a single unit cell is input. The generated path goes from the input point to its reduced cell.

LINE: two points are input. The first point becomes the mobile point, and the other point is used as the fixed point.

CHORD: two points are input. Each point follows a line to its reduced cell. The reported distance is between the two mobile points as they move on their separate lines.

CHORD3: three points are input. The first and second are mobile points as in **CHORD**, but they maintain their initial separation. The midpoint between the two mobile points moves from the initial midpoint towards the third point. The reported distance is between the two mobile points.

TRIANGLE: three points are input. The third point is fixed. The other two move toward the fixed point. The distance between the mobile points is reported.

The calculations of the points' positions along the line are done in the space $\mathbf{S^6}$ (Andrews *et al.*, 2019*a*). However, the distances can be calculated in whichever space the user chooses.

5. Results

Follower is a program with many options. Here we present a few typical graphs of some of the options one might choose.

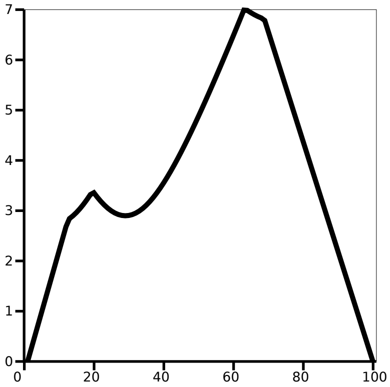

Fig. 2. POINT method with initial mobile point 9.251, 1.253, 8.37126, 146.78, 115.49, 96.40.

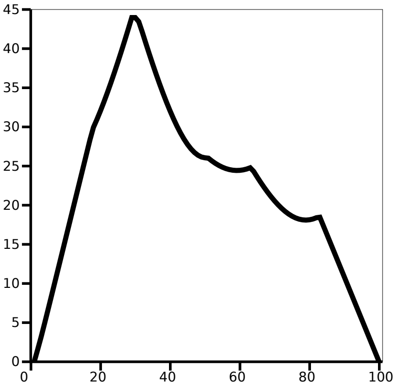

Fig. 3. POINT method with initial mobile point 12.075, 9.894, 7.832, 102.47, 105.51, 132.43.

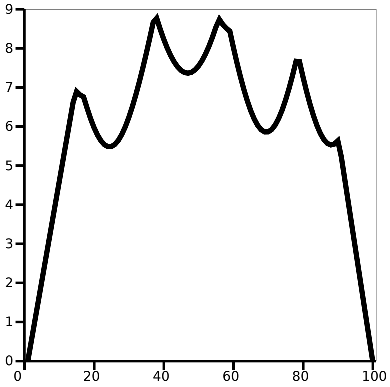

Fig. 4. POINT method with initial mobile point 9.509, 9.353, 9.388 119.74, 117.25, 112.56.

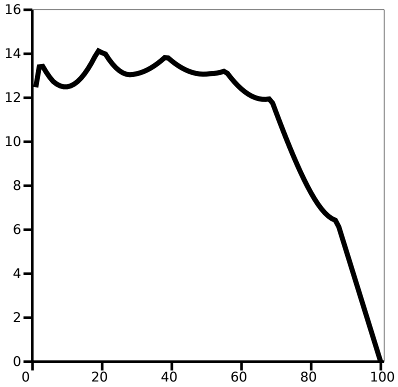

Fig. 5. LINE method with mobile point: 5.31869 7.14692 5.55815 110.63075 91.27139 136.00720, and fixed point: 5.165, 5.417, 5.983, 115.11, 103.91, 92.56.

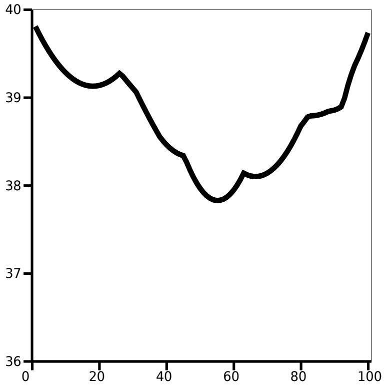

Fig. 6. CHORD method with mobile points: 9.629, 3.180, 8.906, 119.86, 112.27, 100.43 and 8.659 5.863, 5.395, 110.28, 115.18, 124.98.

6. Summary

The Follower algorithm for examining continuous functions is described.

7. Availability

ADD SOME DESCRIPTION

Acknowledgements: Careful copy-editing and corrections by Frances C. Bernstein are gratefully acknowledged. Our thanks to Jean Jakoncic and Alexei Soares for helpful conversations and access to data and facilities at Brookhaven National Laboratory.

References

Andrews, L. C. & Bernstein, H. J. (1988). *Acta Cryst.* A**44**(6), 1009 – 1018.

Andrews, L. C. & Bernstein, H. J. (2014). *J. Appl. Cryst.* **47**(1), 346 – 359.

Andrews, L. C., Bernstein, H. J. & Pelletier, G. A. (1980). *Acta Cryst.* A**36**, 248 – 252.

Andrews, L. C., Bernstein, H. J. & Sauter, N. K. (2019*a*). *Acta Cryst.* A**75**. https://doi.org/10.1107/S2053273318015413.

Andrews, L. C., Bernstein, H. J. & Sauter, N. K. (2019*b*). *Acta Cryst.* A**75**(3), 593–599.

Bernstein, H. J., Andrews, L. C., Diaz Jr, J. A., Jakoncic, J., Nguyen, T., Sauter, N. K., Soares, A. S., Wei, J. Y., Wlodek, M. R. & Xerri, M. A. (2020). *Struct. Dyn.* **7**(1), 014302.

Delaunay, B. N. (1932). *Zeitschrift für Kristallographie*, **84**, 109 – 149.

Nguyen, T., Phan, K. L., Kreitler, D. F., Andrews, L. C., Gabelli, S. B., Kozakov, D., Jakoncic, J., Sweet, R. M., Soares, A. S. & Bernstein, H. J. (2020). *bioRxiv.*

Niggli, P., (1928). Krystallographische und Strukturtheoretische Grundbegriffe, Handbuch der Experimentalphysik, Vol. 7, part 1. Akademische Verlagsgesellschaft, Leipzig.

Synopsis

The Follower algorithm was developed to examine consistency and accuracy in computing "distances" between lattices