Flat Norm Decomposition and Computation

Sandra Auttelet[ı], Jared Brannan[ł], Blake Cecil[ł], Curtis Michels[ł], Kevin Vixie[ł]

Washington State University

Abstract

We computed the flat normalization on irregular graphs, only done previously on regular graphs. When discretizing the flat norm, weights were approximated using minimization techniques. This work can be used in graph theory to compare graphs at different scales. By computing the local minimum, the topology of a surface formed from a randomly connected data set can be measured using level set methods across the gradient of the connections between points. This code uses a low-cost method by minimizing the graph cuts.

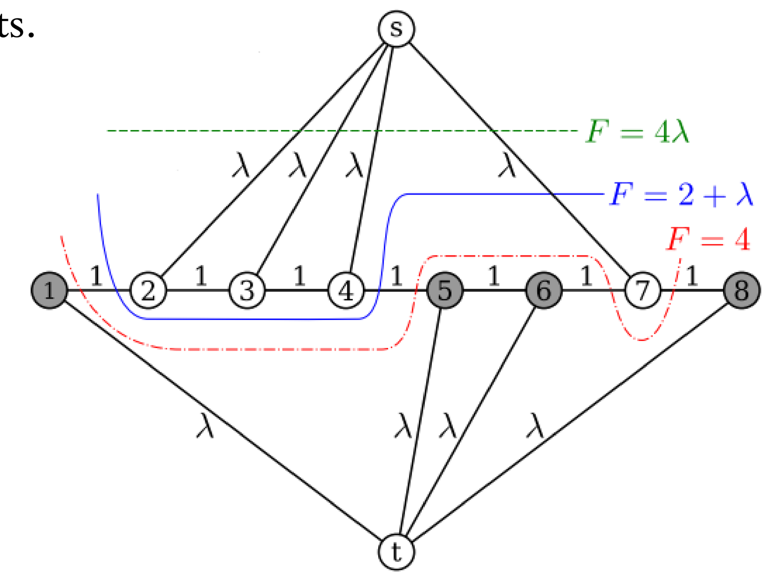


Figure 1: A cut occurs when the flow, F, crosses the connection between the data, black and white circles, and the source node, s. [1]

Objective

The flat norm is a function describing the topology of a surface.

$$\text{Flat Norm} := F(u) = \int_{\Omega} |\nabla u| dx + \lambda \int_{\Omega} |u - f| dx \qquad (1)$$

where Ω is a domain of the input signal u, f is the noisy data, and λ controls smoothing.

The goal of this project was to discretize the flat norm to allow for computation on arbitrary graph sets. The flat norm is computed using min-cut max-flow linear programming techniques.

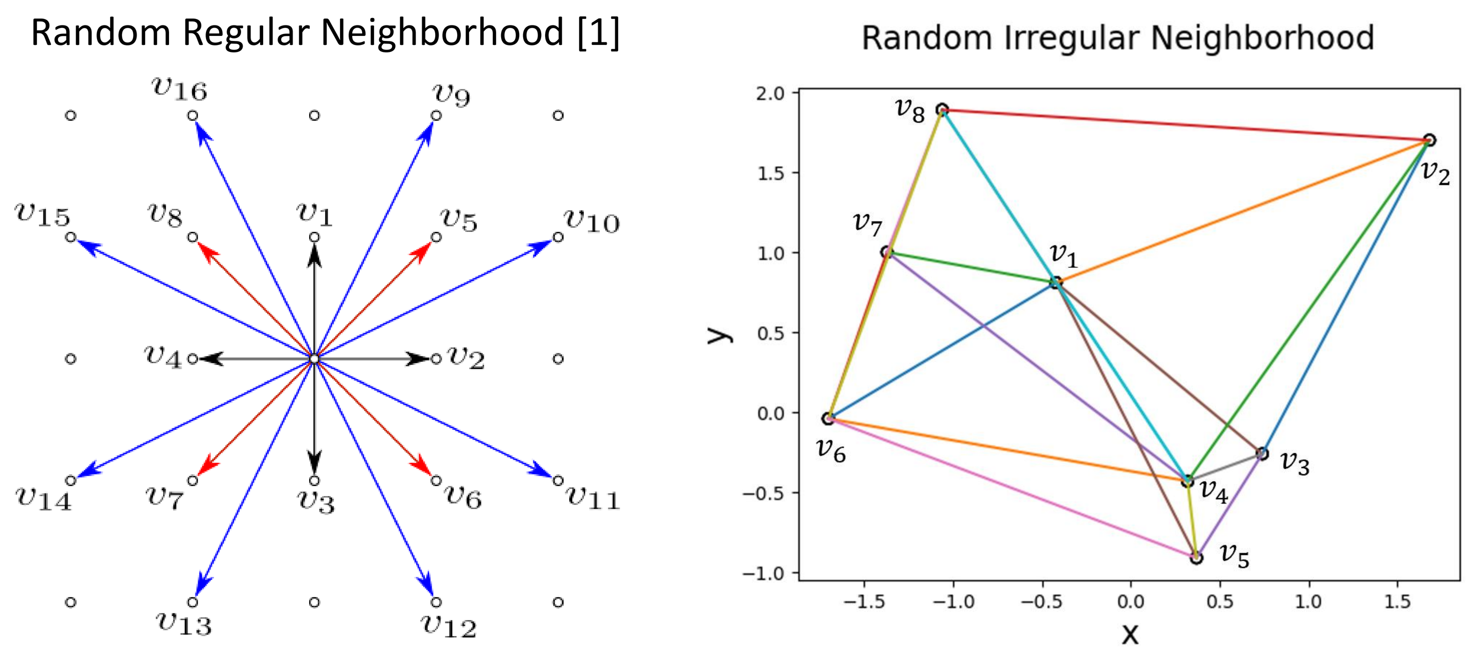


Figure 2: The image on the left is from [1] with weights found by hand. Compared to the right with computationally calculated weights.

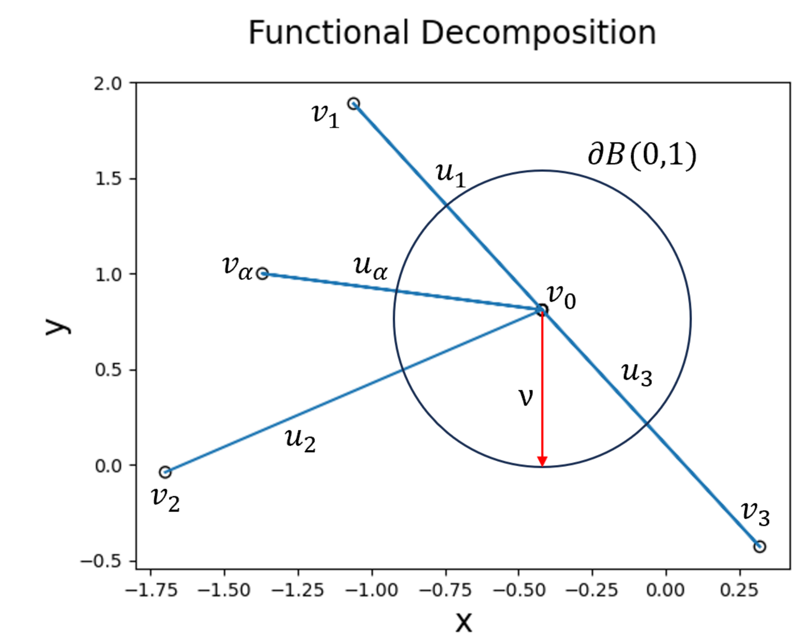
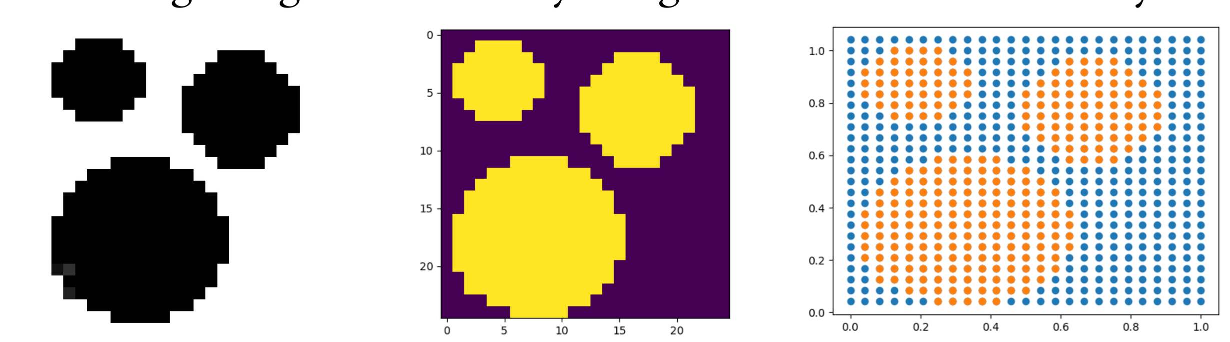


Figure 3: First pick one vertex to call v_0. Then choose one vector for this basis, u_α, and iterate through each connection.

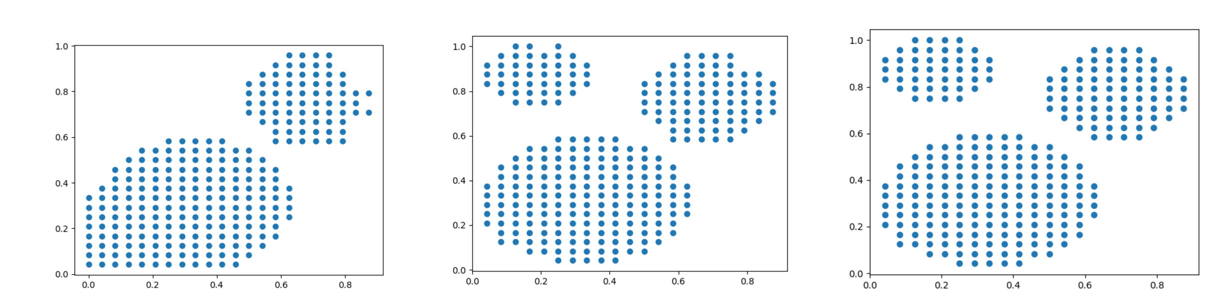
Application of Flat Norm

Figure 10: Image Discretization



The starting image is 25x25 pixels for speed. Once the grid is overlayed with points, some are removed depending on the scale, λ. Increasing the scale to the max level will return the image completely. The scale becomes a tuning feature to attempt to remove noise at a higher frequency than our signal.

Figure 11: Discretized Image with Varying Scale



Decomposition Methods

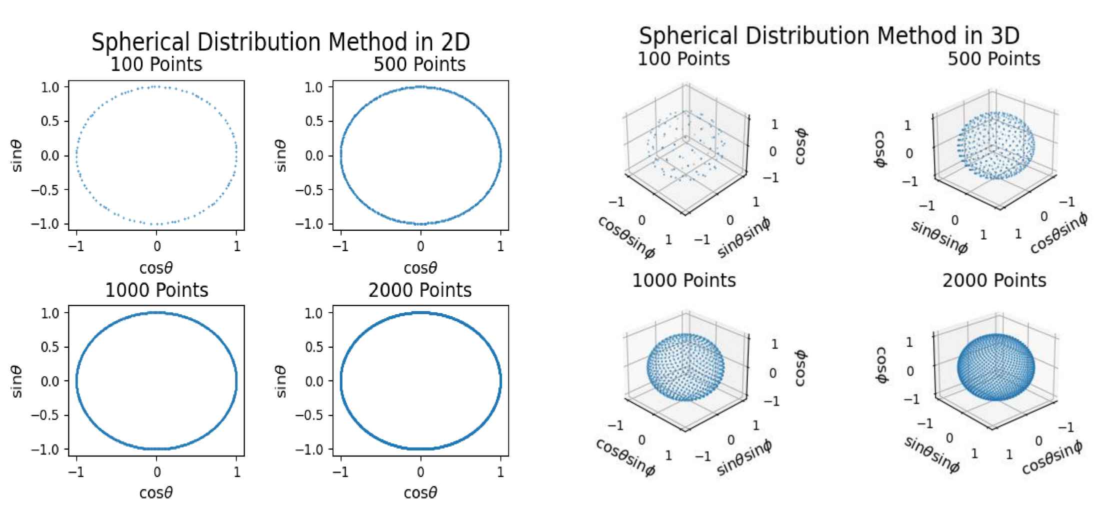


Figure 4: The Spherical Distribution Method used for our C_3 approximation. This method utilized the Fibonacci lattice by implementing the golden ratio. [2]

In [3], they begin with a finite graph and end with the function below to calculate the edge weights:

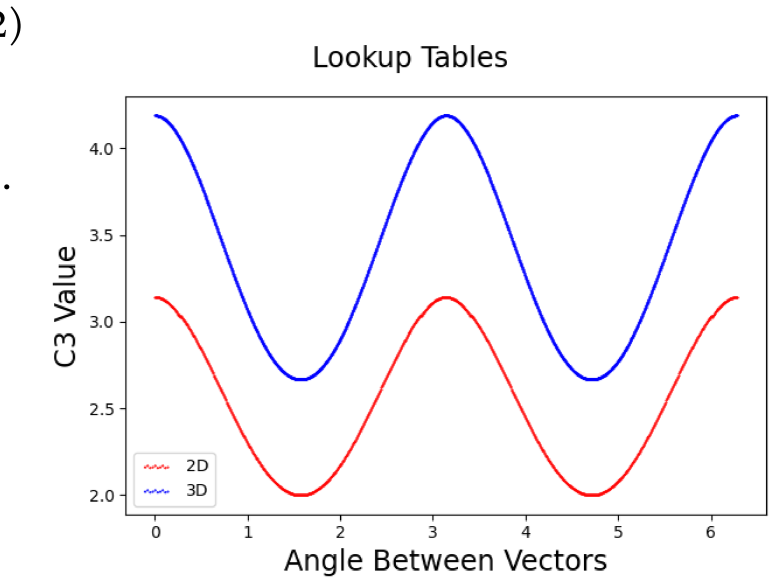
$$2w_j^\alpha |u_\alpha|^2 C_1 + 2 \sum_{m=1;m\neq\alpha}^{N_j} w_j^m |u_m||u_\alpha| C_3 - 2|u_\alpha| C_2. \qquad (2)$$

Equation (2) is specific to a vector, u_α, as shown in Figure 3.

Constants C_1 and C_2 were calculated using a simple volume integral.

C_3 was approximated due to the dependence of the specific vector, u_α, on its nearest neighbors, u_m. The integral was computed using Cubepy.

Figure 5: We built a table with 1000 values for C_3 in 2D and 2000 values in 3D. Tables were built to speed up our computation time.



The max value we obtained in 2D was π and in 3D was $\frac{4\pi}{3}$, the constants used for area and volume calculations on a sphere respectively.

Future Application

We will be collaborating with the LIGO project. Because the min-cut, max-flow can be used for classification, this code can be used to compare "glitches". Types of glitches are defined in [5], and deep learning methods are currently used to classify these signals.

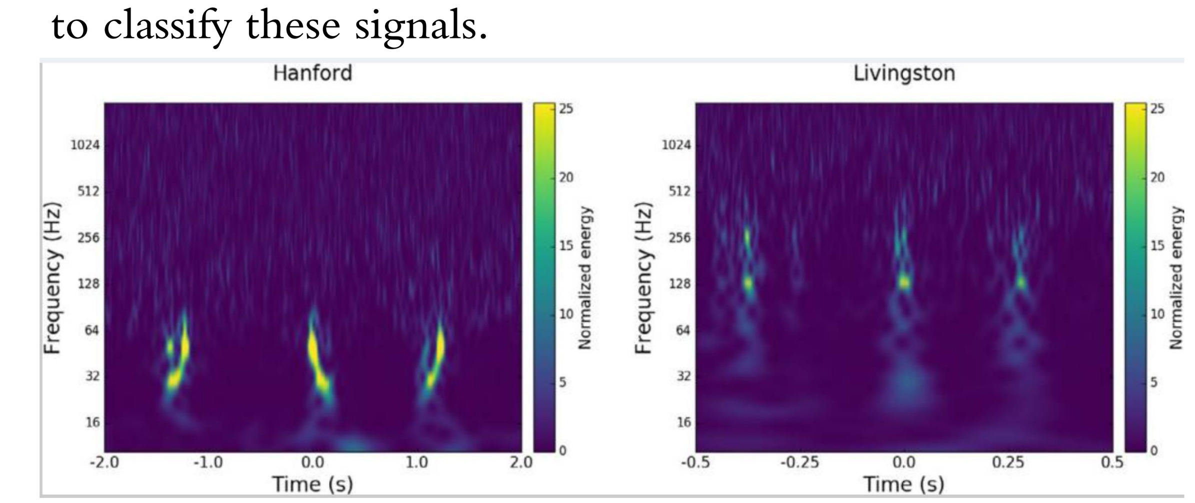


Figure 12: Glitch types: 'Paired Doves' and 'Helix' [5]

Computation Methods

To minimize (2), we first tried the Newton Iteration Method using,

$$w_n = w_{n-1} - \frac{f'(w_{n-1})}{f''(w_{n-1})} \qquad (3)$$

where each weight, w_n, is approximated using the Jacobian and Hessian of each vertex. If $f''(w_{n-1})$ was singular, this method did not work.

Because (2) is convex, we opted to use the Method of Least Squares. This method only requires the Jacobian computation and proved to be as accurate as needed.

Figure 6: Newton Iteration Method

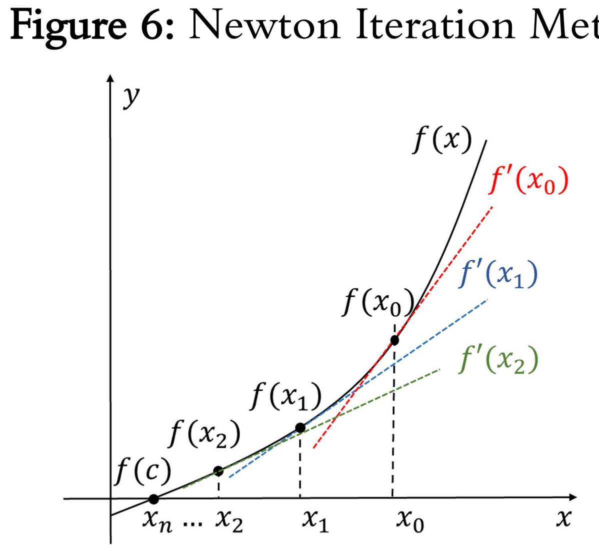
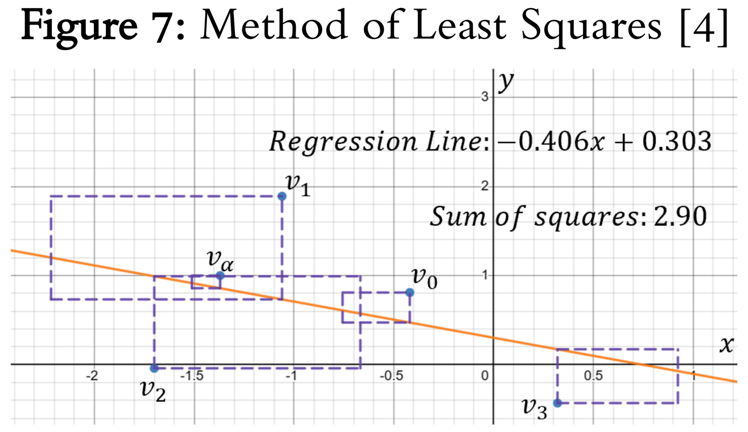


Figure 7: Method of Least Squares [4]



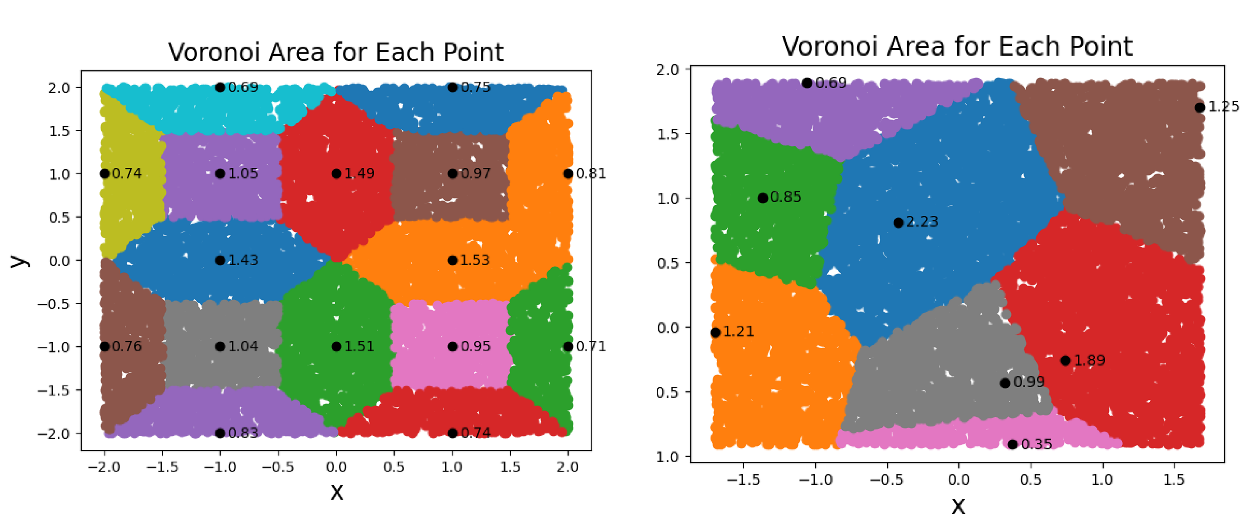


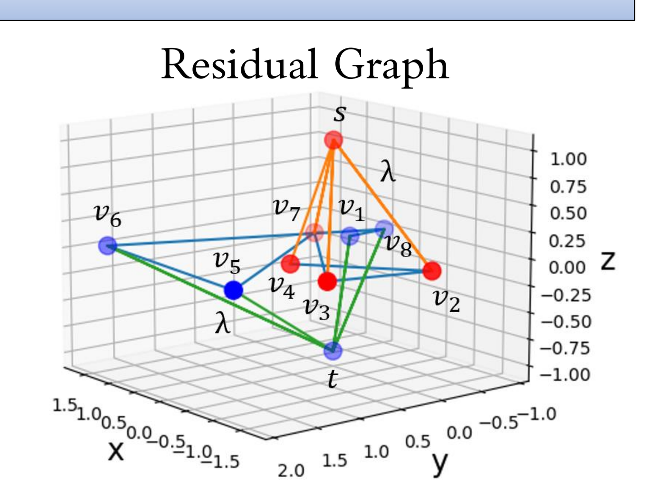
Figure 8: The Voronoi region for each vertex in our graphs from Figure 2.

Each region is approximated using a Monte Carlo method. Because arbitrary graphs are undirected, each edge has a different weight, w_{AB} or w_{BA}, for points A and B. We must normalize our solution, so we have one weight, w, such that

$$w = V_A w_{AB} + V_B w_{BA} \qquad (4)$$

where V_A and V_B denote the Voronoi area.

Figure 9: From min-cut, max-flow of the random neighborhood found using the Edmond Karp flow function.



python matplotlib Pickle SciPy learn NetworkX PyTorch NumPy CubePy

References and Acknowledgements

1. K. Vixie, and K. Clawson and T. Asaki, "Multiscale Flat Norm Signatures of Shapes and Images." *Applied Mathematical Sciences*, Vol. 4, 2010, no. 14, 667 – 680.
2. M. Roberts, "Evenly distributing points on a sphere | Extreme Learning," *Extreme Learning*, Oct. 05, 2020. http://extremelearning.com.au/evenly-distributing-points-on-a-sphere/
3. K. Sabochick, "Derivation Progress," Jun. 24, 2021.
4. Least-Squares Regression Line | Desmos
5. Zevin M, Coughlin S, Bahaadini S, Besler E, Rohani N, Allen S, Cabero M, Crowston K, Katsaggelos AK, Larson SL, Lee TK, Lintott C, Littenberg TB, Lundgren A, Østerlund C, Smith JR, Trouille L, Kalogera V. Gravity Spy: integrating advanced LIGO detector characterization, machine learning, and citizen science. Class Quantum Gravity. 2017;34(No 6):064003. doi: 10.1088/1361-6382/aa5cea. Epub 2017 Feb 28. PMID: 29722360; PMCID: PMC5927381.

ı Sandra Auttelet, an undergraduate in physics and mathematics at Washington State University, would like to thank:

ł Kevin Vixie, mentor and associate professor in mathematics at WSU, for the opportunity to join his research team.

ł Jared Brannan, Curtis Michels and Blake Cecil, graduate students in mathematics at WSU, for working through the code of this functional computation.

This research was funded by the Undergraduate Hacker Research Scholarship for Spring 2022 and Spring 2023, the Leo Millam Undergraduate Research Scholarship for Summer 2022, and by the Tom Dickinson Research Scholarship for Summer 2023.