Flat Norm Decomposition and Computation

Sandy Auttelet[1], Jared Brannan[1], Blake Cecil[1], Curtis Michels[1], Katrina Sabochick[1], Kevin R. Vixie, PhD[1]

[1]Integrated Science and Mathematics, Washington State University

Background

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

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

where Ω is a domain of the input signal u, f is the noisy data, and λ controls smoothing. [1]

Decomposition

Suppose we start with the image of three circles and we attempt to compute the flat norm on this image.

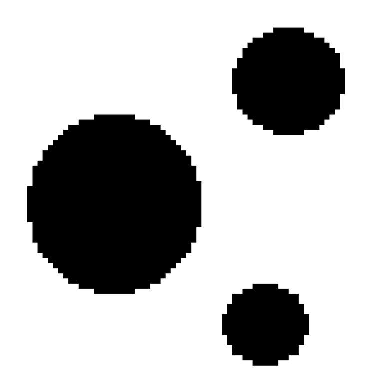


Figure 1. Starting Image

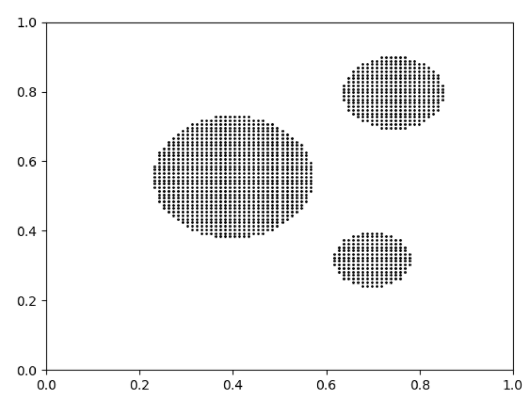


Figure 2. Discretization

First, we define χ_E as the characteristic function on a set E. Then we input u which is an image or signal transformed into a binary input using a threshold value. We discretize the input into our set E by assigning pixels to a point in the case of an image.

Equation (1) is then reduced to

$$F(\Sigma, \lambda) = \text{Per}(\Sigma) + \lambda |\Sigma \triangle \Omega| \qquad (2)$$

where Σ is the support of the input $u = \chi_\Sigma$, $\text{Per}(\Sigma)$ is the perimeter of the set Σ, $\triangle$ denotes the symmetric difference, and Ω is the support of the binary data $f = \chi_\Omega$. [1]

Then if we minimize the penalty term of this function on E, we can decompose the first term below to approximate the edge weights:

$$\min_{W_i} ||F_{W_i, U_i}(\nu) - |\nu|||_2^2 = \sum_{i=1}^{D} w_i^2 C_1^i - 2\sum_{i=1}^{D} w_i C_2^i + 2\sum_{i=1}^{D}\sum_{j=1}^{i-1} w_i w_j C_3^{ij} + \int_{\partial B(0,1)} d\nu. \qquad (3)$$

The last term drops out when we minimize this decomposition so we don't concern ourselves with it here. Then we must discretize (3) for sets. [2]

Spherical Distribution Method

Constants from (4,5) were calculated analytically. Due to the dependence of u_i and its neighboring vector u_j, C_3^{ij} was numerically computed using Cubepy. [3]

Using a spherical decomposition, we define our constants [4]:

$$C_1^i := \int_{\partial B(0,1)} |u_i \cdot \nu| d\nu \qquad (4)$$

$$C_2^i := \int_{\partial B(0,1)} |u_i \cdot \nu|^2 d\nu \qquad (5)$$

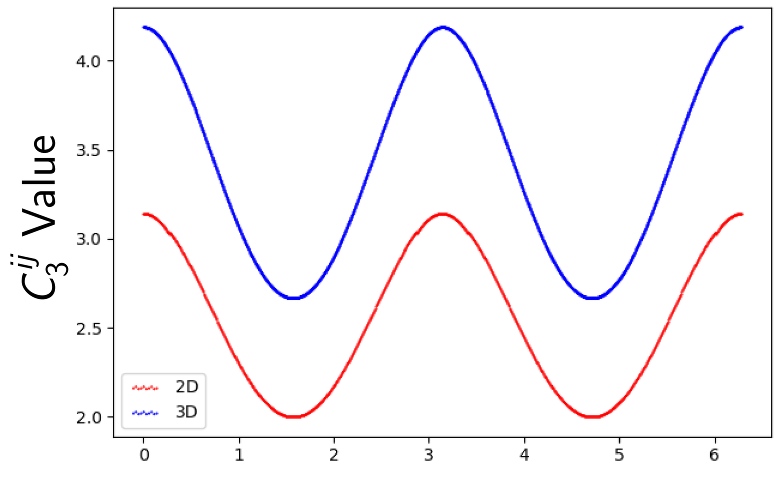
$$C_3^{ij} := \int_{\partial B(0,1)} |u_i \cdot \nu||u_j \cdot \nu| d\nu \qquad (6)$$

where $\nu = (\cos\theta, \sin\theta)$ in the 2D case and $\nu = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$ in the 3D case.

Table 1. Constant Solutions

	2D	3D
C_1^i	$\pi\|\|u_i\|\|^2$	$\frac{4\pi}{3}\|\|u_i\|\|^2$
C_2^i	$4\|\|u_i\|\|$	$2\pi\|\|u_i\|\|$
C_3^{ij}	Numerical	Numerical

Figure 3. A table of C_3^{ij} values to decrease user computation time.



Increasing the degree for each vertex will increase the accuracy of the computation, but runtime will be impacted. For a graph with V vertices of degree D and with edges E,

$$\text{Runtime Complexity} \approx O(|V|\log|V| + |V||D|^3 + |V|^2|E|).$$

The first term is from constructing the k-d tree, the second is from the weight computation and the last term is from the min-cut max-flow algorithm.

Computational Method

Figure 4. Regular Graph Voronoi Regions

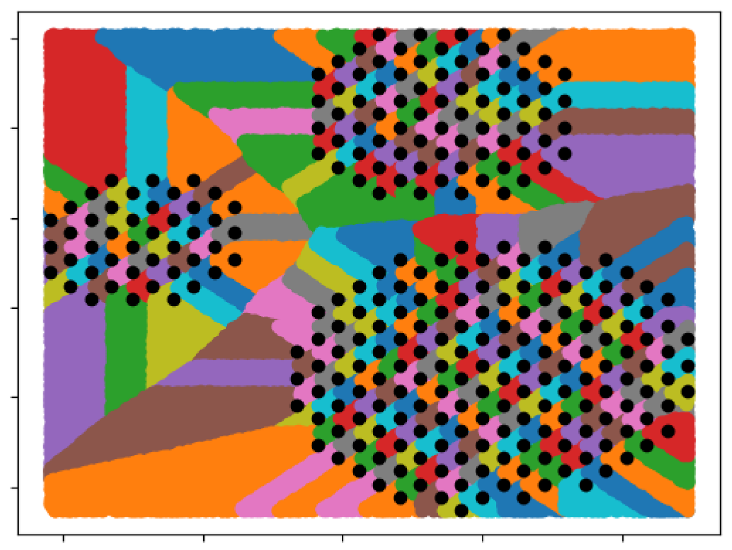
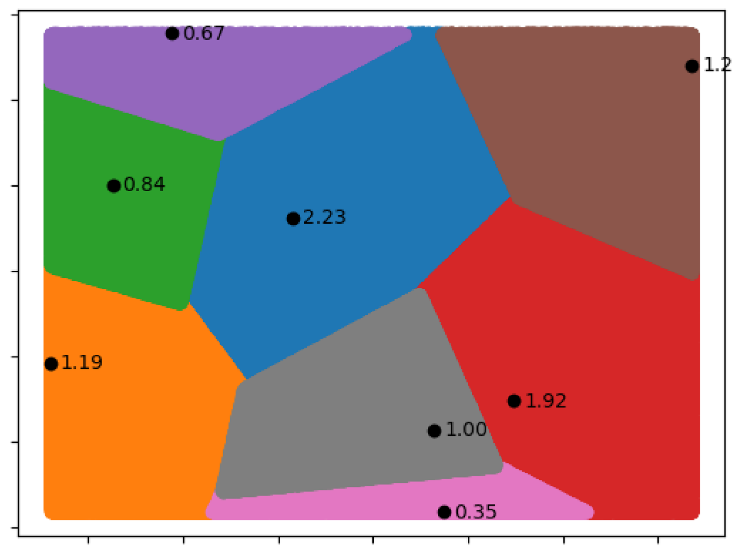


Figure 5. Irregular Graph Voronoi Regions



Voronoi regions were found using a k-d tree method to normalize the weight minimization. This calculation is unnecessary for the regularly spaced points.

Because (3) is convex, we used the Method of Least Squares [5] for each vertex to minimize our function.

We then add a virtual source and sink node with connection weights of λ to our graph to apply the min-cut max-flow algorithm.

Figure 6. 1D Min-cut Max-flow [1]

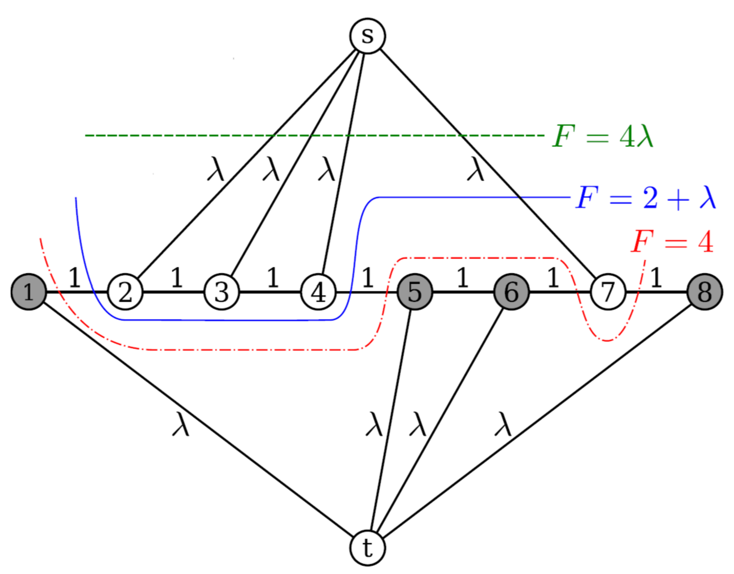
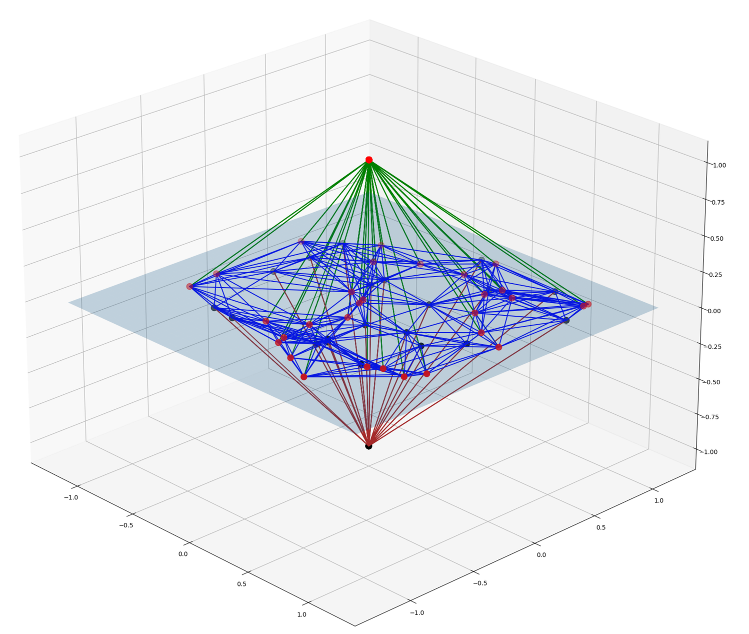


Figure 7. 2D Min-cut Max-flow



Applications

We can use this code to denoise images by employing the smooth weight function on the boundary of shapes. A max scale will return the signal exactly, limiting the mean curvature. When toggling the scale, shape characteristics can be identified.

Figure 8. $\lambda = 0.0025$

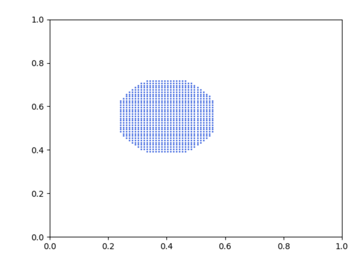


Figure 9. $\lambda = 0.0035$

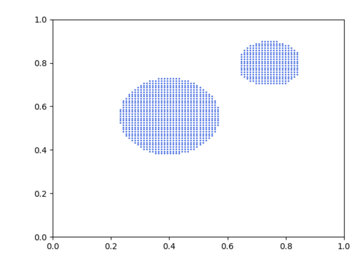
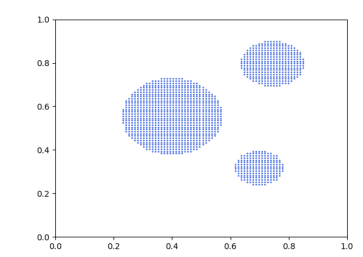
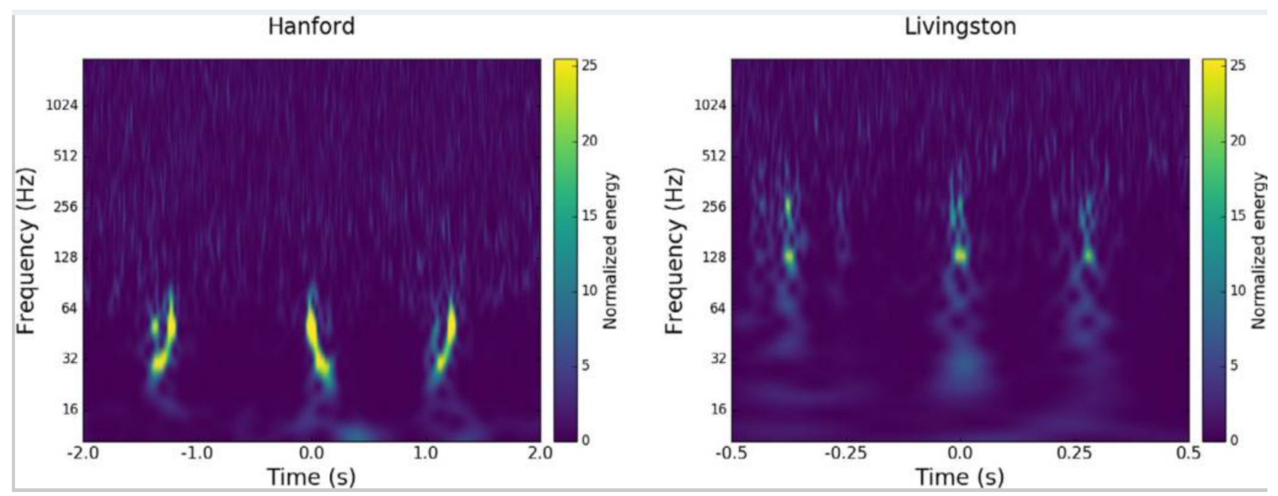


Figure 10. $\lambda = 0.0065$



Future Work

Figure 11. We are collaborating with the LIGO project to classify glitches like "Paired Doves" and "Helix" below. [6]



References and Acknowledgments

[1] Kevin Vixie, Keith Clawson, Thomas Asaki, Gary Sandine, Simon Morgan, and Brandon Price. Multiscale Flat Norm Signatures for Shapes and Images. *Applied Mathematical Sciences*, 4:667–680, 01 2010.

[2] Katrina Sabochick. Derivation progress. *unpublished*, 06 2021.

[3] Jarle Berntsen, Terje O. Espelid, and Alan Genz. An Adaptive Algorithm for the Approximate Calculation of Multiple Integrals. *ACM Trans. Math. Softw.*, 17(4):437–451, dec 1991.

[4] Sandy Auttelet, Jared Brannan, Blake Cecil, Curtis Michels, Katrina Sabochick, and Kevin Vixie. Flat norm on graphs. 2024.

[5] Least squares regression line demo. https://www.desmos.com/calculator/umjqsogp50, 2018.

[6] M Zevin, S Coughlin, S Bahaadini, E Besler, N Rohani, S Allen, M Cabero, K Crowston, A K Katsaggelos, S L Larson, T K Lee, C Lintott, T B Littenberg, A Lundgren, C Østerlund, J R Smith, L Trouille, and V Kalogera. Gravity Spy: integrating advanced LIGO detector characterization, machine learning, and citizen science. *Classical and Quantum Gravity*, 34(6):064003, February 2017.

Thank you to the donors of the Hacker, Leo Millam, and Tom Dickinson Undergraduate Research Scholarships for funding this research.

Thank you to WSU's Mathematics and Statistics Department, Office of Undergraduate Research, and Team Mentor Program for funding travel to this conference.

For any suggestions or questions email: sandra.zichterman@wsu.edu