

**LA-UR-23-20347**

Accepted Manuscript

# **Graph Metric Learning Quantifies Morphological Differences between Two Genotypes of Shoot Apical Meristem Cells in Arabidopsis**

Scott, Cory  
Mjolsness, Eric  
Oyen, Diane Adele  
Kodera, Chie  
Uyttewaal, Magalie  
Bouchez, David

Provided by the author(s) and the Los Alamos National Laboratory (2023-02-02).

**To be published in:** in silico Plants

**DOI to publisher's version:** 10.1093/insilicoplants/diad001

**Permalink to record:**

<http://permalink.lanl.gov/object/view?what=info:lanl-repo/lareport/LA-UR-23-20347>



Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

PAPER

# Graph Metric Learning Quantifies Morphological Differences between Two Genotypes of Shoot Apical Meristem Cells in *Arabidopsis*

Cory Braker Scott<sup>1,2,3\*</sup>, Eric Mjolsness<sup>2,3</sup>, Diane Oyen<sup>3</sup>, Chie Kodera<sup>4,5</sup>,  
Magalie Uyttewaal<sup>4</sup> and David Bouchez<sup>4</sup>

<sup>1</sup>Department of Mathematics and Computer Science, Colorado College, Colorado Springs, CO, 80903, USA, <sup>2</sup>Department of Computer Science, University of California Irvine, Irvine, CA, 92697, USA, <sup>3</sup>Los Alamos National Laboratory, Los Alamos, NM, 87544, USA, <sup>4</sup>Université Paris-Saclay, INRAE, AgroParisTech, Institut Jean-Pierre Bourgin (IJPB), 78000, Versailles, France. and <sup>5</sup>CryoCapCell, Inserm U1195, Université Paris Saclay, Le Kremlin-Bicêtre, France

\*Corresponding author e-mail: cbs@coloradocollege.edu

FOR PUBLISHER ONLY Received on 19 April 2022; revised on 16 December 2022; accepted on 11 January 2023; Published: 30 January 2023

## Abstract

We present a method for learning “spectrally descriptive” edge weights for graphs. We generalize a previously known distance measure on graphs (Graph Diffusion Distance), thereby allowing it to be tuned to minimize an arbitrary loss function. Because all steps involved in calculating this modified GDD are differentiable, we demonstrate that it is possible for a small neural network model to learn edge weights which minimize loss. We apply this method to discriminate between graphs constructed from shoot apical meristem images of two genotypes of *Arabidopsis thaliana* specimens: wild-type and *trm678* triple mutants with cell division phenotype. Training edge weights and kernel parameters with contrastive loss produces a learned distance metric with large margins between these graph categories. We demonstrate this by showing improved performance of a simple k-nearest-neighbors classifier on the learned distance matrix. We also demonstrate a further application of this method to biological image analysis. Once trained, we use our model to compute the distance between the biological graphs and a set of graphs output by a cell division simulator. Comparing simulated cell division graphs to biological ones allows us to identify simulation parameter regimes which characterize mutant vs. wild-type *Arabidopsis* cells. We find that *trm678* mutant cells are characterized by increased randomness of division planes and decreased ability to avoid previous vertices between cell walls.

**Key words:** Cell morphology; graph metrics; morphodynamics; neural networks; spectral graph theory.

## 1. Introduction

Plant organogenesis results in stereotypic patterning of cell shapes and cell wall networks. The *Arabidopsis* shoot apical meristem (SAM) is an organogenic, self-rejuvenating tissue that produces the primordia of aerial organs such as leaves and flowers. The morphogenetic activity of the SAM requires tight spatio-temporal control of the timing and orientation of cell divisions, as well as the direction and rate of cell growth. Cell wall networks resulting from SAM activity have been previously described by cell geometry approaches (Hamant et al., 2008; Montenegro-Johnson et al., 2019; Stamm et al., 2017). These methods can quantify local morphology and can be used to calculate tissue-scale topological statistics. However, previous graph metric approaches to quantifying SAM networks have been limited to using topological features to compare cell morphology. Here, we introduce a method for direct comparison of cell networks, using a trained distance metric that incorporates both topological and geometric information. The main scientific goal of the present work is to show that this type of distance metric, once learned, is able to a) accurately classify plant cells by type, and b) can be interpreted to generate insight into the morphological properties of different cell genotypes. Our numerical experiments demonstrate that incorporating geometric information into a learned distance metric outperforms cell type classifiers operating on topological features, as in Sahlin et al. (Sahlin et al., 2009).

Graph Diffusion Distance (GDD) is a measure of similarity between graphs originally introduced by Hammond et al. (Hammond et al., 2013) and substantially generalized and scaled up by Scott et al. (Scott and Mjolsness, 2021). This metric measures the similarity of two graphs by comparing their respective spectra (the eigenvalues of the graph Laplacian) as they evolve under graph diffusion. However, it is well-known that there exist pairs of *cospectral* graphs which are not isomorphic but have identical spectra. Furthermore, because even a small change to entries of a matrix may change its eigenvalues, another limitation of GDD is that it is sensitive to small changes in the topology of the graph (as well as small variations in edge weights). Finally, since GDD does not make use of edge or node attributes, it cannot distinguish between two different signals on the same source graph, diminishing its applicability in data science. In this work, we introduce several generalizations to GDD which resolve these issues and make it a powerful machine learning tool for datasets of graphs derived from biological microscope images.

### Contributions.

The first contribution of this paper is the development of a differentiable version of the GDD calculation. This allows us to backpropagate error through the GDD calculation to the edge weights of the graph, which in turn allows us to learn discriminative edge weights with a neural network. We apply this method to a dataset of graphs extracted from confocal microscope images of the L1 cell layer of shoot apical meristems (SAMs) of *Arabidopsis thaliana*. Our novel distance calculation method allows us to compare our real graphs with synthetic examples generated via model-derived simulation. On this basis we conclude that a specific genotype of mutant *Arabidopsis* is characterized by increased randomness in the direction of cell division and decreased ability to avoid previous vertices between cell walls.

## 2. Graph Diffusion Distance

**Fig. 1.** From left to right: a small graph, its adjacency matrix, and its graph Laplacian.

We use the definition of Graph Diffusion Distance (GDD) first given by Hammond et al. and later expanded (to cover differently-sized graphs) by Scott et al. This graph comparison metric makes use of the eigenvalues of each graph’s Laplacian matrix, which we define as follows: Let  $w_{ij}$  be the weight of the edge between the two vertices  $v_i$  and  $v_j$ . Then, the graph Laplacian is a matrix  $L = A - D$ , where  $A$  is the (weighted) adjacency matrix of the graph with  $w_{ij}$  as the  $(i, j)$ th entry.  $D$  is a diagonal matrix where the  $i$ th entry on the diagonal is given by  $\sum_k w_{ik}$ . See Figure 1 for an example of a graph and its Laplacian. Given two graphs  $G_1$  and  $G_2$ , let  $L_1$  and  $L_2$  be their respective Laplacian matrices. Furthermore, let  $L_i = U_i \Lambda_i U_i^T$  be the diagonalizations of each Laplacian, so that  $\Lambda_i$  is a diagonal matrix whose  $j$ th diagonal entry,  $\lambda_j^{(i)}$ , is the  $j$ th eigenvalue of  $L_i$ . If  $L$  is a graph Laplacian, then the matrix  $e^{tL}$  (called the *diffusion kernel* of the graph) describes heat flow between vertices of that graph: the  $(i, j)$ th entry of the diffusion kernel describes the amount of heat that has flowed from node  $i$  to node  $j$  after  $t$  time has passed. Then the graph diffusion distance between two graphs of the same size is given by comparing the eigenvalues of their diffusion kernels:

$$\begin{aligned} D(G_1, G_2) &= \sup_t \left\| e^{-tL_1} - e^{-tL_2} \right\|_F = \sup_t \left\| e^{-t\Lambda_1} - e^{-t\Lambda_2} \right\|_F \\ &= \sup_t \sqrt{\sum_{j=1}^n \left( e^{-t\lambda_j^{(1)}} - e^{-t\lambda_j^{(2)}} \right)^2}. \end{aligned} \quad (1)$$

This equation compares the eigenvalues using the *Frobenius norm*, also known as the element-wise L2 norm. The maximization over  $t$  is because we want to compare the two kernels at the most informative time: at both very small and very large values of  $t$ , all graphs look identical (since respectively either no heat has diffused, or heat has spread evenly everywhere). See Figure 2 for an example GDD calculation, evaluated at multiple values of  $t$ . The simplification in Equation 1 (which we will not detail here) relies on the rotation invariance of the Frobenius matrix norm  $\|\cdot\|_F$  and the Taylor expansion of the exponential map. It is clear that this distance measure requires the two graphs to be the same size, since otherwise this matrix difference is not defined.

The generalization to different-sized graphs given by Scott et al. could also be modified in the way we discuss in Section 3, but we do not consider this version of GDD in this paper. Given two graphs  $G_1, G_2$  of differing sizes  $n_1 < n_2$ , we can define graph

**Fig. 2.** This figure shows the result of evaluating the norm in Equation 1 with eigenvalues of two random graphs on 32 nodes, for varying values of  $t$ . This demonstrates why we take the diffusion distance to be the peak of this norm; since at early and late times the difference between the eigenvalues of the two diffusion kernels vanishes.

diffusion distance similarly to Equation 1:

$$\begin{aligned} D(G_1, G_2) &= \sup_{t>0} \inf_{\alpha>0} \inf_{P \in \mathcal{C}(P)} \left\| P e^{\frac{-t}{\alpha} L_1} - e^{-t\sqrt{\alpha} L_2} P \right\|_F \\ &= \sup_{t>0} \inf_{\alpha>0} \inf_{\tilde{P} \in \mathcal{C}(\tilde{P})} \left\| \tilde{P} e^{\frac{-t}{\alpha} \Lambda_1} - e^{-t\sqrt{\alpha} \Lambda_2} \tilde{P} \right\|_F. \end{aligned} \quad (2)$$

In Equation 2,  $\alpha$  is a time-dilation factor which dilates the passage of time in one graph with respect to the other. Dilating time in the two graphs with  $\alpha$  allows us to fairly compare graphs that have radically different sizes. As an example, diffusion on a fine square grid and a coarse square grid look very similar, but with differing timescales.  $P$  is a rectangular matrix which is optimized according to some set of constraints  $\mathcal{C}$ . In the cited paper by Scott et al., the constraint  $\mathcal{C}$  is taken to be orthogonality:  $P^T P = I$ . The reason for requiring  $P$  to be an orthogonal matrix is that optimization over this class of matrices is invariant to rotation of the two coordinate systems of each of the two graphs. Our optimization is then comparing the discrepancy between the two diffusion kernels, regardless of their individual coordinate systems.  $\tilde{P} = U_2^T P U_1$  is a change of basis from graph-space to eigenspace, allowing us to again represent the equation for varying-size GDD as a comparison between lists of eigenvalues.

### 2.1. GDD is a differentiable function of $t$ and edge weights

Once all of the eigenvalues  $\lambda_i$  and orthonormal eigenvectors  $v_i$  (of a matrix  $L$ ) are computed, we may backpropagate through the eigendecomposition as described in (Nelson, 1976) and (Andrew et al., 1993). If our edge weights  $A$  (and therefore the values in the Laplacian matrix  $L$ ) are parametrized by some parameter value  $\theta$ , and our loss function  $\mathcal{L}$  is dependent on the eigenvalues of  $L$ , then we can collect the gradient  $\frac{\partial \mathcal{L}}{\partial \theta}$  as:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_k \left( \frac{\partial \mathcal{L}}{\partial \lambda_k} \frac{\partial \lambda_k}{\partial \theta} \right) = \sum_k \left( \frac{\partial \mathcal{L}}{\partial \lambda_k} v_k^T \frac{\partial L}{\partial \theta} v_k \right), \quad (3)$$

where  $v_k$  is the  $k$ th unit-length eigenvector of  $L$ . This derivation uses the fact that (since the eigenvectors  $v_k$  are orthonormal)

$$\frac{\partial \lambda_k}{\partial \theta} = \frac{\partial}{\partial \theta} (v_k^T L v_k) = \frac{\partial v_k^T}{\partial \theta} L v_k + v_k^T \frac{\partial L}{\partial \theta} v_k + v_k^T L \frac{\partial v_k}{\partial \theta} \quad (4)$$

$$= \lambda_k \frac{\partial}{\partial \theta} (v_k^T v_k) + v_k^T \frac{\partial L}{\partial \theta} v_k = 0 + v_k^T \frac{\partial L}{\partial \theta} v_k. \quad (5)$$

If the entries of  $L$  are computed as a function of  $\theta$  using an automatic differentiation package (such as PyTorch (Paszke et al., 2019)) the gradient matrix  $\frac{\partial L}{\partial \theta}$  is already known before eigendecomposition. We note here that for any fixed value of  $t$ , all of the operations needed to compute GDD are either simple linear algebra or continuous or both. Therefore, for any loss function  $\mathcal{L}$  which takes the GDD between two graphs as input, we may optimize  $\mathcal{L}$  by backpropagation through the calculation of GDD using Equation 3. We note here that although all numerical experiments in this paper use the same-sized version of GDD (Equation 1), this backpropagation will work for the varying-sized version as well, allowing gradients to be used to adjust any of the inputs to the general GDD equation (Equation 2).

## 3. Learning Parameters for Diffusion Kernels

In this section, we describe our method for learning edge weights for Laplacian diffusion kernels, beginning with our generalization of GDD to make it trainable, and then introducing a method for learning edge weights.

### 3.1. Diff2Dist: Differentiable Graph Diffusion Distance

We make two main changes to GDD to make it capable of being tuned to specific graph data. First, we replace the real-valued optimization over  $t$  with a maximum over an explicit list of  $t$  values  $t_1, t_2, \dots, t_p$ . This removes the need for an optimization step inside the GDD calculation. We initialize  $t_i$  to be exponentially distributed in the range  $[e^{-3}, e^3]$ . The  $t$  values are updated via gradient descent during training. Second, we re-weight the Frobenius norm in the GDD calculation with a vector of weights  $\beta_j$  which is the same length as the list of eigenvalues (these weights are normalized to sum to 1). The resulting GDD calculation is then:

$$D(G_1, G_2) = \max_{t \in \{t_1, t_2, \dots, t_p\}} \sqrt{\sum_{j=1}^n \beta_j \left( e^{-t\lambda_j^{(1)}} - e^{-t\lambda_j^{(2)}} \right)^2}. \quad (6)$$

We call this version of GDD *Differentiable Graph Diffusion Distance*, or Diff2Dist. Because this distance calculation is comprised entirely of differentiable components and linear algebra, it may be explicitly included in the computation graph (e.g. in PyTorch) of a machine learning model, without needing to invoke some external optimizer to find the supremum over all  $t$ . We note here that although the  $\max()$  function is not differentiable everywhere, PyTorch has logic for backpropagating through the  $\max()$  operation.

**Fig. 3.** Left: an image of the SAM of a mutant *Arabidopsis* specimen. The original 3D confocal microscope image is here represented as a 2D skeleton. Right: a zoomed-in view of the same specimen, with separate cells false-colored and an example extracted cell neighborhood graph overlaid.

This is accomplished by masking the reverse-mode gradient to only apply to the maximum entry of a given tensor, and breaking ties with the subgradient method - see the PyTorch documentation for more details.  $t_n$  and  $\beta_j$  may be tuned by gradient descent or some other optimization algorithm to minimize a loss function which takes  $D(G_1, G_2)$  as input. Tuning the  $t_n$  values results in a list of values of  $t$  for which GDD is most informative for a given dataset, while tuning  $\beta_j$  reweights GDD to pay most attention to the eigenvalues which are most discriminative. In the experiments in Section 4.2 we demonstrate the efficacy of tuning these parameters using contrastive loss.

### 3.2. Learning Edge Weighting Functions

Here, we note that if graph edge weights are determined by some differentiable function  $f$  parametrized by parameters  $\theta$ , we may still apply all of the machinery of Sections 2.1 and 3.1. A common edge weighting function for graphs embedded in Euclidean space is the *Gaussian Distance Kernel*,  $w_{ij} = \exp\left(\frac{-d_{ij}^2}{2\sigma^2}\right)$ , where  $d_{ij}$  is the distance between nodes  $i$  and  $j$  in the embedding.  $\sigma$  is the standard deviation of the distance kernel and can be chosen *a priori* or tuned in the same way as  $\beta$  and  $t$ , using a numerical optimization procedure. However, the Gaussian distance kernel, while mathematically well-motivated for embedded graphs, is a somewhat arbitrary choice of edge weight, especially in cases where the edges of our graphs have more complicated edge labels. In cases like the data discussed in this paper, our edge labels are vector-valued, and it is therefore advantageous to replace this hand-picked edge weight with weights chosen by a general function approximator, e.g. an artificial neural network (Fukushima and Miyake, 1982), with inputs defined geometrically. As before, the parameters of this ANN can be tuned using gradients backpropagated through the GDD calculation and eigendecomposition.

## 4. Numerical Experiments

### 4.1. Data Description

#### 4.1.1. Arabidopsis Shoot Apical Meristem Dataset

The species *Arabidopsis thaliana* is of high interest in plant morphology studies, since it is a standard genetic model organism whose genome was fully sequenced in 1996, relatively early (Kaul et al., 2000). Additionally its structure makes it relatively easy to capture images of the aerial stem cell niche with active cell division: the *shoot apical meristem* (SAM). Recent work (Schaefer et al., 2017) has found that mutant *Arabidopsis* specimens with a simultaneous loss of function of genes TRM6, TRM7, and TRM8 demonstrate more variance in the placement of new cell walls during cell division. This is thought to be the result of the *trm678* triple mutants having abnormal or absent pre-prophase bands (PPBs) before cell division. The PPB is a microtubule cytoskeleton array which is hypothesized to fine-tune the placement of new cell walls before division (Schaefer et al., 2017).

The dataset used in this paper was prepared as follows:

1. Two genotypes of *Arabidopsis* (wild type and *trm678* mutants: mutants with loss of function of all three of TRM6, TRM7, and TRM8) expressing the PDF1::mCITRINE-KA1 reporter (Stanislas et al., 2018) were sown and kept in short-day conditions (8 hours of light, 16 hours of dark) for 6 weeks.
2. Plants were transferred to long-day conditions (16 hours of light, 8 hours of dark) and kept there until the inflorescence meristem (SAM) had formed. This took two weeks for wild-type plants and three weeks for *trm678* mutants.
3. The SAM of each plant was then dissected and observed with a confocal microscope, Leica SP8 upright scanning confocal microscope equipped with a water immersion objective (HC FLUOTAR L 25x/0.95 W VISIR).
4. This resulted in 3D stacks of the SAM imaged from above, collected for both types of specimens.
5. Each 3D image was converted to a 2D image showing only the cell wall of the top layer of cells in the SAM. In total we used 20 confocal microscope images (13 from wild-type plants and 7 from *trm678* mutants).
6. Each image was segmented into individual cells, by finding connected components separated by cell walls. This step was performed with the watershed algorithm, as implemented in the Python package scikit-learn (with default parameters).
7. We construct a graph from the resulting segmented image. In this graph, two cells are connected by an edge if they share a boundary and their centers of mass (approximated as the mean coordinate of the corresponding region of pixels) are closer than 100 pixels.
8. Finally, we extracted multiple subgraphs from each of these graphs. Each subgraph consists of a cell and its 63 closest neighbors. 63 was chosen so that each of the graphs in our dataset had 64 nodes total; we picked this size of graph because it was the largest power of 2 size for which we could fit reasonable batches in memory. Cell neighborhood selection was limited to the central region of each SAM image, since the primordia (secondary growths surrounding the SAM) are known to have different morphological properties. For each cell neighborhood, we produce a graph by connecting two cells if and only if their shared boundary is 30 pixels or longer (pixel size : 1 px = 0.1818  $\mu\text{m}$ ). For each edge, we save the length of this shared boundary, the angle of the edge from horizontal, and the edge length, as the three geometric attributes input to the ANN that computes graph edge weight.

We extracted 1200 cell neighborhoods in this way, resulting in a dataset of 600 graphs from each *Arabidopsis* genotype. See Figure 3 for an example of cell segmentations and extracted graphs. The dataset of labelled graphs is available in the Supplementary Material accompanying this paper.

#### 4.1.2. Replum Cell Dataset

To further validate our proposed approach, we also demonstrate our model’s ability to learn a distance function on a different cell graph dataset, also derived from *Arabidopsis* cells. This dataset consists of confocal microscope images of the replum of the gynoecium of *Arabidopsis* plants in two varieties: wild-type and mutants with loss of function in all four of the genes TRM1, TRM2, TRM3, and TRM4 (“*trm1234* mutants”). Cellular descriptions in terms of the orientation of cell divisions and directions of cell growth during *Arabidopsis* gynoecium development are still lacking; however, we took advantage of the organisation of the replum epidermis into pseudocells to compare the orientation of cell divisions and the direction of cell growth in the *trm1234* mutants compared to the wild type. We detected cell outlines with the membrane marker PDF1::TdTOMATO-29-1 (adapted from PUBQ::TdTOMATO-29-1, as in (Shapiro et al., 2015)). Images of marked cell outlines were processed and turned into graphs with an identical pipeline to the one described in the previous section, with the exception that for this dataset we had to take subgraphs of size 32 rather than 64 (since there are fewer cells per image, a 64-cell subgraph would be the entire tissue sample in some cases).

#### 4.2. Evaluation of Diff2Dist Variants

We test each of the GDD generalizations proposed, on the task of classifying wild-type vs. mutant morphological graphs. We split our dataset 85%/15% train/test; all metrics we report are calculated on the test set. We compare the following four methods:

1. Original GDD of unweighted graphs, with  $t$  chosen as the arg max of  $D$  over the values  $t_1 \dots t_k$ ;
2. Weighted norm version of Diff2Dist, using graphs with Gaussian kernel edge weights (with  $\sigma$  fixed at  $10^{-3} \cdot w_{\max}$ , where  $w_{\max}$  is the largest edge weight over all graphs in the training dataset), with  $t_1 \dots t_k$  and  $\beta_i$  tuned via backpropagation;
3. Weighted norm version of Diff2Dist using graphs with Gaussian kernel edge weights, with  $t$ ,  $\sigma$ , and  $\beta_i$  all tuned via backprop;
4. Weighted norm version Diff2Dist of graphs with general edge weights parametrized by a small neural network (see below for network specification). Input to this neural network was a vector of all three edge attributes.

For methods 2 and 3, the input to the distance kernel was the distance between nodes in the original image. All parameters were tuned using ADAMOpt (Kingma and Ba, 2014) (with default PyTorch hyperparameters and batch size 256) to minimize the *contrastive loss* function (Hadsell et al., 2006):

$$\mathcal{L}(G_i, G_j) = \frac{1}{2} \left( y_{ij} \max(0, D(G_i, G_j) - \rho_{\text{lower}})^2 + (1 - y_{ij}) \max(0, \rho_{\text{upper}} - D(G_i, G_j))^2 \right). \quad (7)$$

This loss function encourages  $G_i$  and  $G_j$  to be closer than  $\rho_{\text{lower}}$  if they have the same label, and further apart than  $\rho_{\text{upper}}$  if they differ ( $y_{ij}$  is a binary indicator of label agreement). These margins were set ( $\rho_{\text{lower}} = 0.001$ ,  $\rho_{\text{upper}} = 0.33$ ) by trial-and-error on the training set. Training took 600 epochs. For the neural network approach, edge weights were chosen as the final output of a neural network with seven layers of sizes {3, 128, 32, 32, 32, 32, 1} with sigmoid linear unit (SiLU) activations on the first six layers and no activation function on the last layer. We observed that the benefit of our approach was consistent across a variety of neural network architectures; we leave a more thorough evaluation of how network capacity influences this method for future work.

Results of these experiments are presented in Table 1. In this table, we compare the performance of a simple K-nearest-neighbor classifier trained on the distance matrix produced by each variant method of calculating the distance. Reported values are the accuracy of the classifier on the test dataset, and each KNN classifier uses the value of K that achieved the highest accuracy on the training dataset. We see that the learned distance matrix, using neural network weights, yields a classifier with 95% accuracy on the test set. This observation demonstrates that our method is able to learn a distance metric that separates the two categories of graph more effectively than the diffusion distance metric alone.

We also compare our model to a baseline classifier: a neural net trained on per-graph histogram feature vectors. Sahlin et al. (Sahlin et al., 2009) demonstrate that wild-type and mutant tissues in *Arabidopsis* are well-characterized by histograms of per-cell neighbor counts. We train a neural network (with SiLU activations and layer sizes [16,32,32,32,32,128,64,1]) to predict cell type from histograms of neighbor counts for each graph in each dataset. On the SAM dataset, this simple classifier outperforms regular GDD. However, our distance variants with learned components (approaches 2, 3, and 4) outperform this baseline classifier by a substantial margin. For the Replum dataset, all four of our approaches outperform the baseline, but the full neural-net enabled distance measure performs best (98% accuracy).

We also present distance matrices for each approach on the SAM dataset, as well as Isomap (Tenenbaum et al., 2000) embeddings of each (we used the scikit-learn (Pedregosa et al., 2011) implementation of Isomap with 15 neighbors and default hyperparameters). The distance matrices developed using the ANN approach clearly show better separation between the two categories. See Figures 4, 5, 6, and 7. In each of these figures we show the distance matrix for that graph distance method, as well as the result of embedding the distance using Isomap. In the Isomap embeddings, the points are colored by cell type, with semitransparent points representing the training dataset and opaque points representing the test dataset. The distance matrices and resulting embeddings for methods 1 and 2 do not result in a clear separation between the two categories, whereas methods 3 and 4 do. The Isomap embeddings demonstrate that methods 1 and 2 all result in overlapping clusters, since some within-category distances are higher than some inter-category distances. In contrast, methods 3 and 4 (learning weights with a neural network model) both have two distinct clusters. Furthermore, method 4 (Diff2Dist) demonstrates a dramatic improvement over method 3, showing that this version of GDD can be tuned so that the resulting graph distance minimizes an arbitrary loss function (e.g. separates classes of graphs). This shows that our neural network is learning an edge weighting function which highlights some difference between morphological

**Table 1.** Validation set accuracy for a simple K-nearest neighbors classifier for all four methods, as well as a simple classifier as baseline. The test set was the same for each of these tests. For KNN-based methods, the value reported is the highest value of accuracy over all K in the range [3..50].

Method	Accuracy % (SAM Cells)	Accuracy % (Replum Cells)
Histogram classifier	79.4	79.1
GDD only	75.6	85.1
$t$ -tuning and $\beta$ -weights	82.2	86.5
$t$ and $\sigma$ -tuning, $\beta$ -weights	90.0	87.8
ANN Parametrization	<b>97.2</b>	<b>98.6</b>

properties of the two classes of graphs. We note that these results are consistent across both of the *Arabidopsis* datasets considered in this paper.

**Fig. 4.** Top: distance matrix for morphological graphs, generated with method 1 (GDD on unweighted graphs). Bottom: Isomap embedding of this distance matrix, which ensures that points with small distance are placed near each other. Blue points represent the wild type, and orange points represent the mutant. Transparent points represent training set graphs; solid points represent those from the test set. We see that naive GDD leads to an embedding where the two categories of graph overlap, indicating that GDD by itself is not capturing the distinction between these two classes of graph.

**Fig. 5.** As in Figure 4, but for method 2 (weighted norm GDD on graphs with fixed Gaussian kernel edge weights.). This distance matrix and embedding have the same flaws as those in Figure 4.

**Fig. 6.** As in Figure 4, but for method 3 (weighted norm GDD with Gaussian kernel weights with tuned  $\sigma$ ). We see that using a Gaussian distance kernel for edge weights, with radius tuned by gradient descent, results in a graph distance metric which better separates the two categories.

**Fig. 7.** As in Figure 4, but for the ANN-determined edge weights. Replacing the arbitrary Gaussian distance kernel with weights chosen by a machine-learning model makes this approach fully general and produces a distance metric which fully separates the two graph categories. Blue points ( $\simeq$  left lobe): wild type. Orange points ( $\simeq$  right lobe): mutant.

## 5. Analysis of Simulation Parameters

The previous section demonstrates that our modified version of GDD is able to reliably find structural features which distinguish wild-type from mutant. In this section, we use the best trained model (the model with neural-network chosen weights, referred to in the previous section as “approach 4”) as a tool to interpret which morphological features contribute the most to this discrimination. We use the distance metric trained in the previous section to characterize the differences in cell growth between the two types of cells, by comparing the biological graphs to graphs generated by a simulation that incorporates hypothesized mechanisms of growth control.

We generated a population of artificial morphological graphs using the simulation code “Tissue” (Hamant et al., 2008). Tissue uses including finite element mechanical models to simulate growing collections of cells (represented as sets vertices). See the Tissue Gitlab (Jönsson) for instructions on how to use this software. For all of these experiments, we started from the *meristem.init* file which is packaged with Tissue. In a typical Tissue simulation, cell walls act like springs with a fixed spring constant, and all cells grow isotropically at a fixed linear rate. Cell division can be triggered by cell size, shape, or randomly - we configured Tissue so that cells divide when they reach a volume of 40 arbitrary units, with a small random chance  $r_{\text{div}}$  of dividing at any timestep if they are larger than 20 units (see the included configuration file for details). These numbers (40 and 20) were chosen so that the final mesh produced by each simulation had the same distribution of cell volumes as the dataset mention in Section 4.1.1.

By default, Tissue places new cell division planes along the shortest path which divides the two daughter cells into roughly equal volume, following the well known *Errera’s rule* (Besson and Dumais, 2011; Errera, 1886). The shortest path is found by enumerating each possible path and measuring their lengths. We modified this behavior so that, with probability  $r_{\text{angle}}$ , one of the less optimal paths was chosen instead. These parameters were implemented in a new division rule we contributed to the Tissue simulator (our modified version of Tissue’s source code is available in the supplementary material of this manuscript). We also varied two parameters from the original version of Tissue, representing the spring constant of each cell wall and the exclusion radius around each vertex (where new vertices are not allowed to be placed during division). A summary of the values swept over for each parameter is in Table 2. There were  $4 \times 3 \times 4 \times 6 = 288$  combinations of parameters, resulting in that many simulations. Each simulation resulted in a mesh file representing the final positions of cell vertices and cell walls after 10000 timesteps. Each

**Table 2.** Summary of input parameters and values used during comparison of simulations to biologically-derived graphs. Spring constant controls the stiffness of cell walls during the simulation. Vertex exclusion size is the proportional size of an envelop around each vertex where new vertices may not be placed during division.  $r_{\text{div}}$  is the random chance of a cell dividing at a given timestep even when it has not reached 40 units of volume.  $r_{\text{angle}}$  is the probability (given that division has occurred) that the placement of the new cell wall will be random instead of optimal.

Parameter Name	Values used
Spring Constant	0.1, 0.3, 1.0, 3.0
Vertex Exclusion Size	0.1, 0.3, 0.6
Random Division Frequency $r_{\text{div}}$	0.0, 0.00001, 0.00003, 0.0001
Random Division Direction Frequency $r_{\text{angle}}$	0.0, 0.01, 0.03, 0.1, 0.5, 1.0

mesh was converted into an image and processed into a set of graphs (one centered on each cell) exactly as described in Section 4.1.1. This yielded a large secondary dataset of synthetically constructed morphological graphs.

We used the Diff2Dist model (trained on the biological graphs only) to compute the distance between each biological graph and all of the graphs which originated from a simulation. Each biological graph can then be assigned a numerical label for each parameter, by taking the mean of the parameters for the 100 closest simulations. This gives us an estimate of which parameter values a given biological graph is most similar to (under our learned distance estimate, which is shown to separate *trm678* and wild-type graphs).

The results of this experiment can be seen in Figure 8. Comparing biologically derived graphs to synthetically-generated ones in this way allows us to see that wild-type graphs are characterized by higher vertex avoidance (upper right panel) and lower rate of random cell division direction (lower right panel), whereas *trm678* graphs are more likely to have cell divisions in random directions.

**Fig. 8.** Visualizing simulation parameters using Diff2Dist. Each plot shows one point for each morphological graph in our *Arabidopsis* dataset. Points are placed in 2D using Isomap, exactly as in Figure 7. Points are colored according to the parameter values of the nearest simulation-derived graphs, where ‘nearest’ means in the sense of our trained distance metric.

## 6. Code Repository

Code for all of the experiments in this paper is available as an Open Science Framework repository at [https://osf.io/h2fzp/?view\\_only=02bf54a68eef469baa68721430c8c77f](https://osf.io/h2fzp/?view_only=02bf54a68eef469baa68721430c8c77f)

## 7. Conclusion and Future Work

This paper presents a method to compute distance metrics between edge-labelled graphs, in such a way as to respect class labels. This approach is flexible and can be implemented entirely in PyTorch, making it possible to automatically differentiate the distance computation and thereby learn a distance metric between graphs that were previously not able to be discriminated by Graph Diffusion Distance. We demonstrate that our learned distance metric both a) enables more accurate classification of cell types, and also b) allows us to use simulated data to interrogate the trained distance model, to determine what parameters characterize different types. We also demonstrate that our learned distance metric and associated classifier outperform previous classifiers based on topological features, indicating that geometric information is necessary to distinguish cell type.

In the future we hope to apply this method to more heterogeneous graph datasets by including the varying-size version of GDD. Additionally, we aim to apply this approach to datasets representing 3D cell topologies and data from more perturbed mutant plants. Our neural network approach, as described, is not a Graph Neural Network in the sense described by prior works such as (Kipf and Welling, 2016; Bacciu et al., 2020), as there is no message-passing step. We expect message-passing layers to directly improve these results and hope to include them in a future version of differentiable GDD.

## Funding Information

This work was funded in part by Human Frontiers Science Program grant HFSP—RGP0023/2018, U.S. NIH NIDA Brain Initiative grant 1RF1DA055668-01, U.S. NIH National Institute of Aging grant R56AG059602, and a Leverhulme Trust Visiting Professor grant at Sainsbury Laboratory Cambridge University. Research presented in this article was also supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20200041ER. Finally, this work was supported by the Colorado College Department of Mathematics and Computer Science Faculty Research stipend. This work has benefited from the support of IJPB's Plant Observatory technological platforms, and from the support of Saclay Plant Sciences-SPS (ANR-17-EUR-0007).

## Acknowledgements

The authors would like to thank the reviewers. We would also like to thank Jacques Dumais and Olivier Hamant for their insightful comments on an early version of this manuscript.

Accepted Manuscript

## References

- A. L. Andrew, K.-W. E. Chu, and P. Lancaster. Derivatives of eigenvalues and eigenvectors of matrix functions. *SIAM journal on matrix analysis and applications*, 14(4):903–926, 1993.
- D. Bacciu, F. Errica, A. Micheli, and M. Podda. A gentle introduction to deep learning for graphs. *Neural Networks*, 2020.
- S. Besson and J. Dumais. Universal rule for the symmetric division of plant cells. *Proceedings of the National Academy of Sciences*, 108(15):6294–6299, 2011.
- L. Errera. On a fundamental condition of equilibrium for living cells. *CR Hebd Seances Acad Sci*, 103:822–824, 1886.
- K. Fukushima and S. Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.
- R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.
- O. Hamant, M. G. Heisler, H. Jönsson, P. Krupinski, M. Uyttewaal, P. Bokov, F. Corson, P. Sahlin, A. Boudaoud, E. M. Meyerowitz, Y. Couder, and J. Traas. Developmental patterning by mechanical signals in arabidopsis. *Science*, 322(5908):1650–1655, 2008. ISSN 0036-8075. doi: 10.1126/science.1165594. URL <https://science.sciencemag.org/content/322/5908/1650>.
- D. K. Hammond, Y. Gur, and C. R. Johnson. Graph diffusion distance: A difference measure for weighted graphs based on the graph laplacian exponential kernel. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 419–422. IEEE, 2013.
- H. Jönsson. URL <https://gitlab.com/slucu/teamHJ/tissue>.
- S. Kaul, H. L. Koo, J. Jenkins, M. Rizzo, T. Rooney, L. J. Tallon, T. Feldblyum, W. Nierman, M. I. Benito, X. Lin, et al. Analysis of the genome sequence of the flowering plant arabidopsis thaliana. *nature*, 408(6814):796–815, 2000.
- D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- T. Montenegro-Johnson, S. Strauss, M. D. Jackson, L. Walker, R. S. Smith, and G. W. Bassel. 3DCellAtlas Meristem: A Tool for the Global Cellular Annotation of shoot apical meristems. *Plant methods*, 15(1):1–9, 2019.
- R. B. Nelson. Simplified calculation of eigenvector derivatives. *AIAA journal*, 14(9):1201–1205, 1976.
- A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
- P. Sahlin, O. Hamant, and H. Jönsson. Statistical properties of cell topology and geometry in a tissue-growth model. In *International Conference on Complex Sciences*, pages 971–979. Springer, 2009.
- E. Schaefer, K. Belcram, M. Uyttewaal, Y. Duroc, M. Goussot, D. Legland, E. Laruelle, M.-L. de Tauzia-Moreau, M. Pastuglia, and D. Bouchez. The preprophase band of microtubules controls the robustness of division orientation in plants. *Science*, 356(6334):186–189, 2017.
- C. Scott and E. Mjolsness. Graph diffusion distance: Properties and efficient computation. *Plos one*, 16(4):e0249624, 2021.
- B. E. Shapiro, C. Tobin, E. Mjolsness, and E. M. Meyerowitz. Analysis of cell division patterns in the arabidopsis shoot apical meristem. *Proceedings of the National Academy of Sciences*, 112(15):4815–4820, 2015.
- P. Stamm, S. Strauss, T. D. Montenegro-Johnson, R. Smith, and G. W. Bassel. In silico methods for cell annotation, quantification of gene expression, and cell geometry at single-cell resolution using 3dcellatlas. In *Plant Hormones*, pages 99–123. Springer, 2017.
- T. Stanislas, M. P. Platre, M. Liu, L. E. Rambaud-Lavigne, Y. Jaillais, and O. Hamant. A phosphoinositide map at the shoot apical meristem in arabidopsis thaliana. *BMC biology*, 16(1):1–13, 2018.
- J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

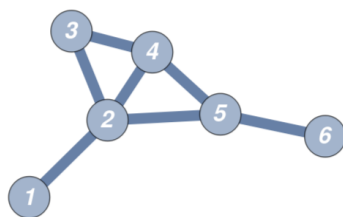
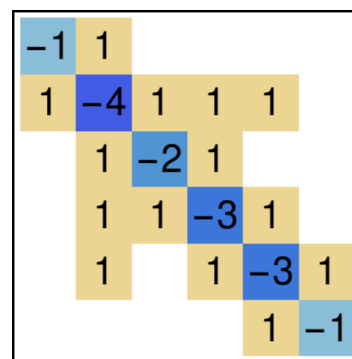
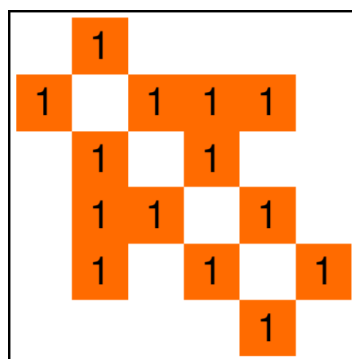


Fig. 1.



Accepted Manuscript

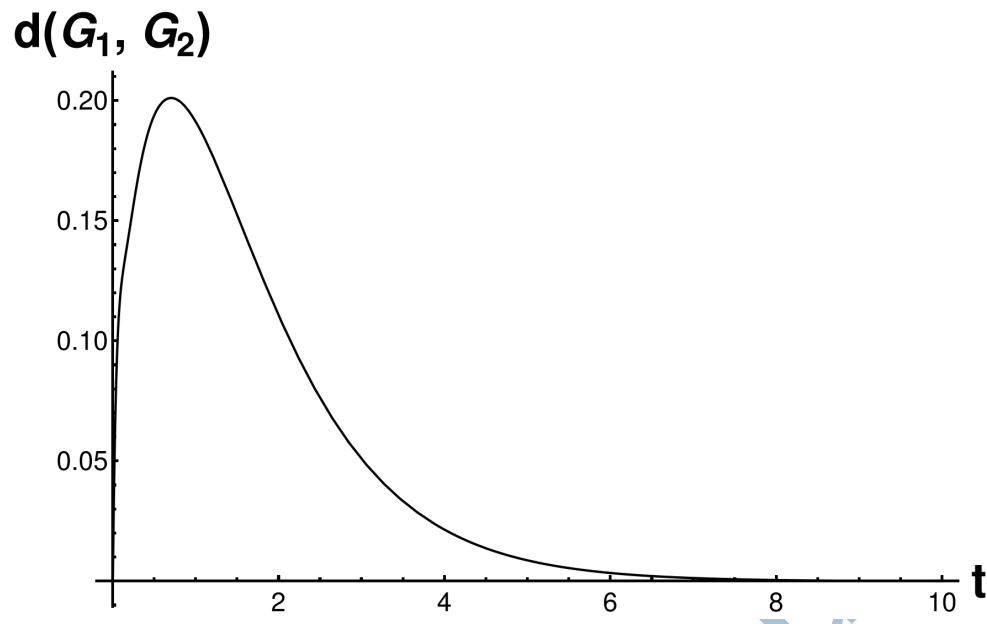


Fig. 2.

Accepted Manuscript

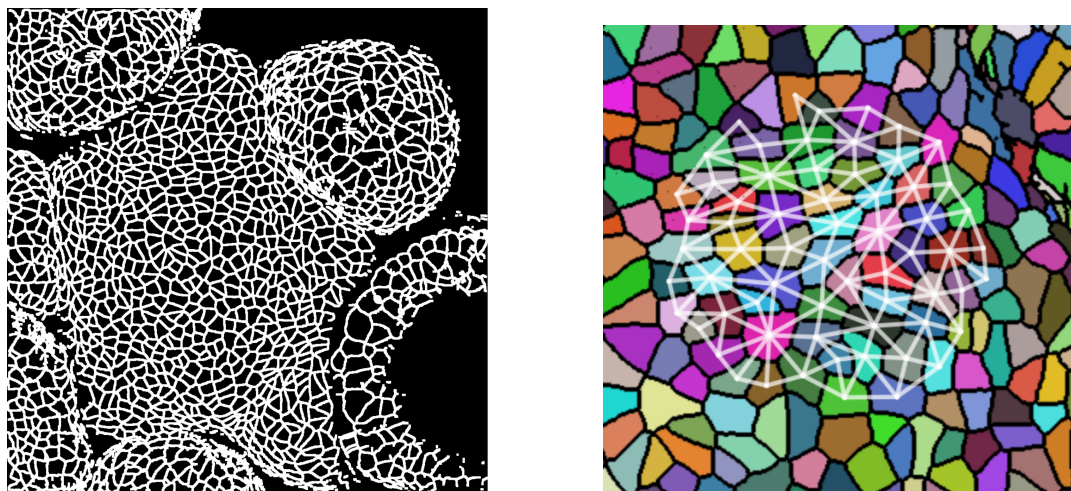


Fig. 3.

Accepted Manuscript

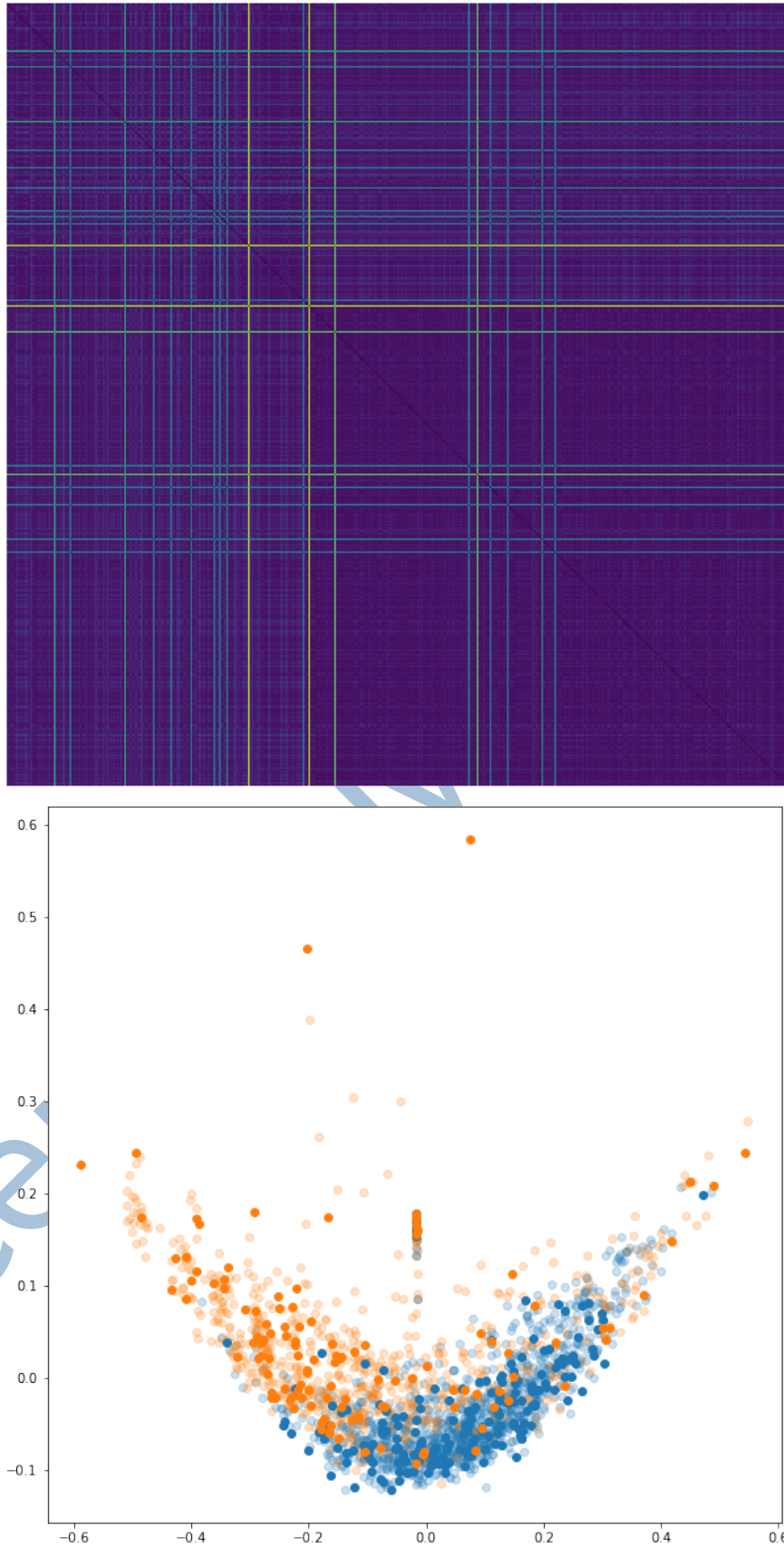


Fig. 4.

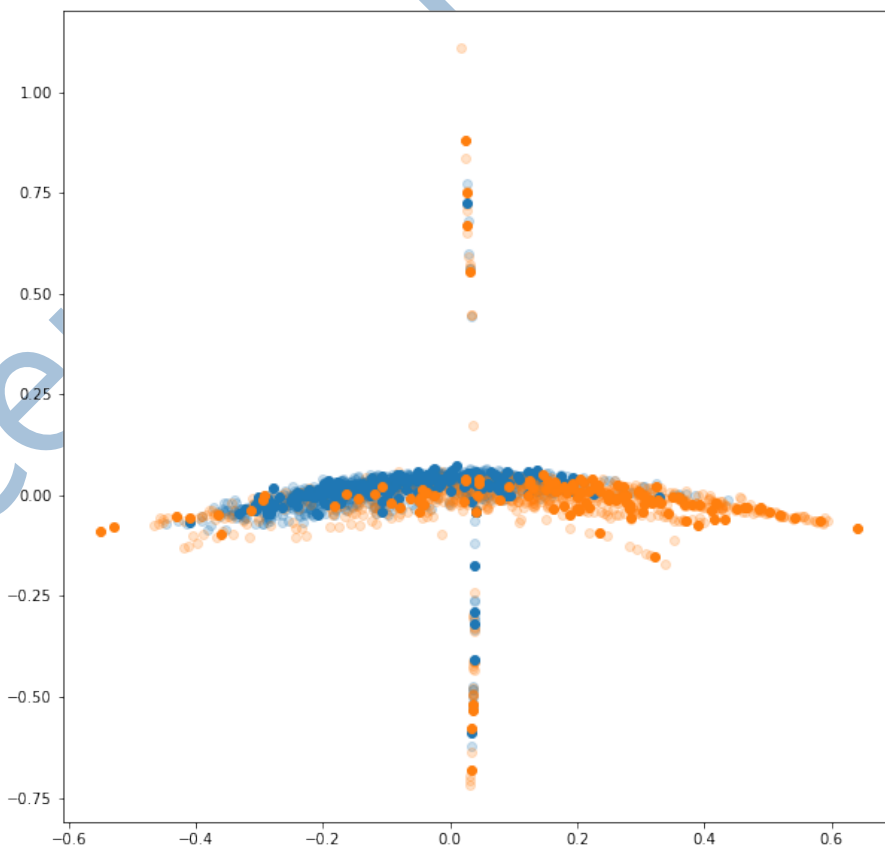
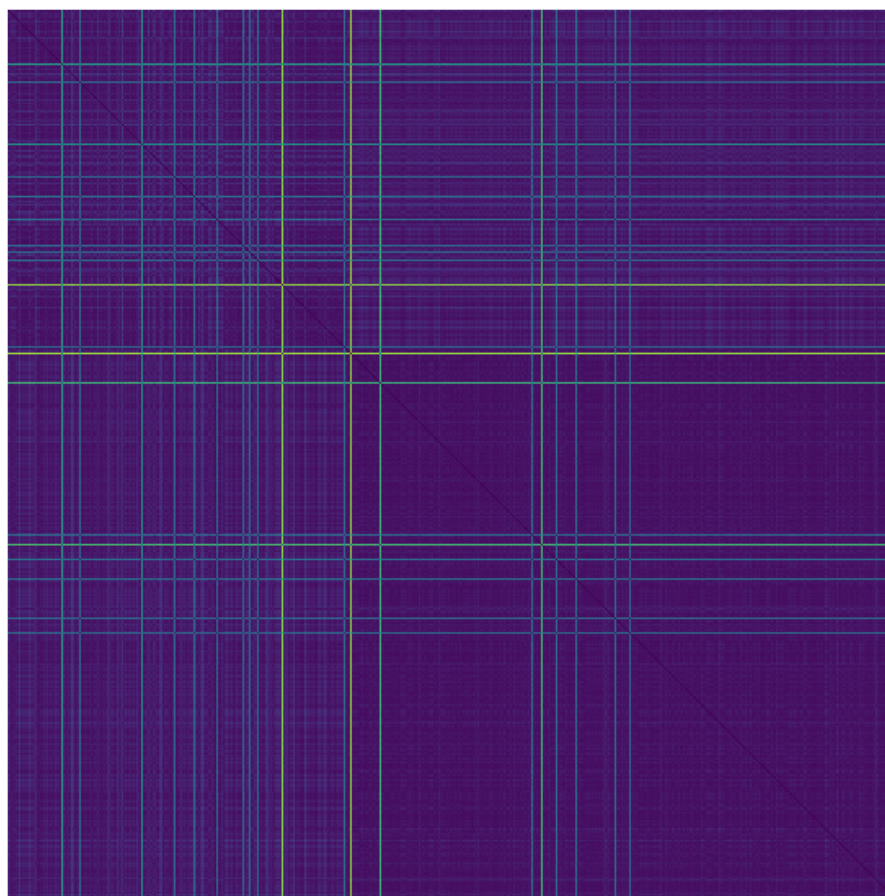


Fig. 5.

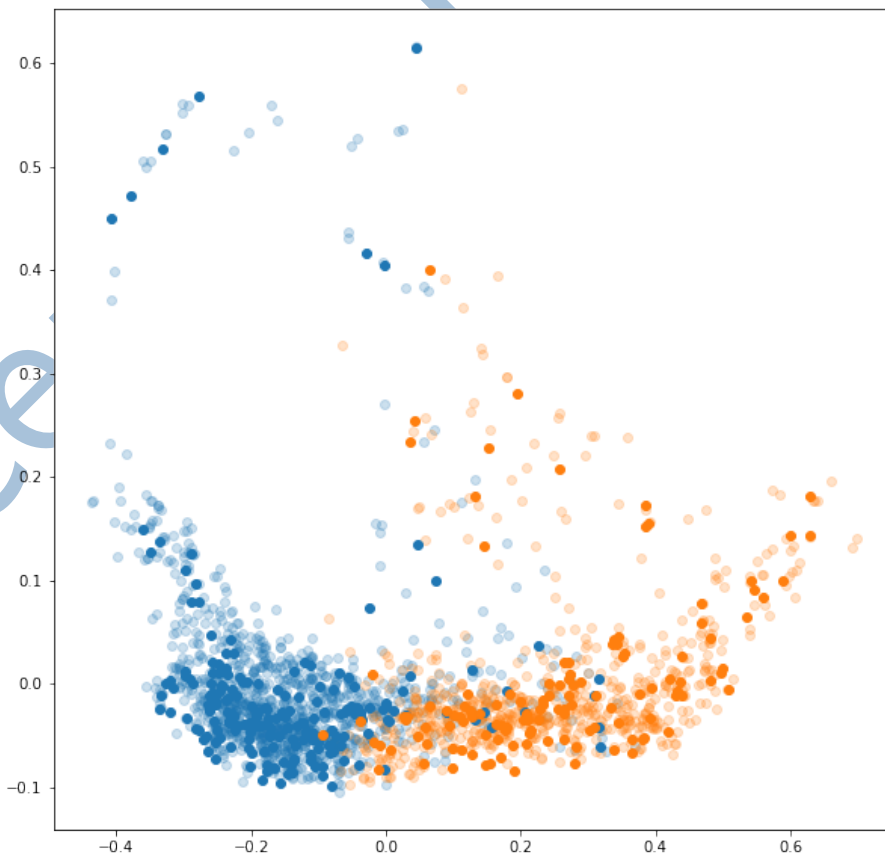
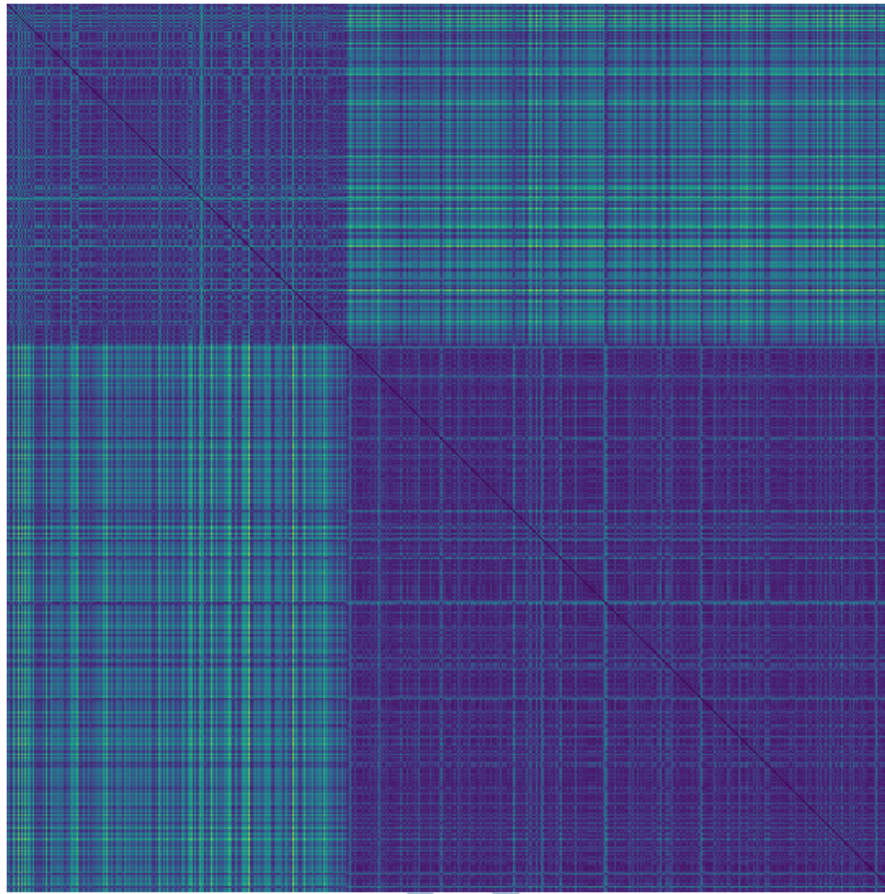


Fig. 6.

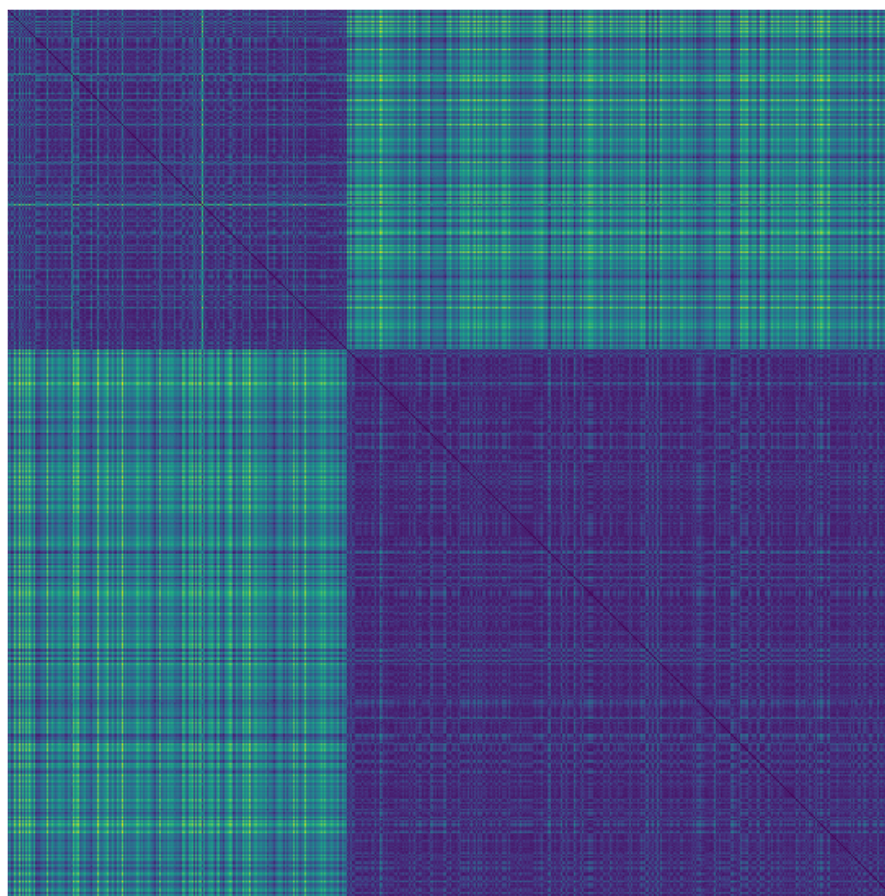


Fig. 7.

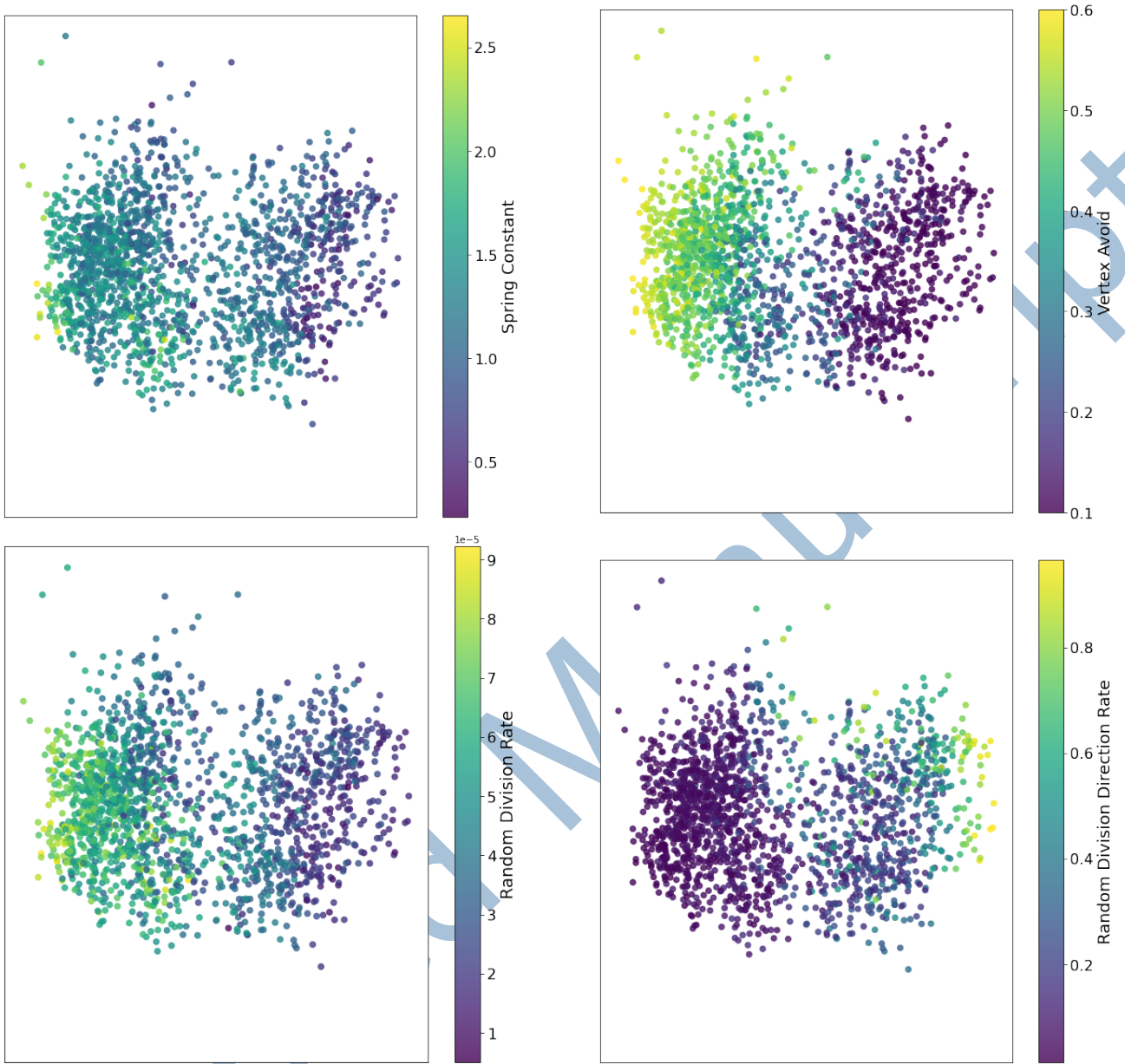


Fig. 8.