RGB image-based data analysis via discrete Morse theory and persistent homology

Chuan Du*, Christopher Szul,* Adarsh Manawa, Nima Rasekh, Rosemary Guzman, and Ruth Davidson**

***Abstract*—Understanding and comparing images for the purposes of data analysis is currently a very computationally demanding task. A group at Australian National University (ANU) recently developed open-source code that can detect fundamental topological features of a grayscale image in a computationally feasible manner. This is made possible by the fact that computers store grayscale images as cubical cellular complexes. These complexes can be studied using the techniques of discrete Morse theory. We expand the functionality of the ANU code by introducing methods and software for analyzing images encoded in red, green, and blue (RGB), because this image encoding is very popular for publicly available data. Our methods allow the extraction of key topological information from RGB images via informative persistence diagrams by introducing novel methods for transforming RGB-to-grayscale. This paradigm allows us to perform data analysis directly on RGB images representing water scarcity variability as well as crime variability. We introduce software enabling a a user to predict future image properties, towards the eventual aim of more rapid image-based data behavior prediction.**

*Corresponding Author **Co-first authors

***Keywords*—Discrete Morse Theory, Image Analysis, Data Behavior Prediction, RGB-to-Grayscale Image Conversion**

I. Introduction

Persistent homology (PH) has been identified as one of the most promising mathematical tools currently available for data analysis [1]. For example, PH has been employed for deriving effective methods in machine learning, [2], and analysis of percolating surfaces and porous materials [3]. Further, PH has been shown to be an effective method for analyzing datasets related to medicine such as the homology of brain networks [4], brain artery tree structure [5], and orthodontics [6].

Discrete Morse theory (DMT) has emerged as a tool that can be used in combination with persistent homology for data analysis [7], [8] because it has been demonstrated that (1) the Betti numbers produced in multi-dimensional persistent homology calculations are *stable* functions, meaning that small perturbations in the data set does not change the resulting Betti numbers [9] (see Section II-B, and that (2) discrete Morse theory, as it can identify the Betti numbers of cell complexes, can be used to make persistent homology computations more efficient [8], [7].

The novel contribution of this manuscript is the re-purposing of open-source code developed in the publications [10], [11] (hereafter referred to as the ANU code in this manuscript) that uses a combination of DMT for the *partitioning* and *skeletonizing* (in other words finding the underlying topological features of) images using differences in grayscale values as the distance function for DMT and PH calculations.

The data analysis methods we have developed for analyzing spatial properties of the images could lead to image-based predictive data analysis that bypass the computational requirements of machine learning, as well as lead to novel targets for which key statistical properties of images should be under study to boost traditional methods of predictive data analysis.

The methods in [10], [11] and the code released with them were developed to handle grayscale images. But publicly available image-based data is usually presented in heat-map data using Red-Green-Blue (RGB) encoding. Since RGB encoding varies by image and uses unequal weighting depending on the image under study, using an "off the shelf" RGB-to-grayscale conversion (for example, using Preview for grayscale conversion on a Macintosh) before running the DMT+PH analysis will not reliably result in an informative persistence diagram suitable for robust data analysis. We have developed an additional code repository that expands the functionality of the repository released with [10], [11] to allow for multiple ways (see Section III for details) to deal with obstacles to (1) robust data analysis and (2) predictions of data behavior based on image analysis alone.

In particular the methods we have developed with the ANU code allow for multiple paths to enable user-defined variability for conversion of RGB-to-grayscale. This empowers the data-informed user of our methods to solve the problem of variation of RGB encoding of publicly available image data. This is important as such variation can lead (see Section III-A) to insufficiently informative data analysis. Further, we exploit the speed and stability of their algorithm while allowing wide application to public image data via user-defined inputs for our code as well as the prediction of data behavior.

For example, such predictive capability has the potential to provide a user-informed image-based alternative to to machine learning methods for weather and climate assessment [12], [13] as well as computationally and statistically costly methods in the social sciences [14]. We note that computational methods for application-specific data analysis are needed experts in myriad fields. The motive of this publication is to place better computational methods for complex data in the hands of field-specific (in such fields as climate science and many social sciences) experts that can interpret meaningful features of images.

As case studies, we use our methods to analyze patterns in two applications. The first is a proof of concept that user-controlled RGB-to-grayscale conversion affects the tractability

*Co-first Authors
**Corresponding Author: redavid2@illinois.edu

of image based data analysis using water scarcity (see Section III-A). Our techniques allow publicly available heat map data from regions with highly variable water scarcity, a common problem for resource management [15] to be used as an analytical tool for such scarcity. We chose this application because it is an important resource management issue [16] where water resource experts could benefit from image-based methods for policy management, as data-gathering in this field is very difficult and time-consuming [17].

The second application of our novel RGB-to-grayscale conversion techniques is to heat-map based crime data in Halifax, Nova Scotia (see Section III-B). We identified this as a potential application because of the abundance of interest in the literature of the statistical properties of data about crime variation [18], as well as interest in fast machine learning techniques (for example) to analyze such data [19].

II. Mathematical Foundations

In this section we review the relevant mathematical definitions for our data analysis techniques and computational methods. In particular, we review the relevant defininitions underlying *discrete Morse theory* (DMT), the discrete vector fields (DVF) that DMT produces, and how these objects aid in the computation of *persistent homology* (PH).

A. Discrete Morse Theory

We refer the reader to [20] for a comprehensive guide to the historical development and concepts of DMT, and largely follow the notation in [20] for the following definitions.

Definition 1. A finite *cell complex* (V, K) is a collection of vertices V where K is a collection of subsets of V satisfying

1) $\{v \in K \mid \text{for all } v \in V\}$, and
2) if α is a cell defined by subset of the vertices of another cell $\beta \in K$, $\alpha \in K$.

We use the notation α^p to indicate that the the dimension of the cell $\alpha \in K$ is p.

Definition 2. A function $f : K \to \mathbb{R}$ is a *discrete Morse function* (DMF) if for every $\alpha^{(p)} \in K$:

(1) $|\{\beta^{(p+1)} \supsetneq \alpha \mid f(\beta) \leq f(\alpha)\}| \leq 1$, and

(2) $|\{\gamma^{(p-1)} \subsetneq \alpha \mid f(\gamma) \geq f(\alpha)\}| \leq 1$.

The use of a DMF allows for the construction of a discrete gradient vector field (a DVF) on a cell complex, which simplifies the computation of the Betti numbers of K. To explain the construction of a DVF we must introduce the notion of critical cells:

Definition 3. (1) $|\{\beta^{(p+1)} \supsetneq \alpha \mid f(\beta) \leq f(\alpha)\}| = 0$, and

(2) $|\{\gamma^{(p-1)} \subsetneq \alpha \mid f(\gamma) \geq f(\alpha)\}| = 0$

Observe that non-critical cells come in pairs, which is relevant to the next definition:

Definition 4. A *discrete vector field* (DVF) on a cell complex induced by a DMF is defined by arrows pointing from a higher-dimensional cell β^p to a lower dimensional cell α^{p-1} such that α^{p-1} is *not* assigned a higher value than β^p by the DMF.

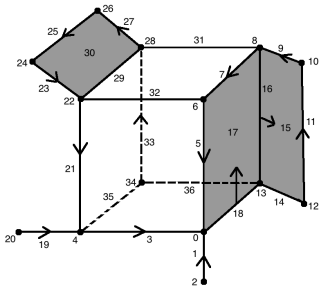

Fig. 1: A DVF on a Cubical Complex

The reason we want to find and understand the critical cells follows [10]. We refer the reader to [21] for an introduction to the topological relevance of *CW* complexes. Informally, CW complexes are complexes where cells of different dimension are glued together under sufficiently flexible topological rules.

Theorem II.1. *[10, Theorem 2.5] Suppose K is a simplicial complex with a DMF. Then K is homotopy equivalent to a CW complex with exactly one cell of dimension p for each critical simplex of dimension p.*

Thus by finding critical cells we are able to recover all the essential topological data of our cell complex. More importantly, by the theorem above, the topological data does not depend on any particular choice of a DMF. Thus our methods are well-defined after the user enables an RGB-to-grayscale encoding.

The code accompanying [10], [11] uses the natural DMF on cubical complexes determined by the grayscale values of the vertices in the image, because computers store images as cubical complexes [22], [23], [24]. Therefore the DVF of interest is also determined by the grayscale values of the vertices of the complex defining a compute graphics image. Figure 1 shows a DVF on a cubical complex with integer values.

The underlying shape is *connected*. This means that it is not broken down into several parts, or in other words, there is a path between any two parts of the complex. Topologically this corresponds to the 0th Betti number being equal to 1. Moreover, the shape has 5 holes. This is clear because out of the 8 rectangles, three of them have been filled out as gray, indicating that elementary topological collapses induced by the DVF do not change the topology of the complex. Thus the first Betti number is 5. Finally, there are no higher dimensional objects in this picture and so all higher Betti numbers are 0.

Now we show how we can recover the same result by looking a the DMF. The 0th Betti number corresponds to the critical 0-cells, which are just the vertices of the cubical complex. Figure 1 has only one critical 0-cell, namely the one labeled *0*. The first Betti number is determined by critical 1-cells, which we visualize as lines.

Looking at our DMF we see that the 1-cells with the label set

$$\{14, 29, 31, 32, 35, 36\}$$

are all critical. This does not match our claim that the first Betti number is 5. However, the rectangle labeled 30 is also critical and has in its boundary the line labeled 29. They thus form a pair that will cancel each other out. Thus we really only have 5 critical lines and no critical rectangles (equivalently 2-cells). So the DMF confirms that the first Betti number is 5 and the higher Betti numbers are 0. Figure 1, from a topological viewpoint, is actually a bouquet of 5 circles.

Notice that the DMF also gives us a concrete method to simplify the computations via the DVF. The arrows in Figure 1 represent elementary topological collapses induced by the DVF on the cubical complex induced by the values on the vertices. By drawing arrows from the higher values to the lower values, we have a concrete method on how to collapse the shape without modifying any crucial information. For example, we can collapse the point labeled *20* onto the point labeled *4* with the path *19* and still have the same topological data. Clearly we can do some of those simplifications by hand without using a DMF just by looking at the shape. However, the strength of a DMF is the ability to have a computational method to simplify the structure of a complex in a way that can be implemented algorithmically as in the ANU codebase. Yet we remind the reader that in this manuscript we present methods for moving this functionality beyond grayscale to allow for user-controlled application-specific image analysis, which affects the topological interpretation of an image.

B. Persistent Homology

Homology groups are algebraic objects that quantify topological structures of different dimensions present in a cell complex. The rank of these groups are called *Betti numbers*. By comparing the Betti numbers of two cell complexes, we can determine how similar the topology of different cell complexes are. For more details see [11, 2.2].

In certain situations a cell complex has some additional information, a *filtration*. A filtration of a cell complex K is a chain of cell complexes

$$K_0 \subset K_1 \subset K_2 \subset ... \subset K.$$

Intuitively, we think of the filtration as giving us instructions for how to build the actual cell complex K. We start with a cell complex K_0 and in each step add some piece until we reach the final stage.

For the specific case where the given cell complex has a filtration, we use *persistent homology* (PH) to compute topological invariants such as the Betti number. Concretely, PH compares the Betti numbers of different filtration levels via a *barcode diagram* (such as those shown in Figures 4, 6, and 8). In [25] it is shown that the barcode diagram gives us a accurate description of the Betti numbers of filtered spaces, by proving that a small change in the original space results in a small change in the barcode diagram. Concretely, any given change in the original space gives us a reasonable bound on the change in the barcode diagram.

PH is optimally suited to understand topological features of a cell complex with a given DMF. That is because the integers a DMF assigns to each cell gives the cell complex a filtration. In this case K_0 is the the point with the lowest DMF value and each next filtration level adds the lowest dimensional cell which has the next lowest DMF value. For example, in Figure 1 K_0 is the point labeled with value 0, K_1 consists of the two points with label set $\{0, 2\}$, and so on. Note we skipped the line labeled 1 as we have not yet included all lower dimensional cells in our filtration; filtrations are stable under multiple ordering of cells just as the choice of multiple DMFs results in DVFs that reduce a cell complex to the same toplogical information. This filtration allows us to compute the PH of a cell complex. This helps us determine how persistent a Betti number is throughout the filtration.

In [11] (and in the ANU code) the authors construct a cell complex with a DMF directly computed from the grayscale values in an image and use the mathematical paradigm described above to recover topological features from noisy data and capture key information of the grayscale picture.

III. Results and Technical Motivation

As explained in Section II-A, DMT allows for the compression of large data sets into critical shapes and points to facilitate the analysis of the homology of the persistent pairs. Informally, one can think of this compression as a method for identifying the key topological features or "key shape features" of an object, as discussed in the example of Figure 1. In this publication all the objects under study are images stored as RGB arrays that encode images as heat maps representing water scarcity variability (see Section III-A) and crime data variability (see Section III-B).

To apply the DMT and PH to analyze heat maps, pictures in .jpg form are first converted into grayscale images using three distinct methods for comparison: *Average, Luminosity, and Custom Convertio*, where color values per pixel (where pixels are vertices in a cubical complex) are transformed into grayscale values per pixel, ranging from 0 (black) to 255 (white). The ANU source code from [10], [11] generates the DMF after assigning 0-cells (pixels) with grayscale values and then choosing integer values for higher-dimesional faces int the cubical complex. Data-informative persistence diagrams are then generated to describe the birth and death time of persistence pairs. With the life-span information of topological features in the input images, patterns of pictures at a specific future time can be predicted and thus the information interpreted from the images can be gained. The results from our two case analyses follow in Sections III-A and III-B. Section III-A provides a proof of concept of our methods for enabling user-informed RGB-to-grayscale conversion, while Section III-B provides an proof of concept of how our methods can be used for predictive data analysis.

A. Application I: Analysis of Variability of Water Scarcity

We chose Kazakhstan as a proof of concept for our analysis of the impact of user-defined RGB-to-grayscale conversion because the unusually high variability of temperature and water scarcity in the unique climate of this country leads to topological representations of images with high variability across the images [26]. Further, the images available at the

Aqueduct Water Risk Atlas website from which Figure 2 was obtained (See Section VI for link information) do not even have a uniform RGB weighting on the images, so we analyze the data extracted from the image in Figure 2 in detail to make the concepts concrete.

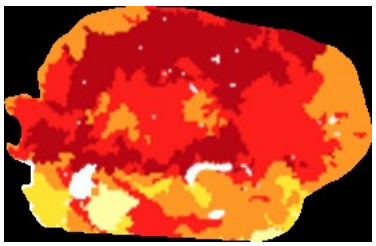

Fig. 2: Open-Source Image of Kazakhstan Seasonal Water Risk obtained from the Aqueduct Water Risk Analysis Atlas at http://bit.ly/2fEoU7Q

Starting with an RGB image of Kazakhstan water scarcity (shown in (Fig. 2), we used three methods to convert RGB images into Grayscale images. First, we used an *Average Method*, obtaining the grayscale values by calculating the average of RGB values. Then we obtain the converted grayscale image in Figure 3 and the corresponding persistence diagram, shown in Figure 4.

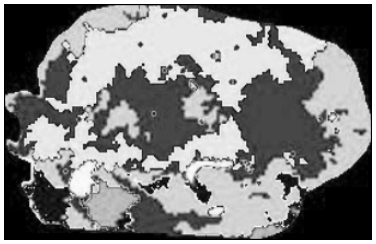

Fig. 3: Grayscale Image of Figure 2 with Average Method

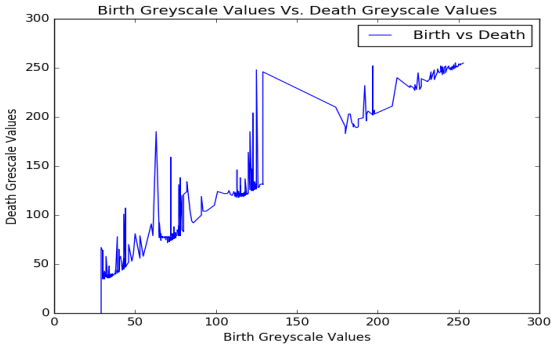

Fig. 4: Persistence Diagram with Average Method

Second, we used a *Luminosity Method*, obtaining the grayscale values by assigning different coefficients as *weights* to RGB values respectively to account for human eye's perception. In detail, since human eyes are more sensitive to color green, The value of green is weighted most heavily. The formula for luminosity is $0.21R + 0.72G + 0.07B$. Then we get the converted grayscale image in Figure 5 and the corresponding persistence diagram, shown in Figure 6.

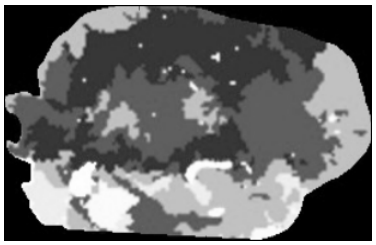

Fig. 5: Grayscale Image of 2 with Luminosity Method

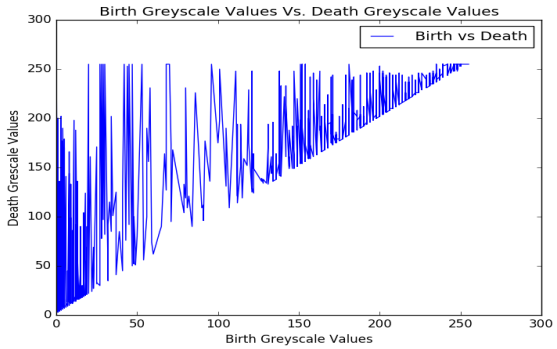

Fig. 6: Persistence Diagram with Luminosity Method

Third, we used the online converter "Convertio" (see https://convertio.co/) in conjunction with the ANU software. This is the *Custom Convertio* method, which produces a distinct converted grayscale image in Figure 7 and corresponding persistence diagram, shown in Figure 8.

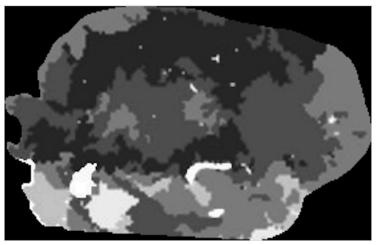

Fig. 7: Grayscale Image of Figure 2 with Convertio

B. Application II: Analysis of Crime Data in Halifax, Nova Scotia

We curated a dataset from Halifax Crime Maps, which is an animated heatmap of density of crimes in Halifax powered

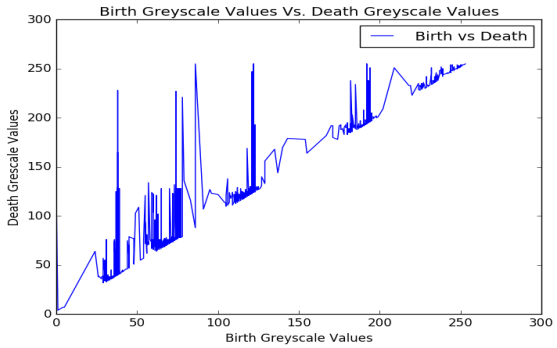

Fig. 8: Persistence Diagram with Convertio Method

by OpenDataHalifax (see http://www.crimeheatmap.ca/). New data is added weekly leading to transformations in the animation.

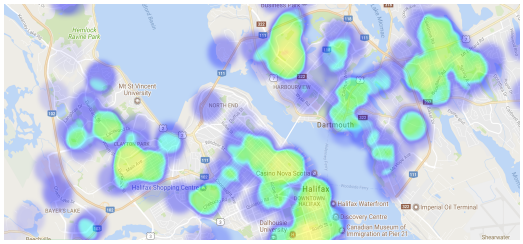

Fig. 9: Feb 1st Crime Data (Original)

To obtain time-sequence data to test predictive analysis, we collected data from OpenDataHalifax dating from February 1st, 2017 to May 24th, 2017, with 15 days between two consecutive images. We cropped the image and retained the part which has the most pattern diversity in the February 1st image, and all the other images are cropped with the same dimensions. We chose this process to retrieve relevant data for meaningful comparison in a topological sense.

After cropping the images, we changed the background to black. The crime maps included information such as local road and river structures that are not relevant to our shape-based data analysis approach. Road and river structures did not change over the time intervals we were studying. Thus we set such background information to black to exclude this data from our analysis and make topologically relevant information easier to extract.

Next, we changed the color saturation and contrast both to 100% in order to see the significant differences among the RGB weightings, so that diverse patterns could be easily distinguished. This was a necessary pre-processing step because of the arbitrary nature of RGB weighting in publicly available image data.

Finally, we transformed all these images from .png files to .pgm files by using the online file type converter Convertio, because the ANU code can only use .pgm files as inputs. See

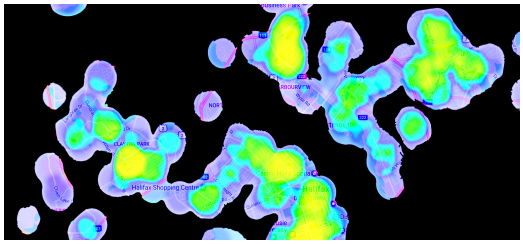

Fig. 10: Feb 1st Crime Data with Saturation and Contrast 100%

Figures 12, 13, and III-C.

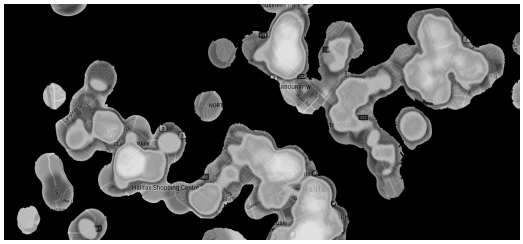

Fig. 11: Feb 1st Crime Data in Grayscale

C. Data Processing Procedure Using the ANU Code and Our Code

As two codebases are meant to be used in unison to perform similar analyses as those performed in this manuscript, we go into detail regarding the flowcharts in Figures 12 and 13 so that the interested user can reproduce our experiments and apply our methods to their own datasets.

1) The .pgm file is first converted to a NetCDF https://github.com/Unidata/netcdf-c/releases/v4.5.0 file via diamorse/util/pgmtonc.C. See [27] for the original reference.
2) Then, the NetCDF file is analyzed for persistence computations via diamorse/python/persistence.py, and outputs a .txt file with entries ”birth”, ”death”, ”dimension”, “creator xyz”, “destructor xyz” and “weight”, where *weight* refers to the surjective assignment of an RGB value to a grayscale value. Note that since we are focusing on 1-dimensional homology, the z-coordinate simply harmonizes our usage of two distinct codebases.
3) Next, the text file data are sorted and key values are extracted to .csv files for each date of crime data we use correspondingly via TextFiletoCSVFile.java (part of our codebase).
4) The .csv files are imported into Mathematica Notebook (in our codebase) analysis and prediction modeling.
5) In our analysis of the Halifax Crime Data, the Mathematica notebook PredictionModeling.nb read in the .csv files created from the ANU code and pulled out the length of the .csv from May 10th 2017. In fact, it pulls it out for all of them, but the key variable is the May

10th, 2017, as it is the starting point of the time-series analysis, as well as being a stable starting variable, as outlined in the next point:

6) PredictionModeling.nb and imports the file from the output .csv files, takes the length of all the .csv files. We chose to do this because the May 10th .csv file has the median length of all six input file sizes. Then we use the starting variable as the May 10th file, but we dropped the May 24th file as it was an outlier in that the number of data points in the .csv file was significantly smaller than the other time-series datasets fof the Halifax crime data. The result of this data analysis decision is shown in Figure 14. Essentially, it was impossible to recover significant data for our predictive modeling.

IV. Methods and Data Analysis

The images consumed in our application are available as public image data encoded in RGB values. The ANU code consumes grayscale images, so before using their code, we must pre-process the image to convert the RGB values to grayscale. While there are many common tools available to do this, we found that most conversions used linear combinations of RGB values that resulted in a loss of data regarding the key shape or topological information of the cell complex (see [28] for a review of unresolved issues identified with this problem).

The consideration of these linear combinations is important as the weighting of RGB values affects the result of the grayscale conversion in a manner that directly influences the homological features of the converted images. In particular, the use of many linear-combination-based RGB tools in our early experiments (evidenced best by the Average Method in Section III-A) led to over-weighting of dark grayscale values resulting from the use of such conversion tools erased the birth and death of homological features of RGB data that were essential to a informative topological analysis. Figure 4 shows the result of the over-weighting of dark grayscale values both in the longer lifetimes of trivial features in Figure 3 as well as the brief lifetime of of the first grayscale value at about 25 on the x-axis.

In particular, too much information is lost about the 1-dimensional homology of an image under analysis if a naïve RGB-to-grayscale conversion is used. The Luminosity Method for this conversion as displayed in the diagram in Figure 6 preserves the most information about birth and death of 1-dimensional homology out of the three methods tested on the Kazakhstan water scarcity data. We argue that shorter birth and death cycles for 1-dimensional homology tells the user of our methods more information usable for custom data analysis.

Comparing Figures 4 and 8, one sees that the Custom Covertio Method does not lose as much life-cycle information for topological features as the Average Method. The Luminosity Method picks up more birth-death cycling than the Custom Covertio method, which in turn picks up more birth-death cycling than the Average Method. Therefore we argue that the Luminosity Method is best suited to the analysis of this image.

A. Mathematica Notebook Calculations and Predictive Analysis

The Mathematica notebook PredictionModeling.nb available on our github (See Section VI) reads through all of the dates encoded in the .csv files we extracted from the Halifax Crime Data website and randomly samples which values to take out of the datasets, as they are too large to enable a full sampling with our computational resources. The number of data points is fixed by the number of data points from the May 10th sample. We chose May 10th as the basepoint proof of concept for this analysis as it had the second fewest number of points. When the notebook samples the data from the .csv files it pulls out the grayscale values from when the pattern was destroyed and created.

The prediction analysis relies on six date-based values explained in Section III-B. The notebook consumes the six date-based values after the three RBG-to-grayscale transformations are completed- i.e. the six places where significant features were born or destroyed. The choice of six dates were due to the limitations of our computational resources.

The code in PredictionModeling.nb took six values for six dates and plotted them onto an x and y plot, as visualized in Figure 14, and looks for significant data points on the plot. This is computationally expensive as the average computing time was on the order of 10 hours for each data point on an Ubuntu Linux operating system (Version 16).

The PredictionModeling.nb notebook creates a list of 35000 functions and then with functions with respect to the x value and the y value as a function of x. There is no z-value in our applications because we are only looking at 1-dimensional homology in the images we analyze. In PredictionModeling.nb the x axis represents the number of data point collected, and the y axis represents how long the pattern persisted.

For predictive analysis, PredictionModeling.nb first imports the nine .csv files for the nine dates as nine separate table variables. The notebook then finds the length for eight of the nine tables. The May 24th Table (an input data point) was dropped due to how small it was. (See the zero value on the plot in Figure 14). PredictionModeling.nb then stores the May 10th length in a separate variable called "keyLength." May 10th is the *key length* because its table was the smallest after the May 24th table was dropped.

The next process in PredictionModeling.nb is to randomly sample from each of the eight dates by sampling their index values. The number of values sampled from is equal to the integer value of the "keyLength." The value stored in keyLength is about 35,000. The index that is pulled out is used to figure out where in the table should the birth and death grayscale values be extracted from. These values are then placed into eight more tables in Mathematica. The death grayscale value is then subtracted from the birth grayscale value. This new value is stored into eight more tables, one for each of the dates where information was collected.

These eight tables are then compiled into one table: a table of tables of integer values. Note, the number eight has nothing to do with the number of data points collected and is purely a computational issue. Next, the value from the first index in

each of the tables for the first six dates is extracted and placed into a separate set. This is done until the keyLength value is met and there are no more points to analyze.

The sets are then plotted with the x-axis being the date value and the y-axis being the grayscale value for the amount of time that the pattern persisted. It does this once for each pattern because it is unnecessary to use create graphics at each step within the Mathematica computations. Graphics such as Figures 12, 13 and 14 are produced solely to understand code functionality.

After the plot such as Figure 14 has been created, the PredictionModeling.nb attempts to find a predictive function using only a single independent variable for the 35,000 sets. The independent variable, x, is extracted from the data for a date we collected: e.g. (1 = Feb 01, 2 = Feb 15, 3 = Mar 01... 6 = Apr 12) and the dependent variable, y, is the predicted length of the lifespan of the pattern for that date for the comparison of index 1.

After the 35,000 functions have been recovered, they are saved to a table. The first function is pulled out of the table and the x variable is replaced with 7 because that is the next value to come; logically this would be at a 15-day interval according to our data curation methods as outlined in Section III-B. We used six dates to train the predictive function and the seventh date is a predictive test value. PredictionModeling.nb solves for what the y-value (the lifespan of the pattern for that date for the comparison of index 1) of the seventh date is expected to be and then compares that to the actual seventh date that we have on record. PredictionModeling.nb looks for a percent error.

As we are not aware of other tools for predictive RGB data prediction methods, it is up to the consumer of this publication to decide a tolerable error percentage. Yet a reasonable recommendation would be to say if the error was was less than five percent, this would be a success: this is a high bound as PredictionModeling.nb would run this computation for each of the 35,000 sets of functions generated by the notebook. There is no reason besides computational resource access at our institution that more than a 7th date could be predicted.

B. Using Our Methods with the ANU Codebase

In Figure 12 we provide a visual representation of how to combine our code with the ANU code for the methods in Section III-A. Our codebase handles the conversion of an RGB image to a grayscale image (pictured in the upper left-hand corner of the Figure) so that we can employ the functionality of the ANU code. In Figure 13 it shown how to use the two codebases in conjunction regarding the methods explained in Section III-B.

The ANU code takes in an image in the .pgm format and then it converts that image to an image in the NETCDF3 format. The code functions in two distinct ways. The first option is that it creates a segmented black and white image and sets the pixels (one-dimensional components of the cubical complex in question) with non-zero values to one. The second option is that the code sets the values of certain pixels, whose values are above a certain specified value, to one.

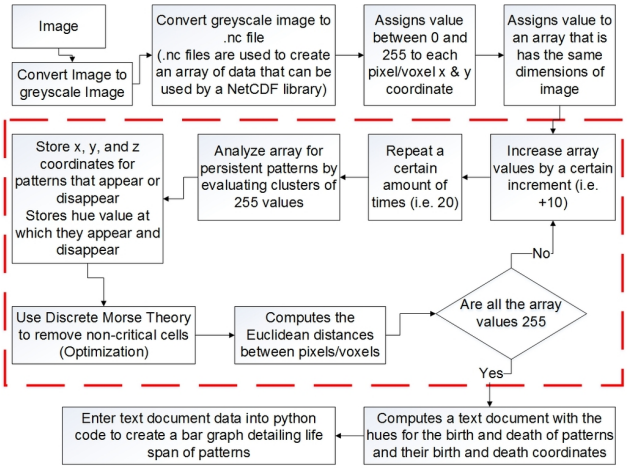

Fig. 12: Flowchart for Data Analysis Using ANU Codebase and Our Codebase

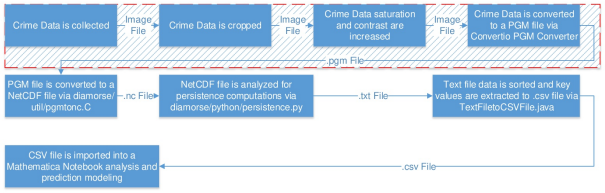

Fig. 13: Flowchart for How ANU Code and our Codebase Work Together for Predictive Analysis for Crime Data

After doing this the code then analyzes the image in the NETCDF3 format to generate data identifying the persistent pairs in the cubical complex. This is accomplished by running the picture through the persistence.py script available in the ANU codebase. This process is outlined in Figure 12.

The ANU code then writes the data in a .txt file. Using our code (See Figure 13, the .txt file is then inserted into a Java converter that changes the .txt file to a .csv file. The .csv file is then given as input to our notebook PredictionModeling.nb, which then takes random points from the dataset and generates functions to predict future trends.

C. Key Commands Used in Data Analysis from the ANU Codebase

We explain here in detail the Custom Convertio Method as outlined in Section III-A. The image.pgm file is an image file that was converted to a .pgm file through Convertio. In the predictive application of this research project, the images are taken in the form of .jpeg files from screenshots of the Hailfax Crime Data. Explanation of how these images are processed before being used in the ANU code can be found in Section III-B.

```
Input File: image.pgm
```

```
Output File: tomo_floatimage.nc
Command: diamorse ./bin/pgmtonc image.pgm
```

The command pgmtonc uses the C source code from the ANU codebase to convert the image.pgm file to the tomofloatimage.nc. The tomofloatimage.nc file is the image file in the NetCDF form. It contains the information present in the image.pgm file in a multidimensional array of integer values. This tomofloatimage.nc file is used in the persistence.py code to create the text file with information on the persistence pairs in the image.pgm file.

We ran the pgmtonc command using terminal on a Linux Ubuntu laptop (Version 14.04) after we changed the directory to the folder containing the folder "bin" with the pgmtonc.c code inside of it. The image.pgm file was in the same directory as the directory with the "bin" folder. This directory was called "diamorse" for us. Filepaths and directory names will of course need to be adjusted depending on the user of our software.

This command was ran for each of the six images pulled in from HailfaxCrimeData because we needed to have separate tomofloatimage.nc files for each of them. To identify which files were which, each .pgm file name was given the month and date of when the data was taken. This file naming automatically came over when the .nc file was created.

```
Input File: tomo_floatimage.nc
Output File: file.txt
Command: diamorse ./python/persistence.py
-t 1.0 -r tomo_floatimage.nc > file.txt
```

The text file that is output from the persistence.py source code gives the information on the persistent pairing in a tabular form. This information is the the grayscale values for when the persistent pair is created and destroyed, the persistent pair's dimension, and the xyz-coordinate locations for when the persistent pair is created and destroyed. The grayscale values for the birth and death of the persistent pairs is what is used in the scope for this project.

We ran the persistence.py command using terminal on Linux Ubuntu Version 14.04. The tomofloatimage.nc file from the above documentation, ./bin/pgmtonc, should be output to the "diamorse" folder and does not need to be moved. When the command is run, the target location is maintained as the "diamorse" folder and the target file type was a text file. We ran this command nine times for each of the tomofloatimage.nc files created from the previous command. The name of the target file was changed to the month and date to match the initial image name.

The grayscale values contained in the text files are the same information that can be used to create figures similar to Figure 4, Figure 6, and Figure 8. In those figures, the vertical bars were created by using the grayscale value for the birth of the persistent pair as the bottom point of the vertical bar and using the grayscale value for the death of the persistent pair as the top of the bar. Sections of the diagram that were not vertical when the birth and death grayscale values were the same. This is apparent in Figure 4 between the 140 and 170 on the birth grayscale Values.

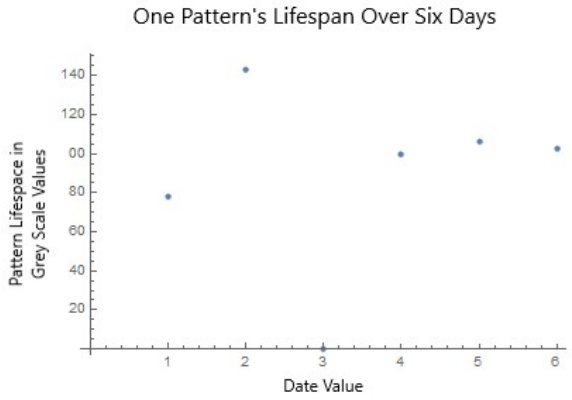

Fig. 14: Mathematica Prediction Diagram

D. Key Commands Used in Data Analysis from our Codebase

The text file from the ANU's persistence.py source code is the input for the TextFiletoCSVFile.java. The java code was written so that it would read in all of the nine text documents that we created at the same time and create nine separate .csv files with the name from the text documents.

We ran the java source code on a Windows 10 machine using the Eclipse IDE with the Java Platform, Standard Edition Development Kit version 9.

```
Input File: file.txt
Output File: file.csv
Command: ./TextFiletoCSVFile.java
```

The file.csv file contains the grayscale value for when the when the persistent pair is created and destroyed from the text file generated from the ANU codebase and it indexes each of the persistent pairs. The index starts at one and finishes at the last entry from the text file. The .csv file was designed to have the first column as the index number, the next column had the grayscale value that the persistent pair was created, and the third and last column had the grayscale value that the persistent pair was destroyed.

Creating .csv files from the text files was done because the information could be easily read into Mathematica and would make the Mathematica notebook easier to write and follow.

```
Input File: file.csv
Output File: notebook.nb
```

PredictionModeling.nb contains a plot of the first pattern over the first six dates, a list of functions for each of the data sets, an estimate for the lifelines to happen in the next date, and a percent error between the estimate and the actual grayscale values.

After point six in Figure 14 point 7 should indicate as a function of x in the y-coordinate. We solve the equation, and give that position in the y-coordinate as a function of x only. Note that in Figure 14 the date 3 is valued zero because the image had no recoverable topological patterns on that date.

V. Discussion and Future Directions

The goal of this manuscript is to contribute to a path towards predictive topological data analysis based only on RGB images (either publicly or privately available) to improve computational feasibility in an age of an overabundance of visual data. The contribution of this manuscript is the development of methods to bypass computational techniques including parallelized machine learning [29], [30], develop new methods for uncovering statistical properties of image-encoded data [3], and add potential for application-based directions for the theory in algebraic publications such as [31].

A. Relevance and Applications of Data Analysis in this Manuscript

The proof of concept of our methods established in Section III-A is that RGB-to-grayscale conversions affect the amount of toplogical features recovered from an image. If time-series data was available for the Kazakhstan water-scarcity image data it is reasonable to assume that water scarcity in this volatile region would be predictable in the near term.

The proof of concept for predictive image-based analysis was established in Section III-B because time-series data was accessible via open access collection by the authors for Halifax, Nova Scotia crime patterns over regular time intervals.

B. Future Questions and Directions

1) In the future we expect that image-based DMT data analysis could be applied to water scarcity or other climactic conditions. This could be promising for any open data source updated at more regular intervals. In an era of rapidly fluctuating climate, additional predictive tools that require minimal computational time should be useful.
2) To allow for concrete data analysis using our methods, we edited the pictures in Section III-A. In particular, we encountered issues with variation in the dimension that we determined were a byproduct of the non-informative background of the time-series crime data. Due to our editing of the the original images from the crime data, we may have introduced bias in the predictive methods, though this is unlikely, as there was no data in the background color, only user-interface based RGB graphics design. Yet for other types of data this question should be investigated.
3) Yet for other images it may be the case that a custom surjection from the RGB scale of $\{0 \dots 255\}^3$ onto the grayscale array of $\{0 \dots 255\}$ may be necessary. Hence the inclusion of the script SurjectionCode.py in our code release (See Section VI) to allow for the application-informed user to customize a mapping (a surjection) onto grayscale values from RGB values that extracts informative birth-death patterns in a barcode diagram. This script is still under development. The goal of this script is to provide a customized user-informed RGB weighting to use with the rest of our codebase, which is still under active development.
4) Running PredictionModeling.nb on a laptop using Ubuntu Verson 14.04, comptuations are not significantly refined further after taking 10, 15, or 50 points and the the PredictionModeling.nb notebook stalls before 9 hours. Errors in our computations showed that limitations of our server access at our home university prevented a complete analysis. With better computational resources we would be able to execute the computational goals of our theoretical development. It is our hope that users with large-scale computing resources (i.e. campus computing clusters instead of desktop or laptop machines) will be able to perform more detailed analyses with our codebase.

VI. Supporting Materials and Software

The data used in Section III-A can be found at Aqueduct Water Risk Atlas website:

http://www.wri.org/applications/maps/aqueduct-atlas/

The data used in Section III-B can be found at Halifax Crime's website: http://www.crimeheatmap.ca/

All novel code developed for this manuscript and used in our data analysis can be found at the github repository https://github.com/redavids/IBTCDA/tree/master. This code requires dependencies explained on the the github repository, including the repository released with the publications [10], [11]. Installation instructions are available at the github.

Acknowledgments

This project was supported by a Mathways Grant NSF DMS-1449269 to the Illinois Geometry Lab. We thank Tyler Graham and Titan Wibowo for their participation and comments during early phases of the project. R.D. was supported by NSF grant DMS-1401591.

References

[1] H. Edelsbrunner and J. Harer, "Persistent homology-a survey," *Contemporary Mathematics*, vol. 453, pp. 257–282, 2008.

[2] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt, "A stable multi-scale kernel for topological machine learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4741–4748.

[3] V. Robins, M. Saadatfar, O. Delgado-Friedrichs, and A. P. Sheppard, "Percolating length scales from topological persistence analysis of micro-CT images of porous materials," *Water Resources Research*, vol. 52, no. 1, pp. 315–329, 2016.

[4] H. Lee, H. Kang, M. K. Chung, B.-N. Kim, and D. S. Lee, "Persistent brain network homology from the perspective of dendrogram," *IEEE transactions on medical imaging*, vol. 31, no. 12, pp. 2267–2277, 2012.

[5] P. Bendich, J. Marron, E. Miller, A. Pieloch, and S. Skwerer, "Persistent homology analysis of brain artery trees," *The annals of applied statistics*, vol. 10, no. 1, p. 198, 2016.

[6] J. Gamble and G. Heo, "Exploring uses of persistent homology for statistical analysis of landmark-based shape data," *Journal of Multivariate Analysis*, vol. 101, no. 9, pp. 2184–2199, 2010.

[7] M. K. Chung, P. Bubenik, and P. T. Kim, "Persistence diagrams of cortical surface data," in *International Conference on Information Processing in Medical Imaging*. Springer, 2009, pp. 386–397.

[8] K. Mischaikow and V. Nanda, "Morse theory for filtrations and efficient computation of persistent homology," *Discrete & Computational Geometry*, vol. 50, no. 2, pp. 330–353, 2013.

[9] A. Cerri, B. D. Fabio, M. Ferri, P. Frosini, and C. Landi, "Betti numbers in multidimensional persistent homology are stable functions," *Mathematical Methods in the Applied Sciences*, vol. 36, no. 12, pp. 1543–1557, 2013.

[10] V. Robins, P. J. Wood, and A. P. Sheppard, "Theory and algorithms for constructing discrete Morse complexes from grayscale digital images," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1646–1658, 2011.

[11] O. Delgado Friedrichs, V. Robins, and A. Sheppard, "Skeletonization and partitioning of digital images using discrete Morse theory," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 654–666, 2015.

[12] C. Priestley and R. Taylor, "On the assessment of surface heat flux and evaporation using large-scale parameters," *Monthly weather review*, vol. 100, no. 2, pp. 81–92, 1972.

[13] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Advances in neural information processing systems*, 2015, pp. 802–810.

[14] J. Gill, *Bayesian methods: A social and behavioral sciences approach*. CRC press, 2014, vol. 20.

[15] T. Oki and S. Kanae, "Global hydrological cycles and world water resources," *Science*, vol. 313, no. 5790, pp. 1068–1072, 2006.

[16] V. Smakhtin, C. Revenga, and P. Döll, "A pilot global assessment of environmental water requirements and scarcity," *Water International*, vol. 29, no. 3, pp. 307–317, 2004.

[17] O. Petit, "Paradise lost? The difficulties in defining and monitoring Integrated Water Resources Management indicators," *Current opinion in environmental sustainability*, vol. 21, pp. 58–64, 2016.

[18] T. Nakaya and K. Yano, "Visualising Crime Clusters in a Space-time Cube: An Exploratory Data-analysis Approach Using Space-time Kernel Density Estimation and Scan Statistics," *Transactions in GIS*, vol. 14, no. 3, pp. 223–239, 2010.

[19] M. H. de Boer, H. Bouma, M. C. Kruithof, F. B. ter Haar, N. M. Fischer, L. K. Hagendoorn, B. Joosten, and S. Raaijmakers, "Automatic analysis of online image data for law enforcement agencies by concept detection and instance search," in *Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies*, vol. 10441. International Society for Optics and Photonics, 2017, p. 104410H.

[20] R. Forman, "A users guide to discrete Morse theory," *Sém. Lothar. Combin*, vol. 48, p. 35pp, 2002.

[21] J. Milnor, "On spaces having the homotopy type of a CW-complex," *Transactions of the American Mathematical Society*, vol. 90, no. 2, pp. 272–280, 1959.

[22] R. Gonzalez-Diaz, M.-J. Jimenez, and B. Medrano, "Encoding specific 3D polyhedral complexes using 3D binary images," in *International Conference on Discrete Geometry for Computer Imagery*. Springer, 2016, pp. 268–281.

[23] J. F. Hughes, *Computer graphics: principles and practice*. Pearson Education, 2014.

[24] V. A. Kovalevsky, "Finite topology as applied to image analysis," *Computer vision, graphics, and image processing*, vol. 46, no. 2, pp. 141–161, 1989.

[25] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer, "Stability of persistence diagrams," *Discrete & Computational Geometry*, vol. 37, no. 1, pp. 103–120, 2007.

[26] V. Yapiyev, Z. Sagintayev, A. Verhoef, A. Kassymbekova, M. Baigaliyeva, D. Zhumabayev, D. Malgazhdar, D. Abudanash, N. Ongdas, and S. Jumassultanova, "The changing water cycle: Burabay National Nature Park, Northern Kazakhstan," *Wiley Interdisciplinary Reviews: Water*, 2017.

[27] R. Rew and G. Davis, "NetCDF: an interface for scientific data access," *IEEE computer graphics and applications*, vol. 10, no. 4, pp. 76–82, 1990.

[28] C. Kanan and G. W. Cottrell, "Color-to-grayscale: does the method matter in image recognition?" *PloS one*, vol. 7, no. 1, p. e29740, 2012.

[29] I. Youkana, J. Cousty, R. Saouli, and M. Akil, "Parallelization strategy for elementary morphological operators on graphs: distance-based algorithms and implementation on multicore shared-memory architecture," *Journal of Mathematical Imaging and Vision*, pp. 1–25, 2017.

[30] R. Reina-Molina, D. Díaz-Pernil, P. Real, and A. Berciano, "Effective homology of k-D digital objects (partially) calculated in parallel," *Pattern Recognition Letters*, vol. 83, pp. 59–66, 2016.

[31] M. Allili, T. Kaczynski, and C. Landi, "Reducing complexes in multidimensional persistent homology theory," *Journal of Symbolic Computation*, vol. 78, pp. 61–75, 2017.