# Speeding up Risk Analyses of U.S. Flood Insurance Loss Data Using the Diffusion Map

Chia Ying Lee*, Rafał Wójcik†, Charlie Wusuo Liu‡, and Jayanta Guin§

***Abstract*—In the insurance industry, catastrophe risk analysis using catalogs of catastrophic events is a major component for quantifying financial risks of an insurance portfolios. To ensure an accurate quantification of risk, particularly for rare, strong catastrophic events, large sizes of catalogs are simulated and used for computing loss estimates location-by-location and event-by-event, but this is computationally intensive. In this paper, we propose to speed up the risk computation by taking a data analytic approach to compress the catalog—specifically, using dimension reduction and clustering. To address the nonlinear geometry of the loss data from the U.S. Flood model, we used a nonlinear dimension reduction technique, the diffusion map. Combined with clustering, we show that it yields accurate catalog compression and produces a realistic representation of hydrometeorological patterns over the entire country. Finally, we discuss how clustering results must be refined to ensure fidelity in retaining the most important catastrophic events, and how in real life, a risk manager can utilize our results to make informed risk management decisions.**

***Index Terms*—Diffusion map, nonlinear dimension reduction, spectral clustering, catastrophe modeling, insurance risk analytics.**

## I. Introduction

THE purpose of catastrophe modeling (known as CAT modelling in the industry) is to anticipate the chances and severity of catastrophic events from earthquakes and hurricanes to terrorism and crop failure, so companies or governments can appropriately prepare for their financial impact.

CAT models provide a robust, structured approach for estimating a wide range of possible future scenario losses from catastrophes, along with their associated probabilities. Loss estimates produced from CAT models can be deterministic for a specific event (e.g., Hurricane Katrina, a magnitude 8.0 earthquake in San Francisco) or probabilistic from an ensemble of hypothetical events [1]. The latter approach uses Monte–Carlo techniques and physical models to simulate large *catalogs* of events. For example historical data on the frequency, location, and intensity of past hurricanes are modeled and used to predict 10,000 catalog years of potential hurricane experience. Each of the 10,000 years should be thought of as a potential realization from a distribution which characterizes the probability of what could happen in the year 2016, for example, instead of simulations or predictions of hurricane activity from now until the year 12016 [2].

Manuscript received August 2, 2016; revised August 4, 2016.

C. Y. Lee, C. Liu are with the Financial and Uncertainty Modeling Group, AIR Worldwide, Boston, MA 02116 USA.

R. Wójcik, is the Manager and Principal Scientist of the Financial and Uncertainty Modeling Group, AIR Worldwide, Boston, MA 02116 USA.

J. Guin is the Executive Vice President and Chief Research Officer of Research at AIR Worldwide, Boston, MA 02116 USA.

e-mail: *clee, †rwojcik, ‡wliu, §jguin @air-worldwide.com

To pass from catalog to financial risk, the risk analysis aggregates event losses over the locations or properties in a particular portfolio, noting that losses are typically modeled by random variables characterized by loss distributions. Then the event losses are aggregated within each catalog year to obtain an aggregate annual loss (AL) distribution for each year. Finally, empirical samples from the overall portfolio loss distribution can be obtained from the mixture of AL distributions. These empirical samples are used to construct the *exceedance probability (EP) curve* [3], which is equivalent to estimating the survival function of the portfolio loss distribution, or 1 minus its cumulative distribution function. The EP curve is the key tool used by insurers to estimate their probabilities of experiencing various levels of loss. In addition, two important risk statistics of the portfolio loss are the Average (Aggregate) Annual Loss (AAL), which measures the expected AL, and the Tail Value at Risk (or $p\%$-TVaR), which measures the expected AL conditional on observing the upper $p\%$ tail of the portfolio loss distribution [2], [4].

Our paper is focused on the risk analysis for the U.S. flood catalog. This catalog relies on complex, physically based probabilistic flood model for the U.S. [5], [6], [7], and requires significant computational cost to estimate portfolio losses for each flood event. This is compounded by the large number of events in the catalog. Therefore, it is desirable to compress the size of the catalog to reduce computational time. Here, we take a clustering approach to catalog compression. This idea stems from the fact that events in the catalog can be split into two groups: strong, infrequent events generating substantial losses and weak, frequent events generating small losses. So, we aim to identify clusters of similar weak events in such a way that the risk analysis from the clusters provide a good approximation for that of the full catalog. At the same time we retain all the original strong events. Inevitably, catalog compression will incur errors in the portfolio's EP curve, AAL and TVaR estimates so it is crucial to find the right patterns in the loss data to minimize these errors.

### A. Basic data set: loss matrix

The flood model for the U.S. simulates on-floodplain riverine flooding for a river network of 1.4 million miles, including all streams with a minimum drainage area of 3.9 square miles, with 335,000 drainage catchments. Off-floodplain flooding is simulated only for areas away from floodplains [5], [7]. Each simulated flood event is characterized by physical model parameters, such as peak flow, peak runoff and catchments affected, from which the losses of exposed property are calculated. Because the extent of

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

losses depend not only on the model's physical parameters but also on the geomorphology of the catchments, large scale weather patterns etc., the relationship between the physical parameters and the losses is very complex. For this reason, we take a loss-based approach to catalog compression, in which industry loss data (the expected total on- and off-floodplain ground-up losses that have been estimated for all insurable property industry-wide), is assumed to be a suitable surrogate data set for risk analysis. The industry loss data implicitly reflects the physical parameters of events in the catalog, the geomorphology of property locations and the exposure information of the properties.

Our basic data set is the loss matrix $L$ comprising of industry losses in each catchment for each event,

$$L_{ij} = \text{Industry loss for event } i \text{ in catchment } j \qquad (1)$$

The dimensionality of $L$ is huge, containing 685,477 events covering 335,000 catchments, but it is also sparse, because the majority of flood events affects only a small proportion of catchments. In practice, loss data is not always available nationwide or at the spatial resolution of catchments. In some data sets, the losses are aggregated to a coarser spatial resolution of zipcodes or counties. The loss data available to us are for

- 6186 catchments in the Northeastern U.S.
- 29911 zipcodes in the entire U.S.
- 3101 counties in the entire U.S.

Because the industry loss data does not contain distributional information, in this paper the portfolio loss distribution is the distribution of the means of the AL distributions. Here, the set of insurable properties within a given zipcode $z$ is treated as a type of portfolio. Aggregating $L$ by catalog years produces $N_y = 10,000$ empirical samples for the zipcode loss distribution,

$$\left\{ AL_*(y,z) := \sum_{\{i:\text{Event } i \in \text{ Year } y\}} L_{iz} \right\}_{y=1}^{N_y}.$$

The EP curve for that zipcode is estimated by sorting the empirical samples in descending order and plotting them against the corresponding probability. The $p$%-TVaR is estimated by averaging the top $p$% of the empirical samples.

### *B. Catalog compression as clustering and multiobjective optimzation*

The clustering approach to the catalog compression seeks to find a *clustering solution* which partitions weak events into disjoint subsets (clusters). Each cluster of events is then represented by a *reference event* (medoid) which is an existing event within that cluster. The losses incurred by an event are approximated by that of its cluster's reference event. This means that only the losses incurred by the reference event need to be computed in the risk analysis, thereby decreasing the time spent for risk computation.

Standard clustering algorithms such as $k$-means or hierarchical clustering [8] are designed to minimize the difference between the losses of an event and its reference event. Our main objective of catalog compression is to maintain accuracy of the EP curve and the risk statistics by minimizing errors in the 1%-TVaR and AAL for zipcode losses:

- Average error in 1%-TVaR

$$\mathcal{F}_{TVaR}(c) = \frac{1}{N_z} \sum_{z=1}^{N_z} \left| TVaR_c(z) - TVaR_*(z) \right| \qquad (2)$$

  where $TVaR_c(z)$ is the 1%-TVaR for zipcode $z$ computed under the clustering solution $c$, and $TVaR_*(z)$ is that computed from the full catalog.
- Average error in AAL

$$\mathcal{F}_{AAL}(c) = \frac{1}{N_z} \sum_{z=1}^{N_z} \left| \frac{1}{N_y} \sum_{y=1}^{N_y} \left( AL_c(y,z) - AL_*(y,z) \right) \right| \qquad (3)$$

  where $AL_c(y,z)$ is annual loss of the $z$-th zipcode in the $y$-th catalog year, under $c$.

In addition to the above objectives we also aim to minimize the ratio:

$$\mathcal{F}_{red}(c) = \frac{\sum_{e_{\text{ref}}} \#(\text{counties affected by reference event } e_{\text{ref}})}{\sum_e \#(\text{counties affected by event } e)} \qquad (4)$$

which counts the reduction in the number of affected counties for which portfolio losses need be computed. The county compression rate $1 - \mathcal{F}_{red}$ is closely related to the *event* compression rate, $1 - \frac{\#\text{reference events}}{\#\text{events}}$. Determining the event compression rate is equivalent to determining the number of clusters as in [28]. The two compression rates are not interchangeable because of the variability in spatial extent of the clustered events: large clusters often comprise of localized low-loss events, while widespread high-loss events tend to become singleton clusters. Because the actual savings in loss computation time depend on the number of portfolio locations for which losses need be computed and the efficiency with which the loss computation is implemented in software, the county compression rate is a preferred surrogate for gain in computing speed.

All three criteria $\mathcal{F}_{TVaR}, \mathcal{F}_{AAL}$ and $\mathcal{F}_{red}$ are nonlinear functions of the data and are not equivalent to the objectives of standard clustering algorithms. The sole application of the latter may not yield an optimal solution so both sets of objectives should be considered as in [22] [23] and [24], [25].

Apart from the above dichotomy, (2), (3) and (4) conflict with each other—no single clustering solution minimizes them simultaneously. Thus, we cast the catalog compression as Multiobjective Optimization Problem (MOP) in [21] to compare different clustering solutions. This formalism seeks to find the *Pareto front*, which is the collection of *Pareto optimal* solutions for which no such solution is better than another by all objectives simultaneously. (See Figure 7 for a visualization of a Pareto front). The catalog compression problem formulated as MOP reads:

$$\min_{c \in \mathcal{C}} \left( \mathcal{F}_{TVaR}(c), \mathcal{F}_{AAL}(c), \mathcal{F}_{red}(c) \right) \qquad \text{(MOP)}$$

where $\mathcal{C}$ is the set of all possible event clustering solutions. The computational cost of the MOP can be reduced by both using more efficient clustering algorithms and pre- or post-processing refinements to finetune clustering solutions as proposed [27]. For catalog compression, the refinement is targeted towards improving the accuracy of estimating mean and tail statistics from the EP curve, by ensuring

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

that error-prone events become singleton clusters—clusters comprising of a single event—thereby eliminating errors due to approximation by a reference event.

### C. The role of nonlinear dimension reduction in event clustering

The fundamental ingredient of any clustering technique is the specification of a metric to quantify similarity between data points. Because of the ease at which Euclidean distances can be computed, some of the most efficient clustering algorithms assume Euclidean metric. In high dimensional applications, however, the choice of the metric is non-obvious and to large extent heuristic. Additionally, due to the curse of dimensionality [9], the concept of proximity, distance or nearest neighbor may not be meaningful. Examples of counterintuitive behavior of the Minkovski norm and its influence on the performance of k-means clustering are given in [10]. To tackle the curse of dimensionality, we aim to find an embedding of the high dimensional loss matrix $L \in R^{N\times p}$ into a low-dimensional space $R^{N\times q}$, $q \ll p$ equipped with an appropriate metric (see Section III).

The well-known linear dimension reduction technique, *principal component analysis* (PCA), achieves this embedding by seeking the best low-rank approximation to identify the best low-dimensional linear subspace that represents the directions of greatest correlations in the data[8].The linearity of the PCA, however, limits its usefulness to situations when the data conform to a Gaussian assumption—an assumption which our loss data does not satisfy. In fact, because of the sparsity of the data, the majority of events incur no loss at most zipcodes. Almost all the events lie on a nonsmooth, nonlinear manifold. In this case PCA will not give a meaningful low-dimensional representation. To tackle this problem, many recent techniques, including locally linear embedding [11], semidefinite embedding [12], Isomap [13], Laplacian eigenmaps [14] and the diffusion map [15], [16], [17], [18], have been proposed. One in particular, the diffusion map (DM), adopts the formalism of diffusion processes on a manifold in order to define a new distance (called the *diffusion distance*). (e.g., [19]). The DM embeds the data into a new coordinate system that preserves the diffusion distance such that the embedding represents the diffusion distance within the first few eigendimensions. This makes the DM particularly attractive to apply to the event clustering problem: it provides a geometry-aware distance that can be used in conjunction with Euclidean-based clustering algorithms. For example, [17] showed a rigorous justification for $k$-means clustering of the diffusion coordinates. In Section III we show that another clustering algorithm, the Growing Neural Gas (GNG) [34] combined with a graph community finding method [33], is particularly suited to our application. In Section II we show computationally fast implementation of DM.

The paper is organized as follows. We first describe the foundation and implementation of the DM in Section II, then give an example of its application to the spectral clustering of catchments and counties into Flood Regions, in Section III. For catalog compression, Section IV-A details the main event clustering procedure, while Section IV-B describes the pre-/post-processing refinements. We further explain the nature of the conflicting TVaR and AAL objectives in Section IV-C. Finally, in Section V, we show that the results of the overall catalog compression methodology yields good solutions, and demonstrate the role of Pareto optimal solutions in making risk management decisions.

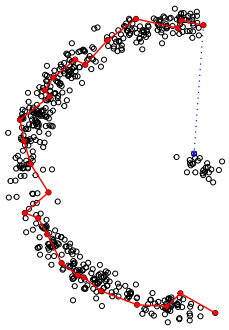

Fig. 1. Intuitively, the diffusion distance reflects the connectivity by short hops (characterized by high probability transitions) between the two ends of the arc (solid lines), in contrast to a lack of connectivity to the points in the center (dotted line). As the diffusion time $t$ increases, the ends of the arc grow closer, in diffusion distance, relative to their diffusion distance to the center data points.

## II. Diffusion map

The DM constructs a nonlinear transformation of high-dimensional data into the *diffusion space*, through a spectral decomposition of the graph Laplacian, defined by a random walk on the graph of data points that respects the connectivity and topological structure of the data manifold. Such decomposition provides an efficient representation of the data in terms of the *diffusion coordinates*, the leading few of which are used to define a low dimensional embedding of the data. In what follows, we give the theoretical framework and discuss computationally fast implementation of the DM.

### A. Theory

The input to the DM is a symmetric weight matrix $W$, where $W_{ij} > 0$ represents similarity between data points $x_i, x_j$. Then, the algorithm constructs a Markov random walk on the weighted graph $G = (X, W)$, with transition matrix $P$ given by

$$P_{ij} = \frac{W_{ij}}{d_i}, \tag{5}$$

where $d_i = \sum_k W_{ik}$ is the degree of vertex $x_i$. Under the positivity condition of $W_{ij}$ and if $G$ is connected, the random walk converges to a unique stationary distribution,

$$\lim_{t\to\infty} P_{ij}^{(t)} = \frac{d_j}{\sum_k d_k} := \phi_j^*. \tag{6}$$

where $P^{(t)} = P^t$ is the $t$-step transition matrix. The random walk favors transitions between similar points, so the stationary distribution concentrates in regions of high data density. The *diffusion distance* $D^{(t)}$, at a *diffusion time* $t$, reflects the connectivity by short and highly probable paths between two data points $x_i, x_j$ (see Fig. 1). This gives rise to its probabilistic definition, which can be re-expressed in a more convenient form in terms of the eigendecomposition

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

**Algorithm 1:** DM algorithm.

**Input :** Weight matrix $W$ (derived from data $X \in \mathbb{R}^{N\times m}$), maximum embedding dimension pmax.

$d \leftarrow$ `rowSums(`$W$`)` `// vertex degrees`
$\phi^* \leftarrow d/$`sum(`$d$`)` `// stationary distribution`
$\Delta \leftarrow$ `diag(`$d^{-1/2}$`)`.
$\widetilde{W} \leftarrow \Delta * W * \Delta$ `// symmetric graph Laplacian`
$(\Lambda, \widetilde{V}) \leftarrow$ `eigendecomposition(`$\widetilde{W}$, pmax`)`
$(t, p) \leftarrow$`getTimeDimension(`$\Lambda$`)` `// (see text)`
$\psi \leftarrow \Delta * \widetilde{V}[:, 1\!:\!p]$ `// right eigenvector`
$\psi \leftarrow \psi *$`diag((t(`$\phi^*$`)*`$\psi$`)^{-1/2})` `// normalize`
$\Psi \leftarrow \psi *$`diag(`$\Lambda[1\!:\!p]$`^t)` `// diffusion coordinates`

**Output:** Diffusion coordinates $\Psi \in \mathbb{R}^{N\times p}$.

**Algorithm 2:** Fast randomized eigendecomposition of symmetric matrices, with power iteration [29, Algorithms 4.4 and 5.3].

**Input :** Symmetric matrix $A \in \mathbb{R}^{n\times n}$, desired number of eigendimensions $p$, and number of power iterations $q$.

$\Omega \leftarrow$ `rnorm(`$n, 2p$`);` `// random Gaussian matrix`
**for** $i \leftarrow 0$ **to** $2q$ **do**
  $(Q, R) \leftarrow$`qr(`$A * \Omega$`);` `// QR decomposition`
  $\Omega \leftarrow Q$;
**end**
$B \leftarrow$ `transpose(`$\Omega$`)` $*A*\Omega$; `// a small matrix`
$(\Lambda, V) \leftarrow$ `eigen(`$B$`);` `// eigendecomposition`
$\Lambda \leftarrow \Lambda$[1:p]; $V \leftarrow V$[,1:p];

**Output:** Eigenvalues $\Lambda \in \mathbb{R}^p$, eigenvectors $V \in \mathbb{R}^{n\times p}$.

of the transition matrix:

$$(D_{ij}^{(t)})^2 := \sum_k \frac{(P_{ik}^{(t)} - P_{jk}^{(t)})^2}{\phi_k^*} \equiv \sum_{k=0}^{N-1} \lambda_k^{2t}(\psi_{ik} - \psi_{jk})^2 \tag{7}$$

where $\lambda_k, \psi_{\cdot,k}$ are the eigenvalues and corresponding right eigenvector of $P$ (normalized w.r.t. the weight $\phi^*$). Note that $\lambda_0 = 1$ and $\psi_{\cdot,0} \equiv 1$ is a constant vector. If the eigenvalues decay sufficiently fast, the summation in Eq. 7 can be approximated by the first $p$ summands, chosen up to error tolerance. By defining the *diffusion map*—the transformation of the data points into the $p$-dimensional diffusion space—as

$$\Psi^{(t)} : x_i \in \mathbb{R}^N \mapsto (\lambda_1^t \psi_{i1}, \ldots, \lambda_p^t \psi_{ip})^T \in \mathbb{R}^p, \tag{8}$$

it follows that the diffusion distance is approximated by the Euclidean distance of the diffusion coordinates,

$$(D_{ij}^{(t)})^2 \approx \|\Psi^{(t)}(x_i) - \Psi^{(t)}(x_j)\|^2. \tag{9}$$

### B. Algorithm and Implementation

Given an input similarity measure $\mathrm{Sim}(\cdot,\cdot)$, a common choice for the weight matrix $W$ is to use a Gaussian kernel,

$$W_{ij} = e^{-(1-\mathrm{Sim}(x_i,x_j))^2/2\kappa^2}.$$

The kernel width, $\kappa$, can be automatically set as the median distance of each data point's $k$-th nearest neighbor [18], with $k$ typically 1% of the data size. The diffusion time $t$ and embedding dimension $p$ should meet an error tolerance for the approximation of $D^{(t)}$; e.g. to satisfy $p = \max\{j : |\lambda_j^t| > \delta|\lambda_1^t|\}$ for a given tolerance $\delta$ [17]. In practice, we choose $\delta = 0.1$ and fix a maximum dimension $p_{max}$ (to limit the number of eigenvectors that must be computed), and then choose $t$ large enough to meet the error tolerance.

Algorithm 1 shows the basic steps of DM implementation. It uses the symmetric graph Laplacian in lieu of the transition matrix, so that eigendecomposition for symmetric matrices can be used. This step is often the computational bottleneck with large data, but we the fast randomized SVD technique of [29] can tackle this problem. Our implementation of the DM algorithm in R/Rcpp is as follows:

- *Fast linear algebra using fast randomized SVD algorithm [29].* Based on a random projection and iterative orthogonalization procedure, the fast randomized SVD algorithm reduces the estimation of the leading singular values to the eigendecomposition of a small matrix. The random projection is justified by the Johnson-Linderstrauss lemma [30]. We used a version for eigendecomposition of symmetric matrices (shown in Algorithm 2) with complexity $\mathcal{O}((q+1)(N^2p + Np^2))$.
- *Memory efficient implementation requiring negligible additional memory allocation.* Memory efficiency is achieved by utilizing packed storage of symmetric matrices [31] and designing modification-in-place subroutines.
- *Optimized BLAS libraries for symmetric packed matrix multiplication.* In Algorithm 2, the multiplication of the (large) input matrix with the (small) random Gaussian matrix is performed by the BLAS routine `dspmv` [31] for symmetric packed matrix-vector multiplication. Note that if the input matrix is too large to fit into RAM, a single-pass matrix multiplication scheme can be adopted to allow entry-wise streaming of the input matrix.
- *Higher eigendecomposition accuracy possible with additional computation.* The option to improve the approximation accuracy utilizes a power iteration feature of the fast randomized eigendecomposition algorithm. The additional cost is due to multiple repetitions of matrix multiplication. Empirically, $q = 2$ is sufficient for accurate results.
- *Speed up with parallelization.* Many of the computations are highly parallelizable.

## III. An Example: Spectral Clustering of Flood Regions using DM

In this section we illustrate the efficacy of the DM using an example of clustering catchments or counties, instead of events, into regions of similar flood activity. To cluster catchments, the loss matrix is transposed, $L'$, so that each data point $z \in \mathbb{R}^N$ represents a catchment's losses from the $N$ events in the catalog. The input to the DM can be any similarity measure computed on $L'$ (e.g. Jaccard and Yule similarities, Simple Matching Coefficient, correlation coefficient, Euclidean distance, etc.), so that it defines a corresponding diffusion distance. Thereafter, a clustering algorithm is applied to the Euclidean distance on the diffusion coordinates, noting that conceptually, it is the closeness with respect to the diffusion distance that the clusters obey. This is illustrated in Figure 2.

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

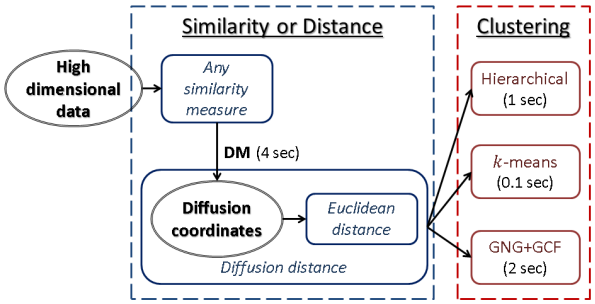


Fig. 2. Workflow for spectral clustering using DM. In parentheses are ballpark computational times taken on the Northeastern U.S. catchment loss data set, computed on a 3.4 GHz Intel i7-4770 processor. Our DM implementation used 8 core parallelization; the other algorithms used a single core. GNG+GCF used the `gmum.r` package [32].

As mentioned before, clustering algorithms that require an Euclidean distance assumption (such as $k$-means, neural gas), as well as those that accept an arbitrary similarity measure (such as hierarchical clustering, agnes) can now be used. Another clustering method that is particularly effective for this application is the Growing Neural Gas algorithm combined with a graph community finding algorithm (GNG+GCF):

- The Growing Neural Gas (GNG) algorithm combines key ideas from competitive Hebbian learning and the Neural Gas algorithm to build a graph of nodes that represent the centroids of clusters, defined through the Voronoi tesselation. An edge between two nodes reflects the density of data points connecting the nodes.
- Graph community finding (GCF) algorithms attempt to find communities of nodes that are highly connected within each community but poorly connected with other communities. One way communities are found is through minimizing the modularity score function, which measures the fraction of edges falling within a community:

$$\frac{\sum_{n,n'} E_{n,n'} \delta_{c_n,c_{n'}}}{\sum_{n,n'} E_{n,n'}}$$

where edge $E_{n,n'} = 1$ if nodes $n, n'$ are connected, and 0 otherwise; $c_n$ is the community to which $n$ belongs. GCF is achieved by the fast greedy modularity optimization algorithm [35], which is of almost linear complexity, $\mathcal{O}(n \log^2 n)$. A drawback of GCF is that the exact number of communities is not known a priori.

To obtain effective clustering using GNG+GCF, the GNG was trained using more nodes than the number of desired regions, so that each node represents a "micro-cluster" of catchments. Then, a flood region is formed by a community of nodes found by the GCF algorithm, which represents the corresponding collection of catchments.

For catchment loss data in Northeastern U.S., we performed the spectral clustering using DM as shown in Figure 2. In our experiments, the Jaccard similarity yielded the best results. It is defined as

$$\text{Sim}_{Jac}(z, z') := \frac{n_{11}}{n_{11} + n_{10} + n_{01}},$$

where the entries of the data points $z, z'$ are first converted to binary values (with 1 and 0 indicating positive and zero loss, respectively), and then $n_{ij} = |\{k : z_k = i \text{ and } z'_k = j\}|$.

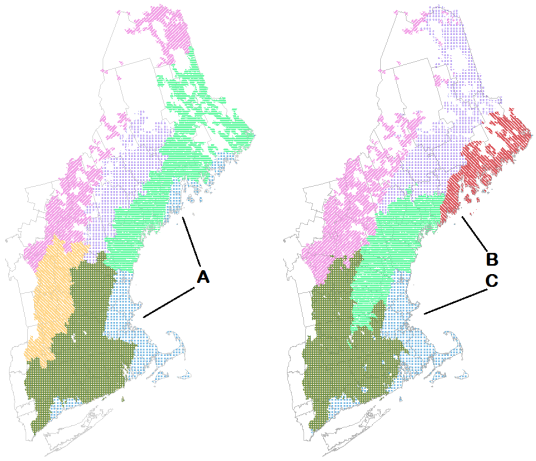


Fig. 3. Map of Northeastern U.S. catchments clustered into six regions, using the benchmark method (LEFT) and GNG+GCF clustering on the diffusion distance (RIGHT). Catchments not affected by any event are not included.

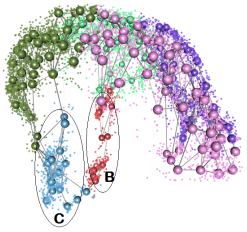


Fig. 4. 3D visualization of the first 3 diffusion coordinates, for diffusion time $t = 5$ and dimension $p = 4$. The GNG+GCF algorithm was used with 200 nodes. The labeled groups of nodes correspond to the Regions B and C in Figure 3 RIGHT

The Jaccard similarity computes overlap ratios and is well-suited to image and text processing applications [36]. Here, two catchments are Jaccard-similar if the events affecting both catchments form a large proportion of those affecting at least one of the catchments; that is, if they have a similar propensity for flooding.

We looked for heuristic qualities of how well hidden relationships in the data were captured, such as the ability to reproduce large scale weather patterns and maintain geographical connectivity of the regions. For comparison, hierarchical clustering based on the correlation coefficient $\text{Corr}(z, z') = Var(z, z')/\sqrt{Var(z)Var(z')}$, a method used in feature cluster analysis, was used as a benchmark.

- The methods shown in Figure 2 and the benchmark method produced regions with fairly cohesive boundaries, and broadly captured weather patterns moving in a northeastern direction.
- Hierarchical and $k$-means clustering on the diffusion distance, as well as the benchmark method, were unable to separate the two coastal regions that are subjected to the same weather patterns but are geographically disconnected. (Region A in Figure 3 LEFT.)
- Only GNG+GCF clustering on the diffusion distance successfully distinguished the two geographically separated coastal regions (Regions B, C in Figure 3 RIGHT)
- The DM provides a visualization of the catchments with similar weather patterns, particularly the 'closeness' of the separate coastal regions (Figure 4). This explains why it is easy for a clustering algorithm to cluster

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

those two regions together. Nonetheless, GNG+GCF distinguishes those two regions because the corresponding groups of nodes are sparsely interconnected despite being close in diffusion distance.

For county loss data for the entire U.S., we again performed spectral clustering using DM with Jaccard similarity, and with diffusion time $t = 1$ and 100 diffusion coordinate dimensions. Figure 5 compares the $k$-means and GNG+GCF clustering on the diffusion distance. The GNG+GCF algorithm identified 20 flood regions, and the number of clusters for $k$-means was fixed at 20 for comparison. Both clustering procedures pick up the general weather patterns in the Tornado Belt and Eastern U.S. where meteorology dominates the flooding patterns, but produce distinctly differing clusters in the drier Central and Western U.S. Once again, GNG+GCF successfully distinguishes Florida from the central part of the U.S., something that $k$-means fails to achieve. Interestingly, the boundaries of the clusters found by GNG+GCF sometimes coincide with the boundaries of the USGS Hydrological Flood Regions [37], which are defined by the geomorphology of river basins. This is significant because it indicates the ability of our clustering method to identify distinct types of patterns—meteorological and hydrological—from the loss data.

## IV. Catalog compression

### A. Using DM in Event Clustering

It is intractable to work directly with the entire pairwise-distance matrix required for the DM, due to the large size of the catalog. Therefore, we partition the catalog into subcatalogs to make it computationally tractable and scalable. This is equivalent to approximating the similarity matrix by a sparse matrix possessing a block diagonal structure, where events in different subcatalogs are assumed to be completely dissimilar. Effectively, we only allow events within the same subcatalog to be clustered together. The subcatalog size should ideally be within the constraints of computer resources. The idea of pre-partitioning the data set is not new, an example being the heuristic CF-tree building procedure of BIRCH [38]. Instead of using the heuristics, however, we applied partitioning schemes that exploit our application-specific knowledge:

- *Partition by total loss:* Compute each event's total loss and define disjoint loss intervals. Each subcatalog comprises of the events whose total event loss lies in a given loss interval.
- *Partition by Flood Regions:* Each subcatalog comprises of the events affecting a given combination of flood regions (regions obtained from the clustering procedure in Section III).

Partitioning by flood regions is tractable only when there are a small number of flood regions, such as in the Northeastern U.S. catchment loss data. For the entire U.S., partitioning by loss is preferred because of its computational tractability. The loss interval partitions were $\$10^{5.5}$, $10^{6}$, $10^{6.25}$, $10^{6.5}, \ldots, 10^{10}$ dollars, and the subcatalog sizes ranged from about 1000 to 50000 events.

To compute the DM, we used the Jaccard similarity (between events), and for each subcatalog computed the diffusion coordinates up to a maximum of $p = 50$ dimensions. Then, $k$-means was applied to the diffusion coordinates to cluster the events within each subcatalog. To promote better accuracy for high-loss events, a different compression rate was used for each subcatalog (Table I): higher loss events are afforded a lower compression rate, and vice versa. Finally, because $k$-means produces the cluster centroids in diffusion space, the last step is to determine the reference event for each cluster. Even for a single objective problem, finding the globally optimal set of reference events is an expensive combinatorial problem. An effective alternative is to adopt a local strategy: select, cluster-by-cluster, the reference event whose total loss is closest to the average total loss for events in the cluster.

TABLE I
Compression rates (ballpark) depending on the subcatalog's loss amount(M=million, B=billion).

| Loss ($) | <1M | 1–10M | ... | 1–10B | >10B |
|---|---|---|---|---|---|
| Compression | 99.5% | 99% | ... | 75% | None |

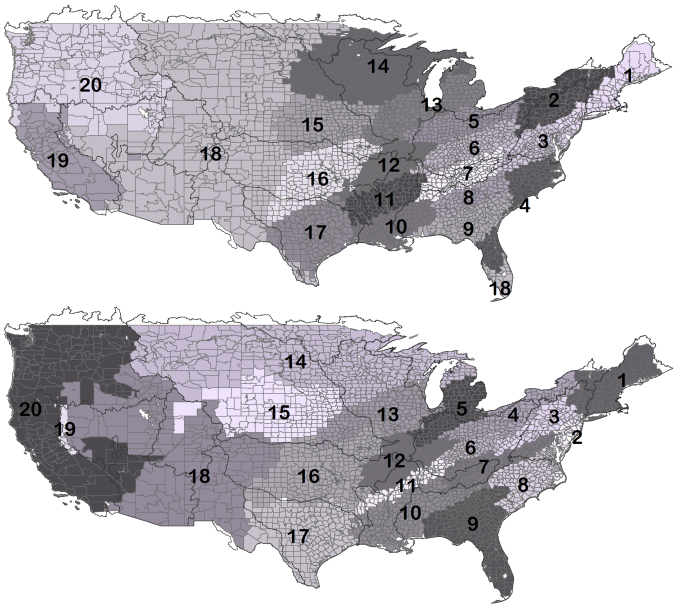


Fig. 5. Map of U.S. counties clustered into 20 Flood Regions using $k$-means (top) and GNG+GCF (bottom) algorithms on the diffusion distance. Overlaid on both maps are the boundaries of the 18 USGS Hydrological Flood Regions.

### B. Pre- and post-processing refinements

We present pre- and post-processing refinements to fine-tune the clustering solutions obtained from the previous section in order to minimize the $\mathcal{F}_{TVaR}$ and $\mathcal{F}_{AAL}$ objectives. The refinements target and correct errors in the risk measures by identifying, a priori and post hoc, events that potentially incur large errors and force them into singleton clusters. This method of improving the accuracy of the clustering solution comes at the expense of compression rate.

*a) Preprocessing:* Prior to clustering, identify "important events" that contribute most to the tail statistics of the EP curve for each county. Each such event becomes a singleton cluster and does not need to be accessed by the clustering algorithm, thereby slightly reducing the computation needed to cluster the remaining events. The important events for

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

**Algorithm 3:** Extremal Error Correction for TVaR.

The zipcodes are handled in the order of their error's magnitude. For each zipcode, the order in which events in the top 100 years (which factor into the 1%-TVaR computation) are converted to singleton clusters depends on whether the TVaR was under- or over-estimated; in the former, events incurring the most negative errors go first. This minimizes the number of additional singleton clusters. The process is iterated until all errors are within the threshold.

**Input :** Error threshold $\epsilon$, loss matrix $L$, clustered loss matrix $M$.

```
zErrors ← computeErrors(L, M);
while max(|zErrors|) > ϵ do
    zRank ← argsort(−zErrors); // decreasing
    for z in zRank do
        if zErrors[z] < −ϵ then // underestimation
            eRank ← argsort(M[:, z] − L[:, z]);
        else
            eRank ← argsort(L[:, z] − M[:, z]);
        end
        i ← 0;
        while |zErrors[z]| > ϵ do
            e ← eRank[i]; i ← i + 1;
            topYears ←computeTopYears(M[:, z]);
            if Year(e) ∈ topYears then
                // Convert event e to singleton
                M[e, z] ← L[e, z];
                zErrors[z] ←updateError(L[:, z], M[:, z]);
            end
        end
    end
    zErrors ← computeErrors(L, M);
end
```

**Output:** New clustering solution.

each county are those that make up at least 95% of its 1%-TVaR estimate, as well as those that make up 50% of its 1.5%-TVaR estimate. The union of important events for all counties is the overall important event set, and comprises 16.5% of the catalog.

*b) Postprocessing:* Upon obtaining a clustering solution, identify the events that contribute most to the TVaR or AAL error, and convert them to singleton clusters. This process necessarily decreases the compression rate. We propose the Extreme Error Correction algorithm to perform this refinement, given a desired error threshold. Algorithm 3 shows the algorithm for correcting TVaR errors; the algorithm for AAL is similar.

### C. Conflicting TVaR and AAL objectives

It is intuitive that $\mathcal{F}_{TVaR}$ and $\mathcal{F}_{red}$ are conflicting objectives, but it is less obvious that the two error metrics $\mathcal{F}_{TVaR}$ and $\mathcal{F}_{AAL}$ conflict. The reason for this conflict is best illustrated by an extreme example: *if in a clustering solution, the important TVaR events are each turned into singleton clusters, and all remaining events form one huge cluster, then one would expect a very good $\mathcal{F}_{TVaR}$ but very bad $\mathcal{F}_{AAL}$*. The act of re-allocating singleton clusters towards maintaining accuracy of important TVaR events, without changing the compression rate, comes at the expense of accuracy for other events and hence at the expense of AAL error.

This trade-off between the two objectives is shown in Figure 6, where the NSGA2 algorithm [26] was used to estimate the Pareto front for the bi-objective optimization of $\mathcal{F}_{TVaR}, \mathcal{F}_{AAL}$. The population size was 200, and a penalty on the event compression rate was imposed on the

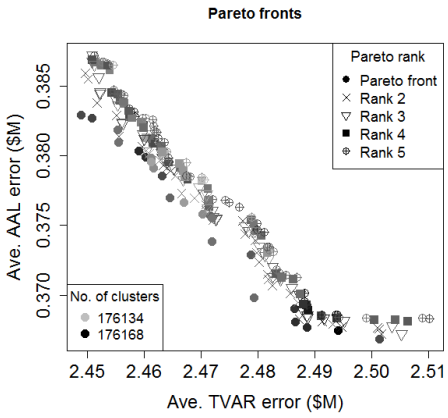


Fig. 6. Estimated (1st) Pareto front in 2-objective space (black). The other solutions are ranked by their closeness to Pareto optimality: the 2nd front becomes the new estimated Pareto front if the 1st front is removed; the $r$-th front becomes the estimated Pareto front if all fronts ranked less than $r$ are removed. All solutions have approximately 74.3% event compression rate.

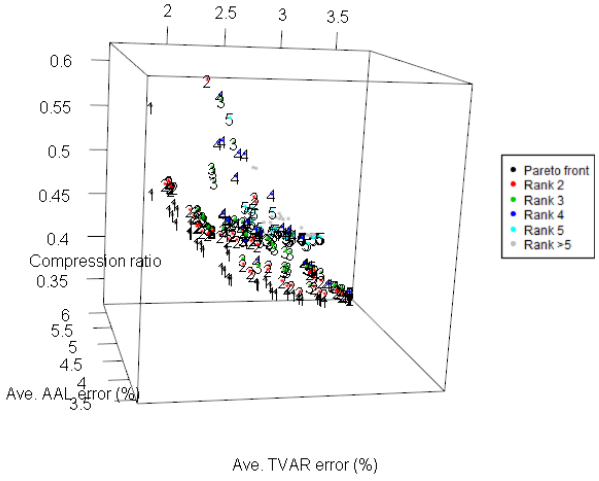


Fig. 7. Estimated Pareto fronts for 3-objective space. On the $z$-axis is the reduction rate $\mathcal{F}_{red}$. Note that the error objectives shown are the average relative errors, instead of the average absolute errors in (2), (3).

fitness function. For the genetic operators in NSGA2, the mutation operator was designed to randomly split or merge clusters. The solutions shown in Figure 6 have roughly the same event compression rate of 74.3%. However, while the $\mathcal{F}_{TVaR}, \mathcal{F}_{AAL}$ trade-off is interesting and subtle, the trade-off with $\mathcal{F}_{red}$ is more significant.

## V. Simulation Results

We ran the catalog compression methodology on the 10K U.S. flood catalog, using industry loss data at the county and zipcode resolutions. A collection of candidate solutions was obtained by running the full methodology with varying pre- and post-refinement parameters, to yield 162 solutions with county compression rates ranging from 40-70%. Figure 7 shows the candidate solutions in 3D-objective space, including the estimated Pareto optimal solutions in black. We observe a trade off between the three objectives, but the most significant trade off is between the 1%-TVaR (or AAL) error and the reduction ratio $\mathcal{F}_{red}$.

At this juncture, a risk manager is enlisted to select the final solution, based on a desired accuracy, compression

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

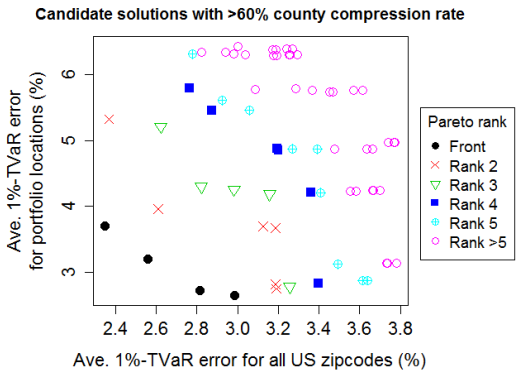


Fig. 8. Estimated Pareto fronts in 2-objective space using TVaR errors for zipcodes and portfolio locations.

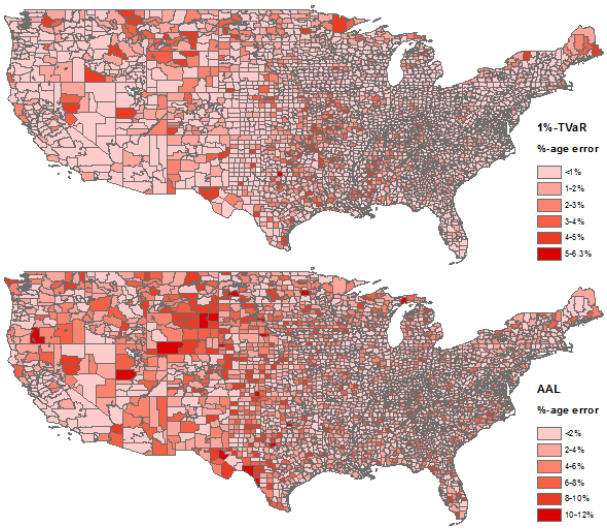


Fig. 9. Map of the U.S. showing 1%-TVaR and AAL errors by county.

rate or other managerial criteria. A typical decision making process may proceed as follows.

Suppose the risk manager desires a county compression rate of at least 60%, and he also has a portfolio of 800 insured locations on the East coast of the U.S. He is concerned with the average TVaR errors for both zipcodes nationwide and in his portfolio, but is willing to accept more error in his portfolio than nationwide, up to 5% error. He looks at the Pareto optimal solutions subject to this criteria (Figure 8), and selects the one with an acceptable level of error: 2.4% error nationwide, and 3.8% error in his portfolio. This solution has 75.1% event compression rate and 60.1% county compression rate, and is produced after the sequential application of extreme error correction to the event clustering solution at the zipcode resolution with a threshold of 10% error in TVaR and 15% error in AAL. To ensure good results at the county resolution, he further passes the selected solution through a final touch-up using extreme error correction at the county resolution with a threshold of 5% error in TVaR and 10% error in AAL. The final solution has 74.9% event compression rate and 59.9% county compression rate. The resulting errors of the final compressed catalog, at the county resolution, are shown in Figure 9. Figure 10 shows a good agreement between the EP curves estimated from the full and the clustered catalogs.

## VI. Conclusion

We showed the application of the DM for dimension reduction and spectral clustering in the context of U.S. flood insurance risk analysis. By applying event clustering in conjunction with a refinement strategy to optimize the estimation accuracy of the risk measures, we can compress the catalog size to speed up loss computation. We showed that even at event compression rates close to 75%, a high accuracy in risk measures is maintained. Risk management decisions are aided by the presentation of Pareto optimal solutions under the multiobjective optimization framework.

This entire methodology of combining general dimension reduction and clustering techniques with problem specific optimization and refinement algorithms can be useful in many other applications. Furthermore, the low dimensional embedding produced by the DM provides a good visualization of the relationships between catchments or counties (or more generally, spatial dimensions). By using an appropriate clustering algorithm, such as GNG+GCF, this clustering methodology is better able to reveal hidden relationships in the data, as illustrated by the weather patterns and geographical connectivity captured in the U.S. county loss data.

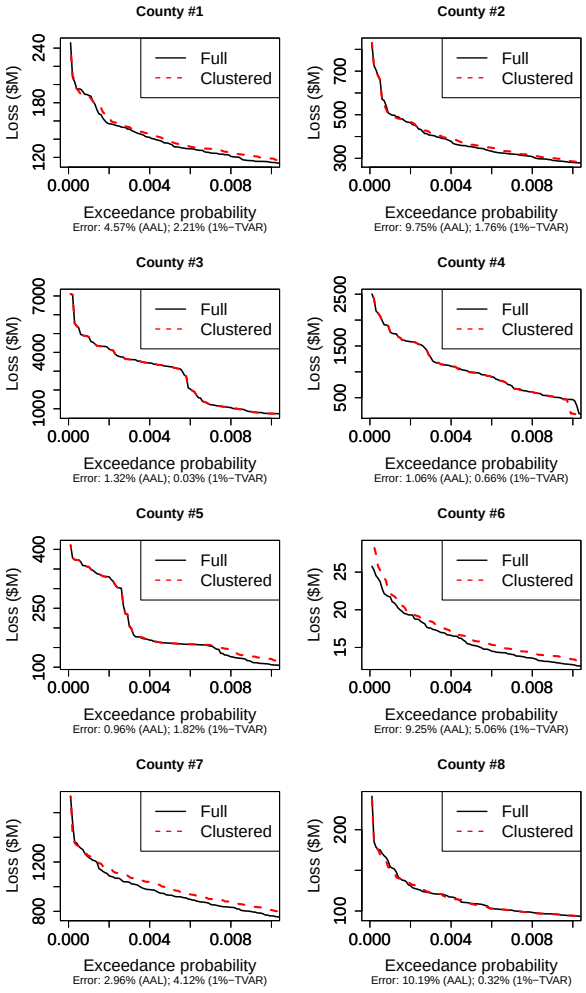


Fig. 10. EP curves constructed from the full and clustered catalogs, for selected counties, including some with the highest errors.

ISBN: 978-988-14047-1-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

## ACKNOWLEDGMENT

The authors would like to thank Dan Reese and Raulina Wojtkiewicz for their assistance with acquiring the data sets.

## REFERENCES

[1] K. Clark, "Catastrophe risk," in *IAA Risk Book – Governance, Management and Regulation of Insurance Operations*. International Actuarial Association / Association Actuarielle Internationale, 2015.

[2] S. Latchman, "Quantifying the risk of natural catastrophes," 2010. [Online]. Available: http://understandinguncertainty.org/node/622

[3] P. Grossi, H. Kunreuther, and D. Windeler, "An introduction to catastrophe models and insurance," in *Catastrophe modeling: a new approach to managing risk*, ser. Huebner International Series on Risk, Insurance and Economic Security, G. Patricia and H. Kunreuther, Eds. Springer Science+Business Media, 2005.

[4] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath, "Coherent measures of risk," *Mathematical Finance*, vol. 9, no. 3, pp. 203–228, 1999.

[5] B. Dodov and A. Weiner, "Introducing the AIR inland flood model for the United States," 2013. [Online]. Available: http://www.air-worldwide.com/Publications/AIR-Currents/2013/Introducing-the-AIR-Inland-Flood-Model-for-the-United-States/

[6] R. Wojtkiewicz, J. Rollins, and V. Foley, "U.S. flood insurance - the NFIP and beyond," http://www.air-worldwide.com/Publications/AIR-Currents/2013/U-S–Flood-Insurance%E2%80%94the-NFIP-and-Beyond/, , 2013.

[7] AIR-WORLDWIDE, "The AIR inland flood model for the United States," Brochure, 2013. [Online]. Available: http://www.air-worldwide.com/publications/brochures/documents/air-inland-flood-model-for-the-united-states/

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*, 2nd ed., ser. Springer Series in Statistics. Springer, New York, 2009.

[9] R. Bellman, *Dynamic Programming*, 1st ed. Princeton, NJ, USA: Princeton University Press, 1957.

[10] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, *Database Theory — ICDT 2001: 8th International Conference London, UK, January 4–6, 2001 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, ch. On the Surprising Behavior of Distance Metrics in High Dimensional Space, pp. 420–434.

[11] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[12] K. Q. Weinberger and L. K. Saul, "Unsupervised learning of image manifolds by semidefinite programming," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, June 2004, pp. II–988–II–995 Vol.2.

[13] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[14] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, pp. 1373–1396, 2003.

[15] R. R. Coifman and S. Lafon, "Diffusion maps," *Appl. Comput. Harmon. Anal.*, vol. 21, no. 1, pp. 5–30, 2006.

[16] S. Lafon, Y. Keller, and R. R. Coifman, "Data fusion and multicue data matching by diffusion maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1784–1797, Nov 2006.

[17] S. Lafon and A. Lee, "Diffusion maps and coarse-graining: a unified framework for dimensionality reduction, graph partitioning, and data set parameterization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1393–1403, Sept 2006.

[18] A. B. Lee and L. Wasserman, "Spectral connectivity analysis," *J. Amer. Statist. Assoc.*, vol. 105, no. 491, pp. 1241–1255, 2010.

[19] R. Talmon, I. Cohen, S. Gannot, and R. R. Coifman, "Diffusion maps for signal processing: A deeper look at manifold-learning techniques based on kernels and graphs," *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 75–86, July 2013.

[20] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[21] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer-Verlag New York Inc., 2007.

[22] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, ser. Wiley Series in Probability and Statistics. Wiley, 2005.

[23] R. T. Ng and J. Han, "CLARANS: a method for clustering objects for spatial data mining," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 5, pp. 1003–1016, Sep 2002.

[24] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern Recognition*, vol. 33, pp. 1455–1465, 2000.

[25] L. Agustin-Blas, S. Salcedo-Sanz, S. Jiménez-Fernández, L. Carro-Calvo, J. Del Ser, and J. Portilla-Figueras, "A new grouping genetic algorithm for clustering problems," *Expert Syst. Appl.*, vol. 39, no. 10, pp. 9695–9703, Aug. 2012.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr 2002.

[27] I. S. Dhillon, Y. Guan, and J. Kogan, "Iterative clustering of high dimensional text data augmented by local search," in *IEEE International Conference on Data Mining (ICDM)*, dec 2002.

[28] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, no. 2, pp. 159–179, 1985.

[29] N. Halko, P. G. Martinsson, and J. A. Tropp, "Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.

[30] S. Dasgupta and A. Gupta, "An elementary proof of the Johnson-Lindenstrauss lemma," Tech. Rep., 1999.

[31] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide: Third Edition*. Society for Industrial and Applied Mathematics, 1999.

[32] W. Czarnecki, S. Jastrzebski, M. Data, I. Sieradzki, M. Bruno-Kaminski, K. Jurek, P. Kowenzowski, M. Pletty, K. Talik, and M. Zgliczynski, *gmum.r: GMUM Machine Learning Group Package*, 2015. [Online]. Available: https://github.com/gmum/gmum.r

[33] I. T. Podolak and S. K. Jastrzebski, *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013*. Springer International Publishing, 2013, ch. Density Invariant Detection of Osteoporosis Using Growing Neural Gas, pp. 629–638.

[34] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems 7*. MIT Press, 1995, pp. 625–632.

[35] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical review. E, Statistical, nonlinear, and soft matter physics (Print)*, vol. 70, no. 6 Part 2, p. 066111, Dec 2004.

[36] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*, 2nd ed. Cambridge University Press, 2014.

[37] U. S. Geological Survey and U. S. Department of Agriculture and Natural Resources Conservation Service, *Federal Standards and Procedures for the National Watershed Boundary Dataset (WBD)*, 2013. [Online]. Available: http://pubs.usgs.gov/tm/11/a3/

[38] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *SIGMOD Rec.*, vol. 25, no. 2, pp. 103–114, Jun. 1996.