Studying Constrained Clustering Problems
Using Homotopy Maps

David R. Easterling
Computer Science Department, Virginia Tech, Blacksburg, VA 24061, dreast@cs.vt.edu

M. Shahriar Hossain
Computer Science Department, University of Texas at El Paso, El Paso, TX 79968, mhossain@utep.edu

Layne T. Watson
Departments of Computer Science, Mathematics, and Oceanic and Aerospace Engineering, Virginia Tech, Blacksburg, VA 24061 ltw@cs.vt.edu

Naren Ramakrishnan
Department of Computer Science, Virginia Tech, Blacksburg, VA 24061 naren@cs.vt.edu

Abstract
Many algorithms for constrained clustering have been developed in the literature that aim to balance vector quantization requirements of cluster prototypes against the discrete satisfaction requirements of constraint (must-link or must-not-link) sets. Significant research has been devoted to designing new algorithms for constrained clustering and understanding when constraints help clustering. However, no method exists to systematically characterize solution spaces as constraints are gently introduced and how to assist practitioners in choosing a sweet spot between vector quantization and constraint satisfaction. We present a homotopy method that can smoothly track solutions from unconstrained to constrained formulations of clustering. Beginning the homotopy zero curve tracking where the solution is (fairly) well-understood, the curve can then be tracked into regions where there is only a qualitative understanding of the solution space, finding multiple local minima along the way. Through experiments, we demonstrate how our homotopy method helps identify better tradeoffs and reveals insight into the structure of solution spaces not obtainable using pointwise exploration of parameters.

Keywords
Constrained Clustering, Homotopy Methods, Optimization, Nonlinear Minimization

1. Introduction
As machine learning permeates multiple fields of science and engineering, new objective functions are continually being proposed to suit the demands of new application domains. Multi-criteria objective functions especially are becoming more prevalent in areas such as mixing labeled and unlabeled data [2, 5, 22], incorporating constraints [10, 24, 32], and transfer learning [17, 23, 33, 34].

One such multi-objective formulation is in the area of constrained clustering. In constrained clustering [8], the goal is not just to obtain clusters that are local in their respective spaces but also to obey a discrete set of a priori must-link (ML) and must-not-link (MNL) constraints between points. Although there are many powerful constrained clustering algorithms published in the literature [8, 20, 14, 11, 15, 21], there is currently a lack of a systematic
mathematical theory to guide the design of formulations, understand tradeoffs, and explore alternatives.

The traditional solution is to introduce a parameter $\lambda$ that balances or weights competing considerations, in this case cluster locality versus constraint satisfaction. Although there are interesting theoretical insights into the complexity of constrained clustering problems [9], there is no existing theory available that can deal with (1) how to efficiently compute solutions parametrically as $\lambda$ varies, (2) how to find and deal with multiple solutions for a fixed $\lambda$ and (3) how to canonically define the best choice of $\lambda$. Since most machine learning formulations involve multiple local optima, repeated optimization for discretely varying values of $\lambda$ yields an incomplete picture of the solution space.

Homotopy methods are systematic approaches to characterize solution spaces by smoothly tracking solutions from one formulation to another (in this case, from an unconstrained formulation to a constrained formulation). This can allow the effect of changing $\lambda$ on the quality and nature of the solutions to be mathematically characterized. Smoothly tracking solutions as $\lambda$ varies provides a holistic understanding of the interplay between the algorithm and a dataset. Beginning the homotopy zero curve tracking where the solution is (fairly) well-understood, the homotopy curve can then be tracked into regions where there is only a qualitative understanding of the solution space, finding multiple local minima (for the same $\lambda$) along the way. By connecting solutions across values of $\lambda$, homotopy methods can provide the raw material for obtaining multiple distinct solutions that can then be aggregated using ensemble techniques.

Initial efforts into the application of homotopy methods to machine learning have been made in [7], where classical continuation is used as a way to study how two diverse information sources should be combined in order to arrive at an integrated model. [16] shows that a general semisupervised formulation for hidden Markov models (HMMs) can be realized using a probability-one homotopy as well. However, the creation of homotopy maps remains a bit of a black art, especially for emerging machine learning formulations.

Our key contributions here are:

1. We present the first results in constructing homotopy maps for constrained clustering problems, which combine quadratic loss functions with discrete evaluations of constraint violations. This is a non-trivial task since there are several discrete aspects to the constrained clustering problem (e.g., discrete assignments of point to clusters, discrete satisfactions or violations of constraints) that need to be accommodated in a traditional homotopy framework.
2. The construction of homotopy maps can often be problem specific and typically requires careful tweaking to ensure boundedness and convergence. Here, we demonstrate a general construct that enables the map to be applied to any constrained clustering problem, similar to existing algorithms for this purpose.
3. We show numerous experimental results demonstrating the scalability, viability, usefulness, superiority, and interpretability of the homotopy map approach to constrained clustering.

2. Some Homotopy for Data Miners

Homotopy methods have their roots in classical continuation methods, which we review first. Continuations are ways to use the known solutions for an ‘easy’ problem to aim to find the solutions for a more ‘difficult’ problem. In a standard continuation method [25, 29], given two differentiable functions $f(x)$ and $g(x)$ such that the zero $x^0$ of $g$ (where $g(x^0) = 0$) is known and the zero $x^*$ for $f(x)$ is sought, let

$$\rho(\lambda, x) = (1 - \lambda)g(x) + \lambda f(x), \quad 0 \leq \lambda \leq 1,$$

where $\rho(\lambda, x)$ is the homotopy function that connects the solutions of $g(x)$ and $f(x)$. By varying $\lambda$, we can track the solutions from $g(x)$ to $f(x)$, allowing us to explore the solution space of the constrained clustering problem.
where \( \lambda \) is the independent variable. The initial condition \( x^0 \) is known, so standard local solution methods (such as Newton’s method) may be employed iteratively as \( \lambda \) varies slowly from 0 to 1 to yield \( x^* \). Such a method, applied to the above problem, would yield the desired series of solutions along a solution curve, provided that the solving method would provide a solution for every \( \lambda \). However, there is no guarantee that a given starting function \( g \) at a given \( x^0 \) will yield a zero of \( f \), as the algorithm may become ill conditioned as the continuation progresses (see Figure 1). In particular, given a discovered solution \( \tilde{\lambda} \) along the curve and a step size of \( \Delta \lambda \), if the Jacobian derivative \( D_x \rho(\tilde{\lambda} + \Delta \lambda, x) \) is not invertible, the local solver will fail.

Continuation can fail if the zeroes of \( \rho \) do not connect the zero \( x_0 \) of \( g \) to a zero \( \bar{x} \) of \( f \) due to the curve failing to exist for some \( \lambda \) or wandering off to infinity without reaching \( \lambda = 1 \), or, as stated above, if the Jacobian becomes nonsingular due to a bifurcation or turning point in the zero curve \( \gamma \), the curve the solver tracks along the continuation as \( \lambda \) varies from 0 to 1. Unlike continuation methods, homotopy methods allow for \( \lambda \) to increase or decrease along the curve by converting them to dependent variables, dependent on the arc length of \( \gamma \) and thus able to vary in more than one direction as the solver explores the zero curve. The solver must still have some mechanism in place to deal with singularities along the curve.

To rectify this situation, a homotopy algorithm was proposed \cite{25} based on the constructive proofs of \cite{6}, which in turn are based on the Brouwer fixed point theorem. This algorithm was subsequently developed in FORTRAN as HOMPACK \cite{30}, and has since been converted into Fortran 90 as HOMPACK90 \cite{31}. The key idea of the algorithm is that if the Jacobian derivative \( D_x \rho(\lambda + \Delta \lambda, x) \) has full rank along the curve, the uniqueness of the solution for a given arc length of the zero path \( \gamma \) is guaranteed. Thus, assuming that \( \gamma \) continues to some length \( \dot{s} \) such that \( \lambda(\dot{s}) = 1 \), \( x(\dot{s}) \) is guaranteed to be a solution to the equation \( f(\dot{x}) = 0 \).

The following are important theorems in producing such curves.

**Theorem 1** \cite{6}. Let \( F : \mathbb{R}^n \to \mathbb{R}^n \) be twice continuously differentiable, and suppose there exists \( r > 0 \) such that for any \( a \in \mathbb{R}^n \) with \( \|a\| < r \), \( z - a \) and \( F(z) \) do not point in opposite directions on \( \{ z \in \mathbb{R}^n \mid \|z\| \leq r \} \). Define \( H : \mathbb{R}^n \times [0,1) \times \mathbb{R}^n \to \mathbb{R}^n \) by

\[
H(a, \lambda, z) = (1 - \lambda)(z - a) + \lambda F(z),
\]
and let $H_a(\lambda, z) = H(a, \lambda, z)$. Then for almost all vectors $a \in \mathbb{R}^n$ with $\|a\| < r$, there exists a zero curve $\gamma$ of $H_a(\lambda, z)$ emanating from $(0, a)$, along which the $n \times (n+1)$ Jacobian matrix $DH_a(\lambda, z)$ has full rank, that does not intersect itself and is disjoint from any other zeros of $H_a$, and accumulates at a point $(1, \bar{z})$ for which $F(\bar{z}) = 0$. Furthermore, if $\text{rank} DH_a(1, \bar{z}) = n$, then the curve $\gamma$ connecting $(0, a)$ to $(1, \bar{z})$ has finite arc length.

**Theorem 2** ([6]). Let $F: \mathbb{R}^n \to \mathbb{R}^m$, $p: \mathbb{R}^m \times [0, 1] \times \mathbb{R}^n \to \mathbb{R}^m$, and $\rho_a(\lambda, x) = \rho(a, \lambda, x)$. Assume that

1. $p$ is transversal to zero (rank $D\rho_p = n$ on $p^{-1}(0)$) and $C^2$;
2. for each fixed $a \in \mathbb{R}^m$, $\rho_a(0, x) = 0$ has a unique nonsingular solution $x_0$ and $\text{rank} \ D_x \rho_a(0, z_a) = n$;
3. $\rho_a(1, z) = f(z)$.
4. for each $a \in \mathbb{R}^m$, the connected component of the zero set $\rho_a^{-1}(0)$ containing $(0, z_a)$ is bounded.

Then, for almost all $a \in \mathbb{R}^m$, there exists a zero curve $\gamma$ of $H_a(\lambda, z)$ emanating from $(0, a)$, along which the $n \times (n+1)$ Jacobian matrix $DH_a(\lambda, z)$ has full rank, that does not intersect itself and is disjoint from any other zeros of $H_a$, and accumulates at a point $(1, \bar{z})$ for which $F(\bar{z}) = 0$. Furthermore, if $\text{rank} DH_a(1, \bar{z}) = n$, then the curve $\gamma$ connecting $(0, a)$ to $(1, \bar{z})$ has finite arc length.

These theorems show that each curve can take one of the following five forms. First, the curve can be a closed loop lying entirely in $(0, 1) \times \mathbb{R}^p$. Second, the curve can turn back and join two solutions of $\rho_a(0, x) = 0$ together. Third, the curve can similarly join two solutions of $\rho_a(1, x) = 0$. Fourth, the curve can be unbounded, with one endpoint in either $\{0\} \times \mathbb{R}^p$ or $\{1\} \times \mathbb{R}^p$. Fifth and finally, the curve can have one endpoint in $\{0\} \times \mathbb{R}^p$ and one endpoint in $\{1\} \times \mathbb{R}^p$, exactly the situation that allows tracking the solution to bear fruit. Since this holds for almost all $a$, $\rho$ can be constructed so that this holds for almost all $x_0$, through the simple use of an $a-x_0$ term. Thus, if the fifth condition can be guaranteed, the resulting homotopy will not have to deal with any irregularities due to singularities along the zero curve.

Since the map is always constructed so that the initial point lies on $g(x_0) = 0$, the first and third types of curves can not apply. If the solution at $\rho_a(0, x)$ is uniquely $x_0$, the second form can not apply. Finally, if $\rho_a^{-1}(0)$ is bounded, the fourth form is not applicable. Thus the only possible result is the smooth continuation of the curve from $\lambda = 0$ to $\lambda = 1$, yielding the desired solution.

In practice, the invertibility of the Jacobian derivative at $\bar{x}$, while convenient, is not necessary, as the solution tends to approach the desired solution as $\lambda \to 1$. This is especially true when applied to the semisupervised clustering problem, as the desired clustering is presumed to present at some point before the complete satisfaction of all constraints, or a direct optimization would be preferable. Homotopy maps that fulfill each of the above requirements are **globally convergent probability-one homotopy maps**. Given time to trace the finite (albeit potentially long) arc length of the solution curve with a robust enough solution tracker, such curves will inevitably yield a useful solution. More to the point in the current study, tracing the solution curve yields the entire trace for analysis, allowing for useful information to be obtained for multiple objectives.

It is possible to convert a homotopy map with multiple zeroes at $\lambda = 0$ into one with a unique zero at $\lambda = 0$. Doing so will help satisfy the above conditions where the “natural” $g(x)$ may have multiple solutions and still guarantee a global convergence. Consider the map $H(\lambda, z) = (1 - \tanh(60\lambda))(z - a_0) + \tanh(60\lambda)[(1 - \lambda)(g(z)) + \lambda(f(z))]$, where $a_0$ is a constant vector, and $\tanh(*)$ is the hyperbolic tangent function. The nature of the hyperbolic tangent function is that $\tanh(X)$ approaches one as $X \to \infty$ for all scalar $X$, and it approaches 0 as $X \to -\infty$. The $\tanh(60\lambda)$ terms in the above equation thus approach one and zero (to
concatenates the argument vectors. Given a set $\hat{\lambda}, z \in \mathbb{R}^{d}$ of two data points (cluster representatives) in $d$ dimensions, let $\lambda, z \in \mathbb{R}^{d}$ of two data points (cluster representatives) in $d$ dimensions. For the purposes of this map, consider each cluster assignment to be a "hard" assignment, that is, each data point is assigned to a single cluster determined by its distance from the cluster representative, not assigned a probability of belonging to each cluster based on those distances.

Let superscripts denote vector indices and subscripts denote components of vectors unless otherwise indicated. Let all norms be 2-norms and all distances be Euclidean distances. Given a set $\hat{X} = \{x^i \mid x^i \in \mathbb{R}^d, i = 1, 2, \ldots, k\}$ of $k$ points (cluster representatives) in $d$ dimensions, let $\hat{X} = \text{vec}(x^1, x^2, \ldots, x^k) \in \mathbb{R}^{kd}$, where vec is the vector function that concatenates the argument vectors. Given a set $\hat{Y} = \{y^i \mid y^i \in \mathbb{R}^d, i = 1, 2, \ldots, n\}$ of $n$ data points in $d$ dimensions, let $Y = \text{vec}(y^1, y^2, \ldots, y^n) \in \mathbb{R}^{nd}$. Represent a constraint by the vector $c = (a, b, z, w) \in \mathbb{R}^{2d+2}$ of two data points $a, b \in \hat{Y}$, an identifier $z = \pm 1$, and a degree-of-belief weight $w \in \mathbb{R}$, where an identifier of $z = 1$ means that $a$ and $b$ are bound by a must-link constraint (i.e., must be in the same cluster) and an identifier of $z = -1$ means that $a$ and $b$ are bound by a must-not-link or cannot-link constraint (must not be in the same cluster). Given a set $\hat{C} = \{c^i \mid c^i \in \mathbb{R}^{2d+2}, i = 1, 2, \ldots, q\}$ of $q$ constraints, let $C = \text{vec}(c^1, c^2, \ldots, c^q) \in \mathbb{R}^{q(2d+2)}$.

### 3.1. Functions

For a data point $y \in \hat{Y}$ and two cluster prototypes $x^j, x^j' \in \hat{X}$ define the function $D : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ by

$$D(x^i, x^j, y) = \left( \max\{0, \|x^i - y\|^2 - \|x^j - y\|^2\} \right)^4.$$  

Note that $D$ is three times continuously differentiable, $D \geq 0$, and $D(x^i, x^j, y) > 0$ if and only if the distance between $y$ and $x^i$ is larger than the distance between $y$ and $x^j$. 
Given \( a, b \in \hat{Y} \), let the must-link function \( F_m : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^{kd} \rightarrow \mathbb{R} \) be defined by

\[
F_m(a, b, X) = \prod_{i=1}^{k} \left( \sum_{j=1, j \neq i}^{k} D(x^i, x^j, a) + D(x^i, x^j, b) \right)
\]

and let the cannot-link function \( F_c : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^{kd} \rightarrow \mathbb{R} \) be defined by

\[
F_c(a, b, X) = \sum_{i=1}^{k} \left( \prod_{j=1, j \neq i}^{k} D(x^i, x^j, a)D(x^i, x^j, b) \right).
\]

Then the following observations are easily verified.

**Observation 1.** \( F_m \) and \( F_c \) are nonnegative and three times continuously differentiable.

**Observation 2.** For any must-link constraint \( c = (a, b, 1, w) \in \hat{C} \), the must-link function \( F_m(a, b, X) = 0 \) if and only if constraint \( c \) is satisfied.

**Observation 3.** For any cannot-link constraint \( c = (a, b, -1, w) \in \hat{C} \), the cannot-link function \( F_c(a, b, X) = 0 \) if and only if constraint \( c \) is satisfied.

**Observation 4.** The penalty function

\[
F(C, X) = \sum_{\{i: z_i = 1\}} F_m(a^i, b^i, X) + \sum_{\{i: z_i = -1\}} F_c(a^i, b^i, X)
\]

is zero if and only if all the constraints in \( \hat{C} \) are satisfied.

It is simple to add a degree-of-belief weight \( w_i > 0 \) to each component of the penalty function \( F \) without eliminating its properties:

\[
F(C, X) = \sum_{\{i: z_i = 1\}} w_i F_m(a^i, b^i, X) + \sum_{\{i: z_i = -1\}} w_i F_c(a^i, b^i, X).
\]

By Observation 4, if it is possible to satisfy all of the constraints, then there exists a vector of cluster representatives \( X \) such that \( F(C, X) = 0 \). This vector of cluster representatives represents a global minimum point of the function \( F \) at which \( \nabla_X F(C, X) = 0 \). This suggests the homotopy map \( \tilde{\rho}_a(\lambda, X) = (1 - \lambda)(X - k_0) + \lambda (\nabla_X F(C, X))^T \).

The advantages of this homotopy map should be immediately evident. When \( \lambda = 0 \), the solution to the above map is simply the solution to the unsupervised clustering problem. When \( \lambda = 1 \), the solution, if one exists, represents a local minimum of the penalty function, which is based on the violation of constraints. This is not to say that the solution generated will satisfy all the constraints if such a solution is possible, as it is fairly easy to construct a degenerate set of constraints so that there is a local solution close to \( x = k_0 \). However, in practice this has not proven to be a problem.

This is a probability-one homotopy map, but while it satisfies conditions (1), (2), and (3) in Theorem 2, it fails to satisfy condition (4), boundedness. Furthermore, there is a trivial solution to all constraints at \( x^1 = x^2 = \ldots = x^k \), where all cluster representatives are equal. Thus, two modifications must be made to the above map. First, a boundedness constraint must be applied to \( F \) to satisfy condition (4). Secondly, a constraint must be imposed to prevent the cluster representatives from degenerative stacking. Converting the minimum of this unconstrained penalty function to the minimum of a constrained function is simply a matter of converting the solution \( \tilde{X} \) from a zero of \( \nabla_X F(C, X) \) to a Karush-Kuhn-Tucker
(KKT) point of the constrained minimization problem, where the constraints (listed below) are formed to force both boundedness and a separation of prototypes. A KKT point is simply a feasible point that satisfies the KKT conditions. Recall that the KKT conditions for minimizing $F(X)$ with inequality constraints require that

$$\nabla_X F(C, X^*) + \sum_{i=1}^{m} \mu_i \nabla g_i(X^*) = 0$$

so that there exists $X^*$, $g_i(X^*) \leq 0$, $\mu_i \geq 0$ and $\mu_i g_i(X^*) = 0$ for all $i = 1, \ldots, m$.

### 3.2. Homotopy map

First, consider the boundedness constraint. A straightforward function $\Psi : \mathbb{R}^{kd} \rightarrow \mathbb{R}$ to achieve boundedness is $\Psi(X) = B - \sum_{i=1}^{n} \|x_i\|^2 \geq 0$, where $B$ is a user-defined boundedness constant. The Lagrangian of the new bounded penalty function is $\hat{L}(X, \mu) = F(C, X) - \mu \Psi(X)$, and its derivative, replacing $\nabla_X F(C, X)$, is $\nabla_X \hat{L}(X, \mu) = \nabla_X F(C, X) - \mu \nabla_X \Psi(X)$. This yields a new variable, the Lagrangian multiplier $\mu$, which in turn adds a new function to the map (since the map must be from $\mathbb{R}^{p+1} \rightarrow \mathbb{R}^p$ for some $p$), along with the requirement that $\mu \geq 0$, $\Psi \geq 0$, and $\mu \Psi = 0$. This naturally leads to the use of the Mangasarian NCP function presented in [18] and modified in [26, 28], given next.

Define the function $\Phi : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ by

$$\Phi(\mu, \Psi(X)) = -|\mu - \Psi(X)|^3 + \mu^3 + \Psi(X)^3 - (1 - \lambda)h_0$$

for some constant $h_0 > 0$. The constant term $h_0$ is designed to force the remaining terms to remain positive for $\lambda < 1$, which enforces the boundedness of $X$ for $\lambda < 1$, since $\Psi(X)$ must remain positive so that $\Phi = 0$. The above homotopy map is then modified to

$$\rho_\ell(\lambda, X) = \left[ (1 - \lambda)(X - k_0) + \lambda(\nabla_X \hat{L}(X, \mu))^T \right] / \Phi(\mu, \Psi(X))$$

Unfortunately, while the stated $\Phi$ enforces a lower bound on $\mu$, it does not enforce an upper bound on $\mu$; in fact, if the solution $X$ approaches the boundary as $\lambda \rightarrow 0^+$, $\mu$ must potentially become arbitrarily large to compensate for this. While this map is better than the previous one in that it prevents cluster representatives from migrating arbitrarily far from the data set, it does not prevent $\mu$ from growing arbitrarily large, although in practice this is not a common occurrence.

Similarly, define the function $G : \mathbb{R}^{kd} \rightarrow \mathbb{R}$ by

$$G(X) = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \max(0, \ell - \|x_i - x_j\|^2)^4, \quad x_i, x_j \in \hat{X}.$$ 

Then $G \geq 0, G \in C^3$ and $G = 0$ unless two cluster representatives $x_i$ and $x_j$ are at a distance less than $\ell$ from each other, where $\ell$ is the user-defined regularization parameter. This represents an equality constraint on the original problem. The updated Lagrangian becomes $\hat{L}(X, \mu) = F(C, X) - \mu \Psi(X) + \nu G(X)$. In turn, $\nabla_X \hat{L}(X, \mu) = \nabla_X F(C, X) - \mu \nabla_X \Psi(X) + \nu \nabla_X G(X)$. The additional function is much simpler here, as $G(X)$ is obviously bounded above by $k(k-1)\ell^4/2$ and below by 0. Let $G(X)$ serve as the final regularization function when $\lambda = 1$, thus fulfilling the equality constraint conditions, and let $\nu$ be uniquely determined at $\lambda = 0$ by some initial $\nu_0$. Since it can be assumed that $G(X_0) = G(k_0) = 0$ for any reasonable $\ell$, the final hard clustering map is $\rho_\ell(\lambda, X, \mu, \nu) =

$$
\left[ (1 - \lambda)(X - k_0) + \lambda(\nabla_X \hat{L}(X, \mu))^T \right] / \Phi(\mu, \Psi(X))

\left[ (1 - \lambda)(\nu - \nu_0) + G(X) \right].$$
3.3. Alternative linear bounding constraints

An alternative bounding function replaces the nonlinear boundedness constraint $\Psi \geq 0$ with a linear bounding constraint. Define $\hat{B} : [0, 1) \times \mathbb{R}^{kd} \to \mathbb{R}^{kd+1}$ by $\hat{B}(\lambda, X) = AX - B - (1 - \lambda)b_0$, so that $\hat{B} \leq 0$ enforces boundedness of $X$, where $A \in \mathbb{R}^{(kd+1) \times kd}$ is constructed to have full column rank, $b_0 > 0$ is some relaxation constant, and $B \in \mathbb{R}^{kd+1}$. In this construction, $\mu \in \mathbb{R}^{kd+1}$ and $\nabla X \hat{B}(\lambda, X) = A$. Letting $\Omega : \mathbb{R}^{kd+1} \times \mathbb{R}^{kd+1} \to \mathbb{R}^{kd+1}$ be defined by $\Omega_i(\mu, -\hat{B}(\lambda, X)) = \Phi(\mu_i, -\hat{B}(\lambda, X)_i)$, $1 \leq i \leq kd+1$, yields

$$\rho_a(\lambda, X, \mu, \nu) = \begin{bmatrix} (1 - \lambda)(X - k_0) + \lambda(\nabla X L(X, \mu, \nu))^T \\ \Omega(\mu, -\hat{B}(\lambda, X)) \\ (1 - \lambda)(\nu - \nu_0) + G(X) \end{bmatrix},$$

where $L(X, \mu, \nu) = F(C, X) + \mu \hat{B}(\lambda, X) + \nu G(X)$.

Letting $-e^t = (-1, \ldots, -1) \in \mathbb{R}^{kd}$ and $I \in \mathbb{R}^{kd \times kd}$ be the identity matrix, a suitable $A$ and $\hat{B}$ would be

$$A = \begin{pmatrix} I \\ -e^t \end{pmatrix}, \quad \hat{B} = \begin{pmatrix} B \ell \\ B_{kd} \end{pmatrix}.$$

Then $\mu$ for this function $\rho_a$ is implicitly bounded where $0 \leq \lambda < 1$ by exactly the same argument employed for linearly constrained convex optimization in [28]. The disadvantage to this method is that it involves a doubling of the number of dimensions to be tracked, involving an increase in the number of computations to be performed at each step. The advantage is that, unlike the previous map, this one is provably globally convergent under weak assumptions (that the constraints that depend on $B$ and $\ell$ are satisfied by the initial solutions), satisfying all four requirements for global convergence.

4. Experimental Results

We evaluate and demonstrate the characteristics of our approach using both synthetic and real datasets. The questions we seek to answer through our experiments are

(1) Do homotopy maps help in trading off locality with constraints satisfied? (Section 4.1)
(2) Can homotopy maps help find better solutions than tailor made algorithms for constrained clustering? (Section 4.2)

(3) How well our algorithm performs when there are faulty constraints compared to other methods? (Section 4.3)

(4) Do homotopy maps reveal insight into the structure of solutions otherwise not obtainable using pointwise exploration of the $\lambda$ parameter? (Section 4.4)

(5) Do linear bounding constraints, which guarantee a globally convergent solution, track a reasonable solution in time comparable to nonlinear bounding constraints? (Section 4.5)

(6) How does the runtime of the framework scale with increasing dimensions, increasing number of clusters, and increasing number of data points? (Section 4.6)

(7) How do the cluster prototypes change their positions over the iterations of the homotopy tracking function? (Section 4.7)

In this section, we use different types of evaluation for constrained clustering as well we homotopy tracking mechanism. The level to which two clusterings are similar is measured using the Rand index [19]. The Rand index ranges from 0 to 1 and a larger Rand index value implies higher similarity between two clusterings. Other evaluation metrics used in this section are sum of squared distance (SSD) to compute the locality of the clusters generated, and straightforward use of the number of constraints violated.

4.1. Constraint Satisfaction vs. Clusters’ Locality

We begin with a simple synthetic dataset, involving 200 points gathered from four Gaussian distributions. We aim to find two clusters from this dataset. There are multiple natural clusterings possible, depending on whether the clusters are organized horizontally or vertically, among other options. We begin with one of these clusterings and as the homotopy curve is tracked, constraints are slowly introduced. We carefully prepared a list of 50 constraints in such a way that the must-link constraints are picked from two different initial clusters and the must-not-link constraints are picked from the same initial clusters. Thus, the clusters are forced to reorganize as $\lambda$ is varied.

Figure 2 (left) shows the data points and the constraints. The solid (green) lines denote the must-link constraints and the dashed (pink) lines denote the must-not-link constraints. During the homotopy curve tracking, the cluster prototypes smoothly traverse the space and finally settle down to a position where a maximum of constraints are satisfied.

In addition to the data points and the constraints, Figure 2 (left) shows the paths of the cluster prototypes as $\lambda$ is varied. Figure 2 (right) depicts that as $\lambda$ increases during the tracking, the number of constraint violations reduces but the sum of squared distance (SSD) increases, as expected. At $\lambda = 0.0$, recall that we have a $k$-means local minimum, where our sum of squared distance is low but the number of constraint violations is high. At the other end of the curve, at $\lambda = 1.0$, the sum-of-squared distance reaches its peak but all the constraints are satisfied. The actual crossing of these curves gives us insights into how much one objective needs to be traded off in achieving another.

As shown in Figure 2 (right), all the constraints are satisfied as the homotopy curve reaches $\lambda = 1$. To compare the homotopy results with a traditional constrained clustering algorithm, we applied the MPC$k$-means [1] algorithm on the same dataset and constraints. Only 52% of the constraints were satisfied by MPC$k$-means.

4.2. Comparison of the Developed Constrained Clustering Method with other Methods

The homotopy based constrained clustering approach is unique from any other constrained clustering method because it allows the user to track dual objectives and opens the opportunity to flexibly analyze the tradeoff between those objectives. In this subsection we take the
result with maximum number of constraint satisfaction and compare it with some already existing constraint clustering methods.

We used two UCI ML repository datasets, “iris” and “bupa”, to illustrate this comparative study. Both the datasets have benchmark classification labels. We first applied $k$-means clustering on the target dataset and then created constraints to drive the solution generated by $k$-means toward the benchmark classification labels. Then we computed the Rand index between the result obtained and the classification labels. We used the same set of constraints for our homotopy based constrained clustering framework and two other constrained clustering algorithms, Metric Pairwise Constraints $k$-means (MPC$k$-means) and Pairwise Constraints $k$-means (PC$k$-means) from [4]. Figure 3 (left) shows the improvements in Rand index gained by our method over MPC$k$-means and PC$k$-means for the iris dataset. We observe that our method gained positive improvement over both MPC$k$-means and PC$k$-means with increasing number of constraints. The comparative gain achieved by the homotopy method against the PC$k$-means method is always higher than the comparative gain achieved against the MPC$k$-means method because the metric learning mechanism of MPC$k$-means helps in satisfying more constraints than are satisfied by PC$k$-means. The homotopy method tends to perform better with larger numbers of constraints, resulting in a larger gain at the right side of the plot. This indicates that the homotopy method provides better desired clustering with larger numbers of constraints, as should be expected from any constraint clustering algorithm. On the other hand, MPC$k$-means and PC$k$-means fail to realize the constraints in the generated clustering when the constraints are large in number.

The maximum improvement we observed with the iris dataset is 32%. Similar behavior is observed for the homotopy based method with the bupa dataset (see Figure 3 (right)). However, the maximum improvement obtained with this dataset was 6%.

4.3. Faulty Constraints

In Section 4.2, we compared our approach with MPC$k$-means and PC$k$-means. The objective was to drive the $k$-means solution toward a given clustering using must-link and must-not-link constraints. While providing feedback, it is possible that a user provides faulty constraints instead of providing all satisfiable constraints. For example, users can provide one constraint as must-link as well as provide the same constraint as must-not-link. In practice, this can occur from a wide array of activities, from crowdsourcing to glean new insight about a clustering problem to polling entertainment users about their preferences and using the results to make suggestions to users that may not be entirely valid due to things such as mood swings and subjective value changes. Additionally, a must-link constraint between
Figure 4. UCI ML repository “pima” (left) and “ionosphere” (right) datasets.

Note. Comparison between homotopy based constrained clustering method with MPC\(k\)-means and PC\(k\)-means when there are unsatisfiable constraints.

points \(p_1\) and \(p_2\) and a must-not-link between points \(p_3\) and \(p_4\) can be conflicting if \(p_1\) is very close to \(p_3\) and \(p_2\) is a close neighbor of \(p_4\). In this section, we verify how our method performs in the presence of such faulty constraints compared to MPC\(k\)-means and PC\(k\)-means. We report a Rand index similar to the plots as we reported in Section 4.2. However, in this section the Rand index is plotted against number of faulty constraints. In addition to MPC\(k\)-means and PC\(k\)-means we also provide the Rand index for unconstrained \(k\)-means in the same plot. We perform this comparative study on two different UCI datasets, “pima” and “ionosphere”.

Based on the results reported in Figure 4 our homotopy based constrained clustering framework performs better than any other method when examined with a range of faulty constraints varying from 5 to 30 out of 100 total constraints. Rand index for \(k\)-means with the pima dataset is 0.54 and 0.59 with the ionosphere dataset. Since \(k\)-means algorithm is not affected by constraints, Rand indices for \(k\)-means remain constant in both the plots of Figure 4. This figure shows that our approach produces better clusterings (by way of a larger Rand index) than any other method and always produces better results than the unconstrained \(k\)-means. This indicates that the homotopy based method can track a good solution even in the presence of faulty constraints. On the other hand, with both the pima and ionosphere datasets, MPC\(k\)-means and PC\(k\)-means perform worse than the unconstrained \(k\)-means in the presence of faulty constraints.

4.4. Tracking Multiple Solutions

Recall that the homotopy method works for multiple solutions for a given dataset and constraint set. Figure 5 shows an example where we track three different solutions for the same dataset and constraints as in Figure 2. One of these three tracking curves (the top curve) corresponds to Figure 2 (right). Conventional optimization algorithms would require a fixed \(\lambda\) to balance the locality and constraint satisfaction trade-off of the objective function whereas homotopy smoothly tracks \(\lambda\) without requiring any user input for balancing the trade-offs. Discretely sampling \(\lambda\) and using a conventional local optimizer would likely detect solutions from different curves of the homotopy map. As an example, Figure 5 shows three points from three different curves which we obtained by a local optimizer at \(\lambda = 0.2\), \(\lambda = 0.4\), and \(\lambda = 0.8\). This demonstrates that discrete sampling of \(\lambda\) does not give us insight into the effect of \(\lambda\) as this would lead us to mistakenly conclude that the effect of \(\lambda\) has an inflection point.
4.5. Linear vs. Nonlinear Bounding Strategies

The linear bounding strategy proposed in Section 3.3 has the theoretical advantage of being provably bounded; that is, the associated $\mu$ components can not diverge to infinity as the algorithm progresses. The nonlinear bounding strategy listed in Section 3.2 does not have this theoretical advantage, but in practice it does not suffer from unrestricted growth in any component preventing the advancement of the homotopy tracking algorithm. While the two different maps can lead to substantially different zero curves, in practice none of the boundedness inequality constraints have become active in any of the experiments performed here, as the zero curve behavior has been far more dominated by the constraints used. Nevertheless, the experiments depicted by Figure 6 were performed to demonstrate the differences in execution time brought about by the varying number of constraints used in the experiments, along with the different functions employed to ensure boundedness.

4.6. Scalability of the Approach

We used synthetic datasets and random constraints for the scalability experiments of this section. Figure 7 depicts the runtime characteristics of our algorithm with increasing number of (a) data points, (b) dimensions, (c) constraints, and (d) clusters. As illustrated by Figure 7(a), our approach scales very well with increasing number of data points. The plot shows that the runtime of the homotopy tracking based constrained clustering algorithm is not significantly affected by the number of data points. The runtime monotonically increases with number of dimensions (see Figure 7(b)), number of constraints (see Figure 7(c)), and number of clusters (see Figure 7(d)). Also note that the runtime includes the time to track a $k$-means solution from an unconstrained setting to complete constraint satisfaction by smoothly generating a large set of intermediate solutions. The time taken is quadratic in both the number of dimensions and the number of clusters, and linear with the number of constraints. Of note is that these experiments were designed to not take advantage of maps that quickly converge to a minimum, but instead to proceed until the maximum number of steps of the solution tracker were traversed, so that the full time difference could be explored without being skewed by early success.
Figure 6. Linear vs. nonlinear bounding requirements.

Comparison between linear and nonlinear bounding strategies

<table>
<thead>
<tr>
<th>Number of constraints</th>
<th>Runtime (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>5</td>
</tr>
<tr>
<td>50</td>
<td>6</td>
</tr>
<tr>
<td>60</td>
<td>7</td>
</tr>
<tr>
<td>70</td>
<td>8</td>
</tr>
<tr>
<td>80</td>
<td>9</td>
</tr>
<tr>
<td>90</td>
<td>10</td>
</tr>
<tr>
<td>100</td>
<td>11</td>
</tr>
<tr>
<td>110</td>
<td>12</td>
</tr>
</tbody>
</table>

Linear bounding
Nonlinear bounding

Note. The plots depict the linear bounding requiring more time than the nonlinear bounding.

Figure 7. Results to verify the scalability of the homotopy method.

(a) (b) (c) (d)

Note. (a) Runtime of our algorithm is not influenced by the number of data points. (b) The plot shows the runtime of the algorithm as a function of the number of data points. (c) Runtime as a function of number of constraints. (d) Runtime as a function of number of clusters.

Figure 8. Six Gaussian distribution dataset.

Note. There are 100 constraints to drive the $k$-means clustering results to an alternative solution. The images show how the cluster prototypes change their positions over the iterations of the homotopy tracking function. Only two of the plots are shown in this figure, out of 375 steps.
4.7. Demonstration of Homotopy Tracking for Alternative Clustering

Alternative clustering (e.g., [12, 15]) is the idea of uncovering multiple clusterings of a dataset so as to suggest varying viewpoints and differing hypotheses. We demonstrate the power of homotopy curve tracking in finding a solution to find an alternative clustering. We utilize a 2D synthetic dataset having six Gaussian distributions, each with 50 points, arranged uniformly around a circle. Each of the images of Figure [S] depicts the dataset. Three initial prototypes generated by k-means are shown by three cross marks having red, blue, and black colors. We generated 100 random constraints in a way that must-link constraints are from two different k-means clusters and must-not-link constraints are from the same cluster. The constraints are shown using green and purple straight lines.

Figure [S] shows two plots illustrating the path of the cluster prototypes during the track of a solution. The plots depict that the solutions (cluster prototypes) smoothly change their positions in the solution space. Note that this demonstration is for one set of starting prototypes that were generated by k-means. For any other local minima generated by k-means, the demonstration of this tracking will be different. The right plot of Figure [S] shows that the prototypes reach a final destination where all the constraints are satisfied and that it is an alternative clustering solution with three clusters for six Gaussian distributions.

5. Discussion

We have developed new homotopy theory for constrained clustering problems and used state-of-the-art mathematical software to characterize multi-criteria problems in constrained clustering. Just as in other applications of homotopy methods to science and engineering, the application of homotopy methods to machine learning problems can usher in greater understanding of solution sets. Besides the strong mathematical foundations and rigorous formalisms brought to classic machine learning problems, this work has the potential to greatly reduce the ad hoc nature of methodological experimentation that is prevalent in practice. The approach given here not only helps extract better patterns from data, but also it helps formally understand the internal workings of machine learning techniques. In the future, we plan to design homotopy maps for other machine learning problems including information bottleneck, time series segmentation, and transfer learning.

References


