The Chinese University of Hong Kong, Shenzhen
SDS · School of Data Science

Andre Milzarek · Fall Semester 2021/22

MDS 6106 – Introduction to Optimization

Final Project

Convex Clustering

The goal of this project is to investigate different optimization models and to utilize minimization methodologies that were introduced in the lecture to solve convex clustering problems for large-scale unsupervised learning.

Project Description. Clustering is one of the most fundamental problems in unsupervised learning. Different from usual classification tasks, unsupervised learning problems do not require labeled training data and try to estimate labels, classes, or clusters directly from the given data. Traditional clustering models, such as K-means and hierarchical clustering, often suffer from poor performance because of the inherent non-convexity of the models and the difficulties in finding global optimal solutions for such models. The clustering results are generally highly dependent on suitable initializations and can differ significantly for different starting points. More importantly, standard clustering models require prior knowledge about the number of clusters which is not available in many applications. Therefore, in practice, K-means is typically applied with different cluster numbers and the user then decides on an appropriate value.

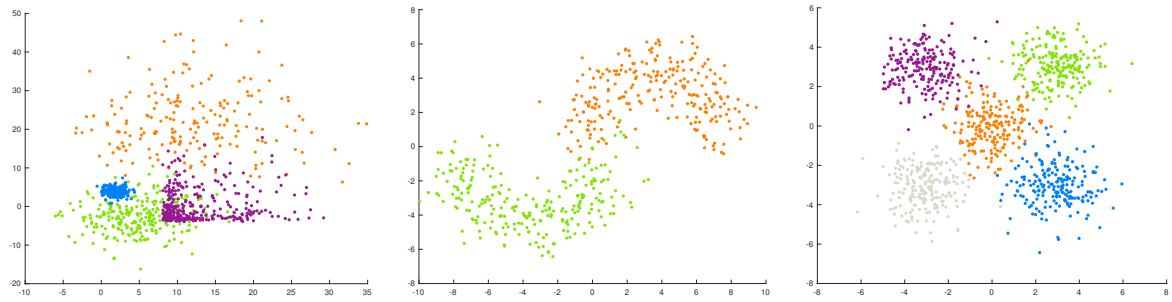

Figure 1: Illustration: Different two-dimensional point clouds.

In order to overcome these issues, new convex clustering models have been recently proposed. Let $\mathbb{R}^{d\times n} \ni A = (a_1, a_2, \ldots, a_n)$ be a given data matrix with n observations and d features. The convex clustering model for these n observations solves the following convex optimization problem:

$$\min_{X\in\mathbb{R}^{d\times n}} \ \frac{1}{2}\sum_{i=1}^{n}\|x_i - a_i\|^2 + \lambda\sum_{i=1}^{n}\sum_{j=i+1}^{n}\|x_i - x_j\|_p \tag{1}$$

where $\lambda > 0$ is a regularization parameter and $\|\cdot\|_p$ denotes the p-norm. Here, $\|\cdot\|$ denotes the standard Euclidean norm (as usual). The p-norm above with $p \geq 1$ ensures the convexity of the model. Typically, p is chosen to be 1, 2, or ∞.

After solving (1) and obtaining the optimal solution $X^* = (x_1^*, \ldots, x_n^*)$, we assign a_i and a_j to the same cluster if and only if $x_i^* = x_j^*$. In other words, x_i^* acts as a centroid (center of the cluster) for the observation a_i. In practice, instead of requiring $x_i^* = x_j^*$, we can assign a_i and a_j to the same cluster if $\|x_i^* - x_j^*\| \leq \varepsilon$ for a given tolerance $\varepsilon > 0$. The key idea behind the convex clustering model is that, if two observations a_i and a_j belong to the same cluster, then their corresponding centroids x_i^* and x_j^* should be the same. The first term in the model (1) is the fidelity or data

term while the second term is the regularization term to penalize the differences between different centroids so as to enforce the property that centroids for observations in the same cluster should be identical.

The advantages of convex clustering lie mainly in two aspects. First, since the clustering model (1) is strongly convex, the optimal solution of the model for a given positive λ is unique and is more easily obtainable than traditional clustering algorithms like K-means. Second, instead of requiring prior knowledge of the number of clusters, we can generate a clustering path via solving (1) for a sequence of positive values of λ, see, e.g., Figure 2.

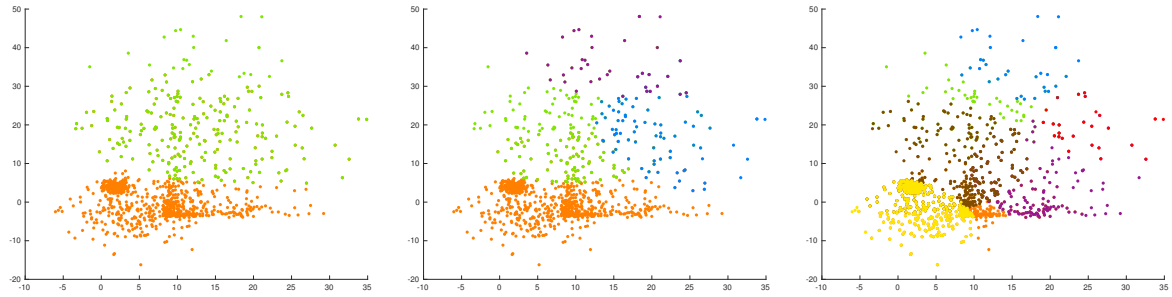

Figure 2: Illustration: Clustering results on the dataset shown in Figure 1 for $\lambda \in \{0.17, 0.1, 0.07\}$, respectively.

In this project, we want to study different optimization approaches for the convex clustering model (1) and potential alternative strategies and compare their performance on several real-word data sets and tasks.

Project Tasks.

1. *Data Preparation.* This first part of the project concerns data generation and data preparation for our optimization problems. We want to discuss two categories of data: self-generated data point clouds in the two-dimensional plane and datasets coming from real-world large-scale applications.

   – Given $n_1, n_2, \ldots, n_p \in \mathbb{N}$, generate different data points $a_1, \ldots, a_n \in \mathbb{R}^2$, $n = \sum_{k=1}^{p} n_k$, with p clusters, such that

   $$a_1, \ldots, a_{n_1} \text{ belong to cluster } \mathcal{C}_1, \quad \ldots, \quad a_{n_1+\cdots+n_{p-1}+1}, \ldots, a_n \text{ belong to cluster } \mathcal{C}_p.$$

   The data points a_i can be generated in many different ways. For instance, let $c_1, \ldots, c_p \in \mathbb{R}^2$ be reference points. Then, we can set

   $$a_j = c_i + \begin{pmatrix} \varepsilon_{j,1} \\ \varepsilon_{j,2} \end{pmatrix}, \quad \varepsilon_{j,1}, \varepsilon_{j,2} \sim N(0, \sigma_i^2)$$

   for all $j = \sum_{k=1}^{i-1} n_k + 1, \ldots, \sum_{k=1}^{i} n_k$ and $i = 1, \ldots, p$. Here, $\varepsilon_{j,1}$ and $\varepsilon_{j,2}$ are error terms that follow a Normal distribution with zero mean and variance σ_i^2, respectively. Data clouds generated by this exemplary model will then cluster around the reference points $c_1, \ldots, c_p$ and the spread of the data points is controlled by the selected variance.

   Generate a variety of data test sets (at least three–four) for the optimization problems following this outlined strategy. Vary the size of the datasets and their shape (see also Figure 1 for an illustration) to cover interesting situations.

   – On Blackboard, we have provided four different real-world datasets used for multi-class classification. The datasets are taken from the `LIBSVM` library `https://www.csie.ntu.edu.tw/~cjlin/libsvm/` and are stored in `MATLAB`'s `mat`-format. You can access the data points and labels via

data set	d	n	classes	description
`wine`	13	178	3	This data is the result of a chemical analysis of wines grown in the same region in Italy but derived from three different types of wine. The analysis determined the quantities of 13 components found in each of the wines.
`vowel`	10	528	11	–
`segment`	19	2310	7	–
`mnist`	784	60000	10	The MNIST database consists of 60000 handwritten digits. The digits have been normalized and centered in fixed-size 28×28 images.

Table 1: A description of the datasets used in the numerical comparison.

`load('dataset_data.mat','A')`, `load('dataset_label.mat','b')`.

An overview of the prepared datasets is given in Table 1. Download the datasets (from BB or the mentioned website) and check if you can access and store the data points in your workspace. Additional tools and different datasets can also be found on the `LIBSVM` website.

2. *Huber-type Clustering.* In the following, we choose $p = 2$ and we first consider a smooth variant of the clustering problem (1):

$$\min_{X \in \mathbb{R}^{d \times n}} \; f_{\texttt{clust}}(X) := \frac{1}{2} \sum_{i=1}^{n} \|x_i - a_i\|^2 + \lambda \sum_{i=1}^{n} \sum_{j=i+1}^{n} \varphi_{\texttt{hub}}(x_i - x_j). \tag{2}$$

Here, $\varphi_{\texttt{hub}}(y)$ denotes the Huber-type version of the Euclidean norm $\|y\|_2$ that was already introduced in the lectures:

$$\varphi_{\texttt{hub}}(y) = \begin{cases} \frac{1}{2\delta}\|y\|^2 & \text{if } \|y\| \leq \delta, \\ \|y\| - \frac{\delta}{2} & \text{if } \|y\| > \delta. \end{cases}$$

The gradient of the function $\varphi_{\texttt{hub}}$ is given by $\nabla \varphi_{\texttt{hub}}(y) = y/\delta$ if $\|y\| \leq \delta$ and $\nabla \varphi_{\texttt{hub}}(y) = y/\|y\|$ for $\|y\| > \delta$.

– Implement the accelerated gradient method (AGM) for the smoothed convex clustering problem (2) with fixed step size and basic extrapolation strategy

$$\alpha_k = \frac{1}{L}, \quad \beta_k = \frac{t_{k-1} - 1}{t_k}, \quad t_k = \frac{1}{2}(1 + \sqrt{1 + 4t_{k-1}^2}), \quad t_{-1} = t_0 = 1.$$

The Lipschitz constant L of $\nabla f_{\texttt{clust}}$ is given by $L = 1 + n\lambda/\delta$. Alternatively, you can also use the following parameter strategies:

$$\alpha_k = \frac{1}{L} \quad \text{and} \quad \beta_k = \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}},$$

where $\mu > 0$ denotes the strong convexity parameter of $f_{\texttt{clust}}$. (This variant of AGM is particularly designed for strongly convex problems).

– Implement the globalized Newton-CG method (Lecture L-09, slide 38) with backtracking that was presented in the lecture to solve the smooth clustering problem (2). (You might first want to implement the standard gradient method as basis of the approach).

You can use standard parameters for the backtracking procedure ($\gamma = 0.1$ or $\gamma = 10^{-2}$, $s = 1$, and $\sigma = \frac{1}{2}$). The maximum number of CG iterations can be set to `cg-max` $= 10$

and as tolerance you can choose $\texttt{cg-tol}_k = \min\{1, \|\nabla f_{\texttt{clust}}(X^k)\|^{0.1}\}\|\nabla f_{\texttt{clust}}(X^k)\|$. Other choices are possible. Notice that the Huber function $\varphi_{\texttt{hub}}$ is not twice differentiable at points y satisfying $\|y\| = \delta$. Instead you can work with the generalized derivative $\tilde{\nabla}^2\varphi_{\texttt{hub}}(y) := \frac{1}{\delta}I$ at those points.

- Run the different methods first on several of the synthetic datasets and compare their performance with respect to the number of iterations and the required cpu-time. Plot and compare the convergence w.r.t. the norm of the gradient $\nabla f_{\texttt{clust}}(X^k)$ or w.r.t. the relative error $|f_{\texttt{clust}}(X^k) - f^*|/\max\{1, |f^*|\}$ where f^* is the best obtained function value. You can choose $\delta \in [10^{-4}, 10^{-1}]$ and $\lambda \in [0.05, 1]$ (other suitable choices are also possible).
- Run (one of) your method(s) for a sequence of regularization parameters $\lambda_1 > \lambda_2 > \cdots > \lambda_\ell > 0$, $\ell \geq 10$, and visualize the obtained clustering results appropriately. When assigning the data points a_i and a_j to the same cluster, you can use the criterion $\|x_i^* - x_j^*\| \leq \varepsilon$ with $\varepsilon \approx 10^{-2}$ (or smaller). How many different clusters do you obtain depending on the choice of λ_i and how do the clusters change? Create suitable plots that show the runtime of your algorithm and the achieved number of clusters for the different choices of λ_i. Discuss your results and observations.

  Note that the solutions obtained for the clustering problem with regularization parameter λ_i can be used as initial point for the next run with λ_{i+1} (or λ_{i-1}).

Hints and Further Guidelines:

- Notice that the data matrix A for the examples given in Table 1 is stored in a `sparse` format (we only save nonzero entries). Make sure that your code does not contain operations that convert this matrix into a `full` matrix (especially when working with some of the large-scale sets).
- Although the optimization variable $X^k \in \mathbb{R}^{d\times n}$ is a matrix, you can treat it as a vector in your algorithms by setting $x = \mathrm{vec}(X) \in \mathbb{R}^{dn}$. Try to avoid long `for loops` in order to evaluate $f_{\texttt{clust}}$ and $\nabla f_{\texttt{clust}}$.
- For debugging, you can test your implementation of the inexact Newton method and the CG method first on simpler and low-dimensional examples.

3. *Weighted Models.* To handle cluster recovery for large-scale data sets, the following weighted convex clustering model modified from (1) can be considered:

$$\min_{X\in\mathbb{R}^{d\times n}} \ \frac{1}{2}\sum_{i=1}^{n}\|x_i - a_i\|^2 + \lambda\sum_{i=1}^{n}\sum_{j=i+1}^{n} w_{ij}\|x_i - x_j\|, \tag{3}$$

where $w_{ij} = w_{ji} \geq 0$ are given weights that are chosen based on the input data A. One can regard the original convex clustering model (1) as a special case of (3), if we take $w_{ij} = 1$ for all $i < j$ in the above weighted convex clustering model. To make the computational cost cheaper when evaluating the regularization terms, one would generally put a non-zero weight only for a pair of points which are nearby each other, and a typical choice of the weights is

$$w_{ij} = \begin{cases} \exp(-\vartheta\|a_i - a_j\|^2) & \text{if } (i,j) \in \mathcal{E}, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where $\mathcal{E} = \bigcup_{i=1}^{n}\{(i,j) : a_j \text{ is among } a_i\text{'s } k\text{-nearest neighbors}, i < j \leq n\}$ and $\vartheta > 0$ is a given positive constant.

– As in part 2.), we want to solve the optimization problem (3) with weights of the form (4) using a Huber-type regularization of the Euclidean norm. Adjust the code of the accelerated gradient method *or* of Newton's method to solve this more general weighted problem. The Lipschitz constant of this new objective function does not change (for general weights the new constants is given by $1 + n\lambda \max_{i,j} |w_{ij}|/\delta$).

– Rerun your code for the data sets tested in part 2.) and report your results. A possible choice of k and ϑ in $\mathcal{E}$ and (4) is given by $k \in \{5, 10\}$ and $\vartheta = 0.5$.

– Give a detailed comparison of your results with the ones obtained in part 2.) – does the weighted clustering model achieve better results and better runtimes?

Hints and Guidelines:

– Many of the parameters and constants mentioned in this and the second part are not fully fixed. Different variants of δ, λ, k, ϑ, ε, or even w_{ij} might yield better or more stable results. For debugging, you can test your implementation first on simpler and low-dimensional examples.

4. *Performance, Extensions, and Stochastic Optimization.* In this final part of the project, we try to investigate additional improvements and variants of the algorithms discussed in the second and third part.

   – Based on your numerical experience, is it possible to further improve the performance of your implementation – i.e., by choosing different linesearch strategies, parameters (within the algorithm), or update rules? Try to revise your code and implement one (some) of the algorithms as efficient as possible. Report your changes and adjustments.

   Potential Key-Words and Ideas: Barzilai-Borwein step sizes; L-BFGS updates or other Hessian approximations; adjust line-search parameters; minimize the number of data-calls and computations involving the data matrix A and the penalization term; more efficient implementations of the cluster penalty term; ...

   – The previous numerical experiments are primarily based on convex Huber-type smoothing techniques for the Euclidean norm. Investigate other types of regularizations, e.g., other norms like the ℓ_1- ($\|\cdot\|_1$) or maximum norm ($\|\cdot\|_\infty$) or other smoothing techniques, like, e.g., the nonconvex `log`-function $\varphi_{\texttt{log}}(y) := \log(1 + \|y\|^2/\nu)$ (for the Euclidean norm). How do these changes affect performance and quality of the recovered clusters? Many other (non)convex variants of the cluster model are possible.

   – Both clustering problems can be expressed as an empirical risk minimization problem of the form

   $$\min_{x \in \mathbb{R}^{dn}} \; f(x) = \sum_{i=1}^{m} f_i(x),$$

   where each $f_i : \mathbb{R}^{dn} \times \mathbb{R} \to \mathbb{R}$ corresponds to our loss model being evaluated at a certain data point $a_i \in \mathbb{R}^n$ or to a cluster penalty term $\|x_i - x_j\|$. If the number of data points n is large or if the dimension dn of the problem is large, the evaluation of the full gradient $\nabla f(x)$ can be very time-consuming and – in some cases – is not even possible. Stochastic gradient methods are based on the idea of using simpler stochastic approximations of the gradient in each iteration. In particular, in each iteration we sample one index i_k from $\{1, ..., m\}$ uniformly at random and perform the update

   $$x^{k+1} = x^k - \alpha_k \nabla f_{i_k}(x^k).$$

Convergence of this simple approach can be guaranteed if the step sizes $\alpha_k = \eta\beta_k$ are diminishing and satisfy $\sum_{k=0}^{\infty} \beta_k = \infty$, $\sum_{k=0}^{\infty} \beta_k^2$, and $\eta > 0$. There are also mini-batch versions, where we select a (larger) subset $\mathcal{S}_k \subset \{1, ..., m\}$ at random and set

$$x^{k+1} = x^k - \frac{\alpha_k}{|\mathcal{S}_k|} \sum_{i \in \mathcal{S}_k} \nabla f_i(x^k).$$

Get familiar with such type of stochastic approach and implement a suitable stochastic gradient-type algorithm for one of the models. Investigate the behavior of the stochastic gradient method on some larger data sets and test different choices of β_k. Compare the performance with the deterministic approaches.

Potential Key-Words and Ideas: Stochastic gradient descent; strong convexity; mini-batch; variance reduction techniques (SVRG); random reshuffling techniques; sketching.

This part of the project is more open and not all of the mentioned points need to be addressed. In particular, other extensions and discussions are possible. Add comments if you have already improved your code and implementations while working on part 2 or 3.

Project Report and Presentation. This project is designed for groups of four or five students. Please send the following information to `217012017@link.cuhk.edu.cn` until **December, 16th, 11:00 pm**:

- Name and student ID number of the participating students in your group, group name.

Please contact the instructor in case your group is smaller to allow adjustments of the project outline and requirements.

A report should be written to summarize the project and to collect and present your different results. The report itself should take no more than 15–20 typed pages plus a possible additional appendix. It should cover the following topics and items:

- What is the project about?
- What have you done in the project? Which algorithmic components have you chosen to implement? What are your main contributions?
- Summarize your main results and observations.
- Describe your conclusions about the different problems and methods that you have studied.

You can organize your report following the outlined structure in this project description. As the different parts in this project only depend very loosely on each other, you can choose to distribute the different tasks and parts among the group members. Please clearly indicate the responsibilities and contributions of each student and mention if you have received help from other groups, the teaching assistant, or the instructor.

Try to be brief, precise and fact-based. Main results should be presented by highly condensed and organized data and not just piles of raw data. To support your summary and conclusions, you can put more selected and organized data into an appendix which is not counted in the page limit. Please use a cover page that shows the names and student ID numbers of your group members.

The deadline for the report submission is **December, 23th, 12:00 pm**. Please submit your report (as pdf-document) and all supporting materials and code online using Blackboard.

The individual presentations of the project are scheduled (tentatively) for **December, 24th**. More information will follow here soon.