Technical Report: On non-convex regularization in machine learning with an application to clinical data

Matthias Lenga[1], *Tim O.F. Conrad*[2]

[1]: Christian-Albrechts-Universität zu Kiel, Institute of Mathematics
[2]: Freie Universitaet Berlin, Institute of Mathematics & Forschungscampus MODAL

Contents

List of Algorithms

1 Non-convex regularization for supervised learning

Motivated by the clinical applications described in [CO15] we test a machine learning algorithm for binary classification with a non-smooth, non-convex regularization function. By renouncing the convexity of the regulating term, we hope to get sparse and jet robust classifiers. We tested the algorithm on proteomics datasets generated by the university hospitals of Leipzig and Heidelberg from [CO15] and on TCGA data on mRNA and miRNA level. These notes are largely expository in nature, with a didactic flavour.

Let $A \in \mathbb{R}^{n\times d}$ denote our data-matrix and $y \in \{-1,1\}^n$ the associated vector of class labels. We consider the optimization problem

$$\min_{(x_0,x)\in\mathbb{R}\times\mathbb{R}^d} \sum_{i=1}^{n} \left(a^{(i)}x + x_0 - y_i\right)^2 + \lambda \sum_{i=1}^{n} \left(|x_i| + \varepsilon\right)^p \tag{1}$$

where $a^{(i)} \in \mathbb{R}^d$ denotes the i-th row of the matrix A and $p, \lambda > 0$ and $\varepsilon \geq 0$. Strictly positive values of ε ensure that even for $0 < p < 1$ the regulation term is Lipschitz-continuous at the origin. Any minimizer (x_0^*, x^*) of (1) must satisfy the first order condition

$$\sum_{i=1}^{n} \left(a^{(i)}x^* + x_0^* - y_i\right) = 0$$

which is equivalent to

$$x_0^* = \overline{y} - \left(\overline{a_1}, ..., \overline{a_d}\right) x^*$$

where $a_i \in \mathbb{R}^n$ denotes the i-th column of A. We center each column a_i by subtracting it's mean $\overline{a_i}$ and replace y_i by $y_i - \overline{y}$ in order to get the intercept-free representation

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^{n} \left(a^{(i)}x - y_i\right)^2 + \lambda \sum_{i=1}^{n} \left(|x_i| + \varepsilon\right)^p \tag{2}$$

of our minimization problem. Here we reused the notation form above for the centered quantities. For $p \geq 1$ this is a convex optimization problem. In the case of ridge regression ($p = 2, \varepsilon = 0$) the vector

$$x^* = \left(A^T A + \lambda I\right)^{-1} Ay$$

is the explicit solution of (2). In the case of Lasso-type shrinkage ($p = 1$) we have to solve a l_1-regularized least squares problem. Even large-scale problems of this type can be solved efficiently, cf. [KI07]. Now we consider the case $0 < p < 1$. Unfortunately the minimization problem is no longer convex and for all choices of $\varepsilon \geq 0$ and $0 < p < 1$ the function

$$x \mapsto \left(\sum_{i=1}^{n} \left(|x_i| + \varepsilon\right)^p\right)^{1/p}$$

is not a proper norm. Nevertheless we want to use functions of the latter type for regularization. We call them with a slight abuse of notation l_p-type penalty terms. A widely used idea is to solve (2) iteratively by replacing the regularization term in each step with a suitable affine majorant which leads to a l_1-regularized least squares problem. To be more particular, in each step we choose

$$x^{k+1} \in \operatorname*{argmin}_{x\in\mathbb{R}^d} \ \|Ax - y\|_2^2 + \lambda \|W^k x\|_1 \tag{3}$$

where

$$W^k := \operatorname{diag}\left[p(|x_1^k| + \varepsilon)^{p-1}, ..., p(|x_d^k| + \varepsilon)^{p-1}\right]$$

depends only om the previous iterate. In the literature this procedure is know as the IRL1 (iteratively reweighted l_1) algorithm. See [CZ14] for a convergence analysis and more details. The first crucial structural property is that the regulator can be decomposed in a concave, increasing function

$G : \mathbb{R}^d_+ \to \mathbb{R}_+$ and the mapping $x \mapsto (|x_1|, ..., |x_d|)$. The second important characteristic is that the quadratic loss term is a convex function which converges to infinity as $\|x\| \to \infty$. Based on these observations the IRL1 algorithm was generalized in [OC13] in order to handle large classes of loss and regulation functions as well as linear constraints. Further analysis and numerical results can be found in [OC15].

We want to outline some ideas from [OC13] and [CZ14]. We dispense with the full generality case for the sake of simplicity. Let $F : \mathbb{R}^d \to \mathbb{R}_+$ be a convex, coercive function and $G : \mathbb{R}^d_+ \to \mathbb{R}_+$ be concave and componentwise increasing. For any vector $x \in \mathbb{R}^d$ we denote by $|x|$ the componentwise absolute value and $\Pi_j x = x_j$ the projection to the j-th component. Moreover we write $\partial h(x)$ for the subdifferential of a convex function h at the point x. The superdifferential of a concave function is denoted by $\hat{\partial}$. The minimization task (2) generalizes to

$$\min_{x \in \mathbb{R}^d} F(x) + G(|x|) \tag{4}$$

and the iteration step (3) becomes

$$x^{k+1} \in \operatorname*{argmin}_{x \in \mathbb{R}^d} \; F(x) + \|W^k x\|_1 \tag{5}$$

where $W^k := \operatorname{diag}\left(\omega_1^k, ..., \omega_d^k\right)$ and $\omega^k \in \hat{\partial} G(|x^k|)$ is arbitrarily chosen. Due to the fact that the function G is increasing, every element in $\hat{\partial} G$ has nonnegative components. First we show that the sequence $F(x^k) + G(|x^k|)$ decreases monotonically as $k \in \mathbb{N}$ tends to infinity. Assume that x^{k+1} minimizes the convex function $x \mapsto F(x) + \|W^k x\|_1$. From [BP12] Theorem 3.57 we know that

$$0 \in \partial F(x^{k+1}) + \partial \|W^k x^{k+1}\|_1$$

is true. This means we can pick $-a^{k+1} \in \partial F(x^{k+1})$ such that $a^{k+1} \in \partial \|W^k x^{k+1}\|_1$. Moreover we find that

$$\partial \|W^k x^{k+1}\|_1 = \sum_{i=1}^d \omega_i^k \partial |\Pi_i x^{k+1}|$$

is true. Hence there exist a vector $\alpha^{k+1} \in [-1, 1]^d$ with $a^{k+1} = W^k \alpha^{k+1}$ and $\alpha_i^{k+1} x_i^{k+1} = |x_i^{k+1}|$. Now we can conclude:

$$\begin{aligned}
&F(x^k) - F(x^{k+1}) + G(|x^k|) - G(|x^{k+1}|) \\
&\geq \langle -a^{k+1}, x^k - x^{k+1} \rangle + \langle \omega^k, |x^k| - |x^{k+1}| \rangle \\
&= -\langle W^k \alpha^{k+1}, x^k - x^{k+1} \rangle + \langle \omega^k, |x^k| - |x^{k+1}| \rangle \\
&= -\sum_{i=1}^d \omega_i^k \alpha_i^{k+1} \left(x_i^k - x_i^{k+1}\right) + \sum_{i=1}^d \omega_i^k \left(|x_i^k| - |x_i^{k+1}|\right) \\
&= -\sum_{i=1}^d \omega_i^k \alpha_i^{k+1} x_i^k + \sum_{i=1}^d \omega_i^k |x_i^{k+1}| + \sum_{i=1}^d \omega_i^k |x_i^k| - \sum_{i=1}^d \omega_i^k |x_i^{k+1}| \\
&= \sum_{i=1}^d \omega_i^k \left(|x_i^k| - \alpha_i^{k+1} x_i^k\right) \geq 0
\end{aligned}$$

This proves the assertion. Due to the fact that $F + G \geq 0$ we conclude that the sequence $F(x^k) + G(|x^k|)$ converges. Moreover we assumed that $F(x) \to \infty$ as $\|x\| \to \infty$. This implies that x^k is bounded which allows us to extract a convergent subsequence. Under some regularity assumptions one can show that every accumulation point of x^k is a stationary point of the target functional, cf. [OC13]. Moreover it was proven that under further regularity assumptions the whole sequence converges. We refer the reader to [OC15] for more details. This motivates to use $|F(x^k) + G(|x^k|) - F(x^{k+1}) - G(|x^{k+1}|)| < \delta$ for some $\delta > 0$ in combination with a maximal iteration number as a break criterion for the IRL1 loop (5). Algorithm 1 contains the detailed procedure. Choosing $p = 1$ leads to the classic lasso penalization

Algorithm 1 l_p-regularized supervised learning

INPUT:
$A \in \mathbb{R}^{n\times d}$ data matrix
$y \in \mathbb{R}^n$ vector of class labels
$p \in (0,1]$ concavity parameter
$\lambda \in (0,\infty)$ impact parameter
$\varepsilon \in [0,\infty)$ adjustment of Lipschitz-constant at origin, need $\varepsilon > 0$ if $p < 1$
$\delta \in (0,\infty)$ precision for break criterion in the IRL1 loop
MaxIt $\in \mathbb{N}$ maximal number of iterations in the IRL1 loop
F sufficiently regular loss function, e.g.

- quadratic loss: $\|Ax + x_0 - y\|_2$
- Hinge loss: $\sum_{i=1}^n \left(1 - y_i(a^{(i)}x + x_0)\right)_+$

OUTPUT:
$(x_0^*, x^*) \in \mathbb{R} \times \mathbb{R}^d$ classifier

function LPSL

$\Phi(\omega_0, \omega) := F(\omega_0, \omega) + \lambda \sum_{j=1}^d (|\omega_j| + \varepsilon)^p$

$(x_0, x) \leftarrow$ starting point from $\mathbb{R} \times \mathbb{R}^d$, e.g. randomly chosen

for $k = 1$ to MaxIt **do**
$W \leftarrow$ diag $\left[p(|x_1| + \varepsilon)^{p-1}, ..., p(|x_N| + \varepsilon)^{p-1}\right]$

$$\begin{aligned} &\text{minimize} && F(\omega_0, \omega) + \lambda \|W\omega\|_1 \\ &\text{subject to} && (\omega_0, \omega) \in \mathbb{R} \times \mathbb{R}^d \end{aligned}$$

gap $\leftarrow |\Phi(x_0, x) - \Phi(\omega_0^*, \omega^*)|$
$(x_0, x) \leftarrow (\omega_0^*, \omega^*)$
if gap $< \delta$ **or** $p == 1$ **then** break
end

return (x_0, x)

end

1.1 Numerical results for dataset UHL with quadratic loss

The IRL1 algorithm was applied to generate classifiers from the dataset provided by the university hospital Leipzig (UHL). For our computations we chose Matlab R2010b with the CVX 2.1 optimization package and the solver Mosek 7.1. Due to the specific properties of the CVX suite it was convenient to use the l_p-penalized l_2-norm

$$\min_{x\in\mathbb{R}^d} \|Ax - y\|_2 + \lambda \sum_{i=1}^n (|x_i| + 0.001)^p \tag{6}$$

instead of a penalized sum of squares as in (2). Furthermore all components of the classifiers smaller than 10^{-5} in absolute value were set to zero. The UHL dataset contains protein data of length 42390 generated by mass spectrometry from 75 patients with pancreatic cancer and an equally sized control group. In this case we used 10-fold cross validation in order to calibrate the impact parameter λ and

the concavity parameter p. The column *size* indicates the number of non-zero components of the classifier in the last cross-validation round. The column *CV-error* contains the average percentage of misclassification on the test data in the ten cross-validation rounds.

p	λ	size	CV-err
0.90	110	61	0.00 %
0.90	145	44	0.00 %
0.90	180	38	0.67 %
0.90	200	42	0.67 %
0.90	225	28	0.67 %
0.90	250	31	2.00 %
0.85	110	41	0.00 %
0.85	145	38	0.67 %
0.85	180	32	0.67 %
0.85	200	30	1.33 %
0.85	225	25	1.33 %
0.85	250	24	1.33 %
0.80	110	37	0.67 %
0.80	145	24	1.33 %
0.80	180	23	0.67 %
0.80	200	19	1.33 %
0.80	225	18	2.00 %
0.80	250	17	2.67 %
0.75	110	25	2.00 %
0.75	145	20	1.33 %
0.75	180	18	1.33 %
0.75	**200**	**15**	**1.33 %**
0.75	225	13	2.00 %
0.75	250	13	2.67 %
0.70	110	20	2.00 %
0.70	145	15	1.33 %
0.70	180	14	3.33 %
0.70	200	12	4.00 %
0.70	225	10	4.67 %
0.70	250	10	4.67 %
0.65	110	15	1.33 %
0.65	145	12	2.00 %
0.65	180	11	4.67 %
0.65	200	12	6.00 %
0.65	225	8	6.67 %
0.65	250	8	6.67 %
0.60	110	11	2.00 %
0.60	145	11	4.67 %
0.60	180	9	6.00 %
0.60	200	8	6.67 %
0.60	225	7	8.67 %
0.60	250	6	8.00 %
0.50	70	14	3.33 %
0.50	110	9	6.67 %

Table 1: parameter selection via 10-fold CV for UHL data with l_p-regularized quadratic loss

For good parameter choices our algorithm generates sparse feature vectors at low error rates. We visualize the content of the table with a plot of the *size* and *CV-err* values as functions of the parameters p and λ. Please note that the p-λ-axis directions in figure one and two are reversed.

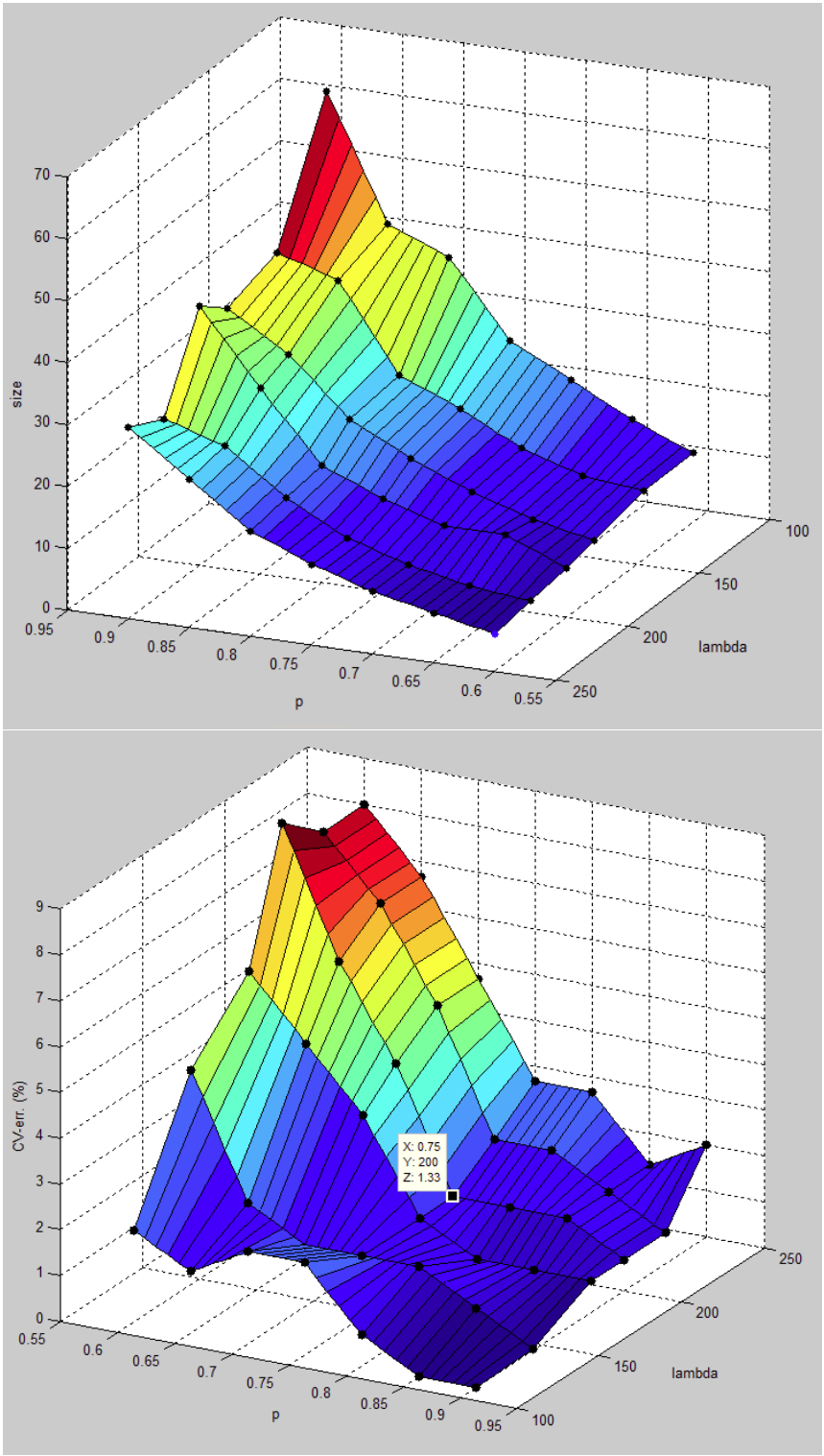

Figure 1: size (top) and CV-error in % (bottom) as functions of p and λ l_p-regularized quadratic loss minimization, dataset UHL

Motivated by the results above we chose $p = 0.75$, $\lambda = 200$ and computed a classifier using the whole dataset as training data. We use a thresholding level of 10^{-5}. Thresholding at a level of 10^{-3} led to

classifiers with the same size as in [CO15].

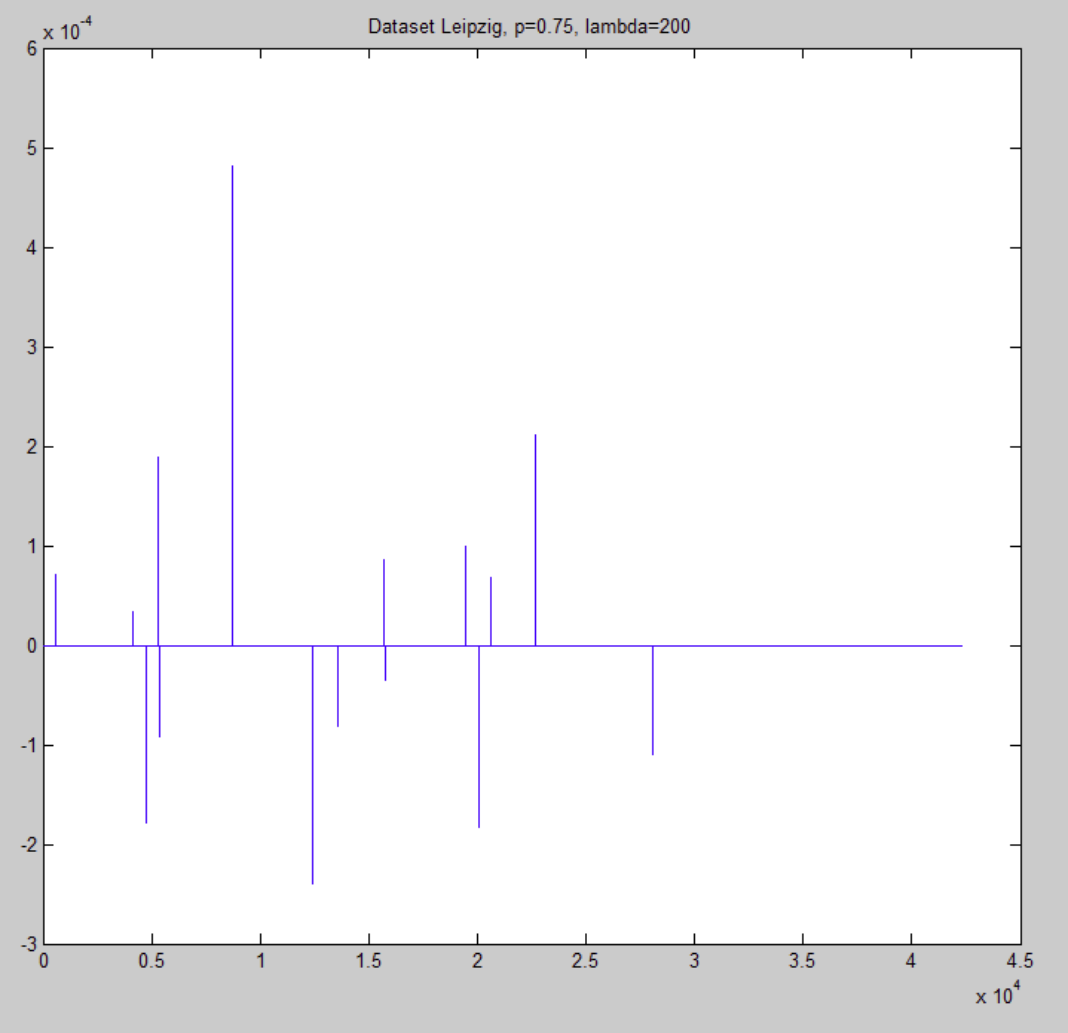

Figure 2: classifier for $p = 0.75$, $\lambda = 200$ obtained via l_p-regularized quadratic loss minimization from the dataset UH Leipzig

Next, the IRL1 algorithm was applied to learn with quadratic loss from the dataset provided by the university hospital Heidelberg (UHH). The UHH dataset contains protein spectra of length 42390 from 144 patients. In this case we used 9-fold cross validation in order to find good choices of λ and p.

p	λ	size	CV-err
0.75	200	20	1.25%
0.75	225	20	1.25%

Table 2: parameter selection via 9-fold CV for UHH data with l_p-regularized quadratic loss

1.2 Numerical results for dataset UHL with the Hinge-loss

The IRL1 algorithm was also applied to learn from the UHL dataset with a l_p-regularized Hinge-loss. This means we solved the following non-convex optimization problem:

$$\min_{(x_0,x)\in\mathbb{R}\times\mathbb{R}^d} \sum_{i=1}^{n} \Big(1 - y_i(a^{(i)}x + x_0)\Big)_+ + \lambda \sum_{i=1}^{n} (|x_i| + 0.001)^p \tag{7}$$

Figure 3 contains the plot of a typical classifier for $\lambda = 180$, $p = 0.65$ and a thresholding level of 10^{-5}. In both cases our classifiers select sparse feature subsets and archive low CV-error rates.

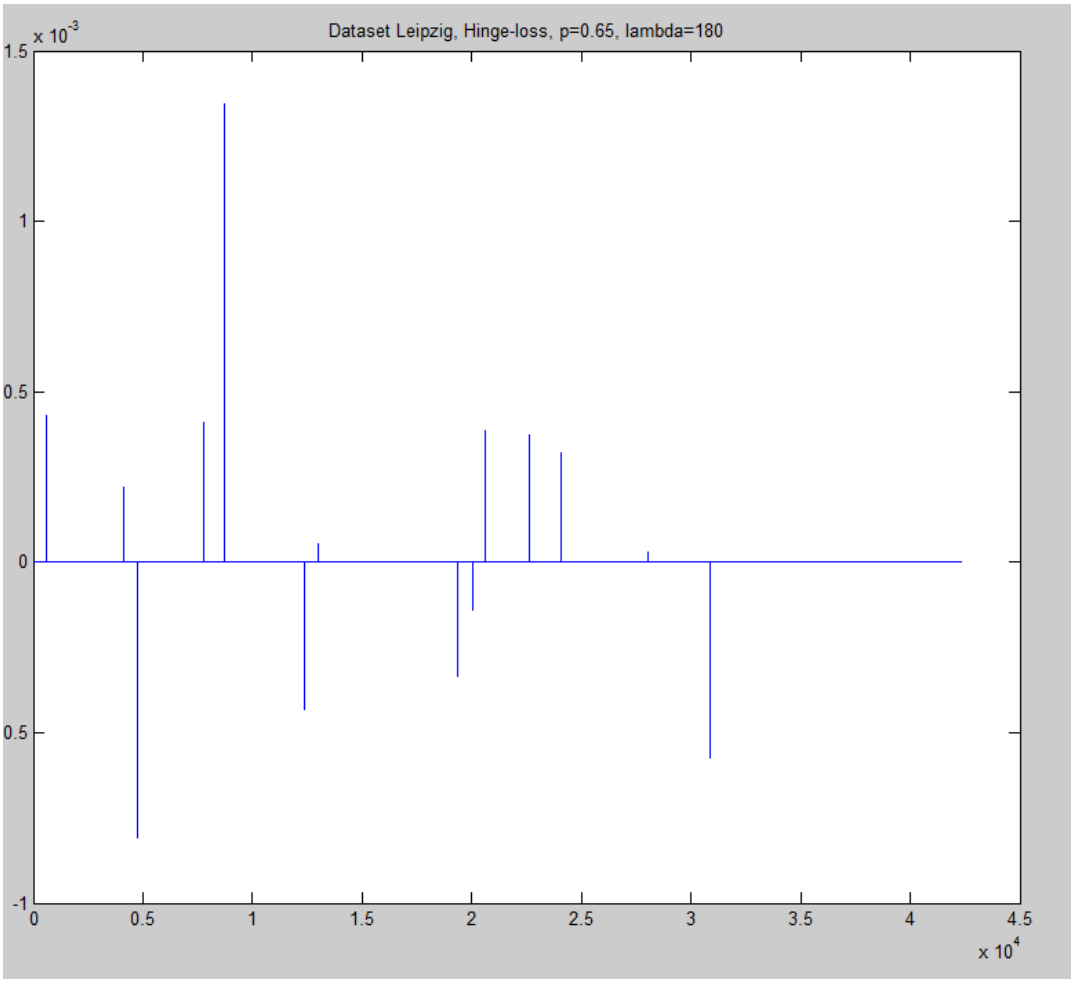

Figure 3: classifier for $p = 0.65$, $\lambda = 180$ obtained via l_p-regularized Hinge-loss minimization from the dataset UH Leipzig

For the Hinge-loss we generated a table which depicts the average optimizer-size and CV-error dependencies on the impact and concavity parameter λ and p. Again, by *CV-error* we mean the average percentage of misclassification on the test data in the ten cross-validation rounds.

p	λ	average size	CV-err
0.90	110	16.8	0.67 %
0.90	145	17.0	0.67 %
0.90	180	17.1	0.67 %
0.90	200	16.8	0.67 %
0.90	225	16.5	1.33 %
0.90	250	17.3	0.67 %
0.85	110	15.9	0.00 %
0.85	145	16.1	0.67 %
0.85	180	16.8	0.67 %
0.85	200	16.2	0.67 %
0.85	225	16.5	1.33 %
0.85	250	16.5	0.67 %
0.80	110	16.6	0.00 %
0.80	145	15.7	0.67 %
0.80	180	15.9	0.67 %
0.80	200	16.0	0.67 %
0.80	225	16.0	1.33 %
0.80	250	16.5	0.67 %
0.75	110	15.6	0.00 %
0.75	145	16.2	0.67 %
0.75	180	15.1	0.67 %
0.75	200	15.4	0.67 %
0.75	225	15.8	1.33 %
0.75	250	16.2	0.67 %

p	λ	average size	CV-err
0.70	110	15.7	0.67 %
0.70	145	15.3	0.67 %
0.70	180	14.9	0.67 %
0.70	200	15.2	0.67 %
0.70	225	15.2	2.00 %
0.70	250	15.8	1.33 %
0.65	110	15.5	0.67 %
0.65	145	15.0	0.67 %
0.65	**180**	**14.7**	**0.67 %**
0.65	200	14.3	1.33 %
0.65	225	14.8	2.00 %
0.65	250	14.6	2.00 %
0.60	110	15.0	0.00 %
0.60	145	14.8	0.67 %
0.60	180	14.9	1.33 %
0.60	200	14.5	1.33 %
0.60	225	13.6	2.67 %
0.60	250	13.9	2.67 %
0.55	110	14.6	0.67 %
0.55	145	14.9	0.67 %
0.55	180	13.6	1.33 %
0.55	200	13.3	1.33 %
0.55	225	12.9	2.67 %
0.55	250	12.5	2.67 %

Table 2: parameter selection via 10-fold CV for UHL data with l_p-regularized Hinge-loss

Similar to figure 1 we visualize the content of the latter table with a plot, cf. figure 4. In figure 1 we observe that the size of the optimizer grows super-linear for $p \to 1$ and decreasing impact parameter λ. For the Hinge-loss, at least from a qualitative point of view, the size of the optimal element behaves differently. Figure 4 shows that the size of the optimizer grows rather logarithmically as impact and concavity decreases. Summarizing table 1 and table 2 we conclude that the l_p-regularized Hinge-loss as well as l_p-regularized quadratic loss produces sparse classifiers with low error rates for our dataset. The Hinge-loss outperforms the l_p-penalized quadratic and is less sensitive to changes in the parameters p and λ.

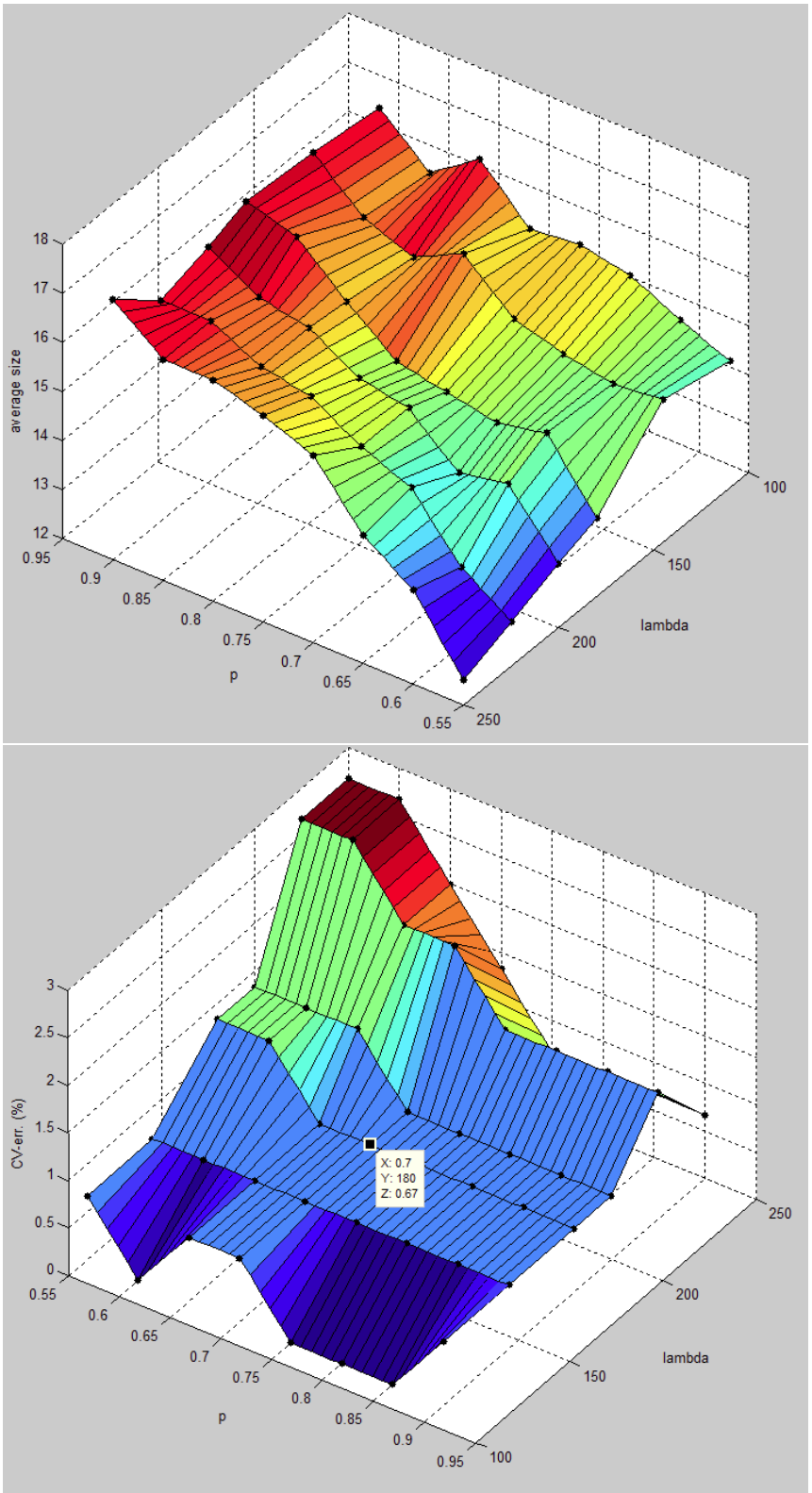

Figure 4: average size (top) and CV-error in % (bottom) as functions of p, λ l_p-regularized Hinge-loss minimization, dataset UHL

1.3 Numerical results for miRNA data from TCGA

We applied the l_p-regularized supervised learning method to breast carcinoma (BRCA) data[1] and lung adenocarcinoma (LUAD) data[2] on miRNA level from the TCGA database. We combined the BCGSC IlluminaHiSeq_miRNASeq data of 457 BRCA patients with the BCGSC IlluminaHiSeq_miRNASeq data of 457 LUAD patients. Afterwards we added the the class labels -1 and 1 for BRCA and LUAD, respectively. This resulted in a data matrix $X \in \mathbb{R}^{914\times 1046}$ and a label vector $y \in \{-1,1\}^{914}$. Again we chose Matlab R2010b with the CVX 2.1 optimization package and the solver Mosek 7.1 for our computations. The l_p-penalized l_2-norm

$$x \mapsto \|Ax - y\|_2 + \lambda \sum_{i=1}^{n} (|x_i| + 0.001)^p$$

was used to learn from the data. We used 10-fold cross-validation in order to assess the quality of our classifiers. In each cross-validation round we calculated a classifier ω^* and the following performance measures:

- The size of the classifier ω^*.
- The AIC value based on the test set: $\mathrm{AIC}_{\mathrm{te}} = N_{te} \log \mathrm{MSE}_{\mathrm{te}}(\omega^*) + 2\ \mathrm{size}(\omega^*)$
- The AIC value based on the training set: $\mathrm{AIC}_{\mathrm{tr}} = N_{tr} \log \mathrm{MSE}_{\mathrm{tr}}(\omega^*) + 2\ \mathrm{size}(\omega^*)$

By $\mathrm{MSE}_{\mathrm{te}}(\omega^*)$ and $\mathrm{MSE}_{\mathrm{tr}}(\omega^*)$ we denote the mean-squared error of ω^* on the test and training data, respectively. Moreover $\mathrm{size}(\omega^*)$ refers to the number of non-zero entries of ω^*. The sizes of the test and training set are denoted by N_{te} and N_{tr}. By averaging the quantities from the different cross-validation rounds, we gain overall performance measures. These were generated for different choices of p and λ. Our results can be found in the table below.

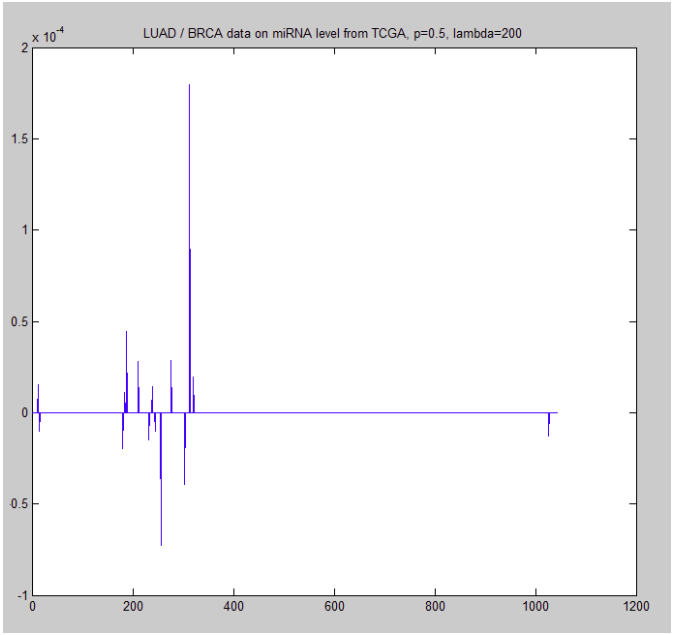

Figure 5: classifier for $p = 0.5$, $\lambda = 200$ obtained via l_p-regularized quadratic loss minimization from LUAD/BRCA data on miRNA level

[1] https://tcga-data.nci.nih.gov/tcga/dataAccessMatrix.htm?mode=ApplyFilter&diseaseType=BRCA
[2] https://tcga-data.nci.nih.gov/tcga/dataAccessMatrix.htm?mode=ApplyFilter&diseaseType=LUAD

p	λ	average size	CV-error	average $\mathrm{AIC}_{\mathrm{te}}$	average $\mathrm{AIC}_{\mathrm{tr}}$
0.4	125	13.6	5.49%	-46.87	-680.11
0.4	150	12	6.81%	-47.20	-652.73
0.4	175	10.2	6.81%	-49.55	-626.53
0.4	200	9.1	7.69%	-46.86	-590.37
0.4	225	8.1	8.79%	-46.44	-561.76
0.4	250	7.8	10.11%	-43.33	-537.51
0.4	275	7.6	10.76%	-40.79	-502.89
0.4	300	6.6	12.41%	-39.03	-466.9
0.5	125	19.7	4.39%	-46.5	-779.66
0.5	150	16.8	4.39%	-50.03	-749.27
0.5	175	14.9	4.50%	-49.28	-716.97
0.5	**200**	**14.0**	**4.83%**	**-48.91**	**-691.56**
0.5	225	12.7	6.26%	-45.99	-662.50
0.5	250	11.7	7.47%	-47.49	-646.11
0.5	275	10.5	6.92%	-48.50	-624.62
0.5	300	9.7	7.80%	-46.29	-597.57
0.6	125	25.9	2.74%	-44.89	-862.60
0.6	150	23.1	3.73%	-44.32	-823.85
0.6	175	22.0	3.73%	-46.95	-803.48
0.6	200	20.0	4.06%	-47.38	-783.55
0.6	225	17.7	3.51%	-49.82	-761.91
0.6	250	16.3	4.50%	-49.56	-746.07
0.6	275	15.4	4.83%	-49.11	-729.97
0.6	300	14.5	5.82%	-48.02	-706.00
0.7	125	35.1	2.63%	-32.26	-915.36
0.7	150	31.0	2.63%	-38.54	-901.29
0.7	175	29.0	2.96%	-39.83	-885.49
0.7	200	26.1	2.63%	-44.22	-864.52
0.7	225	24.6	3.62%	-45.15	-848.28
0.7	250	23.4	3.08%	-46.47	-832.97
0.7	275	22.1	3.40%	-46.60	-816.34
0.7	300	21.4	3.95%	-45.92	-806.35

Table 3: learning from the BCGSC IlluminaHiSeq_miRNASeq combined BRCA/LUAD data with l_p-regularized l_2-norm

1.4 Numerical results for mRNA data from TCGA

We applied the l_p-regularized supervised learning method to breast carcinoma (BRCA) data[3] and lung adenocarcinoma (LUAD) data[4] on mRNA level from the TCGA database. For each cancer type we chose 516 UNC IlluminaHiSeq_RNASeqV2 datasets and prepared them in the same manner as the datasets from section 1.3. This resulted in a data matrix $X \in \mathbb{R}^{1032\times 20502}$ and a label vector $y \in \{-1,1\}^{1032}$. We used the l_p penalized quadratic loss function

$$x \mapsto \|Ax-y\|_2 + \lambda \sum_{i=1}^{n} (|x_i| + 0.001)^p$$

to learn from the combined mRNA data of 400 LUAD and 400 BRCA patients. The test error was calculated on the data of the the remaining 116 LUAD and 116 BRCA patients.

p	λ	size	test-err.	$\mathrm{AIC}_{\mathrm{tr}}$
0.3	370	7	8.6957%	-511.7719
0.3	400	8	10.4348%	-539.7564
0.3	430	8	10.4348%	-516.5935
0.3	460	7	10.4348%	-590.5172
0.35	370	13	4.3478%	-671.3513
0.35	400	12	6.087%	-659.8359
0.35	430	11	6.087%	-617.1605
0.35	460	8	9.5652%	-533.0436
0.4	370	19	0.86957%	-883.1419
0.4	400	18	1.7391%	-857.786
0.4	430	16	2.6087%	-792.1976
0.4	460	15	2.6087%	-756.8803
0.45	370	24	0%	-999.3025
0.45	400	23	0%	-998.952
0.45	430	20	0.86957%	-925.4912
0.45	460	20	0.86957%	-911.4062
0.5	370	27	0%	-1064.833
0.5	400	27	0%	-1044.3473
0.5	430	27	0%	-1040.0019
0.5	460	25	0%	-989.3306
0.55	370	43	0%	-1242.8838
0.55	400	38	0%	-1189.9164
0.55	430	35	0%	-1148.3962
0.55	460	31	0%	-1090.0247
0.6	370	59	0%	-1400.2854
0.6	400	57	0%	-1355.8678
0.6	430	52	0%	-1347.6207
0.6	460	49	0%	-1310.6378

Table 4: learning from the UNC IlluminaHiSeq_RNASeqV2 combined BRCA/LUAD data with l_p-regularized l_2-norm

[3] https://tcga-data.nci.nih.gov/tcga/dataAccessMatrix.htm?mode=ApplyFilter&diseaseType=BRCA
[4] https://tcga-data.nci.nih.gov/tcga/dataAccessMatrix.htm?mode=ApplyFilter&diseaseType=LUAD

2 Spectral clustering

In this section we want to give a brief review of spectral clustering methods. Please see [LU07] and the references therein for a broader overview. Let $x_1, ..., x_n \in \mathbb{R}^d$ be a set of vectors whose similarity is expressed by a positive, symmetric matrix $W \in \mathbb{R}^{n \times n}$. This means W_{ij} is a quantitative measure for the similarity of the points x_i and x_j. Furthermore we make the natural assumption that $W_{ii} = 1$ for all $i \in \{1, ..., n\}$. Some prominent prototypes for similarity matrices can be found in the table below.

name	weight
Laplace kernel	$W_{ij} = \exp\left(-c\|x_i - x_y\|_2\right)$
Gaussian kernel	$W_{ij} = \exp\left(-c\|x_i - x_y\|_2^2\right)$
ε-neighborhood	$W_{ij} = 1$ if $\|x_i - x_j\| < \varepsilon$ and $W_{ij} = 0$ otherwise
mutual k-N-N	$W_{ij} = 1$ if $x_i \in \mathcal{N}_k(x_j) \wedge x_j \in \mathcal{N}_k(x_i)$ and $W_{ij} = 0$ otherwise

Table 3: important similarity matrices

Here $c > 0$ is a fixed constant which allows us to adjusts the similarity decay and $\|\cdot\|$ is a norm on the $\mathbb{R}^d$. By $\mathcal{N}_k(x_i)$ we denote the the set which contains the k elements of $\{x_1, ..., x_n\}$ which have the smallest distance to x_i with respect to some metric. The entries of Laplace and Gauss-type similarity matrices are strictly positive. Sometimes it is convenient to threshold their values by setting all entries $W_{ij} < \varepsilon$ equal to zero.

The undirected graph G with vertices $V := \{x_1, ..., x_n\}$ and adjacency matrix W is called similarity graph. We want to cluster $x_1, ..., x_n$ into disjoint groups such that edges between all nodes within a group have large weights and edges connecting two different groups have low weights. The graph's degree matrix D and Laplacian L are defined by:

$$\begin{aligned} D &:= \operatorname{diag}\left(\sum_{j=1}^{n} W_{1j}, ..., \sum_{j=1}^{n} W_{nj}\right) \\ L &:= D - W \end{aligned}$$

For all $v \in \mathbb{R}^d$ we have

$$\begin{aligned} v^T L v &= \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij} v_i^2 - \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij} v_i v_y \\ &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}(v_i - v_j)^2 \geq 0 \end{aligned}$$

and thus L is a positive semi-definite matrix with non-negative real eigenvalues. The vector $\mathbb{1} = (1, ..., 1)$ is obviously an eigenvector for the eigenvalue 0. We want to show that the multiplicity of the eigenvalue 0 is equal to the number of connected components of G. Assume that our graph is connected and that v is a non-zero element of L's kernel. For any spanning tree T of G we find that

$$0 \leq \sum_{\{x_i, x_j\} \in T} W_{ij}(v_i - v_j)^2 \leq 2 v^T L v = 0$$

is true. Due to the fact that $W_{ij} > 0$ for all edges $\{x_i, x_j\}$ of T we conclude that $v_1 = v_2 = ... = v_n$ and thus the eigenspace of 0 is one-dimensional. Now assume that G has m connected components $G_1, ..., G_m$. We can relabel the vectors $x_1, ..., x_n$ such that L is m-block diagonal.

$$L = \begin{pmatrix} L_1 & & & \\ & L_2 & & \\ & & \ddots & \\ & & & L_m \end{pmatrix}$$

The matrix $L_i \in \mathbb{R}^{|G_i|\times|G_i|}$ is the Laplacian of the sub-graph G_i. We write $\mathbb{1}_{L_i}$ for the vector with ones in the coordinates of the block L_i and zeros elsewhere. The arguments above can be applied to each sub-graph separately which yields that the kernel of L is spanned by $\mathbb{1}_{L_1}, ..., \mathbb{1}_{L_m}$. We conclude that $\dim\ker(L) = m$ and that the kernel basis vectors serve as indicators for the connection components. This gives us a method to cluster the data $x_1, ..., x_n$ in it's similarity equivalence classes.

Using the Laplace or Gaussian similarity measure without thresholding yields complete similarity graphs. Thus clustering the data into its connected components is meaningless. A common way to formulate our clustering task is the following: for $M \in \mathbb{N}$ fixed find a partition $V = C_1 \cup ... \cup C_M$ such that the *ratio cut* functional

$$\mathrm{RC}\,(C_1, ..., C_M) := \sum_{l=1}^{M} \frac{\sum_{x_i \in C_l} \sum_{x_j \in V\setminus C_l} W_{ij}}{|C_l|}$$

is minimal. The numerator reflects our wish that large edge weights between nodes of different clusters should be penalized. The denominator rewards balanced cluster sizes and penalizes singletons. We start with the case $M = 2$ in order to get a better understanding of the functional's structure. For every partition $V = C_1 \cup C_2$ we have:

$$\begin{aligned}
\mathrm{RC}\,(C_1, C_2) &= \frac{1}{|C_1|} \sum_{x_i \in C_1} \sum_{x_j \in C_2} W_{ij} + \frac{1}{|C_2|} \sum_{x_i \in C_1} \sum_{x_j \in C_2} W_{ij} \\
&= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} \left(y_{C_1,i} - y_{C_1,j}\right)^2 + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} \left(y_{C_2,i} - y_{C_2,j}\right)^2 \\
&= y_{C_1}^T L y_{C_1} + y_{C_2}^T L y_{C_2} \\
&= \mathrm{tr}\left(Y^T L Y\right)
\end{aligned}$$

where $y_{C_k,i} := 1/\sqrt{|C_k|}$ if the node x_i is contained in the set C_k and $y_{C_k,i} := 0$ otherwise and $Y := (y_{C_1}, y_{C_2}) \in \mathbb{R}^{n\times 2}$. Let us note that for each choice of $C_1 \subset V$ we get a unique matrix Y. Obviously the vectors y_{C_1} and y_{C_2} are orthonormal. Thus we conclude that the minimization problem of the ratio cut functional can be reformulated as a discrete optimization problem:

$$\begin{aligned}
&\text{minimize} && \mathrm{tr}\left(Y^T L Y\right) \\
&\text{subject to} && Y^T Y = I_{2\times 2} \\
&&& Y = (y_{C_1}, y_{C_2}) \\
&&& C_1 \in 2^V \setminus \{\emptyset, V\}
\end{aligned} \tag{8}$$

By dropping the last constraint we get the following surrogate convex program:

$$\begin{aligned}
&\text{minimize} && \mathrm{tr}\left(Y^T L Y\right) \\
&\text{subject to} && Y^T Y = I_{2\times 2} \\
&&& Y \in \mathbb{R}^{n\times 2}
\end{aligned} \tag{9}$$

The explicit solution can be easily calculated by diagonalization. Let $L = U\,\mathrm{diag}(\lambda_1, ..., \lambda_{n-1}, 0)U^T$ be the spectral decomposition of L where $U = (u_1, ..., u_{n-1}, \mathbb{1}/\sqrt{n})$ is a basis of orthonormal eigenvectors associated to the eigenvalues $\lambda_1 \geq ... \geq \lambda_n = 0$. Due to the fact that $U^T Y$ is a orthonormal matrix, the minimization problem is equivalent to:

$$\begin{aligned}
&\text{minimize} && \mathrm{tr}\left(Y^T\,\mathrm{diag}(\lambda_1, ..., \lambda_n) Y\right) \\
&\text{subject to} && Y^T Y = I_{2\times 2} \\
&&& Y \in \mathbb{R}^{n\times 2}
\end{aligned}$$

A straightforward calculation shows that

$$\mathrm{tr}\left(Y^T\,\mathrm{diag}(\lambda_1, ..., \lambda_n) Y\right) = \sum_{i=1}^{n} \lambda_i Y_{i,1}^2 + \sum_{i=1}^{n} \lambda_i Y_{i,2}^2$$

is true. Keeping the orthogonality relation $Y^TY = I_{2\times 2}$ and $\lambda_n = 0$ in mind, we find that our latter minimization task is solved by $Y = (e_{n_1}, e_n)$. Thus the solution of (9) is given by $Y^* = (u_{n-1}, \mathbb{1}/\sqrt{n})$ with the optimal value λ_{n-1}. Based on the information contained in Y^* we want to reconstruct an approximate solution for (8). Obviously no problem-specific clustering information is contained in the second column of Y^*. We apply the K-means algorithm (cf. [HTF9] section 14.3.6) to the entries of the vector u_{n-1} in order to get a partition $D_1 \cup D_2 = \{1, ..., n\}$. As clustering rule for our data we chose $x_i \in C_j$ iff $\pi_i(u_{n-1}) \in D_j$ for $j = 1, 2$ and $i = 1, ..., n$. The the general case $M \geq 2$ can be treated along the same lines. First we rewrite the ratio cut function in trace form

$$\begin{aligned}
\mathrm{RC}\,(C_1, ..., C_M) &= \sum_{l=1}^{M} \frac{1}{|C_l|} \sum_{x_i \in C_1} \sum_{x_j \in V \setminus C_l} W_{ij} \\
&= \sum_{l=1}^{M} y_{C_l}^T L y_{C_l} \\
&= \mathrm{tr}\left(Y^T L Y\right)
\end{aligned}$$

with y_{C_l} defined as above. This leads to a discrete minimization problem similar to (8) which is again approximated by a convex surrogate:

$$\begin{aligned}
\text{minimize} \quad & \mathrm{tr}\left(Y^T L Y\right) \\
\text{subject to} \quad & Y^T Y = I_{M\times M} \\
& Y \in \mathbb{R}^{n\times M}
\end{aligned}$$

With the same diagonalization trick we see that the solution of the latter program is given by $Y^* = (u_{n-M+1}, ..., u_{n-1}, \mathbb{1}/\sqrt{n})$. Afterwards we apply the K-means algorithm to the rows of Y^* in order to generate a partition $D_1 \cup ... \cup D_M$ of the set $\{1, ..., n\}$. Then we cluster our original data $x_1, ..., x_n$ according to the rule $x_i \in C_j$ iff $\pi_i(u_{n-1}) \in D_j$ for $j = 1, ..., M$ and $i = 1, ..., n$. This clustering technique is widely used in practice. Using other partition score functions than the ratio cut leads to alternative algorithms, cf. the discussion in [LU07] and [NJW2]. We want to make an important concluding remark. The derivation of the approximate convex problem followed by the clustering of the data based on the problem's solution does not provide any bounds for the quality of obtained partition. In [GM98] it was pointed out that the convex approximation approach can lead to a severely suboptimal clustering of the data.

3 Unsupervised learning via subspace clustering

Let $S_1, ..., S_n \subset \mathbb{R}^d$ be a family of linear subspaces with $\dim\left(\bigoplus_{i=1}^n S_i\right) = \sum_{i=1}^n \dim(S_i)$. From each subspace S_i we have $N_i \geq \dim(S_i)$ data points $Y_i \in \mathbb{R}^{d \times N_i}$. Assume that each matrix Y_i contains enough information in order to reconstruct the associated subspace S_i. This means we require that $\operatorname{rank}(Y_i) = \dim(S_i)$ is true for all $i = 1, ..., n$. We define $N := N_1 + ... + N_n$ and $Y := (Y_1, ..., Y_n) \in R^{d \times N}$ and fix an index $i \in \{1, ..., n\}$. For every $y \in S_i$ we can obviously find a vector $x = (x_1, ..., x_n) \in \mathbb{R}^{N_1 + ... + N_n}$ with $x_j = 0$ for all $i \neq j$ such that $y = Yx = Y_i x_i$ is true. We call x a block sparse solution of the linear system. We follow closely the ideas from [EV09]. There it was shown that block sparse solutions can be obtained by solving the following convex optimization problem:

$$\begin{array}{ll} \text{minimize} & \|\omega\|_1 \\ \text{subject to} & Y\omega = y \\ & \omega \in \mathbb{R}^N \end{array} \tag{10}$$

We give a short outline of the proof: Let $\omega^* = (\omega_1, ..., \omega_n) \in \mathbb{R}^N$ be a solution of (10) with $\omega_i \in \mathbb{R}^{N_i}$ and chose $x \in \mathbb{R}^N$ as described above. We have:

$$0 = Y(\omega^* - x) = Y_i(\omega_i^* - x_i) + \sum_{j \neq i} Y_j \omega_j^*$$

Due to the independence of the subspaces $S_1, ..., S_n$ this yields:

$$Y_i(\omega_i^* - x_i) \in S_i \cap \bigoplus_{j \neq i} S_j = \{0\}$$

Thus we conclude that $Y_i \omega_i^* = Y_i x_i = y$ holds. Due to the optimality of ω^* in (10) we must have $\omega_j^* = 0$ for all $j \neq i$. This concludes the proof. This idea is the cornerstone of the sparse subspace clustering algorithm (SSC). For a data matrix $Y = (y_1, ..., y_N) \in \mathbb{R}^{d \times N}$ in column notation and $i \in \{1, ..., N\}$ we consider the following convex problem:

$$\begin{array}{ll} \text{minimize} & \|\omega\|_1 \\ \text{subject to} & Y\omega = y_i \\ & e_i^T \omega = 0 \\ & \omega \in \mathbb{R}^N \end{array} \tag{11}$$

If (11) has a solution we denote it by w_i^*. Otherwise y_i is independent from the space spanned by the other columns and we put $w_i^* := 0$. We define the symmetric similarity matrix $W := I + (|w_1^*|, ..., |w_N^*|) + (|w_1^*|, ..., |w_N^*|)^T \in \mathbb{R}^{N \times N}$ where $|\cdot|$ denotes the component-wise absolute value. We want to remark that $W_{ij} > 0$ implies that y_i and y_j live in the same subspace. Later we will show that this condition is not necessary. From this matrix we can construct the similarity graph G and apply the spectral clustering methods from section 2 in order to find the connected components of the graph. The elements in each connected subgraph live in the same subspace.

As an example let us consider the vectors $y_1 = e_1, y_2 = U_\theta e_1, y_3 = U_\theta^T e_1, y_4 = e_2, y_5 = U_\theta e_2, y_6 = U_\theta^T e_2, y_7 = e_3 \in \mathbb{R}^3$ where $U_\theta \in \mathrm{SO}(3)$ is the counterclockwise rotation by the angle θ in the e_1-e_2-plane. Obviously the vectors $y_1, ..., y_6$ live in the e_1-e_2-plane. An easy but lengthy calculation shows that for sufficiently small angles $\theta > 0$ the similarity matrix W is given by

$$W = \left(\begin{array}{cccccc|c} 1 & * & * & 0 & 0 & 0 & 0 \\ * & 1 & 0 & 0 & * & 0 & 0 \\ * & 0 & 1 & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 1 & * & * & 0 \\ 0 & * & 0 & * & 1 & 0 & 0 \\ 0 & 0 & * & * & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

where $*$ indicates a strictly positive value. The associated similarity graph has three connection components. The spectral clustering algorithm from section 2 returns the clusters $C_1 = \{y_1, y_2, y_3, y_4, y_5, y_6\}$ and $C_2 = \{y_7\}$. We have a look at the magnitude of W's eigenvalues:

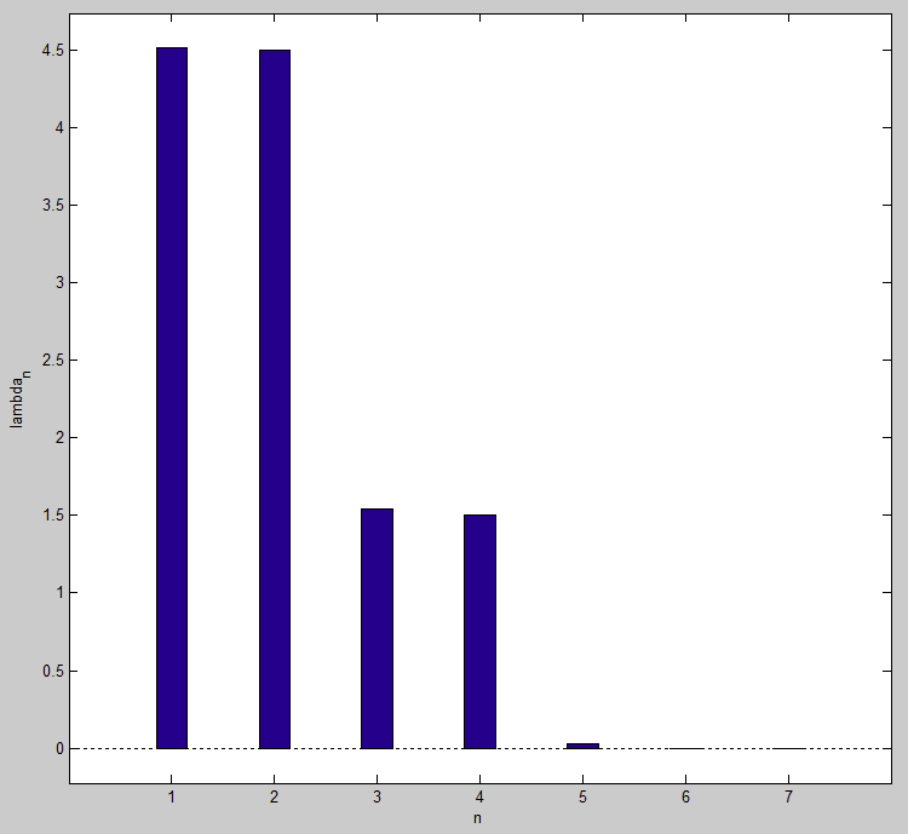

Figure 5: Eigenvalues of the matrix W

The eigenvalue 0 has a multiplicity of two which corresponds to the fact that the similarity graph has two connected components. We observe that the gap between the eigenvalues λ_5 and λ_6 is significantly smaller than the gap between λ_4 and λ_5. Using the eigenvectors associated to $\lambda_5, \lambda_6, \lambda_7$ for spectral clustering yields the partition $C_1' = \{y_1, y_2, y_3\}, C_2' = \{y_4, y_5, y_6\}$ and $C_3' = \{y_7\}$. In the "subspace" context we can interpret this as follows: the vectors y_2 and y_3 are distorted versions of y_1. This means $y_2 = y_1 + n_2$ and $y_3 = y_1 + n_3$ for some noise vectors $n_2, n_3 \in \mathbb{R}^3$ of small magnitude. The same relationship holds for y_5 and y_6 with respect to y_4. Thus the partition C_1', C_2', C_3' represents the true subspace clustering of the undistorted data.

As pointed out in [EV09], it is convenient to allow for some distortion in (11), in order to make the procedure more robust for real world data. Relaxing the equality constraint $Y\omega = y_i$ to the inequality $\|Y\omega - y_i\|_2 \leq \varepsilon$ we can cope with deviations of the magnitude ε. In order to generate the similarity matrix W we then solve

$$\begin{aligned} &\text{minimize} && \|\omega\|_1 \\ &\text{subject to} && \|Y\omega - y_i\|_2 \leq \varepsilon \\ &&& e_i^T \omega = 0 \\ &&& \omega \in \mathbb{R}^N \end{aligned} \tag{12}$$

instead of (11). Applying the methods from convex duality we one can show that a solution w_i^* in (12) is always obtained. Moreover there exists a positive constant γ, such that solving the latter convex

problem is equivalent to the following minimization task:

$$\begin{aligned} &\text{minimize} && \|Y\omega - y_i\|_2 + \gamma\|\omega\|_1 \\ &\text{subject to} && e_i^T\omega = 0 \\ &&& \omega \in \mathbb{R}^N \end{aligned} \tag{13}$$

We remark that the constant γ depends on the noise-level ε and on the data contained in Y. As known from general theory, increasing γ puts higher penalties on non-sparse solutions. Setting $\gamma = 0$ yields the least square solution $w_i^* = Y^+ y_i$. A derivation and a review of some general ideas from convex analysis can be found in the appendix.

As next step in the procedure, spectral clustering is applied to the matrix W. Often the true number of subspaces is unknown. In order to estimate it, one can calculate the eigengap $i^* = \text{argmax}_{i=1,..,N-1}(\lambda_i - \lambda_{i+1})$ from the ordered sequence of the Laplacian's eigenvalues $\lambda_1 \geq ... \geq \lambda_N = 0$. Afterwards the eigenvectors associated to the $N - i^*$ smallest eigenvalue are used for spectral clustering. More details on the sparse subspace clustering algorithm, the data-driven choice of the parameter γ and further theoretical analysis can be found in [SEC4]. Many real world applications require us to to cluster data in affine subspaces. As pointed out in [EV09] the affine subspace clustering task can be tackled by imposing the additional linear constraint $\mathbb{1}^T\omega = 1$ to program (13).

In applications the data matrix $Y \in \mathbb{R}^{d\times N}$ will often have the property that d is much larger than N. For example, the dataset UHL from section 1 contains protein spectra of length 42390 from 150 patients. Clustering the data from UHL with the aforementioned method principally requires us to solve 150 times the convex program (13) with $Y \in \mathbb{R}^{42390\times 150}$. The computational effort can be drastically reduced by a simple transformation: For $Y \in \mathbb{R}^{d\times N}$ with $d \geq N$ we calculate the singular value decomposition $Y = U\Sigma V^T$ where $U \in \mathrm{O}(d), V \in \mathrm{O}(N)$ and $\Sigma \in \mathbb{R}^{d\times N}$. For all $\omega \in \mathbb{R}^N$ we find that

$$\|Y\omega - y_i\|_2 = \|\Sigma V^T(\omega - e_i)\|_2 = \left\|\text{diag}(\sigma_1, ..., \sigma_N)V^T(\omega - e_i)\right\|_2$$

is true. Thus the convex program (13) can be transform to:

$$\begin{aligned} &\text{minimize} && \left\|\text{diag}(\sigma_1, ..., \sigma_N)V^T\omega\right\|_2 + \gamma\|\omega\|_1 \\ &\text{subject to} && e_i^T\omega = -1 \\ &&& \omega \in \mathbb{R}^N \end{aligned} \tag{14}$$

As we see, all the quantities appearing in the latter convex problem are N-dimensional and depend only on the the known data matrix Y, its singular values and its right singular vectors $v_1, ...v_N$. The d-dimensional basis of left singular vectors is not explicitly needed for our computation. For moderately sized values of N this reduced singular value decomposition can be computed at low computational cost. Again, the affine subspace clustering task can be treated by adding an additional constraint to (14) which yields the convex program:

$$\begin{aligned} &\text{minimize} && \left\|\text{diag}(\sigma_1, ..., \sigma_N)V^T\omega\right\|_2 + \gamma\|\omega\|_1 \\ &\text{subject to} && e_i^T\omega = -1 \\ &&& \mathbb{1}^T\omega = 1 \\ &&& \omega \in \mathbb{R}^N \end{aligned} \tag{15}$$

3.1 Numerical results for the dataset UHL

We have applied affine subspace clustering to partition the dataset UHL into two groups. First we used algorithm (15) to generate the similarity matrix W. Afterwards we have used the normalized spectral clustering method from [NJW2] in order to generate the partition. We used Matlab R2010b in combination with the CVX 2.1 optimization package for our computations. As solver we chose SDPT3 version 4.0. The section $p = 1$ in the table below contains our results.

p	γ	error	p	γ/p	error
1	10^{10}	11.33 %	0.8	10^{10}	48.67 %
1	10^{9}	10.00 %	0.8	10^{9}	46.67 %
1	10^{8}	10.00 %	0.8	10^{8}	48.67 %
1	10^{7}	11.33 %	0.8	10^{7}	48.67 %
1	10^{6}	10.00 %	0.8	10^{6}	48.67 %
1	10^{5}	11.33 %	0.8	10^{5}	48.67 %
1	10^{4}	13.33 %	0.8	10^{4}	44.67 %
1	10^{3}	15.33 %	0.8	10^{3}	14.67 %
1	10^{2}	11.33 %	0.8	10^{2}	14.00 %
1	10	15.33 %	0.8	10	15.33 %
1	1	16.67 %	0.8	1	16.67 %
1	10^{-1}	17.33 %	0.8	10^{-1}	16.67 %
1	10^{-2}	17.33 %	0.8	10^{-2}	16.67 %
1	10^{-3}	17.33 %	0.8	10^{-3}	18.67 %
1	10^{-4}	18.67 %	0.8	10^{-4}	18.67 %
1	10^{-5}	18.00%	0.8	10^{-5}	16.67 %
1	10^{-6}	16.67 %	0.8	10^{-6}	16.67 %
1	10^{-7}	18.67 %	0.8	10^{-7}	16.67 %
1	10^{-8}	18.67 %	0.8	10^{-8}	16.67 %
1	10^{-9}	16.67 %	0.8	10^{-9}	-
1	10^{-10}	18.00 %	0.8	10^{-10}	-

Table 4: l_p-regularized affine subspace clustering of UHL data into two partitions

Figure 6 contains the spectrum of the normalized symmetric Laplacian matrix $L_{\text{sym}} = I - D^{-1/2} W D^{-1/2}$ for $\gamma = 10^{-6}$ and $\gamma = 10^{6}$.

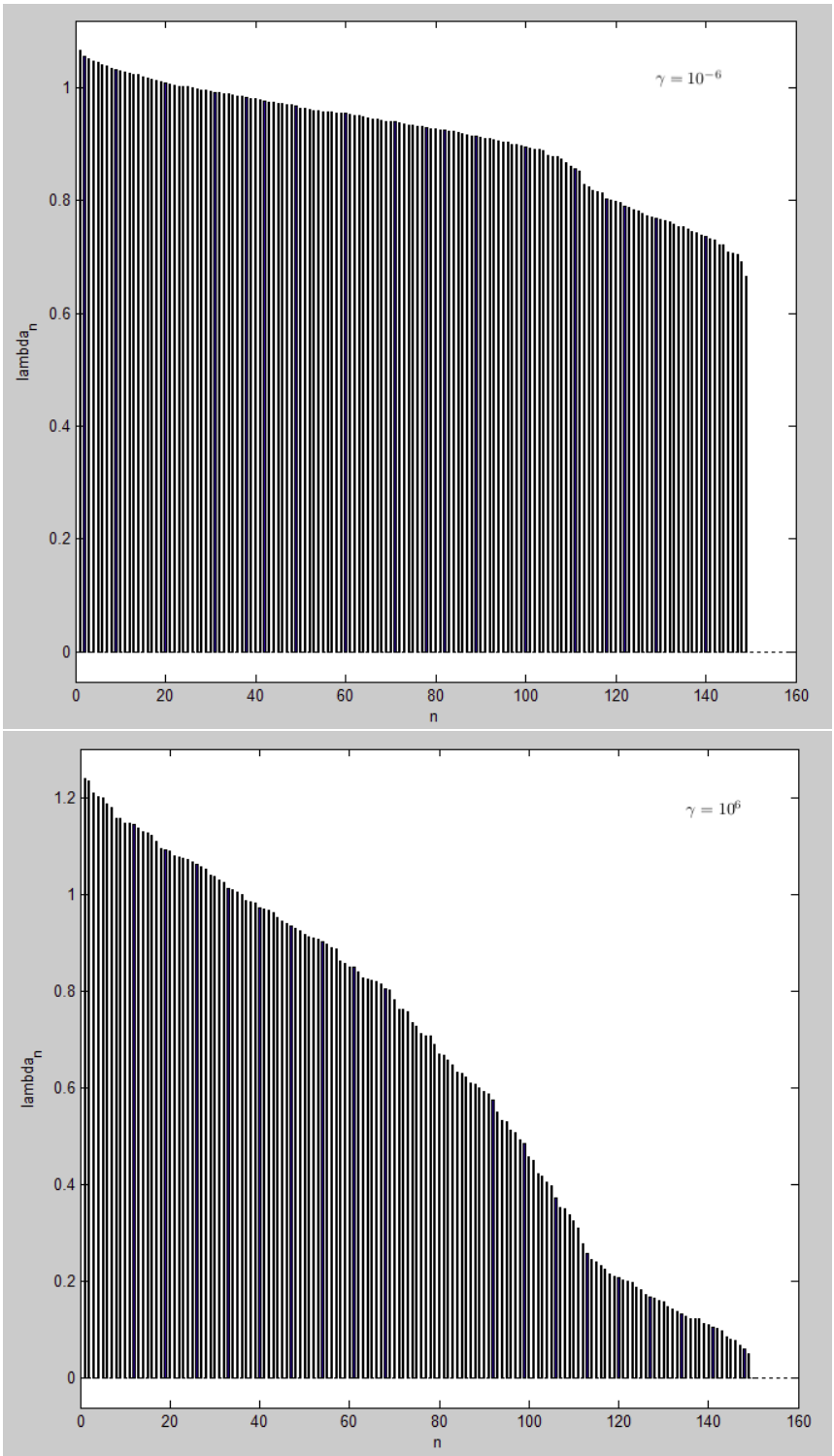

Figure 6: spectrum of L_{sym} for $\gamma = 10^{-6}$ (top) and $\gamma = 10^{6}$ (bottom) l_1-regularized affine subspace clustering, dataset UHL

3.2 Generalized subspace clustering with l_p-type penalties

We want to generalize method (15) by allowing non-convex penalization. For each choice of $\varepsilon, \gamma > 0$, $p \in (0,1)$ and $a \in \{0,1\}$ we generate an affinity matrix W by solving

$$
\begin{aligned}
&\text{minimize} && \left\|\operatorname{diag}(\sigma_1,...,\sigma_N)V^T\omega\right\|_2 + \frac{\gamma}{p}\sum_{j=1}^{N}(|\omega_j|+\varepsilon)^p \\
&\text{subject to} && \omega_i = -1 \\
&&& a\mathbb{1}^T\omega = a \\
&&& \omega \in \mathbb{R}^N
\end{aligned}
\tag{16}
$$

for all all $i \in \{1,...,N\}$. The impact of the penalty term is regulated by the parameter γ, the penalty function's concavity can be adjusted by altering the value of p. Choosing $a = 1$ leads to affine subspace clustering. The parameter ε determines the Lipschitz-constant of the penalty term. As pointed out in [OC13] the the IRL1 procedure from section 1 is applicable to this non-convex linearly constraint optimization task. For each $i \in \{1,...,N\}$ we solve a sequence of convex problems

$$
\omega^{k+1,i} \in \underset{\substack{\omega_i=-1 \\ a\mathbb{1}^T\omega=a \\ \omega\in\mathbb{R}^N}}{\operatorname{argmin}} \ \|\operatorname{diag}(\sigma_1,...,\sigma_N)V^T\omega\|_2 + \gamma\|W^{k,i}\omega\|_1 \tag{17}
$$

where

$$
W^{k,i} := \operatorname{diag}\left[(|\omega_1^{k,i}|+\varepsilon)^{p-1},...,(|\omega_N^{k,i}|+\varepsilon)^{p-1}\right]
$$

and $\omega^{0,i}$ is a randomly chosen starting point. We used $|\omega_1^{k,i} - \omega_1^{k+1,i}| < 10^{-8}$ or $k > 30$ as break criterion for the inner IRL1 loop. We chose the concavity parameter $p = 0.8$ and $\varepsilon = 0.001$. Again Matlab R2010b with CVX 2.1 and SDPT3 version 4.0 was used for the computations. After the affinity matrix W was generated, we applied the normalized spectral clustering (NSC) method from [NJW2] in order to partition the data in two groups. Our results are contained in the right column of Table 4. The detailed procedure is stated in Algorithm 2 below. By rSVD(Y) we denote a reduced singular value decomposition which returns the singular values $\sigma_1 \geq ... \geq \sigma_N \geq 0$ and a matrix $V \in \mathrm{O}(N)$ containing the right singular vectors of Y. Standard computer algebra software like Matlab and Octave offer efficient algorithms for the computation of such matrix decompositions. Choosing $p = 1$ will cause the inner loop to break after a single iteration and the algorithm reduces to the convexly penalized procedure from the beginning of section 3.

Algorithm 2 l_p-regularized subspace clustering

INPUT:
$Y \in \mathbb{R}^{d\times N}$ data matrix
$p \in (0,1]$ concavity parameter
$\gamma \in (0,\infty)$ impact parameter
$\varepsilon \in [0,\infty)$ adjustment of Lipschitz-constant at origin, need $\varepsilon > 0$ if $p < 1$
$a \in \{0,1\}$ switch for affine subspace clustering
$\delta \in (0,\infty)$ precision for break criterion in inner IRL1 loop
MaxIt $\in \mathbb{N}$ maximal number of iterations in inner IRL1 loop

OUTPUT:
Grps $\in \mathbb{R}^N$ vector with group labels

function LPSC

$[\sigma_1, ...\sigma_N, V] \leftarrow \mathrm{rSVD}(Y)$

$F(\omega) := \left\|\mathrm{diag}(\sigma_1, ..., \sigma_N)V^T\omega\right\|_2 + \frac{\gamma}{p}\sum_{j=1}^{N}(|\omega_j| + \varepsilon)^p$

for $i = 1$ to N **do**
$x \leftarrow$ starting point from $\mathbb{R}^N$, e.g. randomly chosen

for $k = 1$ to MaxIt **do**
$W \leftarrow \mathrm{diag}\left[(|x_1| + \varepsilon)^{p-1}, ..., (|x_N| + \varepsilon)^{p-1}\right]$

$$\begin{cases} \text{minimize} & \left\|\mathrm{diag}(\sigma_1, ..., \sigma_N)V^T\omega\right\|_2 + \gamma\left\|W\omega\right\|_1 \\ \text{subject to} & \omega_i = -1 \\ & a\mathbb{1}^T\omega = a \\ & \omega \in \mathbb{R}^N \end{cases}$$

gap $\leftarrow |F(x) - F(\omega^*)|$
$x \leftarrow \omega^*$
if gap $< \delta$ **or** $p == 1$ **then** break
end

$M(:, i) \leftarrow x$
end

Grps $\leftarrow \mathrm{NSC}(|M| + |M^T| + I)$
return Grps

end

It might seem that the choice $p = 1$ outperforms its $p = 0.8$-concave penalty alternative on the dataset UHL due to its high accuracy and low computational cost. We want to remark that due to time constraints the algorithm and its numerical testing leaves a lot of space for improvement. In particular we hope that a careful calibration of the parameters p and λ will improve the accuracy of the clustering procedure. The overall performance of the algorithm depends heavily on the solution of the convex program in the IRL1 loop. Its very specific structure ($\|Ax\|_2 + \|Bx\|_1$ plus affine constraints) allows for very efficient solvers, c.f. [KI07] or recent overview papers on l_1-regularized

least-squares programs. Our numerical experiments were conducted with general purpose solvers as Mosek or SDPT3. Beside performance issues, numerous other questions remain open. For example, it would be interesting to analyze in detail the impact of the parameters p, γ and ε on the cluster assignments. Data-driven methods for parameter choices in the standard subspace clustering scenario can be found in the literature, c.f. [SEC4]. Are these ideas somehow transferable to the non-convex penalty case?

References

[AL06] C. D. Aliprantis: Infinite Dimensional Analysis - A Hitchhiker's Guide, 3rd Edition, Springer, 2006

[BP12] V. Barbu, T. Precupanu: Convexity and Optimization in Banach Spaces, Springer, 2012

[CZ14] X. Chen, W. Zhou: Convergence of the reweighted l_1 minimization algorithm for l_2-l_p minimization, Computational Optimization and Applications, Vol. 59, Issue 1-2, Oct. 2014, p. 47-61

[CO15] T. Conrad et al.: Sparse Proteomics Analysis - A compressed sensing-based approach for feature selection and classification of high-dimensional proteomics mass spectrometry data, 2015

[EV09] E. Elhamifar, R. Vidal: Sparse subspace clustering, In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pages 2790 - 2797

[GM98] S. Guattery, G. Miller: On the quality of spectral separators. Journal on Matrix Analysis and Applications, Vol. 19, Issue 3, p. 701 - 719, 1998

[HTF9] T. Hastie, R. Tibshirani, J. Friedman: The Elements of Statistical Learning, 2nd Edition, Springer 2009

[KI07] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, D. Gorinevsky: An Interior-Point Method for Large-Scale l_1-Regularized Least Squares, Selected Topics in Signal Processing, IEEE Journal of, Vol. 1, Issue 4, Dec. 2007

[LA93] S. Lang: Real and Functional Analysis, 3rd Edition, Springer, 1993

[LU07] U. v. Luxburg: A Tutorial on Spectral Clustering, Statistics and Computing, Vol. 17, Issue 4, p. 395-416, 2007

[NJW2] A. Ng, M. Jordan, Y. Weiss: On spectral clustering - analysis and an algorithm, Advances in Neural Information Processing Systems 14, 2001

[OC13] P. Ochs, A. Dosovitskiy, T. Brox, T. Pock: An Iterated l_1 Algorithm for Non-smooth Non-convex Optimization in Computer Vision, IEEE Conference on Computer Vision and Pattern Recognition, 2013

[OC15] P. Ochs, A. Dosovitskiy, T. Brox, T. Pock: On iteratively reweighted Algorithms for Non-smooth Non-convex Optimization in Computer Vision, SIAM Journal on Imaging Sciences 8, 2015, p. 331-372

[RO74] R. T. Rockafellar: Conjugate Duality and Optimization, CBMS-NSF Regional Conference Series in Applied Mathematics, Vol. 16, 1974

[SEC4] M. Soltanolkotabi, E. Elhamifar, E. J. Candès: Robust subspace clustering, Annals of Statistics, Vol. 42, No. 2, p. 669-699, 2014