Exact Solution of L_∞-norm and L_2-norm Plane Separation

Charles Audet

GERAD and École Polytechnique de Montréal
C.P. 6079, station Centre-ville
Montréal (Qc), Canada, H3C 3A7
charles.audet@gerad.ca

Pierre Hansen

GERAD and HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Qc) Canada, H3T 2A7
pierre.hansen@gerad.ca

Alejandro Karam

HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Qc) Canada, H3T 2A7
alejandro.karam@hec.ca

Chi-to Daniel Ng

Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
lgtctng@polyu.edu.hk

Sylvain Perron *

GERAD and École Polytechnique de Montréal
C.P. 6079, Succ. Centre-ville
Montréal (Qc) Canada, H3C 3A7
sylvain.perron@gerad.ca

November 23, 2004

*Corresponding author, Fax: (514)340-5665

Abstract

We consider the problem of separating two sets of points in an n-dimensional real space with a (hyper)plane that minimizes the sum of L_p-norm distances to the plane of points lying on the wrong side of it. We propose a new mixed integer programming formulation for the L_∞-norm case, and an implementation of a nonconvex quadratic programming algorithm for the L_2-norm case, based on a branch-and-cut approach. Computational results are reported for several publicly available data sets and larger artificial problems. We also explore the accelerating potential of incorporating heuristic bounds in our exact solution approaches.
Key Words: linear discrimination, separating plane, L_p-norm separation.

1 Introduction

A basic problem in supervised classification is the separation of (or discrimination between) two sets of points in the real space $\mathbb{R}^n$ with a (hyper)plane that assigns a half-space to each of the sets. This problem is of interest in data mining and machine learning settings. When the interior of the convex hulls of the sets intersect, perfect linear separation is not possible, as there are no plane leaving all points from the first set on a side of the plane, and those of the other set on the opposite side. A discriminating plane may be found by minimizing some measure of the separation error, such as the total or average number of misclassified points, or some notion of distance of such points to the plane (for a historical overview, see [17] and references therein). This often leads to difficult optimization challenges, and the separation criteria are sometimes chosen so as to keep the problem tractable.

The fairly intuitive objective of minimizing the sum of distances of misclassified points to the plane appears to have been behind several approaches to the separation problem, often leading to various Linear Programming formulations. The objective functions in these programs are, however, only substitutes for the true measures of the distances that ignore the essentially non-linear nature of the optimization problem.

In [9], Mangasarian states the problem precisely in terms of the analytical expression for the sum of L_p-norm distances of misclassified points to the plane. This results in a formulation of the general problem as that of minimizing a convex function (involving a sum of $\max\{0, \cdot\}$ operators) over a unit sphere in a norm dual to that one originally chosen to measure the distances from the misclassified points to the plane. An exact solution is often practical for $p = 1$, as for the L_1-norm the problem can be solved by $2n$ linear programs[1].

Our work takes these efforts a step further: we explore exact solution approaches for two other values of p. We propose a mixed integer formulation for the L_∞-norm and tackle the L_2-norm problem with an adaptation of the branch-and-cut algorithm of Audet *et al.* [2] for non-convex quadratically constrained quadratic programs.

In order to probe the practical applicability, potential scalability and speed of these approaches, we test them on several publicly available data sets, as well as on four series of

[1] An alternative heuristic approach for this case is also presented in [9] which is a successive linearization algorithm applied to a penalty reformulation of the problem. It is indirectly suggested that this technique could also be used for the L_∞-norm case.

randomly generated problems. In the context of these tests, we also explore the effect of using heuristic bounds to accelerate our exact solution implementations.

The rest of the paper is organized as follows. The following section establishes the notation and summarizes some key results from [9]. Section 3 presents our formulation for the L_∞-norm and Section 4 summarizes the branch-and-cut approach implemented for the L_2-norm. We then describe, in Section 5, the data sets used for our numerical tests and discuss the corresponding results. Section 6 concludes.

2 Notation and problem statement

We first establish the notation and state the general optimization problem. The scalar product of two vectors x and y both in $\mathbb{R}^n$, is denoted $x'y$. If M is a matrix, M_i represents its i^{th} row. $\mathbb{R}^n_+$ is the closed positive orthant. When necessary, 0 and 1 denote, respectively, vectors of zeros and ones in appropriate dimensions.

Following [9], we denote $\mathcal{A}$ and $\mathcal{B}$ the two sets of points in $\mathbb{R}^n$ containing respectively m and k points represented by the matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{k\times n}$.

For a given plane

$$P = \left\{ x \,\middle|\, w'x = \gamma \right\} \text{ with } \gamma \in \mathbb{R}, w \in \mathbb{R}^n, w \neq 0 \tag{1}$$

a point $x \in \mathcal{A} \cup \mathcal{B}$ is said to be misclassified when

$$\begin{array}{ll} w'x > \gamma & \text{if } x \in \mathcal{A}, \\ & \text{or} \\ w'x < \gamma & \text{if } x \in \mathcal{B}. \end{array} \tag{2}$$

For a fixed $p \in [1, \infty]$, the L_p-norm distance between a point $x \in \mathcal{A} \cup \mathcal{B}$ and its projection $\pi(x)$ to the plane P is given by

$$\|x - \pi(x)\|_p = \frac{|w'x - \gamma|}{\|w\|'_p} \tag{3}$$

where $\|\cdot\|'_p$ denotes the dual norm of $\|\cdot\|_p$. We recall that, for $1 < p < \infty$, $\|\cdot\|'_p = \|\cdot\|_q$ where $\frac{1}{p} + \frac{1}{q} = 1$. The cases $p = 1$ and $p = \infty$, are defined by a limit argument. Thus,

$$\|\cdot\|'_2 = \|\cdot\|_2, \|\cdot\|'_1 = \|\cdot\|_\infty \text{ and } \|\cdot\|'_\infty = \|\cdot\|_1. \tag{4}$$

Although the formulae for the arbitrary-norm distance from a point to a hyperplane had been derived in other settings (e.g., [12]), [9] appears to be the first to establish and use them explicitly in the context of linear discrimination.

Note that there is one degree of freedom in the characterization of the plane P, which can be used to fix an arbitrary scale. The choice of the scaling constraint

$$\|w\|'_p = 1 \tag{5}$$

removes the denominator from (3) and rules out the null solution $w = 0$ that has haunted some previous formulations of this problem. This approach, which appears to have been first proposed in [4], is used in several studies (see, e.g., [11] and references therein), and elegantly generalizes to the arbitrary-norm case.

The resulting optimization problem (problem(17) of [9]) is

$$\min_{w,\gamma} \left\{ \sum_{i=1}^{m} \max\left\{-w'A_i + \gamma, 0\right\} + \sum_{j=1}^{k} \max\left\{w'B_j - \gamma, 0\right\} \middle| \|w\|'_p = 1 \right\} \tag{6}$$

where $w \in \mathbb{R}^n$ and $\gamma \in \mathbb{R}$.

This program can be reformulated in order to linearize the $\max\{\cdot, 0\}$ operators in the objective function. This results in

$$\min_{w,\gamma,y,z} \left\{ \sum_{i=1}^{m} y_i + \sum_{j=1}^{k} z_j \middle| \begin{array}{c} y_i \geq -w'A_i + \gamma \text{ for } i = 1, .., m \\ z_j \geq w'B_j - \gamma \text{ for } j = 1, .., k \\ \|w\|'_p = 1 \\ y \geq 0, \quad z \geq 0 \end{array} \right\} \tag{7}$$

where $w \in \mathbb{R}^n$, $\gamma \in \mathbb{R}$, $y \in \mathbb{R}^m$ and $z \in \mathbb{R}^k$. The following sections are based on this last formulation.

3 L_∞-norm: A Mixed Integer Formulation

For the L_∞-norm case, the constraint (5) requires

$$\|w\|'_\infty = \|w\|_1 = \sum_{l=1}^{n} |w_l| = 1. \tag{8}$$

This can be addressed by the usual technique to linearize the absolute value operator. We replace w by two non-negative vectors w^+ and w^- in $\mathbb{R}^n_+$ such that

$$w^+ + w^- = |w| \tag{9}$$

$$w^+ - w^- = w \tag{10}$$

$$0 \leq w^+ \leq 1 \qquad 0 \leq w^- \leq 1. \tag{11}$$

We must also include a vector of binary variables $\delta \in \{0,1\}^n$ to force at most one of each pair of variables w_i^+ and w_i^- to be nonzero.

The resulting formulation is:

$$\min_{w^+,w^-,\gamma,\delta,y,z} \left\{ \sum_{i=1}^{m} y_i + \sum_{j=1}^{k} z_j \;\middle|\; \begin{array}{cc} y_i \geq (-w^+ + w^-)' A_i + \gamma & \text{for } i = 1,..,m \\ z_j \geq (w^+ - w^-)' B_j - \gamma & \text{for } j = 1,..,k \\ \sum_{l=1}^{n} (w_l^+ + w_l^-) = 1 & \\ w_l^+ \leq \delta_l & \text{for } l = 1,...,n \\ w_l^- \leq 1 - \delta_l & \text{for } l = 1,...,n \\ y \geq 0, \quad z \geq 0 & \\ w^+ \geq 0, \quad w^- \geq 0 & \\ \delta \in \{0,1\}^n & \end{array} \right\} \tag{12}$$

where $\gamma \in \mathbb{R}$, $y \in \mathbb{R}^m$, $z \in \mathbb{R}^k$, $w^+ \in \mathbb{R}^n$ and $w^- \in \mathbb{R}^n$.

4 L_2-norm: A Branch and Cut Approach

Under the Euclidean norm, the constraint (5) is equivalent to

$$\|w\|_2' = w'w = 1. \tag{13}$$

For this case, we deal with problem (7) directly as a quadratic program with non-convex quadratic constraints (QQP) which may be solved using the branch-and-cut algorithm of Audet *et al.* [2]. This algorithm provides in finite time a globally optimal solution (within given feasibility and optimality tolerances). Its basic idea is to estimate all quadratic terms by successive linearizations, or outer-approximations, within an enumeration tree using Reformulation-Linearization Techniques (RLT) (see, e.g., [2, 15, 16]). The essential idea behind RLT is to replace each squarest term w_i^2 appearing in QQP by a linear one s_i and each bilinear term $w_i w_j$ by a linear one t_{ij}. Then the linear terms are constrained to make sure that s_i

approximates w_i^2 and t_{ij} approximates $w_i w_j$ as closely as possible. Problem (7) with $p = 2$ is a particular QQP without bilinear terms. We specialize the RLT algorithm to QQP with only linear and squarest terms.

The RLT algorithm presented in [2] relies on four classes of linearizations, denoted by $[\cdot]_\ell$, only two of which do not concern bilinear terms. The first one, originally due to Al-Khayyal and Falk [1], contains under-estimations of the square function. For any fixed value $\alpha_i \in \mathbb{R}, i \in \{1, 2, \ldots, n\}$ consider the RLT constraints

$$\underline{S}_i(\alpha_i): \qquad 0 \leq [(w_i - \alpha_i)^2]_\ell = s_i - 2\alpha_i w_i + \alpha_i^2.$$

This inequality defines the half-space tangent to the convex function w_i^2 at the point $w_i = \alpha_i$. When the solution of the relaxation is such that $s_i < w_i^2$, the value of α_i is chosen in such a way as to minimize a maximum error in the interval. The second class of linearizations not related to bilinear term are over-estimations of the square function. For $w_i \in [\alpha_i, \beta_i], i \in \{1, 2, \ldots, n\}$ consider the constraints

$$\overline{S}_i(\alpha_i, \beta_i) : 0 \leq [(w_i - \alpha_i)(\beta_i - w_i)]_\ell \;=\; -\alpha_i\beta_i + (\alpha_i + \beta_i)w_i - s_i.$$

For given values of α_i and β_i, this inequality defines the half-space obtained through the cord from (α_i, α_i^2) to (β_i, β_i^2).

Figure 1 illustrates both the under- and over-estimations of the square function $f(w_i) = w_i^2$ defined for w_i in the interval $[\ell_i, u_i] \subseteq \mathbb{R}$. We relax the requirement that $(w_i, s_i) \in \{(w_i, s_i) : s_i = w_i^2, \ell_i \leq w_i \leq u_i\}$ by imposing that (w_i, s_i) belongs to the triangle defined by $\underline{S}_i(\ell_i)$, $\underline{S}_i(u_i)$ and $\overline{S}_i(\ell_i, u_i)$ (dashed area on Figure 1).

The algorithm is of the branch-and-cut type. At the root of the enumeration tree, an intensive pre-processing phase is performed to tighten as much as possible the lower bound ℓ_i and upper bound u_i on each variable w_i appearing in a quadratic term. We can thus start with a closer outer-approximation using the inequalities $\underline{S}_i(\ell_i)$, $\underline{S}_i(u_i)$ and $\overline{S}_i(\ell_i, u_i)$.

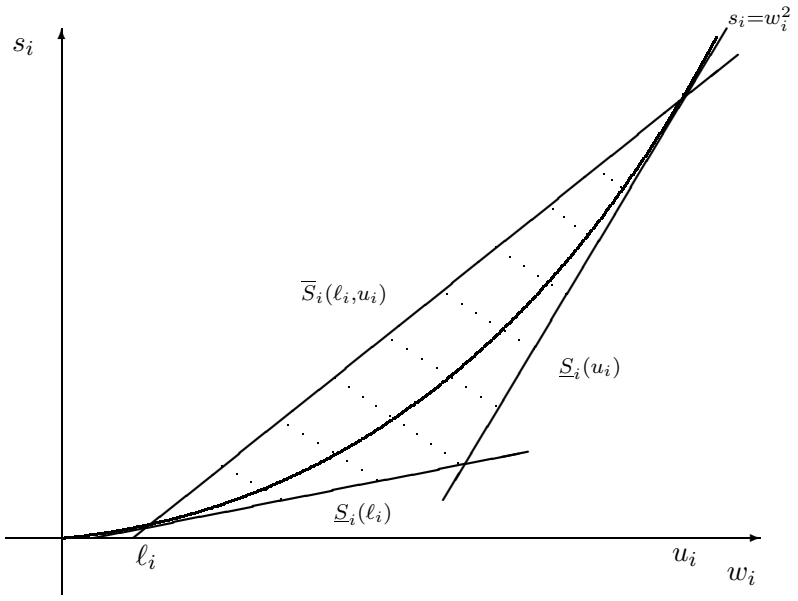

Figure 1: Under and over approximations of the function $f(w_i) = w_i^2$

Then, the algorithm recursively branches on the variables involved in the quadratic terms using a best-first strategy that explores the candidate having the largest linear relaxation value. At each node of the enumeration tree, it identifies the variable w_i for which the error $|w_i^2 - s_i|$ is the largest. The branching process creates two subproblems: one in which $w_i \leq \alpha_i$ and another in which $w_i \geq \alpha_i$ for a carefully chosen value of α_i. The algorithm adds an inequality $\underline{S}_i(\alpha_i)$ and then, for each of the two sons created by the branching process, it introduces, using the separation constraint on w_i, an inequality $\underline{S}_i(\alpha_i, \beta_i)$ (or $\underline{S}_i(\beta_i, \alpha_i)$) which is valid in all nodes of the subtree rooted at this node. The algorithm stops when an optimal solution (within given feasibility and optimality tolerances) is found. The solution is said to be feasible within a given tolerance $\epsilon_r > 0$ if each quadratic term is approximated within ϵ_r, *i.e.*, if $|w_i^2 - s_i| < \epsilon_r$ for all i. A ϵ_r-feasible solution is said to be optimal within a given optimality tolerance $\epsilon_z > 0$, if the difference between the optimal value of the relaxation and the value of the solution is not more than ϵ_z.

Results on problems (7) with $p = 2$ presented in section 5 have been obtained using an implementation of the algorithm (Perron [14]) that allows the solution of large QPP with low quadratic density, i.e., problems having a large number of variables but few quadratic terms.

Interesting instances of problem (7) with $p = 2$ usually have this property, since the number of points $(m + k)$ is usually very large compared to the dimension (n). Note that the number of variables of this QQP is $m + k + n + 1$ and the number of quadratic terms is n. Another feature of this implementation is that the bounds on variables are refined not only at the root node but also at subsequent nodes.

5 Numerical experiments

We tested our formulation (12) for the L_∞-norm case and our solution approach for program (7) for the L_2-norm case on a set of instances from the UCI Repository [5] and two series of random problems created with D. Musicant's NDC generator [13], which produces normally distributed clusters. This generator is publicly available and has been used in other discrimination studies (e.g., [7, 10]). The parameters used for the generation of the artificial sets, as well as the preprocessing steps for the real life instances from the UCI Repository are detailed in Appendix A.

Heuristic approaches are being increasingly used as valuable tools to accelerate exact solution methods (see, e.g., [6, 8]). We measure the effect, on the performance of our implementations, of adding a heuristic bound. We first obtain heuristic solutions to all the problems with an application of Variable Neighborhood Search (VNS) [3], and use the objective value found as a cutoff limit in the branching processes. In addition, for the L_2-norm case, we add to each linear subproblem the constraint that the objective function be no greater than the heuristic bound, and the heuristic solution is used as starting point.

The mixed integer programs for the L_∞-norm were solved directly with ILOG CPLEX 8.1, using default settings. We used the same tool for the linear programming part of the branch and cut algorithm for the L_2-norm case. All tests were performed on dual processor computers[2] running under Linux.

In all tables, the fit column is the full set classification accuracy of the solution found. The objective value is rounded to its approximate accuracy, namely 10^{-5}. Time is measured in CPU seconds, unless otherwise indicated. Time is also reported for the runs where the heuristic solution was used to accelerate the exact solution process. The last column shows

[2]Intel Xeon 3.06 GHz, 1 Mb cache memory, 2 GO RAM.

the time savings obtained by this procedure, considering of course the time used to obtain the heuristic solution[3].

We first discuss the results obtained on the random problems, and then turn to the UCI real life instances.

5.1 Random problems

For the first series of random problems, we fixed the dimension at 6 and explored the effects of increasing the number of points from 2000 to 20000 (by steps of 2000). We then generated a second series of problems with 2000 observations, with dimensions ranging from 4 to 13. We will refer to these two test sets as NDC6d and NDC2k respectively. For each problem size, we generated 10 instances and report the corresponding mean values of the results in tables 1, 2, 3 and 4.

Acceleration by inclusion of a heuristic solution results in significant time savings on relatively large problems, for both the L_2-norm and the L_∞-norm cases. However, the inclusion of the heuristic solution is counterproductive in some cases.

Under the L_2-norm, this is due to the fact that the new bound is used to refine existing bounds (in nodes of depth up to five); the time spent on these refinements is not compensated for by the node reduction effect on problems with few dimensions. The net savings in higher dimensions, as well as in the UCI problems, are dramatic. In fact, instances in 12 and 13 dimensions could not be solved in reasonable time without the heuristic bound.

Under the L_∞-norm, inclusion of the heuristic bound results in net time gains in larger instances than for the L_2-norm case, because the exact solution time is relatively small with respect to the effort of obtaining the heuristic solution in the first place. Since exact solutions are found very quickly for all the UCI problems and the artificial problems with less than about 10000 points or about 10 dimensions, the heuristic enhancement is not appropriate. However, significant net savings are obtained for larger instances, and they increase with the size of the problem.

[3]It is interesting to point out that the VNS heuristic used found the exact solution, to within its accuracy, for most instances. For the others, the solution was quite close. The exact algorithms were thus performing mostly a confirmation exercise.

Table 1: L_∞-norm results on NDC2k series

Problem size		Exact solution				Global time reduction
		Obj	Fit	Time		
Dim.	Nb. points			Without init. sol.	With init. sol.	
4	2000	1.79383	94.31%	0.8	0.337	-1363%
5	2000	1.71293	93.71%	1.5	0.683	-366%
6	2000	2.09555	93.14%	3.9	1.77	-166%
7	2000	2.18904	91.54%	8.8	4.487	-80%
8	2000	2.51591	89.69%	20.6	14.671	-49%
9	2000	2.55585	90.06%	49.4	33.378	-10%
10	2000	2.36433	89.46%	102.8	66.469	11%
11	2000	3.91649	83.59%	330.9	281.062	5%
12	2000	2.52528	88.58%	530.5	423.858	14%
13	2000	3.10935	85.73%	1324.6	1213.909	8%

Table 2: L_∞-norm results on NDC6d series

Problem size		Exact solution				Global time reduction
		Obj	Fit	Time		
Dim.	Nb. points			Without init. sol.	With init. sol.	
6	2000	1.85690	93.13%	3.7	1.7	-140%
6	4000	3.59903	91.92%	14.9	7.8	-43%
6	6000	6.48506	91.53%	40.9	23.3	3%
6	8000	7.23242	92.25%	65.1	37.3	-9%
6	10000	11.00265	91.08%	126.5	73.6	6%
6	12000	12.92623	90.94%	204.0	114.0	28%
6	14000	16.50828	89.58%	320.7	168.8	36%
6	16000	12.06249	92.22%	288.1	160.4	20%
6	18000	17.75625	92.82%	391.3	234.8	27%
6	20000	12.20177	94.49%	368.0	202.8	29%

Differences between L_∞-norm and L_2-norm in full set accuracy do not appear to be very large. However, on both NDC series, the planes found with the L_2-norm criterion provided better fit than those obtained minimizing the L_∞-norm.

Solution time grows exponentially with the dimension, while it appears to follow a power rule with respect to the number of observations. Best-fit details are provided in Appendix B.

Under the L_2-norm, solution of instances of 20000 observations in 6 dimensions took an average of about 1.6 CPU hours, while the L_∞-norm problems on the same data sets where solved in an average of only about six CPU minutes.

Table 3: L_2-norm results on NDC2k series

Problem size		Exact solution				Global time reduction
		Obj	Fit	Time		
Dim.	Nb. points			Without init. sol.	With init. sol.	
4	2000	3.10957	94.41%	5.0	6.3	-238.0%
5	2000	3.45771	93.84%	11.4	17.4	-149.2%
6	2000	4.33043	93.23%	34.5	38.1	-50.8%
7	2000	5.03871	91.75%	118.3	95.6	0.8%
8	2000	5.96413	90.16%	557.4	223.1	54.7%
9	2000	6.29355	90.17%	1796.3	680.1	59.7%
10	2000	6.48006	89.55%	3194.4	1728.4	44.1%
11	2000	11.27714	83.74%	38530.5	17491.7	54.4%
12	2000	7.26097	89.02%	-	18970.8	-
13	2000	9.49367	86.15%	-	60818.1	-

Table 4: L_2-norm results on NDC6d series

Problem size		Exact solution				Global time reduction
		Obj	Fit	Time		
Dim.	Nb. points			Without init. sol.	With init. sol.	
6	2000	3.97815	93.15%	36.0	38.2	-46.6%
6	4000	7.64622	92.22%	212.6	210.6	-12.4%
6	6000	14.21926	91.60%	635.5	726.1	-19.6%
6	8000	15.94860	92.28%	959.7	947.1	-4.4%
6	10000	23.78685	91.04%	1433.3	1950.2	-41.5%
6	12000	27.07589	91.01%	2593.5	2131.5	15.1%
6	14000	35.80013	89.69%	2858.2	3877.6	-38.3%
6	16000	25.78079	92.25%	2146.0	3143.3	-51.7%
6	18000	36.51885	92.80%	5915.7	5700.6	1.7%
6	20000	24.93259	94.59%	5777.2	3508.1	37.2%

As exact solution of L_∞-norm problems is considerably faster than that of L_2-norm problems, we considered worthwhile to test the behavior of our L_∞-norm method on larger problems, for which averaging over several instances would be impractical. We therefore generated two additional series of unique problems, with simpler structure and for which the complexity (as approximated by the number of full-set missclassifications) could be easily controlled. The first set of these additional test problems, denoted Sym2k , considers instances with 2000 observations in 14, 16 and 18 dimensions, while the set which we call Sym6d includes instances in 6 dimensions with up to 100000 points by increments of 10000.

For the simpler, more precisely controlled instances of the Sym2k and Sym6d series (used only with the L_∞-norm criterion), the objective function grows monotonically, as expected, with the problem size. Note, however, that the average objective function values for the NDC series showed some fluctuations, reflecting the greater variability of those problems.

Table 5: L_∞-norm results on Sym2k series

Problem size		Exact solution				Global time reduction
		Obj	Fit	Time		
Dim.	Nb. points			Without init. sol.	With init. sol.	
14	2000	2.54698	92.20%	0.57	0.49	12%
16	2000	2.56076	92.10%	2.66	2.54	4%
18	2000	3.18235	90.20%	16.89	13.82	18%
20	2000	2.52311	92.15%	81.53	60.17	26%

Table 6: L_∞-norm results on Sym6k series

Problem size		Exact solution				Global time reduction
		Obj	Fit	Time		
Dim.	Nb. points			Without init. sol.	With init. sol.	
6	10000	26.2138	88.96%	156.3	96.7	27%
6	20000	49.6972	88.74%	786.4	461.4	37%
6	30000	73.1058	89.11%	1774.4	1197.7	30%
6	40000	100.85	89.01%	3972.7	2326.4	39%
6	50000	116.317	88.95%	6004.2	3428.2	41%
6	60000	137.682	89.07%	7655.7	5050.1	32%
6	70000	159.39	89.23%	15215.0	7495.1	50%
6	80000	190.945	88.86%	21864.7	10639.1	51%
6	90000	202.645	89.02%	20072.5	11650.6	41%
6	100000	218.939	88.99%	35206.3	14020.3	60%

In addition to the four random problem series discussed above we solved, for the L_∞-norm, a single instance of 100000 points in 10 dimensions, generated with the same criteria as the Sym6d and Sym2k problem series. The solution time, accelerated by the inclusion of the heuristic solution, was about 84 CPU hours.

5.2 Real life instances

Tables 7 and 8 present the results obtained for the UCI problems, under the L_∞-norm and the L_2-norm, respectively. This instances are considerably smaller than our random problems, and the use of a heuristic bound does not yield net time savings in the exact solution under the L_∞-norm. Under the L_2-norm, however, the time savings are very significant in all but one problem.

Table 7: L_∞-norm results on UCI instances

Problem			Exact solution				Global
name	size		Obj	Fit	Time		time
	Dim	Nb. points			Without init. sol.	With init. sol.	red.
Cancer	9	683	0.78030	97.51%	1.1	1.06	-694%
Pima	8	768	5.32226	76.04%	8.1	7.72	-50%
Echocardiogram	7	74	0.58059	77.42%	0.1	0.09	-440%
Glass	9	214	0.01209	95.79%	0.4	0.42	-657%
Housing	13	506	0.65243	88.14%	30.4	29.05	-39%
Hepatitis	16	150	0.40146	90.00%	65.7	63.53	-8%

Table 8: L_2-norm results on UCI instances

Problem			Exact solution				Global
name	size		Obj	Fit	Time		time
	Dim	Nb. points			Without init. sol.	With init. sol.	red.
Cancer	9	683	2.06696	97.07%	156.2	57.0	53.38%
Pima	8	768	12.24314	75.39%	611.6	211.2	63.83%
Echocardiogram	7	74	1.20730	75.68%	4.6	1.3	56.06%
Glass	9	214	0.03114	95.33%	4.5	1.4	-72.12%
Housing	13	506	0.89714	84.39%	550.2	30.4	88.89%
Hepatitis	16	150	0.87113	88.00%	14181.3	50.7	99.57%

5.3 Cross validation

The techniques proposed in this paper allow us to compare the classification performance of separating planes obtained by minimizing different norms. Figure 2 summarizes the full-set fit for the UCI problems under the L_2- and L_∞-norms. We also include, for comparison, the results under the L_1-norm obtained by solving the corresponding $2n$ linear programs (see [9]). Figure 3 shows the 10-cross validation mean accuracy of the planes found under the three norms.

To our surprise, the plane obtained under the L_∞-norm performs at least as well as the others, both in full set and 10-cross accuracy, for all problems except *echocardiogram*, which is by far the smallest dataset we consider, with only 74 points.

We note that the L_1-norm plane, which is considerably easier to compute than the others, performs better than the L_2-norm plane in three cases and nearly as well in two others. This suggests that the L_1-norm could be a good bet on large problems for which the alternatives are impractical.

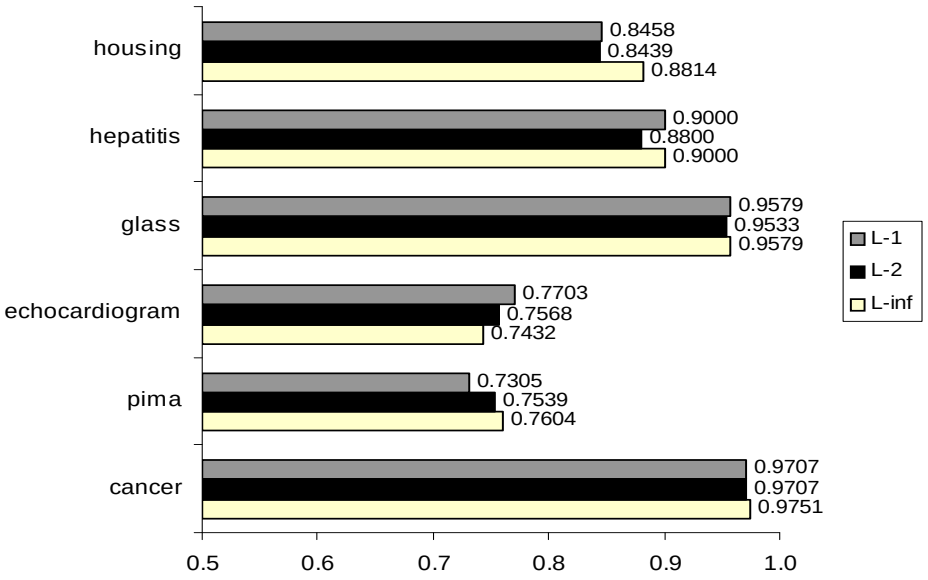

Figure 2: Full set accuracy

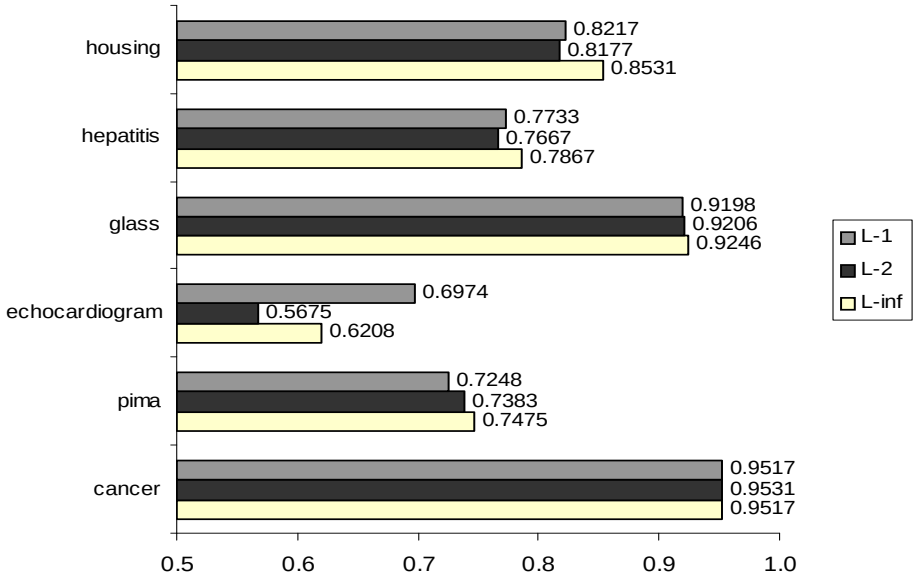

Figure 3: 10-cross validation

6 Conclusion

L_p-norm separation, a classic problem in unsupervised classification, presents important optimization challenges. Despite recent progress, practical techniques for the exact solution of cases other than the L_1-norm have remained unavailable.

We propose and implement two new approaches that make possible the exact solution of fairly large problems for the L_2-norm and L_∞-norm cases. We solve in reasonable computing times examples of up to 20000 points (in 6 dimensions) and 13 dimensions (with 2000 points). A solution in L_∞-norm was also found for an example of 100000 points in 10 dimensions.

We also show that, for sufficiently large problems, computation times can be substantially reduced by incorporating heuristic results in the exact solution process.

Several real-life instances from the UCI Repository are also considered, and we are able to compute and compare full set fit and generalization properties (as estimated by 10-cross validation) for the L_1, L_2 and L_∞-norms.

Acknowledgments: Work of Charles Audet was supported by NSERC grant 239436-01 and FCAR grant NC72792. Work of Pierre Hansen was supported by NSERC grant 105574-02 and FCAR grant 2002-ER-73226. Alejandro Karam is grateful to Mexico's CONACYT for support. Work of Sylvain Perron has been supported by NSERC graduate scholarship 195113 and FCAR graduate scholarship 67567.

A Data Set Details

We considered the instances from the UCI Machine Learning Repository [5] which either have only two classes or could be readily converted into a binary classification problem. We then retained those with very few or no categorical variables.

Cancer refers to the Wisconsin Breast Cancer database. Rows with missing attributes were deleted.

Pima refers to the Pima Indians Diabetes database.

For the *Echocardiogram* problem, all instances with missing labels were deleted, and missing attributes were replaced by the corresponding class means.

For the *Glass* database the two classes considered were window versus non-window sources.

Housing refers to the Boston Housing database.

In the *Hepatitis* database, all observations with more than 6 missing attributes were deleted, as were columns 16 and 18, which had too many missing entries. Missing observations were then replaced with column means (if continuous) or modes. Column 3 trivially separates the set, and was also removed.

Musicant's NDC generator is a MATLAB program. It locates randomly a given number of centers, assigns them to one of two classes by splitting the set with a randomly generated plane, and then produces multivariate normal points from these centers, using a randomly generated covariance matrix. This approach provides some more generality than one might have with other common practices. Note, however, that even within the class of normally distributed problems, some reasonable, interesting configurations (such as having a small cluster centered on the "wrong" side of the plane) are not spanned by NDC. These limitations are inevitable in any exercise with artificial data, and we feel that replicability is facilitated with the use of a publicly available generator. The number of centers was made to be equal to the dimension of the problem and the dispersion parameter $nExpandFactor$ was fixed at 15.

The Sym2k and Sym6d series, used for larger L_∞-norm instances, were constructed by fixing arbitrary centers, assigning one half of the points to each of them (Sym stands for symmetric) and generating independent columns from a normal distribution for each of them. The first center was fixed at the origin and the second one was set along the ray defined by a vector of ones of the appropriate dimension, at a distance adjusted as to keep the L_1-norm full set fit at about 89%. This criterion to approximate and control the difficulty of the problem was used because of the relative ease of obtaining the exact L_1-norm solution.

All databases were linearly standardized to the range $[0, 1]$.

The data sets are publically available at **www.gerad.ca/Charles.Audet**.

B Details on Curve Fitting for Solution Time Behavior

Among the models tried, the best fit for the case with 2000 points was obtained by $t = \kappa_1 \cdot e^{\kappa_2 \cdot n}$, where t is CPU time in seconds. For the problems on 6 dimensions, the best curve was $t = \kappa_3 \cdot (m + k)^{\kappa_4}$, where $m + k$ is the total number of points. The estimation details are as follows. For the L_∞-norm:

- $\kappa_1 = .0336$, $\kappa_2 = .8004$, $R^2 = .998$
- $\kappa_3 = .00000042$, $\kappa_4 = 2.3683$, $R^2 = .994$

For the L_2-norm:

- $\kappa_1 = .0246$, $\kappa_2 = 1.2409$, $R^2 = .985$
- $\kappa_3 = .0000044$, $\kappa_4 = 2.1267$, $R^2 = .974$

These estimates where performed considering together the data from NDC and Sym tables, without heuristic acceleration.

References

[1] F.A. Al-Khayyal and J.E. Falk. Jointly constrained biconvex programming. *Math. Oper. Res.*, 8(2):273–286, 1983.

[2] C. Audet, P. Hansen, B. Jaumard, and G. Savard. A branch and cut algorithm for nonconvex quadratically constrained quadratic programming. *Math. Program.*, 87(1, Ser. A):131–152, 2000.

[3] G. Caporossi, P. Hansen, and A. Karam. Arbitrary-norm plane separation by variable neighborhood search. *in preparation*, 2004.

[4] T.M. Cavalier, J.P. Ignizio, and A.L. Soyster. Discriminant analysis via mathematical programming: certain problems and their causes. *Comput. Oper. Res.*, 16(4):353–362, 1989.

[5] J.M. Christopher and P.M. Murphy. UCI repository of machine learning databases, 1998.

[6] G. Desaulniers, J. Desrosiers, and M.M. Solomon. Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems. In C.C. Ribeiro and P. Hansen, editors, *Essays and surveys in metaheuristics (Angra dos Reis, 1999)*, Oper. Res./Comput. Sci. Interfaces Ser., pages 309–324. Kluwer Acad. Publ., Boston, MA, 2002.

[7] G. Fung and O.L. Mangasarian. Proximal support vector machine classifiers. In *Knowledge Discovery and Data Mining*, pages 77–86, 2001.

[8] P. Hansen, J. Brimberg, D. Urosevic, and N. Mladenovic. Primal-Dual Variable Neighborhood Search for Bounded Heuristic and Exact Solution of the Simple Plant Location

Problem. Les Cahiers du GERAD G-2003-64, Groupe d'études et de recherche en analyse des décisions, october 2003.

[9] O.L. Mangasarian. Arbitrary-norm separating plane. *Operations Research Letters*, 24(1–2):15–23, 1999.

[10] O.L. Mangasarian and D.R. Musicant. Active support vector machine classification. In *NIPS*, pages 577–583, 2000.

[11] P. Marcotte, G. Marquis, and G. Savard. A new implicit enumeration scheme for the discriminant analysis problem. *Computers and Operations Research*, 26(6):625–639, 1995.

[12] E. Melachrinoudis. An analytical solution to the minimum $\mathcal{L}_p$-norm of a hyperplane. *Journal of Mathematical Analysis and Applications*, 211:172–179, 1997.

[13] D.R. Musicant. NDC: normally distributed clustered datasets, 1998.

[14] S. Perron. *Applications jointes de l'optimisation combinatoire et globale*. PhD thesis, École Polytechnique de Montréal, 2004.

[15] H.D. Sherali and A. Alameddine. A new reformulation-linearization technique for bilinear programming problems. *J. Global Optim.*, 2(4):379–410, 1992.

[16] H.D. Sherali and C.H. Tuncbilek. A global optimization algorithm for polynomial programming problems using a reformulation-linearization technique. *J. Global Optim.*, 2(1):101–112, 1992.

[17] A. Stam. Nontraditional approaches to statistical classification: Some perspectives on l-p-norm methods. *Annals of Oper. Res.*, 74:1–36, 1997.