M a t h e m a t i c a
B a l k a n i c a

New Series Vol. 23, 2009, Fasc. 3-4

Interior Point Methods for Self-Dual Linear Optimization Problems Based on Kernel Functions

Mohamed El Ghami[1] *and Trond Steihaug*[2]

Presented by M. El Ghami

In this paper we present the theory and practical aspects of implementing the path following interior point methods for Self-dual linear optimization problems based on kernel functions. We investigate the influence of the choice of the kernel functions on the theoretical complexity results and on the computational behavior of the generic primal-dual algorithm for Linear Optimization. We find that the finite kernel function gives the best results for more than 50 % of the tested problems compared to the standard log-barrier method.

Key Words: Kernel function, Interior-point Algorithm, Large-update, Iteration number.

AMS Subject Classification: 90C51, 90C05

1. Introduction

The study of Interior-Point Methods (IPMs) is currently one of the most active research areas in optimization. The name "interior-point methods" has originated from the fact that the points generated by the algorithms all lie in the interior of the feasible region. This is in contrast with the famous and well-established simplex method where the iterates move along the boundary of the feasible region from one extreme point to another. Nowadays, IPMs provide a powerful tool for solving Linear Optimization (LO) problems. They become an alternative to the simplex based methods and are particularly useful in applications where the size of the problem is very large.

Most IPMs can be categorized into two classes: large-update and small update IPMs depending on the strategy used in the algorithm to update the parameter with respect to the duality gap. It has been proven that the small-update IPMs has the best known worst-case complexity result $O\left(\sqrt{n}\log\frac{n}{\epsilon}\right)$, where n is the size of the problem and ϵ is the accuracy. However, large-update

IPMs perform much better in practice with a best known worst-case complexity $O\left(\sqrt{n}\log n \log \frac{n}{\epsilon}\right)$.

The aim of this paper is to explore the use of path-following IPM algorithms based on kernel functions in practise. For this purpose we developed in MATLAB® a practical variant of an IPM solver for LO problems so-called LP-IPM. We will solve by LP-IPM solver various LO problems from Netlib library testing problems [7] that has become the standard benchmark for testing and comparing LO algorithms. The results from our experiments with LP-IPM demonstrate that path-following algorithms are not only of theoretical interest but also can be effective IPM based software solution for LO.

The paper is organized as follows. In Section 2 we start introducing linear optimization problem and its transformation to self dual problem. In Section 3 where we discuss the classical search direction and the new search direction based on kernel functions. Numerical results i given in Section 4. In the final section we conclude with some remarks.

We use the following notational conventions. Throughout the paper, $\|\cdot\|$ denotes the 2-norm of a vector. The nonnegative orthant and positive orthant are denoted as $\mathbf{R}^n_+$ and $\mathbf{R}^n_{++}$, respectively. Finally, if $z \in \mathbf{R}^n_+$ and $f : \mathbf{R}_+ \to \mathbf{R}_+$, then $f(z)$ denotes the vector in $\mathbf{R}^n_+$ whose i-th component is $f(z_i)$, with $1 \leq i \leq n$.

2. Linear Optimization Problems

In this paper we consider the following *canonical linear* optimization problem:

$$\begin{array}{lrcl} \min & c^T x & & \\ s.t. & Ax & \geq & b, \\ & 0 & \leq & x_i \leq u_i, \quad i \in \mathcal{I}, \\ & 0 & \leq & x_j, \qquad j \in \mathcal{J}, \end{array}$$

where $A \in \mathbf{R}^{m\times n}$ is a real $m \times n$ matrix of rank m, $x, c \in \mathbf{R}^n$, $b \in \mathbf{R}^m$, $u_i \in \mathbf{R}$, with $\mathcal{I}$ and $\mathcal{J}$ index sets such that $\mathcal{I} \bigcup \mathcal{J} = \{1, 2, ..., n\}$ and $\mathcal{I} \bigcap \mathcal{J} = \emptyset$. Without loss of generality the last two inequalities can be written as

$$\bar{F}x \leq u, \;\; x \geq 0, \quad \bar{F} \in \mathbf{R}^{m_f \times n},$$

where $m_f = |\mathcal{I}|$, the rows of $\bar{F}$ are unit vectors and $u = (u_1, u_2, ..., u_{m_f})^T$. Let

$F := -\bar{F}$ and $b_u := -u$, then the above problem can be written as:

$$\begin{array}{lllll} (\mathrm{P}) & \min & c^T x & & \\ & s.t. & Ax & \geq & b, \\ & & Fx & \geq & b_u \\ & & x & \geq & 0, \end{array}$$

the dual problem of (P) is given by

$$\begin{array}{lllll} (\mathrm{D}) & \max & b^T y - b_u^T y_u & & \\ & s.t. & A^T y - F^T y_u & \leq & c, \\ & & y, y_u & \geq & 0. \end{array}$$

For solving (P) and (D), we use the *self-dual embedding* model [14]. We transfer the original problems to a slightly larger LO problem, called augmented problem that always has an optimal solution. The optimal solution of the augmented problem either proves that the original problem does not have an optimal solution or it exists and easily can be converted to an optimal solution of (P) and (D). The embedding problem can be formulated as follows. Let $x^0 \in \mathbf{R}^n_{++}$, $y^0 \in \mathbf{R}^m_{++}$, ${y_u}^0 \in \mathbf{R}^{m_f}_{++}$ and $k^0, \nu^0 \in \mathbf{R}_{++}$ be the given initial interior solution. Define $r_u \in \mathbf{R}^{m_f}$, $r_p \in \mathbf{R}^m$, $r_d \in \mathbf{R}^n$ to be the errors at this initial interior solution, and $r_g, \beta \in \mathbf{R}$, $\bar{n} \in \mathbf{R}_{++}$ be parameters defined as follows

$$\begin{array}{rcl} r_u & = & {y_u}^0 + Fx^0 - b_u, \\ r_p & = & y^0 - Ax^0 + b, \\ r_d & = & x^0 - F^T x^0 + A^T y^0 - c, \\ \beta & = & 1 + b_u^T {y_u}^0 - b^T y^0 + c^T x^0, \\ \bar{n} & = & m_f + m + n + 2. \end{array}$$

The self-dual embedding model can be written as follows

$$\begin{array}{llllll} (SP_0) & \min & & & & \bar{n}\nu \\ & s.t. & & -Fx & +b_u k & +r_u\nu \geq 0 \\ & & & Ax & -bk & +r_p\nu \geq 0 \\ & F^T y_u & -A^T y & & +ck & +r_d\nu \geq 0 \\ & -b_u^T y_u & +b^T y & -c^T x & & +\beta\nu \geq 0 \\ & -r_u^T y_u & -r_p^T y & -r_d^T x & -\beta & \geq \bar{n} \\ & y_u \geq 0, & y \geq 0, & x \geq 0, & k \geq 0, & \nu \geq 0. \end{array}$$

The first four constraints in (SP_0), with $k = 1$ and $\nu = 0$, represent primal and dual feasibility with $y_u \geq 0$, $y \geq 0$, $x \geq 0$ and reversed weak duality

inequality associated with (P) and (D). A normalizing variable ν is added to achieve feasibility of the initial point $(x^0, y^0, {y_u}^0)$ and the fifth constraint is added to ensure self-duality. Let M, z and q be defined as follows

$$M := \begin{bmatrix} 0_{m_f \times m_f} & 0_{m_f \times m} & -F & b_u & r_u \\ 0_{m \times m_f} & 0_{m \times m} & A & -b & r_p \\ F^T & -A^T & 0_{n \times n} & c & r_d \\ -b_u^T & b^T & -c^T & 0 & \beta \\ -r_u^T & -r_p^T & -r_d^T & \beta & 0 \end{bmatrix}, z := \begin{bmatrix} y_u \\ y \\ x \\ k \\ \nu \end{bmatrix}, q := \begin{bmatrix} 0_{m_f} \\ 0_m \\ 0_n \\ 0 \\ \bar{n} \end{bmatrix}.$$

Then (SP_0) can be written as follows

$$(SP) \qquad \min \left\{ q^T z \ : \ Mz \geq -q, \, z \geq 0 \right\},$$

where $M \in \mathbf{R}^{\bar{n} \times \bar{n}}$ is a skew-symmetric ($M^T = -M$) and (SP) is a self-dual LO problem. We associate to any vector $z \in \mathbf{R}^{\bar{n}}$ its *slack vector* $s(z)$ as follows $s(z) := Mz + q$. Then z is feasible for (SP) if and only if $z \geq 0$ and $s(z) \geq 0$. Solving (SP) is equivalent to solve the following system:

$$\begin{aligned} Mz + q &= s(z), \\ zs(z) &= 0 \end{aligned} \tag{1}$$

where $zs(z)$ denotes the coordinatewise product of the vectors z and $s(z)$, i.e.

$$zs(z) = [z_1 s(z)_1; z_2 s(z)_2; ...; z_n s(z)_n].$$

We shall also use the notation

$$\frac{z}{s(z)} = [\frac{z_1}{s(z)_1}, \frac{z_2}{s(z)_2}, ..., \frac{z_n}{s(z)_n}],$$

for $z, s(z) \in \mathbf{R}^n$ such that $s(z)_i \neq 0$, $i = 1, 2, ..., n$.

3. Interior-point Methods for the Self-dual Model

In this section we introduce the new search direction based on the kernel functions. First we recall the classical search direction and introduce the scaled Newton direction.

3.1. Classical search direction

The basic idea underlying IPMs is to replace the second equation in (1) by the nonlinear equation $zs = \mu \mathbf{1}$, with parameter $\mu > 0$, $s = s(z)$ and $\mathbf{1}$ denoting the all-one vector $(1, 1, ..., 1)^T$. The system (1) becomes:

$$\begin{aligned} Mz + q &= s, \quad z \geq 0, s \geq 0, \\ zs &= \mu \mathbf{1}. \end{aligned} \tag{2}$$

We assume that there exist (z^0, s^0) such that [14]

$$Mz^0 + q = s^0, \quad z^0 > 0, \quad s^0 > 0. \tag{3}$$

which is known as the *interior-point condition* (IPC) [14].

Without loss of generality we may assume that $z^0 = \mathbf{1}$. Then its slack vector $s^0 = s(z^0) = \mathbf{1}$. Observe that these z^0 and s^0 solve (2) when $\mu = 1$. If the IPC holds, the parameterized system (2) has a unique solution $z(\mu)$ for each $\mu > 0$; $z(\mu)$ is called the μ-center of (SP). The set of μ-centers (with $\mu > 0$) defines a homotopic path, which is called *the central path* of (SP). If $\mu \to 0$ then the limit of the central path exists. This limit satisfies the complementarity condition, and hence yields optimal solutions for (SP) [14]. IPMs follow the central path approximately. Let us briefly indicate how this works. at the start of our algorithm let $\mu = 1$, with $z(1) = s(1) = \mathbf{1}$. We then decrease μ to $\mu_+ := (1 - \theta)\mu$, for some $\theta \in (0, 1)$ and apply Newton's method to iteratively solve the non-linear equations (1). Thus we need displacements $\Delta z, \Delta s$ such that $z + \Delta z$, $s + \Delta s$ satisfy:

$$\begin{aligned} M(z + \Delta z) + q &= s + \Delta s, \quad z + \Delta z \geq 0, s + \Delta s \geq 0, \\ (z + \Delta z)(s + \Delta s) &= \mu \mathbf{1}. \end{aligned} \tag{4}$$

Neglecting for the moment the inequality constraints, this system can be rewritten as:

$$\begin{aligned} M\Delta z &= \Delta s, \\ s\Delta z + z\Delta s + \Delta z \Delta s &= \mu \mathbf{1} - zs. \end{aligned} \tag{5}$$

Neglect the quadratic term $\Delta z \Delta s$, so for each step we have to solve for Δz and Δs in the following Newton system

$$\begin{aligned} M\Delta z - \Delta s &= 0, \\ s\Delta z + z\Delta s &= \mu \mathbf{1} - zs. \end{aligned} \tag{6}$$

Since M has full row rank, the system (6) uniquely defines a search direction $(\Delta z, \Delta s)$ for any $z > 0$ and $s(z) > 0$; this is the so-called Newton direction and this direction is used in all existing implementations of the primal-dual method. The first equation in (6) take care of the feasibility after a step along the Newton direction, whereas the second equation serves to drive the new iterates to the $\mu-$centers. The second equation is called the *centering equation*. By taking a step along the search direction, with the step size defined by a line search rule,

one constructs a new double $(z, s(z))$, with $z > 0$ and $s(z) > 0$. If necessary, we repeat the procedure until we find iterates that are close enough to $z(\mu)$. Then μ is again reduced by the factor $1-\theta$ and we apply Newton's method targeting at the new $\mu-$centers, and so on. This process is repeated until μ is small enough, say until $\bar{n}\mu \leq \epsilon$; at this stage we have found $\epsilon-$solutions of the problem (SP).

3.2. The scaled Newton direction

Following the approach in [2, 14], we define the vector v by

$$v := \sqrt{\frac{zs}{\mu}}. \tag{7}$$

Note that the pair $(z, s(z))$ coincide with the μ-center $(z(\mu), s(z)(\mu))$ if and only if $v = \mathbf{1}$. Introducing the notations

$$\bar{M} := \mu DMD, \text{ where } D := \operatorname{diag}(d), \ d := \sqrt{\frac{z}{s}}, \tag{8}$$

and defining the *scaled search directions* d_z and d_s by

$$d_z := \frac{v\Delta z}{z}, \qquad d_s := \frac{v\Delta s}{s}, \tag{9}$$

the system (6), can be rewritten as follows

$$\begin{aligned} \bar{M}d_z - d_s &= 0, \\ d_z + d_s &= v^{-1} - v. \end{aligned} \tag{10}$$

Note that d_z and d_s are orthogonal, because d_z belongs to the null space of the matrix $\bar{M}$ and d_s to its row space. Hence $d_z = d_s = 0$ if and only if $v^{-1} - v = 0$, which is equivalent to $v = \mathbf{1}$. We conclude that $d_z = d_s = 0$ holds if and only if the pair $(z, s(z))$ coincides with the μ-center $(z(\mu), s(z)(\mu))$. The second equation in (10) is called the *scaled centering equation.* Note that the right-hand side $v^{-1} - v = -\nabla\Psi_c(v)$ where

$$\Psi_c(v) := \sum_{i=1}^{n} \left(\frac{v_i^2 - 1}{2} - \log v_i \right), \tag{11}$$

and the Hessian $\nabla^2\Psi_c(v) = \operatorname{diag}\left(\mathbf{1} + v^{-2}\right)$. Since $\nabla^2\Psi_c(v)$ is positive definite, then $\Psi_c(v)$ is strictly convex. Moreover, $\nabla\Psi_c(\mathbf{1}) = 0$, it follows that $\Psi_c(v)$ attains its minimal value at $v = \mathbf{1}$, with $\Psi_c(\mathbf{1}) = 0$. Thus it follows that $\Psi_c(v)$ is nonnegative and vanishes if and only if $v = \mathbf{1}$.

3.3. New search direction from kernel functions

In this subsection we introduce the new search direction. First we start with the definition of a kernel function.

Definition 3.1. A twice differentiable function $\psi : (0, \infty) \longrightarrow [0, \infty)$ is called a *kernel function* if

$$\psi(1) = 0, \;\; \psi'(1) = 0, \;\; \psi''(t) > 0, \;\; \forall t > 0.$$

Due to $\psi(1) = 0, \;\; \psi'(1) = 0$ any kernel function is determined by its second derivative:

$$\psi(t) = \int_1^t \int_1^\xi \psi''(\zeta)\, d\zeta d\xi. \tag{12}$$

Following the approach in [3, 4, 5, 13] we replace the scaled barrier function $\Psi_c(v)$ by a strictly convex function $\Psi(v)$, $v \in \mathbf{R}^{\bar{n}}_{++}$ such that $\Psi(v)$ is minimal at $v = \mathbf{1}$ and $\Psi(\mathbf{1}) = 0$. Thus the new scaled centering equation becomes

$$d_z + d_s = -\nabla\Psi(v). \tag{13}$$

Note that $d_z = 0$ and $d_s = 0$ if and only if $v = \mathbf{1}$. To simplify matters we restrict ourselves to the case where $\Psi(v)$ is separable with identical coordinate functions. We can write

$$\Psi(v) = \sum_{i=1}^{\bar{n}} \psi(v_i), \tag{14}$$

where $\psi(t) \; : \; \mathcal{D} \to \mathbf{R}_+$, with $\mathbf{R}_{++} \subseteq \mathcal{D}$, is a kernel function and in addition the following four conditions are satisfied.

$$t\psi''(t) + \psi'(t) \;>\; 0, \quad t < 1, \tag{15-a}$$

$$\psi'''(t) \;<\; 0, \quad t > 0, \tag{15-b}$$

$$2\psi''(t)^2 - \psi'(t)\psi'''(t) \;>\; 0, \quad t < 1, \tag{15-c}$$

$$\psi''(t)\psi'(\beta t) - \beta\psi'(t)\psi''(\beta t) \;>\; 0, \quad t > 1, \quad \beta > 1. \tag{15-d}$$

Observe that $\psi_1(t) = \frac{t^2-1}{2} - \log t$, is the kernel function yielding the Newton direction, as defined by (10). In this general framework we call $\Psi(v)$ a *scaled barrier function.*

$$\Psi(v) = \sum_{i=1}^{\bar{n}} \psi\left(v_i\right) = \sum_{i=1}^{\bar{n}} \psi\left(\sqrt{\frac{z_i s_i}{\mu}}\right). \tag{16}$$

One may easily verify that by application of this definition to the kernel function $\psi_1(t)$, we obtain — up to a constant factor and a constant term — the classical logarithmic barrier function.

Generic Path-Following Algorithm for LO

Input:
A kernel function $\psi(t)$;
a threshold parameter $\tau > 0$;
an accuracy parameter $\epsilon > 0$;
a barrier update parameter θ, $0 < \theta < 1$;
begin
$z := \mathbf{1}$; $s := \mathbf{1}$; $\mu := 1$; $v := \mathbf{1}$;
while $n\mu \geq \epsilon$ **do**
begin
$\mu := (1 - \theta)\mu$;
while $\Psi(v) > \tau$ **do**
begin
Solve $(\Delta z, \Delta s)$
$z := z + \alpha \Delta z$;
$s := s + \alpha \Delta s$;
$v := \sqrt{\frac{zs}{\mu}}$;
end
end
end

Figure 1: Algorithm for LO

Any proximity function $\Psi(v)$ gives rise to a primal-dual IPM, as described in Figure 1. With $\bar{M}$ as defined in (8), the search direction in the algorithm is obtained by solving the system

$$\begin{aligned} \bar{M} d_z - d_s &= 0, \\ d_z + d_s &= -\nabla \Psi(v), \end{aligned} \tag{17}$$

and then computing Δz and Δs by using (9). Solving (17) is equivalent to solve

the following equation

$$(I + \bar{M})d_z = -\nabla\Psi(v), \tag{18}$$

where $I \in \mathbf{R}^{\bar{n}\times\bar{n}}$ denotes identity matrix, and then we compute d_s from

$$d_s = -d_z - \nabla\Psi(v). \tag{19}$$

In this way we avoid one of the disadvantages of the predictor-corrector [9] strategy and we solve the Newton system only once per iteration.
Note that $I + M$ is a nonsingular matrix, and then $(I + M)^{-1}$ exists and we can solve (18) easily.

3.4. Scheme for analyzing the kernel functions

In this section we derive the theoretical complexity results for primal-dual interior point methods based on kernel functions. In Figure 2 we give the scheme for analyzing a kernel-function-based algorithm and we derive the theoretical complexity results for a given function. This scheme is based in the recent results in [2, 4]. As examples of kernel functions we give in Table 1 ten different kernel functions that we will use for our numerical test in the next section. When there are parameters involved in the definition of a kernel function we used the parameters $p \in [0, 1]$, $q \geq 1$, and $\sigma \geq 1$. The first nine kernel functions satisfying the conditions (15-a), (15-b), (15-c) and (15-d). But for ψ_{10} the second inequality in (15-a) ($t \geq \frac{1}{\sigma}$) is more restrictive than the first nine kernel functions, where we have the same inequality for all $t > 0$.

In the analysis, we will ensure that we may apply inequality (15-a) in the region were the iterates of the algorithm occur [4]. Note that this last kernel function is not like the first nine kernel functions that had the property that $\lim_{t\downarrow 0}\psi(t) = \infty$ and $\lim_{t\to\infty}\psi(t) = \infty$. ψ_{10} has the second property, but it fails to have the first property, because

$$\lim_{t\downarrow 0}\psi(t) = \psi(0) = \frac{e^{\sigma} - 1}{\sigma} - \frac{1}{1 + p} < \infty.$$

This means that if either z or s approaches the boundary of the feasible region, then $\Psi(v)$ converges to a finite value, depending on the value of σ.

Given the kernel function $\psi(t)$, the parameters τ, θ, and the step size α should be chosen in such a way that the algorithm is "optimized" in the sense that the number of iterations required by the algorithm is as small as possible. Obviously, the resulting iteration bound will depend on the kernel function, and our main task is to find a kernel function that minimizes the iteration bound in theory and in practice.

We denote by $\varrho : [0,\infty) \to [1,\infty)$ and $\rho : [0,\infty) \to (0,1]$ the inverse functions of $\psi(t)$ for $t \geq 1$, and $-\frac{1}{2}\psi'(t)$ for $t \leq 1$, respectively. In other words

$$s = \psi(t) \quad \Leftrightarrow \quad t = \varrho(s), \quad t \geq 1, \tag{20}$$

$$s = -\tfrac{1}{2}\psi'(t) \quad \Leftrightarrow \quad t = \rho(s), \quad t \leq 1. \tag{21}$$

In the analysis of the algorithm we use a norm-based proximity measure $\delta(v)$, defined by

$$\delta(v) = \frac{1}{2}\|\psi'(v)\| = \frac{1}{2}\sqrt{\sum_{i=1}^{n} \psi'(\lambda_i(v))^2} = \frac{1}{2}\left\|d_z + d_s\right\|. \tag{22}$$

Following the argument in [2, 4], we may simply apply the scheme in Figure 2 to each of the ten kernel functions. The corresponding iteration complexity are given in the third column in Table 1.

i	kernel function ψ_i	iteration complexity	ref.
1	$\frac{t^2-1}{2} - \log t$	$O\left(n \log \frac{n}{\epsilon}\right)$	[6]
2	$\frac{t^2-1}{2} + \frac{t^{1-q}-1}{q(q-1)} - \frac{q-1}{q}(t-1)$	$O\left(qn^{\frac{q+1}{2q}} \log \frac{n}{\epsilon}\right)$	[13]
3	$\frac{t^2-1}{2} + \frac{(e-1)^2}{e}\frac{1}{e^t-1} - \frac{e-1}{e}$	$O\left(n^{\frac{3}{4}} \log \frac{n}{\epsilon}\right)$	[2]
4	$\frac{1}{2}\left(t - \frac{1}{t}\right)^2$	$O\left(n^{\frac{2}{3}} \log \frac{n}{\epsilon}\right)$	[11]
5	$\frac{t^2-1}{2} + e^{\frac{1}{t}-1} - 1$	$O\left(\sqrt{n}\ \log^2 n \log \frac{n}{\epsilon}\right)$	[2]
6	$\frac{t^2-1}{2} - \int_1^t e^{\frac{1}{\xi}-1} d\xi$	$O\left(\sqrt{n}\ \log^2 n \log \frac{n}{\epsilon}\right)$	[2]
7	$\frac{t^2-1}{2} + \frac{t^{1-q}-1}{q-1}$	$O\left(qn^{\frac{q+1}{2q}} \log \frac{n}{\epsilon}\right)$	[12]
8	$\frac{t^{1+p}-1}{1+p} - \log t$	$O\left(n \log \frac{n}{\epsilon}\right)$	[5]
9	$\frac{t^{p+1}-1}{p+1} + \frac{t^{1-q}-1}{q-1}$	$O\left(qn^{\frac{p+q}{q(1+p)}} \log \frac{n}{\epsilon}\right)$	[3]
10	$\frac{t^{1+p}-1}{p+1} + \frac{e^{\sigma(1-t)}-1}{\sigma}, \sigma \geq 1$	$O\left(\sqrt{n} \log n \log \frac{n}{\epsilon}\right)$	[4]

Table 1: Example of kernel functions and complexity results for large-update methods.

Step 0: Specify a kernel function $\psi(t)$; an update parameter θ, $0 < \theta < 1$; a threshold parameter τ; and an accuracy parameter ϵ.

Step 1: Solve the equation $-\frac{1}{2}\psi'(t) = s$ to get $\rho(s)$, the inverse function of $-\frac{1}{2}\psi'(t), t \in (0, 1]$. If the equation is hard to solve, derive a lower bound for $\rho(s)$.

Step 2: Calculate the decrease of $\Psi(v)$ during an inner iteration in terms of δ for the default step size α from

$$f(\alpha) := \Psi\left(\sqrt{(v + \alpha d_x)\,(v + \alpha d_s)}\right) - \Psi\,(v) \leq -\frac{\delta^2}{\psi''(\rho(2\delta))}.$$

Step 3: Solve the equation $\psi(t) = s$ to get $\varrho(s)$, the inverse function of $\psi(t), t \geq 1$. If the equation is hard to solve, derive lower and upper bounds for $\varrho(s)$.

Step 4: Derive a lower bound for δ in terms of $\Psi(v)$ by using

$$\delta(v) \geq \tfrac{1}{2}\psi'\left(\varrho\left(\Psi(v)\right)\right).$$

Step 5: Using the results of step 3 and step 4 find a valid inequality of the form

$$f(\alpha) \leq -\kappa\Psi(v)^{1-\gamma}$$

for some positive constants κ and γ, with $\gamma \in (0, 1]$ as small as possible.

Step 6: Calculate the upper bound of Ψ_0 from

$$\Psi_0 \leq L_\psi(n, \theta, \tau) := n\psi\left(\frac{\varrho\left(\frac{\tau}{n}\right)}{\sqrt{1-\theta}}\right) \leq \frac{n}{2}\psi''(1)\left(\frac{\varrho(\frac{\tau}{n})}{\sqrt{1-\theta}} - 1\right)^2.$$

Step 7: Derive an upper bound for the total number of iterations by using that

$$\frac{\Psi_0^\gamma}{\theta\kappa\gamma}\log\frac{n}{\epsilon}$$

is an upper bound for this number.

Step 8: Set $\tau = O(n)$ and $\theta = \Theta(1)$ to calculate a complexity bound for large-update methods.

Figure 2: Scheme for analyzing a kernel-function-based algorithm.

4. Numerical results

The aim of this section is to investigate the influence of the choice of the kernel function $\psi(t)$ on the computational behavior of the generic primal-dual algorithm for LO, as given in Figure 1. The kernel functions that we used in our experiments are the ten kernel functions ψ_i, $i \in \{1, \ldots, 10\}$, as presented in Table 1. When there are parameters involved in the definition of a kernel function we used several values of these parameters as indicated in Table 2 below. These values were chosen after some preliminary experiments that showed that these values gave the most promising iteration counts for the respective kernel functions. This left us with 18 different kernel functions. For

Kernel function	Parameter values
ψ_2	$q \in \{1.5, 2\}$
ψ_7	$q \in \{1.5, 2\}$
ψ_8	$p = 0.8$
ψ_9	$(p, q) \in \{(0.5, 2),\ (0.8, 1.5),\ (0.8, 2)\}$
$\psi_{10} := \psi_{p,\sigma}$	$(p, \sigma) \in \{(0.5, 1), (0.8, 1), (1, 1)\}$ and$(p, \sigma) \in \{(1, 1.5), (1, 2)\}$

Table 2: Choice of parameters.

the test problems we used problems from the well-known Netlib library.To limit the number of test problems we used only the problems in this library that are known to have optimal solutions. This left us with 95 test problems. To get a first impression of the iteration bounds for the several kernel functions we applied the algorithm to a selection of ten of these problems.

After this first round of experiments we run all the mentioned 95 problems from the Netlib library with the kernel functions that gave the best performance on the aforementioned set of ten problems. We used a straightforward implementation of our algorithm in MATLAB ® [8]. The self-dual embedding model [14] is used to enable the start of the algorithm as indicated in Figure 1, namely with $z = s = \mathbf{1}$ and $\mu = 1$. Our experiments were performed on a standard PC with a Pentium 4 processor and with 1 GB internal memory. Since we wanted to compare iteration numbers for several kernel functions, and since these numbers depend on the parameters[1] τ, θ and the accuracy parameter ϵ. In this way the iteration numbers depend only on the kernel function and the problem instance.

[1]We fixed $\tau = 1$, $\theta = 0.99$ and $\epsilon = 10^{-8}$ in our experiments.

The results of the first round of experiments are given in two tables (Table 3 and Table 4). For each of the ten problems we used **bold** font to highlight the best, i.e., the smallest, iteration number. This information is summarized in Table 5, which gives for each of the ten problems the smallest iteration number, and for which kernel function(s) this was achieved. From Table 5 we conclude that the smallest iteration numbers were realized by five kernel functions. For these five kernel functions we solved in the second round the mentioned 95 problems in the Netlib library. The results of the second round are listed in three tables (Table 6 to Table 8).

LP Problem	Number of iterations for ψ_1, ψ_2, ψ_3, ψ_4, ψ_4, ψ_5, ψ_6 ψ_7 and ψ_8									
	ψ_1	ψ_2	ψ_2	ψ_3	ψ_4	ψ_5	ψ_6	ψ_7	ψ_7	ψ_8
		$q = 1.5$	$q = 2$					$q = 1.5$	$q = 2$	0.8
ADLITTLE	**23**	**23**	24	39	25	?	24	**23**	25	25
AFIRO	**16**	**16**	17	28	17	?	18	**16**	17	19
DEGEN2	**24**	26	26	46	25	?	27	25	25	27
DEGEN3	**30**	31	33	71	34	?	37	**30**	34	32
Grow15	**37**	39	38	77	41	?	43	39	41	39
MAROS	74	76	82	90	87	?	92	69	87	71
SC105	**18**	19	19	36	21	?	19	19	21	22
SC205	**22**	**22**	23	36	24	45	23	23	24	23
SCTAP2	**26**	28	30	41	31	?	32	27	31	30
SHELL	**46**	49	50	81	52	?	55	49	52	51

Table 3: Iteration numbers for ψ_i, $i = 1, ..., 8$.

In the first round we encountered a problem with kernel function ψ_5, indicated by question marks in the corresponding column of Table 3. The reason is the occurrence of the expression $e^{\frac{1}{t}}$ in the definition of this kernel function. For values of t smaller than ≈ 0.0014 the value of this expression goes beyond the size of numbers that can be handled by MATLAB. Since such small values occur in the vector v during the execution of the algorithm for some of the test problems, the programme failed to run for nine of the ten problems in the first round. Only for SC205 it found the solution, but when comparing the number of iterations with the iteration count for the other kernel functions, the result is not very promising.

From the first four tables we may draw a few conclusions. For ψ_8, it becomes clear that smaller values of the parameter p influence the iteration count negatively. Hence, $p = 1$ seems to be the best possible choice, which gives ψ_1, the kernel function of the logarithmic barrier function. Furthermore, the kernel functions ψ_3, ψ_4, ψ_5, ψ_6, ψ_8 and ψ_9 never give the smallest iteration number. Special attention deserves the finite barrier kernel function $\psi_{p,\sigma}$. This

LP Problem	Number of iterations for $\psi_{9(p,q)}$ and $\psi_{10} = \psi_{p.\sigma}$							
	ψ_9	ψ_9	ψ_9	ψ_{10}	ψ_{10}	ψ_{10}	ψ_{10}	ψ_{10}
	$(0.5, 2)$	$(0.8, 1.5)$	$(0.8, 2)$	$(0.5, 1)$	$(0.8, 1)$	$(1, 1)$	$(1, 1.5)$	$(1, 2)$
ADLITTLE	35	27	29	29	26	24	24	25
AFIRO	29	20	21	24	20	**16**	17	17
DEGEN2	42	29	31	34	28	**24**	26	26
DEGEN3	47	33	37	39	36	32	31	31
GROW15	55	43	46	47	40	**37**	38	38
MAROS	95	80	80	77	77	**62**	64	68
SC105	33	22	25	28	22	**18**	19	20
SC205	38	26	28	31	24	**22**	**22**	23
SCTAP2	46	31	34	35	31	**26**	27	27
SHELL	67	52	56	57	53	50	52	51

Table 4: Iteration numbers for ψ_9 in the first four columns and for ψ_{10} in the last five columns.

Problem	Best It	Kernel functions
ADLITTLE	23	ψ_1, $\psi_2(q = 1.5)$, $\psi_7(q = 1.5)$
AFIRO	16	ψ_1, $\psi_2(q = 1.5)$, $\psi_7(q = 1.5)$, $\psi_{1,1}$
DEGEN2	24	ψ_1, $\psi_{1,1}$
DEGEN3	30	ψ_1, $\psi_7(q = 1.5)$
GROW15	37	ψ_1, $\psi_{1,1}$
MAROS	62	$\psi_{1,1}$
SC105	18	ψ_1, $\psi_{1,1}$
SC205	22	ψ_1, $\psi_2(q = 1.5)$, $\psi_{1,1}$, $\psi_{1,1.5}$
SCTAP2	26	ψ_1, $\psi_{1,1}$
SHELL	46	ψ_1

Table 5: Smallest iteration numbers and corresponding kernel function(s)

kernel function differs from the other kernel functions in the sense that it has a finite value at the boundary of the feasible region [4]. The results in Table 4 show that for $\psi_{1,1}$ the iteration numbers in all ten cases are the same (or almost the same) as for the kernel function ψ_1 of the classical logarithmic barrier function.

In the second round we used only the kernel functions that appear in the second column of Table 5, namely ψ_1, ψ_2 and ψ_7, both with $q = 1.5$, and $\psi_{1,1}$ and $\psi_{1,1.5}$. The iteration counts for the 95 problems are listed in Table 6, and Table 8. The results in these three tables justify the conclusion that the kernel functions $\psi_{p,\sigma}$ deserve further investigation. Their performance seems quite promising. These kernel functions are new and have not yet been optimized for

LP Problem	Number of iterations				
	ψ_1	$\psi_{2(q=\frac{3}{2})}$	$\psi_{7(q=\frac{3}{2})}$	$\psi_{1,1}$	$\psi_{1,\frac{3}{2}}$
25FV47	**71**	75	75	**71**	72
80BAU3B	**100**	104	104	103	104
ADLITTLE	**23**	**23**	**23**	24	24
AFIRO	**16**	**16**	**16**	**16**	17
AGG	43	44	44	**42**	43
AGG2	**36**	37	37	39	38
AGG3	**39**	45	41	43	41
BANDM	39	39	40	**38**	40
BEACONFD	**23**	**23**	24	25	25
BLEND	**19**	20	**19**	**19**	20
BNL1	69	70	69	**68**	**68**
BNL2	76	76	79	76	**75**
BOEING1	**42**	44	45	44	44
BOEING2	**35**	**35**	36	36	36
BORE3D	39	39	39	**36**	38
BRANDY	40	41	42	39	**38**
CAPRI	**42**	44	44	**42**	**42**
CYCLE	86	93	90	**77**	86
CZPROB	78	84	86	77	**76**
D2Q06C	**107**	112	112	110	109
D6CUBE	**38**	39	40	39	40
DEGEN2	**24**	26	25	**24**	26
DEGEN3	**30**	31	**30**	32	31
DFL001	**81**	85	85	83	83
E226	**41**	43	44	42	43
ETAMACRO	66	66	67	**64**	**64**
FFFFF800	**64**	66	65	**64**	**64**
FINNIS	60	60	61	**56**	**56**
FIT1D	**32**	33	**32**	33	33
FIT1P	**33**	**33**	35	34	34
FIT2D	42	43	42	**41**	42

Table 6: Numerical Results (I).

practical use. To get a method that is useful from a practical point of view, one has to embed the generic algorithm of Figure 1 in a predictor-corrector scheme as proposed by Mehrotra [9]. A nice example of this is a recent paper of Zhu et al. [15]. They propose an algorithm, in which the search direction determined by the kernel function ψ_7 plays a role. The algorithm starts with a predictor step based on the so-called primal-dual affine scaling direction. If the maximum step size in this direction is sufficiently large, then the algorithm performs after the predictor step a Mehrotra corrector step, based on the classical primal-dual Newton direction, followed by a backtracking line search technique to keep the

LP Problem	Number of iterations				
	ψ_1	$\psi_{2(q=\frac{3}{2})}$	$\psi_{7(q=\frac{3}{2})}$	$\psi_{1,1}$	$\psi_{1,\frac{3}{2}}$
FIT2P	**40**	42	43	**40**	**40**
FORPLAN	**40**	43	43	48	46
GANGES	43	43	**42**	44	44
GFRD-PNC	**32**	35	38	34	36
GREENBEA	**119**	125	130	123	125
GREENBEB	**121**	124	126	123	**121**
GROW15	**37**	39	39	**37**	38
GROW22	41	39	41	40	**38**
GROW7	**35**	36	37	**35**	**35**
ISRAEL	**36**	37	39	37	37
KB2	**30**	**30**	**30**	**30**	**30**
LOTFI	**29**	30	32	31	32
MAROS	74	76	69	**62**	64
MAROS-R7	**37**	**37**	38	**37**	39
MODSZK1	**49**	50	51	51	51
NESM	75	**74**	75	75	75
PEROLD	73	72	75	73	**71**
PILOT	102	102	104	100	**99**
PILOT.JA	**74**	76	76	76	76
PILOT.WE	137	133	132	**125**	131
PILOT4	71	76	76	**70**	**70**
PILOT87	145	149	150	147	**144**
PILOTNOV	56	58	59	**55**	56
RECIPE	**19**	21	21	21	21
SC105	**18**	19	19	**18**	19
SC205	**22**	**22**	23	**22**	**22**
SC50A	18	**17**	18	**17**	18
SC50B	17	17	17	**16**	17
SCAGR25	**32**	33	36	33	33
SCAGR7	**25**	26	26	26	26
SCFXM1	**42**	44	44	43	43
SCFXM2	**52**	**52**	54	**52**	53

Table 7: Numerical results (II).

iterates in a certain neighborhood of the central path. If the maximum feasible step size in the predictor step is not large enough, then the algorithm uses the search direction determined by the kernel function ψ_7 to bring the iterate closer to the central path. From their paper it is clear that the iteration numbers presented in the last tow tables can be reduced by about 50%.

LP Problem	Number of iterations				
	ψ_1	$\psi_{2(q=\frac{3}{2})}$	$\psi_{7(q=\frac{3}{2})}$	$\psi_{1,1}$	$\psi_{1,\frac{3}{2}}$
SCFXM3	57	**51**	54	57	56
SCORPION	**33**	34	36	35	35
SCRS8	51	51	53	**50**	52
SCSD1	**32**	41	46	39	33
SCSD6	50	**45**	54	67	61
SCSD8	45	**36**	50	39	41
SCTAP1	**36**	42	42	**36**	**36**
SCTAP2	**26**	28	27	**26**	27
SCTAP3	**28**	31	29	**28**	29
SEBA	54	56	57	**53**	54
SHARE1B	48	50	50	**47**	48
SHARE2B	**22**	**22**	23	24	23
SHELL	**46**	49	49	50	52
SHIP04L	**30**	**30**	**30**	**30**	31
SHIP04S	**28**	29	31	30	30
SHIP08L	36	34	34	**32**	33
SHIP08S	**27**	30	30	28	28
SHIP12L	**48**	53	55	50	**48**
SHIP12S	**41**	44	44	42	43
SIERRA	**40**	44	45	42	44
STAIR	**33**	**33**	35	34	34
STANDATA	**29**	31	30	30	30
STANDGUB	**29**	31	30	30	30
STANDMPS	**35**	39	38	38	38
STOCFOR1	27	27	27	**25**	**25**
STOCFOR2	65	**64**	66	68	68
STOCFOR3	**120**	121	121	122	123
TRUSS	**61**	62	65	63	64
TUFF	**40**	41	42	41	41
VTP-BASE	**28**	29	**28**	29	**28**
WOOD1P	37	35	40	**33**	34
WOODW	**60**	74	78	62	63

Table 8: Numerical results (III).

5. Conclusions and remarks

In this paper we investigated the influence of the choice of the kernel function on the computational behavior of the generic primal dual path-following algorithm for LO. Numerical results show that by using a kernel functions $\psi_{p,\sigma}$, with finite barrier term, the best iteration complexity was achieved in most of the test problems, especially for the kernel functions $\psi_{1,1}$ and $\psi_{1,1.5}$. Their practical performance seems quite promising for LO. This conclusion is supported in a different implementation [4].

References

[1] E.D. Andersen, J. Gondzio, Cs. Mészáros and X. Xu, Implementation of interior point methods for large scale linear programming. In T. Terlaky, Ed. *Interior Point Methods of Mathematical Programming*, pages 189–252. Kluwer Academic Publishers, The Netherlands, 1996.

[2] Y.Q. Bai, M. El Ghami, and C. Roos, A comparative study of kernel functions for primal-dual interior-point algorithms in linear optimization. *SIAM Journal on Optimization,***15**(1), 2005, 101-128.

[3] Y. Q. Bai, G. Lesaja, C. Roos, G.Q. Wang and M. El Ghami, A Class of Large- and Small-update Primal-dual Interior-point Algorithms for Linear Optimization. *Journal Of Optimization Theory and Applications (JOTA),* **138**, *2008, 341–359.*

[4] M. El Ghami, I. D. Ivanov, J.B.M. Melissen, C. Roos and T. Steihaug, A Polynomial-time Algorithm for Linear Optimization Based on a New Class of Kernel Functions. Journal of Computational and Applied Mathematics, **224**(2), 2009, 500–513.

[5] M. El Ghami, I. D. Ivanov, C. Roos and T. Steihaug, A Polynomial-time Algorithm for LO Based on Generalized Logarithmic Barrier Functions. *International Journal of Applied Mathematics (IJAM),* **21**(1) 2008, 99–115,.

[6] D. den Hertog, *Interior Point Approach to Linear, Quadratic and Convex Programming, Mathematics and its Applications. Kluwer Academic Publishers, Dordrecht*, **277**, 1994.

[7] http://www-fp.mcs.anl.gov/otc/Guide/TestProblems/index.html

[8] http://www.mathworks.com

[9] S. Mehrotra, On an implementation of primal-dual interior point method. SIAM *J. Optimization*, **2**(4),, 1992, 575-601.

[10] G.L. Nemhauser, A.H.G. Rinnooy Kan and M.J.Todd, *Optimization,* Holland, 1989.

[11] J. Peng, C. Roos and T. Terlaky, New complexity analysis of the primal-dual Newton method for linear optimization, *Ann. Oper Res.*, **99**, 2000, 23–39.

[12] J. Peng, C. Roos and T. Terlaky, A new and efficient large-update interior-point method for linear optimization. *Journal of Computational Technologies*, **6**(4), 2001, 61–80.

[13] J. Peng, C. Roos, and T. Terlaky, *Self-Regularity: A New Paradigm for Primal-Dual Interior-Point Algorithms*, Princeton University Press, 2002.

[14] C. Roos, T. Terlaky and J. Ph. Vial, *Theory and Algorithms for Linear Optimization. An Interior-Point Approach*, Springer Science, 2005.

[15]X. Zho, J. Peng, T. Terlaky and G. Zhang, On an implementation of self-regular proximity based feasibles IPMs. www.Optimization-online.org 2004.

Department of Informatics,
University of Bergen
P.O.Box 7803
N-5020 Bergen, NORWAY
E-mail: [1]*melghami@ii.uib.no*
[2]*Trond.Steihaug@ii.uib.no*