Numerical Experience with a
Polyhedral Norm CDT
Trust-Region Algorithm

M. El-Alem
R.A. Tapia

June, 1993

TR93-26

NUMERICAL EXPERIENCE WITH A POLYHEDRAL NORM CDT TRUST-REGION ALGORITHM¹

M. M. El-Alem ² and R. A. Tapia ³

Abstract. In this paper, we study the modification of the Celis-Dennis-Tapia trust-region subproblem which is obtained by replacing the ℓ_2 norm with a polyhedral norm. The polyhedral norm Celis-Dennis-Tapia (CDT) subproblem can be solved using a standard quadratic programming code.

We include computational results which compare the performance of the polyhedral norm CDT trust region algorithm with the performance of other existing codes. The numerical results validate the effectiveness of our approach. These results show that there is not much loss of robustness or speed and suggest that the polyhedral norm CDT algorithm may be a viable alternative. The topic merits further investigation.

Key words. Trust region methods, constrained optimization, Celis-Dennis-Tapia algorithm, successive quadratic programming algorithm, nonlinear programming, computational comparisons.

¹The first author was supported in part by the REDI foundation and the State of Texas Award, Contract Number 1059 as a visiting member of the Center for Research on Parallel Computation, Rice University, Houston, Texas 77251-1892, June 1993 - September 1993. He thanks Rice University for the congenial scientific atmosphere that it provided. The second author was supported in part by the National Science Foundation, Cooperative Agreement Number CCR-8809615, Air Force Office of Scientific Research 89-0363, and Department of Energy DEFG05-86ER25017.

²Assistant Professor, Department of Mathematics, Faculty of Science, Alexandria University, Alexandria, Egypt.

³Professor, Department of Computational and Applied Mathematics and Center for Research on Parallel Computation, Rice University, Houston, Texas 77251-1892.

1 Introduction

In this paper, we propose and study a variant of the Celis-Dennis-Tapia (CDT) trust-region algorithm for equality constrained optimization. Detailed discussion about the CDT algorithm can be found in Refs. 1 - 4. Since the CDT subproblem consists of a quadratic objective function with two quadratic (ℓ_2 norm) inequality constraints, obtaining its solution is not necessarily a straightforward matter (see Refs. 5 - 7). We consider modifying the CDT subproblem by replacing the ℓ_2 norm with a polyhedral norm; specifically a convex combination of the ℓ_1 and ℓ_∞ norms. This polyhedral norm CDT subproblem can be formulated as a quadratic program and solved using off-the-shelf codes. A new subproblem is proposed to be solved when regularity is an issue with the CDT subproblem.

We include computational results which compare the performance of the polyhedral norm CDT algorithm with the performance of the three existing codes VF02AD by Powell, NPSOL by Gill, Murray, Saunders, and Wright, and DNCONG by Schittkowski. The performance of the algorithm is also compared with two experimental codes, NLPTR by Williamson and NDIM by Celis. The results favor the polyhedral norm CDT algorithm and indicate that it deserves further investigation.

The paper is organized as follows. In the remainder of Section 1, we present the successive quadratic programming method for equality constrained optimization. In Section 2, we describe the trust-region subproblems that will be considered in computing the trial steps. In Section 3 and 4, we discuss various approaches for computing the trial steps. The merit function is introduced in Section 5. A formula for updating the penalty parameter is also presented. In Section 6, a procedure for updating the radius of the trust region is presented together with a discussion concerning the criteria for accepting or rejecting the trial steps. In Section 7, we outline the polyhedral norm CDT trust-region algorithm. Section 8 is devoted to the computer implementation of the algorithm and the numerical comparisons. Section 9 contains concluding remarks.

Consider the following equality constrained optimization problem

$$\begin{aligned}
 \text{(EQ)} \quad & \min \quad f(x) \\
 & \text{s. t.} \quad h(x) = 0
 \end{aligned}$$

where $h(x)$ is the vector whose components are $h_i(x)$ $i = 1, \dots, m$. We assume that f and h_i are smooth nonlinear functions defined from $\mathbb{R}^n$ into $\mathbb{R}$. The Lagrangian function $l : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}$ associated with problem (EQ) is

$$l(x, \lambda) = f(x) + \lambda^T h(x),$$

where $\lambda = (\lambda_1, \dots, \lambda_m)^T$ is called the Lagrange multiplier vector.

The well-known successive quadratic programming method (SQP) approximates a minimizer of problem (EQ) by solving a sequence of quadratic programming subproblems.

By the successive quadratic programming quasi-Newton method (SQP), we mean the iterative procedure:

SCHEME 1.1: Successive Quadratic Programming Method

Given $x_0 \in \mathbb{R}^n$, $\lambda_0 \in \mathbb{R}^m$, and $B_0 \in \mathbb{R}^{n \times n}$.

For $k = 1, 2, \dots$ until convergence, do the following steps.

Step 1. Find a solution s_k^{QP} and associated Lagrange multiplier $\Delta\lambda_k^{QP}$ of the quadratic program

$$\begin{aligned}
 \text{(QP)} \quad & \min \quad \nabla_x l_k^T s + \frac{1}{2} s^T B_k s \\
 & \text{s. t.} \quad h_k + \nabla h_k^T s = 0.
 \end{aligned}$$

Step 2. Set $x_{k+1} = x_k + s_k^{QP}$.

Set $\lambda_{k+1} = \lambda_k + \Delta\lambda_k^{QP}$.

Update B_k .

Step 3. Return to Step 1.

In the QP subproblem the matrix B_k is the Hessian of the Lagrangian at (x_k, λ_k) or an approximation to it.

If the solution to problem (QP) exists, then it is also the solution to the following $(n + m) \times (n + m)$ linear system:

$$\begin{bmatrix} B_k & \nabla h_k \\ \nabla h_k^T & 0 \end{bmatrix} \begin{bmatrix} s \\ \Delta\lambda \end{bmatrix} = - \begin{bmatrix} \nabla_x l_k \\ h_k \end{bmatrix}. \quad (1)$$

Hence the SQP method can be viewed as a quasi-Newton method on the extended system

$$\begin{bmatrix} \nabla_x l(x, \lambda) \\ h(x) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

(see Refs. 8 and 9). Therefore, it shares the advantages and the disadvantages of Newton's method. From the good side, it possesses fast local convergence and leads to an easy subproblem at each iteration. However, from the bad side, the step may not exist and the algorithm may not converge if the starting point is far from the solution.

The local convergence analysis for the SQP algorithm is well understood. The area of global convergence is currently receiving much attention.

A fundamental difficulty in the definition of the SQP step is that the second-order sufficiency condition need not hold at each iteration. By this we mean that at any iteration k , the matrix B_k need not be positive definite on the null space of ∇h_k^T ; hence the QP subproblem may not have a solution or a unique solution. This difficulty will not arise near a solution of problem (EQ) if the standard assumptions for the Newton's method hold at the solution. For this reason, the SQP method usually performs very well locally. An effective way of dealing with the lack of positive definiteness on this proper subspace is to use a trust-region globalization strategy. We describe this approach in detail in Section 2.

2 Trust-Region Globalization For Problem (EQ)

Trust-region algorithms have proven to be very effective and robust for solving unconstrained optimization problems. See, for example, Ref. 10. The most natural way to extend the trust-region idea to constrained optimization is to add a trust-region constraint which restricts

the size of the step in problem (QP). This idea leads to the so-called trust-region quadratic programming subproblem (TRQP).

2.1 TRQP Subproblem

Adding a trust-region constraint to problem (QP) gives

$$\begin{aligned} \text{(TRQP)} \quad & \min \quad \nabla_x l_k^T s + \frac{1}{2} s^T B_k s \\ & \text{s. t.} \quad h_k + \nabla h_k^T s = 0 \text{ and } \|s\|_2 \leq \Delta_k. \end{aligned}$$

Subproblem TRQP may have inconsistent constraints, i. e., the hyperplane $h_k + \nabla h_k^T s = 0$ may not intersect the sphere $\|s\|_2 \leq \Delta_k$. Even if they do intersect, there is no guarantee that the trial step s_k will be accepted. So one may have to decrease the radius of the trust region, and again inconsistent constraints may arise. To overcome this difficulty we solve a different problem, the so-called CDT subproblem.

2.2 CDT Subproblem

Celis, Dennis, and Tapia (Ref. 1) suggested the following approach. At each iteration the step is computed by solving the subproblem:

$$\begin{aligned} \text{(CDT)} \quad & \min \quad \nabla_x l_k^T s + \frac{1}{2} s^T B_k s \\ & \text{s. t.} \quad \|h_k + \nabla h_k^T s\|_2 \leq \theta_k \text{ and } \|s\|_2 \leq \Delta_k. \end{aligned}$$

where θ_k is a positive constant. This idea leads to a feasible subproblem provided that θ_k is chosen properly.

2.3 Choice of θ_k

Celis, Dennis and Tapia (Ref. 1) chose θ_k to be $\|h_k + \nabla h_k^T s_k^{\text{cp}}\|_2$, where $s_k^{\text{cp}} = -\alpha_k \nabla h_k h_k$ is the step to the Cauchy point, i.e., the minimizer in the trust region $\{s : \|s\|_2 \leq \Delta_k\}$ of $\Phi(s) = \|h_k + \nabla h_k^T s\|_2^2$ along the line emanating from the origin in the direction of the negative gradient of $\Phi(s)$ at $s = 0$. That is, the Celis-Dennis-Tapia step is chosen from the

set of steps from x_k that produce points which are inside the trust region and give at least as much decrease in the ℓ_2 norm of the residual of the linearized constraints as does the Cauchy step.

A disadvantage of using this choice of θ_k lies in the possible loss of regularity in the subproblem when the Cauchy point is the most linearly feasible point in the trust region. *i.e.* $\theta_k = 0$.

A more general choice of θ_k is suggested by Celis, Dennis, Martinez, Tapia, and Williamson (Ref. 11). (See also Williamson Ref. 12). They choose

$$\theta_k = (1 - \tau_1)\|h_k\|_2 + \tau_1\|h_k + \nabla h_k^T s_k^{\text{cp}}\|_2,$$

for some $0 < \tau_1 \leq 1$, where s_k^{cp} is the step to the Cauchy point and was defined above.

This latter choice of θ_k enforces a fraction of Cauchy decrease on the ℓ_2 norm of the linearized constraints in the CDT subproblem. The choice given by Celis, Dennis and Tapia (Ref. 1) gives at least as much decrease in the ℓ_2 norm of the linearized constraints as does the Cauchy step s_k^{cp} . Another choice of θ_k is given by Powell and Yuan in Ref. 13.

A disadvantage of the Celis-Dennis-Tapia approach lies in the numerical difficulty that arises when θ_k is very small. To overcome this difficulty, we consider an alternative to be described in the following section.

2.4 Generalized TRQP Subproblem

In the SQP method, unless we assume that ∇h_k has full column rank, there is no guarantee that the linearized constraints are consistent, *i.e.* the feasible region may be empty. To overcome this difficulty, in a fashion similar to that used by Byrd and Omojokun (Ref. 14), we replace the quadratic programming (QP) with the more general quadratic programming subproblem

$$\begin{aligned} \text{(GQP)} \quad & \min \quad \nabla_x l_k^T s + \frac{1}{2} s^T B_k s \\ & \text{s. t.} \quad h_k + \nabla h_k^T s = v_k, \end{aligned}$$

where the m -dimensional vector v_k is given by $v_k = h_k + \nabla h_k^T \bar{s}_k$, and

$$\bar{s}_k = \operatorname{argmin}[\|h_k + \nabla h_k^T s\|_2 : s \in \mathbb{R}^n].$$

The advantage of using this approach lies in the fact that the (GQP) subproblem always has consistent constraints and hence, the feasible region is nonempty (*e.g.* the point $\bar{s}_k$ is a feasible point). The only difficulty in the definition of the GQP step is that the second-order sufficiency conditions need not hold. By this we mean that the matrix B_k need not be positive definite on the null space of ∇h_k^T . Our way of dealing with this difficulty is to add a trust-region constraint. This strategy leads to what we call the generalized trust-region quadratic program or problem (GTRQP)

$$\begin{aligned} \text{(GTRQP)} \quad & \min \quad \nabla_x l_k^T s + \frac{1}{2} s^T B_k s \\ & \text{s. t.} \quad h_k + \nabla h_k^T s = u_k \text{ and } \|s\|_2 \leq \Delta_k, \end{aligned}$$

where the m -dimensional vector u_k is given by $u_k = h_k + \nabla h_k^T \hat{s}_k$, and

$$\hat{s}_k = \operatorname{argmin}[\|h_k + \nabla h_k^T s\|_2 : \|s\|_2 \leq \tau_2 \Delta_k, s \in \mathbb{R}^n],$$

for some $0 < \tau_2 \leq 1$.

3 Trial Step Computation

In this section and the next we discuss various approaches for computing the trial steps. Let us start by considering a skeleton version of the basic CDT algorithm

SCHEME 3.1: (ℓ_2 Norm CDT)

Given $x_0 \in \mathbb{R}^n$, and $B_0 \in \mathbb{R}^{n \times n}$.

For $k = 1, 2, \dots$ until convergence, do the following steps.

Step 1. Find a solution s to the CDT subproblem

$$\begin{aligned} \text{(CDT)} \quad & \min \quad \nabla_x l_k^T s + \frac{1}{2} s^T B_k s \\ & \text{s. t.} \quad \|h_k + \nabla h_k^T s\|_2 \leq \theta_k \text{ and } \|s\|_2 \leq \Delta_k, \end{aligned}$$

where B_k is the Hessian of the Lagrangian or an approximation to it.

Step 2. Test whether to accept or reject the step s .

Update Δ_k and θ_k .

Update B_k .

Step 3. Return to Step 1.

As mentioned earlier the major disadvantage of this approach lies in the difficulty of solving the subproblem for the trial step at each iteration. In the following section we present our modified algorithm that is designed to avoid this disadvantage.

4 More Accessible Trust-Region Subproblem

To overcome the computational difficulty of the CDT subproblem we replace the ℓ_2 norm in the trust-region constraint in Algorithm 3.1 with the ℓ_∞ norm and replace the ℓ_2 norm in the linear feasibility constraint in the CDT subproblem by a convex combination of the ℓ_1 and the ℓ_∞ norms. This approach leads to a subproblem which can equivalently be written as a quadratic objective function that is to be minimized subject to a set of linear inequality constraints and simple bounds. This latter subproblem can be readily solved by standard quadratic programming packages. The following is the modified CDT algorithm.

SCHEME 4.1: (Polyhedral Norm CDT)

Given $x_0 \in \mathbb{R}^n$ and $B_0 \in \mathbb{R}^{n \times n}$.

For $k = 1, 2, \dots$ until convergence, do the following steps.

Step 1. Find a solution s to the following subproblem

$$\begin{aligned} \text{(PNC DT)} \quad & \min \quad \nabla_x l(x_k, \lambda_k)^T s + \frac{1}{2} s^T B_k s \\ & \text{s. t.} \quad \|h(x_k) + \nabla h(x_k)^T s\|_{c(\alpha)} \leq \theta_k \text{ and } \|s\|_\infty \leq \Delta_k \end{aligned}$$

where $\|\cdot\|_{c(\alpha)} = (1 - \alpha)\|\cdot\|_1 + \alpha\|\cdot\|_\infty$.

Step 2. Test whether to accept or reject the step s .

Update Δ_k and θ_k .

Update B_k .

Step 3. Go to step 1.

Our motivation for considering a convex combination of the ℓ_1 norm and the ℓ_∞ norm instead of one or the other stems from the following fact. The ℓ_1 unit ball under approximates the ℓ_2 unit ball and the ℓ_∞ over approximates it. So, the $\ell_{c(\alpha)}$ unit ball with appropriate choice of α (see Section 8), gives a better piecewise linear approximation to the ℓ_2 unit ball. Moreover, we believe that in some uncomputational sense, the ℓ_2 unit ball is the optimal choice.

The above algorithm has two disadvantages which are characteristic of the CDT formulation and neither has to do with the choice of norm. Firstly, locally, the PNCDT step is not as good as the SQP step. Secondly, experimentation shows numerical difficulties arise when θ_k is very small. The following alternative algorithm is designed to avoid these two disadvantages.

SCHEME 4.2: (Combination Trial Step)

Solve (QP) to get s^{QP} (if it exists). (See Section 1.)

If s^{QP} exists and $\|s^{QP}\|_2 \leq \Delta_k$

then set $s_k = s^{QP}$.

Else, if θ_k is small

then solve for s the subproblem (VGTRQP) (see below).

Set $s_k = s$.

Else, solve for s the subproblem (PNCDT) (see above).

Set $s_k = s$.

To compute the trial step at most two subproblems have to be solved, the QP subproblem and either the PNCDT or the following variant of the GTRQP subproblem (see Section 2.4)

$$\begin{aligned} \text{(VGTRQP)} \quad & \min \quad \nabla_x l_k^T s + \frac{1}{2} s^T B_k s \\ & \text{s. t.} \quad h_k + \nabla h_k^T s = u_k \text{ and } \|s\|_\infty \leq \Delta_k \end{aligned}$$

where the m-dimensional vector u_k is given by $u_k = h_k + \nabla h_k^T \tilde{s}_k$, and

$$\tilde{s}_k = \operatorname{argmin}[\|h_k + \nabla h_k^T s\|_2 : \|s\|_\infty \leq \tau_2 \Delta_k, s \in \mathbb{R}^n],$$

for some $0 < \tau_2 \leq 1$.

The major advantage of this approach lies in the fact that the modified subproblems PNCDT and VGTRQP can be formulated as quadratic programs which can be solved by a standard quadratic programming routine.

5 Merit Function

Our choice of merit function is the augmented Lagrangian

$$\mathcal{L}(x, \lambda; r) = f(x) + \lambda^T h(x) + r \|h(x)\|_2^2,$$

where $\lambda \in \mathbb{R}^m$ and $r \in \mathbb{R}_+$ is the penalty parameter. For updating the multiplier, we will use the formula $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$, where

$$\Delta\lambda_k = -(\nabla h_k^T \nabla h_k)^{-1} \nabla h_k^T (\nabla l_k + B_k s_k) \quad (2)$$

and s_k is the computed trial step. It can be shown that in the case of SQP, (2) gives $\Delta\lambda_k^{QP}$ (see Section 1).

The actual reduction in the merit function in going from (x_k, λ_k) to $(x_k + s_k, \lambda_k + \Delta\lambda_k)$ is given by

$$A_k = \mathcal{L}(x_k, \lambda_k; r_k) - \mathcal{L}(x_k + s_k, \lambda_k + \Delta\lambda_k; r_k).$$

We can write

$$\begin{aligned} A_k &= l(x_k, \lambda_k) - l(x_k + s_k, \lambda_k) - \Delta\lambda_k^T h(x_k + s_k) \\ &\quad + r_k [\|h(x_k)\|_2^2 - \|h(x_k + s_k)\|_2^2]. \end{aligned}$$

The calculation of the step s_k is based on a quadratic approximation of the Lagrangian function and a linear approximation to the constraints. Now by using these approximations, we can compute the predicted reduction

$$P_k = \mathcal{L}(x_k, \lambda_k; r_k) - \Psi(x_k, s_k, \lambda_k, \Delta\lambda_k; r_k),$$

where $\Psi(x_k, s_k, \lambda_k, \Delta\lambda_k; r_k)$ is an approximation to $\mathcal{L}(x_k + s_k, \lambda_k + \Delta\lambda_k; r_k)$ and is defined by

$$\begin{aligned}\Psi(x_k, s_k, \lambda_k, \Delta\lambda_k; r_k) &= l(x_k, \lambda_k) + \nabla_x l(x_k, \lambda_k)^T s_k + \frac{1}{2} s_k^T B_k s_k \\ &+ \Delta\lambda_k^T [h(x_k) + \nabla h(x_k)^T s_k] + r_k \|h(x_k) + \nabla h(x_k)^T s_k\|_2^2.\end{aligned}$$

Hence,

$$\begin{aligned}P_k &= -\nabla_x l_k^T s_k - \frac{1}{2} s_k^T B_k s_k - \Delta\lambda_k^T (h_k + \nabla h_k^T s_k) \\ &+ r_k [\|h_k\|_2^2 - \|h_k + \nabla h_k^T s_k\|_2^2].\end{aligned}$$

The penalty parameter r_k is updated using the following scheme given by El-Alem (Ref. 3).

SCHEME 5.1: (Updating the Penalty Parameter)

Initialization: set $r_0 = 1$ and choose a small constant $\rho > 0$.

At the k^{th} iteration, after computing s_k , do the following steps

Step 1. Set $r_k = r_{k-1}$.

Step 2. If $P_k \leq \frac{r_k}{2} [\|h_k\|_2^2 - \|h_k + \nabla h_k^T s_k\|_2^2]$, then set

$$r_k = \frac{2\nabla_x l_k^T s_k + s_k^T B_k s_k + 2\Delta\lambda_k^T (h_k + \nabla h_k^T s_k)}{\|h_k\|_2^2 - \|h_k + \nabla h_k^T s_k\|_2^2} + \rho.$$

6 Testing The Step and Updating Δ

Let s_k be the step computed by the algorithm and let $\Delta\lambda_k$ be given by (2). We test whether the point $(x_k + s_k, \lambda_k + \Delta\lambda_k)$ is a better approximation to the solution (x_*, λ_*) than is the point (x_k, λ_k) . First, we compute the ratio $\Gamma_k = A_k/P_k$.

The step is rejected if $\Gamma_k < \eta_1$ where $0 < \eta_1 < 1$ is a small fixed constant. In this case, we decrease the radius of the trust region by setting $\Delta_k = \alpha_1 \|s_k\|_2$, where $\alpha_1 \in (0, 1)$ and go back to Scheme 4.2 to compute another trial step.

If $\Gamma_k \geq \eta_1$, then we accept the step and set $x_{k+1} = x_k + s_k$ and $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$.

When the step is accepted, the trust-region radius is updated by comparing the value of A_k with P_k . Namely, if $\eta_1 \leq \Gamma_k < \eta_2$, where $\eta_1 < \eta_2 < 1$, then the radius of the trust region is updated by the rule: $\Delta_{k+1} = \min[\Delta_{\min}, \alpha_2 \Delta_k]$, where $\Delta_{\min}$ is a fixed constant and $\alpha_2 > 1$. If $\eta_2 \leq \Gamma_k < \eta_3$ where $\eta_2 < \eta_3 < 1$, then the radius of the trust region is updated by the rule: $\Delta_{k+1} = \min[\Delta_{\max}, \alpha_2 \Delta_{\min}, \alpha_3 \Delta_k]$, where $\Delta_{\max} > \Delta_{\min}$ is a fixed constant and $\alpha_3 \geq \alpha_2$.

If $\eta_3 \leq \Gamma_k < \eta_4$ where $\eta_3 \leq \eta_4 < 1$, then the radius of the trust region is updated by the rule: $\Delta_{k+1} = \min\{\Delta_{\max}, \max[\alpha_2 \Delta_{\min}, \alpha_3 \Delta_k]\}$. Finally, if $\Gamma_k \geq \eta_4$, then we increase the trust-region radius by setting: $\Delta_{k+1} = \min[\Delta_{\max}, \max\{\alpha_3 \Delta_{\min}, \alpha_4 \Delta_k\}]$, where $\alpha_4 \geq \alpha_3$.

These notions can be stated in algorithmic form as follows

SCHEME 6.1: (Testing the Step and Updating Δ)

If $\Gamma_k < \eta_1$, then reject the step.

Set $\Delta_k = \alpha_1 \|s_k\|_2$.

Compute another trial step using Scheme 4.2.

Else, if $\eta_1 \leq \Gamma_k < \eta_2$ then set $x_{k+1} = x_k + s_k$,

$$\lambda_{k+1} = \lambda_k + \Delta \lambda_k,$$

$$\Delta_{k+1} = \min[\Delta_{\min}, \alpha_2 \Delta_k]$$

Else, if $\eta_2 \leq \Gamma_k < \eta_3$ then set $x_{k+1} = x_k + s_k$,

$$\lambda_{k+1} = \lambda_k + \Delta \lambda_k,$$

$$\Delta_{k+1} = \min[\Delta_{\max}, \alpha_2 \Delta_{\min}, \alpha_3 \Delta_k],$$

Else, if $\eta_3 \leq \Gamma_k < \eta_4$ then set $x_{k+1} = x_k + s_k$,

$$\lambda_{k+1} = \lambda_k + \Delta \lambda_k,$$

$$\Delta_{k+1} = \min[\Delta_{\max}, \max\{\alpha_2 \Delta_{\min}, \alpha_3 \Delta_k\}],$$

Else, set $x_{k+1} = x_k + s_k$,

$$\lambda_{k+1} = \lambda_k + \Delta \lambda_k,$$

$$\Delta_{k+1} = \min[\Delta_{\max}, \max\{\alpha_3 \Delta_{\min}, \alpha_4 \Delta_k\}].$$

In the case when $\Gamma_k \geq \eta_3$, and s^{QP} exists but $s_k \neq s^{QP}$, then we do only one internal doubling by increasing the radius of the trust region and if s^{QP} lies inside the new trust region, we take it as our trial step. Otherwise, we stay with the old acceptable step and the

old updated trust-region radius. This can be stated in algorithmic form as follows

SCHEME 6.2: (Internal Doubling)

If $\eta_3 \leq \Gamma_k < \eta_4$, s^{QP} exists, and $\|s^{QP}\|_2 > \Delta_k$

then set $\Delta_k = \min[\Delta_{\max}, \max\{\alpha_2\Delta_{\min}, \alpha_3\Delta_k\}]$,

Else, if $\Gamma_k \geq \eta_4$, s^{QP} exists, and $\|s^{QP}\|_2 > \Delta_k$

then set $\Delta_k = \min[\Delta_{\max}, \max\{\alpha_3\Delta_{\min}, \alpha_4\Delta_k\}]$.

If $\|s_k^{QP}\|_2 > \Delta_k$,

then, go back to the last acceptable step and the last corresponding trust-region radius and update it by Scheme 6.1.

Else, if $\Gamma_k < \eta_1$,

then go back to the last acceptable step and the last corresponding trust-region radius and update it by Scheme 6.1.

Else, accept the step and update Δ_k according to Scheme 6.1 above.

7 Polyhedral Norm CDT Trust-Region Algorithm

The following represents an outline of the complete algorithm.

Step 0 :

Set $x_0 \in \mathbb{R}^n$, $B_0 \in \mathbb{R}^{n \times n}$, $\lambda_0 \in \mathbb{R}^m$, $\rho > 0$,

$0 < \alpha_1 < 1 < \alpha_2 \leq \alpha_3 \leq \alpha_4$, $0 < \eta_1 < \eta_2 < \eta_3 < \eta_4 < 1$,

$0 < \tau_1 \leq 1$, $0 < \tau_2 \leq 1$, $\varepsilon > 0$, $\delta_0 > 0$, and $k = 0$.

Step 1 :

if $\|\nabla l_k\|_2 + \|h_k\|_2 < \varepsilon_{tol}$, stop.

Step 2 :

Compute s_k according to Scheme 4.2 above and $\Delta\lambda_k$ according to (2).

Step 3 :

Update the penalty parameter according to (3)

Step 4 :

Test the step and update Δ_k according to Scheme 6.1 above.

Step 5 :

Update B_k .

Step 6 :

Set $k := k + 1$ and go to step 1.

8 Computer Implementation

We designed our numerical investigation with the following questions in mind:

- 1) How does the polyhedral norm CDT algorithm compare to other existing algorithms and do we lose anything when we replace the ℓ_2 norm with a polyhedral norm?
- 2) How does the polyhedral norm, as a way to approximate the CDT subproblem, compare to other ways of approximating the CDT subproblem?
- 3) Is it important that we always take the SQP step whenever it is acceptable?

In the following two subsections we will try to answer these questions. However, more numerical investigation is needed to fully understand these questions and related ones. (See Section 9)

8.1 Performance of The Algorithm

Our experiments were performed on a subset of the Hock and Schittkowski's test problems (Ref. 15). The experiments were done on a Sun 4/490 workstation with 64 Megabytes of memory.

The algorithm described in Section 7 has been programmed and tested on a SUN Microsystems in double precision Matlab with a machine epsilon about 10^{-16} .

The tolerance for the stopping criterion is chosen as follows: minimum step size is 10^{-10} and the tolerance $\varepsilon_{tol} = 10^{-6}$. The maximum number of iterations allowed is 75, and the maximum number of function evaluations is 100.

Successful termination means that the termination condition of the algorithm is met with $\varepsilon_{tol} = 10^{-6}$. On the other hand, if the algorithm generates a step of length less than 10^{-10} , or the number of iterations is greater than 75, or the number of function evaluations is greater than 100, then it is considered an unsuccessful termination.

The initial trust-region radius $\Delta_0 = \|s_0^{QP}\|_2$. However, if s_0^{QP} does not exist then we take $\Delta_0 = \|s_0^{cp}\|_2$. The maximum trust-region radius allowed is $\Delta_{\max} = 50 \Delta_0$. The value of $\Delta_{\min}$ is $0.01 \Delta_0$.

The values of the parameters used for updating the trust-region radius in Scheme 6.1 and Scheme 6.2 are: $\eta_1 = 10^{-6}$, $\eta_2 = 0.1$, $\eta_3 = 0.4$, $\eta_4 = 0.7$, $\alpha_1 = 0.3$, $\alpha_2 = 2$, $\alpha_3 = 4$, and $\alpha_4 = 6$.

The parameter ρ in formula (3) takes the value 0.1. In computing θ_k , we use the formula

$$\theta_k = (1 - \tau_1)\|h_k\|_2 + \tau_1\|h_k + \nabla h_k^T s_k^{cp}\|_2.$$

The value of τ_1 is taken to be 0.7 and the value of τ_2 in the VTRGQP subproblem is 0.8.

We define the polyhedral norm to be $\|\cdot\|_{c(\alpha)} = (1 - \alpha)\|\cdot\|_1 + \alpha\|\cdot\|_\infty$. As a value for α we use $\alpha = \sqrt{2} - 1$. This choice of α results from an attempt to make the polyhedral norm unit ball best approximate the ℓ_2 norm unit ball.

For computing B_k we use a finite difference approximation to the Hessian of the Lagrangian.

In computing the trial step, we first solve the linear system (1) to obtain the s^{QP} step and its associated Lagrange multiplier step $\Delta\lambda^{QP}$. If the step exists and lies inside the trust region, then we take it as our trial step. Our algorithm will reject this step if it produces a negative P_k . In this case, we assume that the s^{QP} step does not exist and proceed with algorithm 4.2.

We now report the numerical results for our trust-region algorithm described in Section 7. The problems that we tested can be found in Hock and Schittkowski (Ref. 15) and will be referenced by the same number given in that book. The results are reported in Tables 1 - 4.

In Tables 1 and 2, our results are compared against the following three software packages: NPSOL by Gill, Murray, Saunders, and Wright (Refs. 16 and 17), DNCONG by

Table 1: Average number of iterations

Problem	NPSOL	DNCONG	NLPTR	PNCDDT
6	10.4	12	11	6.4
7	23	19	16.8	13.5
8	6	5.5	5.5	6
9	5	6.5	5.5	10.5
26	45.7	21.1	19.9	19.1
27	23.7	23	15.2	12.2
39	23	28.8	23.6	17
40	23.3	25.3	8.8	16.5
42	13.6	17.6	8.4	10
60	18.7	20.4	11.7	11.6
77	16.5	19.8	8	9.5
78	19	17.5	7.5	7
79	15.4	15.2	7.4	15

Schittkowski (Ref. 18), which is available in the IMSL MATH/LIBRARY and NLPTR, an experimental code written by Williamson (Ref. 12), which approximates the solution of the CDT subproblem by considering a particular two dimensional subspace. In Table 1, We list the average number of iterations for our algorithm and the above three software packages. In Table 2, we list the average number of function evaluations.

In Table 3, we report the number of unsuccessful terminations for our algorithm and the three software packages mentioned above.

In Table 4, our results are compared against VF02AD by Powell (Refs. 19 - 21), which is available in the Harwell subroutine library, and NDMCDT, an experimental code written by Celis (Ref. 2), which approximates the solution of the l_2 norm CDT subproblem. We present the average number of iterations for our algorithm and the ones in the two software packages. We regret that we were unable to include a table for the average number of function evaluations. The number of iterations was the only information available to us for several of these applications.

The results show that for this set of test problems our algorithm is competitive with other codes listed in the tables with respect to the number of iterations and the number

Table 2: Average no. of function evaluations.

Problem	NPSOL	DNCONG	NLPTR	PNCDDT
6	17.4	14.4	17	13.8
7	39	23.8	20	25.5
8	9	5.5	7	7.5
9	9	7	7	13
26	56	23	21.6	21.6
27	36.8	28.8	21.8	19.5
39	33.6	38.8	31.2	30.4
40	47	38.8	12.5	20.7
42	21.6	20.2	10	13.6
60	28.9	24.5	13.3	13
77	27	26.3	11.3	14.5
78	27.8	23.5	9.8	8.5
79	23.8	18.8	8.8	23.4

Table 3: No. of unsuccessful terminations

NPSOL	DNCONG	NLPTR	PNCDDT
7	9	6	9

of function evaluations. We must add here that the fewest number of iterations does not necessarily imply that the corresponding algorithm is the less expensive. On the other hand, the comparison results in the tables show that for this set of test problems we do not lose robustness when replacing the ℓ_2 norm with the $\ell_{c(\alpha)}$ norm. However, we believe that there is some incompatibility between the PNCDDT and VGTRQP subproblems and the augmented Lagrangian merit function. At some points, the trial steps do not give enough decrease in the merit function. As a result, at these points, the algorithm seems to take an excessive number of function evaluations to move to a new point. This situation is depicted in Problems 27 and 39, where our algorithm at some different starting points takes the maximum number of iterations allowed by the algorithm. (See Ref. 22.)

Table 4: Average number of iterations

Problem	VF02AD	NDIMCDT	PNC DT
60	15	19.5	8.8
77	23.8	16.6	11.4
78	11.2	6	6
79	13.1	7	9.6

8.2 CDT Versus SQP

To emphasize the fact that the PNC DT step is a good step far away from the solution and the SQP step is a good step close to the solution, we ran Algorithm 4.1 (which always takes the PNC DT step as its trial step) then we ran an algorithm that uses Scheme 1.1 to get a trial step, *i. e.* the SQP subproblem is the only subproblem to be solved at each iteration. We compared the results of those two algorithms⁴ with the combined polyhedral norm CDT algorithm that takes either the SQP or the PNC DT step. A typical result is shown in the following figure⁵.

⁴The results were compared for problems and starting points such that the SQP algorithm is well defined and convergent.

⁵Problem number 39 with starting point (2,2,2). The norm of residual in the figure means $\|\nabla_x l_k\|_2 + \|h_k\|_2$.

9 Concluding Remarks

We have presented a polyhedral norm variant of the CDT trust region algorithm for the equality constrained optimization problem. We included computational results which compare the performance of our variant algorithm with the performance of other existing codes.

The numerical results indicate the effectiveness of our approach in obtaining a solution to problem (EQ) using our algorithm from any starting point.

Our numerical results show that there is not much loss of robustness or speed in the performance of the algorithm when replacing the ℓ_2 norm with the polyhedral norm. This study indicates that our approach is a viable alternative and merits further investigation. In particular, further work is needed in the area of the compatibility of the ℓ_2 norm augmented Lagrangian and the polyhedral norm subproblems.

10 References

1. Celis, M. R., Dennis, J. E., and Tapia R. A., A Trust Region Strategy for Nonlinear Equality Constrained Optimization, Numerical Optimization, Edited by P. Boggs, R. Byrd and R. Schnabel, SIAM Publications, Philadelphia, Pennsylvania, pp. 71-82, 1985.
2. Celis, M. R., A Trust Region Strategy for Nonlinear Equality Constrained Optimization, Ph D Thesis, Rice University, Houston, Texas, 1985.
3. El-Alem, M. M., A Global Convergence Theory for a Class of Trust Region Algorithms for Constrained Optimization, Ph D Thesis, Rice University, Houston, Texas, 1988.
4. El-Alem, M. M., A Global Convergence Theory for the Celis-Dennis-Tapia Trust Region Algorithm for Constrained Optimization, SIAM Journal on Numerical Analysis, Vol. 24, pp. 266-290, 1991.

5. Yuan, Y., A Dual Algorithm for Minimizing a Quadratic Function With Two Quadratic Constraints, Technical Report DAMTP-NA3, University of Cambridge, England, 1988.
6. Yuan, Y., On a Subproblem of Trust Region Algorithms For Constrained Optimization, Mathematical Programming, Vol. 47, pp. 53-63, 1990.
7. Zhang, Y., Computing a Celis-Dennis-Tapia Trust Region Step for Equality Constrained Optimization, Mathematical Programming, Vol. 55, pp. 109-124, 1992.
8. Tapia, R. A., Diagonalized Multiplier Methods and Quasi-Newton Methods For Constrained Optimization, Journal of Optimization Theory and Applications, Vol. 22, pp. 135-194, 1977.
9. Tapia, R. A., Quasi-Newton Methods for Equality Constrained Optimization: Equivalence of Existing Methods and a New Implementation, Nonlinear Programming 3, Edited by O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, Academic Press, New York, NY, pp. 125-164, 1978.
10. Dennis, J. E. and Schnabel, R. B., Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.
11. Celis, M. R., Dennis, J. E., Martinez, J. M., Tapia, R. A., and Williamson, K., An Algorithm Based on a Convenient Trust-Region Subproblem for Nonlinear Programming. (in preparation)
12. Williamson, K., A Robust Trust Region Algorithm For Nonlinear Programming, Ph D Thesis, Rice University, Houston, Texas, 1990.
13. Powell, M. J. D. and Yuan, Y, A Trust Region Algorithm For Equality Constrained Optimization, Mathematical Programming, Vol. 49, pp. 189-211, 1991.
14. Byrd, R. and Omojokun, E., Robust Trust Region Methods for Nonlinearly Constrained Optimization, Presented at the 1987 SIAM Conference on Optimization, Houston, Texas.

15. Hock, W., and Schittkowski, K., Test Examples for Nonlinear Programming Codes, Lecture Notes in Economics and Mathematical Systems, Springer Verlag, Berlin, Germany, Vol. 187, 1981.
16. Gill, P. E., Murray, W., Saunders, M. A., and Wright, M., User's Guide for QPSOL : A Fortran Package for Quadratic Programming, Report SOL 84-2, Stanford University, Stanford, California, 1984.
17. Gill, P. E., Murray, W., Saunders, M. A., and Wright, M., User's Guide for NPSOL : A Fortran Package for Nonlinear Programming, Report SOL 83-12, Stanford University, Stanford, California, 1983.
18. Schittkowski, K., NLPQL: A Fortran Subroutine Solving Constrained Nonlinear Programming Problems, Annals of Operations Research, Vol. 5, pp. 485-500, 1986.
19. Powell, M. J. D., A Fast Algorithm for Nonlinearly Constrained Optimization Calculations, Lecture Notes in Mathematics, Springer Verlag, Berlin, Germany, Vol. 630, pp. 144-157, 1978.
20. Powell, M. J. D., Extensions to Subroutine VF02AD, Lecture Notes in Control and Information Science, Springer Verlag, New York, Vol. 38, 1982.
21. Powell, M. J. D., The Performance of Two Subroutines for Constrained Optimization on Some Difficult Test Problem, Numerical Optimization, Edited by R. Boggs, R. Byrd and B. Schnabel, SIAM Publications, SIAM, Philadelphia, Pennsylvania, pp. 71-82, 1984.
22. El-Alem, M. M. and Tapia, R. A., Numerical Experience with a Polyhedral Norm CDT Trust-Region Algorithm, Tech. Rep. TR93-26, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1993.