675

A SCALED TRUST REGION METHOD FOR A CLASS OF CONVEX OPTIMIZATION PROBLEMS *

R. J. Chen and R. R. Meyer

Computer Sciences Department and Mathematics Research Center

The University of Wisconsin-Madison

Madison, Wisconsin 53706 USA

ABSTRACT

Piecewise-linear approximation of nonlinear convex objective functions in linearly constrained optimization produces subproblems that may be solved as linear programs. This approach to approximation may be used for nonseparable as well as separable functions, and for the former class (the focus of this paper), it lies between linear and quadratic approximation with regard to its accuracy. In order to have additional control of the accuracy of the piecewise-linear approximation, we consider two devices : rectangular trust regions and dynamic scaling. The use of rectangular trust regions in conjuction with the type of piecewise-linear approximation considered here actually serves to simplify rather than complicate the approximating problems. This is a result of the equivalence of the trust region and the use of a limited number of segments in the approximation. The approach to dynamic scaling considered here may be applied to problems in which each objective function term is a convex function of a linear function of the variables. This scaling device allows the algorithm to adjust the approximation between an underestimating function (corresponding to a linear approximation) and an overestimating function (the nonseparable analog of the overestimate associated with separable approximation of convex functions.) The scaling factor is adjusted in accordance with the acceptance criteria associated with the trust region method.

Computational experience is cited for some large-scale problems arising from traffic assignment applications. The algorithm considered here also has the property that it allows such problems to be decomposed into a set of smaller optimization problems at each major iteration. These smaller problems correspond to linear single-commodity networks, and may be solved in parallel. Results are given for the distributed solution of these problems on the CRYSTAL multicomputer.

1. Introduction

Piecewise-linear approximation of convex nonlinear objective functions in linearly constrained optimization has the nice property of producing subproblems that may be solved as linear programs. This approach to approximation may be used for nonseparable as well as separable functions, and this paper deals with nonseparable convex objectives that are sums of terms of the form $f_j(\mathbf{c}_j \cdot \mathbf{x})$, where f_j is a continuously differentiable convex function defined on R^1 and $\mathbf{c}_j \cdot \mathbf{x}$ is a linear function of the problem variables $\mathbf{x}$. (See [Meyer 85] for details of a piecewise-linear trust region algorithm for the general

* This research was supported in part by NSF grant DCR-8503148 and AFOSR grant AFOSR-86-0194.

675

nonseparable case.) Although such a problem could be transformed into a separable problem by substitutions of the form $t_j = \mathbf{c}_j \cdot \mathbf{x}$ and the concatenation of this equation to the constraints, this transformation can have undesirable effects. For the multicommodity problems considered below, for example, it would introduce additional coupling between variables and thereby destroy the block structure of the constraints, and this new constraint would also have a different character from the other constraints, resulting in the destruction of the network nature of the initial constraints. Thus, we will demonstrate how objective functions of this type may be treated directly without the introduction of new constraints.

Piecewise-linear approximation lies between linear and quadratic approximation with regard to its accuracy. In order to control the error in the piecewise-linear approximation , we introduce two devices : rectangular trust regions and dynamic scaling. The use of rectangular trust regions in conjuction with this type of piecewise-linear approximation was first described in [Meyer 86]. With this approach, trust region constraints serve to simplify rather than to complicate the approximating subproblem, because the equivalence between the trust region constraints and the use of a limited number of piecewise-linear segments in the approximation allows the former to be handled implicitly. The approach to dynamic scaling considered here is a new approximation tool that takes advantage of the assumed form of the objective function terms. This device allows the algorithm to adjust the approximation between an underestimating function (corresponding to a linear approximation) and an overestimating function (the nonseparable analog of the overestimating property inherent in separable approximation of convex functions.) The scaling factor is adjusted in accordance with the acceptance criteria associated with the trust region method.

For notational simplicity we develop the theory below in the context of objective functions that are sums of terms of the form $f_j(x_{1j} + \cdots + x_{kj})$, i.e., each term involves the sum of a fixed number (namely, k) of variables. The extension of these results to the case in which the argument is a linear function of $\mathbf{x}$ is straightforward. We also consider a specific block structure of the constraints in order to emphasize the decomposition that is possible when piecewise-linear approximation is used appropriately in that context. However, the theory as presented is also independent of the nature of the linear constraints.

Computational experience is cited for some large-scale problems arising from traffic assignment applications. The algorithm considered here has the property that it allows such problems to be decomposed into smaller optimization problems at each major iteration. These smaller problems may be solved in parallel, and computational results are given for the distributed solution of these problems on the CRYSTAL multicomputer.

2. A Multicommodity Problem

Consider a directed network E of m nodes and n arcs. Let A be the node-arc matrix of E and k be the number of commodities sharing the network. Let x_{qj} denote the value of flow on arc j corresponding to commodity q. Let $\mathbf{x}_q$ and $\mathbf{b}_q$ denote the flow vector and supply-demand vector respectively of commodity q. If f_j $(j=1,...,n)$ is a set of continuously differentiable functions corresponding to the n arcs of the network, the corresponding multicommodity problem may be written as

$$\begin{array}{llllr}
\min \sum_{j=1}^{n} f_j(x_{1j}+x_{2j}+\cdots+x_{kj}) & & & & \\
\mathrm{A}\mathbf{x}_1 & & & = \mathbf{b}_1 & \\
& \mathrm{A}\mathbf{x}_2 & & = \mathbf{b}_2 & \\
& & \ddots & \vdots & \text{(MCP)} \\
& & \mathrm{A}\mathbf{x}_k & = \mathbf{b}_k & \\
x_{qj} \geq 0, \quad q=1,2,...,k, \; j=1,2,...,n & & & &
\end{array}$$

For notational convenience, we define

$\mathbf{x}_q := (x_{q1}, x_{q2}, \cdots, x_{qn}) \, \varepsilon \, R^n$ (flow vector of commodity q);

$\mathbf{x}_{.j} := (x_{1j}, x_{2j}, \cdots, x_{kj}) \, \varepsilon \, R^k$ (flow vector in arc j);

$\mathbf{x} := (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_k) \, \varepsilon \, R^{kn}$ (flow vector);

$f(\mathbf{x}) := \sum_{j=1}^{n} f_j(\sum_{q=1}^{k} x_{qj})$ (cost function);

$\Omega_q := \{ \mathbf{x}_q \, \varepsilon \, R_+^n \mid A\mathbf{x}_q = \mathbf{b}_q \}$ (feasible flow region of commodity q);

$\Omega := \{ \mathbf{x} \, \varepsilon \, R_+^{kn} \mid A\mathbf{x}_q = \mathbf{b}_q, \text{for } q=1,2,...,k \}$ (feasible flow region).

Network flow problems of the form MCP include computer network design [Cantor and Gerla 74], [Magnanti and Wong 84], traffic assignment [Bertsekas and Gafni 82], [Dafermos 80], [Dantzig, et al, 79], [Feijoo and Meyer 84], [Lawphongpanich and Hearn 83], [Pang and Yu 82], hydroelectric power systems [Hanscom, et al, 80], and telecommunications networks [McCallum 76]. In many cases, the network and the number of the commodities are large, so such problems can have hundreds of thousands of variables. To solve MCP, several algorithms have been proposed. Among the best known are (1) the Frank-Wolfe approach [LeBlanc, et al, 75]; (2) the column generation approach [Leventhal, et al, 73]; (3) the convex simplex approach [Nguyen 74]. Among these algorithms, only the Frank-Wolfe method leads to the decomposition of MCP into independent subproblems. The disadvantage of the Frank-Wolfe method is that it converges very slowly.

To accelerate convergence and to allow for parallelism, the approach of [Feijoo and Meyer 84] utilizes nonlinear separable approximation of the objective. Convex separable network flow problems have been successfully solved by many programming algorithms based on methods that iteratively generate search directions by solving approximating subproblems. Five algorithms that have been used are the Frank-Wolfe [LeBlanc, et al, 75], [Collins, et al, 78], convex simplex [Nguyen 74], [Rosenthal 81], reduced gradient [Mutagh and Saunders 78], [Dembo and Klincewicz 81], [Beck, at el, 83], piecewise-linearization [Kao and Meyer 81], [Kamesam and Meyer 84], [Monma and Segal 82], and Newton methods [Klincewicz 83]. Convex piecewise-linear networks can be reformulated as linear networks, allowing solution by extremely fast algorithms. The PL-approximation algorithm of [Kamesam and Meyer 84] is thus adopted in [Feijoo and Meyer 84] as a subroutine to solve the separable subproblems. This approach works well for small problems [Feijoo and Meyer 84,85], but for large-scale problems, a drawback appears. The computing time per major iteration increases after several iterations, indicating inefficiency in dealing with the global nature of the approximation. To remedy this, we incorporate in the algorithm both the trust region theory developed in [Meyer 85] and a more flexible method of constructing separable approximations.

3. A Scaled Separable Approximation

To obtain a good objective function approximation, we consider a class of scaled separable approximations in this section. This scaling has the property that the true objective function is bounded below and above by separable functions corresponding to appropriate choices of the scaling parameters. Letting f be convex, for a feasible point $\mathbf{x} \varepsilon \Omega$, we define a shifted function on R^{kn}

$$h(\mathbf{d}) := f(\mathbf{d}+\mathbf{x}) - f(\mathbf{x}).$$

This function corresponds to the change in the objective function resulting from a change of **d** in the current flow **x**. Thus, MCP is equivalent to

$$\min h(\mathbf{d}) \;\; s.t. \;\; \mathbf{d} \in \Omega_{\mathbf{x}} := \{ \mathbf{d} \mid \mathbf{x}+\mathbf{d} \in \Omega \}. \qquad (\text{MCP}_1)$$

For consistency , we list the notation associated with **h** in the same manner as before:

$\mathbf{d}_q := (d_{q1}, d_{q2}, \cdots , d_{qn})$ (flow vector changes in commodity q);

$\mathbf{d}_{\cdot j} := (d_{1j}, d_{2j}, \cdots , d_{kj})$ (flow vector changes in arc j);

$\mathbf{d} = (\mathbf{d}_1, \mathbf{d}_2, \cdots , \mathbf{d}_k)$ (flow vector changes);

$h_j(\cdot) := f_j(\cdot + \sum_q x_{qj}) - f_j(\sum_q x_{qj})$ (shifted cost function for arc j);

$h^S_{qj}(d_{qj},\sigma) := \frac{1}{\sigma} h_j(\sigma d_{qj})$ (term associated with commodity q on arc j, scaled by σ);

$h^S(\mathbf{d},\sigma) := \sum_j \sum_q \frac{1}{\sigma} h_j(\sigma d_{qj})$ (*scaled separable approximation* of h).

For a $\sigma > 0$, $h^S(\mathbf{d},\sigma)$ is called a *scaled separable approximation with scale factor* σ. (A scaled separable approximation with $\sigma=0$ will also be established by considering limits below.) Some simple examples of scaled separable approximations will now be given. (The upper and lower bounds for h will be established in Corollary 3.4.)

Example 1 : $f(x) = x^2$ (k=1, n=1)

[1] $\bar{x} = 0$: ($\bar{x}$ denotes current point)

$h(d) = h_1(d) = d^2$

$h^S(d,\sigma) = \frac{1}{\sigma}(\sigma d)^2 = \sigma d^2$

(1) $\sigma = 1$ (*upper bound*)

$h^S(d,1) = d^2 = h(d)$

(2) $\sigma = 0$ (*lower bound*)

$h^S(d,0) = 0 \; (= f'(0) \cdot d)$

[2] $\bar{x} = 1$:

$h(d) = h_1(d) = (1+d)^2 - 1 = 2d + d^2$

$h^S(d,\sigma) = \frac{1}{\sigma}(2(\sigma d) + (\sigma d)^2) = 2d + \sigma d^2$

(1) $\sigma = 1$ (*upper bound*)

$h^S(d,1) = 2d + d^2 = h(d)$

(2) $\sigma = 0$ (*lower bound*)

$h^S(d,0) = 2d$ ($= f'(1) \cdot d$)

Example 2 : $f(x,y) = (x+y)^2$ (k=2, n=1)

[1] $(\bar{x},\bar{y}) = (0,0)$: ($(\bar{x},\bar{y})$ denotes current point)

$h(\mathbf{d}) = h_1(d_{11}+d_{21}) = (d_{11}+d_{21})^2$

$h^S(\mathbf{d},\sigma) = \frac{1}{\sigma}(\sigma d_{11})^2 + \frac{1}{\sigma}(\sigma d_{21})^2 = \sigma(d_{11}^2 + d_{21}^2)$

(1) $\sigma = 2$ (*upper bound*)

$h^S(d,2) = 2(d_{11}^2 + d_{21}^2)$

(2) $\sigma = 0$ (*lower bound*)

$h^S(\mathbf{d},0) = 0$ ($= \nabla f(\mathbf{0}) \cdot \mathbf{d}$)

[2] $(\bar{x},\bar{y}) = (1,1)$:

$h(\mathbf{d}) = h_j(d_{11}+d_{21}) = (1+d_{11}+1+d_{21})^2 - (1+1)^2 = (d_{11}+d_{21})^2 + 4(d_{11}+d_{21})$

$h^S(\mathbf{d},\sigma) = \frac{1}{\sigma}((\sigma d_{11})^2 + 4(\sigma d_{11})) + \frac{1}{\sigma}((\sigma d_{21})^2 + 4(\sigma d_{21})) = \sigma(d_{11}^2 + d_{21}^2) + 4(d_{11}+d_{21})$

(1) $\sigma = 2$ (*upper bound*)

$h^S(d,2) = 2(d_{11}^2 + d_{21}^2) + 4(d_{11}+d_{21})$

(2) $\sigma = 0$ (*lower bound*)

$h^S(\mathbf{d},0) = 4(d_{11}+d_{21})$ ($= \nabla f(1,1) \cdot \mathbf{d}$)

The error in the approximation in example 2 is $\sigma(d_{11}^2 + d_{21}^2) - (d_{11}+d_{21})^2$ (independent of the base point). The contours of error functions for σ=0, σ=1, and σ=2 are shown in Figure 1, 2, and 3.

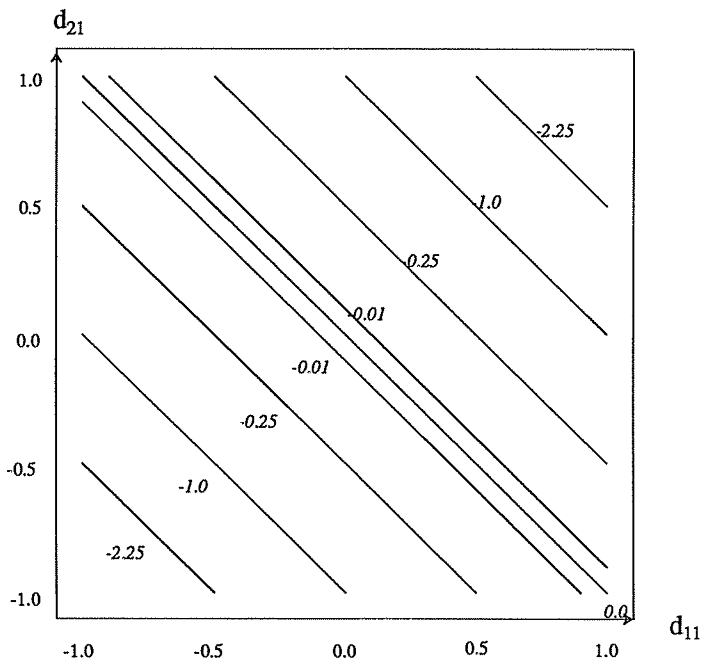


Figure 1 Error Contours for σ=0

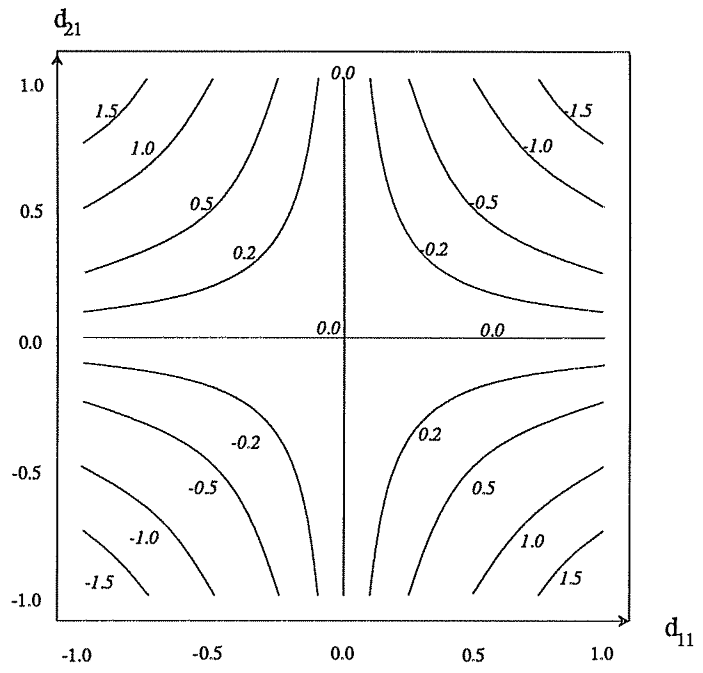


Figure 2 Error Contours for σ=1

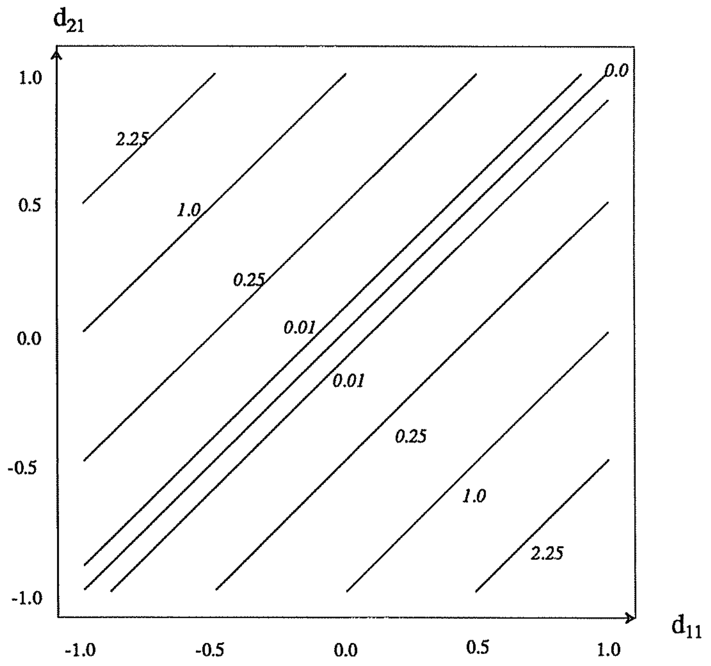


Figure 3 Error Contours for σ=2

Observe that h_j and $h(\mathbf{d})=\sum_j h_j(\sum_q d_{qj})$ are convex, and $h_j(0)=0$ for $j=1,2,\dots,n$. By these properties, we have the following lemma.

Lemma 3.1 $h(\mathbf{d}) \le h^S(\mathbf{d},k)$.

Proof: By the convexity of h_j, and $\sum_{q=1}^{k}\frac{1}{k}=1$, we have

$$h_j(\sum_{q=1}^{k} d_{qj}) = h_j(\sum_{q=1}^{k}\frac{1}{k}\cdot kd_{qj}) \le \sum_{q=1}^{k}\frac{1}{k}\cdot h_j(kd_{qj})$$

The lemma follows by applying $\sum_{j=1}^{n}$ on both sides. □

We now show the monotonicity of $h^S(\mathbf{d},\cdot)$.

Lemma 3.2 For $0<\sigma_1\le\sigma_2$, $h^S(\mathbf{d},\sigma_1) \le h^S(\mathbf{d},\sigma_2)$.

Proof: By the convexity of h_j, and $h_j(0)=0$, we have

$$h_j(\sigma_1 d_{qj}) = h_j(\frac{\sigma_1}{\sigma_2}\cdot\sigma_2 d_{qj}+\frac{\sigma_2-\sigma_1}{\sigma_2}\cdot\sigma_2\cdot 0) \le \frac{\sigma_1}{\sigma_2}\cdot h_j(\sigma_2 d_{qj})$$

The lemma follows by applying $\sum_{j=1}^{n}\sum_{q=1}^{k}\sigma_1^{-1}$ on both sides. □

The following lemma shows that the limit as $\sigma\to 0$ of the separable functions $h^S(\mathbf{d},\sigma)$ is the linearization approximation at 0.

Lemma 3.3 For fixed $\mathbf{d}$, $h^S(\mathbf{d},\sigma) \to \nabla h(\mathbf{0})\mathbf{d}=\nabla f(\mathbf{x})\mathbf{d}$, as $\sigma\to 0^+$.

Proof: By L'Hospital's Rule, we have

$$\lim_{\sigma\to 0^+}\frac{h_j(\sigma d_{qj})}{\sigma} = \lim_{\sigma\to 0^+} h_j'(\sigma d_{qj})\cdot d_{qj} = h_j'(0)\cdot d_{qj};$$

and by the definition of h_j,

$$\frac{\partial h(\mathbf{0})}{\partial d_{qj}} = h_j'(0) = f_j'(\textstyle\sum_q x_{qj}) = \frac{\partial f(\mathbf{x})}{\partial x_{qj}}. \quad \square$$

Accordingly, if we define $h^S(\mathbf{d},0):=\nabla h(\mathbf{0})\mathbf{d}$, then $h^S(\mathbf{d},\cdot)$ is a non-decreasing continuous function on $[0,\infty)$. Since the gradient approximation yields a lower bound, the next corollary notes that the choices $\sigma=0$, $\sigma=k$ yield lower and upper bounds respectively on the objective function.

Corollary 3.4 $h^S(\mathbf{d},0) \le h(\mathbf{d}) \le h^S(\mathbf{d},k)$.

Thus, $h^S(\mathbf{d},k)$ serves as an upper bound of $h(\mathbf{d})$, while $h^S(\mathbf{d},0)$ serves as a lower bound of $h(\mathbf{d})$. To construct a good separable approximation of the objective is a key factor in the efficiency of the trust region method. The continuity and monotonicity of $h^S(\mathbf{d},\cdot)$ indicate that with a proper σ, $h^S(\mathbf{d},\sigma)$ is a suitable approximation. We allow the scalar σ to be adjusted at every iteration of our algorithm. The amount of increase or decrease of σ is determined by the ratio of the true and approximate objective function improvements in the preceding iteration.

The next lemma is used to show that the direction obtained from the separable program (in which the scaled separable approximate function replaces the objective) is a descent direction.

Lemma 3.5 For $\sigma \geq 0$, $\nabla h^S(\mathbf{0},\sigma) = \nabla h(\mathbf{0})$.

Proof: For $\sigma=0$, the result is trivial.

For $\sigma>0$,

$$\frac{\partial h^S(\mathbf{d},\sigma)}{\partial d_{qj}} = \frac{\partial}{\partial d_{qj}}\left(\frac{h_j(\sigma d_{qj})}{\sigma}\right) = h_j'(\sigma d_{qj}).$$

The lemma then follows from

$$\frac{\partial h^S(\mathbf{0})}{\partial d_{qj}} = h_j'(0) = \frac{\partial h(\mathbf{0})}{\partial d_{qj}} \quad .\Box$$

The next lemma establishes the descent relation between f and its separable approximations.

Lemma 3.6 For $\sigma \geq 0$, if $h^S(\mathbf{d},\sigma) < 0$, then $\mathbf{d}$ is a descent direction of f at $\mathbf{x}$.

Proof: For $\sigma=0$, the result is trivial.

For $\sigma>0$, by the convexity of $h^S(\cdot,\sigma)$ and $h^S(\mathbf{0},\sigma)=0$, we have

$$0 > h^S(\mathbf{d},\sigma) \geq h^S(\mathbf{0},\sigma) + \nabla h^S(\mathbf{0},\sigma)\cdot\mathbf{d} = \nabla h(\mathbf{0})\cdot\mathbf{d} = \nabla f(\mathbf{x})\cdot\mathbf{d}$$

So $\mathbf{d}$ is a descent direction of f at $\mathbf{x}$. $\Box$

Assuming a σ has been selected, we consider an approximating separable program

$$\min h^S(\mathbf{d},\sigma) \;\; s.t. \;\; \mathbf{d} \in \Omega_{\mathbf{x}} \qquad \text{(SP)}$$

If we collect terms corresponding to commodity q and define $h_q^S(\mathbf{d}_q,\sigma) := \sum_{j=1}^{n} h_{qj}^S(d_{qj},\sigma)$, then the objective $h^S(\mathbf{d},\sigma) = \sum_{q=1}^{k} h_q^S(\mathbf{d}_q,\sigma)$. The separable program then can be decomposed into k subprograms of the form:

$$\min h_q^S(\mathbf{d}_q,\sigma) \;\; s.t. \;\; \mathbf{d}_q \,\varepsilon\, \Omega_{\mathbf{x}_q} := \{ \, \mathbf{d}_q \mid \mathbf{x}_q + \mathbf{d}_q \,\varepsilon\, \Omega_q \, \} \qquad (\mathrm{SP}_q)$$

The final step in the approximation process is to replace h^S by a piecewise-linear approximation h^{PL}. Specifically, let h^{PL} be the piecewise-linear approximation of h^S with fixed mesh-size $\delta>0$. That is,

$$h^{PL}(\mathbf{d},\sigma) := \sum_j \sum_q \frac{1}{\sigma} h_j^{PL}(\sigma d_{qj}) \, ,$$

where the functions $h_j^{PL}(\sigma d_{qj})$ are piecewise-linear functions that agree with $h_j(\sigma d_{qj})$ at a set of mesh points to be described below. We define PLP and PLP_q analogous to SP and SP_q:

$$\min h^{PL}(\mathbf{d},\sigma) \;\; s.t. \;\; \mathbf{d} \,\varepsilon\, \Omega_{\mathbf{x}} \qquad (\mathrm{PLP})$$

$$\min h_q^{PL}(\mathbf{d}_q,\sigma) \;\; s.t. \;\; \mathbf{d}_q \,\varepsilon\, \Omega_{\mathbf{x}_q}. \qquad (\mathrm{PLP}_q)$$

Moreover, these subproblems can be solved in parallel [Feijoo and Meyer 85]. If the trust region is large enough to contain the original feasible set, then the method of Feijoo and Meyer corresponds to $\sigma=1$, while the Frank-Wolfe method [LeBlanc, et al., 75] corresponds to $\sigma=0$.

Since $h^{PL} \geq h^S$, the preceding lemma can be used to show that a descent direction is also obtained if the optimal value of PLP is less than zero.

Lemma 3.7 For $\sigma\geq 0$, if $h^{PL}(\mathbf{d},\sigma) < 0$, then $\mathbf{d}$ is a descent direction of f at $\mathbf{x}$.

In the algorithms below, we use h^{PL} as the approximating function. This approximation has the advantage that the resulting subproblems may be solved as linear programs.

4. A Trust Region Algorithm

The trust region technique for unconstrained nonlinear programs is described in [More and Sorensen 79], [Fletcher 81], [Sorensen 82] and was further developed by many other authors to solve constrained problems [Zhang, et al, 84], [Vardi 85], [Meyer 85]. The trust region method of this paper, a modified version of that in [Meyer 85], may be contrasted with the trust region algorithm of [Zhang,et al. 84]. The difference lies in the fact that the underlying approximate objective in [Zhang,et al. 84] is a linear function while ours is not only a piecewise-linear separable function as in [Feijoo and Meyer 85], but also is dynamically scaled at each iteration. Computational experience cited in section 5 indicates that the trust region algorithm utilizing scaled PL approximation converges much more rapidly than algorithms based on linear approximation, yet requires only slightly more computing time per iteration.

For the development of our algorithm, if $\mathbf{x}$ is a feasible solution of MCP, we define notation as follows:

(1) the trust region :

$\Lambda_{\alpha_q}=\{\ \mathbf{d}_q\ \varepsilon\ R^n\ \mid -\alpha_q \le d_{qj} \le \alpha_q,\ j=1,...,n\ \}$: *trust region* for commodity q

$\Lambda_\alpha=\{\ \mathbf{d}\ \varepsilon\ R^{kn}\ \mid -\alpha_q \le d_{qj} \le \alpha_q,\ q=1,...,k,\ j=1,...,n\ \}$: *trust region*

$\alpha=(\ \alpha_1,\cdots,\alpha_k\ )'$: *trust region* vector

$\underline{\alpha}$: threshold for α

γ_1, γ_2 : reduction factors

(2) linear approximation :

$\mathbf{d}_q^L(\alpha_q)$: a *solution* of $\mathrm{LP}_q(\alpha_q)$ (min $\nabla h_q(0)\mathbf{d}_q$ *s.t.* $\mathbf{d}_q\ \varepsilon\ \Omega_{\mathbf{x}_q} \cap \Lambda_{\alpha_q}$)

$h^L(\mathbf{d}) := \nabla f(\mathbf{x})\mathbf{d}$

$\mathbf{d}^L(\alpha)$: a *solution* of $\mathrm{LP}(\alpha)$ (min $h^L(\mathbf{d})$ *s.t.* $\mathbf{d}\ \varepsilon\ \Omega_{\mathbf{x}} \cap \Lambda_\alpha$)

(3) separable approximation :

σ : scale factor

$\bar{\sigma}$: upper bound of scale factor

$\mathbf{d}_q^S(\alpha_q,\sigma)$: a *solution* of $\mathrm{SP}_q(\alpha_q,\sigma)$ (min $h_q^S(\mathbf{d}_q,\sigma)$ *s.t.* $\mathbf{d}_q\ \varepsilon\ \Omega_{\mathbf{x}_q} \cap \Lambda_{\alpha_q}$)

$\mathbf{d}^S(\alpha,\sigma)$: a *solution* of $\mathrm{SP}(\alpha,\sigma)$ (min $h^S(\mathbf{d},\sigma)$ *s.t.* $\mathbf{d}\ \varepsilon\ \Omega_{\mathbf{x}} \cap \Lambda_\alpha$)

(4) piecewise-linear approximation :

δ_q : mesh-size used in PL_q ($\delta_q = \alpha_q/2^i$ for some integer i)

δ : mesh-size vector used in PL

$\mathbf{d}_q^{PL}(\alpha_q,\sigma,\delta_q)$: a *solution* of $\mathrm{PL}_q(\alpha_q,\sigma,\delta_q)$ (min $h_q^{PL}(\mathbf{d}_q,\sigma,\delta_q)$ *s.t.* $\mathbf{d}_q\ \varepsilon\ \Omega_{\mathbf{x}_q} \cap \Lambda_{\alpha_q}$)

$\mathbf{d}^{PL}(\alpha,\sigma,\delta)$: a *solution* of $\mathrm{PL}(\alpha,\sigma,\delta)$ (min $h^{PL}(\mathbf{d},\sigma,\delta)$ *s.t.* $\mathbf{d}\ \varepsilon\ \Omega_{\mathbf{x}} \cap \Lambda_\alpha$)

(5) improvement ratios :

$\eta(\alpha,\sigma,\delta) := \dfrac{h^{PL}(\mathbf{d}^{PL}(\alpha,\sigma,\delta),\sigma)}{h^S(\mathbf{d}^S(\alpha,\sigma),\sigma)}$ = ratio of piecewise-linear and separable optima

η_0 : threshold of η-ratio ($0 < \eta_0 < 1$)

$\rho(\mathbf{d},\sigma) := \dfrac{h(\mathbf{d})}{h^{PL}(\mathbf{d},\sigma)}$ = ratio of h and h^{PL} at $\mathbf{d}$

$\rho_0, \rho_1, \rho_2, \rho_3$: thresholds of ρ-ratio ($0<\rho_0<\rho_1<1<\rho_2<\rho_3$)

(6) the line-search :

L : maximum number of function evaluations per line-search

$\mathbf{d}^{LS}(l)$: lth point in line-search

Where not ambiguous, some arguments of $\mathbf{d}_q^L, \mathbf{d}_q^S, \mathbf{d}_q^{PL}, \mathbf{d}^L, \mathbf{d}^S, \mathbf{d}^{PL}$, $h, h^L, h^S, h^{PL}, \eta, \rho$ are omitted; and the current point on which these are based will be added as needed. For simplicity, we add a superscript i on each item of notation to express the association with the ith current solution $\mathbf{x}^i$.

The η-ratio defined in (5) is used in the convergence theory below, but need not be computed in practice. In implementation, we can take the threshold η_0 as a very small positive value so that we don't need to calculate the denominator of η if the numerator is greater than a tolerance. Alternatively, the algorithm PLTR may be modified by bypassing the η-ratio test in **step 2** whenever $h(\mathbf{d}^{PL}) < -\tau$, where τ is a positive tolerance. (the proof of the convergence theorem is easily changed to take into account this modification)

The algorithm to be described below determines for each distinct $\mathbf{x}^i$ a trust region vector that provides at least a value η_0 for the η-ratio and a value ρ_0 for the ρ-ratio. This is accomplished in two steps: 1) the bound for the η-ratio is attained by refining the mesh-size as needed in the piecewise-linear subprograms, and 2) the bound for the ρ-ratio is attained either by decreasing the trust region vector α as needed, or by doing a line-search. In addition, we also use a suitable scale factor to adjust the objective approximation in order to achieve ρ_0-ratio quickly. The use of the line-search in our algorithm (when the ρ-ratio bound is not initially satisfied) is not shared by the traditional trust region methods.

Algorithm PLTR

step 0: initialization:

Select real numbers $\gamma_1, \gamma_2, \underline{\alpha}, \rho_0, \rho_1, \rho_2, \rho_3, \eta_0, \bar{\sigma}$, and integer L;

Select initial values α^0 and σ^0;

(with restrictions $0<\gamma_1<\gamma_2<1$, $0<\underline{\alpha}<\alpha^0$, $0<\rho_0<\rho_1<1<\rho_2<\rho_3$, $0<\eta_0<1$, $0<\sigma^0<\bar{\sigma}$, $0<L$)

Set i=0;

step 1: find a feasible solution:

```
    Find $\mathbf{x}^0 \varepsilon \Omega$;
**while** tolerance of solution not satisfied **do**
    **step 2: solve piecewise-linear programs:**
    $i \leftarrow i+1$;
    ( refine mesh until $\eta$-ratio satisfactory )
    for $q = 1, k$
        $\delta_q^i \leftarrow 2\alpha_q^i$;
        ( reduce mesh size by factor of 1/2 )
        do $\delta_q^i \leftarrow \delta_q^i/2$; solve $PL_q(\alpha_q^i, \delta_q^i, \mathbf{x}^i)$;
        until $h_q^{PL}(\mathbf{d}_q^{PL}(\delta_q^i)) \leq \eta_0 \cdot h_q^S(\mathbf{d}_q^S)$
    endfor;
    **step 3: do $\rho$-ratio test:**
    $\rho^i \leftarrow \rho(\mathbf{d}^{PL})$
    if $\rho^i \geq \rho_0$ then
        **step 4: no line-search needed:**
        $\mathbf{x}^i \leftarrow \mathbf{x}^i + \mathbf{d}^{PL}$;
        case ( update trust region size and scaling factor )
            $\rho^i \leq \rho_1$ :  $\alpha^i \leftarrow \max\{\gamma_2\alpha^i, \underline{\alpha}\}, \sigma^i \leftarrow \min\{2\sigma^i, \bar{\sigma}\}$;
            $\rho_1 < \rho^i \leq \rho_2$ :  $\alpha^i \leftarrow \max\{\gamma_1\alpha^i, \underline{\alpha}\}, \sigma^i \leftarrow \sigma^i$;
            $\rho_2 < \rho^i \leq \rho_3$ :  $\alpha^i \leftarrow \alpha^i, \sigma^i \leftarrow 0.75\sigma^i$;
            $\rho^i > \rho_3$ :  $\alpha^i \leftarrow \alpha^i, \sigma^i \leftarrow 0.5\sigma^i$;
        endcase;
    else
        **step 5: line-search:**
        ( attempt to satisfy $\rho$-ratio test by line-search )
        for $l = 1, L$
            $\rho^i \leftarrow \dfrac{h(\mathbf{d}^{LS}(l))}{h^{PL}(\mathbf{d}^{PL})}$;
            if $\rho^i \geq \rho_0$ then $\mathbf{x}^i \leftarrow \mathbf{x}^i + \mathbf{d}^{LS}(l)$; $\alpha^i \leftarrow \max\{\gamma_2\alpha^i, \underline{\alpha}\}$; $\sigma^i \leftarrow \min\{2\sigma^i, \bar{\sigma}\}$; goto **step 2**;
        endfor;
        ( $\rho$-ratio unattained; contract trust region )
        $\alpha^i \leftarrow \gamma_2\alpha^i$; $\sigma^i \leftarrow \min\{2\sigma^i, \bar{\sigma}\}$;
    endif;
endwhile
```

Remarks:

1) In **step 1**, the feasible solution can be found by solving a linear network program, where the linear objective function in each commodity can be taken to be $f_j'(0)$ for each arc.

2) In the η-ratio test of **step 2**, $h^S(\mathbf{d}^S(\alpha))$ can be replaced by a lower bound, obtained by the primal method or dual method in [Kamesam and Meyer 84], [Kao and Meyer 81] or by $h^L(\mathbf{d}^L(\alpha))$ or by the alternatives discussed above.

3) If needed, the line-search is used in **step 5**. For simplicity, we use golden-section search since typically only one function evaluation is required.

4) The initial value of the trust region vector for each distinct $\mathbf{x}^i$ is at least $\underline{\alpha}$, since the value of x^i changes only in **step 4** or in **step 5**, where the new value of α is set to at least $\underline{\alpha}$.

Before proving the main convergence theorem, we consider two types of optimality conditions for MCP.

Lemma 4.1 $\mathbf{x}$ is an optimal solution of MCP if and only if $\mathbf{d}^L\mathbf{=0}$ solves LP(α,x) for α>0.

Proof: LP(α,x) solved by $\mathbf{d} = \mathbf{0}$

=> KKT for LP(α,x) satisfied at $\mathbf{d=0}$

=> KKT for MCP$_1$(x) satisfied at $\mathbf{d=0}$

=> $\mathbf{d=0}$ is optimal for MCP$_1$(x) because of convexity of h($\mathbf{d}$).

The analogous argument holds in the other direction. □

Lemma 4.2 For $\sigma\geq0$, $\mathbf{x}$ is an optimal solution of MCP if and only if $\mathbf{d}^S\mathbf{=0}$ solves SP(α,σ,x) for α>0.

Proof: The result follows from the above lemma because h^S and and h have the same gradient. □

This lemma implies that if **x** is not a solution of MCP, then for any α>**0** and $\sigma \geq 0$, $h^S(\mathbf{d}^S(\alpha,\sigma),\sigma)<0$.

By arguments analogous to those used in [Meyer 85], it may be shown that for any $\sigma \varepsilon [0,\bar{\sigma}]$, $h^s(\cdot,\sigma) = h^{PL}(\cdot,\sigma)+o(\alpha) = h(\cdot)+o(\alpha)$. By using the first of these properties, we establish the finiteness of the iteration in step 2 of algorithm PLTR.

Lemma 4.3 For $0<\eta_0<1$, α>**0**, and $0 \leq \sigma$, if **x** is not an optimal solution of MCP, then $\eta(\alpha,\sigma,\delta) \geq \eta_0$ with δ obtained from **step 2.**

Proof: Suppressing arguments to simplify notation, we have

$$h^S(\mathbf{d}^S) \leq h^S(\mathbf{d}^{PL}) \leq h^{PL}(\mathbf{d}^{PL}) \leq h^{PL}(\mathbf{d}^S) = h^S(\mathbf{d}^S)+o(\delta).$$

Thus, $h^{PL}(\mathbf{d}^{PL})=h^S(\mathbf{d}^S)+o(\delta)$, and since $h^S(\mathbf{d}^S) \neq 0$, the η-ratio bound is obtained as $\delta \to 0$. □

The following three lemmas may be proved in a manner analogous to the proofs of lemmas 4,5,6 of [Meyer 85].

Lemma 4.4 For $0<\rho_0<1$ and $\bar{\sigma} \geq 0$, if **x** is not an optimal solution of MCP, then for α small enough, $\rho(\mathbf{d}^{PL},\sigma) \geq \rho_0$.

In order to guarantee that the improvement ratio behaves properly in the neighborhood of any accumulation point $\bar{\mathbf{x}}$ of a sequence $\mathbf{x}^i$, we now observe that $h^S(\mathbf{d}^S(\alpha,\sigma,\mathbf{x}),\sigma,\mathbf{x})$ is continuous for a fixed scale factor σ.

Lemma 4.5 For $\sigma \geq 0$, $h^S(\mathbf{d}^S(\alpha,\sigma,\mathbf{x}),\sigma,\mathbf{x})$ is a continuous function of **x** and α for $\mathbf{x}\ \varepsilon\ \Omega$ and $\alpha \geq \mathbf{0}$.

The key factor in the validity proof of the algorithm is the guarantee of a minimum improvement ratio in a neighborhood of non-optimal points. In the following, ρ_0 is a parameter in (0,1), $\mathbf{y}^i\ \varepsilon\ \Omega$, and $\rho(\mathbf{d},\sigma,\mathbf{y}^i)$ is as defined before.

Lemma 4.6 If $\mathbf{y}^i \to \bar{\mathbf{y}}$, where $\bar{\mathbf{y}}$ is not a solution of MCP, then there exists an $\bar{\alpha}$>**0** such that $\rho(\mathbf{d}^{PL},\sigma,\mathbf{y}^i) \geq \rho_0$ for all $\alpha \varepsilon (\mathbf{0},\bar{\alpha})$, $\bar{\sigma} \geq \sigma \geq 0$, and all $\mathbf{y}^i$ sufficiently close to $\bar{\mathbf{y}}$.

The main convergence theorem of the algorithm now follows in a straightforward manner.

Theorem 4.7 If $\bar{\mathbf{x}}$ is an accumulation point of a sequence generated by Algorithm PLTR, then $\bar{\mathbf{x}}$ is an optimal solution of MCP.

Proof:

Assume the result is false, and let $\{\mathbf{x}^{i_k}\}$ be a subsequence such that $\mathbf{x}^{i_k} \to \bar{\mathbf{x}}$. Using the preceding lemma, we consider those sufficiently large i_k such that $\rho(\mathbf{d}^{i_kPL},\sigma,\mathbf{x}^{i_k}) \geq \rho_0$ for all $\alpha \varepsilon (\mathbf{0},\bar{\alpha})$, $\sigma \geq 0$. Moreover, since the initial value of α for each distinct $\mathbf{x}^{i_k}$ is at least $\underline{\alpha}$, it is the case that for arbitrarily large i_k that $\alpha^{i_k} \geq \alpha^* := \min\{\gamma_1 \bar{\alpha}, \underline{\alpha}\}$ (since the trust region vector is not reduced below this quantity to achieve the required improvement ratio) and $\rho^{i_k} \geq \rho_0$. Letting $\theta_0 := h^S(\mathbf{d}^S, \alpha^*, \bar{\sigma}, \bar{\mathbf{x}})$, we then have for the iterations without line-search,

$$h^{i_k}(\mathbf{d}^{i_kPL}) \leq \rho_0 \cdot h^{i_kPL}(\mathbf{d}^{i_kPL}(\alpha^{i_k},\sigma^{i_k}),\sigma^{i_k}) \quad (\text{ step } 4\ )$$

$$\leq \rho_0 \cdot \eta_0 \cdot h^{i_kS}(\mathbf{d}^{i_kS}(\alpha^{i_k},\sigma^{i_k}),\sigma^{i_k}) \quad (\text{ step } 2\ )$$

$$\leq \rho_0 \cdot \eta_0 \cdot h^{i_kS}(\mathbf{d}^{i_kS}(\alpha^{i_k},\bar{\sigma}),\bar{\sigma}) \quad (\text{ Lemma } 3.2\ )$$

$$\leq \rho_0 \cdot \eta_0 \cdot \theta_0 / 2 \quad (\text{ Lemma } 4.5\ )$$

while for the iterations with line-search, the first inequality above holds for $\mathbf{d}^{i_kLS}$ from step 5. However, for $\mathbf{x}^{i_k}$ sufficiently close to $\bar{\mathbf{x}}$, the relations

$$f(\mathbf{x}^{i_{k+1}}) - f(\mathbf{x}^{i_k}) \leq f(\mathbf{x}^{i_k+1}) - f(\mathbf{x}^{i_k}) \quad (\ f^{\,i} \text{ is decreasing }\ )$$

$$= h^{i_k}(\mathbf{d}^{i_kLS}) \ or\ h^{i_k}(\mathbf{d}^{i_kPL}) \quad (\text{ step 4 or step 5 })$$

$$\leq \rho_0 \cdot \eta_0 \cdot \theta_0 / 2$$

contradict $f(\mathbf{x}^{i_k}) \to f(\bar{\mathbf{x}})$. □

5. Computational Results

This section describes computational results for five standard traffic assignment problems (see Figure 4 for the first four networks). The sources of these problems are: [Nguyen and Dupuis 84] (Problem A), [Steenbrink 74] (Problem B), [Bertsekas and Gafni 82] (Problem C), [LeBlanc and Morlok 75] (Sioux Falls Problem for Sioux Falls, South Dakota), and [Nguyen and Dupuis 84] (Hull Problem for Hull, Canada). Table 1 shows the comparison of three sequential algorithms executed on a Microvax II. The sequential version of PLTR (in which the single commodity subproblems are solved sequentially) may be compared with the Frank-Wolfe trust region algorithm [Zhang, et al. 84] and the reduced gradient algorithm (as implemented in MINOS, see [Murtagh and Saunders 79]). Table 2 demonstrates the performance of the parallel version of the PLTR algorithm as implemented on the Crystal Multicomputer (a local area network of Vax750s). All the computer codes are written in standard FORTRAN 77 using double precision throughout. The compiler used is Berkeley 4.3 FORTRAN 77. The parallel program uses SAP (Simple Application Package), which is a communication package for Crystal nodes. The timings reported are exclusive of input and output.

The following remarks explain certain details in the implementation of our method on the Crystal Multicomputer.

1) **initialization:**

Choose $\gamma_1=0.5$, $\gamma_2=0.75$, $\rho_0=0.3$, $\rho_1=0.8$, $\rho_2=1.3$, $\rho_3=2$, $L=3$, and $\sigma_0=1$.

2) **feasible solution:**

The initial iterate $\mathbf{x}^0$ was taken to be the element of Ω corresponding to the solution of the linear network problem with costs given by $f_j'(0)$.

3) **network subroutine:**

A modified version of RNET [Grigoriadis and Hsu 79] is used to solve piecewise-linear subproblems.

4) **line-search:**

In general, the optimal solution from the trust region is satisfactory, and a line-search is unnecessary. When a line-search is called for, golden section search is used for simplicity. Typically, only one function evaluation is needed in this case to achieve the required ρ-ratio .

5) **stopping criterion:**

We calculate a lower bound for MCP every few iterations. Given a current feasible solution $\mathbf{x}^i$, a lower bound is obtained by computing the optimal objective value of the linearized problem :

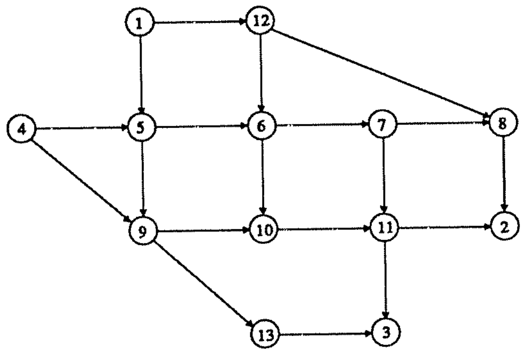
$$\min f(\mathbf{x}^i)+\nabla f(\mathbf{x}^i)\mathbf{x} \quad s.t. \quad \mathbf{x} \, \varepsilon \, \Omega$$

(This problem may be decomposed and solved in parallel.) If the gap between the current objective value and the best lower bound is below the preset tolerance then stop. The test results indicate that the lower bounds obtained from the linearized linear programs are not very tight relative to the upper bounds from the feasible solutions. We also stop the algorithm if the improvements of the objective functions are sufficiently small in 3 consecutive iterations. The eight-figure agreement in the objective values produced by PLTR and MINOS for most of the test problems suggests that the results are correct to more significant figures than the lower bound would indicate.

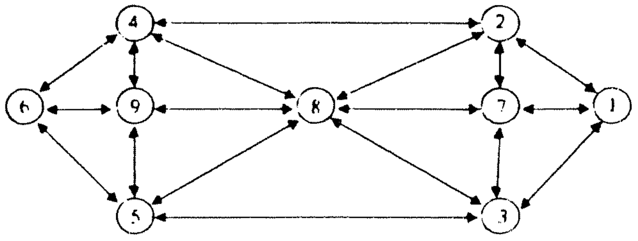
6) **parallel implementation on Crystal:**

A description of the Crystal Multicomputer and its communications software is given in [DeWitt, et al. 84] and [Feijoo 85]. Additional details of the implementation of parallel algorithms for traffic assignment on CRYSTAL are provided in [Feijoo and Meyer 84], [Feijoo 85], and [Chen and Meyer 86].

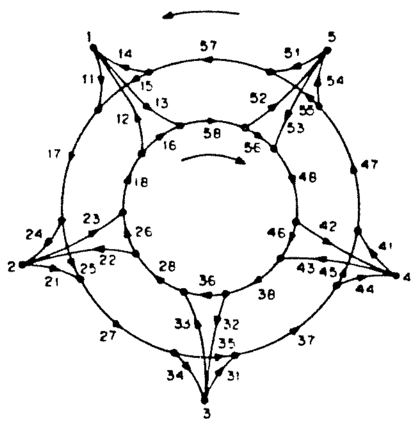
(1) Problem A



(2) Problem B



(3) Problem C



(4) Sioux Falls Problem

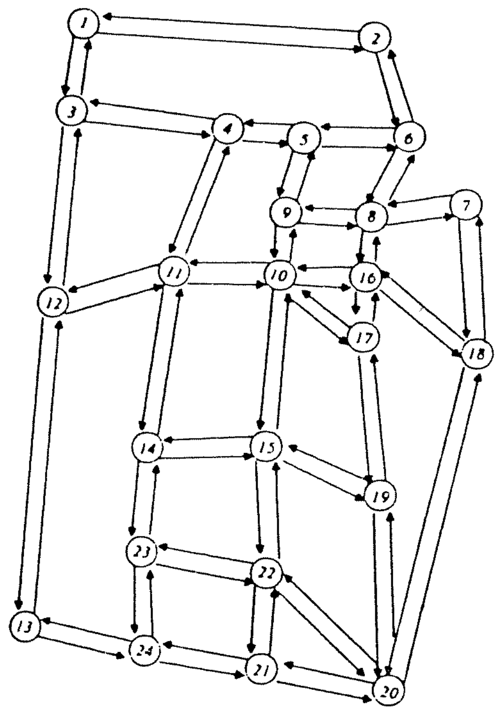


Figure 4 Four test problems

Problem A $f = \sum a_j t_j^2 + b_j t_j$

Dimension				Algorithm	FWTR	PLTR	RG
Comms	4	ODs	4	Iter	24	14	21
Nodes	13	Arcs	19	CPU	4s	2.9s	6.3s
Cnstrs	52	Vars	76	Obj	85028.071	85028.071	85028.071
LB	85021.021			Gap	7.050	7.050	7.050

Problem B $f = \sum a_j t_j^2 + b_j t_j$

Dimension				Algorithm	FWTR	PLTR	RG
Comms	12	ODs	12	Iter	57	23	106
Nodes	9	Arcs	36	CPU	33.1s	18.8s	32.9s
Cnstrs	108	Vars	432	Obj	16957.685	16957.674	16957.674
LB	16957.086			Gap	0.599	0.588	0.588

Problem C $f = \sum a_j t_j^3 + b_j t_j^2 + c_j t_j$

Dimension				Algorithm	FWTR	PLTR	RG
Comms	5	ODs	5	Iter	35	23	54
Nodes	25	Arcs	40	CPU	17.6s	14.9s	17.5s
Cnstrs	125	Vars	200	Obj	5934.7617	5924.3427	5924.3427
LB	5923.9289			Gap	10.8328	0.4138	0.4138

Sioux Falls Problem $f = \sum a_j t_j^5 + b_j t_j$

Dimension				Algorithm	FWTR	PLTR	RG
Comms	24	ODs	552	Iter	175	75	733
Nodes	24	Arcs	76	CPU	12m 17s	6m 23s	12m 14s
Cnstrs	576	Vars	1824	Obj	730121.42	721391.91	721391.91
LB	721387.97			Gap	8733.45	3.94	3.94

Hull Problem $f = \sum_j t_j a_j \left(1 + \frac{\alpha_j}{\beta_j + 1} \left(\frac{t_j}{b_j}\right)^{\beta_j}\right)$

Dimension				Algorithm	FWTR	PLTR	RG
Comms	16	ODs	142	Iter	100	23	3348
Nodes	423	Arcs	798	CPU	81m	23m	457m
Cnstrs	6768	Vars	12768	Obj	31205.213	31194.605	31194.645
LB	31194.581			Gap	10.632	0.024	0.064

Table 1 Comparison of FWTR, PLTR, and RG Algorithms on Microvax II RC

Problem A	**Parallel PLTR (21 Iters)**		
Machines	1	3	5
CPU	24.6s	15.3s	11.0s
Speedup	1	1.6	2.3
Efficiency	1	0.54	0.45

Problem B	**Parallel PLTR (37 Iters)**					
Machines	1	3	4	5	7	13
CPU	2m 13s	1m 14s	52.7s	42.8s	33.7s	24.3s
Speedup	1	1.8	2.5	3.1	4.0	5.5
Efficiency	1	0.60	0.63	0.62	0.57	0.42

Problem C	**Parallel PLTR (35 Iters)**	
Machines	1	6
CPU	1m 24.7s	24.8s
Speedup	1	3.4
Efficiency	1	0.57

Sioux Falls Problem	**Parallel PLTR (100 Iters)**						
Machines	1	3	4	5	7	9	13
CPU	27m 53s	14m 45s	10m 17s	8m 1s	5m 47s	4m 30s	3m 25s
Speedup	1	1.9	2.7	3.5	4.8	6.2	8.2
Efficiency	1	0.63	0.68	0.70	0.69	0.69	0.63

Hull Problem	**Parallel PLTR (44 Iters)**				
Machines	1	3	5	9	17
CPU	1h 59m	1h 2m	32m 33s	18m 39s	9m 21s
Speedup	1	1.9	3.7	6.4	12.7
Efficiency	1	0.64	0.73	0.71	0.74

Table 2 Performance on the Crystal Multicomputer (a local area network of Vax750s)

6. Conclusions

This paper establishes the convergence of a trust region method based on a dynamically-scaled piecewise-linear approximation of the objective function. The scaling feature allows the approximation to be adjusted from an underestimating to an overestimating approximation in order to enhance the likelihood of satisfying the criteria of the trust region approach.

With respect to the set of test problems considered, the convergence rate of this method (PLTR) is , as expected, superior to that of the corresponding linear trust region method (denoted here as FWTR). Since the time per iteration is comparable for these two approaches, PLTR enjoys a definite advantage with respect to total time. Because of the manner in which iterations are defined in the MINOS code, it is more difficult to make this type of comparison with the second-order reduced-gradient method (denoted here as RG) in MINOS. However, the time required to produce eight-figure objective function accuracy in PLTR is consistently less than in MINOS, and for the largest problem in the test set (6768 constraints and 12,768 variables), PLTR is roughly 20 times faster.

Another important property of PLTR is that it allows the traffic assignment problems considered here to be decomposed into single commodity problems, so that parallel computation may be utilized. Computational results with a corresponding block-Jacobi-type parallel algorithm on the CRYSTAL multicomputer show that good speedups are attained with such an approach. Research is continuing on parallel algorithms that are more closely related to block-Gauss-Seidel methods in the sense that they utilize information from the most recently updated blocks of variables (these are commodities in the traffic assignment context). This alternative approach to parallel computation will be described in [Chen and Meyer 86].

References

Beck, P., Lasdon, L., and Engquist, M. [1983]: "A reduced gradient algorithm for nonlinear network problems", *ACM Transactions on Mathematical Software, Vol. 9, No. 1, 57-70.*

Bertsekas, D. P. and Ganfi, E. M. [1982]: "Projection methods for variational inequalities with application to the traffic assignment problem", *Mathematical Programming Study 17, 139-159.*

Cantor, D. G. and Gerla, M. [1974]: "Optimal routing in packet switched computer networks", *IEEE Transactions on Computing C-23, 1062-1068.*

Chen, R. J. and Meyer, R. R. [1986]: "Parallel optimization for traffic assignment", to appear as University of Wisconsin-Madison Computer Sciences Department Tech. Rpt.

Collins, M., Cooper, L., Helgason, R. V., Kennington,J. L., and LeBlanc, L. J. [1978]: "Solving the pipe network analysis problem using optimization techniques", *Management Science, 24, 747-760.*

Dafermos, S. C. [1980]: "Traffic equilibrium and variational inequalities", *Transportation Science 14, 42-54.*

Dantzig, G., Harvey, R., Lansdowne, Z., Robinson, D., and Maier, S. [1979]: "Formulating and solving the network design problem by decomposition", *Transportation Res. 13B, 5-17.*

Dembo, R. S. and Klincewicz, J. G. [1981]: "A scaled reduced gradient algorithm for network flow problems with convex separable costs", *Mathematical Programming Studies, 15, 125-147.*

DeWitt, D., Finkel, R., and Solomon, M. [1984]: "The CRYSTAL multicomputer: design and implementation experience", University of Wisconsin-Madison Computer Sciences Department Tech. Rpt. #553.

Feijoo, B. [1985]: "Piecewise-linear approximation methods and parallel algorithms in optimization", University of Wisconsin-Madison Computer Sciences Department Tech. Rpt. #598.

Feijoo, B. and Meyer, R. R. [1984]: "Piecewise-linear approximation methods for nonseparable convex optimization", University of Wisconsin-Madison Computer Sciences Department Tech. Rpt. #521.

Feijoo, B. and Meyer, R. R. [1985]: "Optimization on the Crystal multicomputer", in *Computing 85*, G. Bucci and G. Valle eds., Elsevier Science Publishers.

Fletcher, R. [1981]: *Practical Methods of Optimization*, John Wiley.

Grigoriadis, M. D. and Hsu, T. [1979]: "RNET the Rutgers minimum cost network flow subroutine", *SIGMAP BULLETIN, 17-18.*

Hanscom, M. A., Lafond, L., Lasdon, L. and Pronovost, G. [1980]: "Modeling and resolution of the medium term energy planning problem for a large hydro-electric system", *Management Science, 26, 659-668.*

Kamesam, P. V. and Meyer, R. R. [1984]: "Multipoint methods for separable nonlinear networks", *Mathematical Programming Study 22, 185-205.*

Kao, C. Y. and Meyer, R. R. [1981]: "Secant approximation methods for convex optimization", *Mathematical Programming Study 14, 143-162.*

Klincewicz, J. G. [1983]: "A newton method for convex separable network flow problems", *Networks 13, 427-442.*

Lawphongpanich, S. and Hearn, D. W. [1983]: "Restricted simplicial decomposition with application to the traffic assignment problem", University of Florida Department of Industrial and System Engineering Research Rpt. 83-8.

LeBlanc, L. J., Morlok, E. K., and Pierskalla, W. P. [1975]: "An efficient approach to solving the road network equilibrium traffic assignment problem", *Transportation Res. 9, 309-318.*

Leventhal T. L., Nemhauser G. L., and Trotter Jr., L. E. [1973]: "A column generation algorithm for optimal traffic assignment", *Transportation Science 7, 168-176.*

Magnanti, T. L. and Wong, R. T. [1984]: "Network design and transportation planning: models and algorithm", *Transportation Science 18, 1-55.*

McCallum, C. J. [1976]: "A generalized upper bounding approach to communications network planning problem", *Networks 7, 1-23.*

Meyer, R. R. [1985]: "Trust region methods for piecewise-linear approximation", University of Wisconsin-Madison Computer Sciences Department Tech. Rpt. #626.

Monma, C. L. and Segal, M. [1982]: "A primal algorithm for finding minimum-cost flows in capacitated networks with applications", *Bell System Tech. J. 61, 949-968.*

More, J. J. and Sorensen, D. C. [1979]: "On the use of directions of negative curvature in a modified Newton method", *Mathematical Programming, 16, 1-20.*

Murtagh, B. A. and Saunders, M. A. [1978]: "Large-scale linearly constrained optimization", *Mathematical Programming, 14, 41-72.*

Nguyen, S. [1974]: "An algorithm for the traffic assignment problem", *Transportation Science 8, 203-216.*

Nguyen, S. and Dupuis, C. [1984]: "An efficient method for computing traffic equilibria in networks with asymmetric transportation", *Transportation Science 18, 185-202.*

Pang, J. S. and Yu, C. S. [1984]: "Linearized simplicial decomposition methods for computing traffic equilibria on networks", *Networks 14, 427-438.*

Rosenthal, R. E. [1981]: "A nonlinear network flow algorithm for maximization of benefits in a hydroelectric power system", *Operations Res. 29, 763-786.*

Sorensen, D. C. [1982]: "Newton method with a model trust region modification", *SIAM J. Numerical Analysis, 19, 409-426.*

Vardi, A. [1985]: "A trust region algorithm for equality constrained minimization: convergence properties and implementation" *SIAM J. Numerical Analysis, 22, 575-591.*

Zhang, J., Kim, N. H., and Lasdon, L. [1984]: "An improved successive linear programming algorithm", University of Texas Graduate School of Business Working paper 84/85-3-2, Austin.