MONOTROPIC PROGRAMMING: A GENERALIZATION OF LINEAR PROGRAMMING AND NETWORK PROGRAMMING

R.T. Rockafellar

Reprint of an article published in *Convexity and Duality in Optimzation*, J. Ponstein (ed.), Springer-Verlag Lecture Notes in Economics and Mathematical Systems No. 256, 1985, pp. 10–36.

MONOTROPIC PROGRAMMING: A GENERALIZATION OF LINEAR PROGRAMMING AND NETWORK PROGRAMMING

*R. Tyrrell Rockafellar**

Duality schemes have been developed for almost all types of optimization problems, but the classical scheme in linear programming has remained the most popular and indeed the only one that is widely familiar. No doubt this is due its simplicity and ease of application as much as its close connection with computation, which is a property shared with other dualities.

Linear programming duality has an appealing formulation in terms of a matrix or tableau in which each row or column corresponds to both a primal variable and a dual variable. A role assigned to one of these paired variables automatically entails a certain role for the other. In the usual way of thinking, decision variables in one problem correspond to slack variables in the other. The only flexibility is whether a decision variable is nonnegative or unrestricted, in accordance with which the corresponding slack variable is tied to an inequality or an equation.

A vast extension of this approach to duality is possible without sacrificing the sharpness of results or even the "concreteness" of representation. The extension is achieved by admitting far more general roles for the variables in a primal-dual pair. To appreciate how this is possible, it is necessary first to free oneself from the idea that there is something inherent in a variable being either a decision variable or a sort of slack variable. Such a distinction is actually an impediment even in standard linear programming, because it relates more to the initial tableau being used than to something inherent in the variables themselves. Whether a variable is "independent" or "dependent" in the expression of a problem and its constraints can change when pivoting transformations are applied to the tableau.

In the generalized duality scheme which we call ***monotropic programming*** the "role" of a variable is something apart from its incidental position in a tableau. It is given by specifying both an ***interval*** (the range of values permitted for the variable) and a ***cost expression*** defined over the interval. The cost expression is a convex function of the variable (possibly linear or piecewise linear). and it might be everywhere zero on the interval, in which case one ***could*** appropriately to think of the variable as a slack variable in a broadened sense.

The pair consisting of an interval and a convex cost expression on that interval can be identified with a so-called proper convex function on the real line. Subject to a minor regularity condition ("closedness", a property referring to endpoints), such a function can be dualized: the ***conjugate*** function furnishes the interval and associated cost expression defining the "role" of the dual variable.

Conjugates of convex functions of a single variable can readily be constructed by a process of generalized differentiation, inversion and reintegration. This feature gives monotropic programming duality a potential for much wider use in applications than other, more abstract forms of

* Supported in part by a grant from the National Science Foundation at the University of Washington, Seattle.

duality. Another interesting feature is the way that monotropic programming associates with each primal-dual pair of variables a certain "maximal monotone relation". The prototypes for such relations in ordinary linear programming are the complementary slackness relations used in characterizing optimality. In network programming, however, the relations have an importance all of their won. They describe the combinations of flow and tension (potential difference) that can occur in the various arcs of the network, more or less like generalizations of Ohm's Law, which in classical electrical theory corresponds to an arc representing an ideal resistor.

The theory of monotropic programming duality is presented in complete detail in the author's 1984 book [1]. Nevertheless a briefer introduction to the subject may be helpful, because the results and even the basic ideas are not yet widely known but very applicable. This is the justification for the present article.

In order not to encumber the exposition, proofs are omitted. Nothing is said about the history of the subject, the theories that are related to it, or the people who have made major contributions. All that can be found in [1].

1. Linear Systems of Variables.

A fundamental concept in monotropic programming is that of a "finite collection of real variables which are interrelated in a homogeneous linear manner". Let us denote the variables by x_j, $j \in J$, where J is some finite index set (such as $\{1,\ldots,n\}$ or $\{0,1,\ldots,n\}$ or $\{1,\ldots,n\} \times \{1,\ldots,m\}$; flexibility in this respect is helpful). We can think of vectors

$$x = (\ldots, x_j, \ldots) \in R^J$$

as corresponding to an assignment of a real number value to each variable. Such vectors can be added or multiplied by scalars as usual:

$$(\ldots, x_j, \ldots) + (\ldots, x'_j, \ldots) = (\ldots, x_j + x'_j, \ldots)$$
$$\lambda(\ldots, x_j, \ldots) = (\ldots, \lambda x_j, \ldots).$$

In this sense R^J is a vector space (identifiable with R^n when $J = \{1,\ldots,n\}$). To say that our variables x_j are related in a homogeneous linear manner is to say that the value vectors x we are interested in form a subset C of R^J with the property

$$x \in C, x' \in C \Longrightarrow x + x' \in C,$$
$$x \in C, \lambda \in R \Longrightarrow \lambda x \in C.$$

In other words, C is a (linear) subspace of R^J.

Formally speaking, then, a *linear system of variables* is simply the designation of a subspace C of R^J for some finite index set J. Often this subspace is described for us initially by a set of homogeneous linear equations

$$(1) \qquad \sum_{j\in J} e_{ij}x_j = 0 \text{ for } i \in I,$$

where I is some other index set, but this description is not unique and can be re-expressed in many different ways. (The restriction to homogeneous linear relations at this stage is merely a theoretical device, but an important one. General linear constraints will be handled below by specifying for each j a real interval C_j in which x_j must lie. Included in this is the possibility that some of these intervals consist of a single point, so that the corresponding x_j's must take on fixed values. Then (1) turns into a system of inhomogeneous linear equations in the remaining x_j's.)

A common way for a linear system of variables to be described is through a set of equations of the special form

$$(2) \qquad \sum_{l\in L} a_{kl}x_l = x_k \text{ for } k \in K,$$

where L and K are separate index sets, $J = K \cup L$. These could, of course, be written as

$$-x_k + \sum_{l\in L} a_{kl}x_l = 0 \quad \text{for} \quad k \in K$$

in order to fit the pattern of (1). Very useful in this connection is the ***tableau notation*** for (2) in Figure 1. When a linear system of variables is given in this way initially, the variables x_l for $l \in L$ may be thought of as "inputs" and the variables x_k for $k \in K$ as "outputs". From a mathematical point of view, however, such a distinction is of little importance.

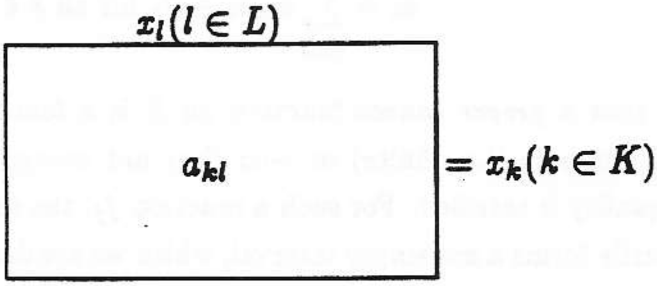


Figure 1.

In fact ***any*** linear system of variables can be represented as in (2) in a ***multiplicity of ways***. We speak of these as ***Tucker representations*** of the system or of the subspace $C \subset R^J$. Specifically, one can pass from (1) to (2) by solving for a maximal set of x_j's (corresponding to some index set $K \subset J$) in terms of the remaining x_j's (corresponding to the complementary index set $L = J \backslash K$).

Thus there is a one-to-one correspondence between Tucker representations of the given C and certain partitions of J into a pair of subsets, K and L. Obviously there are many (but only *finitely* many) such representations, and they all yield tableaus of the same size: the number of elements of L must equal the dimeusion of C in R^J.

It is possible to pass from any one Tucker representation to any other by a series of *pivoting transformations* of the tableau, each such transformation involving an exchange of an index k_0 in K with an index l_0 in L. This is a central idea in computational procedures in monotropic programming.

2. Monotropic Programming Problems.

We suppose now that we are given a linear system of variables, designated by a certain subspace $C \subset R^J$, and also for each index $j \in J$ a *closed proper convex* function f_j on R. We denote by C_j the (nonempty) interval of R where f_j has finite values (the set dom f_j in convex analysis); more about this in a moment. The corresponding *monotropic programming* problem that we shall be concerned with is

$$\text{(P)} \qquad \begin{aligned} &\text{minimize } \sum_{j\in J} f_j(x_j) =: F(x) \text{ over all} \\ &x = (\ldots, x_j, \ldots) \in C \text{ satisfying } x_j \in C_j \text{ for all } j \in J. \end{aligned}$$

Note that in terms of any Tucker representation of C this takes the form

$$\text{(P}'\text{)} \qquad \begin{aligned} &\text{minimize } \sum_{k\in K} f_k(x_k) + \sum_{l\in L} f_l(x_l) \\ &\text{subject to } x_l \in C_l \text{ for all } l \in L, \\ &\qquad x_k = \sum_{l\in L} a_{kl} x_l \in C_k \text{ for all } k \in K. \end{aligned}$$

Here we recall that a *proper convex* function on R is a function defined on *all* of R with values that are real numbers (i.e. finite) or $+\infty$ (but not *everywhere* $+\infty$) and such that the usual convexity inequality is satisfied. For such a function f_j, the set of points where the value of f_j is not $+\infty$ necessarily forms a nonempty interval, which we are denoting here by C_j. *Closedness* of f_j is a mild semicontinuity condition on the behavior of f_j at the endpoints of C_j (to the extent that these are finite): the value of f_j at such an endpoint must coincide with the limiting value obtained as the endpoint is approached from within C_j. (This allows for the possibility that $f_j(x_j) \to +\infty$ as x_j approaches a finite endpoint of C_j from within C_j; thus the closedness of f_j does not require the closedness of C_j.)

For readers unaccustomed to dealing with $+\infty$, it is essential to realize that the introduction of $+\infty$ is merely a notational device for the representation of constraints which happens to be very useful in theory, particularly in understanding duality. Every pair C_j, f_j, consisting of a nonempty real interval C_j and an arbitrary finite-valued convex function f_j on C_j in the traditional sense, can be identified uniquely and unambiguously with a certain proper convex function on R: simply regard $f_j(x_j)$ as $+\infty$ for every $x_j \in C_j$.

An x that satisfies $x \in \mathcal{C}$ and the interval constraints $x_j \in C_j$ in (P) (or the corresponding conditions in (P′)) is said to be a *feasible solution* to our problem, of course. The feasible solutions form a convex subset of R^J on which the objective F is a finite convex function. (Observe that $F(x) < \infty$ if and only if $f_j(x_j) < \infty$ for all j, or in other words, $x_j \in C_j$ for all j. Thus the interval constraints in (P) would be implicit in the minimization even if we had not listed them, which we did for emphasis.)

As represented in the form (P′), our problem could be viewed as one in terms of the variables x_l alone: a certain convex function on R^L which is *preseparable* (expressible as a sum of linear functions of the variables x_l composed with convex functions) is to be minimized subject to a system of linear constraints. To adopt this view strongly, however, would be to miss one of the main features of monotropic programming. Here we are referring to the fact that the representation (P′) is in no way unique, and by passing between various such representations in terms of pivoting we hope to gain computational advantage and insight into the underlying problem (P).

3. Categories of Monotropic Programming.

Many cases of problem (P) are of interest and serve to illuminate the scope of monotropic programming. To look at a "degenerate" example first, suppose that every function f_j is just the *indicator* of a closed interval C_j:

$$f_j(x_j) = \delta(x_j|C_j) = \begin{cases} 0 & \text{when } x_j \in C_j, \\ \infty & \text{when } x_j \notin C_j. \end{cases} \tag{3}$$

Then we have a ***pure feasibility problem***; the objective $F(x)$ has value 0 for all feasible solutions x, and the problem reduces simply to finding such a feasible x, any one at all.

More generally, certain of the function f_j, but not all, may be indicators as in (3). These functions then serve only to represent certain constraints $x_j \in C_j$. They make no contribution to the "cost" $F(x)$ of a feasible solution x.

These ideas can be made clearer by considering how ***linear programming problems*** of the standard sort fit the model of monotropic programming. In the case of such problems we imagine the linear system of variables to be given initially in terms of relations of type (2), so that (P′) is the form to aim at. Suppose the linear programming problem is

$$\text{minimize } \sum_{l \in L} c_l x_l \text{ subject to } x_l \geq 0 \text{ for all } l \in L \text{ and}$$

$$(4)\qquad \sum_{l\in L} a_{kl}x_l \begin{cases} \geq b_k & \text{for } k \in K_+, \\ = b_k & \text{for } k \in K_0, \\ \leq b_k & \text{for } k \in K_- \end{cases}$$

where L, K_+, K_0 and K_- are separate index sets. We can identify this problem with (P′) in the case of $K = K_+ \cup K_0 \cup K_-$ and

$$(5)\qquad f_l(x_l) = c_l x_l + \delta(x_l|C_l), \quad f_k(x_k) = \delta(x_k|C_k),$$

where $C_l = [0, \infty)$ for all $l \in L$ and

$$(6)\qquad C_k = \begin{cases} [b_k, \infty) & \text{for } k \in K_+, \\ [b_k, b_k] & \text{for } k \in K_0, \\ (-\infty, b_k] & \text{for } k \in K_-. \end{cases}$$

Here it would be easy to introduce upper bounds on the variables x_l: take C_l to be an interval of the form $[0, d_l]$. Another extension would be to allow constraints like $\sum_{l\in L} a_{kl}x_l \leq b_k$ to be violated, but at a penalty. If the penalty is linear, this would correspond to taking f_k as in Figure 2, and analogously for indices k in K_0 or K_+. Then the objective F would no longer be linear, but piecewise linear.

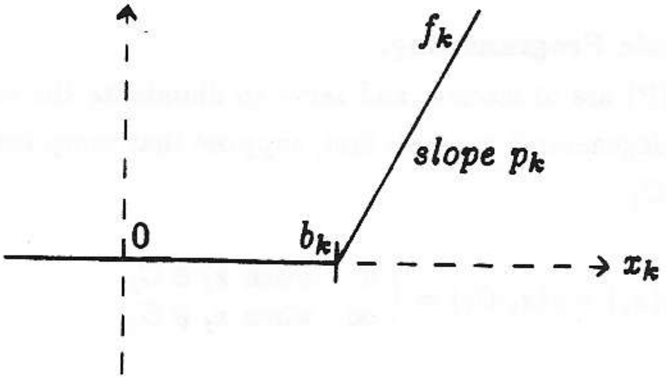


Figure 2.

In general we can identify ***piecewise linear programming*** as the branch of monotropic programming where each of the functions f_j is piecewise linear relative to C_j with finitely many pieces (i.e. f_j is a polyhedral convex function on R in the terminology of convex analysis). Similarly, if each f_j is piecewise quadratic we have ***piecewise quadratic programming,*** which can be shown in particular to encompass all of convex quadratic programming, and which also allows for constraint penalties as in Figure 2 but with quadratic portions.

A noteworthy feature of monotropic programming that will be viewed below is the duality which is possible in these categories of problems. For instance, the dual of a piecewise linear problem will be another piecewise linear problem which can readily be constructed. The duality theory of linear programming itself is, in contrast, relatively cumbersome and limited. The dual of a *linear programming problem in the general sense*, i.e., the case of (P) or (P′) where each f_j is linear (we should really say "affine") relative to C_j, can be obtained in the traditional framework only by reducing first to a standard type of linear programming problem through modification of the constraints by the introduction of auxiliary variables. This is a drawback in particular for linear programming problems with upper bounds, and the case of linear constraint penalties is even worse. In the context of monotropic programming, the dual of a linear programming problem in the general sense will be piecewise linear rather than linear, but it can be generated directly.

4. Network Programming as a Special Case.

A network, or directed graph, as shown in Figure 3, is defined mathematically in terms of finite sets I and J, comprised of the *nodes* i and *arcs* j of the network, and the *incidence matrix*

$$(7) \qquad e_{ij} = \begin{cases} 1 \text{ if } i \text{ is the initial node of the arc } j, \\ -1 \text{ if } i \text{ is the terminal node of the arc } j, \\ 0 \text{ if } i \text{ is not a node of } j. \end{cases}$$

(Arcs that start and end at the same node are excluded.)

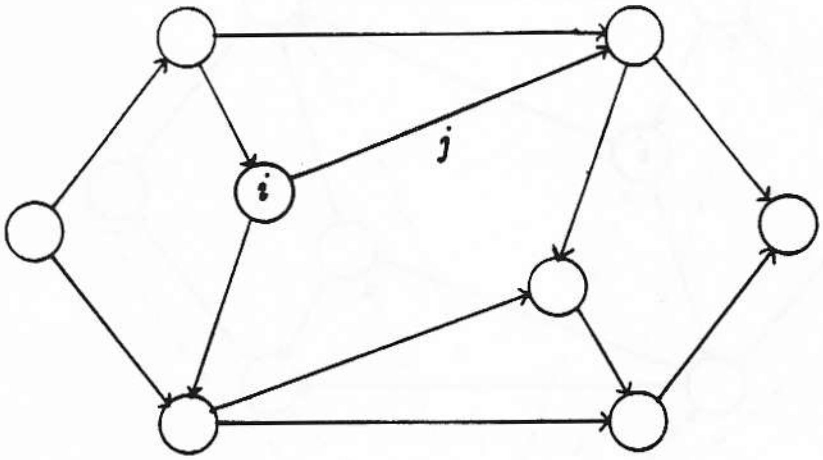


Figure 3.

For each arc $j \in J$, let x_j denote the amount of *flow* in j (a positive quantity being interpreted as material moving in the direction of the arrow that represents j). The linear system we are interested in consists of the variables x_j, $j \in J$, as related by the conditions

$$\sum_{j \in J} e_{ij} x_j = 0 \text{ for all } i \in I.$$

These conditions are Kirchoff's node laws. They say that at each node i, what enters equals what leaves, or in other words, flow is conserved. The vectors $x = (\ldots, x_j, \ldots) \in R^J$ satisfying these conditions are called *circulations*, and the subspace C that they form is the *circulation space* for the network.

The possible Tucker representations (2) in this case correspond one-to-one with the arc sets K that are *spanning trees* for the network. Pivoting from one such representation to another can be carried out "combinatorially" through the manipulation of such trees and their associated cuts and circuits, instead of numerical operations on the coefficients in the tableau.

What then is the interpretation of the monotropic programming problem (P)? For each arc j the flow x_j is restricted to a certain interval C_j and assessed at a certain cost $f_j(x_j)$ (possibly zero). Subject to these interval constraints, one seeks a circulation x that minimizes total cost.

The choice of the intervals C_j in such a problem can reflect restrictions on the direction of flow as well as its magnitude in the arc j. Thus for $C_j = [0, \infty)$ we simply have the condition that the flow in j must be from the initial node to the terminal node, whereas for $C_j = [0, c_j]$ the flow must in addition be bounded in magnitude by c_j. Similarly for $C_j = (-\infty, \infty)$ and $C_j = [-c_j, c_j]$: in the first instance there is no restriction on x_j whatsoever, whereas in the second the direction is unrestricted but $|x_j| \leq c_j$. The case of a single point interval $C_j = [c_j, c_j]$ corresponds to a preassigned value for the flow in the arc j namely, $x_j = c_j$.

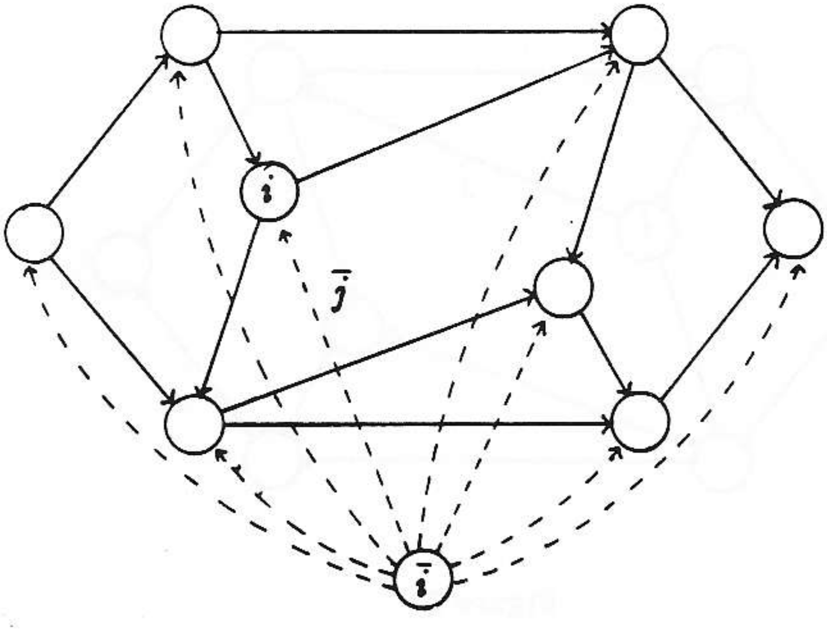


Figure 4.

The relationship between this version of network programming and other, more traditional modes is clarified in Figure 4. Problems for the network in Figure 3 that might ordinarily be conceived in terms of flows that are *not* necessarily conserved at every node are represented in terms of circulations in the augmented network which has a distribution node $\bar{i}$ (a "ground" node in electrical theory). The flow $x_{\bar{j}}$ in the arc $\bar{j}$ in Figure 4 that connects this node $\bar{i}$ to one of the other nodes i corresponds in Figure 3 to an amount of material entering the network at i (positive or negative). Thus a requirement $x_{\bar{j}} = [c_{\bar{j}}, c_{\bar{j}}]$ in this case could be interpreted as specifying the amount entering the network in Figure 3 at i. (If $c_{\bar{j}} > 0$, i would be a supply point, whereas if $c_{\bar{j}} < 0$, i would be a demand point; if $c_{\bar{j}} = 0$, i would be neither.) More general conditions $x_{\bar{j}} \in C_{\bar{j}}$ in Figure 4 could be interpreted as allowing for a certain range of supply or demand at i in Figure 3.

From these considerations it is evident that monotropic programming problems for flows in networks are *generalized transportation problems* with possibly nonlinear costs. In the pure feasibility case they are generalized distribution problems connected with the satisfaction of various requirements of capacity, supply and demand.

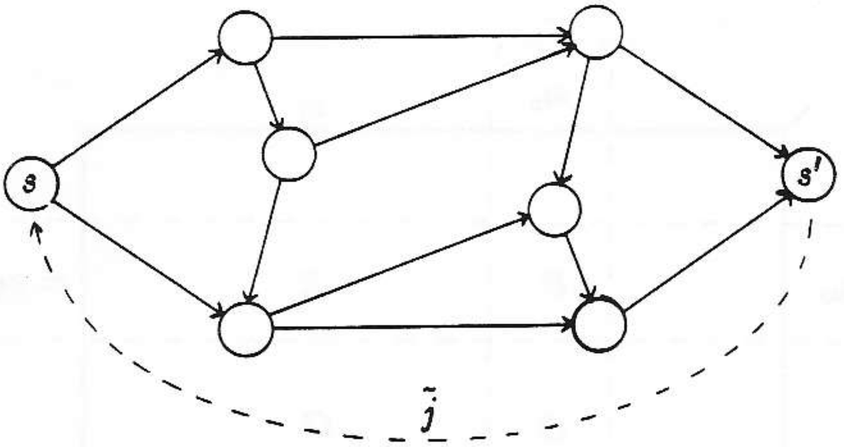


Figure 5.

Even the max flow problem and its generalizations fit this mold. Figure 5 indicates the modified network that would correspond to maximizing in the network of Figure 3 the flow from a certain node s to another node s' subject to capacity conditions $x_j \in C_j$. Over all feasible *circulations* in the modified network, we seek to maximize the flow in the "feedback" arc $\tilde{j}$ (or minimize its negative). This is the monotropic programming problem that corresponds to taking

$$\begin{aligned} &f_{\tilde{j}}(x_{\tilde{j}}) = -x_{\tilde{j}} \quad (C_{\tilde{j}} = (-\infty, \infty)), \\ &f_j(x_j) = \delta(x_j | C_j) \ \text{ for all other arcs } j. \end{aligned} \tag{8}$$

Other important classes of monotropic programming problems for flows in networks involve linear systems of variables more general than the circulation system so far described. For example, there are problems for *network with gains*, where the amount flowing in the arc j can be amplified or attenuated by a certain factor. (The incidences in (7) are replaced by more general numbers.) Such problems too lend themselves to combinatorial rules of pivoting in the manipulation of

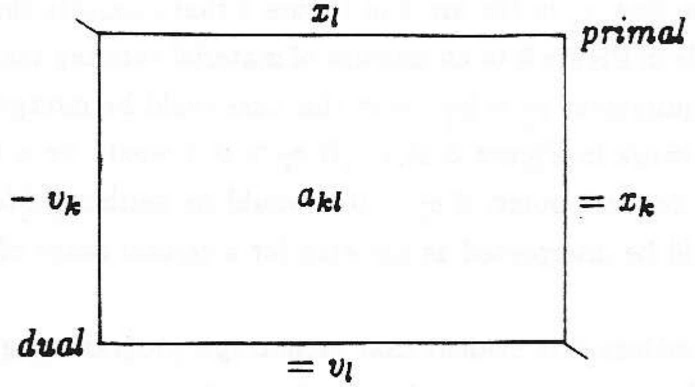


Figure 6.

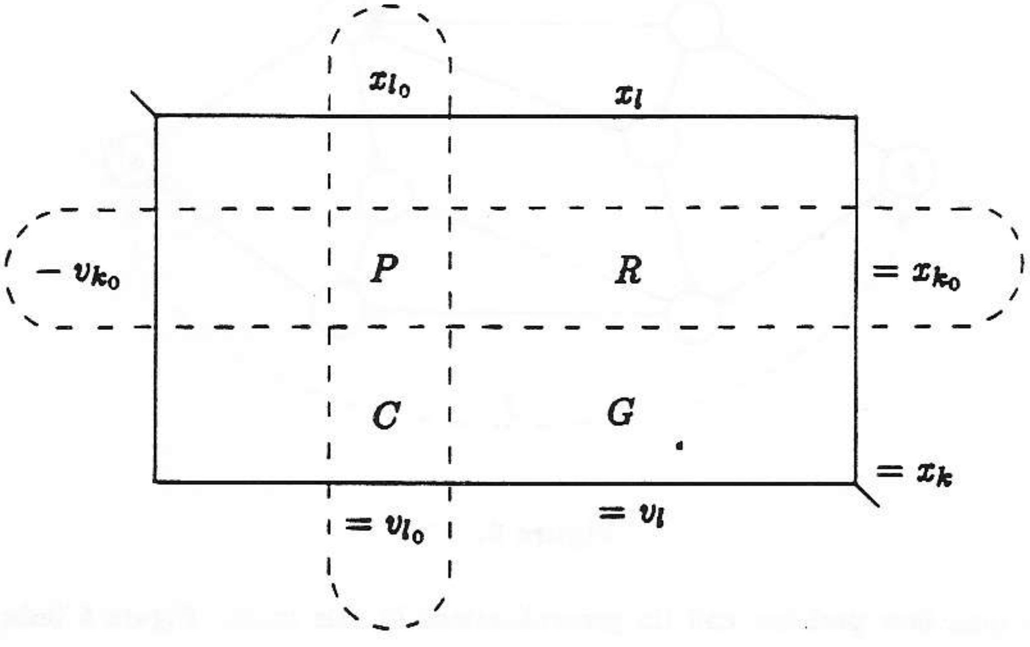


Figure 7.

The numerical formulas for the transformed coefficients are

$$P' = 1/P, \quad C' = C/P, \quad R' = -R/P, \quad G' = G - CR/P. \tag{12}$$

One must always remember, however, that in important cases such as occur in network programming such formulas for the coefficients can be by-passed, because it is possible to store the Tucker representations combinatorially in terms of spanning trees and generate particular coefficients a_{kl} from this as needed. In other cases, for instance in traffic problems, a decomposition is possible in which pivoting is carried out partly by numerical formula and partly by combinatorial techniques.

Each variable x_j in the primal linear system is paired with a variable v_j in the dual linear system, and in applications this pairing usually has a natural significance. The case of network

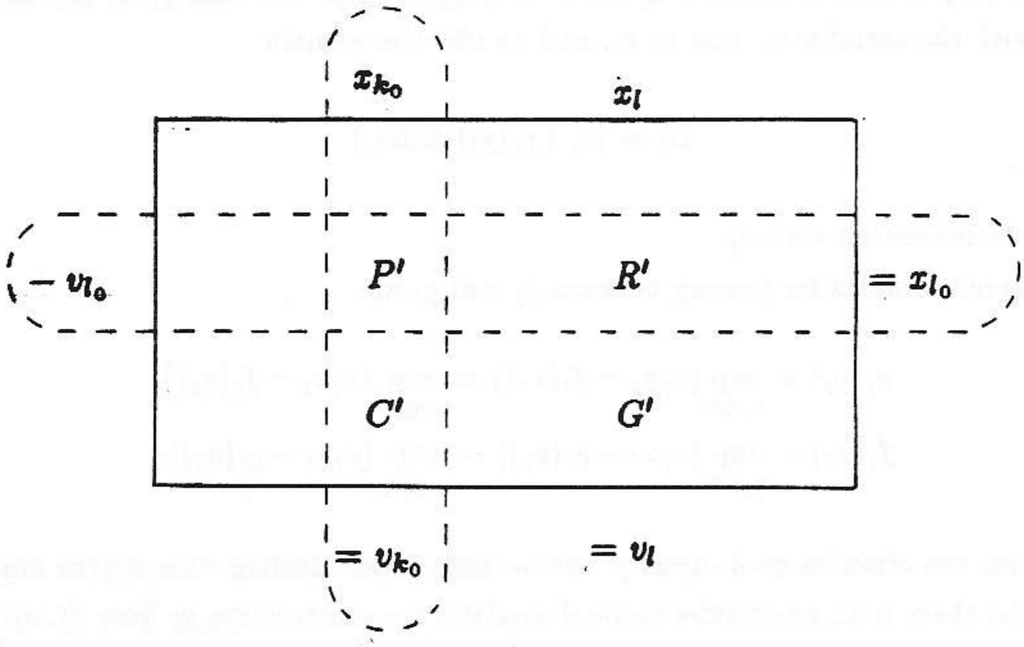


Figure 8.

programming furnishes a memorable illustration. In that case the space C consists of the flow vectors $x \in R^J$ that satisfy the homogeneous equations (1) for the incidence matrix (7). In geometric terms we therefore can view C as the space of vectors orthogonal to the rows of the incidence matrix (one row for each node $i \in I$), and it follows then by elementary linear algebra that the complementary space $D = C^{\perp}$ is the subspace spanned by these rows. Thus (taking $-u_i$ as the notation for the coefficient of the i^{th} row in a general linear combination of the rows, for reasons apparent in a moment) we see that D consists of the vectors $v = (\dots, v_j, \dots)$ expressible in the form

$$v_j = -\sum_{i \in I} u_i e_{ij} \text{ for } j \in J \tag{13}$$

by some choice of numbers u_i. Referring to the definition of e_{ij}, we see further that (13) reduces to $v_j = u_{i_2} - u_{i_1}$, where i_1 is the initial node of the arc j and i_2 the terminal node.

The interpretation is this. The vector $u = (\dots, u_i, \dots) \in R^I$ is a *potential* on the nodes of the network, and $v = (\dots, v_j, \dots)$ is the corresponding vector of potential differences or *tensions*, the *differential* of u. Thus D is the tension or *differential space* of the network.

6. Conjugate Costs and Monotone Relations.

The notion of a linear system of variables has been dualized, but that is not the only ingredient in a monotropic programming problem. We must also dualize the data embodied in the specification of a closed proper convex function f_j on R for each of the variables x_j, which includes the specification of the interval C_j associated with x_j (C_j being the effective domain of f_j). The machinery for this is already well developed in convex analysis. The natural dual of f_j is the closed proper convex function g_j on R *conjugate* to f_j. We take g_j as the cost function associated with the variable v_j dual to x_j, and its effective domain

$$D_j = \{v_j \mid g_j(v_j) \text{ finite } \}$$

as the interval associated with v_j.

The general formulas for passing between f_j and g_j are

$$\begin{aligned} g_j(v_j) &= \sup_{x_j \in R} \{v_j x_j - f_j(x_j)\} = \sup_{x_j \in C_j} \{v_j x_j - f_j(x_j)\}, \\ f_j(x_j) &= \sup_{v_j \in R} \{v_j x_j - g_j(v_j)\} = \sup_{v_j \in D_j} \{v_j x_j - g_j(v_j)\}. \end{aligned} \tag{14}$$

These formulas can often be used directly, but because we are dealing with convex functions of a single variable, there is an alternative method available for constructing g_j from f_j, or vice versa. This method, which involves inverting a generalized derivative relation and integrating, is often very effective and easy to carry out, and it yields other insights as well.

For every value of x_j the set

$$\partial f_j(x_j) = \{v_j \in R \mid f_j(x_j + t) \geq f_j(x_j) + v_j t \text{ for all } t \in R\} \tag{15}$$

consists of the "subgradients" of f_j at x_j in the general terminology of convex analysis, but in the one-dimensional case we are involved with here it is more appropriate to think of a "range of slopes" of f_j at x_j. This set is always a closed interval: in terms of the right derivative $f'_{j+}(x_j)$ and the left derivative $f'_{j-}(x_j)$ one has

$$\partial f_j(x_j) = \{v_j \in R \mid f'_{j-}(x_j) \leq v_j \leq f'_{j+}(x_j)\}. \tag{16}$$

(For this to make sense even when $x_j \notin C_j$, the convention is adopted that $f'_{j-}(x_j)$ and $f'_{j+}(x_j)$ are both ∞ when x_j lies to the right of C_j but both $-\infty$ when x_j lies to the left of C_j; then f'_{j+} and f'_{j-} are nondecreasing functions on R.)

Technically one must view ∂f_j as a "multifunction" rather than a function, because $\partial f_j(x_j)$ can be empty or have more than one element. There is a remarkable function-like character, however, which becomes clear upon inspection of the graph set

$$\Gamma_j = \text{gph } \partial f_j = \{(x_j, v_j) \in R \times R \mid v_j \in \partial f_j(x_j)\}, \tag{17}$$

as illustrated in Figures 9 and 10.

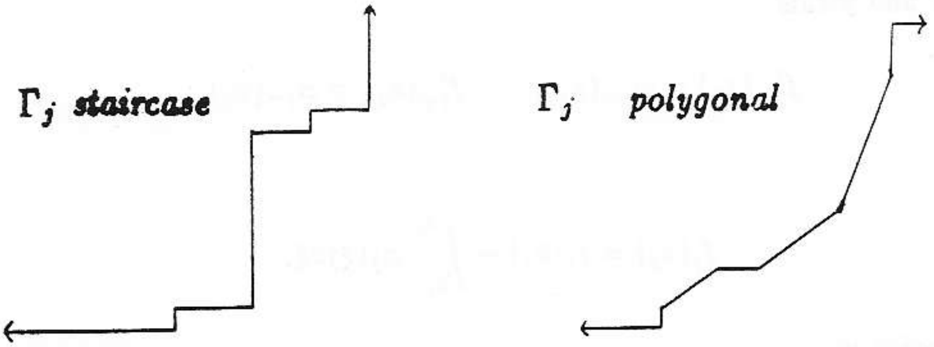


Figure 11.

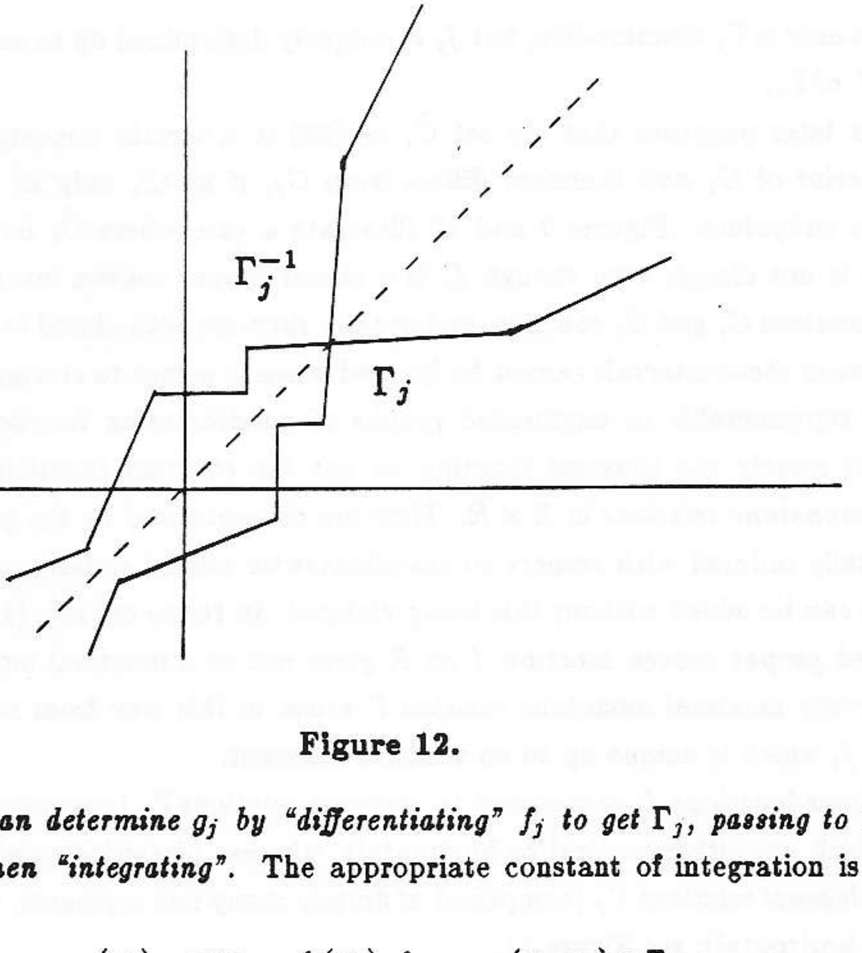


Figure 12.

Therefore one can determine g_j by "differentiating" f_j to get Γ_j, passing to the "inverse" relation Γ_j^{-1}, and then "integrating". The appropriate constant of integration is fixed by the equation

$$g_j(\overline{v}_j) = \overline{v}_j \overline{x}_j - f_j(\overline{x}_j) \text{ for any } (\overline{x}_j, \overline{v}_j) \in \Gamma_j,$$

which is a consequence of (14) and (15). Related to the finiteness interval D_j associated with g_j, as $\tilde{C}_j$ was to C_j, is the interval

$$\begin{aligned} \tilde{D}_j = &\text{ projection of } \Gamma_j \text{ on the vertical axis} \\ = &\{v_j \in R \mid \partial g_j(v_j) \neq \emptyset\} \subset D_j. \end{aligned} \tag{24}$$

This always includes the interior of D_j.

has this property and yields

$$f'_{j-}(x_j) = \varphi_{j-}(x_j), \qquad f'_{j+}(x_j) = \varphi_{j+}(x_j). \tag{20}$$

Moreover

$$f_j(x_j) = f_j(\overline{x}_j) + \int_{\overline{x}_j}^{x_j} \varphi_j(\xi)d\xi, \tag{21}$$

where $\overline{x}_j$ is any point in

$$\begin{aligned} \tilde{C}_j &= \text{projection of } \Gamma_j \text{ on the horizontal axis} \\ &= \{x_j |\ \partial f_j(x_j) \neq \emptyset\} \subset C_j. \end{aligned} \tag{22}$$

In this sense, not only is Γ_j function-like, but f_j is uniquely determined up to an additive constant as the "integral" of Γ_j.

We note for later purposes that the set $\tilde{C}_j$ in (22) is a certain nonempty *interval* which includes the interior of C_j and therefore differs from C_j, if at all, only in a possible lack of one or the other endpoints. Figures 9 and 10 illustrate a case where $\tilde{C}_j$ does differ from C_j, and actually C_j is not closed, even though f_j is a closed proper convex function. In the great majority of applications C_j and $\tilde{C}_j$ coincide, and usually they are both closed too, but the possible discrepancy between these intervals cannot be ignored when it comes to stating theorems, cf. §8.

The sets Γ representable as augmented graphs of nondecreasing functions φ on R as in (16) (with φ not merely the constant function ∞ nor the constant function $-\infty$) are the so called *maximal monotone relations* in $R \times R$. They are characterized by the property that their elements are totally ordered with respect to coordinatewise partial ordering of $R \times R$, and no further elements can be added without this being violated. In terms of (16), (17), (18), and (19), then, every closed proper convex function f on R gives rise to a maximal monotone relation Γ in $R \times R$, and every maximal monotone relation Γ arises in this way from some closed proper convex function f, which is unique up to an additive constant.

Piecewise linear functions f_j correspond to *staircase* relations Γ_j (comprised of finitely many line segments which are either vertical or horizontal), whereas piecewise quadratic functions f_j correspond to *polygonal* relations Γ_j (comprised of finitely many line segments, which do not have to be vertical or horizontal); see Figure 11.

The crucial fact for the purpose of constructing the conjugate g_j of f_j is that

$$v_j \in \partial f_j(x_j) \iff x_j \in \partial g_j(v_j),$$

or in other words

$$\text{gph } \partial g_j = \Gamma_j^{-1} = \{(v_j, x_j) |\ (x_j, v_j) \in \Gamma_j\}; \tag{23}$$

see Figure 12.

It is obvious from this method of constructing conjugates that g_j is piecewise linear when f_j is piecewise linear, and g_j is piecewise quadratic when f_j is piecewise quadratic. Indeed, Γ_j^{-1} is "staircase" when Γ_j is "staircase", and Γ_j^{-1} is "polygonal" when Γ_j is "polygonal". Furthermore the construction is quite easy to carry out in such cases. The conclusion to be drawn in the context of the next section will be that the dual of any piecewise linear or piecewise quadratic problem in monotropic programming can readily be written down.

7. Dual Problem and Equilibrium Problem.

In terms of a linear system of variables corresponding to a subspace $\mathcal{C}$ of R^J and a closed proper convex function f_j assigned to each $j \in J$ we have already introduced the ***primal monotropic programming problem***

$$\text{(P)} \qquad \begin{array}{l} \text{minimize} \ \displaystyle\sum_{j\in J} f_j(x_j) =: F(x) \ \text{over all} \\ \qquad x = (\ldots, x_j, \ldots) \in \mathcal{C} \ \text{satisfying} \ x_j \in C_j \ \text{for all} \ j \in J. \end{array}$$

We now introduce the ***dual monotropic programming problem***

$$\text{(D)} \qquad \begin{array}{l} \text{maximize} \ \ -\displaystyle\sum_{j\in J} g_j(v_j) =: G(v) \ \text{over all} \\ \qquad v = (\ldots, v_j, \ldots) \in \mathcal{D} \ \text{satisfying} \ v_j \in D_j \ \text{for all} \ j \in J \end{array}$$

and the ***monotropic equilibrium problem***

$$\text{(E)} \qquad \begin{array}{l} \text{find} \ x = (\ldots, x_j, \ldots) \in \mathcal{C} \ \text{and} \ v = (\ldots, v_j, \ldots) \in \mathcal{D} \\ \qquad \text{such that} \ (x_j, v_j) \in \Gamma_j \ \text{for all} \ j \in J. \end{array}$$

Here $\mathcal{D} \subset R^J$ gives the dual linear system as in §5, g_j is the cost function conjugate to f_j (with D_j its interval of finiteness), and Γ_j is the corresponding maximal monotone relation in $R \times R$ as in §6.

It is important to realize that because any member of the triple of elements f_j, g_j, Γ_j, can be determined from any other, it is also true that any one of the problems (P), (D), and (E) generates the others. (The choice of constants of integration in passing from (E) to (P) and (D) makes no real difference, since it only affects the objective functions by a constant.) Applications do occur where (E) is paramount, as will be explained at the end of this section. The general connection between the problems is that (E) focuses on joint optimality conditions for the solutions to (P) and (D), or when viewed in the other direction, that (P) and (D) furnish variational principles for the solutions to (E).

Parallel to the expression of (P) in terms of a Tucker representation for the primal linear system as in (P′) one has the expression of (D) as

$$(\mathrm{D}') \qquad \begin{aligned} &\text{maximize} \quad -\sum_{l\in L} g_l(v_l) - \sum_{k\in K} g_k(v_k) \\ &\text{subject to} \quad v_k \in D_k \text{ for all } k \in K, \\ &\qquad\qquad -\sum_{k\in K} v_k a_{kl} = v_l \in D_l \text{ for all } l \in L. \end{aligned}$$

Both problems can be viewed along with (E) in terms of a joint Tucker tableau for the primal and dual systems as in Figure 6. Obviously the general mathematical nature of the dual problem is the same as that of the primal problem, except for a change of sign in the objective (so that a concave function is maximized instead of a convex function minimized). The dual of the dual is the primal again; complete symmetry reigns.

It is hardly possible within the confines of this article to do more than hint at the wide range of situations encompassed by this paradigm. Some general observations that can be made on the basis of the preceding discussion are these. The dual of a piecewise linear problem is piecewise linear, and the dual of a piecewise quadratic problem is piecewise quadratic. In network programming, the dual of a problem involving flows is a problem involving potentials, and conversely.

A more specific illustration is the dual of the basic linear programming problem (4) in the case of upper bounds on the variables x_l, where

$$(25) \qquad \begin{aligned} f_l(x_l) &= c_l x_l + \delta(x_l|[0,d_l]) \quad \text{for } l \in L, \\ f_k(x_k) &= \begin{cases} \delta(x_k, [b_k, \infty)) & \text{for } k \in K_+, \\ \delta(x_k, [b_k, b_k]) & \text{for } k \in K_0. \end{cases} \end{aligned}$$

(For brevity we omit the index set K_- at this time.) Simple calculations on the basis of (14) reveal that

$$(26) \qquad \begin{aligned} g_l(v_l) &= \begin{cases} 0 & \text{if } v_l \le c_l, \\ d_l(v_l - c_l) & \text{if } v_l > c_l, \end{cases} \\ g_k(v_k) &= \begin{cases} b_k v_k + \delta(v_k|(-\infty, 0]) & \text{for } k \in K_+, \\ b_k v_k & \text{for } k \in K_0. \end{cases} \end{aligned}$$

Thus in the corresponding dual (in form (D')) we maximize $-\sum\limits_{k\in K} b_k v_k$ subject to $-v_k \ge 0$, $-\sum_{k\in K} v_k a_{kl} = v_l \le c_l$, except that the constraint $v_l \le c_l$ can be violated to the tune of a *linear penalty* with cost coefficient d_l. Ordinary linear programming duality is included here as a special case: it corresponds to an "infinite penalty" for constraint violation. A change of variables $w_k = -v_k$ would reduce notation to the customary pattern of signs. Incidentally, the introduction of a linear penalty for violation of the constraints $x_k \ge b_k$ in (P) would correspond dually to the introduction of an upper bound on the variable $w_k = -v_k$.

The nature of the relations Γ_l and Γ_k in this linear programming example is instructive too. These relations are displayed in Figure 13.

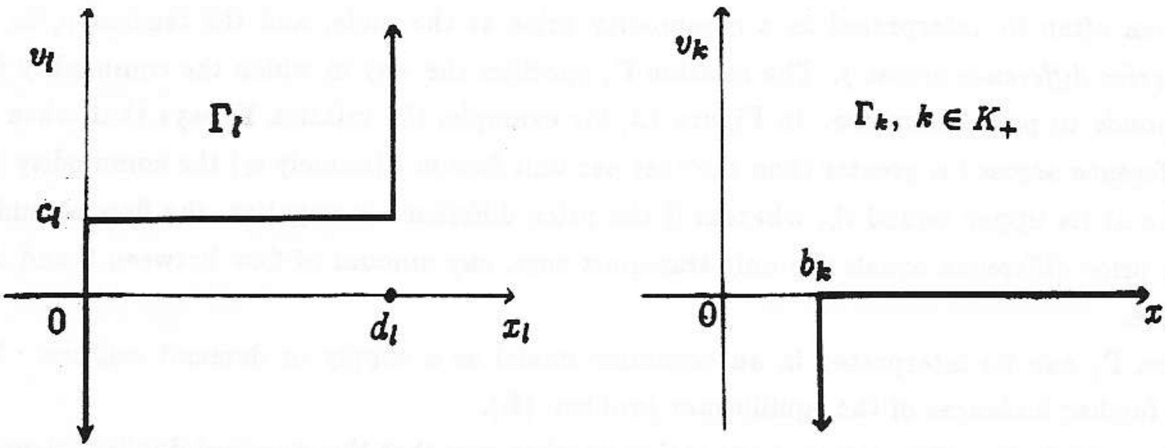


Figure 13.

For $k \in K_0$, Γ_k is simply the vertical line through the point b_k on the x_k-axis. Then the condition (x_k, v_k) reduces to the equation $x_k = b_k$ with no restriction placed on v_k, but the relations in Figure 13 express *complementary slackness* of various sorts. Thus for $k \in K_+$, for instance, the pairs $(x_k, v_k) \in \Gamma_k$ are the ones that satisfy

$$x_k \geq b_k, \quad -v_k \geq 0, \quad (x_k - b_k)(-v_k) = 0.$$

In network programming, on the other hand, the relations Γ_j can take on quite a different sort of meaning. In fact context x_j is the flow in the arc j, and v_j is the tension (potential difference). The condition $(x_j, v_j) \in \Gamma_j$ in the equilibrium problem (E) says that for the arc j only certain combinations of flow and tension are admissible. One is reminded of classical electrical networks and Ohm's Law, where $x_j = r_j v_j$ for a certain coefficient $r_j > 0$, the *resistance* of the arc j. Then Γ_j is a line in $R \times R$ with slope r_j.

Other classical electrical examples are those where the arc j represents an ideal *battery* (Γ_j a horizontal line through the point c_j on the v_j-axis, c_j being the potential difference across the terminals of the battery regardless of the current passing through it), an ideal *generator* (Γ_j a vertical line through the point b_j on the x_j-axis, b_j being the fixed current supplied by the generator regardless of the potential difference across its terminals), or an ideal *diode* (Γ_j the union of the nonnegative x_j-axis with the nonpositive v_j-axis). More general characteristic curves Γ_j can be obtained by imagining arcs j that represent nonlinear resistors or two-terminal "black boxes" with internal networks made up of various components such as have already been mentioned.

A fundamental problem in electrical networks, expressed in a mathematically polished, modern form, is the following. Given for each arc j of the network a "characteristic curve" Γ_j which is a *maximal monotone relation* in $R \times R$, find a circulation vector x and a differential vector v such that for every j the pair (x_j, v_j) lies on Γ_j. This is problem (E) for the linear systems of variables associated with the network.

Equilibrium problems in hydraulic networks arise similarly, and they also appear in the analysis of traffic. In economic networks (and general transportation problems) the potential at a node can often be interpreted as a commodity price at the node, and the tension v_j in the arc j is *price difference* across j. The relation Γ_j specifies the way in which the commodity flow in j responds to price difference. In Figure 13, for example, the relation Γ_l says that when the price difference across l is greater than the cost per unit flow in l (namely c_l) the commodity flow should be at its upper bound d_l, whereas if the price difference is negative, the flow should be 0. If the price difference equals the unit transport cost, any amount of flow between 0 and d_l is acceptable.

Often Γ_j can be interpreted in an economic model as a supply or demand relation. This leads to further instances of the equilibrium problem (E).

Before concluding this section, we mention another way that the standard duality scheme in linear programming can be viewed as a special case of monotropic programming duality. For this we imagine a Tucker tableau as in Figure 14 with two distinguished indices $\overline{k}$ and $\overline{l}$.

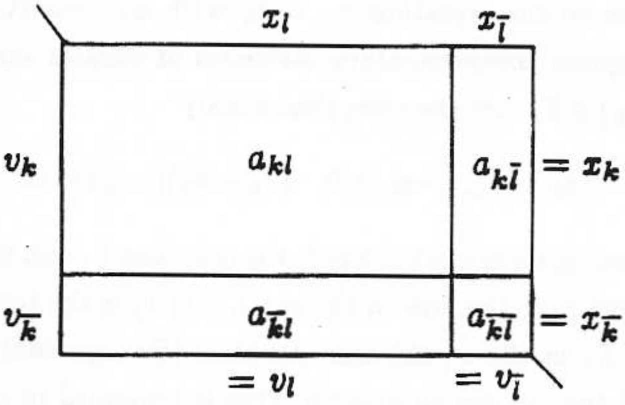


Figure 14.

We take

$$f_j(x_j) = \delta(x_j \| [0,\infty)) = \begin{cases} 0 & \text{if } x_j \geq 0 \\ \infty & \text{if } < 0 \end{cases}$$

for every $j \in J = K \cup L$ except $\overline{k}$ and $\overline{l}$, but

$$f_{\overline{k}}(x_{\overline{k}}) \equiv x_{\overline{k}} \qquad (C_{\overline{k}} = (-\infty, \infty)),$$

$$f_{\overline{l}}(x_{\overline{l}}) = \delta(x_{\overline{l}} | [1,1]) = \begin{cases} 0 \text{ if } x_{\overline{l}} = 1, \\ \infty \text{ if } x_{\overline{l}} \neq 1. \end{cases}$$

The primal problem is then

$$(\mathrm{P}_0) \qquad \begin{gathered} \text{minimize } \; x_{\overline{k}} = \sum_{l \neq \overline{l}} a_{\overline{k}l} x_l + a_{\overline{k}\overline{l}} \; \text{ subject to} \\ x_l \geq 0 \text{ for } l \neq \overline{l}, \quad x_k = \sum_{l \neq \overline{l}} a_{kl} x_l + a_{k\overline{l}} \geq 0 \text{ for } k \neq \overline{k}. \end{gathered}$$

One has

$$g_j(v_j) = \delta(v_j|(-\infty,0]) = \begin{cases} 0 & \text{if } -v_j \leq 0 \\ \infty & \text{if } -v_j < 0 \end{cases}$$

for every j except $\overline{k}$ and $\bar{l}$, but

$$g_{\bar{l}}(v_{\bar{l}}) \equiv v_{\bar{l}} \qquad (D_{\bar{l}} = (-\infty,\infty))$$

$$g_{\overline{k}}(v_{\overline{k}}) = \delta(v_{\overline{k}}|[1,1]) = \begin{cases} 0 & \text{if } v_{\overline{k}} = 1, \\ \infty & \text{if } -v_{\overline{k}} \neq 1, \end{cases}$$

so the dual problem is

(D$_0$)

$$\text{minimize} \quad -v_{\bar{l}} = \sum_{k \neq \overline{k}} v_k a_{\overline{k}l} + a_{\overline{kl}} \ \text{ subject to}$$

$$-v_k \geq 0 \text{ for } k \neq \overline{k}, \qquad -v_l = \sum_{k \neq \overline{k}} v_k a_{kl} + a_{k\bar{l}} \geq 0 \text{ for } l \neq \bar{l}.$$

For every j other than $\overline{k}$ and $\bar{\imath}$, the relation Γ_j is given by the union of the nonnegative x_j axis and the nonpositive v_j axis; it expresses "complementary slackness". The relation $\Gamma_{\overline{k}}$ is the horizontal line through at level 1 on the vertical scale, whereas $\Gamma_{\bar{l}}$ is the vertical line through level 1 on the horizontal scale.

8. The Main Theorems of Monotropic Programming.

The connection between problems (P), (D) and (E) is very tight. The theory is every bit as complete and constructive as in the familiar case of linear programming, but it applies to an enormously richer class of optimization models.

Duality Theorem. *If any one of the following conditions is satisfied,*

(a) the primal problem (P) *has feasible solutions and finite optimal value* inf(P), *or*

(b) the dual problem (D) *has feasible solutions and finite optimal value* sup(D), *or*

(c) the primal problem (P) *and the dual problem* (D) *both have feasible solutions,*

then all three hold, and

$$\inf(P) \quad = \quad \sup(D).$$

Equilibrium Theorem. *A pair (x,v) solves the equilibrium problem* (E) *if and only if x solves the primal problem* (P) *and v solves the dual problem* (D).

Results on the existence of solutions to (P), (D) and (E) require a distinction between the intervals C_j and $\tilde{C}_j$, and between D_j and $\tilde{D}_j$, where $\tilde{C}_j$ and $\tilde{D}_j$ are given by (22) and (24). Whereas a *feasible* solution to (P) is an $x \in C$ such that $x_j \in C_j$ for all $j \in J$, a *regularly feasible* solution is defined to be an $x \in C$ such that $x_j \in \tilde{C}_j$ for all $j \in J$. Likewise a *regularly feasible* solution to (D) is defined to be a $v \in D$ such that $v_j \in \tilde{D}_j$. This distinction falls away in the case of so-called *regular* monotropic programming problems, where $C_j = \tilde{C}_j$ and $D_j = \tilde{D}_j$ for all $j \in J$ (as is true in particular when $\tilde{C}_j$ and $\tilde{D}_j$ are closed, i.e. when Γ_j projects horizontally and vertically onto closed intervals). Important to keep in mind as regular problems in this sense are the problems of piecewise linear programming or piecewise quadratic programming.

Existence Theorem.

(a) The primal problem (P) *has an optimal solution if and only if* (P) *has a feasible solution and* (D) *has a regularly feasible solution.*

(b) The dual problem (D) *has an optimal solution if and only if* (D) *has a feasible solution and* (P) *has a regular feasible solution.*

(c) The equilibrium problem (E) *has a solution if and only if* (P) *and* (D) *both have regularly feasible solutions.*

Corollary. *In the case of regular problems* (P) *or* (D)*, an optimal solution exists if a feasible solution exists and the optimal value is finite.*

By combining the existence theorem with the equilibrium theorem, we obtain the following characterization of optimal solutions to (P) and (D) that is the basis of most computational procedures in monotropic programming.

Optimality Theorem.

(a) Suppose that the primal problem (P) *has at least one regularly feasible solution. Then for x to be an optimal solution to* (P) *it is necessary and sufficient that x be regularly feasible and such that the dual linear system has a vector v satisfying:*

$$v_j \in \partial f_j(x_j) = [f'_{j-}(x_j), f'_{j+}(x_j)] \quad \text{for all} \quad j \in J$$

(Such a v is an optimal solution to (D)*.)*

(b) Suppose that the dual problem (P) *has at least one regularly feasible solution. Then for v to be an optimal solution to* (D) *it is necessary and sufficient that v be regularly feasible and such that the primal linear system has a vector x satisfying*

$$x_j \in \partial g_j(v_j) = [g'_{j-}(v_j), g'_{j+}(v_j)] \text{ for all } j \in J$$

(Such an x is an optimal solution to (P)*.)*

9. Solution by Pivoting Methods.

An important feature of monotropic programming problems is the possibility of solving them by techniques based on repeated pivoting transformations of Tucker tableaus for the underlying linear systems. Such techniques can in some cases be viewed as generalizations of the various forms of the simplex method in linear programming, but they may also be based on distinctly different approaches involving phenomena of another order.

Our aim is to exploit the fact that the primal and dual linear systems of variables can be represented in more than one way by tableaus of the kind in Figure 6. Even though only one such tableau may have been given to us initially, others can be generated by pivoting. We wish to make use of their special properties as a means of constructing a sequence of feasible solutions to (P) or (D) that converges to, or in finitely many steps actually reaches, an optimal solution.

The optimality test provided by the last theorem in the preceding section has a central role in this context. In terms of a Tucker tableau associated with a partition of J into index sets K and L, it says the following: *a regularly feasible solution x to* (P) *is optimal if and only if for some choice of values v_k in the intervals $\partial f_k(x_k)$, $k \in K$, the corresponding values $v_l = -\sum_{k\in K} v_k a_{kl}$ satisfy $v_l \in \partial f_l(x_l)$ for all $l \in L$.* Likewise in the dual problem, *a regularly feasible solution v to* (D) *is optimal if and only if for some choice of values $x_l \in \partial g_l(v_l)$, $l \in L$, the corresponding values $x_k = \sum_{l\in L} a_{kl} x_l$ satisfy $x_k \in \partial g_k(v_k)$ for all $k \in K$.* Our attention actually is directed at the negatives of these conditions, which, as it turns out, can be stated in much sharper form than might be expected. This sharper form is based on the next theorem, which we present in terms of general intervals D'_j and C'_j, not just $\partial f_j(x_j)$ and $\partial g_j(v_j)$, because of its other applications.

Feasibility theorem.

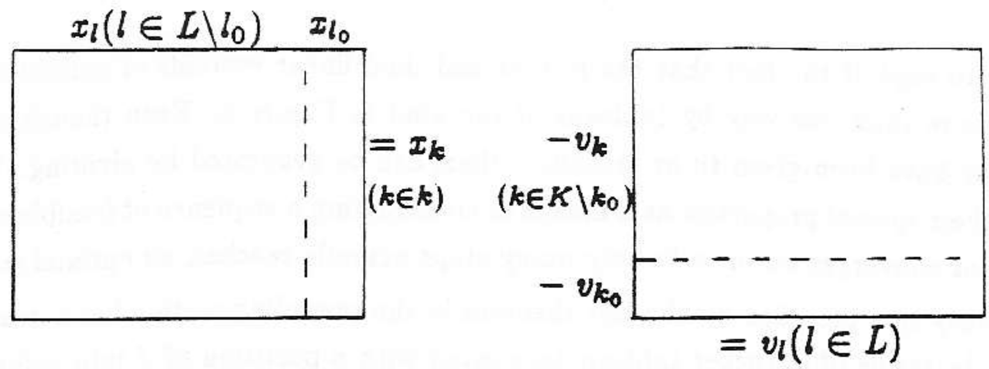
(a) Let C'_j denote a nonempty real interval (not necessarily closed) for each $j \in J$. If there does not exist an $x \in C$ satisfying $x_j \in C'_j$ for every $j \in J$, then in fact there exists a Tucker representation (with J partitioned into some K and L) and an index $k_0 \in K$ such that for no choice of values $x_l \in C'_l$ for $l \in L$ does the number $x_{k_0} = \sum_{l\in L} a_{k_0 l} x_l$ satisfy $x_{k_0} \in C'_{k_0}$.

(b) Let D'_j denote a nonempty real interval (not necessarily closed) for each $j \in J$. If there does not exist a $v \in D$ satisfying $v_j \in D'_j$ for every $j \in J$, then in fact there exists a Tucker representation (with J partitioned into some K and L) and an index $l_0 \in L$ such that for no choice of values $v_k \in C'_k$ for $k \in K$ does the number $v_{l_0} = -\sum_{k\in K} v_k a_{kl_0}$ satisfy $v_{l_0} \in C'_{l_0}$.

This result is valid in particular as a test of feasibility in (P) and (D) (the case of $C'_j = C_j$, $D'_j = D_j$) and regular feasibility ($C'_j = \tilde{C}_j$, $D'_j = \tilde{D}_j$). Our concern at present, though, is with the case of $C'_j = \partial g_j(v_j)$ and $D'_j = \partial f_j(x_j)$, where (a) and (b) characterize nonoptimality of v in (D) and of x in (P), respectively. Pivoting rules do exist for producing special Tucker tableaus and indices k_0 or l_0 with the properties in the theorem. Rather than explain such rules here, which would take too much space, we shall try to indicate the way that the conditions provided by the theorem can be used in optimization.

The basic idea we need to work with is that of trying to improve a given feasible solution x to (P) by changing only one "independent" variable at a time, the primal independent variables relative to a particular Tucker tableau being the ones indexed by the set L in the partition $J = (K|L)$. Consider the situation in Figure 15, where a certain index $l_0 \in L$ has been singled out (first diagram).

We want to look at feasible modifications of x that leave fixed all the values x_l for $l \neq l_0$.



primal case *dual case*

Figure 15.

Such a modification depends only on specifying the value of a single parameter $t \in R$:

$$x'_{l_0} = x_{l_0} + t, \qquad x'_l = x_l \text{ for all } l \in L\backslash l_0,$$

$$x'_k = \sum_{l \in L} a_{kl} x'_l = x_k + a_{kl_0} t \text{ for } k \in K.$$

The corresponding objective value is

$$\varphi(t) = F(x') = \sum_{l \neq l_0} f_l(x_l) + f_{l_0}(x_{l_0} + t) + \sum_{k \in K} f_k(x_k + a_{kl_0} t),$$

where $\varphi(0) = F(x)$ is the objective value already at hand. We are interested in the existence of a t such that $\varphi(t) < \varphi(0)$, because x' is then a "better" feasible solution to (P) than x. (It is still feasible because $F(x') < F(x) < \infty$ implies $x'_j \in C_j$.) When such a t exists, we shall say that ***monotropic improvement*** of x is possible in (P) which respect to the Tucker representation in question and the index $l_0 \in L$. The exact choice the stepsize t, whether by complete minimization of φ or some other device, need not concern us here.

The important thing is that the condition in part (b) of the Feasibility Theorem in the case of the intervals $D'_j = \partial f_j(x_j)$ holds for a particular Tucker representation and index $l_0 \in L$ if and only if monotropic improvement of x is possible for this Tucker representation and index. This can be verified by a calculation of the right and left derivatives of the convex function φ at 0. A similar result holds for the dual problem, where one is interested in improving a given regularly feasible solution v without changing the values v_k for $k \neq k_0$ in a certain tableau (see Figure 15 again, second diagram).

The situation is summarized in the next result.

Improvement Theorem.

(a) If x is a regularly feasible solution to (P) which is not optimal, then monotropic improvement of x is possible with respect to some Tucker representation and index $l_0 \in L$.

(b) If v is a regularly feasible solution to (D) which is not optimal, then monotropic improvement is possible with respect to some Tucker representation and index $k_0 \in K$.

This theorem leads to optimization procedures which alternate between pivoting routines that construct special Tucker tableaus and line searches that minimize special convex functions φ. In network programming, of course, where the Tucker tableaus correspond to spanning trees, the pivoting can be carried out in a combinatorial manner. The monotropic improvement steps then amount to modifying a circulation x only around one closed path at a time, and modifying a differential v only across one cut at a time.

10. Generalized Simplex Methods.

A nice illustration of these ideas, although by no means the only one, is the way that the simplex method of linear programming can be extended to monotropic programming problems in general. Let us call a value of x_j a *breakpoint* of f_j if $f'_{j-}(x_j) \neq f'_{j+}(x_j)$, i.e. if Γ_j has a vertical segment at x_j. (A finite endpoint of C_j that belongs to C_j is a breakpoint in this sense, in particular.) Let us say further that a regularly feasible solution x to (P) is *nondegenerate* if there is a Tucker tableau with partition $J = (K|L)$ such that x_k is not a breakpoint of f_k for any $k \in K$.

Such a tableau, if one exists, is very easy to construct by pivoting: simply exchange breakpoint indices with nonbreakpoint indices until all the breakpoint indices correspond to columns of the tableau instead of rows. One then has a quick test of the optimality of x. The intervals $\partial f_k(x_k)$ for $k \in K$ all consist of just a single point, namely the derivative value $f'_k(x_k)$; taking this as v_k and defining $v_l = -\sum_{k \in K} v_l a_{kl}$, check whether $v_l \in \partial f_l(x_l)$ for every $l \in L$. If "yes", then x is an optimal solution to (P) (and v is an optimal solution to (D); cf. the optimality theorem in §8). If "no", then for some index l_0 one has $v_{l_0} \notin \partial f_{l_0}(x_{l_0})$. Such an index l_0 satisfies the condition in the feasibility theorem in §9 for the intervals $C'_j = \partial f_j(x_j)$, and it therefore signals the possibility of monotropic improvement of x. This improvement can be carried out, and x replaced by x'. If x' is again nondegenerate, the tableau can be restored to proper form with respect to x' if necessary, and the optimality test repeated.

For problems in piecewise linear programming, some refinements of this general simplex procedure are possible. Let us say that x is *quasi-extreme* for (P) if there is a Tucker tableau with partition $J = (K|L)$ such that x_l is a breakpoint of f_l for every $l \in L$. Inasmuch as there are only finitely many breakpoints for each of the cost functions in a piecewise linear programming problem, and also only finitely many possible Tucker tableaus, there can be only finitely many feasible solutions that are quasi-extreme. If the procedure already described is initiated with such a feasible solution x, and if the line search in each monotropic improvement step is carried out with exactitude (which is easy because of piecewise linearity), then all the successive feasible solutions generated by the procedure will be quasi-extreme. Under the *nondegeneracy assumption* that every quasi-extreme feasible solution to (P) is nondegenerate, the procedure can be continued until, after finitely many steps, optimality is achieved.

This general version of the simplex method reduces to the classical method when applied to a linear programming problem in standard form. It also includes the modified simplex method for linear programming problems with upper bounds. It can be used directly on problems obtained from linear programming problems by penalty representation of constraints.

A dual version of all this can be written down in terms of problem (D), of course.

References

1. R.T. Rockafellar, *Network Flows and Monotropic Optimization*, Wiley-Interscience, 1984.