THE PROXIMITY OF
(ALGEBRAIC) GEOMETRIC PROGRAMMING
TO LINEAR PROGRAMMING

by

R. J. Duffin and E. L. Peterson

Report 72-13

May 1972

/ps -- 5-16-72

THE PROXIMITY OF (ALGEBRAIC) GEOMETRIC PROGRAMMING TO LINEAR PROGRAMMING

by

R. J. Duffin and Elmor L. Peterson

ABSTRACT

Geometric programming with (posy)monomials is known to be synonomous with linear programming, This note reduces algebraic programming to geometric programming with (posy)binomials.

* Carnegie-Mellon University, Pittsburgh, Pennsylvania, 15213. Partially supported by the Army under research grant DA--AROD-31-124-71-G17.

** Northwestern University, Evanston, Illinois, 60201. Partially supported by the Northwestern University Urban Systems Engineering Center under research grant #731.

1. Introduction. In [3] we demonstrate the reduction of each well-posed "algebraic program" to an equivalent "posynomial program" in which a posynomial is to be minimized subject only to inequality posynomial constraints (some of which may have a "reversed direction"). Of course, each of those posynomial programs can be reformulated so that its objective function is a (posy)monomial in that it includes only one posynomial term. (To make such a reformulation, simply minimize an additional independent variable that is constrained to be at least as large as the posynomial objective function.)

The purpose of this note is to show that each of those posynomial programs can be further reformulated so that every constraint function is a (posy)binomial in that it includes at most only two posynomial terms. This reformulation is rather striking in view of Federowicz's observation [5, Appendix D] that (posy)monomial programming (i.e., posynomial programming with (posy)monomial objective and constraint functions) is synonomous with linear programming. In fact, we suspect that the resulting proximity of algebraic programming to linear programming may have important computational and theoretical implications.

2. The Reformulation, In [3] (and [2]) we actually show that one need only consider "prototype posynomial constraints'[1]

$$P(t) \text{ £ } 1 \tag{1}$$

and "reversed posynomial constraints"

$$P(t) \wedge 1. \tag{2}$$

If the posynomial $P(t)$ is not already a (posy)binomial it must have the form

$$P(t) = \sum_{1}^{n-2} u_{\pm}(t) + u_{n-1}(t) + u_{n}(t),$$

where its number of terms $n \wedge 3$.

Introducing an additional independent variable s, we readily see that the prototype constraint (1) is equivalent to the two prototype constraints

$$u_{n-l}(t) + u_{n}(t) \wedge s \tag{la}$$

and

$$\sum_{1}^{n-2} u.(t) + s \leq 1. \tag{lb}$$

We also see that the reversed constraint (2) is equivalent to the two reversed constraints

$$U_{n-l}(t) + U_{n}(t) \wedge s \tag{2a}$$

and

$$\sum_{1}^{n-2} u_i(t) + s \geq 1. \tag{2b}$$

After division by the positive variable s, constraint (1a) clearly turns into a prototype (posy)binomial constraint, and constraint (2a) clearly turns into a reversed (posy)binomial constraint, each of which constrains the vector variable (s,t). Of course, constraint (1b) is a prototype posynomial constraint, and constraint (2b) is a reversed posynomial constraint, each of which has only $(n-1)$ terms. Thus, the number of terms in each of the resulting constraints does not exceed $(n-1)$, and a repetition of the preceding technique a total of $(n-3)$ additional times leads to the presence of only (posy)binomial constraints.

If $n \geq 4$, the first additional repetition should probably replace $u_{n-3}(t) + u_{n-2}(t)$ (rather than $u_{n-2}(t) + s$) with an additional independent variable r, so that as many as possible of the resulting terms involve only a single scalar variable in linear fashion.

Every repetition of the preceding technique applied to either a single prototype posynomial constraint or a single reversed posynomial constraint clearly increases by one the total number of terms in the program formulation. But since every repetition also increases by one the total number of independent variables, the program's "degree of difficulty" remains constant.

Although every repetition increases by one both the number of rows and the number of columns in the program's "exponent matrix", most of the additional matrix entries are zero, and the other entries are either minus one or plus one. Hence, this reformulation should not drastically increase the number of matrix computations needed to compute feasible solutions to the corresponding "geometric dual program" [3].

We expect this reformulation to be especially useful in conjunction with the "linearization procedure" of Duffin [2], the "condensation procedure" of Avriel and Williams [1], and the "inversion procedure" of Duffin and Peterson [4].

References

[1] Avriel, M., and A. C. Williams, "Complementary Geometric Programming", SIAM Jour. Appl. Math. 19(1970), 125-141.

[2] Duffin, R. J., "Linearizing Geometric Programs", SIAM Rev. 12(1970), 211-227.

[3] Duffin, R. J., and E. L. Peterson, "Geometric Programming with Signomials", Jour. Opt. Th. Appls., to appear.

[4] Duffin, R. J., and E. L. Peterson, "Reversed Geometric Programs Treated by Harmonic Means", to appear.

[5] Duffin, R. J., E. L. Peterson, and C. Zener, Geometric Programming - Theory and Applications, John Wiley, New York, 1967.