

Reversed Geometric Programs Treated by Harmonic Means

R. J. DUFFIN & ELMOR L. PETERSON

Communicated by WALTER NOLL

Introduction. A “posynomial” is a (generalized) polynomial with arbitrary real exponents, but positive coefficients and positive independent variables. Each posynomial program in which a posynomial is to be minimized subject to only upper-bound inequality posynomial constraints is termed a “prototype geometric program.” Each posynomial program in which a posynomial is to be minimized subject to at least one lower-bound inequality posynomial constraint as well as upper-bound inequality posynomial constraints is termed a “reversed geometric program.”

Originally developed by Duffin, Peterson and Zener [13], prototype geometric programming provides a powerful method for studying many problems in optimal engineering design [28–30, 2, 5, 14, 18, 27]. However, many other important optimization problems can be modelled accurately only by using more general types of algebraic functions. Hence the question of extending the applicability of geometric programming to those larger classes of programs has received considerable attention.

In particular, Section III.4 of [13] presents various techniques for transforming a limited class of algebraic programs into equivalent prototype geometric programs, but many of the most important optimization problems are not within that limited class.

Initial attempts at rectifying this situation were made by Passy and Wilde [21], and Blau and Wilde [6]. They generalized some of the prototype concepts and theorems in order to treat “signomial programs” (namely, programs in which a “signomial”—*i.e.*, the difference of two posynomials—is to be minimized subject to signomial constraints). Subsequently, Duffin and Peterson [11, 12] advanced that work in the still more general setting of algebraic programming. In particular, Appendix A of [11] shows how to transform each well-posed algebraic program into a corresponding equivalent *finite* family of signomial programs; and Section 2 of [11] shows how to further transform each signomial program into a corresponding equivalent reversed geometric