LARGE SCALE GEOMETRIC PROGRAMMING: AN APPLICATION IN CODING THEORY

Yaw O. Chang* and John K. Karlof
Mathematical Sciences Department
The University of North Carolina at Wilmington

*This research was partially supported by the Faculty Research and Development Fund of the University of North Carolina at Wilmington

SCOPE AND PURPOSE

Geometric programming is an optimization technique originally developed for solving a class of nonlinear optimization problems found in engineering and design. Previous applications have generally been small scale and highly nonlinear. In a geometric programming problem all the constraints as well as the objective function are posynomials. The Gaussian channel is a communications model in which messages are represented by vectors in R^n. The transmitter has a finite set of messages available and these messages are transmitted over a noisy channel to a receiver. The received message equals the sent vector perturbed by a Gaussian noise vector. The receiver must decide which of the messages was sent. We use groups of orthogonal matrices to generate the message set and the result is called a group code. The problem of finding the best code generated by a particular group is called the initial vector problem. Previous attempts to find a general solution to this problem have been unsuccessful. Although, it has been solved in several special cases. We write this problem as a nonlinear programming problem, transform it to a generalized geometric programming problem, and use the double condensation technique developed by Avriel, Dembo, and Passy and modified by Burns to develop an algorithm for its solution. We apply this algorithm to groups in which the resulting initial vector problem has close to 50,000 constraints.

ABSTRACT

An algorithm is presented that solves the initial vector problem for group codes for the Gaussian channel. The Gaussian channel is a communications model in which messages are represented by vectors in R^n and transmitted over a noisy channel. In this paper, the set of messages, called a code, is generated by a group of permutation matrices and the problem of finding the code with the largest minimum distance between codewords for a particular permutation group is called the initial vector problem. This problem is an especially large scale nonlinear programming problem. We transform it to a generalized geometric programming problem and use the double condensation method to develop an algorithm for its solution.

1. INTRODUCTION

Geometric programming is an optimization technique originally developed for solving a class of nonlinear optimization problems found in engineering design [14, 15]. In a geometric programming problem all the constraints as well as the objective function are posynomials (sum of terms having a positive real coefficient multiplied by a product of variables where each variable is raised to an arbitrary real constant power) and all the variables are strictly positive. That is, a function of the form:

$$P(x) = \sum_{i=1}^{m} c_i \prod_{j=1}^{n} x_j^{a_{ij}}$$

where $c_i > 0$ and $a_{ij} \in R$. In a generalized geometric programming problem, the coefficients may be arbitrary real numbers. Avriel, Dembo, and Passy [1] describe a technique called double condensation that solves a generalized geometric programming problem by solving a series of linear programming problems. This technique was later modified by Burns [4] to apply to problems with equality constraints.

The Gaussian channel is a communications model in which messages are represented by vectors in R^n and transmitted over a noisy channel. We generate the set of messages by groups of permutation matrices and the resulting message set is called a permutation code. Given an arbitrary permutation group, the problem of finding an optimal code generated by that group may be written as a nonlinear programming problem [10]. The resulting problem is called the initial vector problem and in it each group element together with its inverse create a constraint. Since we will be considering groups as large as 95,000 elements, the associated nonlinear programming problem is especially large scale. In this paper, we transform this large scale nonlinear

programming problem to a generalized geometric programming problem and use the double condensation method to develop an algorithm for its solution. Previous applications of geometric programming have been highly nonlinear problems with few constraints and variables [9, for survey article].

In section 2, we describe the Gaussian channel, permutation codes, and present some previous results on solving the initial vector problem. In section 3 we describe the double condensation technique and present our specialized algorithm. Section 4 contains a numerical example and in section 6 we present computational results.

2. GROUP CODES FOR THE GAUSSIAN CHANNEL

The Gaussian channel is a communications model introduced by Shannon in 1948 in which messages are represented by vectors in R^n and transmitted over a noisy channel. We assume the transmitter has a finite set of messages available and sends one every T seconds. When the vector x is transmitted, the received signal is represented by a vector $y = x + z$ which consists of the sent vector and a Gaussian noise vector z whose components are statistically independent Gaussian random variables of mean zero and variance N. Upon receiving y, the receiver must decide, presumably in some optimum manner, which of the messages was sent. The set of vectors representing the messages is called a code and the process of deciding is called decoding.

In this paper, we consider codes generated by groups of orthogonal n by n matrices (a real matrix, A, is orthogonal if $A^t = A^{-1}$) and more specifically by groups of permutation matrices (A is a permutation matrix if exactly one entry in each row and column is equal to one, and all other entries are zero). Group codes were first defined by Slepian [12] in 1968.

Definition 2.1 *A set X of M n-dimensional unit vectors is an (M,n) group code if there exists a group G of orthogonal n by n matrices and a unit vector x such that $X = \{gx : g \in G\}$. The code X is said to be generated by the group G and the initial vector x. If G is a group of permutation matrices then X is called a permutation code.*

If the set of vectors available to the transmitter is an (M,n) group code where each codeword has the same *a priori* probability of transmission, then the decoding criteria for the receiver that minimizes the average error probability is maximum likelihood [3, section 6.2]. That is, the optimum receiver

decodes y as the codeword $g'x$ that minimizes the set of Euclidean distances $\{d(y, gx) : g \in G\}$. One advantage of using a group code is that all the codewords are on an equal footing; each has the same error probability and the same disposition of codeword neighbors. Thus, the best group codes are those with a large nearest neighbor distance between codewords. This is formulated in the following problem.

Initial Vector Problem: Given a group G of orthogonal n by n matrices, find an initial vector that maximizes the minimum distance between the codewords of all codes generated by G, i.e., find a unit vector x in R^n such that $\min\{d(x, gx)\} = \max \min\{d(z, gz)\}$ where the maximum is taken over all unit vectors z in R^n and the minimum is taken over all nonidentity elements g in G. The vector x is called an optimal initial vector for G and the associated code is called an optimal code generated by G.

Previous attempts to provide a general solution to this problem have not been successful, although it has been solved in some special cases [2, 7, 8, 13]. The evidence so far is that is a very difficult problem and it does not have a closed-form solution.

Suppose G is a group of orthogonal n by n matrices and $V_G = \{y \in R^n : gy = y, \forall g \in G\}$. Let $(\ ,\ )$ denote the ordinary inner product and define $V_G^{\perp} = \{y \in R^n : (y, z) = 0, \forall z \in V_G\}$. Karlof [10] has shown that if $V_G \neq \{0\}$, any optimal initial vector for G must lie in $V_G^{\perp}$. Karlof [10] presented a numerical algorithm for the initial vector problem that incorporates this fact. This algorithm is a two phase method. Phase I is a modification of Topkis and Veinott's feasible directions method that identifies a set of active constraints and a starting point for phase II. Phase II uses a Newton-Raphson technique on the equations formed by the active constraints, the

equations defining $V_G^{\perp}$ (if $V_G \neq \{0\}$), and the equation $\sum x_i^2 = 1$. Karlof has applied this algorithm to many permutation groups, the largest one being the Mathieu group of degree 11 and order 7920. It is also shown in [10] that if G is a doubly transitive permutation group, then $V_G^{\perp} = \{y : \sum y_i = 0\}$. Let G be a doubly transitive permutation group. Now, since $d^2(gx, x) = 2 - 2(gx, x)$ and $(gx, x) = (g^{-1}x, x)$, an equivalent formulation of the initial vector problem(IVP) is

$$\min z_{n+1}$$

$$\sum_{i=1}^{n} z_i = 0$$

$$\sum_{i=1}^{n} z_i^2 = 1$$

$$(gz, z) \leq z_{n+1} \quad g \in S$$

$$z = (z_1, \ldots, z_n) \in R^n, z_{n+1} \in R$$

where S is a complete list of the group elements excluding inverses and the identity. We now let M denote the cardinality of S. Note that this formulation of problem (IVP) is a geometric programming problem except the variables are unrestricted. It can be transformed to a generalized geometric programming problem by replacing $z_i = z_i^+ - z_i^-$ and restricting $z_i^{\pm} \geq \rho$ for some $\rho > 0$. We present an algorithm that uses the double condensation methods to solve this transformed formulation of the initial vector problem. We apply this algorithm to permutation groups as large as 95,040.

3. ALGORITHM

A generalized geometric programming problem is of the form:

$$\min P_0(x) - Q_0(x)$$

$$\begin{aligned} P_k(x) - Q_k(x) &= 0, k = 1, 2, \ldots, M' \\ P_k(x) - Q_k(x) &\leq 0, k = M' + 1, M' + 2, \ldots, N \end{aligned} \tag{3.1}$$

$$0 < x_j^{LB} \leq x_j \leq x_j^{UB} j = 1, \ldots, n$$

where $P_k(x)$ and $Q_k(x)$ are *posynomials* of the general form:

$$P_k(x) = \sum_{i=1}^{I_k} u_{ik}(x) = \sum_{i=1}^{I_k} c_{ik} \prod_{j=1}^{n} x_j^{a_{jik}},$$

$$Q_k(x) = \sum_{l=1}^{L_k} v_{lk}(x) = \sum_{l=1}^{L_k} d_{lk} \prod_{j=1}^{n} x_j^{b_{jlk}},$$

and $c_{ik} > 0$, for all i, k, and $d_{lk} > 0$ for all l, k.

Before we give the description of the double condensation method [4] for solving problem IVP, we describe the condensation technique. The classical *arithmetic-geometric inequality* stating that the weighted arithmetic mean of positive numbers a_1, a_2, ..., a_n is greater than or equal to the geometric mean may be written as follows:

$$\sum_{i=1}^{n} a_i \geq \prod_{i=1}^{n} \left(\frac{a_i}{w_i} \right)^{w_i} \tag{3.2}$$

where $\sum w_i = 1$, and $w_i > 0$ for all i. Equality holds in (3.2) if and only if

$$\frac{a_1}{w_1} = \frac{a_2}{w_2} = \ldots = \frac{a_n}{w_n}.$$

Let $p(x) \leq 1$ be a functional constraint where $p(x)$ is a posynomial of the form

$$p(x) = \sum_{i=1}^{I} c_i \prod_{j=1}^{n} x_j^{a_{ij}}, c_i > 0;$$

and $x' > 0$ be given. Define

$$w_i = \frac{c_i \prod_{j=1}^{n} {x'}_j^{a_{ij}}}{p(x')}, \forall i = 1, \ldots, I.$$

Then $w_i > 0$ and $\sum_{i=1}^{I} w_i = 1$. Therefore, as a direct consequence of (3.2) we have

$$p(x, x') \leq p(x) \leq 1, \tag{3.3}$$

where

$$p(x, x') = \prod_{i=1}^{I} \left(\frac{c_i \prod_{j=1}^{n} x_j^{a_{ij}}}{w_i} \right)^{w_i}.$$

The monomial $p(x, x')$ is the condensed form of $p(x)$, and equals the original posynomial at the point of condensation x'. Note that (3.3) implies that any x that satisfies $p(x) \leq 1$ will also satisfy $p(x, x') \leq 1$, but not vice versa.

We convert problem IVP to a generalized geometric programming problem by writing the variables $z_i = z_i^+ - z_i^- (i = 1, \ldots, n)$ and restricting $z_i^+, z_i^- \geq \rho > 0$. It is no loss of generality to restrict $z_{n+1} > 0$ since the maximum inner product is always positive. Thus, this generalized geometric programming problem has $2n + 1$ variables:

$$\begin{aligned} \min z_{n+1} & \\ \sum_{i=1}^{n} z_i^+ - \sum_{i=1}^{n} z_i^- &= 0 \\ \sum_{i=1}^{n} ({z_i^+}^2 + {z_i^-}^2) - 2\sum_{i=1}^{n} z_i^+ z_i^- &= 1 \\ (gz^+, z^+) + (gz^-, z^-) - [(gz^+, z^-) + (gz^-, z^+)] &\leq z_{n+1} g \in S \end{aligned} \tag{3.4}$$

$$\rho \leq z_i^+, z_i^-, 0 < z_{n+1}$$

where $z^{\pm} = (z_1^{\pm}, z_2^{\pm}, \ldots, z_n^{\pm})$.

An equivalent version of (3.4) is:

$$\begin{aligned} \min z_{n+1} \\ \\ \frac{\sum_{i=1}^n z_i^+}{\sum_{i=1}^n z_i^-} &= 1 \\ \frac{\sum_{i=1}^n {z_i^+}^2 + {z_i^-}^2}{1 + 2\sum_{i=1}^n z_i^+ z_i^-} &= 1 \\ \frac{(gz^+, z^+) + (gz^-, z^-)}{z_{n+1} + (gz^+, z^-) + (gz^-, z^+)} &\leq 1 \\ \\ \rho \leq z_i^+, z_i^-, 0 < z_{n+1} \end{aligned} \tag{3.5}$$

.

We now describe how to solve (3.5) by solving a serie of linear programming problems via the double condensation method. Let $\overline{z} \in R^{2n+1}$ be a feasible solution of (3.5). Using the condensation technique described above, the denominators of (3.5) can be condensed to monomials and (3.5) becomes a posynomial geometric programming problem. Now we perform another condensation on the new posynomial geometric programming problem at $\overline{z}$ to obtain a the following monomial programming problem. We let $z = (z_1^+, z_1^-, z_2^+, z_2^-, \ldots, z_n^+, z_n^-, z_{n+1})$.

$$\begin{aligned} \min z_{n+1} \\ \\ C_k(z) &= 1, k = 1, 2. \\ C_k(z) &\leq 1, k = 3, \ldots, M+2, \end{aligned}$$

$$\rho \leq z_i^+, z_i^-, 0 < z_{n+1}$$

.

where $C_k(z)$ is a monomial.

The natural logarithm function $\ln y$ is monotonicly increasing and is defined for $y > 0$. Therefore, (3.6) can be transformed to a linear programming problem through a change of variables: $y_j^{\pm} = \ln z_j^{\pm}, j = 1, \ldots, n$ and $y_{n+1} = \ln(z_{n+1} + 1)$. Also, we set $\rho = 1$ so that $y_j^{\pm} \geq 0$. Problem LP($\overline{z}$):

$$\max -y_{n+1}$$

$$\begin{aligned} f_1 y &= b_1 \\ f_2 y &= b_2 \\ Ay &\leq B \end{aligned}$$

$$0 \leq y_i^+, y_i^- \leq \ln 2$$

$$0 \leq y_{n+1}$$

where f_1, f_2, and b_1, b_2 are the left side and right side respectively obtained from the two equality constraints of problem (3.6), A is an M by 2n+1 matrix and B is an M dimensional column vector obtained from the inequality constraints of problem (3.6). We restrict $0 \leq y_i^{\pm} \leq \ln 2$ so that $1 \leq z_i^{\pm} \leq 2$ and then $0 \leq z_i \leq 1$. Since y_{n+1} is restricted ≥ 0 in problem LP($\overline{z}$), $z_{n+1} = e^{y_{n+1}} - 1$.

This linear programming problem has $M + 2$ constraints and 2n+1 variables. The algorithm will be applied to groups with M as large as 50,000 and n as large as 20. Thus, in this case, the problem would have on the order

of 50,000 constraints and, after adding slacks, on the order of 50,000 variables. However, the dual of the problem has only 2n+1 constraints all of type greater than or equal to. The right hand side values of these constraints are all zero except for the last constraint where the value is minus one. Since the primal has two equalities, M inequalities, and $2n$ upper bound constraints, the dual has 2 unrestricted and $M + 2n$ restricted variables. We multiply all the constraints of the dual by minus one, add slack variables, write each of the two unrestricted variables as the difference of two restricted variables and obtain a linear programming problem in standard form with 2n+1 equality constraints and $M + 4n + 5$ variables. This is a great improvement over the number of constraints in the primal. Solving the dual has other advantages. We may take the slack variables as our beginning basis. The double condensation algorithm of [4] involves condensing the most violated inequality constraint of the original problem, transforming it to linear form, adding it to the primal problem, and resolving the new linear programming problem. But adding a constraint to the primal is equivalent to adding a variable to the dual. So when solving the dual problem with the new variable added, we may use our current optimal solution as our starting basic feasible solution and pivot on the new variable.

We incorporate the above observations in the following algorithm which employs the double condensation algorithm of [4] to solve the initial vector problem.

STEP 0: Form problem IVP. Let $\bar{z}^0$ be a feasible solution of problem IVP. Set K=0 and OBJ=$\bar{z}^0_{n+1}$ and choose $\epsilon > 0$.

STEP 1: Apply the double condensation technique and linearization procedure to form linear program LP($\bar{z}^K$). Form the dual DL($\bar{z}^K$) of linear

program LP($\bar{z}^K$). Solve DL($\bar{z}^K$) using the slack variables as the beginning basic feasible solution. Let x^{K+1} be the optimal basic feasible solution to DL($\bar{z}^K$). Compute the corresponding solution $\bar{z}^{K+1}$ to problem IVP.

STEP 2: Evaluate the inequality constraints in problem IVP at $\bar{z}^{K+1}$. If $\bar{z}^{K+1}$ satisfies all the inequality constraints, then go to step 5.

STEP 3: Recondense the most violated constraint at $\bar{z}^{K+1}$, transform it to linear form, and append it to DL($\bar{z}^K$) as a new variable.

STEP 4: Solve the appended problem DL($\bar{z}^K$) using x^{K+1} as the beginning basic feasible solution and pivot on the new variable. Let x^{K+1} be the optimal basic feasible solution to the appended problem DL($\bar{z}^K$). Compute the corresponding solution $\bar{z}^{K+1}$ to problem IVP. Go to step 2.

STEP 5: If $\mid \bar{z}_{n+1}^{K+1} - OBJ \mid < \epsilon$, then stop. The solution is $\bar{z}^{K+1}$. Otherwise, set K=K+1, $OBJ = \bar{z}_{n+1}^{K}$ and go to step 1.

Remarks

1. Steps 2 thru 4 form a method to obtain an optimal solution to the posynomial geometric programming problem. When a solution $\bar{z}^{K+1}$ passes the test in STEP 3, it is a global optimum to the posynomial goemetric programming problem. The linear constraint generated in STEP 4 reduces the feasible region of $LP(\bar{z}^K)$ so that $\bar{z}^{K+1}$ no longer belongs to it but all the feasible solutions of the posynomial geometric programming problem still remain a subset of this new feasible region. The new linear constraint is generally referred to as a *cut*.

2. This algorithm is not guaranted to converge to a global optimum of the original problem. Avriel, Dembo, and Passy [1] prove only convergence to a Kuhn-Tucker point for a generalized geometric programming problem without equality constraints.

3. The equality constraints are not involved in the calculations of STEP 3 and do not change during each major interation.

4. EXAMPLE

In this section, we present an example of the algorithm described in section 3 applied to a small permutation group of degree 4. The group is the alternating group on four elements, $A_4 = \{(1), (1,2)(3,4), (1,3)(2,4), (1,4)(2,3), (1,2,3), (1,3,2), (1,3,4), (1,4,3), (2,3,4), (2,4,3), (1,2,4), (1,4,2)\}$. Problem IVP from section 2 is:

$$\min\ z_5$$

$$\begin{aligned}
z_1 + z_2 + z_3 + z_4 &= 0 \\
z_1^2 + z_2^2 + z_3^2 + z_4^2 &= 1 \\
z_1z_2 + z_2z_1 + z_3z_4 + z_4z_3 &\leq z_5 \\
z_1z_3 + z_2z_4 + z_3z_1 + z_4z_2 &\leq z_5 \\
z_1z_4 + z_2z_3 + z_3z_2 + z_4z_1 &\leq z_5 \\
z_1z_3 + z_2z_2 + z_3z_4 + z_4z_1 &\leq z_5 \\
z_1z_4 + z_2z_1 + z_3z_3 + z_4z_2 &\leq z_5 \\
z_1z_1 + z_2z_4 + z_3z_2 + z_4z_3 &\leq z_5 \\
z_1z_2 + z_2z_3 + z_3z_1 + z_4z_4 &\leq z_5 \\
z &\in R^5
\end{aligned}$$

To change problem IVP to a generalized geometic programming problem, we write $z_i = z_i^+ - z_i^-(i = 1, \ldots, 4)$ and restrict $z_i^+, z_i^- \geq 1$. Then the problem has 9 constraints and 9 variables.

STEP 0: Choose $\bar{z}^0 = (.4082, .4082, -.8165, 0.0, .8333)$ then OBJ=$\bar{z}^0_{n+1}$ = .8333. Set K=0 and $\epsilon = .0001$.

STEP 1: After applying the double condensation technique, problem LP($\bar{z}^0$) is formed:

$$f_1 = (.094, -.067, .094, -.067, -.133, .242, .000, .000, .000) \quad b_1 = .209$$

$$f_2 = (.292, -.208, .292, -.208, .208, -.377, .208, -.208, .000) \quad b_2 = -.025$$

$$A = \begin{pmatrix} .100 & -.057 & .100 & -.057 & .006 & .011 & -.119 & .136 & -.140 \\ -.142 & .151 & .025 & .018 & .084 & -.076 & .084 & -.042 & -.133 \\ .025 & .018 & -.142 & .151 & .084 & -.076 & .084 & -.042 & -.133 \\ -.069 & .078 & .109 & -.049 & .049 & -.034 & -.017 & .047 & -.137 \\ .044 & -.031 & .044 & -.031 & -.125 & .226 & .062 & -.062 & -.140 \\ .109 & -.049 & -.069 & .078 & .045 & -.034 & -.016 & .047 & -.137 \\ -.015 & .052 & -.015 & .052 & .081 & -.080 & .016 & .016 & -.133 \end{pmatrix}$$

$$B = (.029, -.051, -.051, -.011, .080, -.011, -.037)^T.$$

The dual, DL($\bar{z}^0$), of the above problem is formed. After putting it in standard form, it has 9 constraints and 28 variables. After 8 pivots, the optimal basic feasible solution is x^1 with basic variables $x^1_2 = 4.690, x^1_3 = .0010, x^1_5 = 2.246, x^1_9 = 4.898, x^1_{10} = .0011, x^1_{21} = .0331, x^1_{23} = .0334, x^1_{24} = .0281, x^1_{26} = .0400$. The corresponding solution to LP($\bar{z}^0$) is

$$y = (.692, .000, .0621, .000, .000, .570, .000, .147, .519)^T$$

and to problem IVP is $\bar{z}^1 = (.999, .064, -.768, -.159, .680)$ where $\bar{z}^{1\pm}_i = e^{y^{\pm}_i}$ for $i = 1, \ldots, 4$, $\bar{z}^1_i = \bar{z}^{1^+}_i - \bar{z}^{1^-}_i$ and $\bar{z}^1_5 = e^{y_5} - 1$.

STEPS 2 and 3: We evaluate problem IVP at $\bar{z}^1$. The most violated inequality constraint is the sixth one. It is recondensed at $\bar{z}^1$ and appended to DL($\bar{z}^0$) as a new variable. The new coefficient in the objective function is .101 and the new column is $(.275, -.150, -.077, .062, -.011, .005, -.055, .053, -.122)^T$.

STEP 4: The appended problem is solved using x^1 as the beginning basic feasible solution and pivoting on the new variable. An optimal solution is obtained after 2 pivots. The corresponding solution (again denoted by $\bar{z}^1$) to problem IVP is $\bar{z}^1 = (.849, .150, -.768, -.159, .680)$ and we return to step 2.

STEPS 2, 3, and 4: The sixth constraint is again the most violated inequality and is recondensed at $\bar{z}^1$ and appended to DL($\bar{z}^0$) as a new variable. The appended problem is solved after 2 pivots and the corresponding solution to problem IVP is $\bar{z}^1 = (.839, .157, -.768, -.159, .680)$.

STEP 2: $\bar{z}^1$ satisfies all the inequality constraints of problem IVP.

STEP 5: $\mid \bar{z}^1_5 - OBJ \mid = \mid .680 - .833 \mid > .0001$. Set $K = 1, OBJ = .680$ and return to step 1.

REMARK: The algorithm converges after 3 more iterations with

$$\begin{array}{rcccccccc} \bar{z}^2 & = & (& .69000 & .16250 & -.68933 & -.16273 & .45013 &) \\ \bar{z}^3 & = & (& .68819 & .16246 & -.68819 & -.16246 & .44721 &) \\ \bar{z}^4 & = & (& .68819 & .16246 & -.68819 & -.16246 & .44721 &) \end{array}$$

5. COMPUTATIONAL RESULTS

In this section, we present the results of some computational experiments in which the algorithm of section 3 was used to calculate the optimal initial vectors of several well-known permutation groups. The groups were generated using a table of generators in [11] and group algorithms from [5]. The computations were done on a VAX 6420 at the University of North Carolina at Wilmington and a CRAY Y-MP at the North Carolina Supercomputer Center.

Recall from the discussion of section 3, the linear programming problem DL($\overline{z}^K$) has a very special structure. The right hand side values of the constraints are all zero except for the last constraint. As a consequence, DL($\overline{z}^K$) is highly degenerate which can cause computational problems. To avoid this situation, we employ a perturbation method found in [6]. We found that this perturbation results in a substantial decrease in the number of pivots. In Table 5.1, we compare the number of pivots with perturbation and without perturbation required at the beginning of each major iteration (i.e., everytime DL($\overline{z}^K$) was formed in STEP 1) for the numerical example of the permutation group A_4 presented in section 4. We can see the number of pivots decreased significantly even for such a small group. We also compare the total number of pivots required of the entire calculations for the groups A_4, M_{11}, and $PSL(4,2)$. The group M_{11} eventually encounters cycling without perturbation and the calculation could not be finished. These results are shown in Table 5.2.

We tested the proposed algorithm on five large permutation groups: they are the symmetric group S_8, the Mathieu groups M_{11} and M_{12}, the projective unimodular group $PSL(4,2)$, and a group of order 40,320 and degree 16

which we call P_{16}. The initial starting solutions were generated randomly by IMSL subroutine RNUN. See Table 5.3 for the results, and note the following nomenclature:

$|G|$: order of the permutation group,

M: number of less than or equal to constraints in problem (IVP),

n: degree of the permutation group,

INT: number of major iterations,

MIN DIS: minimum distance of optimal code.

Table 5.1: Comparison for group $A4$.

iteration	No. of pivots without perturbation	No. of pivots with perturbation
1	23	8
2	19	10
3	23	7
4	50	7

Table 5.2: Comparison of total number of pivots for three groups

Group	Total No. of pivots without pertur.	Total No. of pivots with pertur.
A_4	121	38
M_{11}	–	805
$PSL(4,2)$	28,309	1,892

Table 5.3: Computational results for 5 large groups

Group	$\|G\|$	M	n	INT	MIN DIS
S_8	40,320	20,497	8	4	.218
M_{11}	7,920	4,042	11	5	.633
M_{12}	95,040	47,965	12	13	.488
$PSL(4,2)$	20,160	10,237	15	8	.635
P_{16}	40,320	20,377	16	11	.690

6. CONCLUSIONS

The algorithm developed to solve the initial vector problem in this paper differs from the previous method [10] in several respects. In [10], Karlof employed a two phase algorithm. Phase I was a modification of Topkis and Veinott's feasible directions method and was used to identify a set of active constraints. Phase II used a Newton-Raphson technique on the equations formed by the active constraints. Phase I failed to converge for large groups. The largest group solved was of order 7,920. Convergence of phase I was

also sensitive to the choice of the starting point. On the other hand, the algorithm developed in this paper never failed to converge and the starting points were choosen randomly. The largest group solved in this paper was of order 95,040. This was the largest group we were able to generate. We believe this algorithm has the potential to handle much larger groups.

References

[1] M. Avriel, R. Dembo and U. Passy, "Solution of generalized geometric programs," *Int. J. Numer. Methods Engrg.*, **9**, 149-169 (1975).

[2] I. Blake, "Distance properties of group codes for the Gaussian channel," *SIAM J. Appl. Math.*, **23, No. 3**, 312-324, (1972).

[3] I. Blake and R. Mullin, *The Mathematical Theory of Coding*, Academic Press, New York (1975).

[4] S. A. Burns, "Generalized geometric programming with many equality constraints," *Int. J. Numer. Methods Engrg.*, **24**, 725-741 (1987).

[5] J. Cannon, "The basis of a computer system for modern algebra," *SYMAC'81, Proc. 1981 ACM Symp. Symbolic and Algebraic Computation*, (1981).

[6] A. Charnes, "Optimality and degeneracy in linear programming," *Econometrica*, **20**, 160-170, (1952).

[7] D. Djokovic and I. Blake, "An optimization problem for unitary and orthogonal representations of finite groups," *Trans. Amer. Math. Soc.*, **164**, 267-274, (1972).

[8] C. Downey and J. Karlof, "The analysis of optimal [M,3] group codes for the Gaussian channel," *Util. Math.*, **18**, 51-70, (1980).

[9] J. Ecker, "Geometric progamming: methods, computations, and applications," *SIAM Review*, **22, No. 3**, 338-362, (1980).

[10] J. Karlof, "Permutation codes for the gaussian channel," *IEEE Trans. on Inform. Theory.* **IT-35, No. 4**, 726-732, (1989).

[11] C. Sims, "Computational methods for permutation groups," in *Computational Problems in Abstract Algebra*. Oxford, UK:Pergamon (1970).

[12] D. Slepian, "Group codes for the Gaussian channel," *Bell Syst. Tech. J.*, **17**, 575-602, (1968).

[13] ___, "Permutation modulation," *Proc. IEEE*, **53**, 228-236, (1965).

[14] C. Zener,"A mathematical aid in optimizing engineering design," *Proc. Nat. Acad. Sci. U.S.A.*, **47**, 537-539,(1961).

[15] ___,"A further mathematical aid in optimizing engineering design," *Ibid.*, **48**, 518-522, (1962).