# The Multiplicative Algebraic Reconstruction Technique Solves the Geometric Programming Problem

Charles Byrne (Charles_Byrne@uml.edu)
Department of Mathematical Sciences
University of Massachusetts Lowell
Lowell, MA 01854, USA

October 23, 2007

**Abstract**

The geometric programming problem (GP) is to minimize a posynomial

$$g(t) = \sum_{i=1}^{I} c_i \Big( \prod_{j=1}^{J} t_j^{a_{ij}} \Big),$$

over $t = (t_1, ..., t_J)$ positive, with $c_i > 0$ and $a_{ij}$ real. The dual geometric programming problem is to maximize

$$v(\delta) = \prod_{i=1}^{I} \Big( \frac{c_i}{\delta_i} \Big)^{\delta_i},$$

over all positive vectors $\delta$ with $\sum_{i=1}^{I} \delta_i = 1$, and $\sum_{i=1}^{I} a_{ij}\delta_i = 0$, for $j = 1, ..., J$. Maximizing $v(\delta)$, subject to these linear constraints, is equivalent to minimizing the Kullback-Leibler distance

$$KL(\delta, c) = \sum_{i=1}^{I} \Big( \delta_i \log \Big( \frac{\delta_i}{c_i} \Big) + c_i - \delta_i \Big),$$

subject to the same constraints. We can use the iterative multiplicative algebraic reconstruction technique (MART) to solve the DGP problem, even though the system of linear equations involves a matrix with some negative entries. When the solution of the DGP problem is positive, we can use it to obtain the solution of the GP problem.

## 1 Introduction

The Geometric Programming (GP) Problem involves the minimization of functions $g(t) = g(t_1, ..., t_J)$ of a special type, known as posynomials. The first systematic treatment of geometric programming appeared in the book by Duffin, Peterson and Zener [3], the founders of geometric programming. As we shall see, the Generalized Arithmetic-Geometric Mean (GAGM) Inequality plays an important role in the theoretical treatment of geometric programming.

The GAGM Inequality, applied to the function $g(t)$, leads to the inequality

$$g(t) \geq v(\delta) = v(\delta_1, ..., \delta_I),$$

where $v(\delta)$ is a second function of a particular type. The dual geometric programming (DGP) problem is to maximize the function $v(\delta)$ over positive vectors $\delta$ satisfying certain linear constraints. The function $v(\delta)$ is closely related to cross-entropy and the DGP problem can be solved using the iterative multiplicative algebraic reconstruction technique (MART), even though the system of linear equality constraints involves a matrix with some negative entries.

We begin by describing the GP problem and deriving the DGP problem, following the treatment in Peressini, Sullivan and Uhl [6]. Then we state and prove the main convergence theorem for the MART. Finally, we return to the DGP problem, to show how the linear constraints in that problem can be modified to fit the requirements of the MART.

# 2 The GP and DGP Problems

## 2.1 An Example of a GP Problem

The following optimization problem was presented originally by Duffin, *et al.* [3] and discussed by Peressini *et al.* in [6]. It illustrates well the type of problem considered in geometric programming. Suppose that 400 cubic yards of gravel must be ferried across a river in an open box of length $t_1$, width $t_2$ and height $t_3$. Each round-trip cost ten cents. The sides and the bottom of the box cost 10 dollars per square yard to build, while the ends of the box cost twenty dollars per square yard. The box will have no salvage value after it has been used. Determine the dimensions of the box that minimize the total cost.

With $t = (t_1, t_2, t_3)$, the cost function is

$$g(t) = \frac{40}{t_1 t_2 t_3} + 20t_1t_3 + 10t_1t_2 + 40t_2t_3, \tag{2.1}$$

which is to be minimized over $t_j > 0$, for $j = 1, 2, 3$. The function $g(t)$ is an example of a posynomial.

## 2.2 Posynomials and the GP Problem

Functions $g(t)$ of the form

$$g(t) = \sum_{i=1}^{I} c_i \Big( \prod_{j=1}^{J} t_j^{a_{ij}} \Big), \tag{2.2}$$

with $t = (t_1, ..., t_J)$, the $t_j > 0$, $c_i > 0$ and $a_{ij}$ real, are called *posynomials*. The *geometric programming problem*, or the GP problem, is to minimize a given posynomial over positive $t$. In order for the minimum to be greater than zero, we need some of the $a_{ij}$ to be negative.

We denote by $u_i(t)$ the function

$$u_i(t) = c_i \prod_{j=1}^{J} t_j^{a_{ij}}, \tag{2.3}$$

so that

$$g(t) = \sum_{i=1}^{I} u_i(t). \tag{2.4}$$

For any choice of $\delta_i > 0$, $i = 1, ..., I$, with

$$\sum_{i=1}^{I} \delta_i = 1,$$

we have

$$g(t) = \sum_{i=1}^{I} \delta_i \Big( \frac{u_i(t)}{\delta_i} \Big). \tag{2.5}$$

Applying the Generalized Arithmetic-Geometric Mean (GAGM) Inequality, we have

$$g(t) \geq \prod_{i=1}^{I} \Big( \frac{u_i(t)}{\delta_i} \Big)^{\delta_i}. \tag{2.6}$$

Therefore,

$$g(t) \geq \prod_{i=1}^{I} \Big( \frac{c_i}{\delta_i} \Big)^{\delta_i} \Bigg( \prod_{i=1}^{I} \prod_{j=1}^{m} t_j^{a_{ij}\delta_i} \Bigg), \tag{2.7}$$

or

$$g(t) \geq \prod_{i=1}^{I} \Big( \frac{c_i}{\delta_i} \Big)^{\delta_i} \Big( \prod_{j=1}^{m} t_j^{\sum_{i=1}^{I} a_{ij}\delta_i} \Big), \tag{2.8}$$

Suppose that we can find $\delta_i > 0$ with

$$\sum_{i=1}^{I} a_{ij}\delta_i = 0, \tag{2.9}$$

for each $j$. Then the inequality in (2.8) becomes

$$g(t) \geq v(\delta), \tag{2.10}$$

for

$$v(\delta) = \prod_{i=1}^{I} \Big(\frac{c_i}{\delta_i}\Big)^{\delta_i}. \tag{2.11}$$

## 2.3 The Dual GP Problem

The *dual geometric programming problem*, or the DGP problem, is to maximize the function $v(\delta)$, over all *feasible* $\delta = (\delta_1, ..., \delta_I)$, that is, all positive $\delta$ for which

$$\sum_{i=1}^{I} \delta_i = 1, \tag{2.12}$$

and

$$\sum_{i=1}^{I} a_{ij}\delta_i = 0, \tag{2.13}$$

for each $j = 1, ..., J$. Clearly, we have

$$g(t) \geq v(\delta), \tag{2.14}$$

for any positive $t$ and feasible $\delta$. Of course, there may be no feasible $\delta$, in which case the DGP problem is said to be *inconsistent*.

As we have seen, the inequality in (2.14) is based on the GAGM Inequality. We have equality in the GAGM Inequality if and only if the terms in the arithmetic mean are all equal. In this case, this says that there is a constant $\lambda$ such that

$$\frac{u_i(t)}{\delta_i} = \lambda, \tag{2.15}$$

for each $i = 1, ..., I$. Using the fact that the $\delta_i$ sum to one, it follows that

$$\lambda = \sum_{i=1}^{I} u_i(t) = g(t), \tag{2.16}$$

and

$$\delta_i = \frac{u_i(t)}{g(t)}, \tag{2.17}$$

for each $i = 1, ..., I$. As the theorem below asserts, if $t^*$ is positive and minimizes $g(t)$, then $\delta^*$, the associated $\delta$ from Equation (2.17), is feasible and solves the DGP problem. Since we have equality in the GAGM Inequality now, we have

$$g(t^*) = v(\delta^*).$$

The main theorem in geometric programming is the following.

**Theorem 2.1** *If $t^* > 0$ minimizes $g(t)$, then (DGP) is consistent. In addition, the choice*

$$\delta_i^* = \frac{u_i(t^*)}{g(t^*)} \tag{2.18}$$

*is feasible and solves (DGP). Finally,*

$$g(t^*) = v(\delta^*); \tag{2.19}$$

*that is, there is no duality gap.*

**Proof:** We have

$$\frac{\partial u_i(t^*)}{\partial t_j} = \frac{a_{ij} u_i(t^*)}{t_j^*}, \tag{2.20}$$

so that

$$t_j^* \frac{\partial u_i(t^*)}{\partial t_j} = a_{ij} u_i(t^*), \tag{2.21}$$

for each $j = 1, ..., J$. Since $t^*$ minimizes $g(t)$, we have

$$0 = \frac{\partial g(t^*)}{\partial t_j} = \sum_{i=1}^{I} \frac{\partial u_i(t^*)}{\partial t_j}, \tag{2.22}$$

so that, from Equation (2.21), we have

$$0 = \sum_{i=1}^{I} a_{ij} u_i(t^*), \tag{2.23}$$

for each $j = 1, ..., J$. It follows that $\delta^*$ is feasible. Since we have equality in the GAGM Inequality, we know

$$g(t^*) = v(\delta^*). \tag{2.24}$$

Therefore, $\delta^*$ solves the DGP problem. This completes the proof. ∎

## 2.4 Solving the GP Problem

The theorem suggests how we might go about solving the GP problem. First, we try to find a feasible $\delta^*$ that maximizes $v(\delta)$. This means we have to find a positive solution to the system of $m+1$ linear equations in $n$ unknowns, given by

$$\sum_{i=1}^{I} \delta_i = 1, \tag{2.25}$$

and

$$\sum_{i=1}^{I} a_{ij}\delta_i = 0, \tag{2.26}$$

for $j = 1, ..., J$, such that $v(\delta)$ is maximized. If there is no such vector, then the GP problem has no minimizer. Once the desired $\delta^*$ has been found, we set

$$\delta_i^* = \frac{u_i(t^*)}{v(\delta^*)}, \tag{2.27}$$

for each $i = 1, ..., I$, and then solve for the entries of $t^*$. This last step can be simplified by taking logs; then we have a system of linear equations to solve for the values $\log t_j^*$.

# 3 The MART

The MART [4], which can be applied only to linear systems $y = Px$ in which the matrix $P$ has non-negative entries and the vector $y$ has only positive entries, is a sequential, or row-action, method that uses one equation only at each step of the iteration. We present the general MART algorithm, and then two versions of the MART.

## 3.1 The Algorithms

The general MART algorithm is the following [1].

**Algorithm 3.1 (The General MART)** *Let $x^0$ be any positive vector, and $i = k(\mathrm{mod}\, I) + 1$. Having found $x^k$ for positive integer $k$, define $x^{k+1}$ by*

$$x_j^{k+1} = x_j^k \Big(\frac{y_i}{(Px^k)_i}\Big)^{\gamma_j \delta_i P_{ij}}. \tag{3.1}$$

The parameters $\gamma_j > 0$ and $\delta_i > 0$ are to be chosen subject to the inequality

$$\gamma_j \delta_i P_{ij} \leq 1,$$

for all $i$ and $j$.

The first version of MART that we shall consider, MART I, uses the parameters $\gamma_j = 1$, and

$$\delta_i = 1/\max\{P_{ij}\,|j = 1, ..., J\}.$$

**Algorithm 3.2 (MART I)** *Let $x^0$ be any positive vector, and $i = k(\bmod I) + 1$. Having found $x^k$ for positive integer $k$, define $x^{k+1}$ by*

$$x_j^{k+1} = x_j^k\Big(\frac{y_i}{(Px^k)_i}\Big)^{m_i^{-1}P_{ij}}, \tag{3.2}$$

*where $m_i = \max\{P_{ij}\,|j = 1, 2, ..., J\}$.*

Some treatments of MART leave out the $m_i$, but require only that the entries of $P$ have been rescaled so that $P_{ij} \leq 1$ for all $i$ and $j$; this corresponds to the choices $\gamma_j = 1$ and

$$\delta_i = \delta = 1/\max\{P_{ij}\,|i = 1, ..., I,\ j = 1, ..., J\},$$

for each $i$. Using the $m_i$ is important, however, in accelerating the convergence of MART.

The second version of MART that we shall consider, MART II, uses the parameters $\gamma_j = s_j^{-1}$, and

$$\delta_i = 1/\max\{P_{ij}s_j^{-1}\,|j = 1, ..., J\}.$$

**Algorithm 3.3 (MART II)** *Let $x^0$ be any positive vector, and $i = k(\bmod I) + 1$. Having found $x^k$ for positive integer $k$, define $x^{k+1}$ by*

$$x_j^{k+1} = x_j^k\Big(\frac{y_i}{(Px^k)_i}\Big)^{n_i^{-1}s_j^{-1}P_{ij}}, \tag{3.3}$$

*where $n_i = \delta_i^{-1} = \max\{P_{ij}s_j^{-1}\,|j = 1, 2, ..., J\}$.*

Note that the MART II algorithm can be obtained from the MART I algorithm if we first rescale the entries of the matrix $P$, replacing $P_{ij}$ with $P_{ij}s_j^{-1}$, and redefine the vector of unknowns, replacing each $x_j$ with $x_js_j$.

The MART can be accelerated by relaxation, as well.

**Algorithm 3.4 (Relaxed MART I)** *Let $x^0$ be any positive vector, and $i = k(\bmod I) + 1$. Having found $x^k$ for positive integer $k$, define $x^{k+1}$ by*

$$x_j^{k+1} = x_j^k\Big(\frac{y_i}{(Px^k)_i}\Big)^{\tau_i m_i^{-1}P_{ij}}, \tag{3.4}$$

*where $\tau_i$ is in the interval $(0, 1)$.*

As with ART, finding the best relaxation parameters is a bit of an art.

## 3.2 Convergence of MART

In the consistent case, by which we mean that $Px = y$ has nonnegative solutions, we have the following convergence theorem for MART [1]. We assume that $s_j = \sum_{i=1}^{I} P_{ij}$ is positive, for all $j$.

**Theorem 3.1** *In the consistent case, the general MART algorithm converges to the unique non-negative solution of $Px = y$ for which the weighted cross-entropy*

$$\sum_{j=1}^{J} \gamma_j^{-1} KL(x_j, x_j^0)$$

*is minimized. The MART I algorithm converges to the unique nonnegative solution of $Px = y$ for which the cross-entropy $KL(x, x^0)$ is minimized. The MART II algorithm converges to the unique non-negative solution of $Px = y$ for which the weighted cross-entropy*

$$\sum_{j=1}^{J} s_j KL(x_j, x_j^0)$$

*is minimized.*

As with ART, the speed of convergence is greatly affected by the ordering of the equations, converging most slowly when consecutive equations correspond to nearly parallel hyperplanes.

**Open Question:** When there are no nonnegative solutions, MART does not converge to a single vector, but, like ART, is always observed to produce a limit cycle of vectors. Unlike ART, there is no proof of the existence of a limit cycle for MART.

## 3.3 Proof of Convergence for MART I

We assume throughout this proof that $\hat{x}$ is a nonnegative solution of $Px = y$. The following lemma provides an important property of the KL distance; the proof is easy and we omit it.

**Lemma 3.1** *Let $x_+ = \sum_{j=1}^{J} x_j$ and $z_+ > 0$ Then*

$$KL(x, z) = KL(x_+, z_+) + KL(x, \frac{x_+}{z_+} z). \tag{3.5}$$

For $i = 1, 2, ..., I$, let

$$G_i(x, z) = KL(x, z) + m_i^{-1} KL((Px)_i, y_i) - m_i^{-1} KL((Px)_i, (Pz)_i). \tag{3.6}$$

**Lemma 3.2** *For all $i$, we have $G_i(x, z) \geq 0$ for all $x$ and $z$.*

**Proof:** Use Equation (3.5). ∎

Then $G_i(x, z)$, viewed as a function of $z$, is minimized by $z = x$, as we see from the equation

$$G_i(x, z) = G_i(x, x) + KL(x, z) - m_i^{-1} KL((Px)_i, (Pz)_i). \tag{3.7}$$

Viewed as a function of $x$, $G_i(x, z)$ is minimized by $x = z'$, where

$$z'_j = z_j \Big( \frac{y_i}{(Pz)_i} \Big)^{m_i^{-1} P_{ij}}, \tag{3.8}$$

as we see from the equation

$$G_i(x, z) = G_i(z', z) + KL(x, z'). \tag{3.9}$$

We note that $x^{k+1} = (x^k)'$.

Now we calculate $G_i(\hat{x}, x^k)$ in two ways, using, first, the definition, and, second, Equation (3.9). From the definition, we have

$$G_i(\hat{x}, x^k) = KL(\hat{x}, x^k) - m_i^{-1} KL(y_i, (Px^k)_i). \tag{3.10}$$

From Equation (3.9), we have

$$G_i(\hat{x}, x^k) = G_i(x^{k+1}, x^k) + KL(\hat{x}, x^{k+1}). \tag{3.11}$$

Therefore,

$$KL(\hat{x}, x^k) - KL(\hat{x}, x^{k+1}) = G_i(x^{k+1}, x^k) + m_i^{-1} KL(y_i, (Px^k)_i). \tag{3.12}$$

From Equation (3.12) we can conclude several things:

- 1) the sequence $\{KL(\hat{x}, x^k)\}$ is decreasing;
- 2) the sequence $\{x^k\}$ is bounded, and therefore has a cluster point, $x^*$; and
- 3) the sequences $\{G_i(x^{k+1}, x^k)\}$ and $\{m_i^{-1} KL(y_i, (Px^k)_i)\}$ converge decreasingly to zero, and so $y_i = (Px^*)_i$ for all $i$.

Since $y = Px^*$, we can use $x^*$ in place of the arbitrary solution $\hat{x}$ to conclude that the sequence $\{KL(x^*, x^k)\}$ is decreasing. But, a subsequence converges to zero, so the entire sequence must converge to zero, and therefore $\{x^k\}$ converges to $x^*$.

Finally, since the right side of Equation (3.12) is independent of which solution $\hat{x}$ we have used, so is the left side. Summing over $k$ on the left side, we find that

$$KL(\hat{x}, x^0) - KL(\hat{x}, x^*) \tag{3.13}$$

is independent of which $\hat{x}$ we use. We can conclude then that minimizing $KL(\hat{x}, x^0)$ over all solutions $\hat{x}$ has the same answer as minimizing $KL(\hat{x}, x^*)$ over all such $\hat{x}$; but the solution to the latter problem is obviously $\hat{x} = x^*$. This concludes the proof. ∎

The proof of convergence of the general MART is similar, and we omit it. The interested reader may consult [1].

# 4 Returning to the DGP

As we have just seen, the MART can be used to minimize the function $KL(\delta, c)$, subject to linear equality constraints, provided that the matrix involved has nonnegative entries. We cannot apply the MART yet, because the matrix $A^T$ probably has some negative entries.

The entries on the bottom row of $A^T$ are all one, as is the bottom entry of the column vector $u$, since these entries correspond to the equation $\sum_{i=1}^{I} \delta_i = 1$. By adding suitably large positive multiples of this last equation to the other equations in the system, we obtain an equivalent system, $B^T\delta = s$, for which the new matrix $B^T$ and the new vector $s$ have only positive entries.

For example, the linear system of equations $A^T\delta = u$ corresponding to the posynomial in Equation (2.1) is

$$A^T\delta = u = \begin{bmatrix} -1 & 1 & 1 & 0 \\ -1 & 0 & 1 & 1 \\ -1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Adding two times the last row to the other rows, the system becomes

$$B^T\delta = s = \begin{bmatrix} 1 & 3 & 3 & 2 \\ 1 & 2 & 3 & 3 \\ 1 & 3 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}.$$

The matrix $B^T$ and the vector $s$ are now positive. We are ready to apply the MART.

The matrix $P$ is now $B^T$, and $y = s$. Selecting as our starting vector $x^0 = \delta^0 = c$, the MART converges to $\delta^*$ solving the DGP problem (if a solution exists). Using $\delta^*$, we find the optimal $t^*$ solving the GP problem.

The MART iteration is as follows. With $j = k(\text{mod}\, J+1)+1$, $\mu_i = \max\{B_{ij}\,|j = 1, 2, ..., J\}$ and $k = 0, 1, ...$, let

$$\delta_i^{k+1} = \delta_i^k \Big(\frac{s_j}{(B^T\delta^k)_j}\Big)^{\mu_i^{-1}B_{ij}}.$$

# 5 Final Comments

Minimizing $KL(\delta, c)$ over a more general set of linear equality constraints is more difficult, since we still need to find an equivalent system of linear equations that involves a matrix with only non-negative entries.

Besides the MART, there are related algorithms that can also be used here. The MART has been extended to fully simultaneous and block-iterative versions, the *simultaneous MART* (SMART) and the *rescaled block-iterative* SMART. For details, see [2].

# References


[1] Byrne, C. (2005) "Choosing parameters in block-iterative or ordered-subset reconstruction algorithms." *IEEE Transactions on Image Processing*, **14 (3)**, pp. 321–327.

[2] Byrne, C. (2005) *Signal Processing: A Mathematical Approach*, AK Peters, Publ., Wellesley, MA.

[3] Duffin, R., Peterson, E., and Zener, C. (1967) *Geometric Programming: Theory and Applications*. New York: Wiley.

[4] Gordon, R., Bender, R., and Herman, G.T. (1970) "Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and x-ray photography." *J. Theoret. Biol.* **29**, pp. 471–481.

[5] Kullback, S. and Leibler, R. (1951) "On information and sufficiency." *Annals of Mathematical Statistics* **22**, pp. 79–86.

[6] Peressini, A., Sullivan, F., and Uhl, J. (1988) *The Mathematics of Nonlinear Programming*. New York: Springer-Verlag.