LINEAR PROGRAMMING
SIMPLEX-LIKE SOLUTION TECHNIQUE
(EDGE FOLLOWER)

Definitions:

Parameters

A Mutation Geometry based linear programming problem (MGLP) consists of N variables and M constraints. In this algorithm description, the symbol " < " means less than or equal. The symbol " > " means greater than or equal.

Constraints

Constraints are numbered from 1 to M. In MGLP, the N coordinate constraints (x > 0 etc.) are treated just like any other constraint. They are numbered 1 to N. Therefore, since the first N constraints are the coordinate constraints, there are always more constraints than variables. For a maximization problem, all constraints are converted to the < type constraint (negative RHSs are allowed). For a minimization type problem, all constraints are converted to the > type constraint (again, negative RHSs are allowed). For brevity, how to treat equality constraints (a trivial matter - but one that requires a great deal of explanation), is not discussed here. Again, it is important to understand that coordinate constraints are treated just like any other constraint. Note also, that there are no such things as slack or artificial variables in MGLP - ever.

In the illustrations (EX-1 through EX-14), the constraint numbers are near the small arrows and have a small circle drawn around them.

INDEX

The INDEX is a list of the N constraints whose simultaneous solution defines the current solution vector. The INDEX is similar to the Simplex basis. The main difference is that in Simplex, variables enter and leave the basis while in MGLP, constraints enter and leave the INDEX. These constraints may be coordinate constraints or external constraints.

GAMMA VECTORS

A system of N equations (constraints treated as equations) may be written:

$$\overrightarrow{A} * \overrightarrow{X} = B$$

Where $\overrightarrow{A}$ and $\overrightarrow{X}$ are vector quantities. Each equation in the above system may be written:

$$\overrightarrow{a} * \overrightarrow{x} = b$$

Each equation defines a plane. The vector quantity $\overrightarrow{a}$ is the normal to this plane. A system of N equations defines a point in N space. A system of N-1 equations defines a line of intersection in N space. The vector cross product of N-1 normals, is perpendicular to the normals and has the direction of the line of intersection of the N-1 planes. Consequently, by taking the vector cross product of N-1 normals (constraints), the edges of the convex hull can be generated. This vector cross product is called the GAMMA vector in Mutation Geometry and is the cardinal principal of Mutation Geometry based linear programming.

In this paper, GAMMA vectors are calculated using column cofactors. There are many faster ways to calculate them. The simple methods illustrated in this description would not be used in a practical implementation.

T - VALUES

T-values are the parametric distance between hyperplanes (convex hull vertices), along the direction of the GAMMA vectors. During each iteration, a T-value is computed for each constraint (including of course the N coordinate constraints). The corresponding constraint that gave the minimum non zero T-value will be the closest hyperplane. A T-value of zero defines a degenerate constraint system. It will occur when more than N constraints pass through the same point. How to handle such a condition is a trivial matter which for brevity will not be discussed here. It should be pointed out, however, that the resolution of a degeneracy is a deterministic process - and continual cycling is not possible. Also, the resolution of a degeneracy does not involve perturbing the constraint matrix in any way.

A T-value is computed for each of the M constraints which are not in the INDEX. The formula for computing T-values is given below. The derivation of this formula is given elsewhere in this paper.

$$T_i = (b_i - \vec{a_i} * \vec{r_o})/(\vec{a_i} * \vec{g})$$

Where i = 1, 2, 3, ... M constraints

b_i = the right hand side of constraint i

$\vec{a_i}$ = the coefficient vector of constraint i

$\vec{r_o}$ = the current solution vector

$\vec{g}$ = the current GAMMA vector

$*$ = the vector scalar product

The minimum T-value is used each iteration to calculate a new solution vector. The equation used to calculate the new solution vector is given below:

$$\vec{r} = \vec{r_o} + T \vec{g}$$

where $\vec{r}$ = the new solution vector

$\vec{r_o}$ = the current solution vector

T = the minimum T-value found above

$\vec{g}$ = the current GAMMA vector

A more detailed explanation of T-value computation is given below in the algorithm description area.

The beginning:

SOLUTION ALGORITHM

There are several MGLP techniques for finding a feasible starting solution. Since they all utilize techniques discussed thus far, and the description of them would be very long, for brevity, we start the discussion of the MGLP algorithm with a very simple starting solution technique that would not be used in practice but will illustrate the principles involved.

In describing the MGLP algorithm I will refer to the sheets labeled EX-1 through EX-14. This is only a two variable problem but I believe the "graphic" will be invaluable in describing the action (at least for me)!

We start the description with a statement of the problem - a minimization problem with two variables and nine (two coordinate plus seven external) constraints (see page EX-1 of the illustrated example).

STARTING POINT

We start with a statement of the initial conditions (see page EX-1):

Initial solution vector: $\vec{r}_o$ = (0 0)

Initial INDEX = (1 , 2) (coordinate constraints)

Select (any) N-1 coordinate constraints to form a GAMMA vector. We select constraint number two.

constraint 2: (0 1) > 0

Using the left hand side of constraint number two, form the GAMMA vector (vector cross product) by column cofactors

GAMMA vector $\vec{g}$ = (1 0)

Notice that the GAMMA vector (labeled $\vec{g}$, page EX-1) points along the direction of the positive x axis.

Using the T-value formula, compute the intersection points to all other constraints that are NOT in the INDEX. These intersections are along the positive x axis. (SEE PAGE EX-1, EX-2).

T_i = (b_i - $\vec{a}_i$ * $\vec{r}_o$)/($\vec{a}_i$ * $\vec{g}$)

i = 3 , 4 , 5 , 6 , 7 , 8 , 9

(constraints 1 , 2 are in the INDEX)

T_1 = ignore (INDEX constraint)

T_2 = ignore (INDEX constraint)

T_3 = (7 - (1 7) * (0 0))/((1 7) * (1 0)) = 7

T_4 = (12 - (2 6) * (0 0))/((2 6) * (1 0)) = 6

T_5 = (15 - (3 5) * (0 0))/((3 5) * (1 0)) = 5

T_6 = (16 - (4 4) * (0 0))/((4 4) * (1 0)) = 4

T_7 = (15 - (5 3) * (0 0))/((5 3) * (1 0)) = 3

T_8 = (12 - (6 2) * (0 0))/((6 2) * (1 0)) = 2

T_9 = (7 - (7 1) * (0 0))/((7 1) * (1 0)) = 1

In order to find a feasible starting solution, we have to pick a point that satisfies all constraints. For a pure minimization problem, such as this example, that point lies along the axis and can be found by selecting the largest T-value. In this example, that point is at the intersection of constraints 2 and 3. Therefore, the constraint leaving the INDEX is constraint number 1, and the constraint entering the INDEX is constraint number 3. Using the vector equation:

$$\vec{r} = \vec{r_0} + T\,\vec{g}$$

$$= (\;0 \quad 0\;) + 7\,(\;1 \quad 0\;)$$

$$= (\;7 \quad 0\;)$$

We generate a starting vector which satisfies all constraints (see page EX-2). The initial objective function value is computed from:

$$z = \vec{r} * \overrightarrow{obj}$$

$$= (\;7 \quad 0\;) * (\;1 \quad 1\;)$$

$$= 7$$

We now have a feasible starting point and all the information we need to start the solution process. This information is repeated below and is also shown on page EX-2.

Starting solution vector: $\vec{r} = (\;7 \quad 0\;)$

Starting INDEX: INDEX = 2 , 3

Starting z: $z = 7$

ITERATION 1

The first thing we must do is to select the N-1 constraints from the N constraints in the INDEX to form the GAMMA vectors. The general rule is:

1) Always retain the latest constraint to enter the INDEX when computing GAMMA vectors

2) Calculate the N-1 GAMMA vectors by ignoring in turn each of the remaining N-1 constraints in the INDEX

In our example, N = 2. Therefore, there will only be one GAMMA vector. Since constraint 3 was the last to enter the INDEX, we ignore constraint 2 and calculate the column cofactors of the left hand side to get the GAMMA vector:

Constraint 3: 1 7 > 7

GAMMA vector: $\vec{g} = (\;7 \quad -1\;)$

We now have to test this GAMMA vector to ensure that it is going in the correct direction. If the GAMMA vector is not pointed in the correct direction, it must be multiplied by -1 to change its direction. For minimization type problems, the following condition must hold; THE SCALAR PRODUCT OF THE GAMMA VECTOR AND THE LEFT HAND SIDE OF THE CONSTRAINT LEFT OUT OF THE INDEX WHEN THE GAMMA VECTOR WAS COMPUTED, MUST BE GREATER THAN ZERO. Since constraint number 2 was left out in computing this GAMMA vector, the following condition must hold:

$$(\text{GAMMA vector } \vec{g}) * \overrightarrow{(\text{constraint number } 2)} > 0 \ ?$$

$$(7 \quad -1) * (0 \quad 1) = -1$$

Since this GAMMA vector fails the test, it must be multiplied by -1 to change its direction. The correct GAMMA vector is then:

$$\text{GAMMA vector } \vec{g} = (-7 \quad 1)$$

Notice that this GAMMA vector lies along constraint number 3 (page EX-4).

A GAMMA vector test value of zero indicates that the constraint hyperplanes are orthogonal. In that situation, this test cannot be used to determine proper GAMMA vector direction. There are several other trivial ways to test for proper GAMMA vector orientation, but for the sake of brevity, they will not be discussed here.

We next compute the parametric distance to all constraint hyperplanes that are not in the INDEX via the T-value equation. The T-value for constraint number 1 is computed below.

$$\text{INDEX} = 2 , 3$$

$$\text{Solution vector } \vec{r_o} = (7 \quad 0)$$

$$\text{GAMMA vector } \vec{g} = (-7 \quad 1)$$

$$T_i = (b_i - \vec{a_i} * \vec{r_o}) / (\vec{a_1} * \vec{g})$$

$$T_1 = (0-(1 \quad 0) * (7 \quad 0))/((1 \quad 0) * (-7 \quad 1)) = 1.000$$

I refer you to page EX-3, EX-4, and EX-5 for the complete picture. Notice that constraint number 4 yielded the smallest T-value greater than zero. That constraint will be the next constraint entering the INDEX. Also notice that constraint number 2 will be the constraint leaving the INDEX. At the bottom of page EX-3 you will notice that the solution vector has been updated with the vector equation:

$$\vec{r} = \vec{r_o} + T\,\vec{g}$$

$$= (\ 7 \quad 0\ ) + 0.250\ (\ -7 \quad 1)$$

$$= (\ 5.250 \quad 0.250\ )$$

Also notice that the objective function value has become smaller thus indicating that we are going in the right direction. We have now updated all variables and are ready to proceed to the next iteration. Page EX-5 summarizes the situation at the end of iteration 1.

ITERATION 2

Iteration 2 (see EX-6 through EX-8) follows the same format as iteration 1;

Compute a GAMMA vector

Test (and correct if necessary) the GAMMA vector direction

Compute the T-values for all constraints not in the INDEX using the T-value equation

Select the minimum T-value

Update the solution vector using the $\vec{r}$ vector equation

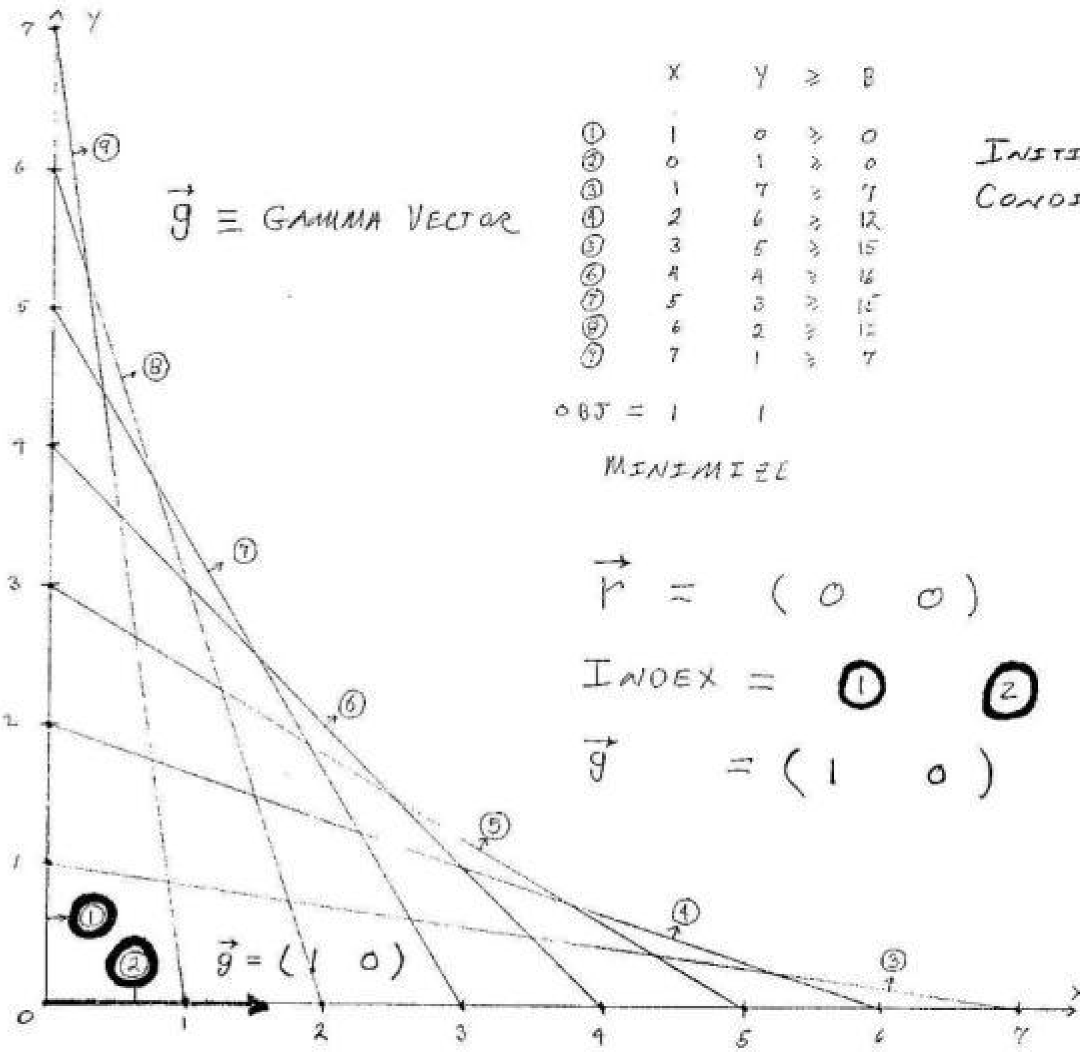
Calculate the new z value using the objective function vector and the new solution vector

Test the new z value against the old z value to see if the optimum has been reached. If so, stop.

Update the INDEX

Continue on to the next iteration

This process is repeated until the optimum is found. In the worked example, the optimum is found at iteration 4 (page EX-13 and EX-14).



$\vec{g} \equiv$ GAMMA VECTOR

	X	Y	≥	B
①	1	0	≥	0
②	0	1	≥	0
③	1	7	≥	7
④	2	6	≥	12
⑤	3	5	≥	15
⑥	4	4	≥	16
⑦	5	3	≥	15
⑧	6	2	≥	12
⑨	7	1	≥	7
OBJ =	1	1		

MINIMIZE

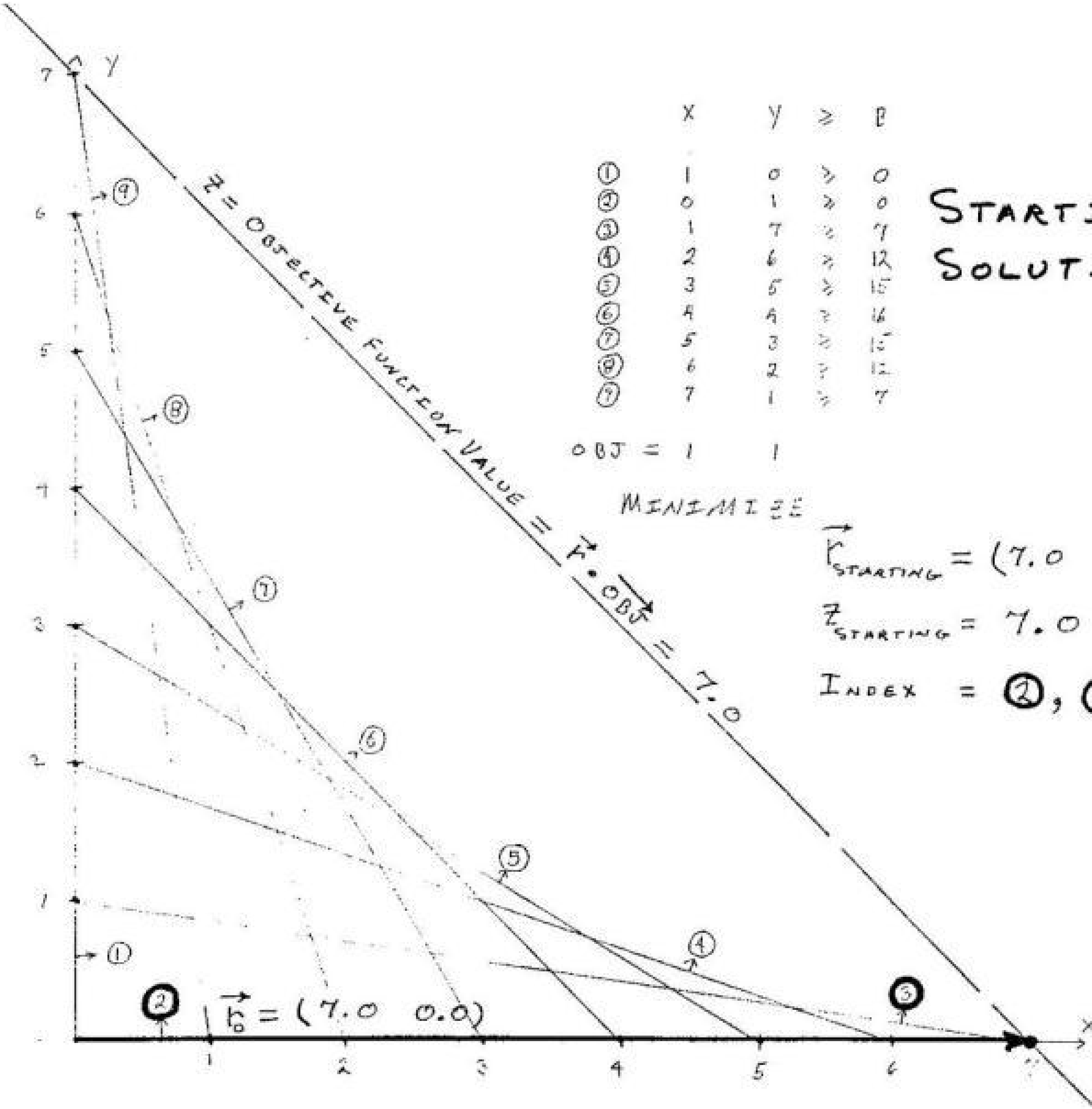
INITIAL CONDITIONS

$\vec{r} = (0 \quad 0)$

INDEX = 1 2

$\vec{g} = (1 \quad 0)$

EX-1



STARTING SOLUTION

	x	y	≥	B
1	1	0	≥	0
2	0	1	≥	0
3	1	7	≥	7
4	2	6	≥	12
5	3	5	≥	15
6	4	4	≥	16
7	5	3	≥	15
8	6	2	≥	12
9	7	1	≥	7
OBJ =	1	1		

MINIMIZE

$\vec{r}_{STARTING} = (7.0 \quad 0.0)$

$Z_{STARTING} = 7.0$

INDEX = 2, 3

EX-2

*************** NEW ITERATION ***************

ITERATION COUNT = 1

GAMMA VECTOR NUMBER = 1

* GAMMA MATRIX = 1.0 7.0 (ORIGINAL)

* GAMMA MATRIX = 1.0 7.0 (REDUCED)

GAMMA VECTOR g = 7.0 -1.0 COLUMN COFACTORS OF GAMMA MATRIX

GAMMA VECTOR TEST = -1.0

GAMMA VECTOR g = -7.0 1.0

GRADIENT = -6.0

*** COMPUTED FROM

$$T_i = (b_i - \vec{a}_i \cdot \vec{r}_0)/(\vec{a}_i \cdot \vec{g})$$

	CONSTRAINT NUMBER =	1	T VALUE =	1.000
***	CONSTRAINT NUMBER =	4	T VALUE =	0.250 (min)
	CONSTRAINT NUMBER =	5	T VALUE =	0.375
	CONSTRAINT NUMBER =	6	T VALUE =	0.500
	CONSTRAINT NUMBER =	7	T VALUE =	0.625
	CONSTRAINT NUMBER =	8	T VALUE =	0.750
	CONSTRAINT NUMBER =	9	T VALUE =	0.875

INCOMING CONSTRAINT = 4

** COMPUTED FROM $\vec{r} = \vec{r}_0 + T\vec{g}$

OUTGOING CONSTRAINT = 2

NEW INDEX = 3 4

** NEW r VECTOR = 5.250 0.250

NEW OBJECTIVE FUNCTION VALUE = 5.50

* IN A TWO VARIABLE PROBLEM, THE <u>ORIGINAL</u> GAMMA MATRIX (a 1x2 IN THIS EXAMPLE) WILL BE THE SAME AS A <u>REDUCED</u> GAMMA MATRIX

$T_i = (b_i - \vec{a}_i \cdot \vec{r}_0) / (\vec{a}_i \cdot \vec{g})$

SELECT $T_i > 0$

GRADIENT = $\vec{g} \cdot \overrightarrow{OBJ}$

	X	Y	≥	B
①	1	0	≥	0
②	0	1	≥	0
③	1	7	≥	7
④	2	6	≥	12
⑤	3	5	≥	15
⑥	4	4	≥	16
⑦	5	3	≥	15
⑧	6	2	≥	12
⑨	7	1	≥	7

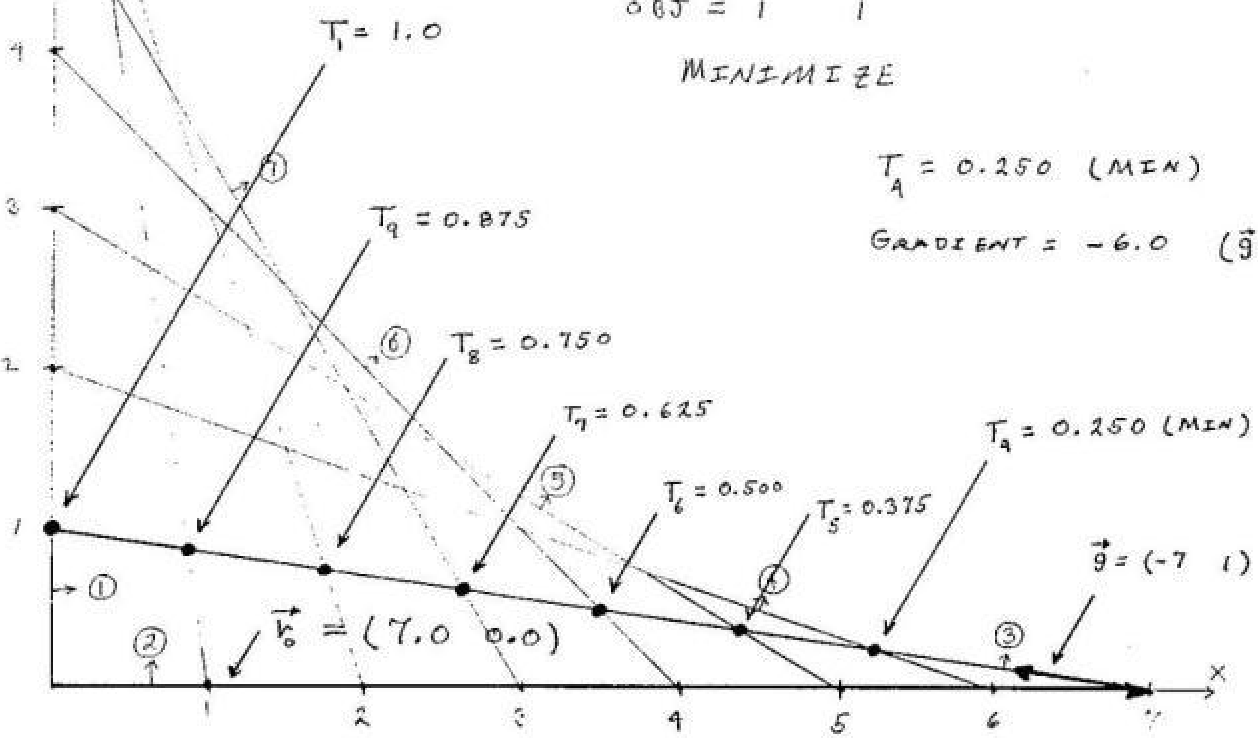
OBJ = 1 1

MINIMIZE

ITERATION # 1

$T_4 = 0.250$ (MIN)

GRADIENT = -6.0 $(\vec{g} \cdot \overrightarrow{OBJ})$



EX-4

$$\vec{r} = \vec{r}_o + T_{MIN}\,\vec{g}$$

$$= (7 \;\; 0) + 0.250\,(-7 \;\; 1)$$

$$= (5.25 \;\; 0.25)$$

	X	Y	≥	B
1	1	0	≥	0
2	0	1	≥	0
3	1	7	≥	7
4	2	6	≥	12
5	3	5	≥	15
6	4	4	≥	16
7	5	3	≥	15
8	6	2	≥	12
9	7	1	≥	7

OBJ = 1 1

MINIMIZE

ITERATION # 1 (CONTINUED)

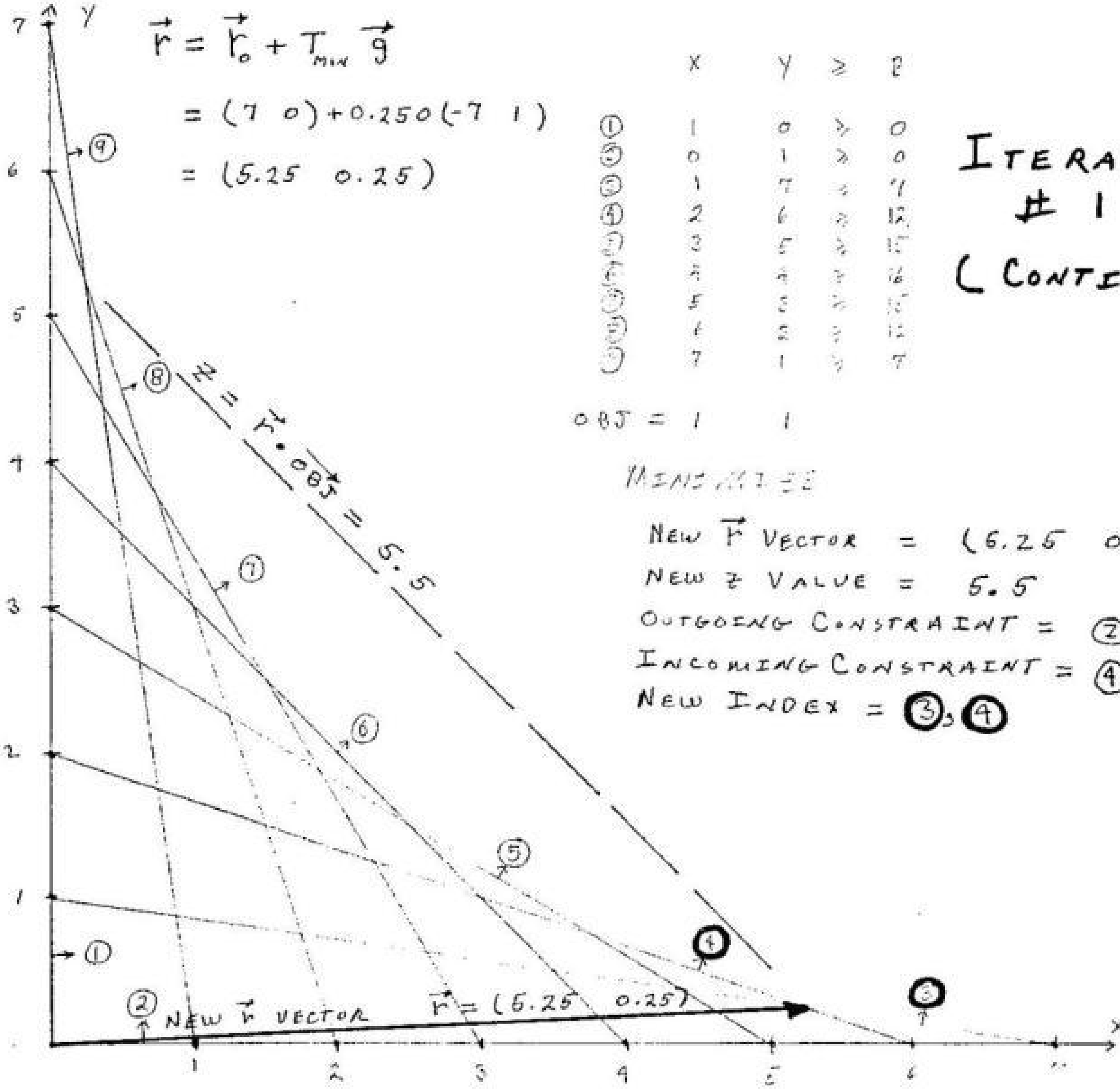
NEW $\vec{r}$ VECTOR = (5.25 0.25)

NEW Z VALUE = 5.5

OUTGOING CONSTRAINT = 2

INCOMING CONSTRAINT = 4

NEW INDEX = 3, 4



EX-5

EX-6

*************** NEW ITERATION ***************

ITERATION COUNT = 2

GAMMA VECTOR NUMBER = 1

* GAMMA MATRIX = 2.0 6.0

* GAMMA MATRIX = 2.0 6.0

GAMMA VECTOR g = 6.0 -2.0

GAMMA VECTOR TEST = -8.0

GAMMA VECTOR g = -6.0 2.0

GRADIENT = -4.0

CONSTRAINT NUMBER	=	1	T VALUE =	0.875	
CONSTRAINT NUMBER	=	2	T VALUE =	-0.125	
CONSTRAINT NUMBER	=	5	T VALUE =	0.250	(min)
CONSTRAINT NUMBER	=	6	T VALUE =	0.375	
CONSTRAINT NUMBER	=	7	T VALUE =	0.500	
CONSTRAINT NUMBER	=	8	T VALUE =	0.625	
CONSTRAINT NUMBER	=	9	T VALUE =	0.750	

INCOMING CONSTRAINT = 5

OUTGOING CONSTRAINT = 3

NEW INDEX = 4 5

NEW r VECTOR = 3.750 0.750

NEW OBJECTIVE FUNCTION VALUE = 4.50

* SEE P3

ITERATION #2

$T_i = (b_i - \vec{a}_i \cdot \vec{r}_o)/(\vec{a}_i \cdot \vec{g})$

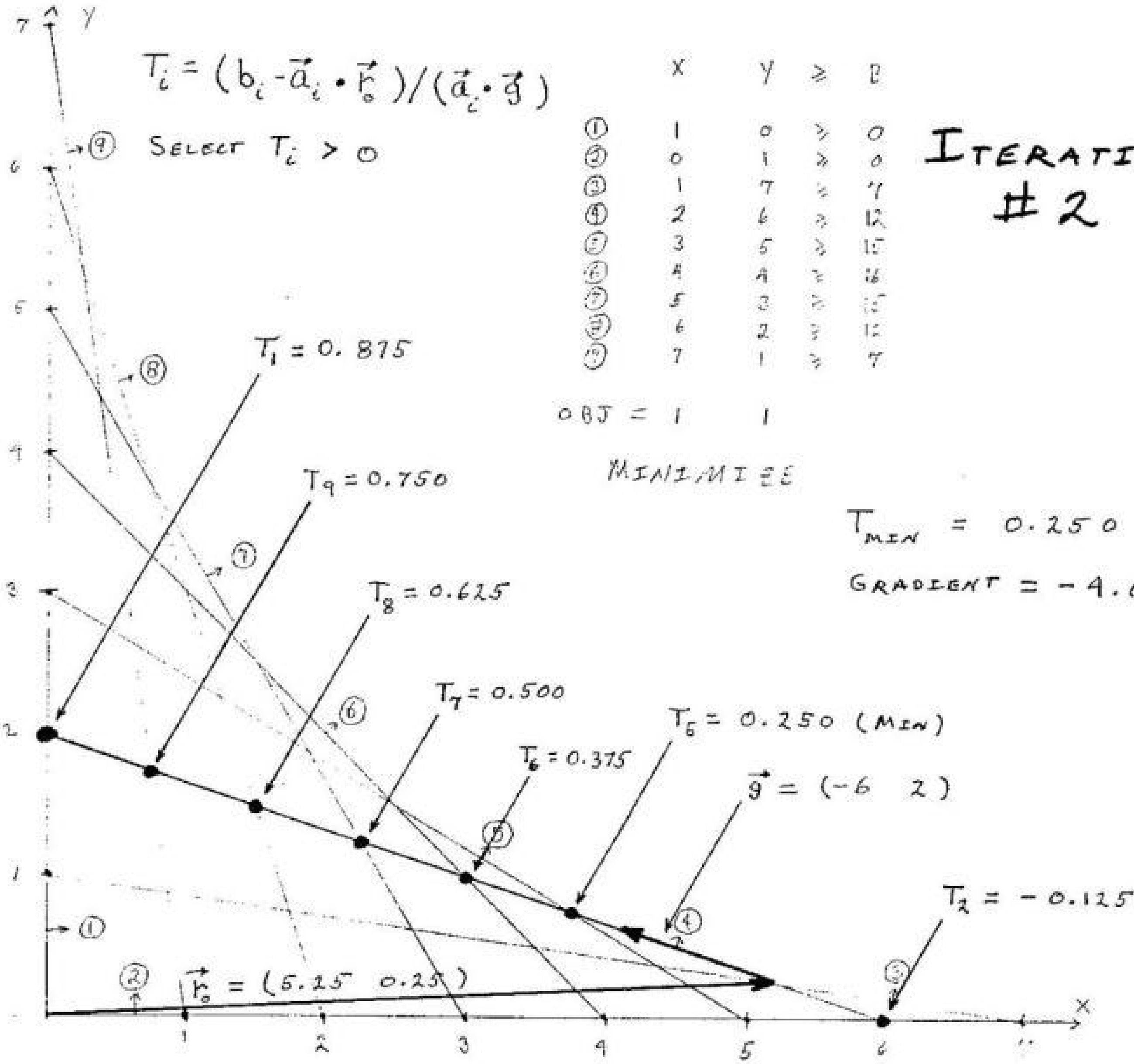
SELECT $T_i > 0$

	X	Y	≥	B
1	1	0	≥	0
2	0	1	≥	0
3	1	7	≥	7
4	2	6	≥	12
5	3	5	≥	15
6	4	4	≥	16
7	5	3	≥	15
8	6	2	≥	12
9	7	1	≥	7
OBJ =	1	1		

MINIMIZE

$T_{MIN} = 0.250$

GRADIENT = -4.0



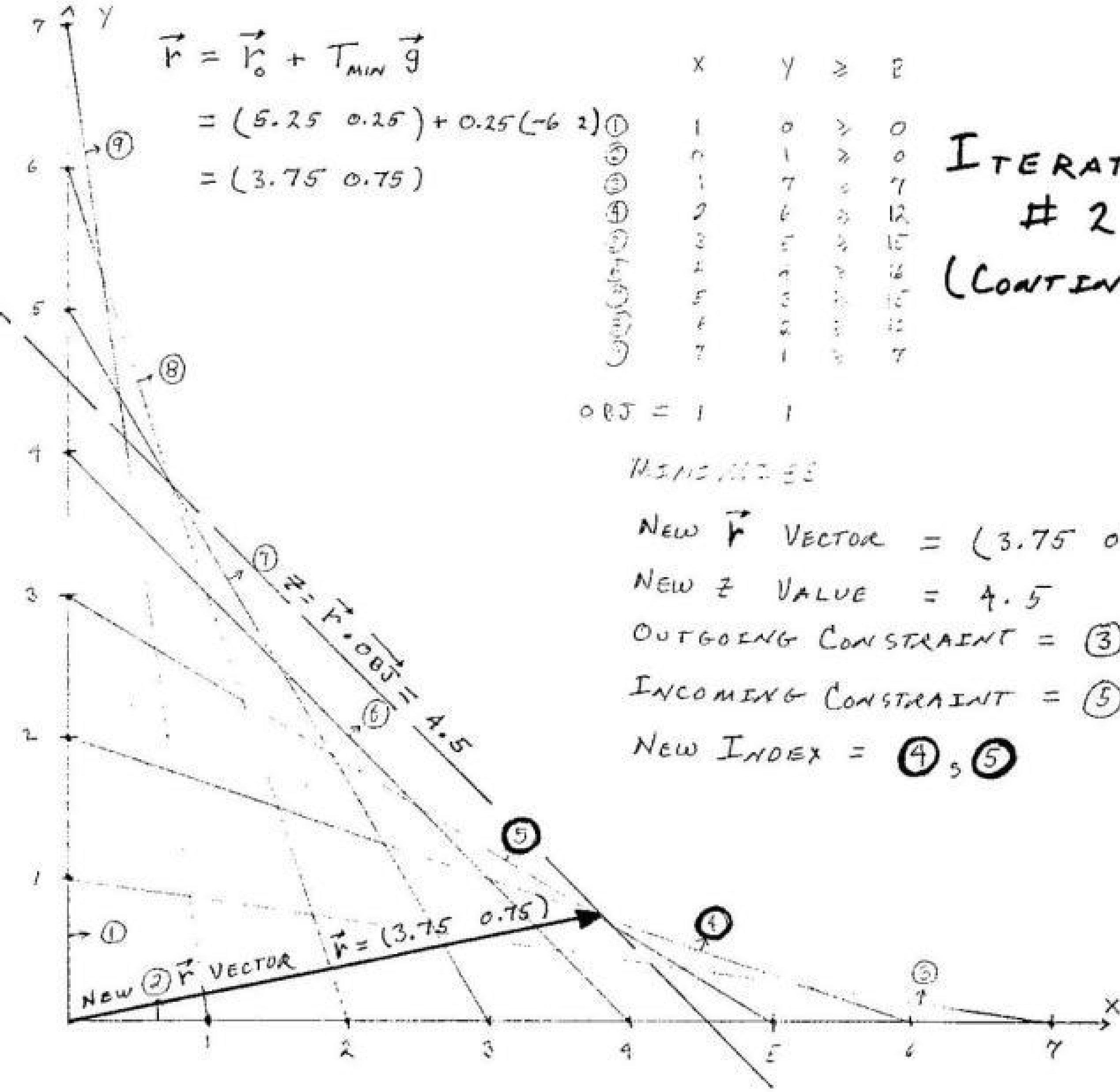
EX-7

ITERATION # 2 (CONTINUED)

$$\vec{r} = \vec{r}_0 + T_{MIN}\,\vec{g}$$
$$= (5.25 \quad 0.25) + 0.25(-6 \quad 2)$$
$$= (3.75 \quad 0.75)$$

	x	y	≥	b
①	1	0	≥	0
②	0	1	≥	0
③	1	7	≥	7
④	2	6	≥	12
⑤	3	5	≥	15
⑥	4	4	≥	16
⑦	5	3	≥	15
⑧	6	2	≥	12
⑨	7	1	≥	7
OBJ =	1	1		

MINIMIZE



NEW $\vec{r}$ VECTOR = (3.75 0.75)

NEW z VALUE = 4.5

OUTGOING CONSTRAINT = ③

INCOMING CONSTRAINT = ⑤

NEW INDEX = ④, ⑤

EX-8

EX-9

*************** NEW ITERATION ***************

ITERATION COUNT	=	3		
GAMMA VECTOR NUMBER	=	1		
* GAMMA MATRIX	=	3.0	5.0	
* GAMMA MATRIX	=	3.0	5.0	
GAMMA VECTOR g	=	5.0	-3.0	
GAMMA VECTOR TEST	=	-8.0		
GAMMA VECTOR g	=	-5.0	3.0	
GRADIENT	=	-2.0		
CONSTRAINT NUMBER	=	1	T VALUE =	0.750
CONSTRAINT NUMBER	=	2	T VALUE =	-0.250
CONSTRAINT NUMBER	=	3	T VALUE =	-0.125
CONSTRAINT NUMBER	=	6	T VALUE =	0.250 (min)
CONSTRAINT NUMBER	=	7	T VALUE =	0.375
CONSTRAINT NUMBER	=	8	T VALUE =	0.500
CONSTRAINT NUMBER	=	9	T VALUE =	0.625
INCOMING CONSTRAINT	=	6		
OUTGOING CONSTRAINT	=	4		
NEW INDEX	=	5	6	
NEW r VECTOR	=	2.500	1.500	

NEW OBJECTIVE FUNCTION VALUE = 4.0

* See P3

ITERATION # 3

$$T_i = (b_i - \vec{a}_i \cdot \vec{r}_0)/(\vec{a}_i \cdot \vec{g})$$

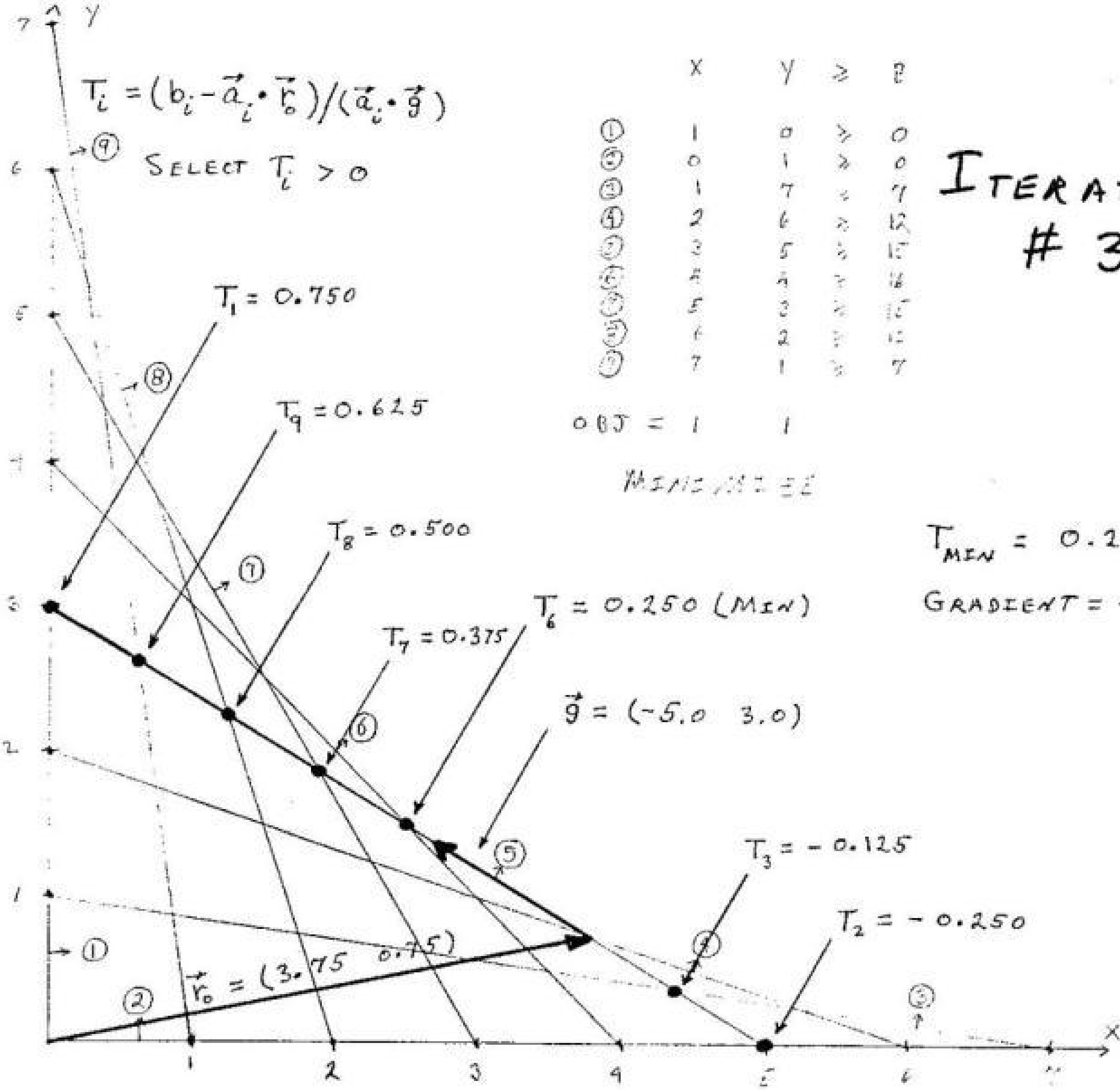
SELECT $T_i > 0$

	X	Y	≥	B
1	1	0	≥	0
2	0	1	≥	0
3	1	7	≥	7
4	2	6	≥	12
5	3	5	≥	15
6	4	4	≥	16
7	5	3	≥	15
8	6	2	≥	12
9	7	1	≥	7
OBJ =	1	1		

MINIMIZE

$T_{MIN} = 0.250$

GRADIENT = −2.0



EX-10

$$\vec{r} = \vec{r}_0 + T_{MIN}\ \vec{g}$$
$$= (3.75\ \ 0.75) + 0.250\,(-5.0\ \ 3.0)$$
$$= (2.5\ \ 1.5)$$

ITERATION #3 (CONTINUED)

	X	Y	≥	B
1	1	0	≥	0
2	0	1	≥	0
3	1	7	≥	7
4	2	6	≥	12
5	3	5	≥	15
6	4	4	≥	16
7	5	3	≥	15
8	6	2	≥	12
9	7	1	≥	7
OBJ =	1	1		

MINIMIZE

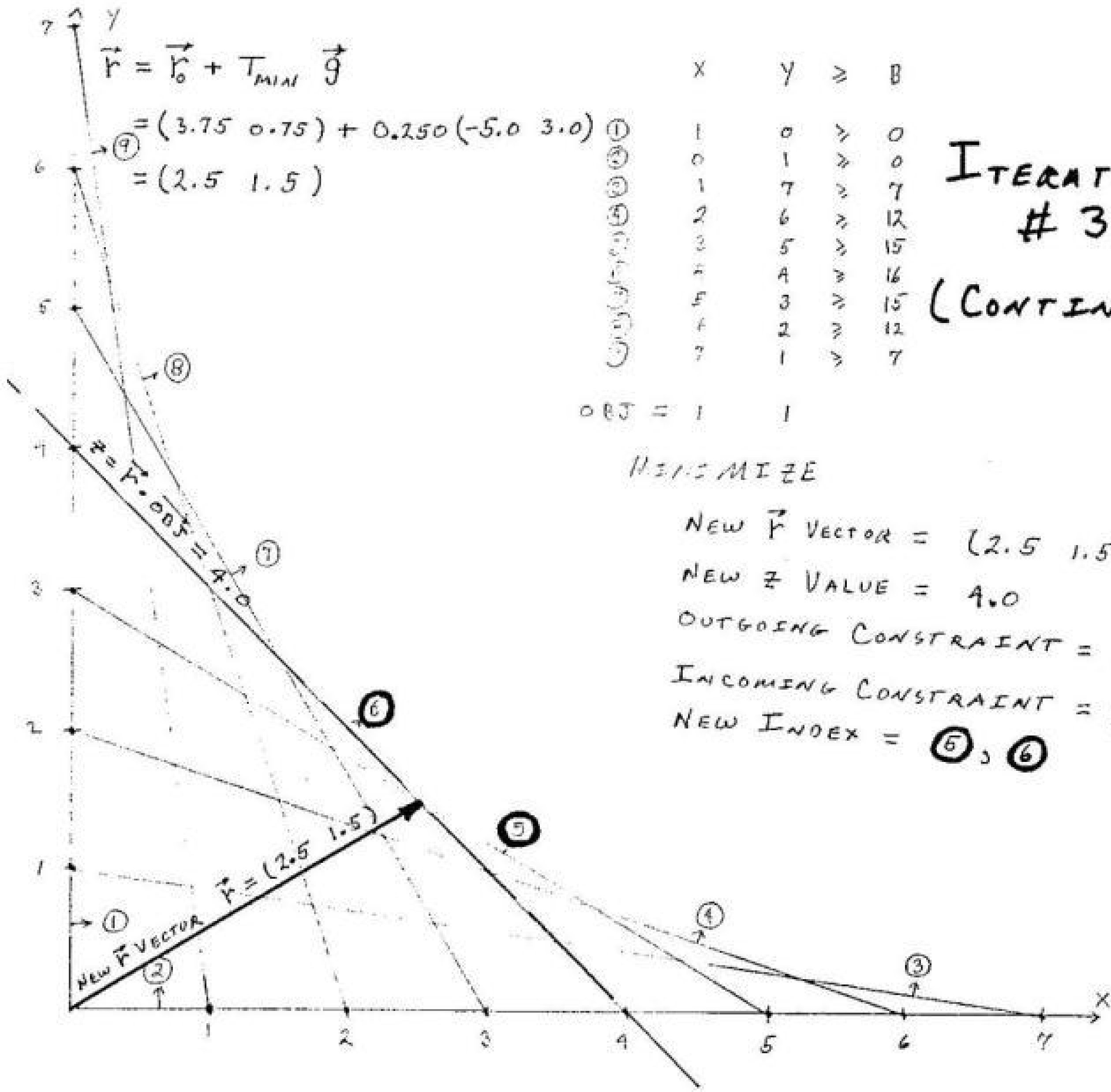
NEW $\vec{r}$ VECTOR = (2.5 1.5)

NEW Z VALUE = 4.0

OUTGOING CONSTRAINT = 4

INCOMING CONSTRAINT = 6

NEW INDEX = 5, 6



EX-12

*************** NEW ITERATION ***************

ITERATION COUNT	=	4	
GAMMA VECTOR NUMBER	=	1	
* GAMMA MATRIX	=	4.0	4.0
* GAMMA MATRIX	=	4.0	4.0
GAMMA VECTOR g	=	4.0	-4.0
GAMMA VECTOR TEST	=	-8.0	
GAMMA VECTOR g	=	-4.0	4.0
GRADIENT	=	0.0	

********** OPTIMAL SOLUTION FOUND **********

********** ALTERNATE OPTIMAL SOLUTION EXISTS **********

* SEE P 3

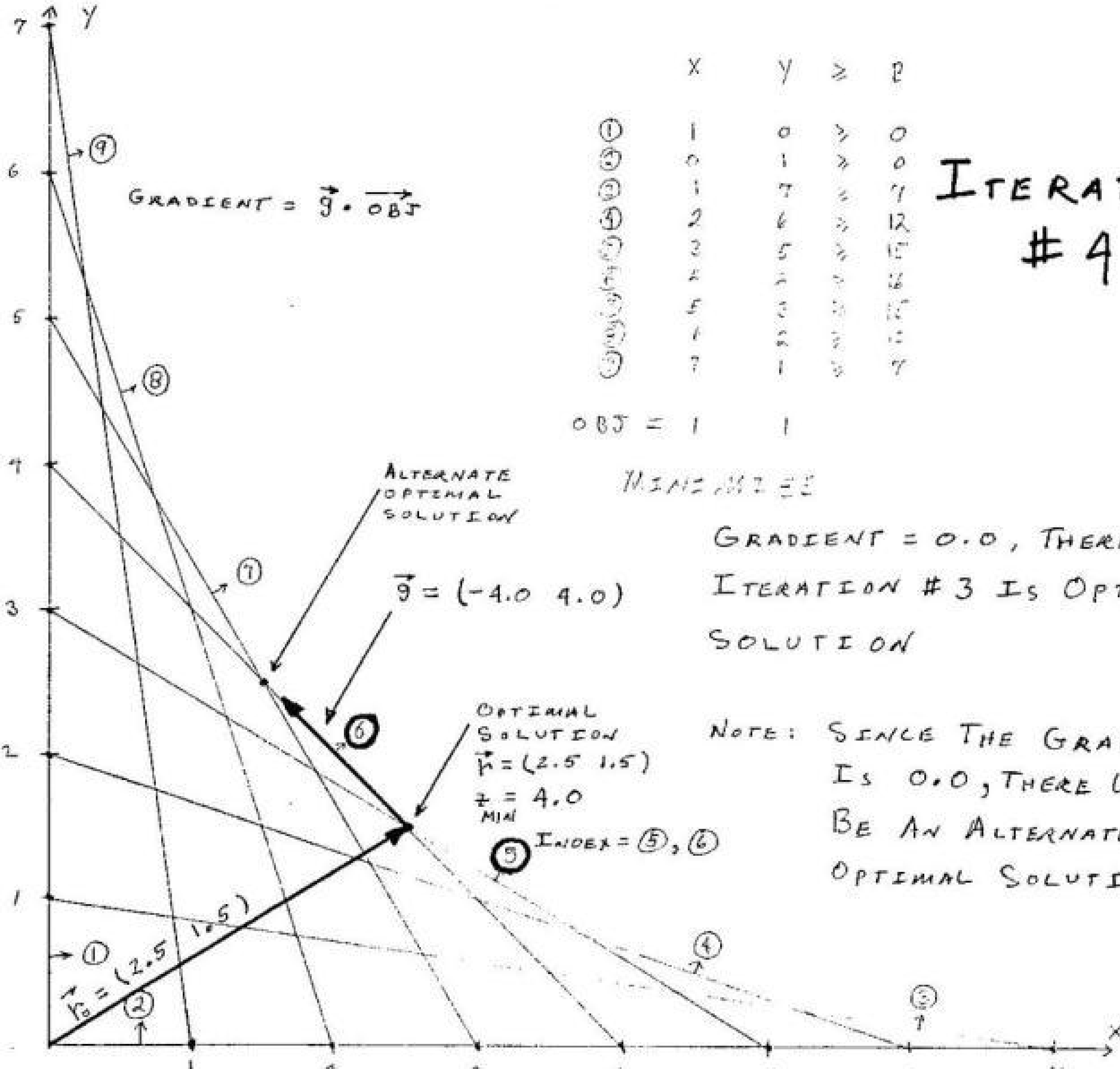
ITERATION #4

	x	y	≥	b
1	1	0	≥	0
2	0	1	≥	0
3	1	7	≥	7
4	2	6	≥	12
5	3	5	≥	15
6	4	4	≥	16
7	5	3	≥	15
8	6	2	≥	12
9	7	1	≥	7
OBJ =	1	1		

MINIMIZE

GRADIENT = 0.0, THEREFORE ITERATION #3 IS OPTIMAL SOLUTION

NOTE: SINCE THE GRADIENT IS 0.0, THERE WILL BE AN ALTERNATE OPTIMAL SOLUTION



EX-13

EX-14

********** OPTIMAL SOLUTION **********

CPU TIME	=	0:00:00.14
NUMBER OF ITERATIONS	=	4
INDEX CONSTRAINT NUMBER	=	5
INDEX CONSTRAINT NUMBER	=	6
OPTIMUM r VECTOR	=	2.50 1.50
OBJECTIVE FUNCTION VALUE	=	4.0
MINIMUM SLACK	=	-6.0
MAXIMUM SLACK	=	-12.0