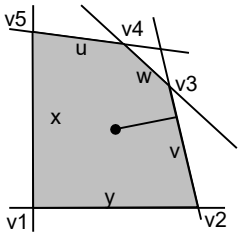
Simplex Attempt 2 PhL 11.27.04

[This document is not the final version, but it is reasonably good and contains useful proofs of things that are not included inthe final version, so we shall keep this document.]

A Prototype Problem

Consider a problem in a physical space E^2 with three ≤ inequalities. Here is a picture:

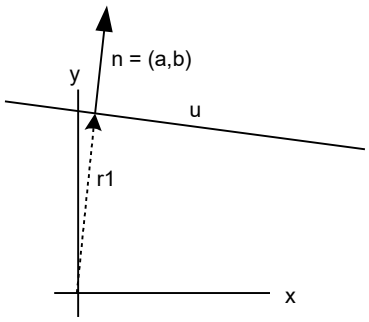


We have the usual x and y axes, and the shaded area is the feasible region for (x,y). The x axis points to the right as usual (from v1 to v2), but we label that boundary with the letter y since the boundary is y = 0. Similarly the boundary labeled w is the line that makes w = 0.

The point shown as a black dot has some coordinates (x,y) but also has coordinates (u,v,w) which are the distances from the point to the boundaries. An arrow marks the distance from the v boundary to the point, for example. The variables u,v,w are the so-called slack variables that cause the boundary inequalities to become equalities, in the usual manner. Inside the feasible region, all 5 variables are positive. Obviously, if we regard x and y as the 2 independent variables, then u,v,w are all dependent variables. The feasibility region is a 2D surface in E^2 and the polygon around this region is a polygon in E^2 . The dimension here 2 is usually called n, and the physical space is then E^n. Notice that the polygon has 5 vertices in our example, which we have labeled v1 through v5. The solution to our problem, to min or max some lincom of x and y, which we might call M(x,y), will always be at one of the vertices -- this is a fundamental theorem that we are not proving here.

Conventions for the boundary lines

In the above picture, the u boundary is defined by au*x + bu*y = cu, and the shaded side is ≤ cu, but for our discussion here we remove the "u" labels. The equation a*x + b*y = c describes a line with normal vector **n** = (a,b). Here is a picture:

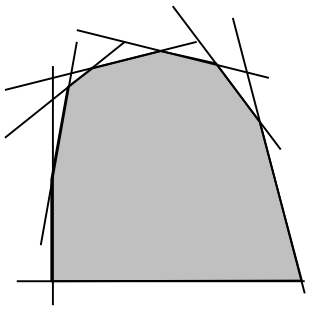


We shall require that our equation ax + by = c is signed and scaled such that a > 0 and $a^2 + b^2 = 1$. This can certainly be done without any "loss of generality". Then the vector **n** = (a,b) is a unit length normal vector normal to the line. We know this because any line can be described by **n•r** = constant, and in the above case, we have **n•r** = ax + by = c. If we select **r** = $\mathbf{r_1}$ as shown, then $\mathbf{n \bullet r_1} = r_1$. We may conclude that r_1 = c is the distance from the origin to the closest approach of the line to the origin.

If we consider any line that is (1) signed by our convention just noted, a>0, and (2) which has a positive y intercept y_0, then we know that c > 0. The reason is that c = **n•r** = $ay_0 > 0$.

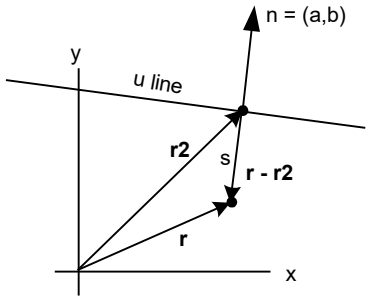
Notice that we have said nothing about the sign of b, it can be either sign and the above conclusions are true, so c > 0 whether the line tilts to the right or to the left.

We shall assume that all our boundary line equations are expressed with these conventions, so that all three c values cu,cv and cw are positive in our prototype problem above. Just to clarify the point, here is another "problem" with 6 boundary lines, and for each of these lines, we have c > 0.



Clarification of the meaning of a "slack variable"

Consider this picture:



What is the distance s from some arbitrary point **r** to the boundary line? By this we mean the distance of closest approach of the boundary line to point **r**. Here is a simple calculation to get the answer. Point $\mathbf{r}_2$ is selected on the line so that the vector $\mathbf{r}_2-\mathbf{r}$ is lined up with the line's (unit length) normal **n**. We then have

$$\mathbf{r}_2-\mathbf{r} = s\mathbf{n}$$

If we dot both sides with **n**, we know that $\mathbf{n}\bullet\mathbf{r}_2 = c$ since $\mathbf{r}_2$ is on the line. So we get

$$c - \mathbf{n}\bullet\mathbf{r} = s \qquad \Rightarrow \qquad s = -ax - ay + c$$

This distance $s \geq 0$ is called "the slack variable" for this boundary line and for the u boundary line, it is called u in the discussion below. We will have similar slack variables w and v for the other two boundary lines. We have just shown that a slack variable *is* the distance from a point to the boundary line, providing we respect our conventions above.

A 3x3 System of Equations

Our boundary inequalities are these (with conventions as noted above; note we use two-letter constants.)

```
au*x + bu*y  ≤ cu
av*x + bv*y  ≤ cv
aw*x + bw*y  ≤ cw
```

We define three positive slack variables u,v and w which convert these inequalities to equalities,

```
u = cu - au*x - bu*y
v = cv - av*x - bv*y
w = cw - aw*x - bw*y
```

where, as noted earlier, the three c's are positive. We can write this in matrix form as follows

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} cu - au*x - bu*y \\ cv - av*x - bv*y \\ cw - aw*x - bw*y \end{bmatrix}$$

This is a 3x3 system of the form A**q**=**s** where the matrix A is the identity matrix, so we really have **q** = **s** . We can regard the three columns of this matrix A=I as forming **basis vectors in E^3**. This is an orthogonal basis and we could then express any column of any 3x3 matrix as a lincom of these basis vectors. More abstractly, we can express any column of three numbers as a lincom of these basis vectors.

This basis in E^3 should not in any way be confused with the basis (x,y) that we used in E^2 above. The space (x,y) is a completely different space from our (u,v,w) space. Each space of course can have one or more bases, so we use the same term "basis" when talking about either of these spaces.

The Mapping from E^2 to E^3.

Now, for a given (x,y) in E^2 we can solve the above equations for (u,v,w) in E^3. Here is a better way to write the above equations to highlight this mapping:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} -au & -bu \\ -av & -bv \\ -aw & -bw \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} cu \\ cv \\ cw \end{bmatrix} \qquad \mathbf{u} = M\mathbf{x} + \mathbf{c}$$

where we show vector notation on the right. We might express this mapping as f: $E^2 \rightarrow E^3$. We want to learn a few things about this mapping.

A. Show that the entire x,y E^2 plane maps into a plane in E^3

First, can we find a fixed vector **n** in E^3 and a constant D such that **n**•**u** = D for all **u** generated by our mapping? If we can, then the plane of E^2 maps into a plane in E^3 described by the equation **n**•**u** = D. This is a plane that has normal vector **n**, and which is displaced distance D from the origin.

From our mapping equation, **n**•**u** = **n**•(M**x**) + **n**•**c** . Now consider:

n•(M**x**) = n_1 (-au*x -bu*y) + n_2 (-av*x -bv*y) + n_3 (-aw*x -bw*y)

Then

- **n**•(M**x**) = [n_1*au + n_2*av + n_3*aw] x + [n_1*bu + n_2*bv + n_3*bw] y

= **n**•**a** x + **n**•**b** y **a** = (au,av,aw) and similar for b

Now as long as vectors **a** and **b** do not align, we can select **n** to be **a x b** and for that **n**, we will have **n**•**a** = 0 and **n**•**b** = 0 and therefore **n**•(M**x**) = 0 for all **x**. For this **n = a x b**, we then have **n**•**u** = **n**•**c** = **(axb)**•**c** ≡ D. That is, D is a constant independent of x. This shows that the plane E^2 maps into a plane in E^3.

B. Show that points/lines in E^2 map into points/lines in E^3

The fact that points map into points is pretty clear. We put an E^2 point $\mathbf{x}$ into $\mathbf{u} = M\mathbf{x} + \mathbf{c}$ and out pops the point $\mathbf{u}$ in E^3. This mapping is not 1-to-1 because many points in E^3 map back to the same $\mathbf{x}$ in E^2. This is clear from the fact that the above equation set is 3 equations in 2 unknowns x and y. The mapping cannot possibly be 1-to-1 because E^3 is a bigger space than E^2. We will show right away that the entire E^2 maps into a certain plane in E^3 so the range of the mapping is that plane. Since this plane is less than the entire space E^3, the mapping is "into".

Now consider an arbitrary line in E^2 written as $\mathbf{x} = \alpha\mathbf{x_1} + \mathbf{e}$ where $\mathbf{x_1}$ defines an arbitrary vector in some direction, and α then maps out the line as α takes all real values. The line passes through arbitrary point $\mathbf{e}$ when $\alpha = 0$. Here is what happens to that line as it goes through the mapping to the E^3 space:

$$\mathbf{u} = M\mathbf{x} + \mathbf{c} = M(\alpha\mathbf{x_1} + \mathbf{e}) + \mathbf{c} = \alpha [\, M\mathbf{x_1}] + [\mathbf{c} + M\mathbf{e}]$$

$$= \alpha\mathbf{u_1} + \mathbf{E} \qquad \text{where } \mathbf{u_1} = M\mathbf{x_1} \text{ and } \mathbf{E} = \mathbf{c} + M\mathbf{e}$$

The result is some straight line in E^3 that passes through a point $\mathbf{E}$ and has direction $\mathbf{u_1}$.

C. Show that the mapping does *not* preserve angles or line segment lengths, so shape is not preserved

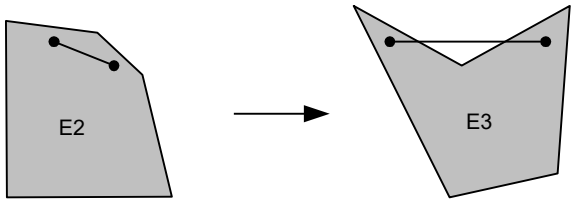
As an example, let $\mathbf{x} = (1,0)$. In this case we find $\mathbf{u_x} = M\mathbf{x}+\mathbf{c} = -\mathbf{a}+\mathbf{c}$. On the other had, it we start with x = (0,1), we get $\mathbf{u_y} = -\mathbf{b}+\mathbf{c}$. It is clear that our two starting vectors in E^2 are perpendicular, but in E^3 the dot product is going to be some strange number $\mathbf{u_x} \bullet \mathbf{u_y} = (-\mathbf{a}+\mathbf{c}) \bullet (-\mathbf{b}+\mathbf{c})$. Since a,b and c are arbitrary, this result is not going to be 0 in general, so the 90 degree angle is not preserved.

As for length of a line segment, consider again $\mathbf{x} = \alpha\mathbf{x_1}$ where now $\mathbf{x_1} = (1,0)$. Let α range from 0 to 1, so we have a segment of unit length. In E^3 this becomes $\mathbf{u} = \alpha\, M\mathbf{x_1} + \mathbf{c} = -\alpha\, \mathbf{a} + \mathbf{c}$. The endpoints of this line segment are at $\mathbf{u_1} = \mathbf{c}$ and $\mathbf{u_2} = -\mathbf{a} + \mathbf{c}$. Since $\mathbf{a}$ and $\mathbf{c}$ can be anything, the distance between these two points is generally not going to be 1.

So we have shown by counterexamples that, in general, angle and segment length is not preserved by our mapping. This means that if we map a polygon, we will end up with a polygon lying on some plane in E^3 but its shape will be distorted from that of the original polygon.

D. Show that convexity of the polygon is preserved by the mapping.

Here we are worried that maybe our mapped polygon in E^3 might look as shown on the right,



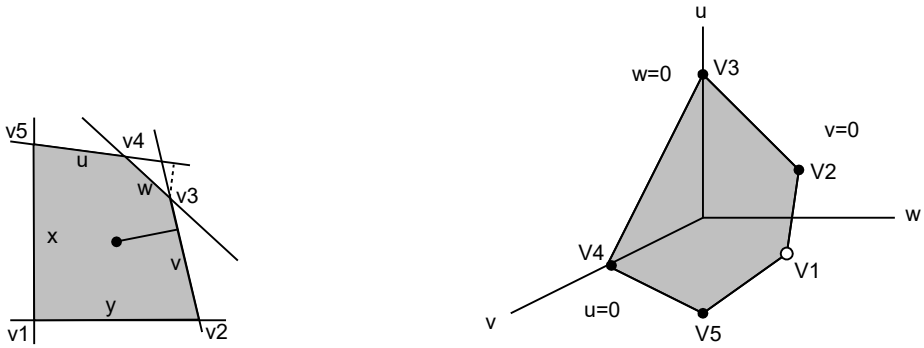
To disprove this possibility, we consider a hypothetical line segment that maps as shown. If the polygon is convex in E^3 then we should be able to find a segment that maps as shown. However, every point in the feasible region of E^2 must map into the feasible region of E^3. After all, we *obtain* the feasible region in E^3 from our mapping by letting the (x,y) coordinates run over all points in the feasible region in E^2. Since the E^2 polygon is convex, any line segment you draw in it must be fully contained in the feasible region.

Thus, the reverse mapping of our hypothetical segment in E^3 must be entirely contained in E^2. But then all forward mapped points on this segment must lie inside the feasible region in E3. The above situation is therefore a contradiction and cannot happen.

E. Application to our Problem.

Based on the above, we know that the polygonal feasible region in E^2 shown in the figure above maps into a polygonal feasible region in E^3. One might at first think the polygon would be mapped into some non-coplanar object in E^3, but we know that the mapping of the feasible region and its polygon boundary lie in a plane in E^3. Since points map into points and line segments into line segments, the mapped boundary is a polygon, but of a shape different from the shape of the polygon in E^2. So we can imagine our mapped feasible region as some tilted polygon sitting in the E^3 space. If the original polygon has N vertices $\{ v_i \}$ in E^2, the mapped polygon will still have N vertices $\{ V_i \}$ in E^3.

But we know much more about the polygon in E^3. In fact, we can draw quite a detailed picture. First we show the picture, then we explain why it looks the way it does. The original E^2 polygon is on the left, while the mapped E^3 one is on the right.



In our 3D picture on the right, the u axis points up, and the u=0 plane is the one containing the v and w axes, and we have labeled it in the figure. Thus u=0 is the "bottom wall" of the first octant in E^3 . Similarly, the left wall is w=0 and the right wall is v=0. The polygon in E^3 is tilted at some strange angle, and the black-dot vertices we claim all lie on the walls, whereas vertex v1 lies out in the middle of the octant somewhere. How do we know this?

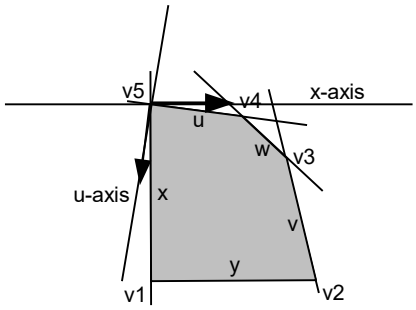
On the left, look at the vertex v3. It lies on the boundaries w=0 and v=0 at the same time (since it lies on both these lines in E^2). This means V3 on the right must lie somewhere on the positive u axis because it must lie on both the w=0 and v=0 planes. In fact, this point lies above the origin by the distance from the point v3 in E^2 to the u line in E^2 (this distance is shown as a little dotted line on the left. Our picture on the right is not drawn to scale). We proved this little fact earlier in this document.

Similarly, vertex V4 is located on the positive v axis since it must be in the u=0 plane and in the w=0 plane. Vertex V5 lies somewhere on the u=0 plane (bottom wall), and vertex V2 lies on the v=0 plane (the right wall). Virtex V1 is the only one which does not lie on a wall! It is at V1 = c, as noted earlier.

So our mapping takes the point **v1** = (0,0) and maps it to **V1** out in the middle of E^3 somewhere, while all the other polygon vertices are mapped to locations on the walls. Since the components of vector **c** are all positive (see earlier), we see that the entire tilted polygon lies in the first octant of E^3.

The Notion of Swapping a Coordinate.

We could, if we liked, replace the E^2 "basis" (x,y) with another basis (x,u) by "swapping" the basis vectors y and u, which we might write as $y \leftrightarrow u$. In our example, (x,y) forms an orthogonal basis in E^2 whereas (x,u) is not an orthogonal basis because the directions of increase of x and u are not perpendicular. Nevertheless, we can locate any point uniquely in E^2 as some (x,u) and then we can regard (y,v,w) as the dependent variables. Here is a picture of the E^2 situation after such a swap, where we show a new origin and the new basis vectors as arrows:



Notice that the new origin (x,u) = (0,0) is at vertex v5. Before this swap, we had (x,y) = (0,0) at vertex v1. So doing this swap has moved the (0,0) point from v1 to v5 and we have thus "traversed" a side of our E^2 polygon by doing the swap. This would not be the case had we chosen to swap $y \leftrightarrow w$, for example. That swap would give us an origin for (x,w) above v5 where the w boundary intersects the y axis. This origin is of very little interest to us, so we would not do such a swap.

Now let's look back at out boundary equations,

```
u = cu - au*x - bu*y
v = cv - av*x - bv*y
w = cw - aw*x - bw*y
```

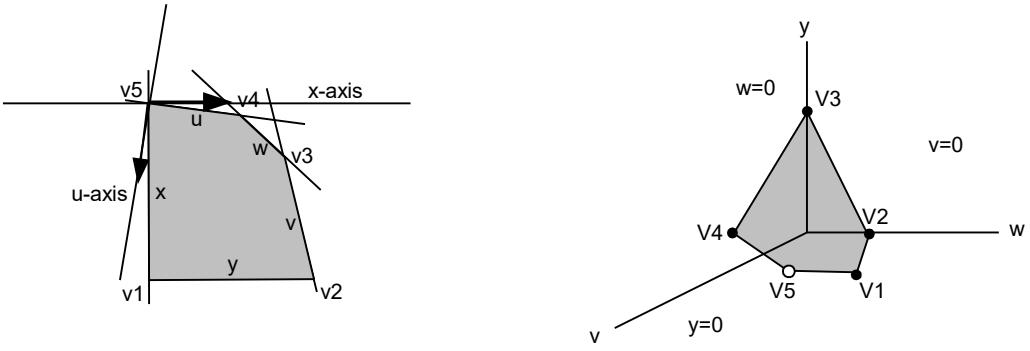
We could solve the first equation for y and substitute that into the last two equations. We can write the solution of the first equation for y as $y = \alpha x + \beta u + \gamma$ and then rewrite our equations as:

$$
\begin{aligned}
y &= \alpha x + \beta u + \gamma \\
v &= cv - av*x - bv*(\alpha x + \beta u + \gamma) \;=\; \alpha' x + \beta' u + \gamma' \\
w &= cw - aw*x - bw*(\alpha x + \beta u + \gamma) \;=\; \alpha'' x + \beta'' u + \gamma''
\end{aligned}
$$

where we have defined ' and " constants just to keep notation simple. In matrix and vector form we have

$$
\begin{bmatrix} y \\ v \\ w \end{bmatrix} = \begin{bmatrix} \alpha & \beta \\ \alpha' & \beta' \\ \alpha'' & \beta'' \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + \begin{bmatrix} \gamma \\ \gamma' \\ \gamma'' \end{bmatrix} \qquad \mathbf{u} = T\mathbf{x} + \mathbf{g}
$$

which of course is just some other linear transformation. Although all the constants in T and g are unpleasant functions of the original constants, we can still draw a picture of the mapped polygon in this new E^3 space.



The explanation is similar to our previous mapping. This time, all five vertices in E^3 lie on the walls except V_5 . Notice this interesting fact. In our original basis which had the origin (0,0) at v_1 , we found that mapped vertex V_1 was the one lying out in the middle of the first E^3 octant. After our swap $y \leftrightarrow u$. our E^2 origin has moved to v_5 and now the mapped vertex V_5 is the one lying in the middle of the octant.

Suppose after doing the $y \leftrightarrow u$ swap we just described, we were to do a swap $x \leftrightarrow w$. Before this swap we have (x,u) as our basis, and this swap would take us to the basis (w,u). We could draw a picture like the left one above but with the origin moved to v4 and new basis arrows. If we drew our mapped polygon in a space labeled (x,y,v), we would find that vertex V_4 is the one vertex out in the octant and not on the walls in E^3. The reason is that all the other vertices are on at least one "wall". This one vertex V_4 lies on the u and w boundaries but neither of these variables is an axis of the E^3 space, so this point is just out there somewhere.

So here is what we have learned. If we start with the (x,y) basis in E^2, we have our origin (0,0) at v_1. If we do the swap $y \leftrightarrow u$, our origin moves to v_5. If we then do the swap $x \leftrightarrow w$, we move our origin to v_4. We can keep doing swaps and work our way around the perimeter of the polygon. The sequence of swaps required to do this is:

(x,y)

$y \leftrightarrow u$

(x,u)

$x \leftrightarrow w$

(w,u)

$u \leftrightarrow v$

(w,v)

$y \leftrightarrow w$

(y,v)

$v \leftrightarrow x$

(y,x)

We are interested in thinking about the vertices of our polygon because we know that the solution to our linear programming problem is going to be at one of these vertices.

The Simplex Table

Now we can write the same equations we wrote above in a different manner:

```
au*x  +   bu*y   +u              = cu
av*x  +   bv*y        +v        = cv
aw*x  +   bw*y              +w   = cw
```

In the original basis, for any specified (x,y), the above equations form a system of 3 equations in 3 unknowns (a 3x3 system) which unknowns are (u,v,w). You think of (x,y) as sort of pre-selected parameters and they play the same role as the constants cx when you think of the 3x3 linear system. Now it is certainly *possible* to write the above equations in this matrix form where we have a 5x3 matrix on the left,

$$\begin{bmatrix} au & bu & 1 & 0 & 0 \\ av & bv & 0 & 1 & 0 \\ aw & bw & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \\ w \end{bmatrix} = \begin{bmatrix} cu \\ cv \\ cw \end{bmatrix}$$

and we often abbreviate this matrix equation by writing the following table

x	y	u	v	w	const	
au	bu	1	0	0	cu	u
av	bv	0	1	0	cv	v
aw	bw	0	0	1	cw	w

The column headings remind you which variable that column stands for. The row labels on the right side remind you "which is which" of our three boundary equations. The two columns on the right are called "the stub" of the table, and sometimes these columns are put on the left like this

	con	x	y	u	v	w
u	cu	au	bu	1	0	0
v	cv	av	bv	0	1	0
w	cw	aw	bw	0	0	1

This is how the table appears in Scheid Chapter 27, for example.

Now WHY would we want to write our equations using the non-square 5x3 matrix shown above, and in its table abbreviations?

The Simplex Problem

We are trying to locate the point lying in the feasible region in E^2 that maximizes or minimizes some quantity M = Ax + By. That feasible region is the shaded region in E^2 shown above. We know from a

fundamental theorem (not proved here, but which probably seems reasonable) that the solution point of our problem must lie at one of the feasible region (polygon in our case) vertices in E^2. One way to solve the problem is to compute the 5 vertex coordinates $v_i = (x_i, y_i)$, compute the five values $M_i = Ax_i + By_i$ and just see which vertex gives the largest or smallest M_i. More or less, this is what the Simplex method will do, but it will not have to check all 5 vertices because it will be smart about doing things. When there are more physical coordinates (instead of just n=2), and more boundary inequalities (instead of just m=3), there can be a lot of "vertices" to check, so Simplex helps out a lot.

The first step is to take the equation M = Ax + By and add it to our set of equations above.

x	y	u	v	w	const	
au	bu	1	0	0	cu	u
av	bv	0	1	0	cv	v
aw	bw	0	0	1	cw	w
A	B	0	0	0	M	M

Recall our discussion earlier about changing basis in E^2 where we might go from (x,y) to (x,u) by doing the swap $y \leftrightarrow u$. In this case, we uniquely represent a E^2 point using (x,u) and we can compute the other coordinates (y,v,w) using this transformation (the first row says $y = \alpha x + \beta u + \gamma$),

$$\begin{bmatrix} y \\ v \\ w \end{bmatrix} = \begin{bmatrix} \alpha & \beta \\ \alpha' & \beta' \\ \alpha'' & \beta'' \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + \begin{bmatrix} \gamma \\ \gamma' \\ \gamma'' \end{bmatrix} \qquad \mathbf{u} = T\mathbf{x} + \mathbf{g}$$

What does this look like in our Simplex table world? Here is the answer:

x	y	u	v	w	const	
$-\alpha$	1	$-\beta$	0	0	γ	y
$-\alpha'$	0	$-\beta'$	1	0	γ'	v
$-\alpha''$	0	$-\beta''$	0	1	γ''	w
$(A+B\alpha)$	0	$B\beta$	0	0	$(M-B\gamma)$	M

The first row, for example, says that $-\alpha x + y - \beta u = \gamma$ which is the same as $y = \alpha x + \beta u + \gamma$ which is indeed the first of our three equations above. In this way we verify the first three Simplex rows. The last row comes from the fact that M = Ax + By, but our E^2 basis is now (x,u), so we use $y = \alpha x + \beta u + \gamma$ to write $M = Ax + B(\alpha x + \beta u + \gamma) = (A+B\alpha)x + B\beta u + B\gamma$ which means $(A+B\alpha)x + B\beta u = (M - B\gamma)$.

Now let's go back to our table before the swap and interpret it. We had

x	y	u	v	w	const	
au	bu	1	0	0	cu	u
av	bv	0	1	0	cv	v
aw	bw	0	0	1	cw	w
A	B	0	0	0	M	M

Remember that this is a short-hand for this matrix equation,

$$\begin{bmatrix} au & bu & 1 & 0 & 0 \\ av & bv & 0 & 1 & 0 \\ aw & bw & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \\ w \end{bmatrix} = \begin{bmatrix} cu \\ cv \\ cw \end{bmatrix}$$

and this scalar equation: Ax + Bx + 0u + 0v + 0w = M (the bottom row of the table). We could incorporate this last equation into the matrix equation by adding a new row to the matrix and by then adding M as the bottom vector on the right,

$$\begin{bmatrix} au & bu & 1 & 0 & 0 \\ av & bv & 0 & 1 & 0 \\ aw & bw & 0 & 0 & 1 \\ A & B & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \\ w \end{bmatrix} = \begin{bmatrix} cu \\ cv \\ cw \\ M \end{bmatrix}$$

Now, we apply this table (or matrix equation) to the vertex v1 which has (x,y) = (0,0). We find that:

u = cu
v = cv
w = cw
M = 0

Now after we do our first swap, the table looks like this

x	y	u	v	w	const	
−α	1	−β	0	0	γ	y
−α'	0	−β'	1	0	γ'	v
−α"	0	−β"	0	1	γ"	w
(A+Bα)	0	Bβ	0	0	(M−Bγ)	M

If we apply this table to the vertex v5 which has (x,u) = (0,0), we find that

y = γ
v = γ'
w = γ"
M = Bγ

If we are trying to maximize the quantity M, and if Bγ > 0, then we conclude that vertex v5 is better than vertex v1.

Although we have not mentioned it yet, it should be clear that, with the first table, we built the E^3 space with coordinates (u,v,w) and these are the columns that have the simple unit vectors like (0,0,1). The second table implies the E^3 space with coordinates (y,v,w) and this table has the unit vectors in those three columns. These unit vectors can be thought of directly as the three axes of the E^3 space.

Using the Gauss-Jordan elimination method

Once again, in our original basis, the Simplex table encodes this matrix equation:

$$\begin{bmatrix} au & bu & 1 & 0 & 0 \\ av & bv & 0 & 1 & 0 \\ aw & bw & 0 & 0 & 1 \\ A & B & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \\ w \end{bmatrix} = \begin{bmatrix} cu \\ cv \\ cw \\ M \end{bmatrix}$$

We know that we are allowed to multiply any row by a constant and the equation represented by that row is not altered. For example, we could multiply the 3rd row by 2 to get

$$\begin{bmatrix} au & bu & 1 & 0 & 0 \\ av & bv & 0 & 1 & 0 \\ 2aw & 2bw & 0 & 0 & 2 \\ A & B & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \\ w \end{bmatrix} = \begin{bmatrix} cu \\ cv \\ 2cw \\ M \end{bmatrix}$$

We know we can add any row to a given row because adding two true equations yields a true equation, so we replace our "any row" with an equivalent equation. So combining these ideas, we are allowed to change any row by adding to it an arbitrary multiple of any other row.

Using these rules, it is relatively simple to move a unit vector like (1,0,0) from one column to another. We know this is what one of our carefully ordered swaps does, so we can accomplish the swap by just fiddling with the Gauss-Jordan rules to get the unit vectors where we want them. In that version of the table, we will go to the polygon vertex that has "the other two" coordinates 0, and we observe the value in the bottom entry of the right side vector (where M is right now). We go with the largest or smallest M, depending on what we are doing.