Simplex Attempt 3 PhL 11.28.04

[In this document, I review the previous Simplex 2 presentation, and then discover the idea of adding M to the matrix equation. I then do battle with Scheid's notation but don't get happiness with it.]

Review of #2 document.

We start with a simple example where the feasible region is in E^2 (a polygon which includes the origin), and there are three far boundaries so the co-space is E^3 . This example is n=2 and m=3. We define some conventions for writing the equations which make the slack variables be directly interpretable as distances from their boundaries in the E^2 space. We draw the most general polygon that fits into the mold of our sample problem. We think of the E^2 variables as parameters, and the E^3 space then has a 3x3 system that we can solve for the slack variables.

We then study this mapping of $E^2 \rightarrow E^3$. Various properties are derived which probably apply to any linear transformation: points, lines and planes map into same. Distances and angles are not preserved, but convexity is preserved. We find that our polygon in E^2 maps into a tilted polygon in E^3 that lies in the first octant. Most of the points of this polygon (all but one in fact) lie on the "walls" in E^3. The reason is that the walls represent the boundary lines, and only the origin vertex (x,y) = (0,0) is then not on a wall.

We then show that you can move around the polygon from vertex to vertex by doing a specific sequence of variable swaps. That is, you exchange a variable in the E^2 space with one in the E^3 space. The reason this takes you on a tour of the vertices is this: a vertex is in fact a place where two boundary lines meet, and those two lines have variable names like (x,u). The vertex here is then has x=0 and u=0. So you are moving your E^2 space origin from one vertex to the next. In the E^3 space of the other 3 variables, we find that the Simplex table has simple unit vectors.

Why do we get these unit vectors? In the original problem statement, each equation gets one slack variable, so in the starting position we do have 3 unit vectors in E^3 which are for (u,v,w). If we take (x,y) = (0,0), we find that (u,v,w) = (3x3 identity)(cu,cv,cw), so the slack variables take their max values.

Question " Why does a coordinate swap move a unit vector from one column to another?

Answer: In our original system we had these equations,

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

$$\mathbf{u} = M\mathbf{x} + \mathbf{c} \qquad (1)$$

Our E^3 variables are those on the LHS. Since these equations are *solutions for those variables*, in each equation the variable appears *just one time with coefficient unity*. That is the reason that we get the unit vectors in our matrix equation:

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

Now, when we "swap a pair of variables" what we really mean is that we are swapping one variable on the LHS of (1) for one variable on the RHS of (1). Consider y ↔ u. We are then looking for a new set of solution equations that have (y,v,w) on the LHS and (x,u) on the RHS. This means we have to somehow eliminate y so it does not appear anywhere on the RHS, and u appears in its place. We need to find an expression y = y(u,x). This comes from the first equation. I will do the general case soon. We then replace all the y's with this expression, and we then have our solution, which is

```
y = α x + β u + γ        =>   y - α  x - β  u = γ
v = α'x + β'u + γ'       =>   v - α' x - β' u = γ'
w = α"x + β"u + γ"       =>   w - α" x - β" u = γ"
```

This is "after swap". Again, since these are *solutions* for the three variables on the left, we know that the three columns in the matrix (2) for these variables will have the unit vectors! Very good. So yes, a swap just moves a unit vector from the one column to another. If course data in the other columns is also altered. In the above case we now have:

$$\begin{bmatrix} -\alpha & 1 & -\beta & 0 & 0 \\ -\alpha' & 0 & -\beta' & 1 & 0 \\ -\alpha'' & 0 & -\beta'' & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \gamma \\ \gamma' \\ \gamma'' \end{bmatrix}$$

The altered data is in the columns that lie under the new "parameter" variable headings, because all the other columns have unit vectors.

Now, let's try to add the M equation. I had M = - Ax - By + C (to be general say). We can just add this equation to the list above. Here:

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

$$\mathbf{u} = M\mathbf{x} + \mathbf{c} \qquad (1)$$

It really is on a completely equal footing. Since we have a "solution" for M as well as the other three variables, the M column in the table or matrix will always be 0,0,0,1. Let's write this down:

$$\left[\begin{array}{cc|cccc} au & bu & 1 & 0 & 0 & 0 \\ av & bv & 0 & 1 & 0 & 0 \\ aw & bw & 0 & 0 & 1 & 0 \\ A & B & 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ u \\ v \\ w \\ M \end{bmatrix} = \begin{bmatrix} cu \\ cv \\ cw \\ C \end{bmatrix}$$

I like it! Now the unit vectors are extended to one more dimension because we have solutions of four equations. Now if we do a swap (but never swap M) between a variable on the left side and one on the right side of the little line, what will happen is the unit vector will move from the original cospace column to the column of the swapped out variable on the left.

What numbers get changed in this process? The row fiddling is just a graphical method of solving the equations for the new variables, nothing more and nothing less. So look at the above equation and think about what will change if we swap y and u. I circle things that will get changed:

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

We will first scale the first equation to make bu→1, which changes all the other nonzero values in that row, so I circle them. Then we will add multiples of this first row to all the other rows. This changes the other rows only where the first row has non-zero values, so that accounts for the rest of the circles. The second column of course will end up being 1,0,0,0. So the answer is: a variable swap changes all entries in the "other" parameter columns on the left side (here, that is only the x column), and changes the entries in the formerly unit vector column (that of the variable swapped out of the right, and finally the constants column on the right. Said another way, all columns change except the unchanged unit vectors. This is pretty reasonable, we are changing basis and we expect all vectors to have new components in the new basis. More on this soon.

Now once we have done this swap, we are going to examine things when all the variables on the left are 0 and we hope that is a vertex in our E^n space. The above equations are then diagonal and we have the solution u = cu', v = cv', w = cw', M = C'. So the right column IS our solution at this vertex.

Now how do we decide which columns or variable pairs to swap? I will answer that later. But what about the observation about the M solution. In the above example, suppose A and B are positive. We see this last equation as -Ax - By + M = C. Suppose we trying to minimize M. We can see that increasing x or y will do this. We could then benefit from swapping out either one of these because after the swap, our new coordinates on the left will be (x,u) say, and our test point will still be (0,0) so x=0, but we know that we will end up with some positive y value, so we know M will be smaller after this swap.

So, there are several issues here. One is that we sort of look at the bottom row to see which variable on the left we would like to swap (perhaps one with largest + coefficient if we are trying to minimize M). But then we have to select a variable on the right to swap it with. We would like to do that in a way that moves us to a new vertex. I see how to do that in our test problem, but not in the general case.

OK, here is the rule on picking the variable on the right side to swap with. We want to pick the one that is going to first "hit a boundary" as we increase y. Look again at our equations,

```
u = cu - au*x - bu*y
v = cv - av*x - bv*y
```

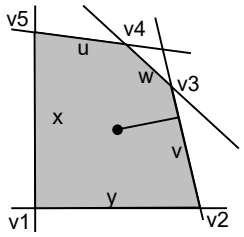
```
w = cw - aw*x - bw*y
```

As we have x = 0, and we increase y, which one hits first?

```
u = cu  - bu*y   = cu(1 - y/y1)        y1 = cu/bu
v = cv  - bv*y   = cv(1 - y/y2)        y2 = cv/bv
w = cw  - bw*y   = cw(1 - y/y3)        y3 = cw/bw
```

These are the values where each equation will hit the limit, like u=0. Suppose y1 is the smallest. Then we will hit u=0 in the first equation with y1. In the second, (1 - y1/y2) will be >0 so we end up with some positive v and so on. So the rule is this: compute the ratios shown on the right and pick the smallest. That tells you which variable to swap. And you will end up at the u=0 boundary with x = 0, so you will end up at a new vertex (I think). Modification of this rule: if the yi ratio value is infinite, then for that variable increasing y has no effect, so we ignore it. And if yi is negative, increasing y increases the variable, so we are never going to hit the u=0 boundary. We are assuming here that the three cx are > 0. So we only consider ratios which are positive and finite. That is how the rule needs to be modified.

How would this rule work in my problem:



I start off at v1. I think about increasing y. With x = 0, I can see clearly that I am going to hit the u line before I hit the w line and then the v line, so u is my swap choice. And I do get then to v5.

Now, when you find the smallest ratio, that row is the variable you are going to swap with so that location is where you want to cause a 1 to appear. This point is called the pivot.

In the blue book example, her first swaps out z with v, then swaps y with w. Since the bottom row numbers are all positive now, he says no more swaps will help.

How do we know that? On page 67 bottom, he knows that increasing x will worsen his situation due to the sign. Notice that, at this point, columns x, v and w are now "left side" columns, so we are thinking now about the point (x,v,w) = (0,0,0) and this is a certain vertex. The variables are all zero at this vertex. In doing another swap, we can only increase one of these three. But increasing any of them hurts, so we are done! So yes, I like his rule for doneness. Once you get to a vertex such that increasing any of the LHS variables hurts, you know you are done! Thus, you don't really have to exhaust all the vertices.

Now what on earth is Scheid doing in his writeup?

He adds the slack variables. Now I see what he calls them x_{n+1} through x_{n+m}, because he wants them to be on an equal footing with the other ones. Compare to x,y and then w,u,v. Once Scheid has his matrix with the m unit vectors on the right, and the filled columns on the left. He rewrites the entire matrix equation in column vector notation. The n left variables are the physical ones we start with, the m rightmost variables are the slack variables.

Now for the starting point I would have taken the first n variables to be 0, as we set (x,y) = 0. However, he instead takes the last n variables to be 0. In my example, that is like taking (v,w) = 0. That is

OK, it is still a vertex somewhere. Then the first m variables are nonzero and we get the simplified equation with the zeroed out terms missing. This would be my 3x3 system in (x,y,u). It is certainly possible to write H in terms of the (x,y,u) variables. He has an mxm problem with a selected m active coordinates, so H is a function of the m active coordinates.

So equations (1) and (2) are just fine. In (3) he says he can write any column vector as a lincom of the m independent ones, and he calls the coefficients $v_{k\ell}$ which is fine by me.

Now I think I better understand. When we writes "include a piece px_k for k > m" : the leftmost m variables are active so are on the "right side" of my usual picture. The "left side" variables in my picture are now all zero. We want to think about increasing just ONE of them, the way blue book increases z from 0.

We now come to the infamous equation (5). It still is unclear, so let me now try to rewrite Scheid the way I would do it. I will go ahead and set the last n variables to 0 as he does, even though some of these are slack variables. This corresponds to picking a vertex out in the volume somewhere far from the origin. Any vertex is OK to start. I agree with (1), and I agree with (2). I agree with (3) . Equation (4) at this point is just a definition. But suppose I add my last row to his matrix, it would look like this:

$c_1 \;\; c_2 \;\; \ldots\ldots\ldots \;\; c_{n+m}$ then M in the variable vector, and some constant b_{m+1} on the right.

This would imply that we have

$$x_1c_1 + x_2c_2 + \quad \ldots\ldots \quad + x_{n+m}\,c_{n+m} + M = b_{m+1}$$

Now if the last n variables are 0, we get

$$x_1c_1 + x_2c_2 + \ldots\ldots \; x_mc_m + M = b_{m+1}$$

so we identify $H_1 = b_{m+1} - M$. I am unclear as to why he puts the 1 subscript on H.

Now again we must face up to equation (5), the great barrier! Look back at (1) which is our matrix equation with the last n x's variables set to 0. There is some v_k vector of course, so suppose we "move out" into our feasible space by moving p units away from our starting vertex in the "k" direction. Then yes, we are just adding a term to equation (1). But look at what we have added:

$$p\,\mathbf{v_k} \;=\; p\,[\,\sum_{i=1}^{m} \; v_{ik}\,\mathbf{v_i}\,] \;=\; \text{amount added on the right of equation (1)}$$

We have displayed the amount added as a sum of m pieces, each associated with one of the basis vectors. So if we subject this off, then (6) is exactly the same as (1). Here is what equation (1) was saying:

$$\sum_{i=1}^{m} \; x_i\,\mathbf{v_i} \;=\; \mathbf{b} \qquad\qquad (1)$$

So do this

$$\sum_{i=1}^{m} \; x_i\,\mathbf{v_i} \;-\, p\,\mathbf{v_k} + p\,\mathbf{v_k} = \mathbf{b}$$

which says

$$\sum_{i=1}^{m} x_i \mathbf{v_i} - p \sum_{i=1}^{m} v_{ik} \mathbf{v_i} + p \mathbf{v_k} = \mathbf{b}$$

which says

$$\sum_{i=1}^{m} (x_i - pv_{ik}) \mathbf{v_i} + p \mathbf{v_k} = \mathbf{b} \qquad \textbf{(5)}$$

So we are over that hurdle! At this point, we have m+1 columns active in our matrix, but he is going to get rid of one of these columns by choosing p such that $(x_i - pv_{ik})$ for some i.

So what exactly is happening here? We have the rightmost n variables = 0, the leftmost m are active. We are exploring activating one of the zero variables. If we turn it on by amount p, then we have to reduce the coefficients of the other basis vectors exactly as shown. One of our assumptions is that we insist that ALL n+m variables are ≥ 0.

Now why do we require that $(x_i - pv_{ik}) \geq 0$? I know that we want $x_i \geq 0$ for all i. Let's rewrite (5) in this way

$$\sum_{i=1}^{m} x_i' \mathbf{v_i} + x_k \mathbf{v_k} = \mathbf{b}$$

Who says that the feasible space must have positive coefficients? Well x_i' is *the value of* the x_i coordinate after we add our v_k piece. As we move in the k direction, all the other coordinates are going to maybe decrease, and we have to make sure we don't hit the wall on any of them. So we require $x_i' \geq 0$ because we require that the value of the $\mathbf{v_i}$ variable be positive.

So in our current framework, we select $p = \min (x_i/v_{ik})$ for only $v_{ik} > 0$. Notice that the v_{ik} are not yet displayed in any matrix. I think he should have used some other letter for v_{ik} such as d_{ik} . For example, I know that $v_{11} = 1$ and $v_{22} = 1$ from the expansion of unit vectors equation, but $a_{11} \neq 1$, etc.

But fine, I will stick for now with his v_{ik} notation. (it causes too many v's to be flying around!) I agree to select p as in his equation (7).

Now lets go back to equation (4) and see where it is coming from. If we included the "last equation" in our matrix, we would say this, as noted above.

$$x_1c_1 + x_2c_2 + \quad ..... \quad + x_{n+m}c_{n+m} + M = b_{m+1}$$

Now if the last n variables are 0, we get

$$x_1c_1 + x_2c_2 + ... \quad x_mc_m + M = b_{m+1}$$

I can rewrite this as

$$-M + b_{m+1} = x_1c_1 + x_2c_2 + ... \quad x_mc_m = H_1$$

Now as we turn on our k basis vector by amount x_k, what is going to happen to the above equation, which is our last matrix equation. I am inclined to try this:

$$x_1c_1 + x_2c_2 + ... \quad x_mc_m + pc_k - pc_k = H_1$$

Then we somehow think maybe the $-x_kc_k$ term can be written so this becomes true

$$x_1'c_1 + x_2'c_2 + ... \quad x_m'c_m + pc_k = H_1$$

and this would say that

$$\sum_{i=1}^{m} (x_i - pv_{ik})\, c_i + pc_k = H_1$$

but this also disagrees with (6) unless $h_k = 0$.

****

Here I just repeat the last attempt with slightly different language, and conclusion is the same.

NOW, imagine this equation as the bottom equation in his matrix, but not shown by him. We make all the vectors one longer with last components called c_i. Then the transformation shown above,

$$\sum_{i=1}^{m} (x_i - pv_{ik})\ \mathbf{v_i} + p\,\mathbf{v_k} = \mathbf{b}$$

takes this form on the last row, (nothing ever changes the rightmost unshown M column)

$$\sum_{i=1}^{m} (x_i - pv_{ik})\ c_i + p\,c_k + M = b_{m+1}$$

or

$$\sum_{i=1}^{m} (x_i - pv_{ik})\ c_i + p\,c_k = H_1$$

This looks like (7) but only if $h_k = 0$, so something is wrong!

****

OK, the battle continues to rage concerning Scheid (4) and (6). What happens if I insert the expression (4) in for h_k in (6) ?

$$h_k \equiv \sum_{i=1}^{m} v_{ik}\, c_i - c_k \qquad (4)$$

$$\sum_{i=1}^{m} (x_i - pv_{ik})\, c_i \;+\; pc_k = H_1 \;- ph_k \qquad (6)$$

Combining these I get

$$\sum_{i=1}^{m} (x_i - pv_{ik})\, c_i \;+\; pc_k = H_1 \;- p\,[\, \sum_{i=1}^{m} v_{ik}\, c_i \;- c_k\,]$$

Cancel the sums involving v_{ik} and we end up with

$$\sum_{i=1}^{m} x_i\, c_i \;+\; pc_k = H_1 \;- p\,[\, - c_k\,]$$

which says

$$\sum_{i=1}^{m} x_i\, c_i = H_1$$

So why does it work here but not when I did it before??? I think I made the wrong association with M. Given that we strictly define H_1 as shown, let's compare (6) to my "last row equation" ,

$$\sum_{i=1}^{m} (x_i - pv_{ik})\, c_i \;+\; pc_k = H_1 \;- ph_k \qquad \text{// (6)}$$

$$\sum_{i=1}^{m} (x_i - pv_{ik})\, c_i \;+\; p\, c_k + M = b_{m+1} \qquad \text{// my last row equation}$$

I now make this connection:

$$-M + b_{m+1} = H_1 - ph_k$$

Let's now replace "my" variables M and b_{m+1} with a new variable I will just call H.

$$H = -M + b_{m+1} = H_1 - ph_k$$

So now we see that H_1 was the value of H for our initial vertex selection! And if $h_k > 0$, then our proposed motion in the k direction is helping us! Scheid's discussion is missing the mention of a variable like H and that is confusing as hell!

Now, let's take a closer look at this "last row" business. I had this to start with

$$x_1c_1 + x_2c_2 + \quad \ldots\ldots \quad + x_{n+m}\, c_{n+m} + M = b_{m+1}$$

as my last row, but all c's are zero after m, so the last row contracts to this

$$x_1c_1 + x_2c_2 + \quad \ldots\ldots \quad + x_m\, c_m + M = b_{m+1}$$

Now, if we go a bit in the k direction, this row becomes this

$$\sum_{i=1}^{m} (x_i - pv_{ik})\, c_i + pc_k + M = b_{m+1}$$

which we can write as

$$\sum_{i=1}^{m} (x_i - pv_{ik})\, c_i + pc_k = -M + b_{m+1} = H$$

I think I now see what he is doing. We want to write H in a form that shows it is decreasing, so we write

$$H = H_1 - ph_k$$

where we separate out the original H_1 and define h_k buy THIS equation. Then we solve and find what h_k turns out to be.

Suppose Scheid had chosen the last m columns to be his basis vectors, the ones that have unit vectors in them. And suppose he had chosen the first n variables to be the ones that are initially zero. Then turning on a new variable k, we would have k in this first set. What could we then say about the h_k?

(1) for j in the last columns of unit vectors, we have $h_j = v_{jj}c_j - c_j = c_j - c_j = 0$, and this agrees with my usual "starting position".

His approach is just really ugly.

Here is my rewrite:

(1) Write the matrix as he has it, select the last m columns as the basis vectors, and choose the starting vertex to be the first n variables. Then equation (1) would become

$$\sum_{i=n}^{n+m} x_i \mathbf{v}_i = \mathbf{b} \qquad \text{// because the first } x_i \text{ are 0} \qquad (1)$$

Similarly, we would have that

$$H_1 = \sum_{i=1}^{m} x_i c_i = 0 \qquad \text{// again, because the first n } x_i \text{ are 0.} \qquad (2)$$

Then for (3) we would have this

$$\mathbf{v}_j = \sum_{i=n+1}^{n+m} v_{ij} \mathbf{v}_i \qquad j = 1 \text{ to } n+m \qquad (3)$$

Now what does the new h_j equation look like? I think this: (first v_{ij} index goes over basis vectors)

$$h_j = \sum_{i=n+1}^{n+m} v_{ij}\, c_i \;-\; c_j \qquad \text{// notice same sum appearing} \quad (4)$$

Now if j is in the range of the unit vectors, which is j = n+1 to n+m, we get

$$v_{ij} = \delta_{i,j} \qquad j \text{ in } n+1 \text{ to } n+m$$

and in this case we get

$$h_j = \sum_{i=n+1}^{n+m} \delta_{i,j}\, c_i \;-\; c_j \;=\; c_j - c_j = 0$$

which agrees with the idea that in your initial stance, $h_j = 0$ in the unit vector bottom rows.
On the other hand, for j in the first n columns we have

$$v_{ij} = a_{ij} \qquad j = 1 \text{ to } m \qquad i = n+1 \text{ to } n+m$$

so we would find then that

$$h_j = \sum_{i=n+1}^{n+m} a_{ij}\, c_i \;-\; c_j$$