# Linear Programming Functions for the TI-Nspire CX CAS Handheld and Associated Emulators.

Language: English

Version: 1.0

Date: 5/11/2018

Author: James O. Thompson

Email: jodaddy101@hotmail.com

# Table of Contents

# 1. Introduction

The functions presented here are not intended to solve linear programming problems. Neither are they intended to teach linear programming. Instead they should complement a good introductory text on the subject. Such a text can be found in “Finite Mathematics” by Lial, Greenwell and Ritchey. This book is frequently used in a college course for students majoring in business, management, economics, etc.

Any student attempting to learn linear programming without, at least, help with a modern calculator such as the TI-84 is sorely disadvantaged. That student would be totally overwhelmed with the mountain of trivial calculations required to solve the most basic linear programming problem. Even with the help of the matrix features of the TI-84, the flood of trivial calculations obscures the beauty of linear programming solution methods. Burdened with repetitive, redundant obscuring calculations such a disadvantaged student would be able to solve few of the many rich exercises at the end of each section of the book. With the help of the functions presented here the student should be able to solve many of the interesting problems posed at the end of each book section. That … and have time for other concurrent college courses as well.

The functions presented here are for the student equipped with a TI-Nspire CX CAS handheld or an emulator on a PC, iPad, etc. The “CAS” (Computer Algebra System) part is essential. The functions cannot be installed on the TI-Nspire without the CAS part.

The functions should be introduced piece meal through the course as aids to automate methods already taught from the book. The student should know the process being automated and use the automation only to avoid repetitive calculations not contributing to the learning process.

The functions are presented in three sets in Sections 4, 5 and 6 corresponding to the approach taken in “Finite Mathematics”. Section 4 addresses Standard Maximization Problems, Section 5 addresses Standard Minimization Problems, and Section 6 the Nonstandard Problems. Section 7 shows combining functions but in so doing hides the underlying simplex methodology. For that reason, I debated including those functions at all. Use the Section 7 functions sparingly. Don’t use them to avoid learning the algorithms of solving linear programming problems.

Following standards from the book, the subscripted variables $x_j$, $y_j$,, $s_i$, $a_i$, $z_0$ and $w_0$ are reserved. There will be conflicts if you create variables with those subscripted names in your TI-Nspire calculations. In all the problems, feasible solutions require that all $x_i$ and $y_i$ be greater than or equal to zero. This constraint will not be pointed out with each problem definition.

The TI-Nspire can operate in either exact or approximate mode for calculations. Approximate mode introduces rounding and other problems so always use exact expressions when using these linear programming functions. This can be done by multiplying entire equations by a fraction removing constant or entering fractions in a numerator/denominator form.

## 2. Platforms

The functions presented here will execute on the TI-Nspire CX CAS handheld, the TI-Nspire CX CAS Student Software and associated emulators executing on the iPad.

## 3. Installation

All of these functions are bundled into a single file, “simplex.tns”. This file is to be installed in the “mylib” folder where it will join the other files (numtheory.tns and linalgcas.tns) from the default installation. The file can be transferred to the handheld using the process described for file transfers in the handheld owner’s manual. On a PC the “mylib” subdirectory is in the User’s Documents subdirectory which, unfortunately, Microsoft likes to hide. See this web reference on how to locate the User’s Documents subdirectory.

After installing this file, you must “Refresh Libraries”. On the handheld, press Doc, then choose the “Refresh Libraries” option. On the Student Software, choose the Tools menu, then the “Refresh Libraries” option. Following this the functions may be accessed as the file name (simplex) followed by the backslash (\) followed by the function name. e.g., “simplex\pblmmax()”.

Following installation, the function source code may be inspected and modified as desired. Furthermore, it may be distributed freely and for free.

## 4. Standard Maximization Problems

This section describes one particular type of linear programming problem, the Standard Maximization Problem.

a. Definition of the Standard Maximization Problem.
In this section we will address only Standard Maximization Problems such as the following:

Maximize: $z_0 = 8x_1 + 3x_2 + x_3$

Subject to: $x_1 + 6x_2 + 8x_3 \le 118$

$x_1 + 5x_2 + 10x_3 \le 220$

b. simplex\PblmMax(numVars, numEqns)
The subscripted variables are difficult to enter so we introduce our first linear programming function. The numVars argument notes the number of variables and the numEqns the number of equations. This function creates a template for entering standard maximization problems customized for our number of variables and equations. We copy this template into an assignment statement, edit the ?'s with our coefficients and our problem definition is complete as the following example shows. For now, always leave the relation operator as is for all standard maximization problems. Also, the coeffecients for the objective function should all be positive.

**simplex\pblmmax(3,2)** ▸

| _ | $x_1$ | $x_2$ | $x_3$ | _ | _ |
|---|---|---|---|---|---|
| *maximize_z* | ? | ? | ? | _ | _ |
| *subject_to* | _ | _ | _ | _ | _ |
| *eqn1* | ? | ? | ? | "≤" | ? |
| *eqn2* | ? | ? | ? | "≤" | ? |

**a:=**

| _ | $x_1$ | $x_2$ | $x_3$ | _ | _ |
|---|---|---|---|---|---|
| *maximize_z* | 8 | 3 | 1 | _ | _ |
| *subject_to* | _ | _ | _ | _ | _ |
| *eqn1* | 1 | 6 | 8 | "≤" | 118 |
| *eqn2* | 1 | 5 | 10 | "≤" | 220 |

▸

| _ | $x_1$ | $x_2$ | $x_3$ | _ | _ |
|---|---|---|---|---|---|
| *maximize_z* | 8 | 3 | 1 | _ | _ |
| *subject_to* | _ | _ | _ | _ | _ |
| *eqn1* | 1 | 6 | 8 | "≤" | 118 |
| *eqn2* | 1 | 5 | 10 | "≤" | 220 |

c. simplex\PblmToTab(pblm)
Our next function creates a tableau from a problem definition. We know a tableau consists of the negative of the objective function on the bottom row, the constraint coefficients in a matrix and slack variables introduced for each equation as per the following.

$$b:=\text{simplex\pblmtotab}(a) \rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & s_1 & s_2 & z_0 & \_ \\ 1 & 6 & 8 & 1 & 0 & 0 & 118 \\ 1 & 5 & 10 & 0 & 1 & 0 & 220 \\ -8 & -3 & -1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

This function doesn't seem to be doing much at this time but we will build on it in future sections of this document. It will grow into one of our most useful functions. Note that the top row of this matrix contains the header for the variables thus the row numbers of the coefficients in the matrix will appear to be one larger than book texts show without such a header.

d. simplex\newTab(numVars, numSlacks, numArts)
Occasionally the book will call for creating a tableau directly. This function will create a tableau template customized for our number of regular variables, number of slack variables and number of artificial variables as per the following example. As in the problem definition template, copy this into an assignment statement and edit the coefficients accordingly.

$$\text{simplex\newtab}(3,2,0) \rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & s_1 & s_2 & z_0 & \_ \\ ? & ? & ? & 1 & 0 & 0 & ? \\ ? & ? & ? & 0 & 1 & 0 & ? \\ ? & ? & ? & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$c:=\begin{bmatrix} x_1 & x_2 & x_3 & s_1 & s_2 & z_0 & \_ \\ 1 & 6 & 8 & 1 & 0 & 0 & 118 \\ 1 & 5 & 10 & 0 & 1 & 0 & 220 \\ -8 & -3 & -1 & 0 & 0 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & s_1 & s_2 & z_0 & \_ \\ 1 & 6 & 8 & 1 & 0 & 0 & 118 \\ 1 & 5 & 10 & 0 & 1 & 0 & 220 \\ -8 & -3 & -1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

e. simplex\pivot(tab, row, col)
The book on linear programming describes a pivot. A pivot turns the pivot column into a basic variable with a single non-zero entry in that column. A pivot is simply a multiply-row-and-add operation where each row requires a separate such operation. Let's create a function that repeats that for each row and save ourselves considerable work as per the following example. Assume we wish to pivot upon the 3rd row, first column. Note:

remember with our header row the row number is one greater than one would expect.

$$\mathbf{b}:=\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & s_1 & s_2 & s_3 & s_4 & z_0 & \_ \\ 16 & 19 & 23 & 15 & 21 & 1 & 0 & 0 & 0 & 0 & 42000 \\ 15 & 10 & 19 & 23 & 10 & 0 & 1 & 0 & 0 & 0 & 25000 \\ 9 & 16 & 14 & 12 & 11 & 0 & 0 & 1 & 0 & 0 & 23000 \\ 18 & 20 & 15 & 17 & 19 & 0 & 0 & 0 & 1 & 0 & 36000 \\ -37 & -34 & -36 & -30 & -35 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{c}:=\mathbf{simplex\backslash pivot}(\mathbf{b},3,1)$$

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & s_1 & s_2 & s_3 & s_4 & z_0 & \_ \\ 0 & \frac{25}{3} & \frac{41}{15} & \frac{-143}{15} & \frac{31}{3} & 1 & \frac{-16}{15} & 0 & 0 & 0 & \frac{46000}{3} \\ 1 & \frac{2}{3} & \frac{19}{15} & \frac{23}{15} & \frac{2}{3} & 0 & \frac{1}{15} & 0 & 0 & 0 & \frac{5000}{3} \\ 0 & 10 & \frac{13}{5} & \frac{-9}{5} & 5 & 0 & \frac{-3}{5} & 1 & 0 & 0 & 8000 \\ 0 & 8 & \frac{-39}{5} & \frac{-53}{5} & 7 & 0 & \frac{-6}{5} & 0 & 1 & 0 & 6000 \\ 0 & \frac{-28}{3} & \frac{163}{15} & \frac{401}{15} & \frac{-31}{3} & 0 & \frac{37}{15} & 0 & 0 & 1 & \frac{185000}{3} \end{bmatrix}$$

f. simplex/unitize(tab, col)

Continuing with the same problem, the book teaches us how to determine the next pivot to continue the optimization process. We take the most negative indicator in the lower row and that will be our pivot column. In our case, column 5. Within that column we form quotients with the right most column and our selected pivot column and choose the least non-negative one. After some painful arithmetic, we discover our next pivot cell is row 5, column 5. Performing that pivot we get to:

$$d := \text{simplex\pivot}(c,5,5)$$

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $z_0$ | _ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | $\frac{-73}{21}$ | $\frac{1496}{105}$ | $\frac{214}{35}$ | 0 | 1 | $\frac{74}{105}$ | 0 | $\frac{-31}{21}$ | 0 | $\frac{136000}{21}$ |
| 1 | $\frac{-2}{21}$ | $\frac{211}{105}$ | $\frac{89}{35}$ | 0 | 0 | $\frac{19}{105}$ | 0 | $\frac{-2}{21}$ | 0 | $\frac{23000}{21}$ |
| 0 | $\frac{30}{7}$ | $\frac{286}{35}$ | $\frac{202}{35}$ | 0 | 0 | $\frac{9}{35}$ | 1 | $\frac{-5}{7}$ | 0 | $\frac{26000}{7}$ |
| 0 | $\frac{8}{7}$ | $\frac{-39}{35}$ | $\frac{-53}{35}$ | 1 | 0 | $\frac{-6}{35}$ | 0 | $\frac{1}{7}$ | 0 | $\frac{6000}{7}$ |
| 0 | $\frac{52}{21}$ | $\frac{-68}{105}$ | $\frac{388}{35}$ | 0 | 0 | $\frac{73}{105}$ | 0 | $\frac{31}{21}$ | 1 | $\frac{1481000}{21}$ |

It's easy determining our next pivot column since there is only one negative indicator in the lower row. It's column 3. Now we do some more, even more painful arithmetic to … whoa doggies, let's make this a bit easier. Let's unitize our next pivot column. That is, multiply each row by a positive number such that each entry is a one, negative one, or a zero. That would avoid the fraction in our denominator when computing our quotients. This leaves us with:

$$\text{simplex\unitize}(d,3)$$

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $z_0$ | _ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | $\frac{-365}{1496}$ | 1 | $\frac{321}{748}$ | 0 | $\frac{105}{1496}$ | $\frac{37}{748}$ | 0 | $\frac{-155}{1496}$ | 0 | $\frac{5000}{11}$ |
| $\frac{105}{211}$ | $\frac{-10}{211}$ | 1 | $\frac{267}{211}$ | 0 | 0 | $\frac{19}{211}$ | 0 | $\frac{-10}{211}$ | 0 | $\frac{115000}{211}$ |
| 0 | $\frac{75}{143}$ | 1 | $\frac{101}{143}$ | 0 | 0 | $\frac{9}{286}$ | $\frac{35}{286}$ | $\frac{-25}{286}$ | 0 | $\frac{5000}{11}$ |
| 0 | $\frac{40}{39}$ | -1 | $\frac{-53}{39}$ | $\frac{35}{39}$ | 0 | $\frac{-2}{13}$ | 0 | $\frac{5}{39}$ | 0 | $\frac{10000}{13}$ |
| 0 | $\frac{52}{21}$ | $\frac{-68}{105}$ | $\frac{388}{35}$ | 0 | 0 | $\frac{73}{105}$ | 0 | $\frac{31}{21}$ | 1 | $\frac{1481000}{21}$ |

The Unitize() function did it's job well but we still have those nasty fractions in our right most column. We now use the feature of the TI-Nspire to go to approximate results by using a Cntl-Enter with our unitize function.

| $x_4$ | $x_5$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $z_0$ | _ |
|---|---|---|---|---|---|---|---|
| 0.429144 | 0. | 0.070187 | 0.049465 | 0. | -0.10361 | 0. | 454.545 |
| 1.2654 | 0. | 0. | 0.090047 | 0. | -0.047393 | 0. | 545.024 |
| 0.706294 | 0. | 0. | 0.031469 | 0.122378 | -0.087413 | 0. | 454.545 |
| -1.35897 | 0.897436 | 0. | -0.153846 | 0. | 0.128205 | 0. | 769.231 |
| 11.0857 | 0. | 0. | 0.695238 | 0. | 1.47619 | 1. | 70523.8 |

Now we see clearly the least non-negative row and know our next pivot is row 2, column3.

e:=simplex\pivot(d,2,3)

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $z_0$ | _ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | $\frac{-365}{1496}$ | 1 | $\frac{321}{748}$ | 0 | $\frac{105}{1496}$ | $\frac{37}{748}$ | 0 | $\frac{-155}{1496}$ | 0 | $\frac{5000}{11}$ |
| 1 | $\frac{591}{1496}$ | 0 | $\frac{1257}{748}$ | 0 | $\frac{-211}{1496}$ | $\frac{61}{748}$ | 0 | $\frac{169}{1496}$ | 0 | $\frac{2000}{11}$ |
| 0 | $\frac{427}{68}$ | 0 | $\frac{77}{34}$ | 0 | $\frac{-39}{68}$ | $\frac{-5}{34}$ | 1 | $\frac{9}{68}$ | 0 | 0 |
| 0 | $\frac{1303}{1496}$ | 0 | $\frac{-775}{748}$ | 1 | $\frac{117}{1496}$ | $\frac{-87}{748}$ | 0 | $\frac{41}{1496}$ | 0 | $\frac{15000}{11}$ |
| 0 | $\frac{51}{22}$ | 0 | $\frac{125}{11}$ | 0 | $\frac{1}{22}$ | $\frac{8}{11}$ | 0 | $\frac{31}{22}$ | 1 | $\frac{779000}{11}$ |

g. simplex\solution(tab)

Continuing with the same problem, we know we have found the optimal solution because there are no negative indicators in the last row. Now we just need to interpret the tableau and extract those variable values. The rules are clear: if a column contains more than one entry the variable value is zero. If there is only one, it's the ratio between the last column and that value. Again, know the rule and how to apply it, then use this function to make it easy and accurate.

simplex\solution(e)

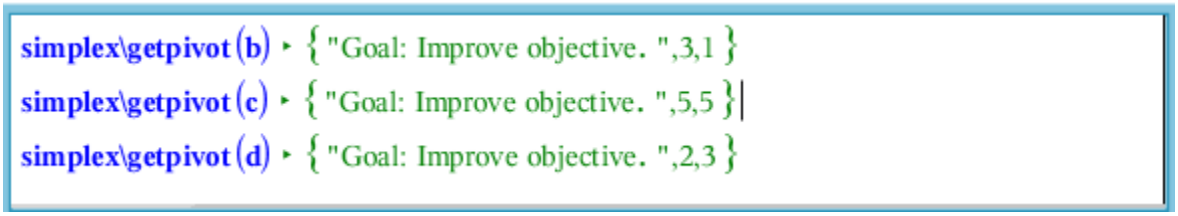
$$\left\{x_1=\frac{2000}{11}, x_2=0, x_3=\frac{5000}{11}, x_4=0, x_5=\frac{15000}{11}, s_1=0, s_2=0, s_3=0, s_4=0, z_0=\frac{779000}{11}\right\}$$

h. simplex\getpivot(tab)

One final function and we are finished with solving Standard Maximization Problems. Determining the next pivot is a pretty straight forward process. Choose the column with

the largest negative number in the last row, then form the quotients of the last column and that column, and finally choose the least non-negative row. Let's automate that and be finished.

We still have the three tableaus (b, c, and d) from the above exercises. Let's use our last function on those three tableaus as follows:




This function returns a list containing three items: the goal of the pivot, the row of the pivot and the column of the pivot. If you check you will find those pivots agree with what we determined manually above.

i. Putting it all together

The examples above have used the TI-Nspire Notes Application. The equations are in Math Boxes. Math Boxes have an attribute that allows you to hide the output from display. The following shows the entire process in solving this particular Standard Maximization Problem from problem definition through to extracting the solution. Most

of the actual output has been hidden to conserve space.

**simplex\pblmmax**(5,4) ▸

| _ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | _ | _ |
|---|---|---|---|---|---|---|---|
| *maximize_z* | ? | ? | ? | ? | ? | _ | _ |
| *subject_to* | _ | _ | _ | _ | _ | _ | _ |
| *eqn1* | ? | ? | ? | ? | ? | "≤" | ? |
| *eqn2* | ? | ? | ? | ? | ? | "≤" | ? |
| *eqn3* | ? | ? | ? | ? | ? | "≤" | ? |
| *eqn4* | ? | ? | ? | ? | ? | "≤" | ? |

**a**:=

| _ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | _ | _ |
|---|---|---|---|---|---|---|---|
| *maximize_z* | 37 | 34 | 36 | 30 | 35 | _ | _ |
| *subject_to* | _ | _ | _ | _ | _ | _ | _ |
| *eqn1* | 16 | 19 | 23 | 15 | 21 | "≤" | 42000 |
| *eqn2* | 15 | 10 | 19 | 23 | 10 | "≤" | 25000 |
| *eqn3* | 9 | 16 | 14 | 12 | 11 | "≤" | 23000 |
| *eqn4* | 18 | 20 | 15 | 17 | 19 | "≤" | 36000 |

**b**:=**simplex\pblmtotab**(**a**)

**simplex\getpivot**(**b**) ▸ { "Goal: Improve objective. ",3,1 }

**c**:=**simplex\pivot**(**b**,3,1)

**simplex\getpivot**(**c**) ▸ { "Goal: Improve objective. ",5,5 }

**d**:=**simplex\pivot**(**c**,5,5)

**simplex\getpivot**(**d**) ▸ { "Goal: Improve objective. ",2,3 }

**e**:=**simplex\pivot**(**d**,2,3)

**simplex\getpivot**(**e**) ▸ Optimum solution has been attained.

**simplex\solution**(**e**)

▸ $\left\{ x_1=\frac{2000}{11}, x_2=0, x_3=\frac{5000}{11}, x_4=0, x_5=\frac{15000}{11}, s_1=0, s_2=0, s_3=0, s_4=0, z_0=\frac{779000}{11} \right\}$

## 5. Standard Minimization Problems

This section describes another type of linear programming problem, the Standard Minimization Problem.

### a. Dual/Shadow Problems

Let's take another look at the initial tableau for the last problem of the previous section.

$$a \rightarrow \begin{bmatrix} \_ & x_1 & x_2 & x_3 & x_4 & x_5 & \_ & \_ \\ maximize\_z & 37 & 34 & 36 & 30 & 35 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 19 & 23 & 15 & 21 & "\leq" & 42000 \\ eqn2 & 15 & 10 & 19 & 23 & 10 & "\leq" & 25000 \\ eqn3 & 9 & 16 & 14 & 12 & 11 & "\leq" & 23000 \\ eqn4 & 18 & 20 & 15 & 17 & 19 & "\leq" & 36000 \end{bmatrix}$$

$$b:=simplex\backslash pblmtotab(a) \rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & s_1 & s_2 & s_3 & s_4 & z_0 & \_ \\ 16 & 19 & 23 & 15 & 21 & 1 & 0 & 0 & 0 & 0 & 42000 \\ 15 & 10 & 19 & 23 & 10 & 0 & 1 & 0 & 0 & 0 & 25000 \\ 9 & 16 & 14 & 12 & 11 & 0 & 0 & 1 & 0 & 0 & 23000 \\ 18 & 20 & 15 & 17 & 19 & 0 & 0 & 0 & 1 & 0 & 36000 \\ -37 & -34 & -36 & -30 & -35 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Keep in mind that each row represents an equality equation. As the $x_i$ grow larger, seeking a maximum, the poor little $s_i$ are between a rock and a hard place. They have to yield to the growing $x_i$ without exceeding the rightmost constraint constant. When the problem finally gets to the solution to the maximization problem, the $s_i$ have become as small as they could possibly get. In other words, they are a minimum.

While solving a maximization problem, we have solved a minimization problem as well. If we were to "hide" our minimization variables as slack variables we could solve a certain minimization problem in the process. These complimentary problems are known as "Dual" or "Shadow" problems.

### b. Simplex\dual(pblm)

Fortunately, as a book on solving linear equation will note, these "Dual" problems are easily derived with some matrix transforms and shuffling. The objective and constraint coefficients are exchanged and, with a few transforms, we see the Dual. Continuing with our example problem:

```
simplex\dual(
 _           x1   x2   x3   x4   x5   _    _
 maximize_z  37   34   36   30   35   _    _
 subject_to  _    _    _    _    _    _    _
 eqn1        16   19   23   15   21   "≤"  42000
 eqn2        15   10   19   23   10   "≤"  25000
 eqn3        9    16   14   12   11   "≤"  23000
 eqn4        18   20   15   17   19   "≤"  36000 )

 _           y1     y2     y3     y4     _    _
 minimize_w  42000  25000  23000  36000  _    _
 subject_to  _      _      _      _      _    _
 eqn1        16     15     9      18     "≥"  37
 eqn2        19     10     16     20     "≥"  34
 eqn3        23     19     14     15     "≥"  36
 eqn4        15     23     12     17     "≥"  30
 eqn5        21     10     11     19     "≥"  35
```

This follows the custom of naming minimization problem variables as $y_i$ as a reminder of the problem type. Also, the objective function is noted as a "w" instead of a "z".

Guess what we get if we take the Dual of a Dual?

```
simplex\dual(simplex\dual(
 _           x1   x2   x3   x4   x5   _    _
 maximize_z  37   34   36   30   35   _    _
 subject_to  _    _    _    _    _    _    _
 eqn1        16   19   23   15   21   "≤"  42000
 eqn2        15   10   19   23   10   "≤"  25000
 eqn3        9    16   14   12   11   "≤"  23000
 eqn4        18   20   15   17   19   "≤"  36000 ))

 _           x1   x2   x3   x4   x5   _    _
 maximize_z  37   34   36   30   35   _    _
 subject_to  _    _    _    _    _    _    _
 eqn1        16   19   23   15   21   "≤"  42000
 eqn2        15   10   19   23   10   "≤"  25000
 eqn3        9    16   14   12   11   "≤"  23000
 eqn4        18   20   15   17   19   "≤"  36000
```

Is this fun or what?

c. Solving our first Standard Minimization Problem.

We'll contrive a problem that we already know how to solve. Note we have a mate to our old simplex\pblmmax() for minimization problems. i.e., simplex\pblmmin().

$$\text{simplex\pblmmin}(4,5)\rightarrow \begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & ? & ? & ? & ? & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & ? & ? & ? & ? & "\geq" & ? \\ eqn2 & ? & ? & ? & ? & "\geq" & ? \\ eqn3 & ? & ? & ? & ? & "\geq" & ? \\ eqn4 & ? & ? & ? & ? & "\geq" & ? \\ eqn5 & ? & ? & ? & ? & "\geq" & ? \end{bmatrix}$$

$$\mathbf{a}:= \begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$$\mathbf{b}:=\text{simplex\dual}(\mathbf{a})\rightarrow \begin{bmatrix} \_ & x_1 & x_2 & x_3 & x_4 & x_5 & \_ & \_ \\ maximize\_z & 37 & 34 & 36 & 30 & 35 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 19 & 23 & 15 & 21 & "\leq" & 42000 \\ eqn2 & 15 & 10 & 19 & 23 & 10 & "\leq" & 25000 \\ eqn3 & 9 & 16 & 14 & 12 & 11 & "\leq" & 23000 \\ eqn4 & 18 & 20 & 15 & 17 & 19 & "\leq" & 36000 \end{bmatrix}$$

Now it's off to the races as before solving a maximization problem: Note we are hiding

the Math Box outputs to conserve space.

```
c:=simplex\pblmtotab(b)
simplex\getpivot(c) → {"Goal: Improve objective. ",3,1}
d:=simplex\pivot(c,3,1)
simplex\getpivot(d) → {"Goal: Improve objective. ",5,5}
e:=simplex\pivot(d,5,5)
simplex\getpivot(e) → {"Goal: Improve objective. ",2,3}
f:=simplex\pivot(e,2,3)
simplex\getpivot(f) → Optimum solution has been attained.
```

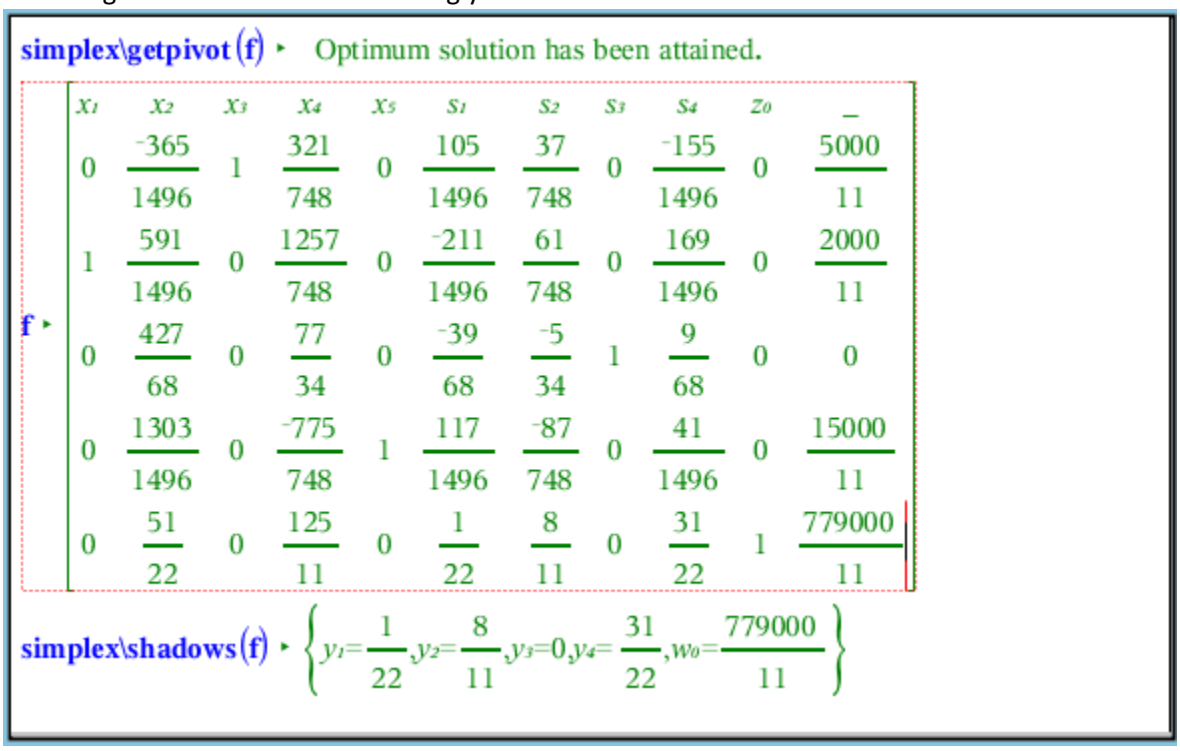
$$f \rightarrow \begin{array}{cccccccccccc} x_1 & & x_2 & & x_3 & & x_4 & & x_5 & s_1 & s_2 & s_3 & s_4 & z_0 & \_ \\ 0 & & \frac{-365}{1496} & & 1 & & \frac{321}{748} & & 0 & \frac{105}{1496} & \frac{37}{748} & 0 & \frac{-155}{1496} & 0 & \frac{5000}{11} \\ 1 & & \frac{591}{1496} & & 0 & & \frac{1257}{748} & & 0 & \frac{-211}{1496} & \frac{61}{748} & 0 & \frac{169}{1496} & 0 & \frac{2000}{11} \\ 0 & & \frac{427}{68} & & 0 & & \frac{77}{34} & & 0 & \frac{-39}{68} & \frac{-5}{34} & 1 & \frac{9}{68} & 0 & 0 \\ 0 & & \frac{1303}{1496} & & 0 & & \frac{-775}{748} & & 1 & \frac{117}{1496} & \frac{-87}{748} & 0 & \frac{41}{1496} & 0 & \frac{15000}{11} \\ 0 & & \frac{51}{22} & & 0 & & \frac{125}{11} & & 0 & \frac{1}{22} & \frac{8}{11} & 0 & \frac{31}{22} & 1 & \frac{779000}{11} \end{array}$$

At this point we have finished the optimization of both our maximization and minimization problems and the answers are somewhere hidden in the final tableau of the f variable.

d. Simplex\shadows(tab)

We could easily extract the answer for our maximization problem using our old simplex\solution() function but we aren't interested in that. We want the solution to the dual/shadow minimization problem. Remember we "hid" our original minimization $y_i$ variables as $s_i$ variables. We did some tricky stuff using transforms, etc. Bottom line, as the books will note, our solutions are hidden in the bottom row under the associated $s_i$ variables. i.e., $y_1=s_1$, etc. We need a function that will extract those solutions while

renaming the variables back accordingly.



e. Putting it all together while hiding output on intermediate calculations.

$$\mathbf{a}:=\begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$\mathbf{b}:=\mathbf{simplex\backslash dual}(\mathbf{a})$

$\mathbf{c}:=\mathbf{simplex\backslash pblmtotab}(\mathbf{b})$

$\mathbf{simplex\backslash getpivot}(\mathbf{c}) \rightarrow \{"Goal: Improve objective. ",3,1\}$

$\mathbf{d}:=\mathbf{simplex\backslash pivot}(\mathbf{c},3,1)$

$\mathbf{simplex\backslash getpivot}(\mathbf{d}) \rightarrow \{"Goal: Improve objective. ",5,5\}$

$\mathbf{e}:=\mathbf{simplex\backslash pivot}(\mathbf{e},5,5)$

$\mathbf{simplex\backslash getpivot}(\mathbf{e}) \rightarrow \{"Goal: Improve objective. ",2,3\}$

$\mathbf{f}:=\mathbf{simplex\backslash pivot}(\mathbf{e},2,3)$

$\mathbf{simplex\backslash getpivot}(\mathbf{f}) \rightarrow$ Optimum solution has been attained.

$$\mathbf{simplex\backslash shadows}(\mathbf{f}) \rightarrow \left\{ y_1=\frac{1}{22}, y_2=\frac{8}{11}, y_3=0, y_4=\frac{31}{22}, w_0=\frac{779000}{11} \right\}$$

## 6. Non-standard Problems

This section described the third type of linear programming problems, Non-Standard Problems.

a. Where are we going?

Currently we can solve Standard Maximization Problems. That is, the inequalities are all ≤ relations. We can also solve Standard Minimization Problems. That is, all ≥ relations. We do that by converting into Standard Maximization Problems. That's the best we can do for now.

In this section we address the rest of the possible relations in three steps. First, maximization problems with a mixture of ≤ and ≥ relations. Second, we add = relations. And last, we add non-standard minimization problems.

b. Mixed ≥ and ≤ constraints.

When we started with all ≤ constraints, we started with a feasible solution. i.e., the origin. When we add some ≥ constraints, we exclude the origin thus aren't starting with a feasible solution.

Following our book, Finite Mathematics, we add "surplus slack variables" which, unlike standard slack variables, start with negative signs. We also reorder our equations such that all the surplus slack variables are last. Finally, we do special pivots to get a feasible solution before we begin pivoting to the optimum solution.

Consider the following problem which has a mixture of ≤ and ≥ relations:

$$\mathbf{a}:=\begin{bmatrix} \_ & x_1 & x_2 & x_3 & \_ & \_ \\ maximize\_z & 120 & 40 & 60 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 1 & 1 & 1 & "\leq" & 100 \\ eqn2 & 1 & 1 & 1 & "\geq" & 60 \\ eqn3 & 10 & 4 & 7 & "\leq" & 500 \end{bmatrix}$$

Our old friend simplex\pblmtotab shows us it's previously unexercised ability to reorder and introduce surplus slacks:

$$\mathbf{b}:=\mathbf{simplex\backslash pblmtotab}(\mathbf{a}) \rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & s_1 & s_2 & s_3 & z_0 & \_ \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 100 \\ 10 & 4 & 7 & 0 & 1 & 0 & 0 & 500 \\ 1 & 1 & 1 & 0 & 0 & -1 & 0 & 60 \\ -120 & -40 & -60 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Our book, Finite Mathematics, tells us how to select a pivot to progress to a feasible solution. We take the -1 in our surplus slack, move to the left to the most distant positive cell, then form quotients in that column, select the least and that is our next pivot:

$$\text{simplex\getpivot}(b) \rightarrow \{\text{"Goal: Make } s_3 \geq 0.\text{"},3,1\}$$

$$c := \text{simplex\pivot}(b,3,1)$$

$$\text{simplex\getpivot}(c) \rightarrow \{\text{"Goal: Make } s_3 \geq 0.\text{"},4,2\}$$

$$d := \text{simplex\pivot}(c,4,2)$$

$$\text{simplex\getpivot}(d) \rightarrow \text{Optimum solution has been attained.}$$

$$\text{simplex\solution}(d) \rightarrow \left\{x_1=\frac{130}{3}, x_2=\frac{50}{3}, x_3=0, s_1=40, s_2=0, s_3=0, z_0=\frac{17600}{3}\right\}$$

That first pivot didn't get us to a feasible solution so the getpivot suggested another, using the same algorithm, which did get us there and also, luckily, also got us the optimum solution.

So, with modifications to pblmtotab and getpivot, we can process inequalities with both ≥ and ≤ relations.

c. Adding = relations.

Now our last relation, the = sign. Our book tells us to introduce an "artificial" variable which we name $a_i$ to distinguish. Our simplex\pblmtotab, not surprisingly, already knows how to do that. Consider:

$$a := \begin{bmatrix} \_ & x_1 & x_2 & \_ & \_ \\ maximize\_z & 3 & 2 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ \\ eqn1 & 1 & 1 & \text{"="} & 50 \\ eqn2 & 4 & 2 & \text{"}\geq\text{"} & 120 \\ eqn3 & 5 & 2 & \text{"}\leq\text{"} & 200 \end{bmatrix}$$

$$b := \text{simplex\pblmtotab}(a) \rightarrow \begin{bmatrix} x_1 & x_2 & s_1 & s_2 & a_1 & z_0 & \_ \\ 5 & 2 & 1 & 0 & 0 & 0 & 200 \\ 4 & 2 & 0 & -1 & 0 & 0 & 120 \\ 1 & 1 & 0 & 0 & 1 & 0 & 50 \\ -3 & -2 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

And we know that we must pivot to force all artificial variables to zero before considering surplus slack or optimum pivots. Once we determine an artificial isn't zero, the pivot selection is the same process as for surplus slack variables. We find that simplex\getpivot is also smarter than we needed in the past.

```
simplex\getpivot(b) → {"Goal" Zero a1.",3,1}
c:=simplex\pivot(b,3,1)
simplex\getpivot(c) → {"Goal" Zero a1.",4,2}
d:=simplex\pivot(c,4,2)
simplex\getpivot(d) → {"Goal: Improve Objective.",2,4}
e:=simplex\pivot(d,2,4)
simplex\getpivot(e) → Optimum solution has been attained.
simplex\solution(e) → {x1=100/3, x2=50/3, s1=0, s2=140/3, a1=0, z0=400/3}
```

Actually, it took two pivots to care for the artificial variable but, in the process, we took care of the surplus slack variable as well. One more pivot for optimization and we have our solution.

d. Non-standard Minimization Problems.

We can now solve any combination of maximization problems. Let's work on getting the same power with minimization problems. Our book makes an interesting observation: if we were to plot the objective equation as it attains it's peak, the negative of that attains a valley. And … vice versa. Thus, if we were to take the negation of our objective in a minimization process, that would be a maximization process which we know how to solve. Could it possibly be that simple?

Guess what our simplex\pblmtotab function already knows how to do:

```
a:=
[   _          y1   y2   _    _ ]
[ minimize_w   3    2    _    _ ]
[ subject_to   _    _    _    _ ]
[   eqn1       1    3    "≤"  6 ]
[   eqn2       2    1    "≥"  3 ]

b:=simplex\pblmtotab(a) →
[ y1  y2  s1  s2  w0  _ ]
[ 1   3   1   0   0   6 ]
[ 2   1   0   -1  0   3 ]
[ 3   2   0   0   -1  0 ]
```

With that we are off to the races solving a maximization problem:

$$\text{simplex\getpivot}(\mathbf{b}) \rightarrow \{\text{"Goal: Make } s_2 \geq 0.\text{"},3,1\}$$

$$\mathbf{c} := \text{simplex\pivot}(\mathbf{b},3,1)$$

$$\text{simplex\getpivot}(\mathbf{c}) \rightarrow \text{Optimum solution has been attained.}$$

$$\text{simplex\solution}(\mathbf{c}) \rightarrow \left\{y_1=\frac{3}{2}, y_2=0, s_1=\frac{9}{2}, s_2=0, w_0=\frac{9}{2}\right\}$$

e. Where we ended!
We can now solve just about any optimization problem. Maximization, minimization, any mixture of ≥, ≤, and = constraints. Just … bring it on!

This does pose an interesting question: with the minimization process we just learned, why do we need the dual/shadow solution method that is restricted to Standard Maximization/Minimization Problems? Won't the newly learned process for minimization problems work for standard problems as well. The answer is: yes. But … not always as well.

Let's go back to the minimization problem we worked earlier:

$$\mathbf{a} := \begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & \text{"}\geq\text{"} & 37 \\ eqn2 & 19 & 10 & 16 & 20 & \text{"}\geq\text{"} & 34 \\ eqn3 & 23 & 19 & 14 & 15 & \text{"}\geq\text{"} & 36 \\ eqn4 & 15 & 23 & 12 & 17 & \text{"}\geq\text{"} & 30 \\ eqn5 & 21 & 10 & 11 & 19 & \text{"}\geq\text{"} & 35 \end{bmatrix}$$

Solving this using the dual/shadows method is as follows:

```
b:=simplex\dual(a)
c:=simplex\pblmtotab(b)
simplex\getpivot(c) → { "Goal: Improve Objective. ",3,1 }
d:=simplex\pivot(c,3,1)
simplex\getpivot(d) → { "Goal: Improve Objective.",5,5 }
e:=simplex\pivot(d,5,5)
simplex\getpivot(e) → { "Goal: Improve Objective. ",2,3 }
f:=simplex\pivot(e,2,3)
simplex\getpivot(f) → Optimum solution has been attained.
simplex\shadows(f) → { y1=1/22, y2=8/11, y3=0, y4=31/22, w0=779000/11 }
```

Note that this method needed three pivots. Now use the minimization process learned in this section:

```
m:=simplex\pblmtotab(a)
simplex\getpivot(m) → { "Goal: Make s5≥0. ",4,1 }
n:=simplex\pivot(m,4,1)
simplex\getpivot(n) → { "Goal: Make s5≥0. ",6,4 }
p:=simplex\pivot(n,6,4)
simplex\getpivot(p) → { "Goal: Make s4≥0. ",5,2 }
q:=simplex\pivot(p,5,2)
simplex\getpivot(q) → { "Goal: Make s2≥0. ",3,3 }
r:=simplex\pivot(q,3,3)
simplex\getpivot(r) → { "Goal: Make s1≥0. ",3,8 }
s:=simplex\pivot(r,3,8)
simplex\getpivot(s) → { "Goal: Make s1≥0. ",2,6 }
t:=simplex\pivot(s,2,6)
simplex\getpivot(t) → Optimum solution has been attained.
simplex\solution(t)
→ { y1=1/22, y2=8/11, y3=0, y4=31/22, s1=0, s2=51/22, s3=0, s4=125/11, s5=0, w0=779000/11 }
```

Both methods reached the same solution. This method, because of surplus slack

variables, required six pivots.

When working large problems, potentially hundreds of variables/equations, pivots can be very expensive. If a minimization process is within the Standard Minimization Problem constraints the dual/shadow process can be significantly more efficient.

## 7. Combined Functions

This section begins combining functions somewhat obscuring the underlying linear programming methods.

a. Where we are now.
   Up to now our objective has been to support learning Linear Programming from a book such as Finite Mathematics. Our functions have been carefully restricted to removing burdensome, tedious calculations but, and this is important, none have done anything the student cannot do himself. You should go back to each of the functions to this point and ask yourself if you could do the calculations using only a TI-84. If not, you should go back and review until you can.

b. Where we are going in this section.
   Now we will orient ourselves more to just getting to the solution.

c. Simplex\autopivot
   We note that the simplex/getpivot and simplex\pivot usually come in pairs. Let's combine them into a single function that actually does a pivot if appropriate and, if not appropriate, returns the reason. The six pivots come a bit easier:

$$\mathbf{a}:=\begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$\mathbf{b}:=\mathbf{simplex\backslash pblmtotab}(\mathbf{a})$

$\mathbf{c}:=\mathbf{simplex\backslash autopivot}(\mathbf{b})$

$\mathbf{d}:=\mathbf{simplex\backslash autopivot}(\mathbf{c})$

$\mathbf{e}:=\mathbf{simplex\backslash autopivot}(\mathbf{d})$

$\mathbf{f}:=\mathbf{simplex\backslash autopivot}(\mathbf{e})$

$\mathbf{g}:=\mathbf{simplex\backslash autopivot}(\mathbf{f})$

$\mathbf{h}:=\mathbf{simplex\backslash autopivot}(\mathbf{g})$

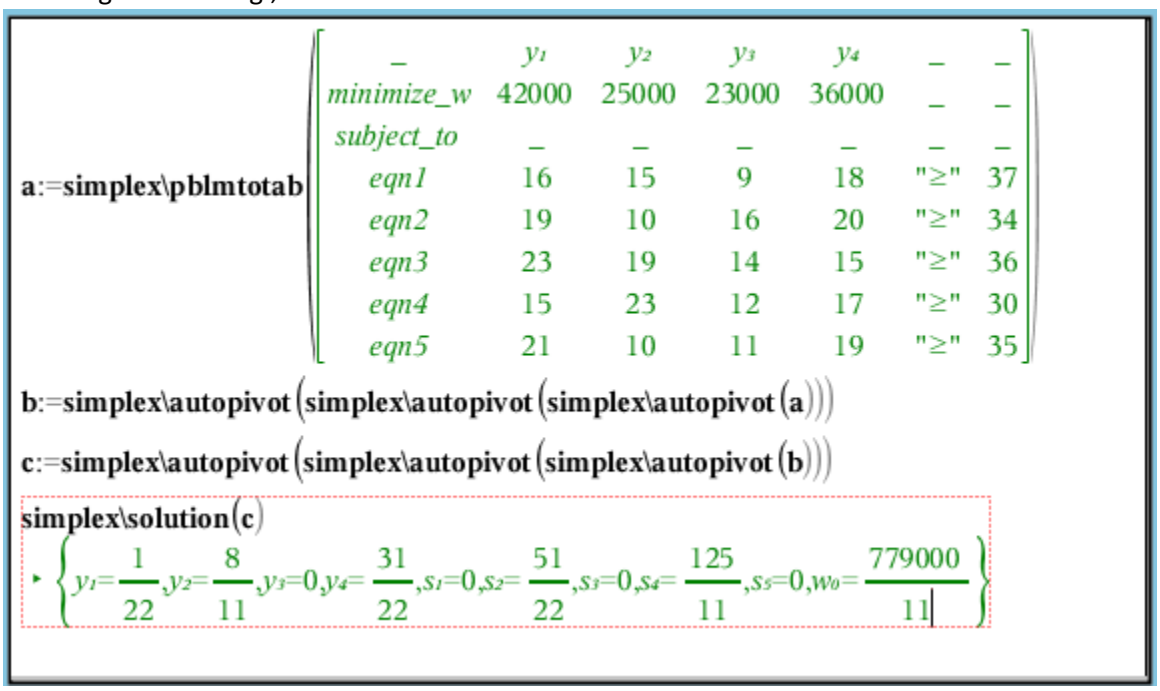
$\mathbf{j}:=\mathbf{simplex\backslash autopivot}(\mathbf{h})$ ▸ Optimum solution has been attained.

$\mathbf{simplex\backslash solution}(\mathbf{h})$

$$\blacktriangleright \left\{ y_1=\frac{1}{22}, y_2=\frac{8}{11}, y_3=0, y_4=\frac{31}{22}, s_1=0, s_2=\frac{51}{22}, s_3=0, s_4=\frac{125}{11}, s_5=0, w_0=\frac{779000}{11} \right\}$$

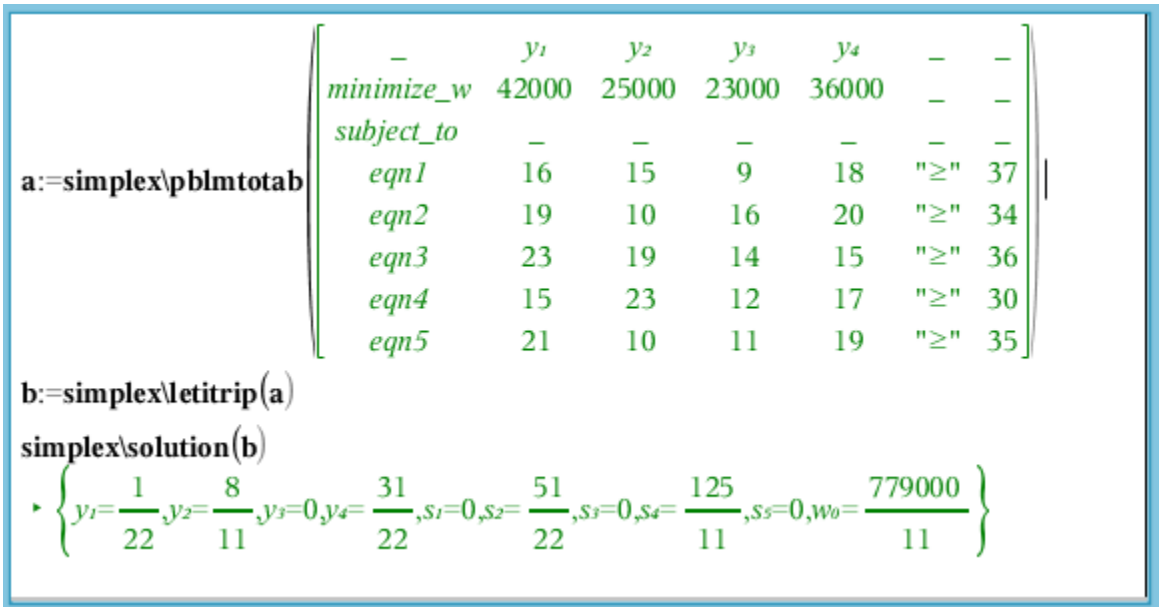
d. Nesting functions
   Notice how frequently the output of one function is the input of another. We could take

advantage of that. E.g., as follows:




e. Simplex\letitrip (Let-it-rip)

Look above where we end up repeating autopivots until there are no more to be done. Let's create a function that just starts pivoting and returns the last successful pivot when it is appropriate to stop. Our problem now appears as:




The process using dual/shadows is as follows:

$$\mathbf{a}:=\mathbf{simplex\backslash dual}\left(\begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}\right)$$

$$\mathbf{b}:=\mathbf{simplex\backslash pblmtotab}(\mathbf{a})$$

$$\mathbf{c}:=\mathbf{simplex\backslash letitrip}(\mathbf{b})$$

$$\mathbf{simplex\backslash shadows}(\mathbf{c}) \rightarrow \left\{y_1=\frac{1}{22}, y_2=\frac{8}{11}, y_3=0, y_4=\frac{31}{22}, w_0=\frac{779000}{11}\right\}$$

We could nest stuff:

$$\mathbf{a}:=\begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$$\mathbf{simplex\backslash solution}(\mathbf{simplex\backslash letitrip}(\mathbf{simplex\backslash pblmtotab}(\mathbf{a}))) \rightarrow \left\{y_1=\frac{1}{22}, y_2=\frac{8}{11}, y_3=0, y_4=\frac{31}{22}, s_1=0, s_2=\frac{51}{22}, s_3=0, s_4=\frac{125}{11}, s_5=0, w_0=\frac{779000}{11}\right\}$$

f. Simplex\theworks

The problem in nesting is that life isn't always pure. The nesting works well assuming there is an optimal solution. But it is badly behaved if there is no feasible solution or unbounded solutions. It's best if we surround the nesting with some logic that considers anomalies. theWorks function does that. Here it is for a well behaved problem:

$$\mathbf{a}:=\begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$$\mathbf{simplex\backslash theworks}(\mathbf{a})$$

$$\left\{ y_1=\frac{1}{22}, y_2=\frac{8}{11}, y_3=0, y_4=\frac{31}{22}, s_1=0, s_2=\frac{51}{22}, s_3=0, s_4=\frac{125}{11}, s_5=0, w_0=\frac{779000}{11} \right\}$$

And for a problem with no feasible solutions:

$$\mathbf{a}:=\begin{bmatrix} \_ & x_1 & x_2 & \_ & \_ \\ maximize\_z & 1 & 1 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ \\ eqn1 & 1 & 1 & "\leq" & 2 \\ eqn2 & 1 & 1 & "\geq" & 5 \end{bmatrix}$$

$$\mathbf{simplex\backslash theworks}(\mathbf{a}) \rightarrow \text{Error: The problem has no feasible solution.}$$

And for an unbounded region:

$$\mathbf{a}:=\begin{bmatrix} \_ & x_1 & x_2 & \_ & \_ \\ maximize\_z & 1 & 1 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ \\ eqn1 & 1 & 1 & "\geq" & 2 \\ eqn2 & 1 & 1 & "\geq" & 5 \end{bmatrix} \rightarrow \begin{bmatrix} \_ & x_1 & x_2 & \_ & \_ \\ maximize\_z & 1 & 1 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ \\ eqn1 & 1 & 1 & "\geq" & 2 \\ eqn2 & 1 & 1 & "\geq" & 5 \end{bmatrix}$$

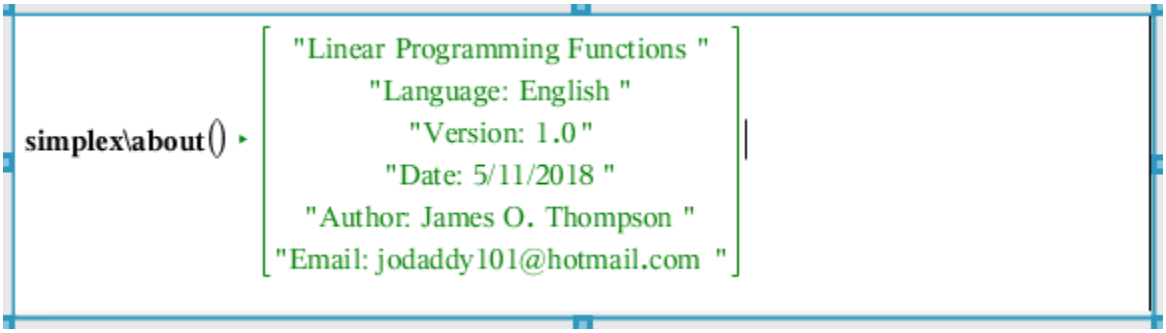
$$\mathbf{simplex\backslash theworks}(\mathbf{a}) \rightarrow \text{Error: Unbounded feasible region.}$$

## 8. Alphabetical Function Listing

These functions are for the end user and are in the library as public functions.

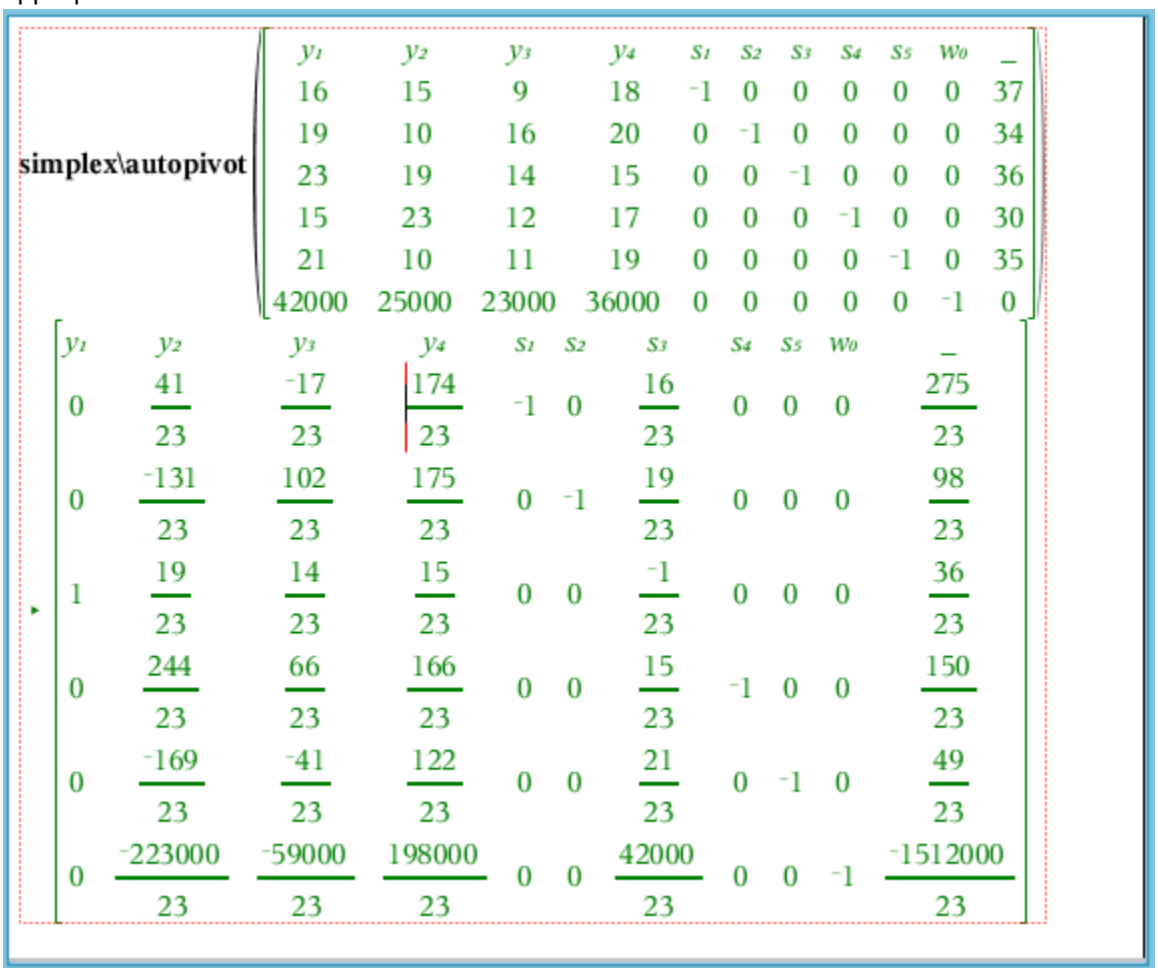
a. About()

This function gives some of the standard data:




b. AutoPivot(tab)

This function takes a tableau, executes a getpivot and either executes the pivot returning the next tableau or returns the message noting why more pivots are not appropriate.




c. Dual(pblm)

This function takes a Standard Maximization/Minimization Problem and returns it's

dual.

d:=simplex\dual

| _ | $y_1$ | $y_2$ | $y_3$ | $y_4$ | _ | _ |
|---|---|---|---|---|---|---|
| *minimize_w* | 42000 | 25000 | 23000 | 36000 | _ | _ |
| *subject_to* | _ | _ | _ | _ | _ | _ |
| *eqn1* | 16 | 15 | 9 | 18 | "≥" | 37 |
| *eqn2* | 19 | 10 | 16 | 20 | "≥" | 34 |
| *eqn3* | 23 | 19 | 14 | 15 | "≥" | 36 |
| *eqn4* | 15 | 23 | 12 | 17 | "≥" | 30 |
| *eqn5* | 21 | 10 | 11 | 19 | "≥" | 35 |

| _ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | _ | _ |
|---|---|---|---|---|---|---|---|
| *maximize_z* | 37 | 34 | 36 | 30 | 35 | _ | _ |
| *subject_to* | _ | _ | _ | _ | _ | _ | _ |
| *eqn1* | 16 | 19 | 23 | 15 | 21 | "≤" | 42000 |
| *eqn2* | 15 | 10 | 19 | 23 | 10 | "≤" | 25000 |
| *eqn3* | 9 | 16 | 14 | 12 | 11 | "≤" | 23000 |
| *eqn4* | 18 | 20 | 15 | 17 | 19 | "≤" | 36000 |

d. GetPivot(tab)
This function takes a tableau and returns either the next suggested pivot or a message indicating why more pivots are not appropriate.

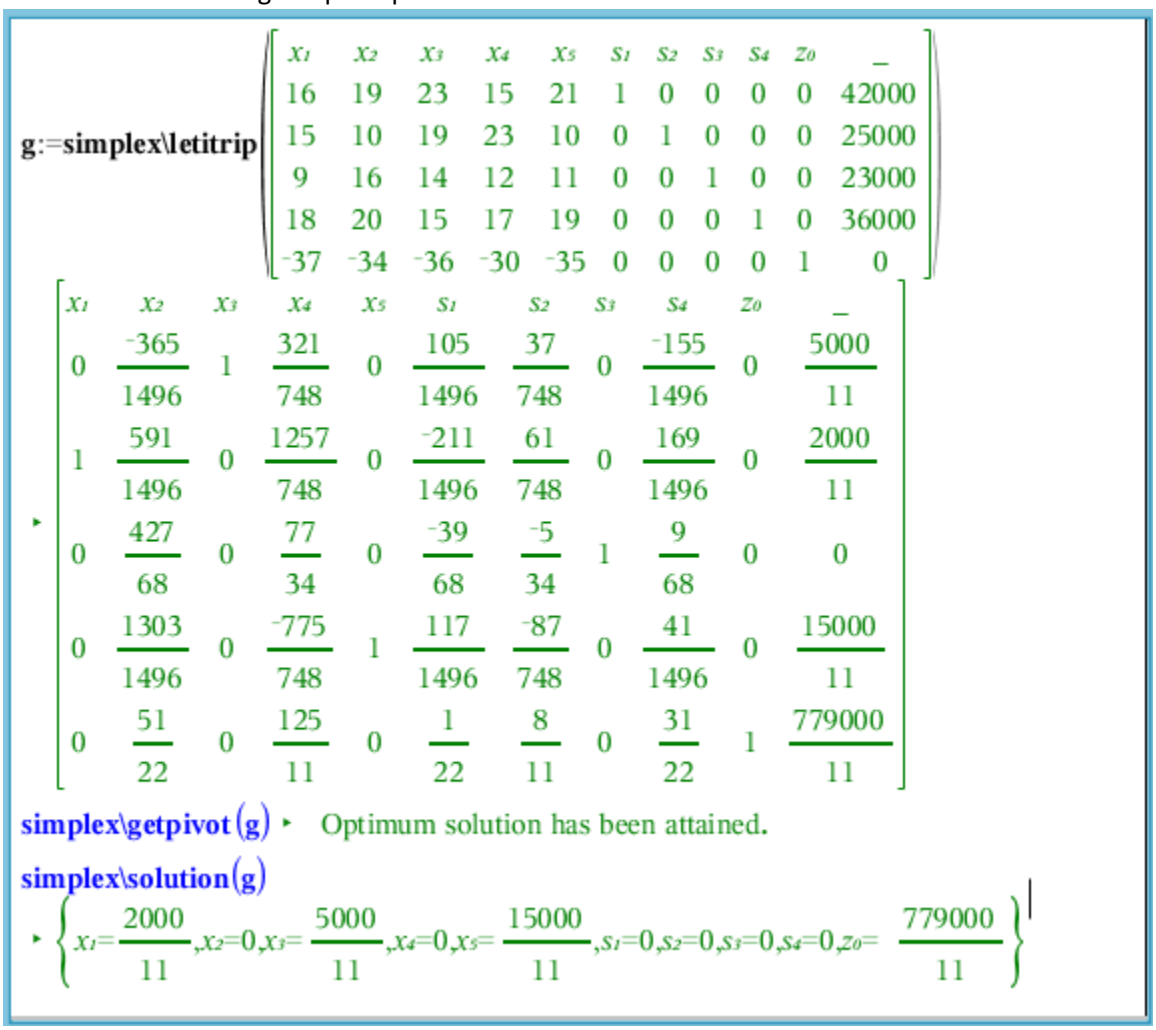
simplex\getpivot

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $z_0$ | _ |
|---|---|---|---|---|---|---|---|---|---|---|
| 16 | 19 | 23 | 15 | 21 | 1 | 0 | 0 | 0 | 0 | 42000 |
| 15 | 10 | 19 | 23 | 10 | 0 | 1 | 0 | 0 | 0 | 25000 |
| 9 | 16 | 14 | 12 | 11 | 0 | 0 | 1 | 0 | 0 | 23000 |
| 18 | 20 | 15 | 17 | 19 | 0 | 0 | 0 | 1 | 0 | 36000 |
| -37 | -34 | -36 | -30 | -35 | 0 | 0 | 0 | 0 | 1 | 0 |

{"Goal: Improve Objective. ",3,1}

e. LetItRip(tab)
This function takes a tableau and performs pivots until no longer appropriate returning the last successful pivot. Use a getpivot on the returned tableau to determine the

reason for terminating the pivot process.




f. NewTab(numVars, numslacks, numArts)
This function is used for cases where you want to begin with a tableau rather than a problem definition. It returns a template tableau to be copied into an assignment

statement where you edit the ?s to produce a tableau.

simplex\newtab(2,3,4) ▸

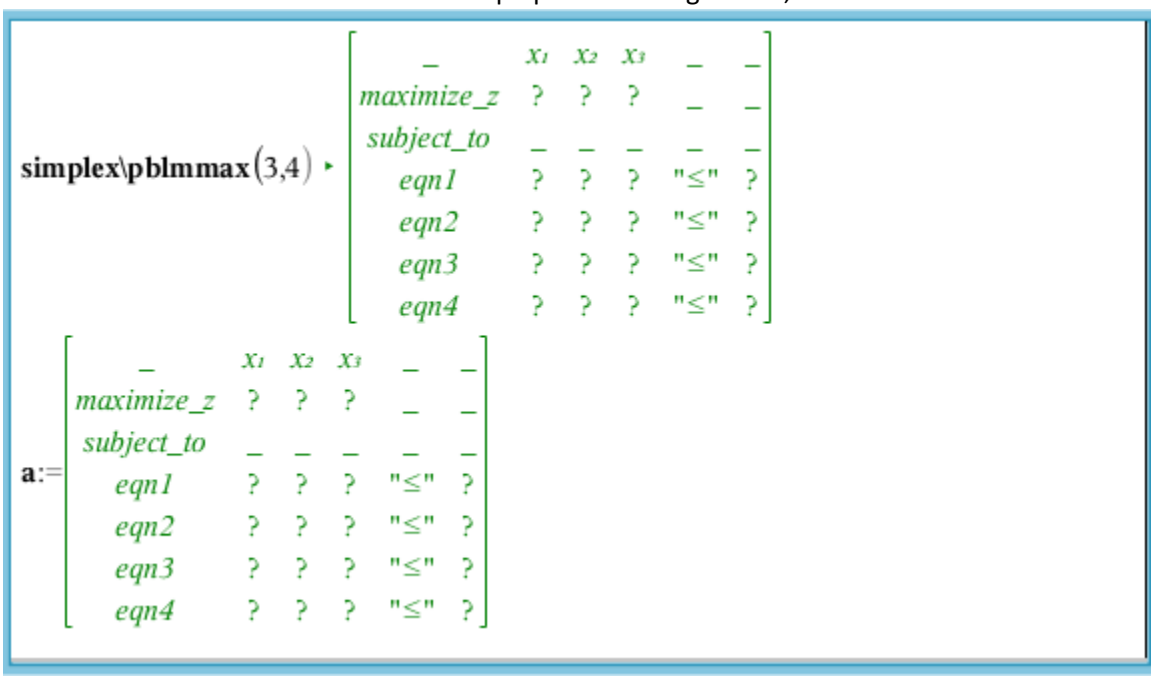
| $x_1$ | $x_2$ | $s_1$ | $s_2$ | $s_3$ | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $z_0$ | _ |
|---|---|---|---|---|---|---|---|---|---|---|
| ? | ? | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

a:=

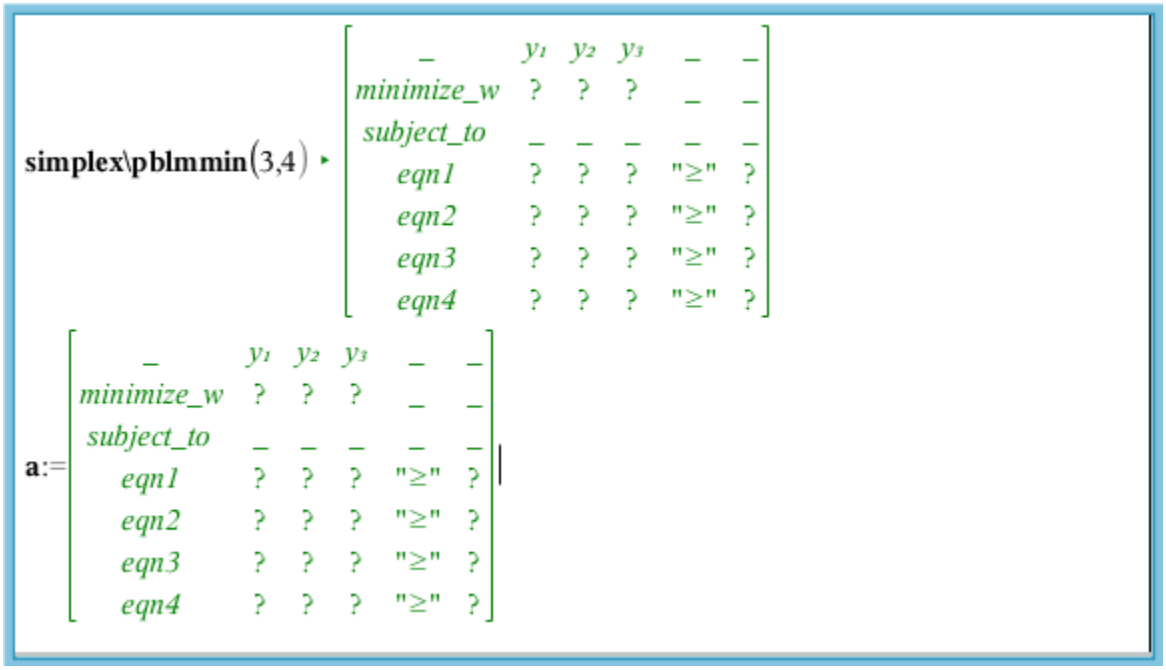
| $x_1$ | $x_2$ | $s_1$ | $s_2$ | $s_3$ | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $z_0$ | _ |
|---|---|---|---|---|---|---|---|---|---|---|
| ? | ? | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | ? |
| ? | ? | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

g. PblmMax(numVars, numEqns)
This function will return a template for a maximization problem definition, tailored for the stated number of variables and equations. The template can be copied into an assignment statement and edited to complete the problem definition. The ?'s may be

edited for coefficients and the relationship operator changed to ≤, = or ≥.




h. PblmMin(numVars, numEqns)
This function will return a template for a minimization problem definition, tailored for the stated number of variables and equations. The template can be copied into an assignment statement and edited to complete the problem definition. The ?'s may be edited for coefficients and the relationship operator changed to ≤, = or ≥.




i. PblmToTab(pblm)
This function will take a problem definition and return an appropriate tableau. It converts minimization problems to maximization by negating the objective function. It creates regular, surplus and artificial variables as appropriate for the various

relationships.

$$\mathbf{m}:=\begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\leq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "=" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "=" & 35 \end{bmatrix}$$

$$\mathbf{simplex\backslash pblmtotab}(\mathbf{m})$$

$$\begin{bmatrix} y_1 & y_2 & y_3 & y_4 & s_1 & s_2 & s_3 & a_1 & a_2 & w_0 & \_ \\ 19 & 10 & 16 & 20 & 1 & 0 & 0 & 0 & 0 & 0 & 34 \\ 16 & 15 & 9 & 18 & 0 & -1 & 0 & 0 & 0 & 0 & 37 \\ 15 & 23 & 12 & 17 & 0 & 0 & -1 & 0 & 0 & 0 & 30 \\ 23 & 19 & 14 & 15 & 0 & 0 & 0 & 1 & 0 & 0 & 36 \\ 21 & 10 & 11 & 19 & 0 & 0 & 0 & 0 & 1 & 0 & 35 \\ 42000 & 25000 & 23000 & 36000 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}$$

j. Pivot(tab, row, col)

This function accepts a tableau along with the desired pivot row and column. It performs the pivot and returns the new tableau. Note that with the header row, row

numbers will be one larger than expected.

$$\mathbf{d} \rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & s_1 & s_2 & s_3 & s_4 & z_0 & \_ \\ 16 & 19 & 23 & 15 & 21 & 1 & 0 & 0 & 0 & 0 & 42000 \\ 15 & 10 & 19 & 23 & 10 & 0 & 1 & 0 & 0 & 0 & 25000 \\ 9 & 16 & 14 & 12 & 11 & 0 & 0 & 1 & 0 & 0 & 23000 \\ 18 & 20 & 15 & 17 & 19 & 0 & 0 & 0 & 1 & 0 & 36000 \\ -37 & -34 & -36 & -30 & -35 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{simplex\backslash pivot}(\mathbf{d},3,1) \rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & s_1 & s_2 & s_3 & s_4 & z_0 & \_ \\ 0 & \frac{25}{3} & \frac{41}{15} & \frac{-143}{15} & \frac{31}{3} & 1 & \frac{-16}{15} & 0 & 0 & 0 & \frac{46000}{3} \\ 1 & \frac{2}{3} & \frac{19}{15} & \frac{23}{15} & \frac{2}{3} & 0 & \frac{1}{15} & 0 & 0 & 0 & \frac{5000}{3} \\ 0 & 10 & \frac{13}{5} & \frac{-9}{5} & 5 & 0 & \frac{-3}{5} & 1 & 0 & 0 & 8000 \\ 0 & 8 & \frac{-39}{5} & \frac{-53}{5} & 7 & 0 & \frac{-6}{5} & 0 & 1 & 0 & 6000 \\ 0 & \frac{-28}{3} & \frac{163}{15} & \frac{401}{15} & \frac{-31}{3} & 0 & \frac{37}{15} & 0 & 0 & 1 & \frac{185000}{3} \end{bmatrix}$$

k. Shadows(tab)

If a problem has been solved via dual\shadows, the solution is in the tableau but beneath the slack variables rather than in the rightmost column. This function will

extract the solution accordingly renaming the slack variables in the process.

$$\mathbf{c} \rightarrow \begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$$\mathbf{d} := \mathbf{simplex\backslash letitrip}(\mathbf{simplex\backslash pblmtotab}(\mathbf{simplex\backslash dual}(\mathbf{c})))$$

$$\rightarrow \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & s_1 & s_2 & s_3 & s_4 & z_0 & \_ \\ 0 & \frac{-365}{1496} & 1 & \frac{321}{748} & 0 & \frac{105}{1496} & \frac{37}{748} & 0 & \frac{-155}{1496} & 0 & \frac{5000}{11} \\ 1 & \frac{591}{1496} & 0 & \frac{1257}{748} & 0 & \frac{-211}{1496} & \frac{61}{748} & 0 & \frac{169}{1496} & 0 & \frac{2000}{11} \\ 0 & \frac{427}{68} & 0 & \frac{77}{34} & 0 & \frac{-39}{68} & \frac{-5}{34} & 1 & \frac{9}{68} & 0 & 0 \\ 0 & \frac{1303}{1496} & 0 & \frac{-775}{748} & 1 & \frac{117}{1496} & \frac{-87}{748} & 0 & \frac{41}{1496} & 0 & \frac{15000}{11} \\ 0 & \frac{51}{22} & 0 & \frac{125}{11} & 0 & \frac{1}{22} & \frac{8}{11} & 0 & \frac{31}{22} & 1 & \frac{779000}{11} \end{bmatrix}$$

$$\mathbf{simplex\backslash shadows}(\mathbf{d}) \rightarrow \left\{ y_1 = \frac{1}{22}, y_2 = \frac{8}{11}, y_3 = 0, y_4 = \frac{31}{22}, w_0 = \frac{779000}{11} \right\}$$

l. Solution(tab)

This function accepts the tableau for a solved problem, extracts and displays the

solution.

$$\mathbf{c} \rightarrow \begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$$\mathbf{d} := \mathbf{simplex\backslash letitrip}(\mathbf{simplex\backslash pblmtotab}(\mathbf{c}))$$

$$\blacktriangleright \begin{bmatrix} y_1 & y_2 & y_3 & y_4 & s_1 & s_2 & s_3 & s_4 & s_5 & w_0 & \_ \\ 0 & 0 & \frac{-427}{68} & 0 & \frac{-591}{1496} & 1 & \frac{365}{1496} & 0 & \frac{-1303}{1496} & 0 & \frac{51}{22} \\ 0 & 0 & \frac{-77}{34} & 0 & \frac{-1257}{748} & 0 & \frac{-321}{748} & 1 & \frac{775}{748} & 0 & \frac{125}{11} \\ 1 & 0 & \frac{39}{68} & 0 & \frac{211}{1496} & 0 & \frac{-105}{1496} & 0 & \frac{-117}{1496} & 0 & \frac{1}{22} \\ 0 & 1 & \frac{5}{34} & 0 & \frac{-61}{748} & 0 & \frac{-37}{748} & 0 & \frac{87}{748} & 0 & \frac{8}{11} \\ 0 & 0 & \frac{-9}{68} & 1 & \frac{-169}{1496} & 0 & \frac{155}{1496} & 0 & \frac{-41}{1496} & 0 & \frac{31}{22} \\ 0 & 0 & 0 & 0 & \frac{2000}{11} & 0 & \frac{5000}{11} & 0 & \frac{15000}{11} & -1 & \frac{-779000}{11} \end{bmatrix}$$

$$\mathbf{simplex\backslash solution}(\mathbf{d})$$

$$\blacktriangleright \left\{ y_1 = \frac{1}{22}, y_2 = \frac{8}{11}, y_3 = 0, y_4 = \frac{31}{22}, s_1 = 0, s_2 = \frac{51}{22}, s_3 = 0, s_4 = \frac{125}{11}, s_5 = 0, w_0 = \frac{779000}{11} \right\}$$

m. TheWorks(pblm)

This function does it all. It accepts a problem definition, computes the appropriate tableau, performs required pivots until a solution is reached, then displays the solution. If a problem has no feasible solution or an unbounded solution, it displays a message to

that effect.

$$\mathbf{c} \rightarrow \begin{bmatrix} \_ & y_1 & y_2 & y_3 & y_4 & \_ & \_ \\ minimize\_w & 42000 & 25000 & 23000 & 36000 & \_ & \_ \\ subject\_to & \_ & \_ & \_ & \_ & \_ & \_ \\ eqn1 & 16 & 15 & 9 & 18 & "\geq" & 37 \\ eqn2 & 19 & 10 & 16 & 20 & "\geq" & 34 \\ eqn3 & 23 & 19 & 14 & 15 & "\geq" & 36 \\ eqn4 & 15 & 23 & 12 & 17 & "\geq" & 30 \\ eqn5 & 21 & 10 & 11 & 19 & "\geq" & 35 \end{bmatrix}$$

$$\mathbf{simplex\backslash theworks}(\mathbf{c})$$

$$\rightarrow \left\{ y_1=\frac{1}{22}, y_2=\frac{8}{11}, y_3=0, y_4=\frac{31}{22}, s_1=0, s_2=\frac{51}{22}, s_3=0, s_4=\frac{125}{11}, s_5=0, w_0=\frac{779000}{11} \right\}$$

n. Unitize(tab, col)

This function will unitize a column of the given tableau. It does that by multiplying each row by a unitizing constant. This leaves a column containing only 1, -1 or zero. This can be useful in computing quotients for a pivot.

$$\mathbf{d} \rightarrow \begin{bmatrix} y_1 & y_2 & y_3 & y_4 & s_1 & s_2 & s_3 & s_4 & s_5 & w_0 & \_ \\ 16 & 15 & 9 & 18 & -1 & 0 & 0 & 0 & 0 & 0 & 37 \\ 19 & 10 & 16 & 20 & 0 & -1 & 0 & 0 & 0 & 0 & 34 \\ 23 & 19 & 14 & 15 & 0 & 0 & -1 & 0 & 0 & 0 & 36 \\ 15 & 23 & 12 & 17 & 0 & 0 & 0 & -1 & 0 & 0 & 30 \\ 21 & 10 & 11 & 19 & 0 & 0 & 0 & 0 & -1 & 0 & 35 \\ 42000 & 25000 & 23000 & 36000 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}$$

$$\mathbf{simplex\backslash unitize}(\mathbf{d},2)$$

$$\rightarrow \begin{bmatrix} y_1 & y_2 & y_3 & y_4 & s_1 & s_2 & s_3 & s_4 & s_5 & w_0 & \_ \\ \frac{16}{15} & 1 & \frac{3}{5} & \frac{6}{5} & \frac{-1}{15} & 0 & 0 & 0 & 0 & 0 & \frac{37}{15} \\ \frac{19}{10} & 1 & \frac{8}{5} & 2 & 0 & \frac{-1}{10} & 0 & 0 & 0 & 0 & \frac{17}{5} \\ \frac{23}{19} & 1 & \frac{14}{19} & \frac{15}{19} & 0 & 0 & \frac{-1}{19} & 0 & 0 & 0 & \frac{36}{19} \\ \frac{15}{23} & 1 & \frac{12}{23} & \frac{17}{23} & 0 & 0 & 0 & \frac{-1}{23} & 0 & 0 & \frac{30}{23} \\ \frac{21}{10} & 1 & \frac{11}{10} & \frac{19}{10} & 0 & 0 & 0 & 0 & \frac{-1}{10} & 0 & \frac{7}{2} \\ 42000 & 25000 & 23000 & 36000 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}$$

## 9. Helper Function Listing

These functions are not for the end user and are in the library as private functions. Because they are not for the end user they are not documented further.

a. AddSubscript
b. Feasible
c. getMsg
d. IsPblm
e. IsTableau
f. MatInsert
g. NewFunction
h. PblmNew
i. PivotToFix
j. PivotToImprove
k. PIvotToRow