Math/CS/Econ 210 Spring 2021 —HW # 4 Assignment Due Thu. Mar. 11, 5 PM

Submission Instructions: Problems 1, 3, 4, 5, and 6 can be done with your lab partner, with one of you submiting the notebook by downloading it as an `ipynb` file and uploading that to the HW4 Moodle link. You can submit five separate notebooks or merge them into a single notebook. Also, each student should upload a file with their solution to Problem 2.

Reading: Strayer 2.6 (and the lecture slides/recordings for Fri 3/5 and Mon 3/8). Note that we are skipping Strayer 2.4 and 2.7 for now; they present a way to solve minimization problems, but you will use a different method to solve minimization problems on this HW. We will come back to these ideas soon.

Reminder of HW collaboration rules: I encourage you to discuss HW problems with other students in the class and/or with me. Your submission should reflect your personal understanding, so you must write the solutions yourself without referring to notes from your collaborative work. For computer problems, this means you may not cut-and-paste (or anything of that nature) except from sample code that I provide or from your own previous work.

1 (CODING):

- Download the notebook `hw4prob1` from Moodle (Python section).
- Read the top portion, which is a demonstration of `while` statements.
- The next few cells provide my code for printing a tableau, `pivot`, `target`, and `select`.

Your job in Problem 1 is to write your own implementation of `SimplexBF`.

A skeleton code is provided near the bottom of the notebook. As you'll see in the skeleton, the function should return value 0 if it proceeds to a solution (you make all the $c_j \leq 0$) or the value -1 if the problem is unbounded.

Three details:

(1) Since `pivot` automatically prints the new tableau after computing it, your `SimplexBF` code will automatically show all the tableaus that are computed to help you assess whether it is working correctly.

(2) I suggest using a `while` loop (see tutorial at top of notebook), since we never know how many pivots will be required.

(3) Recall that my version of `pivot` outputs a new tableau but does not change the input tableau. When I run `pivot` by hand, I do something like:

```
a2 = pivot(a,1,1,indep_names,dep_names)
a3 = pivot(a2,2,2,indep_names,dep_names)
a4 = pivot(a3,1,3,indep_names,dep_names)
etc
```

so I make new names for each new tableau and feed each new tableau's variable name into the next call to `pivot`. However, inside `simplexbf`, we will pivot an unknown number of times, so the above approach is not practical. Instead, I suggest you do this:

```
anew = pivot(a,pivrow,pivcol,indep_names,dep_names)
a = np.copy(anew)
```

With these two commands, after `pivot` computes a new tableau `anew`, we copy the result back into `a`, so that every time we call `pivot`, the input tableau can be `a`.

NEXT PAGE FOR PROBLEM 2

2 (THEORY): This problem relates to the Theorem below.

SimplexNBF Theorem, Part I: When applying a step of `SimplexNBF` to some tableau, if the chosen Target (following the rule of `SimplexNBF`) is $b_i < 0$, then

$$b_{i,new} \geq b_i, \quad \text{and}$$

$$b_{k,new} \geq 0 \;\text{ for all } k > i.$$

(Here, $b_{i,new}$ and $b_{k,new}$ denote the values of b_i and b_k in the new tableau.)

Rather than prove this theorem in full generality, consider the specific setup below that captures the key ideas:

$$\begin{array}{|ccccc|l}
\cdots & x_3 & \cdots & x_n & -1 & \\
\hline
\cdots & a_{13} & \cdots & a_{1n} & b_1 & = -t_1 \\
\cdots & a_{23} & \cdots & a_{2n} & b_2 & = -t_2 \\
\cdots & a_{33} & \cdots & a_{3n} & b_3 & = -t_3 \\
\cdots & a_{43} & \cdots & a_{4n} & b_4 & = -t_4 \\
\hline
\cdots & c_3 & \cdots & c_n & d & = obj \\
\hline
\end{array}$$

Assume that $b_2 < 0$, $b_3 \geq 0$, and $b_4 \geq 0$, so that our choice of Target in `SimplexNBF` is b_2.

Assume also that $a_{23} < 0$, and that a_{23} is the *only* negative a-value in row 2.

Assume also that $a_{33} > 0$ and $a_{43} < 0$.

(a) Explain why the following three inequalities are true, for the case that $b_2/a_{23} > b_3/a_{33}$:

$$b_{2,new} \geq b_2, \qquad b_{3,new} \geq 0, \qquad b_{4,new} \geq 0.$$

(b) Explain why the same three inequalities are true for the case that $b_2/a_{23} < b_3/a_{33}$.

NEXT PAGE FOR PROBLEM 3

3 (THEORY, WITH SOME COMPUTATION BUT NOT CODING): This problem relates to the Theorem below.

SimplexNBF Theorem, Part II: If we are running `SimplexNBF`, and the choice of Target is b_i, and $a_{ij} > 0$ for all columns j, then the problem is infeasible.

(a) Use this theorem to show that there are no points (x, y, z, w) in $\mathbb{R}^4$ that obey:

$$\begin{aligned} -2x - 10y + 3z - 20w &\leq 8 \\ -x + y - 3w &\leq -6 \\ x + 4y - z + 8w &\leq -1 \\ x, y, z, w &\geq 0 \end{aligned}$$

(To do this, download the usual pivot notebook from Moodle (Python section), upload it to either the Jupyterhub or `colab.research.google.com`, and edit the last cell to use `SimplexNBF` to get to the desired conclusion.)

(b) Are there points (x, y, z, w) in $\mathbb{R}^4$ that obey:

$$\begin{aligned} -2x - 10y + 3z - 20w &\leq 8 \\ -x + y - 3w &\leq -6 \\ x + 4y - z + 8w &\geq -1 \\ x, y, z, w &\geq 0 \end{aligned}$$

(I just flipped the last inequality relative to part (a).). Explain how you know.

NEXT PAGE FOR PROBLEM 4

Problems 4-6 are minimization problems, which you should solve by using the "multiply by -1" trick shown in the Day 10 (Th/F) slides or pre-recorded lecture. In case that process is unclear, Problem 4 includes some guidance on the steps.

4. (COMPUTATION BUT NOT CODING) Find the minimum value of $x + 2y + z$ subject to the constraints

$$\begin{aligned} 3x + 6y + 2z &\geq 12 \\ 2x + y + 3z &\geq 10 \\ z &\leq 2 + x + 2y \\ x, y, z &\geq 0 \end{aligned}$$

as follows:

(a) Reformulate the problem as a Canonical Min problem.

(b) Use the "multiply by -1" trick to write down a Canonical Max problem that is equivalent to this Canonical Min problem, in the sense that the Canonical Max problem is maximized at the same point where the Canonical Min problem is minimized, and the optimal objective function values are related by a sign flip.

(c) Solve the problem in (b), using our standard pivot notebook to do the pivots. Based on that result, say what the solution of the original problem is (what is the minimum value and where does it occur?).

NEXT PAGE FOR PROBLEM 5

5 (COMPUTATION BUT NOT CODING, Strayer Ch. 2, # 9b, with some guidance/rewording): A hotel needs to have clean towels for each day of a three-day period. They can purchase new towels for \$1 per towel, or they have two options for washing dirty towels and then reusing them: they can pay \$0.40 per dirty towel to have it back the next day, or \$0.25 per dirty towel to have it back in two days. If the hotel needs 300, 200, and 400 clean towels on the three days, how can the hotel minimize costs?

If you think about it, there are four variables in this problem:

$$
\begin{aligned}
x &= \text{\# of towels to buy new, available on Day \# 1} \\
y &= \text{\# of dirty towels to send for next-day washing after Day \# 1} \\
z &= \text{\# of dirty towels to send for second-day washing after Day \# 1} \\
w &= \text{\# of dirty towels to send for next-day washing after Day \# 2}
\end{aligned}
$$

To set up the constraints, I found it useful to fill out the following table:

Day #	# clean towels hotel has at start of day	# used during the day	# sent to next-day wash at end of day	# sent to 2nd-day wash at end of day
1	x	300	y	z
2		200	w	0
3		400	0	0

To fill in the missing blanks, think about how many clean towels you have left over from the previous day and how many you get back from the cleaning service(s).

Then write down the constraints, based on the facts that each day (1) you must have at least as many clean towels as you need to use and (2) you can't wash more towels than you used.

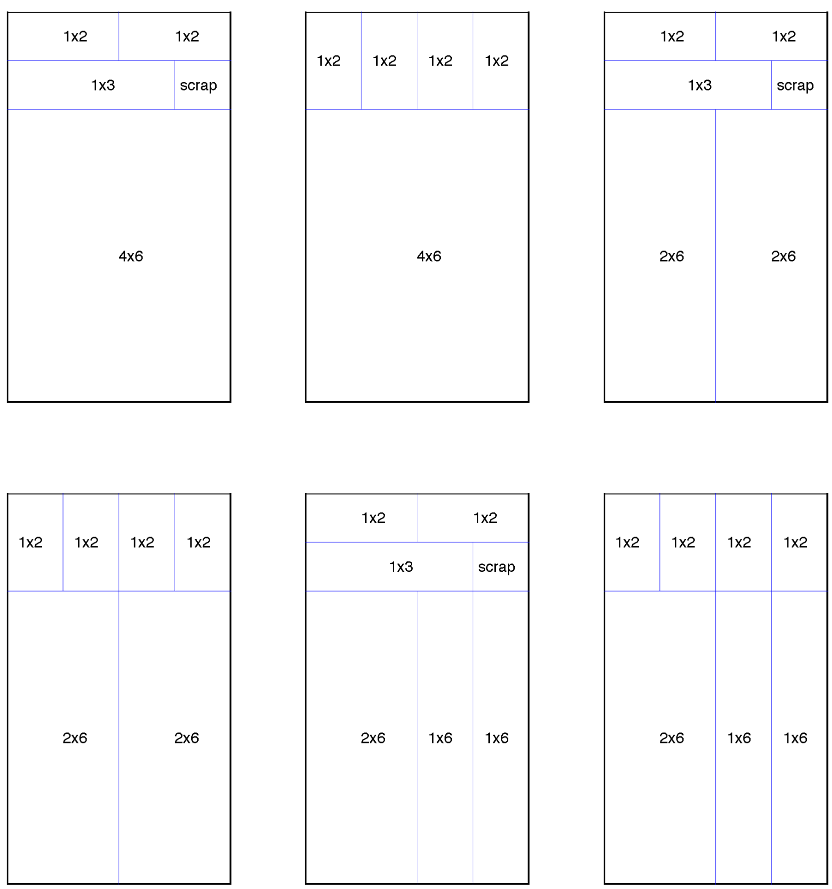
Once you have the problem set up, solve it, using our standard pivot notebook to compute the pivots.

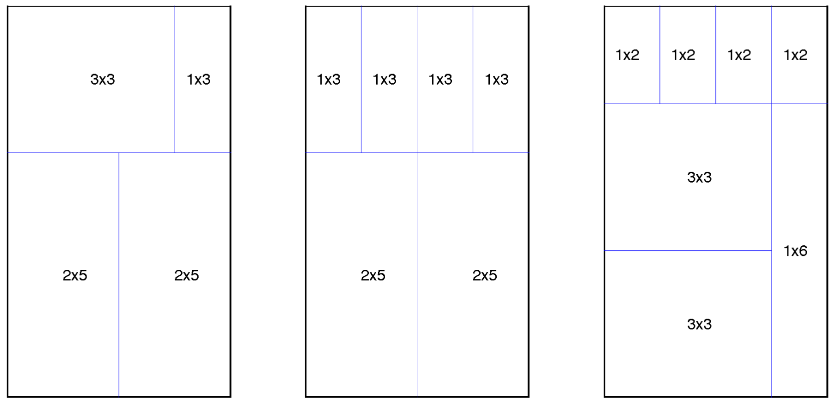
NEXT PAGE FOR PROBLEM 6

6 (COMPUTATION, WITH A NUGGET OF CODING IN PART (B)): You manage a lumberyard that has a large stockpile of plywood sheets that are 4 ft $\times$ 8 ft rectangles. You arrive at work one day and your assistant tells you that you need to fill the following orders for rectangular plywood sheets of various dimensions:

Dimensions (all lengths in ft)	# of orders
4×6	15
2×6	50
2×5	50
3×3	40
1×6	40
1×2	245
1×3	100

Your cutting department has provided you with diagrams of the nine different ways they can cut up a 4×8 sheet into small pieces:





(a) How can you fulfill the orders while using as few 4×8 sheets as possible? [We'll talk about this problem on Monday if you're not sure how to get started.] **Warning: this will take a decent number of pivots, around 10.**

(b) After you solve part (a), your assistant reminds you that you can make at most 15 versions of each of the nine cutting patterns, since each pattern is handled by one particular machine, which can only produce 15 per day. Re-answer the question with these additional constraints.

To do part (b), you will need to incorporate a 9×9 *identity matrix (a square grid of numbers that are all zero, except for ones down the diagonal) into the tableau. Rather than type this out by hand, read through the* `append` *demo on Moodle (Python section) to learn how to use* `np.append` *and* `np.transpose` *to build matrices from smaller pieces. Then use these ideas to build the tableau in (b). The command* `mat = np.identity(9)` *sets* `mat` *equal to a* 9×9 *identity matrix.*