Suggestions for Linear System Solver

Another common feature of professional symbolic algebra systems is the ability to solve systems of equations. As the last required piece of this lab, we'll add some of this functionality to our system, in the form of a solver for systems of linear equations.

You could implement this as a function `solve_linear_system(equations)`, where `equations` is a list of symbolic expressions. This function should return a dictionary mapping variable names (strings) to floating point values representing the values of those variables that are necessary to solve the system.

For the sake of simplicity, we will make some relatively strict assumptions about the form of the equations that are given to the `solve_linear_system` function. In particular, we will expect that each equation is given as a *sum* (or difference) of terms, where each term is either a `Var`, a `Num`, or a `Mul` of a `Num` and a `Var`; and we will assume that each given expression is equal to zero.

For example, to solve the system of equations given by $x + 3y = -5$ and $4x - 3y = 10$, we could do the following to find values for x and y:

```
>>> x, y = Var('x'), Var('y')
>>> solve_linear_system([x + 3*y + 5, 4*x - 3*y - 10])
{'x': 1.0, 'y': -2.0}
```

As we have seen throughout 6.009, the choice of representation for solving a problem can have a big effect on the resulting code, both in terms of how concisely it can be written and in terms of how efficient the code is in the end.

We probably have a good sense of how we can solve systems of linear equations, using something like the *substitution method*. Consider the problem of finding values for x and y that satisfy the two equations:

$$3x + 4y = 33$$

and

$$5x - 2y = 3$$

On paper, you might approach this by solving the second equation for x (in terms of y), getting $x = \frac{2}{5}y + \frac{3}{5}$, and then substituting that into the first equation, getting the following:

$$\frac{6}{5}y + \frac{9}{5} + 4y = 33$$

Then, solving for y yields $y = 6$, and substituting $y = 6$ into the x expression tells us that $x = 3$.

Solving two equations in two unknowns by hand is straightforward, and we could maybe even think about how to replicate that process using our symbolic algebra framework directly. But as we move to more equations in more unknowns, the process gets complicated (both for humans and for our symbolic algebra framework). As such, we're going to suggest using a different approach (and internal representation) for this problem. This does not mean that we should *change* the existing

representation (which works well for the kinds of problems we've seen so far), but that for this part, we might not be making much use of our structures from earlier in the lab.

Again, we could try to write a computer program to perform the substitution method, but this method is complicated and computationally inefficient. By contrast, *Gaussian elimination* is efficient, and it's also more straightforward to implement on a computer.

1) Gaussian Elimination

In this section, we will describe the *Gaussian Elimination* algorithm for solving systems of linear equations. This algorithm works by manipulating systems of linear equations represented as matrix operations.

The example set of linear equations from above can be represented by the following matrix equation:

$$\begin{bmatrix} 3 & 4 \\ 5 & -2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 33 \\ 3 \end{bmatrix}$$

The goal of the algorithm is to solve for the middle vector $\begin{bmatrix} x \\ y \end{bmatrix}$.

Generally, we will say that we are solving the equation $Ax = b$, where A is a square matrix of coefficients, x is a vector of variables (to be solved for), and b is a vector containing the solutions to the linear equations.

1.1) Row-echelon Form

A crucial part of the algorithm is a step called *forward elimination*, which converts the matrix A into *row-echelon* form. If a matrix is in row-echelon form, that means that reading from left to right, each row will start with at least one more zero term than the row above it.

We can do this by applying the following algorithm (described in pseudocode). Note that, below, we use the notation A_{ij} to denote the value in row i and column j in matrix A, and b_i to denote the value in position i in the vector b.

- For each value c between 0 and $w - 1$, inclusive, where w is the number of columns in A:
  - Find the row $s \geq c$ in A whose c^{th} element has the highest magnitude.
    - If A_{sc} is 0, there is no unique solution to this system (your code should retun `None` in this case).
    - Otherwise, swap row s with row c in matrix A, and swap values s and c in vector b.
  - (After the swap described above), for each row $j > c$, substract off a scaled version of row c, scaled by a constant factor x chosen so as to make $A_{jc} = 0$ (and modify the associated value in b accordingly, as well!).

1.2) Back-solving For Variables

Once the matrix A has been converted to row-echelon form, it is possible to back-solve for the values of each of the variables in the vector x.

Starting from the bottom row A, notice that only one variable has a non-zero coefficient associated with it in that equation. Thus, we can use it (and the corresponding value in b) to solve for that variable.

Once we know that variable's value, we can use that, along with the next row from the bottom in A, to solve for the next variable. We can repeat this procedure for each row in A (from each row, we should be able to determine the value of one more variable).

2) Examples

Here, we'll walk through a few examples (which could make good test cases for your code, including for any helper functions you write!).

2.1) Example 1

In this section, we'll work through solving the set of linear equations described above. Our initial setup for that problem was as follows:

$$\begin{bmatrix} 3 & 4 \\ 5 & -2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 33 \\ 3 \end{bmatrix}$$

2.1.1) Forward Elimination

To begin, we look at the first column ($i = 0$). We look for the row with the highest magnitude value in column 0, and swap it with the current row 0 (both in b and in A). In this case, we want the row with the leading 5 to be the top row. So after this swap, we have:

$$\begin{bmatrix} 5 & -2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3 \\ 33 \end{bmatrix}$$

(Note that this did not change the system of equations described at all, just the order in which we will consider them). Now, we want to subtract a scaled version of row 0 from row 1 in order to eliminate the value A_{10}. The right scaling factor for this is $3/5$.

Subtracting $3/5$ times row 0 from row 1, we have:

$$\begin{bmatrix} 5 & -2 \\ 3 - \frac{3}{5}(5) & 4 - \frac{3}{5}(-2) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3 \\ 33 - \frac{3}{5}(3) \end{bmatrix}$$

Simplifying:

$$\begin{bmatrix} 5 & -2 \\ 0 & 26/5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3 \\ 156/5 \end{bmatrix}$$

Now we have our matrix in row-echelon form, so we can move on to solving for the variables.

2.1.2) Back-Solving

We had our system in the following form:

$$\begin{bmatrix} 5 & -2 \\ 0 & 26/5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3 \\ 156/5 \end{bmatrix}$$

Note that the bottom row represents the equation

$$\frac{26}{5}y = \frac{156}{5}$$

Solving, we find that $y = 6$. Then, the next row up represents the equation $5x - 2y = 3$. Now that we know y, we can solve this equation as well:

$$5x - 2(6) = 3$$

$$5x = 15$$

$$x = 3$$

So our final solution is

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \end{bmatrix}$$

2.1.3) Python Form

In our representation for this lab, we could write code like the following to solve this system:

```
>>> x, y = Var('x'), Var('y')
>>> eqns = [
...     4*y + 3*x - 33,
...     5*x - 2*y - 3,
... ]
>>> solve_linear_system(eqns)
{'x': 3.0, 'y': 6.0}
```

2.2) Example 2

Consider solving the following set of linear equations:

$$x + 3y + 7z = 18$$

$$3x + 6y + 9z = 33$$

$$3x + 9y + 15z = 48$$

In matrix form, this system can be represented as:

$$\begin{bmatrix} 1 & 3 & 7 \\ 3 & 6 & 9 \\ 3 & 9 & 15 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 18 \\ 33 \\ 48 \end{bmatrix}$$

2.2.1) Forward Elimination

To begin, we look at the first column ($i = 0$). We look for the row with the highest magnitude value in column 0, and swap it with the current row 0 (both in b and in A). In this case, we want either of the rows with the leading 3's to be the top row, so we'll pick the middle one. After swapping row 0 with row 1, we have:

$$\begin{bmatrix} 3 & 6 & 9 \\ 1 & 3 & 7 \\ 3 & 9 & 15 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 33 \\ 18 \\ 48 \end{bmatrix}$$

(Note that this did not change the system of equations described at all, just the order in which we will consider them). Now, we want to subtract a scaled version of row 0 from the other rows in order to eliminate the values A_{10} and A_{20}.

To do this, we need to subtract $1/3$ times row 0 from row 1, and 1 times row 0 from row 2.

After this subtraction, we have:

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 1 & 4 \\ 0 & 3 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 33 \\ 7 \\ 15 \end{bmatrix}$$

From here on out, row 0 will remain unchanged.

Now we are moving on to the second column ($i = 1$). Again, if there is a row below this one that has a higher magnitude value in column 1, we want to swap those two rows in the matrix. In this case, the bottom row has value 3 in column 1, so we want to swap it with row 1. After swapping rows 1 and 2, we have:

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 3 & 6 \\ 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 33 \\ 15 \\ 7 \end{bmatrix}$$

Now we need to eliminate column 2 from all rows below this one. We can do this by subtracting $1/3$ times row 1 from row 2:

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 3 & 6 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 33 \\ 15 \\ 2 \end{bmatrix}$$

Now we have our matrix in row-echelon form, so we can move on to solving for the variables.

2.2.2) Back-Solving

We had our system in the following form:

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 3 & 6 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 33 \\ 15 \\ 2 \end{bmatrix}$$

Note that the bottom row represents the equation

$$2z = 2$$

Solving, we find that $z = 1$. Then, the next row up represents the equation $3y + 2z = 15$. Now that we know z, we can solve this equation as well:

$$3y + 6(1) = 15$$

$$3y = 9$$

$$y = 3$$

And our final equation is $3z + 6y + 9z = 33$. Knowing values for y znd z, we can solve for x:

$$3x + 6(3) + 9(1) = 33$$

$$3x = 6$$

$$x = 2$$

So our final solution is

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$$

2.2.3) Python Form

In our representation for this lab, we could write code like the following to solve this system:

```
>>> x, y, z = Var('x'), Var('y'), Var('z')
>>> eqns = [
...     x + 3*y + 7*z - 18,
...     3*x + 6*y + 9*z - 33,
...     3*x + 9*y + 15*z - 48,
... ]
>>> solve_linear_system(eqns)
{'x': 2.0, 'y': 3.0, 'z': 1.0}
```