EQUATION-SOLVING WITH A SPECIAL FOCUS ON THE LINEAR CASE: A PRIMER

MATH 196, SECTION 57 (VIPUL NAIK)

Executive summary

Words ...

(1) We need to solve equations both for determining the parameters in a general functional description of a model, and for using existing models to estimate the values of real-world variables. Equations are also useful in solving max/min problems.
(2) When evaluating equation-solving algorithms for suitability, we should consider time, memory, parallelizability, precision, and elegance.
(3) The dimension of the solution space to a system of equations that has no redundancies or inconsistencies is expected to be (number of variables) - (number of equations). If there are fewer equations than variables, we expect that the system is *underdetermined*. If there are more equations than variables, we expect that the system is *overdetermined*. Generally, this means that either some of the equations are redundant (i.e., they do not provide additional information) or the equations are inconsistent.
(4) For diagonal systems of equations, i.e., systems of equations where each equation involves only one variable, we can solve the equations separately, then string together the coordinates of the solution. The process is parallelizable. The solution set satisfies a *rectangular completion property* (note: this isn't a standard term, it's just a shorthand for something that's hard to describe without full symbolism; you might prefer to think of it as a *coordinate interchange property* if that name connects better with the concept for you).
(5) For triangular systems of equations, we need to solve the equations in a particular order, and if we do so, we have to solve a series of equations in one variable. The solution set need not satisfy the rectangular completion property. In case of multiple solutions, we need to make cases and branch each case. Parallelization is possible between the branches but not between the equations themselves.
(6) The same remarks apply for triangular systems of linear equations, except that there is no branching to worry about.
(7) When we manipulate equations (by adding, subtracting, and multiplying) to obtain new equations, we must remember to keep the old equations around to protect against loss of information. However, keeping *all* the old equations around can result in a memory overload. The general idea is to keep enough of the old equations around that the other old equations that we are discarding can be recovered from the new system. In other words, we want our transformations to be *reversible*.
(8) One manipulation technique is to use one of the equations to eliminate one of the variables by expressing it in terms of the other variables. If we are able to do this systematically all the way through, we would have reduced the original system to a triangular system.
(9) Another manipulation technique is to add and subtract multiples of one equation to another equation. We generally keep the equation that is being multiplied before the addition, and discard the equation that is not being multiplied by anything, because that guarantees reversibility. If the value we are multiplying with is a nonzero constant, we can also keep the other equation instead.
(10) We typically add and subtract equations with the explicit goal of eliminating variables or eliminating difficult expressions in terms of the variables.

Actions ...

(1) Please be sure not to throw out information when manipulating the system. When in doubt, store more rather than less.

(2) Please remember that, as soon as you discover an inconsistency, you can abandon the equation-solving attempt (or the branch you are in). All previous solutions that you have found don't amount to anything. In a diagonal system, if any one equation in the system has no solution for its corresponding variable, the system as a whole has no solution. In a triangular system, as soon as we discover an equation that has no solution, discard the branch (from the most recent branching point onward).

1. Equation-solving: why should I care?

In the context of creating and using models, equation-solving occurs in two contexts:

(1) Using input-output pairs (or *observations*) to determine or estimate parameters.
(2) Using outputs from known sets of functions to determine the values of certain inputs (a sort of *triangulation process*).

In application (1), we get a linear equation if and only if the generic functional forms are linear in the parameters (they need not be linear in the inputs). In application (2), we get a linear equation for each functional form that is linear in the inputs.

Apart from model construction and analysis, equation-solving is also key to optimization problems (finding maxima and minima of functions): a first-cut shortlist for the points of local maximum and minimum for a function is the set of critical points, which is obtained by setting all the partial derivatives to equal zero.

Equation-solving is key to going from theoretical models to concrete expressions.

The purpose of this course (and more so of this part of the course) is to become adept at solving linear equations. However, this is a good opportunity to reflect on the general methods of solving equation of all sorts, including equations that are not linear.

2. What should we worry about when (creating procedures for) solving equations?

Before proceeding, it might be worthwhile reviewing the list of things to be watchful for when solving equations and creating procedures for the task of solving equations:

- *Time* is a major constraint. There's a lot more to life than solving equations. The faster, the better.
- *Memory* is another constraint. Manipulating large systems of equations can take a lot of space. Lots of memory usage can also affect time indirectly. Ideally, our equation-solving process should use as little "working memory" as possible.
- *Parallelizability* is great. Parallelizability means that if multiple processors are available, the task can be split across them in a meaningful manner that cuts down on the time required. Some kinds of procedures are parallelizable, or partially so. Others are not parallelizable.
- *Numerical precision* is another issue, particularly in cases where the numbers involved are not rational numbers. The degree of precision with which we measure solutions is important. Lots of major errors occur because of inappropriate truncation. Hypersensitivity to small errors is a bad idea.
- *Elegance and code simplicity*: Another criterion that is arguably important is the complexity of the code that powers the algorithm. Given two algorithms that take about the same amount of time to run, the algorithm that is based on easier, simpler code is preferable, since the code is easier to maintain, tweak, and debug. Elegance is also valuable for aesthetic reasons, which you may or may not care about.

3. Dimension and codimension theory

Brief recap of dimension theory: suppose we start with $\mathbb{R}^n$. We now impose m mutually independent constraints in the form of equations. In other words, we start with all points in $\mathbb{R}^n$, and each constraint whittles down our set of points. The general expectation is that each scalar equation should whittle down the dimension of the set by 1. Thus, the dimension of the subset of $\mathbb{R}^n$ satisfying m independent constraints should be $n - m$.

We can think of the original n (the dimension of the variable space) as the number of variables and the value m (the number of constraints) as the number of equation. Thus:

Dimension of solution space = Number of variables - Number of equations (generically)

In particular:

- If $m < n$, we generically expect that the solution space is positive-dimensional. In other words, we should expect lots of points, possibiy infinitely many of them, that satisfy the system of constraints. We say that such a system is *underdetermined*.

  The key word is *generically*. If the constraints happen to not be independent (either because they are *redundant* or because they are *inconsistent*) a lot of weird things could happen to the solution set. Inconsistency could result in an empty solution set. Redundancy could result in a solution set of higher dimension, because we effectively "wasted" some constraints.
- If $m = n$, we generically expect that the solution space is zero-dimensional. In other words, we should expect to get finitely many solution candidates, or at any rate, a discrete solution set. In the linear situation, the solution will be unique, but for polynomial situations, we could get finitely many solutions without having a unique solution.
- If $m > n$, the solution space is expected to have negative dimension and we say that the system is *overdetermined*. What this essentially means is that it is not really possible for the constraints to be independent. If there is a there there, so to speak, then the system will be consistent and we will still get solutions. Otherwise, the system will not be consistent and we will get an empty solution set.

4. Diagonal systems

Consider a situation with three variables x, y, and z. Suppose we are given the three equations:

$$\begin{aligned} x^2 &= 2 \\ \sin y &= 0 \\ 2z - z^2 &= -10 - z \end{aligned}$$

In principle, this is a system of three equations in three variables. In practice, the system is particularly nice, because each equation "talks about" only one variable. Thus, we can solve each equation as an equation in one variable.

The first equation pins down the value of x to $x = \pm\sqrt{2}$. Applied as a constraint on the entire xyz-space $\mathbb{R}^3$, this brings us down from all of $\mathbb{R}^3$ to the union of the two planes $x = \sqrt{2}$ and $x = -\sqrt{2}$, both parallel to the yz-plane.

The second equation tells us that y is an integer multiple of π. Note that the solution set, though infinite, is discrete and zero-dimensional. (The precise meaning of these concepts is beyond our present scope). Viewed as an equation on $\mathbb{R}^3$, the solution set to this is the union of planes of the form $y = n\pi, n \in \mathbb{Z}$, each parallel to the xz-plane. If we view this together with the first equation, then we have *two constraints* in $\mathbb{R}^3$ and should expect a one-dimensional solution set. Indeed we do get such a solution set: the set of all lines of the form $x = \pm\sqrt{2}, y = n\pi, n \in \mathbb{Z}$.

The third equation is an equation purely in terms of z. If we rearrange and solve, we get $z = 5$ or $z = -2$. The solution set in $\mathbb{R}^3$ for this alone is a union of the two planes $z = 5$ and $z = -2$. The overall solution set of all three equations together is the intersection of all three solution sets, namely, the zero-dimensional (but infinite) set of points of the form:

$$\{(x, y, z) : x = \pm\sqrt{2}, y = n\pi, n \in \mathbb{Z}, z = 5 \text{ or } z = -2\}$$

We call equation systems of this type diagonal systems of equations. Note the following interesting things about diagonal systems of equations:

- The solution procedure for diagonal systems is simply to solve each of the equations individually as an equation in one variable, then string together the solutions in coordinates.

  This is not to say or imply that a diagonal system can always be solved. Diagonal systems can also be difficult to solve. However, the difficulty with diagonal systems does not lie with the entanglement of the variables. Rather, the difficulty, if there is any, must be with solving one of the equations in one variable. There is no additional complexity arising from the presence of multiple variables, since they do not interact at all in the equations.

Further, even if one or more of the individual equations in the diagonal system does not admit an exact solution using known methods, it may be possible to determine approximate solutions using numerical analysis methods. If approximate solutions for each variable can be determined, this gives approximate solutions for the system as a whole.

- The solution set also has a rectangular completion property as follows: if two points are in it, then so are any other points where we swap in and out the coordinates of the point. In the three-dimensional case, if (x_1, y_1, z_1), and (x_2, y_2, z_2) are solutions, so are (x_1, y_2, z_1), (x_2, y_2, z_1) and all the other points we can get in this fashion.

Exercise for the reader: Evaluate the procedure for solving diagonal systems for time, memory, parallelizability, and numerical precision.

Answer:

- *Time*: In a serial computation context, it is basically the sum of the times taken to solve each equation individually.
- *Memory*: If we solve the equations serially, the memory overhead at any point in time is:
  - For the equations already solved: Store the solution set.
  - For the equation being solved: The full working memory needed for its solution procedure.
  - For the equations yet to be solved: Store the equations.
- *Parallelizability*: The system is massively parallelizable. Each equation can be solved independently of the others.
- *Numerical precision*: This is a relative non-issue for the single variable to multivariable transition.
- *Elegance and code simplicity*: Assuming we can invoke modular code to solve the equations in one variable, the code to solve a diagonal system in terms of solving the individual equations is quite simple.

4.1. **Diagonal systems of linear equations.** A diagonal system of linear equations is particularly easy to solve. Unlike the general case, we know that each linear equation in one variable can be one of these three things:

- It has a unique solution
- It is inconsistent, i.e., it simplifies to $0 = 1$.
- It conveys no information, i.e., it simplifies to $0 = 0$.

If, for our linear system, the "inconsistent" case arises for any equation, then there are no solutions to the system. Otherwise, in each coordinate, we either have a unique possibility or the whole real line. Overall, then, our solution set will be an affine subspace (whatever that means) parallel to one of the coordinate subspaces, where some coordinates are completely and uniquely fixed, and other coordinates are completely free to vary.

If a variable does not appear in any equation, it is free to vary over all reals. For completeness, we may throw in a "conveys no information" equation of the form $0x_i = 0$ for each such variable.

5. Non-diagonal systems: general remarks

5.1. **The entanglement of variables.** Conceptually, we can think of diagonal systems as *disentangled* systems of equations: the entanglements between the variables have been removed so that we can concentrate on each variable individually. For general systems of equations, there exist entanglements between the variables. Consider, for instance:

$$\begin{aligned} x^2 - 2xy &= y^2 + z^2 - 3xyz \\ -x^3 - 4y^3 &= -5 \\ xyz &= 1 + (x-1)z^2 \end{aligned}$$

This kind of system is extremely difficult because the variables are entangled with each other. It is not easy to look at this and solve for one variable at a time, because *every* equation is a jumble of the variables.

What we need to do is *rethink* the system in a way that the variables are more clearly separated. The goal is to move to an *equivalent* system that is more separated. We will now proceed to see a number of techniques used for this end. Before that, however, we need to note a few caveats.

5.2. **Protecting against the loss of information.** Let's look at a particular system of equations in two variables that actually came up naturally when trying to find critical points for a function in a multivariable calculus course:

$$\begin{aligned} 4x^3 - 2xy^2 &= 0 \\ -2x^2y + 4y^3 &= 0 \end{aligned}$$

Here is what some people did: multiply the first equation by x, the second equation by y, and subtract. The mixed products cancel under subtraction, and we are left with:

$$4x^4 - 4y^4 = 0$$

This simplifies to giving the union of the line $y = x$ and the line $y = -x$.

Is that the solution set to the original system? Definitely, any solution to the original system must satisfy the new equation $4x^4 - 4y^4 = 0$, and hence, must be on at least one of the two lines. However, the new equation does not convey all the information stored in the original pair of equations. It is a single constraint, whereas the original system had two constraints. Thus, it is possible (and in this case true) that the solution set to the new equation is strictly bigger than the solution set to the old system.

What should we do instead? There are many alternate approaches to solving the question, but this approach can be easily fixed. All we need to do is remember to keep the old system alongside the new equation. Thus, instead of just finding the solutions to the new equation, we test these solutions against the old system. In this case, testing either of the solution lines against either of the equations in the old system gives us a unique solution $x = 0$, $y = 0$.

So, one approach we can use is that even as we try to add, subtract, and manipulate our old system to create a new system, we preserve all the old equations, to avoid any loss of information. This approach, however, is not scalable, because it leads to "equation bloat" as a number of redundant equations are kept on the books for fear of losing something valuable. Usually, it suffices to keep *some* of the old equations on the books, just enough that we do not lose information, but not enough to create unnecessary redundancy and not enough to take too much space.

The key idea is to think of the process as transforming the entire system together, and then ask of a transformation: is it *reversible*? Can we recover the old system from the new one? Certain kinds of transformation rules are reversible by nature. If we stick to those, we can do the manipulations mechanically and not have to explicitly think about loss of information.

Alternative solution approach for the example used. An alternative solution approach is to note that the solutions to the two equations are respective:

$$\begin{aligned} 4x^3 - 2xy^2 = 0 &\iff x = 0 \text{ or } y^2 = 2x^2 \\ -2x^2y + 4y^3 = 0 &\iff y = 0 \text{ or } x^2 = 2y^2 \end{aligned}$$

Let's try all four possibilities:

- $x = 0$ and $y = 0$: This is one solution.
- $x = 0$ and $x^2 = 2y^2$: This solves to $x = 0$, $y = 0$.
- $y^2 = 2x^2$ and $y = 0$: This solves to $x = 0$, $y = 0$.
- $y^2 = 2x^2$ and $x^2 = 2y^2$: Combining, we get $4x^2 = x^2$, so $x = 0$. Plug in and get $y = 0$. Thus, this solves to $x = 0$, $y = 0$.

All four cases end up giving the same point solution $x = 0$, $y = 0$. The fact that all cases end up giving the same solution is a special feature of this system of equations, and is not illustrative of any more general principle.

6. Triangular systems

6.1. **An example and the general idea.** We've already talked about diagonal systems, but there is another kind of system that bears special mention: *triangular system*.

A triangular system of equations is a system where we can arrange the variables so that there is one equation involving only the first variable, another equation involving only the first two, another equation involving only the first three, and so on. For instance, this system is triangular:

$$\begin{aligned} x^2 &= 2 \\ 2x + x^2y &= x^3y^2 \\ xy + xz + yz &= 0 \end{aligned}$$

The first equation involves only x, hence can be solved for x. The second equation involves only x and y. We can plug in solution(s) to the first equation into the second equation, making it an equation in one variable y. Solve this for y. Having got possible values of x and y, we can plug these into the third equation to solve for z.

Note that the existence of multiple solutions at any stage means that we need to make cases for the next stage. In our case, the first equation has two solutions $x = \sqrt{2}$ and $x = -\sqrt{2}$. We have to make two branches for the entire rest of the system based on what solution we choose. Eventually, we will pool together all solutions. Similarly, if in either branch, we get multiple solutions for y, we must branch further. The presence of such branching renders the later coordinates potentially dependent on the earlier coordinates. Thus, the solution set need not have the rectangular completion property observed for diagonal systems.

Solution details for this example. The first equation gives two cases: $x = \sqrt{2}$ and $x = -\sqrt{2}$. Let's consider the subcase $x = \sqrt{2}$. Plugging this into the second equation, we get:

$$2\sqrt{2} + 2y = 2\sqrt{2}y^2$$

Simplify and solve this quadratic equation to get:

$$y = \sqrt{2} \text{ or } y = -1/\sqrt{2}$$

We can further make branches based on the value of y. If we choose $y = \sqrt{2}$, the third equation becomes:

$$2 + 2\sqrt{2}z = 0$$

This simplifies to $z = -1/\sqrt{2}$.

Alternatively, if we choose $y = -1/\sqrt{2}$, we get:

$$-1 + z/\sqrt{2} = 0$$

This simplifies to $z = \sqrt{2}$.

Thus, we get two solutions from the subcase $x = \sqrt{2}$:

- $x = \sqrt{2}, y = \sqrt{2}, z = -1/\sqrt{2}$
- $x = \sqrt{2}, y = -1/\sqrt{2}, z = \sqrt{2}$

Now, consider the case $x = -\sqrt{2}$. In this case, the second equation becomes:

$$-2\sqrt{2} + 2y = -2\sqrt{2}y^2$$

Simplify and solve this quadratic equation to get:

$$y = -\sqrt{2} \text{ or } y = 1/\sqrt{2}$$

If we choose $y = -\sqrt{2}$, we get:

$$2 - 2\sqrt{2}z = 0$$

This simplifies to $z = 1/\sqrt{2}$.
If we choose $y = 1/\sqrt{2}$, we get:

$$-1 - (1/\sqrt{2})z = 0$$

This solves to $z = -\sqrt{2}$.
Thus, we get two solutions from the subcase $x = -\sqrt{2}$:

- $x = -\sqrt{2}, y = -\sqrt{2}, z = 1/\sqrt{2}$
- $x = -\sqrt{2}, y = 1/\sqrt{2}, z = -\sqrt{2}$

Overall, we get four solutions:

- $x = \sqrt{2}, y = \sqrt{2}, z = -1/\sqrt{2}$
- $x = \sqrt{2}, y = -1/\sqrt{2}, z = \sqrt{2}$
- $x = -\sqrt{2}, y = -\sqrt{2}, z = 1/\sqrt{2}$
- $x = -\sqrt{2}, y = 1/\sqrt{2}, z = -\sqrt{2}$

It is clear by looking at this solution set that it does not satisfy the "rectangular completion property." Specifically, we cannot swap values of coordinates between different solutions and still be guaranteed to have a solution.

6.2. **Evaluating the pros and cons of triangular systems.** Both diagonal systems and triangular systems are "essentially solved" types of systems of equations. In both types of systems, we have a strategy to reduce a multivariable system to single variable equations. There are a few ways, however, in which triangular systems are more complicated than diagonal systems. Some of these are stated below:

- In a diagonal system, the individual equations in the separate variables can be solved separately. In other words, the computations can be carried out in *parallel* and the solutions pooled together. Diagonal systems are massively parallelizable.

  For a triangular system, on the other hand, we need to solve the equations *serially* in a very particular order. We first solve the equation with only one variable, then the equation with that and another variable, and so on. Since the process of solving the second equation relies on the solution to the first, the process of solving triangular systems is inherently harder to parallelize.
- In a diagonal system, there is no branching into cases. In a triangular system, on the other hand, multiple solutions to one equation lead to a branching of cases for solving the remaining equations. Note that these branches can be dealt with through parallel processing, but the degree of branching required is generally *a priori* unclear since it depends at each stage on how many solutions each particular equation will turn out to have.

Note that in the case of a triangular system of *linear* equations, the branching issue does not arise, because at each stage, we must get a unique solution. The first issue, namely the issue of non-parallelizability, is still an important problem.

Exercise for the reader: Evaluate the solution procedure for triangular systems for time, memory, parallelizability, and numerical precision.

- *Time*: Depends both on the number of equations and on the number of solutions to each equation. Consider the case where the triangular system has n equations that are of degrees $m_1, m_2, \ldots, m_n$ respectively. The total number of equations we effectively need to solve could be as large as $1 + m_1 + m_1m_2 + \cdots + m_1m_2 \ldots m_{n-1}$.

  We could do a "depth-first" or a "breadth-first" algorithm.
- *Memory*: If proceeding serially, we need to store the solution strings accumulated so far, the working memory for whatever current equation we are working on, and all the equations we are yet to work on. The amount of information that needs to be stored will, like time, depend on the number of solutions to the first few equations.

- *Parallelizability*: We cannot parallelize the system itself. However, we could parallelize the branching process. Every time we encouter multiple solutions, we create multiple new processor nodes and seed them with the information on the solution string so far. If we use this process, the time for every serial strand involves solving n equations. However, the unpredictability in the number of processors needed and the complexity of passing information around are important problems.
- *Numerical precision*: This could be an issue, particularly with polynomials of high degree where precise algebraic expressions for the solutions do not exist. Unlike the diagonal case, the problem is that we need solution values to some equations to plug into other equations. Thus, numerical imprecision for some solutions could lead us to go awry later. Or to put it another way, *errors could compound*.
- *Elegance and code simplicity*: The code becomes complicated because it involves recursion and/or branching. Note that this complexity does not arise in the case of a triangular system of linear equations. Parallelizing the code imposes additional coding challenges.

6.3. **The no solutions case.** For a triangular system, if you obtain an equation that has no solutions, that means that the sub-branch you are working in has no solution. Look for the most recent branching point, and discard all work below that. For instance, consider:

$$\begin{aligned} x^2 &= 3 \\ y^2 &= 2 \\ z &= x + y \\ w^2 &= z \end{aligned}$$

If we choose $x = -\sqrt{3}$, $y = \sqrt{2}$, then $z = \sqrt{2} - \sqrt{3}$, but $w^2 = z$ has no solution. This means that there are no solutions in the branch $x = -\sqrt{3}$, $y = \sqrt{2}$. Similarly, there are no solutions in the branch $x = -\sqrt{3}$, $y = -\sqrt{2}$, so we need to discard those branches.

6.4. **Triangular systems of linear equations.** As noted above, triangular systems of linear equations need to be solved serially, but there are no major branching issues.

Recall the following three cases for a linear equation in one variable:

- It has a unique solution
- It is inconsistent, i.e., it simplifies to $0 = 1$.
- It conveys no information, i.e., it simplifies to $0 = 0$.

As long as our solution process sticks with the first case, we will keep building a unique solution string for the variables solved for so far. If we ever hit the second case, there is no solution, and the process can terminate. If we hit the third case for some variable, then that variable is free to vary over all reals. Subsequent variables will not be determined as unique real numbers, but as linear expressions in that variable, which we can treat as a parameter. What we will get, essentially, is a *family* of solutions. Consider, for instance, the system:

$$\begin{aligned} 2x_1 &= 6 \\ x_1 + x_2 + x_3 &= 7 \\ x_1 - x_2 + x_3 - x_4 &= 10 \end{aligned}$$

There is a missing equation relating x_1 and x_2, so we can fill in such a missing "conveys no information" equation, basically saying that x_2 is free to take any value. We get $x_1 = 3$. The variable x_3 is $7 - x_1 - x_2 = 4 - x_2$. We cannot simplify it further because x_2 is not unique.

We can now solve the fourth equation to write x_4 and we get:

$$x_4 = -3 - 2x_2$$

Thus, our general solution is the line:

$$\{(3, x_2, 4 - x_2, -3 - 2x_2) : x_2 \in \mathbb{R}\}$$

Note that the dimension of this solution set is 1, and it involves a free parameter. This is to be expected because we had four variables but only three equations. Geometrically, it is a line in $\mathbb{R}^4$.

Keep in mind the nature of the solution set you see above. You'll see this sort of description repeatedly, in somewhat different forms, throughout the course. Make sure you have a clear understanding of what it means.

6.5. **A cool fact about linear systems.** One cool thing about systems of linear equations is that every system of linear equations is equivalent to a triangular system. This statement deserves more elaboration, which we shall provide later.

7. Typical manipulation techniques

7.1. **Eliminating a variable through substitution.** In this technique, we solve for one variable in terms of the others from one equation, then replace that variable by the expression in terms of the others into all the other equations. Note that this process:

- reduces the number of variables by one
- reduces the number of equations by one (namely, the equation that we manipulate to solve for one variable in terms of the others).

The simultaneous reduction in the number of variables and the number of equations is no coincidence. Recall that the (expected) dimension of the solution space is given by (number of variables) - (number of equations). The dimension of the solution space should be invariant under manipulation processes. When we eliminate a variable, we also eliminate an equation.

7.1.1. *Example 1 (not solved completely).* Consider the example:

$$\begin{aligned} x^2 - 2xy &= y^2 + z^2 - 3xyz \\ -x^3 - 4y^3 &= -5 \\ xyz &= 1 + (x-1)z^2 \end{aligned}$$

Note that the third equation is linear in y, so we can simplify it to get y in terms of x and z, then eliminate y. Explicitly:

$$y = \frac{1 + (x-1)z^2}{xz}$$

Note that there is a caveat case where the expression does not work. This is the case $x = 0$ and $z = \pm 1$. This case would need to be dealt with separately to see if it is consistent with the other equations. Setting that aside for now, we can plug in the expression for y in the other two equations to get:

$$\begin{aligned} x^2 - 2x\frac{1 + (x-1)z^2}{xz} &= \left(\frac{1 + (x-1)z^2}{xz}\right)^2 + z^2 - 3x\left(\frac{1 + (x-1)z^2}{xz}\right)z \\ -x^3 - 4\left(\frac{1 + (x-1)z^2}{xz}\right)^3 &= -5 \end{aligned}$$

Note that this system is still too complicated.

7.1.2. *Example 2 (solved completely, but only one case can be handled by hand calculation).* In a simpler world, we can keep eliminating variables one by one, then we ultimately get one equation in one variable, that we can solve to get its value. Now, we need to find the other variables. This process now works as a triangular system! We now turn to an example of such a system.

$$\begin{aligned} x + y + z &= 6 \\ x^2 + 2y + 3z &= 16 \\ x^2 + y^2 + z^2 &= 14 \end{aligned}$$

Use the first equation to eliminate z by writing it as $z = 6 - x - y$. Now, the second and third equation become:

$$\begin{aligned} x^2 + 2y + 3(6 - x - y) &= 16 \\ x^2 + y^2 + (6 - x - y)^2 &= 14 \end{aligned}$$

We can use the first of these to eliminate y by getting $y = x^2 - 3x + 2$. Thus, we are left with one equation:

$$x^2 + (x^2 - 3x + 2)^2 + (6 - x - x^2 + 3x - 2)^2 = 14$$

This simplifies to:

$$2(3 + 2x + 5x^2 - 5x^3 + x^4) = 0$$

This is a degree four polynomial equation in one variable. It turns out that it has two real roots and two complex roots. The real roots are $x = 3$ (determinable by inspection) and a more complicated one whose approximate vaue is 2.546 (that requires considerably computation or help from a graphing calculator or computational software). Consider the two cases:

- $x = 3$: We can back-substitute in the expression for y in terms of x and then in the expression for z in terms of y to get $y = 2$ and $z = 1$. Thus, one solution is $x = 3$, $y = 2$, $z = 1$.
- $x \approx 2.546$: This solution (which is difficult to compute by hand) is $x \approx 2.546$, $y \approx 0.844$, $z \approx 2.61$. The values of y and z are obtained by back-substitution.

In fact, properly considered, our system should be viewed as the following triangular system:

$$\begin{aligned} 2(3 + 2x + 5x^2 - 5x^3 + x^4) &= 0 \\ y &= x^2 - 3x + 2 \\ z &= 6 - x - y \end{aligned}$$

Note that this eliminate-and-substitute approach can rarely be done all the way to the point that we get a triangular system. In the linear situation, however, this can always be done.

7.1.3. *A linear example.* We will see a lot of linear examples, so we will be brief here.

$$\begin{aligned} x + y + z &= 6 \\ 2x + 3y + 5z &= 16 \\ x - y + 3z &= -2 \end{aligned}$$

We can use the first equation to obtain:

$$z = 6 - x - y$$

We can substitute in the other two equations:

$$\begin{aligned} 2x + 3y + 30 - 5x - 5y &= 16 \\ x - y + 18 - 3x - 3y &= -2 \end{aligned}$$

The system simplifies to:

$$\begin{aligned} 3x + 2y &= 14 \\ 2x + 4y &= 20 \end{aligned}$$

We can use the first equation to write $y = (14 - 3x)/2$. We can then plug into the third equation and obtain:

$$2x + 2(14 - 3x) = 20$$

This simplifies to $x = 2$. We substitute $y = (14 - 3x)/2 = 4$ and $z = 6 - x - y = 0$. The unique solution is thus $x = 2$, $y = 4$, $z = 0$.

7.2. **Adding and subtracting equations.** Given two equations:

$$\begin{aligned} F(x_1, x_2, \ldots, x_n) &= 0 \\ G(x_1, x_2, \ldots, x_n) &= 0 \end{aligned}$$

Suppose λ is a real number. We can replace this system by the system:

$$\begin{aligned} F(x_1, x_2, \ldots, x_n) + \lambda G(x_1, x_2, \ldots, x_n) &= 0 \\ G(x_1, x_2, \ldots, x_n) &= 0 \end{aligned}$$

Basically, we have added a multiple of the second equation to the first equation. Note that it is *not* enough to just store the equation:

$$F(x_1, x_2, \ldots, x_n) + \lambda G(x_1, x_2, \ldots, x_n) = 0$$

because as a *single* equation, this loses some of the information present in the original system. However, if we retain G alongside, we can recover the original system from the new system. The reason is that starting with the new system, subtracting λ times the second equation from the first recovers the first equation of the old system.

Note that if $\lambda \neq 0$, we could also store $F + \lambda G$ and F together instead of $F + \lambda G$ and G. However, it is better to store G. If we do this, it more readily generalizes to the case where λ is replaced by some function $H(x_1, x_2, \ldots, x_n)$. Explicitly:

$$\begin{aligned} F(x_1, x_2, \ldots, x_n) &= 0 \\ G(x_1, x_2, \ldots, x_n) &= 0 \end{aligned}$$

is equivalent to:

$$\begin{aligned} F(x_1, x_2, \ldots, x_n) + H(x_1, x_2, \ldots, x_n)G(x_1, x_2, \ldots, x_n) &= 0 \\ G(x_1, x_2, \ldots, x_n) &= 0 \end{aligned}$$

for any function $H(x_1, x_2, \ldots, x_n)$.

Suppose we multiply *both* F and G by constants and then add. If the coefficient on F in the sum is always nonzero, we can store the new equation alongside G and throw out F. If the coefficient on G in the sum is always nonzero, we can store the new equation alongside F and throw out G. If both coefficients are always nonzero, we can store the new equation alongside whichever one of F and G we please, and throw out the other.

7.3. **Adding and subtracting with the goal of eliminating.** We can in principle add and subtract equations in any number of ways, but only some of the ways of doing so are *useful*. One laudable goal is to add and subtract in a manner that eliminates a variable or an expression. Consider, for instance, the system:

$$\begin{aligned} x + \cos(y - x^2) &= 0 \\ y + x\cos(y - x^2) &= \pi/2 \end{aligned}$$

In this system, what's bugging us is the presence of $\cos(y - x^2)$, which is a trigonometric function of a polynomial, and is hard to work with. Let's try to eliminate this expression by adding and subtracting the equations. We can subtract x times the first equation from the second equation, but importantly, keep this alongside the first equation. Remember, we keep the equation which *is* being multiplied, so that we can recover the original system. We have:

$$\begin{aligned} x + \cos(y - x^2) &= 0 \\ y - x^2 &= \pi/2 \end{aligned}$$

We can now use the second equation to eliminate y by expressing it in terms of x, and get $y = x^2 + (\pi/2)$. Plugging thus into the first equation, we get:

$$x + \cos(\pi/2) = 0$$

Since $\cos(\pi/2) = 0$, we get $x = 0$, and plugging back gives $y = \pi/2$. Thus, the unique solution is $x = 0$, $y = \pi/2$.

7.4. **Linear systems: elimination is always possible.** Suppose we have a linear system of equations. One of the nice things about such a system is that it is always possible to add and subtract equations to eliminate variables. In fact, we can construct a systematic procedure to solve systems of linear equations by first adding and subtracting them in order to convert to a triangular system, then solving the resultant triangular system. We will see this procedure, called *Gauss-Jordan elimination*, in a subsequent lecture.