CS 70 Discrete Mathematics and Probability Theory
Spring 2020

# HW 2

Due: Fri 2/7 at 11:59 PM
Grace period until Sun 2/9 11:59 PM

## 1 Gaussian Elimination

Gaussian Elimination for solving systems of linear equations is an algorithm that should be very familiar to you. We will have need of it later in the course, and so it is nice for you to actually prove some basic properties that you already know that Gaussian Elimination satisfies.

We will prove the termination of Gaussian Elimination Algorithm for input augmented matrix $A \in \mathbb{R}^{n\times(n+1)}$. Let $A[i][j]$ denote the entry of $A$ in the $i$th row and $j$th column. Let $A[i]$ denote the $i$th row of $A$, where $i$ can range from 1 to $n$.

The first equation of the system of linear equations is:

$$A[1][1]\cdot x_1 + A[1][2]\cdot x_2 + \ldots + A[1][n]\cdot x_n = A[1][n+1]$$

In general, for $1 \le i \le n$, the $i$th equation of the system is:

$$A[i][1]\cdot x_1 + A[i][2]\cdot x_2 + \ldots + A[i][n]\cdot x_n = A[i][n+1]$$

Here is the algorithm:

Initialize $r = 1$, $c = 1$.

While $c \le n$ and $r \le n$, repeat the following:

1. Pick a row $A[i]$ such that $i \ge r$ and $A[i][c] \ne 0$, and swap it with row $A[r]$. If no such $A[i]$ exists, increment $c$ by 1 and skip the following three steps.
2. Scale row $A[r]$ by multiplying it by $\frac{1}{A[r][c]}$, so $A[r][c]$ becomes a 1.
3. For each value of $r < t \le n$, scale the updated row $A[r]$ by $-A[t][c]$ and add it to row $A[t]$ such that $A[t][c] = 0$. In other words $A[t] = A[t] - A[t][c]A[r]$.
4. Increment $c$ by 1 and increment $r$ by 1

At this point, the matrix is in an upper-triangular form (you will have to prove this later in this problem). There are many things that could happen.

If $r < n+1$, then we terminated by running out of rows to work on. In this case, it turns out you can figure out if there are no valid solutions or infinitely many solutions.

If $r = c = n+1$, then we continue onward to the backward pass of this algorithm, known as backsubstitution:

Initialize $r = n$.
Directly inspect the matrix to identify the value of $x_n$ as $A[n][n+1]$.
While $r > 1$:

1. Use the values $x_r \ldots x_n$ and plug them into the linear equation corresponding to $A[r-1]\vec{x} = A[r-1][n+1]$.
2. Solve the resulting linear equation for $x_{r-1}$
3. Decrement $r$ by 1.

This gives us the unique solution if one exists.

(a) Prove using induction that the forward pass of this algorithm always terminates for augmented matrix $A \in \mathbb{R}^{n\times(n+1)}$.

(b) Prove using induction that the backward pass of the algorithm terminates if reached.

(c) Prove using induction that the downward pass of the algorithm terminates with an upper-triangular matrix — all the entries below the diagonal are zero.

(d) Prove that the algorithm is correct for augmented matrix $A \in \mathbb{R}^{n\times(n+1)}$ for the case when a unique solution exists. To do this, you should first prove a lemma that shows that any solution to the original system of equations remains a solution to the modified system of equations at all iterations of the downward pass of the algorithm, and vice-versa: all solutions of the modified system of equations at all iterations are valid solutions to the original system of equations.

## 2 Strong Induction

Use strong induction to show that for all natural numbers $n$ there exist natural numbers $x$, $y$, and $z$ such that $6^n = x^2 + y^2 + z^2$. *Hint: You may want to write out a decomposition for n=1, 2, 3*

## 3 Induction on Reals

Induction is always done over objects like natural numbers, but in some cases we can leverage induction to prove things about real numbers (with the appropriate mapping). We will attempt to prove the following by leveraging induction and finding an appropriate mapping.

Bob the Bug is on a window extending to the ground, trying to escape Sally the Spider. Sally has built her web from the ground to 2 inches up the window. Every second, Bob jumps 1 inch vertically up the window, then loses grip and falls to half his vertical height.

Prove that no matter how high Bob starts up the window, he will always fall into Sally's net in a finite number of seconds.

## 4 Nothing Can Be Better Than Something

In the stable matching problem, suppose that some jobs and candidates have hard requirements and might not be able to just settle for anything. In other words, in addition to the preference orderings they have, they prefer being unmatched to being matched with some of the lower-ranked entities (in their own preference list). We will use the term entity to refer to a candidate/job. A matching could ultimately have to be partial, i.e., some entities would and should remain unmatched.

Consequently, the notion of stability here should be adjusted a little bit to capture the autonomy of both jobs to unilaterally fire employees and employees to just walk away. A matching is stable if

- there is no matched entity who prefers being unmatched over being with their current partner;
- there is no matched/filled job and unmatched candidate that would both prefer to be matched with each other over their current status;
- similarly, there is no unmatched job and matched candidate that would both prefer to be matched with each other over their current status;
- there is no matched job and matched candidate that would both prefer to be matched with each other over their current partners; and
- there is no unmatched job and unmatched candidate that would both prefer to be with each other over being unmatched.

(a) Prove that a stable pairing still exists in the case where we allow unmatched entities.

*(HINT: You can approach this by introducing imaginary/virtual entities that jobs/candidates "match" if they are unmatched. How should you adjust the preference lists of jobs/candidates, including those of the newly introduced imaginary ones for this to work?)*

(b) As you saw in the lecture, we may have different stable matchings. But interestingly, if an entity remains unmatched in one stable matching, it/she must remain unmatched in any other stable matching as well. Prove this fact by contradiction.

## 5 Inductive Charging Revisited

Recall the setup from the previous HW. There are $n$ cars on a one-way circular track. Among all of them, they have exactly enough fuel (in total) for one car to exactly circle the track. Each car burns exactly the same amount of fuel per unit of distance. Two cars at the same location may transfer fuel between them.

(a) Prove that there exists one car that can circle the track, by gathering fuel from other cars along the way. (That is, one car moving and all others stopped). Hint: Use induction and the previous part of this problem from HW1.

*(HINT: The problem as written is about these isolated points called "cars" that have fuel associated with them. These exist at the beginning of your thinking, however they're not what you have when you have already made some progress. It can be useful to think about generalizing points to segments that have a starting and end point associated with them, together with some remaining fuel. Inside each segment, there can be a story of how you got here.)*

## 6 Stable Matching for Classes!

Let's consider the system for getting into classes. For simplicity, we will start with the problem assigning students to lab sections first, since it is clear that there are a finite number of seats. We are given $n$ students and $m$ lab sections. Each lab section $\ell$ has some number, $q_\ell$ of seats, and we assume that the total number of students is larger than the total number of seats (i.e. $\sum_{\ell=1}^{m} q_\ell < n$) and so some students are going to end up with no lab. Each student ranks the $m$ lab sections in order of preference, and the instructor for each lab ranks the $n$ students. Our goal is to find an assignment of students to seats (one student per seat) that is *stable* in the following sense:

- There is no student-lab pair $(s,\ell)$ such that $s$ prefers $\ell$ to her allocated lab section and the instructor for $\ell$ prefers $s$ to one of the students assigned to $\ell$. (This is like the stability criterion you have seen for jobs: it says there is no student-lab pair that would induce that lab instructor to kick out an existing student and take this new student instead.)
- There is no lab section $\ell$ for which the instructor prefers some unassigned student $s$ to one of the students assigned to $\ell$. (This extends the stability criterion to take account of the fact that some students are not assigned to labs.)

Note that this problem is almost the same as the Stable Matching problem for jobs/internships presented in the lecture note, with two differences: (i) there are more students than seats; and (ii) each lab section can have more than one seat.

Perhaps you will agree that this extended model is more realistic, even for the jobs context!

(a) Explain how to modify the propose-and-reject algorithm so that it finds a stable assignment of students to seats. [*Hint*: What roles of students/instructors will be in the propose-and-reject algorithm? What does "candidates have a job offer in hand (on a string)" mean in this setting?]

(b) State a version of the Improvement Lemma (see the Stable Matchings Lecture Note) that applies to your algorithm, and prove that it holds.

(c) Use your Improvement Lemma to give a proof that your algorithm terminates, that every seat is filled, and that the assignment your algorithm returns is stable.

(d) Let us consider the potential of students to want to swap lab sections, subject to global stability (i.e. the swap can't make the matching as a whole unstable). Either prove that your algorithm will not have any such swap requests or modify it to have no such swap requests and prove that the modified one will not have any students wanting to stably-swap sections.

## 7 Grid Induction

Pacman is walking on an infinite 2D grid. He starts at some location $(i, j) \in \mathbb{N}^2$ in the first quadrant, and is constrained to stay in the first quadrant (say, by walls along the x and y axes). Every second he does one of the following (if possible):

(i) Walk one step down, to $(i, j-1)$.

(ii) Walk one step left, to $(i-1, j)$.

For example, if he is at $(5,0)$, his only option is to walk left to $(4,0)$; if Pacman is instead at $(3,2)$, he could walk either to $(2,2)$ or $(3,1)$.

Prove by induction that no matter how he walks, he will always reach $(0,0)$ in finite time. (*Hint*: Try starting Pacman at a few small points like $(2,1)$ and looking all the different paths he could take to reach $(0,0)$. Do you notice a pattern?)

## 8 Well-Ordered Grid

Consider an infinite sheet of graph paper such that each square contains a natural number. Suppose that the number in each square is equal to the average of the numbers in the four neighboring squares.

(a) By the Well-Ordering Principle, there must be some smallest number in the grid (call it $n$). Prove that for any square containing $n$, the four squares adjacent to it must also contain $n$.

(b) Prove that each square in the infinite grid contains the same number.

## 9 Losing Marbles

Two EECS70 GSIs are playing a game, where there is an urn that contains some number of red marbles (R), green marbles (G), and blue marbles (B). There is also an infinite supply of marbles outside the urn. When it is a player's turn, the player may either:

(i) Remove one red marble from the urn, and add 3 green marbles.

(ii) Remove two green marbles from the urn, and add 7 blue marbles.

(iii) Remove one blue marble from the urn.

These are the only legal moves. The last player that can make a legal move wins. We play optimally, of course – meaning we always play one of the best possible legal moves.

(a) Prove by induction that, if the urn initially contains a finite number of marbles at the start of the game, then the game will end after a finite number of moves.

(b) If the urn contains 2 green marbles and $B$ blue marbles initially, then who will win the game? Prove it. In this case, does it matter what strategy the players use?

(c) If the urn contains $(R, G, B)$ red, green, and blue marbles initially, then who will win the game? Prove it. In this case, does it matter what strategy the players use?

## 10 Make Your Own Question

Make your own question on this week's material and solve it.

## 11 Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student! We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

1. **What sources (if any) did you use as you worked through the homework?**
2. **If you worked with someone on this homework, who did you work with?** List names and student ID's. (In case of homework party, you can also just describe the group.)
3. **How did you work on this homework?** (For example, *I first worked by myself for 2 hours, but got stuck on problem 3, so I went to office hours. Then I went to homework party for a few hours, where I finished the homework.*)
4. **Roughly how many total hours did you work on this homework?**