Mathematical Techniques for Computer Science Revision Notes

James Brown

April 29, 2017

Contents

1 Introduction

These are notes I have written in preparation of the 2017 Mathematical Techniques for Computer Science exam. This year the module was run by Achim Jung (A.Jung@cs.bham.ac.uk).

2 Systems of Linear Equations

A linear equation which contains more than one variable will usually have many solutions. In order to pin this down to exactly one solution, so that all equations are satisfied simultaneously, we need as many equations are there are unknowns. Two unknowns, two equations required and so on.

In these systems of equations, we don't really care about the variable names. We introduce a new notation which is much more consise by removing variable names:

$$\left(\begin{array}{ccc|c} 1 & 5 & -2 & -11 \\ 3 & -2 & 7 & 5 \\ -2 & -1 & -1 & 0 \end{array}\right)$$

Figure 1: Consise notation for systems of linear equations

2.1 Gaussian Elimination

In Computer Science terms, this is best described as a recursive algorithm with a **base case** and a **general case**.

- **Base case:** There is only one equation and one unknown in the form $ax = b$. We can directly solve this: $x = b/a$.
- **General case:** There are n equations and each equation contains n unknowns. We use the first equation to *eliminate* the first unknown in the remaining $n - 1$ equations. We then recursively apply Gaussian elimination to the remaining $n - 1$ equations which each have $n - 1$ unknowns.

This method builds a staircase of zeroes beneath our variables, which is often called **echelon form**.

$$\left(\begin{array}{ccc|c} x_1 & * & * & * \\ 0 & x_2 & * & * \\ 0 & 0 & x_3 & * \end{array}\right)$$

2.2 Special Cases for Gaussian Elimination

2.2.1 Contradictory Equations

In the base case $ax = b$, it can happen that $a = 0$, so the equation is really $0 = b$. Furthermore, if it is the case that $b \neq 0$ then the equation is contradictory and cannot be solved. In this case, the algorithm should exit and indicate that a solution can not be found. This may be by raising an exception for example. An example of equations that may lead to this is as follows:

$$x_1 - 2x_2 = 3$$
$$2x_1 - 4x_2 = 7$$

2.2.2 Irrelevant Equations

This is like the previous special case, but both sides of the equation are zero (in $ax = b$, both a and b are zero). This means the equation does not constrain the value of x in any way. Due to this, the value of the unknown can be chosen freely. Take the following equations:

$$\begin{aligned} x_1 - 2x_2 &= 3 \\ 2x_1 - 4x_2 &= 6 \end{aligned}$$

Here the second equation is redundant so we are only left with one equation for two unknowns. This gives the following assignments:

$$\begin{aligned} x_1 &= 3 + x_2 \\ x_2 &: \text{ chosen freely} \end{aligned}$$

2.2.3 First Equation Unusable for Eliminating the First Unknown

If the coefficient of the first unknown in the first equation is zero, not matter how much we add it to another equation the first variable of the other will not be affected. In order to get around this, we simply exchange the first equation with another one for which the first equation is not zero and run the algorithm on this rearranged setup.

2.2.4 Can't Use Any of the Given Equations to Eliminate the First Unknown

This is similar to the case where we have an irrelevant equation. It means that the first variable is not constrained at all by the given system. The algorithm to solve this should report back that the first variable can be chosen freely.

2.3 General Gaussian Elimination

We may encounter a situation where we have two equations for one unknown, which forces us to consider more general systems for linear equations where the number of unknowns does not neccesarily match the number of equations. Luckily, the algorithm does not need much adjustment, all we need to do is readjust the base case.

- **Base Case 1:** Only one unknown is left over but there may be more than one equation. We solve each equation independently and compare the answers. If they agree, that is what we return. If they disagree, then the system has no solution.
- **Base Case 2:** Only one equation is left over but more than one unknown appears in it (let's say m many). In this case, $m - 1$ unknowns can be chosen freely and the other is computed from the equation. It does not matter which of the $m - 1$ unknowns is chosen and which is computed. For example, if we are left with $x - 2y + z = 3$, we set $x = 3 + 2y - z$ and say y and z can be freely chosen.

2.3.1 Extended Gaussian Elimination

Once we have reached **echelon form** and checked that the system is not contradictory, we can elminate entries above the pivots (the staircase values). This helps us to write the general solution, as we can easily read of the values.

3 Fields

There are many **number systems** which are of interest to a computer scientist. The mathematical term for a number system is a **field**. The requirement is that the elements of a field can be added and multiplied. There also must be a 'zero' and a 'one' which must satisfy

$$\begin{aligned} x + 0 &= x \\ x * 1 &= x \end{aligned}$$

All the usual rules of arithmetic are also valid:

$$\begin{gathered}
x + y = y + x \\
x + (y + z) = (x + y) + z \\
x * y = y * x \\
x * (y * z) = (x * y) * z \\
x * (y + z) = x * y + x * z
\end{gathered}$$

We also require that the following equations can always be solved:

$$\begin{aligned}
&a + x = 0 \\
&a * x = 1 \text{ assuming } a \neq 0
\end{aligned}$$

From these, we can derive many other rules of arithmetic. For example, we can show $x * 0 = 0$ is true in any field.

3.1 Galois Field

One particularly useful finite field is GF(2), which has just two elements: zero and one. No field can have fewer than this. We define addition and multiplication in the following ways:

+	0	1
0	0	1
1	1	0

Figure 2: Addition in GF(2)

*	0	1
0	0	0
1	0	1

Figure 3: Multiplication in GF(2)

Gaussian elimination works exactly the same over GF(2) as it did with ordinary numbers. This fact is exploited extensively in coding theory and cryptography.

4 Analytic Geometry

4.1 In the Plane

Given two points with coordinates $\begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$ and $\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$, the distance d between them is computed with $d = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$. This works for both negative and positive coordinates.

4.1.1 Vectors

We can also interpret a pair of numbers $\begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ as a movement in the plane v_1 units parallel to the x-axis and v_2 units parallel to the y-axis. Lower case letters with arrows above are used to notate vectors: $\overrightarrow{v}$, $\overrightarrow{w}$, $\overrightarrow{u}$ etc. A vector also has a length, once again computed by the Pythagorean theorem $|\overrightarrow{v}| = \sqrt{v_1^2 + v_2^2}$. This is the distance a point travels under the movement described by $\overrightarrow{v}$. A vector of length 1 is called a **unit vector**.

Index