Math 1600 Lecture 4, Section 2, 12 Sept 2014

Announcements:

Read Section 1.3 for next class. Work through recommended homework questions.

Tutorials start **next week**, and include a **quiz** covering Sections 1.1, 1.2 and the code vectors part of 1.4. It does not cover the Exploration after Section 1.2.
Questions are *similar* to homework questions, but may be slightly different.
There will be **two true/false questions**, for which you must explain your answers.
The quizzes last 20 minutes, and are at the end of the tutorial, so you have time for questions at the beginning. You must write in the tutorial you are **registered** in. Different sections have different quizzes, but it is still considered an academic offense to share information about quizzes.
No calculators or other aids are permitted on quizzes or exams.

Today is the last day you can **switch tutorial sections**. This must be done via paper add/drop in MC104, 9:30-3:30. No line ups! Updated counts:

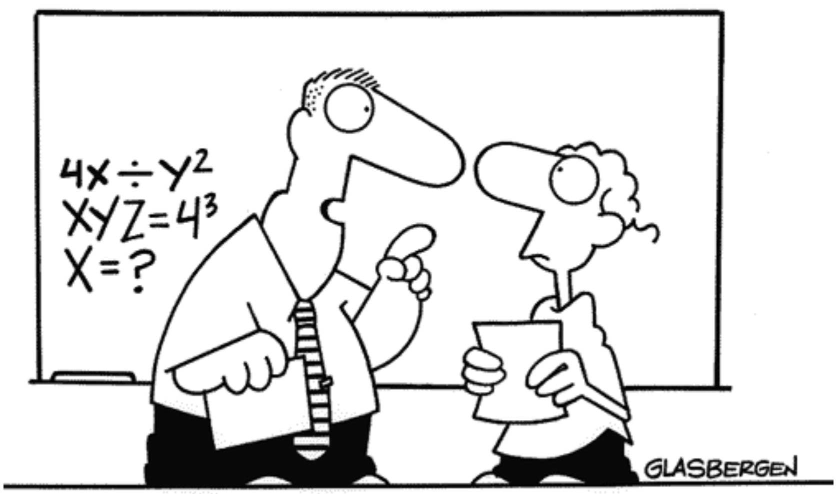
```
003  W   9:30AM   KB-K103     40 !
009  W   9:30AM   UCC-65      17 *
008  W  11:30AM   UCC-60      40 !
006  W   3:30PM   UC-202      35
005  Th 11:30AM   SSC-3010    39 !
007  Th 12:30PM   MC-17       35
004  Th  2:30PM   UC-202      34
```

Office hours: Monday, 3:00-3:30 and Wednesday, 11:30-noon, MC103B.

Help Centers: Monday-Friday 2:30-6:30 in MC 106 starting Wednesday, September 17.

Lecture notes (this page) available from course web page.

A copy of the Solutions Manual has been put on reserve in Taylor Library.



"Algebra class will be important to you later in life because there's going to be a test six weeks from now."

The midterm is in 6 weeks!

Partial review of last lecture:

Section 1.2: Length and Angle: The Dot Product

Definition: The **dot product** or **scalar product** of vectors $\vec{u}$ and $\vec{v}$ in $\mathbb{R}^n$ is the real number defined by

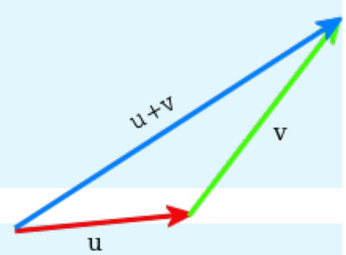
$$\vec{u} \cdot \vec{v} := u_1 v_1 + \cdots + u_n v_n .$$

This has familiar properties; see Theorem 1.2.

Definition: The **length** or **norm** of $\vec{v}$ is the scalar $\|\vec{v}\|$ defined by

$$\|\vec{v}\| := \sqrt{\vec{v} \cdot \vec{v}} = \sqrt{v_1^2 + \cdots + v_n^2} .$$

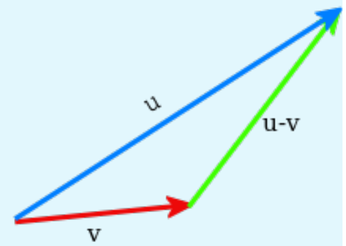
A vector of length 1 is called a **unit** vector.



Theorem 1.5: The Triangle Inequality: For all $\vec{u}$ and $\vec{v}$ in $\mathbb{R}^n$,

$$\|\vec{u}+\vec{v}\| \leq \|\vec{u}\| + \|\vec{v}\|.$$

We define the **distance** between vectors $\vec{u}$ and $\vec{v}$ by the formula



$$d(\vec{u},\vec{v}) := \|\vec{u}-\vec{v}\| = \sqrt{(u_1-v_1)^2+\cdots+(u_n-v_n)^2}.$$

Angles from dot product

Theorem 1.4: The Cauchy-Schwarz Inequality: For all $\vec{u}$ and $\vec{v}$ in $\mathbb{R}^n$,

$$|\vec{u}\cdot\vec{v}| \leq \|\vec{u}\| \, \|\vec{v}\|.$$

We can therefore use the dot product to *define* the **angle** between two vectors $\vec{u}$ and $\vec{v}$ in $\mathbb{R}^n$ by the formula

$$\cos\theta = \frac{\vec{u}\cdot\vec{v}}{\|\vec{u}\| \, \|\vec{v}\|}, \quad \text{i.e.,} \quad \theta := \arccos\left(\frac{\vec{u}\cdot\vec{v}}{\|\vec{u}\| \, \|\vec{v}\|}\right),$$

where we choose $0 \leq \theta \leq 180^\circ$. This makes sense because the fraction is between -1 and 1.

To help remember the formula for $\cos\theta$, note that the denominator normalizes the two vectors to be unit vectors. The formula can also be written

$$\vec{u}\cdot\vec{v} = \|\vec{u}\| \, \|\vec{v}\| \, \cos\theta.$$

New material

On board: Angle between $\vec{u} = [1,2,1,1,1]$ and $\vec{v} = [0,3,0,0,0]$.

For a random example, you'll need a calculator, but for hand calculations you can remember these cosines:

$$\cos 0^\circ = \frac{\sqrt{4}}{2} = 1, \qquad \cos 30^\circ = \frac{\sqrt{3}}{2}, \qquad \cos 45^\circ = \frac{\sqrt{2}}{2} = \frac{1}{\sqrt{2}},$$

$$\cos 60^\circ = \frac{\sqrt{1}}{2} = \frac{1}{2}, \qquad \cos 90^\circ = \frac{\sqrt{0}}{2} = 0,$$

using the usual triangles.

Orthogonal Vectors

How can we tell whether two vectors are orthogonal / perpendicular?
Easy: $\theta = 90^\circ$ is the only angle for which $\cos\theta = 0$.
So $\vec{u}$ and $\vec{v}$ are **orthogonal** if and only if $\vec{u} \cdot \vec{v} = 0$.

Example: If $\vec{u} = [1, 2, 3]$ and $\vec{v} = [1, 1, -1]$ in $\mathbb{R}^3$, then $\vec{u} \cdot \vec{v} = 1 \cdot 1 + 2 \cdot 1 + 3 \cdot (-1) = 1 + 2 - 3 = 0$, so $\vec{u}$ and $\vec{v}$ are orthogonal.

Also, $\vec{u} = [1, 2, 3]$ and $\vec{v} = [1, 1, 1]$ in $\mathbb{Z}_3^3$ are orthogonal, since $\vec{u} \cdot \vec{v} = 1 + 2 + 3 = 6 = 0 \pmod 3$.

An applet illustrating the dot product. (Another one: javascript, and java.)

Pythagorean theorem in $\mathbb{R}^n$: If $\vec{u}$ and $\vec{v}$ are orthogonal, then

$$\|\vec{u} + \vec{v}\|^2 = \|\vec{u}\|^2 + \|\vec{v}\|^2.$$

Explain on board, using Theorem 1.2.

Projections

Use board to derive formula for **the projection of $\vec{v}$ onto $\vec{u}$**:

$$\mathrm{proj}_{\vec{u}}(\vec{v}) = \left(\frac{\vec{u} \cdot \vec{v}}{\vec{u} \cdot \vec{u}}\right) \vec{u}.$$

Here $\vec{u}$ must not be $\vec{0}$, but $\vec{v}$ can be any vector. To help remember the formula, note that the denominator ensures that the answer does not depend on the length of $\vec{u}$.

This applet is useful for understanding projections as well. Java version.

Example: If $\vec{u} = [-1, 1, 0]$ and $\vec{v} = [1, 2, 3]$ then

$$\begin{aligned}\mathrm{proj}_{\vec{u}}(\vec{v}) = \frac{\vec{u}\cdot\vec{v}}{\vec{u}\cdot\vec{u}}\,\vec{u} &= \frac{-1+2+0}{1+1+0}\,[-1,1,0] \\ &= \frac{1}{2}\,[-1,1,0] = [-\frac{1}{2}\,,\frac{1}{2}\,,0]\end{aligned}$$

Questions

True/false: If $\vec{u}$, $\vec{v}$ and $\vec{w}$ are vectors in $\mathbb{R}^n$ such that $\vec{u}\cdot\vec{v} = \vec{u}\cdot\vec{w}$ and $\vec{u} \neq \vec{0}$, then $\vec{v} = \vec{w}$.

False. For example, if $\vec{u} = [1, 0]$, $\vec{v} = [0, 1]$ and $\vec{w} = [0, 2]$, then $\vec{u}\cdot\vec{v} = 0$ and $\vec{u}\cdot\vec{w} = 0$ but $\vec{v} \neq \vec{w}$.

True/false: If $\vec{u}$ is orthogonal to both $\vec{v}$ and $\vec{w}$, then $\vec{u}$ is orthogonal to $2\vec{v} + 3\vec{w}$.

True, because $\vec{u}\cdot(2\vec{v}+3\vec{w}) = 2\,\vec{u}\cdot\vec{v} + 3\,\vec{u}\cdot\vec{w} = 2(0) + 3(0) = 0\,$.

You only answer **true** if a statement is *always* true. You justify this answer by giving a general explanation of why it is always true, not just an example where it happens to be true.

You answer **false** if a statement can in *some case* be false. You justify this answer by giving an explicit example where the statement is false.

Question: Suppose I tell you that $\vec{u}\cdot\vec{v} = 1/2$ and $\vec{u}\cdot\vec{w} = -1$. What is $\vec{u}\cdot(2\vec{v}+3\vec{w})$?

Solution: $\vec{u}\cdot(2\vec{v}+3\vec{w}) = 2\,\vec{u}\cdot\vec{v} + 3\,\vec{u}\cdot\vec{w} = 2(1/2) + 3(-1) = -2\,$.

Question: Does $\mathrm{proj}_{\vec{u}}(\vec{v})$ always point in the same direction as $\vec{u}$?

Solution: No. It is always parallel, but might point in the opposite direction. For example, if $\vec{u} = [1, 0]$ and $\vec{v} = [-1, 1]$ then $\mathrm{proj}_{\vec{u}}(\vec{v}) = [-1, 0] = -\vec{u}$.

Section 1.4: Applications: Code Vectors (we aren't covering force vectors)

We're going to study a way to encode data that allows us to detect transmission errors. Used on CDs, UPC codes, ISBN numbers, credit card numbers, etc.

Example 1.37: Suppose we want to send the four commands "forward", "back", "left" and "right" as a sequence of 0s and 1s. We could use the following code:

$$\text{forward} = [0,0], \quad \text{back} = [0,1], \quad \text{left} = [1,0], \quad \text{right} = [1,1].$$

But if there is an error in our transmission, the Mars rover will get the wrong message and will drive off of a cliff, wasting billions of dollars of taxpayer money (but making for some good NASA jokes).

Here's a more clever code:

$$\text{forward} = [0,0,0], \quad \text{back} = [0,1,1], \quad \text{left} = [1,0,1], \quad \text{right} = [1,1,0].$$

If any single *bit* (binary digit, a 0 or a 1) is flipped during transmission, the Mars rover will notice the error, since all of the **code vectors** have an **even** number of 1s. It could then ask for retransmission of the command.

This is called an **error-detecting code**. Note that it is formed by adding a bit to the end of each of the original code vectors so that the total number of 1s is even.

In vector notation, we replace a vector $\vec{b} = [v_1, v_2, \ldots, v_n]$ with the vector $\vec{v} = [v_1, v_2, \ldots, v_n, d]$ such that $\vec{1} \cdot \vec{v} = 0 \pmod 2$, where $\vec{1} = [1, 1, \ldots, 1]$.

Exactly the same idea works for vectors in $\mathbb{Z}_3^n$; see Example 1.39 in the text.

Note: One problem with the above scheme is that **transposition** errors are not detected: if we want to send $[0,1,1]$ but the first two bits are exchanged, the rover receives $[1,0,1]$, which is also a valid command. We'll see codes that can detect transpositions.