EECS 70 Discrete Mathematics and Probability Theory
Spring 2013 Anant Sahai Note 0

A perspective on EECS as a whole and 70's role within it

In this introductory mini-lecture, we will try to convey the big picture at a "cultural" level. 70 is a basic lower-division course. It is one of the entry-points[1] into the EECS department, along with its siblings 20, 40, 61A and 61B. The only prerequisites are "sophomore mathematical maturity" — which is University code for saying you really should understand the material in Math 1A and 1B as well as be comfortable doing calculations — and some basic programming experience so you can code up simple things if and when that's needed.

As you might've heard, this is a pretty mathematical course. The official title is "Discrete Mathematics and Probability Theory" and the course catalog description describes the content as:

> Logic, infinity, and induction; applications include undecidability and stable marriage problem. Modular arithmetic and GCDs; applications include primality testing[2] and cryptography. Polynomials; examples include error correcting codes and interpolation. Probability including sample spaces, independence, random variables, law of large numbers; examples include load balancing, existence arguments, Bayesian inference.

That probably looks like a grab bag of esoteric math stuff. You might be wondering, why is there such a mathematical course sitting right at the threshold of the majors (or minors) offered by the EECS department? After all, isn't EECS a collection of practical fields? Isn't it about making computers, designing cellphones and wireless networks, programming applications, creating robots, making search engines, creating computer graphics, and enabling all sorts of advanced technology?

What does a bunch of math have to do with anything? Especially things like proofs. Many of you probably think that what practical engineers and designers need are ways to do calculations, while the proofs are something arcane that are best left to the mathematicians who care about these sort of things. The reality is quite different. To paraphrase Prof. Bob Gallager[3], "in Engineering problems, the theorems seldom apply directly, but the proofs often do." Some of you may think that Mathematical rigor is stifling. In reality, mastering the rigor of proofs liberates your creativity — you are free to follow flights of fancy and speculation, secure in the knowledge that eventually, rigor will catch you and tell you if you're making a mistake.[4]

In this lecture note, we will try to metaphorically describe how mathematics relates to EECS. Although math is important for all science and engineering disciplines, the relationship is particularly close and intense when it comes to EECS. When you normally think of the power and utility of applied math, your intuitive

[1] Regardless of which door you are coming through: College of Engineering or Letters and Science.

[2] We're actually going to skip primality testing this term.

[3] A now retired Prof. at MIT, who has made foundational contributions to information theory, the theory of distributed computation, and computer networks.

[4] Rigor (together with the ability to simulate) lets you "fail fast, fail often, and fail cheap" — and this is of the secrets to letting creativity lead you to innovation and deeper understanding.

sense is for mathematics as a descriptive and predictive tool. Something that you can use to help you understand the real world.

Within EECS, that dimension is certainly present. But there is another one too.

Math as magic or wizardry

To understand the intimate relationship that EECS has with mathematics, the playful metaphor of magic is useful. As human beings who are a part of a culture in which "magic" plays an important role in literature, mythology, video-games, movies, etc. you have an intuitive appreciation for magic as an idea — even if we understand that magic is not really a part of our natural world, it is definitely a part of our range of mental models.

What makes magic different from mere understanding is its active force. In stories, a mage wishes not only to understand magic, but to manifest it in a controlled manner. To make magic flow through objects and animate them. That's how folks in EECS intuitively regard math.

Physical electronics

In devices and physical electronics, what folks in EECS do is look for physical materials that "resonate" or respond with certain mathematical properties, and then to weave the materials together into basic devices that are predictably responsive to mathematics. The objects are real, but they respond in mathematical ways that are particularly interesting.

Certain key mathematical operations are desired (e.g. gain (amplification), switching, inversion, integration, and memory, among others.) and we strive to realize these in ever more efficient ways. But at the same time, folks hunt to discover novel devices that exhibit mathematically interesting or powerful behaviors that might be different. For example, people try to build micromechanical resonators or devices that better harness effects related to quantum superposition[5]. This approach is also being applied to biological systems — rather than simply asking how biology works, EECS people engaged in synthetic biology strive to discover the mathematical potential of biological activity.

All this requires an eye and a taste for what sort of mathematical behaviors are interesting, how to recognize them when you see them, and how to enhance them once you do. To use a magical analogy, the discovery of novel devices is like the discovery of new magical herbs, magical ingredients, magical sounds, and magical strokes. But the true power of these magical ingredients is realized in their combination and interconnection. Herbs and ingredients have to be mixed together into potions, magical sounds have to be combined into magical words, and magical strokes have to be put together into magical characters.

Circuits

Magic potions, magical words, and magical characters: this is the metaphoric level that is represented by circuits. Devices have be combined and interconnected in a way that lets the math better course through them. Circuits create pathways for mathematical computations. Knowing what can and cannot be harnessed in combination helps one see what one is looking for.

Modern EECS systems involve circuits consisting of billions of discrete entities. But these are built out of smaller groupings in a hierarchical manner, where math is both used to understand how to analyze the

[5] This is an important part of the quest towards quantum computation.

interconnections as well as to inspire what the goals of the interconnection itself should be. Once woven together, imperfect devices bring forth more ideal and complex mathematical behaviors.

Modern electronic devices and circuits are truly revolutionary. But within this tremendous general advance, there are three particularly immense capabilities that modern devices and circuits have enabled:

- Automated reliable communication over long distances of vast amounts of information.
- Automated reliable and modifiable storage of vast amounts of information over time.
- Automated programmable universal computation at high speed.

The three of these ongoing triumphs of EECS combine to make our current age possible. Each one has an intuitively magical quality to it that is directly perceptible even to an untrained person.

Programming universal computers

Yet even among the three capabilities above, the third (high speed universal computing) resonates most strongly with magical connotations for human beings simply because we use language — and the realization of universal programmable machines makes actually real what had only been mythical before: magical artifacts that are animated by incantations, incantations that must obey the strict rules of their language.

For people in EECS, the Mathematical Realm is at least as real as it is for most mathematicians. As the famous English mathematician Hardy said: "A mathematician, like a painter or a poet, is a maker of patterns. If his patterns are more permanent than theirs, it is because they are made with *ideas*." But in a sense, for people in EECS, the Mathematical Realm is *even more real* because its inhabitants can actually be made to visit us — subtle mathematical patterns can become physically manifest and useful.

Every computer program is a spell or incantation of sorts. These incantations, whatever the surface form that their language might seem to take, are at their heart mathematical in the source of their power. But what kind of incantations are these? In this metaphor, a universal computer is a kind of magical vessel. And with the right incantations, we actually compel mathematical beings to be summoned forth to animate the vessel. Programming is a kind of conjuring.

And unlike a pure mathematician's math that can surprise only people who can understand it, the math that we in EECS summon forth can surprise even those that can't understand it.

Programming matter

For EECS people, the idea of programming extends beyond universal computers. Being able to express incantations and abstractions using language is so powerful that it is used even for circuits and the like. Mathematics here is not just the language used to express the desired behavior, it is also used to automatically transform one representation of the behavior into others that are more compatible with the mathematical nature of the physical substrate.

Connections to other courses and a longer-term view

The centrality of mathematics to EECS means that there is no way that a single semester course like 70 can possibly complete the story for you. Instead, our primary goal in this course is to give you a taste of the field, with a particular emphasis on showing you both the elegance and power of mathematical thinking. There is

an aesthetic dimension here (one of the most important things about Math is that it is beautiful, elegant, and its different branches weave together in surprising ways.) and we want you to begin to "feel" what math can do and what powerful ideas are like. To do this, we are going to have to build some foundations and tools, but these will really be fleshed out and elaborated on in subsequent courses.

Follow-on Courses

Here is a partial list of courses that build substantially upon the material in 70 and can be taken immediately after 70 (assuming you've got the other prereqs):

- EECS 126 — This course in the department really builds on the basics of probability that you learn in the latter half of 70 and deepens your rigorous understanding of the basics as well as building more advanced concepts, in the context of many different applications. A solid understanding of the material in 126 is sufficient to access most of the first-year graduate courses, as far as probability goes. (It also prepares you for 189 — Machine Learning.)
- EECS 188 — This course builds on a part of the probability material covered in 70, especially inference. This combines optimization algorithms with probability — to give you access to the basic tools in modern "AI." (It can be good to take this concurrently with 126 — they complement each other conceptually.)
- EECS 121 — This course builds upon both the probability material and the error-correcting-code material in 70. It shows how mathematical thinking shows up in problems of digital communication and storage, and by going in-depth into these applications, it shows how to model problems so that it is easier to apply math to them.
- EECS 170 — This course builds upon the modulo arithmetic parts of 70 as well as the proof techniques. It shows how you can think about algorithms as mathematical objects. After 170, you can go on to take 172 (which will go more deeply into concepts related to uncomputability that we introduce at the end of 70) as well as 174 (which will build on the probability material in 70 and apply it to algorithms).
- EECS 122 — This course builds upon some of the probabilistic analysis learned in 70, especially the material on load balancing, to help understand computer networks.
- EECS 161 — This course builds upon the RSA material and secret-sharing, and shows how to weave cryptographic protocols out of those primitives and to apply them to computer security questions.
- Math 113 — This course in the Math department will give you the tools to build beyond the basic modulo arithmetic and polynomial properties you learn in 70.

A longer term view

Fields constantly evolve and the education you receive now should lay the foundation for lifelong[6] learning and flexibility. It is hard to predict with any certainty what is going to happen in the four decades to come. However, we can discern trends that are likely to be active, at least for the coming decade or two.

[6] Instead of merely aiming at your first job.

- **Not just computation alone.** The triad of computation, communication, and storage will continue to transform the world. But many of the advances will involve these interacting with the physical world around us, the biological world within us, and social world among us instead of being simply being about new "apps" used by individual humans and new computing/communication/storage platforms to host those apps.
- **Not just abstractions.** Abstractions and hierarchical/layered approaches to design will continue to be important. But as Moore's law changes its qualitative character (not just faster, cheaper, and more efficient serial computation "for free") and new technologies like 3D printing, custom printable circuits, synthetic biological computing, etc. become widespread, the nature of design will increasingly involve exploiting synergies across these. Programmability and creative design will increasingly reach down into lower-layer aspects that had previously been considered to be fixed commodities.
- **Not just models.** The use of well-designed hand-crafted models for problems will also continue. But as it becomes possible to acquire, store, and process increasing amounts of information, models will increasingly be complemented with data and what can be learned from it. Increasingly, the right answer to a challenge will be how to acquire the right additional data that will make the problem easier rather than simply figuring out a purely computational approach to solve a hard question.
- **Not just exact answers.** The digital revolution will continue to leverage its ability to give stark yes/no answers to well-defined questions. But increasingly, approximate answers will play a role in the design and operation of useful systems.

Students will be well served by being prepared for these ongoing trends by having exposure to courses that help them understand the basics in different areas, but also courses where they see how such issues are approached mathematically and how the mathematics helps guide the engineering.