A Self-Learning, Modern Computer Science Curriculum

Table of Contents

1 Introduction

> "The great idea of constructive type theory is that there is no distinction between programs and proofs… Theorems are types (specifications), and proofs are programs" (Robert Harper)

This is a collection of modern resources on various undergrad level computer science topics, for someone with an interest in theory. Use WorldCat or LibGen if you can't buy these books. You don't have to do everything here, just whatever interests you. If I list a resource here it's because I either completed the whole resource or I used parts of it for something I needed to learn and found it a quality enough resource to include here for somebody else who wants deeper coverage of these topics.

1.1 Studying Strategies

On Isaac Newton's iteration method to self-learn geometry:

> "He bought Descartes' Geometry and read it by himself .. when he was got over 2 or 3 pages he could understand no farther, than he began again and got 3 or 4 pages father till he came to another difficult place, than he began again and advanced farther and continued so doing till he made himself master of the whole without having the least light or instruction from anybody" (King's Cam.,Keynes MS 130.10,fol. 2/v/)

Numerous anecdotes exist on studying strategies, like the Feynman method explained here "If you can’t, out loud or on paper, explain the idea without confusion or contradiction, stop and figure it out right there". There's some books that model that method, like Gilbert Strang's *Calculus* has you reciting back the entire chapter you just read.

If you ignore the copious amounts of marketing on his site, Cal Newport has some other interesting anecdotes on studying, such as how he was able to get the best grade in his Discrete Mathematics class, and the rest of the site is full of advice on studying, how to schedule yourself and deliberate practice.

My personal advice is just commit some time everyday to learning, and life will give you days off as things come up, you don't really need a schedule just do something daily and magically by the end of the year you will have achieved a huge amount of work. Always get the errata for what you're reading, even course notes sometimes have errata on the author's page, and always take something a little harder than your skill level so then it becomes a research exercise backfilling all the requirements. For example many people want to relearn math they forgot, so they start working through some enormous 1000+ page pre-calculus book and lose interest after the first few chapters. Instead make the goal to learn calculus, and start there, using the huge pre-calc book as your research project. This is what the exercises are for you pick up all the algebra from highschool you forgot while grinding through a calc book.

1.2 Discord channel

If you want to learn this material with a community, some anons have started a Discord channel.

2 Preliminaries

2.1 Introduction to Computer Science

I've rewritten this entire section into a workshop we work through Brown University's CS019 class and Tao's *Analysis I*, plus many other topics to fill in beginner blanks.

They're draft quality notes but the idea is if you get stuck on something, you can look at them like asking a TA for help.

- (Full Course) CS019 Accelerated Intro to CS
  - Lectures here
  - Use the lastest version of PAPL
  - You can complete this course on a phone/tablet if needed, you use a simple online IDE for the assignments

Torrent archive

```
2018 Lectures, labs, recorded recitations
magnet:?xt=urn:btih:891283aff7f336ba6c56ed47e03a2f72a9b186d0&dn=
```

2.2 Tools

MIT's the missing semester of your CS education. Editors such as emacs/vim are also introduced here but any IDE will do you prefer like Atom or online ones like replit. The 1980s book The Unix Programming Environment by Kernighan & Pike walks through Awk/Grep/Sed and shell scripting too.

2.3 Assumed Mathematical Background

> "So I went to Case, and the Dean of Case says to us, it's a all men's school, "Men, look at, look to the person on your left, and the person on your right. One of you isn't going to be here next year; one of you is going to fail." So I get to Case, and again I'm studying all the time, working really hard on my classes, and so for that I had to be kind of a machine. In high school, our math program wasn't much, and I had never heard of calculus until I got to college. But the calculus book that

> we had was (in college) was great, and in the back of the book there were supplementary problems that weren't assigned by the teacher. So this was a famous calculus text by a man named George Thomas (second edition), and I mention it especially because it was one of the first books published by Addison-Wesley, and I loved this calculus book so much that later I chose Addison-Wesley to be the publisher of my own book. Our teacher would assign, say, the even numbered problems, or something like that (from the book). I would also do the odd numbered problems. In the back of Thomas's book he had supplementary problems, the teacher didn't assign the supplementary problems; I worked the supplementary problems. I was scared I wouldn't learn calculus, so I worked hard on it, and it turned out that of course it took me longer to solve all these problems than the kids who were only working on what was assigned, at first. But after a year, I could do all of those problems in the same time as my classmates were doing the assigned problems, and after that *I could just coast in mathematics, because I'd learned how to solve problems*" (Don Knuth)

Try the new math from scratch experimental curriculum we'll try and teach ourselves the math needed to try most of the courses here. There's also calculus and linear algebra.

3 Functional Programming

3.1 Principles of Functional Programming

Robert Harper keeps a blog and a follow up post on the success of teaching this material to undergrads. Since the CMU 15-150 course is locked down now we can instead do Dan

Licata's version at Wesleyan (PhD advised by Robert Harper) that has short recorded lectures and assignments open to the public, it's very similar to the original 15-150 curriculum.

- (Full Course) COMP 212 - Functional Programming
  - This course differs from CS19 in that you write proofs of specs, focus on parallelism

```
The labs and hw are locked for COMP212 now, but you can get the
 - de-select the video lectures if you want a smaller download

magnet:?xt=urn:btih:4802b90e2b2c0f419690b0a09ee8e2128a92d32f&dn=
```

3.2 CS 3110 Data Structures and Functional Programming

Cornell's version uses OCaml, their book now has embedded YouTube videos and you can download a VM image to run OCaml as they have set it up in the course. Some overlap with CS19 in the exercises if you already did CS19, COMP212 focuses more on parallelism and the exercises much more difficult, try both courses they compliment each other.

4 Algebra

It is possible in a functional language like ML to do algebra with types, proving two types are isomorphic with the desired properties of reflexivity, symmetry, and transitivity. It's also possible to abstract Lists and Trees into polynomials, as every polynomial looks like a sum of terms. As you will learn in 15-150 *Principles of Functional Programming* "most functional datastructures have constant time access near the outer layer of their structure, ie: the head of a list or the root of a tree.

However, access at some random point inside the structure is typically linear since looking at some element of a list is linear in the length of the list, and looking at some element of a tree is linear in the depth of the tree. Datatype derivatives allow constant time access to the entire structure."

4.1 Linear Algebra

I now have my own Modern Applied Linear Algebra workshop if you want to do MIT's latest 18.06, Terence Tao's notes and some Wildberger lectures.

- (Full Course) CS053 - Coding the Matrix
  - A lot of linear algebra over the complex field
  - No formal prerequisites except assumes you know how to do basic proofs, book uses python bu you can do this course in any language, they all have linear algebra libraries
    - A crash course in linear algebra if you've taken some before

4.2 Abstract Algebra

See the crypto workshop, abstract algebra will done there.

5 Discrete Mathematics

5.1 Math Workshop

Poh-Shen Loh has lectures on his YouTube channel for CMU's 21-228 Discrete Math class. We do it here.

5.2 Discrete Math with Standard ML

If you don't know how to program this book will teach you by modeling math functions in SML, polynomials into lists,

sets into types, creating relations and proving their results with programs.

- (Book/Lectures) Discrete Mathematics and Functional Programming - Thomas VanDrunen
  - Lectures exist on the author's homepage, this book is used for a one semester university course with additional elective chapters in Graph Theory, Complexity Theory, Automata, etc.

5.3 Great Theoretical Ideas in Computer Science

This is a crash course in multiple topics such as Probability, Linear Algebra, Modular Arithmetic, Polynomials, Cryptography and Complexity Theory.

- YouTube lectures (missing some) and course notes with solutions on Anil Ada's page. Panopto still has the 2013 course here taught by a younger Ryan O'donnell and the missing lectures not in the youtube playlist.

6 Computer Systems & Architecture

> "[Computer science] is not really about computers - and it's not about computers in the same sense that physics is not really about particle accelerators, and biology is not about microscopes and Petri dishes…and geometry isn't really about using surveying instruments. Now the reason that we think computer science is about computers is pretty much the same reason that the Egyptians thought geometry was about surveying instruments: when some field is just getting started and you don't really understand it very well, it's very easy to confuse the essence of what you're doing with the tools that you use." (Hal Abelson)

6.1 x86 Computer Systems

This covers architecture from a programmer's perspective, such as how to write cache friendly code/optimizations, assembly, how compilers work, return oriented programming (ROP) to bypass stack protections, the memory hierarchy, and networks. You could read K&R's *The C Programming Language* for a brief intro, though this course will explain C as you go anyway and fully covers pointers at the assembly language level making it self contained.

- (Full Course) 15-213 Intro to Computer Systems (CMU)
  - Recorded lectures here or download them here
    - YouTube has lectures uploaded for this course too for multiple semesters
  - Uses CS:APP book, 3rd version for x86-64 beware of the global version errata
  - If you want a well written crash course in C read these notes)
  - The labs (Data Lab, Bomb Lab ect) are on the book website
    - Try the embedded security CTF here after you finish the Attack Lab

I do some of this course here

6.2 RISC-V Architecture

MIT's 6.004 course covers RISC-V: The free and open RISC Instruction Set Architecture. Take this if you are interested in programming FPGAs or cheap embedded modules.

- (Full Course) 6.004 Computation Structures
  - Recorded YouTube lectures here
  - Book used is Computer Organization and Design: RISC-V Edition

6.3 Operating Systems

MIT teaches OS engineering using a rewrite of the sixth edition Unix(v6) similar to the classic Lions' Commentary but in ANSI-C called xv6. The idea is you read the source as you read book, to understand the system. MIT has a full course w/YouTube lectures here or find most recent semester.

The latest xv6 source and text are available via:

- git clone git://github.com/mit-pdos/xv6-riscv.git
- git clone git://github.com/mit-pdos/xv6-riscv-book.git
    - If you have problems building the book, get the pdf

7 Compilers

The course has a recorded screencast

- (Full Course) CSE 131: Compilers
    - Recorded lectures/podcast here uses OCaml
    - There exists a tutorial by Abdulaziz Ghuloum in Scheme
    - See also the MIT course 18

8 Database Systems

Interesting post on the future of database systems by Andy Pavlo.

- (Full Course) 15-445 Intro to Database Systems
    - Recorded lectures on YouTube, course changes every year take the most recent version
    - Uses readings from this book, assumes you have taken 15-213 and know some basic set theory

8.1 Advanced Database Systems

At the end of the semester the grading scripts are publically available.

- (Full Course) 15-721 Advanced Database Systems
  - Recorded lectures on YouTube (find the most recent ones)
  - Everything is in C++17 which you can teach yourself from the bounty of books/documentation available via search engine.
  - Purpose of this course is to learn how to design and build your own dbms

9 Web

Here are the (poor) notes I took for MIT's 6.148 Web Programming Competition and MIT's 6.170 Software Studio. I did most of these assignments on a phone using Eruda since I wrote this while on lunch breaks. The MIT workshop and 6.170 are actually pretty good, giving you just enough to make a MVP yourself, expecting you to fill in the rest of the details on your own now that you know the basics of express, react, CSS etc.

- An advanced introduction to JavaScript

10 Data Science

10.1 Practical Data Science

Uses Python, teaches a crash course in linear algebra and probability

- (Full Course) 15-388 Practical Data Science
  - Recorded lectures are here and archived here
  - Introduction to the 'full stack' of data science analysis: data collection and processing, data visualization and presentation, statistical model

building using machine learning, and big data techniques for scaling these methods.
    - Cornell also has a free text with interactive jupyter notebooks (in Python)

10.2 Introduction to Computational Thinking

MIT course featuring the 3blue1brown linear algebra guy as a lecturer.

- (Full Course) MIT 18.S191 Intro to Computational thinking sp2021
    - Image transformations, learning, how a differential model works

10.3 Advanced Data Science

Cozma Shalizi is an ex Physicist, so he approaches statistics from that viewpoint which is refreshing. He is a research statistician so this won't be a recipe book.

- (Book) Advanced Data Science from an Elementary Point of View
    - Some overlap with his other book *The Truth About Linear Regression*
        - Don't have the prereqs? See his notebooks like how to teach statistics

11 Symbolic Programming

Try Sussman's new book *Software Design for Flexibility* you only need the appendix on Scheme to start, most of this is self-contained, assumes you know the equivalent of his other book *Structure and Interpretation of Computer Programs* or a CS19 style intro course.

12 Natural Language Processing

- (Full Course) 11-411 Natural Language Processing
  - Recorded lectures on YouTube find the most recent ones
    - Some notes on NLP algorithms

13 Programming Language Theory

I go through Robert Harper's PFPL book here there's also an Agda workshop with screencasts as a replacement for Pierce's *Types and Programming Languages*.

14 Isolating Software Failure, Proving Safety and Testing

How to verify software, and strategies of programming that minimize catastrophe during failure. I do some of this here.

- (Lecture Notes) 15-316 Software Foundations of Security and Privacy
  - Covers side channel attacks, provable privacy, web security, proving the validity of a memory sandbox
  - Read about tests, strategies to safe program design (in OCaml), and proving safety from the 2017 version: git clone https://github.com/jeanqasaur/cmu-15316-spring17.git
  - MIT's 6.858 *Computer Systems Security* is fully open with lectures

Brown's CS195y *Logic For Systems* I made a torrent for if you want to model state in a way similar to Alloy, where you can prove it's logic before writing the program. Think about designing a security level scheme where different users have different access levels, you don't want some unforseen combination of states to result in granting

access that shouldn't be granted. The *Software Abstractions* book and 195y course help you with this by providing a way to model your security scheme before you write the code using something similar to set theory proofs.

```
CS195y partial archive, some recorded lectures missing:
magnet:?xt=urn:btih:3187a9951b43b85771a975653680a4a8d9faf61d&dn=
```

14.1 Physical Systems Software Security

- (Full Course) 15-424 Foundations of Cyber-Physical Systems
  - Course (with recorded lectures) if you're interested in programming drones/space shuttles/robots/cars and other things that cannot fail from avoidable errors. Find the latest version.
  - Mainly self contained, you may have to look up some lectures on differential equations, try Strang's 18.085 MIT lectures for *Computational Engineering* on OCW.

15 Workshop in Algorithms

If you want to learn beginner to advanced algorithms try here where we practice mixing competitive programming with standard algorithms texts.

Here's a partial archive of 15-451 if you want to learn about NP-Completeness, approximation algorithms, online algorithms, graph compression, multiplicative weights, computational geometry, fast multiplication, embeddings, FFT some other topics:

```
Recorded lectures
15-451 partial archive:
magnet:?xt=urn:btih:0cd3da156921d9264a7e06b3fb148130037e1228&dn=
```

15.1 Introduction to Parallel and Sequential Algorithms

Search libgen for 'Parallel and Sequential Algorithms' and get latest version. I archived all the lecture notes and some of the labs before they disappeared here. The book is excellent, it goes through many common algorithms you already know about and then teaches you how to make them more parallel and is used in CMU's 15-210 course which now is sealed up behind campus logins.

15-853 *Algorithms in the Real World* is another excellent resource that covers some 15-210 content and more.

15.2 Functional Data Structures

"A stroll through intermediate data structures and their associated algorithms, from the point of view of functional programming." Note how pattern matching makes implementing these seemingly complicated structures a much easier task.

- (Book) Functional Data Structures
  - Exercises and examples done in OCaml, no formal background beyond a typical introductory CS course
  - A more advanced book Purely Functional Data Structures by Chris Okasaki uses an imaginary version of SML with Haskell solutions in the appendix.

15.3 Advanced Data Structures

Taught by Erik Demaine's at MIT, mixes 2017 lectures with MIT OCW 2012 lectures, I used this course to learn cache-oblivious data structures while taking 15-853 *Algorithms in the Real World*.

- (Full Course) 6.851 Advanced Data Structures

16 Complexity Theory

16.1 Undergraduate Complexity Theory

Expands on the lectures in 15-251.

- (Full Course) 15-455 Undergraduate Complexity Theory
  - Recorded lectures here
  - Uses Part III of the Sipser book *Introduction to the Theory of Computation*
  - A 2017 survey of the current state of P vs. NP by Scott Aaronson

16.2 Graduate Complexity Theory

You may want to try solving some of the problems in this domain.

- (Full Course) 15-855 Graduate Computational Complexity Theory
  - Recorded lectures here
  - Uses the book (free draft) *Computational Complexity - A Modern Approach*

16.3 Useful Math for Theoretical CS

These scribed lecture notes give you a diverse background in math useful for theoretical CS, from the excellent book The Nature of Computation such as these slides from *A Theorist's ToolKit* which describe how to find research,

how to write math in LaTeX, how to give a talk, resources on math.overflow etc.

Prof Ryan O'Donnell has now uploaded lectures.

17 Introduction to Quantum Computing

This is a graduate introduction to quantum computation/information theory, from the perspective of theoretical computer science.

- (Full Course) 15-859BB: Quantum Computation
  - Recorded lectures on YouTube.
  - The prereqs are an undergrad background in complexity theory (15-251 or 15-455), linear algebra, and discrete probability.
  - "90% of the understanding of the quantum circuit model is achieved by reviewing three purely 'classical' topics: classical Boolean circuits; reversible classical circuits; and randomized computation"
    - A series of curated papers on Quantum Computing

18 Performance Engineering

If you've taken 15-213 Computer Systems or 6.004 RISC-V Architecture, and understand basic graph algorithms and matrix multiplication you will understand this course, as it covers performance analysis, instruction-level optimizations, caching optimizations, and other techniques for high performance, scalable systems. The course is done in C but the concepts apply to any compiled language.

- (Lecture Notes) - 6.172 Performance Engineering of Software Systems
  - Click on 'Export Materials ZIP' before they're gone

- There are older lectures for this course on MIT OCW/YouTube with similar content
- See the first slide 'Intro & Matrix Multiplication' for a good desc of this course

19 Topics in Mathematics for Finance

Notice there's a button for "Export Materials ZIP". 18.642 also has an older course on Open CourseWare with lecture videos. You may also find interesting this Quantitative trading summary and this old post on pricing & hedging models.

20 Jobs

This service matches your skills to people who want to pay you. Jane Street Capital is a finance tech company that hires functional programmers worldwide, you may want to apply there (after practicing a lot on Kattis). There are numerous opportunities to apprentice as a security researcher. There are also bounties for writing features available on topcoder, gitcoin or bountysource A large collection of remote job listing and consulting platforms is here.

I recommend checking local schools and labs in your area for research programmer positions, students often do not apply to these campus jobs as they are chasing internships and it's where I got started.

21 Workshop in Computational Type Theory

I decided to make a new workshop here to learn cubical/computational type theory.

22 Workshop in Machine Learning/AI

There exists an ongoing workshop to teach ourselves machine learning/ neural networks doing 3 courses at once.

22.1 General AI

You can still watch Berkeley's CS188 on Youtube, uses Norvig's new 4th edition book out in 2020 but any edition you can find will do including the old Lisp addition that is available free on github. Just write your own autonomous agents for exercises. Sussman's grad course is the perfect match to CS188 reviewing older papers with unsolved problems or deep ideas, like his own *Art of the Propagator* paper and implementation. There is also AI readings to explain intelligence from a computational point of view both these are nice compliments.

22.2 Math Background for ML

This free book *Mathematics for Machine Learning* will describe the math often used in ML. There's also a series of recorded lectures and recommended texts from CMU and a crash course in linear algebra here and a set of notes. You may also enjoy the Vector Boot Camp from Brown for the multivariable calculus content.

22.3 Statistics Theory

- (Full Course) 36-705 Intermediate Statistics
  - The author of *All of Statistics* Larry Wasserman covers Chapters 1 - 12 from his book and more covering the fundamentals of theoretical statistics in these recorded lectures
  - The lectures assume you already read Chapter 1 - 2
  - There are many probability lectures around to go with the beginning chapters such as here

    - The quality of these can be changed to 1080p but the audio is terrible, however you can extract the audio easily, and use Audacity (free) to suppress some of the background noise and boost his voice.

22.4 Modern Regression

CMU professor Cosma Shalizi has a great set of lecture notes *The Truth About Linear Regression* in fact if you go through his extremely large page on notebooks he will explain the insight to virtually everything statistics related such as what are stochastic differential equations and notes about teaching statistics which books are recommended. I wish more professors did this.

22.5 Introduction to Machine Learning

Torrent archive:

```
Recorded zoom lectures, homework:
magnet:?xt=urn:btih:2e1005d058b5f4c357d7338c85937aabdd91dcdc&dn=
```

- (Full Course) 10-301/601 Sp 2020 Introduction to Machine Learning
    - Assignments are in your language of choice
    - We're going to do this in the AI workshop

22.6 Introduction to Practical Machine Learning

"This course will teach you basic skills to decide which learning algorithm to use for what problem, code up your own learning algorithm and evaluate and debug it." You don't have to use Python, the prof in these lectures shills Julia for students to try instead.

- (Full Course) CS4780 Machine Learning for Intelligent Systems

    - YouTube lectures, additional reading recommendations and videos for each topic
    - Kaggle has numerous tutorials and competitions available
    - We do this course in the AI workshop

22.7 Graduate Introduction to ML

There is also Cornell's advanced ML class though they deleted the recorded lectures like most other schools post-covid19

- (Full Course) 10-701 PhD Introduction to ML Fall 2020
    - Lectures are on YouTube, we go through this course in the AI workshop
    - Use another semester for recommended reading
    - Same professor teaches 10-708 for problems with a very large number of attributes and huge datasets
    - Try 18-337j a class in optimization (numerical analysis) and parallel ML using Julia, lectures are on YouTube

22.8 Advanced Statistical learning theory

- (Lectures) - 10-702 Statistical Machine Learning Theory
    - A 2016 second grad course in statistical learning theory
    - Assumes you have taken 36-705 (his book *All of Statistics*) and some kind of ML intro class at the level of 10-701
    - Intended to be more advanced than the book *The Elements of Statistical Learning Theory*
        - This course is now called 36-708 Statistical Methods for Machine Learning w/course notes

22.9 Convex Optimization

Used to work on a variety of problems, recently research is in non-convex optimization of neural networks, search google scholar for 'non-convex'.

- (Full Course) 10-725 Convex Optimization
  - Search YouTube for '10-725' to get most recent recorded lectures
  - Completely self contained, see the syllabus
  - Some of this is in Wasserman's 10-702 above

22.10 Deep Learning

- (Full Course) 11-785 Introduction to Deep Learning
  - Recorded lectures here to learn to build and tune deep learning models
    - If the playlist is deleted, which is frequently, (use youtube-dl to archive) search YouTube for "CMU 11-785"
    - Some slides on techniques for making deep learning robust to adversarial manipulation
    - Thorough lecture notes and jupyter books on scalable machine learning systems

22.11 Further Research

- Search 'AI competitions' and try some past competitions
- Open challenges in ML
- Algorithmia.com an open marketplace for AI algorithms
- Cosma Shilizi's notes on machine learning and learning theory

23 Workshop in Cryptography

I am working through Tanja Lange's course in post-quantum crypto and cryptanalysis in the new crypto workshop

Daniel J. Bernstein's posts on the IETF Crypto Forum Research Group [Cfrg] archive is a master class in modern cryptanalysis and he rips apart bad standards/protocol designs, we will try to understand some of these in the crypto workshop above.