ECE3340
Numerical Methods for Electrical and Computer Engineering

PROF. HAN Q. LE
If you love math & computer coding, you will very likely enjoy the course

...If you hate math, or suffer doing math & computer coding, well,... are you sure you are in the right place?
An example of “numerical math” (not pure math)
An example of pure (number theory) math

Prove that: $2+2=4$

Or...

Prove that there are more irrational numbers between $(0, 1)$ than all integers or rational numbers

Or...

No three integers $a$, $b$, $c$ can satisfy the equation below for integer $n>2$

$$a^n + b^n = c^n$$
Obviously, this course is about applied math for scientific and electrical engineering applications.

Physical quantities, e. g. mass, velocity, voltage, current, frequency, temperature, pressure, flux, power, energy, and whatnots... are expressed as real numbers with certain behaviors & relationships.

A practical scientific/engineering problem isn’t solved with abstract symbols ($v, i, \omega, P...$), but with numerical values: A need of numerical methods to obtain accurate and precise numerical representations of the physical quantities of interest for understanding (“insight”) and applications.
Example

Q.1 What is a most common name for this circuit?

Examples without inductor L

Q.2 What is a most common name for these two circuits? (extra credits if you name each one differently based on its function and not by a permutation of letters)
Switch A was connected for a long time \( t < 0 \) and switch B was open. Then switch A is opened at time \( t_A \leq 0 \); simultaneously or after \( t_A \), switch B is instantaneously closed at \( t = 0 \).

Consider this circuit that has no resistors, no dissipative elements (things that absorb power or energy permanently and not give back to the circuit). Assume perfect capacitor and perfect inductor.

Q.3 What is the voltage \( v_{\text{out}} \) before and after the switches are activated? Sketch your guess what it looks like (only “guessing” is asked, no solving equation or anything complicated).
Q.4 For all time t<0, input x[t]=1 Volt. Suddenly at t=0, the input is switched to 0 V (with zero impedance at the input). Sketch your best guess what voltage y[t] looks like as a function of time for t>=0. Extra credit if you write the correct mathematical expression for y[t].

(Yes, you can Internet-search for an answer, here is an example: http://mathlets.org/mathlets/series-rlc-circuit/)

Numerical calculation can give us practical results (with values for actual application) as well as insight for designing, modification, and invention!
When design a circuit, we want to know how it works, what it does for the intended application. We want numerical simulation results such as this:

This is what “numerical methods” is about. This is your objective: learn how to use computers to apply to ECE problems.
Although not scientific or engineering, this is an example of a numerical problem involving computer and programming.
Without a computer, it may take days, weeks for someone to do by hand, which serves no useful purpose other than knowing the company to send your resume in (if looking for a job)... (“googling” it!)
Just like half of the world's spoken tongues, most of the 2,300-plus computer programming languages are either endangered or extinct. As powerhouses C++, Visual Basic, Cobol, Java and other modern source codes dominate our systems, hundreds of older languages are running out of life.

An ad hoc collection of engineers electronic lexicographers, if you will—aim to save, or at least document the lingo of classic software. They're combing the globe's 9 million developers in search of coders still fluent in these nearly forgotten lingus frangas. Among the most endangered are Ada, APL, B (the predecessor of C), Lap, Oberon, Smalltalk, and Simula.

Code-chucker Grady Booch, Rational Software's chief scientist, is working with the Computer History Museum in Silicon Valley to record and, in some cases, maintain languages by writing new compilers so our ever-changing hardware can grok the code. Why bother? "They tell us about the state of software practice, the minds of their inventors, and the technical, social, and economic forces that shaped history at the time," Booch explains. "They'll provide the raw material for software archaeologists, historians, and developers to learn what worked, what was brilliant, and what was an open failure." Here's a peek at the strongest branches of programming's family tree. For a nearly exhaustive rundown, check out the Language List at http://www.informatik.uni-freiburg.de/Java/misc/lang_list.html. —Michael Mendes
Most Popular Coding Languages of 2014

- Python 30.3%
- C++ 13%
- Ruby 10.6%
- Java 22.2%
- Javascript 5.2%
- C# 5%
- C 4.1%
- PHP 3.3%
- Go 1.5%
- Bash 1.1%
- Clojure 0.2%
- Objective C 0.4%
- Perl 1.6%
- Scala 1%
- TCL 0.03%
- Lua 0.04%

@codeeval
www.codeeval.com
Most Popular Desktop IDEs & Code Editors in 2014

- 28.09% Notepad++
- 23.14% SublimeText
- 17.67% Eclipse
- 8.30% NetBeans
- 4.77% IntelliJ
- 4.43% Vim
- 3.89% Visual Studio
- 3.71% PhpStorm
- 3.00% Atom
- 1.94% Emacs
- 1.06% Zend

Get more information at https://blog.codeanywhere.com/most-popular-ides-code-editors
Juno: ≈ 28,000,000 Downloads

- IDE for Java EE Developers: 34%
- IDE for Java and DSL Developers: 14%
- IDE for Java and Report Developers: 7%
- For RCP and RAP Developers
- Modeling Tools
- For Testers
- IDE for Automotive Software Developers
- For Parallel Application Developers
- For Scout Developers
- For Mobile Developers
- Classic: 24%
Numerical methods for scientific and electrical engineering applications

- This course is hardware-and-software-agnostic.
- Computers & software are crucial, but your choice (Mac OS, Windows, linux, ... C++, C#, BASIC...).
- The course content is platform&IDE-independent.
About of this course (1)

- Mathematics is beautiful by itself. But this course is not about mathematics – It’s about math application to scientific/ engineering problems.

- “Just-in-time” learning: We start out with problem examples, and learn what needed to solve. \(\text{(learn as we go)}\).

- The official software package of Course materials is \textit{Mathematica®}. Recommend Math 9 version (if you can still find it), otherwise, Math 10 is OK (slower and buggier). You can do HW in any computer language of your preference (C++, C#, BASIC, FORTRAN, MATLAB, Java, or even Excel...).
U of Houston licensed software

Free IDE stuffs (many...)

- MATLAB
- Wolfram Mathematica
- Get the Wolfram CDF Player
- Eclipse
- Python
- JavaScript IDE
- Facebook

official for Course lectures & HW (cdf is a free browser plug-in)
Hold it, wait... MATLAB and Mathematica? Are they computer languages also?
Higher level - specialized for scientific/engineering app.

Lower level implementation – general app. (web, games)
Course material formats for download

At least 1 of these formats, often 2 or more

Alternative formats
(rare – converted to pdf or xps whenever possible)
About this course (2)

- Review and learn some essential mathematics that have widespread applications in scientific/engineering problems.
- The three most important things of the course are application, application, and application. “It’s all about APPs!”
- Computer usage is of course the essential vehicle of learning. Bring your computer to do work in class as often as possible.

What this course is not

- Learning computer programming. That’s in other courses, e.g. ECE 3331. Practice makes perfect: whether you learned it in 3331 or you may already be an APP developer since HS, you are strongly encouraged to hone your programming skill by practicing what you learned. (Will help you to review/refresh your memory as much as possible).
If you love **math AND computer programming**, you will very likely enjoy the course.

- The more coding you do, the better you will be at it
- Reinforcing your knowledge of electrical engineering (in this case, we pick circuit to be the example topic for numerical math)

Math knowledge at MATH 3321 level and proficiency in computer programming at ECE 3331 level or better are prerequisite.
ECE 3340 - Numerical Methods for Electrical and Computer Engineers
Credit Hours: 3.0
Lecture Contact Hours: 3    Lab Contact Hours: 0

Prerequisite: ECE 3331 and MATH 3321.

Basic linear algebra and numerical methods with electrical engineering applications. Emphasis on use of computer-based solution techniques.

We will not learn “numerical methods” in theory and abstract, but by specific examples and applications in electrical engineering.
Expected work in this course

- **Active lecture and class participation**: you will participate to contribute to lecture and discussion in class

- **Classwork is more important than HW**: Bring paper, pens and your computer to do work in class. Turn in at the end of the class for credit.

- Most important: take ownership of your own education: contribute to your own “lecture.”

- **Homework**: First, try to do on your own. Then get help and collaborate if needed (peer learning). Submit on time and follow policy on late HW.

- A good combination of personal best effort and/or peer interactive learning can help you a great deal.

- **Tests/exams**: to be determined. No need for them if the other two categories are “good enough.”

- **Final personalized project**: to be determined.
Two reference textbooks

Very easy to read and practical (used in ECE 2331)

This book is a classic!

And like many other classics...
A classic is something that everybody wants to have read and nobody wants to read.

~ Mark Twain
• A beautiful work – a lifetime of wisdom from the author, R. W. Hamming.

• It’s hard to absorb the wisdom of a lifetime in a short semester. *(you wouldn’t need taking this course if you read and understand the book)*

• We will use it as a reference textbook:
  • Much of things in the book are already used and implemented in commercially available or free software.
  • Some are still relevant, some are less crucial because of increasing computing power and more sophisticated (intelligent) software.

Also, very useful website by Dr. Mayerich
http://stim.ee.uh.edu/education/ece-3340-numerical-methods/
In practice

- Rarely, if ever - do practicing scientists/engineers have to write own codes for utility apps such as find roots, integration, differential eqs., FDTD, FEM, special functions, linear/nonlinear systems,... and many others.

- There are huge libraries for those things in commercial or free software, developed by dedicated professional using well established or latest algorithm development, debugged, optimized for CPU speed, memory... Why re-invent the wheel?

- Actually, in a professional world, we would insist any user to use established commercial software - don't write your own except for own research: higher risk of bugs and errors. If you need a car, would you rather buy one or drive a contraption built by your colleagues?

- This course helps you become an intelligent, informed user of scientific/engineering software to be a professional electrical engineer - but not to develop you into a professional software developers: this is the job for computer science major at PhD level.
HW 0 (yes, it counts)
Find the first 8-20 consecutive digits in π decimal expansion that is a prime number which also contains your birth month (1-12), birth date (1-31), and last two digits of your birth year (in any order & overlap OK). This is your private encryption key.

Everyone has the same answer ("googling it")

Any two in the class have the same birthdate?
A preview for the next lecture

Introduction to the limit of computer numerical accuracy

CHECK OUT THE PERFORMANCE OF YOUR COMPUTER
Run the APP to get familiar with it. We’ll go through details next time.

<table>
<thead>
<tr>
<th>Basic check of your machine floating-point precision</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Smallest machine-handled number</strong></td>
</tr>
<tr>
<td>Decimal: $2.22507 \times 10^{-308}$</td>
</tr>
<tr>
<td><strong>Largest machine-handled number</strong></td>
</tr>
<tr>
<td>Decimal: $1.79769 \times 10^{308}$</td>
</tr>
<tr>
<td><strong>Smallest relative difference between 2 machine numbers</strong></td>
</tr>
<tr>
<td>Decimal: $2.22045 \times 10^{-16}$</td>
</tr>
</tbody>
</table>
Test floating point input with machine-precision number

Input a real number between 0. and 1.

Select display precision

number input: 0.3
actual value: 0.2999999999 9999988889 77698

discrepancy (error): 3.70074 \times 10^{-17}

Test floating point input with machine-precision number

Input a real number between 0. and 1.

Select display precision

Computer internal value vs. input

Blue: ideal
Red: actual

Input = 1 + x \times 10^{-16}
Test your computer with basic function accuracy

Floating point value of abs of $1/Tan\left[\frac{\pi}{2} \pm 2\pi n\right]$ (Cot)

Sign of $Tan\left[\frac{\pi}{2} \pm 2\pi n\right]$
Test your computer with basic function accuracy

Floating point value of $\frac{1}{n} \log(e^n) - 1$

Machine precision
Mathematica analytic precision
Floating point value of $\frac{1}{N} e^{\log(N)} - 1$