ECE 5650/4650
Modern Digital Signal Processing

Fall Semester 2025

Instructor: Dr. Mark Wickert **Office:** EN-242 **Phone:** 719-332-6996
mwickert@uccs.edu **Fax:** 719-262-3589
https://faculty.uccs.edu/mwickert/

Office Hrs: Wed. after class including during/after the ECE 4680 DSP lab, others by appointment, including *Slack*, *Zoom*, or *Teams*. (my preference is Slack)

Required Texts: Alan V. Oppenhiem and Ronald V. Schafer, *Discrete-Time Signal Processing*, third edition, Prentice-Hall, Englewood Cliffs, New Jersey, 1999.

Notes Course lecture notes are posted on the course Web Site as PDF files. Students are encouraged to download and print them.

Modeling Software: Python and Julia Jupyter notebook and wxMaxima. Consider Julia using Pluto *reactive* notebooks. A Web Site document will explain the install details.

Grading:
1.) Graded homework assignments, including Python/Julia assignments 25%
2.) Final computer project worth 20%/15%. Grade option with final.
3.) Two "Hour" exams at 15% each, 30% total.
4.) Final exam worth 25%/30%. Grade option with final computer project

reorder planned

Topics	Text Sections
1. Introduction/Course Overview	1
2. Discrete-time signals and systems	2.0–2.9
3. The z-transform	3.0–3.4
4. Sampling of continuous-time signals and Discrete-time random signals	4.0–4.6 2.10, App. A 4.7–4.9
5. Transform analysis of linear time-invariant systems	5.0–5.7
6. Structures for discrete-time systems and Finite precision issues	6.0–6.9
7. The discrete Fourier transform	8.0–8.7, 8.9?
8. Computation of the discrete Fourier transform	9.0–9.6
9. Applications of the DFT	Portions of 10.0–10.6

Important: If you are a student with a disability and believe you will need accommodations for this class, it is your responsibility to register with Disability Services and provide them with documentation of your disability. They will work with you to determine what accommodations are appropriate for your situation. To avoid any delay, you should contact Disability Services as soon as possible. Please note that accommodations are not retroactive and disability accommodations cannot be provided until a Faculty Accommodation Letter has been given to me. Please contact Disability Services for more information at Main Hall, 3rd floor, Room 324, (719) 255-3354, or dservice@uccs.edu.

Learning Outcomes

The expected learning outcomes of this course are a more in-depth treatment of discrete-time signals and systems as first started in the UCCS course ECE2610 or similar from another university. As a discipline within electrical engineering this known as digital signal processing (DSP). Specifically the student will learn how to model discrete-time signals and systems in the time domain; extend the time domain modeling to the frequency domain using the discrete-time Fourier transform (DTFT); working signals and linear time invariant (LTI) systems using z-transform (ZT) techniques; sampling theory and multi-rate sampling theory as found in modern DSP; discrete-time random processes and modeling quantization/fixed-point arithmetic; DSP problem solving using time, frequency, and z-domains effectively; properties of LTI systems having minimum phase and linear phase; choice of various DSP implementation topologies; the value and power of the discrete Fourier transform (DFT) and its efficient implementation via fast Fourier transform (FFT) algorithms; Simulation of DSP algorithms and subsystems using Python with the Scipy stack and Julia which can also leverage the Scipy stack.

Slack

We will be using Slack *Channels* for group discussions and Slack *Direct Messaging* for audio and screen sharing. You will receive an invite to join the *DSP Comm Courses* Slack workspace. Download is under *Resources*.

Setting up Python via miniconda

To set up Python alone but with the advantage of virtual environments, miniconda works very well (on Windows install for Powershell). The helper document new_python_virtual_environment.pdf is provided. To run real-time DSP code in Jupyter Lab or vs code using pyaudio_helper. pyaudio_helper runs under Python 3.13.

vs code

Microsoft Visual Studio Code (vs code) is great for writing and debugging Python and Julia code source code files or Jupyter Lab notebooks. *Extensions* are available for Python, Julia, and Jupyter Lab, and many others. See the Julia_with_Python_Setup doc linked below for vs code setup info

Julia and Python, with Jupyter Lab together

Documents linked on the course Web Site provide details on how to install both Python and Julia along with various Julia and Python packages (Julia_with_Python_Setup.pdf). Under Jupyter Lab you create notebook documents (.ipynb) for either Python of Julia.
Note: Jupyter Lab is the perfect place to write code, document code, write text using markdown, import figures, and typeset math equations using LaTeX syntax. Julia makes a significant step in solving the *two language problem* (Python for rapid prototyping and C/C++ for execution speed). You will also see the *reactive* Julia Pluto notebook in action in class.

wxMaxima

Similar to Mathematica, as it is a computer algebra system (CAS), but open-source. Simpler to use than Python, but not as fast at numerical DSP type operations. I will be showing off the new DSP package in class.

Typst

Typst is a new writing tool, similar to LaTeX, but it has a very nice cloud-based interface with real-time preview and one click PDF export. Why? To allow you to create documents that are a hybrid of scanned hand written solutions, typed out problems with text and math, and code with plots. I will supply a homework template and examples and you can decide if you want to try it.