CS 3430: Scientific Computing with Python
Syllabus, Spring 2020

Professor:	Vladimir Kulyukin
Email:	vladimir.kulyukin@usu.edu
Office:	Old Main 402D
Office hours:	Thursday 4:30 - 6:00pm, by appointment
Class meeting time:	TR, 3:00 - 4:15pm
Class location:	Old Main 121.

Prerequisites: CS 1400, CS 1401, CALC I (or equivalent), Completion of University Studies Breadth of Life Sciences (BLS) course, Fulfillment of University Studies Quantitative Literacy (QL) requirement.

Course Description: Scientific Computing (SciComp) is the collection of algorithms, tools, and theories required to solve numerical problems in various areas of science, engineering, mathematics, and economics. Most of these algorithms, tools, and theories originated in mathematics (calculus, linear algebra, number theory, etc.) long before the advent of digital computers. These algorithms, tools, and theories used to be called (and are still called by some scientists today) Numerical Analysis. However, SciComp is a more common term today after digital computers have become mainstream and many methods of Numerical Analysis have been computerized. Python is a convenient tool for SciComp due to a large and ever increasing number of SciComp libraries avaiable in Python. We will use Python 3. Given the diversity of interests of the students in this class (computer science, engineering, physics, economics, statistics, psychology, etc.), you should expect some problems and applications to come from areas outside of your particular area of study or interest. Since SciComp as an area of Computer Science is deeply rooted in mathematics, you will see a fair amount of mathematics in this course. In the end, it is my hope that you will gain a clear understanding of the role of SciComp in modern science and technology and will become better scientists for it.

Objectives: The primary objective of this course is to introduce you to the basics of SciComp with Python. By the end of the class, you will have mastered or become familiar with the following concepts and topics: Gaussian-Jordan Reduction; Matrix Transposes, Inverses, Determinants; Cramer's Rule; LU-Decomposition; Linear Programming; Simplex Algorithm; Differentiation and Differentiation Engines; Newton-Raphson Algorithm; Central Divided Difference; Integration Approximation; Richardson Extrapolation; Romberg Integration; Edge Detection with Gradients; Line Detection with Hough Transform; Image Histograms; Digital Particle Image Velocimetry; Decision Trees; Huffman Encoding; Time Series Comparision; Levenstein Distance; Dynamic Time Warping; Public-Key Cryptography. By the end of the course, you will also have had programming experience with such Python libraries as numpy, scipy, scikit-learn, OpenCV, and sympy.

Note on Python: When Python was not one of the official languages of the CS undergraduate curriculum, I used to dedicate the first 3 weeks of this class for an intensive Python tutorial. *Since Python is now the language of CS1, the Python tutorial is no longer part of this course.* You're expected to know the basics of Python or learn them as the course unfolds.

Course Fees: $145 for Graduate Teaching Assistant.

Textbooks: There are no required texts for this class. Required reading materials will be given in class or, if there are no copyright issues, posted on Canvas. There are no required Python books either. I think that buying Python textbooks is becoming increasingly unnecessary, because there are so many great online resources. You can start with `www.python.org` and follow the tutorial links. Many Python texts are available as e-books on `http://library.usu.edu/`.

Python IDEs: There is no required Python IDE. So use your favorite IDE you're most comfortable with. There is a basic Python IDE called IDLE that you can download from `www.python.org`. There is PyCharm (`https://www.jetbrains.com/pycharm/`), which is much more sophisticated but requires a lot of space and dependencies. You can also use command line and a Python-friendly editor (e.g., Vi or Emacs). Ultimately, you'll submit your Py source (not IDE-specific projects), so it doesn't really matter which IDE you use as far homework submission is concerned.

Assignments and Exams: We'll have regular weekly/bi-weekly assignments and 3 in-class exams. The assignments will typically be assigned on Saturday and due the following Saturday by midnight. Most assignments will require coding. Exams will be mostly conceptual and may ask you to read/write short code segments. All assignments will be posted, submitted, and graded on Canvas. Summary lecture slides and source code will be available in Canvas to help you with your assignments and exams. *Keep in mind that since it is physically impossible for me to put all information discussed in class on lecture slides, slides will necessarily miss some information covered in lectures.* There is no substitute for paying attention and taking good notes. Be cognizant of due dates. No late submissions will be accepted unless there is a valid, documented reason. *Valid reasons are: military duty, university sponsored events, jury duty, serious illness/medical condition treated by a physician, death in the immediate family, and childbirth. All of these events must be supported by written documentation.* Grading of the programming assignments will be based on the following criteria: 1) Completion of program features as outlined in the assignment description; 2) Ability to handle error conditions (i.e. the program

does not crash); 3) Clear and concise code comments; 4) Code readability: meaningful variable, function, and class names, functional abstraction.

Grading: There will be a total of 100 points. The components of your final grade are given in the following table. Regular attendance sign-up sheets will be passed either at the beginning or end of each class period. *If the sign-up sheets show me that you regularly miss classes without a valid reason, I may not be able to answer your emails concerning homework assignments that rely on some material covered in specific lectures.*

Grading Component	Points
Homework	45
Attendance/Participation	5
Exam 1	10
Exam 2	10
Exam 3	30

Grade Ranges: I'll use the following grade ranges to assign final grades.

Grade	Percent
A	95 - 100
A-	88 - 94
B+	86 - 87
B	83 - 85
B-	80 - 82
C+	76 - 79
C	72 - 75
C-	70 - 71
D	60 - 69
F	0 - 59

Grade Argumentation: If you think that the grader or I have made a mistake grading your code/exam, let me know as soon as possible. You deserve credit if and when it is due. If you are concerned about your grade, do not ask what you can do to raise it one or two weeks before the finals week. That is way too late. In order for me to help you, we both need time to solve issues as they arise. Come and see me as soon as you perceive a problem. Obviously, you should have satisfied the course prerequisites. If you have not, you should complete them before taking this class. *Do not attempt to negotiate grades.* If you just need B- instead of a C+ or A instead of A- for whatever reasons, it won't happen. Students must be officially registered for this course. No assignments/exams of any kind will be graded for students whose names do not appear on the class list. *No make-up exams or assignments will be given.* Any conflict with an exam time must be resolved prior to the exam. Acceptable conflicts that will be accommodated are: military duty, university sponsored events, jury duty, serious illness/medical condition treated by a physician, death in the immediate family, and childbirth. All of these events must be supported by written documentation. All arrangements for making up work must be made with me personally.

Plagiarism: The basic rule about cheating is DO NOT DO IT! You are free to talk with each other about regular assignments in general terms. However, you must write your own solutions and/or computer code. No collaboration is allowed on any homework assignments or exams. Activities that violate the plagiarism policy include (but are not limited to): copying another person's work on a programming project, homework assignment, or exam; using any reference (e.g., a third-party software library, a blog post, a book) not authorized by the instructor on a programming project, homework assignment, or exam. You can always ask me before you do something. Additional examples of cheating include turning in another person's work as your own, allowing someone else to copy your work, doing work for another person, letting another person have access to your solution, using unapproved materials during a test, turning in duplicate or near-duplicate assignments, and working with another person to complete an assignment. *The penalty for violating the plagiarism policyis an F in the course.* University authorities will be informed of cheating incidents in a formal letter. The requirement to do your own work does NOT mean working together and turning in two solutions unless such collaboration is explicitly allowed by the instructor. You are also responsible for protecting your code. Take care to keep your code unavailable to other people. Occurrences of cheating will have serious consequences for all involved. The first incident will carry a penalty of the student being given negative points equal to the value of the assignment or test. A second occurrence is grounds for a failing grade in the course and possible University action. Files submitted for homework may be electronically compared to detect cheating.

ADA Compliances: If a student has a disability that will likely require some accommodation by the instructor, the student must contact the instructor and document the disability through the Disability Resource Center, preferably during the first week of the course. Any request for special considerations relating to attendance, pedagogy, taking of examinations, etc. must be discussed with and approved by the instructor. In cooperation with the Disability Resource Center, course materials can be provided in alternative formats, e.g., large print, audio, diskette, or Braille.

Incompleteness: The University policy will be adhered to for incompletes. This means that an incomplete cannot be given to prevent receipt of a bad grade. Under no circumstances can an incomplete be given for which a re-take of the class is required to make up the work. In such situations, a withdrawal or late withdrawal is required. If a student feels that an incomplete is appropriate, it is their responsibility to immediately discuss the matter with their instructor.

Add and Drop Dates: The university's academic calendar can be found at `http://www.usu.edu/registrar/`. Be cognizant of add/drop dates and plan accordingly. Also, note that a student who signs up after the beginning of classes is still responsible for any

homework assigned and due before that date.
Contact: My contact information is given at the beginning of this document. Emails sent and/or received on Saturdays/Sundays will not be answered until the following Monday. Saturdays are my R&D days. Sundays are for my family. So plan accordingly.
Fairness: CS 3430 is typically a large class, which means that my flexibility to meet your individual needs is limited. There will be no allowances made for making up work. The only guarantee I can give you is that everyone will be treated the same way and subject to the same standards, without bias. I will treat everyone equally, and will be consistent. This is the only way I can be fair to all of you.
Electronic Devices: *No smartphones are allowed in class.* They must be turned off or put in sleep mode for the duration of the class period. If you expect an emergency phone call (e.g., medical emergency in the immediate family, a childbirth, etc.), put your smartphone in vibration mode. A student found to be violating this policy may be subject to disciplinary action (e.g., negative points). You can use your laptops for note taking during lectures. *No picture taking or audio recording is allowed in class.*