Scientific Computation
Spring 2022
Project 3

Due: Friday March 25th 17:00 GMT

In addition to this pdf, there are three main files for this assignment: 1) *part1.py* and 2) *part2.py* are Python modules which you will complete and submit on Blackboard (see below for details) and 3) *report3.tex*, a template file for your report which will also be submitted on Blackboard. The discussion and figure(s) described below should be placed in this report. Additionally, there are two data files, *data1.npz* and *data2.npz*, which are needed for part 1.

Part 1

1. (4 pts) You have been provided with the data file *data1.npz* which contains global temperature data for an entire year. Code is provided in the function, *applyPCA*, to load the entire dataset and display one day of data. This code creates a 365 x 71 x 144 numpy array, T, where $T[i, j, k]$ corresponds to the daily average temperature on day i at latitude, $lat[j]$ and longitude, $lon[k]$, where lat and lon are arrays that are also loaded. Apply PCA to this temperature data and analyze the results. Your analysis should (1) identify 2 non-trivial trends related to how the temperatures vary with time and (2) focus on trends associated with what you consider to be the most important principal components. Code for the PCA computation should be placed in *applyPCA*. This function should return the two-dimensional transformed data matrix. Code for your analysis using PCA results should be placed in *analyzePCA*. Present your analysis along with any supporting figures generated in *analyzePCA* in your report.

2. (6 pts) It is common for weather and climate data to have missing or corrupted values. In this question, you will explore the application of a modified version of the recommender system presented in lecture 14 to estimate such missing values. Here, you will be working with temperature data for a single day stored as a matrix in *data2.npz*.

   (a) The function *rec1* estimates values for missing data in the input matrix, R. Elements in R equal to -1000 are assumed to be "missing data". The code attempts to find the a x p matrix, A, and p x b matrix, B such that the cost

c is minimized where

$$c = \sum \sum_{i,j \in K} (R_{ij} - \tilde{R}_{ij})^2 + \lambda \left[\sum_{i=1}^{a} \sum_{j=1}^{p} (A_{ij})^2 + \sum_{i=1}^{p} \sum_{j=1}^{b} (B_{ij})^2 \right],$$

and $\tilde{\mathrm{R}} = \mathrm{AB}$. This cost function includes a l_2-*regularization* term which penalizes cases where the elements of A and B have relatively large magnitudes, and the input parameter l corresponds to λ in the equation and sets the strength of this penalty. For this question, you should develop and apply a set of tests that critically assess how well this function estimates missing data. You should work with the given temperature data matrix (T), and your tests should set some number of elements in T to -1000. You should carefully consider how well the method works for a given p and if it improves as p is varied. Your analysis should include the following:

- A case where approximately 0.5% of the elements are removed from T. It is up to you to decide which elements to remove.
- Consideration of the influence of λ and results for $\lambda = 0$ and at least one non-zero value of λ.
- The case $p = 6$ and at least one other value of p.

Place the code for your analysis in the function, *analyzeRec1*, and add an explanation of your findings with supporting figures to your report. You do not need to assess the efficiency of the function. Note that iterations generally take longer to run as p increases and that it can take a large number of iterations to produce useful results.

(b) It has been observed that recommender systems for user-item ratings matrices can be (modestly) improved by including vectors to account for biases in user and item ratings. For example, if user i tends to rate items highly, the system would estimate a positive value for the i^{th} element in a user bias vector, β. Here, you will develop a modified version of *rec1* which implements such an approach. The modified cost function is,

$$c = \sum \sum_{i,j \in K} (R_{ij} - \beta_i - \tilde{R}_{ij})^2 + \lambda \left[\sum_{i=1}^{a} \sum_{j=1}^{p} (A_{ij})^2 + \sum_{i=1}^{p} \sum_{j=1}^{b} (B_{ij})^2 \right] + \eta \sum_{i=1}^{a} (\beta_i)^2,$$

where β is an a-element vector which must be determined along with A and B, and η is a regularization parameter similar to λ which will be provided as input. Add/modify code in *rec*2 so that it accounts for this modified cost function. You will need to modify the code which updates the elements of A and B. After all elements of the two matrices have been updated, add code to set β so that it minimizes the cost function (given the updated A and B). So, a new β should be generated each iteration. Your code should be efficient to the same degree as the provided code. Add a brief description of the additions/modifications you have made to your report. You do not need to present or discuss results generated by your code.

Part 2

1. (5 pts) The function *solvePDE* computes numerical solutions to a system of two nonlinear partial differential equations of the form,

$$\frac{\partial u}{\partial t} = f(u, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, v; \alpha),$$
$$\frac{\partial v}{\partial t} = g(u, v, \frac{\partial v}{\partial x}, \frac{\partial^2 v}{\partial x^2}; \alpha),$$

   where u and v are functions of space and time, $u(x,t)$, $v(x,t)$, and α is a model parameter specified as input. Initial conditions are generated within the code, and solutions are computed on a spatial grid with $0 \leq x < 1$ (see function documentation for more details).

   Analyze and compare results for simulations with $\alpha = 3$ and $\alpha = 0.1$. Typically simulations contain an initial transient as the system responds to the initial conditions followed by a relatively settled dynamical state. Discard the transient in your analysis, and focus on fluctuations of u in space and time (you may vary Nt and T as needed. Also consider the global qualitative dynamics (e.g. the system is steady (no time-dependence), simple sinusoidal oscillations in time, ...). Carefully analyze if/to what degree chaotic dynamics are present. Qualitative observations should be supported by quantitative results and well-designed figures. Add the code used in your analysis to the function, *analyzePDE*, and add the discussion of your findings along with supporting figures to your report.

2. (5 pts) In the previous question first derivatives were computed with discrete Fourier transforms, but this would no longer be possible if we imposed the boundary conditions, $\frac{\partial u}{\partial x} = 0$ at $x = 0$ and $x = 1$. Finite-difference methods are then an option that can be considered. A 4th-order method has been provided in the function *fd4*. You will implement an implicit finite-difference method and critically compare it to *fd4*. The derivative, $\frac{du}{dx} = u'$ should be computed on the grid, $x_j = jh$, $j = 0, 1, ..., n-1$. Let $h = 1/(n-1)$. Then, the implicit scheme is defined as follows. For $j = 3, 4, ..., n-4$:

$$\alpha u'_{j-1} + u'_j + \alpha u'_{j+1} = \frac{1}{h}\left[\frac{c}{6}(u_{j+3} - u_{j-3}) + \frac{b}{4}(u_{j+2} - u_{j-2}) + \frac{a}{2}(u_{j+1} - u_{j-1})\right],$$

   and the coefficients are provided in the function *implicitFD*. For $j = 1, 2$:

$$u'_j + \alpha u'_{j+1} = \frac{1}{h}\left[a u_j + b u_{j+1} + c u_{j+2} + d u_{j+3}\right],$$
$$\alpha = 3, a = -17/6, b = 3/2, c = 3/2, d = -1/6,$$

   and for $j = n-3, n-2$:

$$u'_j + \alpha u'_{j-1} = -\frac{1}{h}\left[a u_j + b u_{j-1} + c u_{j-2} + d u_{j-3}\right],$$
$$\alpha = 3, a = -17/6, b = 3/2, c = 3/2, d = -1/6.$$

   At the boundaries, $u'_0 = u'_{n-1} = 0$.

(a) Complete *implicitFD* so that it efficiently implements this scheme to compute first derivatives of the m columns of a input two-dimensional n x m array provided as input. Each column of the array should contain data, $u(x_j),\ j = 0, 1, 2, ..., n-1$, which should be differentiated.

(b) Design a set of computational tests that support a critical comparison of the methods implemented in *implicitFD* and *fd4*. Assume that m is comparable to n, and assess the effectiveness of these methods for multiscale problems. Ultimately, your analysis should reflect a clear understanding of the cost and accuracy of the method and your implementation. Add your analysis and accompanying figures to your pdf. Place the code used for the analysis in *analyzeFD*.

Note: You may import and use any modules we have used during the term. Please do not use any other modules without permission.

Further guidance

- You should submit both your completed python file and a pdf containing your discussion and figure(s). You are not required to use the provided latex template, any well-organized pdf is fine. To submit your assignment, go to the module Blackboard page and click on"Project 3". There will be an option to attach your files to your submission. (these should be named *project*3*.py* and *report*3*.pdf*). After attaching the files, submit your assignment, and include the message, "This is my own work unless indicated otherwise." to confirm the work as your own.

- Please do not modify the input/output of the provided functions without permission. This does not mean you are required to use the default values provided for some input variables. You may create additional functions as needed, and you may use any code that I have provided during the term.

- Marking will be based on the correctness of your work, the soundness of your analysis, and the degree to which your submission reflects a good understanding of the material covered up to the release of this assignment. You should aim to keep the pdf version of your report to less than 5 pages of text with 20 or less figures, however you will not be penalized if you exceed this guidance.

- Open-ended questions require sensible time-management on your part. Do not spend so much time on this assignment that it interferes substantially with your other modules. If you are concerned that your approach to the assignment may require an excessive amount of time, please get in touch with the instructor.

- Questions on the assignment should be asked in private settings. This can be a "private" question on Ed (which is distinct from "anonymous"), using the "Chat" on Teams during a Q&A session, or by arrangement with the instructor.

- Please regularly backup your work. For example, you could keep an updated copy of your files on OneDrive.

- In order to assign partial credit, we need to understand what your code is doing, so please add comments to the code to help us.

- You have been asked to submit code in Python functions, but it may be helpful to initially develop code outside of functions so that you can easily check the values of variables in a Python terminal.