Project 2 Instructions, CSE 6730 / CX 4230, Spring 2020

For Project 2, you are to work in small teams to develop at least one tutorial on any subject related to the topics covered in Part 2 of this course. (Information on teaming and scoping of your work appear in the guidelines below.) Since Project 2 is due shortly after we will have completed these topics, you may need to "read ahead" on your own. That should work out fine since the class lectures are likely to cover theoretical or conceptual ideas that complement the "hands-on" work you need to do in your project.

Milestones and grade weights. There are three milestones for this project:

1. Proposal 2 ("soft" deadline of Mar 9, "hard" deadline of Mar 16): This **ungraded checkpoint** exists for you to report your final teaming arrangements (see below) and summarize your initial discussions about possible directions.
2. Checkpoint 2 (due Mar 30, automatically extended to Apr 6 and no late penalties up until Apr 10): In this **graded checkpoint**, you need to show that you have made reasonable progress toward the completion of the project. If you've hit any snags, you need to describe them at this point. This checkpoint exists mostly to force you to start early and will be worth 5% of your final course grade.
3. Project 2 (due Apr 21, automatically extended to Apr 28 -- final deadline!): Your project is due on this date and will be worth ~~17% (project) + 3% (demo) =~~ 20% of your final grade. You will demo your work in-class.

Teaming. You may choose your own teams (2-3 people; please refer to the syllabus) and must have them in place by the Proposal deadline (see **Milestones**).

Inevitably, conflicts within the team may arise. Please communicate with one another, and if needed, work with the teaching staff to resolve problems early on, ideally, well before Checkpoint 2. After that time, we will consider all teams "final" and you are stuck with one another until the end.

Guidelines. We want you to imagine that you are preparing an in-depth tutorial on some subject related to the topics of Part 2 of this course. The reason we are saying "tutorial" is the philosophy that you learn something better when you have to teach it to someone else.

Your project should have three components: exposition, code, and results. That is, it should explain a subject in a tutorial style, with code examples that can be executed to illustrate the subject. These should present these together in an "executable report," which is designed to be interactive. There are several tools and platforms that support the creating such reports, including Jupyter notebooks, Matlab LiveScripts, and RMarkdown. (If you plan to use something else, check-in with the teaching staff early on to make sure it meets our expectations.)

The parts of your project should be "coherent." That is, we are trying to discourage each team member doing a smaller project that has nothing to do with those of the rest of the team. (To repeat, it is not acceptable for a team of three to do three separate and unrelated individual projects and simply "staple them" together.)

We've prepared one example based on the Susceptible-Infected-Recovered disease model explained in an early class ("Infection, 3-ways"). It has five parts and covers three conceptual models. However, these parts are "coherent" in that they are explained as three ways of approaching the modeling of the same phenomenon. It also happens to be written in a literal tutorial style with "exercises" and sample solutions, but you do not need to take the same approach. This demo is available on the GT GitHub:

https://github.gatech.edu/cse6730modsim/sir-demo

Graduate vs. undergraduate teams. (*This information differs slightly from the syllabus*.) We expect all graduate teams to pick one system or phenomenon and model and simulate it in **k** ways, where **k** is the number of members in the team. For undergraduate teams, you can do **max(1, k-1)** ways. Please ask Prof. Vuduc if you really want to do a "mixed" graduate-undergraduate team.

Where to find inspiration. Here are some reference materials you can consult to find projects. The SIR infection model from above comes from O'Leary (2009) in the list below. Except where indicated, these should be available electronically either by direct link (e.g., Smith? book) or through the GT Library (use the GT Library proxy bookmarklet or search for the book via library.gatech.edu).

- Sayama's book (2015) -- lots of examples, exercises, and mini-projects that you can go deeper on or extend
- *Scientific computing with case studies*, by Dianne O'Leary (2009), SIAM.
- *Mathematical modeling of zombies*, by Robert Smith? (2014). (Yes, the question mark at the end of Smith?'s name is for reals.) You might find inspiration from some of the mathematical techniques for conceptual modeling that Smith? applies to consider various aspects of how zombies might behave "in the real world."
- *The structure and dynamics of networks*, by Mark Newman, Duncan J. Watts, and Albert-László Barabási (2006).
- *Mathematical models: mechanical vibrations, population dynamics, and traffic flow*, by Richard Haberman (1998), SIAM.
- *Neural ordinary differential equations* (NeurIPS'18 best paper). Some of you have asked about how to connect data or ML to the topics of the first part of this course. Here is one

way (lots of related cited papers contained therein) that makes one perhaps-unexpected connection!
- *Tutorial on the dynamics of biological neurons* (spiking models), by Jack Terwilliger (2018).
- *A primer on mathematical models in biology*, by Segel & Edelstein-Keshet (2013).
- *Cellular automata tutorial*, by Jarkko Kari (2008?). A good summary of key technical results related to CA models.
- *The nature of code*, by Daniel Shiffman. Section 7.9 has an interesting list of application ideas for cellular automata (as does Sayama's book).
- *Traffic and related self-driven many particle systems*, by Dirk Helbing. [Link via GT Library Proxy]

Evaluation. Based on the description above, we will evaluate your submission along these dimensions:

- Exposition -- How well does the tutorial explain the problem(s) and solution technique(s)? Is it concise, but also clear, readable, and precise? Does a prospective reader take away insight?
- Code -- Is it clear and readable? Does the implementation pay attention to efficiency considerations?
- Results -- Does the notebook contain simulation examples to help illustrate the problems and techniques? Does it use visualization appropriately?

There is no hard "weighting" among these categories, but do try to pay equal attention to all criteria.

Peer assessment. The entire team receives the same base grade. However, at the final submission, we may ask each of you to submit your own "peer assessment" in which you evaluate the contributions of your teammates. In the event of discrepancies, a small additional component of the grade may be based on teammate evaluations. The purpose of this assessment is to create an incentive for you to find ways to work well together and contribute equally to the overall product.

Submission instructions. We will post more detailed guidelines for each submission as we get closer to each milestone's deadline and update the document here.

For all milestones: You will use the GT GitHub instance (github.gatech.edu) to submit some of your milestones. Start by creating an initial repository there to hold the materials for you and your teammates. You can make this private if you wish, but then you will need to share it with all of the teaching staff so we can grade it (more detailed instructions to come).

1st milestone: Project 2 Proposal ("Soft" deadline of Mon Mar 9, "hard" deadline of Mar 16) (required to receive a grade but otherwise worth "0" points)

Instructions and submission link:
https://gatech.instructure.com/courses/103474/assignments/412548

Project 2 Checkpoint (Due Mon Mar 30 -- automatic extension until Mon Apr 6)

Submission: A single PDF document.

There are three components for the submission for the checkpoint:

1. A clear and detailed description of your project (max 2 pages, excluding references).

At this stage, you should have a clearer idea of the project and the details that you expect would be part of the final submission.

Some things to include:
a) An abstract summarizing the system and the goals of the project
b) Description of the system being studied
c) Expected approaches that would be used to study the system
d) Platform(s) of development
e) Literature review (if possible)

2. An update of the current state of the project and initial results, if any (max 2 pages)

Some things to include:
1. A "show of progress" via some working code, analysis, or initial modeling attempts
2. If there have been any major changes in direction or "course-corrections" since your original proposal, you can describe them here.
3. Division of labor: How will you divide up the remaining work among your team? In particular, we will be looking to see that you've given thought to how to ensure your project justifies a multi-person (2 or 3, depending on team size) effort.

3. A Git repository for the final submission

This component of the checkpoint would ensure that you have set up a git repository on the Georgia Tech GitHub repository. This is where you will be sharing your final project implementation with the instructors.

Things we will evaluate:
a) A GT GitHub link to the project repository is included in the report
b) The repository has correct permissions, that is shared with all the teammates, instructors and TA. **(Usernames: rvuduc3, qyang61, ekim79, pyu73, tfuruya6)**
c) At least one file, e.g., README or the Project 2 Checkpoint report is in the repository.

NOTE: Although we recommend it, you don't have to use git for your development. You just have to share it with us on Git.

In case you have not interacted with GT GitHub or Git in general. Following is a brief tutorial for the setting up a GT git repository and pushing (read, saving) your local copy on the GitHub server.

1. Log on to GT GitHub (https://github.gatech.edu/) using your GT credentials.
2. Name your new repository (or "repo").
3. Choose the privacy setting for the repo. If you choose to make it public, anyone with a link and a GT account would be able to access your repo and its contents. This can be updated later.
4. Follow the instruction on the next page to set up a local repo or for a Linux (terminal) based system use the following.
   For other systems follow this link (https://hackmd.io/s/B1Tgv0bNE):
5. **NOTE: For Windows users, download Git for windows and install it. You can type the following commands into the git for windows terminal. Alternately you can download and use GitHub desktop if you want a UI like repo management instead of all these commands.**
   a. `cd <project_folder>` *# go to you project folder*
   b. `vim README.md` *# optional: create a readme file for your repo*
   c. `git init` *# initialize local git repo*
   d. `git add README.md` *# optional: add file to local git*
   e. `git commit -m "first commit"` *# describe the state of the repo.*
      # examples could be:
      # "added feature X", "completed part Y"
   f. `git remote add origin https://github.gatech.edu/<gt_username>/<repo_name>.git`
   g. `git push -u origin master` *# upload the current repo state to GT GitHub*
6. Once the repos are set up, work as usual in your project directory. When it is time to push changes:

   a. `git add --a`   // to add all the new files to git.
      ProTip: use `git status` to view the current state of all files added to git or updated since the last commit.
   b. `git commit -m "second commit"`
   c. `git push`
7. Collaborators can be added through the settings section on your repo.
   `https://github.gatech.edu/<gt_username>/<repo_name>/settings/collaboration`
   ProTip: You can search and add people using GT usernames.

You are always welcome to visit the TA office hours, in case you find something unclear.

Project 2 - Final report and in-class demo (due Tu Apr 21, including in-class demo; automatic extension until Wed Apr 22 @ 11:59 pm ET)

Here is what you need to do for your final Project 2 submission. There are two parts.

Part 1. For your final tutorial project submission, each team will upload a single PDF into the assignment on Canvas named, [Project 2 Submission (link TBD)]. This PDF should contain the following information.

- A single cover page, which shows the title of your tutorial, your team number, a list of your team's members, and a link to your GitHub repo so we can verify what you implemented. (Presumably, this repo is the same as what you indicated in your Project 2 Checkpoint.)
- The tutorial itself
- After the tutorial contents, provide a brief description of how you split the work among the members of the team.

You do not need to put a copy of this PDF in your repository, although you are welcome to do so.

Part 2. Separate from the above submission, which only needs to be done once for the team, each of you must **individually** submit a [Teammate Assessment for Project 2 (link TBD)] assignment on Canvas. It is simply a text entry box in which you will enter the following information:

- Your team number and list of teammates
- Assign a letter grade (A through F) to yourself and each of your teammates based on your assessment of their contributions to the project. In each case, explain your grading.

(Note: Do not interpret this grade as one for your project; instead, think of it only as a grade based on the amount of effort expended to complete the project.)

- We will not share your teammate assessment with anyone else on your team. It will only be used for us to identify issues and try to resolve them after the fact.