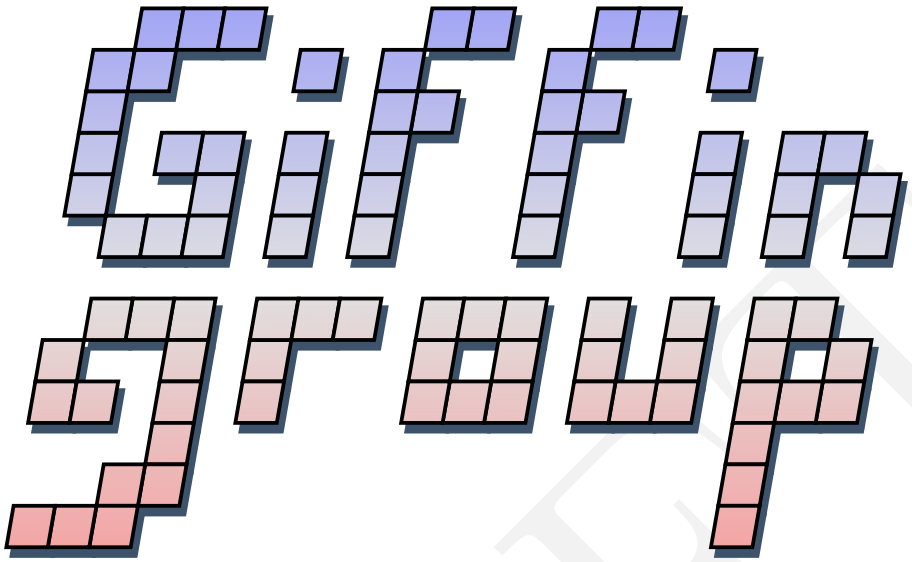

GRADUATE HANDBOOK

Brian Doran Giffin, Ph.D.
Assistant Professor
Oklahoma State University

August 20, 2025

Abstract

This document is intended to equip new and continuing graduate students with the necessary tools to be successful in their academic pursuits and beyond. The contents of this handbook are divided into different chapters focusing on: policies and procedures specific to the research group; expectations for graduate students to conduct ethical, high-quality research; guidance toward developing professional competencies spanning a variety of topics relevant to an academic environment; and additional resources.

Contents

1 Introduction

Welcome to the Giffin Group! Whether you have a strong passion for *computational mechanics*, or merely a burgeoning interest in the subject, it is my personal mission to equip you with the knowledge, skills, and experience necessary to thrive in this exciting and diverse field of study. In spite of the fact that our group is situated within an engineering department, you will discover that computational mechanics finds relevance in a much wider range of applications, and requires a much broader foundation of knowledge. Successful research which advances the current state of knowledge in this area necessitates expertise encompassing the complete spectrum of STEM disciplines: science (mechanics and material science), technology (computer science), engineering (civil/mechanical/aerospace), and mathematics (differential equations, etc.). While the need to develop competencies in all of these areas can feel overwhelming at times, the rewards are well worth the investment of effort. A wealth of engaging career options are available for computational mechanicians within academia, industry and national laboratories, with ample opportunities to work on challenging problems or transition into adjacent fields.

As an advisor, my personal academic philosophy can be summarized through a few simple (if a bit cliché) aphorisms:

- *You are your own greatest asset;* Invest in yourself, do not cheat yourself out of valuable opportunities for learning and growth, have patience, and eventually your investments will pay dividends.
- *Our legacy is not determined by what we do or what we create, but by the people that we personally impact.*
- *What you get out of your academic experience is proportional to what you put in.*

I encourage you to develop your own independent sense of motivation and a set of values that will help you persevere through your academic journey. While the experience of pursuing a graduate degree can feel isolating and disconnected at times, remember that you are part of a community that is genuinely invested in your success. Whenever you can, seek to contribute and give back to that community in ways that align with your personal values and passions.

While this document is primarily geared towards new students joining the research group, the hope is that it may also serve as a centralized location for helpful information that will aid you throughout your academic travails. Chapter 2 covers general policies and procedures governing research group operations. Chapter 3 discusses the expectations for successful graduate study and research. Chapter 4 offers personal advice on how to develop competencies in a variety of relevant research activities, whereas chapter 5 highlights several helpful external resources. This is a living document and is subject to ongoing revision; suggestions and contributions are welcome!

2 Policies and Procedures

This section covers general policies and procedures governing research group operations, including: organized procedures for planning research activities; meeting frequency and format; practices for managing data and software generated within the research group; and institutional requirements for the responsible conduct of research.

2.1 Milestones & Development

The key to academic success lies in the following step-wise approach:

1.) Make a plan

   a.) Make sub-plans

2.) Set concrete, achievable goals to mark progress toward completion of short- and long-term plans

3.) Score measurable wins at regular intervals to maintain morale and momentum

It is very easy to fall into the trap of feeling stuck or stalled. The remedy is to continuously remind yourself of your progress, and to feel good about smaller daily accomplishments. Provided you've set up a plan for yourself, you'll be sure that you're on the right track in working towards achieving your larger goals. To this end, it is vital for students to participate in the planning process, and to reflect on and share their progress on a regular basis. The following sections outline the proposed system adopted within the research group to instill these principles and guide students toward academic success.

2.1.1 Semester Planning and Review

Semester Goals

Students are expected to participate in planning research activities and setting personal goals at the beginning of each semester. Goals should align with the overall objectives of the student's chosen research project, and should be **SMART**: **S**pecific, **M**easurable, **A**chievable, **R**elevant, and **T**ime-bound (Doran, 1981). Set goals should include (but are not limited to) the following:

- Planned coursework
- Service activities: TA appointment; mentorship; professional organizations
- Milestones for degree completion: formation of a thesis/dissertation committee; drafting and submitting a plan of study; scheduling and participation in qualifying exams/defense; graduation filing

- Progress on thesis/dissertation
- Professional training and skill development
- Research outcomes: literature survey, infrastructure/tools/methods, results, publications, presentations, etc.
- Targeted opportunities: fellowships/scholarships, internships, travel awards, conferences, etc.
- Required resources and support: equipment/supplies; time commitment from advisor;

Students should draft a brief document outlining their goals, and share this document with their advisor for further review and discussion during an in-person meeting. Goals should be mutually agreed upon between the student and their advisor to ensure that they are sufficiently ambitious while also setting realistic expectations.

Semester Plan

Once a set of goals have been agreed upon, students should plan out how they will accomplish their stated goals. This should entail:

- A breakdown of each goal into discrete, ordered tasks.
- A timeline for the expected completion status of tasks for each week of the semester.

Students should share their proposed semester plan for formal review by their advisor, and schedule a meeting to discuss potential revisions/adjustments. This plan will form the basis for the primary points of discussion during weekly one-on-one meetings.

Semester Review

At the end of each semester, students will participate in a review process to summarize and document their progress towards completing their degree, and assess whether they successfully accomplished their agreed upon goals. If certain goals were not met, discussion should focus on the reasons why to adjust and inform the plans for the subsequent semester.

2.1.2 Annual Assessment

At the end of each academic calendar year (close to the start of the Fall semester), students must participate in an annual assessment to formally document their progress and development. This is an institutional requirement, with the outcomes of this assessment held on file within the student's formal record at the university.

The elements of the annual assessment process include:

- Completion of an annual assessment document, including the following details:
  - Basic information about the student: name, degree objective, date entered program, GPA.

  - The student's self report/assessment, including: completed coursework; completed degree requirements; research outcomes (publications, presentations, etc.); GRA/GTA service; fellowships; honors; extenuating circumstances; and other accomplishments.
  - The advisor's assessment of the student: areas of strength; opportunities for growth; setting milestones for the next year; estimation of graduation date based on degree progress.
  - Attachment of the student's current CV.
  - Additional commentary from the student and advisor.
- An in-person meeting between the student and their advisor to discuss details presented in the assessment document.

In advance of the review meeting, a template Annual Assessment document will be shared with the student for them to complete. Following the meeting, additional commentary should be added to the document, as needed. Once complete, both the student and the advisor must sign the document prior to the assessment being formally archived.

2.2 Meetings & Reporting

To maintain steady progress towards degree completion, students are expected to participate in regular one-on-one meetings with their advisor, as well as group meetings within the broader context of the research group.

2.2.1 Individual Weekly Meetings

Students should schedule a recurring weekly one-on-one meeting with their advisor to discuss progress on assigned research tasks. In preparation for each meeting, students should share the planned agenda for the meeting with their advisor to ensure effective use of time. For the meeting itself, students should draft a brief set of slides outlining:

- The conclusions reached and tasks set out from the previous meeting.
- The list of tasks worked on over the current week, and their completion status.
- Demonstration of progress towards completing the listed tasks (e.g. data, results, and figures) for review and discussion.
- Specific questions or points for in-depth discussion to solicit guidance on next steps.
- A proposed list of tasks for the following week, with potential alternatives.

During the meeting, it is the student's responsibility to take notes regarding the outcomes of the discussion, including direction received on follow-up tasks ("action items") to be completed in advance of the next meeting. The notes from each meeting should be shared with the advisor shortly afterward for review and revision. Students may take notes using whatever tools or methods they prefer (including audio recording), but the finalized meeting notes should be shared in a centralized document with a bulleted list summarizing the action items covered during the meeting. One institutionally sponsored option for automated note-taking is the Zoom AI Companion, enabling transcription and summary features supporting a variety of alternative summary templates.

2.2.2 Group Meetings

The advisor of the research group will schedule regular group meetings (on either a monthly or bi-weekly basis) involving all members of the research team. The purpose of this meeting is to communicate shared information and resources relevant to all group members, and to provide students with the opportunity to share progress on their individual research projects with the rest of the group. Group activities may include: workshops to develop writing, programming, and general research skillsets; technical seminars delivered by the advisor or individual students on selected topics of mutual interest; discussion of selected journal articles collectively reviewed within the group; and open discussions regarding improvements to research infrastructure and group operations. The meeting will conclude by establishing a plan for the following meeting's activities, and compiling a list of group action items. Suggestions for group meeting activities may be submitted in advance of each meeting for consideration by the group.

2.3 Data Management and Sharing

All research products (including data, documents, software, etc.) generated in the course of conducting university-sponsored research projects must be made publicly accessible (or at least accessible within the research group), and be appropriately organized and documented with accompanying meta-data explaining the nature of each research product and how it should be interpreted. To this end, research products should be archived in a manner which reflect FAIR (Findable, Accessible, Interoperable, Reusable) principles (Wilkinson et al., 2016).

Within the research group, the following tools/infrastructure should be used to share information:

- **Software projects**: All software tools should be subject to revision control, and hosted in a GitHub repository permitting shared access within the research group.
- **Manuscripts**: All documents intended for eventual publication (including journal articles, theses, and dissertations) should be shared via Overleaf.
- **Other**: All other persistent research products should be shared in a common OneDrive folder made accessible to all members of the research group.

Some exceptions may be granted for temporary files, or very large datasets which exceed the OneDrive storage limits (in such cases, data must be backed up on the student's assigned computer workstation).

2.4 Software Development Practices

Industry standard software development practices should be exercised within the research group to ensure that all internally managed software tools are: reliable, well-documented, clearly and cleanly organized, extensible, performant, and distributable across multiple target platforms. Such practices encompass (but are not limited to) the following:

- **Version Control**: Students are expected to utilize git to manage ongoing revisions to software. All software projects subject to revision control should be managed

in GitHub. When working on a new (possibly experimental) feature which may diverge from the main repository, it is standard practice to create a branch on which any anticipated changes will be made. Once the changes are complete and the software satisfies basic requirements regarding code quality, the created branch should be merged into the main branch.

- **Issue Tracking**: Implementation of self-contained features, improvements, or fixes to address software bugs should be appropriately documented and tracked using GitHub Issues. Larger development efforts encompassing multiple issues should be organized as Milestones.

- **Code Reviews**: When contributing major revisions or modifications to a shared software project for which you are not the primary owner, it is standard practice to have another developer review your code before these changes are accepted and merged into the main branch. The practice of soliciting a code review is usually formalized through a Pull Request, during which the reviewer offers specific feedback on things that the requester should address prior to the merge. If you are the primary author of a given software project, you do not need to adopt this practice, but you should consider implementing such practices for larger/shared projects managed between multiple contributors.

- **Cross-platform Support**: To ensure that software projects can be more widely adopted and distributed both within and outside of the research group, software projects should be designed to support compilation and execution on a range of alternative target platforms: from personal computers running different operating systems (MacOS, Windows, Linux), to supercomputing facilities with MPI and GPU-enabled hardware. At a minimum, cross-platform compilation tools such as CMake should be used to design the build system for projects written in C++.

- **Dependency Management**: Large compiled scientific software projects which rely on third party libraries with deep dependency trees can become a nightmare to manage. Some of these concerns can be ameliorated through the use of standardized package managers such as Homebrew (for MacOS) or apt (for Ubuntu). These tools work well on systems for which you have admin privilidges. However, these tools cannot be used to install system-wide packages on shared supercomputing platforms. In such cases, Spack offers a viable alternative. Nonetheless, it is generally a good idea to minimize the number of third party library dependencies for a given project to avoid potential stability and portability issues.

- **Verification and Unit Testing**: To ensure that software projects maintain previous/existing functionality following subsequent code changes, it is essential to adopt and implement software quality assurance practices, includinag verification and unit testing strategies. Such practices implement simple programs which verify that the software is behaving as indended. Formalized unit test frameworks such as GoogleTest can be integrated with CMake using BLT. Continuous Integration (CI) features available in GitHub allow for unit tests can run automatically using GitHub Actions.

- **Documentation**: Software tools produced by the research group (especially ones intended for public consumption) should be appropriately documented. At a minimum, this entails providing a descriptive README.md document within the project

repository (including instructions on how to build and run the project), but should be supplemented by additional content as needed. This may include:

  - *Examples* demonstrating how to use the software in a representative collection of common use-cases.
  - *User documentation* explaining the problem that your software project is trying to provide a solution for, and a more detailed explanation of how to use your tool. Web-based documentation can generated using Sphinx and distributed via GitHub Pages.
  - *Auto-generated API specifications* produced using Doxygen for C++, or Sphinx for Python code.
  - *Code comments* providing a narrative description of what each section of code is doing for ease of navigation by other developers (*including your future self*!)

- **Software Design Principles**: The software architecture for a given project should be guided by a set of agreed upon design principles, chosen to appropriately prioritize target design objectives (such as computational performance or extensibility/modularity). Such principles are frequently standardized through different philosophies (e.g. SOLID or GRASP).

- **Code Style and Readability**: While there are many alternative ways to write code that looks "pretty," this is an inherently subjective assessment. It is up to the individual developer or development team to establish consistency with respect to coding style. Nonetheless, there are some common industry standards such as the Google Style Guide (for C++) or PEP 8 (for Python) which may serve as a helpful baseline. Adopting good coding style ultimately contributes to improved readability. This encompasses details such as how to name variables (with longer, more descriptive names) and when/how to include descriptive comments. Enforcement of a consistent coding style can be facilitated through automated style checking tools such as Cppcheck or YAPF, both of which can be integrated with CMake using BLT and implemented in an automated CI workflow.

- **Licencing**: All publicly distributed software should have an accompanying license associated placing restrictions on how the software can be used, redistributed, or modified. GitHub provides additional guidance on licensing a repository and how to choose a license for a given project.

- **Citability**: Zenodo provides integration with GitHub to assign a DOI to specified releases of a given reponsitory. Additionally, GitHub provides a formal CITATION file specification for ease of formatting citations.

While it may not always be practical to implement all of the above practices, students who have an active interest in purusing a career involving the development of scientific software should attempt to learn about and incorporate such practices within their research projects to the greatest extent possible, while simultaneously balancing such activities with the primary objective of achieving research outcomes.

2.5 Research Integrity & Compliance

All students are *required* to complete assigned trainings regarding the responsible conduct of research, and should adhere to all supplementary policies regarding academic honesty and integrity as set out by the university. Further expectations particular to the research group are discussed in the subsequent sections.

2.5.1 Use of AI

Students are expected to conduct research tasks independently (including writing, programming, and data generation) while exercising judicious use of AI to assist in these processes. Blind utilization of AI to generate research content (particularly if such content is intended for publication) is unacceptable. Common tools exist (and will be periodically utilized within the research group) to check whether content has been AI-generated. Students identified as having utilized AI tools in an irresponsible manner will be confronted, with remedial actions implemented as needed.

Nonetheless, it is expected that students will utilize AI tools to assist with the generation of research content. Appropriate uses of AI within a research context include:

- **Brainstorming**: AI can be a great tool for coming up with innovative ideas that you yourself might not have thought of on your own. It is acceptable (even encouraged) to use AI as a sounding board. Ideally, this should be a back-and-forth exchange where you have a general idea of how you want, involving multiple iterations with an AI tool to further develop the concept you seek to execute on. This is expecially helpful in drafting an outline for planned work (including writing outlines and schedules), or coming up with solutions to specific research problems.

- **Literature Review**: When exploring a new research area, it's helpful to quickly gain a 10,000 ft view of the broader field of research (ideally without having to conduct an extensive literature survey). Review articles can sometimes provide the necessary context to better understand the current state of research in a particular topic, but such articles may not be up-to-date or cover emerging subjects. AI can be a useful tool in such situations to sift through the literature and identify relevant publications that can provide a point of entry to better understand the subject matter. However, there is no replacement for actually reading others' published works. The only way to become a competent technical writer is to read journal articles in detail. While not all articles warrant a thorough reading, students should nonetheless reserve time in their schedules to conduct "deep dives" on topics of direct important to their research. In summary: AI can be a useful tool to find relevant articles worth fully reading in detail, but you should still read these identified articles to fully understand their content and be able to explain it when asked in the context of a research meeting.

- **Data Analysis**: If you are trying to discren patterns from data to gain insights regarding what might be going on, or how the data could be represented more compactly in terms of a reduced order model, this is one of the ideal applications for AI.

- **Editing and Reviewing**: If you're drafting a manuscript intended for submission to a journal and want to get feedback on structure, organization, logical flow, clar-

ity, brevity, or language/grammar/syntax checking, AI tools can provide helpful suggestions for improvements while preserving your original written content and narrative style. If you intend to share a document for review by other members of the research group, you should adopt a practice of conducting a preliminary editorial review (potentially guided by AI tools) to screen for language-related issues in your writing.

- **Boilerplate Code**: If you need a short snippet of code to perform a simple task that would otherwise entail extensive time spent exhaustively learning, using AI tools to generating such code may be acceptable. Nonetheless, you should be transparent about the use of such tools in this context, and it may introduce questions regarding authorship and licensure of software.

The university library provides additional guidance on the responsible use of AI in academic research and writing, including how to cite the use of AI tools in your published work.

Regardless of how you use AI tools, you should always be careful about how you interact with these tools, and you should understand their inherent limitations. In particular, certain AI tools suffer from "sycophantic" behavior, which can lead to confirmation bias in your use of the tool. There is ultimately no substitute for your own judgement, and you should interpret content produced by AI tools with healthy skepticism, seeking independent verification as needed.

Remember: the intent of a graduate degree program is to gain knowledge and skills that will serve you in your career. While it is tempting to use AI tools to accelerate your progress towards completing your degree, bear in mind that *what you get out of your academic experience is proportional to what you put in*: while the process of learning to write/program can be arduous, it is better to develop these skills now while you have the opportunity to grow in these areas early in your career.

If you do utilize AI in preparing any research products, you must be transparent about your use of such tools, and explain how you used AI in your work. Be prepared to justify and defend your use of these tools, and how your utilization of AI adheres to the standards of conduct outlined above. Be aware that many peer-reviewed journal enforce strict standards on what constitutes appropriate use of AI, and you should inform yourself of these requirements in advance of drafting and submitting a manuscript for review.

Lastly, it is worth remembering that AI tools do not always guarantee data privacy and security. It is therefore important to emphasize the importance of proper handling of unpublished research data or intellectual property, particularly if such materials are subject to additional regulations or are sensitive in nature. Exercise extreme caution when selecting a particular AI tool for use in conducting research tasks, and in deciding what information is shared with these tools. Avoid the use of untrusted AI tools. When in doubt, seek clarification within the research group to determine which tools are appropriate for use in a given context.

2.5.2 Authorship and Attribution

Students must comply with common standards for authorship within the academic community. It is unacceptable reproduce the data, code, figures/images, or written work of others without proper attribution (citation) or permission from the originating author(s).

When in doubt about whether certain information can be utilized in your work (particularly if said work is intended to be distributed to a public audience), seek the opinion of the author. Certain data sources and software tools that have been published with a DOI permit their re-use according to an accompanying license agreement. It is not acceptable to directly utilize figures or graphics taken from others' journal publications without the express permission of the author.

3 Expectations

This chapter outlines the expectations that all students participating in the research group should adhere to. Failure to meet outlined expectations will warrant discussion and remedial action, pending discussion of extenuating circumstances.

3.1 Professional Conduct

All members of the research group are expected to comport themselves with mutual respect and transparency. Students should strive to foster a collaborative and inclusive environment which prioritizes group success and cohesion over individual achievement. The personal views, beliefs, and characteristics of other members of the research group should be respected and appreciated, regarding individual differences as potential strengths which bolster the diversity of viewpoints and ideas held by the group as a whole. In general: treat others the way you would want to be treated.

Communications between members of the research group should prioritize honesty and transparency. Criticisms should be direct, but focus on constructive opportunities for improvement and continued development. Students are expected to be responsive to both in-person and electronic communications, and are responsible for providing timely feedback when requested.

3.2 Teaching/Service Balance

Students appointed as a TA for a portion of their overall time are expected to balance their responsibilities between their obligations to the instructor for their assigned class, as well as to their ongoing research project(s). To the extent possible, students should fulfill the minimum number of required hours in service of their TA appointment, but should otherwise strive to work efficiently to stay within this allotment of time.

Additionally, students are expected to participate in a variety of service activities as directed by their research advisor. Such activities may include: educational outreach activities, participation in professional societies and organizations, mentorship of fellow graduate students within and outside of the research group, assisting in the writing and preparation of research grant proposals, peer review of journal articles, and organizing or leading delegated group meeting activities.

3.3 Attendance at Conferences

Students are strongly encouraged to attend professional/academic conferences relevant to their research topic. MS students should aim to attend 1-2 conferences during the

completion of their degree, while PhD students should attempt to participate in 3 conferences. Attendance at conferences is conditioned upon the student's participation in the conference proceedings, including either delivering an oral presentation, or presenting a poster. Several financial support mechanisms exist within the department, and may be offered through other scholarships or travel awards sponsored by the conference organizers. Students are expected to apply for relevant scholarships and awards to facilitate attendance to relevant conference whenever possible.

Relevant conferences should be identified early when establishing a comprehensive plan towards completion of the degree. Some relevant conferences include (but are not limited to):

- **U.S. National Congress on Computational Mechanics (USNCCM)**
  Domestic conference, hosted in odd years, focused on a wide range of subjects related to computational mechanics, held in late July, accepting abstracts in January.
- **World Congress on Computational Mechanics (WCCM)**
  International conference, hosted in even years, focused on a wide range of subjects related to computational mechanics, held in late July, accepting abstracts in January.
- **International Conference on Computational Contact Mechanics**
  International conference (European), hosted in odd years, focused primarily on computational contact and interface mechanics, held in early July, accepting abstracts in February.
- **Engineering Mechanics Institute (EMI) Conference**
  Domestic conference, hosted annually, focused on a variety of topics relevant to theoretical and applied engineering mechanics, held in early June, accepting abstracts in December/January.
- **Natural Hazards Research Institute (NHERI) Computational Symposium**
  Domestic conference, focused on a broad range of computational approaches related to risk assessment for civil infrastructure systems subject to natural hazards.

3.4 Publications

Students are expected to contribute to peer-reviewed publications summarizing the outcomes of their research project(s). MS students should produce sufficient data/results for 2 publications, whereas PhD students should target 4 publications. These specific objectives should drive progress in conducting research over the course of the student's degree program. Specific reccomendations regarding the preparation and submission process for publications are outlined in the subsequent sections.

3.4.1 Format

Students are expected to use LaTeX to format all documents intended for publication in peer-reviewed journals. Many journals have specific formatting requirements which must be adhered to, and there are usually manuscript templates that can serve as a helpful guide on how to typeset manuscripts for submission.

3.4.2 Collaboration and Sharing

All manuscripts generated within the research group intended for eventual publication should be edited and shared between all co-authors using Overleaf. Students should regularly contribute to shared manuscripts, and solicit review of specific sections from their advisor. Comments and revisions will be provided directly through Overleaf.

3.4.3 Language

All written content shared for review by co-authors of a given manuscript should be subject to a preliminary language editing review conducted by the student. Use of AI-assisted grammar/language editing tools is permitted (even encouraged) to ensure that the resulting written content is free of language-related errors. Written sections submitted for review which contain a significant number of language-related error will be returned to the student for correction.

3.4.4 Figures

Figures fit for publication should be formatted with the highest quality possible, ideally in scalable vector graphic formats (.svg, .pdf, .eps) instead of raster image formats (.png, .jpeg, .gif) whenever possible. Certain journals prefer particular file formats (e.g. .eps) for the sake of consistency and to ensure the highest quality for publication. A variety of tools may be used to generate high quality figures, including (but not limited to):

- **MATLAB**: Easily integrated into existing MATLAB scripts, and allows exporting to .pdf format for highest quality. MATLAB also has functionality to use a LaTeX interpreter and Computer Modern font styles. MATLAB's `patch` plotting command is also very handy for visualizing 2D/3D finite element meshes.
- **Python**: Matplotlib provides much of the same functionality (and perhaps more) compared to MATLAB.
- **Tikz**: This is arguably the *fanciest* option for generating structured/mathematical images directly within the context of a LaTeX document, although it has the steepest learning curve.
- **PowerPoint**: If you're already skilled at creating images using MS Office tools (e.g. for presentations), PowerPoint allows for exporting images to pdf format. This is great for simple diagrams and the like.
- **Inkscape**: This is a fairly versatile (and free) graphical design tool with a moderate learning curve.
- **Blender**: Great for flashy visualizations of 3D datasets with control over lighting and texturing.
- **ParaView**: A very versatile post-processing tool intended for visualizing scientific data (especially for finite element analyses) contained in a variety of different mesh database formats.

When creating figures, bear in the mind the following graphic design considerations:

- *Clarity*: Ensure that figures are clear and easy to understand. Don't over-utilize symbols, abbreviations, or other annotations; prefer more descriptive labels that have an immediate interpretation. If your figure wouldn't make sense when taken out of the context of your manuscript as a whole, then you should make adjustments to address this. If something is not clear, consider changing how you are visualizing the data, or add additional context (e.g. a figure lengend, or supplementary description in the figure caption).

- *Simplicity*: Don't include excessive details that obfuscate the meaning of the figure you are presenting. Figures should be presented in a minimalistic fashion to convey a specific point or observation that you are trying to communicate to the reader. Focus on the intended purpose of a given figure in relation to how you are presenting your results and conclusions. In some cases, you may find that certain figures are not necessary if they do not have a specific purpose. In other cases, you may find that you are trying to present too much information in a single figure, in which case it may be better to present the information in multiple separate figures or subfigures.

- *Legibility*: Don't make font too small to the point where it is impossible to read. Conversely, don't make font too large to the point where it looks absurd or cartoonish. A good rule of thumb is to try to have the text in your figures be the same size as the text as it would appear in your manuscript (some journals note this as a requirement). When possible, you should also try to match the font in your figures to be consistent with the font used in your manuscript for the sake of consistency.

- *Colors*: Moderate use of colors can be helpful for clarity, but don't over-utilize colors if it's not necessary. Choose a consistent (minimalistic) and complementary color palette, potentially taking readers who may be colorblind into consideration, and ensuing that the figure is still interpretable when printed in greyscale (if possible).

3.5 Time/Project Management

Students are expected to manage their time effectively and work productively within the allotted number of hours required by their RA appointment to make progress on their chosen research project. When possible, a consistent daily work schedule should be adopted to establish a structured routine. Students should communicate their daily schedule to their advisor to set expectations regarding their anticipated reponsiveness to electronic communications and availability for scheduling meetings.

Students are expected to fulfill their various responsibilities pertaining to coursework, TA assignments, and RA appointments. However, students should regard research and professional development as their primary obligations, as the outcomes of these efforts will directly contribute to their current academic and future professional success.

Students should establish their own consistent time management practices to ensure that they are making progress to accomplish agreed upon goals, and to demonstrate this progress to their advisor. Documenting where one's time is being spent in other areas of professional develop will assist with planning and setting realistic/achievable goals.

While students are free to self-manage their time using whatever tools or methods they prefer, several specific reccomendations are provided below:

- **Microsoft Planner**: Offers low-complexity features for planning and managing multiple concurrent "plans," supports alternative ways to visualize tasks (as a task list, Kanban board, or calendar), allows for synced collaboration and coordination within a team/organization, and enables direct integration with other MS tools (Outlook, Teams, etc.).
- **GitHub Issues**: For projects managed within a GitHub repository, tracking and documenting progress using Issues provides a good way to integrate planning within the GitHub workflow, including built-in functionality for seeking peer review and revision.

3.6 Graduation Requirements

Students are expected to complete their degree within an agreed upon timeline, and must maintain steady progress towards the fulfillment of all program requirements as outlined in the graduate program degree requirements. This includes maintaining good academic standing in all courses in which the student has enrolled.

3.7 Enrollment in Coursework and Research Credits

MS students pursuing the thesis degree option must enroll in a minimum of 30 credit hours over the span of the 2 year degree program, including 6 hours of research credit hours and 24 credit hours of graduate coursework. The majority of courses taken should be selected from available offerings within the department; 2 courses from outside of the department may be chosen to supplement the graduate cirriculum, provided that such courses are relevant to the student's research topic and/or professional development. Students should communicate with the head of the research group to seek guidance and approval regarding enrollment in courses.

PhD students with an MS degree must complete a minimum of 48 credit hours, including 18 hours of coursework and a minimum of 30 credit hours of dissertation research. The PhD program offers potentially greater flexibility to take more courses outside of the department, pending approval from the students' dissertation committee members.

Students must complete submit a program of study which outlines their anticipated coursework/enrollments, and their projected graduation date. MS students should plan to form their thesis committee roughly one year in advance of their anticipated graduation date. PhD students should plan to submit their plan of study by the end of their first year.

Students participating in research under a GRA appointment are advised to enroll in a minimum of 1 research credit hour each semester to keep track of progress towards the completion of their thesis.

3.7.1 Thesis/Dissertation Committee

Both MS and PhD students must assemble a thesis/dissertation committee prior to submission of their plan of study. MS students should identify 3 thesis committee members among the faculty members in the department, whereas PhD students must additionally seek a dissertation committee member from outside the department. Committee

members should be chosen based on relevant expertise. It is common practice to solicit interest from a prospective committee member before including them on your plan of study. It is beneficial (but not required) to have taken courses taught by your committee members. Students should seek the guidance of their primary advisor before reaching out to prospective committee members.

3.7.2 Oral Defense (Thesis)

MS students must participate in an oral defense, during which they present their research in the format of an hour-long presentation. The primary audience consists of the student's thesis committee members, along with other faculty and graduate students within the department. Students must respond to questions from their committee members about their research and defend the validity of their research methodology while demonstrating their general knowledge of their area of specialization.

Students are encouraged to attend the oral defense of other students in the department to show support and interest, and to learn about the process in preparation for their own defense. Students should also arrange to practice their oral defense with members of the research group to help refine their presentation based on constructive feedback.

3.7.3 Thesis/Dissertation Document

MS (PhD) students must complete and submit a thesis (dissertation) document which provides a comprehensive yet concise summary of the research performed during completion of their degree. This document must by reviewed and approved by the students' thesis/dissertation committe, and is subject to revisions pending constructive feedback offered by the committee members. If the student has completed and submitted one or more journal articles, these works may be included directly within their thesis/dissertation as individual chapters; this is the preferred approach for completing the thesis/dissertation, as it ensures that the student's work will be disseminated within the broader academic community. In any case, the thesis/dissertation should also include a more detailed introductory chapter providing additional context regarding the chosen research project and the motivation for pursuing this subject.

The specific organization of the thesis/dissertation is somewhat flexible, although the following format is suggested as a reference guide:

- **Introduction/Background**: Include an abstract which provides a very brief overview of the content of the entire document to inform the reader of the overall scope of work.
  - **Motivation** Explain what the problem is that is being addressed, and why this problem necessitates ongoing research/investigation. Cite appropriate literature describing how/why this problem has relevance to the scientific/engineering community, and to society more broadly.
  - **Literature Review** Provide a somewhat critical review of what others have done to address the problem identified in the Motivation section, focusing on the limitations of others' approaches, and what remaining gaps exist in the literature to help contexualize the contributions that your work is making in solving the stated research problem. The document should provide a more comprehensive summary of concurrent approaches undertaken by others than

a journal publication would expound upon. This section should also provide some high-level context regarding other major themes of ongoing interest in the broader research community which relate to the work being presented.

  - **Overview/Objectives** Outline the research objectives that will be pursued in the presented work, and the expected outcomes in relation to the overarching motivation for the work.
  - **Organization** Ouline what content will appear in each subsequent chapter of the document, for ease of navigation by the reader.

- **Methodology** Present the details of the chosen research methodology undertaken to investigate or address the stated research problem. This may be organized into multiple chapters, as appropriate, but should follow a logical organization of information which guides the reader through your approach without assuming significant prior knowledge.

- **Results and Discussion** Present the results obtained from the application of the described research methodology in a well-organized format, and comment on the nature of the results. If your research project was motivated by a central hypothesis, assess whether the stated hypothesis tested valid/invalid, and your degree of confidence in making this assessment.

- **Conclusions and Future Work** Focus on the limitations of the research methodology and scope of work presented in the results, and highlight areas for improvement and continuation of the work. Emphasize the broader societal impacts of your work, and how your contributions may influence other ongoing efforts in the research community.

4 Guidance

This chapter provides supplementary guidance for students seeking to develop in strategic competencies relevant to the research group. Additional topics will be expanded upon based on specific requests or reccomendations.

4.1 Writing

Writing is arguably the most vitally important skill that one develops in the process of completing one's degree. In many cases, the ideas underpinning the research are never fully synthesized until they are put into writing and subject to multiple rounds of revisions. It is expected of all graduate students that they reserve time a fixed amount of time for technical writing every day. This aids in establishing writing as a consistent practice and habit, and also helps to synthesize and catalogue thoughts and progress during your academic journey.

While there are many schools of thought with respect to becoming a successful writer, there are several key strategies discussed subsequently which are reccomended to ensure that students participating in collaborative writing projects can work effectively.

4.1.1 Outlining

The writing of any research document should begin with an outline. Start by identifying the major sections/chapters in your document, and then proceed in drafting more detailed outlines for each of these subsections in the document. This approach should be applied recursively to draft more granular outlines for each subsection or paragraph appearing in each section. If drafting a document using LaTeX, the outline of content appearing in the document should be included as comments explaining what information should appear in each section.

When participating in the writing of a shared document, it is expected that students will first draft a high-level outline of the document for review by their advisor. At this stage, it is much easier to make adjustment to the structure and logical flow of the document, and to avoid wasted time and effort on writing that is not directly relevant to include in the final document.

4.1.2 Managing References

When drafting a large document such as one's thesis/dissertation, and even for journal articles or research proposals, it becomes necessary to keep track of a large number of citations/references. Several alternative bibliography/citation management tools can be utilized at the discretion of the individual student.

One suggested tool which has institutional support and broad adoption is Zotero, which offers options for sharing bibliographies with other researchers working within a larger group, compatability with MS Word, and integration with common web browsers. OSU also offers detailed documentation and workshops related to this tool to help you get started more quickly.

4.2 Programming

Effectiveness in completing assigned research tasks hinges upon a strong foundational understanding of computer programming. This spans a very broad range of skills centered around good software development practices (refer to section 2.4), and an understanding of how computer hardware and software systems interact. Moreover, students must develop technical competencies (discussed in the subsequent sections) to be able to work effectively within a software project team which shares common computational infrastructures and tools.

4.2.1 Multi-Language Software Projects

The two primary programming languages used for developing software projects within the research group are Python and C++. Python is nominally an interpreted language, meaning that code written in the Python language does not need to be compiled before it is ultimately executed. By comparison, C++ is a compiled language, requiring that code be converted into object code first before it can be executed. Python offers a great deal of flexibility and ease of use, including the ability to easily interface with a wide range of third party packages. C++ has a somewhat steeper learning curve, but provides tighter control over low-level representations of data which can lead to greater computational performance. Most scientific software intended for use in running large-scale simulations is developed in C++, whereas Python is widely adopted for creating scripted workflows and handling pre- and post-processing tasks.

In an ideal world, there would only be one "lingua fraca" in which all software projects are written, but in practice it is necessary to use different programming languages in different settings based on their relative strengths. In some cases, this can introduce difficulties: for certain projects that span a range of tasks (from low-level high-throughput computationally intensive operations, up to high-level lightweight workflows or meta-analyses), it might not be reasonable to develop the entire project in a single language. In such cases, it may be beneficial to write the computationally intensive portions of the program in a low-level language (like C++), while the high-level portions of the project are written in another scripting-orientation language (like Python). This is precisely how several of the projects developed within our research group are structured, with the goal of getting the best of both worlds.

The disadvantage of developing a multi-language software project is that it requires careful attention to how the different software components are interfaced. Usually, this requires defining an application programming interface (API) which "wraps" the exposed functionality of the lower-level software components within a small handful of functions that can be called by the higher-level components. Additionally, one must ensure compatibility and consistency in the way that data types are handled and transferred through the developed API. There are several methods for interfacing C++ and Python code within a multi-language project; arguably, the simplest approach entails the use of ctypes. For

specific examples of how this can be achieved, refer to the example software projects listed section 5.4.

4.2.2 High Performance Computing

High performance computing (HPC) encompassess a broad range of subjects focused around the accurate and efficient execution of scientific computing workflows at very large computational scales. In many cases, HPC is synonymous with parallel computing and the use of advanced *supercomputing* systems which are not commonly made available to the public. Examples of such systems include:

- OSU HPCC (High Performance Computing Center at OSU)
- TACC (Texas Advanced Computing Center at UT Austin)

Several U.S. national laboratories offer advanced open-source software frameworks specifically designed for execution on high performance computing systems, including:

- **RADIUSS** (including MFEM) - Lawrence Livermore National Laboratory
- **SEACAS** - Sandia National Laboratories
- **MOOSE** - Idaho National Laboratory

5 Resources and Support

Additional external resources are collected here for students to pursue on an as-needed basis.

5.1 Campus Resources

A wide range of resources are available to students and can be found online: https://okstate.com. Several specific resources of direct relevance to our research group are highlighted below:

- **Graduate College**
  https://gradcollege.okstate.edu
  Centralized collection of resources for graduate students, including information for new students, annoucements regarding campus events, and more.
- **Office of International Students and Scholars**
  https://iss.okstate.edu
  Provides information relevant to international students.
- **Writing Center**
  https://cas.okstate.edu/osu_writing_center/
  Offers one-on-one consultation and tutoring services to help revise and provide feedback on writing and related media (including presentations and figures).
- **High Performance Computing Center**
  https://hpcc.okstate.edu
  Provides information on OSU's supercomputing facilities, including user support and additional resources on how to request an account and get started.
- **Software Distribution Center**
  https://it.okstate.edu/services/software-distribution/
  Provides access to enterprise and research software.
- **CEAT Information Technology**
  https://ceat.okstate.edu/itservices/
  Offers services and support for computer equipment, and access to common Engineering software tools (MATLAB, Ansys, Office 365, and more).

5.2 Books

If you are developing a new software project from scratch, consider organizing your project in a manner similar to the following open-source projects currently maintained by our research group:

- **The Finite Element Method**
  (Hughes, 2003)
  A very comprehensive and rigorous coverage of classical finite element methods.

- **Computational Inelasticity**
  (Simo and Hughes, 1998)
  An excellent introduction to nonlinear constitutive modeling (including finite deformation plasticity and viscoelasticity).

- **Computational Contact Mechanics**
  (Wriggers and Laursen, 2006)
  Fantastic overview of contact mechanics and its practical implementation in a finite element context.

5.3 Tutorials

While learning any new skill/domain of knowledge is primarily benefited through participation in formal training and coursework, there are many fantastic online tutorials that can get you up to speed on specific topics or tools is a relatively short amount of time. A few specific reccomendations (mostly related to programming) are highlighted below:

- **Continuum Mechanics (Bob McGinty)**
  https://www.continuummechanics.org
  An introduction to the subject of continuum mechanics, providing definitions and comprehensive explanations relevant to the subject.

- **The Not So Short Introduction to LaTeX**
  https://tobi.oetiker.ch/lshort/lshort.pdf
  A comprehensive introduction to typesetting in LaTeX.

- **Git Guide**
  https://github.com/git-guides
  A brief but effective summary of how to use git in combination with GitHub for software version control.

- **Learn C++**
  https://www.learncpp.com
  An excellent introductory tutorial covering most of the core features of the C++ language.

- **High Performance Computing**
  https://hpc.llnl.gov/documentation/tutorials
  Lawrence Livermore National Laboratory provides a wide range of tutorials on various subjects related to high performance computing.

- **NHERI SimCenter**
  https://simcenter.designsafe-ci.org
  The Natural Hazards Engineering Research Institute (NHERI) features a wide range of computational tools and resources and accompanying documentation/tutorials geared towards an academically-focused community.

5.4 Example Software Projects

If you are developing a new software project from scratch, consider organizing your project in a manner similar to the following open-source projects currently maintained by our research group:

- **SWIRL**
  https://github.com/bdgiffin/SWIRL
  Structural Wind-borne debris Impact Risk assessment Library.

- **REMAT**
  https://github.com/bdgiffin/remat
  REversible physics and MATerial model library.

Bibliography

George T Doran. There's a S.M.A.R.T. way to write management's goals and objectives. *Management Review*, 70(11):35–36, 1981.

Thomas JR Hughes. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2003.

Juan C Simo and Thomas JR Hughes. *Computational inelasticity*. Springer, 1998.

Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9, 2016.

Peter Wriggers and Tod A Laursen. *Computational contact mechanics*, volume 2. Springer, 2006.