General advices

- http://yyahnwiki.appspot.com/Research_advices
- U. Alon, How to choose a good scientific problem
- http://managingbias.fb.com - Managing unconscious bias. Valuable resource from Facebook about our unconscious biases.
    - https://implicit.harvard.edu/implicit/ - Take tests and realize how biased you are!

- Merton's CUDOS:
    - **Communalism:** All scientists should have equal access to scientific goods (intellectual property) and there should be a sense of common ownership in order to promote collective collaboration, secrecy is the opposite of this norm.
    - **Universalism**: All scientists can contribute to science regardless of race, nationality, culture, or gender.
    - **Disinterestedness**: according to which scientists are supposed to act for the benefit of a common scientific enterprise, rather than for personal gain.
    - **Originality**: requires that scientific claims contribute something new, whether a new problem, a new approach, new data, a new theory or a new explanation.
    - **Skepticism (Organized Skepticism)**: Skepticism means that scientific claims must be exposed to critical scrutiny before being accepted.

- Don't be afraid to discuss about your career, ask for the reference letter, and other stuff outside research. I'm happy to talk.
- If you aim to stay in academia it is a good exercise to maintain a research statement. It helps you to plan your career direction and to think about the relevance of your research in broader contexts. If you plan to go to industry, it is a good idea to have your product (software, visualization, …) portfolio on http://github.com.
- Michael Faraday said: "the secret is comprised in three words — **Work, finish, publish**." For me, "**finish**" is the most important word.

- Social aspects of science

- Don't: Professor, Professor Ahn, Dr. Ahn, Professor YY / Do: YY

Meetings

Group meeting (2018 Summer): W 11am-Noon (Info East 322).

CX Reading group (2018 Summer): ?

http://yongyeol.com/group/calendar.html You can add this calendar (+ button at the right bottom). Volunteer to get a presentation spot (send an email to Nathaniel).

Other meetings you may want to attend (feel free to add):

- CNetS group talk series
- NaN meeting: http://cnets.indiana.edu/groups/nan/meetings
- Sporns lab meeting: http://www.indiana.edu/~cortex/index.html
- CogLunch: http://cogs.indiana.edu/events/cognitive-lunch.php

Lunch, tea, and beer. They are great opportunities to relax and talk about everything (and practice your english if you're not proficient enough). Great ideas often come with tea or beer.

You can check YY's travel schedule here: http://yongyeol.com/travel/.

Communication

Mailing list: yy-l@indiana.edu

To subscribe, send an empty message to yy-l-subscribe@indiana.edu. It is an open mailing list that anyone can join. Use it. Share ideas and interesting things.

Project discussion & information sharing: https://yy.slack.com

Slack.com is a channel-based messaging + Information sharing service. You can think of it as a modern IRC. You can also download desktop & mobile apps. Feel free to peek into channels of your interest.

Computing

Resources

KB: What linux systems are available?

Mandatory usage guidelines before using any computing resources: http://carl.cs.indiana.edu/cnets-howto/

Main server: **snowball.cs.indiana.edu** (192GB RAM + Several TB Storage)

The large storage is mounted at `/l/nx/`. If you want to have a home directory here send Rob an email. Put data into `/l/nx/data/`. If you have any special request (e.g. db server, packages, etc.), send an email to Rob and me.

Using Jupyter notebook on the server

- https://youtu.be/IUHmL1Ze2Ss

Security

It's becoming alarmingly easy to brute forcing your short passwords and there are always security holes you should be careful about. Some suggestions:

- Use long password. see http://xkcd.com/936/
- Use password managers such as **1password, lastpass,** and **dashlane** to manage long, site specific passwords.
- Use two-step verification, especially for services like Google.
- Always backup. Today is the International Backup Awareness Day.

Workflow

Please use https://github.com (or https://github.iu.edu) for your projects. It is also possible to use dropbox or other tools, but I strongly encourage to use the github to store code, paper, and small data. When the project ends, **you should publish your code and data so that others can replicate your results**. Organize it carefully so that anyone can easily access it and reproduce our results. To learn git, see http://yyahnwiki.appspot.com/Git

Use Github's wonderful functionalities: pull requests, issue tracker, and project tracker.

Here's a sample git repo template for research projects: https://github.com/yy/project-template I'd be happy to explain the structure.

Learn powerful text editors. In doing so, your productivity will keep increasing for many years. I use Vim, but there are other good options (Emacs, sublime text, etc.). Vim and Emacs are particularly versatile since practically every machine has them.

It is a good idea to learn build tools (e.g. GNU make, Snakemake). I use Snakemake. A nice use of make in research: http://www.narrykim.org/s/park-dicer-2011/ Also see "An efficient workflow for reproducible science"

I highly recommend to use Jupyter notebook (now Jupyter Lab) for experiments, where you can maintain documentation, code, and results in one place. At the same time, try to package your code so that we can test and reuse.

Plain text formats are prefered when writing documents (e.g. Markdown, reStructuredText, LaTeX, etc.). They work better with revision control systems and are accessible. Collaborative

platforms such as Google Docs, sharelatex, overleaf are good alternatives. Latexmk is a pretty convenient tool for automatically monitor and compile tex files.

Maintain a summary document that contains method & data descriptions and results (plots/tables + captions), to keep track of your research progress.

Some tips:

- **Know what you're doing and why**: Always ask yourself: what is the main question that your paper tries to address? what is the main result of the paper? Can you summarize it in a single sentence? Why is your problem important in the broader contexts?
- **Be obsessive about robustness**: Always ask yourself: can this be an artifact or a result of systematic biases? What could produce such artifacts? Think of a situation of retracting your paper because someone discovers an embarrassing error in your paper.
- **Start small**: Start with small examples (or small samples of data) that you can exactly understand every detail and can iterate quickly. Make sure everything is correct and then scale up.
- **Test your code**: Create tests for your code base. https://docs.pytest.org/en/latest/
- **Document, document, and document**: always write down details about data and methods. Where did the data come from? How did you process the data? What are the parameters for the simulation? What exactly did you do? Make sure to make everything reproducible (even the manual curations, by creating a data file of curation operations and script).
- **PEPKAC**

Feel free to use my wiki (http://yyahnwiki.appspot.com/Home) to collect and organize references and thoughts.

Homepage

You need a homepage.

Put your CV on your homepage *even if you don't have any paper yet*. Periodically update your CV and homepage.

A good option: https://pages.github.com

Reference Letter

1. You should first discuss with me before putting my name as a reference.
2. You should allow me about one month to write it. In other words, don't expect an

awesome letter to be written within a few days.
3. It is usually much easier for me if you provide me with some material, such as your CV, description of your projects, your roles in the projects, the points (qualities) that you want to emphasize, and other useful information.

There are some helpful articles about it.
http://yyahnwiki.appspot.com/Research_advices#h_949f0dc37da5a88f07d9f89096ab2320

Plagiarism and Misconduct

http://yyahnwiki.appspot.com/Plagiarism
https://www.indiana.edu/~academy/firstPrinciples/index.html

Plagiarism or misconduct may end your career (possibly mine too). Write from scratch. It's dangerous to write while looking at another paper. Making the project reproducible by ensuring data provenance and code publication is helpful to avoid any slippery slope.

It is always better to find errors than publishing them. It is always better to put erratum or retract a paper than not doing anything about it. If you find something's wrong it is never late to report it even after the paper is published.

Classes

As Matt Might wrote, an easy way to fail a Ph.D. is focusing on grades or coursework too much because the *grades become irrelevant* after graduation. What matter after graduation are what you have produced (papers for academia and papers/software for industry), what kinds of skills you mastered, and how others think about you. Courses should serve your research and career. Good grades should not be your objective. Use courses to initiate collaborations or learning & applying new methodologies.

yy is teaching mainly:
- I606: Network Science
- I422/I590: Data Visualization
- I709: Complex Systems Seminar II

Please share other useful courses (feel free to add):
- Machine learning
- Applied machine learning
- Network Neuroscience by Olaf Sporns

Teaching

Doing an AI (TA) requires significant time commitment so you will have less time for your research. However, I'd recommend to do some AIs even when funds are available for RA ship, especially if you are interested in academic positions. Teaching experience is helpful to assess whether, and how much, you like teaching (although doing an AI is different from being an instructor). Because most academic career path involves teaching as a substantial part of the job, having some teaching experience is very helpful. Then, teaching a topic is a really good way to learn the topic because you really need to understand the topic to teach it. So if there is a course about a topic that you want to master, it is not a good idea to serve as an AI. Finally, if you are doing an AI, try to excel. Having some teaching experience can be helpful for your job prospect, especially if you can substantiate your excellence in teaching in the form of awards and reviews, such as 'best AI' award, a really strong course evaluation, etc.

Recommended Books and Resources

Programming (Python)

http://yyahnwiki.appspot.com/Python

1. https://www.python.org/doc/ (Python official tutorials)

Networks

http://yyahnwiki.appspot.com/Network%20science

Information theory

1. MacKay's information theory book: https://www.amazon.com/Information-Theory-Inference-Learning-Algorithms/dp/0521642981 and lectures: https://www.youtube.com/playlist?list=PLruBu5BI5n4aFpG32iMbdWoRVAA-Vcso6

Statistics

http://yyahnwiki.appspot.com/Statistics#h_6225eb5bf8a031f750a1b03f810ccc6a

1. An Introduction to Statistics: http://work.thaslwanter.at/Stats/html/index.html - doing statistics with Python.

Machine learning

http://yyahnwiki.appspot.com/Machine%20learning

1. An Introduction to Statistical Learning with Applications in R: http://www-bcf.usc.edu/~gareth/ISL/ - A very good & easy introduction
2. Machine Learning: A Probabilistic Perspective http://www.amazon.com/Machine-Learning-Probabilistic-Perspective-Computation/dp/0262018020/ - Grad level textbook
3. https://www.youtube.com/playlist?list=PLD0F06AA0D2E8FFBA Youtube video series by mathematicalmonk