**ORIE 4580/5580**
**Course Project: Gender Bias in Corporate Settings**

In celebration of Claudia Goldin, the 2023 Nobel Prize in Economics, and a Cornell alumni!

# 1 Overview

The aim of the project is to use simulation modeling to study gender disparities in corporations, and to perform counterfactual studies to understand how this can be remedied.

Before you begin, its worth getting some sense of what the issue is:

- To convince yourself that gender-bias could be an issue, try a gender-career implicit-bias test (seriously, take it if you have never done so, *even if you are sure you are not biased!*)
- To learn more about it (and maybe to get some ideas of what you can try and model), check out the extensive literature on gender bias in Silicon valley (to take just one example. . . ) - for example, see the great work done by Elephant in the Valley (associated article in Time), some of the many gender diversity survey of the SV workforce, and the many articles (like this and this) with ideas on what to do.

Your aim in this project will be to build and study simple models of how small local biases can lead to large global imbalances in corporations. This is an important idea in many famous simulation models; in particular, it is closely related to the famous Schelling model of segregation in economics (which won a Nobel prize). The methodology I propose below to study this is loosely based on this article, which also is worth reading before doing this project. In the process, you will also grapple with issues in simulating and reporting outcomes for steady-state model, as well as comparing different systems.

# 2 Outline and Main Deliverables

Below I will propose a simple model for studying the mix of employees in a firm. You should feel free to use this as a starting point, and then build on it (or not):

- The employees are classified into four levels: executives (E), senior-management (S), management (M) and junior employees (J). The model takes as input some target number of employees at each level (for example, it could be $5, 20, 100$ and $400$ respectively), which is decided by some external reasons (budget, need, etc.).
- Each employee in level $i \in \{E, S, M, J\}$ independently retires/leaves the company after some $Exp(\lambda_L)$ time. The rate of departures may be higher at higher levels (so $\lambda_E \geq \lambda_S \geq \lambda_M \geq \lambda_J$) if you believe older people retire sooner, or the opposite if you believe there is more churn at lower levels (you could study both to see which is worse).
- The rate of people dropping out of the workforce also has a gender specific component; in particular, there is evidence that women drop out faster than men (in particular, at

junior levels) due to childbirth and familial commitments. You can model this by saying a woman at level $L$ has an additional timer $Exp(\kappa_L)$, and exits the company at the minimum of the two timers (which is equivalent to saying that each woman in level $L$ exits after $Exp(\lambda_L + \kappa_L)$ time.

- When an executive-level employee retires/leaves, the vacant position is filled by promoting some chosen senior-manager. Similarly, any vacancy created at a lower level either due to a promotion or a departure is immediately filled by promoting an employee from the lower level. Finally, junior-level vacancies are filled by hiring new employees (assume there is an infinite pool of such potential hires).

Let $V_E(t), V_S(t), V_M(t)$ and $V_J(t)$ be counting processes for the number of vacancies as the four levels. Convince yourself that in the above model, these are Poisson processes, and find the corresponding rate of each of these processes (are these processes independent?). For numerics, as a start, you can set all $\lambda = 1$ and all $\kappa_L = \kappa$ (and vary $\kappa$).

Here are some thoughts on what you could use this model to study:

- Let $X_L(t), Y_L(t)$ represent the number of female and male employees in each level $L \in \{E, S, M, J\}$ at time $t$. Assume new employees are equally likely to be male or female, and moreover, whenever a vacancy is created at a higher level, then the most senior employee at each level is promoted. Write a simulation model for this system, starting from two different initial states: *i.* all male employees, and *ii.* all female employees. Plot the fraction of women at each level over time for both these starting states, and use this to determine an appropriate relaxation time $\tau_R$ for the steady-state simulation.
  *Note: To model promotions, you could store ordered lists of employees (and their gender) at each level. One can avoid this (how?), but it may be useful for complex models.*
- Next, run a number (say 100) of independent simulations for upto time $5\tau_R$, and plot the expected value (and 95% CIs) for the fraction of women at each level over time. Also plot the first few sample paths to compare.
- Next, suppose the candidate for promotion at each level is chosen not strictly based on seniority, but instead, by selecting between the seniormost 3 candidates at each level. In particular, if the three seniormost candidates $\{1, 2, 3\}$ at level $i$ have been at that level for time $T_1 > T_2 > T_3$ respectively, then candidate $i$ is chosen for promotion with probability proportional to $e^{\gamma T_i + \alpha \mathbb{1}[i \textbf{ is male}]}$ (this is sometimes referred to as the *multinomial logit/logistic* selection rule). Here, $\gamma, \alpha$ are parameters; the former represents a preference for seniority, while $\alpha$ (which can be positive or negative) represents a *bias* towards selecting male candidates.
  Adapt your simulation from the previous parts for this selection model. Note that you need to store the time spent by each employee at their current level (resetting it when the employee is promoted). To make sure your model is correct, set $\alpha$ to 0, and $\gamma$ to a very large value (which increases the probability the most senior is picked), and benchmark against the results you get in the previous parts.
- Finally, set $\gamma = 1$, and study effect of different choices of $\alpha$ on the gender ratio at different levels in the firm. You can choose what experiments to do to best study this.
  *Note: One good way to think about the bias is that given male and female employees with exactly the same time spent at a level, the male employee is $e^{\alpha}/(1 + e^{\alpha})$ more likely to be picked then the female employee. A 5% bias means setting $\alpha$ s.t. $e^{\alpha}/(1 + e^{\alpha}) = 0.05$.*

This gives a simple initial model for this setting (and is at least what we would expect you show us in your demo – see below). In particular, note that there are 2 main parameters that model disadvantage women face: the bias $\alpha$ and the additional drop-out rate $\kappa$. *Which has a stronger effect on the (lack of) diversity? How does it change across layers?* (i.e., which can better explain decreasing diversity as we go up the pyramid?)

As with any 'real' project, though, your brief is fairly open-ended. As an example, here are some things you could study:

- How can you test that your simulation is correct? In particular, are there special settings (number of levels, parameters $\lambda, \kappa, \alpha, \gamma$) for which you can work out the system performance metrics, and use this to check your sim? For example, what if there was just one level
- Can you try and figure out a 'law' (i.e., a formula) that governs the bias at a certain level w.r.t. the level of bias? Try for flatter hierarchies (i.e., one or two levels).
- You can compare different ways of 'mitigating' bias:
  - Having fixed 'quotas' (for example, requiring that there are at least $a$ members of each gender in the the last $k > a$ promotions.
  - The 'Rooney rule' - requiring that the set of candidates considered for promotion have at least one candidate of each gender (see this and this paper for more details.)

Our ultimate aim is to make sure you know how to build a simulator for a complex system, use it to explore design choices and their impact on participants, and finally, can communicate your findings to a client. We now provide some guidelines on how you should approach the project and report.

## 2.1 Deadlines and Deliverables

The project should be done in **teams of 4-5 students** (not more, not less). Please try to form your teams by the end of the week and upload it on Gradescope (we will create a template to do so after next class). If there are students who do not have teams, we will group them up by Monday.

Your deliverable is in two parts: a basic simulation model, and the project report.

- Your simulation model (code + demo) is due on **Friday, December 8** (our official 'final exam' date).
  You must demonstrate your code to us over zoom (no submissions necessary). We will upload a sign-up sheet today with 15 minute time slots; all teams should enter at least 4 time-slots they can attend, and then we will make the slots. Ideally, you should come as a team to demonstrate your code, but all team members need not attend.
- For your final report, we want to encourage you to use *reproducible science* practices. A simple way to do this is to present your report in a main Jupyter notebook, with separate files containing any code you need to include, and any data you generate in past experiments. You can import these into the main report notebook, use Markdown to describe your findings, and in addition, give the reader an opportunity to run your

code with their own parameter settings.

- Your final reports are due on Friday **December 15 at 11.59am ET** (i.e., noon) on Gradescope. You can upload one Jupyter notebook (compulsory), plus at most one additional pdf document (feel free to put all information in the notebook though).

## 2.2 Report Guidelines

Your project report should be made up of two sections: a *main findings section* describing your analysis in a way that is accessible to non-experts, and a *technical appendix* to allow other simulation specialists to repeat/test your analysis. The main part of the report should be designed for non-specialists.

Your project report should not be very long. It is difficult to quantify this in a notebook, but in particular, the text sections (if written separately in a document) should not exceed **10 pages** with **1.5-spacing** and at least **12pt font**. Technical appendices are not included in the page count.

Below are possible sections you may want to consider in your project report – also see the Cornell COVID modeling report as an example. If you feel uncomfortable with any portion of the outline suggested below except the Executive Summary, then feel free to modify the suggested outline:

- Executive summary: This is the part that decision-makers (executives/senators/etc.) will read. Give a synopsis of what you did, what you discovered, and how the tools you developed can be used further. Think what you would want to know if you were a high-level manager and this was all that you read. This section should be short (at most two pages of text).

- Modeling approach, assumptions, parameters: Describe the simulation model in general terms. This section is for intelligent non-specialists, who want to ensure that your model is reasonable and captures all of the important aspects. Describe also what inputs are needed to drive your model and how you obtained them. You may want to defer a large part of this section to the appendices if it is predominately technical in nature, but you should at least describe what you found in the data in a manner suitable for intelligent non-specialists.

- Model Details: Describe the model in technical detail, in a way accessible to experts. You can import diagrams to detail your Markov chain. Also, explain in broad strokes how you checked that your model was correct as implemented (you can refer to detailed tests in the appendix sections). The goal is to convince decision makers that your model can be trusted to correctly answer the questions it was designed to answer.

- Model analysis: Describe how you used your model to study different questions about vaccination strategies. For each analysis, clearly explain what you are testing, and then present the output (with confidence bounds, axis labels and legends). Analyze the results and make recommendations. Think about sensitivity analysis for quantities that are uncertain or approximate, and how robust your recommendation is to such changes. Your readers may not be simulation specialists and cannot possibly know all that simulation has to offer. Therefore be creative and try to go above and beyond what has been asked for.

- Conclusions: Briefly recap your findings and recommendations.
- Technical Appendices: These are intended for a simulation specialist who has training at the level of having taken ORIE 4580/5580 in the past. Readers of this section understand simulation concepts and have some familiarity with Python. Think separately about the model and the implementation, and give any critical details about the model, and its implementation. In particular, you should detail any functions that your code makes available, and give the reader an opportunity to test out parts of the code (for example, via animations, etc.). The aim is to provide details for someone who can then build on your code-base.

## 2.3 General Recommendations

- An open-ended project like this rarely has completely specified deliverables, and often it is up to you to decide how much to do, and how to manage your time and effort. If you have doubts over any particular details of the problem, make assumptions about it, and use your report to outline and justify these assumptions. We will be available to help with this, but may choose not to specify all details.
- We are interested in average performance measures, as well as histograms. Also, think about (and report) other performance measures that could be of interest.
- Please try to plan your project and design your code to be modular. Use classes/functions wherever possible to ensure that the code is easy to build on. Also make sure the code is well commented and documented, as this will help a lot when working with teammates.
- Please remember that this is a team project – make sure you work well as a team to get things done. Good code commenting, dividing responsibilities, version control, planning meetings, etc. are particularly important here, since you are working separately.