Gabe
GabeL@virginia.edu
https://gabriel-vzh2vs.github.io/SYS3062-Website/
UVA's Systems and Information Engineering

January 13th, 2026

SYS 3062: Lab 0

Introduction to Simulation Lab

DES integrates theory and practice to model and analyze stochastic systems through computational experiments.

- **Theory:** Modeling verification & validation, types of uncertainties, and stochastic processes.
- **Analytics:** Output analysis (means, variances) and Input analysis (Distribution Modeling, Goodness of Fit).
- **Methods:** Monte Carlo method, Queuing Systems, and Random Number Generation.
- **Software:** Python (primary) and XLRisk (spreadsheet plugin) with students able to use any professional software if they wish.

GabeL@virginia.edu

What is Lab?

- Hands-On
- Rigorous
- Collaborative

GabeL@virginia.edu

Lab Day-to-Day Structure

The Lab will be composed of the following elements:

- Q&A Session for Prelab/Course Content
- Lab Work Session: Debugging, Answering Student Questions

The time blocking diagram of the typical lab is below:

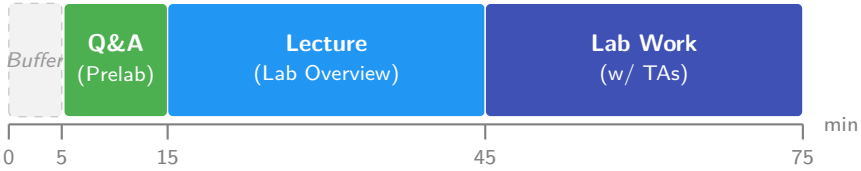


Figure: Timeline of the 75-minute lab session structure.

GabeL@virginia.edu

Q&A = Pre-lab and Lecture Discussion

GabeL@virginia.edu

Lab Lecture Helps You Grow!

Every lab will be based on a set of Prelabs and a single lab from the textbook! Lab lectures will have two components:

- Conceptual Information explaining the Simulation Model we are building in Lab;
- Pseudo-Code with Code Diagrams for the overall flow of the model.

GabeL@virginia.edu

Lab Pseudo-code Example

Definitions

- `multisession`: Spawns background Workers.
- `expand.grid`: Flattens loops.

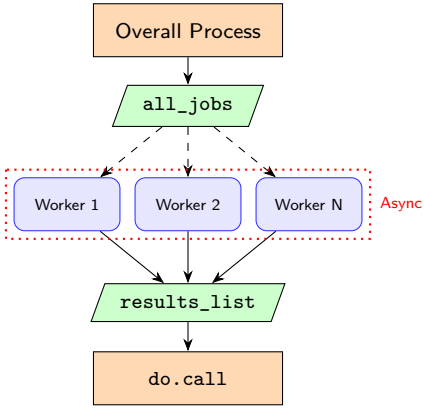
Pseudo-code for Process

```
# Setup
plan(multisession)

# 1. Flatten
all_jobs <- expand.grid(...)

# 2. Async
res <- future_lapply(
  X = 1:nrow(all_jobs),
  FUN = function(i) {...},
  future.globals = c(...),
  future.packages = c(...)
)

# 3. Aggregate
final <- do.call(rbind, res)
```



GabeL@virginia.edu

Lab Work = Building the Model from the Lab

GabeL@virginia.edu

Lab Difficulty

There are three levels of difficulty for the labs noted by the following colors:

- **Green**: Labs that we consider approachable and should take without assistance less than 30 minutes to complete;
- **Black**: Labs that we estimate it will take an hour to complete this task without assistance;
- **Blue**: Labs that are made to prepare the student for the project, we recommend taking 2-3 hours to complete this lab.

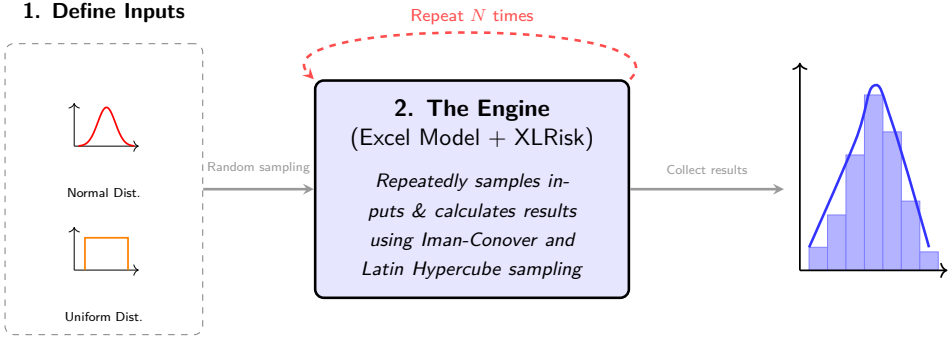
Important Note

Generally, attending lab makes it faster and easier to complete the Labs. This is one of the reasons you should be here! Besides Lab technically being mandatory.

GabeL@virginia.edu

What is XLRisk?

- An open-source add-in that brings Monte Carlo simulation directly into Excel.
- It allows us to replace uncertain single numbers (e.g., future demand, costs) with **random variables** from distributions.



GabeL@virginia.edu

Step 1: Download the Plugin

XLRisk is an open-source Excel add-in.

Download Source

1. Go to the GitHub repository: `https://github.com/pyscripter/XLRisk`
2. Look for the file named **XLRisk.xlam**.
3. Download this file to a folder you can easily access (e.g., `Documents/ExcelAddIns`).

Important Note

Do not open the file directly by double-clicking it. You must load it through Excel's menu system.

GabeL@virginia.edu

Step 2: Installation (Windows)

Windows often blocks downloaded macros for security.

1. **Unblock the File:**
   - Right-click the downloaded `XLRisk.xlam` file.
   - Select **Properties**.
   - At the bottom of the General tab, check **Unblock**.
   - Click **Apply** and **OK**.
2. **Load into Excel:**
   - Open Excel. Go to `File > Options > Add-ins`.
   - At the bottom, **Excel Add-ins** and click **Go...**
   - Click **Browse...**, select your file, and click **OK**.

GabeL@virginia.edu

Step 2: Installation (macOS)

XLRisk is compatible with Excel for Mac.

Mac Instructions

1. Open Excel for Mac.
2. Go to the **Tools** menu in the top bar.
3. Select **Excel Add-ins...**
4. Click **Browse...**
5. Navigate to where you saved `XLRisk.xlam`.
6. Select the file and click **Open**.
7. Ensure the checkbox next to XLRisk is ticked and click **OK**.

GabeL@virginia.edu

Step 3: Verification & Resources

Verify Installation:

- You should now see a new tab in the Excel Ribbon labeled **XLRisk**.
- Click it to see the simulation tools.

Need Help?

- Visual, step-by-step screenshots are available in the course online textbook.
- You can ask the TAs or Instructors for clarification! The textbook is also a great reference for most of the course content.

GabeL@virginia.edu

Software Setup: Python Environment

For setting up Python, we focus on reproducibility and ease of use.

The Challenge:

- "It works on my machine" issues.
- Annoying manual installations.
- Conflicting library versions.

The Solution: `uv`

- An extremely fast Python package manager.
- Handles virtual environments automatically.

Goal

You focus on the modeling and learning the concepts, and not the boilerplate!

GabeL@virginia.edu

Step 0: Install Git

Before we begin, you need **Git** to download the course materials.

Check if you have it

Open a terminal (Command Prompt or Terminal) and run:

```
git --version
```

If you see a version number, you are all set!

If you need to install it

- **Windows:** Download the installer from:
  `https://git-scm.com/download/win`
- **macOS:** Run this command in the terminal:
  `xcode-select --install`
- **Linux:** Use your package manager (e.g., `sudo apt install git`).

GabeL@virginia.edu

Step 1: Install uv

This Software Package uses uv to manage Python dependencies automatically.

macOS / Linux

```
curl -LsSf https://astral.sh/uv/install.sh | sh
```

Windows

```
powershell -c "irm https://astral.sh/uv/install.ps1 | iex"
```

Note: You may need to restart your terminal after installation.

GabeL@virginia.edu

Step 2: Clone Repository

Download the course repository to your local machine.

Terminal Commands

```
git clone https://github.com/Gabriel-vzh2vs/SYS3062SoftwarePackage
cd SYS3062SoftwarePackage
```

Ensure you are in the correct directory before proceeding to the next step.

GabeL@virginia.edu

You do not need to manually install libraries. `uv` handles the environment setup.

Run Command

```
uv run main.py
```

- **First Run:** `uv` will automatically create a virtual environment and download dependencies.
- **Subsequent Runs:** Execution will be instant.

GabeL@virginia.edu

Step 4: VS Code Preparation

1 Open the Project

- Go to `File > Open Folder...`
- Select the `SYS3062SoftwarePackage` folder you just cloned from git.

2 Install Extensions

- Open the Extensions panel (Box icon on the left).
- **Required:** Install the Python extension (by Microsoft).
- **Optional:** Install UV Wingman for easier package management.

GabeL@virginia.edu

Step 5: Connect Environment

VS Code needs to know where your `uv`-managed libraries are.

Select Interpreter

1. Press `Ctrl + Shift + P` (Win/Linux) or `Cmd + Shift + P` (Mac).
2. Type and select: **Python: Select Interpreter**.
3. Choose the option containing `('.venv': venv)`.

Troubleshooting

If you don't see the `.venv` option, ensure you ran `uv run main.py` in the terminal at least once to create the environment.

Verification: Look for `(.venv)` in the bottom-right corner or your terminal.

GabeL@virginia.edu

Demo

Now, it's Demo Time! We will go over a few demos using Simpy, Salabim, and custom Gabe-Grade Code:

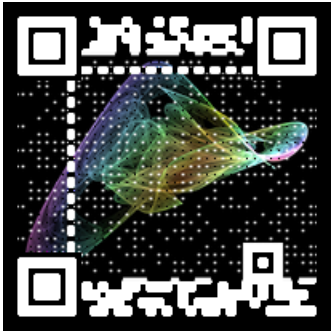
- Simpy + Ciw: Booking Model with Steamlit
- Salabim: Stop Lights Model
- Anylogic: Needles Model (ABM)

GabeL@virginia.edu

Feedback Form

Throughout the course, I may ask for your feedback on:

- **Course Materials**
- **Lecture & Lab**
- **Assignments**



GabeL@virginia.edu

First, I will thank you all for coming to lab at 5 pm on a Tuesday!

Now, you need to do the following tasks:

- **Get** your name card, so I can try to remember your names!
- **Read** Prelab 1 for next week!

It's going to be a great semester!

GabeL@virginia.edu