

# Rethinking Computing Education with Vocareum and Canvas



**Prof. Alexander (Alex) Dowling**

**adowling@nd.edu    dowlinglab.nd.edu**

Department of Chemical and Biomolecular Engineering

University of Notre Dame

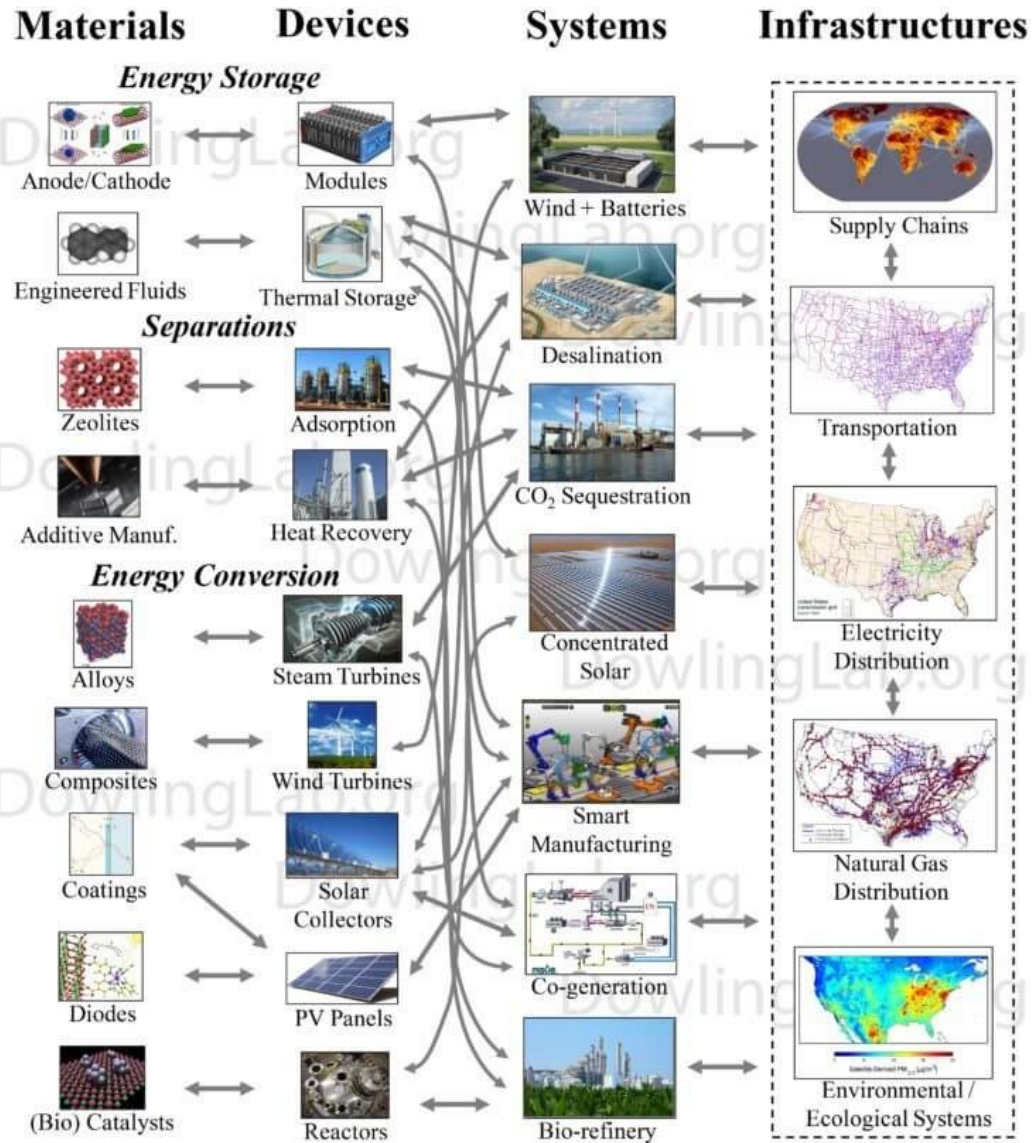
November 18, 2021

**colab**

**vocareum**

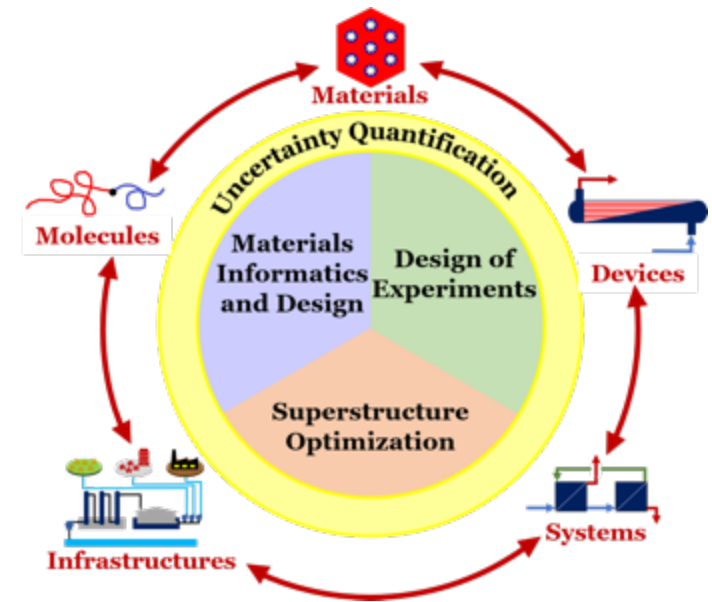


# Research: Process Systems Engineering

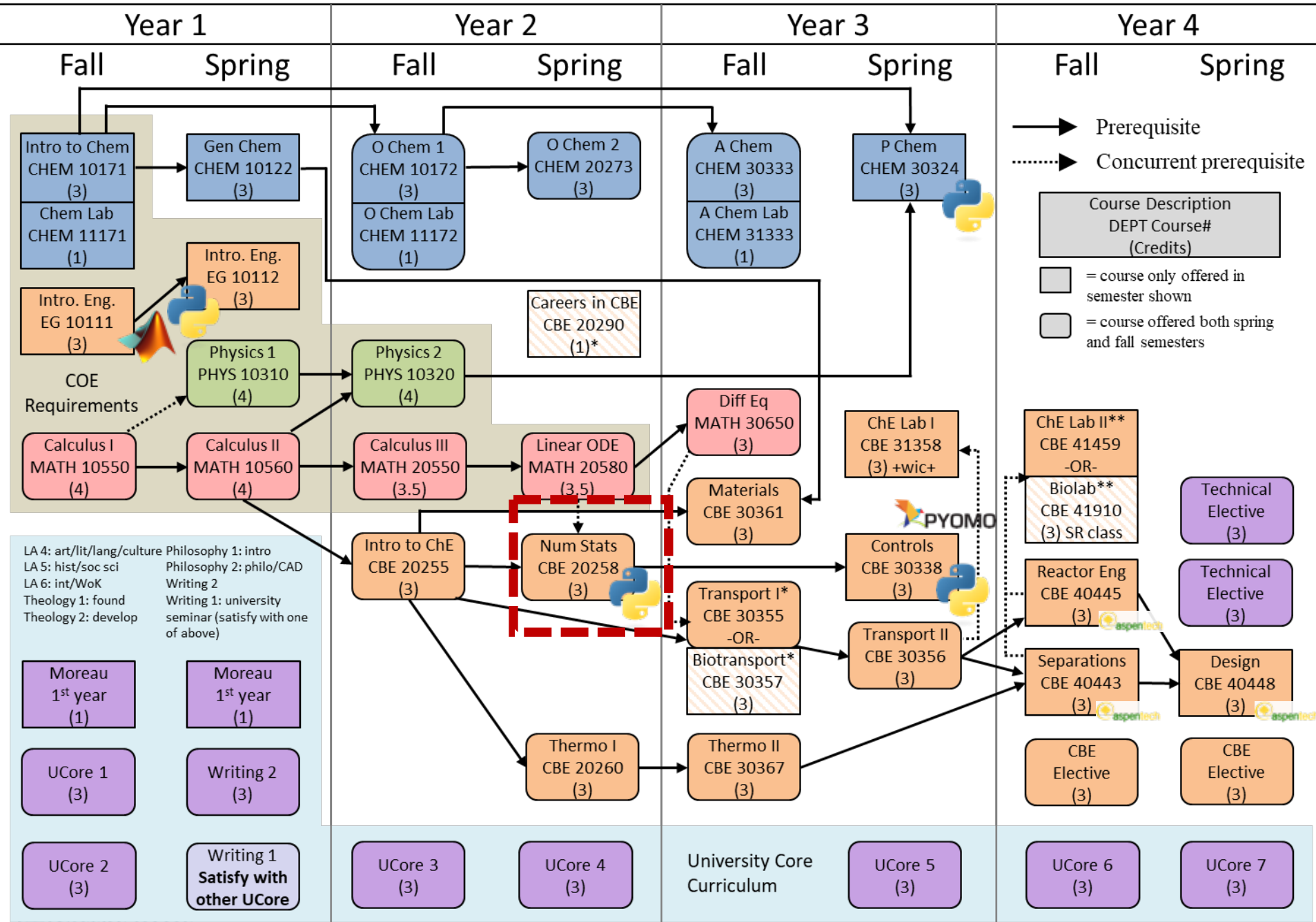


## Themes

- Mathematical Modeling
- Computational Optimization
- Applied Statistics and Uncertainty Quantification
- Energy, Sustainability, & Systems Biology Applications



# Chemical Engineering Suggested 4 Year Curriculum University of Notre Dame



## Current Practice: Computing & Statistics

MATLAB in freshman engineering sequence

Sophomore-required **Numerical & Statistical Analysis (NSA)**

Ad-hoc computing & statistics in upper-level classes:

*"You learned this as sophomores... just figure it out" – Prof. Anonymous*

## Vision

*Vertically integrate computing and statistics throughout the undergraduate curriculum*

# Modernizing Numerical and Statistical Analysis

## Backward Course Design Set Clear Learning Objectives

At the end of the semester, you should be able to...

1. **Create mathematical models** and **apply computational methods** to analyze systems using basic principles of chemical engineering (e.g., mass and energy balances, thermodynamic equilibrium, etc.)
2. **Analyze data** and **quantify uncertainty** using standard statistical techniques and mathematical models grounded in engineering fundamentals
3. Independently plan, implement, and debug short (100 to 300 lines) **Python computer programs** to analyze data, solve engineering mathematical models, and visualize results

## Major Changes

### Reorganized class topics

- Removed advanced topics (QR factorization, compression with SVD, trust regions, BVPs, PDEs)
- Emphasized fundamentals, especially probability & statistics
- Added mass and energy balance examples

Switched to **Python**, with great student buy-in

Incorporated **active learning** into lectures

### Shortened assignments

# Active Learning is Essential for Computing and Statistics

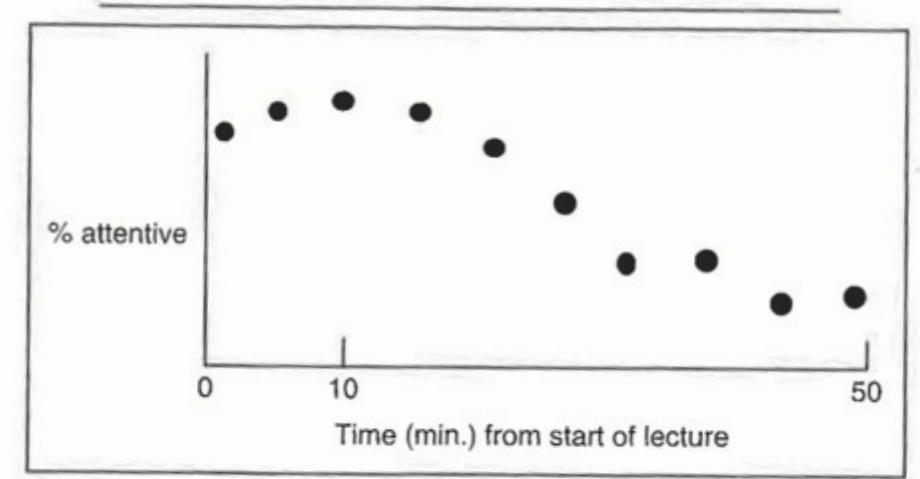


Figure 6.3-1: Attentiveness versus Time in Lecture—No Activities

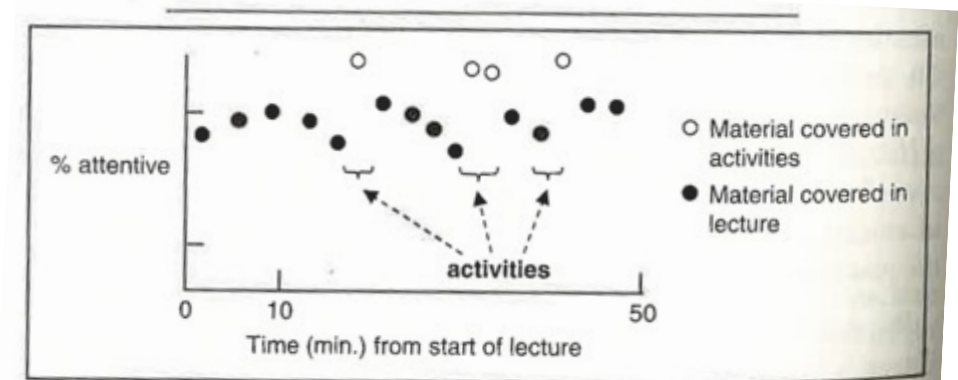


Figure 6.3-2: Attentiveness versus Time in Lecture—Activities Interspersed

# Spring 2019: Cloud-based Google Colaboratory (Jupyter Notebooks)

colab.research.google.com

## Benefits of Google Colaboratory:

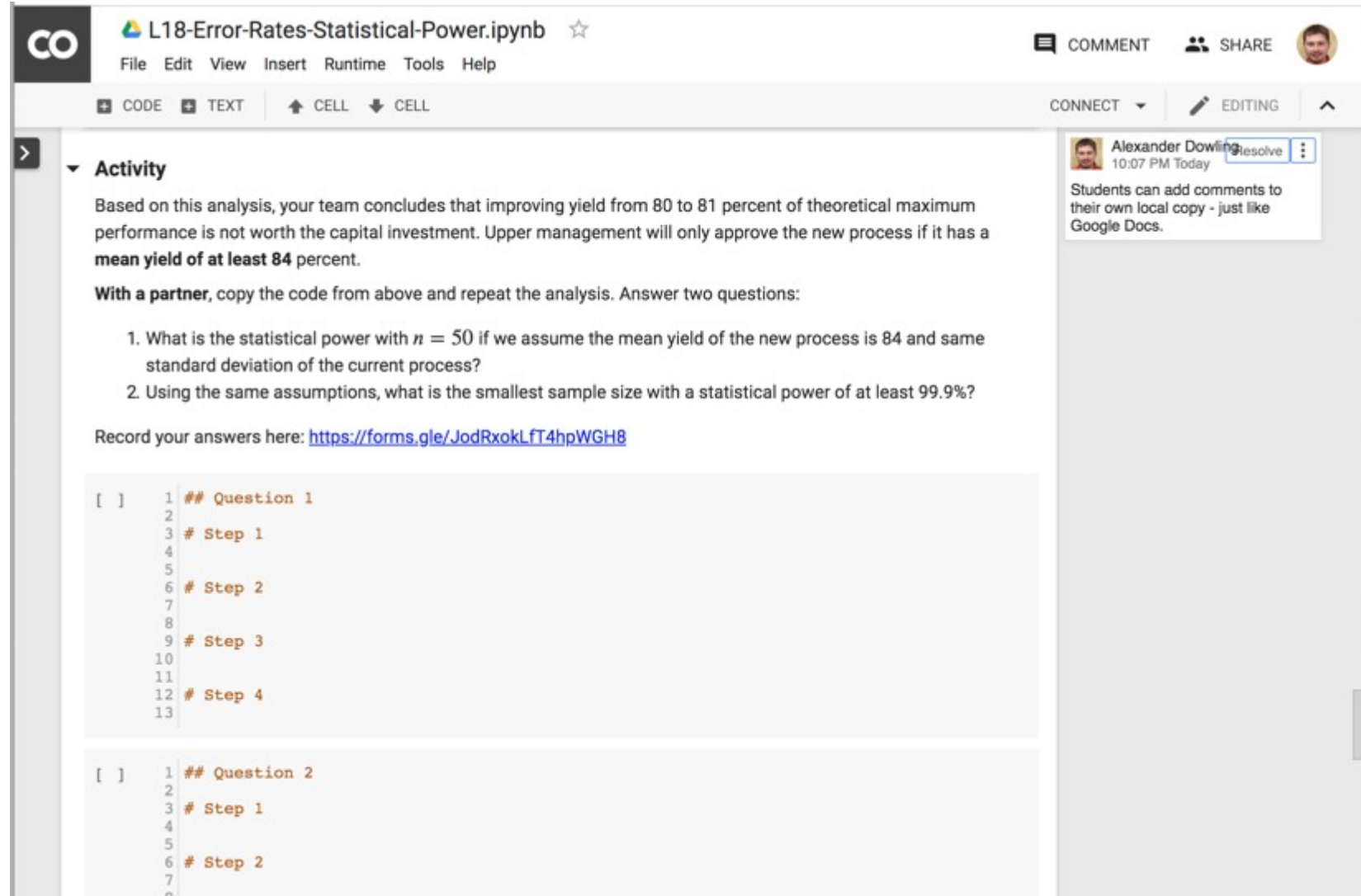
Like Google Docs, but for code

Integrated with **Google Drive**:  
automatic versioning, easy sharing

Removes barriers to access:  
students can complete assignments  
from **any internet connect computer**  
– no need to support 80+ local Python  
installations

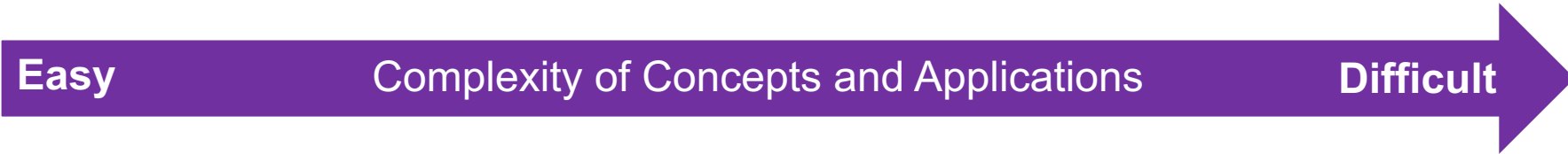
Facilitates **active learning**

Free

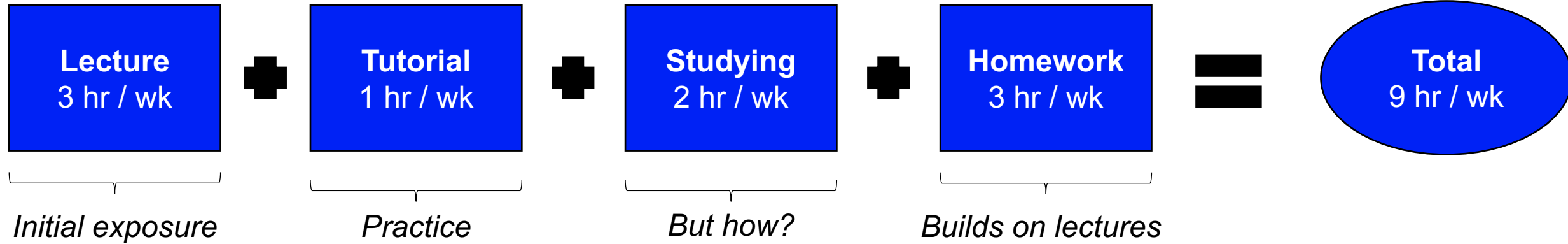


The screenshot shows a Google Colaboratory notebook titled "L18-Error-Rates-Statistical-Power.ipynb". The interface includes a top navigation bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help" menus. Below the navigation bar are tabs for "CODE", "TEXT", and "CELL". The main content area is divided into an "Activity" section and a code editor. The "Activity" section contains a paragraph of text: "Based on this analysis, your team concludes that improving yield from 80 to 81 percent of theoretical maximum performance is not worth the capital investment. Upper management will only approve the new process if it has a mean yield of at least 84 percent." This is followed by a task instruction: "With a partner, copy the code from above and repeat the analysis. Answer two questions:" and a list of two questions. The first question asks for the statistical power with  $n = 50$  given a mean yield of 84 and a standard deviation. The second question asks for the smallest sample size for a statistical power of at least 99.9%. A link to a Google Form is provided for recording answers. The code editor shows two code cells, each containing a series of numbered comments: "## Question 1", "# Step 1", "# Step 2", "# Step 3", and "# Step 4". A right-hand sidebar shows a comment from Alexander Dowling at 10:07 PM, stating that students can add comments to their local copy like Google Docs.

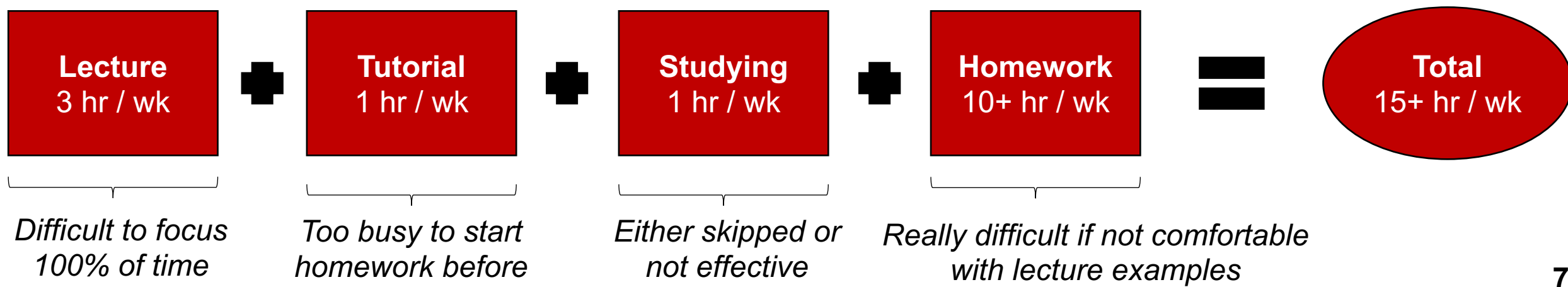
# Making your time more effective



## Traditional Class (plan)



## Traditional Class (reality for many)



# Making your time more effective

Easy

Complexity of Concepts and Applications

Difficult

*This Semester*

**Class Preparation**  
2 hr / wk

*Initial exposure at home*



**Tutorial**  
1 hr / wk

*Practice & jump-start homework*



**Studying**  
1 hr / wk

*I'll teach you how to do this & give extra practice problems*



**Class\***  
3 hr / wk

*Problem solving together*

**Homework\***  
2 - 4 hr / wk

*Easy extensions of home and class activities*



**Total**  
9 - 11 hr / wk

*This is 100% on task time... i.e., Facebook closed, not watching Netflix, not texting*

*\*We'll start some homework problems during class.*

# Fall 2019 - today: Cloud-based Vocareum (Jupyter Notebooks)

www.vocareum.com

## Benefits of Vocareum:

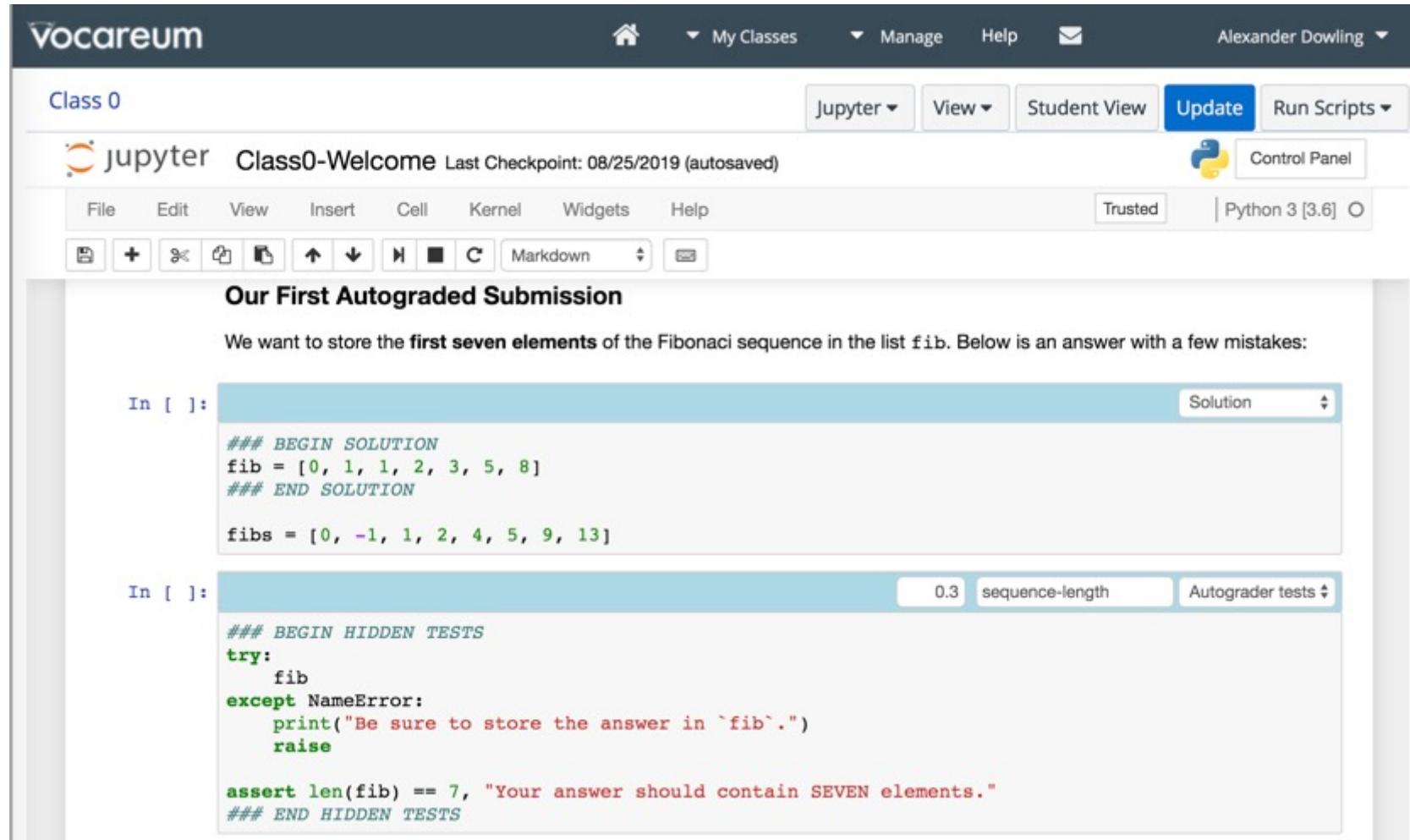
Many of the same cloud-based benefits as Colaboratory

Integrated with **Learning Management System** (e.g., Canvas) and gradebook

Supports **autograding** via nbgrader (with some enhancements)

Supports plagiarism detections (if you want it)

Paid service, but responsive technical support



The screenshot displays the Vocareum interface for a Jupyter Notebook. The top navigation bar includes the Vocareum logo, a home icon, and dropdown menus for 'My Classes', 'Manage', 'Help', and a user profile for 'Alexander Dowling'. Below this, the notebook title is 'Class 0' with buttons for 'Jupyter', 'View', 'Student View', 'Update', and 'Run Scripts'. The Jupyter logo and 'Class0-Welcome' are visible, along with the last checkpoint information: 'Last Checkpoint: 08/25/2019 (autosaved)'. A 'Control Panel' button is also present. The main menu includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. The notebook content shows a code cell titled 'Our First Autograded Submission' with the following text: 'We want to store the **first seven elements** of the Fibonacci sequence in the list `fib`. Below is an answer with a few mistakes:'. The code cell contains two parts: a solution and hidden tests. The solution code is: 

```
### BEGIN SOLUTION
fib = [0, 1, 1, 2, 3, 5, 8]
### END SOLUTION

fibs = [0, -1, 1, 2, 4, 5, 9, 13]
```

 The hidden tests code is: 

```
### BEGIN HIDDEN TESTS
try:
    fib
except NameError:
    print("Be sure to store the answer in `fib`.")
    raise

assert len(fib) == 7, "Your answer should contain SEVEN elements."
### END HIDDEN TESTS
```

 The interface also shows a 'Trusted' status and 'Python 3 [3.6]' kernel information.

# Fall 2019 - today: Cloud-based Vocareum (Jupyter Notebooks)

www.vocareum.com

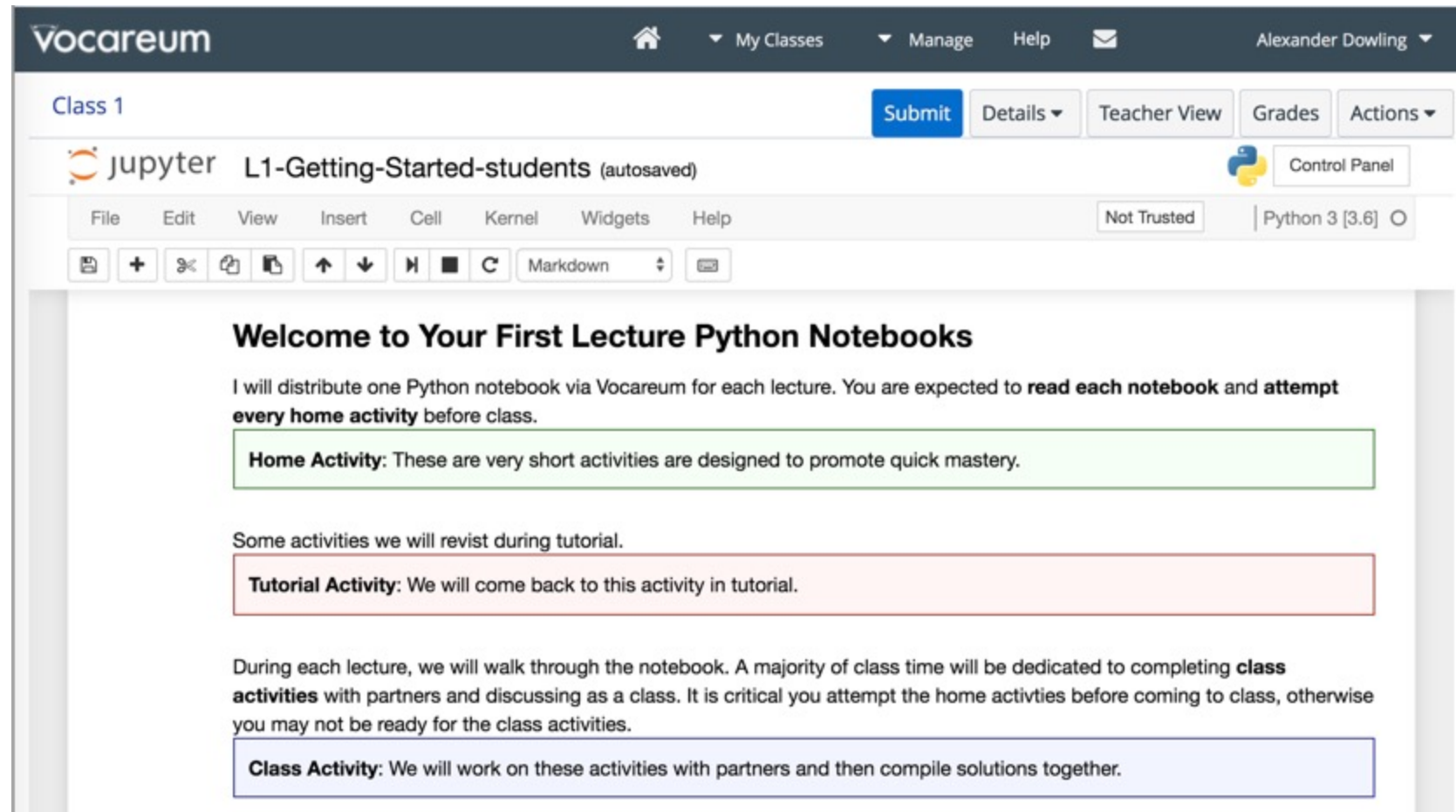
## Benefits of Vocareum:

Many of the same cloud-based benefits as Colaboratory

Integrated with **Learning Management System** (e.g., Canvas) and gradebook

Supports **autograding** via nbgrader (with some enhancements)

Supports plagiarism detections (if you want it)



The screenshot displays the Vocareum web interface. At the top, there's a navigation bar with 'Vocareum' logo, a home icon, 'My Classes', 'Manage', 'Help', and a user profile 'Alexander Dowling'. Below this, a 'Class 1' header is visible with buttons for 'Submit', 'Details', 'Teacher View', 'Grades', and 'Actions'. The main content area shows a Jupyter Notebook titled 'L1-Getting-Started-students (autosaved)'. The notebook's menu bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. A 'Control Panel' with a Python logo is on the right. The notebook content starts with a heading 'Welcome to Your First Lecture Python Notebooks' followed by a paragraph: 'I will distribute one Python notebook via Vocareum for each lecture. You are expected to **read each notebook** and **attempt every home activity** before class.' Below this are three highlighted activity boxes: a green box for 'Home Activity: These are very short activities are designed to promote quick mastery.', a red box for 'Tutorial Activity: We will come back to this activity in tutorial.', and a blue box for 'Class Activity: We will work on these activities with partners and then compile solutions together.'

**Bottom Line:** Autograder (Vocareum) enables accountability for meaningful home activities before class, which translates to more engaging class sessions.

# Canvas *Assignments* give students landing page

FA21-CBE-20258-01 > Assignments

FA21

Search for Assignment

+ Group + Assignment

0% of Total

▼ Assignments

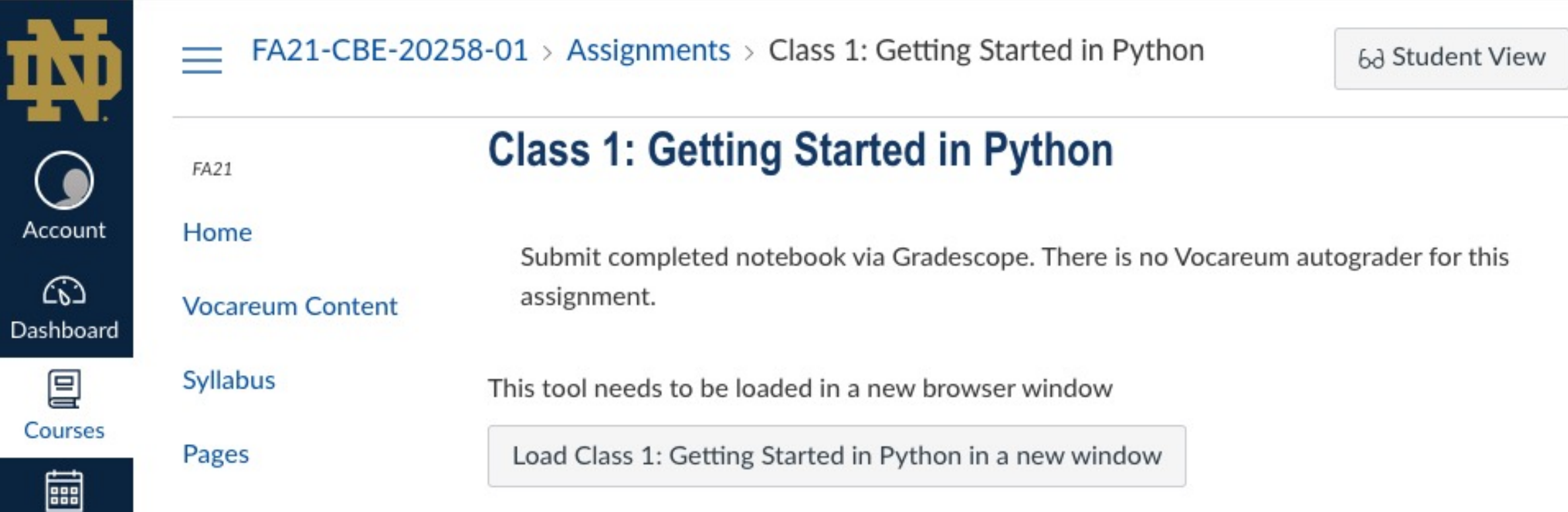
FA21 Vocareum Labs CBE-20258

2% of Total

▼ Class Participation

Tutorial 1 / Class 0: Welcome to Vocareum	Due Aug 23 at 6pm   1 pts	✓
Class 1: Getting Started in Python	Due Aug 26 at 9:25am   1 pts	✓
Class 2: Functions, Scoping, and Recursion	Due Aug 26 at 9:25am   1 pts	✓

# Each Canvas *Assignment* includes a link to Vocareum



The screenshot shows a Canvas LMS interface. On the left is a dark blue sidebar with the University of Notre Dame logo at the top, followed by icons and labels for 'Account', 'Dashboard', 'Courses', and a calendar icon. The main content area has a breadcrumb trail: 'FA21-CBE-20258-01 > Assignments > Class 1: Getting Started in Python'. A 'Student View' button is in the top right. The title 'Class 1: Getting Started in Python' is centered. Below it, the text reads: 'Submit completed notebook via Gradescope. There is no Vocareum autograder for this assignment.' A 'Vocareum Content' link is highlighted in blue. Below that, the text says: 'This tool needs to be loaded in a new browser window'. At the bottom, a button reads: 'Load Class 1: Getting Started in Python in a new window'.

# Instructor Manually Creates Each *Assignment* in Canvas

FA21-CBE-20258-01 > Assignments > Class 1: Getting Started in Python

FA21

Account

Dashboard

Courses

Calendar

Inbox

History

Commons

Help

Home

Vocareum Content

Syllabus

Pages

Announcements

Assignments

Grades

People

Files

Gradescope

Panopto Video

New Analytics

Discussions

Outcomes

Rubrics

Published

Details

Mastery Paths

Class 1: Getting Started in Python

Edit View Insert Format Tools Table

12pt Paragraph | **B** *I* U **A**

Submit completed notebook via Gradescope. There is no Vocareum autograder for this assignment.

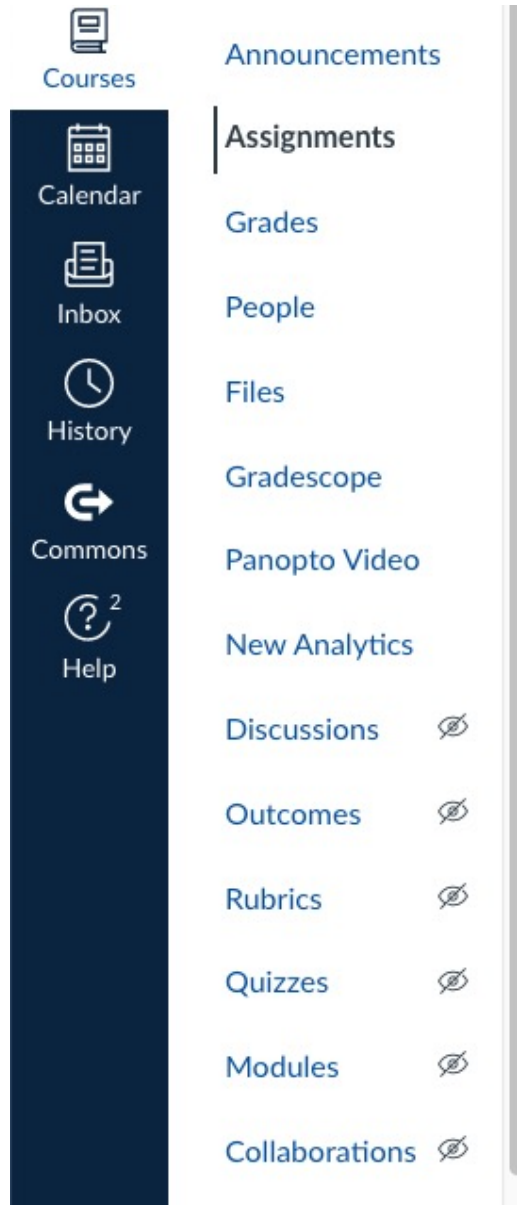
Points 1

Change title here

Write instructions here

Points in Canvas and Vocareum must match, otherwise grades will not transfer

# Instructor Manually Creates Each *Assignment* in Canvas



The sidebar shows the following navigation options:

- Courses
- Calendar
- Inbox
- History
- Commons
- Help
- Announcements
- Assignments
- Grades
- People
- Files
- Gradescope
- Panopto Video
- New Analytics
- Discussions
- Outcomes
- Rubrics
- Quizzes
- Modules
- Collaborations

Submission Type

External Tool

External Tool Options

Enter or find an External Tool URL

<https://labs.vocareum.com/lti/vclab.php> Find

Load This Tool In A New Tab

Select "External Tool"

Copy this URL from Vocareum (easy)

Check this box

Submission Attempts

Allowed Attempts

Unlimited

Assign

Assign to

Everyone

Due

Aug 26, 2021, 9:25 AM

Set due date here

# Vocareum + Gradescope for Jupyter Notebooks

Typical assignments require three submissions:

1. Vocareum (autograder)
2. Gradescope Notebook
3. Gradescope Written

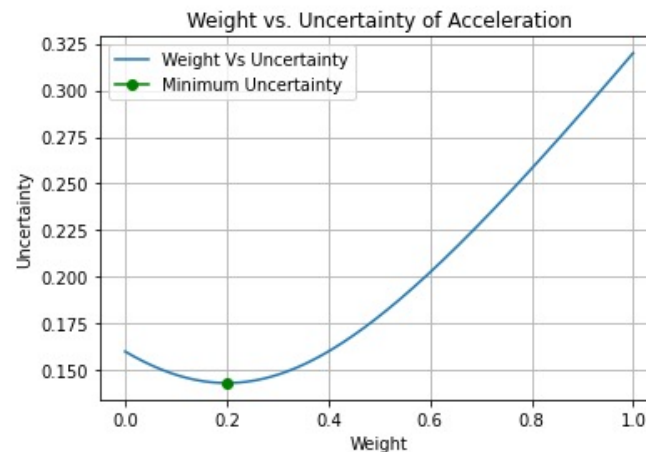
```
U[i] = math.sqrt((w[i]**2)*(U_A1**2) + ((1-w[i])**2)*(U_A2**2))

#plot w vs uncertainty
plt.plot(w,U, label = 'Weight Vs Uncertainty')
plt.plot(w[np.argmin(U)],np.min(U), marker = 'o',
color = 'green', label = 'Minimum Uncertainty')
plt.xlabel('Weight')
plt.ylabel('Uncertainty')
plt.title('Weight vs. Uncertainty of Acceleration')
plt.legend()
plt.grid(True)
plt.show

# Find the minimum uncertainty
index = np.argmin(U)
weight = w[index]

#Calculate Acceleration
A3 = round(weight*A1 + (1-weight)*A2,2)

#Calculate Uncertainty
U_A3 = round(math.sqrt((weight**2)*(U_A1**2) + ((1-weight)**2)*(U_A2**2)),2)
```



2:1c ▾

13 OF 13 GRADED

TOTAL POINTS

**0.5 / 0.5 pts**

[Rubric Settings](#)

[Collapse View](#)

**1** -0.0

Correct

[+ Add Rubric Item](#)

[Create Group](#)

[Import...](#)

**SUBMISSION SPECIFIC ADJUSTMENTS**

Point Adjustment

**APPLY PREVIOUSLY USED COMMENTS**

# nbpages + Google Colab

## CBE60499

Nonlinear and Stochastic Optimization. <https://ndcbe.github.io/CBE60499/>

[View the Project on GitHub](#) ndcbe/CBE60499

## CBE60499

[Table of Contents](#)

[Data Index](#)

[Figure Index](#)

[Python Module Index](#)

[Tag Index](#)

### Chapter 1.0 Getting Started with Pyomo

- [1.1 60 Minutes to Pyomo: An Energy Storage Model Predictive Control Example](#)
- [1.2 Pyomo Mini-Project: Receding Horizon Stochastic Control](#)

### Chapter 2.0 Optimization Modeling with Applications

*This notebook contains material from [CBE60499](#); content is available [on Github](#).*

[< 1.1 60 Minutes to Pyomo: An Energy Storage Model Predictive Control Example](#) | [Contents](#) | [Tag Index](#) | [2.0 Optimization Modeling with Applications](#) >

[Open in Colab](#) [Github](#) [Download](#)

In [ ]:

```
# IMPORT DATA FILES USED BY THIS NOTEBOOK
import os, requests

file_links = [("data/Prices_DAM_ALTA2G_7_B1.csv", "https://ndcbe.github.io/CBE60499/data/Prices_DAM_ALTA2G_7_B1.csv")]

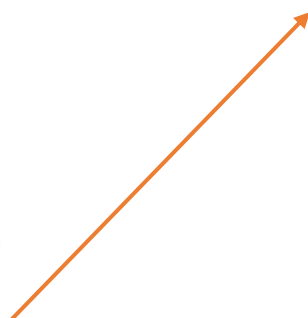
# This cell has been added by nbpages. Run this cell to download data files re

for filepath, fileurl in file_links:
    stem, filename = os.path.split(filepath)
    if stem:
        if not os.path.exists(stem):
            os.mkdir(stem)
    if not os.path.isfile(filepath):
        with open(filepath, 'wb') as f:
            response = requests.get(fileurl)
            f.write(response.content)
```

## 1.2 Pyomo Mini-Project: Receding Horizon Stochastic Control

**Deadline:** Friday, March 5, 2021

### 1.2.1 Assignment Goals



# Which platform?



- Closed ecosystem, requires authentication (e.g., Canvas)
- More effort for students to access in future
- + Easy to control access
  
- + **Autograder** with Canvas integration
  
- More effort to setup/manage
  
- + Responsive tech support
  
- + Easy to access all versions of student submissions

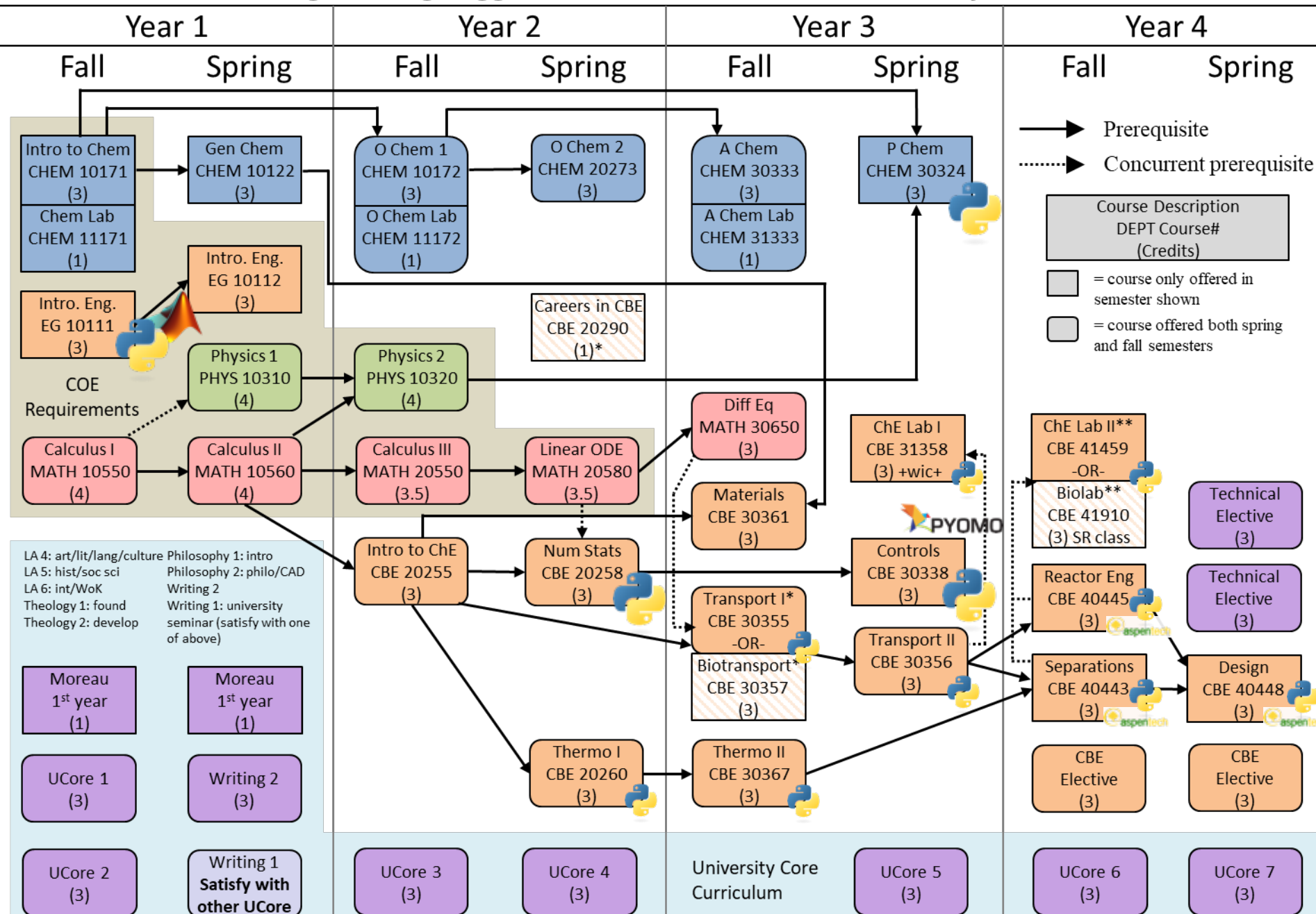
*Great for computing focused undergraduate classes*



- Sharing via Google Drive or website (nbpages)
- + Easy to disseminate
- Limited control over access
  
- Only manual grading via Gradescope
  
- + Easier to setup
  
- On your own, fingers crossed Google does not end support for Colab ;)
  
- + Google Drive automatically stores ~100 versions history for students

*Great for graduate classes and occasional class assignments/examples*

# Chemical Engineering Suggested 4 Year Curriculum University of Notre Dame



## Vision

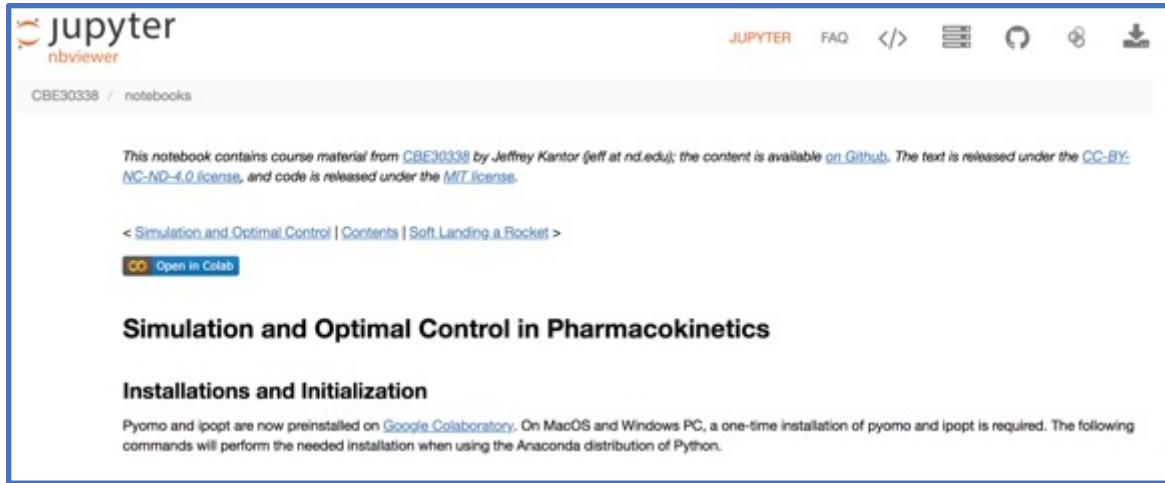
*Vertically integrate computing and statistics throughout the undergraduate curriculum*

## Library of Cloud-based Jupyter Notebooks

*Complement existing core CBE classes with examples that use computing and statistics for problem solving*



# Special Thanks



<https://github.com/jckantor>

Chemical Process Control  
Introduction to Chemical Engineering Analysis  
Introduction to Operations Research  
Process Operations

**Prof. Jeff Kantor**



**Prof. Yamil Colón**



**Vocareum Pilot**

**Pat Miller**  
**Xiaojing Duan**

**Kaneb Center**

**Kevin Barry**  
**Dan Hubert**  
**Kristi Rudenga**

# Rethinking Computing Education with Vocareum and Canvas



**Prof. Alexander (Alex) Dowling**

**adowling@nd.edu    dowlinglab.nd.edu**

Department of Chemical and Biomolecular Engineering

University of Notre Dame

November 18, 2021

**colab**

**vocareum**

