Calculus Unraveled

Intuition, Proofs, and Python

Dr. Mike X Cohen

0.1 Front matter

This page contains some important details about the book that basically no one reads but somehow is always in the first page.

ISBN: 9798309397228, edition 1.

This book was written and formatted in LaTeX by Mike X Cohen.

0.2 Book cover

The cover of this book was designed by Yuva Oz (art-4-science.com).

0.3 Dedication

If you're reading this, then the book is dedicated to you. I wrote this book for *you*. Now turn the page and start learning calculus!

0.4 Forward

The past is immutable and the present is fleeting. Forward is the only direction.

Contents

Chapter 1

Introduction to this book

1.1 What is calculus and why learn it?

Calculus is the study of *change*. In particular, the idea of calculus is to use simple mathematical techniques (mostly differentiation and integration, which are just fancy ways of expressing subtraction and addition) to understand the complicated behavior of mathematical functions (a function is a set of rules that governs how numbers behave).

I learned, forgot, and re-learned calculus several times in my life. I consider this an advantage in teaching calculus, because each time I re-learned it, I saw calculus from a different perspective and with a new appreciation for its conceptual simplicity, beauty, and occasional surprising conclusions. I am reasonably confident that you will also learn and forget calculus (perhaps you already have). That's just how it goes.

There are many reasons why someone (e.g., *you*) would want to learn calculus. Here's a non-exhaustive list:

1. You want to understand other topics that have calculus as a prerequisite, such as physics, engineering, computational biology, finance, or myriad other technical subjects.
2. You are a practicing or aspiring data scientist and want a deeper understanding of statistical and machine-learning algorithms.
3. You like to challenge yourself and learn math as a hobby (yes, this really is true for lots of people!).
4. You don't actually *want* to learn calculus; it's required for your degree or educational program. There's nothing wrong with this motivation!

1.2 Why learn calculus from this book?

There are many calculus textbooks — a few of which are excellent, while many others are either impenetrable or are irrelevant to modern computer-based applications.

Here's the point: If you want a calculus book like any other calculus book, then buy any other calculus book.

I've tried to organize and write this book to be different and unique, to provide a learning approach and perspective that is rigorous but approachable, intuitive yet practical, blending theory with hands-on coding and problem-solving techniques. You will solve exercises, create high-quality graphics, and code algorithms. I believe this will help you understand modern, real-world applications of calculus without getting bogged down in endless repetitive equations to solve.

Most importantly, I use Python code to help you understand concepts in math, because *you can learn a lot of math with a bit of coding*. Many more people in the world have coding experience than have formal math training. In over 20 years of teaching, I have witnessed uncountable situations where learners can't make sense of equations — until they implement them in code, make graphs, change parameters, and solve code-based exercises.

1.3 Target audience

I envision three types of people who would benefit from this book.

1. Students using this book to help them learn calculus. Perhaps your assigned calculus book is overly stuffy or leaves everything "as an exercise to the reader."
2. Professionals in fields related to the quantitative sciences, including data science, engineering, physics, and computational biology, who want to understand the analyses they are applying, develop new algorithms, or advance in their career by increasing their math and math-coding knowledge.
3. Math enthusiasts. You might be surprised how many people choose to learn math in their free time, simply as an enjoyable way to pass the time while increasing mental acuity.

1.4 Prerequisites

The obvious You need to *want* to learn calculus. You need an intention and a goal. Maybe that goal is to pass your university calculus exam, maybe that goal is to get a data-science job, maybe that goal is to understand machine-learning algorithms, or maybe your goal is simply to have an intellectually stimulating hobby. It doesn't matter what your goal is (it is, after all, *your* goal).

I hope that having an intention to learn is almost enough for you to succeed in learning calculus from this book. All the points below are minor in comparison to the importance of having a reason to make even a small but consistent effort to learn a new skill.

High-school math You need to be comfortable with arithmetic and basic algebra. Can you solve for x in $4x^2 = 9$? Then you have enough algebra knowledge to continue. Other concepts will be introduced as the need arises.

There are formulas, equations, and algorithms in this book, but I try to explain them in plain English, illustrate them with graphs and diagrams, and provide you with Python code so you can explore the math through simulations and visualizations.

Programming It is no understatement to write that modern applied mathematics relies 100% on programming. There is simply no way to implement math without knowing at least a little bit of coding.

Fortunately, there are well-developed coding libraries that implement the low-level details. This means that you don't need to be a professional programmer to understand all the code in this book. But you do need to be comfortable with coding.

In this book, I use Python, because it is one of the most popular languages for modern applied math (statistics, machine-learning, deep learning, etc). I'm pretty sure it won't *always* be so popular; some other language will be developed that is better, easier, and faster[1]. But the good news is that all programming languages share some similarities, so learning calculus in Python will help you apply math in any other language. In other

[1] Julia Programming Language looks promising, for example.

words, learning coding is time *invested*, not time *wasted*, even if you use a different language in practice. Anyway, ChatGPT (or other advanced language AI) can translate Python code into R, MATLAB, Julia, or other languages with decent accuracy.

To be clear: You do not *need* any coding to work through this book. You can skip all the code and all the coding exercises and focus on the conceptual and interpretational topics. But I designed this book such that a deep understanding of the material comes from working through the code and code exercises.

People learn best by seeing and doing, rather than by letting their eyes bounce over words and equations. This is a guiding philosophy of my writing and my teaching.

1.4.1 Brand new to Python?

I assume some basic Python proficiency in this book. If you're brand-new to Python, then you should probably wait to delve into this book until after taking an intro-Python course or book. There are myriad free and paid resources available for learning Python.

I tried to keep the code fairly simple in this book, while being advanced enough to express the nuanced principles of calculus. The point is to use Python as a tool to help you learn and understand calculus, not to use gratuitously fancy and dense code. And if you want to challenge yourself, then please embellish my code! You should see my code as a starting point for your own explorations and applications, not as immutable text with no room for customization.

1.5 Exercises

I am sure you have heard this before: *Math is not a spectator sport.* If you simply read this book without solving any exercises, then sure, you'll learn something and I hope you find the book useful. But to really understand calculus, you need to solve calculus problems.

I designed the exercises to require some effort and creativity. They can

be solved only through coding. These are opportunities for you to explore concepts, visualizations, and parameters in ways that are difficult or impossible to do without code.

I very strongly encourage you to work through the exercises[2]. They are not just busy work; they are ways to solidify, explore, and expand your understanding of calculus in ways that are not possible from only reading the chapters. The exercises also provide a wealth of code that you can use to continue learning and applying calculus concepts.

I provide my code solutions to all exercises, but keep in mind that there are many correct coding solutions; the point is for you to explore and understand calculus using code, not to reproduce my code exactly.

Feel free to look at my code solutions as much as you like. It's not cheating! This is especially true if you are new to Python, and if you understand the calculus concepts but struggle with the coding syntax. The purpose of the exercises is to give you opportunities to learn, explore, and expand your knowledge; they are not meant to be quantitative assessments.

In addition to my coding solutions, I have also created online videos in which I explain the Python code and solutions. You can find those videos on my YouTube channel www.youtube.com/@mikexcohen1, or see the online code on github for a direct link.
`https://github.com/mikexcohen/calculus_book`

1.6 Using the code with this book

You can use any IDE that you find most comfortable. I wrote all the Python code using Google's Colab service, which is free and integrates well with their other products, including Google Drive. Therefore, I recommend using Colab to follow along in this book, especially if you are relatively new to Python. The main libraries I use are `numpy`, `scipy`, `sympy`, and `matplotlib`.

Getting the book code into Google Colab involves three steps: download the code from github, upload the code to Google Drive, open the code files in Google Colab.

[2] I mean, like, really super-duper a lot encourage this.

Download the code The code for this book is available at https://github.com/mikexcohen/Calculus_book.
If you are comfortable with git, then you can clone this repository to sync the files locally. If you have no idea what that previous sentence means, then don't worry! You can get all the code without knowing anything about git, and without needing to log in, sign up, give an email address, pay, or anything else.

Simply go to that URL, look for the green button that says "Code", click on that button, and look for the link that says "Download zip" (see Figure 1.1A). This will download one zip file that contains all the code material that you will need for this entire book.

Unpack that zip file on your computer.

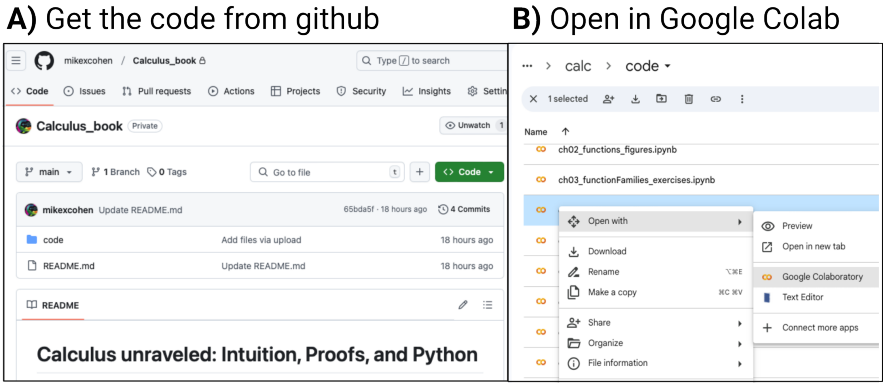

Figure 1.1: Screenshots of getting the book code from my github to your Google-colab. It might look slightly different on your computer.

Upload the code to Google Drive Google Drive is a free cloud-based storage service that Google provides to its users. You do not need to pay to use Drive, but you do need a Google account.

In a browser, go to drive.google.com (log in to your Google account if you're not already logged in). Create a folder for this book. Find the files you downloaded to your local computer, and simply select and drag them into the Drive folder to upload the files. Note that Drive does not unpack zip files, so you'll need to unzip the files on your local computer first.

Now those files are stored in the cloud. That's convenient because you can access them from any internet-connected computer. In fact, you can delete the files from your local computer if they bother you for some reason.

Open the files in Colab Select a Python notebook file (they end with extension ".ipynb"), and either double-click the file or right-click and select Open With, Google Colaboratory (Figure 1.1B). It will open a new tab with the notebook file. If you have absolutely no idea what you are looking at, then you need to learn some Python before continuing with this book. If you are familiar with Python but are new to Colab, then you can consider watching a YouTube video about Colab to become familiar with the environment. But the Colab interface is easy and won't take long to master.

It is also possible to import the files directly from github onto Colab via the File, Open menu options. But this imports only one file at a time, so I think it's easier to upload them all at once to your Drive.

(A note on paying for Google services: You can pay to upgrade your Drive storage limit and/or to get better access to Colab servers. Paying for these services is absolutely not necessary for this book, but something you might consider if you anticipate heavy usage for other applications.)

I recommend creating a copy of my code to modify. That way, you have the original version of my code, and you can feel comfortable making whatever changes you like to your version. Of course, you can always download a fresh version from my github repository.

Using a different Python IDE Using Google's Colab service will help ensure that you can reproduce all the results and figures in this book. Therefore, I recommend using Colab.

You can use any other IDE, on a cloud or on your local computer. But keep in mind that some things in Python are IDE- and version-dependent. It is possible that you will need to make some modifications to my code, for example for visualizations and LaTeX formatting in `sympy` output. Please understand that I cannot provide detailed support for difficulties or errors encountered when using the code outside of Colab.

1.6.1 Code organization

Each chapter has two Python notebook files: one file to create the figures in the text, and one file that provides full solutions to all of the exercises. (Some figures appear in the text but are created as part of the exercises; these instances are indicated in the code files with comments.)

The file naming is by chapter number, chapter label, and file contents. For example, here are the two file names for Chapter 4:
`ch04_limits_figures.ipynb`
`ch04_limits_exercises.ipynb`

Within each file, there is a subheading for each figure or exercise. A few figures in the book were made in Inkscape; these do not have corresponding code cells.

1.6.2 Modifying and reposting my code

Modifying my code Yes, please do! I very much hope that you see my code not as immutable text, but instead as a source of inspiration for you to modify, adapt, and explore.

Reposting my code I am occasionally asked whether I allow people to post my code, or modifications to it, on websites, blogs, github, or the like. My answer is *yes!* Absolutely, please feel free to use and share my code as you like. I ask only that you cite the source at the top of the code file. You can include the url to the book on github or the link to the book on amazon.com or wherever you purchased it. It can be something as simple as:

```
This code is modified from Mike X Cohen's book
on calculus; for code and links to the book, see
https://github.com/mikexcohen/calculus_book
```

Using other computer languages I have zero doubt that all exercises and visualizations can be solved using MATLAB, R, Julia, Mathematica, C++, or any other numerical processing language. If you want to use another language, then that's great! But I cannot support questions or difficulties for other languages.

If you would like to translate all of the book code into a different language, please contact me. I would be happy to link to your github page from mine. At the time of this writing, ChatGPT is impressive, but imperfect in translating a large amount of code. Any AI-translated code would need to be thoroughly checked by a human expert.

1.7 Online resources

Online course This book is based on online courses that I created. The book and the courses are similar, but not redundant. You don't need to enroll in the online course to follow along with this book, or the other way around.

Some people prefer to learn from online videos while others prefer to learn from textbooks. I try to cater to both types of learners. Following both may be beneficial, but I want to make it clear that the two resources are independent, and it is not my intention to upsell you.

You can find a list of all my online courses at sincxpress.com.

This online course is separate from the free video explanations of the exercises that accompany this book. Those videos are designed for this book and are available on my YouTube channel or via a direct link from the github repository that contains the code for this book.

Searching for explanations Although I have tried to write this book as a self-contained resource to master calculus, it is naive to think that everyone will find it the perfect resource that I intend it to be. Everyone learns differently, and everyone has a different way of understanding mathematical concepts.

If you struggle to understand something, don't jump to the conclusion that you aren't smart enough to understand it; a simpler possibility is that the explanation I find intuitive is not the explanation that you find intuitive. I try to give several explanations of the same concepts, in hopes that you'll find traction with at least one of them. If none of those works for you, don't hesitate to search the Internet or other textbooks if you need different or alternative explanations.

Online code The book itself has only the occasional snippet of code. Instead, *all* of the code — including the code to create the figures in the book and the solutions to the exercises — is available at
`https://github.com/mikexcohen/calculus_book`

The best way to learn from this book is to have the code in front of you while reading the book. You can see how the statistics concepts are

implemented and how I created the figures. You can adjust the code to explore the concepts, and you can compare your exercise solutions with mine.

I apologize for stating multiple times that the code is available online and not entirely printed in the book, but I have had quite a few experiences of people complaining that I don't make my code available or that my code is incomplete — even giving my books and courses poor ratings online. I'm not angry about it, because I know that those people were just so excited to learn that they rushed through the introductory material (can you blame them??).

1.8 AI assistance

I wrote this book in 2024, when large language models like ChatGPT were becoming very popular and very hyped. It is impossible to predict how technology will develop and impact life in the future; but deep, complex language models like ChatGPT, Claude, and Gemini have the potential to have a significant impact on the way we write and learn.

I spent a lot of time exploring ChatGPT (models 3.5, and 4/o), and considering whether it could be useful for my books. ChatGPT is a remarkably good writer, but there were three limitations that prevented me from incorporating it into my writing (this book and my other books and courses): First, several explanations were incorrect, although the writing style was so fluid and authoritative that it would be easy to be fooled. Second, I found ChatGPT4's writing style to be stuffy, overly formal, and sesquipedalian[3], like a gifted teenager who consults his thesaurus too often. Third, ChatGPT often equivocates and hedges, refusing to state that something is correct or incorrect, right or wrong, or appropriate or inappropriate.

There were times that I struggled with phrasing a sentence, and which I asked ChatGPT4 to rewrite. In none of those cases did I take ChatGPT's suggestions verbatim (I have tried and failed repeatedly to get ChatGPT to emulate my writing style), but in all of those cases ChatGPT helped me improve the readability of those sentences.

[3] Yes, I had to look up this word; I love the meaning but never remember the word itself.

Most of the AI assistance in creating this book came from me asking ChatGPT to help implement some aspects of coding, including details of visualizations, debugging, and styling LaTeX. Interestingly, ChatGPT often failed to provide a correct solution, but usually gave good tips that helped me find the right solution.

Anyway, the point is that on the cusp of what may be the "era of AI writing," I can assure you that this book is fully human-written, with the occasional assistance from ChatGPT-4 to smooth out clunky sentences and help with coding issues.