**CS exercise as a class, coefficients, covariates:**

<https://docs.google.com/spreadsheets/d/1dQ6Cx7FGLpA5IqK6Pkpaa1FtLko684oDYNq4AFuIYMw/edit?usp=sharing>

**ChatGPT Prompt:**

I've seen this ery cool graph where people will run a lot of different specifications on the same question in the same dataset, where maybe the specifications are this or that fixed effet or this or that control and there might be like 50 of them.  And they will produce a line, and all the coefficients with their confidence intervals (upp and lower) are on the line.  ANd maybe a mean is shown, or maybe a color for the statistically significant ones is slightly different.  I'd like you to make thatin python.

Student Simple coefficient SE Lower 95% CI Upper 95% CI

Scott 0.1249335 0.0438772 0.0389358 0.2109311

Joseph 0.1230617 0.0399419 0.044777 0.2013463

Doreen 0.1084573 0.0386034 0.0327961 0.1841186

Urvi -0.2362015 0.1556495 -0.541269 0.0688659

Godwin 0.0865534 0.0544744 -0.0202145 0.1933212

Nuzhat 0.1010415 0.0317243 0.0388629 0.16322

Zhou 0.1103233 0.0377639 0.0363073 0.1843392

Kabir 0.1287336 0.0376312 0.0549777 0.2024894

Meghana 0.1144449 0.0637527 -0.010508 0.2393979

bond 0.098682 0.046948 0.0066655 0.1906984

Altaf 0.1233474 0.0391245 0.0466647 0.20003

mo 0.083183 0.0472836 -0.0094912 0.1758571

dehua 0.1212049 0.0408121 0.0412146 0.2011953

Mo2 0.1377255 0.0756722 -0.0105893 0.2860403

**A graph with lines and dots

AI-generated content may be incorrect.**

# Fix: Convert color values to a list instead of a pandas Series

colors\_list = df["Significant"].map({True: "green", False: "gray"}).tolist()

# Create the plot again

plt.figure(figsize=(10, 8))

# Plot each coefficient with its CI individually for full control over color

for i, row in df.iterrows():

    plt.errorbar(

        x=row["Estimate"], y=i,

        xerr=[[row["Estimate"] - row["Lower\_CI"]], [row["Upper\_CI"] - row["Estimate"]]],

        fmt='o', color='black', ecolor=colors\_list[i], elinewidth=2, capsize=4

    )

plt.yticks(df.index, df["Student"])

plt.axvline(0, color='red', linestyle='--', linewidth=1)

plt.title("Specification Curve: CS ATT Estimates with 95% CI", fontsize=14)

plt.xlabel("Estimate")

plt.ylabel("Specification (Student)")

plt.grid(True, axis='x')

plt.tight\_layout()

plt.show()

**ChatGPT Prompt:**Hey could you use python and put this into python and make a kernel density of these 11 coefficients. Title the figure "Distribution of CS ATT Estimates" and take the mean of them, put that somewhere labeled, take the standard deviation of them, put that somewhere.

Scott 0.1249335

Joseph 0.1230617

Doreen 0.1084573

Urvi -0.2362015

Godwin 0.0865534

Nuzhat 0.1010415

Zhou 0.1103233

Kabir 0.1287336

Meghana 0.1144449

bond 0.098682

Altaf 0.1233474

mo 0.083183

dehua 0.1212049

Python code:

import matplotlib.pyplot as plt

import numpy as np

from scipy.stats import gaussian\_kde

# Re-defining the ATT estimates from the list

att\_values = np.array([

    0.1249335, 0.1230617, 0.1084573, -0.2362015, 0.0865534,

    0.1010415, 0.1103233, 0.1287336, 0.1144449, 0.098682,

    0.1233474, 0.083183, 0.1212049

])

# Calculate KDE

kde = gaussian\_kde(att\_values)

x\_vals = np.linspace(min(att\_values) - 0.1, max(att\_values) + 0.1, 1000)

y\_vals = kde(x\_vals)

# Calculate mean and std

mean\_val = np.mean(att\_values)

std\_val = np.std(att\_values)

# Plot

plt.figure(figsize=(10, 6))

plt.plot(x\_vals, y\_vals, color='skyblue', linewidth=3)

plt.fill\_between(x\_vals, y\_vals, color='skyblue', alpha=0.4)

# Mean and SD lines

plt.axvline(mean\_val, color='red', linestyle='--', label=f"Mean = {mean\_val:.3f}")

plt.text(mean\_val, max(y\_vals) \* 0.8, f"Mean = {mean\_val:.3f}", color='red', ha='center')

plt.text(mean\_val, max(y\_vals) \* 0.7, f"SD = {std\_val:.3f}", color='black', ha='center')

plt.title("Distribution of CS ATT Estimates", fontsize=16)

plt.xlabel("ATT Estimate")

plt.ylabel("Density")

plt.grid(True)

plt.legend()

plt.tight\_layout()

plt.show()

A graph of a distribution of cs

AI-generated content may be incorrect.