

# Using Seaborn Styles

INTERMEDIATE DATA VISUALIZATION WITH SEABORN



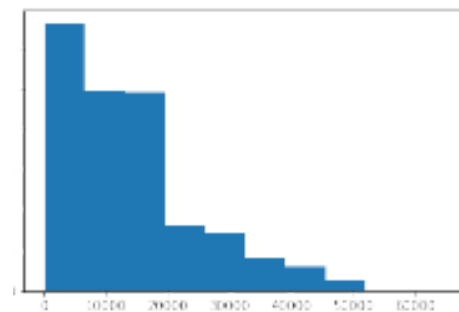
**Chris Moffitt**  
Instructor

# Setting Styles

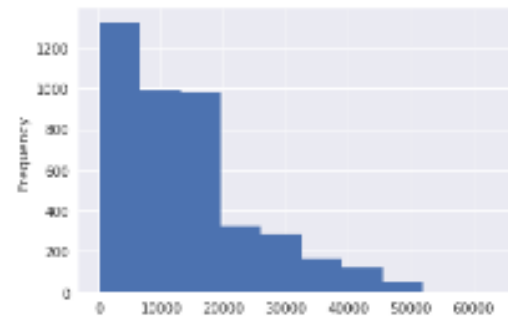
- Seaborn has default configurations that can be applied with `sns.set()`
- These styles can override matplotlib and pandas plots as well

```
sns.set()  
df['Tuition'].plot.hist()
```

Pandas histogram



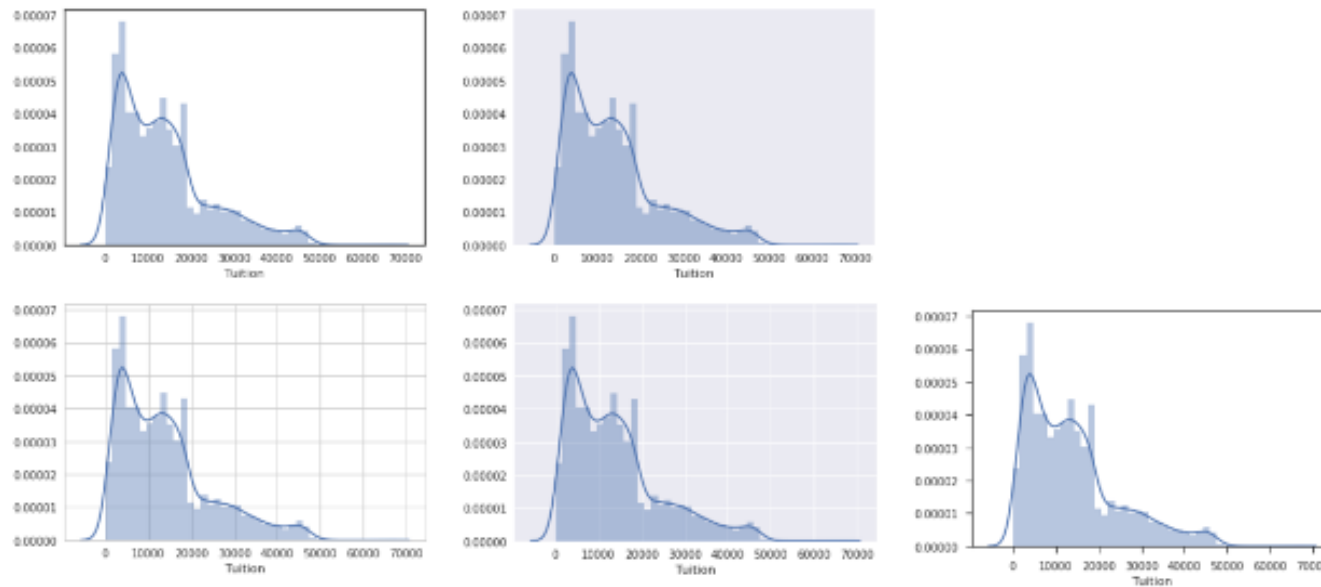
default



Seaborn style

# Theme examples with sns.set\_style()

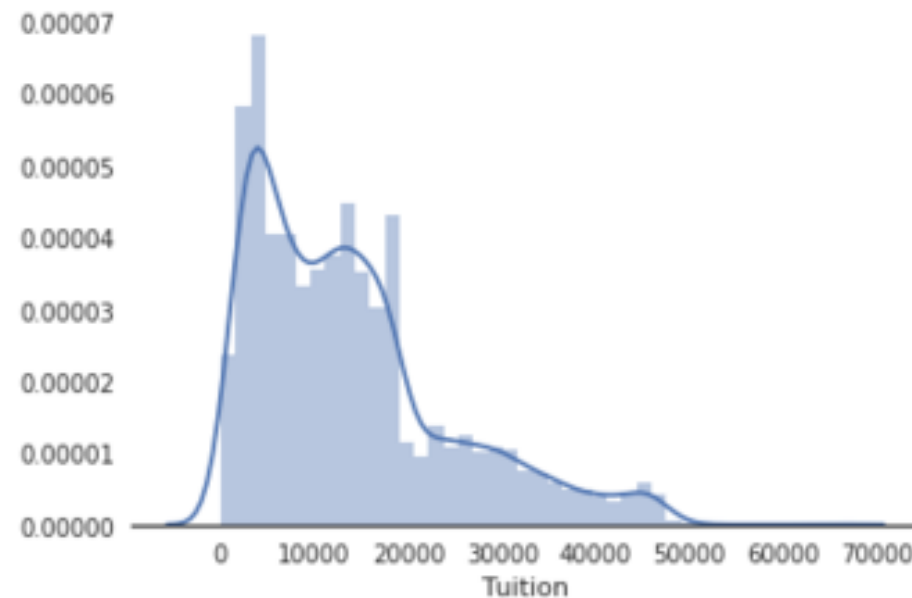
```
for style in ['white', 'dark', 'whitegrid', 'darkgrid',  
             'ticks']:  
    sns.set_style(style)  
    sns.distplot(df['Tuition'])  
    plt.show()
```



# Removing axes with `despine()`

- Sometimes plots are improved by removing elements
- Seaborn contains a shortcut for removing the spines of a plot

```
sns.set_style('white')  
sns.distplot(df['Tuition'])  
sns.despine(left=True)
```



# Let's practice!

INTERMEDIATE DATA VISUALIZATION WITH SEABORN

# Colors in Seaborn

INTERMEDIATE DATA VISUALIZATION WITH SEABORN

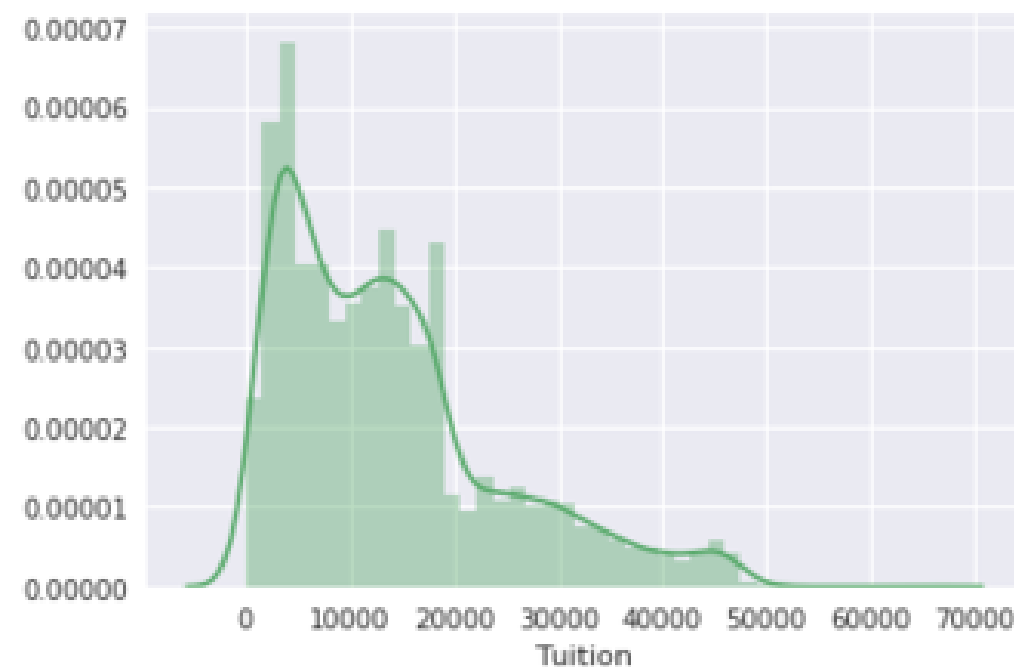


**Chris Moffitt**  
Instructor

# Defining a color for a plot

- Seaborn supports assigning colors to plots using `matplotlib` color codes

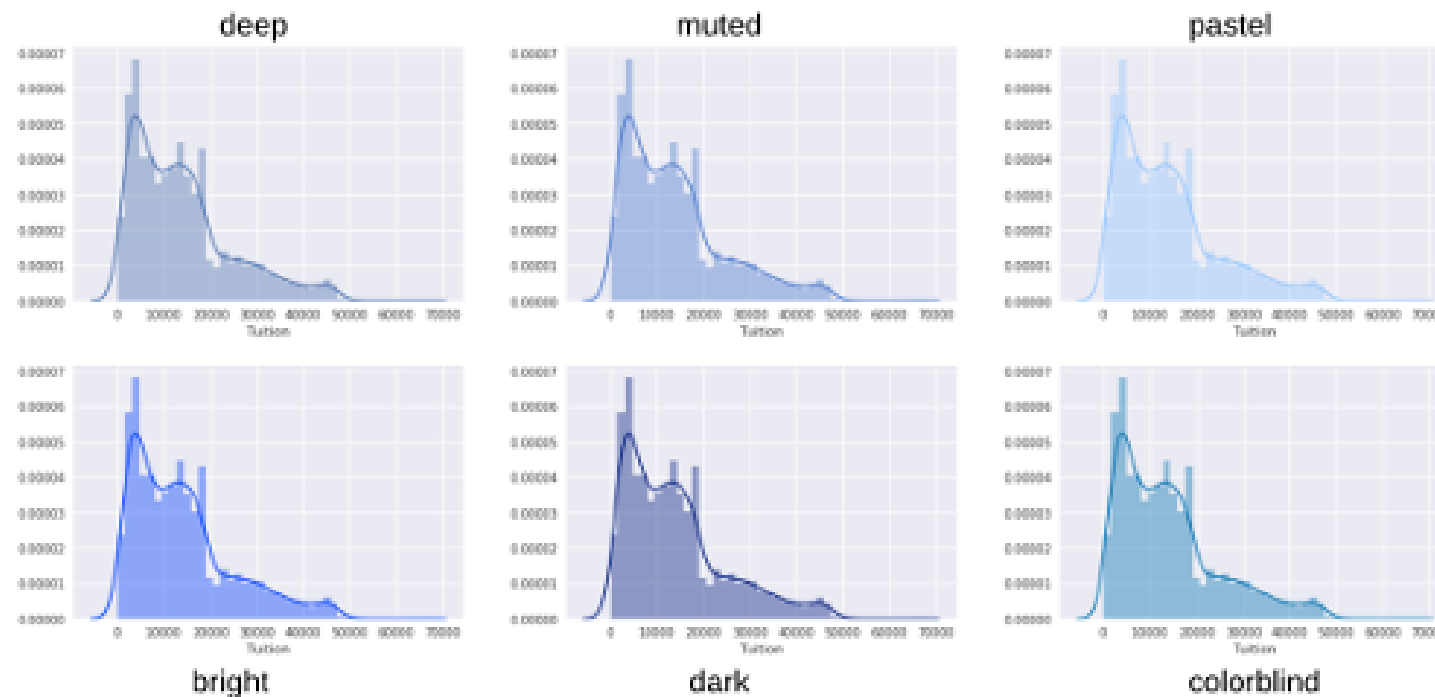
```
sns.set(color_codes=True)  
sns.distplot(df['Tuition'], color='g')
```



# Palettes

- Seaborn uses the `set_palette()` function to define a palette

```
for p in sns.palettes.SEABORN_PALETTES:  
    sns.set_palette(p)  
    sns.distplot(df['Tuition'])
```





# Displaying Palettes

- `sns.palplot()` function displays a palette
- `sns.color_palette()` returns the current palette

```
for p in sns.palettes.SEABORN_PALETTES:  
    sns.set_palette(p)  
    sns.palplot(sns.color_palette())  
plt.show()
```



# Defining Custom Palettes

- Circular colors = when the data is not ordered

```
sns.palplot(sns.color_palette(  
    "Paired", 12))
```



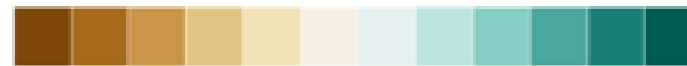
- Sequential colors = when the data has a consistent range from high to low

```
sns.palplot(sns.color_palette(  
    "Blues", 12))
```



- Diverging colors = when both the low and high values are interesting

```
sns.palplot(sns.color_palette(  
    "BrBG", 12))
```



# Let's practice!

INTERMEDIATE DATA VISUALIZATION WITH SEABORN

# Customizing with matplotlib

INTERMEDIATE DATA VISUALIZATION WITH SEABORN

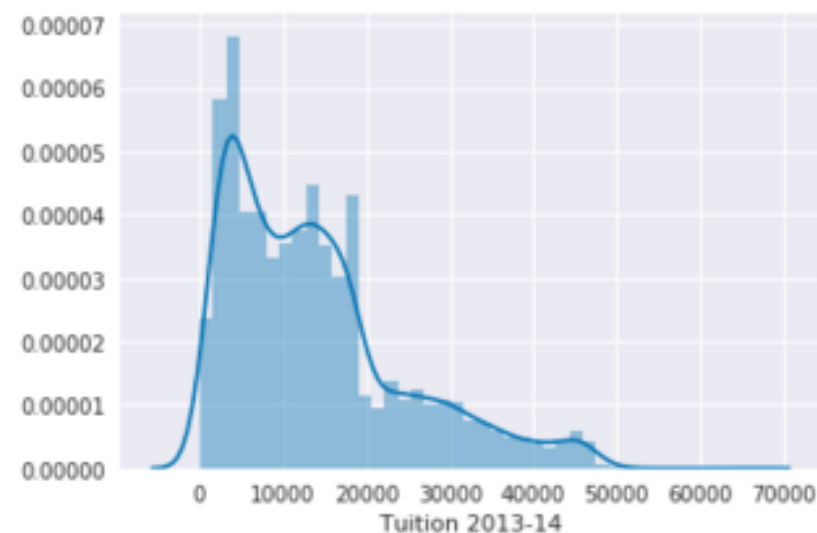


**Chris Moffitt**  
Instructor

# Matplotlib Axes

- Most customization available through `matplotlib` `Axes` objects
- `Axes` can be passed to seaborn functions

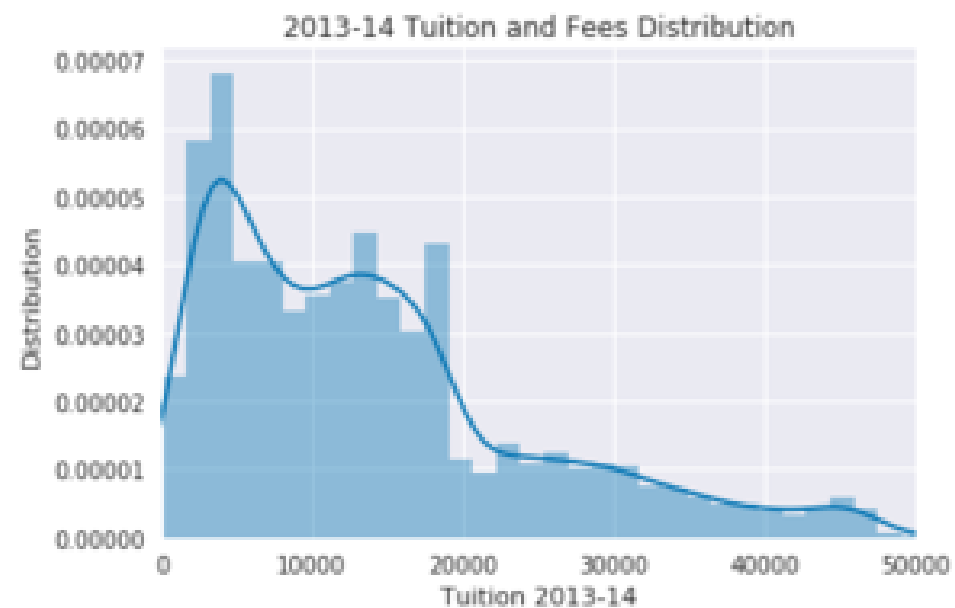
```
fig, ax = plt.subplots()
sns.distplot(df['Tuition'], ax=ax)
ax.set(xlabel="Tuition 2013-14")
```



# Further Customizations

- The `axes` object supports many common customizations

```
fig, ax = plt.subplots()
sns.distplot(df['Tuition'], ax=ax)
ax.set(xlabel="Tuition 2013-14",
       ylabel="Distribution", xlim=(0, 50000),
       title="2013-14 Tuition and Fees Distribution")
```



# Combining Plots

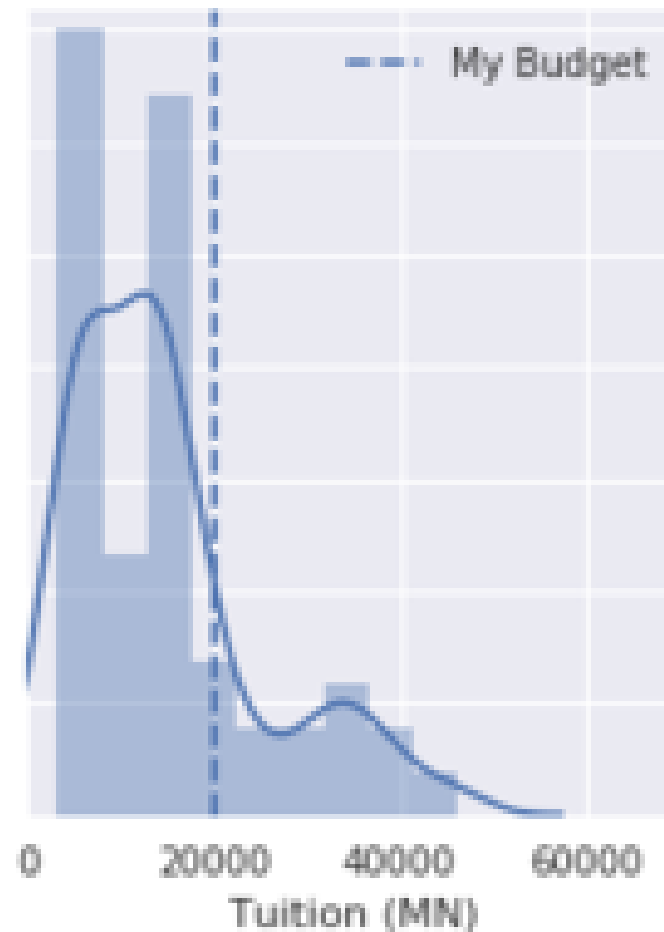
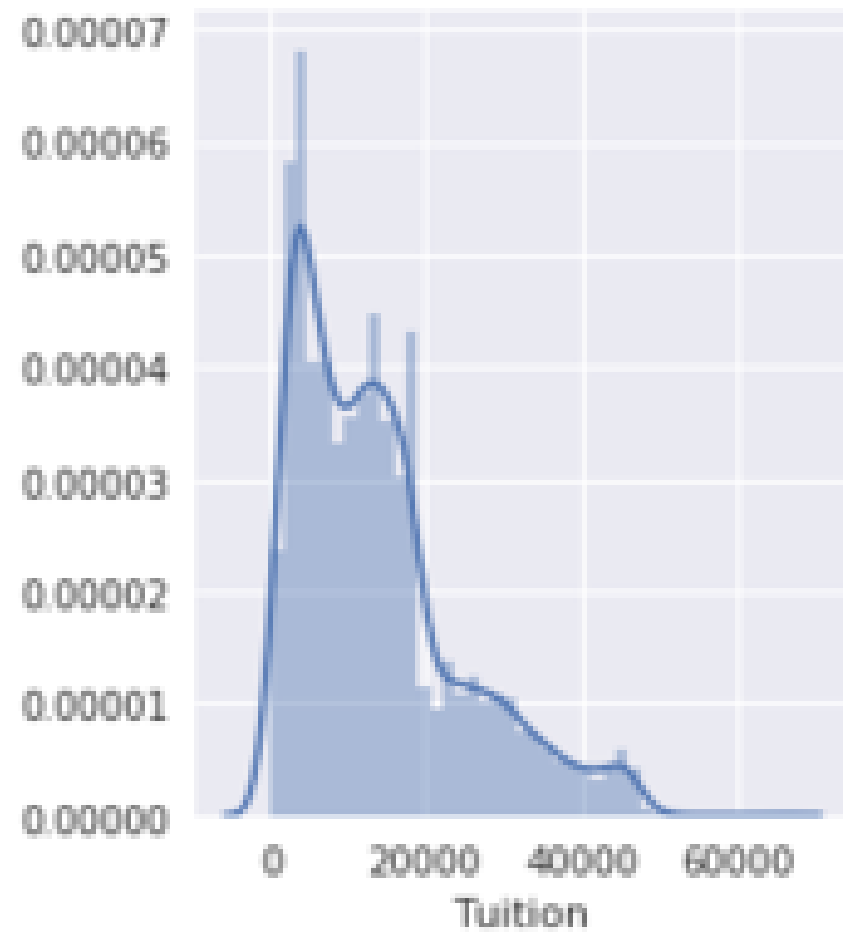
- It is possible to combine and configure multiple plots

```
fig, (ax0, ax1) = plt.subplots(
    nrows=1, ncols=2, sharey=True, figsize=(7,4))

sns.distplot(df['Tuition'], ax=ax0)
sns.distplot(df.query(
    'State == "MN"')['Tuition'], ax=ax1)

ax1.set(xlabel="Tuition (MN)", xlim=(0, 70000))
ax1.axvline(x=20000, label='My Budget', linestyle='--')
ax1.legend()
```

# Combining Plots





# Let's practice!

INTERMEDIATE DATA VISUALIZATION WITH SEABORN