

```

#Import relevant libraries
import warnings                                # `do not disturbe`
mode
warnings.filterwarnings('ignore')

import numpy as np                             # vectors and
matrices
import pandas as pd                           # tables and data
manipulations
import matplotlib.pyplot as plt               # plots
import seaborn as sns                         # more plots

from dateutil.relativedelta import relativedelta # working with dates
with style
from scipy.optimize import minimize           # for function
minimization

import statsmodels.formula.api as smf         # statistics and
econometrics
import statsmodels.tsa.api as smt
import statsmodels.api as sm
import scipy.stats as scs

from itertools import product                 # some useful
functions
from tqdm import tqdm_notebook

from mpl_toolkits import mplot3d
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import statistics
from scipy.stats import pearsonr
import matplotlib.dates as mdates

df =
pd.read_excel('/content/drive/MyDrive/ACADEMIA/Papers_review/Prof.
Karabo/Agrowaste_data.xlsx')
df.head()

```

	Experiment	Enzyme fraction (Kda)	Enzyme fraction class	\
0	1.0	<3	1.0	
1	2.0	<3	1.0	
2	3.0	>3	2.0	
3	4.0	>3	2.0	
4	5.0	<10	3.0	

Particle size class	Particle size (µm)	Time (hrs)	TRS (g/L)	TPC
---------------------	--------------------	------------	-----------	-----

(g/L)				
0	1.0	>75 - <106	24.0	31.646975
5.629286				
1	2.0	>106	24.0	33.475547
4.928571				
2	1.0	>75 - <106	24.0	31.075547
5.475000				
3	2.0	>106	24.0	39.018404
6.424286				
4	1.0	>75 - <106	24.0	38.839832
4.892143				

```
df.describe().transpose()
```

	count	mean	std	min
25% \				
Experiment	16.0	8.500000	4.760952	1.000000
4.750000				
Enzyme fraction class	16.0	2.500000	1.154701	1.000000
1.750000				
Particle size class	16.0	1.500000	0.516398	1.000000
1.000000				
Time (hrs)	16.0	48.000000	24.787093	24.000000
24.000000				
TRS (g/L)	16.0	56.002171	27.000270	31.075547
33.377332				
TPC (g/L)	16.0	4.309905	1.680511	1.114138
3.003362				

	50%	75%	max
Experiment	8.500000	12.250000	16.000000
Enzyme fraction class	2.500000	3.250000	4.000000
Particle size class	1.500000	2.000000	2.000000
Time (hrs)	48.000000	72.000000	72.000000
TRS (g/L)	42.821702	67.671875	115.187500
TPC (g/L)	4.910357	5.428060	6.607241

```
df.columns
```

```
Index(['Experiment', 'Enzyme fraction (Kda)', 'Enzyme fraction class',
      'Particle size class', 'Particle size (μm)', 'Time (hrs)', 'TRS
(g/L)',
      'TPC (g/L)'],
      dtype='object')
```

```
tpc_std = df['TPC (g/L)'].std()
trs_std = df['TRS (g/L)'].std()
```

```
fig = plt.figure(figsize=(8, 5))
ax = fig.add_axes([0,0,1,1])
ax2 = ax.twinx()
```

```

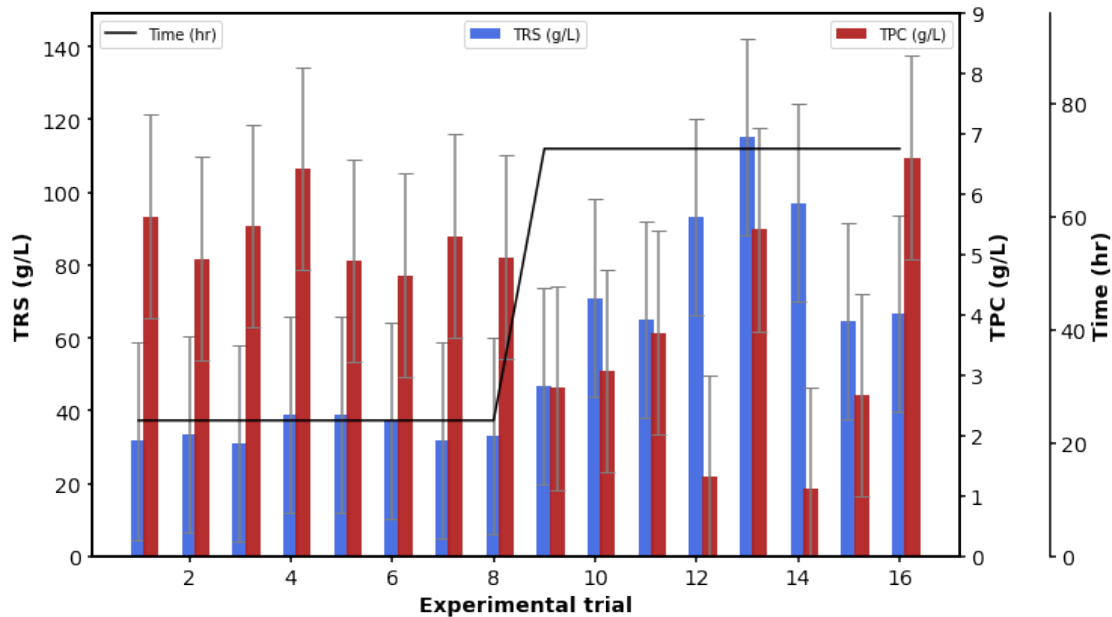
ax3 = ax.twinx()
ax.bar(df['Experiment'] + 0.00, df['TRS (g/L)'], color = 'royalblue',
width = 0.3, label='TRS (g/L)', yerr=trs_std, ecolord='grey',
capsize=5, alpha=.95)
ax2.bar(df['Experiment']+ 0.25, df['TPC (g/L)'], color = 'firebrick',
width = 0.3, label='TPC (g/L)', yerr=tpc_std, ecolord='grey',
capsize=5, alpha=.95)
ax3.plot(df['Experiment'], df['Time (hrs)'], color = 'black',
label='Time (hr)')
ax.set_xlabel('Experimental trial', size=14, fontweight = 'bold')
ax.set_ylabel('TRS (g/L)', size=14, fontweight = 'bold')
ax.legend( loc='upper center')
ax2.legend(loc='best')
ax2.set_ylim(0, 9)
ax3.set_ylim(0, 96)
ax3.legend(loc='upper left')
ax2.set_ylabel('TPC (g/L)', size=14, fontweight = 'bold')
ax3.set_ylabel('Time (hr)', size=14, fontweight = 'bold')

ax3.spines['right'].set_position(('outward', 60))
#ax3.xaxis.set_ticks([])

ax.tick_params(axis='x', labelsize=14)
ax.tick_params(axis='y', labelsize=14)
ax2.tick_params(axis='y', labelsize=14)
ax3.tick_params(axis='y', labelsize=14)

# change all spines
for axis in ['top', 'bottom', 'left', 'right']:
    ax.spines[axis].set_linewidth(1.5)
    ax2.spines[axis].set_linewidth(1.5)
    ax3.spines[axis].set_linewidth(1.5)
# increase tick width
ax.tick_params(width=1.5)
ax2.tick_params(width=1.5)
ax3.tick_params(width=1.5)

```



```

fig = plt.figure(figsize=(8, 5))
ax = fig.add_axes([0,0,1,1])
ax2 = ax.twinx()
ax3 = ax.twinx()
ax.bar(df['Experiment'] + 0.00, df['TRS (g/L)'], color = 'royalblue',
width = 0.3, label='TRS (g/L)', yerr=trs_std, ecolor='grey',
capsize=5, alpha=.95)
ax2.bar(df['Experiment']+ 0.25, df['TPC (g/L)'], color = 'firebrick',
width = 0.3, label='TPC (g/L)', yerr=tpc_std, ecolor='grey',
capsize=5, alpha=.95)
ax3.plot(df['Experiment'] ,df['Enzyme fraction class'], color =
'black', label='Enzyme fraction class')
ax.set_xlabel('Experimental trial', size=13, fontweight = 'bold')
ax.set_ylabel('TRS (g/L)', size=13, fontweight = 'bold')
ax.legend( loc='upper center')
ax2.legend(loc='best')
ax2.set_ylim(0, 9)
ax3.set_ylim(0, 5)
ax3.legend(loc='upper left')
ax2.set_ylabel('TPC (g/L)', size=13, fontweight = 'bold')
ax3.set_ylabel('Enzyme fraction class', size=13, fontweight = 'bold')

ax3.spines['right'].set_position(('outward', 60))

ax.tick_params(axis='x', labels=14)
ax.tick_params(axis='y', labels=14)
ax2.tick_params(axis='y', labels=14)
ax3.tick_params(axis='y', labels=14)

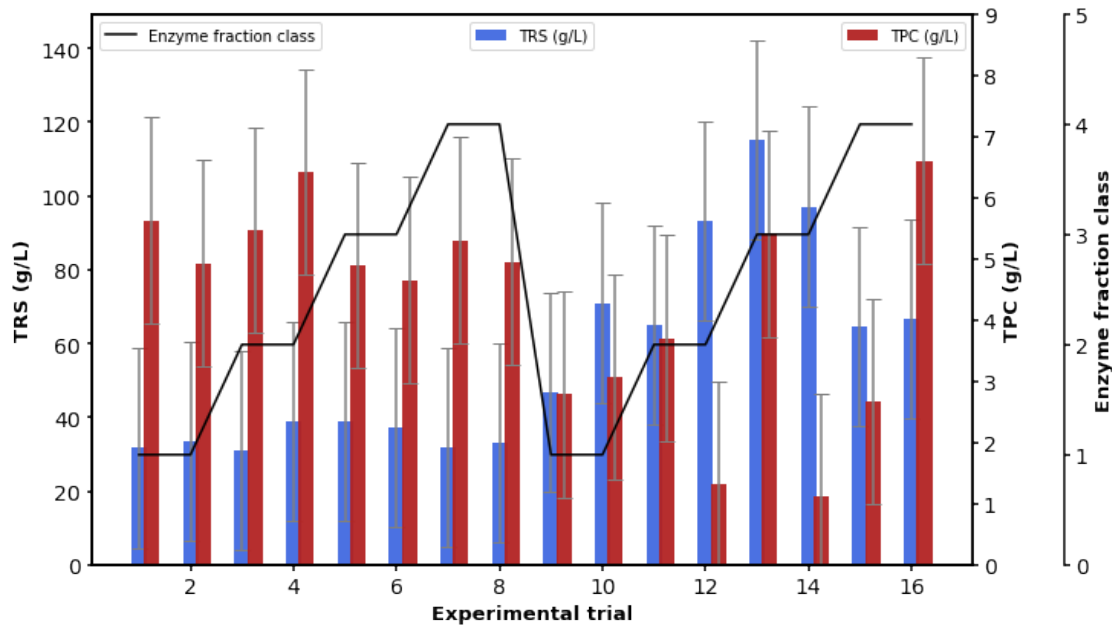
# change all spines
for axis in ['top','bottom','left','right']:

```

```

ax.spines[axis].set_linewidth(1.5)
ax2.spines[axis].set_linewidth(1.5)
ax3.spines[axis].set_linewidth(1.5)
# increase tick width
ax.tick_params(width=1.5)
ax2.tick_params(width=1.5)
ax3.tick_params(width=1.5)

```



```

fig = plt.figure(figsize=(8, 5))
ax = fig.add_axes([0,0,1,1])
ax2 = ax.twinx()
ax3 = ax.twinx()
ax.bar(df['Experiment'] + 0.00, df['TRS (g/L)'], color = 'royalblue',
width = 0.3, label='TRS (g/L)', yerr=trs_std, ecolor='grey',
capsize=5, alpha=.95)
ax2.bar(df['Experiment']+ 0.25, df['TPC (g/L)'], color = 'firebrick',
width = 0.3, label='TPC (g/L)', yerr=tpc_std, ecolor='grey',
capsize=5, alpha=.95)
ax3.scatter(df['Experiment'] ,df['Particle size class'], color =
'black', label='Particle size class', marker='D')
ax.set_xlabel('Experimental trial', size=13, fontweight = 'bold')
ax.set_ylabel('TRS (g/L)', size=13, fontweight = 'bold')
ax.legend( loc='upper center')
ax2.legend(loc='best')
ax2.set_ylim(0, 9)
ax3.set_ylim(0, 3)
ax3.legend(loc='upper left')
ax2.set_ylabel('TPC (g/L)', size=13, fontweight = 'bold')
ax3.set_ylabel('Particle size class', size=13, fontweight = 'bold')

ax3.spines['right'].set_position(('outward', 60))

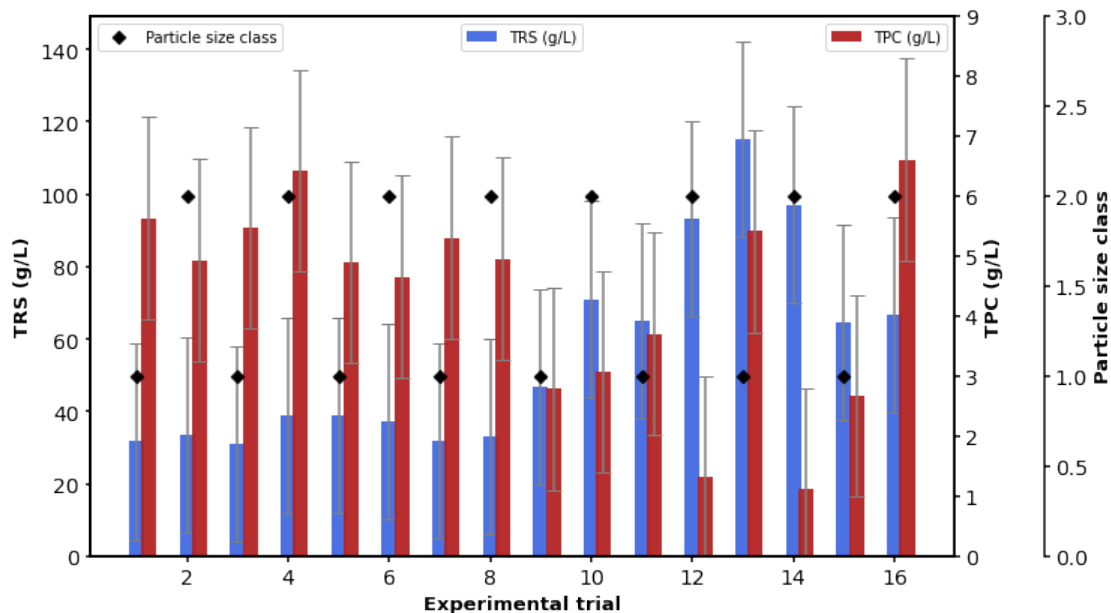
```

```

ax.tick_params(axis='x', labelsz=14)
ax.tick_params(axis='y', labelsz=14)
ax2.tick_params(axis='y', labelsz=14)
ax3.tick_params(axis='y', labelsz=14)

# change all spines
for axis in ['top', 'bottom', 'left', 'right']:
    ax.spines[axis].set_linewidth(1.5)
    ax2.spines[axis].set_linewidth(1.5)
    ax3.spines[axis].set_linewidth(1.5)
# increase tick width
ax.tick_params(width=1.5)
ax2.tick_params(width=1.5)
ax3.tick_params(width=1.5)

```



```

fig = plt.figure(figsize=(10, 6))
ax = fig.add_axes([0,0,1,1])
ax2 = ax.twinx()
ax3 = ax.twinx()
ax4 = ax.twinx()
ax5 = ax.twinx()

ax.bar(df['Experiment'] + 0.00, df['TRS (g/L)'], color = 'royalblue',
width = 0.25, label='TRS (g/L)', yerr=trs_std, ecolor='grey',
capsize=5, alpha=.95)
ax2.bar(df['Experiment']+ 0.25, df['TPC (g/L)'], color = 'firebrick',
width = 0.25, label='TPC (g/L)', yerr=tpc_std, ecolor='grey',
capsize=5, alpha=.95)
ax3.plot(df['Experiment'] ,df['Time (hrs)'], color = 'black',
label='Time (hr)')

```

```

ax4.plot(df['Experiment'], df['Enzyme fraction class'], color =
'orange', label='Enzyme fraction class')
ax5.scatter(df['Experiment'], df['Particle size class'], color =
'darkgreen', label='Particle size class', marker='D')
ax.set_xlabel('Experimental trial', size=13, fontweight = 'bold')
ax.set_ylabel('TRS (g/L)', size=13, fontweight = 'bold')
ax.legend( loc='upper center')
ax2.legend(loc='upper right')
ax4.legend(loc='lower left')
ax5.legend(loc='lower right')
ax2.set_ylim(0, 9)
ax3.set_ylim(0, 96)
ax4.set_ylim(0, 5)
ax5.set_ylim(0, 3)
ax3.legend(loc='upper left')
ax2.set_ylabel('TPC (g/L)', size=13, fontweight = 'bold')
ax3.set_ylabel('Time (hr)', size=13, fontweight = 'bold')
ax4.set_ylabel('Enzyme fraction class', size=12, fontweight = 'bold')
ax5.set_ylabel('Particle size class', size=12, fontweight = 'bold')

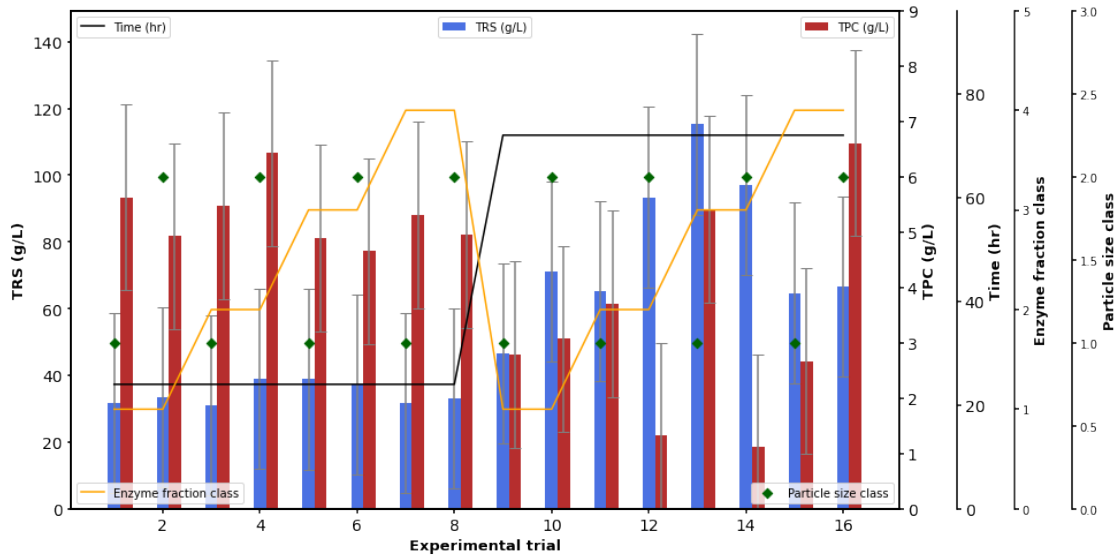
ax3.spines['right'].set_position(('outward', 50))
ax4.spines['right'].set_position(('outward', 100))
ax5.spines['right'].set_position(('outward', 150))
#ax3.xaxis.set_ticks([])

ax.tick_params(axis='x', labelsize=14)
ax.tick_params(axis='y', labelsize=14)
ax2.tick_params(axis='y', labelsize=14)
ax3.tick_params(axis='y', labelsize=14)

# change all spines
for axis in ['top', 'bottom', 'left', 'right']:
    ax.spines[axis].set_linewidth(1.5)
    ax2.spines[axis].set_linewidth(1.5)
    ax3.spines[axis].set_linewidth(1.5)
    ax4.spines[axis].set_linewidth(1.5)
    ax5.spines[axis].set_linewidth(1.5)

# increase tick width
ax.tick_params(width=1.5)
ax2.tick_params(width=1.5)
ax3.tick_params(width=1.5)
ax4.tick_params(width=1.5)
ax5.tick_params(width=1.5)

```



```
combined_data_corrMatrix = df[['Enzyme fraction class', 'Particle size
class', 'Time (hrs)', 'TRS (g/L)', 'TPC (g/L)']].corr()
fig, ax = plt.subplots(figsize=(10,8))
sns.heatmap(combined_data_corrMatrix, annot = True, cmap="YlGnBu",
fmt='.3g', ax=ax)
```

```
ax.tick_params(axis='x', labelsz=12)
ax.tick_params(axis='y', labelsz=12)
```

```
for label in ax.get_xticklabels():
    if label.get_text() == "TPC (g/L)":
        label.set_size(12)
        label.set_weight("bold")
        label.set_color("red")
```

```
for label in ax.get_xticklabels():
    if label.get_text() == "TRS (g/L)":
        label.set_size(12)
        label.set_weight("bold")
        label.set_color("red")
```

```
for label in ax.get_yticklabels():
    if label.get_text() == "TPC (g/L)":
        label.set_size(12)
        label.set_weight("bold")
        label.set_color("red")
```

```
for label in ax.get_yticklabels():
    if label.get_text() == "TRS (g/L)":
        label.set_size(12)
        label.set_weight("bold")
        label.set_color("red")
```





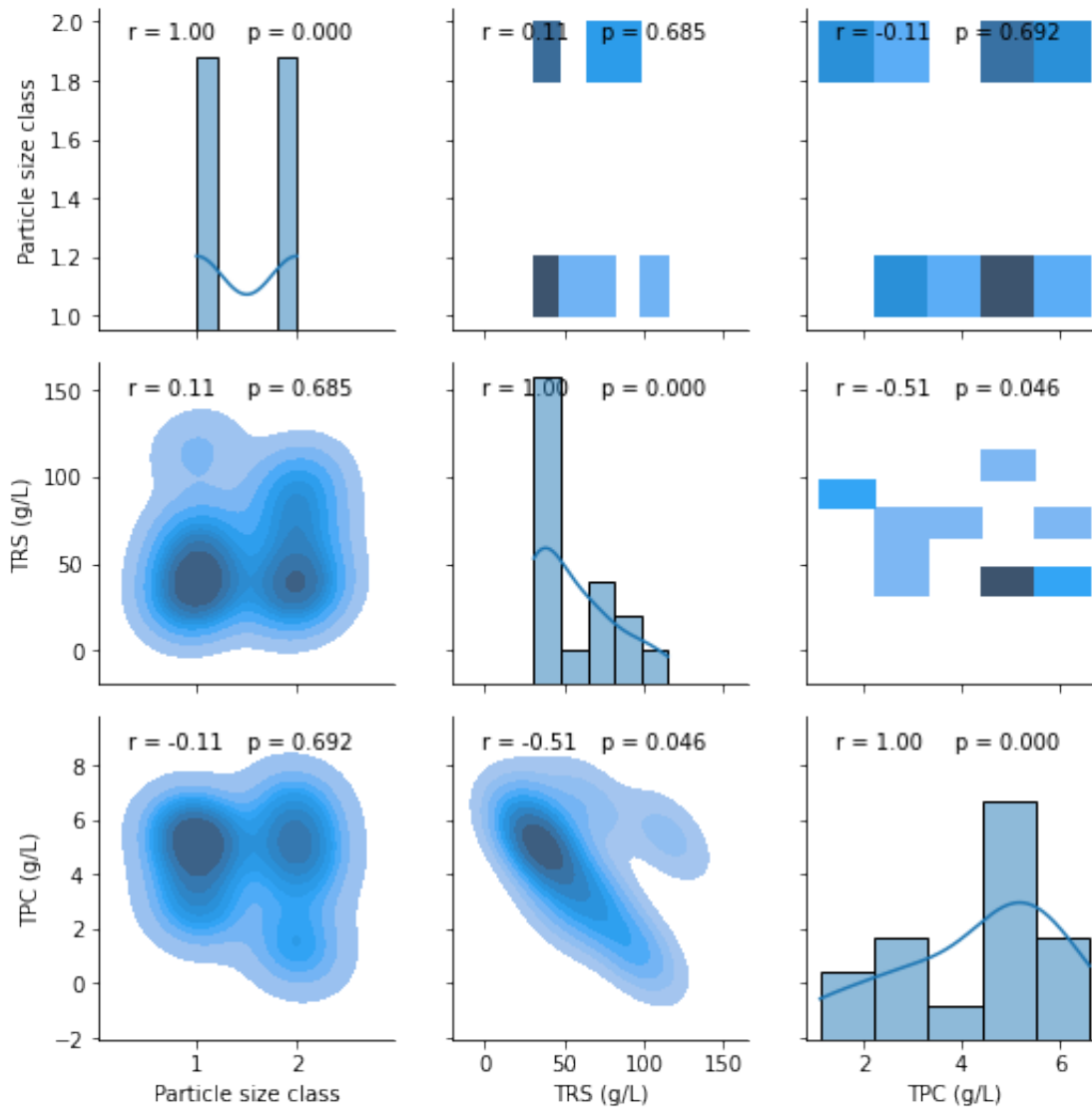
```
from scipy.stats import pearsonr
```

```
data_set = df[['Particle size class', 'TRS (g/L)', 'TPC (g/L)']]
```

```
def corrfunc(x, y, **kws):
    (r, p) = pearsonr(x, y)
    ax = plt.gca()
    ax.annotate("r = {:.2f} ".format(r),
                xy=(.1, .9), xycoords=ax.transAxes)
    ax.annotate("p = {:.3f}".format(p),
                xy=(.5, .9), xycoords=ax.transAxes)
```

```
g = sns.PairGrid(data_set)
g.map(corrfunc)
g.map_upper(sns.histplot)
g.map_lower(sns.kdeplot, fill=True)
g.map_diag(sns.histplot, kde=True)
```

```
<seaborn.axisgrid.PairGrid at 0x7f308cfad910>
```



```
from scipy.stats import pearsonr
```

```
data_set2 = df[['Enzyme fraction class', 'TRS (g/L)', 'TPC (g/L)']]
```

```
def corrfunc(x, y, **kws):
    (r, p) = pearsonr(x, y)
    ax = plt.gca()
    ax.annotate("r = {:.2f} ".format(r),
                xy=(.1, .9), xycoords=ax.transAxes)
    ax.annotate("p = {:.3f} ".format(p),
                xy=(.5, .9), xycoords=ax.transAxes)
```

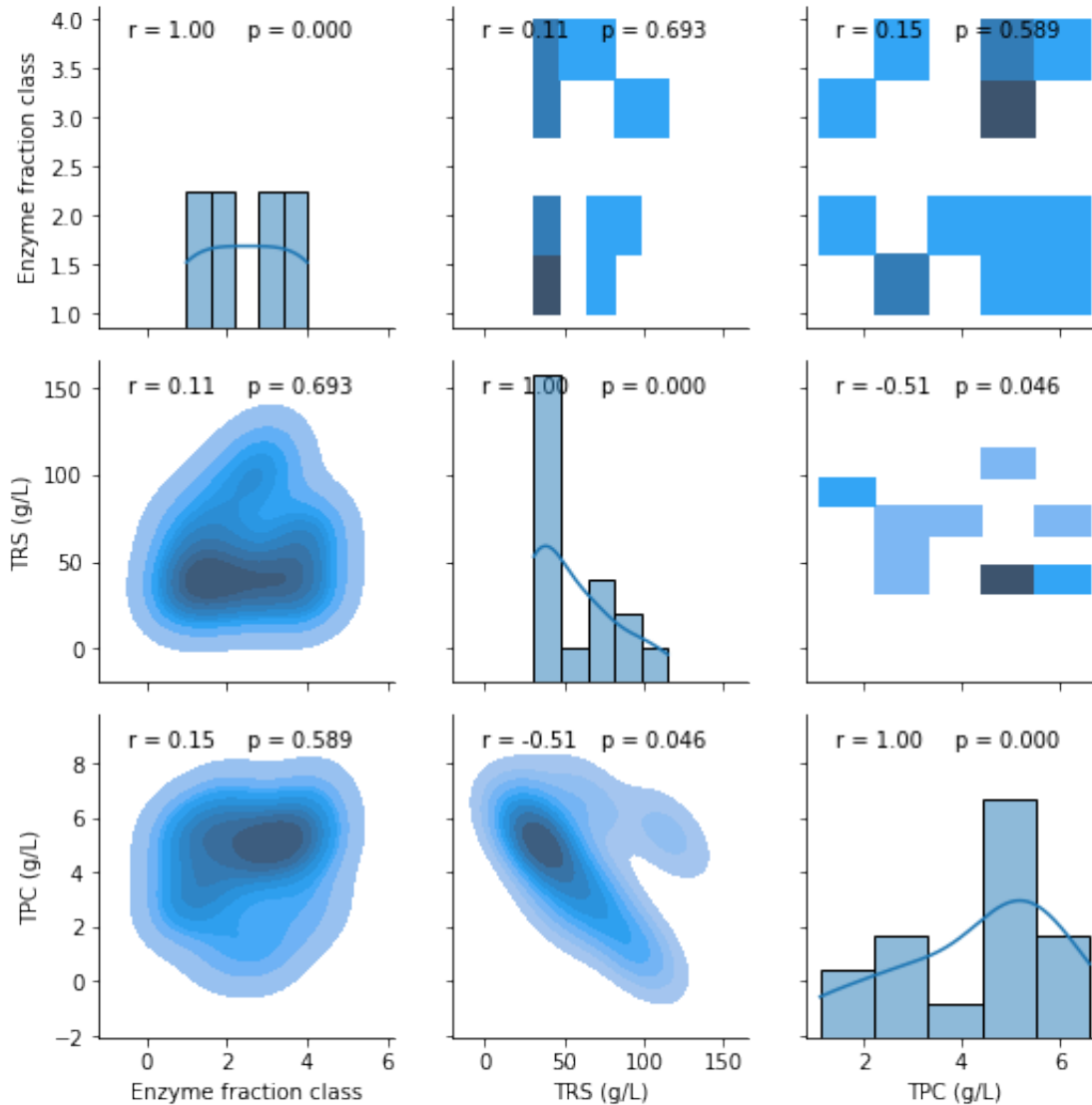
```
g = sns.PairGrid(data_set2)
g.map(corrfunc)
```

```

g.map_upper(sns.histplot)
g.map_lower(sns.kdeplot, fill=True)
g.map_diag(sns.histplot, kde=True)

<seaborn.axisgrid.PairGrid at 0x7f308f871710>

```



```

from scipy.stats import pearsonr

data_set3 = df[['Time (hrs)', 'TRS (g/L)', 'TPC (g/L)']]

```

```

def corrfunc(x, y, **kws):
    (r, p) = pearsonr(x, y)
    ax = plt.gca()
    ax.annotate("r = {:.2f} ".format(r),

```

```

        xy=(.1, .9), xycoords=ax.transAxes)
    ax.annotate("p = {:.3f}".format(p),
               xy=(.5, .9), xycoords=ax.transAxes)

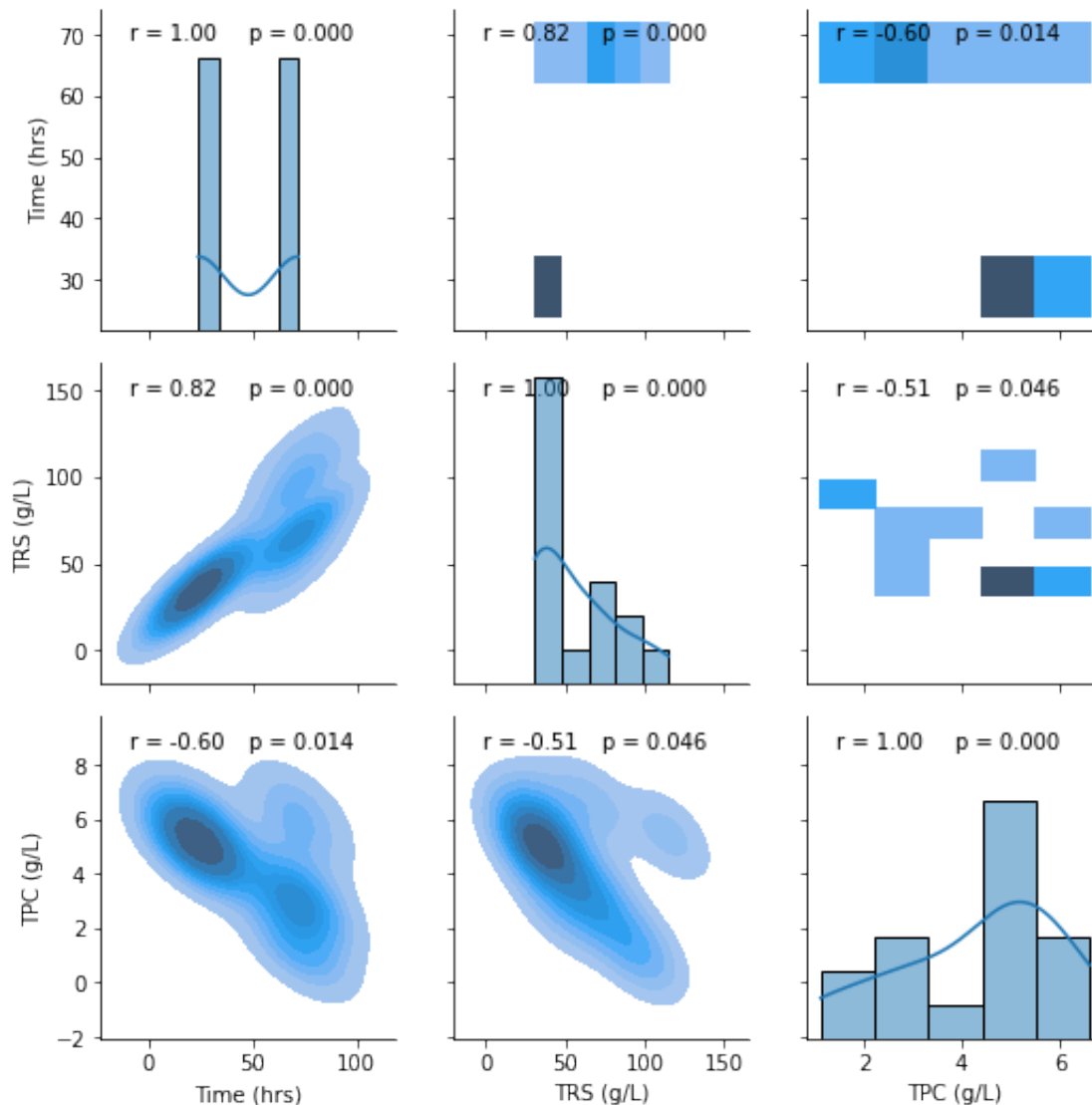
```

```

g = sns.PairGrid(data_set3)
g.map(corrfunc)
g.map_upper(sns.histplot)
g.map_lower(sns.kdeplot, fill=True)
g.map_diag(sns.histplot, kde=True)

```

<seaborn.axisgrid.PairGrid at 0x7f308a805710>



```

from mpl_toolkits import mplot3d
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter

```

```

import statistics

x = df['Experiment']
y = df['Enzyme fraction class']
z = df['TRS (g/L)']

# Creating figure
fig = plt.figure(figsize=(10, 6))
ax = plt.axes(projection='3d')

# Creating color map
my_cmap = plt.get_cmap('hot')

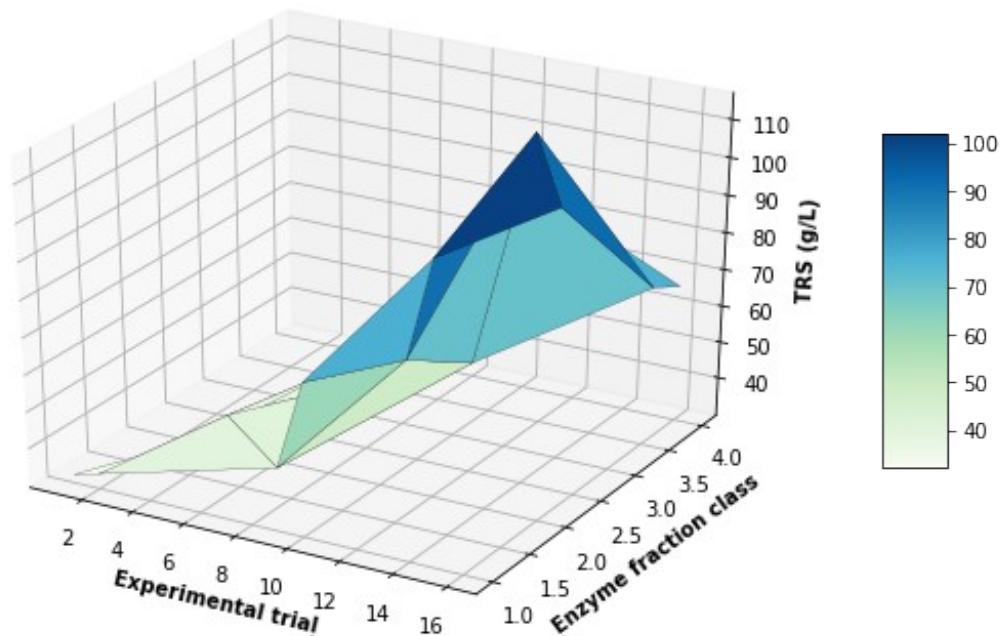
# Creating plot
trisurf = ax.plot_trisurf(x, y, z,
                           cmap = cm.GnBu,
                           linewidth = 0.2,
                           antialiased = True,
                           edgecolor = 'black')
fig.colorbar(trisurf, ax = ax, shrink = 0.5, aspect = 5)
#ax.set_title('Tri-Surface plot')

# Adding labels
ax.set_xlabel('Experimental trial', fontweight='bold')
ax.set_ylabel('Enzyme fraction class', fontweight='bold')
ax.set_zlabel('TRS (g/L)', fontweight='bold')

# show plot
plt.show

<function matplotlib.pyplot.show>

```



```

y = df['Particle size class']
x = df['Experiment']
z = df['TRS (g/L)']

# Creating figure
fig = plt.figure(figsize=(10, 6))
ax = plt.axes(projection='3d')

# Creating color map
my_cmap = plt.get_cmap('hot')

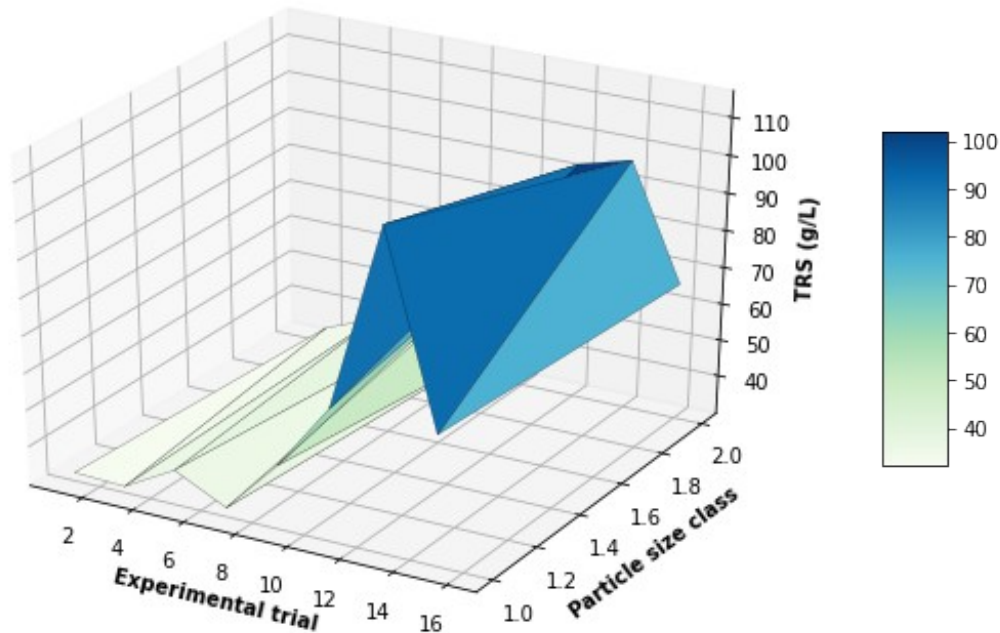
# Creating plot
trisurf = ax.plot_trisurf(x, y, z,
                           cmap = cm.GnBu,
                           linewidth = 0.2,
                           antialiased = True,
                           edgecolor = 'black')
fig.colorbar(trisurf, ax = ax, shrink = 0.5, aspect = 5)
#ax.set_title('Tri-Surface plot')

# Adding labels
ax.set_ylabel('Particle size class', fontweight='bold')
ax.set_xlabel('Experimental trial', fontweight='bold')
ax.set_zlabel('TRS (g/L)', fontweight='bold')

# show plot
plt.show

```

```
<function matplotlib.pyplot.show>
```



```
y = df['Time (hrs)']  
x = df['Experiment']  
z = df['TRS (g/L)']
```

```
# Creating figure
```

```
fig = plt.figure(figsize=(10, 6))  
ax = plt.axes(projection='3d')
```

```
# Creating color map
```

```
my_cmap = plt.get_cmap('hot')
```

```
# Creating plot
```

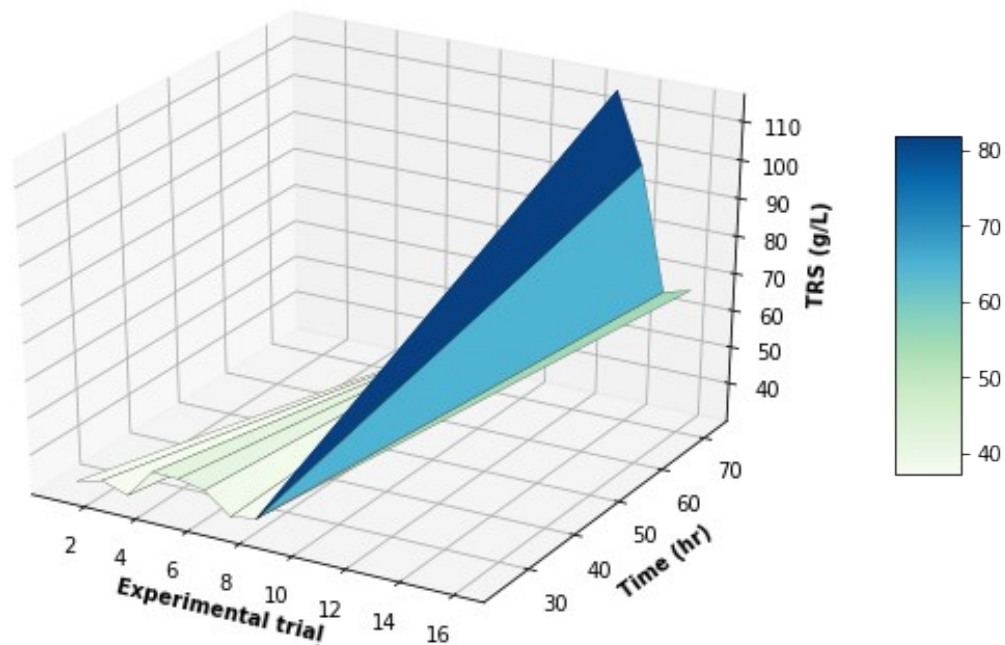
```
trisurf = ax.plot_trisurf(x, y, z,  
                           cmap = cm.GnBu,  
                           linewidth = 0.2,  
                           antialiased = True,  
                           edgecolor = 'black')  
fig.colorbar(trisurf, ax = ax, shrink = 0.5, aspect = 5)  
#ax.set_title('Tri-Surface plot')
```

```
# Adding labels
```

```
ax.set_ylabel('Time (hr)', fontweight='bold')  
ax.set_xlabel('Experimental trial', fontweight='bold')  
ax.set_zlabel('TRS (g/L)', fontweight='bold')
```

```
# show plot
plt.show

<function matplotlib.pyplot.show>
```



```
x = df['Experiment']
y = df['TPC (g/L)']
z = df['TRS (g/L)']

# Creating figure
fig = plt.figure(figsize =(10, 6))
ax = plt.axes(projection ='3d')

# Creating color map
my_cmap = plt.get_cmap('hot')

# Creating plot
trisurf = ax.plot_trisurf(x, y, z,
                           cmap = cm.GnBu,
                           linewidth = 0.2,
                           antialiased = True,
                           edgecolor = 'black')
fig.colorbar(trisurf, ax = ax, shrink = 0.5, aspect = 5)
#ax.set_title('Tri-Surface plot')

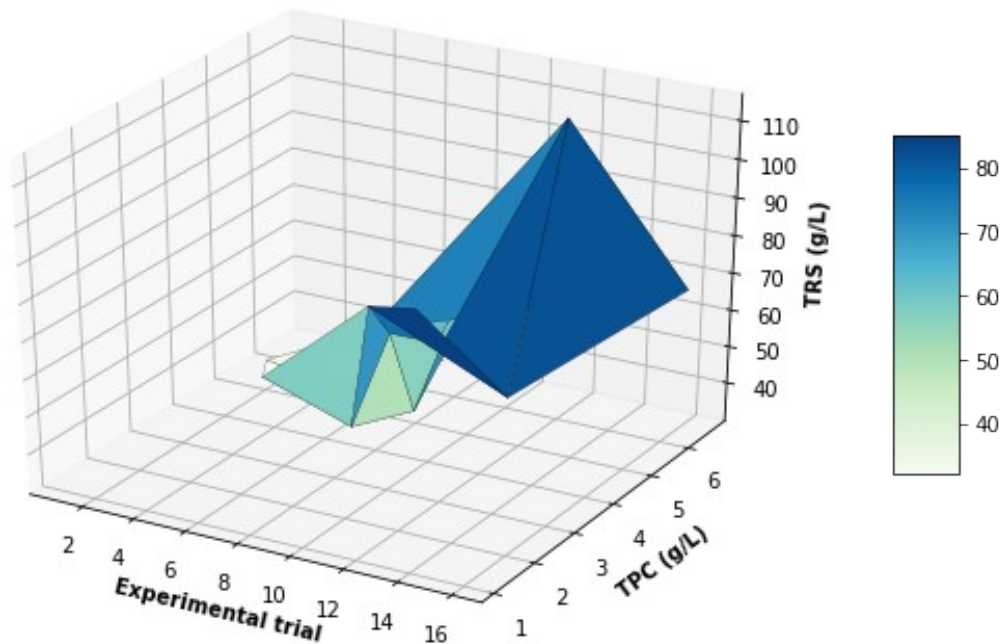
# Adding labels
ax.set_xlabel('Experimental trial', fontweight ='bold')
```



```
ax.set_ylabel('TPC (g/L)', fontweight = 'bold')
ax.set_zlabel('TRS (g/L)', fontweight = 'bold')
```

```
# show plot
plt.show
```

```
<function matplotlib.pyplot.show>
```



```
df.columns
```

```
Index(['Experiment', 'Enzyme fraction (Kda)', 'Enzyme fraction class',
      'Particle size class', 'Particle size (μm)', 'Time (hrs)', 'TRS
      (g/L)',
      'TPC (g/L)'],
      dtype='object')
```

```
df.columns
```

```
Index(['Experiment', 'Enzyme fraction (Kda)', 'Enzyme fraction class',
      'Particle size class', 'Particle size (μm)', 'Time (hrs)', 'TRS
      (g/L)',
      'TPC (g/L)'],
      dtype='object')
```

```
table = pd.DataFrame()
particle = pd.DataFrame()
time = pd.DataFrame()
enzyme = pd.DataFrame()
table_3 = pd.DataFrame()
```

```

table['trs_p_1'] = df[df["Particle size class"]==1]["TRS (g/L)"]
table_3['trs_p_2'] = df[df["Particle size class"]==2]["TRS (g/L)"]
table['tpc_p_1'] = df[df["Particle size class"]==1]["TPC (g/L)"]
table_3['tpc_p_2'] = df[df["Particle size class"]==2]["TPC (g/L)"]
table['trs_t_24'] = df[df["Time (hrs)"]==24]["TRS (g/L)"]
table['trs_t_72'] = df[df["Time (hrs)"]==72]["TRS (g/L)"]
table['tpc_t_24'] = df[df["Time (hrs)"]==24]["TPC (g/L)"]
table['tpc_t_72'] = df[df["Time (hrs)"]==72]["TPC (g/L)"]
table['trs_e_1'] = df[df["Enzyme fraction class"]==1]["TRS (g/L)"]
table['trs_e_2'] = df[df["Enzyme fraction class"]==2]["TRS (g/L)"]
table['trs_e_3'] = df[df["Enzyme fraction class"]==3]["TRS (g/L)"]
table['trs_e_4'] = df[df["Enzyme fraction class"]==4]["TRS (g/L)"]
table['tpc_e_1'] = df[df["Enzyme fraction class"]==1]["TPC (g/L)"]
table['tpc_e_2'] = df[df["Enzyme fraction class"]==2]["TPC (g/L)"]
table['tpc_e_3'] = df[df["Enzyme fraction class"]==3]["TPC (g/L)"]
table['tpc_e_4'] = df[df["Enzyme fraction class"]==4]["TPC (g/L)"]

```

```
time.head()
```

```
Empty DataFrame
```

```
Columns: []
```

```
Index: []
```

```

f, axes = plt.subplots(3,2, figsize=(16,18), sharex=False)
sns.kdeplot(df[df["Particle size class"]==1]["TRS (g/L)"], shade=True,
ax=axes[0,0], color='blue', label="TRS with particles size >75µm x
<106µm", alpha=.6)
sns.kdeplot(df[df["Particle size class"]==2]["TRS (g/L)"], shade=True,
ax=axes[0,0], color='purple', label="TRS with particles size >106µm",
alpha=.6)
axes[0,0].legend()
axes[0,0].set_xlabel("TRS (g/L)", fontweight='bold')
axes[0,0].set_ylabel("Density", fontweight='bold')
axes[0,0].legend(loc="upper right")

```

```

sns.kdeplot(df[df["Particle size class"]==1]["TPC (g/L)"], shade=True,
ax=axes[0,1], color='blue', label="TPC with particles size >75µm x
<106µm", alpha=.6)
sns.kdeplot(df[df["Particle size class"]==2]["TPC (g/L)"], shade=True,
ax=axes[0,1], color='purple', label="TPC with particles size >106µm",
alpha=.6)
axes[0,1].legend()
axes[0,1].set_xlabel("TPC (g/L)", fontweight='bold')
axes[0,1].set_ylabel("Density", fontweight='bold')
axes[0,1].legend(loc="upper left")

```

```

sns.kdeplot(df[df["Time (hrs)"]==24]["TRS (g/L)"], shade=True,
ax=axes[1,0], color='orange', label="TRS with a processing time of 24
hrs", alpha=.6)
sns.kdeplot(df[df["Time (hrs)"]==72]["TRS (g/L)"], shade=True,
ax=axes[1,0], color='green', label="TRS with a processing time of 72

```

```

hrs", alpha=.6)
axes[1,0].legend()
axes[1,0].set_xlabel("TRS (g/L)", fontweight='bold')
axes[1,0].set_ylabel("Density", fontweight='bold')
axes[1,0].legend(loc="upper right")

sns.kdeplot(df[df["Time (hrs)"]==24]["TPC (g/L)"], shade=True,
ax=axes[1,1], color='orange', label="TPC with a processing time of 24
hrs", alpha=.6)
sns.kdeplot(df[df["Time (hrs)"]==72]["TPC (g/L)"], shade=True,
ax=axes[1,1], color='green', label="TPC with a processing time of 72
hrs", alpha=.6)
axes[1,1].legend()
axes[1,1].set_xlabel("TPC (g/L)", fontweight='bold')
axes[1,1].set_ylabel("Density", fontweight='bold')
axes[1,1].legend(loc="upper left")

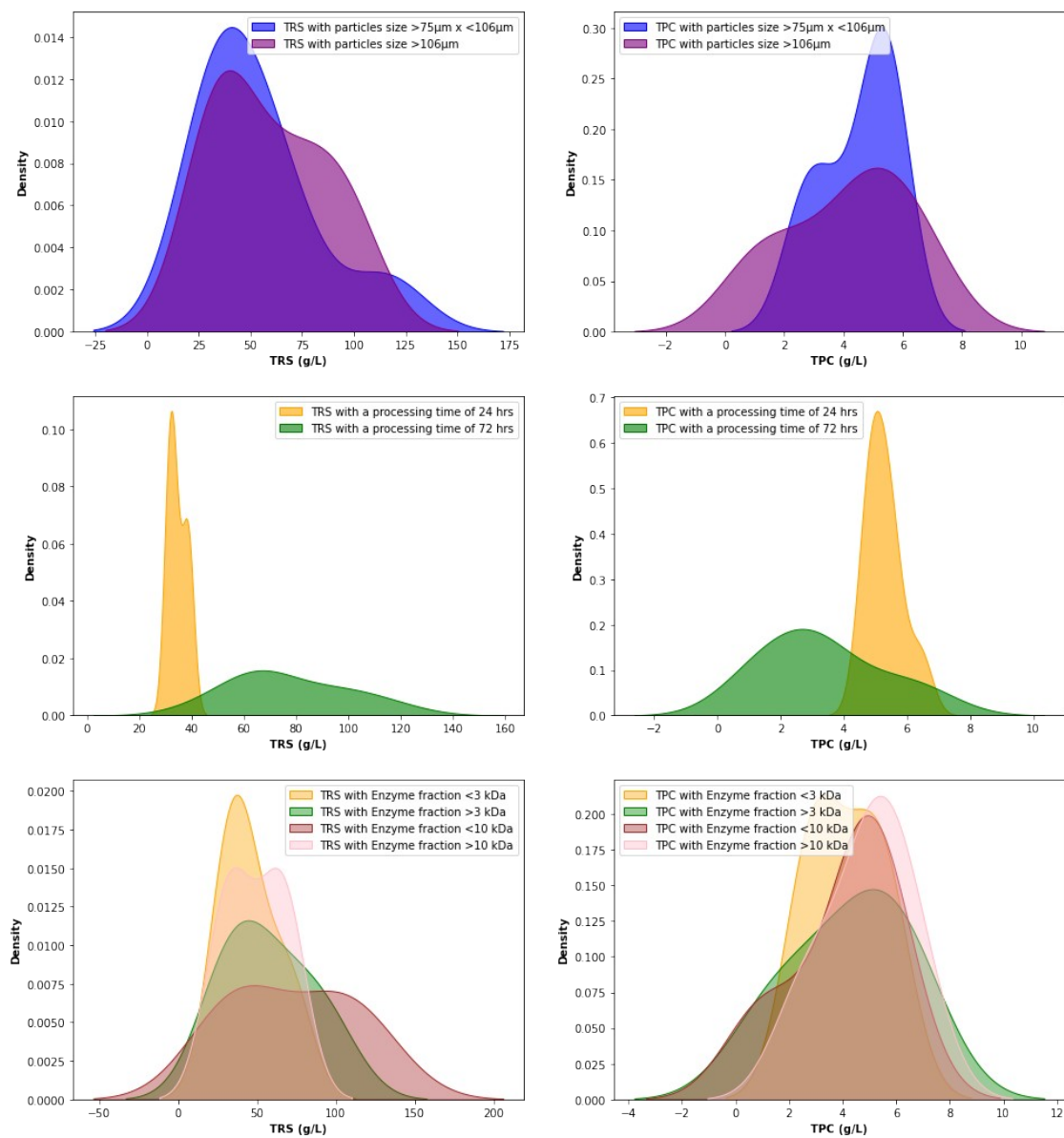
sns.kdeplot(df[df["Enzyme fraction class"]==1]["TRS (g/L)"],
shade=True, ax=axes[2,0], color='orange', label="TRS with Enzyme
fraction <3 kDa", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==2]["TRS (g/L)"],
shade=True, ax=axes[2,0], color='green', label="TRS with Enzyme
fraction >3 kDa", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==3]["TRS (g/L)"],
shade=True, ax=axes[2,0], color='brown', label="TRS with Enzyme
fraction <10 kDa", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==4]["TRS (g/L)"],
shade=True, ax=axes[2,0], color='pink', label="TRS with Enzyme
fraction >10 kDa", alpha=.4)
axes[2,0].legend()
axes[2,0].set_xlabel("TRS (g/L)", fontweight='bold')
axes[2,0].set_ylabel("Density", fontweight='bold')
axes[2,0].legend(loc="upper right")

sns.kdeplot(df[df["Enzyme fraction class"]==1]["TPC (g/L)"],
shade=True, ax=axes[2,1], color='orange', label="TPC with Enzyme
fraction <3 kDa", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==2]["TPC (g/L)"],
shade=True, ax=axes[2,1], color='green', label="TPC with Enzyme
fraction >3 kDa", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==3]["TPC (g/L)"],
shade=True, ax=axes[2,1], color='brown', label="TPC with Enzyme
fraction <10 kDa", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==4]["TPC (g/L)"],
shade=True, ax=axes[2,1], color='pink', label="TPC with Enzyme
fraction >10 kDa", alpha=.4)
axes[2,1].legend()
axes[2,1].set_xlabel("TPC (g/L)", fontweight='bold')
axes[2,1].set_ylabel("Density", fontweight='bold')

```

```
axes[2,1].legend(loc="upper left")
```

```
<matplotlib.legend.Legend at 0x7f308f854490>
```



```
table_3.describe().transpose()
```

	count	mean	std	min	25%	50%
\						
trs_p_2	8.0	58.877729	26.795536	33.082689	36.405904	52.790452
tpc_p_2	8.0	4.134929	2.107982	1.114138	2.636379	4.791429
		75%	max			

```

trs_p_2  76.593750  97.125000
tpc_p_2   5.323393   6.607241

table['tpc_p_2'] = table_3['tpc_p_2']
table['trs_p_2'] = table_3['trs_p_2']

table = table.drop(['tpc_p_2'], axis=1)

describe_df = table.describe().transpose().drop([], axis=1)
final_table = describe_df.append(table_3.describe().transpose())
final_table.to_csv('/content/drive/MyDrive/ACADEMIA/Papers_review/
Prof. Karabo/final_table.csv')

table_1 = table.drop(['tpc_p_1',
                    'tpc_t_24', 'tpc_t_72',
                    'tpc_e_1', 'tpc_e_2', 'tpc_e_3', 'tpc_e_4'], axis=1)

table_1.columns

Index(['trs_p_1', 'trs_t_24', 'trs_t_72', 'trs_e_1', 'trs_e_2',
      'trs_e_3',
      'trs_e_4', 'trs_p_2'],
      dtype='object')

f, axes = plt.subplots(3, figsize=(7, 12), sharex=False)
sns.kdeplot(table_1['trs_p_1'], shade=True, ax=axes[0], color='blue',
label="Total reducible sugars (g/L) with particles >75µm and <106µm",
alpha=.6)
sns.kdeplot(table_1["trs_p_2"], shade=True, ax=axes[0],
color='purple', label="Total reducible sugars (g/L) with particles
>106µm", alpha=.6)
axes[0].legend(loc="center left")

sns.kdeplot(table_1['trs_t_24'], shade=True, ax=axes[1],
color='orange', label="Total reducible sugars (g/L) with a processing
time of 24 hrs", alpha=.6)
sns.kdeplot(table_1['trs_t_72'], shade=True, ax=axes[1],
color='green', label="Total reducible sugars (g/L) with a processing
time of 72 hrs", alpha=.6)
axes[1].legend(loc="center left")

sns.kdeplot(table_1['trs_e_1'], shade=True, ax=axes[2],
color='orange', label="Total phenolic compounds (g/L) with Enzyme
fraction <3 kDa", alpha=.4)
sns.kdeplot(table_1['trs_e_2'], shade=True, ax=axes[2], color='green',
label="Total phenolic compounds (g/L) with Enzyme fraction >3 kDa",
alpha=.4)
sns.kdeplot(table_1['trs_e_3'], shade=True, ax=axes[2], color='brown',
label="Total phenolic compounds (g/L) with Enzyme fraction <10 kDa",
alpha=.4)
sns.kdeplot(table_1['trs_e_4'], shade=True, ax=axes[2], color='pink',

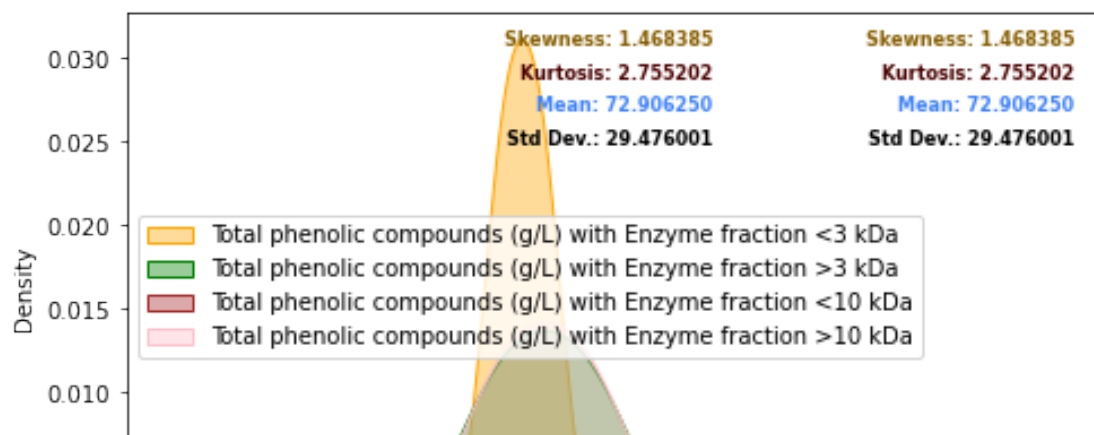
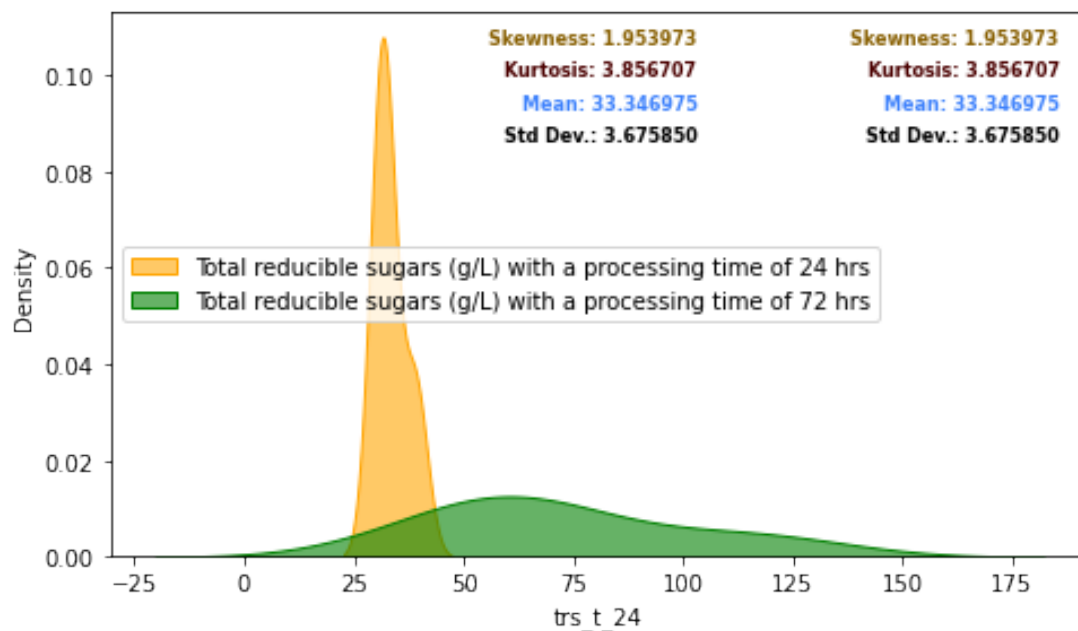
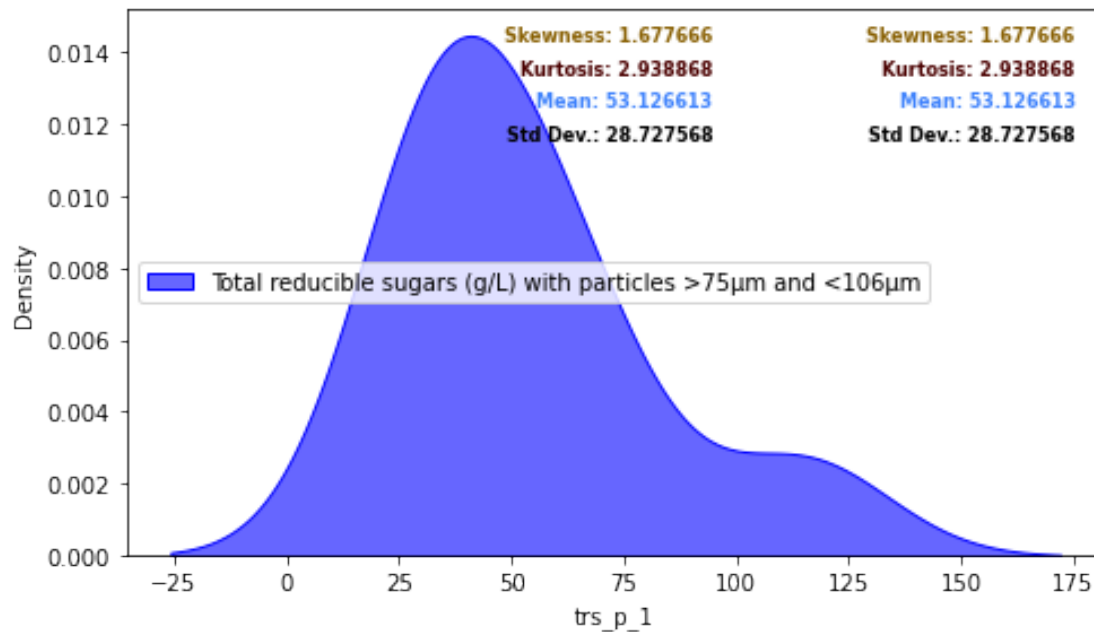
```

```

label="Total phenolic compounds (g/L) with Enzyme fraction >10 kDa",
alpha=.4)
axes[2].legend(loc="center left")

p = 0.97
for i, ax in enumerate(axes.reshape(-1)):
    ax.text(x=0.60, y=p, transform=ax.transAxes, s="Skewness: %f" %
table_1.iloc[:,i].skew(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:poo brown')
    ax.text(x=0.60, y=p - 0.06, transform=ax.transAxes, s="Kurtosis:
%f" % table_1.iloc[:,i].kurt(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:dried blood')
    ax.text(x=0.60, y=p - 0.12, transform=ax.transAxes, s="Mean: %f"
% table_1.iloc[:,i].mean(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:dodger blue')
    ax.text(x=0.60, y=p - 0.18, transform=ax.transAxes, s="Std Dev.:
%f" % table_1.iloc[:,i].std(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:black')
    ax.text(x=0.97, y=p, transform=ax.transAxes, s="Skewness: %f" %
table_1.iloc[:,i].skew(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:poo brown')
    ax.text(x=0.97, y=p - 0.06, transform=ax.transAxes, s="Kurtosis:
%f" % table_1.iloc[:,i].kurt(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:dried blood')
    ax.text(x=0.97, y=p - 0.12, transform=ax.transAxes, s="Mean: %f"
% table_1.iloc[:,i].mean(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:dodger blue')
    ax.text(x=0.97, y=p - 0.18, transform=ax.transAxes, s="Std Dev.:
%f" % table_1.iloc[:,i].std(),\
            fontweight='demibold', fontsize=8, verticalalignment='top',
horizontalalignment='right',\
            color='xkcd:black')
plt.tight_layout()

```



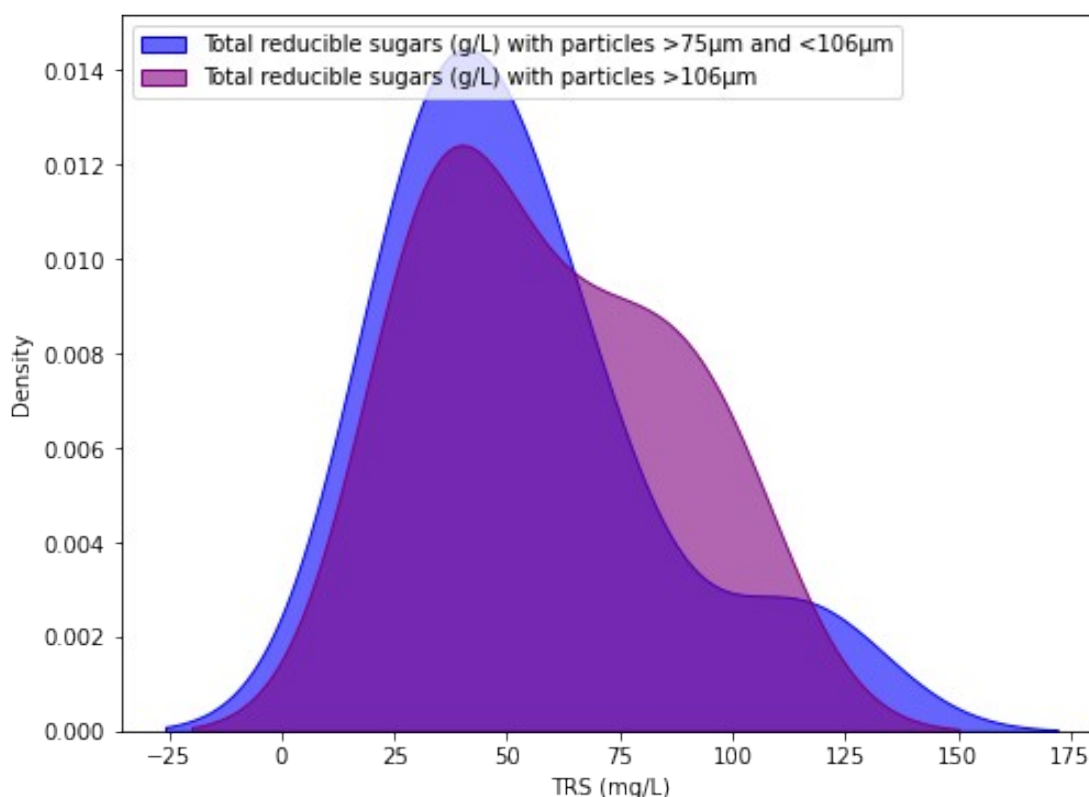


```
fig, ax = plt.subplots(figsize=(8,6))

sns.kdeplot(df[df["Particle size class"]==1]["TRS (g/L)"], shade=True,
ax=ax, color='blue', label="Total reducible sugars (g/L) with
particles >75µm and <106µm", alpha=.6)
sns.kdeplot(df[df["Particle size class"]==2]["TRS (g/L)"], shade=True,
ax=ax, color='purple', label="Total reducible sugars (g/L) with
particles >106µm", alpha=.6)

ax.set_xlabel("TRS (mg/L)")
ax.set_ylabel("Density")
ax.legend(loc="upper left")

<matplotlib.legend.Legend at 0x7f308fae2250>
```



```
fig, ax = plt.subplots(figsize=(8,6))

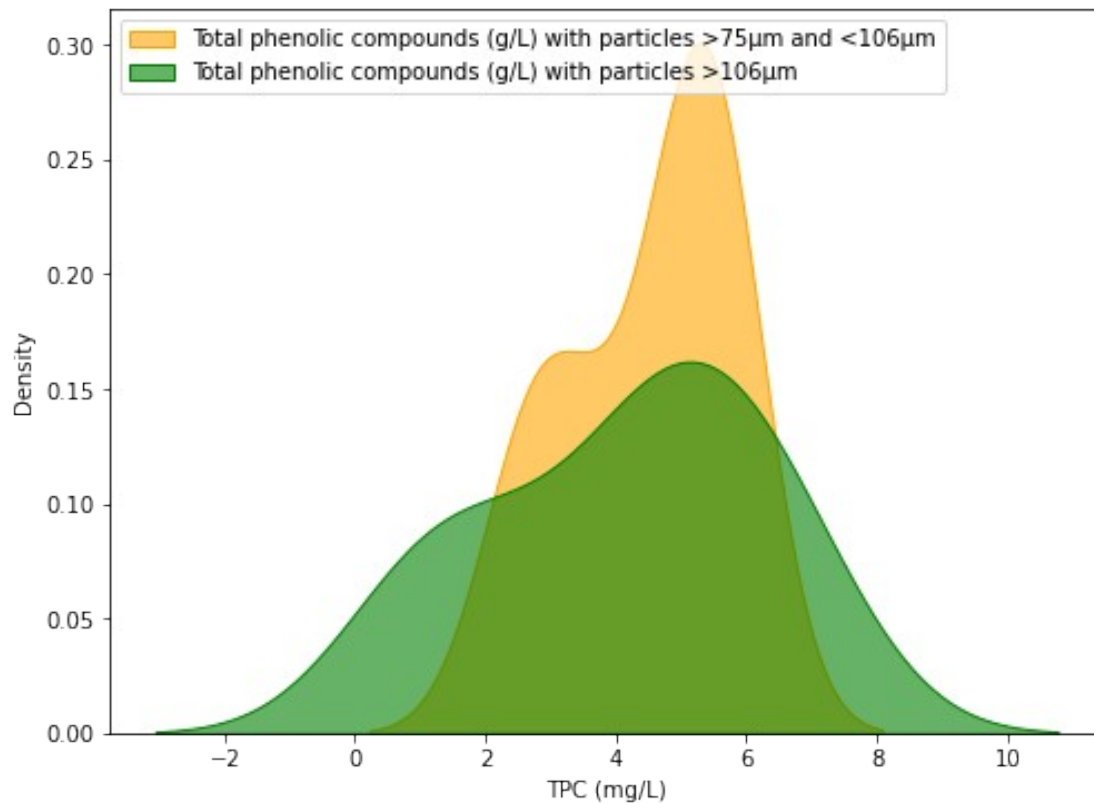
sns.kdeplot(df[df["Particle size class"]==1]["TPC (g/L)"], shade=True,
ax=ax, color='orange', label="Total phenolic compounds (g/L) with
particles >75µm and <106µm", alpha=.6)
sns.kdeplot(df[df["Particle size class"]==2]["TPC (g/L)"], shade=True,
ax=ax, color='green', label="Total phenolic compounds (g/L) with
particles >106µm", alpha=.6)

ax.set_xlabel("TPC (mg/L)")
```



```
ax.set_ylabel("Density")
ax.legend(loc="upper left")
```

<matplotlib.legend.Legend at 0x7f3087dba090>

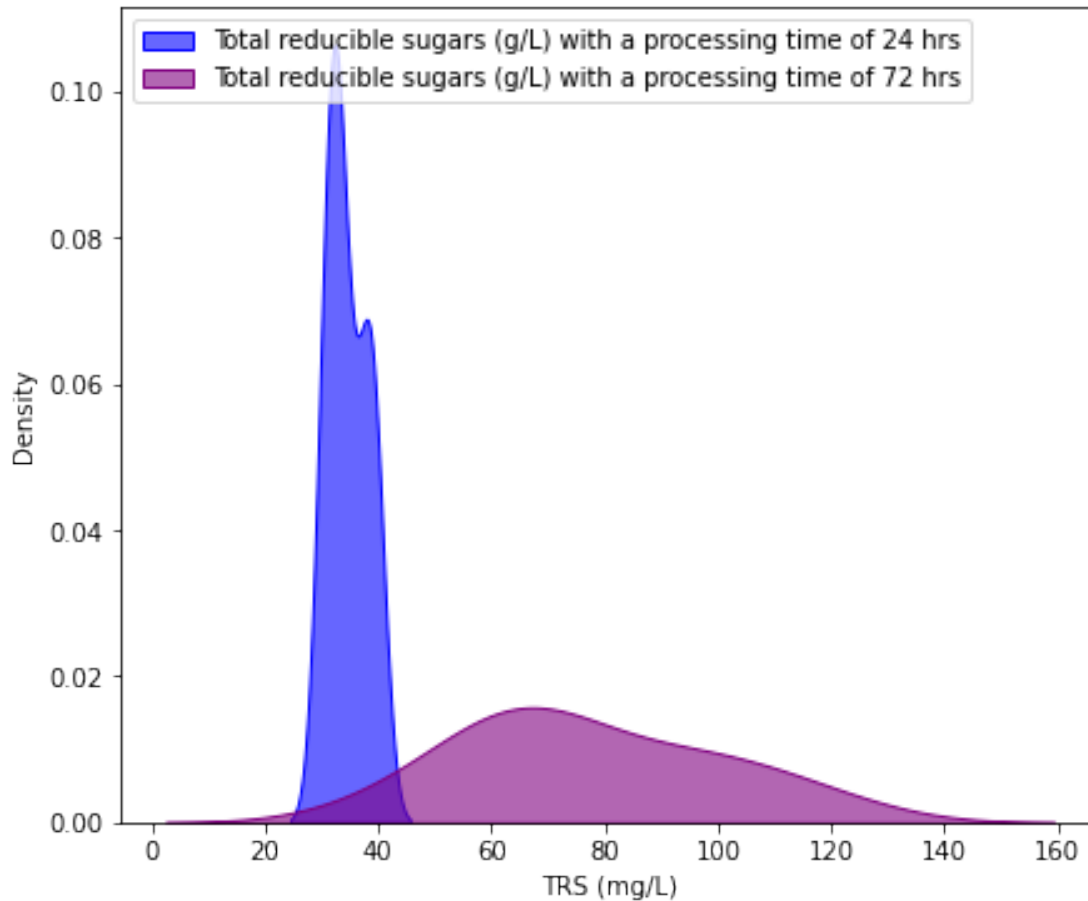


```
fig, ax = plt.subplots(figsize=(7,6))
```

```
sns.kdeplot(df[df["Time (hrs)"]==24]["TRS (g/L)"], shade=True, ax=ax,
color='blue', label="Total reducible sugars (g/L) with a processing
time of 24 hrs", alpha=.6)
sns.kdeplot(df[df["Time (hrs)"]==72]["TRS (g/L)"], shade=True, ax=ax,
color='purple', label="Total reducible sugars (g/L) with a processing
time of 72 hrs", alpha=.6)
```

```
ax.set_xlabel("TRS (mg/L)")
ax.set_ylabel("Density")
ax.legend(loc="upper left")
```

<matplotlib.legend.Legend at 0x7f3087881210>

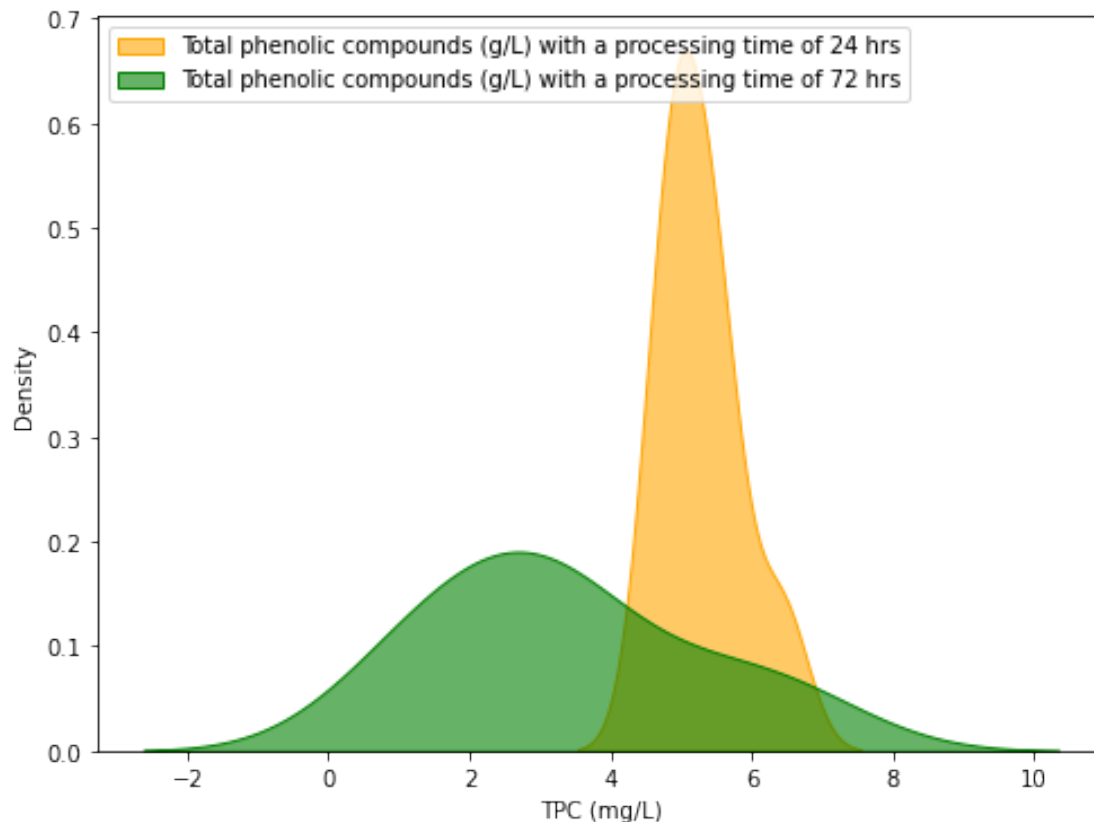


```
fig, ax = plt.subplots(figsize=(8,6))

sns.kdeplot(df[df["Time (hrs)"]==24]["TPC (g/L)"], shade=True, ax=ax,
color='orange', label="Total phenolic compounds (g/L) with a
processing time of 24 hrs", alpha=.6)
sns.kdeplot(df[df["Time (hrs)"]==72]["TPC (g/L)"], shade=True, ax=ax,
color='green', label="Total phenolic compounds (g/L) with a processing
time of 72 hrs", alpha=.6)

ax.set_xlabel("TPC (mg/L)")
ax.set_ylabel("Density")
ax.legend(loc="upper left")

<matplotlib.legend.Legend at 0x7f3087841990>
```



```
df.columns
```

```
Index(['Experiment', 'Enzyme fraction (Kda)', 'Enzyme fraction class',
      'Particle size class', 'Particle size (µm)', 'Time (hrs)', 'TRS
(g/L)',
      'TPC (g/L)'],
      dtype='object')
```

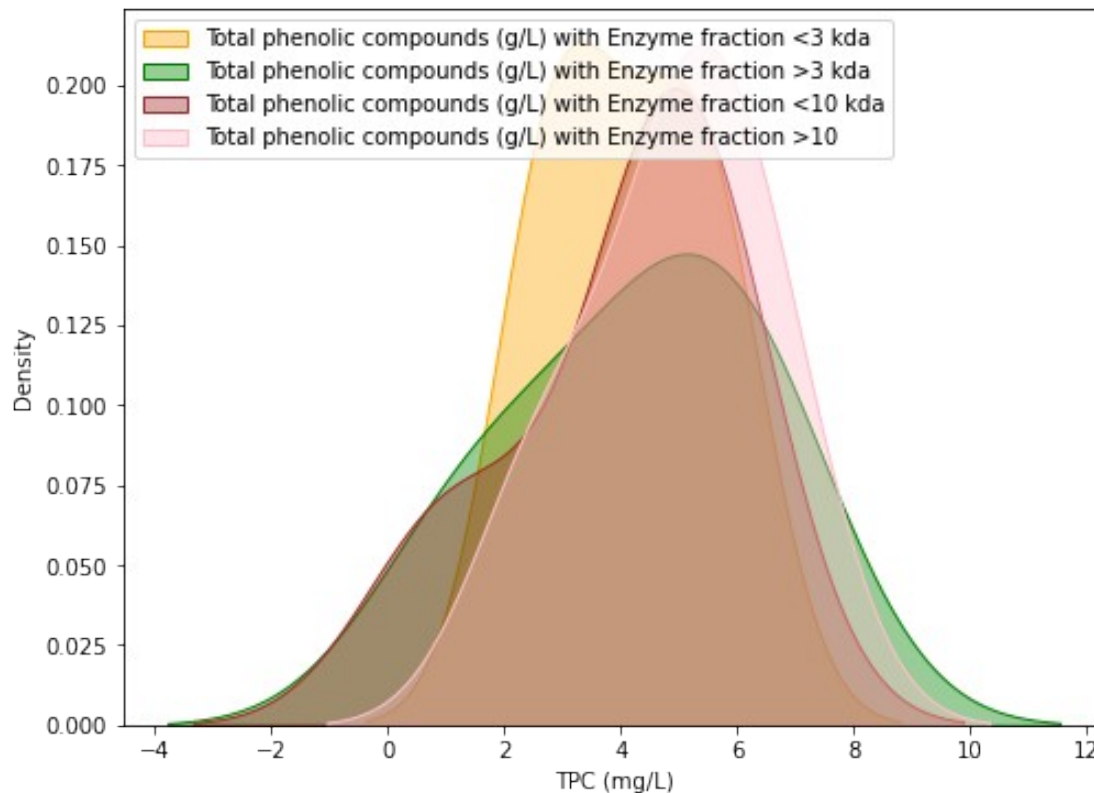
```
fig, ax = plt.subplots(figsize=(8,6))
```

```
sns.kdeplot(df[df["Enzyme fraction class"]==1]["TPC (g/L)"],
            shade=True, ax=ax, color='orange', label="Total phenolic compounds
(g/L) with Enzyme fraction <3 kda", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==2]["TPC (g/L)"],
            shade=True, ax=ax, color='green', label="Total phenolic compounds
(g/L) with Enzyme fraction >3 kda", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==3]["TPC (g/L)"],
            shade=True, ax=ax, color='brown', label="Total phenolic compounds
(g/L) with Enzyme fraction <10 kda", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==4]["TPC (g/L)"],
            shade=True, ax=ax, color='pink', label="Total phenolic compounds (g/L)
with Enzyme fraction >10", alpha=.4)
```

```
ax.set_xlabel("TPC (mg/L)")
```

```
ax.set_ylabel("Density")
ax.legend(loc="upper left")
```

<matplotlib.legend.Legend at 0x7f3087687910>



```
fig, ax = plt.subplots(figsize=(8,6))
```

```
sns.kdeplot(df[df["Enzyme fraction class"]==1]["TRS (g/L)"],
shade=True, ax=ax, color='orange', label="Total reducible sugars (g/L)
with Enzyme fraction <3", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==2]["TRS (g/L)"],
shade=True, ax=ax, color='green', label="Total reducible sugars (g/L)
with Enzyme fraction >3", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==3]["TRS (g/L)"],
shade=True, ax=ax, color='brown', label="Total reducible sugars (g/L)
with Enzyme fraction <10", alpha=.4)
sns.kdeplot(df[df["Enzyme fraction class"]==4]["TRS (g/L)"],
shade=True, ax=ax, color='pink', label="Total reducible sugars (g/L)
with Enzyme fraction >10", alpha=.4)
```

```
ax.set_xlabel("TPC (mg/L)")
ax.set_ylabel("Density")
ax.legend(loc="upper left")
```

<matplotlib.legend.Legend at 0x7f3087628410>

