MSE5820X_Project-1_Johnson

February 25, 2025

1 Project 1

Grant Johnson MSE 5820X

[282]:
```
# Import necessary packages
import math
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
from scipy.stats import pearsonr
import seaborn as sns
import matplotlib.lines as mlines
```

1.1 1 Data Analysis of Gleeble Hardness/Microstructure Data

The Gleeble is a piece of equipment that uses Joule heating to locally heat a sample to introduce changes in the microstructure due to annealing temperature, cooling rate, etc.

1.1.1 1.1 Hardness Profile with Error Bars

The first graphic uses a profile along the gauge length of the specimen using a number of points. Each row corresponds to hardness tests that are 1 mm away from each other along the gauge length.The cell below defines functions for the statistics of this. The error bars in the graphic are the 95% confidence interval.

[283]:
```
# Create a function that passes back the z-value from the Student's T␣
↪Distribution for the 95% confidence interval
def Zvalue(n):
    z = 0
    if n==20:
        z = 2.093
    elif n == 19:
        z = 2.101
    elif n == 18:
        z = 2.110
    elif n == 17:
        z = 2.120
    elif n == 16:
```

```
        z = 2.131
    elif n == 15:
        z = 2.145
    elif n == 14:
        z = 2.160
    elif n == 13:
        z = 2.179
    elif n == 12:
        z = 2.201
    elif n == 11:
        z = 2.228
    elif n == 10:
        z = 2.262
    return z

# Create a function that performs the necessary statistical functions, namely mean, standard deviation, and 95% confidence interval
def Stats(list):
    n = len(list)
    ave = np.average(list)
    std = np.std(list, ddof=1)
    z = Zvalue(n)
    conf = std * z / np.sqrt(n)
    return ave, conf
```

[284]:

```
# Import data for Hardness profile
df21 = pd.read_csv('Gleeble_2101T1_Hardness.csv')
df21
```

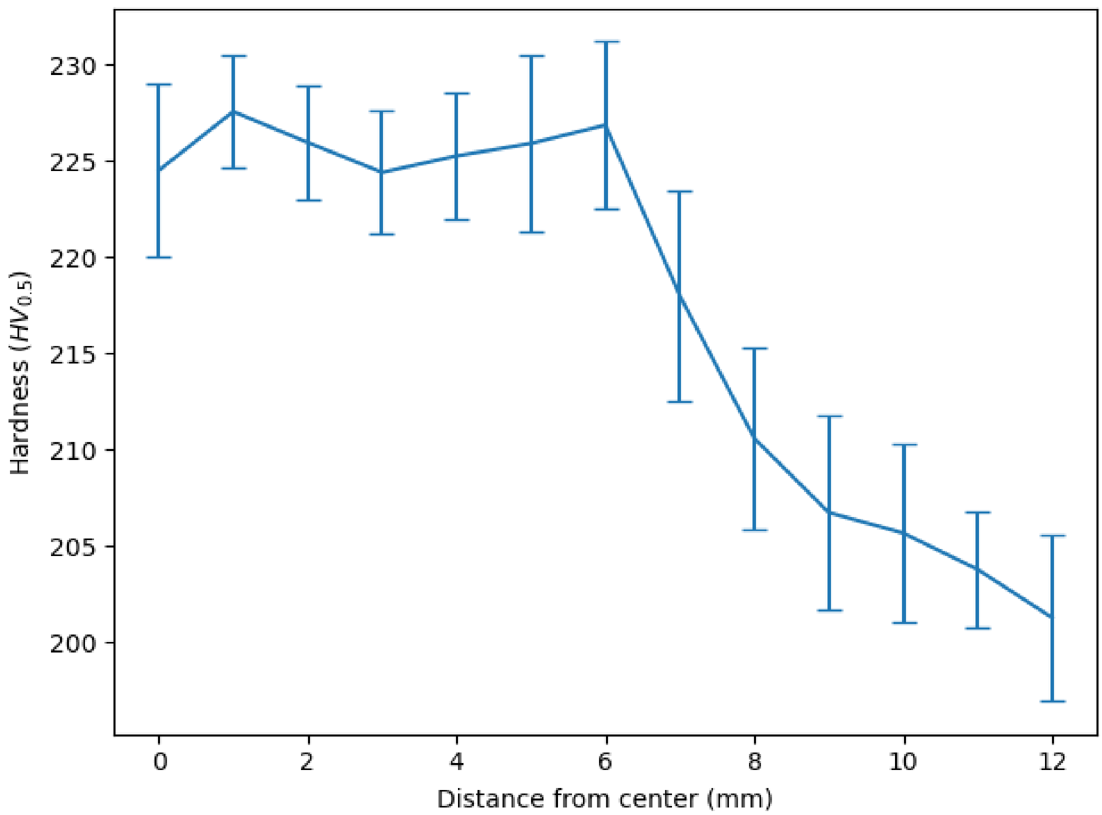
[284]:

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	202	229	229	218	224	224	210	211	214	212	233.0	207	221
1	206	228	225	213	231	223	223	220	196	200	205.0	216	214
2	230	230	214	232	223	224	233	230	226	190	202.0	202	205
3	215	241	221	210	214	226	230	216	219	221	206.0	204	198
4	217	221	234	233	227	240	217	230	213	217	201.0	199	201
5	233	221	221	221	226	233	220	228	209	209	209.0	199	213
6	240	224	220	229	220	215	231	232	214	210	210.0	201	195
7	228	241	228	220	231	229	215	203	208	209	198.0	207	190
8	232	224	223	232	227	220	225	225	219	221	208.0	206	198
9	221	234	229	226	227	223	232	227	197	207	227.0	207	203
10	224	226	232	236	237	248	222	234	191	204	205.0	212	195
11	218	236	225	226	215	205	234	204	205	195	199.0	200	208
12	229	228	217	223	223	232	231	200	206	229	202.0	208	194
13	229	225	231	225	235	210	226	205	228	209	202.0	211	201
14	223	219	219	227	223	224	239	208	219	216	202.0	214	191
15	238	228	235	224	217	221	213	238	211	198	202.0	200	208

16	227	225	230	227	231	223	231	208	220	202	194.0	194	192
17	227	220	221	230	234	234	246	215	215	191	206.0	197	192
18	220	225	236	216	228	235	237	214	198	191	197.0	196	215
19	231	226	229	220	212	229	222	212	204	204	NaN	196	192

[285]:

```
# Initialize lists
sets = []
list_ave = []
list_conf = []

# For each column, get rid of NaN values and use the Stats
# function previously defined to get the average and
# 95% confidence interval and add them both to lists

for col in df21.columns:
    set_n = pd.to_numeric(df21[col]).dropna().values
    sets.append(set_n)

    ave, conf = Stats(set_n)
    list_ave.append(ave)
    list_conf.append(conf)

# Plot line plot with error bars added using 95% confidence interval
plt.errorbar(range(len(list_ave)), list_ave, yerr=list_conf, capsize=5)
plt.xlabel('Distance from center (mm)')
plt.ylabel('Hardness ($HV_{0.5}$)')
plt.tight_layout()
```



1.1.2 1.2 Pearson Correlation Matrix

Analysis of correlation between properties and microstructure. This is used to understand which microstructural features correlate to other microstructural features as well as the hardness.

[286]:
```
# Create a Pandas Dataframe with the data
df_original = pd.read_csv('Gleeble_Data.csv')
perimeter = df_original.pop('Perimeter Fraction - Ferrite')
df_original.insert(9, 'Perimeter Fraction - Ferrite', perimeter)
df = df_original.drop(columns=['Sample','Distance from center']) # Remove the␣
↪columns that can't be used for correlation coefficient measurement
df
```

[286]:
```
    Hardness  Area Fraction - Ferrite  Mean Intercept - Ferrite  \
0    224.500                   58.310                   0.01960
1    227.550                   59.872                   0.01944
2    225.950                   60.905                   0.01925
3    224.400                   60.146                   0.01970
4    225.250                   61.108                   0.02110
5    225.900                   60.462                   0.02136
6    226.850                   61.668                   0.02213
```

7	218.000	60.948	0.02317
8	210.600	62.174	0.02820
9	206.750	59.706	0.02861
10	205.684	59.036	0.03148
11	203.800	55.241	0.03011
12	201.300	54.660	0.03177
13	218.050	60.419	0.02404
14	222.000	62.070	0.02505
15	223.250	58.570	0.02455
16	226.700	61.242	0.02770
17	229.850	58.631	0.02638
18	226.850	61.502	0.02692
19	223.000	61.062	0.02815
20	221.100	62.509	0.03107
21	224.300	61.014	0.03182
22	222.050	57.874	0.03144
23	216.250	54.891	0.02927
24	215.800	58.449	0.03542
25	219.700	56.196	0.03722
26	240.050	78.980	0.02332
27	244.250	78.998	0.02298
28	241.400	79.751	0.02440
29	241.750	78.080	0.02351
30	244.450	79.551	0.02564
31	247.250	80.040	0.02336
32	248.550	78.228	0.02308
33	245.900	77.877	0.02615
34	242.450	75.459	0.02956
35	248.950	73.406	0.02828
36	243.400	68.346	0.02549
37	244.200	67.463	0.02486
38	245.150	67.088	0.02393
39	241.350	68.121	0.02552
40	248.300	68.297	0.02423
41	246.800	70.001	0.02453
42	245.800	68.835	0.02657
43	247.600	66.115	0.02540
44	251.000	68.719	0.02529
45	249.550	71.070	0.02842
46	229.600	67.000	0.02759
47	227.300	64.045	0.03120
48	234.350	62.001	0.02760

	Mean Inverse Intercept - Ferrite	Mean Nearest Neighbor - Ferrite	\
0	153.2538	0.00790	
1	154.4899	0.00893	
2	153.8796	0.00981	

3	156.9621	0.00821
4	147.8884	0.01037
5	144.4656	0.00899
6	149.5963	0.00846
7	151.2930	0.01007
8	145.4572	0.00984
9	145.0801	0.00995
10	138.7236	0.00980
11	139.7309	0.01094
12	124.1582	0.01138
13	160.8898	0.01485
14	159.1685	0.01352
15	170.9181	0.01251
16	151.8022	0.01411
17	160.8095	0.01115
18	150.9376	0.01070
19	155.7744	0.01275
20	143.2028	0.01153
21	139.2476	0.01224
22	149.0552	0.01104
23	150.3371	0.01083
24	126.9429	0.01253
25	129.8316	0.01048
26	148.3800	0.01007
27	152.7000	0.01136
28	144.8700	0.01311
29	146.9300	0.01516
30	133.0800	0.01449
31	167.6000	0.00924
32	177.6800	0.00904
33	166.9700	0.00936
34	154.6500	0.00897
35	149.5900	0.00937
36	126.6443	0.00745
37	136.4853	0.00679
38	129.8744	0.00735
39	119.8798	0.00763
40	127.1412	0.00708
41	131.6589	0.00655
42	116.8838	0.00838
43	124.7735	0.00814
44	133.4531	0.00646
45	119.5470	0.00733
46	117.8363	0.00661
47	113.4243	0.00630
48	108.3189	0.00644

	Mean Average Neighbor - Ferrite	Mean Equivalent Diameter - Ferrite \
0	0.02884	0.00308
1	0.03280	0.00297
2	0.03172	0.00298
3	0.02914	0.00287
4	0.03446	0.00362
5	0.03242	0.00335
6	0.03020	0.00343
7	0.03363	0.00443
8	0.03307	0.00417
9	0.03162	0.00347
10	0.02905	0.00372
11	0.03303	0.00396
12	0.03328	0.00383
13	0.04517	0.00585
14	0.04235	0.00531
15	0.03728	0.00503
16	0.04445	0.00538
17	0.03355	0.00515
18	0.03480	0.00515
19	0.03817	0.00432
20	0.03889	0.00436
21	0.03863	0.00458
22	0.03320	0.00451
23	0.03250	0.00473
24	0.03781	0.00462
25	0.03366	0.00436
26	0.03316	0.00275
27	0.03839	0.00307
28	0.04115	0.00318
29	0.04308	0.00364
30	0.04580	0.00389
31	0.03008	0.00288
32	0.02851	0.00296
33	0.03166	0.00340
34	0.02966	0.00348
35	0.02966	0.00353
36	0.02431	0.00297
37	0.02115	0.00334
38	0.02400	0.00363
39	0.02421	0.00336
40	0.02187	0.00332
41	0.02059	0.00278
42	0.02843	0.00376
43	0.02527	0.00401
44	0.01972	0.00281
45	0.02356	0.00288

46	0.02025	0.00284
47	0.01958	0.00257
48	0.02170	0.00256

	Perimeter Fraction - Ferrite	Mean Roundness - Ferrite	\
0	105.84209	0.67292	
1	108.05757	0.66591	
2	109.67937	0.66170	
3	104.56396	0.66158	
4	103.00866	0.65383	
5	97.59301	0.66888	
6	98.57366	0.66595	
7	90.52619	0.65565	
8	78.02542	0.65286	
9	72.71754	0.65415	
10	66.79651	0.64864	
11	65.38474	0.66998	
12	59.46019	0.66812	
13	86.78765	0.61765	
14	82.42330	0.63325	
15	79.54571	0.62601	
16	78.51186	0.62512	
17	72.24089	0.62817	
18	74.63544	0.64772	
19	77.26384	0.63652	
20	72.19610	0.64017	
21	70.56164	0.63644	
22	65.41272	0.63576	
23	67.78556	0.63630	
24	61.87719	0.65924	
25	51.47358	0.65990	
26	140.04188	0.69279	
27	141.38577	0.69779	
28	128.91032	0.69045	
29	131.69406	0.68622	
30	126.30851	0.69010	
31	136.88990	0.69438	
32	138.89600	0.69723	
33	118.14263	0.70343	
34	100.85276	0.70359	
35	97.91358	0.70774	
36	113.21703	0.69779	
37	113.67964	0.69753	
38	114.81712	0.69608	
39	108.93672	0.70356	
40	116.92546	0.70293	
41	119.73931	0.70264	

42	103.84728	0.70393
43	107.18885	0.69735
44	112.73474	0.70566
45	102.82460	0.70278
46	101.61360	0.70938
47	84.85848	0.71058
48	89.06463	0.70593

	Area Fraction - Austenite	Mean Intercept - Austenite \
0	41.690	0.01269
1	40.128	0.01220
2	39.095	0.01161
3	39.854	0.01206
4	38.892	0.01240
5	39.538	0.01294
6	38.332	0.01243
7	39.052	0.01350
8	37.826	0.01574
9	40.294	0.01774
10	40.964	0.02000
11	44.759	0.02406
12	45.340	0.02685
13	39.581	0.01546
14	37.930	0.01466
15	41.430	0.01631
16	38.758	0.01567
17	41.369	0.01735
18	38.498	0.01666
19	38.938	0.01717
20	37.491	0.01802
21	38.986	0.01930
22	42.126	0.02230
23	45.109	0.02314
24	41.551	0.02437
25	43.804	0.02949
26	21.020	0.00673
27	21.002	0.00668
28	20.249	0.00697
29	21.920	0.00741
30	20.449	0.00730
31	19.960	0.00646
32	21.772	0.00701
33	22.123	0.00858
34	24.541	0.01124
35	26.594	0.01241
36	31.654	0.01092
37	32.537	0.01105

38	32.912	0.01107
39	31.879	0.01121
40	31.703	0.01065
41	29.999	0.00986
42	31.165	0.01192
43	33.885	0.01243
44	31.281	0.01102
45	28.930	0.01153
46	33.000	0.01280
47	35.955	0.01660
48	37.999	0.01584

	Mean Inverse Intercept - Austenite	Mean Nearest Neighbor - Austenite \
0	139.7774	0.01634
1	143.4454	0.01598
2	150.1647	0.01413
3	144.5700	0.01576
4	142.9964	0.01605
5	137.6376	0.01721
6	144.3766	0.01634
7	142.0857	0.01677
8	140.1642	0.01627
9	141.6000	0.01471
10	137.0224	0.01438
11	125.9825	0.01344
12	105.6388	0.01578
13	140.8495	0.01498
14	144.4057	0.01490
15	143.1647	0.01478
16	140.7715	0.01700
17	134.6026	0.01588
18	142.0102	0.01596
19	145.1660	0.01495
20	148.3932	0.01490
21	144.9996	0.01571
22	134.4456	0.01639
23	129.2328	0.01556
24	125.2627	0.01631
25	106.6900	0.01722
26	226.8800	0.00717
27	227.4400	0.00707
28	220.5400	0.00764
29	214.5800	0.00824
30	214.5400	0.00842
31	241.1100	0.00695
32	237.2500	0.00677
33	225.7600	0.00705

34	204.8500	0.00744
35	188.7400	0.00728
36	161.9437	0.01301
37	166.1110	0.01202
38	164.5542	0.01312
39	160.2694	0.01326
40	168.6409	0.01169
41	176.8403	0.01111
42	157.1459	0.01196
43	162.3956	0.01109
44	180.6860	0.01079
45	184.5539	0.00996
46	178.9364	0.00971
47	167.9322	0.01050
48	173.5860	0.01098

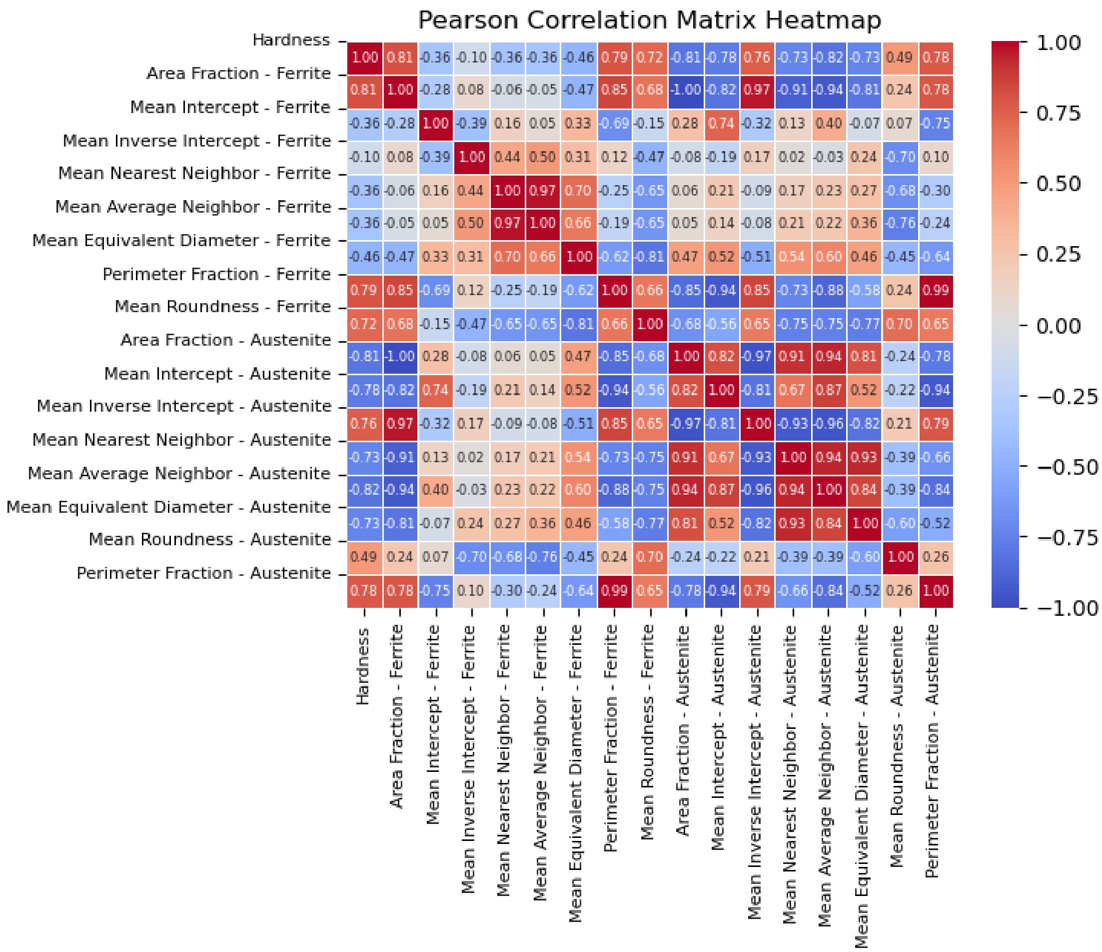
	Mean Average Neighbor - Austenite	Mean Equivalent Diameter - Austenite \
0	0.03784	0.01446
1	0.03686	0.01361
2	0.03297	0.01182
3	0.03650	0.01367
4	0.03707	0.01381
5	0.03859	0.01457
6	0.03717	0.01304
7	0.04102	0.01292
8	0.04082	0.01154
9	0.03945	0.01047
10	0.04172	0.01085
11	0.04157	0.01023
12	0.04794	0.01182
13	0.03797	0.01156
14	0.03749	0.01147
15	0.03705	0.01051
16	0.04044	0.01262
17	0.04104	0.01086
18	0.03945	0.01081
19	0.03829	0.01104
20	0.03969	0.01048
21	0.04338	0.01089
22	0.04313	0.01047
23	0.04589	0.01021
24	0.04628	0.01139
25	0.05296	0.01241
26	0.01771	0.00513
27	0.01717	0.00502
28	0.01878	0.00550
29	0.02017	0.00572

30	0.02049	0.00570
31	0.01797	0.00494
32	0.01776	0.00482
33	0.01947	0.00463
34	0.02183	0.00478
35	0.02187	0.00464
36	0.03057	0.00869
37	0.03026	0.00671
38	0.03290	0.00810
39	0.03158	0.00809
40	0.02759	0.00643
41	0.02723	0.00659
42	0.02935	0.00700
43	0.02832	0.00640
44	0.02680	0.00581
45	0.02449	0.00512
46	0.02431	0.00478
47	0.02760	0.00504
48	0.02948	0.00584

	Mean Roundness - Austenite	Perimeter Fraction - Austenite
0	0.61706	110.09240
1	0.61043	111.36840
2	0.61591	112.33990
3	0.61524	107.72711
4	0.60052	105.46279
5	0.60926	99.72018
6	0.62318	101.95247
7	0.62418	92.62049
8	0.61795	79.45474
9	0.61741	74.51297
10	0.61789	68.47362
11	0.63400	67.41116
12	0.63554	60.56054
13	0.61307	89.04080
14	0.61854	83.66781
15	0.62247	82.45415
16	0.62135	81.55681
17	0.62174	73.79129
18	0.62284	75.11709
19	0.61286	78.68125
20	0.61315	72.66786
21	0.61119	71.54017
22	0.61858	66.91269
23	0.61016	70.28938
24	0.60588	63.18054
25	0.61784	53.20667

26	0.62046	132.96804
27	0.62838	134.83625
28	0.62362	123.15026
29	0.61919	125.63513
30	0.61812	119.99902
31	0.62568	130.42571
32	0.62267	132.80770
33	0.62864	111.93677
34	0.62871	96.08251
35	0.62842	92.30758
36	0.65383	113.96244
37	0.66714	116.31294
38	0.65080	115.85799
39	0.65840	109.74235
40	0.66456	117.63661
41	0.65997	121.45707
42	0.67511	102.15824
43	0.67218	107.29663
44	0.66759	114.49442
45	0.67932	100.53219
46	0.68626	102.12388
47	0.68500	87.33570
48	0.66459	91.35140

[287]:
```
# Creating a correlation coefficient matrix
pearson_corr = df.corr(method='pearson')

# Calculating correlation and p-values for variables of interest
r_value, p_value = pearsonr(df['Hardness'],df['Perimeter Fraction - Austenite'])
r_value1, p_value1 = pearsonr(df['Hardness'],df['Area Fraction - Ferrite'])

# Plot the heatmap using the seaborn package
ax = sns.heatmap(
    pearson_corr,
    annot=True,
    cmap='coolwarm',
    fmt='.2f',
    linewidths=0.5,
    annot_kws={"size":6}
    )
plt.title('Pearson Correlation Matrix Heatmap')
plt.xticks(fontsize=8)
ax.set_yticks(range(len(pearson_corr)))
ax.set_yticklabels(pearson_corr.index, rotation=0, fontsize=8)

plt.show()
```



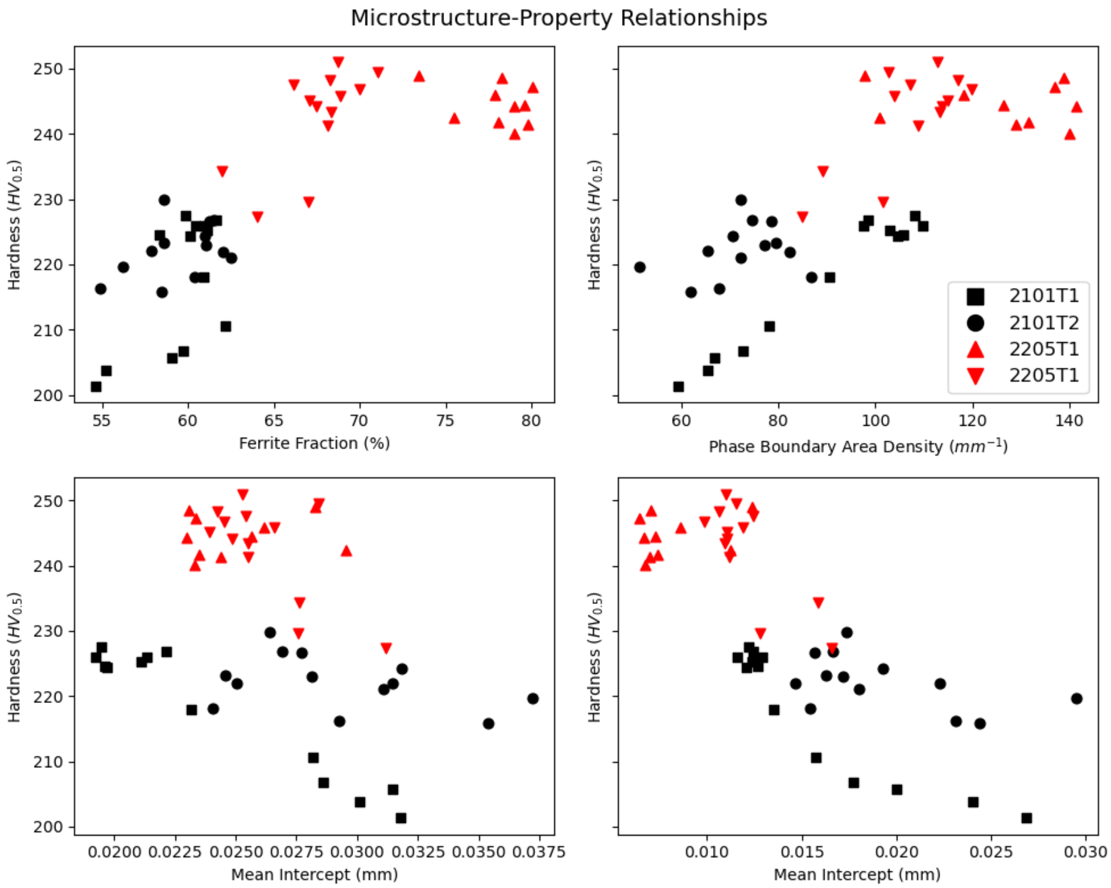
1.1.3 1.3 Subplots of Microstructure/Property Relationships

Using the Pearson correlation matrix, we can plot a few of the relationships that we have seen. Specifically, the phase fraction, the phase sizes, and the phase boundary perimeter fraction.

[288]:
```
df0 = df_original

# Create a dictionary for what the color and shape of each scatter plot point
marker_map = {
    '2101T1': 's',
    '2101T2': 'o',
    '2205T1': '^',
    '2205T2': 'v'
}
color_map = {
    '2101T1': 'k',
    '2101T2': 'k',
```

```
    '2205T1': 'r',
    '2205T2': 'r'
}

# Create a 2x2 plot matrix
fig, ((ax0, ax1), (ax2, ax3)) = plt.subplots(2, 2, sharey=True, figsize=(10,8))
#fig.text(0.05,0.5,r'Hardness ($HV_{0.5}$)', va='center', rotation=90,␣
 ↪fontsize=12)
fig.suptitle('Microstructure-Property Relationships',fontsize=14)
for _, row in df0.iterrows():
    # Use the previous dictionary to assign a color and a shape based on the␣
 ↪sample column
    ax0.scatter(row['Area Fraction - Ferrite'], row['Hardness'],
                marker=marker_map[row['Sample']],
                color=color_map[row['Sample']], s=40)
    ax1.scatter(row['Perimeter Fraction - Ferrite'], row['Hardness'],
                marker=marker_map[row['Sample']],
                color=color_map[row['Sample']], s=40)
    ax2.scatter(row['Mean Intercept - Ferrite'], row['Hardness'],
                marker=marker_map[row['Sample']],
                color=color_map[row['Sample']], s=40)
    ax3.scatter(row['Mean Intercept - Austenite'], row['Hardness'],
                marker=marker_map[row['Sample']],
                color=color_map[row['Sample']], s=40)

#ax0.set_title('Hardness vs. Ferrite Fraction', fontsize=11)
ax0.set_xlabel('Ferrite Fraction (%)')
ax0.set_ylabel(r'Hardness ($HV_{0.5}$)')
#ax1.set_title('Hardness vs. Phase Boundary Fraction', fontsize=11)
ax1.set_xlabel('Phase Boundary Area Density ($mm^{-1}$)')
ax1.set_ylabel(r'Hardness ($HV_{0.5}$)')
#ax2.set_title('Hardness vs. Ferrite Mean Intercept', fontsize=11)
ax2.set_xlabel('Mean Intercept (mm)')
ax2.set_ylabel(r'Hardness ($HV_{0.5}$)')
#ax3.set_title('Hardness vs. Austenite Mean Intercept', fontsize=11)
ax3.set_xlabel('Mean Intercept (mm)')
ax3.set_ylabel(r'Hardness ($HV_{0.5}$)')

# Create a legend
black_square = mlines.
 ↪Line2D([],[],color='k',marker='s',linestyle='None',markersize=10,label='2101T1')
black_circle = mlines.
 ↪Line2D([],[],color='k',marker='o',linestyle='None',markersize=10,label='2101T2')
red_up = mlines.Line2D([],[],␣
 ↪color='r',marker='^',linestyle='None',markersize=10,label='2205T1')
red_down = mlines.Line2D([],[],␣
 ↪color='r',marker='v',linestyle='None',markersize=10,label='2205T1')
```

```
ax1.legend(handles=[black_square, black_circle, red_up, red_down], loc='lower right',fontsize=12)

plt.tight_layout()
plt.show()
```



1.2 2 Data Analysis of Porosity in L-DED Sample

A separate dataset, this data includes information about defects within a laser directed energy deposition build of 2205 duplex stainless steel. The information includes the location, the size, and the shape of the defects.

[289]:
```
df1 = pd.read_csv('Pre-S1_Porosity.csv')
df1 = df1.drop(['Unnamed: 10','Unnamed: 11','avg','st dev'], axis=1)
df1
```

[289]:
```
   Feature  Area (um^2)  Roundness  CentroidX (um)  CentroidY (um)  \
0     1938    20.228864   1.073835     7532.205622     1271.324358
1     2343    21.914603   1.068422     8990.169230     2247.136570
```

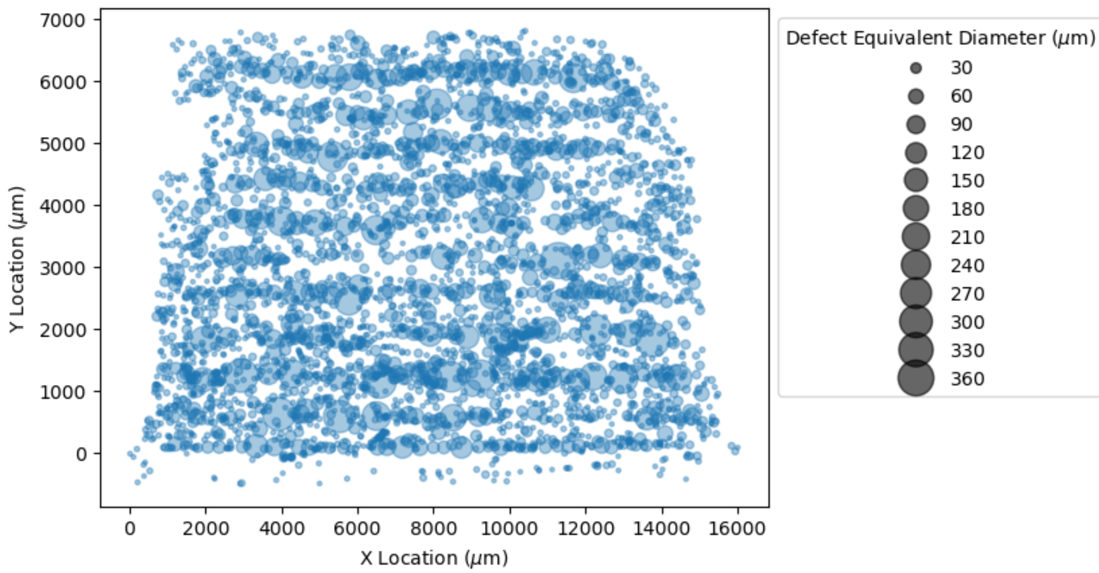
```
2         3017    21.914603   1.068422    11214.259910     4423.187930
3         3605    21.914603   1.068422    13646.737370     3895.404590
4         1872    21.493168   1.058099     7307.539222     1188.304896
…          …         …          …            …            …
3892        87  1517.164821   0.156178     1164.103625     7122.672042
3893      1293   206.924424   0.151512     5298.580717     5703.856783
3894      1213   271.403929   0.136949     5092.539888     5485.956073
3895      1249   297.111444   0.121646     5208.686982     6838.742971
3896      1510   131.909052   0.108585     6158.218961     4062.039555

      First Moment of Area (um^3)  Eccentricity  Equivalent Diameter (um)  \
0                       34.211366      0.104211                  5.075056
1                       38.458722      0.000000                  5.282285
2                       38.458722      0.000000                  5.282285
3                       38.458722      0.000000                  5.282285
4                       37.405651      0.260868                  5.231248
…                             …           …                       …
3892                 97337.013720      0.993401                 43.951271
3893                  5083.596222      0.999510                 16.231585
3894                  8683.724579      0.998557                 18.589304
3895                 13695.897910      0.981735                 19.449783
3896                  4125.912918      0.995060                 12.959623

      Nearest Neighbor Distance (um)  Average Neighbor Distance (um)
0                          96.337506                      240.048939
1                         107.661792                      220.165750
2                         147.851573                      298.032672
3                          92.825953                      155.324208
4                          93.983282                      292.442586
…                                …                            …
3892                       95.234881                      150.218060
3893                       52.467702                      109.163800
3894                       95.184914                      193.835165
3895                       19.095491                       61.437162
3896                      161.462618                      259.043767

[3897 rows x 10 columns]
```

1.2.1 2.1 Location and Size Graphical Representation

This graphic shows where the defects occur and how large each defect is as a bubble. The defects are concentrated in lines along the layers of the build.

[290]:
```
# Flip and translate data in y-direction to represent it in the correct orientation relative to original figure.
df1['CentroidY_new'] = -df1['CentroidY (um)'] + 7700
```

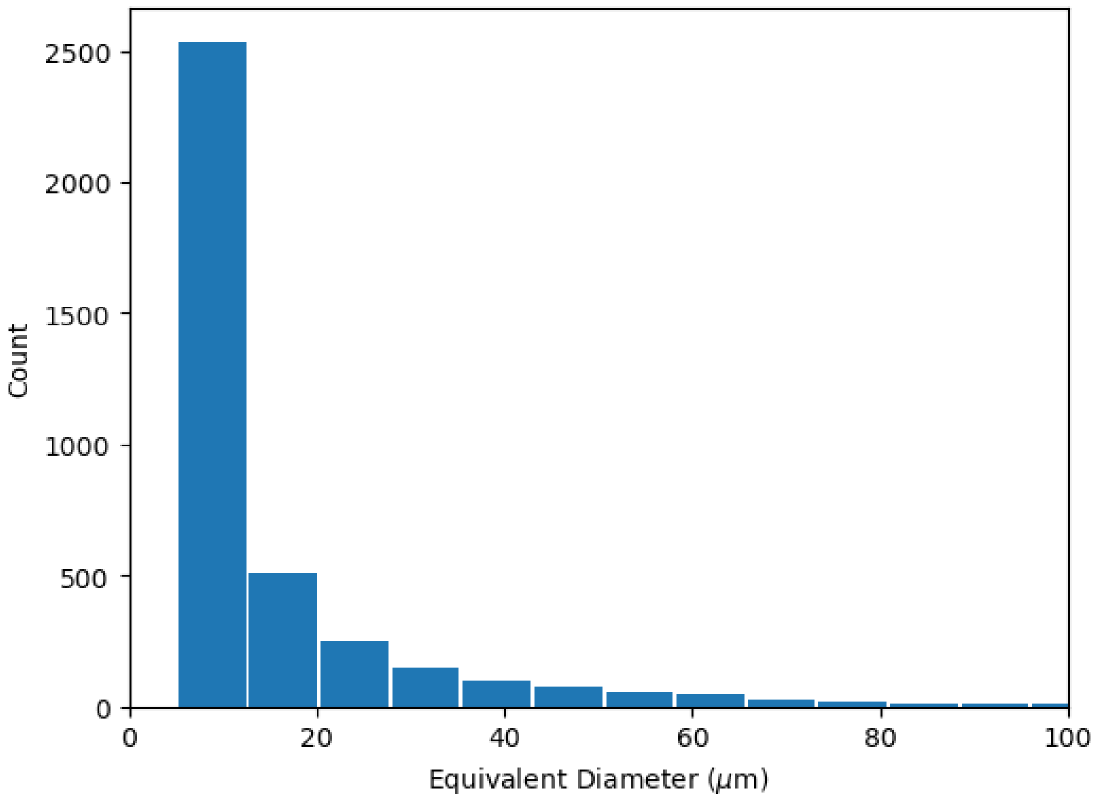
```
fig, ax = plt.subplots()

scatter = ax.scatter(df1['CentroidX (um)'], df1['CentroidY_new'],␣
 ↪s=df1['Equivalent Diameter (um)'], alpha=0.4)
handles, labels = scatter.legend_elements(prop='sizes', alpha=0.6)
legend2 = ax.legend(handles, labels, title=r'Defect Equivalent Diameter␣
 ↪($\mu$m)', bbox_to_anchor=(1,1))
plt.xlabel(r'X Location ($\mu$m)')
plt.ylabel(r'Y Location ($\mu$m)')

plt.show()
```



1.2.2 2.2 Defect Size Histogram

To see the size distribution of the defects, a histogram of the defect sizes is shown.

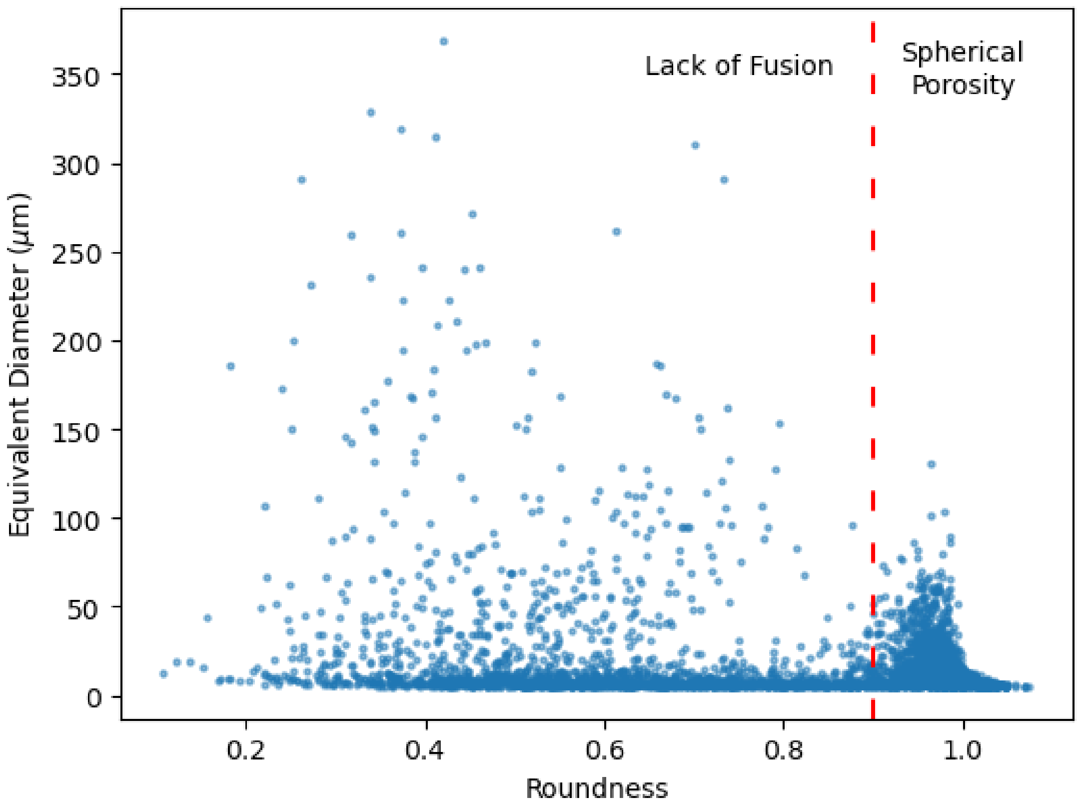
[291]:
```
plt.hist(df1['Equivalent Diameter (um)'], bins=48,rwidth=0.95)
plt.xlim(0,100)
plt.xlabel(r'Equivalent Diameter ($\mu$m)')
plt.ylabel('Count')

plt.show()
```



1.2.3 2.3 Equivalent Diameter vs. Roundness

One way to distinguish between lack of fusion defects (defects between layers due to insufficient energy that lack remelting) and spherical porosity (from gas entrapment or keyhole) can be seen by using the roundness (a measure of the perimeter to the area where 1 is a perfect circle. 0.9 is used as a threshold above which the defect is labeled as spherical porosity.

[292]:
```
fig, ax = plt.subplots()

scatter = ax.scatter(df1['Roundness'],df1['Equivalent Diameter (um)'],alpha=0.
 ↪5,s=5)
plt.axvline(x=0.9, color='r', linestyle=(0, (5, 8)))
plt.xlabel('Roundness')
plt.ylabel(r'Equivalent Diameter ($\mu$m)')
ax.text(0.75,350, 'Lack of Fusion',horizontalalignment='center')
ax.text(1.0,340, 'Spherical\nPorosity',horizontalalignment='center')

plt.show()
```


Lack of Fusion
Spherical Porosity
Equivalent Diameter (μm)
Roundness
0
50
100
150
200
250
300
350
0.2
0.4
0.6
0.8
1.0

[ ]: