

Sample

April 27, 2021

```
[1]: import pandas as pd
from copy import deepcopy
import matplotlib.pyplot as plt
import re
import numpy as np
from matplotlib import gridspec
import matplotlib
```

1 Helper functions

These are borrowed from the `Convert.ipynb` file.

```
[2]: headings = ['Building Identifier',
                 'Country',
                 'City',
                 'Quality / Stage of Data',
                 'Construction Date',
                 'Building Type',
                 'Gross Floor Area']
```

```
[3]: df = pd.read_excel('../Dataset/dataset.xlsx',header=1).drop('Unnamed: 0',axis=1)
```

```
[4]: df
```

```
[4]:   Building Identifier Country City Quality / Stage of Data \
0              1      CA  TOR              00IFC
1              2      CA  TOR              00IFC
2              3      CA  TOR              00IFC
3              4      CA  TOR              00IFC
4              5      CA  TOR              00IFC
5              6      CA  TOR              00IFC
6              7      CA  TOR              00IFC
7              8      CA  TOR              00IFC
8              9      CA  TOR              00IFC
9             10      CA  TOR              00IFC
10            11      CA  TOR              00IFC
11            12      CA  TOR              00IFC
```

12	13	CA	TOR	00IFC
13	14	CA	TOR	00IFC
14	15	CA	TOR	00IFC
15	16	CA	TOR	00IFC
16	17	CA	TOR	00IFC
17	18	CA	TOR	00IFC
18	19	CA	TOR	00IFC
19	20	CA	TOR	00IFC
20	21	CA	TOR	00IFC
21	22	CA	TOR	00IFC
22	23	CA	TOR	00IFC
23	24	CA	TOR	00IFC
24	25	CA	TOR	00IFC
25	26	CA	TOR	00IFC
26	27	CA	WIN	00IFC
27	28	CA	TOR	00IFC
28	29	CA	TOR	00IFC
29	30	CA	TOR	00IFC
30	31	CA	TOR	00IFC
31	32	CA	TOR	00IFC
32	33	CA	TOR	00IFC
33	34	CA	TOR	00IFC
34	35	CA	TOR	00IFC
35	36	CA	TOR	00IFC
36	37	CA	TOR	00IFC
37	38	CA	TOR	00IFC
38	39	CA	TOR	00IFC
39	40	US	NEW	00IFC
40	41	CA	TOR	00IFC
41	42	CA	TOR	00IFC
42	43	CA	TOR	00IFC
43	44	CA	TOR	00IFC
44	45	CA	TOR	00IFC
45	46	CA	TOR	00IFC
46	47	CA	TOR	00IFC
47	48	CA	RIC	0IARC
48	49	CA	TOR	00IFC
49	50	CA	TOR	00IFC
50	51	CA	TOR	00IFC
51	52	CA	TOR	00IFC
52	53	CA	TOR	00IFC
53	54	CA	TOR	00IFC
54	55	CA	TOR	00IFC
55	56	CA	TOR	00IFC
56	57	CA	TOR	00IFC
57	58	CA	TOR	00IFC
58	59	CA	TOR	0IFBP

59

60

CA TOR

OIFBP

	Construction Date	Building Type	Gross Floor Area \
0	2021	SND	521.18
1	2021	SND	389.24
2	2021	SND	411.64
3	2021	SND	269.56
4	2011	OFF	11248.00
5	2011	APB	11317.00
6	2021	SND	445.99
7	2021	SND	438.45
8	2021	SND	714.07
9	2021	SND	343.24
10	2009	OFF	73083.00
11	1917	SMR	199.93
12	2021	SND	226.89
13	2021	SND	611.73
14	2021	SND	343.44
15	2021	SND	613.38
16	1969	SNR	413.72
17	1969	SNR	333.49
18	2021	SND	178.38
19	2021	SND	323.80
20	2020	SND	837.56
21	2021	SND	587.86
22	2021	SND	568.21
23	2021	SMD	234.73
24	2021	SND	294.84
25	2021	SND	496.77
26	2007	OFF	73600.00
27	2021	SND	643.30
28	2021	SND	701.61
29	2021	SMD	257.75
30	2021	SND	378.70
31	2021	SND	324.16
32	2020	SND	533.53
33	2020	SMD	254.05
34	2021	SND	423.03
35	2021	SND	328.16
36	2021	SND	421.59
37	2020	SND	628.59
38	2021	SND	464.51
39	2017	EDU	8983.00
40	2021	SND	346.14
41	1913	SNR	161.08
42	2021	SND	891.97
43	2021	SND	525.61

44	2021	SND	502.87
45	2021	SND	379.18
46	2021	SND	549.65
47	2016	EDU	6819.00
48	2020	SND	393.82
49	2021	SND	648.14
50	1988	INS	21934.00
51	2018	APB	53146.02
52	2018	MIX	33975.25
53	2017	APB	69784.00
54	2017	APB	39409.04
55	2016	APB	53871.00
56	2020	LNW	137.23
57	2020	LNW	144.92
58	2019	LNW	83.10
59	2021	LNW	234.79

	000_G2010.20.000_03 00 00.00_kg_1	000_B1010.20.000_03 00 00.00_kg_1 \
0	NaN	NaN
1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	13704.0	1.776816e+06
5	NaN	1.514400e+06
6	NaN	NaN
7	NaN	NaN
8	NaN	NaN
9	NaN	NaN
10	58008.0	4.029264e+06
11	NaN	NaN
12	NaN	NaN
13	NaN	NaN
14	NaN	NaN
15	NaN	NaN
16	NaN	NaN
17	NaN	NaN
18	NaN	NaN
19	NaN	NaN
20	NaN	NaN
21	NaN	NaN
22	NaN	NaN
23	NaN	NaN
24	NaN	NaN
25	NaN	NaN
26	NaN	4.480680e+06
27	NaN	NaN
28	NaN	NaN

29	NaN	NaN
30	NaN	NaN
31	NaN	NaN
32	NaN	NaN
33	NaN	NaN
34	NaN	NaN
35	NaN	NaN
36	NaN	NaN
37	NaN	NaN
38	NaN	NaN
39	NaN	2.191431e+04
40	NaN	NaN
41	NaN	NaN
42	NaN	NaN
43	NaN	NaN
44	NaN	NaN
45	NaN	NaN
46	NaN	NaN
47	NaN	3.756000e+04
48	NaN	NaN
49	NaN	NaN
50	NaN	NaN
51	NaN	NaN
52	NaN	NaN
53	NaN	NaN
54	NaN	NaN
55	NaN	NaN
56	NaN	NaN
57	NaN	NaN
58	NaN	NaN
59	NaN	NaN

	000_C1010.10.000_04	22	00.00_kg_1	...	000_B2010.10.000_07	46	16.00_kg_2	\
0			NaN	...			NaN	
1			NaN	...			NaN	
2			NaN	...			NaN	
3			NaN	...			NaN	
4			19397.560000	...			NaN	
5			53877.650000	...			NaN	
6			NaN	...			NaN	
7			NaN	...			NaN	
8			NaN	...			NaN	
9			NaN	...			NaN	
10			562574.500000	...			NaN	
11			NaN	...			NaN	
12			NaN	...			NaN	
13			NaN	...			NaN	

14		NaN	...	NaN
15		NaN	...	NaN
16		NaN	...	NaN
17		NaN	...	NaN
18		NaN	...	NaN
19		NaN	...	NaN
20		NaN	...	NaN
21		NaN	...	NaN
22		NaN	...	NaN
23		NaN	...	NaN
24		NaN	...	NaN
25		NaN	...	NaN
26	354208.227500		...	NaN
27		NaN	...	NaN
28		NaN	...	NaN
29		NaN	...	NaN
30		NaN	...	NaN
31		NaN	...	NaN
32		NaN	...	NaN
33		NaN	...	NaN
34		NaN	...	NaN
35		NaN	...	NaN
36		NaN	...	NaN
37		NaN	...	NaN
38		NaN	...	NaN
39	8666.292723		...	NaN
40		NaN	...	NaN
41		NaN	...	NaN
42		NaN	...	NaN
43		NaN	...	NaN
44		NaN	...	NaN
45		NaN	...	NaN
46		NaN	...	NaN
47		NaN	...	NaN
48		NaN	...	NaN
49		NaN	...	NaN
50		NaN	...	NaN
51	8194.250000		...	NaN
52	191988.905000		...	NaN
53	82694.400000		...	NaN
54	46298.790000		...	NaN
55	422839.793489		...	NaN
56		NaN	...	NaN
57		NaN	...	NaN
58		NaN	...	NaN
59		NaN	...	67.3

	001_B2010.80.000_07 27 00.00_kg_2	001_B2010.80.000_07 21 13.00_kg_2 \
0	NaN	NaN
1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN
5	NaN	NaN
6	NaN	NaN
7	NaN	NaN
8	NaN	NaN
9	NaN	NaN
10	NaN	NaN
11	NaN	NaN
12	NaN	NaN
13	NaN	NaN
14	NaN	NaN
15	NaN	NaN
16	NaN	NaN
17	NaN	NaN
18	NaN	NaN
19	NaN	NaN
20	NaN	NaN
21	NaN	NaN
22	NaN	NaN
23	NaN	NaN
24	NaN	NaN
25	NaN	NaN
26	NaN	NaN
27	NaN	NaN
28	NaN	NaN
29	NaN	NaN
30	NaN	NaN
31	NaN	NaN
32	NaN	NaN
33	NaN	NaN
34	NaN	NaN
35	NaN	NaN
36	NaN	NaN
37	NaN	NaN
38	NaN	NaN
39	NaN	NaN
40	NaN	NaN
41	NaN	NaN
42	NaN	NaN
43	NaN	NaN
44	NaN	NaN
45	NaN	NaN

46	NaN	NaN
47	NaN	NaN
48	NaN	NaN
49	NaN	NaN
50	NaN	NaN
51	NaN	NaN
52	NaN	NaN
53	NaN	NaN
54	NaN	NaN
55	NaN	NaN
56	NaN	NaN
57	NaN	NaN
58	NaN	NaN
59	37.3	112.67

	001_B2010.10.000_09 24 23.00_kg_2	OB1_A5020.10.000_06 11 00.00_kg_2	\
0	NaN	NaN	
1	NaN	NaN	
2	NaN	NaN	
3	NaN	NaN	
4	NaN	NaN	
5	NaN	NaN	
6	NaN	NaN	
7	NaN	NaN	
8	NaN	NaN	
9	NaN	NaN	
10	NaN	NaN	
11	NaN	NaN	
12	NaN	NaN	
13	NaN	NaN	
14	NaN	NaN	
15	NaN	NaN	
16	NaN	NaN	
17	NaN	NaN	
18	NaN	NaN	
19	NaN	NaN	
20	NaN	NaN	
21	NaN	NaN	
22	NaN	NaN	
23	NaN	NaN	
24	NaN	NaN	
25	NaN	NaN	
26	NaN	NaN	
27	NaN	NaN	
28	NaN	NaN	
29	NaN	NaN	
30	NaN	NaN	

31	NaN	NaN
32	NaN	NaN
33	NaN	NaN
34	NaN	NaN
35	NaN	NaN
36	NaN	NaN
37	NaN	NaN
38	NaN	NaN
39	NaN	NaN
40	NaN	NaN
41	NaN	NaN
42	NaN	NaN
43	NaN	NaN
44	NaN	NaN
45	NaN	NaN
46	NaN	NaN
47	NaN	NaN
48	NaN	NaN
49	NaN	NaN
50	NaN	NaN
51	NaN	NaN
52	NaN	NaN
53	NaN	NaN
54	NaN	NaN
55	NaN	NaN
56	NaN	NaN
57	NaN	NaN
58	NaN	NaN
59	2655.54	277.59

	OB1_A5020.10.000_06 11 00.00_kg_1	OB1_A5020.10.000_09 21 16.00_kg_1	\
0	NaN	NaN	
1	NaN	NaN	
2	NaN	NaN	
3	NaN	NaN	
4	NaN	NaN	
5	NaN	NaN	
6	NaN	NaN	
7	NaN	NaN	
8	NaN	NaN	
9	NaN	NaN	
10	NaN	NaN	
11	NaN	NaN	
12	NaN	NaN	
13	NaN	NaN	
14	NaN	NaN	
15	NaN	NaN	

16	NaN	NaN
17	NaN	NaN
18	NaN	NaN
19	NaN	NaN
20	NaN	NaN
21	NaN	NaN
22	NaN	NaN
23	NaN	NaN
24	NaN	NaN
25	NaN	NaN
26	NaN	NaN
27	NaN	NaN
28	NaN	NaN
29	NaN	NaN
30	NaN	NaN
31	NaN	NaN
32	NaN	NaN
33	NaN	NaN
34	NaN	NaN
35	NaN	NaN
36	NaN	NaN
37	NaN	NaN
38	NaN	NaN
39	NaN	NaN
40	NaN	NaN
41	NaN	NaN
42	NaN	NaN
43	NaN	NaN
44	NaN	NaN
45	NaN	NaN
46	NaN	NaN
47	NaN	NaN
48	NaN	NaN
49	NaN	NaN
50	NaN	NaN
51	NaN	NaN
52	NaN	NaN
53	NaN	NaN
54	NaN	NaN
55	NaN	NaN
56	NaN	NaN
57	NaN	NaN
58	NaN	NaN
59	889.66	854.98

000_C1010.10.000_07 21 13.00_kg_1	00R_B3010.90.000_07 21 13.00_kg_1 \
NaN	NaN

1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN
5	NaN	NaN
6	NaN	NaN
7	NaN	NaN
8	NaN	NaN
9	NaN	NaN
10	NaN	NaN
11	NaN	NaN
12	NaN	NaN
13	NaN	NaN
14	NaN	NaN
15	NaN	NaN
16	NaN	NaN
17	NaN	NaN
18	NaN	NaN
19	NaN	NaN
20	NaN	NaN
21	NaN	NaN
22	NaN	NaN
23	NaN	NaN
24	NaN	NaN
25	NaN	NaN
26	NaN	NaN
27	NaN	NaN
28	NaN	NaN
29	NaN	NaN
30	NaN	NaN
31	NaN	NaN
32	NaN	NaN
33	NaN	NaN
34	NaN	NaN
35	NaN	NaN
36	NaN	NaN
37	NaN	NaN
38	NaN	NaN
39	NaN	NaN
40	NaN	NaN
41	NaN	NaN
42	NaN	NaN
43	NaN	NaN
44	NaN	NaN
45	NaN	NaN
46	NaN	NaN
47	NaN	NaN

48	NaN	NaN
49	NaN	NaN
50	NaN	NaN
51	NaN	NaN
52	NaN	NaN
53	NaN	NaN
54	NaN	NaN
55	NaN	NaN
56	NaN	NaN
57	NaN	NaN
58	NaN	NaN
59	127.47	420.29

00R_B1020.20.000_07 51 13.00_kg_1

0	NaN
1	NaN
2	NaN
3	NaN
4	NaN
5	NaN
6	NaN
7	NaN
8	NaN
9	NaN
10	NaN
11	NaN
12	NaN
13	NaN
14	NaN
15	NaN
16	NaN
17	NaN
18	NaN
19	NaN
20	NaN
21	NaN
22	NaN
23	NaN
24	NaN
25	NaN
26	NaN
27	NaN
28	NaN
29	NaN
30	NaN
31	NaN
32	NaN

33	NaN
34	NaN
35	NaN
36	NaN
37	NaN
38	NaN
39	NaN
40	NaN
41	NaN
42	NaN
43	NaN
44	NaN
45	NaN
46	NaN
47	NaN
48	NaN
49	NaN
50	NaN
51	NaN
52	NaN
53	NaN
54	NaN
55	NaN
56	NaN
57	NaN
58	NaN
59	315.22

[60 rows x 2090 columns]

```
[5]: name_conversion = pd.read_csv('name_conversion.csv')
      building_name_conversion = pd.read_csv('building_type_name_conversion.csv')
```

```
[6]: building_name_map = {k['Building Code']:k['Building Type'] for _,k in
      ↪building_name_conversion.iterrows() }
```

```
[7]: name_map = {k.Code:k.Category for _,k in name_conversion.iterrows() }
```

```
[8]: additional_categories_map = {v:k for k,v in {
      'Continuous Footings':'OCF',
      'Foundation Walls':'OFW',
      'Spread Footings':'OSF',
      'Column Piers':'OCP',
      'Columns Supporting Floors':'CSF',
      'Floor Girders and Beams':'FGB',
      'Floor Trusses':'OFT',
      'Floor Joists':'OFJ',
```

```

'Columns Supporting Roofs':'CSR',
'Roof Girders and Beams':'RGB',
'Roof Trusses':'ORT',
'Roof Joists':'ORJ',
'Parking Bumpers':'OPB',
'Precast Concrete Stair Treads':'PCS',
'Roof Curbs':'ORC',
'Exterior Wall Construction':'EWC',
'Composite Decking':'CPD',
'Cast-in-Place concrete':'CIC',
'Floor Structural Frame':'FSF',
'Associated Metal Fabrications':'AMF',
'Floor Construction Supplementary Components':'FCS',
'Roof Construction Supplementary Components':'RCS',
'Residential Elevators':'ORE',
'Vegetated Low-Slope Roofing':'VLR',
'Swimming Pools':'SWP',
'Excavation Soil Anchors':'ESA',
'Floor Trusses':'FTS',
'Roof Window and Skylight Performance':'RWS',
'Rainwater Storage Tanks':'RST',
'Gray Water Tanks':'GWT'}.items()
}

additional_categories_map['OFT'] = 'Floor Trusses'

```

2 1. Plot sample figures

Here we plot building material mass.

```

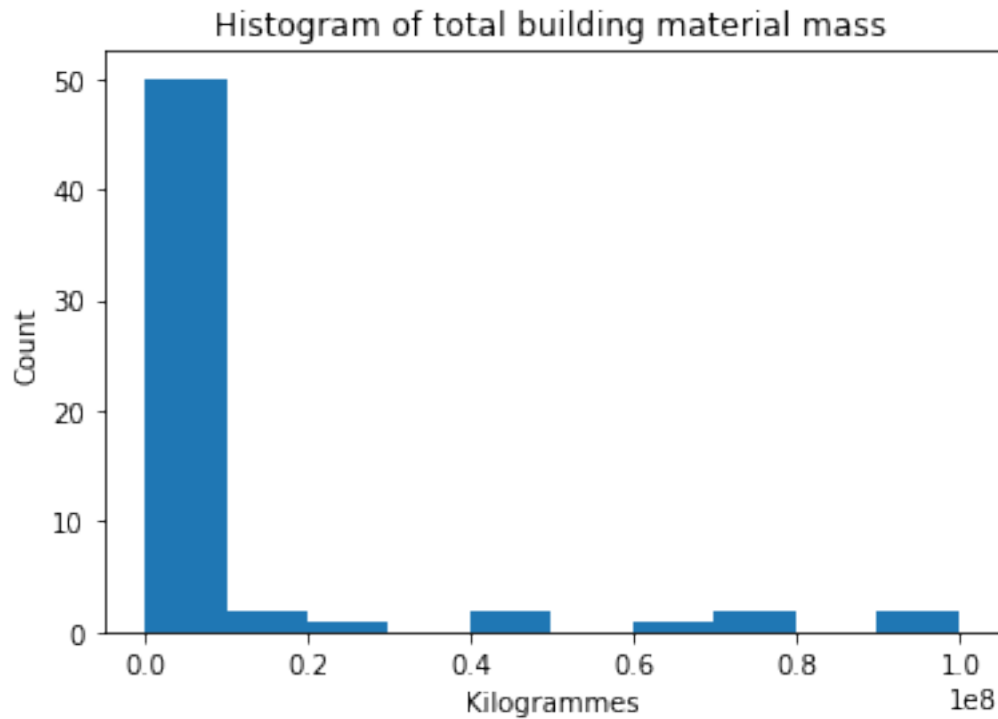
[9]: plt.hist(df[[c for c in df.columns if 'kg' in c]].sum(axis=1));
plt.title('Histogram of total building material mass')
plt.xlabel('Kilogrammes')
plt.ylabel('Count');

```

```

[9]: Text(0, 0.5, 'Count')

```



3 2. Investigate a specific material

In this example, we select only columns that match the MasterFormat code for Structural Concrete. Then, we aggregate based on Level 2 UniFormat code.

```
[10]: cols = [d for d in df.columns if '03 31 00' in d]
```

```
[11]: f = lambda x: re.split('[_\\.\\ ]',x)[1][0:3]
concrete_df = pd.concat([df[headings],df[cols].groupby(f,axis=1).sum()],axis=1).
    ↪rename(columns=name_map)
```

```
[12]: concrete_df
```

```
[12]: Building Identifier Country City Quality / Stage of Data \
0          1          CA  TOR          00IFC
1          2          CA  TOR          00IFC
2          3          CA  TOR          00IFC
3          4          CA  TOR          00IFC
4          5          CA  TOR          00IFC
5          6          CA  TOR          00IFC
6          7          CA  TOR          00IFC
7          8          CA  TOR          00IFC
8          9          CA  TOR          00IFC
```

9	10	CA	TOR	00IFC
10	11	CA	TOR	00IFC
11	12	CA	TOR	00IFC
12	13	CA	TOR	00IFC
13	14	CA	TOR	00IFC
14	15	CA	TOR	00IFC
15	16	CA	TOR	00IFC
16	17	CA	TOR	00IFC
17	18	CA	TOR	00IFC
18	19	CA	TOR	00IFC
19	20	CA	TOR	00IFC
20	21	CA	TOR	00IFC
21	22	CA	TOR	00IFC
22	23	CA	TOR	00IFC
23	24	CA	TOR	00IFC
24	25	CA	TOR	00IFC
25	26	CA	TOR	00IFC
26	27	CA	WIN	00IFC
27	28	CA	TOR	00IFC
28	29	CA	TOR	00IFC
29	30	CA	TOR	00IFC
30	31	CA	TOR	00IFC
31	32	CA	TOR	00IFC
32	33	CA	TOR	00IFC
33	34	CA	TOR	00IFC
34	35	CA	TOR	00IFC
35	36	CA	TOR	00IFC
36	37	CA	TOR	00IFC
37	38	CA	TOR	00IFC
38	39	CA	TOR	00IFC
39	40	US	NEW	00IFC
40	41	CA	TOR	00IFC
41	42	CA	TOR	00IFC
42	43	CA	TOR	00IFC
43	44	CA	TOR	00IFC
44	45	CA	TOR	00IFC
45	46	CA	TOR	00IFC
46	47	CA	TOR	00IFC
47	48	CA	RIC	0IARC
48	49	CA	TOR	00IFC
49	50	CA	TOR	00IFC
50	51	CA	TOR	00IFC
51	52	CA	TOR	00IFC
52	53	CA	TOR	00IFC
53	54	CA	TOR	00IFC
54	55	CA	TOR	00IFC
55	56	CA	TOR	00IFC

56	57	CA	TOR	00IFC
57	58	CA	TOR	00IFC
58	59	CA	TOR	0IFBP
59	60	CA	TOR	0IFBP

	Construction Date	Building Type	Gross Floor Area	Foundations	\
0	2021	SND	521.18	1.709236e+05	
1	2021	SND	389.24	1.082862e+05	
2	2021	SND	411.64	1.909299e+05	
3	2021	SND	269.56	6.736923e+04	
4	2011	OFF	11248.00	0.000000e+00	
5	2011	APB	11317.00	0.000000e+00	
6	2021	SND	445.99	1.295202e+05	
7	2021	SND	438.45	1.174431e+05	
8	2021	SND	714.07	1.927680e+05	
9	2021	SND	343.24	9.564723e+04	
10	2009	OFF	73083.00	0.000000e+00	
11	1917	SMR	199.93	9.927316e+04	
12	2021	SND	226.89	5.835472e+04	
13	2021	SND	611.73	2.061282e+05	
14	2021	SND	343.44	1.436814e+05	
15	2021	SND	613.38	1.789777e+05	
16	1969	SNR	413.72	9.293583e+04	
17	1969	SNR	333.49	1.186380e+05	
18	2021	SND	178.38	6.408230e+04	
19	2021	SND	323.80	4.733438e+04	
20	2020	SND	837.56	2.605656e+05	
21	2021	SND	587.86	2.455371e+05	
22	2021	SND	568.21	1.415184e+05	
23	2021	SMD	234.73	8.560216e+04	
24	2021	SND	294.84	7.580863e+04	
25	2021	SND	496.77	1.205336e+05	
26	2007	OFF	73600.00	0.000000e+00	
27	2021	SND	643.30	9.718853e+04	
28	2021	SND	701.61	1.810933e+05	
29	2021	SMD	257.75	8.183304e+04	
30	2021	SND	378.70	1.477228e+05	
31	2021	SND	324.16	1.188635e+05	
32	2020	SND	533.53	1.627046e+05	
33	2020	SMD	254.05	8.882102e+04	
34	2021	SND	423.03	9.980270e+04	
35	2021	SND	328.16	1.238544e+05	
36	2021	SND	421.59	1.760423e+05	
37	2020	SND	628.59	2.298828e+05	
38	2021	SND	464.51	1.886381e+05	
39	2017	EDU	8983.00	0.000000e+00	
40	2021	SND	346.14	9.748630e+04	

41	1913	SNR	161.08	5.362299e+04
42	2021	SND	891.97	2.157609e+05
43	2021	SND	525.61	2.567725e+05
44	2021	SND	502.87	1.372402e+05
45	2021	SND	379.18	1.437386e+05
46	2021	SND	549.65	1.435894e+05
47	2016	EDU	6819.00	0.000000e+00
48	2020	SND	393.82	7.294707e+04
49	2021	SND	648.14	2.216331e+05
50	1988	INS	21934.00	0.000000e+00
51	2018	APB	53146.02	1.115822e+07
52	2018	MIX	33975.25	4.220040e+06
53	2017	APB	69784.00	7.912944e+06
54	2017	APB	39409.04	9.350736e+06
55	2016	APB	53871.00	1.627512e+06
56	2020	LNW	137.23	3.111394e+04
57	2020	LNW	144.92	3.241172e+04
58	2019	LNW	83.10	3.347723e+04
59	2021	LNW	234.79	8.400714e+04

	Subgrade	Enclosures	Slabs-On-Grade	Substructure	Interior	\
0		0.0	6.721219e+04		0.0	
1		0.0	3.576043e+04		0.0	
2		0.0	3.246461e+04		0.0	
3		0.0	1.595211e+04		0.0	
4		0.0	0.000000e+00		0.0	
5		0.0	0.000000e+00		0.0	
6		0.0	3.521918e+04		0.0	
7		0.0	4.289057e+04		0.0	
8		0.0	8.446873e+04		0.0	
9		0.0	2.033114e+04		0.0	
10		0.0	0.000000e+00		0.0	
11		0.0	1.971760e+04		0.0	
12		0.0	1.435987e+04		0.0	
13		0.0	4.140039e+04		0.0	
14		0.0	2.246836e+04		0.0	
15		0.0	4.219445e+04		0.0	
16		0.0	3.376814e+04		0.0	
17		0.0	2.622366e+04		0.0	
18		0.0	2.343862e+04		0.0	
19		0.0	2.368485e+04		0.0	
20		0.0	6.344851e+04		0.0	
21		0.0	6.865710e+04		0.0	
22		0.0	6.684690e+04		0.0	
23		0.0	1.294360e+04		0.0	
24		0.0	1.791821e+04		0.0	
25		0.0	5.137996e+04		0.0	

26	0.0	0.000000e+00	0.0
27	0.0	5.230228e+04	0.0
28	0.0	6.233222e+04	0.0
29	0.0	1.211886e+04	0.0
30	0.0	3.514722e+04	0.0
31	0.0	2.011968e+04	0.0
32	0.0	3.674638e+04	0.0
33	0.0	1.160387e+04	0.0
34	0.0	3.329286e+04	0.0
35	0.0	1.931159e+04	0.0
36	0.0	3.304437e+04	0.0
37	0.0	5.528816e+04	0.0
38	0.0	2.866777e+04	0.0
39	0.0	0.000000e+00	0.0
40	0.0	2.237098e+04	0.0
41	0.0	1.235658e+04	0.0
42	0.0	5.949332e+04	0.0
43	0.0	3.378685e+04	0.0
44	0.0	3.951047e+04	0.0
45	0.0	2.913799e+04	0.0
46	0.0	3.506390e+04	0.0
47	0.0	0.000000e+00	0.0
48	0.0	3.364275e+04	0.0
49	0.0	6.099032e+04	0.0
50	0.0	0.000000e+00	0.0
51	2728008.0	3.647520e+05	11033448.0
52	1705680.0	3.834720e+05	5400288.0
53	3246168.0	1.407000e+06	14052000.0
54	3567720.0	9.045840e+05	7607280.0
55	3438168.0	7.174800e+05	22907184.0
56	0.0	1.439848e+04	0.0
57	0.0	2.000253e+04	0.0
58	0.0	5.412759e+03	0.0
59	0.0	1.962799e+04	0.0

	Substructure Related Activities	Superstructure \
0	0.0	1.938810e+03
1	0.0	1.397610e+03
2	0.0	1.528710e+02
3	0.0	1.212090e+01
4	0.0	0.000000e+00
5	0.0	0.000000e+00
6	0.0	5.332590e+02
7	0.0	1.970790e+03
8	0.0	4.049670e+03
9	0.0	9.440170e+02
10	0.0	0.000000e+00

11	0.0	0.000000e+00
12	0.0	9.785830e+02
13	0.0	5.381500e+02
14	0.0	0.000000e+00
15	0.0	0.000000e+00
16	0.0	0.000000e+00
17	0.0	7.514840e+03
18	0.0	0.000000e+00
19	0.0	2.111800e+03
20	0.0	3.270810e+03
21	0.0	2.533580e+03
22	0.0	6.016340e+02
23	0.0	1.827610e+03
24	0.0	5.977480e+02
25	0.0	2.540900e+03
26	0.0	0.000000e+00
27	0.0	7.189470e+02
28	0.0	2.276420e+02
29	0.0	1.587900e+03
30	0.0	1.096510e+04
31	0.0	5.530400e+03
32	0.0	1.360980e+03
33	0.0	2.177290e+03
34	0.0	6.524310e+02
35	0.0	3.944150e+03
36	0.0	4.401230e+02
37	0.0	8.518740e+02
38	0.0	2.593160e+03
39	0.0	0.000000e+00
40	0.0	2.360810e+02
41	0.0	0.000000e+00
42	0.0	8.599660e+02
43	0.0	1.038810e+03
44	0.0	4.881840e+02
45	0.0	1.267510e+03
46	0.0	1.154890e+03
47	0.0	0.000000e+00
48	0.0	1.835120e+02
49	0.0	1.041320e+03
50	0.0	0.000000e+00
51	133464.0	2.780006e+07
52	112872.0	2.226535e+07
53	169896.0	3.204622e+07
54	276264.0	1.483577e+07
55	93048.0	3.239134e+07
56	0.0	0.000000e+00
57	0.0	0.000000e+00

58	0.0	0.000000e+00
59	0.0	0.000000e+00

	Exterior Vertical Enclosures	Exterior Horizontal Enclosures \
0	0.0	0.0
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	0.0
5	0.0	0.0
6	0.0	0.0
7	0.0	0.0
8	0.0	0.0
9	0.0	0.0
10	0.0	0.0
11	0.0	0.0
12	0.0	0.0
13	0.0	0.0
14	0.0	0.0
15	0.0	0.0
16	0.0	0.0
17	0.0	0.0
18	0.0	0.0
19	0.0	0.0
20	0.0	0.0
21	0.0	0.0
22	0.0	0.0
23	0.0	0.0
24	0.0	0.0
25	0.0	0.0
26	0.0	0.0
27	0.0	0.0
28	0.0	0.0
29	0.0	0.0
30	0.0	0.0
31	0.0	0.0
32	0.0	0.0
33	0.0	0.0
34	0.0	0.0
35	0.0	0.0
36	0.0	0.0
37	0.0	0.0
38	0.0	0.0
39	0.0	0.0
40	0.0	0.0
41	0.0	0.0
42	0.0	0.0

43	0.0	0.0
44	0.0	0.0
45	0.0	0.0
46	0.0	0.0
47	0.0	0.0
48	0.0	0.0
49	0.0	0.0
50	0.0	0.0
51	727896.0	537984.0
52	405408.0	392400.0
53	328032.0	799872.0
54	119088.0	0.0
55	159336.0	0.0
56	0.0	0.0
57	0.0	0.0
58	0.0	0.0
59	0.0	0.0

	Interior Construction	Conveying	Plumbing	Special Construction \
0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0
8	11307.2	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0
12	0.0	0.0	0.0	0.0
13	0.0	0.0	0.0	0.0
14	0.0	0.0	0.0	0.0
15	0.0	0.0	0.0	0.0
16	0.0	0.0	0.0	0.0
17	0.0	0.0	0.0	0.0
18	0.0	0.0	0.0	0.0
19	0.0	0.0	0.0	0.0
20	0.0	0.0	0.0	0.0
21	0.0	0.0	0.0	0.0
22	0.0	0.0	0.0	0.0
23	0.0	0.0	0.0	0.0
24	0.0	0.0	0.0	0.0
25	0.0	0.0	0.0	0.0
26	0.0	0.0	0.0	0.0
27	0.0	0.0	0.0	0.0

28	0.0	0.0	0.0	0.0
29	0.0	0.0	0.0	0.0
30	0.0	0.0	0.0	0.0
31	0.0	0.0	0.0	0.0
32	0.0	0.0	0.0	0.0
33	0.0	0.0	0.0	0.0
34	0.0	0.0	0.0	0.0
35	0.0	0.0	0.0	0.0
36	0.0	0.0	0.0	0.0
37	0.0	0.0	0.0	0.0
38	0.0	0.0	0.0	0.0
39	0.0	0.0	0.0	0.0
40	0.0	0.0	0.0	0.0
41	0.0	0.0	0.0	0.0
42	0.0	0.0	0.0	0.0
43	0.0	0.0	0.0	0.0
44	0.0	0.0	0.0	0.0
45	0.0	0.0	0.0	0.0
46	0.0	0.0	0.0	0.0
47	0.0	0.0	0.0	0.0
48	0.0	0.0	0.0	0.0
49	0.0	0.0	0.0	0.0
50	0.0	0.0	0.0	0.0
51	6816696.0	2494560.0	0.0	80592.0
52	5893176.0	1829328.0	48816.0	62280.0
53	9050592.0	2304480.0	172032.0	0.0
54	5180976.0	861888.0	130152.0	0.0
55	5604960.0	1664448.0	0.0	220992.0
56	0.0	0.0	0.0	0.0
57	0.0	0.0	0.0	0.0
58	0.0	0.0	0.0	0.0
59	0.0	0.0	0.0	0.0

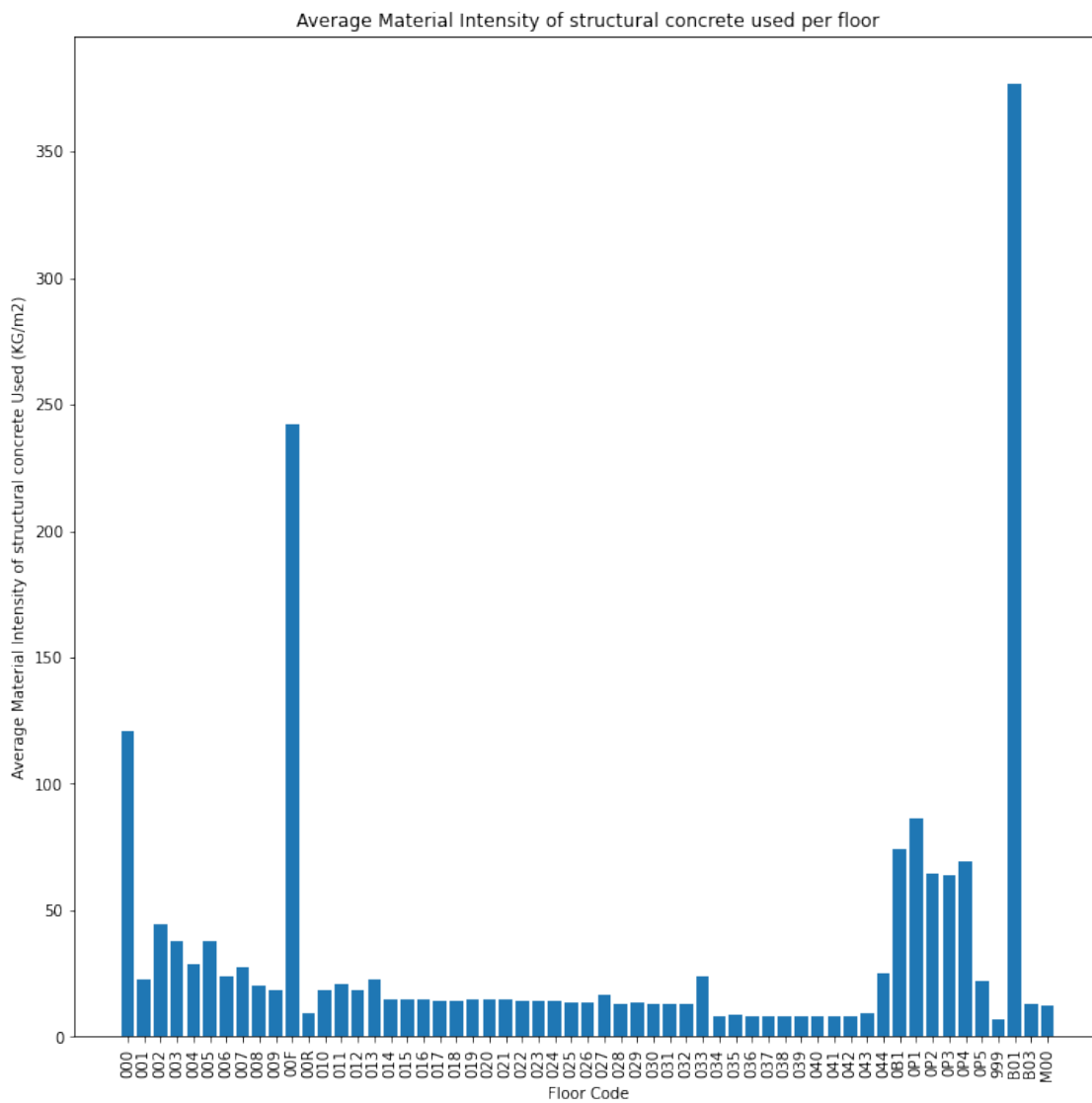
Site Improvements

0	0.0
1	0.0
2	0.0
3	0.0
4	0.0
5	0.0
6	0.0
7	0.0
8	0.0
9	0.0
10	0.0
11	0.0
12	0.0

13	0.0
14	0.0
15	0.0
16	0.0
17	0.0
18	0.0
19	0.0
20	0.0
21	0.0
22	0.0
23	0.0
24	0.0
25	0.0
26	0.0
27	0.0
28	0.0
29	0.0
30	0.0
31	0.0
32	0.0
33	0.0
34	0.0
35	0.0
36	0.0
37	0.0
38	0.0
39	0.0
40	0.0
41	0.0
42	0.0
43	0.0
44	0.0
45	0.0
46	0.0
47	0.0
48	0.0
49	0.0
50	0.0
51	0.0
52	0.0
53	18384.0
54	97560.0
55	0.0
56	0.0
57	0.0
58	0.0
59	0.0


```
[13]: grouping_function = lambda x: x.split('_')[0] #This function takes in a full
      ↪ column name, like "000_G2010.20.000_03 00 00.00_m3_1", and returns only the
      ↪ floor.
to_draw = df[cols].groupby(grouping_function,axis=1).sum().replace(0,np.NaN).
      ↪div(df['Gross Floor Area'],axis='rows').mean()
plt.figure(figsize=(12,12))
plt.bar(to_draw.keys(), to_draw.values)
plt.xticks(rotation=90)
plt.title('Average Material Intensity of structural concrete used per floor')
plt.ylabel('Average Material Intensity of structural concrete Used (KG/m2)')
plt.xlabel('Floor Code');
```

```
[13]: Text(0.5, 0, 'Floor Code')
```



Now, we will aggregate to Level 3 MasterFormat codes, and display these values for the first three entries.

```
[14]: f = lambda x: name_map[re.split('_\.\\ ',x)[1][0:3]] #This function takes in a  
      ↪ full column name and returns only the Level 3 MasterFormat code.  
      concrete_df = df[cols].groupby(f,axis=1).sum()
```

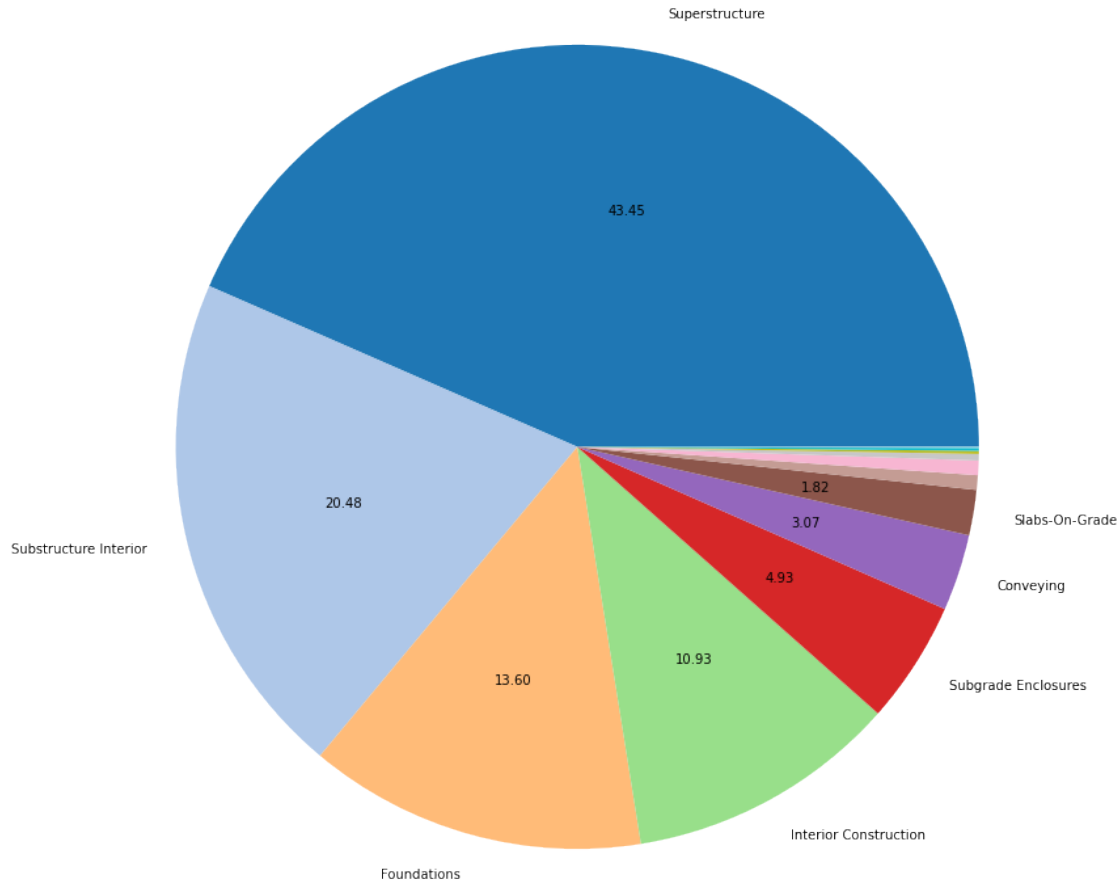
```
[15]: concrete_df.mean().sort_values(ascending=False)
```

```
[15]: Superstructure                2.156826e+06  
      Substructure Interior         1.016670e+06  
      Foundations                  6.750260e+05  
      Interior Construction         5.426285e+05  
      Subgrade Enclosures           2.447624e+05  
      Conveying                     1.525784e+05  
      Slabs-On-Grade                9.043012e+04  
      Exterior Vertical Enclosures  2.899600e+04  
      Exterior Horizontal Enclosures 2.883760e+04  
      Substructure Related Activities 1.309240e+04  
      Special Construction          6.064400e+03  
      Plumbing                      5.850000e+03  
      Site Improvements             1.932400e+03  
      dtype: float64
```

3.1 Pie chart version A: on-pie chart labels for all > 1%

```
[16]: def my_autopct(pct):  
      ↪ return ('%.2f' % pct) if pct > 1 else ''  
      to_plot = concrete_df.mean().sort_values(ascending=False)  
      to_plot.plot.pie(figsize=(12,12),colormap='tab20',autopct=my_autopct,labels=[k,  
      ↪ if v > 30000 else '' for k,v in to_plot.items()])  
      plt.ylabel('')  
      plt.title('Percentage of total steel used in each building element category');  
      # plt.legend(loc='center left',bbox_to_anchor=(-0.20, 0.75));  
      plt.tight_layout();
```

Percentage of total steel used in each building element category



3.2 Pie version B: external legend with slice labels

```
[17]: fig = plt.figure(figsize=(16,12))
gs = gridspec.GridSpec(2, 2, width_ratios=[3, 1])
ax0 = plt.subplot(gs[:,0])

def my_autopct(pct):
    return ('%.2f' % pct) if pct > 1 else ''
to_plot = concrete_df.mean().sort_values(ascending=False)
to_plot.plot.pie(ax=ax0,colormap='tab20',autopct=my_autopct,labeldistance=None)
plt.ylabel('')
plt.legend(loc='center left',bbox_to_anchor=(-0.20, 0.75));
plt.tight_layout();

ax1 = plt.subplot(gs[0,1])
```

```

f = lambda x: \
    additional_categories_map[re.split('_\.\ ',x)[3]] \
    if \
    re.split('_\.\ ',x)[3] != '000' \
    else \
    name_map['.'.join(re.split('_\.\ ',x)[1:3])]

superstructure_df = df[[c for c in cols if 'B10' in c]].groupby(f,axis=1).sum()
to_plot = superstructure_df.mean().sort_values(ascending=False)
def my_autopct(pct):
    return ('%.2f' % ((pct * 0.4335))) if pct > 1 else ''
to_plot.plot.pie(ax=ax1,colormap='Paired',autopct=my_autopct,labeldistance=None)
plt.ylabel('')
plt.legend(loc='center right',bbox_to_anchor=(1, -0.65));
plt.tight_layout();

transFigure = fig.transFigure.inverted()

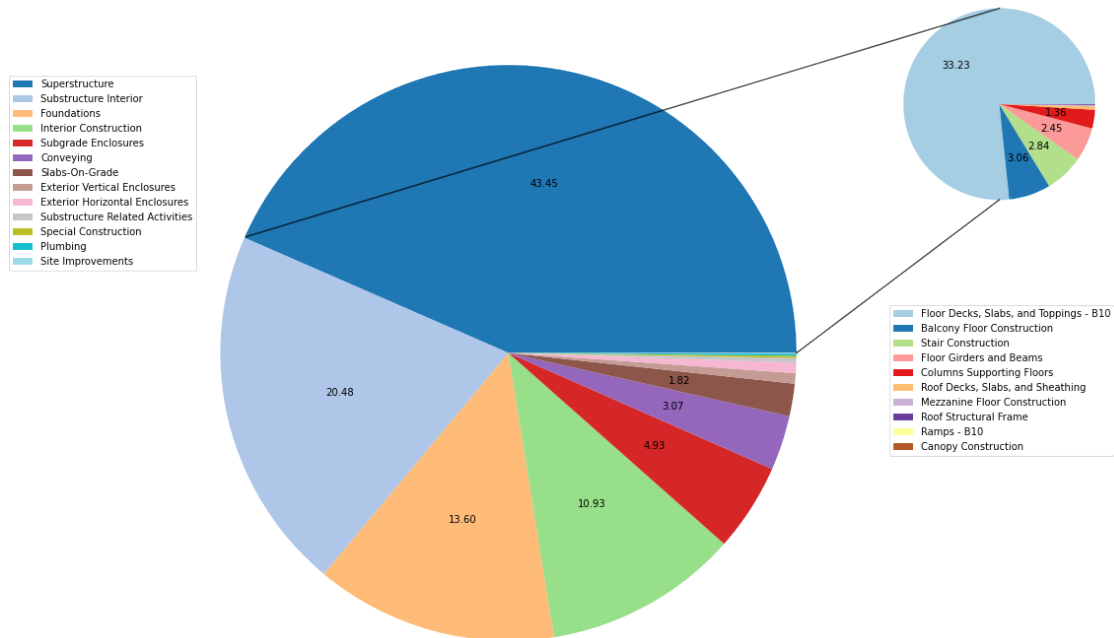
coord1a = transFigure.transform(ax0.transData.transform([1,0]))
coord2a = transFigure.transform(ax1.transData.transform([0,-0.72]))

coord1b = transFigure.transform(ax0.transData.transform([-0.91,0.35]))
coord2b = transFigure.transform(ax1.transData.transform([0,0.72]))

linea = matplotlib.lines.Line2D((coord1a[0],coord2a[0]),(coord1a[1],coord2a[1]),
                                transform=fig.transFigure,c='black',alpha=0.7)
lineb = matplotlib.lines.Line2D((coord1b[0],coord2b[0]),(coord1b[1],coord2b[1]),
                                transform=fig.transFigure,c='black',alpha=0.7)
fig.lines = linea,lineb,

plt.savefig('concrete_breakdown_pie.pdf')

```



We can produce a pie chart for a single building, also.

```
[18]: mf_codes = pd.read_csv('mf_name_conversion.csv')
```

```
[19]: tofind = [
    'Plain Steel Reinforcement Bars',
    'Reinforcement Bars',
    'Structural Steel Framing',
    'Fabric and Grid Reinforcing',
    'Metal Doors',
    'Metal Roof Panel',
    'Metal Stairs',
    'Metal Railings',
    'Steel Decking',
    'Steel Joist Framing',
    'Steel'
] #List of terms we are looking to identify in column names.

tokeep = [
    c for c in mf_codes.Title.values if any(t in c for t in tofind)
] #For each codes' corresponding in MasterFormat

steel_codes = mf_codes[mf_codes.Title.isin(tokeep)]
```

```
[20]: columns_to_keep = []
      for column in df.columns:
          if 'kg' in column:
              code = re.split('_',column)[2]
              for k,c in steel_codes.values:
                  if c in code:
                      columns_to_keep.append(column)
```

```
[21]: f = lambda x: mf_codes[mf_codes.Code == str.replace(re.split('_',x)[2],'00','').
↳strip('.').values[0][0]
      steel_df = df[columns_to_keep].groupby(f,axis=1).sum()
```

```
[22]: (steel_df>0).sum(axis=1).sort_values()
```

```
[22]: 15      1
      42      1
      22      1
      36      1
      7       1
      34      1
      31      1
      35      1
      55      2
      58      2
      40      2
      41      2
      1       2
      43      2
      24      2
      23      2
      21      2
      20      2
      54      2
      44      2
      17      2
      16      2
      30      2
      14      2
      45      2
      12      2
      11      2
      32      2
      9       2
      33      2
      3       2
      18      2
      0       3
```

```

52     3
53     3
56     3
46     3
39     3
29     3
37     3
28     3
27     3
26     3
25     3
13     3
10     3
2      3
38     3
5      3
6      3
8      3
57     4
4      4
49     4
50     4
48     4
47     4
19     4
51     4
59     4
dtype: int64

```

```

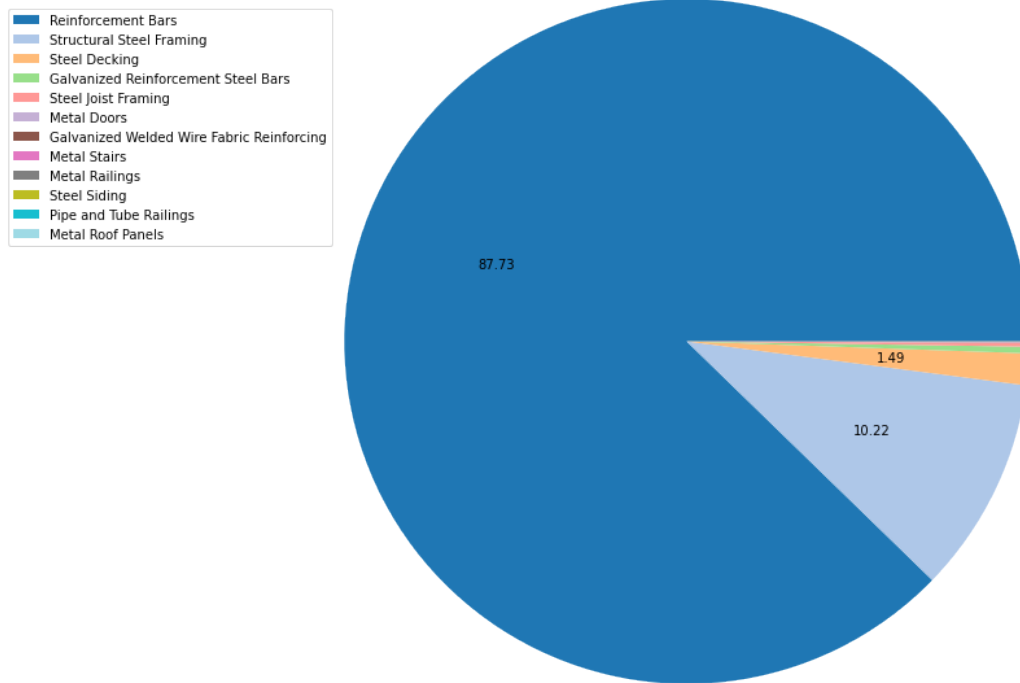
[23]: def my_autopct(pct):
        return ('%.2f' % (pct)) if pct > 1 else ''
to_plot = steel_df.sum().sort_values(ascending=False)
to_plot.plot.
    ↳ pie(figsize=(12,12), colormap='tab20', autopct=my_autopct, labeldistance=None)
plt.legend(loc='center left', bbox_to_anchor=(-0.30, 0.75));

plt.ylabel('')
plt.title(f'Types of steel use in all buildings in terms of MasterFormat_
    ↳ categories');
plt.tight_layout();

plt.savefig('steel_composition_pie.pdf')

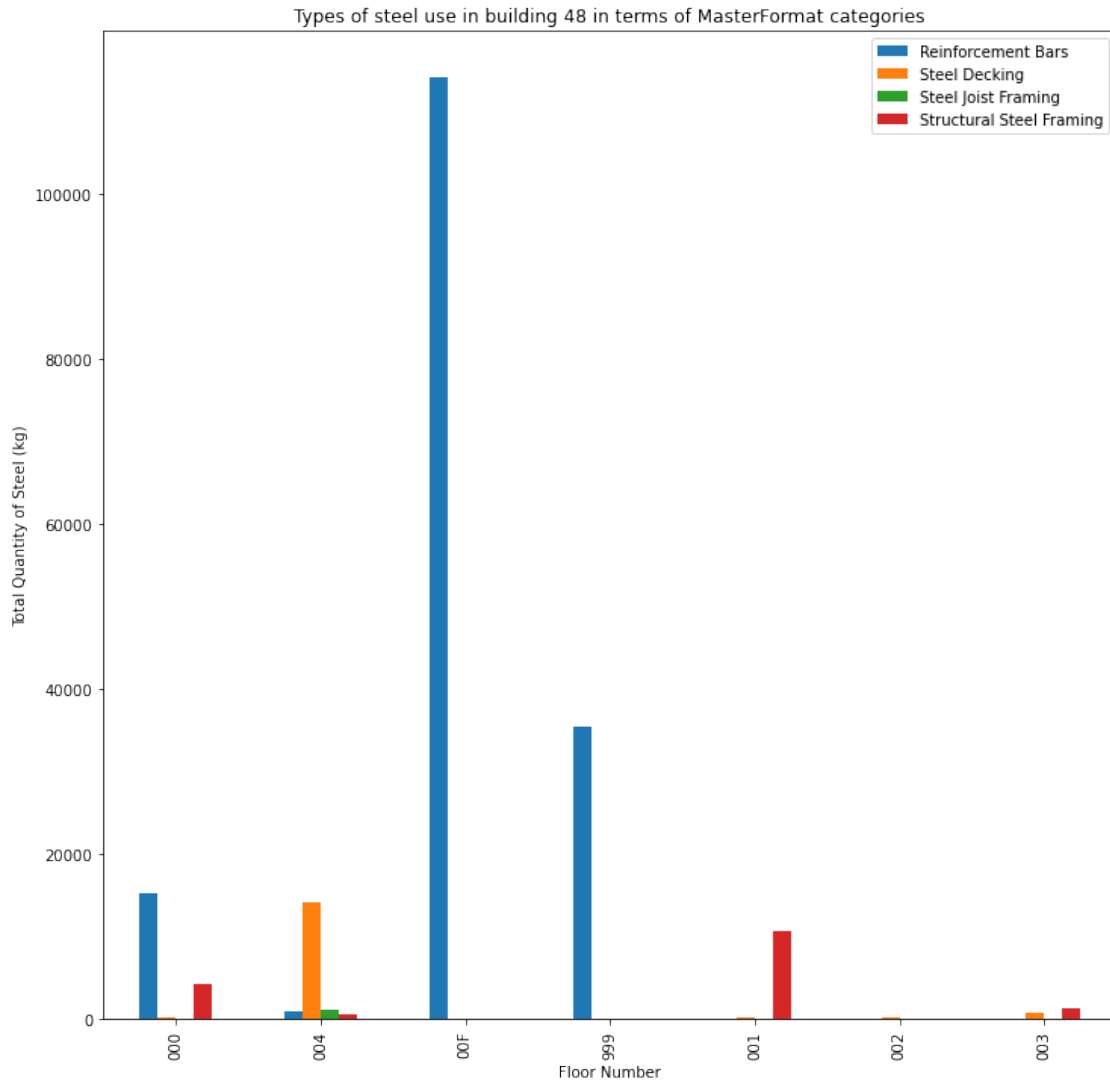
```

Types of steel use in all buildings in terms of MasterFormat categories



```
[24]: f = lambda x: mf_codes[mf_codes.Code == str.replace(re.split('_',x)[2],'00','').
    ↳strip('.')].values[0][0] + '/' + x.split('_')[0]
tdf = df[colums_to_keep].groupby(f,axis=1).sum().iloc[47,:]
tdf = tdf[tdf>0]
```

```
[25]: from collections import defaultdict
todf = defaultdict(dict)
for (a,b),c in zip(tdf.keys().str.split('/'),tdf.values):
    todf[a][b] = c
toplot = pd.DataFrame(todf)
toplot.plot.bar(figsize=(12,12));
plt.xlabel('Floor Number')
plt.ylabel('Total Quantity of Steel (kg)')
plt.title('Types of steel use in building 48 in terms of MasterFormat_
    ↳categories')
plt.savefig('bar_steel_onebuildingtype_byfloor.pdf')
```

We can also calculate the average for each Level 3 MasterFormat code by year of construction:

```
[26]: concrete_df = pd.concat([df[headings[1:]], df[cols].groupby(f,axis=1).
    ↳sum()],axis=1)
    concrete_df.groupby('Construction Date').mean()
```

```
[26]:
```

Construction Date	Gross Floor Area	Structural Concrete/000 \
1913	161.080000	0.000000e+00
1917	199.930000	0.000000e+00
1969	373.605000	0.000000e+00
1988	21934.000000	0.000000e+00
2007	73600.000000	0.000000e+00
2009	73083.000000	0.000000e+00

2011	11282.500000	0.000000e+00
2016	30345.000000	3.595656e+06
2017	39392.013333	4.084352e+06
2018	43560.635000	5.893680e+06
2019	83.100000	0.000000e+00
2020	418.528571	4.914431e+03
2021	445.404444	0.000000e+00

	Structural Concrete/001	Structural Concrete/002 \
Construction Date		
1913	1944.380000	0.0
1917	4972.300000	0.0
1969	7262.220500	0.0
1988	0.000000	0.0
2007	0.000000	0.0
2009	0.000000	0.0
2011	0.000000	0.0
2016	0.000000	2680512.0
2017	0.000000	989280.0
2018	0.000000	1511892.0
2019	0.000000	0.0
2020	4923.690714	0.0
2021	11399.123858	0.0

	Structural Concrete/003	Structural Concrete/004 \
Construction Date		
1913	0.0	0.0
1917	0.0	0.0
1969	0.0	0.0
1988	0.0	0.0
2007	0.0	0.0
2009	0.0	0.0
2011	0.0	0.0
2016	1686228.0	1057032.0
2017	1232336.0	778480.0
2018	1347936.0	1323132.0
2019	0.0	0.0
2020	0.0	0.0
2021	0.0	0.0

	Structural Concrete/005	Structural Concrete/006 \
Construction Date		
1913	0.0	0.0
1917	0.0	0.0
1969	0.0	0.0
1988	0.0	0.0
2007	0.0	0.0

2009	0.0	0.0
2011	0.0	0.0
2016	1056780.0	1129680.0
2017	683496.0	679376.0
2018	2164812.0	969060.0
2019	0.0	0.0
2020	0.0	0.0
2021	0.0	0.0

	Structural Concrete/007	Structural Concrete/008	...	\
Construction Date			...	
1913	0.0	0.0	...	
1917	0.0	0.0	...	
1969	0.0	0.0	...	
1988	0.0	0.0	...	
2007	0.0	0.0	...	
2009	0.0	0.0	...	
2011	0.0	0.0	...	
2016	1809852.0	857976.0	...	
2017	632520.0	651080.0	...	
2018	752208.0	734688.0	...	
2019	0.0	0.0	...	
2020	0.0	0.0	...	
2021	0.0	0.0	...	

	Structural Concrete/0B1	Structural Concrete/0P1	\
Construction Date			
1913	0.000000	0.0	
1917	0.000000	0.0	
1969	0.000000	0.0	
1988	0.000000	0.0	
2007	0.000000	0.0	
2009	0.000000	0.0	
2011	0.000000	0.0	
2016	0.000000	2206668.0	
2017	0.000000	3402456.0	
2018	0.000000	3713916.0	
2019	5412.758585	0.0	
2020	0.000000	0.0	
2021	545.221944	0.0	

	Structural Concrete/0P2	Structural Concrete/0P3	\
Construction Date			
1913	0.0	0.0	
1917	0.0	0.0	
1969	0.0	0.0	
1988	0.0	0.0	

2007	0.0	0.0
2009	0.0	0.0
2011	0.0	0.0
2016	1715028.0	1596444.0
2017	2513320.0	2469984.0
2018	2637060.0	2756916.0
2019	0.0	0.0
2020	0.0	0.0
2021	0.0	0.0

	Structural Concrete/OP4	Structural Concrete/OP5 \
Construction Date		
1913	0.0	0.0
1917	0.0	0.0
1969	0.0	0.0
1988	0.0	0.0
2007	0.0	0.0
2009	0.0	0.0
2011	0.0	0.0
2016	0.0	0.0
2017	1895672.0	508328.0
2018	4093284.0	0.0
2019	0.0	0.0
2020	0.0	0.0
2021	0.0	0.0

	Structural Concrete/999	Structural Concrete/B01 \
Construction Date		
1913	0.0	64035.190000
1917	0.0	114018.460000
1969	0.0	132278.015000
1988	0.0	0.000000
2007	0.0	0.000000
2009	0.0	0.000000
2011	0.0	0.000000
2016	155076.0	0.000000
2017	7736.0	0.000000
2018	558516.0	0.000000
2019	0.0	0.000000
2020	0.0	140301.728571
2021	0.0	161741.376806

	Structural Concrete/B03	Structural Concrete/M00
Construction Date		
1913	0.000000	0.0
1917	0.000000	0.0
1969	0.000000	0.0

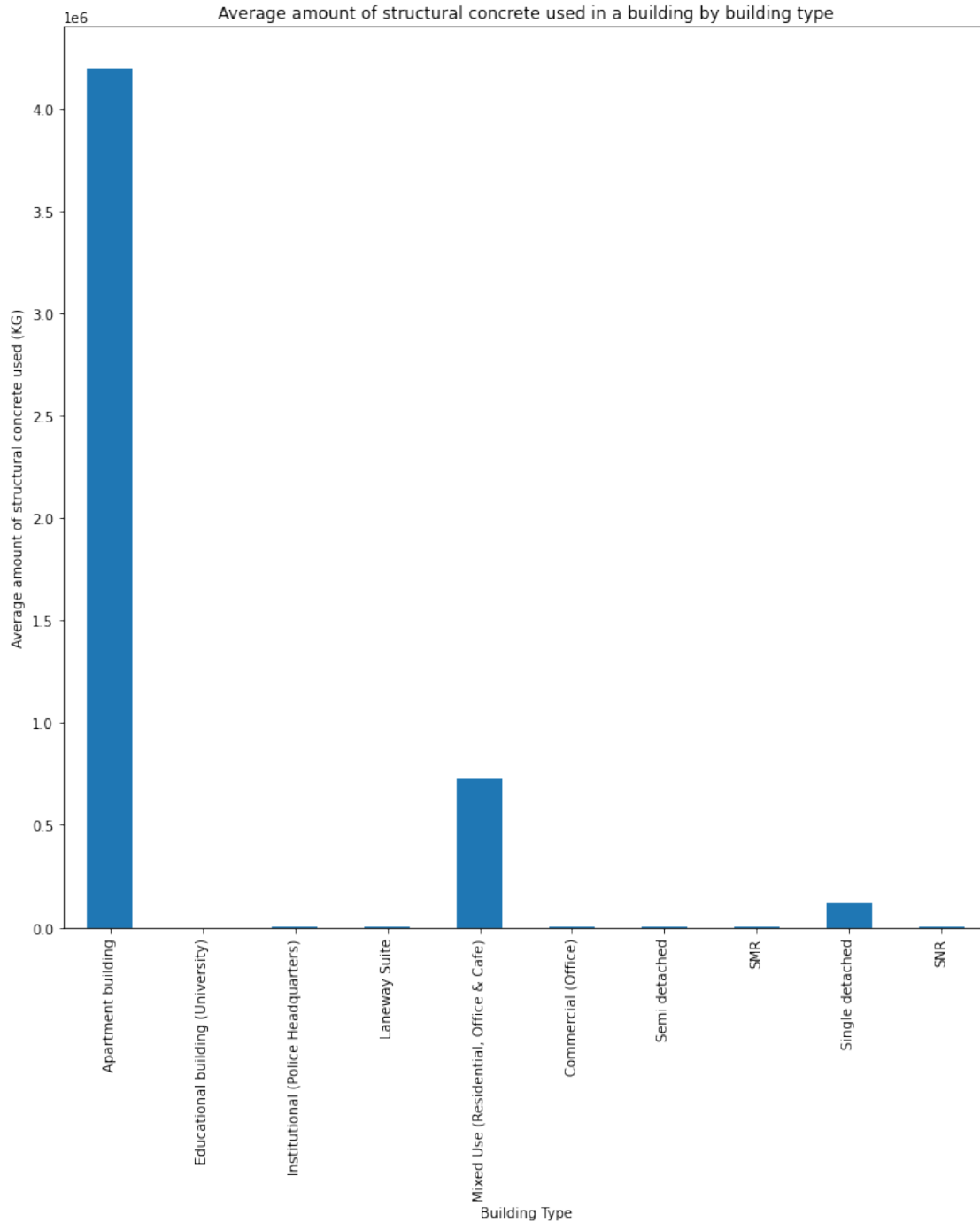
1988	0.000000	0.0
2007	0.000000	0.0
2009	0.000000	0.0
2011	0.000000	0.0
2016	0.000000	82056.0
2017	0.000000	0.0
2018	0.000000	597624.0
2019	0.000000	0.0
2020	988.177143	0.0
2021	0.000000	0.0

[13 rows x 58 columns]

We can get the average amount of steel in KG used per building type:

```
[27]: concrete_df.groupby('Building Type').sum().mean(axis=1).
      ↪ rename(index=building_name_map).plot(kind='bar',figsize=(12,12))
plt.ylabel('Average amount of structural concrete used (KG)')
plt.title('Average amount of structural concrete used in a building by building_
      ↪ type');
```

```
[27]: Text(0.5, 1.0, 'Average amount of structural concrete used in a building by
building type')
```



4 3. Uncertainty by Building Type

In this section, we look at the uncertainty code associated with each material takeoff. We collect these by building type and then report the number of each value per type of building.

```
[28]: uncertainty_level = {}
      for k,v in df.iterrows():
          #Initialise empty lists for each building type as they occur
          if v['Building Type'] not in uncertainty_level.keys():
              uncertainty_level[v['Building Type']] = []
          #Append the uncertainty value for each column that is non-NaN
          for key in v[~v.isna()].keys()[7:]:
              uncertainty_level[v['Building Type']].append(key.split('_')[-1])

[29]: from collections import Counter

[30]: for k,v in uncertainty_level.items():
      uncertainty_level[k] = Counter(v) #Construct a Counter object per building_
      ↪type

[31]: uncertainty_level

[31]: {'SND': Counter({'1': 1662, '2': 641, '4': 293}),
      'OFF': Counter({'1': 494, '3': 307}),
      'APB': Counter({'1': 1171, '2': 1, '3': 971}),
      'SMR': Counter({'1': 21, '2': 27, '4': 8}),
      'SNR': Counter({'1': 58, '2': 70, '4': 56}),
      'SMD': Counter({'1': 170, '2': 34, '4': 19}),
      'EDU': Counter({'1': 93, '3': 24, '2': 6}),
      'INS': Counter({'1': 90, '3': 77, '2': 1}),
      'MIX': Counter({'1': 363, '3': 276}),
      'LNW': Counter({'2': 46, '1': 142, '4': 19})}
```

Next, we aggregate columns by the purpose of the material and uncertainty combined, and report the average by building type.

```
[32]: f = lambda x: name_map[re.split('_\.\ ',x)[1][0]] + '/' + x.split('_')[-1].
      ↪split('.')[0] #From a full code, return only the use code and uncertainty_
      ↪code.
      by_function_df = pd.concat([df[headings[1:]],df[cols].groupby(f,axis=1).
      ↪sum()],axis=1)

[33]: by_function_df.groupby('Building Type').mean().rename(index=building_name_map).
      ↪drop(['Construction Date'],axis=1).round(2)
```

```
[33]:
```

	Gross Floor Area	Interiors/1 \
Building Type		
Apartment building	45505.41	5330644.8
Educational building (University)	7901.00	0.0
Institutional (Police Headquarters)	21934.00	0.0
Laneway Suite	150.01	0.0
Mixed Use (Residential, Office & Cafe)	33975.25	5893176.0

Commercial (Office)	52643.67	0.0
Semi detached	248.84	0.0
SMR	199.93	0.0
Single detached	478.40	305.6
SNR	302.76	0.0

	Services/1	Shell/1	Shell/2 \
Building Type			
Apartment building	1525512.0	21949118.40	0.00
Educational building (University)	0.0	0.00	0.00
Institutional (Police Headquarters)	0.0	0.00	0.00
Laneway Suite	0.0	0.00	0.00
Mixed Use (Residential, Office & Cafe)	1878144.0	23063160.00	0.00
Commercial (Office)	0.0	0.00	0.00
Semi detached	0.0	1864.27	0.00
SMR	0.0	0.00	0.00
Single detached	0.0	1547.01	13.19
SNR	0.0	2504.95	0.00

	Sitework/1 \
Building Type	
Apartment building	23188.8
Educational building (University)	0.0
Institutional (Police Headquarters)	0.0
Laneway Suite	0.0
Mixed Use (Residential, Office & Cafe)	0.0
Commercial (Office)	0.0
Semi detached	0.0
SMR	0.0
Single detached	0.0
SNR	0.0

	Special Construction And Demolition/1 \
Building Type	
Apartment building	60316.8
Educational building (University)	0.0
Institutional (Police Headquarters)	0.0
Laneway Suite	0.0
Mixed Use (Residential, Office & Cafe)	62280.0
Commercial (Office)	0.0
Semi detached	0.0
SMR	0.0
Single detached	0.0
SNR	0.0

	Substructure/1	Substructure/2 \
Building Type		

Apartment building	20539176.00	0.00
Educational building (University)	0.00	0.00
Institutional (Police Headquarters)	0.00	0.00
Laneway Suite	58217.18	44.81
Mixed Use (Residential, Office & Cafe)	11822352.00	0.00
Commercial (Office)	0.00	0.00
Semi detached	97640.84	0.00
SMR	110089.90	8900.86
Single detached	180887.68	5347.87
SNR	93180.79	19334.28

Substructure/4

Building Type	
Apartment building	0.00
Educational building (University)	0.00
Institutional (Police Headquarters)	0.00
Laneway Suite	1850.97
Mixed Use (Residential, Office & Cafe)	0.00
Commercial (Office)	0.00
Semi detached	0.00
SMR	0.00
Single detached	0.00
SNR	0.00

Next, we report the total amount of material falling under each uncertainty code by year of construction.

```
[34]: f = lambda x: x.split('_')[-1].split('.')[0] #Select only the uncertainty code.
print('Average amount of material used per building, by year and uncertainty_
      ↳code (kg)')
display(pd.concat([df['Construction Date'],df[[c for c in df.columns if 'kg' in_
      ↳c]].groupby(f,axis=1).sum()]),axis=1).groupby('Construction Date').mean().
      ↳round(0))
```

Average amount of material used per building, by year and uncertainty code (kg)

	1	2	3	4
Construction Date				
1913	90840.0	8231.0	0.0	7754.0
1917	137657.0	24862.0	0.0	20786.0
1969	124421.0	90335.0	0.0	32662.0
1988	27569611.0	2134.0	832782.0	0.0
2007	89001799.0	0.0	2715390.0	0.0
2009	96695376.0	0.0	2989172.0	0.0
2011	12969995.0	436848.0	393956.0	0.0
2016	37771429.0	837792.0	988332.0	0.0
2017	40843021.0	0.0	1112868.0	0.0
2018	54609667.0	0.0	1537632.0	0.0

2019	53403.0	2376.0	0.0	0.0
2020	222826.0	26710.0	0.0	28228.0
2021	243613.0	27693.0	0.0	41449.0

5 4. Material Intensity

We can easily calculate material intensity by dividing takeoffs which are measured in kilograms by the Gross Floor Area:

```
[35]: kilogram_columns = [d for d in df.columns if 'kg' in d]
df_mi = df[kilogram_columns].div(df['Gross Floor Area'],axis=0)
```

```
[36]: kilogram_columns = [d for d in df.columns if 'kg' in d]
df_mi = df[kilogram_columns].div(df['Gross Floor Area'],axis=0)
f = lambda x: name_map[re.split('_\\.\\ ',x)[1][0:3]]
pd.concat([df[headings[1:]],df_mi[kilogram_columns].groupby(f,axis=1).
    ↪sum(),axis=1)[df['Building Type'] == 'SND']
```

```
[36]: Country City Quality / Stage of Data Construction Date Building Type \
0 CA TOR 00IFC 2021 SND
1 CA TOR 00IFC 2021 SND
2 CA TOR 00IFC 2021 SND
3 CA TOR 00IFC 2021 SND
6 CA TOR 00IFC 2021 SND
7 CA TOR 00IFC 2021 SND
8 CA TOR 00IFC 2021 SND
9 CA TOR 00IFC 2021 SND
12 CA TOR 00IFC 2021 SND
13 CA TOR 00IFC 2021 SND
14 CA TOR 00IFC 2021 SND
15 CA TOR 00IFC 2021 SND
18 CA TOR 00IFC 2021 SND
19 CA TOR 00IFC 2021 SND
20 CA TOR 00IFC 2020 SND
21 CA TOR 00IFC 2021 SND
22 CA TOR 00IFC 2021 SND
24 CA TOR 00IFC 2021 SND
25 CA TOR 00IFC 2021 SND
27 CA TOR 00IFC 2021 SND
28 CA TOR 00IFC 2021 SND
30 CA TOR 00IFC 2021 SND
31 CA TOR 00IFC 2021 SND
32 CA TOR 00IFC 2020 SND
34 CA TOR 00IFC 2021 SND
35 CA TOR 00IFC 2021 SND
36 CA TOR 00IFC 2021 SND
37 CA TOR 00IFC 2020 SND
```

38	CA	TOR	00IFC	2021	SND
40	CA	TOR	00IFC	2021	SND
42	CA	TOR	00IFC	2021	SND
43	CA	TOR	00IFC	2021	SND
44	CA	TOR	00IFC	2021	SND
45	CA	TOR	00IFC	2021	SND
46	CA	TOR	00IFC	2021	SND
48	CA	TOR	00IFC	2020	SND
49	CA	TOR	00IFC	2021	SND

	Gross Floor Area	Conveying	Exterior Horizontal Enclosures \
0	521.18	0.0	11.137992
1	389.24	0.0	5.461939
2	411.64	0.0	3.786074
3	269.56	0.0	6.503479
6	445.99	0.0	11.933511
7	438.45	0.0	12.707195
8	714.07	0.0	12.865930
9	343.24	0.0	4.300619
12	226.89	0.0	12.424245
13	611.73	0.0	5.140200
14	343.44	0.0	6.494467
15	613.38	0.0	13.090524
18	178.38	0.0	9.782438
19	323.80	0.0	9.824569
20	837.56	0.0	13.521848
21	587.86	0.0	6.949783
22	568.21	0.0	12.754287
24	294.84	0.0	3.650542
25	496.77	0.0	5.352985
27	643.30	0.0	11.769043
28	701.61	0.0	11.799093
30	378.70	0.0	5.522739
31	324.16	0.0	5.361174
32	533.53	0.0	8.494907
34	423.03	0.0	11.102019
35	328.16	0.0	10.234937
36	421.59	0.0	12.223172
37	628.59	0.0	10.408758
38	464.51	0.0	4.118745
40	346.14	0.0	11.787081
42	891.97	0.0	10.710312
43	525.61	0.0	18.918490
44	502.87	0.0	6.014586
45	379.18	0.0	6.169302
46	549.65	0.0	11.310711
48	393.82	0.0	16.116861

49 648.14 0.0 9.684756

	Exterior Vertical	Enclosures	Foundations	...	Interior Finishes	\
0		136.939623	335.649367	...	8.309413	
1		69.018253	281.318698	...	6.490936	
2		101.450370	464.462195	...	4.574905	
3		188.215196	255.359136	...	8.510443	
6		61.325975	295.116668	...	6.391063	
7		130.552921	269.468463	...	6.584780	
8		104.310510	276.917123	...	6.563894	
9		210.632241	283.893850	...	8.940907	
12		186.668275	261.874926	...	6.134611	
13		102.332008	343.714248	...	7.638991	
14		147.104280	424.099610	...	7.860800	
15		156.986570	298.537712	...	8.068881	
18		112.523711	371.149916	...	9.551856	
19		186.570501	148.769711	...	9.483653	
20		91.689386	317.583491	...	7.152371	
21		94.557055	428.185321	...	6.754074	
22		83.789887	255.012975	...	7.860492	
24		127.856507	261.274626	...	4.807604	
25		89.883144	251.725837	...	5.921358	
27		83.949693	156.365248	...	8.492430	
28		53.418023	266.164355	...	7.952623	
30		164.214896	403.602589	...	7.221059	
31		190.512918	377.853541	...	6.597902	
32		68.518430	309.062696	...	6.648595	
34		154.072547	243.607664	...	4.717349	
35		184.202156	388.744353	...	5.648226	
36		158.716507	424.443503	...	5.625641	
37		136.076590	369.744859	...	5.699975	
38		151.068033	412.845205	...	7.621364	
40		146.479339	287.564257	...	7.916204	
42		213.677214	245.205806	...	7.577250	
43		109.529933	498.010299	...	7.954358	
44		91.481074	278.679758	...	4.564488	
45		172.418003	391.303861	...	6.339432	
46		127.866168	266.468237	...	6.701647	
48		140.069509	188.980245	...	10.629628	
49		131.118584	347.187490	...	5.089382	

	Plumbing	Site Improvements	Slabs-On-Grade	Special Construction	\
0	0.0	0.0	273.972401	0.0	
1	0.0	0.0	192.874465	0.0	
2	0.0	0.0	170.733356	0.0	
3	0.0	0.0	124.186526	0.0	
6	0.0	0.0	153.061618	0.0	

7	0.0	0.0	211.910108	0.0
8	0.0	0.0	266.709576	0.0
9	0.0	0.0	138.510228	0.0
12	0.0	0.0	129.263543	0.0
13	0.0	0.0	165.513154	0.0
14	0.0	0.0	129.532248	0.0
15	0.0	0.0	166.414337	0.0
18	0.0	0.0	223.398638	0.0
19	0.0	0.0	158.178114	0.0
20	0.0	0.0	143.282268	0.0
21	0.0	0.0	237.918968	0.0
22	0.0	0.0	199.364347	0.0
24	0.0	0.0	131.174185	0.0
25	0.0	0.0	242.284758	0.0
27	0.0	0.0	152.407914	0.0
28	0.0	0.0	169.419640	0.0
30	0.0	0.0	179.868896	0.0
31	0.0	0.0	132.696247	0.0
32	0.0	0.0	135.390288	0.0
34	0.0	0.0	147.458950	0.0
35	0.0	0.0	128.887840	0.0
36	0.0	0.0	147.225241	0.0
37	0.0	0.0	186.334547	0.0
38	0.0	0.0	145.273403	0.0
40	0.0	0.0	139.821081	0.0
42	0.0	0.0	138.994603	0.0
43	0.0	0.0	139.646277	0.0
44	0.0	0.0	182.059329	0.0
45	0.0	0.0	158.446049	0.0
46	0.0	0.0	154.805714	0.0
48	0.0	0.0	198.860705	0.0
49	0.0	0.0	199.209464	0.0

	Subgrade Enclosures	Substructure Interior \
0	9.652903	0.000000
1	6.851955	0.000000
2	11.298572	0.000000
3	4.351465	0.000000
6	9.478642	0.054452
7	4.218921	0.000000
8	8.902623	0.000000
9	9.601245	0.000000
12	3.818403	0.935612
13	7.722754	0.000000
14	9.135529	0.000000
15	4.868508	0.467438
18	0.000000	0.000000

19	4.617006	0.000000
20	7.131170	0.000000
21	7.959752	0.000000
22	6.339651	0.000000
24	7.469048	0.000000
25	9.448689	0.078017
27	0.000000	0.096759
28	11.919460	0.000000
30	7.509119	0.330172
31	5.073992	0.000000
32	8.867868	0.000000
34	0.000000	0.000000
35	4.762839	0.000000
36	9.538939	0.000000
37	6.039206	1.461249
38	9.071017	0.000000
40	7.568785	0.394416
42	4.540919	0.371810
43	6.720435	0.000000
44	6.092739	0.000000
45	9.489156	0.195110
46	6.042229	0.499896
48	6.057127	1.647329
49	7.221222	1.208104

	Substructure Related Activities	Superstructure	Water And Gas Mitigation
0	0.0	30.228003	0.0
1	0.0	26.271523	0.0
2	0.0	23.756286	0.0
3	0.0	30.517748	0.0
6	0.0	39.906513	0.0
7	0.0	39.907474	0.0
8	0.0	38.291591	0.0
9	0.0	35.370538	0.0
12	0.0	35.355314	0.0
13	0.0	33.388004	0.0
14	0.0	39.370016	0.0
15	0.0	40.958564	0.0
18	0.0	63.006044	0.0
19	0.0	36.597047	0.0
20	0.0	28.734226	0.0
21	0.0	37.457583	0.0
22	0.0	36.265538	0.0
24	0.0	30.389475	0.0
25	0.0	43.728928	0.0
27	0.0	35.393414	0.0
28	0.0	39.408113	0.0

30	0.0	82.392236	0.0
31	0.0	46.380703	0.0
32	0.0	25.469871	0.0
34	0.0	35.666107	0.0
35	0.0	49.404111	0.0
36	0.0	34.035382	0.0
37	0.0	47.065025	0.0
38	0.0	37.921434	0.0
40	0.0	27.740220	0.0
42	0.0	29.045531	0.0
43	0.0	33.265489	0.0
44	0.0	37.265275	0.0
45	0.0	46.860447	0.0
46	0.0	31.152827	0.0
48	0.0	49.899420	0.0
49	0.0	38.021046	0.0

[37 rows x 21 columns]

```
[37]: master_format_convert = {v:k for k,v in {
    'Concrete':'03',
    'Masonry':'04',
    'Metals':'05',
    'WoodPlasticsAndComposites':'06',
    'ThermalAndMoistureProtection':'07',
    'Finishes':'09',
    'Openings':'08',
    'Earthwork':'31',
    'ExteriorImprovements':'32'
}.items() }
```

```
[38]: f = lambda x: master_format_convert[re.split('[_\\.\\ ]',x)[4]]
toplot = pd.concat([df[headings[1:]],df_mi[kilogram_columns].groupby(f,axis=1).
    ↪sum()),axis=1).sort_values(['Building Type'])
```

```
[39]: types_to_keep = ['APB','SND','SNR','SMR','SMD','ADU','SEC','ROW','LNW']
toplot = toplot[toplot['Building Type'].isin(types_to_keep)]
building_type_map = {
    'APB':'Mid to high-rise buildings',
    'SND':'Newly Constructed Single family dwellings',
    'SNR':'Renovated Single family dwellings',
    'SMD':'Newly Constructed Single family dwellings',
    'SMR':'Renovated Single family dwellings',
    'ADU':'Newly Constructed Single family dwellings',
    'SEC':'Newly Constructed Single family dwellings',
    'ROW':'Newly Constructed Single family dwellings',
    'LNW':'Laneway Houses'
}
```

```

}

topplot['Building Type'] = topplot['Building Type'].replace(building_type_map)
topplot = topplot.sort_values('Building Type')

```

```

[40]: fig, ax = plt.subplots(figsize=(10,7))

cols = topplot.columns[6:]
margin_bottom = np.zeros(len(topplot))

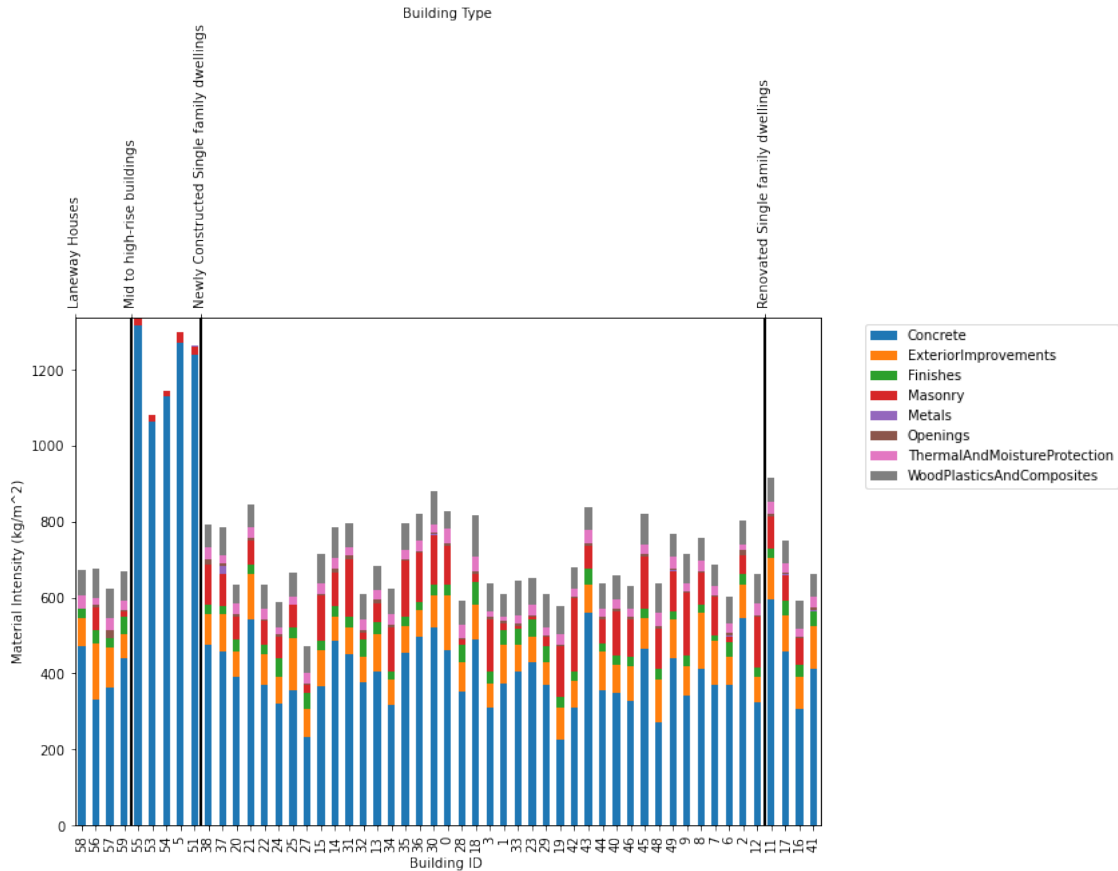
cmap = plt.get_cmap('tab10')

for num, col in enumerate(cols):
    values = topplot[col].values

    topplot[col].plot.bar(x='Year',y='Value', ax=ax, stacked=True,
                        bottom = margin_bottom, color=cmap(num),
                        label=col)
    margin_bottom += values
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.ylabel('Material Intensity (kg/m^2)')
plt.xlabel('Building ID ')
ax2 = ax.twinx()
ax2.set_xlim(0, len(topplot))
ax2.set_xticks([k for k,v in enumerate(topplot['Building Type'].values) if v !=
    topplot['Building Type'].values[k-1] or k==0])
for tick in ax2.get_xticklabels():
    tick.set_rotation(90)
ax2.set_xticklabels([v for k,v in enumerate(topplot['Building Type'].values) if
    v != topplot['Building Type'].values[k-1] or k==0])
ax2.set_xlabel("Building Type")
plt.grid(color='black',linewidth=2)

plt.show()

```

```
[41]: df_mi = df[kilogram_columns].div(df['Gross Floor Area'],axis=0)
```

```
[42]: df_mi = df[kilogram_columns].div(df['Gross Floor Area'],axis=0)
df_mi = df_mi.div(df_mi.sum(axis=1),axis=0) * 100
f = lambda x: name_map[re.split('_\.\.',x)[1][0]]
topplot = pd.concat([df[headings[1:]],df_mi[kilogram_columns].groupby(f,axis=1).
    ↳sum()),axis=1).sort_values('Building Type')
topplot = topplot[topplot['Building Type'].isin(types_to_keep)]
topplot['Building Type'] = topplot['Building Type'].replace(building_type_map)
topplot = topplot.sort_values('Building Type')
fig, ax = plt.subplots(figsize=(10,7))

cols = topplot.columns[6:]
margin_bottom = np.zeros(len(topplot))

cmap = plt.get_cmap('tab10')

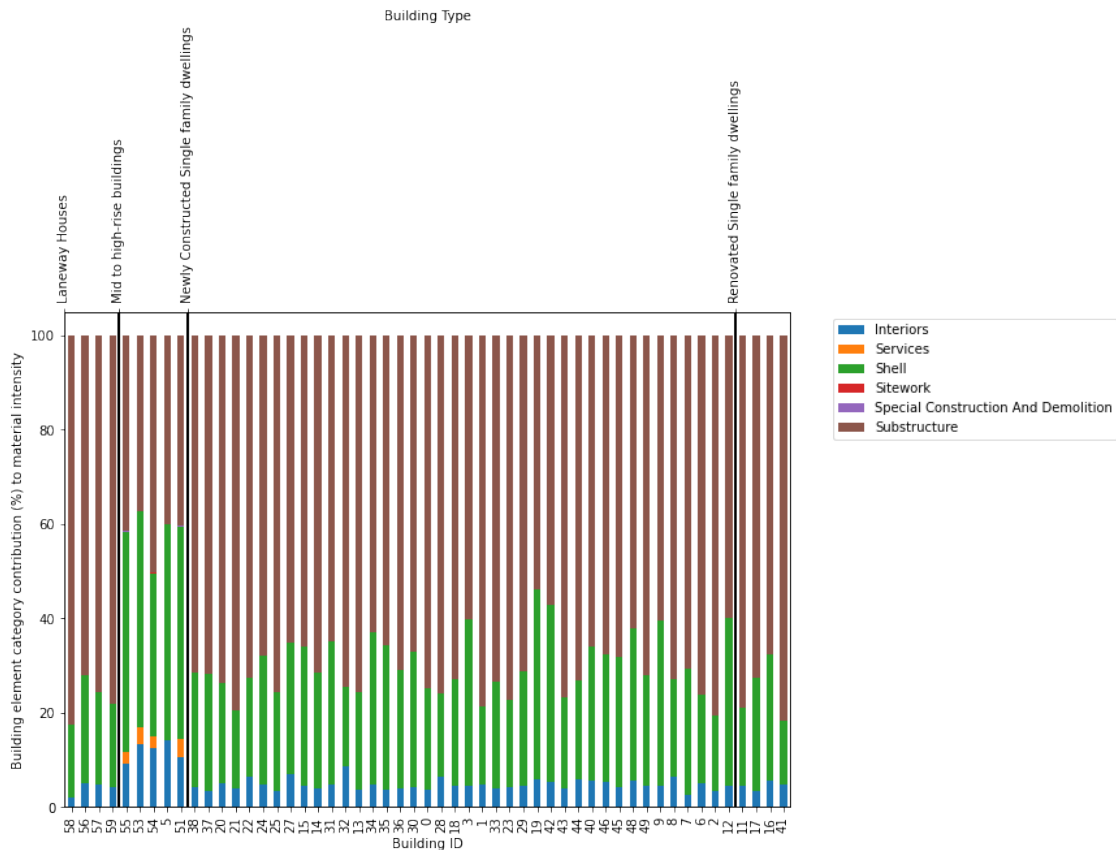
for num, col in enumerate(cols):
    values = topplot[col].values
```

```

topplot[col].plot.bar(x='Year',y='Value', ax=ax, stacked=True,
                     bottom = margin_bottom, color=cmap(num),
→label=col)
    margin_bottom += values
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xlabel('Building ID')
plt.ylabel('Building element category contribution (%) to material intensity')

ax2 = ax.twinx()
ax2.set_xlim(0, len(topplot))
ax2.set_xticks([k for k,v in enumerate(topplot['Building Type'].values) if v !=
→topplot['Building Type'].values[k-1] or k==0])
for tick in ax2.get_xticklabels():
    tick.set_rotation(90)
ax2.set_xticklabels([v for k,v in enumerate(topplot['Building Type'].values) if
→v != topplot['Building Type'].values[k-1] or k==0])
ax2.set_xlabel("Building Type")
plt.grid(color='black',linewidth=2)
plt.show()

```



```
[43]: f = lambda x: name_map[re.split('_\.\\ ',x)[1][0]] + '/' + re.split('_\.\\ 
      ↪',x)[-1]
      toplot = df_mi[kilogram_columns].groupby(f,axis=1).sum()
```

```
[44]: df_mi = df[kilogram_columns].div(df['Gross Floor Area'],axis=0)
      df_mi = df_mi.div(df_mi.sum(axis=1),axis=0)
      f = lambda x: name_map[re.split('_\.\\ ',x)[1][0]] + '/' + re.split('_\.\\ 
      ↪',x)[-1]
      toplot = df_mi[kilogram_columns].groupby(f,axis=1).sum()
      for i in range(1,5):
          toplot[f'Total/{i}'] = 0
      for k,v in toplot.iteritems():
          toplot[f'Total/{k.split("/") [1]}'] += v
      toplot_out = deepcopy(toplot)
      for k,v in toplot.iteritems():
          toplot_out[k] = (v/toplot[[c for c in toplot.columns if k.split('/')[0] in_
      ↪c]].sum(axis=1)) * int(k.split('/')[1])
      f = lambda x: x.split('/')[0]
      toplot_out = pd.concat([df['Building Type'],toplot_out.groupby(f,axis=1).
      ↪sum()],axis=1).sort_values('Building Type')
      toplot_out = toplot_out.reset_index()
      toplot_out['index'] += 1
      toplot_out['index'] = toplot_out['index'].astype('str')
```

```
[45]: toplot_out = toplot_out[toplot_out['Building Type'].isin(types_to_keep)]
      toplot_out['Building Type'] = toplot_out['Building Type'].
      ↪replace(building_type_map)
      toplot_out = toplot_out.sort_values('Building Type')
```

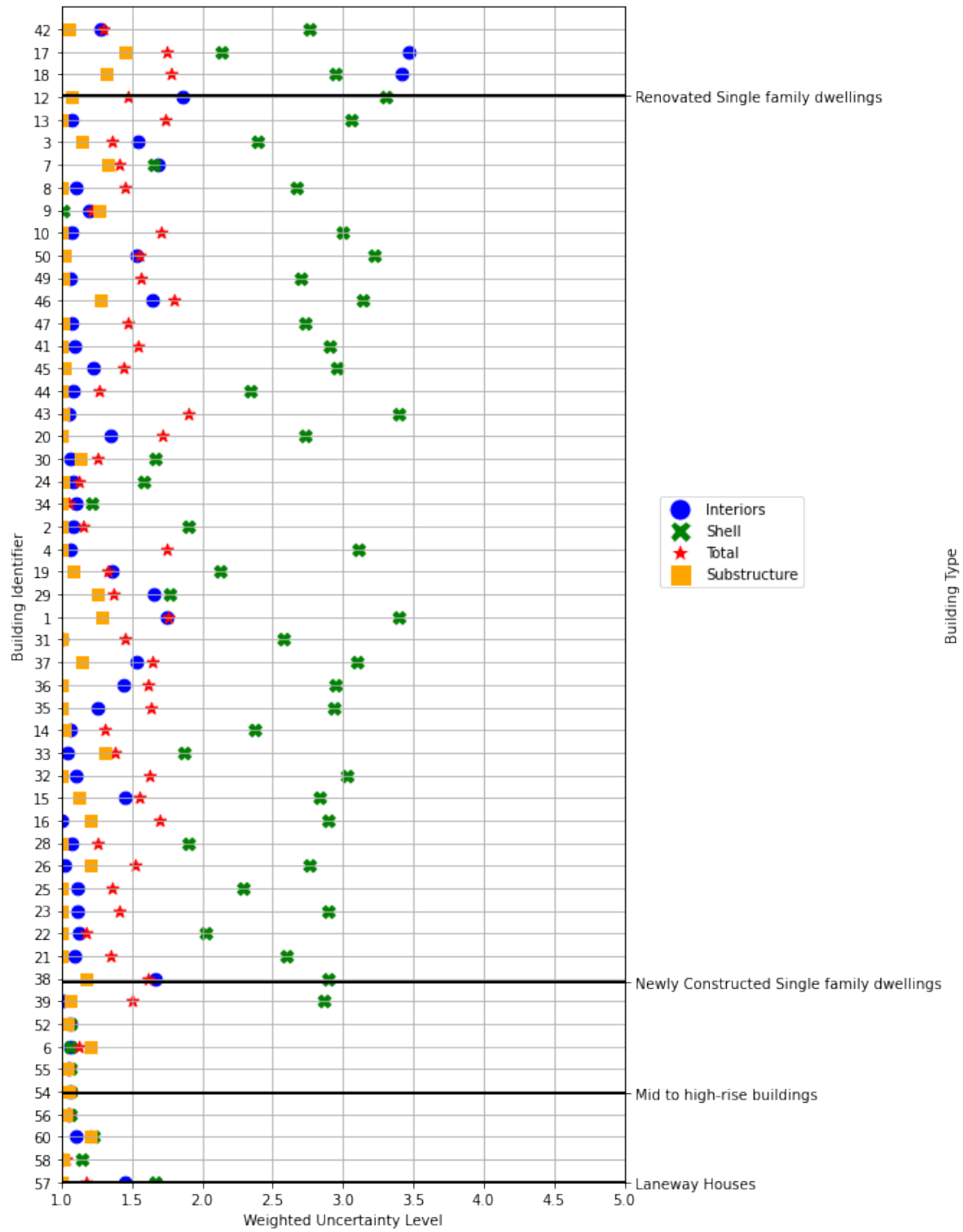
```
[46]: from matplotlib.lines import Line2D
      fig, ax = plt.subplots(figsize=(7,15))
      ax.set_xlim(1,5)
      ax.set_ylim(1,len(toplot_out))
      # ax.set_yticks(toplot_out['index'])
      handles = []
      for v,m,c in_
      ↪[('Interiors','o','blue'),('Shell','X','green'),('Total','*','red'),('Substructure','s','or
      ↪
          ax.scatter(x=toplot_out[v].values,y=toplot_out['index'].values, marker=m,
      ↪color=c, s=75)
          handles.append(
              Line2D([0], [0], marker=m, color='w', label=v,
                      markerfacecolor=c, markersize=15)
          )
      plt.legend(handles=handles,bbox_to_anchor=(1.05, 0.5), loc='lower left')
      plt.ylabel('Building Identifier')
```

```

plt.xlabel('Weighted Uncertainty Level')
plt.grid()
ax2 = ax.twinx()
ax2.set_ylim(0, len(toplot_out))
ax2.set_yticks([k for k,v in enumerate(toplot_out['Building Type'].values) if v_
    ↪ != toplot_out['Building Type'].values[k-1] or k==0])
# for tick in ax2.get_yticklabels():
#     tick.set_rotation(90)
ax2.set_yticklabels([v for k,v in enumerate(toplot_out['Building Type'].values)_
    ↪ if v != toplot_out['Building Type'].values[k-1] or k==0])
ax2.set_ylabel("Building Type")

plt.grid(color='black',linewidth=2)

```



[]:

[]: