00-00-PLOTS

March 8, 2021

Simple Plots

[34]:
```
from mpmath import mpf, mp
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
mp.dps = 3
# load data
filepath = '../data/roots_200.csv'
roots = pd.read_csv(filepath)
```

[35]:
```
#load up the discriminant roots
d_roots = pd.read_csv('../data/discriminant_roots_100.csv')
d_roots
```

[35]:

	Unnamed: 0	real	imaginary	p	q
0	0	0.250000	0.000000	0	1
1	0	0.250000	0.000000	0	1
2	1	-0.250000	0.000000	1	1
3	2	0.000000	-0.500000	1	2
4	3	0.000000	0.500000	1	2
…	…	…	…	..	..
15648	1717	1.023439	1.793734	34	47
15649	1718	-0.549302	2.066788	34	47
15650	1719	-0.549302	-2.066788	34	47
15651	1720	0.283700	-2.230023	34	47
15652	1721	0.283700	2.230023	34	47

[15653 rows x 5 columns]

[36]:
```
roots
```

[36]:

	Unnamed: 0	real	imaginary	p	q
0	0	1.000000	0.000000	1	3
1	1	0.500000	0.000000	1	4
2	2	0.381966	0.000000	1	5
3	3	2.618034	0.000000	1	5
4	4	0.333333	0.000000	1	6

```
…              …        …          …   …   …
463827        3471 -1.535960  -2.661993  143  317
463828        3472 -1.850274  -2.699986  143  317
463829        3473 -1.850274   2.699986  143  317
463830        3474  0.296217   2.925352  143  317
463831        3475  0.296217  -2.925352  143  317

[463832 rows x 5 columns]
```

[37]:
```
xstrings = roots['real'].to_numpy()
ystrings = roots['imaginary'].to_numpy()
qstrings = roots['q'].to_numpy()
pstrings = roots['p'].to_numpy()
xs = []
ys = []
qs = []
es = []
ps = []
for x in xstrings:
    if x not in roots.columns:
        xs.append(mpf(x))
for y in ystrings:
    if y not in roots.columns:
        ys.append(mpf(y))
for q in qstrings:
    if q not in roots.columns:
        qs.append(float(q))
for p in pstrings:
    if p not in roots.columns:
        ps.append(float(p))
```

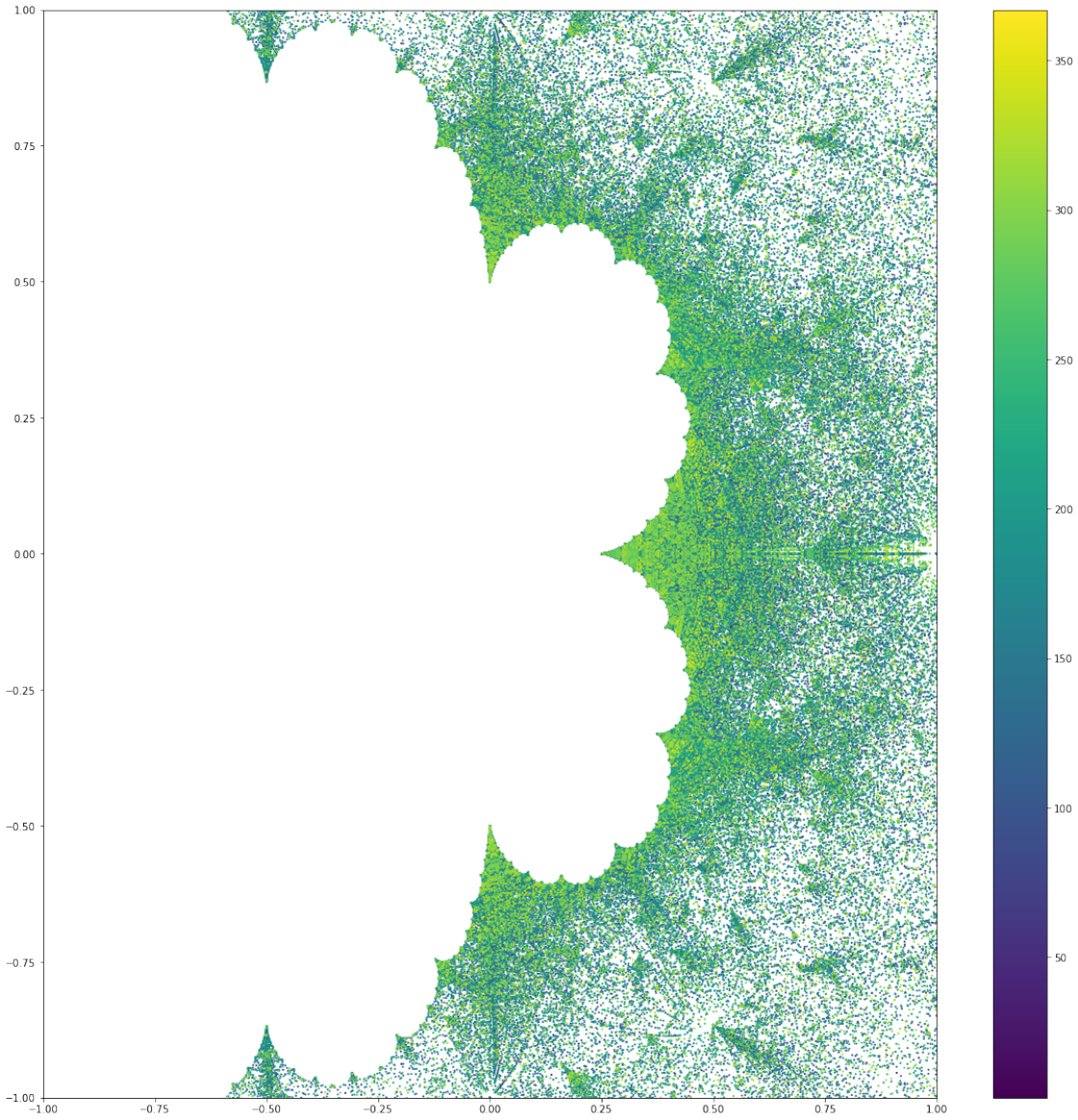
[38]:
```
# repeat for the discriminant
d_xstrings = d_roots['real'].to_numpy()
d_ystrings = d_roots['imaginary'].to_numpy()
d_qstrings = d_roots['q'].to_numpy()
d_pstrings = d_roots['p'].to_numpy()
d_xs = []
d_ys = []
d_qs = []
d_es = []
d_ps = []
for x in d_xstrings:
    if x not in d_roots.columns:
        d_xs.append(mpf(x))
for y in d_ystrings:
    if y not in d_roots.columns:
        d_ys.append(mpf(y))
```

```
for q in d_qstrings:
    if q not in d_roots.columns:
        d_qs.append(float(q))
for p in d_pstrings:
    if p not in d_roots.columns:
        d_ps.append(float(p))
```

[39]:
```
ratio = np.array(ps)/np.array(qs)
```

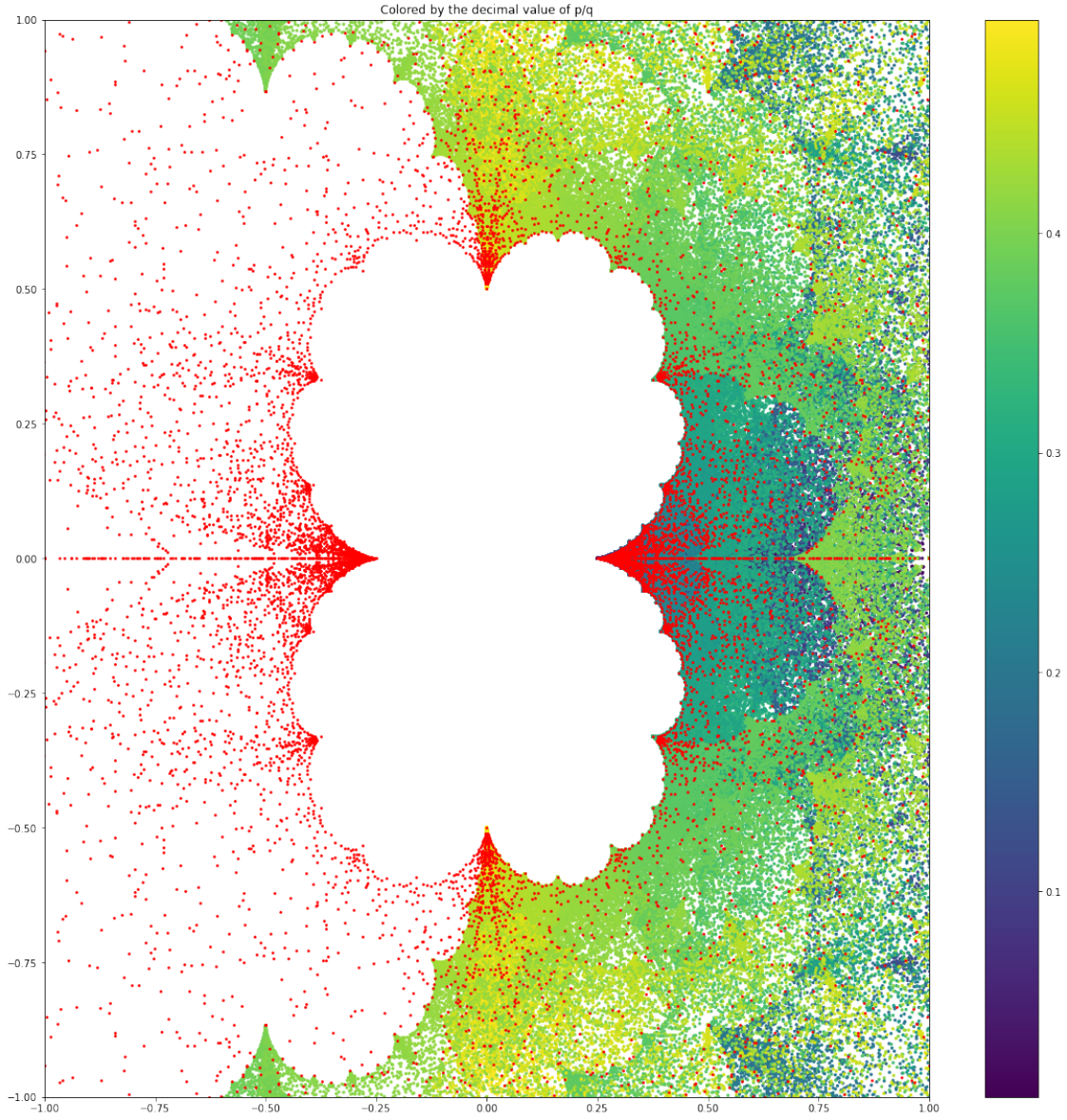
[40]:
```
fig, axs = plt.subplots(1,figsize=(20,20))
s = axs.scatter(xs,ys, c=qs, s=1)
lim = 1
plt.xlim(-lim,lim)
plt.ylim(-lim,lim)
fig.colorbar(s)
```

[40]:
```
<matplotlib.colorbar.Colorbar at 0x7fdefb7343d0>
```



[41]:
```
# color by ratio
fig, axs = plt.subplots(1,figsize=(20,20))
s = axs.scatter(xs,ys, c=ratio, s=5)
lim = 1
plt.xlim(-lim,lim)
plt.ylim(-lim,lim)
fig.colorbar(s)
# plot discriminant roots in red
axs.scatter(d_xs,d_ys,c='r', s=3)
axs.set_title("Colored by the decimal value of p/q")
```

[41]:
```
Text(0.5, 1.0, 'Colored by the decimal value of p/q')
```


Colored by the decimal value of p/q
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
0.4
0.3
0.2
0.1

[ ]:

[ ]: